prof. RUSU CONSTANTIN

ELECTRONICĂ DIGITALĂ

- AUXILIAR CURRICULAR -

BISTRIȚA - 2017

CUPRINS

PREFAȚĂ 1
CAPITOLUL 1. BAZELE ALGEBREI LOGICE 2
1.1. PREZENTAREA SISTEMELOR DE NUMERAŢIE 2
1.1.1 SISTEMUL DE NUMERAŢIE ZECIMAL 2
1.1.2 SISTEMUL DE NUMERAŢIE BINAR 3
1.1.3 SISTEMUL DE NUMERAŢIE OCTAL 5
1.1.4 SISTEMUL DE NUMERAŢIE HEXAZECIMAL 6
1.2. CONVERSII GENERALE ÎNTRE SISTEMELE DE NUMERAŢIE 7
1.2.1 CONVERSII DIN BINAR, OCTAL, HEXAZECIMAL 7
1.2.2 CONVERSII DIN ZECIMAL ÎN BINAR 8
1.2.3 CONVERSII DIN ZECIMAL ÎN OCTAL 9
1.2.4 CONVERSII DIN ZECIMAL ÎN HEXAZECIMAL 9
1.3. OPERAŢII CU NUMERE NEZECIMALE 10
1.3.1 OPERAŢII CU NUMERE BINARE 10
1.3.2 OPERAŢII CU NUMERE OCTALE ŞI HEXAZECIMALE 16
1.4. CODAREA NUMERELOR BINARE 21
1.4.1 REPREZENTAREA ÎN SISTEM BINAR A NUMERELOR NEGATIVE 21
1.4.2 CODURI NUMERICE 23
1.4.3 CODURI ALFANUMERICE 27
REZUMATUL CAPITOLULUI 29
EVALUAREA CUNOȘTINȚELOR 31
CAPITOLUL 2. FUNCŢII LOGICE 32
2.1 AXIOMELE ŞI TEOREMELE ALGEBREI LOGICE 32
2.2 PREZENTAREA FUNCŢIILOR LOGICE 33
2.3 REPREZENTAREA FUNCŢIILOR LOGICE 34
2.3.1. REPREZENTAREA FUNCŢIILOR LOGICE PRIN TABELA DE ADEVĂR 34
2.3.2 REPREZENTAREA FUNCŢIILOR LOGICE PRIN DIAGRAME VEITCH – KARNAUGH 36
2.4. SIMPLIFICAREA FUNCŢIILOR LOGICE 39
2.4.1 TRANSFORMAREA TABELULUI DE ADEVĂR ÎN EXPRESII LOGICE 39
2.4.2 MINIMIZAREA FUNCŢIILOR LOGICE 42
REZUMATUL CAPITOLULUI 48
EVALUAREA CUNOȘTINȚELOR 49

CAPITOLUL 3. PORŢI LOGICE 50
3.1. PORŢI LOGICE ELEMENTARE 50
3.1.1 POARTA LOGICĂ NU (NOT) 50
3.1.2 POARTA LOGICĂ SAU (OR) 51
3.1.3 POARTA LOGICĂ ŞI (AND) 51
3.1.4 POARTA LOGICĂ SAU – NU (NOR) 52
3.1.5 POARTA LOGICĂ ŞI – NU (NAND) 53
3.1.6 POARTA LOGICĂ SAU – EXCLUSIV (XOR) 54
3.2. IMPLEMENTAREA FUNCŢIILOR LOGICE CU PORŢI LOGICE 55
3.2.1 ANALIZA CIRCUITELOR LOGICE 55
3.2.2 SINTEZA CIRCUITELOR LOGICE 59
REZUMATUL CAPITOLULUI 63
EVALUAREA CUNOȘTINȚELOR 64
CAPITOLUL 4. CIRCUITE LOGICE ELEMENTARE 66
4.1. CIRCUITE LOGICE CU COMPONENTE DISCRETE 66
4.1.1 PORŢI LOGICE ELEMENTARE CU COMPONENTE PASIVE 66
4.1.2 PORŢI LOGICE ELEMENTARE CU COMPONENTE ACTIVE 68
4.2. CIRCUITE LOGICE ÎN TEHNOLOGIE INTEGRATĂ 73
4.2.1 CIRCUITE LOGICE INTEGRATE BIPOLARE 73
4.2.2 CIRCUITE LOGICE INTEGRATE MONOPOLARE 77
REZUMATUL CAPITOLULUI 80
4.3. LUCRĂRI DE LABORATOR 81
LUCRARE DE LABORATOR 1 81
LUCRARE DE LABORATOR 2 83
LUCRARE DE LABORATOR 3 85
CAPITOLUL 5. CIRCUITE LOGICE COMBINAŢIONALE 87
5.1. GENERALITĂŢI 87
5.2. CODIFICATOARE 89
5.3. DECODIFICATOARE 92
5.4. MULTIPLEXOARE 99
5.5. DEMULTIPLEXOARE 107
5.6. COMPARATOARE NUMERICE 113
5.7. SUMATOARE 119
5.8. CONVERTOARE DE COD 123
REZUMATUL CAPITOLULUI 126

5.9. LUCRĂRI DE LABORATOR 128
LUCRARE DE LABORATOR 4 128
LUCRARE DE LABORATOR 5 130
CAPITOLUL 6. CIRCUITE LOGICE SECVENŢIALE 132
6.1. GENERALITĂŢI 132
6.2. CIRCUITE BASCULANTE BISTABILE 134
6.2.1 CIRCUITE BASCULANTE BISTABILE DE TIP RS 135
6.2.2 CIRCUITE BASCULANTE BISTABILE DE TIP JK 139
6.2.3 CIRCUITE BASCULANTE BISTABILE DE TIP D 142
6.2.4 CIRCUITE BASCULANTE BISTABILE DE TIP T 143
6.3. NUMĂRĂTOARE 144
6.3.1 NUMĂRĂTOARE ASINCRONE 145
6.3.2 NUMĂRĂTOARE SINCRONE 150
6.3.3 APLICAŢII ALE NUMĂRĂTOARELOR 152
6.4. REGISTRE 154
REZUMATUL CAPITOLULUI 163
6.5 LUCRĂRI DE LABORATOR 166
LUCRARE DE LABORATOR 6 166
LUCRARE DE LABORATOR 7 168
LUCRARE DE LABORATOR 8 170
LUCRARE DE LABORATOR 9 172
LUCRARE DE LABORATOR 10 174
LUCRARE DE LABORATOR 11 176
LUCRARE DE LABORATOR 12 178
BIBLIOGRAFIE 180

PREFAȚĂ

Electronica este o disciplină tehnico-științifică în care teoria se îmbină în mod armonios și indispensabil cu practica. Electronica este o ramură de vârf a industriei atât în zilele noastre cât și în toate epocile viitoare.

Încă de la începuturile sale electronica a atras în special tinerii dornici de a realiza și experimenta diverse construcții.

Printr-o muncă bine dirijată în care se îmbină armonios însușirea elementelor teoretice cu realizarea construcțiilor practice, tânărul licean de azi va fi specialistul de mâine.

Auxiliarul curricular ***Electronică Digitală*** se adresează elevilor care urmează cursurile unui liceu tehnologic sau ale unei școli de arte și meserii, domeniul electronică și automatizări precum și specializările care derivă din acest domeniu.

Auxiliarul curricular este structurat în cinci capitole. În fiecare capitol sunt tratate noțiunile teoretice de bază corespunzătoare temei respective, lucrări de laborator și simulări cu ajutorul calculatorului. Capitolul se încheie cu un rezumat și un test de verificare a cunoștințelor.

Autorul urează mult succes celor care utilizează acest auxiliar curricular și le dorește să îmbine cât mai plăcut și armonios cunoștințele teoretice cu abilitățile tehnice pentru a-și dezvolta cât mai mult puterea de creație tehnică.

ÎN ELECTRONICĂ VIITORUL RĂMÂNE DESCHIS TUTUROR POSIBILITĂȚILOR.

Prof. RUSU CONSTANTIN
Colegiul Tehnic INFOEL - BISTRIȚA

CAPITOLUL 1. BAZELE ALGEBREI LOGICE

1.1. PREZENTAREA SISTEMELOR DE NUMERAŢIE

Orice sistem de numeraţie este caracterizat prin **caractere** care reprezintă numărul propriu-zis, şi **baza** sau rădăcina sistemului de numeraţie care reprezintă numărul de simboluri permise pentru reprezentarea numărului (**Tabel 1.1**).

Tabel 1.1 Sisteme de numeraţie

Sistem de numeraţie	Baza	Caractere permise
ZECIMAL	10	0 ; 1 ; 2 ; 3 ; 4 ; 5 ; 6 ; 7 ; 8 ; 9 .
BINAR	2	0 ; 1.
OCTAL	8	0 ; 1 ; 2 ; 3 ; 4 ; 5 ; 6 ; 7.
HEXAZECIMAL	16	0; 1; 2; 3; 4; 5; 6; 7; 8; 9; A; B; C; D; E; F.

1.1.1 SISTEMUL DE NUMERAŢIE ZECIMAL

Acest sistem este un ***sistem de numeraţie poziţional*** se utilizează cel mai frecvent. Conform tabelului 1, sistemul zecimal utilizează **10 caractere (cifre)** şi are baza **10** deoarece pentru reprezentarea unui număr în acest sistem sunt permise 10 caractere.

Un număr din sistemul zecimal se reprezintă printr-un şir de cifre în care fiecare dintre poziţiile cifrelor are o anumită ***pondere***.

Ponderea unei poziţii este egală cu **10 la puterea** dată de ***numărul de ordine*** al poziţiei respective.

Numărul de ordine al poziţiei este pozitiv pentru partea întreagă a numărului zecimal şi negativ pentru partea fracţionară a numărului zecimal.

Valoarea numărului de ordine pentru partea întreagă este **0** pentru **unităţi**, **1** pentru **zeci**, **2** pentru **sute**, **3** pentru **mii**, etc.

Valoarea numărului de ordine pentru partea zecimală este **-1** pentru **unităţi**, **-2** pentru **zeci**, **-3** pentru **sute**, **-4** pentru **mii**, etc.

Valoarea unui număr zecimal este suma ponderată a cifrelor sale.

Exemple de numere scrise în sistemul zecimal:

$5627 = (5627)_{10} = 5 \cdot 10^3 + 6 \cdot 10^2 + 2 \cdot 10^1 + 7 \cdot 10^0 = 5000 + 600 + 20 + 7$

$245{,}37 = (245{,}37)_{10} = 2 \cdot 10^2 + 4 \cdot 10^1 + 5 \cdot 10^0 + 3 \cdot 10^{-1} + 7 \cdot 10^{-2} =$

$= 200 + 40 + 5 + 0{,}30 + 0{,}07$

1.1.2 SISTEMUL DE NUMERAŢIE BINAR

Acest sistem este un sistem de numeraţie poziţional care utilizează **2** caractere (**0** şi **1**) şi are baza **2**. Deoarece numerele binare pot fi prelucrate direct de circuitele digitale (logice), sistemul de numeraţie binar se utilizează pentru transmiterea informaţiilor gestionate de un calculator şi a semnalelor în montaje cu circuite digitale.

O informaţie elementară gestionată de calculator poate fi asociată cu două niveluri de tensiune: **0 V(care corespunde caracterului 0)** şi **+5 V(care corespunde caracterului 1).**

Caracterele utilizate în sistemul binar se numesc **cifre binare sau *biţi***.

Un grup de **8 biţi** formează **un octet** sau ***1 byte***.

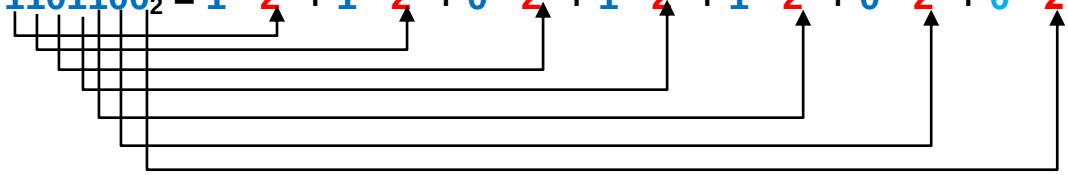
Bitul cel mai din **stânga** al unui număr binar se numeşte **bitul de cel mai mare ordin** sau ***bitul cel mai semnificativ*** **(MSB – most significant bit)**

Bitul cel mai din **dreapta** al unui număr binar se numeşte **bitul de cel mai mic ordin** sau ***bitul cel mai puţin semnificativ*** **(LSB – least significant bit)**

Un număr binar este format dintr-un şir de caractere **0** sau **1**. Reprezentarea unui număr binar este asemănătoare cu reprezentarea numărului zecimal cu deosebirea că se schimbă ponderea din 10 în 2.

Exemple de numere binare şi echivalentele lor zecimale:

$$1101100_2 = 1 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 108_{10}$$



$$1001{,}010 = 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} + 0 \cdot 2^{-3} = 9{,}25_{10}$$

Codul BCD

Codul BCD, numit şi codul 8421 permitea scrierea cifrelor de la 0 la 9 în sistemul binar utilizând pentru fiecare cifră un ansamblu de **4 cifre binare** (4 biţi) (**Tabel1. 2**).

Tabel 1.2 Reprezentarea numerelor în cod BCD

Cifra	Cod BCD $2^3\ 2^2\ 2^1\ 2^0$	ZECIMAL
0	0 0 0 0	$0\cdot 2^3 + 0\cdot 2^2 + 0\cdot 2^1 + 0\cdot 2^0 = 0$
1	0 0 0 1	$0\cdot 2^3 + 0\cdot 2^2 + 0\cdot 2^1 + 1\cdot 2^0 = 1$
2	0 0 1 0	$0\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 0\cdot 2^0 = 2$
3	0 0 1 1	$0\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 = 3$
4	0 1 0 0	$0\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + 0\cdot 2^0 = 4$
5	0 1 0 1	$0\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + 1\cdot 2^0 = 5$
6	0 1 1 0	$0\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2^1 + 0\cdot 2^0 = 6$
7	0 1 1 1	$0\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 = 7$
8	1 0 0 0	$1\cdot 2^3 + 0\cdot 2^2 + 0\cdot 2^1 + 0\cdot 2^0 = 8$
9	1 0 0 1	$1\cdot 2^3 + 0\cdot 2^2 + 0\cdot 2^1 + 1\cdot 2^0 = 9$

1.1.3 SISTEMUL DE NUMERAŢIE OCTAL

Acest sistem de numeraţie utilizează **8 caractere** (vezi **tabelul 1.1**) şi are baza **8**.

Reprezentarea unui număr octal este asemănătoare cu reprezentarea numărului zecimal cu deosebirea că se schimbă ponderea din 10 în 8.

Exemple de numere octale şi echivalentele lor zecimale:

$3081_8 = 3 \cdot 8^3 + 0 \cdot 8^2 + 8 \cdot 8^1 + 1 \cdot 8^0 = 3 \cdot 512 + 0 \cdot 64 + 8 \cdot 8 + 1 \cdot 1 = 1601_{10}$

$12{,}4_8 = 1 \cdot 8^1 + 2 \cdot 8^0 + 4 \cdot 8^{-1} = 1 \cdot 8 + 2 \cdot 1 + 4 \cdot \frac{1}{8} = 10{,}5_{10}$

La fiecare caracter din sistemul de numeraţie octal îi corespunde un şir de 3 biţi (deoarece cu un şir de 3 biţi se pot realiza 8 combinaţii) după cum este prezentat în **Tabelul 1.3**

Tabel 1.3 Reprezentarea numerelor în octal

OCTAL	BINAR $2^2\ 2^1\ 2^0$	ZECIMAL
0	0 0 0	$0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 0$
1	0 0 1	$0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 1$
2	0 1 0	$0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 2$
3	0 1 1	$0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 3$
4	1 0 0	$1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 4$
5	1 0 1	$1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 5$
6	1 1 0	$1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 6$
7	1 1 1	$1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 7$

Pentru **conversia numerelor binare în numere octale** se împart biţii numărului binar în grupe de câte 3 pornind de la dreapta (sau de la virgulă) spre stânga:

$1011100101001_2 = 001\ \ 011\ \ 100\ \ 101\ \ 001 = 13451_8$

$110{,}01_2 = 110\ ,\ 010 = 6{,}2_8$

Pentru **conversia numerelor octale în numere binare** se înlocuieşte fiecare caracter din octal cu şirul corespunzător de 3 biţi:

$2106_8 = 010\ \ 001\ \ 000\ \ 110 = 010001000110_2$

$204{,}51 = 010\ \ 000\ 100\ ,\ 101\ \ 001 = 010000100{,}101001_2$

1.1.4 SISTEMUL DE NUMERAŢIE HEXAZECIMAL

Acest sistem de numeraţie utilizează 16 caractere (vezi **tabelul 1.1**) şi are baza **16**. Reprezentarea unui număr hexazecimal este asemănătoare cu reprezentarea numărului zecimal cu deosebirea că se schimbă ponderea din 10 în 16.
La fiecare caracter din sistemul de numeraţie hexazecimal îi corespunde un şir de 4 biţi (deoarece cu un şir de 4 biţi se pot realiza 16 combinaţii) după cum este prezentat în **Tabelul 1.4**

TABEL 1.4 Reprezentarea numerelor în hexazecimal

HEXA ZECIMAL	BINAR 2^3 2^2 2^1 2^0	ZECIMAL
0	0 0 0 0	$0\cdot 2^3 + 0\cdot 2^2 + 0\cdot 2^1 + 0\cdot 2^0 = 0$
1	0 0 0 1	$0\cdot 2^3 + 0\cdot 2^2 + 0\cdot 2^1 + 1\cdot 2^0 = 1$
2	0 0 1 0	$0\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 0\cdot 2^0 = 2$
3	0 0 1 1	$0\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 = 3$
4	0 1 0 0	$0\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + 0\cdot 2^0 = 4$
5	0 1 0 1	$0\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + 1\cdot 2^0 = 5$
6	0 1 1 0	$0\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2^1 + 0\cdot 2^0 = 6$
7	0 1 1 1	$0\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 = 7$
8	1 0 0 0	$1\cdot 2^3 + 0\cdot 2^2 + 0\cdot 2^1 + 0\cdot 2^0 = 8$
9	1 0 0 1	$1\cdot 2^3 + 0\cdot 2^2 + 0\cdot 2^1 + 1\cdot 2^0 = 9$
A	1 0 1 0	$1\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 0\cdot 2^0 = 10$
B	1 0 1 1	$1\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 = 11$
C	1 1 0 0	$1\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + 0\cdot 2^0 = 12$
D	1 1 0 1	$1\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + 1\cdot 2^0 = 13$
E	1 1 1 0	$1\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2^1 + 0\cdot 2^0 = 14$
F	1 1 1 1	$1\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 = 15$

Exemple de numere hexazecimale şi echivalentele lor zecimale:

$218_{16} = 2\cdot 16^2 + 1\cdot 16^1 + 8\cdot 16^0 + = 2\cdot 256 + 1\cdot 16 + 8\cdot 1 = 536_{10}$

$BAC_{16} = B\cdot 16^2 + A\cdot 16^1 + C\cdot 16^0 = 11\cdot 256 + 10\cdot 16 + 12\cdot 1 = 2988_{10}$

Pentru **conversia numerelor binare în numere hexazecimale** se împart biţii numărului binar în grupe de câte 4 biţi de la dreapta la stânga:

$1011110101101_2 = 0001\ \ 0111\ \ 1010\ \ 1101 = 17AD_8$

Pentru **conversia numerelor hexazecimale în numere binare** se înlocuieşte fiecare caracter din hexazecimal cu şirul corespunzător de 4 biţi:

$DAC_{16} = 1101\ \ 1010\ \ 1100 = 110110101100_2$

1.2. CONVERSII GENERALE ÎNTRE SISTEMELE DE NUMERAŢIE

1.2.1 CONVERSII DIN BINAR, OCTAL, HEXAZECIMAL

Conversia din **binar** în **octal** sau **hexazecimal** se face prin ***substituţie*** (se împarte numărul binar în grupe de câte 3 sau 4 biţi şi se înlocuieşte fiecare grupă cu caracterul corespunzător – conform **tabel 1.3** şi **tabel 1.4**)

Conversia din **octal** în **binar** sau **hexazecimal** se face prin ***substituţie*** (caracterele numărului octal se înlocuiesc cu grupe de 3 sau 4 biţi

OBS. Conversia din **octal** în **hexazecimal** nu se face direct, mai întâi se converteşte din **octal** în **binar** apoi din **binar** în **hexazecimal**

Conversia din **hexazecimal** în **binar** sau **octal** se face prin ***substituţie*** (caracterele numărului hexazecimal se înlocuiesc cu grupe de 3 sau 4 biţi.

OBS. Conversia din **hexazecimal** în **octal** nu se face direct, mai întâi se converteşte din **hexazecimal** în **binar** apoi din **binar** în **octal**

Conversia din **binar, octal , hexazecimal** în **zecimal** se face prin ***adunare*** (algoritmi de conversie sunt prezentaţi în secţiunea **1.1.**)

Conversia din **zecimal** în **binar, octal, hexazecimal** se face prin ***împărţire*** (algoritmi de conversie vor fi prezentaţi în continuare).

Metodele de conversie între cele mai uzuale baze de numeraţie sunt prezentate în

Tabelul 1.5 Metode de conversie

CONVERSIE	METODĂ	EXEMPLE
Din **BINAR** în		
OCTAL	**Substituţie**	$1100101_2 = 001\ \ 100\ \ 101_2 = 145_8$
HEXAZECIMAL	**Substituţie**	$111010010011_2 = 1110\ \ 1001\ \ 0011_2 = E93_{16}$
ZECIMAL	**Adunare**	$10011_2 = 1\cdot2^4+0\cdot2^3+0\cdot2^2+1\cdot2^1+1\cdot2^0 = 19_{10}$
Din **OCTAL** în		
BINAR	**Substituţie**	$2105_8 = 010\ \ 001\ \ 000\ \ 101_2 = 010001000101_2$
HEXAZECIMAL	**Substituţie**	$625_8 = 110\ \ 010\ \ 101_2 = 0001\ \ 1001\ \ 0101_2 = 195_{16}$
ZECIMAL	**Adunare**	$207_8 = 2\cdot8^2 + 0\cdot8^1 + 7\cdot8^0 = 128 + 0 + 7 = 135_{10}$
Din **HEXAZECIMAL** în		
BINAR	**Substituţie**	$D0C_{16} = 1101\ 0000\ 1100_2 = 110100001100_2$
OCTAL	**Substituţie**	$EA_{16} = 1110\ \ 1010_2 = 011\ \ 101\ \ 010_2 = 352_8$
ZECIMAL	**Adunare**	$BEC_{16} = 11\cdot16^2 + 14\cdot16^1 + 12\cdot16^0 = 3820_{10}$

1.2.2 CONVERSII DIN ZECIMAL ÎN BINAR

Conversia din **zecimal** în **binar** se face prin împărţirea numărului zecimal la **2** astfel:

179 : 2 = 89 rest 1 **(LSB - cifra cea mai puţin semnificativă)**

89 : 2 = 44 rest 1

44 : 2 = 22 rest 0

22 : 2 = 11 rest 0

11 : 2 = 5 rest 1

5 : 2 = 2 rest 1

2 : 2 = 1 rest 0

(MSB – cifra cea mai semnificativă) **1 : 2 = 0 rest 1**

Caracterele numărului în binar este format de valorile resturilor scrise de la MBS spre LBS

179_{10} = 10110011_2

OBSERVAŢII:

Împărţirea se face până când **deîmpărţitul** (numărul care se împarte) **este mai mic** decât **împărţitorul** (la conversia în binar **împărţitorul** este **0**).

La ultima împărţire – când deîmpărţitul este mai mic decât împărţitorul – rezultatul împărţirii este 0 iar ***restul este egal cu deîmpărţitul***

1 : 2 = 0 rest 1

O altă metodă este împărţirea numărului succesiv la 2 şi în coloana din stânga se scriu rezultatele împărţirii la 2 iar în coloana din dreapta resturile obţinute:

179	2
89	1
44	1
22	0
11	0
5	1
2	1
1	0
0	1

179 : 2 = 89 rest 1
89 : 2 = 44 rest 1
44 : 2 = 22 rest 0
22 : 2 = 11 rest 0
11 : 2 = 5 rest 1
5 : 2 = 2 rest 1
2 : 2 = 1 rest 0
1 : 2 = 0 rest 1

43	2
21	1
10	1
5	0
2	1
1	0
0	1

43 : 2 = 21 rest 1
21 : 2 = 10 rest 1
10 : 2 = 5 rest 0
5 : 2 = 2 rest 1
2 : 2 = 1 rest 0
1 : 2 = 0 rest 1

179_{10} = 10110011_2 **43_{10} = 101011_2**

1.2.3 CONVERSII DIN ZECIMAL ÎN OCTAL

Conversia din **zecimal** în **octal** se face prin împărţirea numărului zecimal la **8** astfel:

1962 : 8 = 245 rest 2 (LSB)

245 : 8 = 30 rest 5

30 : 8 = 3 rest 6

3 : 8 = 0 rest 3 (MSB)

$\Rightarrow 1962_{10} = 3652_8$

1962	8
245	2
30	5
3	6
0	3

$\Rightarrow 1962_{10} = 3652_8$

1.2.4 CONVERSII DIN ZECIMAL ÎN HEXAZECIMAL

Conversia din **zecimal** în **hexazecimal** se face prin împărţirea numărului zecimal la **16** astfel:

2988 : 16 = 186 rest 12 (LSB)

186 : 16 = 11 rest 10

11 : 16 = 0 rest 11 (MSB)

Dacă restul este un număr (dacă nu este o cifră de la 0 la 9) pentru fiecare număr se scrie caracterul corespunzător conform **tabelului 1.4**

11 → B ; 10 → A ; 12 → C $\Rightarrow 2988_{10} = BAC_{16}$

2988	16
186	12 → C
11	10 → A
0	11 → B

$\Rightarrow 2988_{10} = BAC_{16}$

1.3. OPERAŢII CU NUMERE NEZECIMALE

1.3.1 OPERAŢII CU NUMERE BINARE

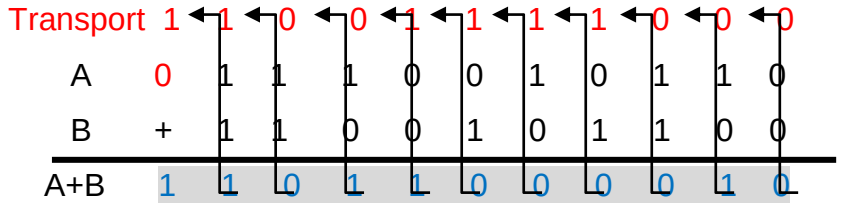
A. ADUNAREA NUMERELOR BINARE

Reguli de bază:

- 0 + 0 = 0 transport 0;
- 0 + 1 = 1 transport 0;
- 1 + 0 = 1 transport 0;
- 1 + 1 = 0 transport 1.

Pentru a aduna două numere binare se adună între ei biţii numerelor (începând de la dreapta la stânga) iar la acest rezultat se adaugă transportul (care poate fi 0 sau 1) conform regulilor de mai sus.

Exemple de adunări cu numere binare



Algoritmul de realizare a adunării de mai sus:

- Se adună biţii de pe prima coloană din dreapta. Rezultatul se trece sub coloană iar transportul deasupra celei de-a doua coloane din dreapta;
- Se adună biţii de pe a doua coloană din dreapta. Rezultatul se adună cu transportul de deasupra coloanei apoi se trece rezultatul obţinut sub coloană iar transportul se trece deasupra celei de-a treia coloane din dreapta;
- Se continuă adunarea după acest algoritm până se ajunge la prima coloana din stânga.

0 + 0 = 0 transport 0
1 + 0 + 0 = 1 transport 0
1 + 1 + 0 = 0 transport 1
0 + 1 + 1 = 0 transport 1
1 + 0 + 1 = 0 transport 1 $\Rightarrow$ **11011000010**
0 + 1 + 1 = 0 transport 1
0 + 0 + 1 = 1 transport 0
1 + 0 + 0 = 1 transport 0
1 + 1 + 0 = 0 transport 1
1 + 1 + 1 = 1 transport 1
1 + 0 = 1

OBSERVAȚIE: Dacă într-o adunare numărul de caractere **1** este **impar** atunci rezultatul adunării este impar, adică **1,** iar dacă este par rezultatul adunării este **0**.

Transport		0	0	0	0	0	0	0	0
A		0	1	0	0	1	1	0	0
B	+	1	0	0	1	0	0	0	1
A + B		1	1	0	1	1	1	0	1

Transport		1	1	1	1	1	1	1	0
A		0	1	1	1	1	1	1	1
B	+	0	0	1	1	1	1	1	1
A + B		1	0	1	1	1	1	1	0

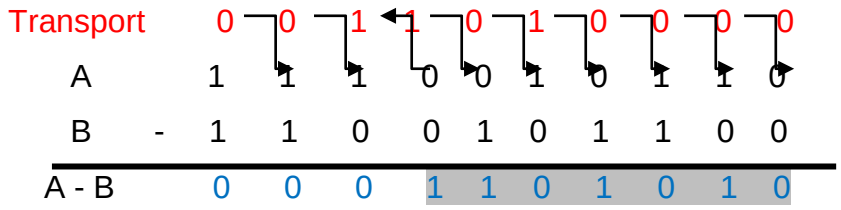
B. SCĂDEREA NUMERELOR BINARE

Reguli de bază:

- 0 - 0 = 0 împrumut 0;
- 1 - 0 = 1 împrumut 0;
- 1 - 1 = 0 împrumut 0;
- 0 - 1 = 1 împrumut 1.

Pentru a scade două numere binare se scad între ei biţii numerelor (începând de la dreapta la stânga) iar din acest rezultat se scade împrumutul (care poate fi 0 sau 1) conform regulilor de mai sus.

Exemple de scăderi cu numere binare



Algoritmul de realizare a scăderii de mai sus:

- Se scad din biţii numărului A biţii numărului B de pe prima coloană din dreapta. Rezultatul se trece sub coloană iar împrumutul deasupra celei de-a doua coloane din dreapta.
- Se scad biţii de pe a doua coloană din dreapta. Din rezultat se scade împrumutul de deasupra coloanei apoi se trece rezultatul obţinut sub coloană iar împrumutul se trece deasupra celei de-a treia coloane din dreapta.
- Se continuă scăderea după acest algoritm până se ajunge la prima coloana din stânga.

Împrumut	0	0	1	1	0	0	1	1	0
A		1	1	0	0	1	1	0	0
B	-	1	0	0	1	0	0	0	1
A - B		0	0	1	1	1	0	1	1

Împrumut	0	0	0	0	0	0	1	0	0
A		1	0	1	1	1	1	0	1
B	-	1	0	0	0	0	0	1	1
A - B		0	0	1	1	1	0	1	0

Împrumut	0	1	0	1	0	1	0	1	0
A		1	0	1	0	1	0	1	0
B	-	0	1	0	1	0	1	0	1
A - B		0	1	0	1	0	1	0	1

Împrumut	0	1	1	0	0	1	1	0	0
A		1	0	0	1	1	0	0	1
B	-	0	1	1	0	0	1	1	0
A + B		0	0	1	1	0	0	1	1

C. ÎNMULŢIREA NUMERELOR BINARE

Reguli de bază:

- 0 x 0 = 0;
- 1 x 0 = 0;
- 0 x 1 = 0;
- 1 x 1 = 1.

Pentru a înmulţii două numere binare A (deînmulţit) şi B(înmulţitor) se procedează exact ca la înmulţirea a două numere zecimale:

- Se înmulţeşte pe rând fiecare cifră a înmulţitorului cu cifrele deînmulţitului;
- Se scriu rezultatele obţinute unul sub altul decalându-le cu o unitate spre stânga;
- Se adună pe verticală cifrele rezultatelor fiecărei înmulţiri *respectând regulile de adunare a numerelor binare.*

Exemple de înmulţiri a numerelor binare

Exemplul 1.

```
  51          1 1 0 0 1 1  deînmulţit
x 13        x     1 1 0 1  înmulţitor
 153          1 1 0 0 1 1
+ 51       0 0 0 0 0 0
 663     1 1 0 0 1 1                  produse parţiale care se adună
      + 1 1 0 0 1 1
      1 0 1 0 0 1 0 1 1 1  PRODUS
```

Exemplul 2.

```
  125           1 1 1 1 1 0 1   deînmulţit
x  24         x     1 1 0 0 0   înmulţitor
 500          0 0 0 0 0 0 0
 250        0 0 0 0 0 0 0
3000      0 0 0 0 0 0 0                produse parţiale care se adună
        1 1 1 1 1 0 1
      1 1 1 1 1 0 1
      1 0 1 1 1 0 1 1 0 0 0
```

D. ÎMPĂRŢIREA NUMERELOR BINARE

Algoritmul de împărţire a două numere binare are la bază metoda împărţirii a două numere întregi. Fiind dat deîmpărţitul D şi împărţitorul Î, pentru operaţia de împărţire trebuie să se determine câtul C şi restul R, astfel încât să fie satisfăcută relaţie:

D = Î x C + R

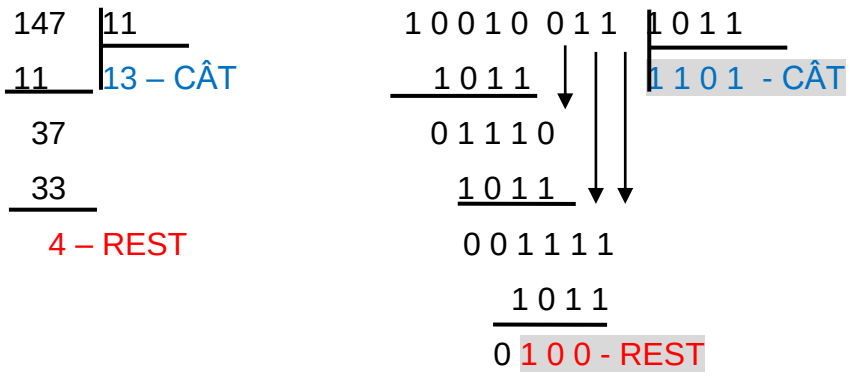
Operaţia de împărţire în cazul numerelor binare, se va reduce la o serie de scăderi ale împărţitorului din restul parţial ţinând cont de următoarele reguli:

- Dacă *restul este mai mare decât împărţitorul câtul este 1*;
- Dacă *restul este mai mic decât împărţitorul câtul este 0*.

La efectuarea scăderilor *se respectă regulile de scăderea a numerelor binare*.

Exemple de împărţire a numerelor binare

Exemplul 1.



Algoritmul împărţirii deîmpărţitului 10010011 la împărţitorul 1011:

- Deoarece împărţitorul 1011 este mai mare decât primii 4 biţi ai deîmpărţitului 1001 împărţitorul se va împărţii la primi 5 biţi ai deîmpărţitului 10010;
- Deoarece 10010 este mai mare decât 1011.

  Deci *primul bit al câtului este 1*;
- Înmulţim câtul 1 cu împărţitorul 1011 şi trecem rezultatul în stânga sub primi 5 biţi ai deîmpărţitului;
- Scădem 1011 din 10010 (respectând regulile scăderii în binar) şi obţinem restul 111;
- Coborâm bitul deîmpărţitului, care este 0 (vezi săgeata) şi obţinem restul 1110;
- Deoarece restul 1110 este mai mare decât împărţitorul 1011 câtul este 1.

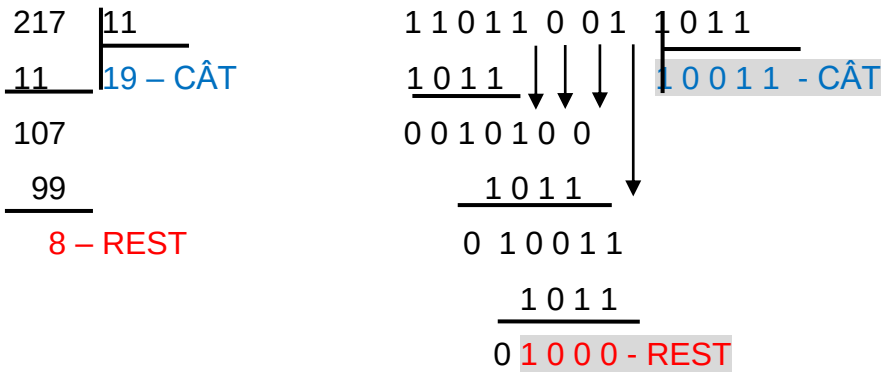
  Deci *al doilea bit al câtului este 1*;

- Înmulţim câtul 1 cu împărţitorul 1011 şi trecem rezultatul în stânga sub restul 1110;
- Scădem 1011 din 1110 (respectând regulile scăderii în binar) şi obţinem restul 11;
- Coborâm bitul deîmpărţitului, care este 1 (vezi săgeata) şi obţinem restul 111;
- Deoarece restul 111 este mai mic decât împărţitorul 1011 câtul este 0.

  Deci *al treilea bit al câtului este 0*;
- Coborâm bitul deîmpărţitului, care este 1 (vezi săgeata) şi obţinem restul 1111;
- Deoarece restul 1111 este mai mare decât împărţitorul 1011 câtul este 1.

  Deci *al patrulea bit al câtului este 1*;
- Înmulţim câtul 1 cu împărţitorul 1011 şi trecem rezultatul în stânga sub restul 1111;
- Scădem 1011 din 1111 (respectând regulile scăderii în binar) şi obţinem restul 100.

Exemplul 2.



Algoritmul împărţirii deîmpărţitului 1101100 la împărţitorul 1011:

- Deoarece numărul format din primi 4 biţi ai deîmpărţitului 1101 este mai mare decât 1011.

  Deci *primul bit al câtului este 1*;
- Înmulţim câtul 1 cu împărţitorul 1011 şi trecem rezultatul în stânga sub primi 4 biţi ai deîmpărţitului;
- Scădem 1011 din 1101 (respectând regulile scăderii în binar) şi obţinem restul 10;
- Coborâm bitul deîmpărţitului, care este 1 (vezi săgeata) şi obţinem restul 101;
- Deoarece restul 101 este mai mic decât împărţitorul 1011 câtul este 0.

  Deci *al doilea bit al câtului este 0*;
- Coborâm bitul deîmpărţitului, care este 0 (vezi săgeata) şi obţinem restul 1010;
- Deoarece restul 1010 este mai mic decât împărţitorul 1011 câtul este 0.

Deci *al treilea bit al câtului este 0*;

- Coborâm bitul deîmpărţitului, care este 0 (vezi săgeata) şi obţinem restul 10100;
- Deoarece restul 10100 este mai mare decât împărţitorul 1011 câtul este 1.

  Deci *al patrulea bit al câtului este 1*;
- Înmulţim câtul 1 cu împărţitorul 1011 şi trecem rezultatul în stânga sub restul 10100;
- Scădem 1011 din 10100 (respectând regulile scăderii) şi obţinem restul 1001;
- Coborâm bitul deîmpărţitului, care este 1 (vezi săgeata) şi obţinem restul 10011;
- Deoarece restul 10011 este mai mare decât împărţitorul 1011 câtul este 1.

  Deci *al cincilea bit al câtului este 1*;
- Înmulţim câtul 1 cu împărţitorul 1011 şi trecem rezultatul în stânga sub restul 10011;
- Scădem 1011 din 10011 (respectând regulile scăderii) şi obţinem restul 1000.

1.3.2 OPERAŢII CU NUMERE OCTALE ŞI HEXAZECIMALE

A. ADUNAREA NUMERELOR OCTALE

Reguli:

- Adunarea se face ca în sistemul zecimal, prin scrierea numerelor unul sub altul;
- Dacă prin adunarea caracterelor de pe o coloana se depăşeşte valoarea 7 numărul obţinut se scrie ca o sumă de 2 numere (un număr reprezintă baza sistemului adică 8 iar celălalt reprezintă valoarea cu care s-a depăşit baza) astfel:
  8 = 8 + 0 ; 9 = 8 + 1 ; 10 = 8 + 2 ; ..... 14 = 8 + 6;
- Numărul care reprezintă *baza* (care are valoarea în octal 1) *se transportă* deasupra următoarei coloane din stânga;
- Suma cifrelor de pe coloana respectivă se adună cu transportorul de deasupra coloanei care ATENŢIE! are valoarea 1;
- Numărul care reprezintă *valoarea cu care s-a depăşit baza* este *rezultatul adunării* de pe coloana respectivă în cazul în care suma numerelor de pe coloana respectivă este mai mare decât 7;
- Dacă suma numerelor de pe o coloană este mai mică sau egală cu 7, rezultatul obţinut reprezintă *rezultatul adunării* de pe coloana respectivă.

Exemple de adunare a două numere octale

Exemplul 1.

```
    1    1    0
    3    7    2    1₈          3721₈ + 1363₈ = 5305₈
+   1    3    6    4₈
-------------------------
(5+0) (8+3) (8+0) (5+0)
  5    3    0    5₈
```

1 + 4 = 5 transport 0 ⇒ prima cifră (din dreapta) este 5

2 + 6 + 0 = 8 = 8 + 0 = 0 transport 1 ⇒ a doua cifră este 0

7 + 3 + 1= 11 = 8 + 3 = 3 transport 1 ⇒ a treia cifră este 3

3 + 1 + 1 = 5 transport 0 ⇒ a patra cifră este 5

Exemplul 2.

```
   1
   1 7 0 2₈
+  2 1 3 1₈
-------------
   4 0 3 3₈
```

Exemplul 3.

```
 1 1 1
     5 7 5₈
+    2 7 6₈
-------------
 1  0 7 3₈
```

Exemplul 4.

```
 1 1
     2 7₈
+    7 7₈
-------------
 1  2  6₈
```

B. SCĂDEREA NUMERELOR OCTALE

Reguli:

- Scăderea se face ca în sistemul zecimal, prin scrierea numerelor unul sub altul;
- Dacă prin scăderea caracterelor de pe o coloana rezultatul obţinut este negativ (numărul de sus este mai mic decât numărul de jos), se împrumută de pe următoarea coloană din stânga o unitate în octal care înseamnă opt unităţi în zecimal;
- Se face suma algebrică dintre împrumut şi numerele de pe coloana respectivă iar în urma calculului se obţine cifra corespunzătoare rezultatului de pe acea coloană;
- Unitatea (1) împrumutată de pe o coloană se scade din cifra de sus a coloanei de unde a fost împrumutată.

Exemple de adunare a două numere octale

Exemplul 1.

```
  -1      -1
   4       5       3₈        457₈ – 264₈ = 167₈
 - 2       6       4₈
-------------------------
(4-1-2=1) (8+5-1-6=6) (8+3-4=7)
   1       6       7₈
```

- Scad numerele de pe coloana din dreapta ⇒ 3 – 4 < 0 ⇒ împrumut o unitate octală de pe coloana din mijloc;
- Adun împrumutul la diferenţa numerelor de pe coloană ⇒ 8+3-4=7 ⇒ cifra 7;
- Scad din diferenţa numerelor de pe coloana din mijloc împrumutul ⇒ 5 – 6 – 1 < 0 ⇒ împrumut o unitate octală de pe coloana din stânga;
- Adun împrumutul la diferenţa numerelor de pe coloană 8+5-6-1=6 ⇒ cifra 6;
- Din diferenţa numerelor de pe coloana din stânga scad unitatea împrumutată ⇒ 4 – 2 -1 = 1 ⇒ cifra 1.

Exemplul 2.

```
  -1 -1
   6  1  2₈
 - 4  5  7₈
-------------
   1  3  3₈
```

Exemplul 3.

```
   -1
    5 3 2₈
  - 2 5 1₈
-------------
    2 6 1₈
```

Exemplul 4.

```
    -1
    3 6 2₈
  - 1 3 8₈
-------------
    2 2 2₈
```

C. ADUNAREA NUMERELOR HEXAZECIMALE

Reguli:

- Adunarea se face ca în sistemul zecimal, prin scrierea numerelor unul sub altul
- Înainte de a efectua adunările, caracterele alfabetice (A, B,C,D,E,F) se înlocuiesc cu valorile lor în zecimal (10,11,12,13,14,15) – vezi tabelul 1.3 din secţiunea 1.1;
- Dacă prin adunarea caracterelor de pe o coloana se depăşeşte valoarea 15 numărul obţinut se scrie ca o sumă de 2 numere (un număr reprezintă baza sistemului adică 16 iar celălalt reprezintă valoarea cu care s-a depăşit baza) astfel:

  16 = 16 + 0 ; 17 = 16 + 1 ; 18 = 16 + 2 ; ..... 31 = 16 + 15;

- Numărul care reprezintă *baza* (care are valoarea în hexazecimal 1) *se transportă* deasupra următoarei coloane din stânga;
- Suma cifrelor de pe coloana respectivă se adună cu transportorul de deasupra coloanei care ATENŢIE! are valoarea 1;
- Rezultatul adunării se transformă în hexazecimal (conform tabelului 1.1 din secţiunea 3) şi reprezintă *rezultatul adunării* de pe coloana respectivă.

Exemple de adunare a două numere hexazecimale

Exemplul 1.

$$\begin{array}{r} 6\ D\ 8\ A\ 3\ 2_{16} \\ +\ 3\ \ 3\ E\ 4\ C\ 8_{16} \\ \hline \end{array} \Rightarrow \begin{array}{r} \quad +1\ \ +1 \\ 6\ \ 13\ \ 8\ \ 10\ \ 3\ \ 2 \\ +\ 3\ \ 3\ \ 15\ \ 4\ \ 12\ \ 8 \\ \hline 10\ \ (16+1)\ \ (16+6)\ \ 14\ \ 15\ \ 10 \end{array} \Rightarrow \begin{array}{r} 6\ D\ 8\ A\ 3\ 2_{16} \\ +\ 3\ \ 3\ E\ 4\ C\ 8_{16} \\ \hline A\ 1\ 6\ E\ F\ A_{16} \end{array}$$

$2 + 8 = 10 = A_{16}$ transport 0 $\Rightarrow$ prima cifră (din dreapta) este A

$3 + 12 = 15 = F$ transport 0 $\Rightarrow$ a doua cifră este F

$10 + 4 = 14 = E$ transport 0 $\Rightarrow$ a treia cifră este E

$8 + 14 = 22 = 16 + 6 = 6$ transport 1 $\Rightarrow$ a patra cifră este 6

$13 + 3 + 1 = 17 = 16 + 1 = 1$ transport 1 $\Rightarrow$ a cincea cifră este 1

$6 + 3 + 1 = 10 = A$ transport 0 $\Rightarrow$ a şasea cifră este A

Exemplul 2.

$$\begin{array}{r} 1\ 1\ 1 \\ A\ 3\ D\ 4_{16} \\ +\ C\ F\ E\ B_{16} \\ \hline 1\ 7\ 3\ B\ F_{16} \end{array}$$

Exemplul 3.

$$\begin{array}{r} 1\ 1\ 1 \\ 2\ A\ 5\ 7_{16} \\ +\ 5\ 7\ B\ 9_{16} \\ \hline 8\ 2\ 1\ 0_{16} \end{array}$$

Exemplul 4.

$$\begin{array}{r} 1\ \ 1 \\ 1\ 9\ B\ 9_{16} \\ +\ C\ 7\ E\ 6_{16} \\ \hline E\ 1\ 9\ F_{16} \end{array}$$

D. SCĂDEREA NUMERELOR HEXAZECIMALE

Reguli:

- Scăderea se face ca în sistemul zecimal, prin scrierea numerelor unul sub altul;
- Înainte de a efectua scăderile, caracterele alfabetice (A, B,C,D,E,F) se înlocuiesc cu valorile lor în zecimal (10,11,12,13,14,15) – vezi tabelul 1.3 din secţiunea 1.1;
- Dacă prin scăderea caracterelor de pe o coloana rezultatul obţinut este negativ (numărul de sus este mai mic decât numărul de jos), se împrumută de pe

următoarea coloană din stânga o unitate în hexazecimal care înseamnă 16 unităţi în zecimal;

- Se face suma algebrică dintre împrumut şi numerele de pe coloana respectivă iar în urma calculului se obţine cifra corespunzătoare rezultatului de pe acea coloană;
- Unitatea (1) împrumutată de pe o coloană se scade din cifra de sus a coloanei de unde a fost împrumutată.

Exemple de scădere a două numere hexazecimale

Exemplul 1.

$$\begin{array}{r} 5\ C\ 2\ B_{16} \\ -\ 3\ A\ C\ F_{16} \\ \hline 2\ 1\ 5\ C_{16} \end{array} \Leftrightarrow \begin{array}{cccc} & \overset{-1}{} & \overset{-1}{} & \\ 5 & 12 & 2 & 11_{16} \\ -\ 3 & 10 & 12 & 15_{16} \\ \hline (5-3=2) & (12-1-10=1) & (16+2-1-12=5) & (16+11-15=12=C) \\ 2 & 1 & 5 & C_8 \end{array}$$

- Scad numerele de pe coloana din dreapta $\Rightarrow 11 - 15 < 0 \Rightarrow$ împrumut o unitate hexazecimală de pe coloana din mijloc;
- Adun împrumutul la diferenţa numerelor de pe coloană $\Rightarrow 16 + 11 - 15 = 12 = C$;
- Fac suma algebrică a numerelor de pe următoarea coloană din stânga

$\Rightarrow 2 - 1 - 12 < 0 \Rightarrow$ împrumut o unitate hexazecimală de pe următoarea coloană;

- Adun împrumutul la diferenţa numerelor de pe coloană $16 + 2 - 1 - 12 = 5$;
- Fac suma algebrică a numerelor de pe următoarea coloană din stânga

$\Rightarrow 12 - 1 - 10 = 1$

- Fac suma algebrică a numerelor de pe următoarea coloană din stânga

$\Rightarrow 5 - 3 = 2$

Exemplul 2.

$$\begin{array}{r} \overset{-1}{A}\ \overset{-1}{3}\ D\ 4_{16} \\ -\ 7\ 5\ 1\ B_{16} \\ \hline 2\ E\ B\ 9_{16} \end{array}$$

Exemplul 3.

$$\begin{array}{r} \overset{-1}{C}\ \overset{-1}{E}\ D\ 0_{16} \\ -\ 1\ F\ 0\ C_{16} \\ \hline B\ F\ C\ 4_{16} \end{array}$$

Exemplul 4.

$$\begin{array}{r} \overset{-1}{F}\ 2\ C\ 3_{16} \\ -\ 9\ D\ 6\ E_{16} \\ \hline 5\ 5\ 5\ 5_{16} \end{array}$$

1.4. CODAREA NUMERELOR BINARE

Codificare presupune realizarea unei schimbări a formei de exprimare a informaţiei, altfel spus o translatare de limbaj.

1.4.1 REPREZENTAREA ÎN SISTEM BINAR A NUMERELOR NEGATIVE

Pentru reprezentarea în binar a unui număr negativ, primul bit din stânga reprezentării numărului este utilizat ca bit de semn astfel:

0 pentru numere **pozitive (+)**

1 pentru numere **negative (-)**

A. CODUL DIRECT

Pentru numerele negative cu **n** biţi, bitul de semn este **1** iar ceilalţi **n-1** biţi servesc pentru reprezentarea valorii absolute a numărului.

Exemplu: Reprezentarea numărului **-5** pe opt biţi în cod direct.

Convertim numărul 5 din baza 10 în baza 2 $\Rightarrow$ $5_{10} = 101_2$

Valoarea absolută a numărului **- 5** reprezentat pe 8 biţi este $\mathbf{00000101_2}$

Pentru numărul **– 5** primul bit din stânga este **1**

Numărul - 5 pe opt biţi în cod direct are valoarea $\mathbf{10000101_2}$

B. CODUL INVERS (complement faţă de 1)

Pentru numerele negative cu **n** biţi, bitul de semn este **1** iar ceilalţi **n-1** biţi servesc pentru reprezentarea valorii absolute **NEGATE** a numărului. Negarea se realizează la nivel de bit prin transformarea biţilor **0** în **1** şi a biţilor **1** în **0**.

Exemplu: Reprezentarea numărului **- 5** pe opt biţi în cod invers

Valoarea absolută a numărului **– 5** este **0000101**.

Valoarea absolută **NEGATĂ** a numărului **– 5** este **1111010**

Pentru numărul **– 5** primul bit din stânga este **1**

Numărul - 5 pe opt biţi în cod invers are valoarea $\mathbf{11111010_2}$

Valoarea numerică a unui număr negativ **N** reprezentat pe **n** biţi în cod invers se calculează cu formula:

$$C_1(N) = 2^n - 1 - V$$

unde: **n** – este numărul de biţi al reprezentării

V – este valoarea absolută a numărului reprezentat.

Exemplu: Valoarea numerică numărului **- 5** pe opt biţi în cod invers

$$C_1(N) = 2^8 - 1 - 5 = 256 - 1 - 5 = 250$$

$\mathbf{11111010_2 = 1 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 250_{10}}$

C. CODUL COMPLEMENTAR (complement faţă de 2)

Pentru reprezentarea numerelor negative în cod complementar se parcurg etapele:

Se reprezintă numărul negativ în valoare absolută pe opt biţi

Se transformă biţii **0** în **1** şi biţii **1** în **0**

Rezultatul obţinut se adună cu **1**

Exemplu: Reprezentarea numărului **- 5** pe opt biţi în cod complementar

Valoarea absolută a numărului **– 5** este **|- 5| = 5**

Numărul **5** în sistem binar pe opt biţi are valoarea **00000101**

După transformare se obţine numărul **11111010**

Adunăm numărul obţinut cu **1**

$$\begin{array}{r} 11111010 + \\ \underline{\qquad\quad 1} \\ 11111011 \end{array}$$

Numărul negativ – 5 în cod complementar are valoarea 11111011

Valoarea numerică a unui număr negativ **N** reprezentat pe **n** biţi în cod complementar se calculează cu formula:

$$C_2(N) = 2^n - V$$

unde: **n** – este numărul de biţi al reprezentării

V – este valoarea absolută a numărului reprezentat.

Exemplu: Valoarea numerică numărului **- 5** pe opt biţi în cod complementar.

$$C_2(N) = 2^8 - 5 = 256 - 5 = 251$$

$$11111011_2 = 1\cdot2^7 + 1\cdot2^6 + 1\cdot2^5 + 1\cdot2^4 + 1\cdot2^3 + 0\cdot2^2 + 1\cdot2^1 + 1\cdot2^0 = 251_{10}$$

CONCLUZII:

În codul complementar ***bitul din stânga*** rămâne întotdeauna ***bit de semn***.

Avantajul reprezentării numerelor în cod complementar faţă de reprezentarea în celelalte coduri este că prin adunarea numărului reprezentat cu complementul său faţă de 2 se obţine rezultatul **0**.

Codul complementar este cel mai utilizat pentru reprezentarea numerelor algebrice în calculator.

1.4.2 CODURI NUMERICE

Sistemele digitale efectuează calculele interne cu ajutorul numerelor binare dar majoritatea utilizatorilor preferă să lucreze cu numere zecimale. Din această cauză au fost create interfeţe cu exteriorul a sistemelor digitale care pot prelua, prelucra şi afişa valori zecimale.

Prin urmare un număr zecimal este reprezentat într-un sistem digital printr-un şir de biţi, diverse combinaţii ale valorilor din şir reprezentând diferite numere zecimale. Mulţimea formată din şiruri de *n* biţi, în care fiecare şir de biţi reprezintă câte un număr sau element, se numeşte ***COD***.

O combinaţie determinată de valorile a *n* biţi se numeşte ***CUVÂNT DE COD***.

Pentru reprezentarea cifrelor sistemului de numeraţie zecimal sunt necesari minimum 4 biţi deoarece numărul de cifre zecimale este 10, iar acest număr este mai mare decât $\mathbf{2^3}$ care se reprezintă pe **4** biţi.

A. CODURI ZECIMAL – BINARE (BCD)

În clasa de coduri zecimal-binare (**B**inary **C**oded **D**ecimal) mulţimea **X** a sursei primare de informaţii care trebuie codificată este formată din simbolurile cifrelor sistemului zecimal, iar mulţimea cuvintelor de cod trebuie să conţină cel puţin 10 cuvinte distincte.

X = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}.

Cuvintele de cod trebuie să aibă cel puţin **4 biţi,** deoarece $\mathbf{2^3} < 10 < \mathbf{2^4} = \mathbf{16}$

Stabilind corespondenţa între cele **10** cifre ale sistemului zecimal şi cele **16** cuvinte binare de **4 biţi**, se pot obţine în total $\boldsymbol{C_{16}^{10}}$ **= 29.059.430.400** posibilităţi de codificare.

Codurile zecimal – binare se clasifică astfel (vezi **tabelul 1.6**):

- Coduri ponderate:
  - Codul 8421;
  - Codul 2421;
  - Codul 4221;
  - Codul 7421;
- Coduri neponderate:
  - Codul Exces 3;
  - Codul Gray;
  - Codul 2 din 5;
  - Codul 8421 cu bit de paritate.

Tabelul 1.6. Coduri zecimal-binare

Numere în zecimal	CODURI ZECIMAL-BINARE							
	Coduri ponderate				Coduri neponderate			
	8421	2421	4221	7421	Exces3	Gray	2 din5	8421 cu bit de paritate impară
0	0000	0000	0000	0000	0011	0000	00011	10000
1	0001	0001	0001	0001	0100	0001	00101	00001
2	0010	0010	0010	0010	0101	0011	00110	00010
3	0011	0011	0011	0011	0110	0010	01001	10011
4	0100	0100	0100	0100	0111	0110	01010	00100
5	0101	1011	1001	0101	1000	0111	01100	10101
6	0110	1100	1100	0110	1001	0101	10001	10110
7	0111	1101	1101	0111	1010	0100	10010	00111
8	1000	1110	1110	1001	1011	1100	10100	01000
9	1001	1111	1111	1010	1100	1101	11000	11001

A1. CODURI PONDERATE

Cel mai utilizat cod ponderat este codul 8421. Acest cod se mai numeşte codul zecimal-binar natural **NBCD** (**N**atural-**B**inary-**C**oded-**D**ecimal), în terminologia curentă este definit impropriu doar codul **BCD**.

Bitul **0** are ponderea **1(2^0)**, bitul **1** are ponderea **2 (2^1)**, bitul **2** are ponderea **4 (2^2)**, bitul **3** are ponderea **8 (2^3)**. Deci în codul 8421 ponderile biţilor sunt 8, 4, 2, 1.

Se observă că ponderea unui bit este egală cu notaţia codului corespunzătoare bitului respectiv.

Aceeaşi regulă de fixare a ponderii bitului din cuvântul de cod, egală cu cea din notaţia codului, se respectă la toate celelalte coduri ponderate.

După cum se observă din **Tabelul 1.6** pentru fiecare caracter zecimal corespunde un cod de 4 biţi. Pentru a transforma codul binar în număr zecimal se înmulţeşte baza sistemului binar (**2**) cu ponderea bitului corespunzător şi se adună rezultatele.

Exemple:

Codul $\mathbf{0111_{8421}}$ se scrie $\mathbf{0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 0 + 4 + 2 + 1 = 7}$

Codul $\mathbf{0111_{8421}}$ se mai poate scrie $\mathbf{0 \cdot 8 + 1 \cdot 4 + 1 \cdot 2 + 1 \cdot 1 = 0 + 4 + 2 + 1 = 7}$

Codul $\mathbf{1110_{2421}}$ se scrie $\mathbf{1 \cdot 2^1 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 2 + 4 + 2 + 0 = 8}$

Codul $\mathbf{1110_{2421}}$ se mai poate scrie $\mathbf{1 \cdot 2 + 1 \cdot 4 + 1 \cdot 2 + 0 \cdot 1 = 2 + 4 + 2 + 1 = 8}$

Codul $\mathbf{1101_{4221}}$ se scrie $\mathbf{1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^1 + 1 \cdot 2^0 = 4 + 2 + 0 + 1 = 7}$

Codul $\mathbf{1101_{4221}}$ se mai poate scrie $\mathbf{1 \cdot 4 + 1 \cdot 2 + 0 \cdot 2 + 1 \cdot 1 = 4 + 2 + 0 + 1 = 7}$

Codul $\mathbf{1010_{7421}}$ se scrie $\mathbf{1 \cdot 7 + 0 \cdot 4 + 1 \cdot 2 + 0 \cdot 1 = 7 + 0 + 2 + 0 = 9}$

Numerele pot fi reprezentate în **BCD** prin cuvinte de orice lungime folosindu-se câte **1 octet (8 biţi)** pentru fiecare combinaţie de două cifre. Numerele **BCD** precedate de semn prezintă un bit suplimentar pentru semn (primul bit din stânga).

A2. CODURI NEPONDERATE

1. Codul EXCES 3

Codul EXCES 3 se obţine din cuvântul de cod **8421**, al cifrei zecimale respective, la care se adună **0011**, adică **3** în binar.

EXEMPLU:

Reprezentarea cifrei 8 în cod EXCES 3.

Cifra **8** în codul **8421** are valoarea **1000**

Pentru reprezentarea în codul EXCES 3 se adună **1000 + 0011 = 1011**

Valoarea cifrei **8** în codul **EXCES 3** este **1011**

Utilizând codul **EXCES 3**, se poate face distincţie între lipsa unei informaţii înscrise într-un registru sau locaţie de memorie şi înscrierea valorii zero. (0000 reprezintă lipsa unei informaţii, iar zero este codificat prin 0011)

2. Codul 2 din 5

Acest cod se utilizează pentru reprezentarea numerelor zecimale printr-un grup de **5 biţi** din care numai doi biţi sunt semnificativi (au valorile egale cu **1**). În acest fel se realizează o unicitate a reprezentării, deoarece din cele 32 numere posibile cu 5 biţi (2^5) numai 10 satisfac condiţia 2 din 5. Numerele care satisfac condiţia **2 din 5** sunt prezentate în **tabelul 1.6**.

Acest cod creează posibilitatea *detectării erorilor multiple la transmiterea informaţiei*.

3. Codul 8421 cu bit de paritate.

Acest cod este un ***cod detector de erori***. Codul conţine un bit suplimentar numit **bit de paritate** care este primul bit din stânga numărului reprezentat în acest cod. Codul se obţine din codul **8421** prin adăugarea unui bit de paritate în faţa codului **8421** care reprezintă un anumit număr. Bitul de paritate se poate alege astfel încât numărul total al biţilor cu valoare **1**, în exprimarea numărului, să fie **par** respectiv **impar**.

Acest cod se utilizează pentru *verificarea transmiterii corecte a informaţiei*

4. Codul GRAY

Codul Gray este un cod digital care acceptă modificarea unui singur bit din cuvântul de cod, la trecerea dintre două cuvinte de cod succesive (trecerea de la o cifră zecimală la următoarea cifră zecimală).

Această proprietate face ca acest cod să fie utilizat la dispozitivele de codare circulare (diverse traductoare unghiulare de poziţie).

Codul **gray** se obţine din codul **8421** astfel (vezi **tabelul 1.7**):

- $\mathbf{G_0}$ – repetă primele **două** locaţii ale lui $\mathbf{B_0}$, după care se reflectă din două în două locaţii astfel: **01 10 01 10 01 10 01 10;**
- $\mathbf{G_1}$ – repetă primele **patru** locaţii ale lui $\mathbf{B_1}$**,** după care se reflectă din patru în patru locaţii astfel: **0011 1100 0011 1100;**
- $\mathbf{G_2}$ – repetă primele **opt** locaţii ale lui $\mathbf{B_2}$**,** după care se reflectă din opt în opt astfel: **00001111 11110000;**
- $\mathbf{G_3}$ – repetă $\mathbf{B_3}$.

Tabelul 1.7 – Tabelul de adevăr al convertorului de cod **8421 – gray**

Număr zecimal	CODUL 8421				CODUL GRAY			
	B3	B2	B1	B0	G3	G2	G1	G0
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	1	0	0
9	1	0	0	1	1	1	0	1
10	1	0	1	0	1	1	1	1
11	1	0	1	1	1	1	1	0
12	1	1	0	0	1	0	1	0
13	1	1	0	1	1	0	1	1
14	1	1	1	0	1	0	0	1
15	1	1	1	1	1	0	0	0

Codul **Gray** are proprietatea de adiacenţă, adică trecerea de la o cifră zecimală la următoarea sau precedenta necesită modificarea unui singur bit din cuvântul de cod.
Codul **Gray** este util pentru mărimile care cresc sau descresc succesiv.

1.4.3 CODURI ALFANUMERICE

Codurile alfanumerice conţin cifre, litere şi semne speciale care se numesc ***caractere***.

Cel mai utilizat cod alfanumeric este codul **ASCII** *(The American Standard Code for Information Interchange – codul american standardizat pentru schimbul de informaţii)*

Codul **ASCII** utilizează **7 biţi** pentru a codifica 128 de caractere diferite (vezi **Tabelul 1.8**).

Codul **ASCII** conţine litere mari, litere mici, cifre, sisteme de punctuaţie şi diverse caractere de comandă care nu se tipăresc.

Tabelul 1.8 – Codul **ASCII**

$b_3b_2b_1b_0$	$b_6\ b_4\ b_5$							
	000	**001**	**010**	**011**	**100**	**101**	**110**	**111**
0000	NULL	DLE		0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

EXEMPLE de reprezentare în ASCII a caracterelor:

C – 100 0011 (coloana **100** linia **0011**)

& – 010 0110 (coloana **010** linia **0110**)

9 - 011 1001 (coloana **011** linia **1001**).

REZUMATUL CAPITOLULUI

- ***Sistemul de numerație zecimal*** utilizează **10 caractere (cifre)** şi are baza **10** deoarece pentru reprezentarea unui număr în acest sistem sunt permise 10 caractere (0, 1, 2, 3, 4, 5, 6, 7, 8, 9).
- ***Sistemul de numerație binar*** este un sistem de numeraţie poziţional care utilizează **2** caractere (**0** şi **1**) şi are baza **2**.
- ***Sistemul de numeraţie octal*** utilizează **8 caractere** (0, 1, 2, 3, 4, 5, 6, 7) şi are baza **8**.
- La fiecare caracter din sistemul de numeraţie octal îi corespunde un şir de 3 biţi:
  0 ⇔ 000 = 0x2^2 + 0x2^1 + 0x2^0, …….., 7 ⇔ 111= 1x2^2 + 1x2^1 + 1x2^0 **= 4+2+1 = 7**
- ***Sistemul de numeraţie hexazecimal*** utilizează 16 caractere (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F) şi are baza **16**.
- La fiecare caracter din sistemul de numeraţie hexazecimal îi corespunde un şir de 4 biţi:
  0 ⇔ 0000 = 0x2^3+0x2^2+0x2^1+0x2^0, ….., **F ⇔ 1111 = 1x2^3+**1x2^2+1x2^1+1x2^0 **= 15**
- ***Conversia unui număr din altă bază într-un număr în baza 10***:
  - ***11011_2*** = 1x2^4 + 1x2^3 + 0x2^2 + 1x2^1 + 1x2^0 = 16 + 8 + 0 + 2 + 1 = ***27_{10}***
  - ***7021_8*** = 7x8^3 + 0x8^2 + 2x8^1 + 1x8^0 = 3584 + 0 + 16 + 1 = ***3601_{10}***
  - ***$1AF_{16}$*** = 1x16^2 + Ax16^1 + Fx16^0 = 256 + 10x16 + 15x1 = 256+160+15 = ***431_{10}***
- ***Conversia unui număr din baza 10 într-un număr din altă bază:***
  - Pentru conversia în baza 2 se împarte succesiv la 2 astfel:

    83 : 2 = 41 rest **1**
    41:2 = 20 rest **1**
    20:2 = 10 rest 0
    10:2 = 5 rest 0
    5:2 = 2 rest 1
    2:2 = 1 rest **0**
    1:2 = 0 rest **1**

    $83_{10} = 1010011_2$

  - Pentru conversia în baza 8 se împarte succesiv la 8 astfel:

    1080 : 8 = 135 rest **0**
    135:8 = 16 rest **7**
    16:8 = 2 rest **0**
    2:8 = 0 rest **2**

    $1080_{10} = 2070_8$

- Pentru conversia în baza 16 se împarte succesiv la 16 astfel:

$254_{10} = FE_8$.

- Pentru **conversia numerelor binare în numere octale** se împart biţii numărului binar în grupe de câte 3 pornind de la dreapta (sau de la virgulă) spre stânga:

10011100_2 = 010 011 100 = ***234_8***.

- Pentru **conversia numerelor octale în numere binare** se înlocuieşte fiecare caracter din octal cu şirul corespunzător de 3 biţi:

742_8 = 111 100 010 = ***111100010_2***

- Pentru **conversia numerelor binare în numere hexazecimale** se împart biţii numărului binar în grupe de câte 4 biţi de la dreapta la stânga (pentru completarea primei grupe din stânga se adaugă **0**) :

1010011_2 = 0101 0011 = ***53_{16}***

- Pentru **conversia numerelor hexazecimale în numere binare** se înlocuieşte fiecare caracter din hexazecimal cu şirul corespunzător de 4 biţi:

$DAC12_{16}$ = **1101 1010 1100 0001 0010 =** ***1101101011000010010_2*****.**

- Reguli de bază la adunarea numerelor binare:
  - 0 + 0 = 0 transport 0;
  - 0 + 1 = 1 transport 0;
  - 1 + 0 = 1 transport 0;
  - 1 + 1 = 0 transport 1.
- Reguli de bază la scăderea numerelor binare:
  - 0 - 0 = 0 împrumut 0;
  - 1 - 1 = 0 împrumut 0;
  - 1 - 0 = 1 împrumut 0;
  - 0 - 1 = 1 împrumut 1.
- Reguli de bază la înmulțirea numerelor binare:
  - 0 x 0 = 0;
  - 1 x 0 = 0;
  - 0 x 1 = 0;
  - 1 x 1 = 1.

EVALUAREA CUNOȘTINȚELOR

1. Efectuați următoarele conversii între sisteme de numerație:
   a. $10110111_2 = ?_8$;
   b. $174003_8 = ?_2$;
   c. $1101011010_2 = ?_{16}$;
   d. $FA35_{16} = ?_2$;
   e. $1100011_2 = ?_{10}$;
   f. $12345_{10} = ?_{16}$;
   g. $255_{10} = ?_2$;
   h. $7531_{10} = ?_8$;
   i. $2467_8 = ?_{16}$;
   j. $FE12A_{16} = ?_8$;
2. Convertiți următoarele numere din octal în binar și hexazecimal:
   a. $713621_8 = ?_2 = ?_{16}$;
   b. $13045_8 = ?_2 = ?_{16}$;
   c. $2304_8 = ?_2 = ?_{16}$;
   d. $777_8 = ?_2 = ?_{16}$;
   e. $111{,}111_8 = ?_2 = ?_{16}$;
3. Convertiți următoarele numere din hexazecimal în binar și octal:
   a. $BABA_{16} = ?_2 = ?_8$;
   b. $F1E2_{16} = ?_2 = ?_8$;
   c. $9B8C2_{16} = ?_2 = ?_8$;
   d. $89D67A_{16} = ?_2 = ?_8$;
   e. $DEAD,BEEF_{16} = ?_2 = ?_8$;
4. Adunați următoarele perechi de numere binare:
   a. 111001 + 10001 = ?;
   b. 1001100 + 111110 = ?;
   c. 11110000 + 10000001 = ?;
   d. 101010 + 101010 = ?
5. Adunați următoarele perechi de numere hexazecimale:
   a. F35B + 27E6 = ?;
   b. B9D4 + 4F5A = ?;
   c. 1234 + ABCD =?;
   d. AB67 + EF89 = ?.

CAPITOLUL 2. FUNCŢII LOGICE

2.1 AXIOMELE ŞI TEOREMELE ALGEBREI LOGICE

Algebra logică are la bază ***principiul dualităţii*** potrivit căruia toate axiomele şi teoremele rămân valabile dacă se fac schimbările **"+"** cu **"•"** respectiv **"0"** cu **"1"**.
Semnul **"+"** reprezintă **ADUNARE** logică. Semnul **"•"** reprezintă **ÎNMULŢIRE** logică.
Conform principiului dualităţii fiecare axiomă şi teoremă are două forme.

AXIOMELE ALGEBREI LOGICE

1. ASOCIATIVITATEA: **(A+B)+C = A+(B+C) = A+B+C (A•B)•C = A•(B•C) = A•B•C**

2. COMUTATIVITATEA: **A + B = B + A A • B = B • A**

3. DISTRIBUTIVITATEA: **A • (B+C) = A • B + A • C A + B • C = (A+B) • (A+C)**

4. ELEMENT NEUTRU: **A + 0 = 0 + A = A A • 1 = 1 • A = A**

5. COMPLEMENTUL: **A + $\overline{A}$ = 1 A • $\overline{A}$ = 0**

TEOREMELE ALGEBREI LOGICE

1. IDEMPOTENŢA

A + A + A + ...+ A = A A • A • A • .....• A = A

2. ELEMENTE NEUTRE

A + 1 = 1 A • 0 = 0

3. ABSORBŢIA

A + A • B = A A • (A + B) = A

4. ABSORBŢIA INVERSĂ

$\overline{A} + \overline{A} \cdot B = \overline{A}$ $\overline{A} \cdot (\overline{A} + B) = \overline{A}$

A + $\overline{A}$ • B = A + B A • ($\overline{A}$ + B) = A • B

5. DUBLA NEGAŢIE (INVOLUŢIA)

$\overline{\overline{A}} = A$

6. TEOREMELE LUI DE MORGAN

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

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

Pentru înţelegerea şi demonstrarea axiomelor, teoremelor sau a altor relaţii în algebra logică se ţine cont de următoarele reguli:

A şi **B** pot fi înlocuite cu **0** sau **1**. Dacă **A = 0** atunci **B = 1** şi invers

0 • 0 = 0 0 + 0 = 0 0 • 1 = 0 $\overline{0} = 1$

1 • 1 = 1 1 + 1 = 1 1 • 0 = 0 $\overline{1} = 0$

2.2 PREZENTAREA FUNCŢIILOR LOGICE

Algebra booleană operează pe o mulţime **B = { x I x ∈ {0,1} }.**

În această mulţime se definesc 3 legi de compoziţie:

Complementarea (inversarea logică, negarea , "NU", „**NOT**"**)**

Disjuncţia (suma logică, reuniunea , „SAU", „**OR**")

Conjuncţia (produsul logic, intersecţia, „ŞI" , „**AND**")

O funcţie ***f* : $B^n \rightarrow B$** se numeşte ***funcţie booleană***.

O funcţie booleană de **n** variabile **$y = f(x_1, x_2, x_3, ...x_n)$** se caracterizează prin faptul că atât variabilele cât şi funcţia nu pot lua decât două valori distincte **0** şi **1**.

Din cele prezentate mai sus rezultă că în algebra booleană sunt **trei** funcţii elementare:

Funcţia NU → (NOT) → NEGAŢIE
Funcţia SAU → (OR) → ADUNARE
Funcţia ŞI → (AND) → ÎNMULŢIRE

Prin combinarea celor trei funcţii logice elementare se mai obţin încă patru funcţii logice:

Funcţia SAU – NU → (NOR) → NEGAREA SUMEI LOGICE
Funcţia ŞI – NU → (NAND) → NEGAREA PRODUSULUI LOGIC
Funcţia SAU – EXCLUSIV → (XOR) → SUMA MODULO 2
Funcţia SAU – EXCLUSIV - NU → (NXOR) → NEGARE SUMĂ MODULO 2

În **tabelul 2.1** sunt prezentate funcţiile logice elementare utilizate în algebra logică.

Tabelul 2.1 – FUNCŢII LOGICE ELEMENTARE

Nr. crt.	Denumirea funcţiei logice	Operaţia realizată	Expresia funcţiei logice
1	NU (NOT)	Inversare	$Y = \overline{A}$
2	SAU (OR)	Sumă logică	$Y = A + B$
3	ŞI (AND)	Produs logic	$Y = A \cdot B$
4	SAU - NU (NOR)	Negarea sumei logice	$Y = \overline{A + B}$
5	ŞI – NU (NAND)	Negarea produsului logic	$Y = \overline{A \cdot B}$
6	SAU - EXCLUSIV (XOR)	Sumă modulo 2	$Y = A \oplus B$
7	SAU - EXCLUSIV NEGAT (NXOR)	Negarea sumei modulo 2	$Y = \overline{A \oplus B}$

Funcţiile logice de bază prezentate mai sus se implementează (realizează) cu ajutorul unor circuite fizice numite ***porţi logice***.

Aceste dispozitive sunt prezentate în capitolul **PORŢI LOGICE** .

2.3 REPREZENTAREA FUNCŢIILOR LOGICE

Pentru reprezentarea funcţiilor se folosesc în mod curent 2 metode:

- Reprezentarea prin tabela de adevăr;
- Reprezentarea prin diagrame Veitch – Karnaugh.

2.3.1. REPREZENTAREA FUNCŢIILOR LOGICE PRIN TABELA DE ADEVĂR

TABELA DE ADEVĂR - stabileşte corespondenţa dintre valorile de adevăr ale variabilelor de intrare şi valoarea de adevăr a funcţiei în fiecare punct al domeniului de definiţie.

TABELUL DE ADEVĂR AL FUNCŢIEI NU (NOT)

A	$Y = \overline{A}$
0	1
1	0

TABELUL DE ADEVĂR AL FUNCŢIEI SAU (OR)

A	B	Y = A + B
0	0	0
0	1	1
1	0	1
1	1	1

TABELUL DE ADEVĂR AL FUNCŢIEI ŞI (AND)

A	B	Y = A • B
0	0	0
0	1	0
1	0	0
1	1	1

TABELUL DE ADEVĂR
AL FUNCŢIEI SAU - NU (NOR)

A	B	Y = $\overline{A + B}$
0	0	1
0	1	0
1	0	0
1	1	0

TABELUL DE ADEVĂR
AL FUNCŢIEI ŞI - NU (NAND)

A	B	Y = $\overline{A \bullet B}$
0	0	1
0	1	1
1	0	1
1	1	0

TABELUL DE ADEVĂR
AL FUNCŢIEI SAU-EXCLUSIV
(XOR)

A	B	Y = $A \oplus B$ $Y = \overline{A} \cdot B + A \cdot \overline{B}$
0	0	0
0	1	1
1	0	1
1	1	0

TABELUL DE ADEVĂR
AL FUNCŢIEI SAU-EXCLUSIV- NEGAT
(NXOR)

A	B	Y = $\overline{A \oplus B}$ $Y = A \cdot B + \overline{A} \cdot \overline{B}$
0	0	1
0	1	0
1	0	0
1	1	1

2.3.2 REPREZENTAREA FUNCŢIILOR LOGICE PRIN DIAGRAME VEITCH – KARNAUGH

Diagramele Veitch - Karnaugh se utilizează pentru minimizarea unei funcţii logice, în scopul obţinerii unei expresii algebrice cât mai simple, care permite implementarea unui circuit digital cu un număr minim de porţi logice.

Diagrama Karnaugh simplifică o funcţie logică cu mai multe intrări (maxim 8).

O ***diagramă Karnaugh*** este o reprezentare grafică a tabelului de adevăr corespunzător unei funcţii logice. Diagrama unei funcţii logice cu ***n*** intrări, este un tablou 2^n celule, câte o celulă pentru fiecare combinaţie de intrare posibilă.

Liniile şi coloanele unei diagrame Karnaugh sunt etichetate astfel încât combinaţia de intrare a oricărei celule să poată fi aflată cu uşurinţă din denumirea liniei şi coloanei la intersecţia cărora se află celula respectivă.

În fiecare celulă a diagramei se scrie o valoare logică **0** sau **1** care reprezintă valoarea de adevăr a funcţiei când variabilele de intrare au valorile coordonatelor celulei respective.

În celula unei diagrame mai poate fi scris (cu dimensiuni mici) numărul ***mintermenului*** corespunzător din tabelul de adevăr. Mintermenul reprezintă valoarea zecimală a numărului binar format din biţii variabilelor de intrare (mai simplu, reprezintă numărul de ordine al rândului din tabelul de adevăr cu precizarea că numărătoarea începe de la **0**.

a. Diagrama Karnaugh pentru o funcţie cu două variabile

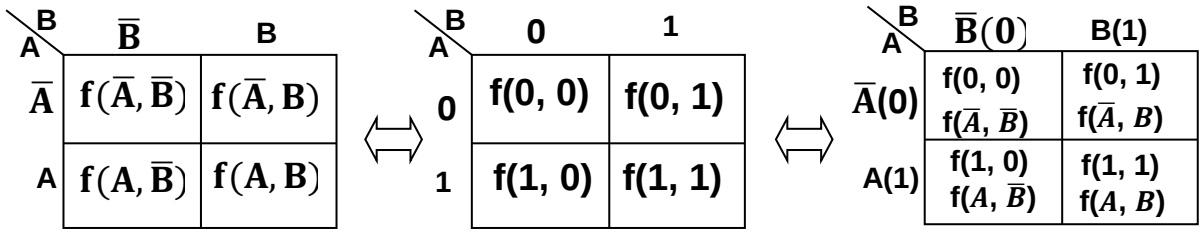


Figura 2.1 Versiunea simplificată a unei diagrame Karnaugh cu 2 variabile

Transformarea tabelului de adevăr a unei funcţii cu două variabile în diagramă Karnaugh este prezentată în **figura 2.2**

Mintermen	A	B	f
0	0	0	a
1	0	1	b
2	1	0	c
3	1	1	d

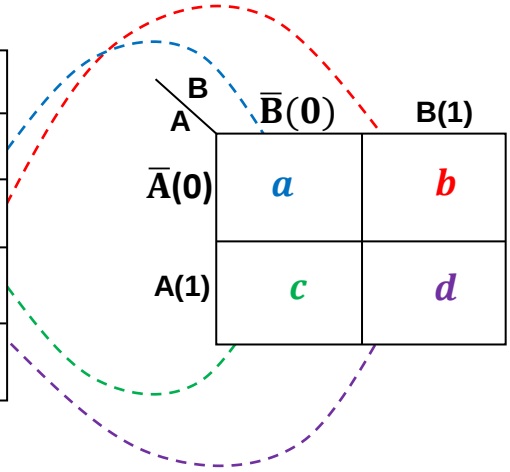


Figura 2.2 Corespondenţa dintre tabela de adevăr şi diagrama Karnaugh

b. Diagrama Karnaugh pentru o funcţie cu trei variabile

A \ BC	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
$\bar{A}$	$f(\bar{A},\bar{B},\bar{C})$	$f(\bar{A},\bar{B},C)$	$f(\bar{A},B,C)$	$f(\bar{A},B,\bar{C})$
A	$f(A,\bar{B},\bar{C})$	$f(A,\bar{B},C)$	$f(A,B,C)$	$f(A,B,\bar{C})$

$\Leftrightarrow$

A \ BC	00	01	11	10
0	$f(0,0,0)$	$f(0,0,1)$	$f(0,1,1)$	$f(0,1,0)$
1	$f(1,0,0)$	$f(1,0,1)$	$f(1,1,1)$	$f(1,1,0)$

Figura 2.3 Versiunea simplificată a unei diagrame Karnaugh cu 3 variabile

Transformarea tabelului de adevăr a unei funcţii cu trei variabile în diagramă Karnaugh este prezentată în **figura 2.4**

Mintermen	A	B	C	f
0	0	0	0	a
1	0	0	1	b
2	0	1	0	c
3	0	1	1	d
4	1	0	0	e
5	1	0	1	f
6	1	1	0	g
7	1	1	1	h

A \ BC	$\bar{B}\bar{C}(00)$	$\bar{B}C(01)$	$BC(11)$	$B\bar{C}(10)$
$\bar{A}(0)$	0 a	1 b	3 d	2 c
$A(1)$	4 e	5 f	7 h	6 g

Figura 2.4 Corespondenţa dintre tabela de adevăr şi diagrama Karnaugh

c. Diagrama Karnaugh pentru o funcţie cu patru variabile

$A \cdot B$ \ $C \cdot D$	$\bar{C} \cdot \bar{D}$	$\bar{C} \cdot D$	$C \cdot D$	$C \cdot \bar{D}$
$\bar{A} \cdot \bar{B}$	$f(\bar{A}, \bar{B}, \bar{C}, \bar{D})$	$f(\bar{A}, \bar{B}, \bar{C}, D)$	$f(\bar{A}, \bar{B}, C, D)$	$f(\bar{A}, \bar{B}, C, \bar{D})$
$\bar{A} \cdot B$	$f(\bar{A}, B, \bar{C}, \bar{D})$	$f(\bar{A}, B, \bar{C}, D)$	$f(\bar{A}, B, C, D)$	$f(\bar{A}, B, C, \bar{D})$
$A \cdot B$	$f(A, B, \bar{C}, \bar{D})$	$f(A, B, \bar{C}, D)$	$f(A, B, C, D)$	$f(A, B, C, \bar{D})$
$A \cdot \bar{B}$	$f(A, \bar{B}, \bar{C}, \bar{D})$	$f(A, \bar{B}, \bar{C}, D)$	$f(A, \bar{B}, C, D)$	$f(A, \bar{B}, C, \bar{D})$

⇕

$A \cdot B$ \ $C \cdot D$	00	01	11	10
00	$f(0,0,0,0)$	$f(0,0,0,1)$	$f(0,0,1,1)$	$f(0,0,1,0)$
01	$f(0,1,0,0)$	$f(0,1,0,1)$	$f(0,1,1,1)$	$f(0,1,1,0)$
11	$f(1,1,0,0)$	$f(1,1,0,1)$	$f(1,1,1,1)$	$f(1,1,1,0)$
10	$f(1,0,0,0)$	$f(1,0,0,1)$	$f(1,0,1,1)$	$f(1,0,1,0)$

Figura 2.5 Versiunea simplificată a unei diagrame Karnaugh cu 4 variabile

Mintermen	A	B	C	D	f
0	0	0	0	0	a
1	0	0	0	1	b
2	0	0	1	0	c
3	0	0	1	1	d
4	0	1	0	0	e
5	0	1	0	1	f
6	0	1	1	0	g
7	0	1	1	1	h
8	1	0	0	0	i
9	1	0	0	1	j
10	1	0	1	0	k
11	1	0	1	1	l
12	1	1	0	0	m
13	1	1	0	1	n
14	1	1	1	0	o
15	1	1	1	1	p

AB \ CD	$\bar{C}\bar{D}$(00)	$\bar{C}D$(01)	CD(11)	$C\bar{D}$(10)
$\bar{A}\bar{B}$(00)	0 a	b	3 d	2 c
$\bar{A}B$(01)	4 e	5 f 7	h	6 g
AB(11)	12 m	13 n	15 p	14 o
$A\bar{B}$(10)	8 i	9 j	11 l	10 k

Figura 2.6 Corespondenţa dintre tabela de adevăr şi diagrama Karnaugh

2.4. SIMPLIFICAREA FUNCŢIILOR LOGICE

În proiectarea sistemelor digitale, implementarea circuitelor digitale se bazează pe algebra booleană. Între gradul de complexitate al funcţiei logice care descrie un circuit şi gradul de complexitate al circuitului respectiv există o strânsă legătura. Dacă reuşim sa simplificăm expresia funcţiei logice vom reduce automat şi complexitatea circuitului.

Implementarea practică a circuitului se realizează pe baza formei minimizate a funcţiei logice care descrie circuitul numeric, ceea ce conduce la o configuraţie optimă de circuit.

2.4.1 TRANSFORMAREA TABELULUI DE ADEVĂR ÎN EXPRESII LOGICE

Procesul de proiectare a circuitelor digitale începe adeseori de la un tabel de adevăr. După cum am văzut în secţiunea **2.2**, tabelul de adevăr stabileşte corespondenţa dintre valorile de adevăr ale variabilelor de intrare şi valoarea de adevăr a funcţiei circuitului respectiv. În funcţie de starea logică a variabilelor de intrare, funcţia logică a circuitului are o anumită formă. Înainte de a fi simplificată, funcţia logică trebuie determinată.

Pe baza tabelului de adevăr o funcţie logică se determină relativ simplu, după următorul algoritm:

- Se identifică în tabelul de adevăr liniile în care valoarea variabilei de ieşire **f** este **1**;
- Se face produsul variabilelor de intrare de pe liniile respective (câte un produs pentru fiecare linie);
- Forma algebrică a funcţiei logice **f** este suma acestor produse.

OBSERVAŢII:

- Dacă pe o linie a tabelului de adevăr, valoarea logică a unei variabile de intrare este **0** în expresie produsului apare forma negată a variabilei respective;
- Dacă pe o linie a tabelului de adevăr, valoarea logică a unei variabile de intrare este **1** în expresie produsului apare forma normală a variabilei respective.

Exemplu: Deducerea expresiei funcţiei logice care are următorul tabel de adevăr:

Dec	Hex	A	B	C	Fn
0	0	0	0	0	0
1	1	0	0	1	0
2	2	0	1	0	0
3	3	0	1	1	1
4	4	1	0	0	0
5	5	1	0	1	1
6	6	1	1	0	1
7	7	1	1	1	1

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

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

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

$A \cdot B \cdot C$

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

În cele se urmează se va prezenta prin câteva exemple determinarea unei funcţii logice pornind de la tabelul de adevăr.

EXEMPLUL 1.

Dec	Hex	A	B	C	Fn
0	0	0	0	0	0
1	1	0	0	1	1
2	2	0	1	0	1
3	3	0	1	1	1
4	4	1	0	0	1
5	5	1	0	1	0
6	6	1	1	0	1
7	7	1	1	1	0

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

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

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

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

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

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

EXEMPLUL 2.

Dec	Hex	A	B	C	D	Fn	
0	0	0	0	0	0	0	
1	1	0	0	0	1	0	
2	2	0	0	1	0	0	
3	3	0	0	1	1	0	
4	4	0	1	0	0	0	
5	5	0	1	0	1	1	$\overline{A} \cdot B \cdot \overline{C} \cdot D$
6	6	0	1	1	0	0	
7	7	0	1	1	1	1	$\overline{A} \cdot B \cdot C \cdot D$
8	8	1	0	0	0	0	
9	9	1	0	0	1	0	
10	A	1	0	1	0	1	$A \cdot \overline{B} \cdot C \cdot \overline{D}$
11	B	1	0	1	1	1	$A \cdot \overline{B} \cdot C \cdot D$
12	C	1	1	0	0	0	
13	D	1	1	0	1	1	$A \cdot B \cdot \overline{C} \cdot D$
14	E	1	1	1	0	1	$A \cdot B \cdot C \cdot \overline{D}$
15	F	1	1	1	1	1	$A \cdot B \cdot C \cdot D$

$$f = \overline{A} \cdot B \cdot \overline{C} \cdot D + \overline{A} \cdot B \cdot C \cdot D + A \cdot \overline{B} \cdot C \cdot \overline{D} + A \cdot \overline{B} \cdot C \cdot D + \\ + A \cdot B \cdot \overline{C} \cdot D + A \cdot B \cdot C \cdot \overline{D} + A \cdot B \cdot C \cdot D$$

EXEMPLUL 3.

Dec	Hex	A	B	C	D	Fn	
0	0	0	0	0	0	1	$\overline{A} \cdot \overline{B} \cdot \overline{C} \cdot \overline{D}$
1	1	0	0	0	1	0	
2	2	0	0	1	0	1	$\overline{A} \cdot \overline{B} \cdot C \cdot \overline{D}$
3	3	0	0	1	1	0	
4	4	0	1	0	0	0	
5	5	0	1	0	1	1	$\overline{A} \cdot B \cdot \overline{C} \cdot D$
6	6	0	1	1	0	0	
7	7	0	1	1	1	1	$\overline{A} \cdot B \cdot C \cdot D$
8	8	1	0	0	0	1	$A \cdot \overline{B} \cdot \overline{C} \cdot \overline{D}$
9	9	1	0	0	1	0	
10	A	1	0	1	0	1	$A \cdot \overline{B} \cdot C \cdot \overline{D}$
11	B	1	0	1	1	0	
12	C	1	1	0	0	0	
13	D	1	1	0	1	1	$A \cdot B \cdot \overline{C} \cdot D$
14	E	1	1	1	0	0	
15	F	1	1	1	1	1	$A \cdot B \cdot C \cdot D$

$$f = \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot \overline{D} + \overline{A} \cdot \overline{B} \cdot C \cdot \overline{D} + \overline{A} \cdot B \cdot \overline{C} \cdot D + \overline{A} \cdot B \cdot C \cdot D + \\ + A \cdot \overline{B} \cdot \overline{C} \cdot \overline{D} + A \cdot \overline{B} \cdot C \cdot \overline{D} + A \cdot B \cdot \overline{C} \cdot D + A \cdot B \cdot C \cdot D$$

EXEMPLUL 4.

Dec	Hex	A	B	C	D	Fn	
0	0	0	0	0	0	0	
1	1	0	0	0	1	0	
2	2	0	0	1	0	0	
3	3	0	0	1	1	0	
4	4	0	1	0	0	0	
5	5	0	1	0	1	1	$\rightarrow \overline{A} \cdot B \cdot \overline{C} \cdot D$
6	6	0	1	1	0	0	
7	7	0	1	1	1	1	$\rightarrow \overline{A} \cdot B \cdot C \cdot D$
8	8	1	0	0	0	0	
9	9	1	0	0	1	0	
10	A	1	0	1	0	1	$\rightarrow A \cdot \overline{B} \cdot C \cdot \overline{D}$
11	B	1	0	1	1	1	$\rightarrow A \cdot \overline{B} \cdot C \cdot D$
12	C	1	1	0	0	0	
13	D	1	1	0	1	1	$\rightarrow A \cdot B \cdot \overline{C} \cdot D$
14	E	1	1	1	0	1	$\rightarrow A \cdot B \cdot C \cdot \overline{D}$
15	F	1	1	1	1	1	$\rightarrow A \cdot B \cdot C \cdot D$

$$f = \overline{A} \cdot B \cdot \overline{C} \cdot D + \overline{A} \cdot B \cdot C \cdot D + A \cdot \overline{B} \cdot C \cdot \overline{D} + A \cdot \overline{B} \cdot C \cdot D + \\ + A \cdot B \cdot \overline{C} \cdot D + A \cdot B \cdot C \cdot \overline{D} + A \cdot B \cdot C \cdot D$$

2.4.2 MINIMIZAREA FUNCŢIILOR LOGICE

Minimizarea unei funcţii logice se poate realiza prin:

- ***metoda analitică*** – care se bazează pe simplificarea expresiei unei funcţii logice pe baza axiomelor şi teoremelor algebrei booleene;
- ***metoda diagramelor Veitch – Karnaugh*** – care transpune axiomele şi teoreme algebrei booleene pe reprezentare funcţiei cu diagrame Karnaugh.

În cele se urmează se va explica prin câteva exemple simplificarea funcţiilor logice prin ambele metode.

EXEMPLUL 1. Minimizarea funcţiei $f = \overline{A} \cdot B \cdot C + A \cdot \overline{B} \cdot C + A \cdot B \cdot \overline{C} + A \cdot B \cdot C$

1.1 Metoda analitică

$f = \overline{A} \cdot B \cdot C + A \cdot \overline{B} \cdot C + A \cdot B \cdot \overline{C} + A \cdot B \cdot C = B \cdot C \cdot (\overline{A} + A) + A \cdot \overline{B} \cdot C + A \cdot B \cdot \overline{C} =$

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

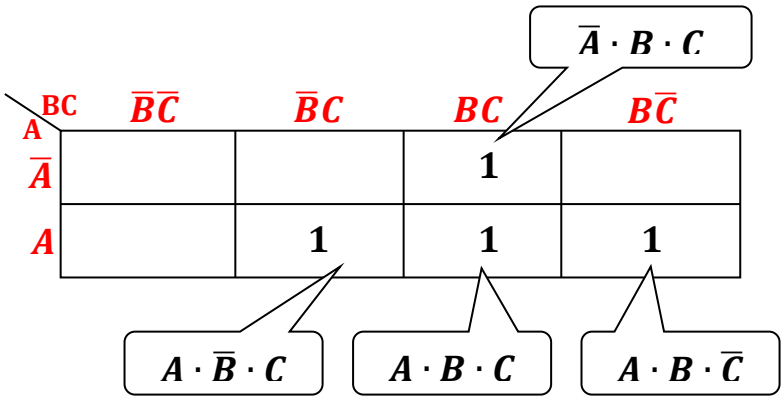
$= C \cdot B + C \cdot A + A \cdot B \cdot \overline{C} = C \cdot B + A \cdot (C + \overline{C} \cdot B) = C \cdot B + A \cdot (C + B) = B \cdot C + A \cdot C + A \cdot B$

Prin metoda analitica se obţine în urma minimizării funcţia: **f = A•B + A•C + B•C**

1.2 Metoda diagramei Karnaugh

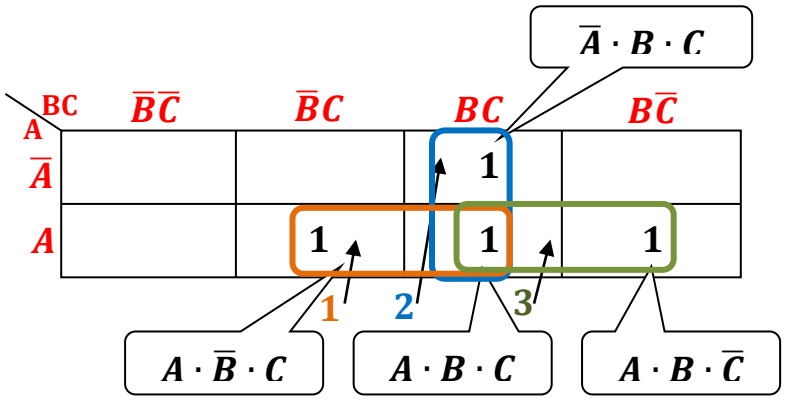
Se parcurg următoarele etape:

- Se scrie expresia funcţiei $f = \overline{A} \cdot B \cdot C + A \cdot \overline{B} \cdot C + A \cdot B \cdot \overline{C} + A \cdot B \cdot C$;
- Se desenează diagrama Karnaugh;
- În celulele diagramei se introduc valorile de **1** corespunzător poziţiei fiecărui produs al sumei funcţiei **f** (coordonatele celulelor pentru funcţia cu 3 variabile sunt prezentate în **figura 2.3** din secţiunea **2.3.2**);



- Identificăm grupuri de celule alăturate care conţin valoarea 1.

- **OBSERVAŢII:**
  - Fiecare grup trebuie să conţină **două sau patru celule adiacente**;
  - Celule adiacente au o latură comună pe verticală sau pe orizontală şi diferă printr-o singură variabilă;
  - Se consideră adiacente şi celulele da la capetele opuse ale unei linii sau coloane;
  - celulă poate face parte din mai multe grupuri;
  - În diagrama de mai jos au fost identificate **3 grupuri** de câte două celule;



- Se caută variabila sau variabilele comune pentru fiecare grup şi scriem pentru fiecare grup în parte, variabila (sau produsul de variabile dacă sunt mai multe) ca rezultat boolean. Rezultatul final este suma rezultatelor fiecărui grup;
- În diagrama de mai sus:
  - pentru grupul 1 sunt comune variabilele **A** şi **C** - **rezultat logic A·C;**
  - pentru grupul 2 sunt comune variabilele **B** şi **C – rezultat logic B·C;**
  - pentru grupul 3 sunt comune variabilele **A** şi **B – rezultat logic A·B;**
- Rezultatul final este : **f = A•B + A•C + B•C.**

EXEMPLUL 2. Minimizarea funcţiei :

$$f = A \cdot \bar{B} \cdot \bar{C} + A \cdot B \cdot \bar{C} + \bar{A} \cdot \bar{B} \cdot C + \bar{A} \cdot B \cdot C + \bar{A} \cdot B \cdot \bar{C}$$

2.1 Metoda analitică

$$f = A \cdot \bar{B} \cdot \bar{C} + A \cdot B \cdot \bar{C} + \bar{A} \cdot \bar{B} \cdot C + \bar{A} \cdot B \cdot C + \bar{A} \cdot B \cdot \bar{C} = A \cdot \bar{C} \cdot (\bar{B} + B) +$$

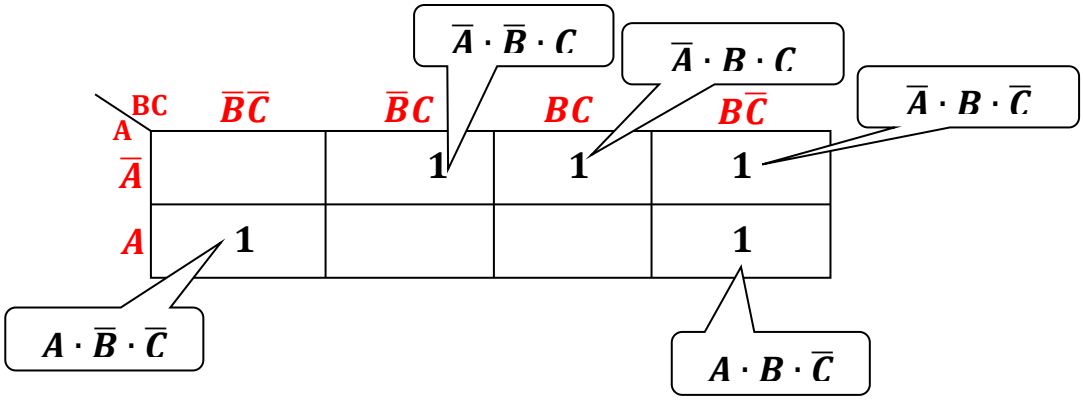
$$+ \bar{A} \cdot C \cdot (\bar{B} + B) + \bar{A} \cdot B \cdot \bar{C} = A \cdot \bar{C} + \bar{A} \cdot C + \bar{A} \cdot B \cdot \bar{C} = A \cdot \bar{C} + \bar{A} \cdot (C + B \cdot \bar{C}) =$$

$$= A \cdot \bar{C} + \bar{A} \cdot C + \bar{A} \cdot B \quad \Rightarrow \quad \mathbf{f = A \cdot \bar{C} + \bar{A} \cdot C + \bar{A} \cdot B}$$

2.2 Metoda diagramei Karnaugh

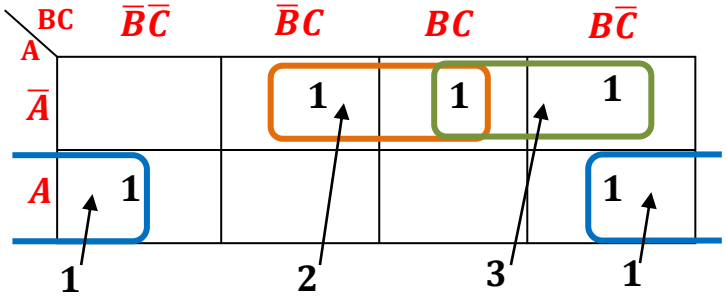
Se parcurg etapele prezentate la punctul 1.2

- În celulele diagramei se introduc valorile de **1** corespunzătoare poziţiei fiecărui produs al sumei funcţiei **f**;



- Identificăm grupuri de celule alăturate care conţin valoarea 1;

  În diagrama de mai jos au fost identificate **3 grupuri** de câte două celule



- În diagrama de mai sus:
  - pentru grupul **1** sunt comune variabilele ***A*** şi $\overline{C}$ **- rezultat logic** $A\cdot\overline{C}$;
  - pentru grupul **2** sunt comune variabilele $\overline{A}$ şi ***C*** **– rezultat logic** $\overline{A}\cdot C$;
  - pentru grupul **3** sunt comune variabilele $\overline{A}$ şi ***B*** **– rezultat logic** $\overline{A}\cdot B$;
- Rezultatul final este : $f = A\cdot\overline{C} + \overline{A}\cdot C + \overline{A}\cdot B$.

În exemplele următoare minimizarea unei funcţii logice se va prezenta numai prin metoda diagramei Karnaugh .

EXEMPLUL 3. Minimizarea funcţiei :

$$f = \overline{A}\cdot B\cdot\overline{C}\cdot D + \overline{A}\cdot B\cdot C\cdot D + A\cdot\overline{B}\cdot C\cdot\overline{D} + A\cdot\overline{B}\cdot C\cdot D + A\cdot B\cdot\overline{C}\cdot D +$$
$$+ A\cdot B\cdot C\cdot\overline{D} + A\cdot B\cdot C\cdot D$$

Se observă că funcţia are **patru** variabile de intrare ⇒ diagrama Karnaugh are 16 celule.

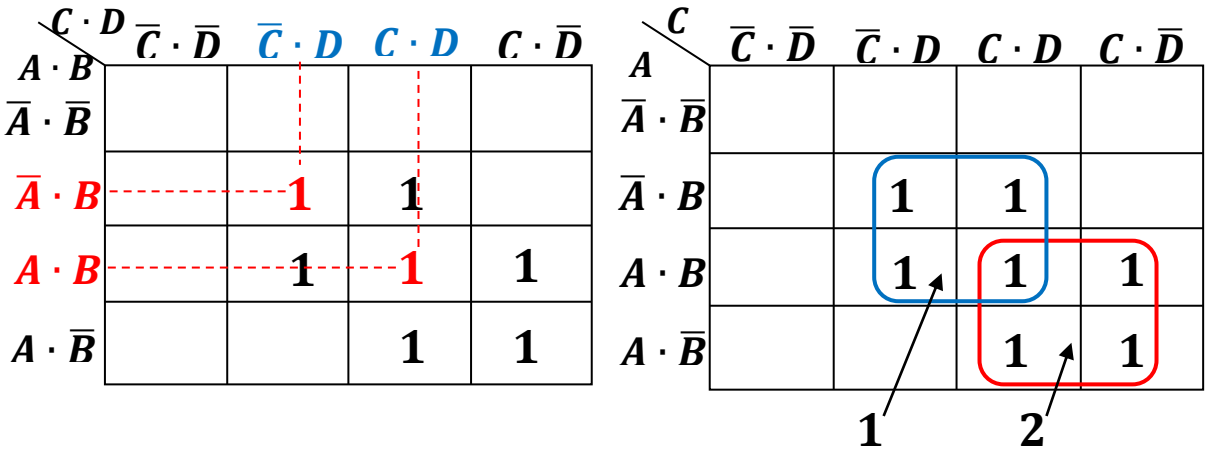
- În celulele diagramei se introduc valorile de **1** corespunzătoare poziţiei fiecărui produs al sumei funcţiei **f**.

Deoarece funcţia are **7 termeni**, pe diagramă în **7 celule** va fi valoarea logică **1.**

Fiecare termen al funcţiei se plasează la adresa corespunzătoare din celulă (vezi **figura 2.4** din secţiunea **2.3.2**). Primele două caractere ale unui termen indică linia iar ultimele două caractere ale termenului indică coloana la intersecţia cărora se plasează caracterul **1** în tabel .**Exemple:**

- termenul $\overline{A} \cdot B \cdot \overline{C} \cdot D$ se plasează la intersecţia **liniei** $\overline{A} \cdot B$ cu **coloana** $\overline{C} \cdot D$

- termenul $A \cdot B \cdot C \cdot D$ se plasează la intersecţia **liniei** $A \cdot B$ cu **coloana** $C \cdot D$



- Identificăm grupuri de celule alăturate care conţin valoarea 1

În general, pe o diagramă Karnaugh se încearcă formarea grupurile cu dimensiunea pătratelor cât mai mare (cu cât dimensiunea pătratului este mai mare cu atât se elimină mai multe caractere din rezultatul final)

În diagrama de mai sus s-au format **2** grupuri cu **pătrate** care au **4 celule** (latura = 2).

- În diagrama de mai sus:
  - pentru grupul **1** sunt comune variabilele ***B*** şi ***D*** **- rezultat logic *B·D***
  - pentru grupul **2** sunt comune variabilele ***A*** şi ***C*** **– rezultat logic *A·C***
- Rezultatul final este : $f = A \cdot C + B \cdot D$

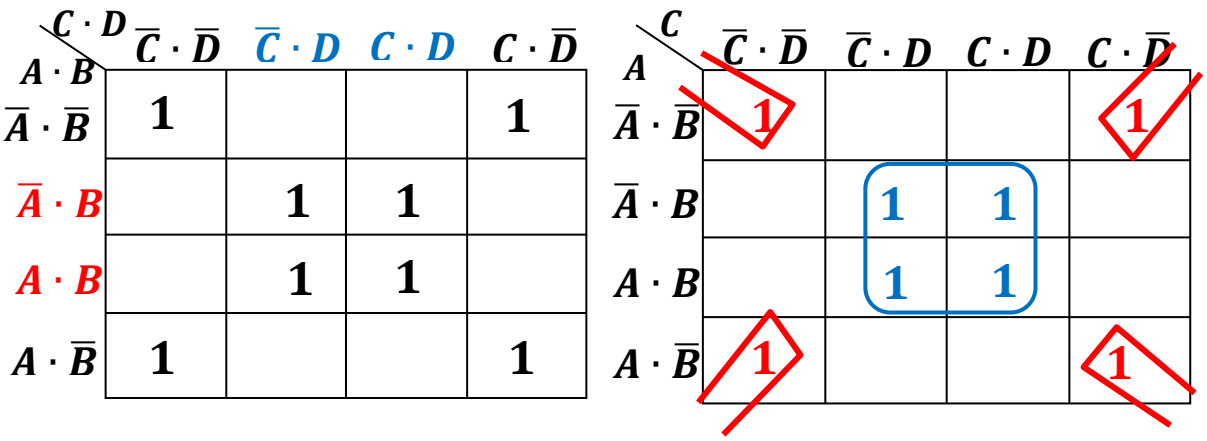
EXEMPLUL 4. Minimizarea funcţiei :

$$f = \overline{A}\cdot\overline{B}\cdot\overline{C}\cdot\overline{D} + A\cdot\overline{B}\cdot\overline{C}\cdot\overline{D} + \overline{A}\cdot\overline{B}\cdot C\cdot\overline{D} + A\cdot\overline{B}\cdot C\cdot\overline{D} + \overline{A}\cdot B\cdot\overline{C}\cdot D + \overline{A}\cdot B\cdot C\cdot D + A\cdot B\cdot\overline{C}\cdot D + A\cdot B\cdot C\cdot D$$

- În celulele diagramei se introduc valorile de **1** corespunzătoare poziţiei fiecărui produs al sumei funcţiei **f**;

Deoarece funcţia are **8 termeni**, pe diagramă în **8 celule** va fi valoarea logică **1.**

Fiecare termen al funcţiei se plasează la adresa corespunzătoare din celulă (vezi **figura 2.6** din secţiunea **2.3.2**).



- Identificăm grupuri de celule alăturate care conţin valoarea **1**;

Celulele din cele patru colţuri ale diagramei dacă au valoarea **1** formează un **pătrat**.

- În diagrama de mai sus s-au format **două** grupuri de **două pătrate** cu câte **4 celule**:
  - pentru pătratul format de celulele din mijlocul diagramei sunt comune variabilele ***B*** (pe cele 2 linii) şi ***D*** (pe cele 2 coloane) **- rezultat logic *B·D***
  - pentru pătratul format de celulele din colţurile diagramei sunt comune variabilele $\overline{B}$ (pe cele 2 linii) şi $\overline{D}$ (pe cele 2 coloane) **– rezultat logic $\overline{B}\cdot\overline{D}$**
- Rezultatul final este : ***f = B · D + $\overline{B}$· $\overline{D}$***.

REZUMATUL CAPITOLULUI

- **TEOREMELE ALGEBREI LOGICE:**
  - **A + 0 = 0 + A = A** **A • 1 = 1 • A = A;**
  - **A +** $\overline{A}$ **= 1** **A •** $\overline{A}$ **= 0;**
  - **A + A +....+ A = A** **A • A • ....• A = A;**
  - **A + 1 = 1** **A • 0 = 0;**
  - **A + A • B = A** **A • (A + B) = A;**
  - $\overline{A} + \overline{A} \bullet B = \overline{A}$ $\overline{A}$ **• (**$\overline{A}$**+ B)=** $\overline{A}$**;**
  - **A +** $\overline{A}$ **• B = A + B** **A • (**$\overline{A}$ **+ B) = A • B;**
  - $\overline{\overline{A}}$ **= A**
  - $\overline{A + B} = \overline{A} \bullet \overline{B}$ $\overline{A \bullet B} = \overline{A} + \overline{B}$**;**
  - $A + B = \overline{\overline{A} \bullet \overline{B}}$ $A \bullet B = \overline{\overline{A} + \overline{B}}$**;**
- **PRINCIPALELE FUNCȚII LOGICE:**
  - **NU (NOT)** **Y =** $\overline{A}$**;**
  - **SAU (OR)** **Y = A + B;**
  - **ȘI (AND)** **Y = A • B;**
  - **SAU-NU (NOR)** **Y =** $\overline{A + B}$**;**
  - **ȘI-NU (NAND)** **Y =** $\overline{A \bullet B}$**;**
  - **SAU-EXCLUSIV (XOR)** **Y =** $A \oplus B = \overline{A} \cdot B + A \cdot \overline{B}$**;**
  - **SAU-EXCLUSIV- NEGAT (NXOR)** **Y =** $\overline{A \oplus B} = A \cdot B + \overline{A} \cdot \overline{B}$**;**
- **SIMPLIFICAREA funcțiilor logice pe baza tabelului de adevăr se face după următorul algoritm:**
  - Se identifică în tabelul de adevăr liniile în care valoarea variabilei de ieşire **f** este **1**;
  - Se face produsul variabilelor de intrare de pe liniile respective (câte un produs pentru fiecare linie);
  - Forma algebrică a funcţiei logice **f** este suma acestor produse.
- **OBSERVAŢII:**
  - Dacă pe o linie a tabelului de adevăr, valoarea logică a unei variabile de intrare este **0** în expresie produsului apare forma negată a variabilei respective;
  - Dacă pe o linie a tabelului de adevăr, valoarea logică a unei variabile de intrare este **1** în expresie produsului apare forma normală a variabilei respective.
- Minimizarea unei funcţii logice se poate realiza prin:
  - ***metoda analitică*** – care se bazează pe simplificarea expresiei unei funcţii logice pe baza axiomelor şi teoremelor algebrei booleene;
  - ***metoda diagramelor Veitch – Karnaugh*** – care transpune axiomele şi teoreme algebrei booleene pe reprezentare funcţiei cu diagrame Karnaugh.

EVALUAREA CUNOȘTINȚELOR

1. Deduceți expresia funcției logice căreia îi corespunde tabelului de adevăr:

a.

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

b.

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

c.

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

2. Simplificați următoarele funcții logice utilizând teoremele algebrei logice:

a. $f = A \cdot A \cdot B + C \cdot \overline{C} \cdot D$;

b. $f = B \cdot B \cdot C + B \cdot \overline{C} \cdot \overline{C} + \overline{A} \cdot B \cdot C + \overline{A} \cdot B \cdot \overline{C}$;

c. $f = \overline{A} \cdot B \cdot C + \overline{A} \cdot B \cdot \overline{C} + A \cdot C$;

d. $f = (A + B) \cdot (\overline{A} + C)$;

e. $f = A \cdot B + A \cdot (B + C) + B \cdot (B + C)$;

f. $f = (A + \overline{A}) \cdot (A \cdot B + A \cdot B \cdot \overline{C})$;

g. $f = \overline{(\overline{A} + \overline{B})} + \overline{(A + \overline{B})}$;

h. $f = A \cdot B \cdot \overline{(\overline{A} + B \cdot C)}$;

i. $f = (A + \overline{B} + C) \cdot \overline{(A \cdot B + \overline{A} \cdot \overline{C})}$;

j. $f = A \cdot \overline{B} + \overline{A} \cdot B + (A + \overline{B}) \cdot A$;

3. Simplificați următoarele funcții logice utilizând metoda diagramei Karnaugh:

a. $f = \overline{A} \cdot \overline{B} \cdot C + A \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot \overline{C} + A \cdot B \cdot C$;

b. $f = \overline{A} \cdot B \cdot C + \overline{A} \cdot B \cdot \overline{C} + A \cdot B \cdot \overline{C} + \overline{A} \cdot \overline{B} \cdot \overline{C}$;

c. $f = \overline{A} \cdot \overline{B} \cdot \overline{C} \cdot \overline{D} + \overline{A} \cdot B \cdot \overline{C} \cdot \overline{D} + A \cdot \overline{B} \cdot \overline{C} \cdot \overline{D} + A \cdot \overline{B} \cdot \overline{C} \cdot D + \overline{A} \cdot \overline{B} \cdot C \cdot \overline{D} + \overline{A} \cdot B \cdot C \cdot \overline{D} + A \cdot \overline{B} \cdot C \cdot \overline{D}$.

CAPITOLUL 3. PORŢI LOGICE

3.1. PORŢI LOGICE ELEMENTARE

Porţile logice sunt dispozitive electronice numerice cu ajutorul cărora sunt implementate funcţiile logice şi matematice. O poartă logică este un amplificator special care acceptă şi generează semnale de tensiune corespunzătoare stărilor logice **0** şi **1**.

Poarta logică are una sau mai multe intrări digitale care formează o combinaţie de valori binare (**0** şi **1**), iar la ieşire o singură stare binară (**0** sau **1**). Datorită acestei proprietăţi o poartă logică este **un circuit combinaţional.**

Fizic, ca şi circuit electric, o poartă logică se reprezintă cu contacte electrice (pentru intrări) şi lampă electrică sau LED pentru ieşire. Pentru toate porţile logice reprezentate electric, cu contacte şi lămpi electrice, se respectă convenţiile:

- **0 logic** este echivalent cu **nivel de tensiune scăzut (L)** sau **0 V (volţi)**;
- **1 logic** este echivalent cu **nivel de tensiune ridicat (H)** sau **+ V(volţi)**;
- contact electric deschis – reprezintă **0 logic la INTRARE**;
- contact electric închis – reprezintă **1 logic la INTRARE**;
- LED stins – reprezintă **0 logic la IEŞIRE**;
- LED aprins – reprezintă **1 logic la IEŞIRE**.

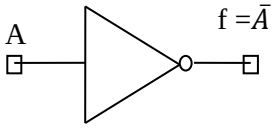
3.1.1 POARTA LOGICĂ NU (NOT)

Tabela de adevăr

A	f
0	1
1	0

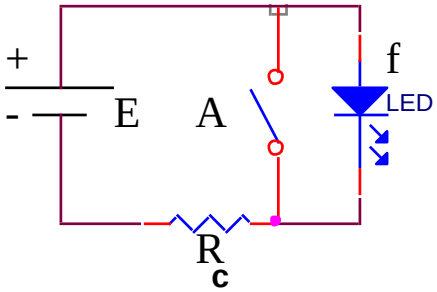
a

Simbolul circuitului logic NU



b

Circuitul logic NU cu contacte



c

Figura 3.1.1 Poarta logică NU

După cum se vede din tabela de adevăr din **figura 3.1.1 a**, poarta logică **NU**, inversează semnalul de intrare. Dacă la intrare este **0 logic** la ieşire este **1 logic** şi invers.

În circuitul din **figura 3.1.1 c**, contactul **A** reprezintă **intrarea** porţii iar **LED-ul f** reprezintă ieşirea porţii.

Când contactul **A** este **deschis (0 logic)**, **LED-ul f** este **aprins (1 logic).**

Când contactul **A** este **închis (1 logic)**, **LED-ul f** este **stins (0 logic).**

Poarta logică **NU** este o poartă elementară cu o singură intrare.

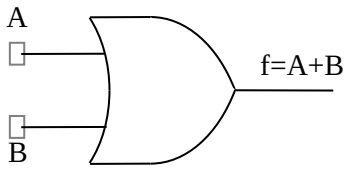
3.1.2 POARTA LOGICĂ SAU (OR)

Tabela de adevăr

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

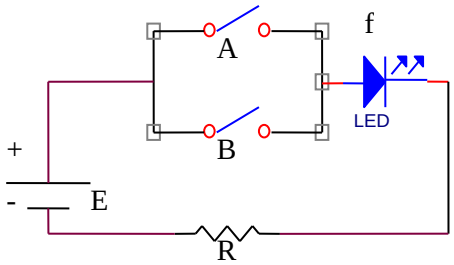
a

Simbolul circuitului logic SAU



b

Circuitul logic SAU cu contacte



c

Figura 3.1.2 Poarta logică SAU

Poarta logică **SAU** implementează funcţia logică **SAU** care este o adunare logică(**disjuncţie**) sau reuniune. ATENŢIE! adunarea logică nu este o adunare aritmetică.

Ieşirea porţii este în **1 logic** dacă cel puţin una din intrările porţii este în **1 logic.**

În schema din **figura 3.1.2 c** LED-ul se aprinde (**1logic**) când cel puţin unul din cele două contacte **A** şi **B** sunt închise (**1 logic).**

Porţile logice SAU pot fi cu două sau cu mai multe intrări.

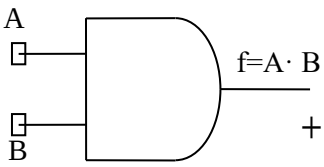
3.1.3 POARTA LOGICĂ ŞI (AND)

Tabela de adevăr

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

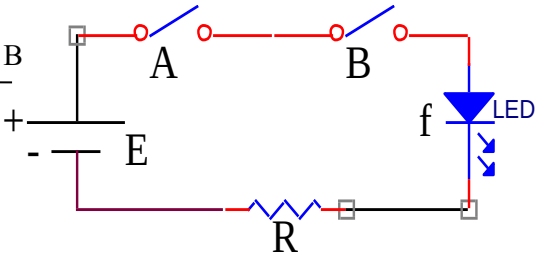
a

Simbolul circuitului logic ŞI



b

Circuit logic ŞI cu contacte



c

Figura 3.1.3 Poarta logică SAU

Poarta logică **ŞI** implementează funcţia logică **ŞI** care este o înmulţire logică(**conjuncţie**) sau intersecţie. ATENŢIE! înmulţirea logică nu este o înmulţire aritmetică.

Ieşirea porţii este în **1 logic** dacă toate intrările porţii sunt în **1 logic.**

În schema din **figura 3.1.3 c** LED-ul se aprinde (**1logic**) când ambele contacte **A** şi **B** sunt închise (**1 logic).**

Porţile logice **ŞI** pot fi cu două sau cu mai multe intrări.

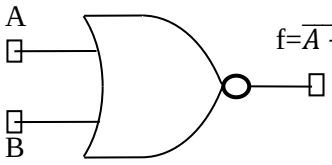
3.1.4 POARTA LOGICĂ SAU – NU (NOR)

Tabela de adevăr

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

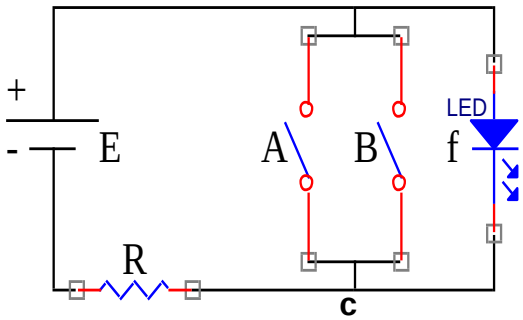
a

Simbolul circuitului SAU-NU



b

Circuitul logic SAU - NU cu contacte



c

Figura 3.1.4 Poarta logică SAU - NU

Poarta logică SAU-NU se obţine prin combinarea unei porţi logice SAU cu o poartă logică NU (vezi **figura 3.1.5**).

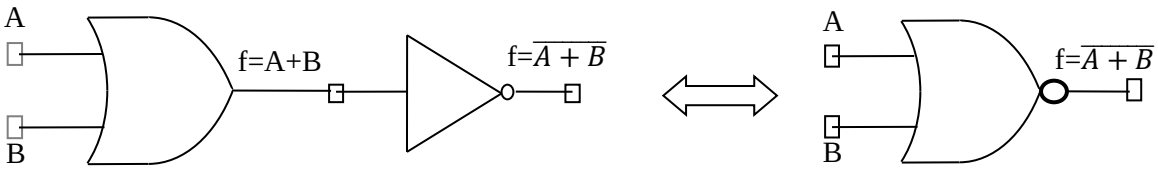


Figura 3.1.5 Obţinerea unei porţi logice SAU - NU

În simbolul porţii logice SAU-NU din **figura 3.1.4 b** negaţia este reprezentată prin cerculeţul de la ieşirea porţii. Prin acest element, simbolul porţii SAU-NU diferă de cel al porţii SAU.

Poarta logică **SAU-NU** implementează funcţia logică **SAU-NU** care este o adunare logică(**disjuncţie**) NEGATĂ.

Ieşirea porţii este în **1 logic** dacă toate intrările porţii sunt în **0 logic.**

În schema din **figura 3.1.4 c** LED-ul se aprinde (**1logic**) când ambele contacte **A** şi **B** sunt deschise (**0 logic).**

Porţile logice SAU-NU pot fi cu două sau cu mai multe intrări.

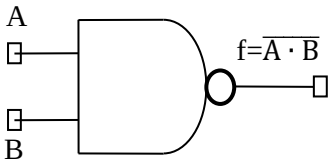
3.1.5 POARTA LOGICĂ ŞI – NU (NAND)

Tabela de adevăr

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

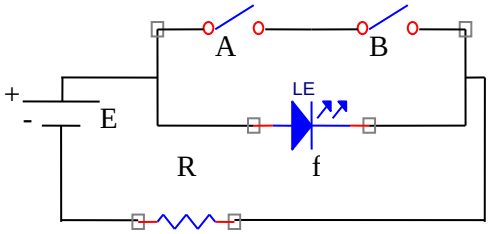
a

Simbolul circuitului ŞI-NU



b

Circuitul logic ŞI – NU cu contacte



c

Figura 3.1.6 Poarta logică ŞI - NU

Poarta logică ŞI-NU se obţine prin combinarea unei porţi logice ŞI cu o poartă logică NU (vezi **figura 3.1.7**).

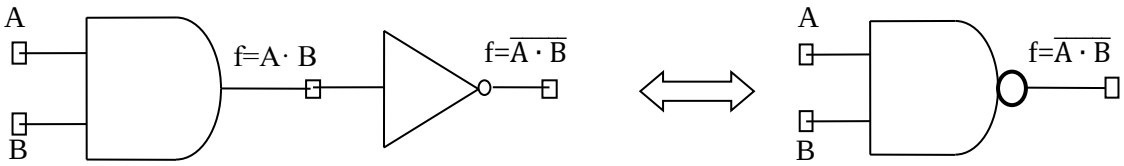


Figura 3.1.7 Obţinerea unei porţi logice ŞI - NU

În simbolul porţii logice ŞI-NU din **figura 3.1.6 b** negaţia este reprezentată prin cerculeţul de la ieşirea porţii. Prin acest element, simbolul porţii ŞI-NU diferă de cel al porţii ŞI.

Poarta logică **ŞI-NU** implementează funcţia logică **ŞI-NU** care este o înmulţire logică(**conjuncţie**) NEGATĂ.

Ieşirea porţii este în **0 logic** dacă toate intrările porţii sunt în **1 logic.**

În schema din **figura 3.1.6 c** LED-ul este stins (**0 logic**) când ambele contacte **A** şi **B** sunt închise (**1 logic)**

Porţile logice ŞI-NU pot fi cu două sau cu mai multe intrări.

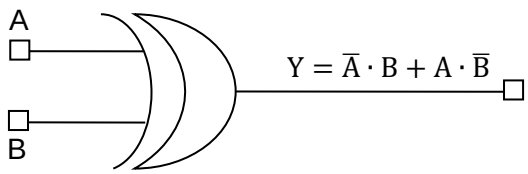
3.1.6 POARTA LOGICĂ SAU – EXCLUSIV (XOR)

Tabela de adevăr

A	B	Y = $A \oplus B$ $Y = \overline{A} \cdot B + A \cdot \overline{B}$
0	0	0
0	1	1
1	0	1
1	1	0

a

Simbolul circuitului SAU-EXCLUSIV



b

Figura 3.1.8 Poarta logică SAU - EXCLUSIV

Poarta logică **SAU-EXCLUSIV** implementează funcţia logică **SAU-EXCLUSIV.**

Ieşirea porţii este în **1 logic** dacă cele două intrări ale porţii sunt complementare (dacă **A** este în **0 logic** atunci **B** trebuie sa fie în **1 logic**, iar dacă **A** este în **1 logic** atunci **B** trebuie să fie în **0 logic**).

În **figura 3.1.9** poarta logică **SAU-EXCLUSIV** este prezentată cu contacte.

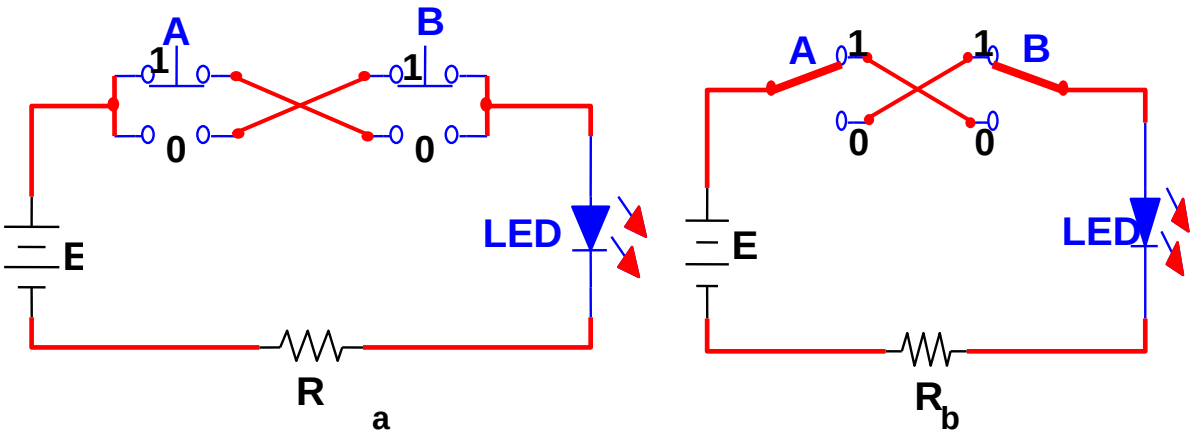


Figura 3.1.9 Poarta logică SAU – EXCLUSIV cu contacte

Intrările **A** şi **B** sunt două butoane cu revenire, care au câte un contact normal închis (**1 logic**) şi un contact normal deschis (**0 logic**), conectate ca în **figura 3.1.9 a**.

Butoanele cu revenire pot fi înlocuite cu comutatoare ca în **figura 3.1.9 b.**

Ieşirea porţii este reprezentată de **LED.**

LED-ul se aprinde numai în situaţia în care un buton este activat (apăsat) iar celălalt buton este dezactivat, sau un comutator este pe poziţia **1** iar celălalt comutator pe poziţia **0**.

3.2. IMPLEMENTAREA FUNCŢIILOR LOGICE CU PORŢI LOGICE

3.2.1 ANALIZA CIRCUITELOR LOGICE

La ***analiza*** circuitelor logice se pleacă de la schema circuitului logic (care se cunoaşte) şi se urmăreşte stabilirea expresiei funcţiei sau funcţiilor logice (care se determină) corespunzătoare circuitului.

Prin analiza circuitelor logice se poate determina şi tabela de adevăr corespunzătoare circuitului logic respectiv.

Determinarea expresiei funcţiei logice de ieşire se face parcurgând schema de la stânga spre dreapta prin scrierea expresiilor de la ieşirea porţilor logice care formează circuitul respectiv din aproape în aproape.

Pentru a înţelege mai bine în cele ce urmează sunt prezentate câteva exemple.

EXEMPLUL 1. Pornind de la schema logică din **figura 3.2.1** să se determine expresia analitică a funcţiei de ieşire a circuitului.

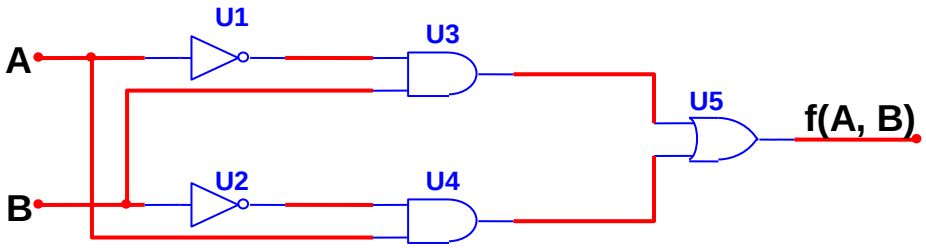


Figura 3.2.1 Schemă logică cu porţi elementare

REZOLVARE

În **figura 3.2.2** sunt prezentate expresiile logice de la ieşirea fiecărei porţi logice din circuit

1. Expresia logică la ieşirea porţii **U1 (NOT)** este $\overline{\mathbf{A}}$.
2. Expresia logică la ieşirea porţii **U2 (NOT)** este $\overline{\mathbf{B}}$.
3. Expresia logică la ieşirea porţii **U3 (AND)** este $\overline{\mathbf{A}} \cdot \mathbf{B}$.
4. Expresia logică la ieşirea porţii **U4 (AND)** este $\overline{\mathbf{B}} \cdot \mathbf{A}$.
5. Expresia logică la ieşirea porţii **U5 (OR)** este $\overline{\mathbf{A}} \cdot \mathbf{B} + \mathbf{A} \cdot \overline{\mathbf{B}}$.
6. Expresia analitică a funcţiei de ieşire este $\mathbf{f} = \overline{\mathbf{A}} \cdot \mathbf{B} + \mathbf{A} \cdot \overline{\mathbf{B}}$.

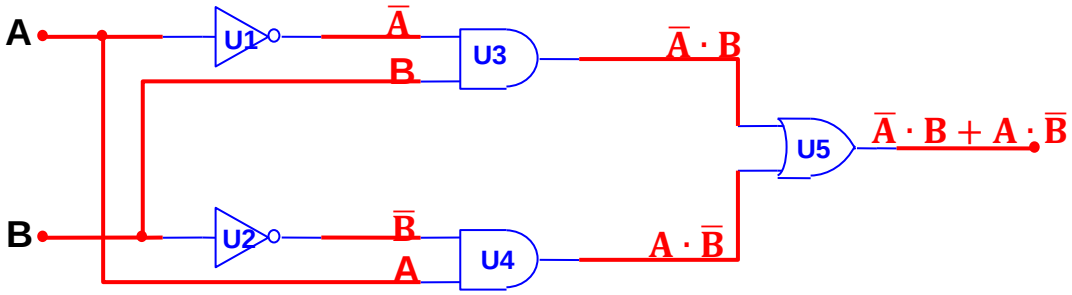


Figura 3.2.2 Schemă logică cu porţi elementare şi expresiile funcţiilor de ieşire

EXEMPLUL 2. Pornind de la schema logică din **figura 3.2.3** să se determine expresia analitică a funcţiei de ieşire a circuitului.

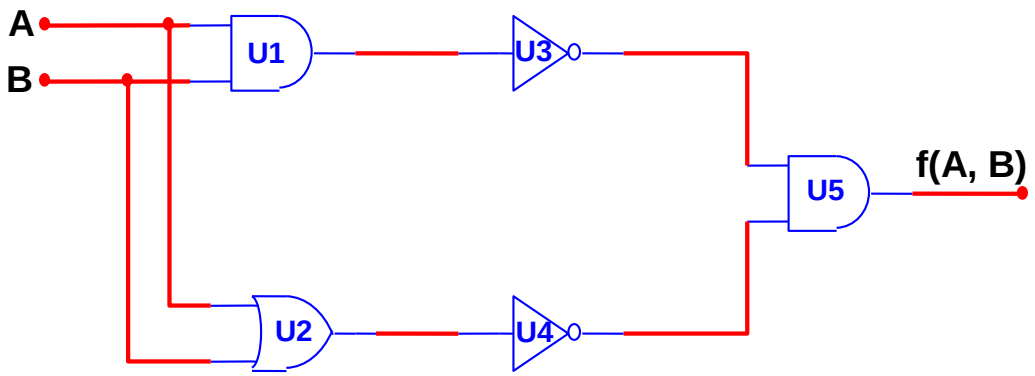


Figura 3.2.3 Schemă logică cu porţi elementare

REZOLVARE

În **figura 3.2.4** sunt prezentate expresiile logice de la ieşirea fiecărei porţi logice din circuit

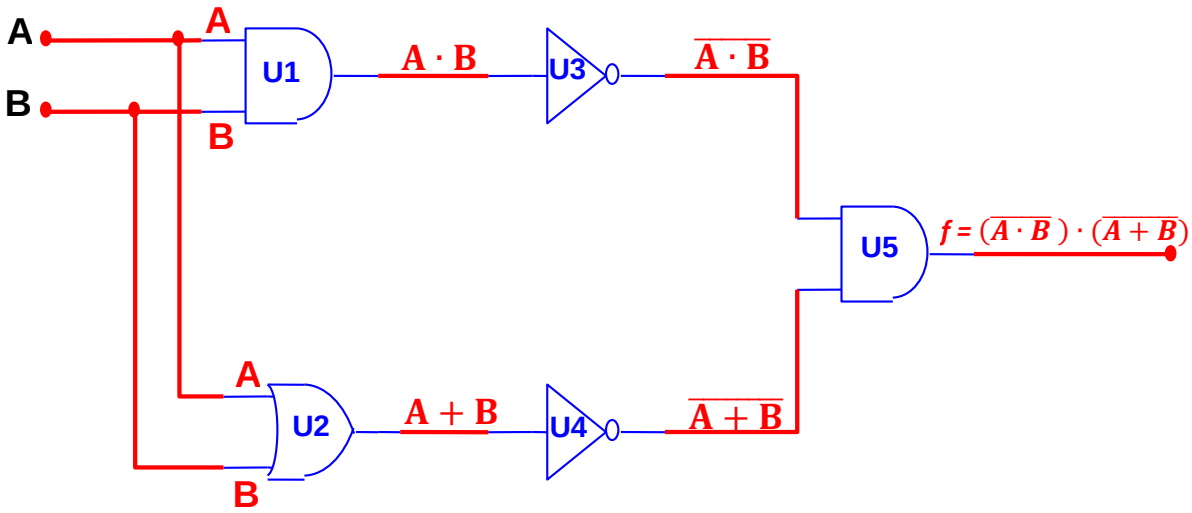


Figura 3.2.4 Schemă logică cu porţi elementare şi expresiile funcţiilor de ieşire

Expresia analitică a funcţiei logice de ieşire este $f = (\overline{A \cdot B}) \cdot (\overline{A + B})$

Funcţia se aduce la o formă mai simplă aplicând axiomele şi teoremele algebrei logice:

$f = (\overline{A \cdot B}) \cdot (\overline{A + B}) = (\overline{A} + \overline{B}) \cdot \overline{A} \cdot \overline{B} = \overline{A} \cdot \overline{A} \cdot \overline{B} + \overline{B} \cdot \overline{A} \cdot \overline{A} = \overline{A} \cdot \overline{B} + \overline{A} \cdot \overline{B} = \overline{A} \cdot \overline{B}$

Forma finală a expresiei analitice a funcţiei de ieşire este $f = \overline{A} \cdot \overline{B} = \overline{A + B}$

EXEMPLUL 3. Pornind de la schema logică din **figura 3.2.5** să se determine expresia analitică a funcţiei de ieşire a circuitului.

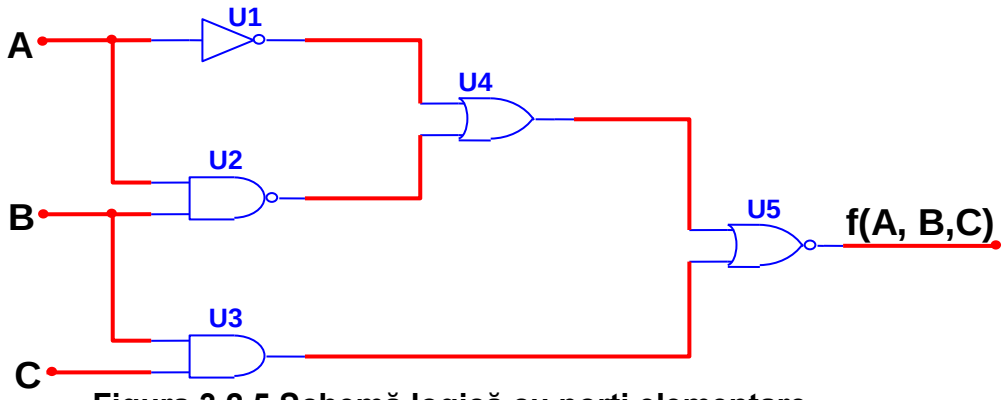


Figura 3.2.5 Schemă logică cu porţi elementare

REZOLVARE

În **figura 3.2.6** sunt prezentate expresiile logice de la ieşirea fiecărei porţi logice din circuit

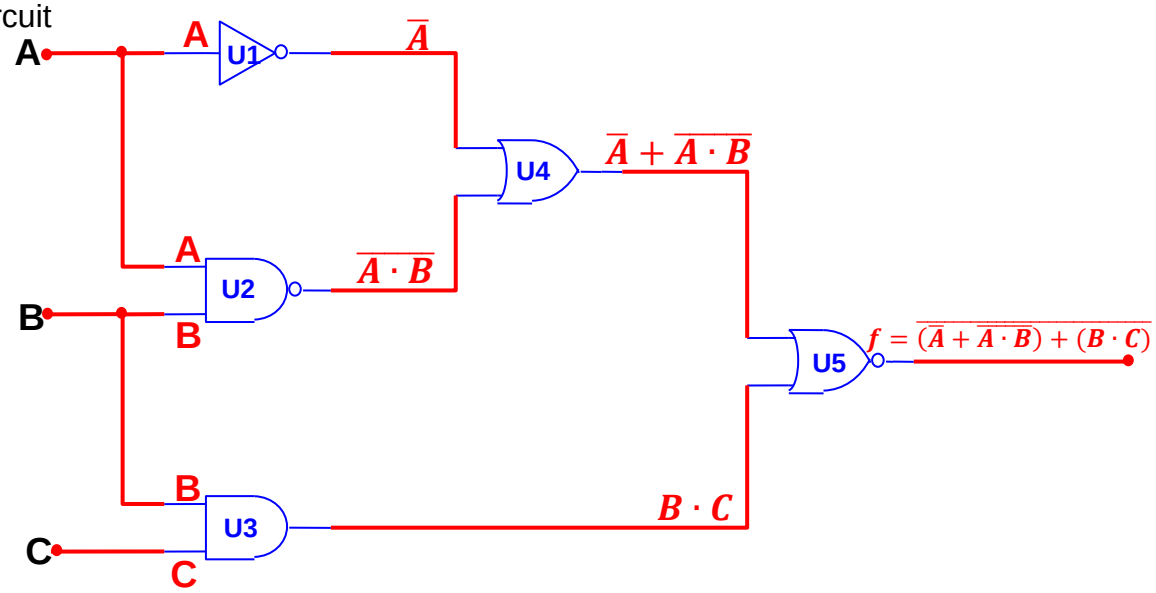


Figura 3.2.6 Schemă logică cu porţi elementare şi expresiile funcţiilor de ieşire

Expresia analitică a funcţiei logice de ieşire este $f = \overline{(\overline{A} + \overline{A \cdot B}) + (B \cdot C)}$

Funcţia se aduce la o formă mai simplă aplicând axiomele şi teoremele algebrei logice:

$$f = \overline{(\overline{A} + \overline{A \cdot B}) + (B \cdot C)} = \overline{(\overline{A} + \overline{A \cdot B})} \cdot \overline{(B \cdot C)} = \overline{\overline{A}} \cdot \overline{\overline{A \cdot B}} \cdot \overline{B \cdot C}$$

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

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

Forma finală a expresiei analitice a funcţiei de ieşire este $f = A \cdot B \cdot \overline{C}$

EXEMPLUL 4. Pornind de la schema logică din **figura 3.2.7** să se determine expresia analitică a funcţiei de ieşire a circuitului.

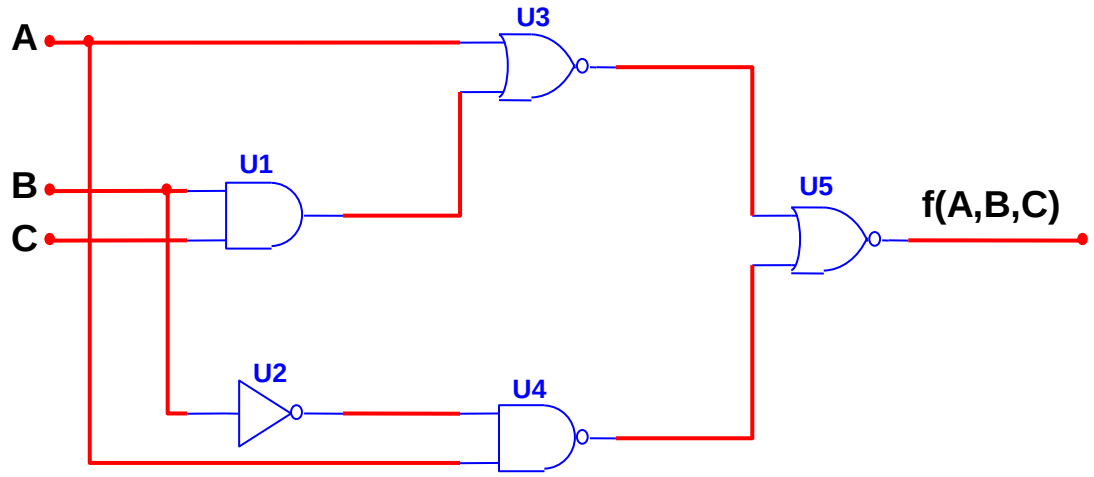


Figura 3.2.7 Schemă logică cu porţi elementare

REZOLVARE

În **figura 3.2.8** sunt prezentate expresiile logice de la ieşirea fiecărei porţi logice din circuit

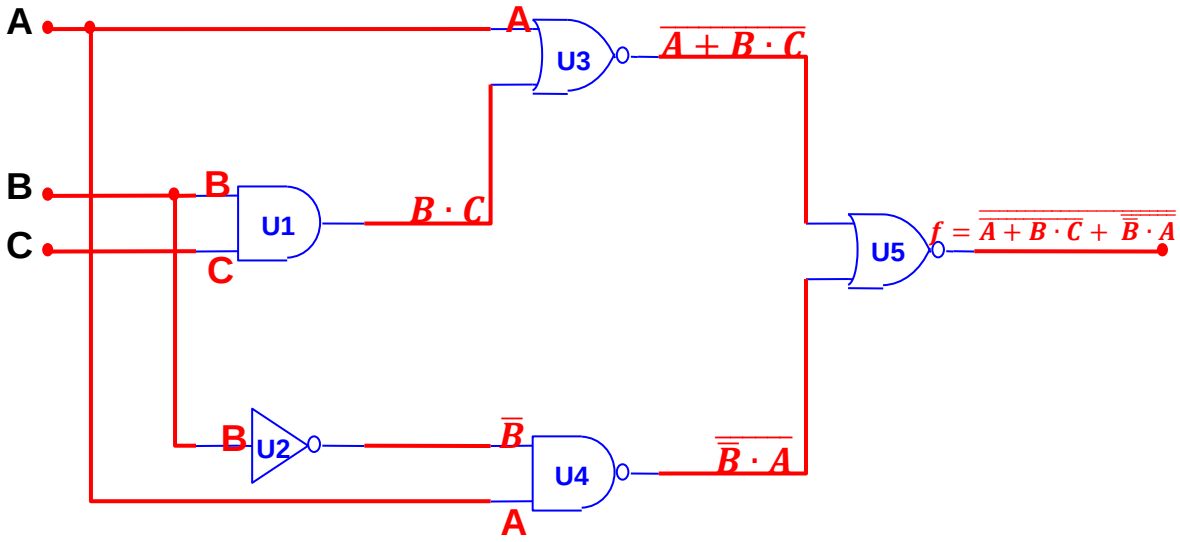


Figura 3.2.8 Schemă logică cu porţi elementare şi expresiile funcţiilor de ieşire

Expresia analitică a funcţiei logice de ieşire este $\mathbf{f = \overline{\overline{A + B \cdot C} + \overline{\overline{B} \cdot A}}}$

Funcţia se aduce la o formă mai simplă aplicând axiomele şi teoremele algebrei logice:

$$f = \overline{\overline{A + B \cdot C} + \overline{\overline{B} \cdot A}} = \left(\overline{\overline{A + B \cdot C}}\right) \cdot \left(\overline{\overline{\overline{B} \cdot A}}\right) = (A + B \cdot C) \cdot (A \cdot \overline{B}) = A \cdot A \cdot \overline{B} + B \cdot C \cdot \overline{B}$$

$$\boldsymbol{f = A \cdot \overline{B}}$$

3.2.2 SINTEZA CIRCUITELOR LOGICE

La ***sinteza*** circuitelor logice se pleacă de la funcţia pe care trebuie să o îndeplinească circuitul (care se cunoaşte) şi se urmăreşte obţinerea unei variante minimale a structurii acestuia (care se determină).

Sinteza circuitelor logice presupune parcurgerea următoarelor etape:

- Definirea funcţiei de ieşire;
- Minimizarea funcţiei de ieşire;
- Desenarea schemei circuitului.

După modul de exprimare a funcţiei de ieşire implementarea circuitului logic se poate face în mai multe variante:

- Cu orice combinaţie de circuite logice elementare;
- Numai cu circuite “NAND”;
- Numai cu circuite “NOR”.

Pentru a înţelege mai bine în cele ce urmează sunt prezentate câteva exemple.

EXEMPLUL 1. Pornind de funcţia $\mathbf{Y = A \oplus B}$ să se realizeze sinteza circuitului corespunzător în mai multe variante.

a) Sinteza utilizând orice combinaţie de circuite logice elementare.

Pornim de la forma canonică a funcţiei $\mathbf{Y = A \oplus B = \bar{A} \cdot B + A \cdot \bar{B}}$

Ştiind că $A \cdot \bar{A} = 0$ şi $B \cdot \bar{B} = 0$ şi înlocuind în forma canonică a funcţiei se obţine:

$$Y = \bar{A} \cdot B + A \cdot \bar{A} + A \cdot \bar{B} + B \cdot \bar{B} = \bar{A}(A + B) + \bar{B}(A + B) = (A + B) \cdot (\bar{A} + \bar{B})$$

Deci forma funcţiei de ieşire a circuitului este $\mathbf{Y = (A + B) \cdot (\bar{A} + \bar{B})}$

Pentru a obţine $\bar{A}$ şi $\bar{B}$ avem nevoie de două porţi **NU(NOT) – U1A, U1B**

Pentru a obţine $(A + B)$ şi $\bar{A} + \bar{B}$ avem nevoie de două porţi **SAU (OR) - U2A, U2B**

Pentru a obţine $(A + B) \cdot (\bar{A} + \bar{B})$ avem nevoie de o poartă **ŞI (AND) – U3A**

În urma implementării se obţine schema logică din **figura 3.2.9**

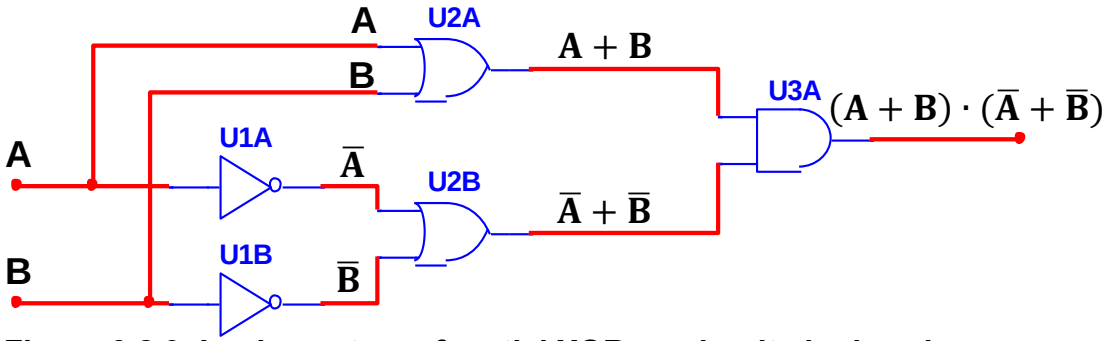


Figura 3.2.9 Implementarea funcţiei XOR cu circuite logice elementare

b) Sinteza utilizând numai circuite ŞI-NU (NAND)

Pornim de la forma canonică a funcţiei $Y = A \oplus B = \overline{A} \cdot B + A \cdot \overline{B}$

Deoarece utilizăm numai circuite NAND trebuie să transformăm funcţia într-un produs negat de doi termeni.

Utilizând formula lui Morgan $A + B = \overline{\overline{A} \cdot \overline{B}}$ se obţine funcţia:

$$Y = A \oplus B = \overline{A} \cdot B + A \cdot \overline{B} = \overline{(\overline{A} \cdot B)} \cdot \overline{(A \cdot \overline{B})}$$

Deci forma funcţiei de ieşire a circuitului este: $Y = \overline{\overline{(\overline{A} \cdot B)} \cdot \overline{(A \cdot \overline{B})}}$

Pentru a obţine $\overline{A}$ şi $\overline{B}$ avem nevoie de două porţi **NAND – U1A, U1B**

Pentru a obţine $\overline{(\overline{A} \cdot B)}$ şi $\overline{(A \cdot \overline{B})}$ avem nevoie de două porţi **NAND – U1C, U1D**

Pentru a obţine $\overline{\overline{(\overline{A} \cdot B)} \cdot \overline{(A \cdot \overline{B})}}$ avem nevoie de o poartă **NAND – U2D**

În urma implementării se obţine schema logică din **figura 3.2.10**

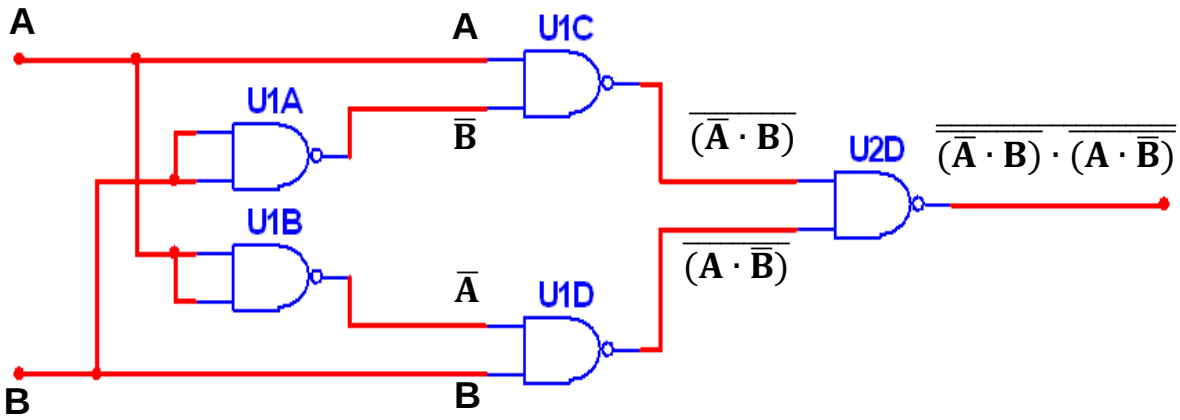


Figura 3.2.10 Implementarea funcţiei XOR cu circuite logice NAND

c) Sinteza utilizând numai circuite SAU-NU (NOR)

Pornim de la forma canonică a funcţiei $\mathbf{Y} = (\mathbf{A} + \mathbf{B}) \cdot (\overline{\mathbf{A}} + \overline{\mathbf{B}})$

Deoarece utilizăm numai circuite NOR trebuie să transformăm funcţia într-o sumă negată de doi termeni.

Utilizând formula lui Morgan $\mathbf{A} \cdot \mathbf{B} = \overline{\overline{\mathbf{A}} + \overline{\mathbf{B}}}$ se obţine funcţia:

$$\boldsymbol{Y} = (\mathbf{A} + \mathbf{B}) \cdot (\overline{\mathbf{A}} + \overline{\mathbf{B}}) = \overline{\overline{(\mathbf{A} + \mathbf{B})} + \overline{(\overline{\mathbf{A}} + \overline{\mathbf{B}})}}$$

Deci forma funcţiei de ieşire a circuitului este: $\boldsymbol{Y} = \overline{\overline{(\mathbf{A} + \mathbf{B})} + \overline{(\overline{\mathbf{A}} + \overline{\mathbf{B}})}}$

Pentru a obţine $\overline{\mathbf{A}}$ şi $\overline{\mathbf{B}}$ avem nevoie de două porţi **NOR – U1A, U1B**

Pentru a obţine $\overline{(\mathbf{A} + \mathbf{B})}$ şi $\overline{(\overline{\mathbf{A}} + \overline{\mathbf{B}})}$ avem nevoie de două porţi **NOR – U1C, U1D**

Pentru a obţine $\overline{\overline{(\mathbf{A} + \mathbf{B})} + \overline{(\overline{\mathbf{A}} + \overline{\mathbf{B}})}}$ avem nevoie de o poartă **NOR – U2D**

În urma implementării se obţine schema logică din **figura 3.2.11**

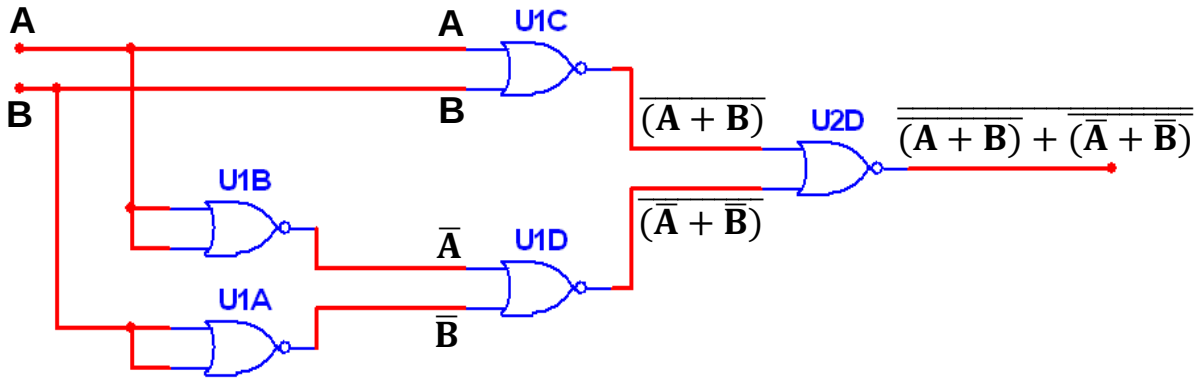


Figura 3.2.11 Implementarea funcţiei XOR cu circuite logice NOR

EXEMPLUL 2. Pornind de funcţia $f = (\overline{\bar{A} + \overline{AB}}) + \overline{BC}$ să se realizeze sinteza circuitului corespunzător utilizând orice combinaţie de circuite logice elementare.

Pentru a obţine $\bar{A}$ avem nevoie de o poartă **NU (NOT) – U1A**

Pentru a obţine $\overline{AB}$ şi $\overline{BC}$ avem nevoie de două porţi **ŞI-NU (NAND) – U2A, U2B**

Pentru a obţine $(\overline{\bar{A} + \overline{AB}})$ avem nevoie de o poartă **SAU-NU (NOR) – U3A**

Pentru a obţine $(\overline{\bar{A} + \overline{AB}}) + \overline{BC}$ avem nevoie de o poartă **SAU (OR) – U4A**

În urma implementării se obţine schema logică din **figura 3.2.12**

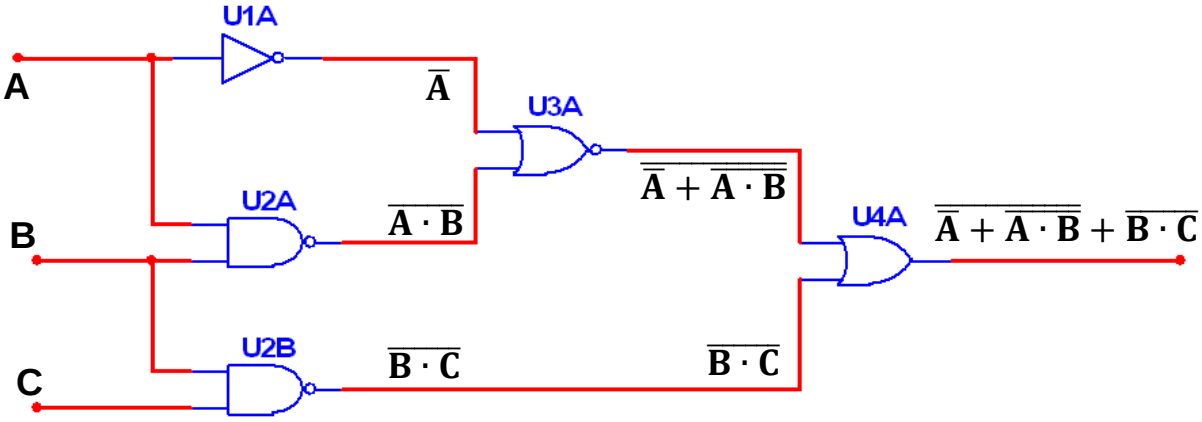


Figura 3.2.12 Implementarea unei funcţii logice cu circuite logice elementare

Numărul porţilor utilizate se pot reduce prin simplificarea funcţie logice.

Ştiind că $\overline{A \cdot B} = \bar{A} + \bar{B}$ şi $\overline{\bar{A} + \bar{B}} = AB$ funcţia iniţială se transformă astfel:

$f = \overline{\bar{A} + \overline{A \cdot B}} + \overline{B \cdot C} = \overline{\bar{A} + \bar{A} + \bar{B}} + \overline{BC} = \overline{\bar{A} + \bar{B}} + \overline{BC} = AB + \overline{BC}$

$f = AB + \overline{BC}$

În urma implementării se obţine schema logică din **figura 3.2.13**

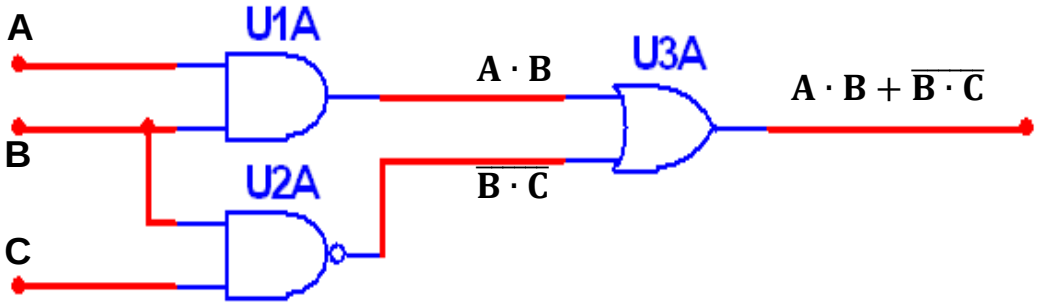
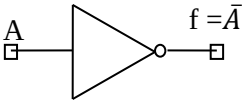
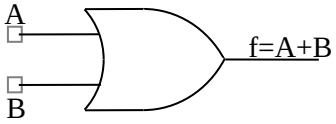
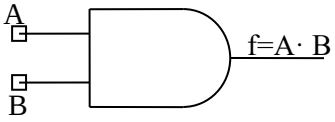


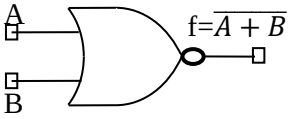
Figura 3.2.13 Implementarea unei funcţii logice cu circuite logice elementare

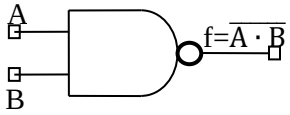
REZUMATUL CAPITOLULUI

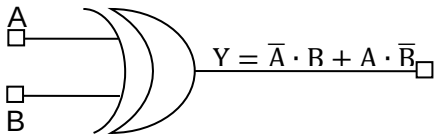
- ***Poarta logică este un circuit combinaţional*** care are una sau mai multe intrări digitale care formează o combinaţie de valori binare (**0** şi **1**), iar la ieşire o singură stare binară (**0** sau **1**).
- Pentru toate porţile logice se respectă convenţiile:
  - **0 logic** este echivalent cu **nivel de tensiune scăzut (L)** sau **0 V**;
  - **1 logic** este echivalent cu **nivel de tensiune ridicat (H)** sau **+ V.**
- Porțile logice elementare se reprezintă astfel:
  - Poarta logică ***NU (NOT)*** 

  - Poarta logică ***SAU (OR)*** 

  - Poarta logică ***ŞI (AND)*** 

  - Poarta logică ***SAU-NU (NOR)*** 

  - Poarta logică ***ŞI-NU (NAND)*** 

  - Poarta logică ***SAU-EXCLUSIV (XOR)*** 

- Pentru a **analiza un circuit logic** se pornește de la schema circuitului logic și se determină expresiei funcţiei logice de ieşire parcurgând schema de la stânga spre dreapta prin scrierea expresiilor de la ieşirea porţilor logice care formează circuitul respectiv din aproape în aproape.
- La ***sinteza*** circuitelor logice se pleacă de la funcţia pe care trebuie să o îndeplinească circuitul şi se urmăreşte obţinerea unei variante minimale a structurii acestuia care se determină parcurgând etapele:
  - Minimizarea funcţiei de ieşire;
  - Desenarea schemei circuitului.

EVALUAREA CUNOȘTINȚELOR

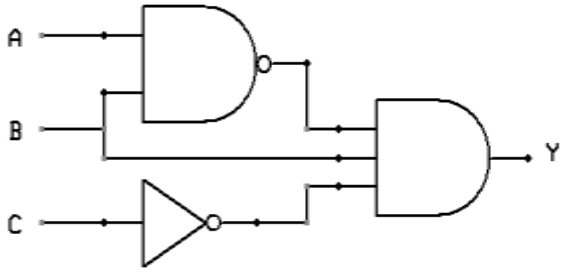
1. Pornind de la schema logică din fiecare imagine:

A. determinați expresia analitică a funcţiei de ieşire a circuitului;

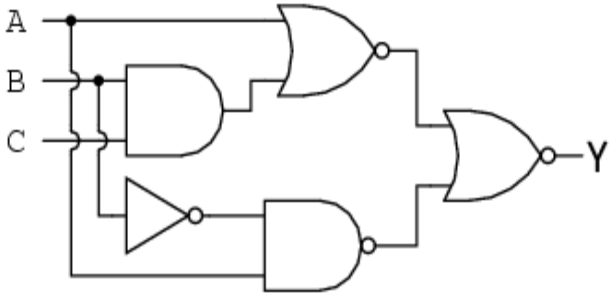
B. simplificați expresia funcției de ieșire;

C. reprezentați schema logică corespunzătoare funcției de ieșire simplificate a circuitului.

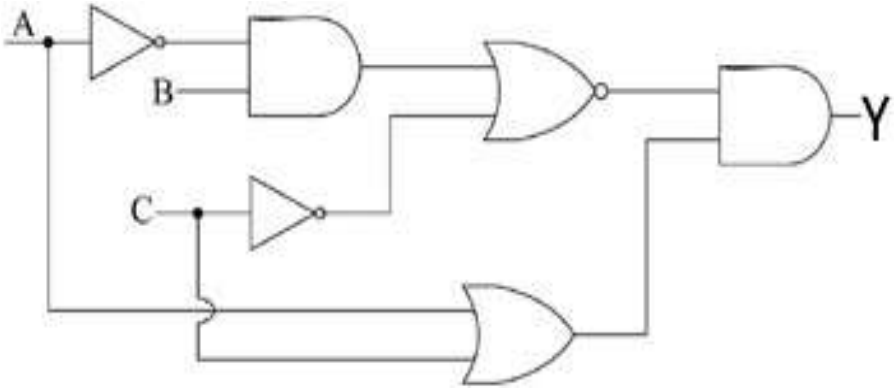
a.



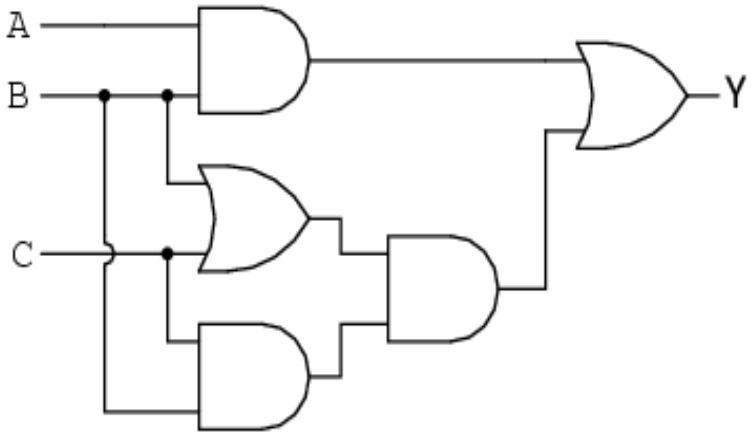
b.



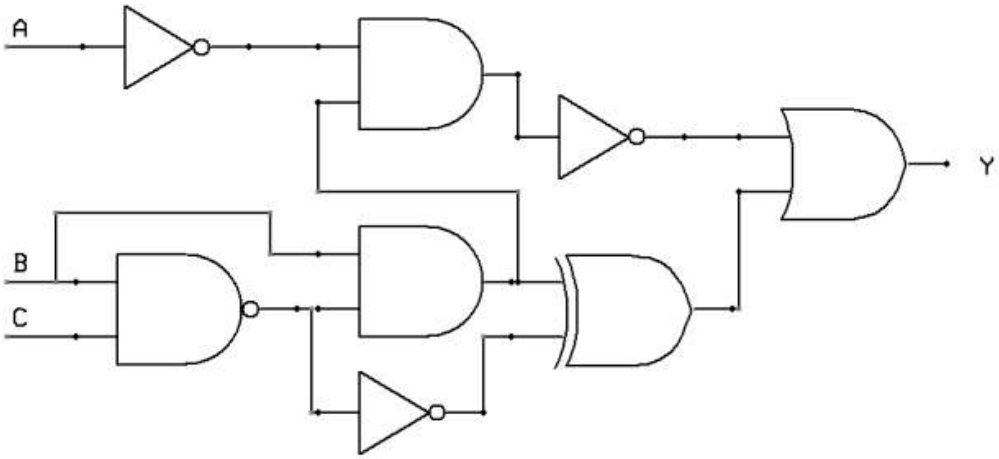
c.



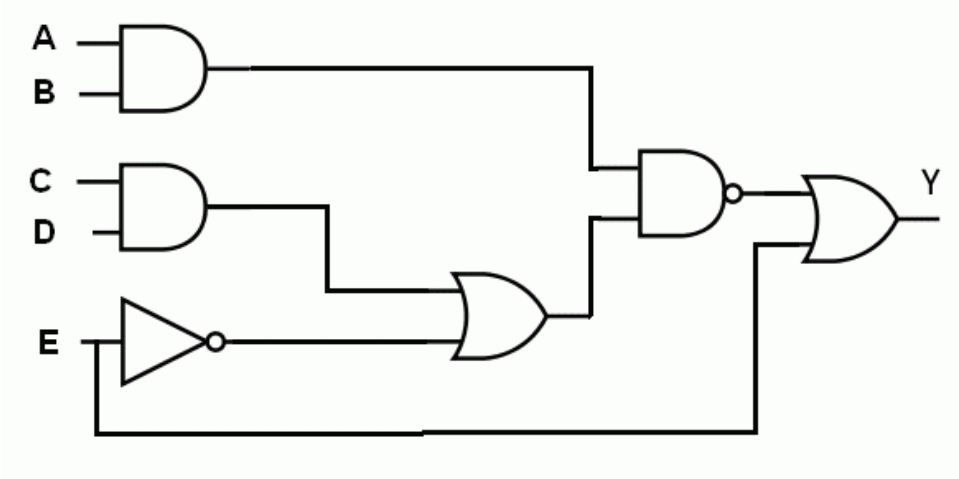
d.



e.



f.



2. Pornind de funcţiile logice prezentate mai jos, pentru fiecare funcție:

A. să se realizeze tabela de adevăr a funcției;

B. să se implementeze funcția dată utilizând orice combinaţie de circuite logice elementare;

C. să se simplifice funcția dată;

D. să se implementeze funcția simplificată utilizând porți logice;

a. $f = A \cdot B + B \cdot C + A \cdot C$;

b. $f = A \cdot (\overline{B + C}) + D$;

c. $f = A \cdot B + B \cdot C \cdot (B + C)$;

d. $f = (\overline{\bar{A} \cdot B + \bar{C}}) \cdot (A + C)$;

e. $f = \overline{\overline{A + BC} + \overline{\overline{A \cdot B}}}$;

f. $f = (\overline{A \cdot \bar{B} \cdot C} + B) \cdot (\overline{A + \bar{C}}) \cdot B$.

CAPITOLUL 4. CIRCUITE LOGICE ELEMENTARE

4.1. CIRCUITE LOGICE CU COMPONENTE DISCRETE

4.1.1 PORŢI LOGICE ELEMENTARE CU COMPONENTE PASIVE

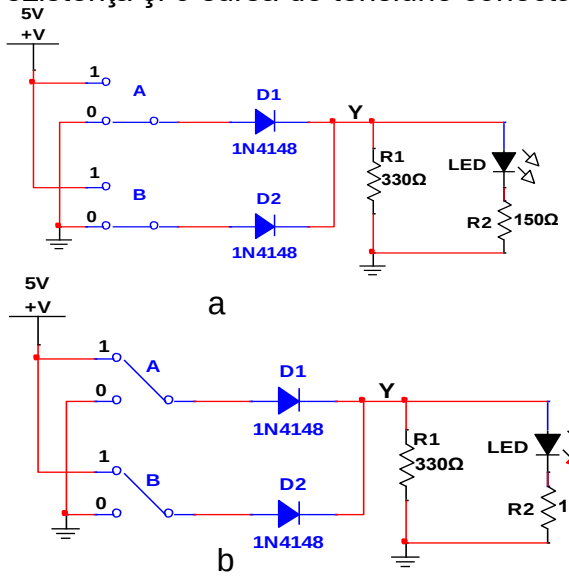
Componente electronice pasive sunt componente care nu au capacitatea de a amplifica semnalul aplicat la intrare. Din această categorie de componente, pentru realizarea porţilor logice elementare, cele mai utilizate sunt rezistoarele şi diodele.

Pentru a înţelege funcţionarea porţilor logice elementare cu se ţine seama de următoarele convenţii:

- "0" logic sau nivel jos – L este echivalent cu 0 volţi;
- "1" logic sau nivel sus – H este echivalent cu +V volţi (tensiunea maximă a sursei de alimentare, care poate fi +5V sau +15V în funcţie de componentele utilizate);
- Pentru comutarea intrărilor porţii logice în "0" sau "1" logic se utilizează câte un comutator pentru fiecare intrare a porţii (în poziţia 0 a comutatorului intrarea este conectată la 0 volţi adică "0" logic, iar poziţia 1 intrarea este conectată la +V adică "1" logic);
- La ieşirea porţii logice se conectează un LED înseriat cu un rezistor care semnifică:
  - "1" logic – dacă este aprins;
  - "0" logic – dacă este stins.

POARTA LOGICĂ "SAU" – cu diode.

Pentru construcţia unei porţi logice SAU se utilizează două sau mai multe diode, o rezistenţă şi o sursă de tensiune conectate ca în **figura 4.1.1**.



a

b

TABELA DE ADEVĂR

A	B	Y=A+B
0	0	0
0	1	1
1	0	1
1	1	1

c

Figura 4.1.1 Poarta logică "SAU" cu diode

FUNCŢIONARE.

Dacă la ambele intrări se aplică 0 volţi, diodele sunt blocate şi la ieşire avem 0 volţi

A şi B în 0 logic ⇒ ieşirea Y în 0 logic ⇒ LED stins (**figura 4.1.1 a**).

Dacă la una din cele două intrări se aplică +5V, dioda corespunzătoare intrării conduce şi la ieşirea avem +5V.

A sau B în 1 logic ⇒ ieşirea Y în 1 logic ⇒ LED aprins (**figura 4.1.1 b**).

Funcţia logică a porţii SAU este **Y=A+B** (**figura 4.1.1 c**).

POARTA LOGICĂ "ŞI" – cu diode.

Pentru construcţia unei porţi logice ŞI se utilizează două sau mai multe diode, o rezistenţă şi o sursă de tensiune conectate ca în **figura 4.1.2**.

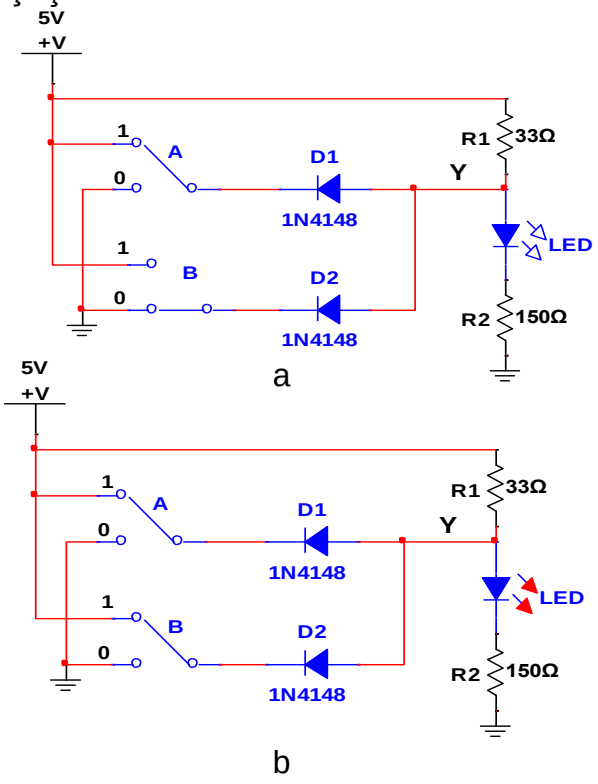


TABELA DE ADEVĂR

A	B	Y=A·B
0	0	0
0	1	0
1	0	0
1	1	1

c

Figura 4.1.2 Poarta logică "ŞI" cu diode

FUNCŢIONARE.

Dacă la una din cele două intrări se aplică 0 volţi, dioda corespunzătoare intrării respective conduce şi la ieşire avem 0 volţi

A sau B în 0 logic ⇒ ieşirea Y în 0 logic ⇒ LED stins (**figura 4.1.2 a**).

Dacă la ambele intrări se aplică +5V, ambele diode sunt blocate şi la ieşirea avem +5V.

A şi B în 1 logic ⇒ ieşirea Y în 1 logic ⇒ LED aprins (**figura 4.1.2 b**).

Funcţia logică a porţii SAU este **Y=A·B** (**figura 4.1.2 c**).

4.1.2 PORŢI LOGICE ELEMENTARE CU COMPONENTE ACTIVE

Acest tip de circuite conţin şi tranzistoare bipolare care sunt componente active de circuit deoarece sunt capabile să amplifice un semnal.

POARTA LOGICĂ "NU" – cu tranzistoare.

Pentru construcţia unei porţi logice NU se utilizează un tranzistor bipolar, mai multe rezistenţe şi o sursă de tensiune conectate ca în **figura 4.1.3**.

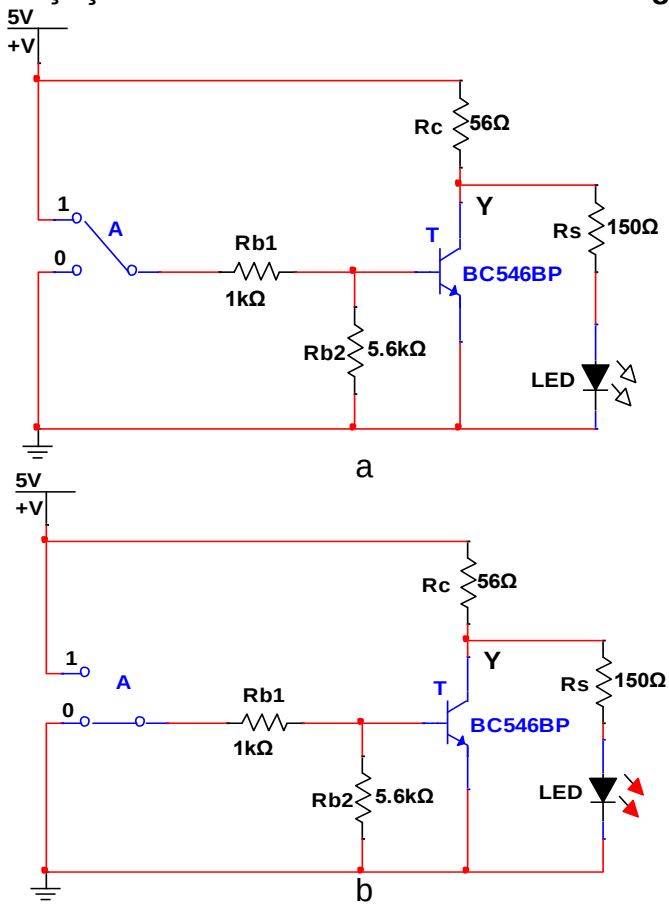


TABELA DE ADEVĂR

A	$Y = \overline{A}$
0	1
1	0

c

Figura 4.1.3 Poarta logică "NU" cu tranzistor

FUNCŢIONARE:

Când rezistenţa Rb1 este conectată prin intermediul comutatorului A la +V, joncţiunea bază-emitor a tranzistorului T este polarizată direct prin intermediul divizorului de tensiune Rb1-Rb2 iar tranzistorul T este saturat, situaţie în care în colectorul tranzistorului tensiunea este foarte mică (aproximativ 0 V).

A în 1 logic ⇒ ieşirea Y în 0 logic ⇒ LED stins (**figura 4.1.3 a**).

Când rezistenţa Rb1 este conectată prin intermediul comutatorului A la 0V, în baza tranzistorului T sunt 0V iar tranzistorul este blocat, situaţie în care în colectorul tranzistorului tensiunea este mare (aproximativ 4 V).

A în 0 logic ⇒ ieşirea Y în 1 logic ⇒ LED aprins (**figura 4.1.3 b**).

POARTA LOGICĂ "SAU" – cu tranzistoare.

Pentru construcţia unei porţi logice SAU se utilizează două tranzistoare bipolare, mai multe rezistenţe şi o sursă de tensiune conectate ca în **figura 4.1.4**.

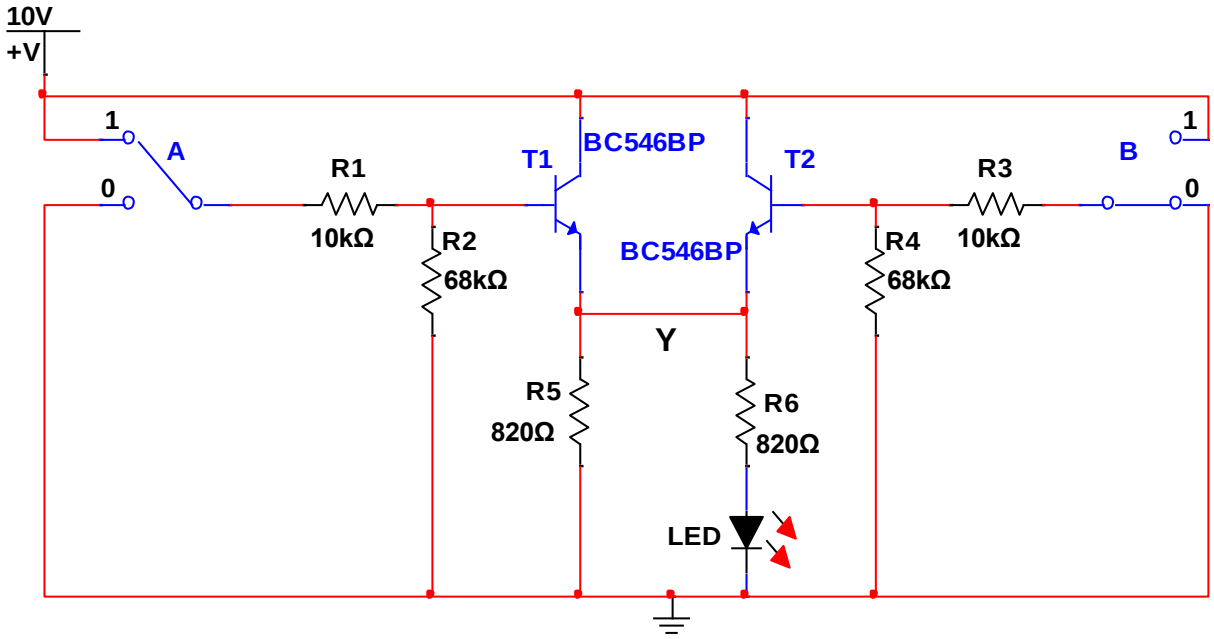


Figura 4.1.4 Poarta logică "SAU" cu tranzistoare

FUNCŢIONARE:

Dacă comutatorul A este conectat la +V, joncţiunea bază-emitor a tranzistorului T1 este polarizată direct prin intermediul divizorului de tensiune R1-R2, iar tranzistorul T1 este saturat. În această situaţie în emitorul tranzistorului T1 tensiunea este aproximativ +7,5 V şi LED-ul luminează.

A în 1 logic ⇒ tranzistorul T1 saturat ⇒ ieşirea Y în 1 logic ⇒ LED aprins

Dacă comutatorul B este conectat la +V, joncţiunea bază-emitor a tranzistorului T2 este polarizată direct prin intermediul divizorului de tensiune R3-R4, iar tranzistorul T2 este saturat. În această situaţie în emitorul tranzistorului T2 tensiunea este aproximativ +7,5 V şi LED-ul luminează.

B în 1 logic ⇒ tranzistorul T2 saturat ⇒ ieşirea Y în 1 logic ⇒ LED aprins

Dacă ambele comutatoare A şi B sunt conectate la 0 V, bazele celor două tranzistoare sunt conectate la 0 V prin intermediul rezistoarelor de 10 KΩ . În această situaţie cele două tranzistoare T1 şi T2 sunt blocate iar LED-ul este stins

Dacă A şi B în 0 logic ⇒ T1 şi T2 blocate ⇒ ieşirea Y în 0 logic ⇒ LED stins

POARTA LOGICĂ "ŞI" – cu tranzistoare.

Pentru construcţia unei porţi logice ŞI se utilizează două tranzistoare bipolare, mai multe rezistenţe şi o sursă de tensiune conectate ca în **figura 4.1.5**.

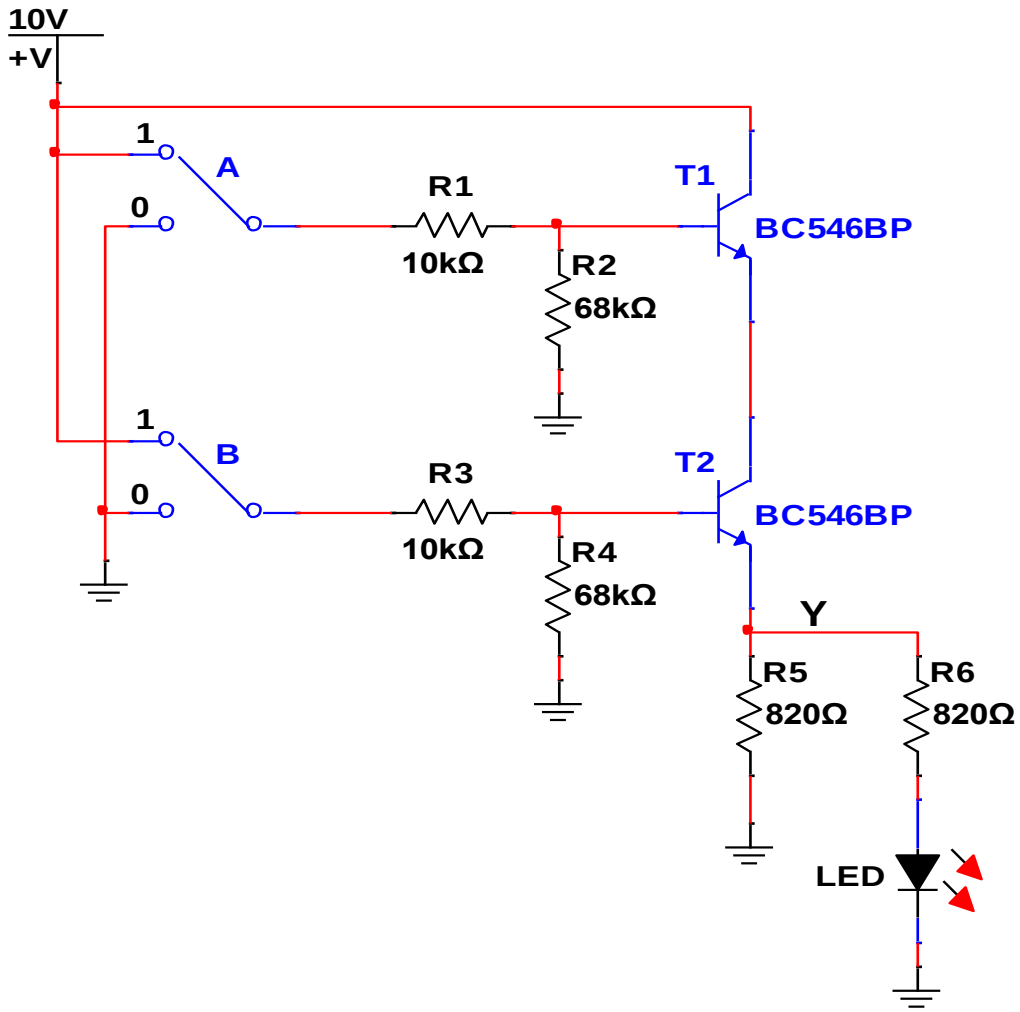


Figura 4.1.5 Poarta logică "ŞI" cu tranzistoare

FUNCŢIONARE:

Dacă comutatorul A sau comutatorul B este conectat la 0 V, tranzistorul corespunzător comutatorului respectiv este blocat. Dacă unul din cele două tranzistoare este blocat, tensiunea în emitorul tranzistorului T2 este aproximativ 0 V. În această situaţie ieşirea Y este în 0 logic şi LED-ul este stins.

Dacă A sau B în 0 logic ⇒ T1 sau T2 blocat ⇒ ieşirea Y în 0 logic ⇒ LED stins

Dacă ambele comutatoare sunt conectate la +V, tranzistoarele T1 şi T2 sunt saturate situaţie în care tensiunea în emitorul tranzistorului T2 este aproximativ + 7,4 V. În această situaţie ieşirea Y este în 1 logic şi LED-ul este aprins.

Dacă A şi B în 1 logic ⇒ T1 şi T2 saturate ⇒ ieşirea Y în 1 logic ⇒ LED aprins.

POARTA LOGICĂ "SAU- NU" (NOR) – cu diode şi tranzistoare.

Pentru construcţia acestei porţi se utilizează o poartă “SAU” construită cu diode (**vezi figura 4.1.1 a**) şi o poartă “NU” construită cu tranzistoare (**vezi figura 4.1.3 a**).

Poarta logică SAU-NU cu diode şi tranzistoare este prezentată în **figura 4.1.6**.

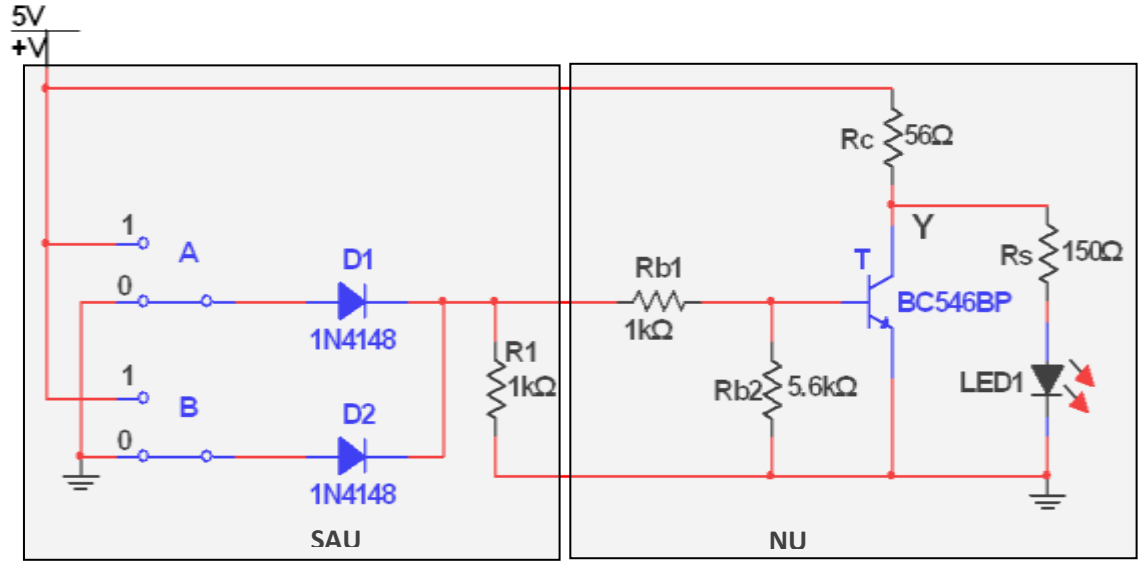


Figura 4.1.6 Poarta logică “SAU-NU” cu diode şi tranzistoare

FUNCŢIONARE:

Dacă comutatorul A sau B sau ambele comutatoare sunt conectate la +V, joncţiunea bază-emitor a tranzistorului T este polarizată direct şi tranzistorul este saturat. În această situaţie în colectorul tranzistorului T este o tensiune de aproximativ 0 V iar LED1 este stins.

Dacă A şi/sau B în 1 logic ⇒ T saturat ⇒ ieşirea Y în 0 logic ⇒ LED stins

Dacă ambele comutatoare sunt conectate la 0 V, ambele diode D1, D2 sun blocate, în baza tranzistorului T este o tensiune de aproximativ 0 V iar tranzistorul T este blocat. În această situaţie în colectorul tranzistorului T este o tensiune de aproximativ 4 V iar LED1 este aprins.

Dacă A şi B în 0 logic ⇒ T blocat ⇒ ieşirea Y în 1 logic ⇒ LED aprins

TABELA DE ADEVĂR

A	B	$Y = \overline{A + B}$
0	0	1
0	1	0
1	0	0
1	1	0

POARTA LOGICĂ "ŞI- NU" (NAND) – cu diode şi tranzistoare.

Pentru construcţia acestei porţi se utilizează o poartă "ŞI" construită cu diode (**vezi fig. 4.1.2 a**) şi o poartă "NU" construită cu tranzistoare (**vezi fig. 4.1.3 a**).

Poarta logică ŞI-NU cu diode şi tranzistoare este prezentată în **figura 4.1.7**.

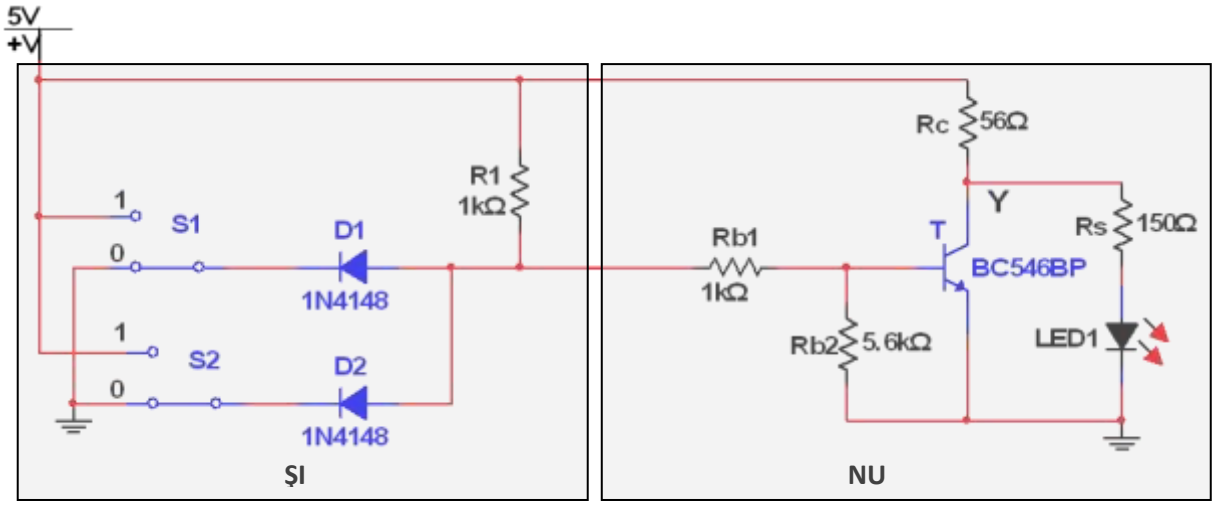


Figura 4.1.7 Poarta logică "ŞI-NU" cu diode şi tranzistoare

FUNCŢIONARE:

Dacă comutatorul A sau B sau ambele comutatoare sunt conectate la 0 V, dioda D1 sau D2 sau ambele diode sunt în conducţie iar în baza tranzistorului T este o tensiune de aproximativ 0 V, tranzistorul T este blocat. În această situaţie în colectorul tranzistorului T este o tensiune de aproximativ 4 V iar LED1 este aprins.

Dacă A şi/sau B în 0 logic ⇒ T blocat⇒ ieşirea Y în 1 logic ⇒ LED aprins

Dacă ambele comutatoare sunt conectate la + V, diodele D1 şi D2 sunt blocate, iar joncţiunea bază-emitor a tranzistorului T este polarizată direct, tranzistorul T este saturat. În această situaţie în colectorul tranzistorului T este o tensiune de aproximativ 0 V iar LED1 este stins.

Dacă A şi B în 1 logic ⇒ T saturat ⇒ ieşirea Y în 0 logic ⇒ LED stins

TABELA DE ADEVĂR

A	B	$Y = \overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

4.2. CIRCUITE LOGICE ÎN TEHNOLOGIE INTEGRATĂ

În prezent, circuitele logice se realizează în exclusivitate prin tehnica integrării monolitice. În funcţie de tehnologia utilizată, circuitele logice integrate se împart în două categorii:

- Circuite integrate bipolare – TTL (au în componenţă tranzistori bipolari);
- Circuite integrate monopolare – MOS (au în componenţă tranzistori cu efect de câmp).

4.2.1 CIRCUITE LOGICE INTEGRATE BIPOLARE

Familia de circuite integrate TTL (Transistor Transistor Logic), este cea mai răspândită familie de circuite integrate logice.

Circuitele sunt realizate cu tranzistori bipolari fără condensatori de cuplaj între ei (cu cuplaj direct).

Cea mai răspândită familie de circuite logice integrate TTL este seria **74xx** pentru aplicaţii comerciale (**tabelul 4.2.1**).

Tabelul 4.2.1 – Exemple de circuite integrate TTL

Codul circuitului integrat	Tipul porţilor	Numărul intrărilor unei porţi	Numărul porţilor pe circuitul integrat
7404	***NOT***	1	6
7408	***AND***	2	4
7411	***AND***	3	3
7421	***AND***	4	2
7432	***OR***	2	4
7400	***NAND***	2	4
7410	***NAND***	3	3
7420	***NAND***	4	2
7430	***NAND***	8	1
7402	***NOR***	2	4
7427	***NOR***	3	3
7486	***XOR***	2	4

PARAMETRII CIRCUITELOR LOGICE INTEGRATE TTL

Tensiunea de alimentare*: 4,75 V .... 5,25 V*

Tensiunile de intrare*:*

0 V..... 0,8 V *sunt interpretate ca* ***0 logic (L)***

2 V..... 5 V *sunt interpretate ca* ***1 logic (H)***

Tensiunea de intrare corespunzătoare nivelului **L**: $\mathbf{V_{IL(MAX)} = 0,8\ V}$

Tensiunea de intrare corespunzătoare nivelului **H**: $\mathbf{V_{IH(MIN)} = 2\ V}$

Domeniul 0,8 V....2 V dintre cele două nivele limită se numeşte ***domeniu de incertitudine***

Tensiunile de ieşire:

0 V..... 0,4 V *sunt interpretate ca* ***0 logic (L)***

2,4 V..... 5 V *sunt interpretate ca* ***1 logic (H)***

Tensiunea de ieşire garantată pentru nivelului **L**: $\mathbf{V_{OL(MAX)} = 0,4\ V}$

Tensiunea de ieşire garantată pentru nivelului **H**: $\mathbf{V_{OH(MIN)} = 2,4\ V}$

Diferenţa, în modul, dintre tensiunea de ieşire garantată şi tensiunea de intrare corespunzătoare reprezintă ***marginea de zgomot*** a porţii.

$$\boldsymbol{V_{IL} - V_{OL} = V_{OH} - V_{IH} = 0,4\ V}$$

Cu toate că este garantată o margine de zgomot de numai 0,4 V, practic, o poartă TTL are o margine de zgomot de 1,4 V.

Curenţii de intrare

Valoarea curentului de intrare garantat pentru nivelul **L**: $\mathbf{I_{IL} = -\ 1,6\ mA}$

Valoarea curentului de intrare garantat pentru nivelul **H**: $\mathbf{I_{IH} = 40\ \mu A}$

Curenţii de ieşire

Valoarea curentului de ieşire garantat pentru nivelul **L**: $\mathbf{I_{0L} = 16\ mA}$

Valoarea curentului de ieşire garantat pentru nivelul **H**: $\mathbf{I_{OH} = -\ 800\ \mu A}$

Factorul de încărcare (sortanţă) - "fan-in" ; "fan-out"

Fan-in reprezintă numărul maxim de ieşiri ce pot fi conectate în paralele la o intrare

Fan-out reprezintă numărul maxim de intrări ce pot fi conectate la o ieşire

Pentru porţile TTL standard ***fan-out = 10***

Timpul de propagare (timpul de întârziere la propagarea informaţiei)

Timpul de propagare pentru variaţia ieşirii din L în H $\boldsymbol{t_{pLH} = 12\ ns}$

Timpul de propagare pentru variaţia ieşirii din H în L $\boldsymbol{t_{pLH} = 8\ ns}$

Valoarea medie tipică a timpului de propagare este **10 ns**.

POARTA TTL ŞI-NU (NAND)

Toate seriile TTL au drept circuit fundamental poarta ŞI-NU (NAND).

Seria TTL 74xx are poarta logică fundamentală realizată cu 4 tranzistori bipolari, conectaţi ca în **figura 4.2.1**.

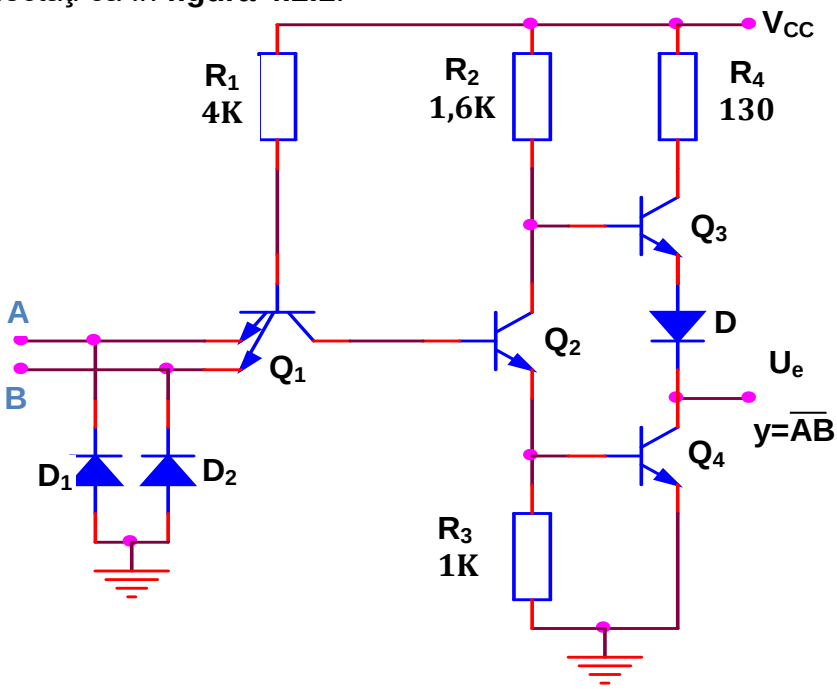


Figura 4.2.1 Poartă TTL standard

Elementele constructive ale circuitului:

- **Tranzistorul multiemitor Q1**- realizează funcţia logică ***ŞI***;
- **Tranzistorul Q2** – realizează funcţia logică ***NU***;
- **Tranzistoarele Q3,Q4, dioda D** – etaj de ieşire contratimp, asigură o impedanţă de ieşire mică**;**
- **Diodele D1, D2** – diode de tăiere, limitează oscilaţiile negative de intrare şi amortizează oscilaţiile parazite.

Funcţionarea circuitului:

- Dacă una dintre intrările **A** sau **B** este în **"0" logic**, tranzistorul **Q1** se saturează.

Q1 saturat ⇒ Q2 blocat ⇒ Q4 blocat şi Q3 saturat

În această situaţie la ieşire este **"1" logic.**

- Dacă ambele intrări **A** şi **B** sunt în **"1" logic**, tranzistorul **Q1** este blocat.

Q1 blocat ⇒ Q2 saturat ⇒ Q4 saturat şi Q3 blocat

În această situaţie la ieşire este **"0" logic.**

REGULI DE UTILIZARE ALE CIRCUITELOR LOGICE DIN FAMILIA TTL

1. Nici o intrare a unui circuit logic TTL nu se lasă flotantă (neconectată).
   a. La circuitele **ŞI** respectiv **ŞI-NU** intrările neutilizate se conectează prin intermediul unei rezistenţe de polarizare $\mathbf{R_p}$ la $\mathbf{+V_{CC}}$
   b. La circuitele **SAU** respectiv **SAU-NU** intrările neutilizate se conectează direct la „masa” montajului.
2. Intrările neutilizate se pot conecta la alte intrări.
3. Ieşirile circuitelor logice nu se conectează niciodată direct la $\mathbf{+V_{CC}}$ sau **masă**.
4. Este interzisă interconectarea ieşirilor a două sau mai multe circuite TTL dacă există posibilitatea ca aceste ieşiri să ajungă la niveluri logice diferite.
5. Decuplarea circuitelor integrate TTL este obligatorie. Deoarece pe durata frontului consumul unei porţi creşte de circa 20 de ori faţă de curentul mediu de alimentare, pentru a evita o cădere de tensiune pe traseele de alimentare mai mare de 0,5 V, se conectează condensatori nepolarizaţi între aceste trasee care decuplează circuitele integrate TTL.

4.2.2 CIRCUITE LOGICE INTEGRATE MONOPOLARE

Circuitele logice integrate realizate în tehnologie monopolară se împart în 3 familii:

- Familia PMOS – utilizează numai tranzistoare MOS cu canal de tip P. Aceste circuite au procesul de fabricaţie simplu dar viteza de comutaţie mică;
- Familia NMOS – utilizează numai tranzistoare MOS cu canal de tip N. Aceste circuite au procesul de fabricaţie mai complicat dar viteza de comutaţie este mare;
- Familia CMOS – utilizează tranzistoare MOS complementare unele cu canal de tip P şi altele cu canal de tip N. Aceste circuite au o viteză de comutaţie medie şi un consum redus de energie.

  Circuitele integrate CMOS sunt la ora actuală cele mai utilizate circuite logice integrate monopolare datorită următoarelor particularităţi:
  - Gamă mare pentru tensiunea de alimentare: **3,5 V ... 15 V**;
  - Putere de consum mică;
  - Viteză de lucru bună;
  - Imunitate la zgomot foarte bună : **45%**;
  - Densitate de integrare mare.

Circuitele logice CMOS se fabrică în mai multe serii, cea mai utilizată fiind seria **40xx** (vezi **tabelul 4.2.2**).

Tabelul 4.2.2 – Exemple de circuite integrate CMOS

Codul circuitului integrat	Tipul porţilor	Numărul intrărilor într-o poartă	Numărul porţilor pe circuitul integrat
MMC 4001	***NOR***	**2**	**4**
MMC 4002	***NOR***	**4**	**2**
MMC 4011	***NAND***	**2**	**4**
MMC 4012	***NAND***	**4**	**2**
MMC 4023	***NAND***	**3**	**3**
MMC 4025	***NOR***	**3**	**3**
MMC 4030	***XOR***	**2**	**4**
MMC 4068	***NAND***	**8**	**1**
MMC 4069	***NOT***	**1**	**6**
MMC 4071	***OR***	**2**	**4**
MMC 4072	***OR***	**4**	**2**
MMC 4073	***AND***	**3**	**3**
MMC 4075	***OR***	**3**	**3**
MMC 4078	***NOR***	**8**	**1**
MMC 4081	***AND***	**2**	**4**
MMC 4082	***AND***	**4**	**2**

În familia de circuite logice CMOS poarta fundamentală este **INVERSORUL** (poarta **NU**).

Inversorul CMOS este prezentat în **figura 4.2.2** şi se compune din doi tranzistori MOS complementari, unul cu canal indus de tip *p*, **pMOS** şi altul cu canal indus de tip *n*, **nMOS** conectaţi în serie, cu grilele (G) şi drenele (D) conectate împreună .

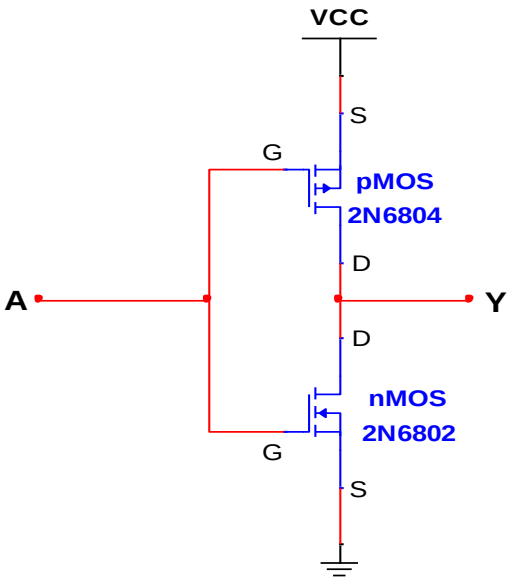


Figura 4.2.2 Inversorul CMOS

FUNCŢIONARE.

Sursa tranzistorului **pMOS** este conectată la **+V_{CC}** iar sursa tranzistorului **nMOS** este conectată la masa montajului (**-**). Grilele celor doi tranzistori reprezintă **intrarea (A)** iar drenele reprezintă **ieşirea (Y)**.

În situaţia în care intrarea **A** este conectată la masă (**0 logic)**, tensiunea pe grila tranzistorului nMOS este sub tensiunea de prag necesară deschiderii tranzistorului situaţie în care tranzistorul **nMOS** este **blocat**. În acelaşi timp tensiunea pe grila tranzistorului pMOS este (în valoare absolută) peste tensiunea de prag situaţie în care tranzistorul **pMOS** este în **conducţie**. Dacă tranzistorul pMOS este în conducţie se comportă ca un întrerupător închis iar la ieşirea **Y** a circuitului va fi +V_{cc} (**1 logic**).

În situaţia în care intrarea **A** este conectată la +V_{CC} (**1 logic)**, tensiunea pe grila tranzistorului pMOS este sub tensiunea de prag necesară deschiderii tranzistorului situaţie în care tranzistorul **pMOS** este **blocat**. În acelaşi timp tensiunea pe grila tranzistorului nMOS este (în valoare absolută) peste tensiunea de prag situaţie în care tranzistorul **nMOS** este în **conducţie**. Dacă tranzistorul nMOS este în conducţie se comportă ca un întrerupător închis iar la ieşirea **Y** a circuitului va fi 0 V (**0 logic**).

În **figura 4.2.3** este prezentată o schemă practică de realizare a unui inversor CMOS cu tranzistori MOS.

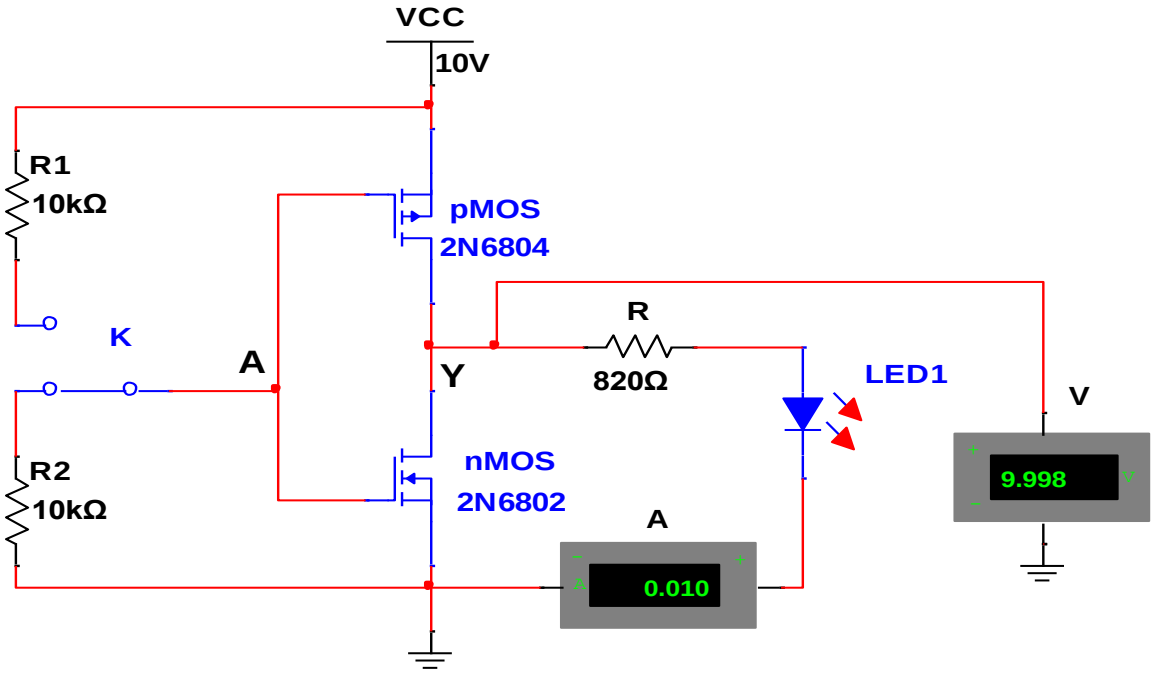


Figura 4.2.3 Poartă logică NU realizată cu tranzistori MOS

Când comutatorul **K** este conectat la masa montajului (prin intermediul rezistorului **R2**), intrarea inversorului **A** este în **0 logic** situaţie în care ieşirea inversorului **Y** este în **1 logic** iar **LED 1 luminează**.

Când comutatorul **K** este conectat la $+V_{CC}$ (prin intermediul rezistorului **R2**), intrarea inversorului **A** este în **1 logic** situaţie în care ieşirea inversorului **Y** este în **0 logic** iar **LED 1 nu luminează**.

REZUMATUL CAPITOLULUI

- Pentru a înţelege funcţionarea porţilor logice elementare cu se ţine seama de următoarele convenţii:
  - "0" logic sau nivel jos – L este echivalent cu 0 volţi;
  - "1" logic sau nivel sus – H este echivalent cu +V volţi (tensiunea maximă a sursei de alimentare, care poate fi +5V sau +15V în funcţie de componentele utilizate);
- Circuite logice elementare se construiesc:
  - Cu componente discrete (diode, tranzistori);
  - În tehnologie integrată:
    - Circuite integrate bipolare – **TTL** (au în componenţă tranzistori bipolari);
    - Circuite integrate monopolare – **MOS** care au în componenţă tranzistori cu efect de câmp și se împart în 3 familii:
      - ***Familia PMOS*** – utilizează numai tranzistoare MOS cu canal de tip P. Aceste circuite au procesul de fabricaţie simplu dar viteza de comutaţie mică;
      - ***Familia NMOS*** – utilizează numai tranzistoare MOS cu canal de tip N. Aceste circuite au procesul de fabricaţie mai complicat dar viteza de comutaţie este mare;
      - ***Familia CMOS*** – utilizează tranzistoare MOS complementare unele cu canal de tip P şi altele cu canal de tip N. Aceste circuite au o viteză de comutaţie medie şi un consum redus de energie. Circuitele integrate CMOS sunt la ora actuală cele mai utilizate circuite logice integrate monopolare.

- Reguli de utilizare ale circuitelor logice din familia TTL:
  - Nici o intrare a unui circuit logic TTL nu se lasă flotantă (neconectată);
  - Ieşirile circuitelor logice nu se conectează niciodată direct la **+V_{CC}** sau **masă**;

- Circuitele integrate CMOS au următoarele particularități:
  - Gamă mare pentru tensiunea de alimentare: **3,5 V ... 15 V**;
  - Putere de consum mică;
  - Viteză de lucru bună;
  - Imunitate la zgomot foarte bună : **45%**.

4.3. LUCRĂRI DE LABORATOR

LUCRARE DE LABORATOR 1

PORȚI LOGICE ELEMENTARE CU DIODE.

- **OBIECTIVE:**
  - Realizarea circuitelor porților logice (SAU, ȘI) cu simulatorul;
  - Realizarea practică a circuitelor porților logice;
  - Realizarea tabelelor de adevăr în funcție de poziția comutatoarelor și indicațiile LED-urilor;

- **RESURSE:**
  - Calculatoare cu soft de simulare a circuitelor electronice;
  - Proiector multimedia;
  - Sursă de tensiune continuă reglabilă;
  - Pistoale de lipit;
  - Accesorii pentru lipit, conductoare;
  - Plăcuțe de lucru;
  - Diode de comutație, rezistoare, comutatoare, LED-uri.

- **DESFĂȘURAREA LUCRĂRII:**

1. Realizează cu simulatorul schemele electronice a porților logice din figurile de mai jos:

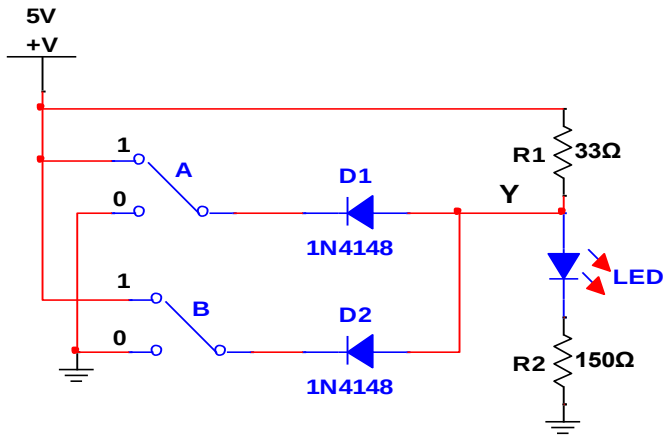


TABELA DE ADEVĂR

A	B	Y=…….
0	0	
0	1	
1	0	
1	1	

Figura 4.3.1 Poarta logică "ŞI" cu diode

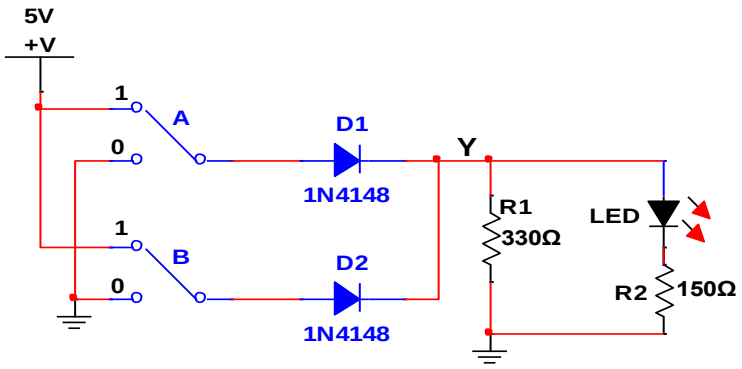


TABELA DE ADEVĂR

A	B	Y=……..
0	0	
0	1	
1	0	
1	1	

Figura 4.3.2 Poarta logică "SAU" cu diode

2. Completează tabela de adevăr pentru fiecare poartă în funcție de pozițiile comutatoarelor **A** și **B** și indicația LED-ului;

3. Realizează pe o placă de probă (pe rând) montajele din figurile de mai sus;

4. Alimentează cu tensiune montajul și se verifică funcționarea corectă a acestuia;

5. Măsoară tensiunile în anodul (+) fiecărei diode, în punctul **Y** și pe LED pentru fiecare poziție a comutatorului corespunzător diodei și notează valorile în tabelele de mai jos:

Tabelul porții logice "ȘI"

U_A[V]	U_B[V]	U_Y[V]	U_{LED}[V]	Stare LED

Tabelul porții logice "SAU"

U_A[V]	U_B[V]	U_Y[V]	U_{LED}[V]	Stare LED

LUCRARE DE LABORATOR 2

PORȚI LOGICE ELEMENTARE CU TRANZISTOARE BIPOLARE.

- **OBIECTIVE:**
  - Realizarea circuitelor porților logice (NU, SAU, ȘI) cu simulatorul;
  - Realizarea practică a circuitelor porților logice;
  - Realizarea tabelelor de adevăr în funcție de poziția comutatoarelor și indicațiile LED-urilor;

- **RESURSE:**
  - Calculatoare cu soft de simulare a circuitelor electronice;
  - Proiector multimedia;
  - Sursă de tensiune continuă reglabilă;
  - Pistoale de lipit;
  - Accesorii pentru lipit, conductoare;
  - Plăcuțe de lucru;
  - Tranzistoare bipolare, rezistoare, comutatoare, LED-uri.

- **DESFĂȘURAREA LUCRĂRII:**

1. Realizează cu simulatorul schemele electronice a porților logice din figurile de mai jos:

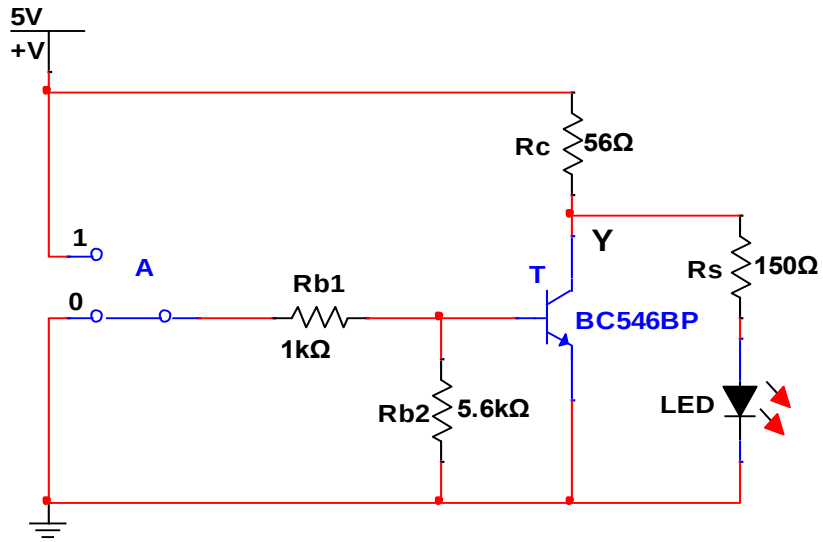


TABELA DE ADEVĂR

A	Y=…….
0	
1	

Figura 4.3.3 Poarta logică "NU" cu tranzistoare bipolare

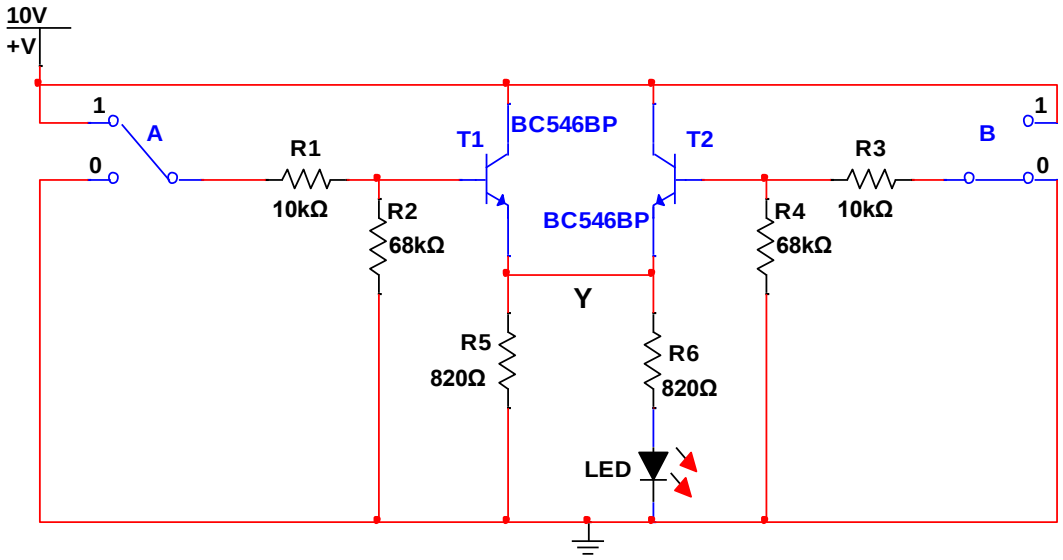


Figura 4.3.4 Poarta logică "SAU" cu tranzistoare

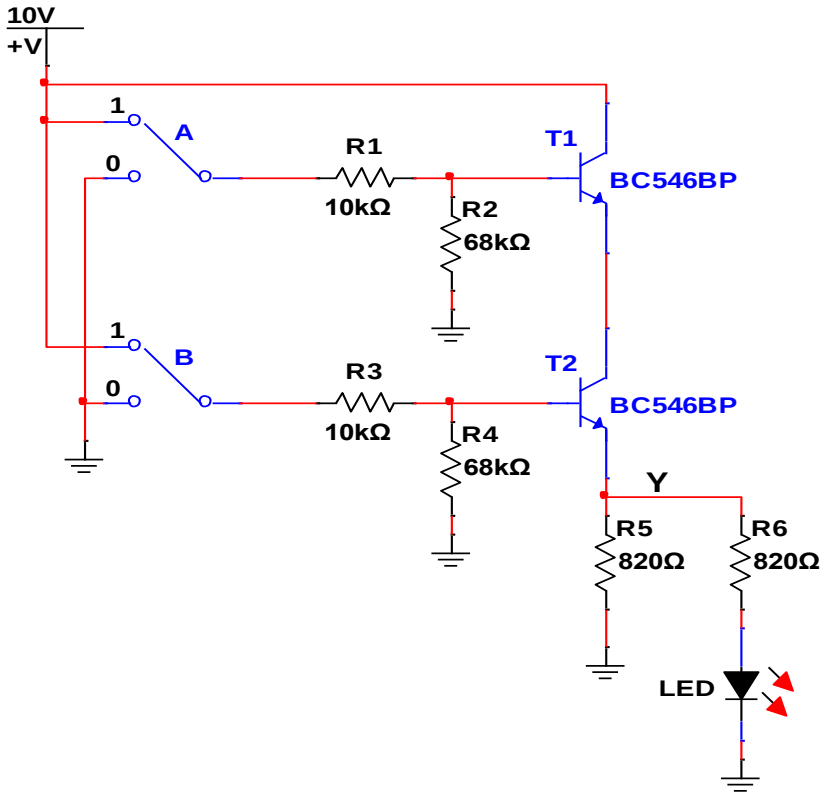


Figura 4.3.5 Poarta logică "ŞI" cu tranzistoare

2. Completează tabela de adevăr pentru poarta logică **"NU"** în funcție de poziția comutatorului **A** și indicația LED-ului;

3. Realizează pe o placă de probă (pe rând) montajele din figurile de mai sus;

4. Alimentează cu tensiune montajul și se verifică funcționarea corectă a acestuia.

LUCRARE DE LABORATOR 3

PORȚI LOGICE ELEMENTARE CU DIODE ȘI TRANZISTOARE BIPOLARE.

- **OBIECTIVE:**
  - Realizarea circuitelor porților logice (SAU-NU, ȘI-NU) cu simulatorul;
  - Realizarea practică a circuitelor porților logice;
  - Realizarea tabelelor de adevăr în funcție de poziția comutatoarelor și indicațiile LED-urilor;

- **RESURSE:**
  - Calculatoare cu soft de simulare a circuitelor electronice;
  - Proiector multimedia;
  - Sursă de tensiune continuă reglabilă;
  - Pistoale de lipit;
  - Accesorii pentru lipit, conductoare;
  - Plăcuțe de lucru;
  - Tranzistoare bipolare, diode de comutație, rezistoare, comutatoare, LED-uri.

- **DESFĂȘURAREA LUCRĂRII:**

1. Realizează cu simulatorul schemele electronice din figurile de mai jos:

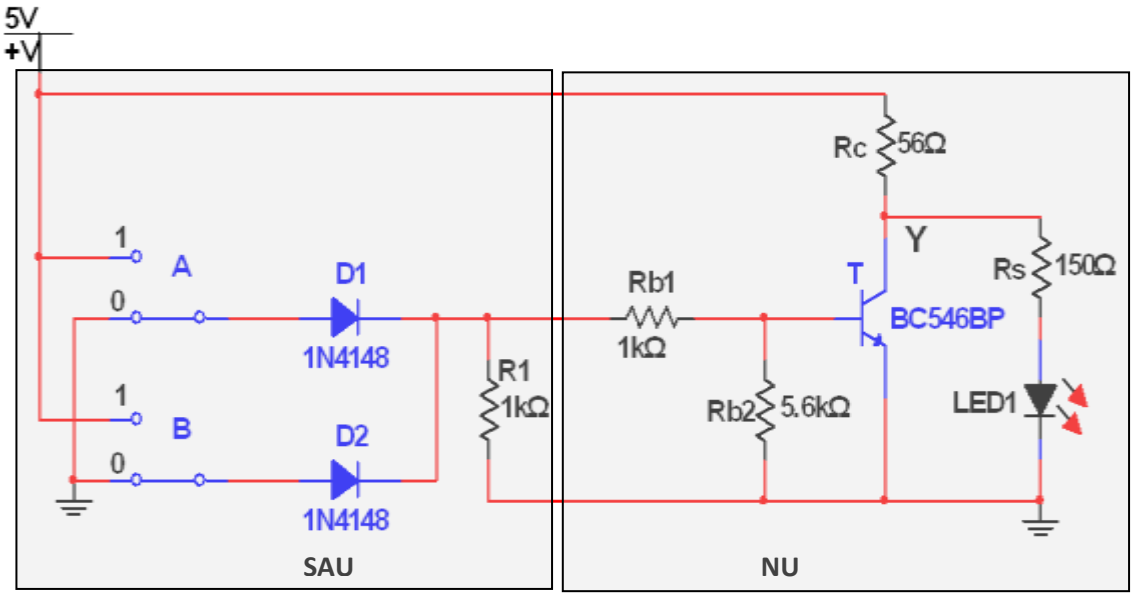


Figura 4.3.6 Poarta logică "SAU-NU" cu diode şi tranzistoare

TABELA DE ADEVĂR

A	B	Y=
0	0	
0	1	
1	0	
1	1	

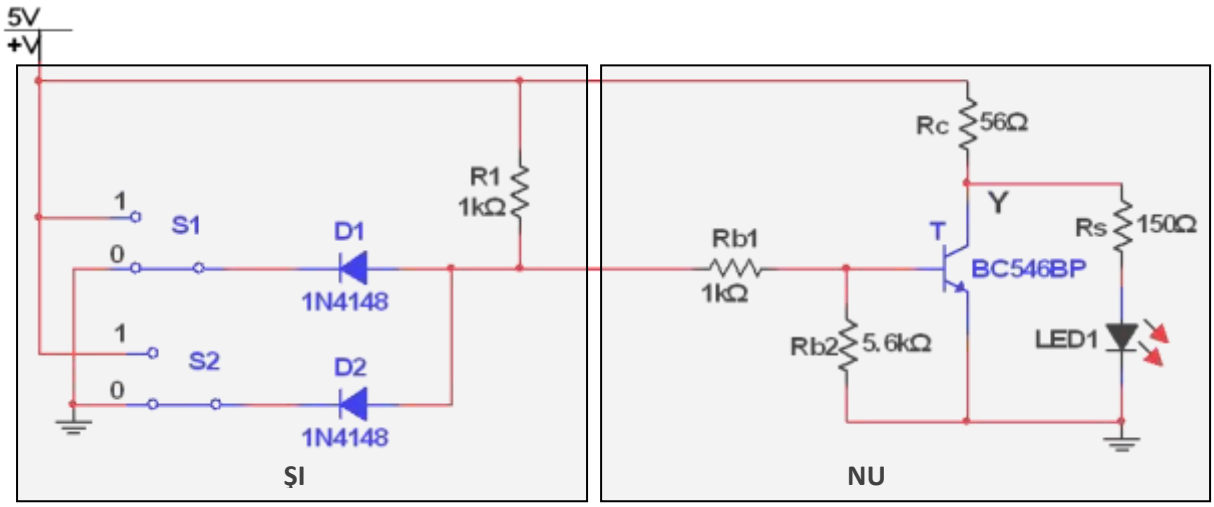


Figura 4.3.7 Poarta logică "ŞI-NU" cu diode şi tranzistoare

TABELA DE ADEVĂR

A	B	Y =
0	0	
0	1	
1	0	
1	1	

2. Completează tabela de adevăr pentru fiecare poartă în funcție de pozițiile comutatoarelor **A** și **B** și indicația LED-ului;

3. Realizează pe o placă de probă (pe rând) montajele din figurile de mai sus;

4. Alimentează cu tensiune montajul și se verifică funcționarea corectă a acestuia.

CAPITOLUL 5. CIRCUITE LOGICE COMBINAŢIONALE

5.1. GENERALITĂŢI

Circuitele logice combinaţionale (CLC) – sunt circuite alcătuite din porţi logice de bază a căror operare poate fi descrisă cu ajutorul algebrei Booleene.

Aceste circuite se caracterizează prin faptul că în fiecare moment starea logică a ieşirii depinde de modul în care se combină nivelurile logice ale intrărilor în acel moment.

CLC nu au capacitatea de memorare a informaţiei (sunt independente de propriile stări anterioare).

Schema bloc a unui CLC cu n intrări şi m ieşiri este dată în **figura 5.1.1**

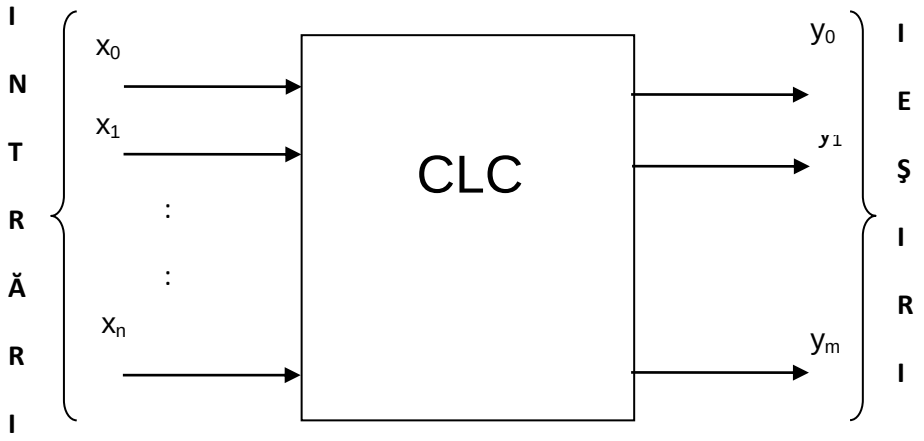


Figura 5.1.1 Schema bloc a unui circuit logic combinaţional

Funcţiile care descriu aceste tipuri de circuite reprezintă funcţii binare prezentate în capitolul 2 şi pot fi scrise sub forma relaţiilor (**5.1.1**)

$$y_0 = f_0(x_0, x_1, \dots\dots\dots x_n)$$
$$y_1 = f_1(x_0, x_1, \dots\dots\dots x_n)$$
$$\dots\dots\dots\dots\dots\dots\dots\dots\dots \quad \textbf{(5.1.1)}$$
$$y_m = f_m(x_0, x_1, \dots\dots\dots x_n)$$

Problema esenţială care trebuie rezolvată cu ajutorul CLC este *implementarea unor funcţii logice cu ajutorul unui număr minim de porţi logice*. Aceste aspecte sunt prezentate în **subcapitolul 3.2**.

În cele ce urmează vor fi studiate numai CLC realizate cu porţi logice care primesc la intrare semnale numerice în logică pozitivă sau logică negativă şi furnizează la ieşire semnale numerice într-o anumită logică.

În logică pozitivă : nivel ridicat de tensiune H ⇔ "1" ⇔ „ADEVĂRAT"
nivel coborât de tensiune L ⇔ „0" ⇔ „ FALS"

În logică negativă : nivel ridicat de tensiune H ⇔ "0" ⇔ „FALS"
nivel coborât de tensiune L ⇔ „1" ⇔ „ ADEVĂRAT"

Porţile logice sunt circuitele logice de bază din structura circuitelor logice combinaţionale. Porta logică reprezintă o implementare fizică a unei funcţii logice. Porţile logice sunt prezentate în **subcapitolul 3.1**.

Pentru prelucrarea datelor în sistemele digitale şi pentru citirea şi afişarea rezultatelor prelucrării, este necesară parcurgerea următoarelor etape:

- ***Codarea şi decodarea*** – transformarea datelor dintr-un cod în altul;
- ***Multiplexarea*** – transmiterea către o ieşire a unei singure informaţii dintr-un grup de informaţii;
- ***Demultiplexarea*** – introducerea succesivă a datelor la diferite adrese posibile.

Pentru efectuarea operaţiilor aritmetice se utilizează circuite logice combinaţionale special construite pentru acest scop numite *circuite numerice* (comparatoare, sumatoare, convertoare de cod).

5.2. CODIFICATOARE

Codificatoarele (CD) – sunt circuite logice combinaţionale cu n intrări şi m ieşiri care furnizează la ieşire un cod de m biţi atunci când numai una din cele n intrări este activă. De regulă intrările codificatoarelor sunt active în 0, deoarece prin activarea unei intrări aceasta este pusă la masa montajului, deci capătă valoarea 0 logic.

Circuitele de codificare primesc la intrare semnale codificate într-un cod diferit de cel binar şi furnizează la ieşire semnale în cod binar sau echivalentul acestuia.

Numărul de biţi ai codului de ieşire (m) este întotdeauna mai mic decât numărul de biţi ai codului de intrare (n) $\boldsymbol{m \geq \log_2 n}$

La codificatorul cu n intrări care are la ieşire un cod de m biţi, număr de cuvinte furnizate la ieşire este $\boldsymbol{n = 2^{m-1}}$ care este egal cu numărul intrărilor acestuia.

Cel mai utilizat codificator este codificatorul zecimal-BCD la intrarea căruia se aplică date în sistemul zecimal iar la ieşire apar date în codul BCD.

Codificatorul SN74148 – este un codificator zecimal-BCD de trei biţi (fig. 5.2.1).

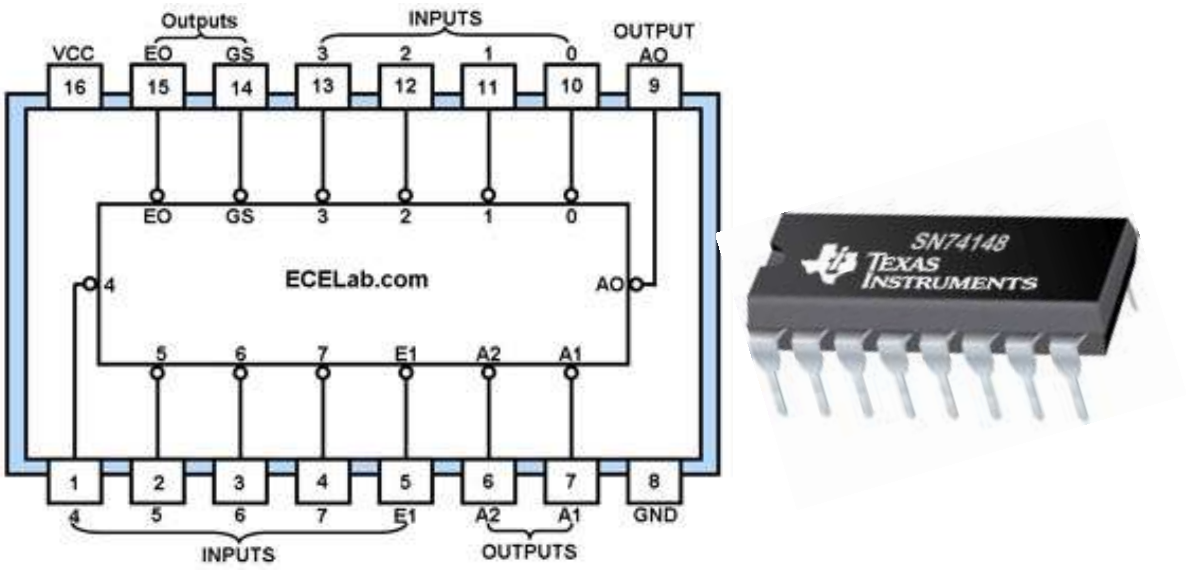


Figura 5.2.1 Codificatorul integrat SN74148

Codificatorul SN74148 este prevăzut cu:

- 8 intrări de date ($\mathbf{I_0 - I_7}$) active în **0**;
- O intrare **EI** (Enable In) pentru validarea circuitului care este activă în **0**;
- 3 ieşiri de date ($\mathbf{A_0, A_1, A_2}$) active în **0**;
- O ieşire suplimentară pentru conectarea în cascadă a mai multor codificatoare **EI** (Enable Out) activă în **0**;
- O ieşire **GS** care devine activă (în 0 logic) când cel puţin una dintre intrările codificatorului este activă.

Tabelul de adevăr al codificatorului este prezentat mai jos

Tabelul 5.2.1 – Tabelul de adevăr al codificatorului SN74148

INTRĂRI									IEŞIRI				
EI	0	1	2	3	4	5	6	7	2^2 A2	2^2 A1	2^0 A0	GS	EO
1	X	X	X	X	X	X	X	X	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	X	X	X	X	X	X	X	0	0	0	0	0	1
0	X	X	X	X	X	X	0	1	0	0	1	0	1
0	X	X	X	X	X	0	1	1	0	1	0	0	1
0	X	X	X	X	0	1	1	1	0	1	1	0	1
0	X	X	X	0	1	1	1	1	1	0	0	0	1
0	X	X	0	1	1	1	1	1	1	0	1	0	1
0	X	0	1	1	1	1	1	1	1	1	0	0	1
0	0	1	1	1	1	1	1	1	1	1	1	0	1

Codificatorul SN74147 – este un codificator zecimal-BCD de 4 biţi (figura 5.2.2).

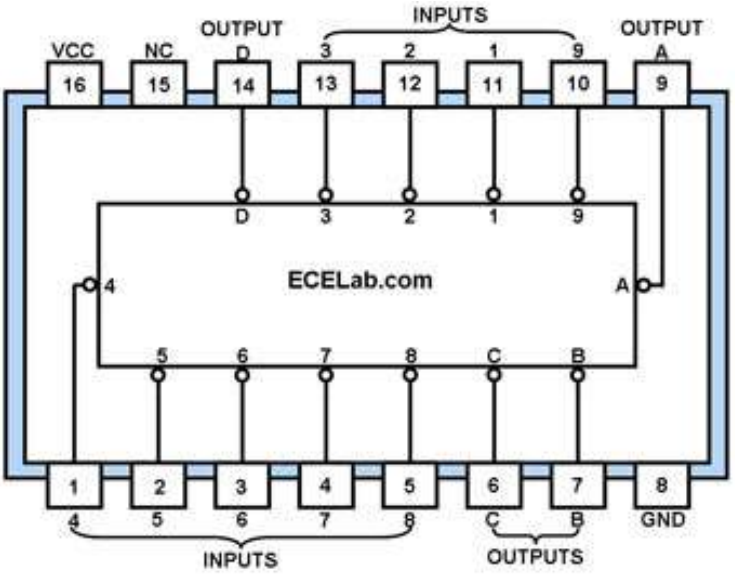


Figura 5.2.2 Codificatorul integrat SN74147

Codificatorul este prevăzut cu: **9 intrări** numerotate de la **1** la **9** active în **0**

4 ieşiri notate cu **A, B, C, D** active în **0**

Acest codificator nu utilizează 10 intrări deoarece se consideră că la intrare este cifra 0 când toate intrările sunt în 1 logic (vezi prima linie din tabel)

Tabelul de adevăr al codificatorului este prezentat mai jos

Tabelul 5.2.2- Tabelul de adevăr al codificatorului SN74147

INTRĂRI									IEŞIRI			
1	2	3	4	5	6	7	8	9	2^3 D	2^2 C	2^1 B	2^0 A
1	1	1	1	1	1	1	1	1	1	1	1	1
X	X	X	X	X	X	X	X	0	0	1	1	0
X	X	X	X	X	X	X	0	1	0	1	1	1
X	X	X	X	X	X	0	1	1	1	0	0	0
X	X	X	X	X	0	1	1	1	1	0	0	1
X	X	X	X	0	1	1	1	1	1	0	1	0
X	X	X	0	1	1	1	1	1	1	0	1	1
X	X	0	1	1	1	1	1	1	1	1	0	0
X	0	1	1	1	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	1	0

5.3. DECODIFICATOARE

Decodificatoarele (DCD) – sunt circuite logice combinaţionale cu n intrări şi m ieşiri ($m=2^n$) care activează o singură ieşire corespunzătoare codului aplicat la intrare.

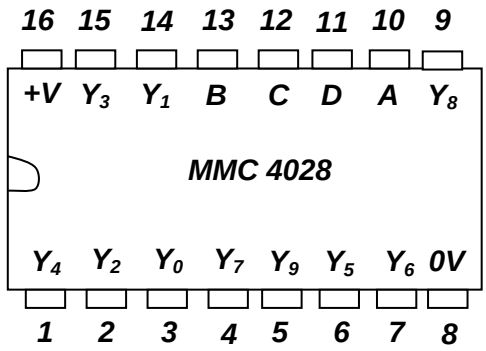
Circuitele de codificare primesc la intrare semnale logice în cod binar sau echivalentul acestuia şi furnizează la ieşire semnale în cod zecimal sau echivalentul acestuia.

Cele mai utilizate decodificatoare sunt: **decodificatorul BCD - zecimal** şi **decodificatorul BCD - 7 segmente**.

1. **Decodificatorul BCD - zecimal** – primeşte la intrare datele în cod BCD şi activează o singură ieşire corespunzătoare codului de intrare.

Acest decodificator este prevăzut cu ***4 intrări*** notate cu **A, B, C, D** (corespunzătoare celor 4 biţi din codul BCD) şi cu ***10 ieşiri*** notate cu **Y0, Y1, Y2,.....Y9** (corespunzătoare celor 10 cifre din codul zecimal). În funcţie de tipul decodificatorului ieşirile sunt **active în 0 logic** sau în **1 logic**.

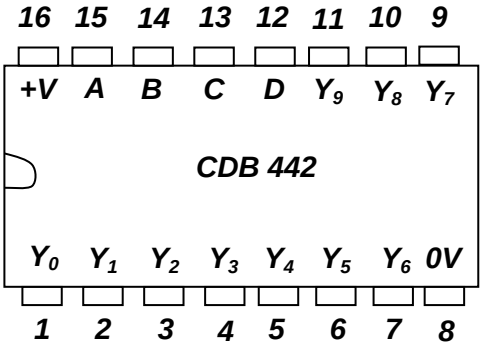
Decodificatorul MMC 4028 are ieşirile active în 1 logic.



Tabelul de adevăr MMC 4028

2^3	2^2	2^1	2^0	IEȘIRI									
D	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	Y9
0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	0
0	0	1	1	0	0	0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0	0	1	0	0	0	0
0	1	1	0	0	0	0	0	0	0	1	0	0	0
0	1	1	1	0	0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	0	0	0	0	1

Decodificatorul CDB 442 are ieşirile active în 0 logic.



Tabelul de adevăr CDB 442

2^3	2^2	2^1	2^0	IEȘIRI									
D	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	Y9
0	0	0	0	**0**	1	1	1	1	1	1	1	1	1
0	0	0	1	1	**0**	1	1	1	1	1	1	1	1
0	0	1	0	1	1	**0**	1	1	1	1	1	1	1
0	0	1	1	1	1	1	**0**	1	1	1	1	1	1
0	1	0	0	1	1	1	1	**0**	1	1	1	1	1
0	1	0	1	1	1	1	1	1	**0**	1	1	1	1
0	1	1	0	1	1	1	1	1	1	**0**	1	1	1
0	1	1	1	1	1	1	1	1	1	1	**0**	1	1
1	0	0	0	1	1	1	1	1	1	1	1	**0**	1
1	0	0	1	1	1	1	1	1	1	1	1	1	**0**

La intrările A, B, C, D se aplică codul binar corespunzător cifrelor de la 0 la 15 (16 combinaţii. Doar 10 din cele 16 combinaţii sunt acceptate, şi anume cele corespunzătoare cifrelor 0 – 9. Celelalte combinaţii reprezintă stări interzise.

Exemplu: dacă **A=0, B=1, C=1, D=0** la ieşirea **Y6** apare nivel logic **0** (0,2...0,4 V), restul ieşirilor au nivel logic 1 (circa 3,4 V).

Acelaşi lucru se întâmplă dacă codul corespunde oricărei cifre de la 0 la 9.

Pentru combinaţiile logice corespunzătoare numerelor de la 10 la 15, ieşirile rămân în starea logică 1.

Aceste decodificatoare se utilizează în:

- Circuite de numărare
- Generatoare de funcţii
- Circuite de comandă la distanţă
- Circuite de selecţie

În **figura 5.3.1** este prezentată schema unei aplicaţii cu decodificatorul MMC 4028.

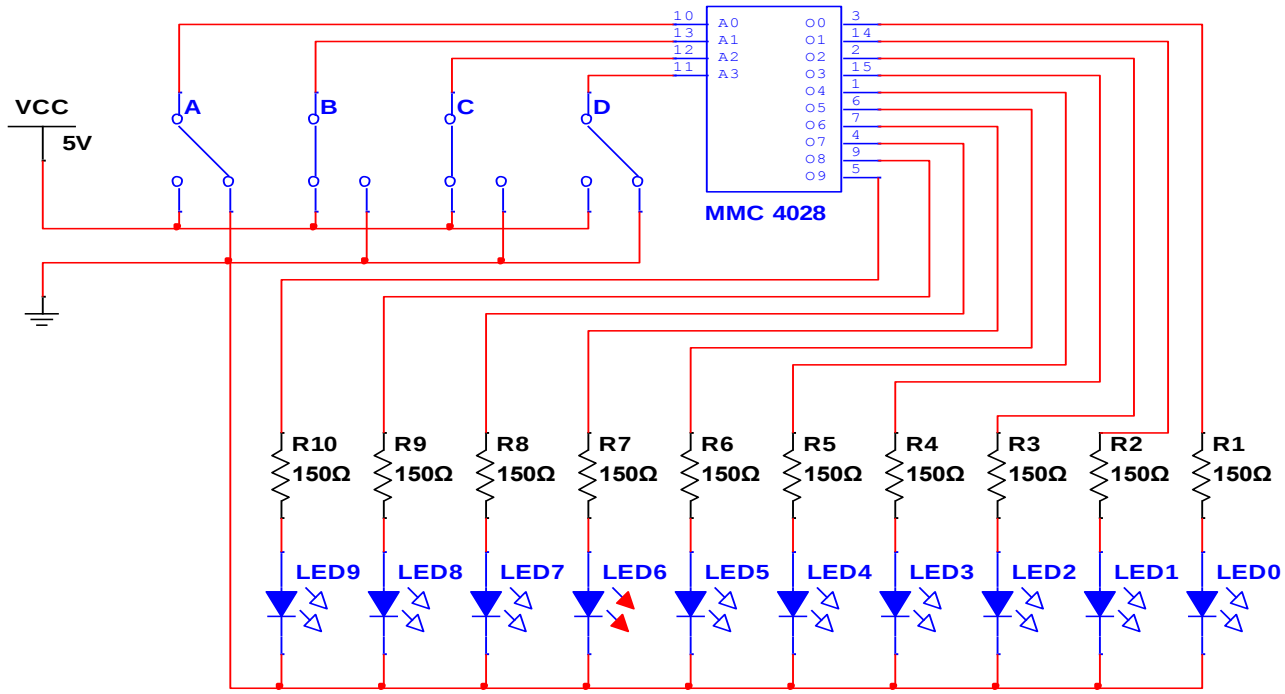


Figura 5.3.1 Aplicaţie cu decodificatorul MMC 4028

Intrările decodificatorului (A0, A1, A2, A3) sunt conectate la comutatoarele A, B, C, D. Aceste comutatoare pot fi poziţionate în 0 logic (0 V) respectiv în 1 logic (+5V).

Ieşirile decodificatorului (Q0, Q1, Q2,.....Q9) sunt conectate prin intermediul rezistoarelor R1, R2, R3,.....R10 la LED-urile LED0, LED1, LED2,.....LED9.

În funcţie de poziţia comutatoarelor A, B, C, D la intrarea decodificatorului se aplică un cod binar corespunzător unei anumite cifre de la 0 la 9 şi luminează LED-ul corespunzător cifrei respective.

În exemplul din figura 5.3.1 comutatoarele B şi C sunt în 1 logic, iar comutatoarele A şi D sunt în 0 logic, combinaţie ce corespunde cifrei 6, situaţie în care LED6 luminează.

Pentru codurile de intrare corespunzătoare numerelor de la 10 la 15 toate LED-urile vor fi stinse deoarece aceste combinaţii reprezintă stări interzise.

2. **Decodificatorul BCD – 7 segmente** – comandă dispozitivele de afişare numerică realizate din 7 segmente luminoase (cu led-uri, cristale lichide). Decodificatorul primeşte la intrare datele în cod BCD şi activează mai multe ieşiri corespunzătoare codului de intrare.

Prin polarizarea directă a segmentelor , în diverse combinaţii, se poate forma orice cifră a sistemului zecimal. Afişajele 7 segmente se construiesc în două variante: cu anodul comun şi cu catodul comun şi sunt prevăzute cu 10 terminale (**figura 5.3.2**)

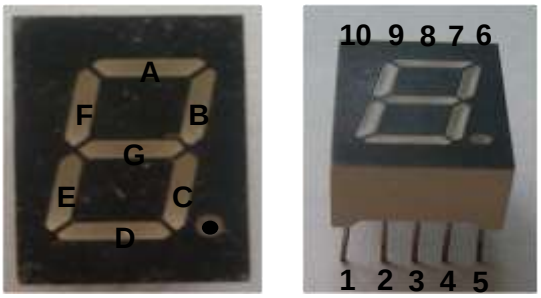
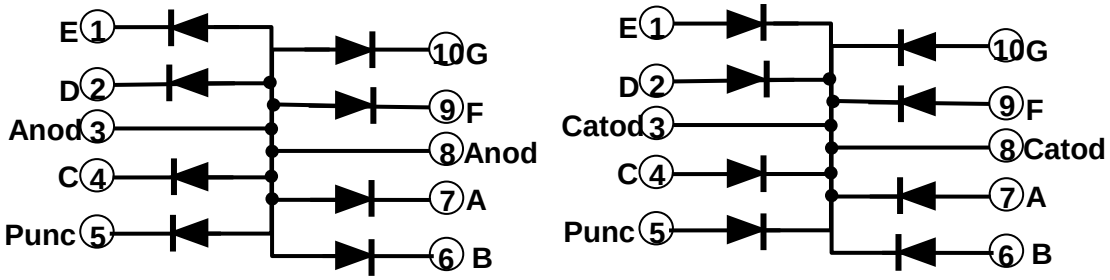


Figura 5.3.2 Afişaj 7 segmente - aranjarea segmentelor-numerotarea terminalelor

KW1 – 501 AS
KW1 – 521 AGA

KW1 – 501 CRB
KW1 – 521 CS



(a) Cu Anod comun **(b) Cu Catod comun**

Figura 5.3.3 Structură afişaj 7 segmente

Pentru activarea unui segment acesta se polarizează direct.

La afişajele cu Anod comun, anodul se conectează spre polul pozitiv al sursei (+) iar segmentul care se activează se conectează spre polul negativ al sursei (-).

La afişajele cu Catod comun, catodul se conectează spre polul negativ al sursei (-) iar segmentul care se activează se conectează spre polul pozitiv al sursei (+).

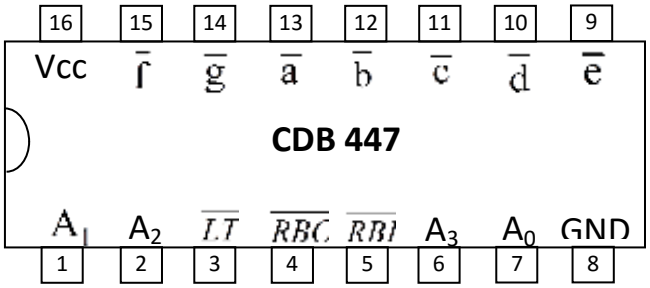
Un segment are următorii parametrii electrici:

Tensiunea directă de polarizare Vf = 1,9 V – 2,2 V (în funcţie de culoarea segmentelor)

Curentul direct If = 10 mA – 20 mA.

Decodificatorul BCD – 7 segmente este prevăzut cu ***4 intrări*** notate cu ***A, B, C, D*** (corespunzătoare celor 4 biţi din codul BCD) şi cu ***7 ieşiri*** notate cu ***a, b, c, d, e, f*** (corespunzătoare celor 7 segmente ale afişajului).

Pentru afişajele cu anodul comun se pot utiliza circuitele integrate: CDB 446; CDB 447; SN74LS47 ; SN7447. În funcţie de combinaţia intrărilor se activează una sau mai multe ieşiri. La aceste decodificatoare ieşirile sunt active în „0” logic.



LT - asigură testarea segmentelor
LT= ”1” – toate segmentele aprinse
RBO- pentru funcţiile de ieşire 0-15
RBI- pentru afişarea lui 0

Figura 5.3.4 Decodificatorului CDB 447

Pentru afişajele cu catodul comun se pot utiliza circuitele integrate: CDB448 ; MMC4248; SN74LS48 ; SN7448 ; HCF 4511 BE. În funcţie de combinaţia intrărilor se activează una sau mai multe ieşiri. La aceste decodificatoare ieşirile sunt active în „1” logic.

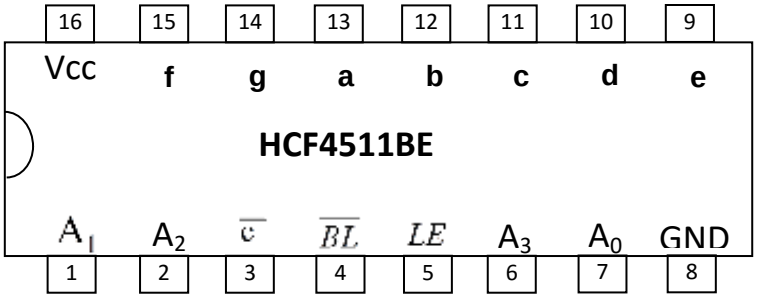


Figura 5.3.4 Decodificatorului HCF 4511 BE

În **figura 5.3.5** este prezentată schema unei aplicaţii cu decodificatorul CDB 447.

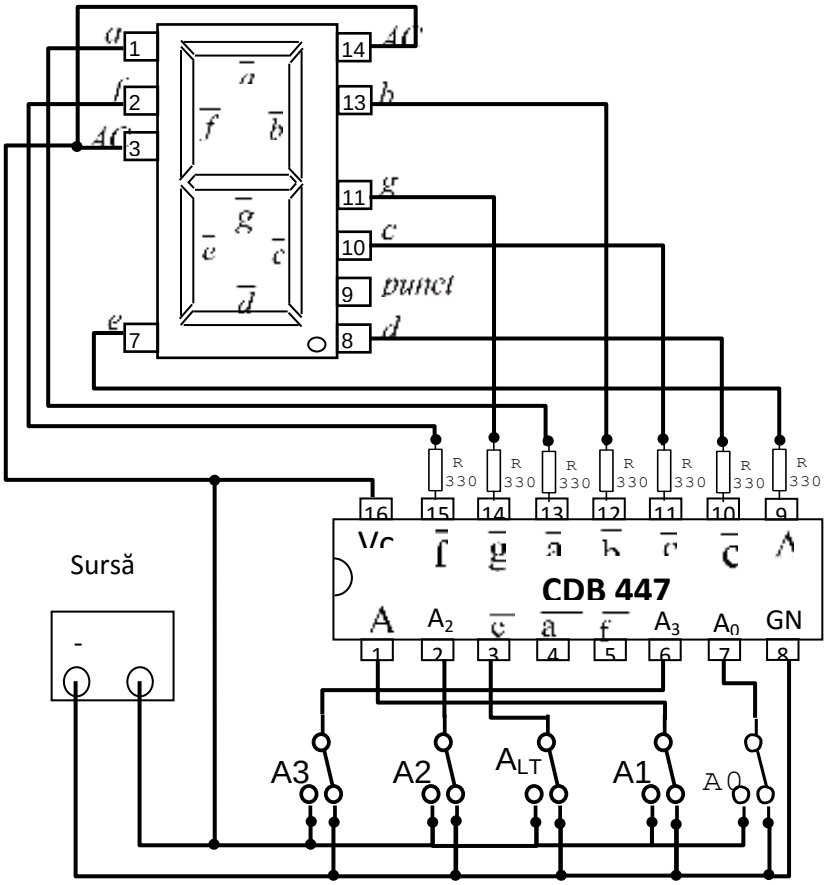


Figura 5.3.5 Comanda unui afişaj 7 segmente cu anodul comun (MDE 2102 R)

Pentru verificarea segmentelor afişajului se poziţionează comutatorul ALT pe (+) apoi se poziţionează înapoi pe (-).

Comutatoarele A0, A1, A2, A3 pot fi poziţionate în 0 logic (0 V) respectiv în 1 logic (+5V). În funcţie de combinaţiile de la intrarea decodificatorului se vor activa segmentele corespunzătoare cifrei respective (vezi tabelul de adevăr CDB 447).

Tabelul de adevăr CDB 447

D	C	B	A	cifra	a	b	c	d	e	f	g
0	0	0	0	**0**	0	0	0	0	0	0	1
0	0	0	1	**1**	1	0	0	1	1	1	1
0	0	1	0	**2**	0	0	1	0	0	1	0
0	0	1	1	**3**	0	0	0	0	1	1	0
0	1	0	0	**4**	1	0	0	1	1	0	0
0	1	0	1	**5**	0	1	0	0	1	0	0
0	1	1	0	**6**	1	1	0	0	0	0	0
0	1	1	1	**7**	0	0	0	1	1	1	1
1	0	0	0	**8**	0	0	0	0	0	0	0
1	0	0	1	**9**	0	0	0	1	1	0	0

În **figura 5.3.6** este prezentată schema unei aplicaţii cu decodificatorul HCF 4511.

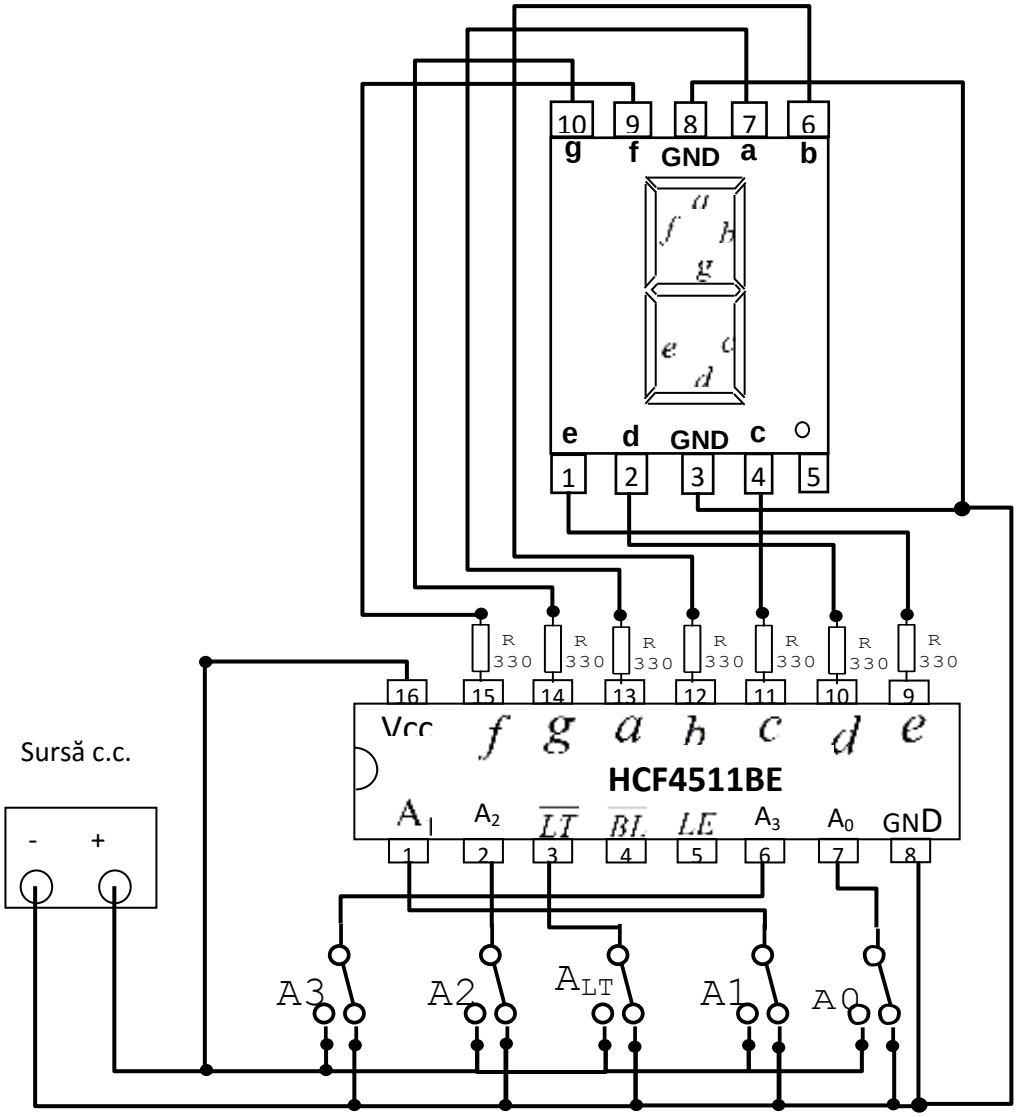


Figura 5.3.6 Comanda unui afişaj 7 segmente cu catodul comun (KW1-501CRB)

În funcţie de combinaţiile de la intrarea decodificatorului se vor activa segmentele corespunzătoare cifrei respective (vezi tabelul de adevăr HCF 4511).

Tabelul de adevăr HCF 4511

D	C	B	A	**cifra**	a	b	c	d	e	f	g
0	0	0	0	**0**	**1**	**1**	**1**	**1**	**1**	**1**	0
0	0	0	1	**1**	0	**1**	**1**	0	0	0	0
0	0	1	0	**2**	1	1	0	**1**	**1**	0	**1**
0	0	1	1	**3**	**1**	**1**	**1**	**1**	0	0	**1**
0	1	0	0	**4**	0	**1**	**1**	0	0	**1**	**1**
0	1	0	1	**5**	**1**	0	**1**	**1**	0	**1**	**1**
0	1	1	0	**6**	0	0	**1**	**1**	**1**	**1**	**1**
0	1	1	1	**7**	**1**	**1**	**1**	0	0	0	0
1	0	0	0	**8**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
1	0	0	1	**9**	**1**	**1**	**1**	0	0	**1**	**1**

5.4. MULTIPLEXOARE

Multiplexoarele (MUX) – sunt circuite logice combinaţionale cu m intrări şi o singură ieşire, care permit transferul datelor de la una din intrări spre ieşirea unică. Selecţia intrării de la care se transferă datele se face prin intermediul unui cuvânt de cod de selecţie numit adresă, cuvânt care are n biţi. Numărul de intrări m este egal cu numărul combinaţiilor logice de adresă 2^n a căror apariţie urmează să autorizeze accesul succesiv al intrărilor către ieşire ($\mathbf{m=2^n}$). Schema de principiul a unui multiplexor este prezentată în **figura 5.4.1**.

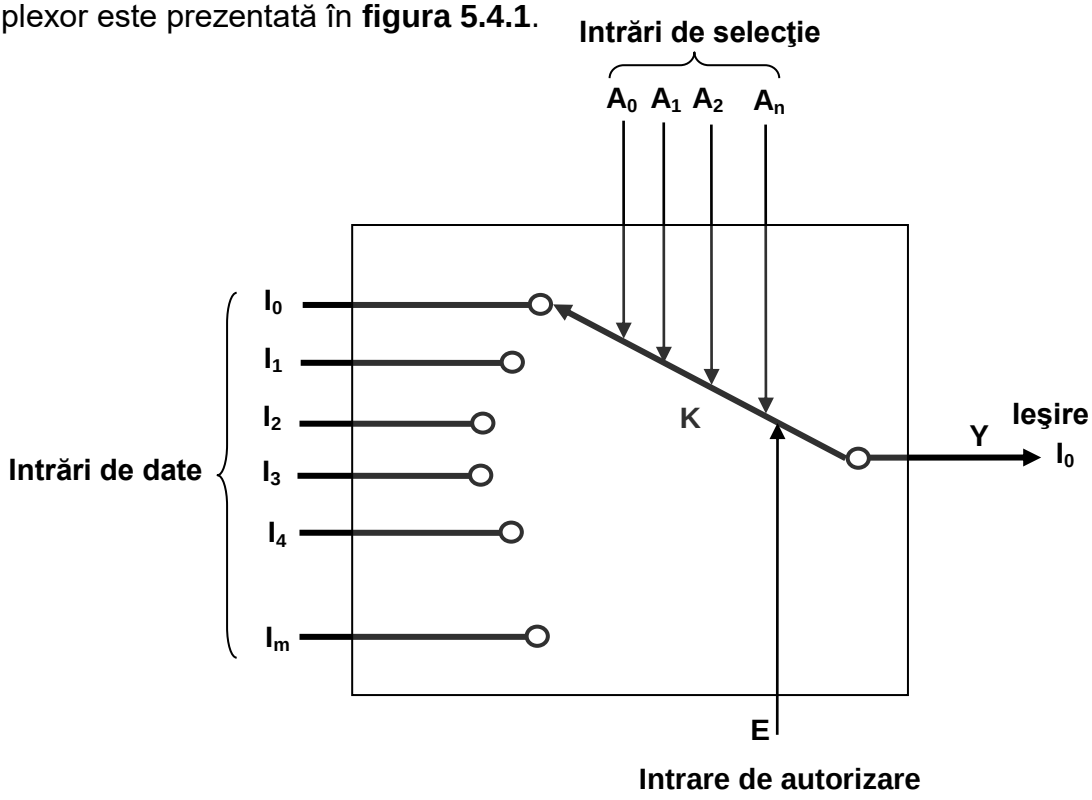


Figura 5.4.1 Schema de principiu a unui multiplexor

În funcţie de poziţia comutatorului K la ieşirea Y va fi transmis semnalul uneia din intrările de date I. Poziţia comutatorului este comandată de nivelul logic al intrărilor de selecţie (A1, A2,...An), care formează adresa unei anumite intrări de date. Multiplexorul mai este prevăzut cu o intrare de autorizare (E) care permite funcţionarea sau blocarea multiplexorului.

În practică se utilizează următoarele tipuri de multiplexoare:

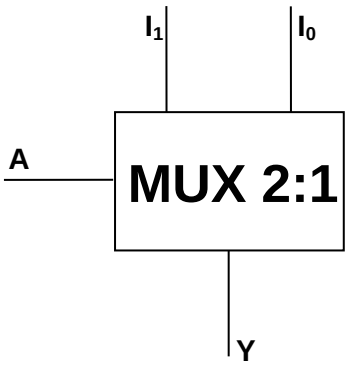
- Cu 2 intrări si o linie de adresă (SN74LS157, CDB 4157);
- Cu 4 intrări şi 2 linii de adresă (SN74LS153, CDB 4153);
- Cu 8 intrări şi 3 linii de adresă (SN74LS151, CDB 4151);
- Cu 16 intrări şi 4 linii de adresă (SN74LS150, CDB 74150).

1. MULTIPLEXOR CU 2 INTRĂRI

Acest multiplexor (**fig.5.4.2 a**) permite transferul datelor de pe intrările de date I0 şi I1 la ieşirea Y în funcţie de starea logică a intrării de selecţie A conform tabelei de adevăr din (**fig. 5.4.2 b**).

Când **A=0** pe ieşirea **Y** se transferă datele de pe intrarea **I0**

Când **A=1** pe ieşirea **Y** se transferă datele de pe intrarea **I1**



a

Intrări			Ieşire
A	I0	I1	Y
0	0	X	0
0	1	X	1
1	X	0	0
1	X	1	1

b

Figura 5.4.2 Multiplexor cu 2 intrări

Realizat cu porţi logice elementare, multiplexorul cu 2 intrări arată ca în **figura 5.4.3**

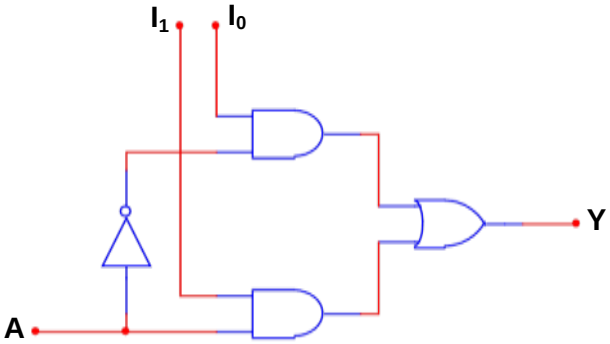
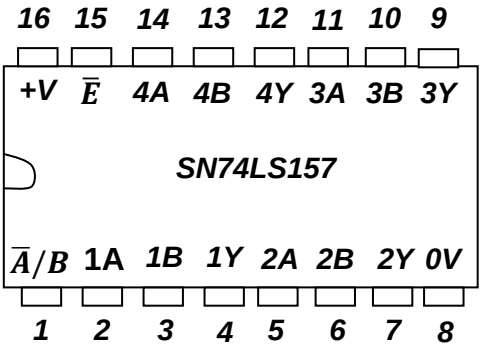


Figura 5.4.3 Multiplexorul cu 2 intrări realizat cu porţi logice

Prezentarea circuitului SN 74LS157 (4 multiplexoare cu 2 intrări)

Configuraţia terminalelor



Tabelul de adevăr

INTRĂRI				Ieşire
$\overline{E}$	$\overline{A}/B$	A	B	Y
1	X	X	X	0
0	1	1	X	1
0	1	0	X	0
0	0	X	1	1
0	0	X	0	0

Figura 5.4.4 Multiplexorul cu 2 intrări SN74SL157

2. MULTIPLEXOR CU 4 INTRĂRI

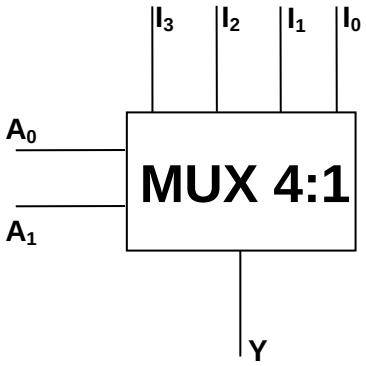
Acest multiplexor (**fig.5.4.5 a**) permite transferul datelor de pe intrările de date I0, I1, I2, I3 la ieşirea Y în funcţie de starea logică a intrărilor de selecţie A0, A1 conform tabelei de adevăr din (**fig. 5.4.5 b**).

Când A1=0, A0=0 (**0**) pe ieşirea Y se transferă datele de pe intrarea **I0**

Când **A1=0, A0=1** (**1**) pe ieşirea Y se transferă datele de pe intrarea **I1**

Când **A1=1, A0=0** (**2**) pe ieşirea Y se transferă datele de pe intrarea **I2**

Când **A1=1, A0=1** (**3**) pe ieşirea Y se transferă datele de pe intrarea **I3**



a

Intrări selecţie		Intrări date				Ieşire
A1	A0	I0	I1	I2	I3	Y
0	0	0	X	X	X	0
0	0	1	X	X	X	1
0	1	X	0	X	X	0
0	1	X	1	X	X	1
1	0	X	X	0	X	0
1	0	X	X	1	X	1
1	1	X	X	X	0	0
1	1	X	X	X	1	1

b

Figura 5.4.5 Multiplexor cu 4 intrări

Realizat cu porţi logice elementare, multiplexorul cu 4 intrări arată ca în **figura 5.4.6**

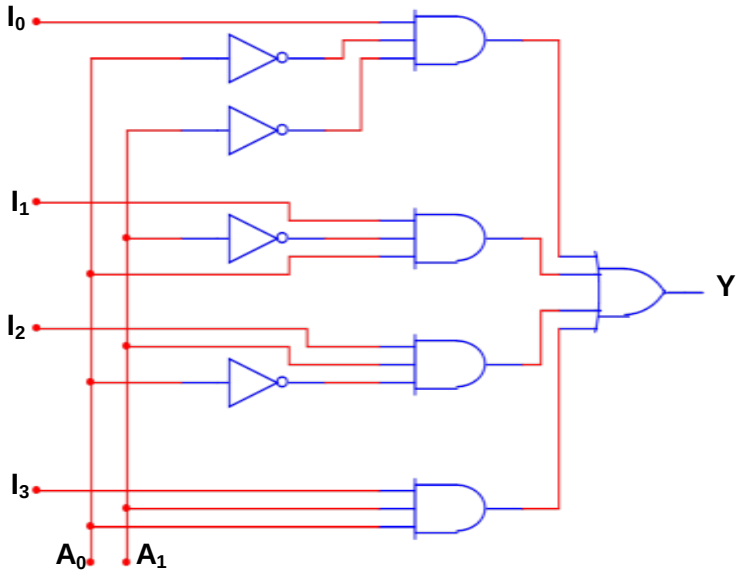
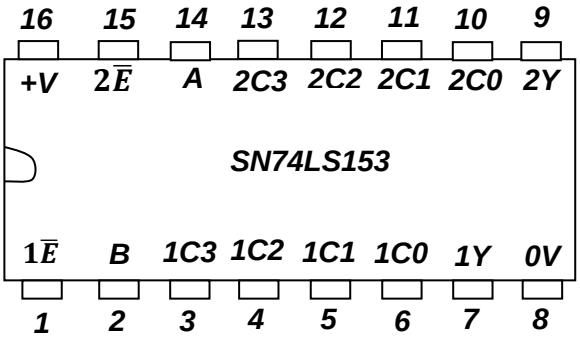


Figura 5.4.6 Multiplexorul cu 4 intrări realizat cu porţi logice

Prezentarea circuitului SN 74LS153 (2 multiplexoare cu 4 intrări)

Configuraţia terminalelor



Tabelul de adevăr

Intrări selecţie		Intrări date				Autorizare	Ieşire
B	A	C0	C1	C2	C3	$\overline{E}$	Y
X	X	X	X	X	X	1	0
0	0	0	X	X	X	0	0
0	0	1	X	X	X	0	1
0	1	X	0	X	X	0	0
0	1	X	1	X	X	0	1
1	0	X	X	0	X	0	0
1	0	X	X	1	X	0	1
1	1	X	X	X	0	0	0
1	1	X	X	X	1	0	1

Figura 5.4.7 Multiplexorul cu 4 intrări SN74SL153

3. MULTIPLEXOR CU 8 INTRĂRI

Acest multiplexor (**fig.5.4.8 a**) permite transferul datelor de pe intrările de date I0, I1, I2, I3, I4, I5, I6, I7, la ieşirea Y în funcţie de starea logică a intrărilor de selecţie A0, A1, A2 conform tabelei de adevăr din (**fig. 5.4.8 b**).

Când **A2=0, A1=0**, **A0=0** (**0**) pe ieşirea **Y** se transferă datele de pe intrarea **I0**

Când **A2=0, A1=0**, **A0=1** (**1**) pe ieşirea **Y** se transferă datele de pe intrarea **I1**

Când **A2=0, A1=1, A0=0** (**2**) pe ieşirea **Y** se transferă datele de pe intrarea **I2**

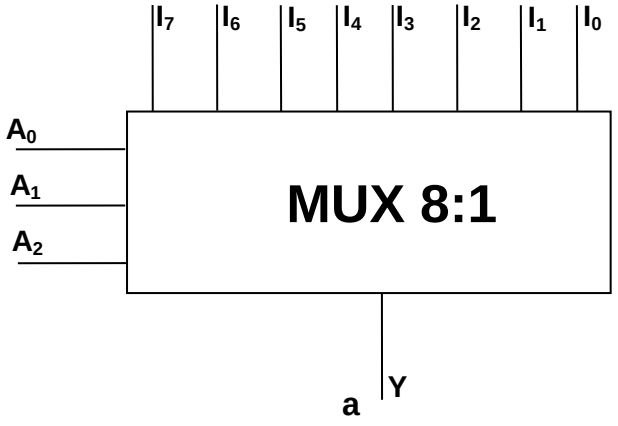
Când **A2=0, A1=1, A0=1** (**3**) pe ieşirea **Y** se transferă datele de pe intrarea **I3**

Când **A2=1, A1=0, A0=0** (**4**) pe ieşirea **Y** se transferă datele de pe intrarea **I4**

Când **A2=1, A1=0, A0=1** (**5**) pe ieşirea **Y** se transferă datele de pe intrarea **I5**

Când **A2=1, A1=1, A0=0** (**6**) pe ieşirea **Y** se transferă datele de pe intrarea **I6**

Când **A2=1, A1=1, A0=1** (**7**) pe ieşirea **Y** se transferă datele de pe intrarea **I7**



a

INTRĂRI SELECŢIE			IEŞIRE
A2	A1	A0	Y
0	0	0	I0
0	0	1	I1
0	1	0	I2
0	1	1	I3
1	0	0	I4
1	0	1	I5
1	1	0	I6
1	1	1	I7

b

Figura 5.4.8 Multiplexor cu 8 intrări

Realizat cu porţi logice elementare, multiplexorul cu 8 intrări arată ca în **figura 5.4.9**.

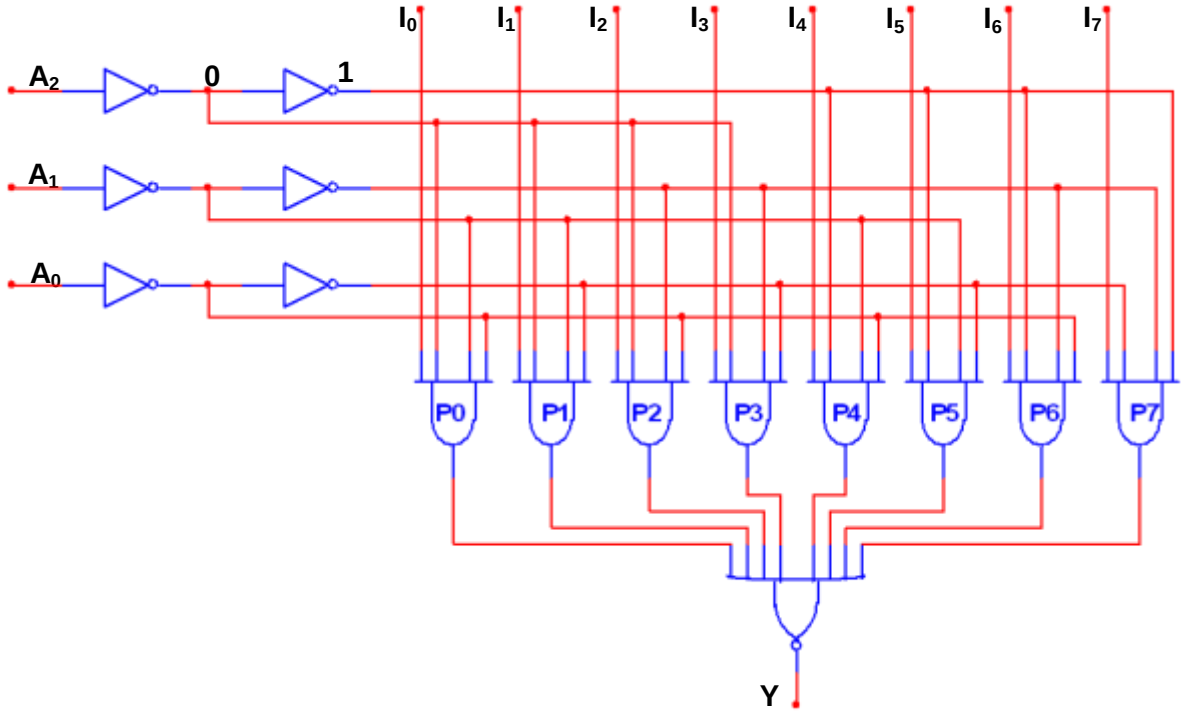
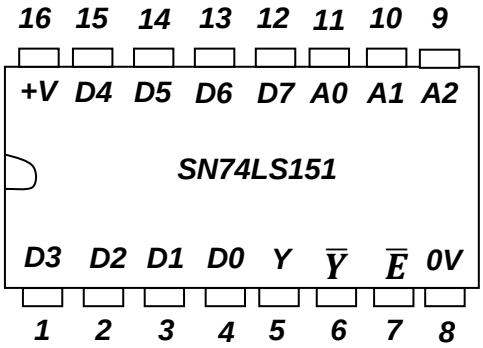


Figura 5.4.9 Multiplexorul cu 8 intrări realizat cu porţi logice

Prezentarea circuitului SN 74LS151 (1 multiplexor cu 8 intrări)

Configuraţia terminalelor



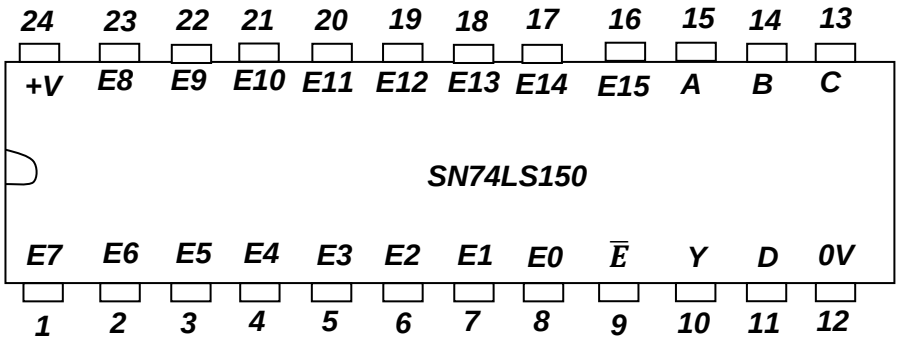
Tabelul de adevăr

INTRĂRI				IEŞIRI	
SELECŢIE			Autorizare	Y	$\overline{Y}$
A2	A1	A0	$\overline{E}$		
X	X	X	1	0	1
0	0	0	0	D0	$\overline{D0}$
0	0	1	0	D1	$\overline{D1}$
0	1	0	0	D2	$\overline{D2}$
0	1	1	0	D3	$\overline{D3}$
1	0	0	0	D4	$\overline{D4}$
1	0	1	0	D5	$\overline{D5}$
1	1	0	0	D6	$\overline{D6}$
1	1	1	0	D7	$\overline{D7}$

Figura 5.4.10 Multiplexorul cu 8 intrări SN74SL151

Prezentarea circuitului SN 74LS150 (1 multiplexor cu 16 intrări)

Configuraţia terminalelor



Tabelul de adevăr

INTRĂRI					Ieşire
D	C	B	A	$\overline{E}$	Y
X	X	X	X	1	1
0	0	0	0	0	$\overline{E0}$
0	0	0	1	0	$\overline{E1}$
0	0	1	0	0	$\overline{E2}$
0	0	1	1	0	$\overline{E3}$
0	1	0	0	0	$\overline{E4}$
0	1	0	1	0	$\overline{E5}$
0	1	1	0	0	$\overline{E6}$
0	1	1	1	0	$\overline{E7}$
1	0	0	0	0	$\overline{E8}$
1	0	0	1	0	$\overline{E9}$
1	0	1	0	0	$\overline{E10}$
1	0	1	1	0	$\overline{E11}$
1	1	0	0	0	$\overline{E12}$
1	1	0	1	0	$\overline{E13}$
1	1	1	0	0	$\overline{E14}$
1	1	1	1	0	$\overline{E15}$

Figura 5.4.11 Multiplexorul cu 16 intrări SN74SL150

VERIFICAREA PRACTICĂ A MULTIPLEXORULUI CU 4 INTRĂRI - SN 74LS153

În figura **5.4.12** este schema unui circuit de verificare practică a unui multiplexor cu 4 intrări realizată cu simulatorul Multisim.

Comutatoarele I0, I1, I2 sunt intrările de date care pot fi 0 logic sau 1 logic în funcţie de poziţia comutatorului.

Comutatoarele A0, A1 sunt intrările de selecţie care pot fi 0 logic sau 1 logic în funcţie de poziţia comutatorului.

Comutatorul E este intrarea de autorizare care poate fi 0 logic sau 1 logic în funcţie de poziţia comutatorului.

La ieşirea circuitului (Y) este conectat prin intermediul unui rezistor R un LED care luminează în 1 logic şi este stins în 0 logic.

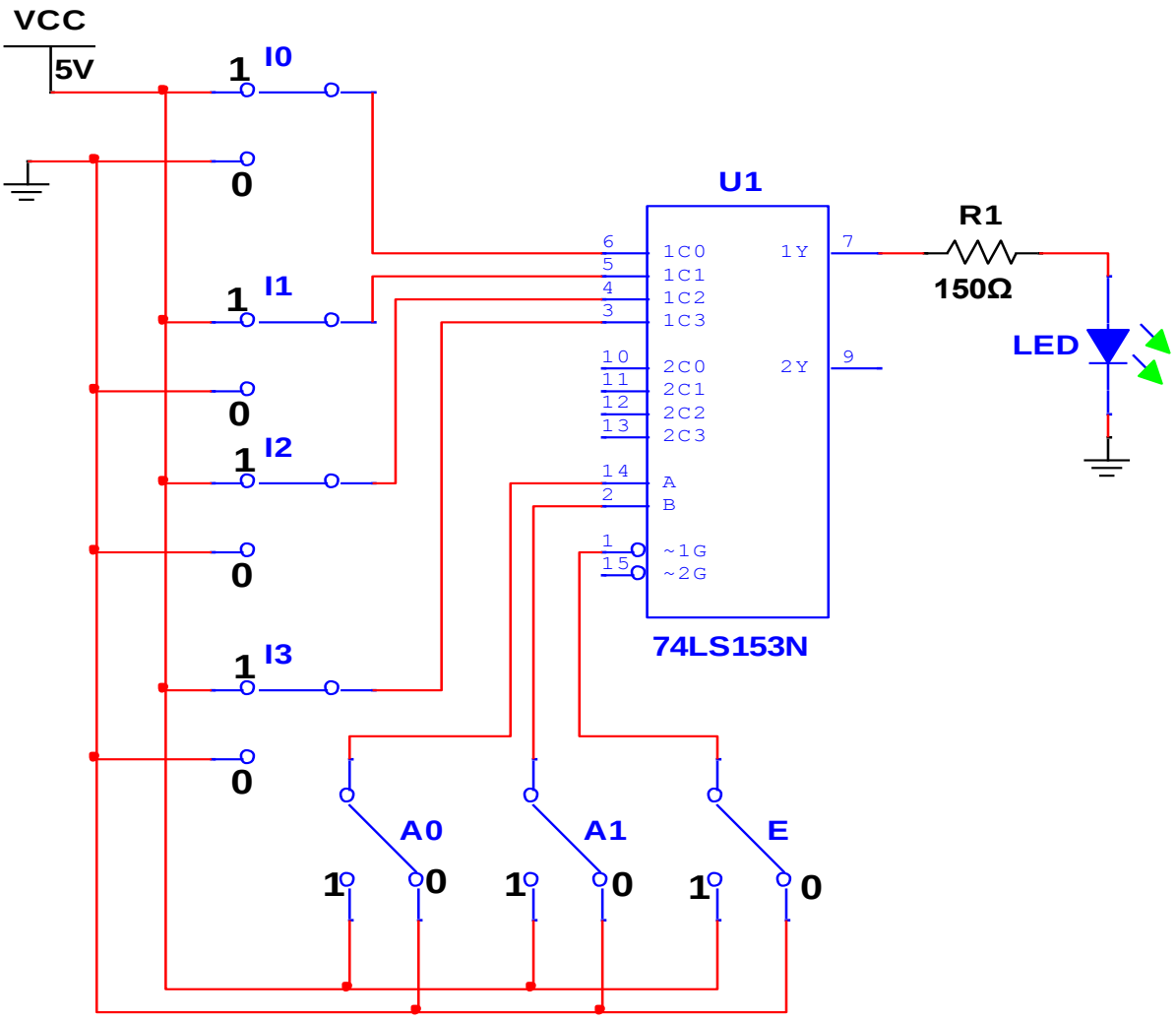


Figura 5.4.12 Schemă de verificare a multiplexorului SN74SL153

Pentru verificarea funcţionării se poziţionează comutatoarele conform tabelei de adevăr din figura 5.4.5 şi se observă starea LED-ului de la ieşirea multiplexorului.

5.5. DEMULTIPLEXOARE

Demultiplexoarele (DMUX) – sunt circuite logice combinaţionale cu o singură intrare şi m ieşiri, care permit transferul datelor de la intrarea unică spre una din cele m ieşiri. Selecţia ieşirii spre care se transferă datele se face prin intermediul unui cuvânt de cod de selecţie numit adresă, cuvânt care are n biţi. Numărul de ieşiri m este egal cu numărul combinaţiilor logice de adresă 2^n a căror apariţie urmează să autorizeze transferul semnalului de intrare succesiv către cele m ieşiri (**$m=2^n$**). Schema de principiul a unui demultiplexor este prezentată în **figura 5.5.1**

.

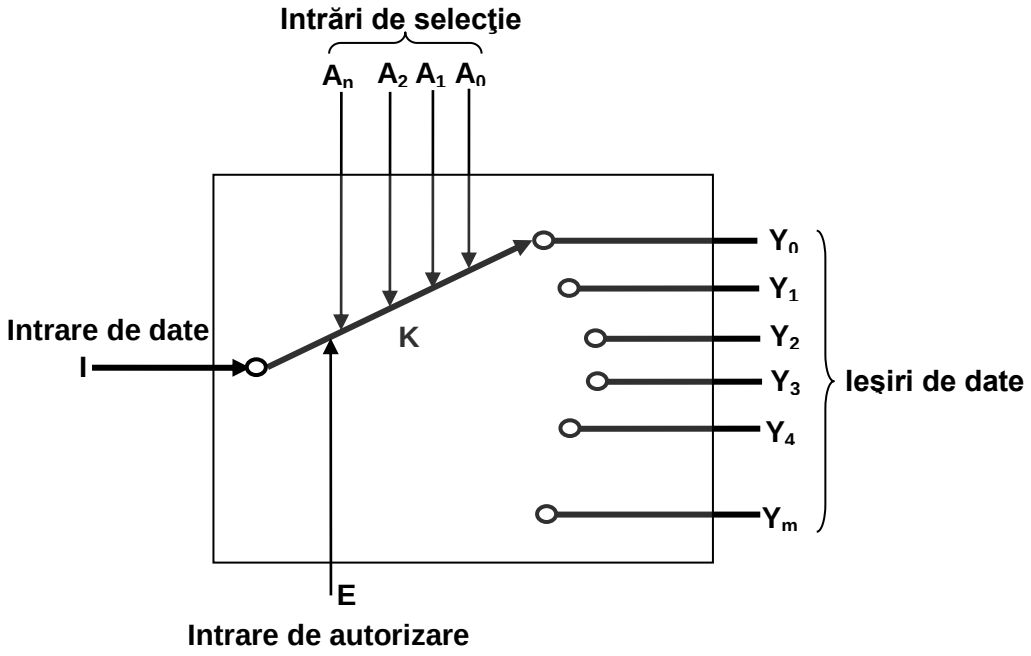


Figura 5.5.1 Schema de principiu a unui demultiplexor

În funcţie de poziţia comutatorului K , semnalul de intrare I va fi transmis uneia din ieşirile de date Y0, Y1, Y2, ...Ym. Poziţia comutatorului este comandată de nivelul logic al intrărilor de selecţie (A1, A2,...An), care formează adresa unei anumite ieşiri de date.

Când codul cuvântului de la intrarea de selecţie (A0,...An) corespunde cu adresa unei ieşiri (Y0,....Ym), semnalul de la intrarea de date (I) este transmis către acea ieşire. Celelalte ieşiri (care nu sunt active) vor trece în 0 logic (la unele circuite în 1 logic).

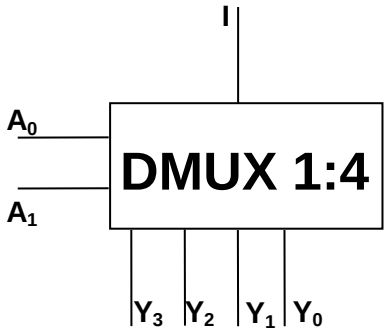
Demultiplexorul mai este prevăzut cu o intrare de autorizare (E) care permite funcţionarea sau blocarea demultiplexorului.

Principala utilizare a demultiplexorului este conversia serie – paralel a datelor binare.

1. DEMULTIPLEXOR CU 4 IEŞIRI

Acest multiplexor (**fig.5.5.2 a**) permite transferul datelor de pe intrarea de date I la una din ieşirile Y0, Y1, Y2, Y3 în funcţie de starea logică a intrărilor de selecţie A0, A1 conform tabelei de adevăr din (**fig. 5.5.2 b**).

Când **A1=0, A0=0** (**0**) semnalul de pe intrarea **I** se transferă pe ieşirea **Y0**
Când **A1=0, A0=1** (**1**) semnalul de pe intrarea **I** se transferă pe ieşirea **Y1**
Când **A1=1, A0=0** (**2**) semnalul de pe intrarea **I** se transferă pe ieşirea **Y2**
Când **A1=1, A0=1** (**3**) semnalul de pe intrarea **I** se transferă pe ieşirea **Y3**



a

Intrări selecţie		Intrare date	Ieşiri de date			
A1	A0	I	Y0	Y1	Y2	Y3
0	0	0	0	0	0	0
0	0	1	1	0	0	0
0	1	0	0	0	0	0
0	1	1	0	1	0	0
1	0	0	0	0	0	0
1	0	1	0	0	1	0
1	1	0	0	0	0	0
1	1	1	0	0	0	1

b

Figura 5.5.2 Demultiplexor cu 4 ieşiri

Realizat cu porţi logice elementare, demultiplexorul cu 4 ieşiri arată ca în figura **5.5.3**

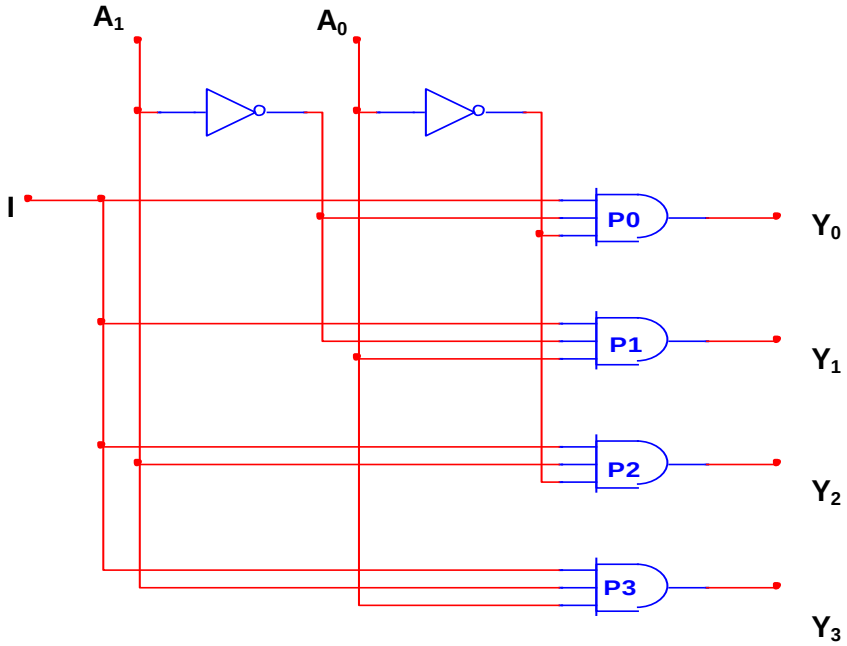
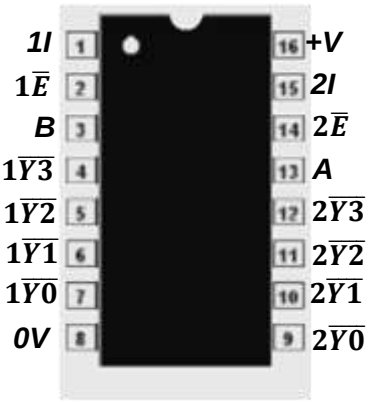


Figura 5.5.3 Demultiplexorul cu 4 ieşiri realizat cu porţi logice

Prezentarea demultiplexorului cu 4 ieşiri - 74LS155N (**figura 5.5.4**)

Configuraţia terminalelor:



Tabelul de adevăr

Intrări selecţie		Intrare autorizare	Intrare date	Ieşiri de date			
A1	A0	$\overline{E}$	I	$\overline{Y0}$	$\overline{Y1}$	$\overline{Y2}$	$\overline{Y3}$
0	0	0	0	1	1	1	1
0	0	0	1	0	1	1	1
0	1	0	0	1	1	1	1
0	1	0	1	1	0	1	1
1	0	0	0	1	1	1	1
1	0	0	1	1	1	0	1
1	1	0	0	1	1	1	1
1	1	0	1	1	1	1	0
X	X	1	X	1	1	1	1

Circuit de verificare a demultiplexorului

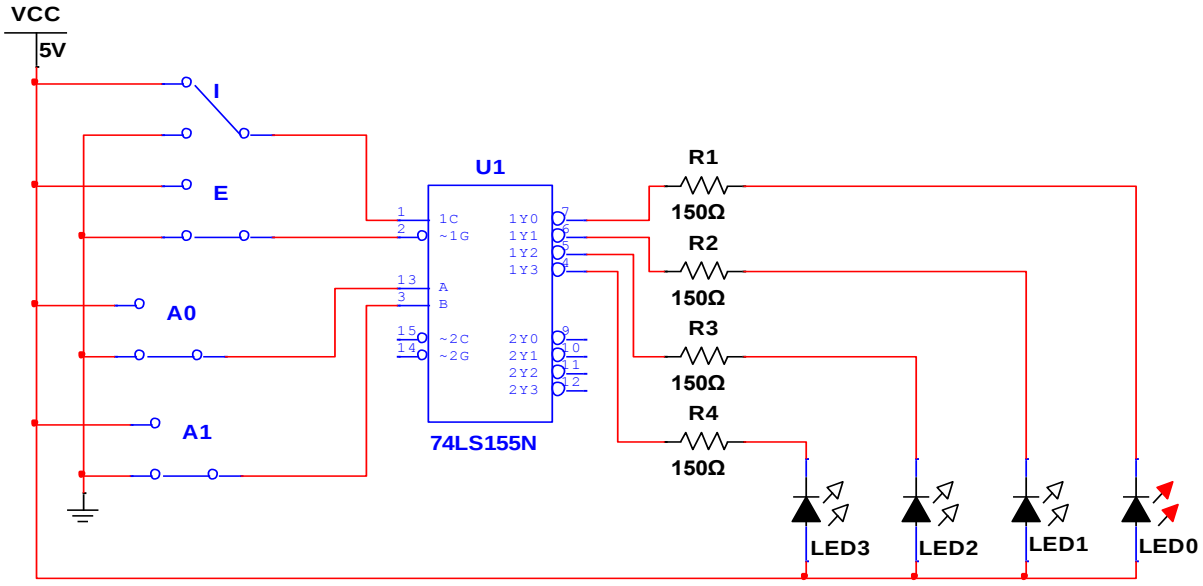


Figura 5.5.4 Demultiplexorul cu 4 ieşiri 74LS155N

2. DEMULTIPLEXOR CU 8 IEŞIRI

Acest multiplexor (**fig.5.5.5 a**) permite transferul datelor de pe intrarea de date I la una din ieşirile Y0, Y1, Y2, Y3, Y4, Y5, Y6, Y7 în funcţie de starea logică a intrărilor de selecţie A0, A1, A2 conform tabelei de adevăr din (**fig. 5.5.5 b**).

Când **A2=0, A1=0, A0=0** (**0**) semnalul de pe intrarea I se transferă pe ieşirea **Y0**

Când **A2=0, A1=0, A0=1** (**1**) semnalul de pe intrarea I se transferă pe ieşirea **Y1**

Când **A2=0, A1=1, A0=0** (**2**) semnalul de pe intrarea I se transferă pe ieşirea **Y2**

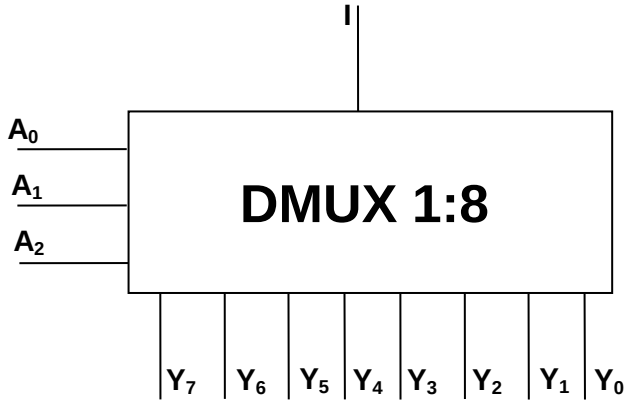
Când **A2=0, A1=1, A0=1** (**3**) semnalul de pe intrarea I se transferă pe ieşirea **Y3**

Când **A2=1, A1=0, A0=0** (**4**) semnalul de pe intrarea I se transferă pe ieşirea **Y4**

Când **A2=1, A1=0, A0=1** (**5**) semnalul de pe intrarea I se transferă pe ieşirea **Y5**

Când **A2=1, A1=1, A0=0** (**6**) semnalul de pe intrarea I se transferă pe ieşirea **Y6**

Când **A2=1, A1=1, A0=1** (7) semnalul de pe intrarea I se transferă pe ieşirea **Y7**



a

INTRĂRI				IEŞIRI							
I	A2	A1	A0	Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
0/1	0	0	0	0	0	0	0	0	0	0	**0/1**
0/1	0	0	1	0	0	0	0	0	0	**0/1**	0
0/1	0	1	0	0	0	0	0	0	**0/1**	0	0
0/1	0	1	1	0	0	0	0	**0/1**	0	0	0
0/1	1	0	0	0	0	0	**0/1**	0	0	0	0
0/1	1	0	1	0	0	**0/1**	0	0	0	0	0
0/1	1	1	0	0	**0/1**	0	0	0	0	0	0
0/1	1	1	1	**0/1**	0	0	0	0	0	0	0

b

Figura 5.5.5 Demultiplexor cu 8 ieşiri

Realizat cu porţi logice elementare, demultiplexorul cu 8 ieşiri arată ca în **figura 5.5.6**

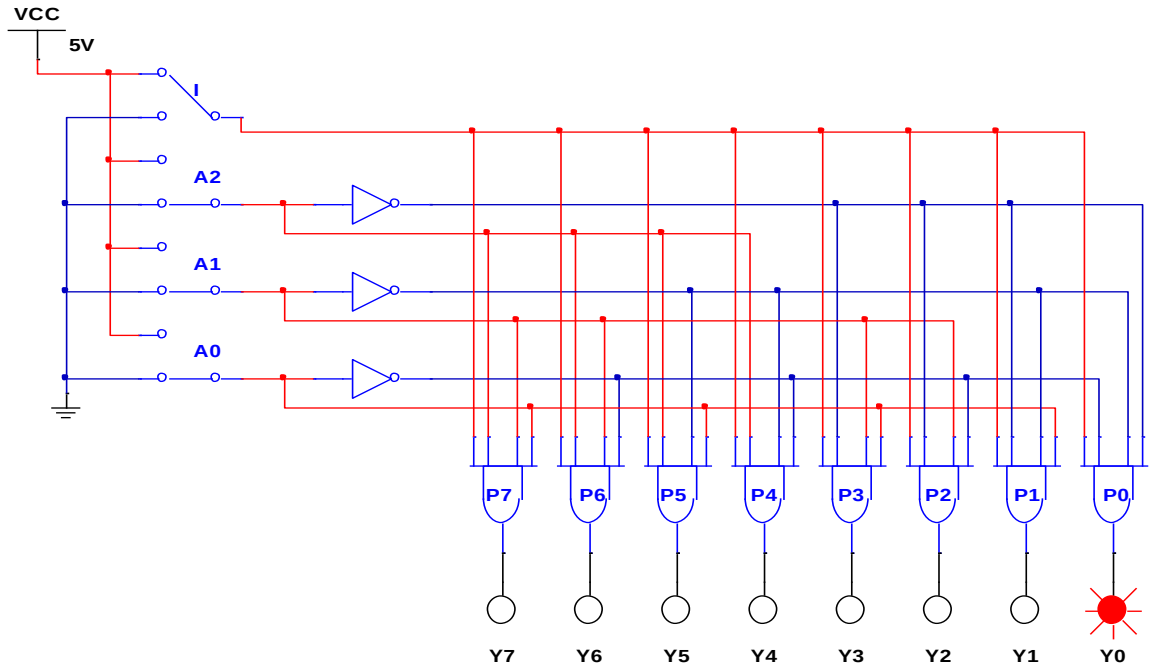
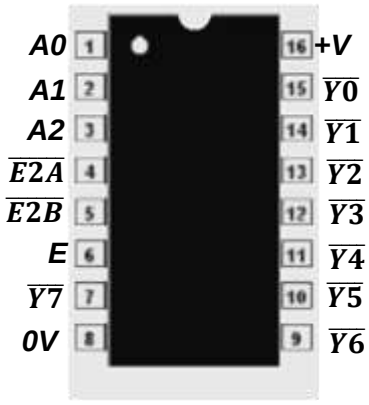


Figura 5.5.6 Circuit de verificare a demultiplexorului cu 8 ieşiri realizat cu porţi logice

Prezentarea demultiplexorului cu 8 ieşiri - 74LS138N (**figura 5.5.7**)

a. Configuraţia terminalelor



b. Tabelul de adevăr

INTRĂRI						IEŞIRI							
$\overline{E2A}$	$\overline{E2B}$	E	A2	A1	A0	Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
0	0	**0/1**	0	0	0	1	1	1	1	1	1	1	**1/0**
0	0	**0/1**	0	0	1	1	1	1	1	1	1	**1/0**	1
0	0	**0/1**	0	1	0	1	1	1	1	1	**1/0**	1	1
0	0	**0/1**	0	1	1	1	1	1	1	**1/0**	1	1	1
0	0	**0/1**	1	0	0	1	1	1	**1/0**	1	1	1	1
0	0	**0/1**	1	0	1	1	1	**1/0**	1	1	1	1	1
0	0	**0/1**	1	1	0	1	**1/0**	1	1	1	1	1	1
0	0	**0/1**	1	1	1	**1/0**	1	1	1	1	1	1	1

Figura 5.5.7 Demultiplexorul cu 8 ieşiri 74LS138N

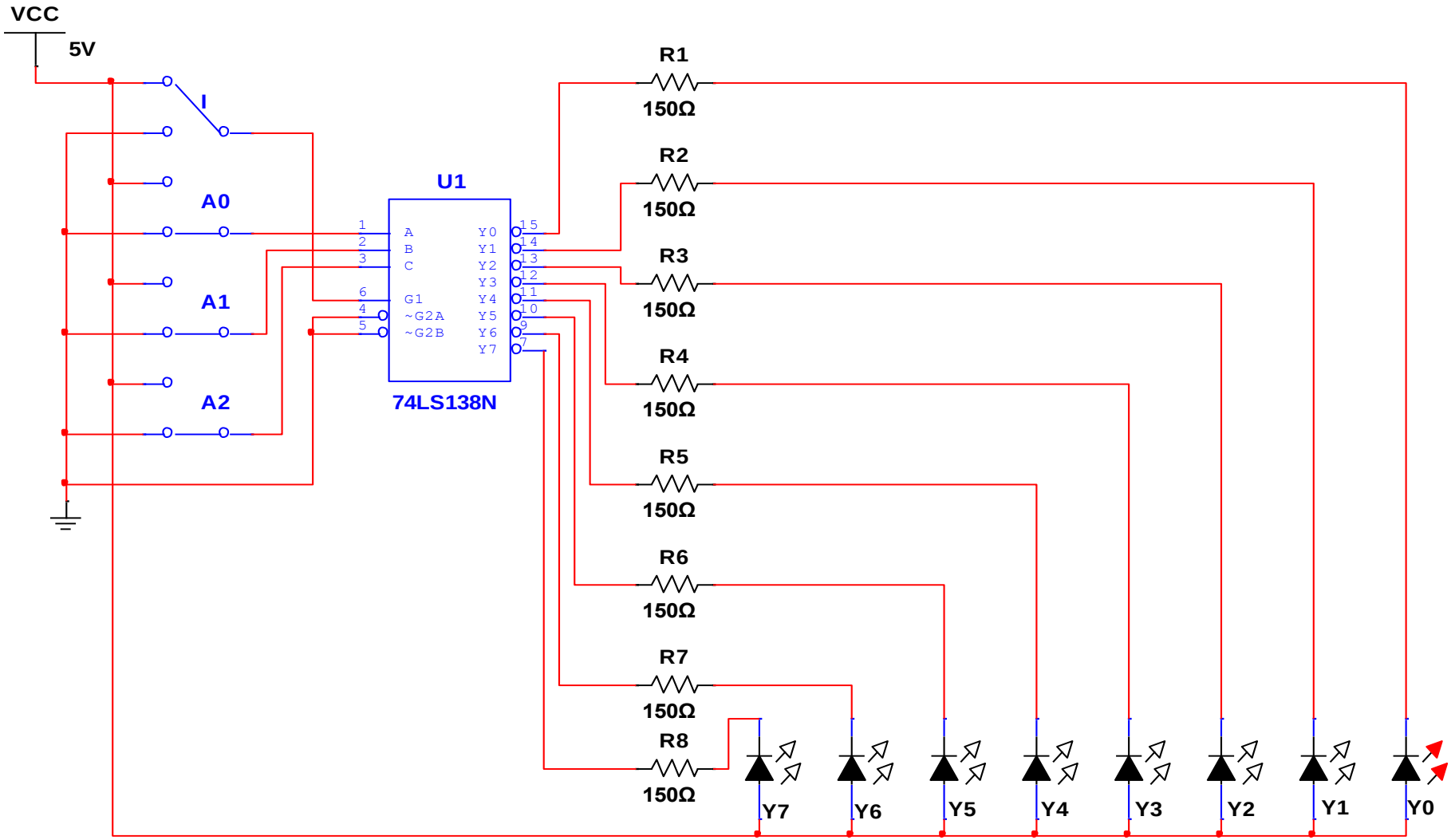


Figura 5.5.7 Circuit de verificare a demultiplexorului cu 8 ieşiri – 74LS138N

5.6. COMPARATOARE NUMERICE

Comparatoarele numerice permit compararea rapidă a două numere binare A şi B şi determinarea valorii relative a acestora (se determină dacă între cele două numere există una din relaţiile **A=B, A>B, A<B**).

Un comparator numeric (**figura 5.6.1**) este prevăzut cu:

- 2n intrări pentru cele 2 numere de n biţi;
- 3 ieşiri cu rezultatul comparaţiei celor 2 numere (A=B, A<B, A>B);
- 3 intrări suplimentare (A=B, A<B, A>B), pentru conectarea în cascadă a mai multor comparatoare atunci când se compară numere cu lungimi mari.

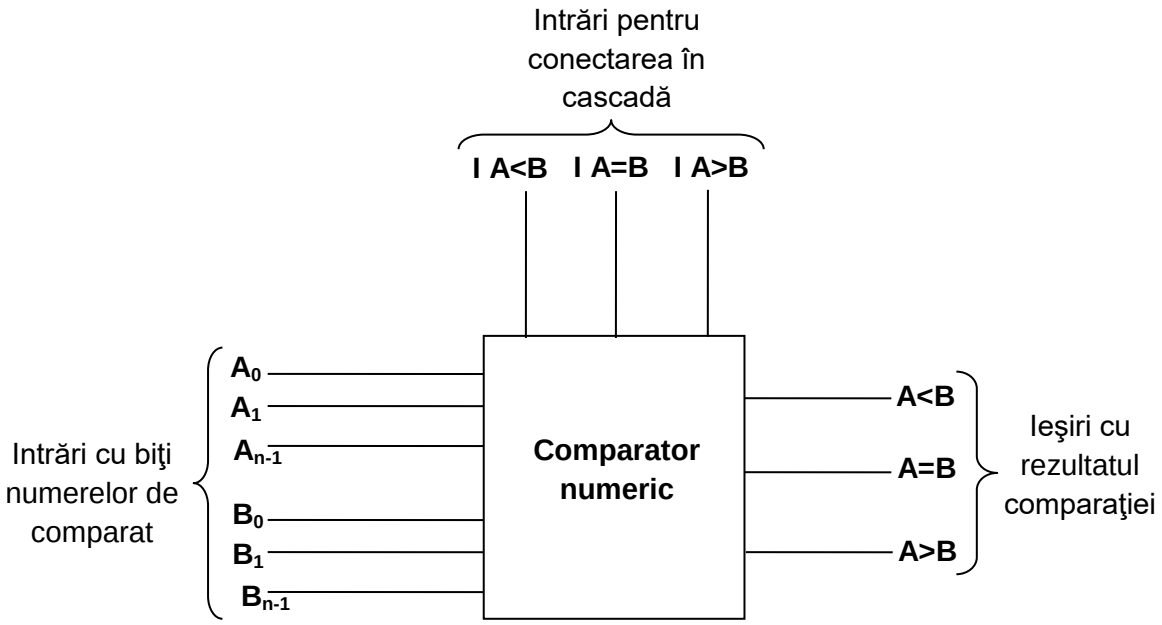


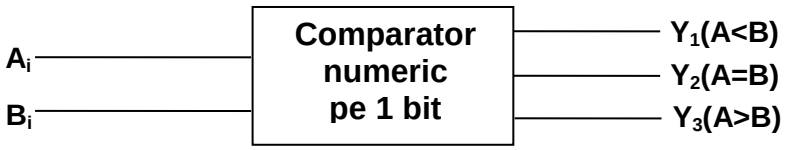
Figura 5.6.1 Schema bloc a unui comparator numeric.

În funcţie de lungimea numerelor de comparat, comparatoarele numerice pot fi:

- Comparatoare numerice pe 1 bit;
- Comparatoare numerice pe 2 biţi;
- Comparatoare numerice pe 4 biţi;
- Comparatoare numerice pe 8 biţi.

1. Comparatorul numeric pe 1 bit.

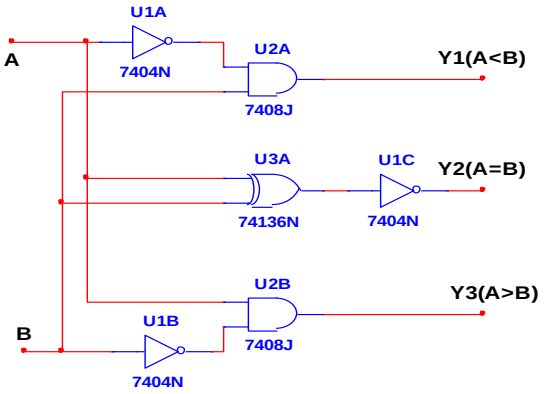
În **figura 5.6.2** sunt prezentate: schema bloc a comparatorului pe 1 bit (**fig. 5.6.2 a**), tabelul de adevăr (**fig. 5.6.2 b**) şi schema logică a comparatorului (**fig. 5.6.2 c**).



a

INTRĂRI		IEŞIRI		
Ai	Bi	Y1(A<B)	Y2(A=B)	Y3(A>B)
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

b



c

Figura 5.6.2 Comparator numeric pe un bit.

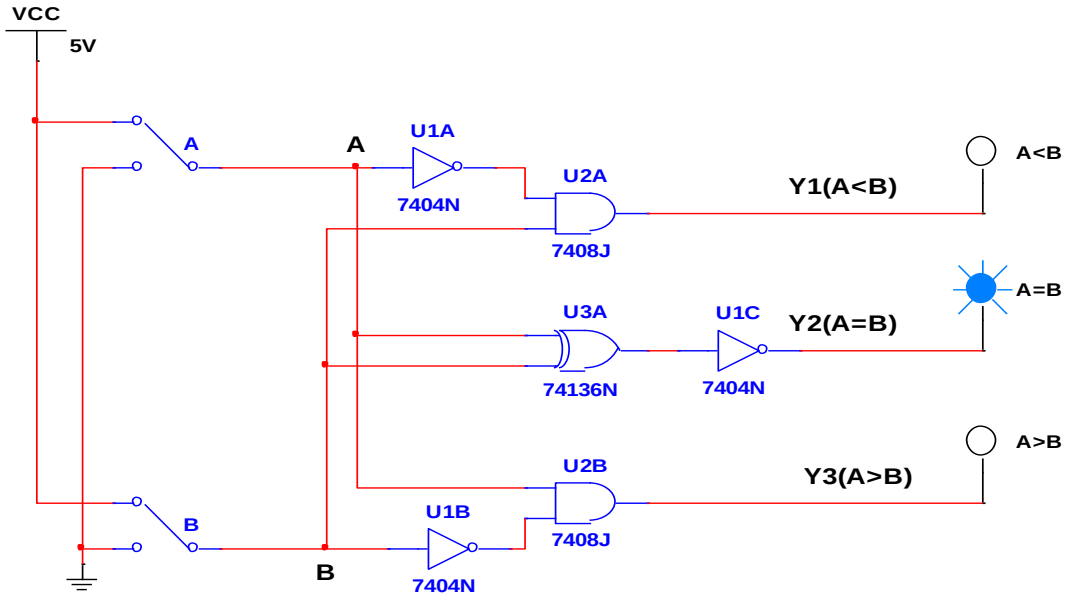
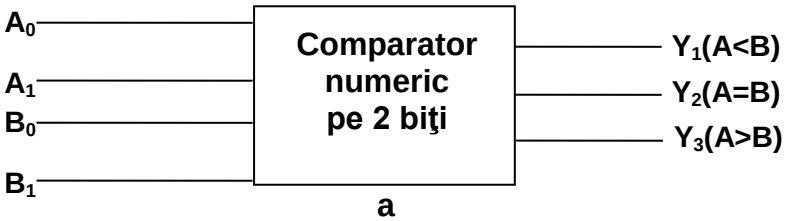


Figura 5.6.3 Circuit de verificare a comparatorului numeric pe un bit.

2. Comparatorul numeric pe 2 biţi.

În **figura 5.6.4** sunt prezentate schema bloc a comparatorului pe 2 biţi (**fig. 5.6.4 a**) şi tabelul de adevăr (**fig. 5.6.4 b**).



a

INTRĂRI				IEŞIRI		
A0	A1	B0	B1	Y1(A<B)	Y2(A=B)	Y3(A>B)
0	0	0	0	0	1	0
0	0	0	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
0	1	1	1	1	0	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	0	1	0

b

Figura 5.6.4 Comparator numeric pe 2 biţi.

În **figura 5.6.5** este prezentată schema de verificare a unui comparator numeric pe 2 biţi realizat cu porţi logice elementare.

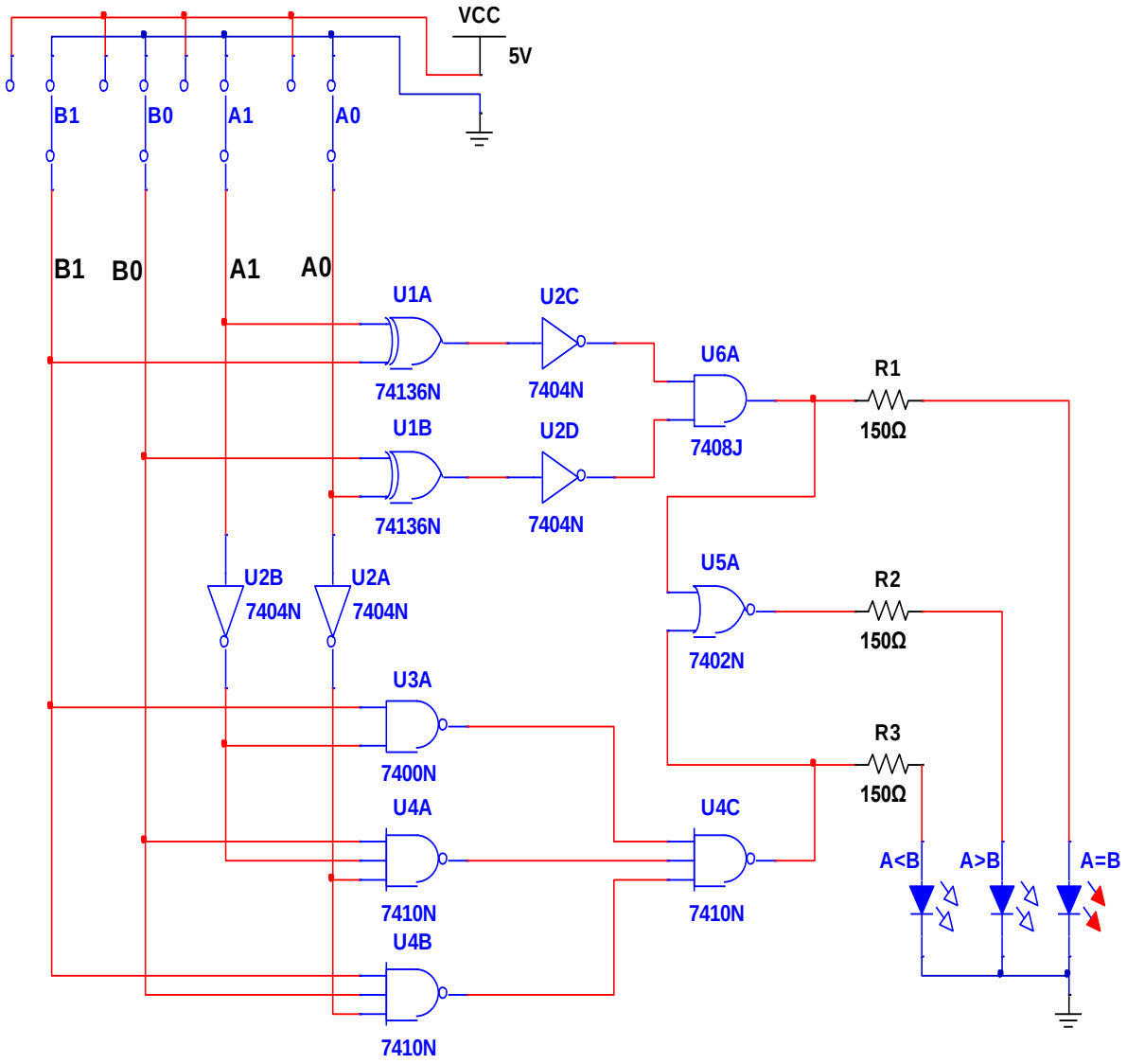
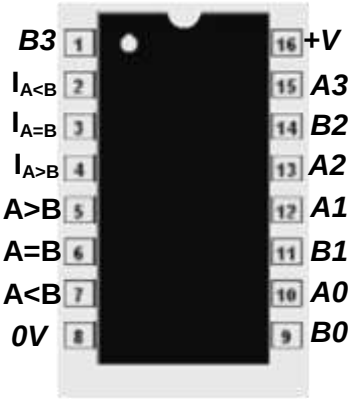


Figura 5.6.5 Verificarea comparatorului numeric pe 2 biţi.

3. Comparatorul numeric pe 4 biţi.

În **figura 5.6.6** este prezentat comparatorul pe 4 biţi – 74LS85N. Spre deosebire de celelalte două tipuri de comparatoare prezentate, acest comparator este prevăzut cu 3 intrări de extindere (I A<B, I A=B, I A>B) pentru conectarea în cascadă cu alt comparator. Acest montaj se utilizează pentru extinderea capacităţii de comparare la 8 biţi.

Configuraţia terminalelor:



Tabelul de adevăr

Compararea intrărilor				Intrări de extindere			Ieşiri		
A3,B3	A2,B2	A1,B1	A0,B0	IA>B	IA<B	IA=B	A>B	A<B	A=B
A3>B3	X	X	X	X	X	X	1	0	0
A3<B3	X	X	X	X	X	X	0	1	0
A3=B3	A2>B2	X	X	X	X	X	1	0	0
A3=B3	A2<B2	X	X	X	X	X	0	1	0
A3=B3	A2=B2	A1>B1	X	X	X	X	1	0	0
A3=B3	A2=B2	A1<B1	X	X	X	X	0	1	0
A3=B3	A2=B2	A1=B1	A0>B0	X	X	X	1	0	0
A3=B3	A2=B2	A1=B1	A0<B0	X	X	X	0	1	0
A3=B3	A2=B2	A1=B1	A0=B0	1	0	0	1	0	0
A3=B3	A2=B2	A1=B1	A0=B0	0	1	0	0	1	0
A3=B3	A2=B2	A1=B1	A0=B0	0	0	1	0	0	1
A3=B3	A2=B2	A1=B1	A0=B0	X	X	1	0	0	1
A3=B3	A2=B2	A1=B1	A0=B0	1	1	0	0	0	0
A3=B3	A2=B2	A1=B1	A0=B0	0	0	0	1	1	0

Figura 5.6.6 Comparatorul pe 4 biţi - 74LS85N

În **figura 5.6.7** este prezentat circuitul de verificare a comparatorului pe 4 biţi – 74LS85N

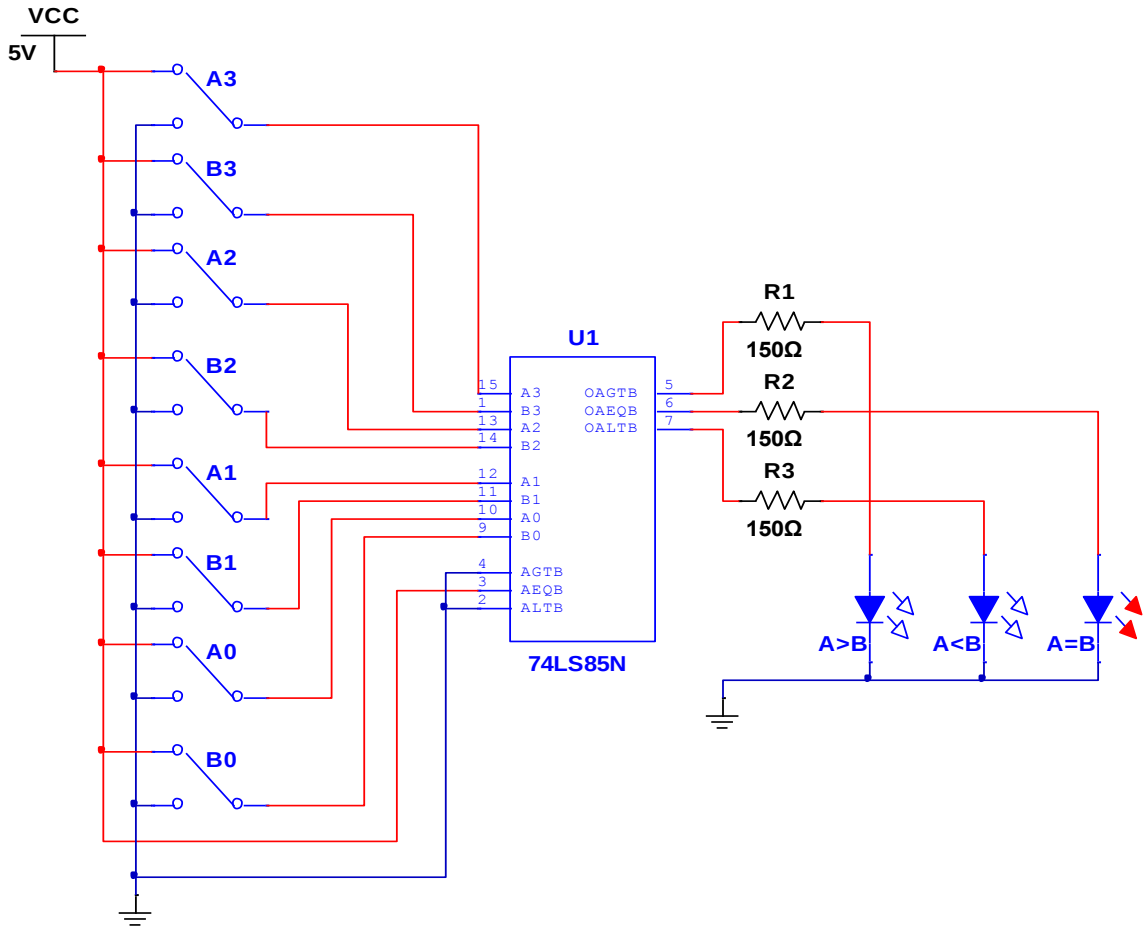


Figura 5.6.7 Verificarea comparatorului pe 4 biţi - 74LS85N

Pentru a obţine un comparator pe 8 biţi se conectează în cascadă două comparatoare pe 4 biţi ca în schema din figura 5.6.8.

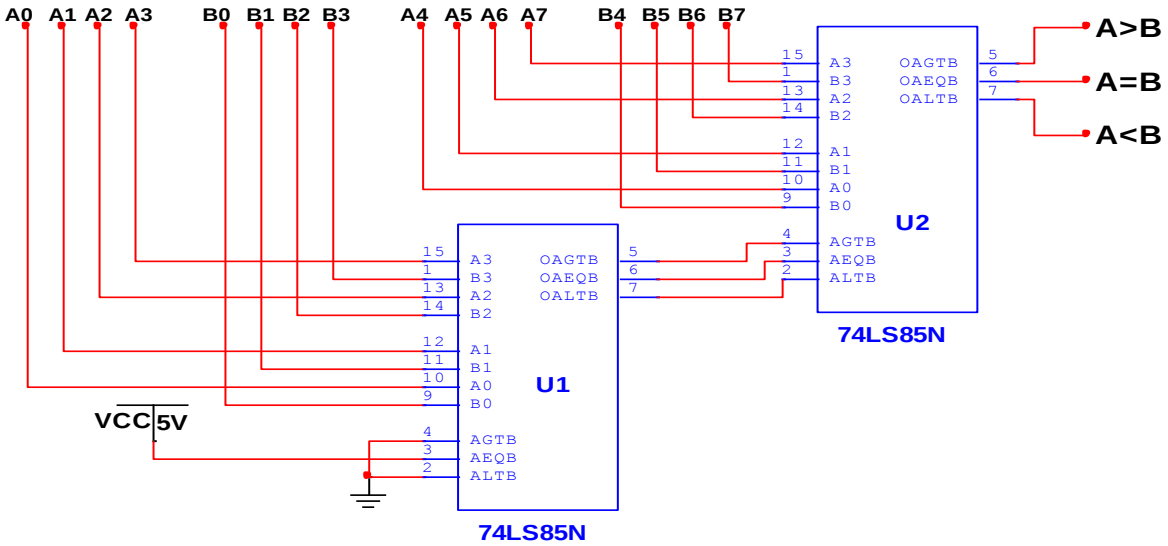


Figura 5.6.8 Schemă comparator pe 8 biţi cu circuite 74LS85N

5.7. SUMATOARE

Sumatoarele sunt circuite logice combinaţionale care realizează operaţii aritmetice (adunarea şi scăderea) cu două numere binare care au un număr egal de biţi.

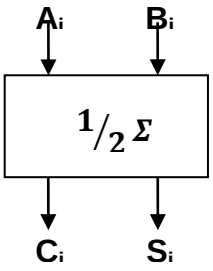
Un sumator pe mai mulţi biţi este construit din mai multe sumatoare pe un bit.

Sumatoarele elementare pe un bit se împart în două categorii:

- Semisumatoare (sumatoare elementare pentru bitul 0) realizează suma a două numere binare de 1 bit fără a ţine seama de transportul de la bitul inferior către rangul următor;
- Sumatoare elementare complete pe 1 bit care ţin seama de transportul de la bitul cu semnificaţie imediat inferioară către rangul următor.

1. Sumatorul elementar pentru bitul 0

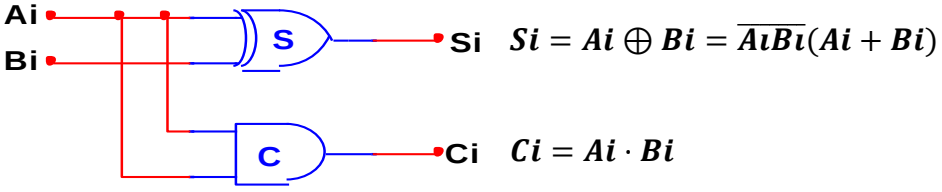
În figura 5.7.1 sunt prezentate: schema bloc, tabelul de adevăr, schema logică a sumatorului elementar pentru bitul 0.



a. Schema bloc

Ai	Bi	Rezultatul adunării	Si	Ci
0	0	00	0	0
0	1	01	1	0
1	0	01	1	0
1	1	10	0	1

b. Tabelul de adevăr



$Si = Ai \oplus Bi = \overline{AiBi}(Ai + Bi)$

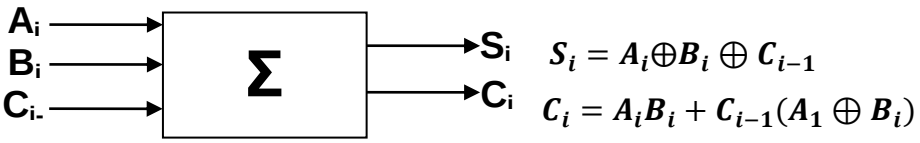
$Ci = Ai \cdot Bi$

c. Schema logică

Figura 5.7.1 Sumatorul elementar pentru bitul 0

2. Sumatorul elementar complet

Acest sumator prezentat în **figura 5.7.2** ia în consideraţie şi transportul de la bitul inferior către rangul următor. Sumatorul adună la intrare 3 biţi: doi biţi de date şi unul de transport, şi furnizează la ieşire un bit sumă şi unul de transport.



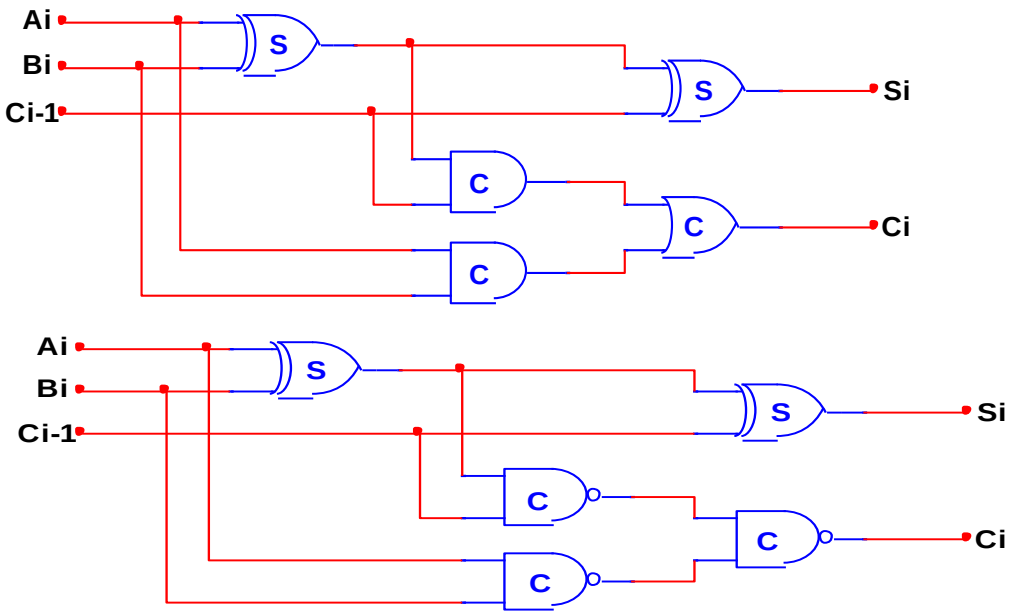
$$S_i = A_i \oplus B_i \oplus C_{i-1}$$

$$C_i = A_i B_i + C_{i-1}(A_1 \oplus B_i)$$

a. Schema bloc

INTRĂRI			Rezultatul adunării	IEŞIRI	
Ai	Bi	Ci-1	SUMA	Ci	Si
0	0	0	00	0	0
0	0	1	01	0	1
0	1	0	01	0	1
0	1	1	10	1	0
1	0	0	01	0	1
1	0	1	10	1	0
1	1	0	10	1	0
1	1	1	11	1	1

b. Tabelul de adevăr



c. Scheme logice

Figura 5.7.2 Sumatorul elementar complet

3. Sumatorul pe 2 biţi

Sumatorul pe 2 biţi se obţine prin interconectarea a 2 sumatoare complete pe un bit.

În **figura 5.7.3** este prezentată schema unei aplicaţii cu sumatorul integrat 74LS183N.

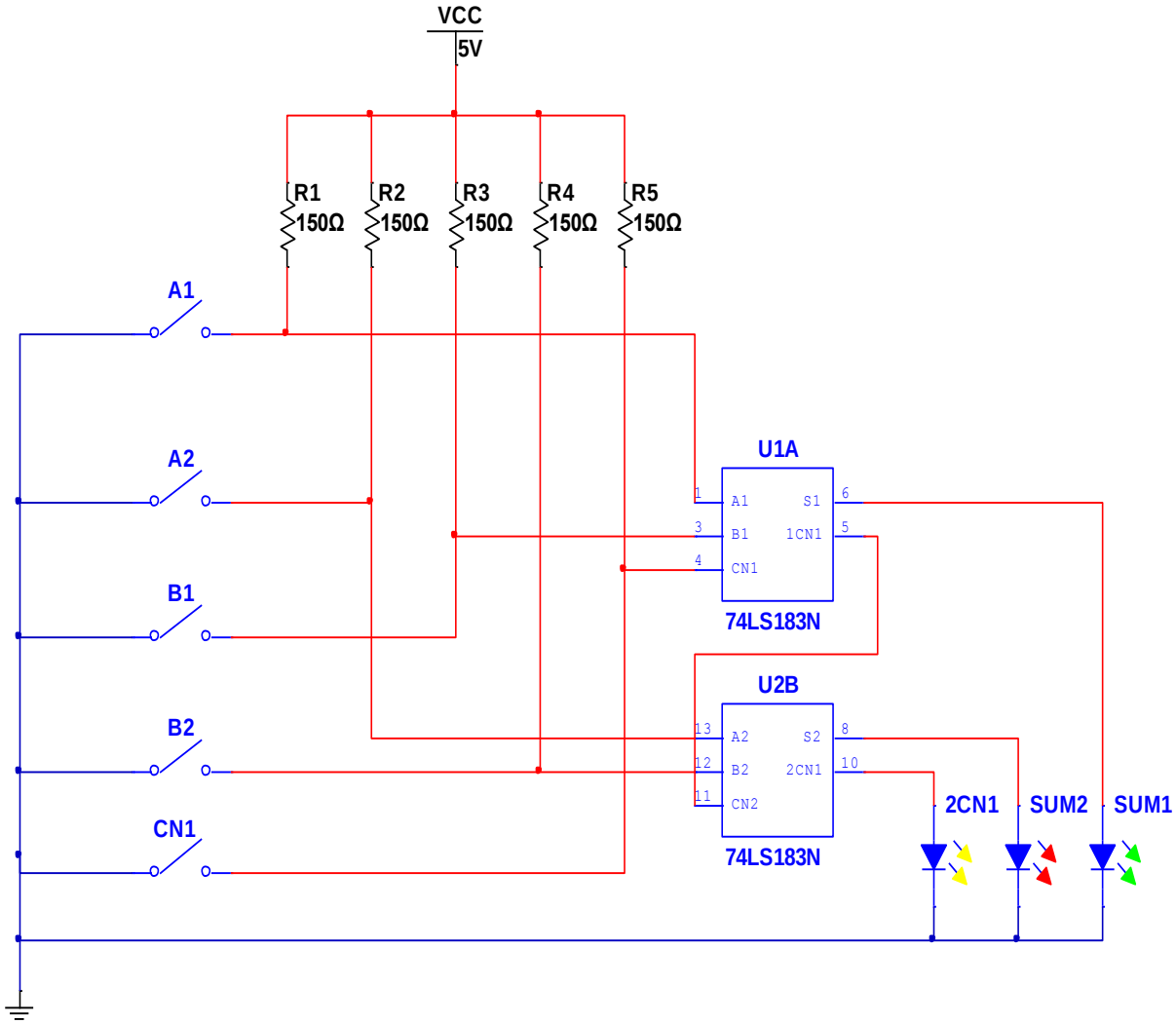


Figura 5.7.4 Sumator pe 2 biţi cu circuitul integrat 74LS183N

Bitul de transport de ieşire 1CN1 (pin 5) de la sumatorul 1, se conectează la bitul de transport de intrare CN2 (pin 11) de la sumatorul 2.

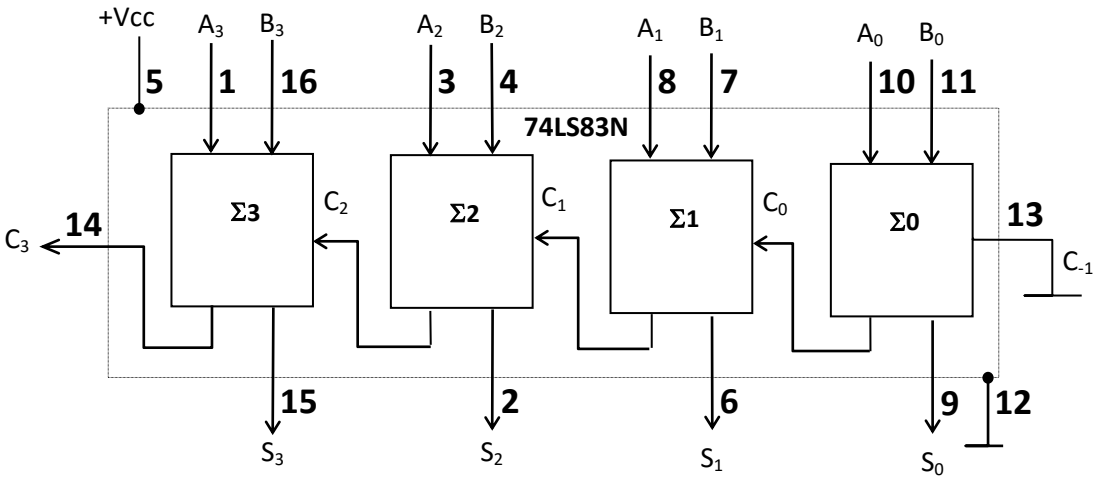
La intrările A1, B1, A2, B2, CN1 se conectează câte un întrerupător care este conectat la masă (0 V) şi o rezistenţă conectată la + 5V. Când întrerupătorul este pe poziţia închis intrarea integratului este în 0 logic iar când întrerupătorul este pe poziţia deschis intrarea integratului este în 1 logic.

La ieşirile S1, S2, 2CN2 sunt conectate LED-uri pentru semnalizare optică.

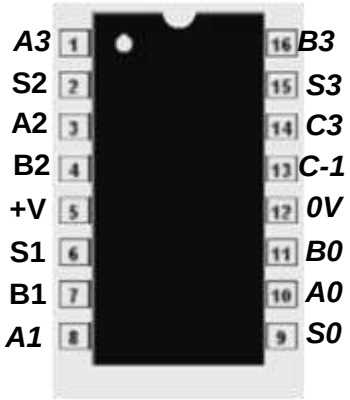
4. Sumatorul cu transport succesiv pe 4 biţi

Acest sumator se obţine prin interconectarea a 4 sumatoare complete pe un bit.

În **figura 5.7.4** este prezentat sumatorul integrat 74LS83N.



a. Schema bloc



b. Configuraţia terminalelor

Figura 5.7.4 Sumatorul elementar complet pe 4 biţi

5.8. CONVERTOARE DE COD

Convertoarele de cod sunt circuite logice combinaţionale care realizează conversia numerelor binare dintr-un cod în alt cod.

La intrarea convertorului se aplică un cod binar iniţial de n biţi iar la ieşire se obţine un alt cod binar final de m biţi.

Organizarea unui convertor de cod se bazează pe un tabel care exprimă corespondenţa dintre codul de intrare şi codul de ieşire, corespondenţă care trebuie să fie unu la unu. Codul de intrare reprezintă un argument în timp ce codul de ieşire este o funcţie de acest argument.

În **figura 5.81** este prezentată schema bloc a unui convertor de cod.

Figura 5.8.1 Schema bloc a convertorului de cod

Convertorul de cod este alcătuit dintr-o pereche decodificator – codificator.

Codul de intrare de n biţi este aplicat mai întâi decodificatorului, rezultând o singură ieşire activă din cele 2n ieşiri. Această ieşire activă a decodificatorului este aplicată la intrarea codificatorului care va genera la ieşirea codificatorului un cod de m biţi.

1. **Convertorul de cod din cod binar natural în cod binar reflectat (Gray).**

În **figura 5.8.2** sunt prezentate schema bloc (**fig.5.8.2 a**) şi schema logică (**fig.5.8.2 b**) a acestui convertor de cod.

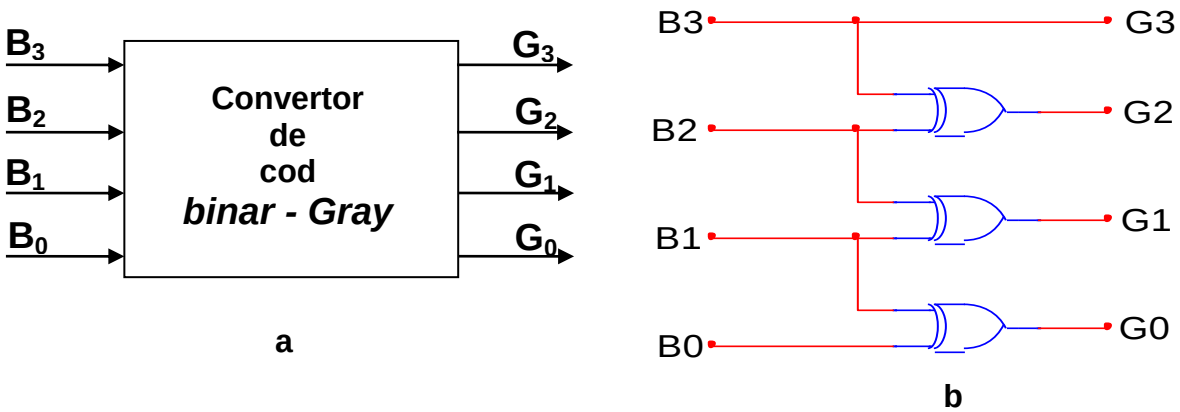


Figura 5.8.2 Convertorul de cod "binar – Gray"

Pentru a înţelege cum este convertit codul binar în cod Gray se studiază tabela de adevăr a convertorului, tabela prezentată mai jos.

Binar natural				Gray			
B3	B2	B1	B0	G3	G2	G1	G0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

Codul Gray este un cod numeric reflectat, care are proprietatea că 2 numere adiacente diferă prin valoarea unui singur bit.

După cum rezultă din tabela de adevăr, codul Gray se obţine din codul binar astfel:

G0 - repetă primele 2 locaţii ale lui B0, după care se reflectă din 2 în 2 locaţii;

G1 - repetă primele 4 locaţii ale lui B1, după care se reflectă din 4 în 4 locaţii;

G2 - repetă primele 8 locaţii ale lui B2, după care se reflectă din 8 în 8 locaţii;

G3 - repetă B3.

2. Convertorul de cod din cod Gray în cod binar natural.

În **figura 5.8.3** sunt prezentate schema bloc (**fig.5.8.3 a**) şi schema logică (**fig.5.8.3 b**) a acestui convertor de cod.

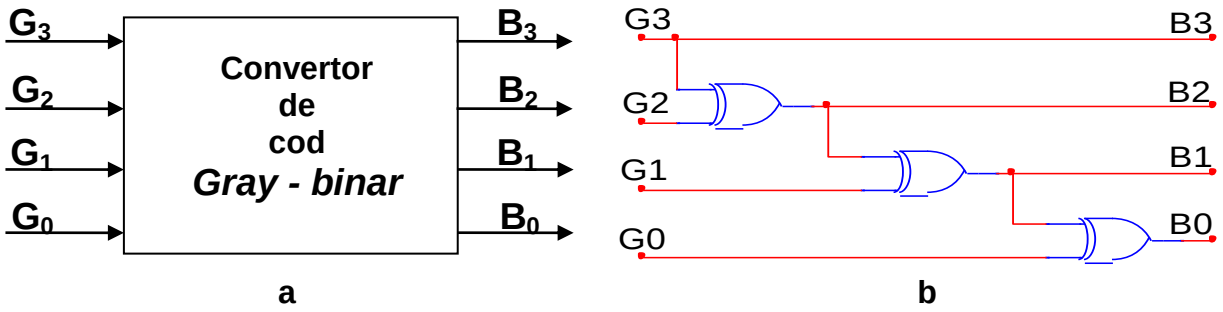


Figura 5.8.3 Convertorul de cod "Gray - binar"

Pentru a înţelege cum este convertit codul Gray în cod binar se studiază tabela de adevăr a convertorului, tabela prezentată mai jos.

Gray				Binar natural			
G3	G2	G1	G0	B3	B2	B1	B0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	1
1	0	0	0	1	1	1	1
1	0	0	1	1	1	1	0
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	1
1	1	1	0	1	0	1	1
1	1	1	1	1	0	1	0

REZUMATUL CAPITOLULUI

- ***Circuitele logice combinaţionale*** (CLC) sunt circuite alcătuite din porţi logice de bază, care se caracterizează prin faptul că în fiecare moment starea logică a ieşirii depinde de modul în care se combină nivelurile logice ale intrărilor în acel moment, fiind independente de propriile stări anterioare.
- Pentru prelucrarea datelor în sistemele digitale şi pentru citirea şi afişarea rezultatelor prelucrării, este necesară parcurgerea următoarelor etape:
  a. ***Codarea şi decodarea*** – transformarea datelor dintr-un cod în altul;
  b. ***Multiplexarea*** – transmiterea către o ieşire a unei singure informaţii dintr-un grup de informaţii;
  c. ***Demultiplexarea*** – introducerea succesivă a datelor la diferite adrese posibile.
- **Codificatoarele (CD)** – sunt circuite logice combinaţionale cu n intrări şi m ieşiri care furnizează la ieşire un cod de m biţi atunci când numai una din cele n intrări este activă.
- Circuitele de codificare primesc la intrare semnale codificate într-un cod diferit de cel binar şi furnizează la ieşire semnale în cod binar sau echivalentul acestuia.
- Cel mai utilizat codificator este ***codificatorul zecimal-BCD*** la intrarea căruia se aplică date în sistemul zecimal iar la ieşire apar date în codul BCD.
- **Decodificatoarele (DCD)** – sunt circuite logice combinaţionale cu n intrări şi m ieşiri ($m=2^n$) care activează o singură ieşire corespunzătoare codului aplicat la intrare.
- Circuitele de codificare primesc la intrare semnale logice în cod binar sau echivalentul acestuia şi furnizează la ieşire semnale în cod zecimal sau echivalentul acestuia.
- Cele mai utilizate decodificatoare sunt: **decodificatorul BCD-zecimal** şi **decodificatorul BCD-7 segmente**.
- **Multiplexoarele (MUX)** – sunt circuite logice combinaţionale cu m intrări şi o singură ieşire, care permit transferul datelor de la una din intrări spre ieşirea unică. Selecţia intrării de la care se transferă datele se face prin intermediul unui cuvânt de cod de selecţie numit adresă, cuvânt care are n biţi.

- **Demultiplexoarele (DMUX)** – sunt circuite logice combinaţionale cu o singură intrare şi m ieşiri, care permit transferul datelor de la intrarea unică spre una din cele m ieşiri. Selecţia ieşirii spre care se transferă datele se face prin intermediul unui cuvânt de cod de selecţie numit adresă, cuvânt care are n biţi.
- **Comparatoarele numerice** permit compararea rapidă a două numere binare A şi B şi determinarea valorii relative a acestora (se determină dacă între cele două numere există una din relaţiile **A=B, A>B, A<B**).
- **Sumatoarele** sunt circuite logice combinaţionale care realizează operaţii aritmetice (adunarea şi scăderea) cu două numere binare care au un număr egal de biţi. Un sumator pe mai mulţi biţi este construit din mai multe sumatoare pe un bit.
- **Convertoarele de cod** sunt circuite logice combinaţionale care realizează conversia numerelor binare dintr-un cod în alt cod. La intrarea convertorului se aplică un cod binar iniţial de n biţi iar la ieşire se obţine un alt cod binar final de m biţi.
- Convertorul de cod este alcătuit dintr-o pereche decodificator – codificator. Codul de intrare de n biţi este aplicat mai întâi decodificatorului, rezultând o singură ieşire activă din cele 2n ieşiri. Această ieşire activă a decodificatorului este aplicată la intrarea codificatorului care va genera la ieşirea codificatorului un cod de m biţi.

5.9. LUCRĂRI DE LABORATOR

LUCRARE DE LABORATOR 4

DECODIFICATORUL BCD - ZECIMAL.

- **OBIECTIVE:**
  - Realizarea schemei circuitului de decodificare cu simulatorul;
  - Realizarea practică a circuitului de decodificare;
  - Realizarea tabelului de adevăr în funcție de poziția comutatoarelor de intrare și indicațiile LED-urilor de ieșire;

- **RESURSE:**
  - Calculatoare cu soft de simulare a circuitelor electronice;
  - Proiector multimedia;
  - Sursă de tensiune continuă reglabilă;
  - Pistoale de lipit;
  - Accesorii pentru lipit, conductoare;
  - Plăcuțe de lucru;
  - Rezistoare, comutatoare, LED-uri, circuite integrate decodificatoare.

- **DESFĂȘURAREA LUCRĂRII:**

1. Realizează cu simulatorul schema electronică din figura de mai jos:

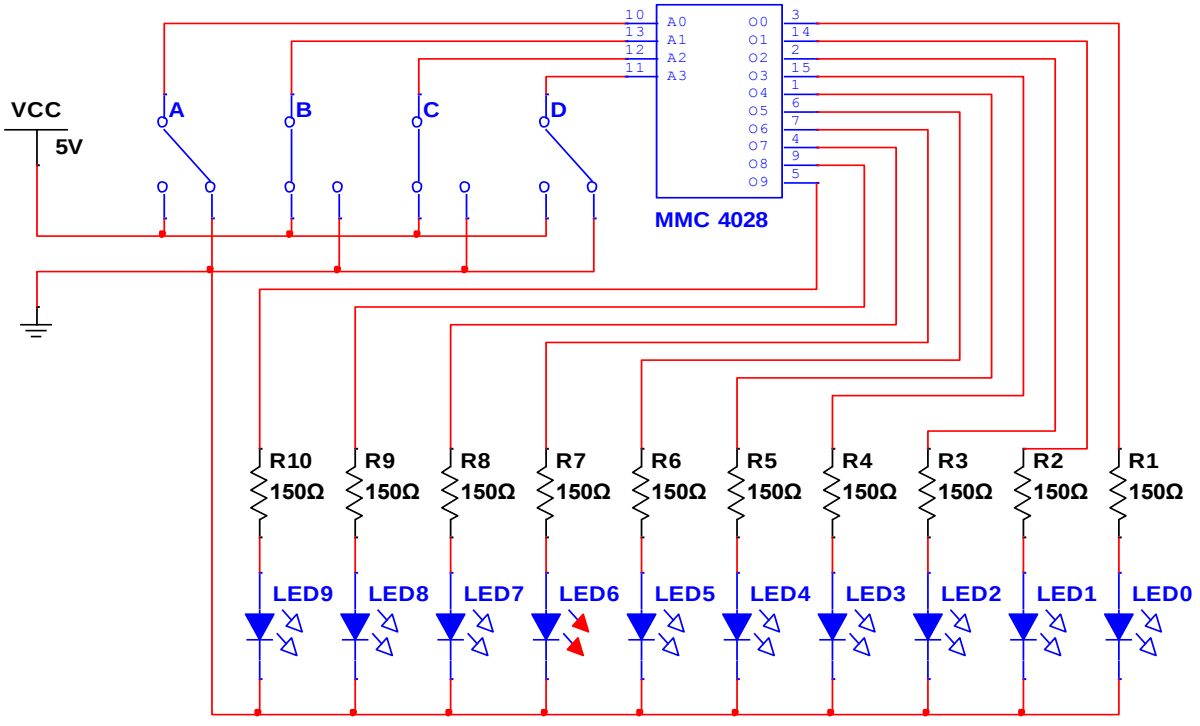


Figura 5.9.1 Aplicaţie cu decodificatorul MMC 4028

2. Realizează practic, pe plăcuţa de probă montajul corespunzător schemei date.

ATENȚIE! Pinul 8 al CI se conectează la (-) iar pinul 16 al CI se conectează la (+).

3. Plasează în soclu de pe placa de probă circuitul integrat (ATENȚIE la poziția CI).

4. Conectează montajul la o sursă de tensiune continuă conform schemei de mai sus, pornește sursa și regleaz-o la valoarea indicată în schemă.

5. Conectează succesiv cele 4 comutatoare de intrare **D**, **C**, **B**, **A** la potenţialul **0V** respectiv **5V** conform tabelului de adevăr de mai jos şi notează în tabel valorile logice ale ieşirilor, **"0"** sau **"1",** în funcţie de starea LED-ului de pe ieşirea respectivă.

Nr. zecimal	INTRĂRI				IEȘIRI									
	D 2^3 8	C 2^2 4	B 2^1 2	A 2^0 1	L0	L1	L2	L3	L4	L5	L6	L7	L8	L9
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
1	0	0	0	1										
2	0	0	1	0										
3	0	0	1	1										
4	0	1	0	0										
5	0	1	0	1										
6	0	1	1	0										
7	0	1	1	1										
8	1	0	0	0										
9	1	0	0	1										

6. OBSERVAȚII:

……..

……..

……..

……..

LUCRARE DE LABORATOR 5

DECODIFICATORUL BCD – 7 SEGMENTE.

- **OBIECTIVE:**
  - Realizarea schemei circuitului de decodificare cu simulatorul;
  - Realizarea practică a circuitului de decodificare;
  - Realizarea tabelului de adevăr în funcție de poziția comutatoarelor de intrare și indicațiile segmentelor afișajului;

- **RESURSE:**
  - Calculatoare cu soft de simulare a circuitelor electronice;
  - Proiector multimedia;
  - Sursă de tensiune continuă reglabilă;
  - Pistoale de lipit;
  - Accesorii pentru lipit, conductoare;
  - Plăcuțe de lucru;
  - Rezistoare, comutatoare, LED-uri, circuite integrate decodificatoare.

- **DESFĂȘURAREA LUCRĂRII:**

1. Realizează cu simulatorul schema electronică din figura de mai jos:

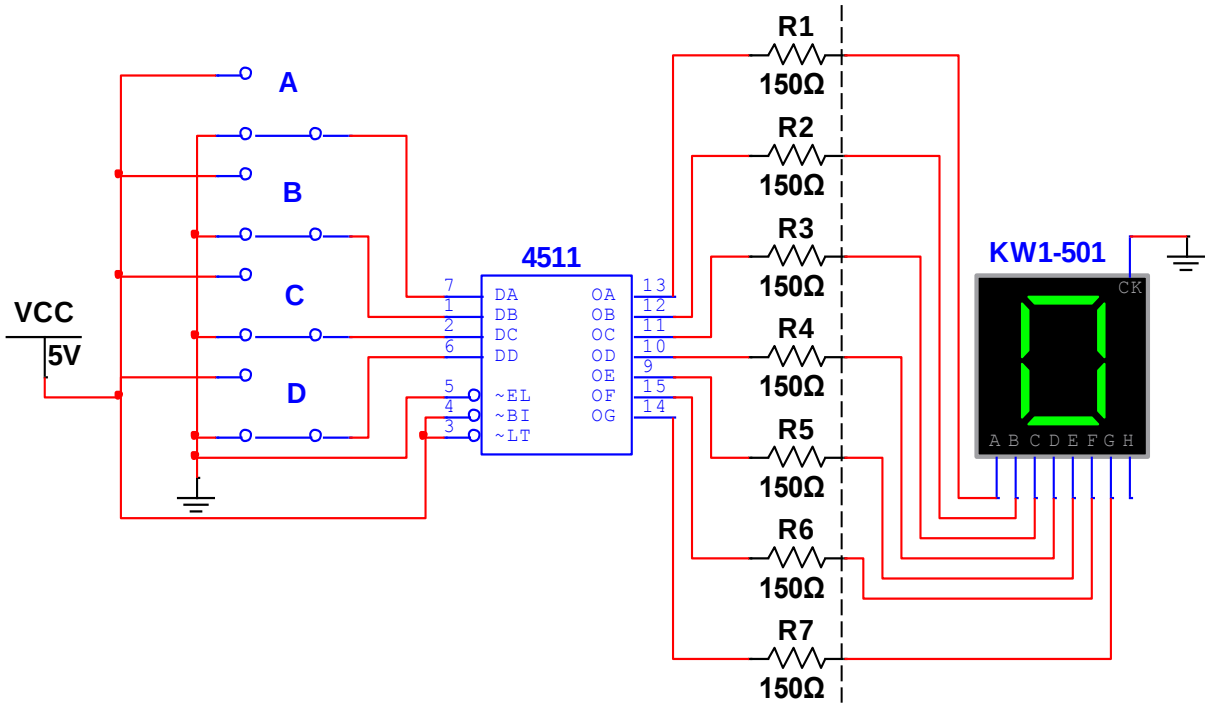


Figura 5.9.1 Aplicaţie cu decodificatorul MMC 4028

2. Realizează practic, pe plăcuţa de probă montajul corespunzător schemei date.

ATENȚIE! Pinul 8 al CI se conectează la (-) iar pinul 16 al CI se conectează la (+).

3. Lipește conductoarele conectate la soclul afișajului la terminalele rezistoarelor R1 - R7 conform schemei.

4. Plasează în soclu de pe placa de probă circuitul integrat (ATENȚIE la poziția CI).

5. Conectează montajul la o sursă de tensiune continuă conform schemei de mai sus, pornește sursa și regleaz-o la valoarea indicată în schemă.

6. Conectează succesiv cele 4 comutatoare de intrare **D**, **C**, **B**, **A** la potenţialul **0V** respectiv **5V** conform tabelului de adevăr de mai jos şi notează în tabel valorile logice ale ieşirilor, **“0”** sau **“1”,** în funcţie de starea segmentului afișajului.

D	C	B	A	**cifra**	a	b	c	d	e	f	g
0	0	0	0	**0**	**1**	**1**	**1**	**1**	**1**	**1**	0
0	0	0	1	**1**							
0	0	1	0	**2**							
0	0	1	1	**3**							
0	1	0	0	**4**							
0	1	0	1	**5**							
0	1	1	0	**6**							
0	1	1	1	**7**							
1	0	0	0	**8**							
1	0	0	1	**9**							

7. OBSERVAȚII:

……..

……..

……..

……..

CAPITOLUL 6. CIRCUITE LOGICE SECVENŢIALE

6.1. GENERALITĂŢI

Circuitele logice secvenţiale (CLS) – sunt circuite logice combinaţionale cu memorie. Aceste circuite se caracterizează prin faptul că în fiecare moment starea logică a ieşirilor depind atât de starea logică a intrărilor cât şi de stările logice anterioare ale intrărilor sau ale circuitului.

Un circuit logic secvenţial se obţine dintr-un circuit logic combinaţional la care se adaugă o serie de elemente de circuit secundare (elemente de memorie), care reprezintă conexiuni de reacţie inversă (prin intermediul elementelor de memorie o parte din ieşirile circuitului sunt conectate la intrările circuitului). În **figura 6.1.1** este reprezentată schema bloc a unui circuit logic secvenţial.

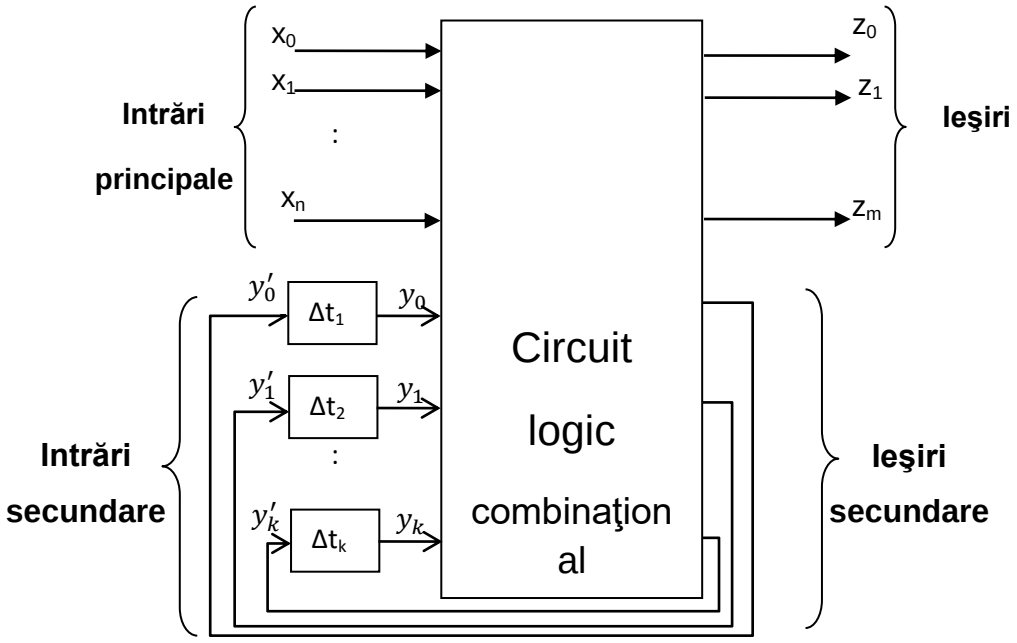


Figura 6.1.1 Schema bloc a unui circuit logic secvenţial

X0, X1, ...Xn – intrări principale accesibile din exterior

Z0, Z1, ...Zm – ieşiri principale accesibile din exterior

Y0, Y1, ...YK – intrări secundare , nu sunt accesibile din exterior.

Starea intrărilor secundare formează starea internă PREZENTĂ a CLS

$\boldsymbol{Y'_0}$, $\boldsymbol{Y'_1}$,..... $\boldsymbol{Y'_K}$ - ieşiri secundare, nu sunt accesibile din exterior

Starea ieşirilor secundare formează starea internă URMĂTOARE a CLS

Δt0, Δt1, ...Δtk – elemente de memorie (de întârziere)

Stările URMĂTOARE devin PREZENTE după un interval de timp determinat de elementele de memorie (întârziere).

La circuitele logice secvenţiale variabilele de intrare, ieşire şi stare pot avea numai două valori “1logic” şi “0 logic” cu un număr finit de stări.

În funcţie de elementele de memorie, care asigură temporizarea semnalelor, circuitele logice secvenţiale se împart în două mari categorii:

circuite secvenţiale asincrone

circuite secvenţiale sincrone

La circuitele secvenţiale asincrone, starea prezentă a circuitului poate fi modificată în orice moment, ca efect al schimbării nivelelor logice aplicate la intrările principale. Fiecare element de memorie este format dintr-un şir de porţi logice prin care întârzie semnalul logic care se propagă prin aceste porţi, deci elementul de memorie este un dispozitiv de întârziere. Deoarece această întârziere nu poate fi controlată, aceste circuite pot deveni instabile, motiv pentru care circuitele secvenţiale asincrone se utilizează foarte rar.

La circuitele secvenţiale sincrone, starea prezentă a circuitului poate fi modificată numai la apariţia unui semnat de temporizare numit semnal de ceas sau tact. Semnalul de ceas este un şir de impulsuri dreptunghiulare care se aplică circuitului printr-o intrare suplimentară numită intrarea semnalului de ceas. Elementele semnalului de ceas sun prezentate în figura 6.1.2.

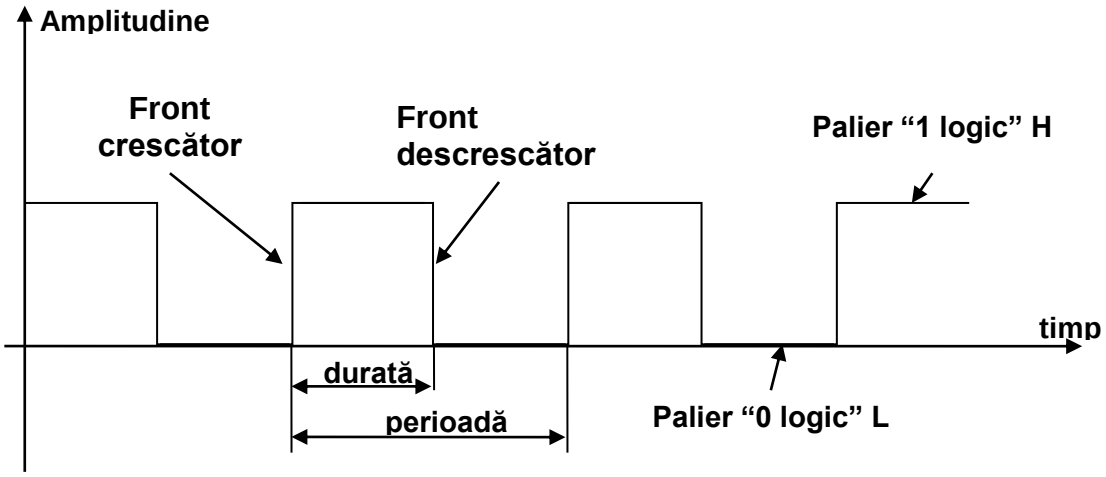


Figura 6.1.2 Elementele unui semnal de ceas (semnal dreptunghiular)

Raportul dintre lăţimea duratei şi perioadei semnalului de ceas se numeşte factor de umplere.

Un semnal de ceas poate fi activ fie pe frontul crescător (atunci când starea circuitului se schimbă pe frontul crescător) sau pe frontul descrescător (atunci când starea circuitului se schimbă pe frontul descrescător).

6.2. CIRCUITE BASCULANTE BISTABILE

Circuitele basculante bistabile (CBB) – sunt cele mai simple circuite logice secvenţiale, cu două stări stabile, utilizate ca elemente de memorie în circuitele logice secvenţiale complexe în scopul memorării stărilor interne ale acestora.

Un CBB este prevăzut cu două sau mai multe intrări şi două ieşiri care sunt complementare una faţă de cealaltă şi funcţionează ca o memorie de 1 bit.

Intrările CBB sunt utilizate pentru a provoca bascularea circuitului (se schimbă stările logice ale ieşirilor) la apariţia unui impuls pe intrare. CBB va rămâne în această stare şi după dispariţia impulsului pe intrare. CBB memorează o anumită informaţie până la apariţia unui impuls pe intrarea acestuia.

În funcţie de numărul intrărilor CBB pot funcţiona în 2 regimuri:

Regim asincron – CBB are numai intrări de date, fără a fi prevăzut cu intrare de tact, la care starea circuitului la ieşire este determinată de combinaţiile de valori ale intrărilor de date (latch-uri).

Regim sincron – CBB are pe lângă intrările de date şi o intrare de tact, care determină momentul în care combinaţiile valorilor ale intrărilor de dare modifică starea ieşirilor circuitului (bistabile).

În funcţie de modul de comandă şi de stările disponibile CBB pot fi:

- De tip RS;
- De tip JK;
- De tip D;
- De tip T.

Tipuri de latch-uri (CBB asincrone):

TTL - 74LS256, 74LS259, 74LS373, 74LS375, 74LS75.

CMOS - 4042, 4043, 4044, 4508.

Tipuri de bistabile (CBB sincrone):

TTL - 74107, 74109, 74112, 74173, 74174, 74175.

CMOS - 4013, 4027, 4076.

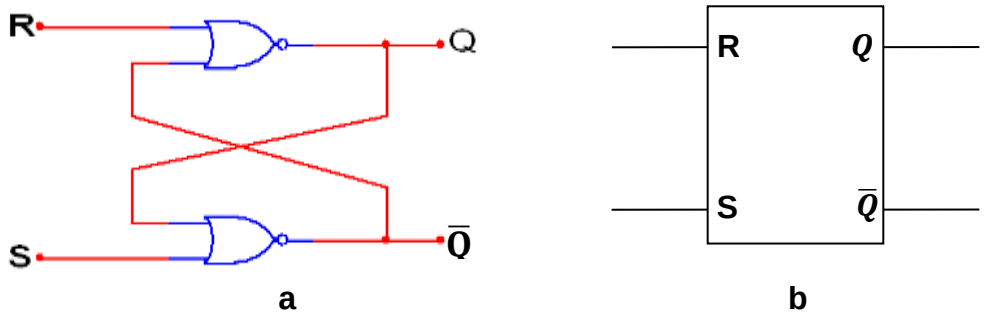
6.2.1 CIRCUITE BASCULANTE BISTABILE DE TIP RS

CBB de tip RS se obţin prin introducerea unei reacţii într-un sistem elementar de ordin 0, obţinând astfel un sistem de ordin 1.

1. **Circuitul basculant bistabil de tip RS ASINCRON**

Acest circuit datorită proprietăţilor sale de memorare este cunoscut şi sub numele de latch (zăvor) şi poate fi realizat cu 2 porţi SAU-NU (NOR) sau 2 porţi ŞI-NU (NAND). Circuitele RS asincrone sunt prevăzute cu 2 intrări R (Reset) readucere în 0 sau ştergere şi S (Set) fixare sau înscriere, precum şi cu 2 ieşiri complementare Q respectiv $\overline{Q}$.

În **figura 6.2.1** sunt reprezentate schema logică (a) şi simbolul (b) unui latch RS cu porţi NOR.



a **b**

Figura 6.2.1 Latch RS cu porţi NOR (SAU-NU)

Pentru a înţelege funcţionarea circuitului se studiază tabela de adevăr al circuitului prezentată mai jos (**Tabelul 6.2.1**).

Tabelul 6.2.1

Rn	Sn	Qn+1
0	0	Qn
1	0	0
0	1	1
1	1	X

Indice n – valoare logică prezentă

Indice n+1 – valoare logică viitoare

Cât timp ambele intrări sunt inactive R=S=0 ieşirile Q şi $\overline{Q}$ nu îşi schimbă stările logice în care se află (circuitul nu comută).

Când pe intrarea S (înscriere) se aplică un impuls pozitiv S=1 ieşirea Q trece în 1 logic iar ieşirea complementară $\overline{Q}$ trece în 0 logic (circuitul trece în starea 1).

Când pe intrarea R (ştergere) se aplică un impuls pozitiv R=1 ieşirea Q trece în 0 logic iar ieşirea complementară $\overline{Q}$ trece în 1 logic (circuitul trece în starea 0).

Dacă ambele intrări sunt active R=S=1 ieşirile Q şi $\overline{Q}$ se află într-o stare nedeterminată influenţată de procesul tehnologic de construcţie al circuitului.

În **figura 6.2.2** sunt reprezentate schema logică (a) şi simbolul (b) unui latch RS cu porţi NAND.

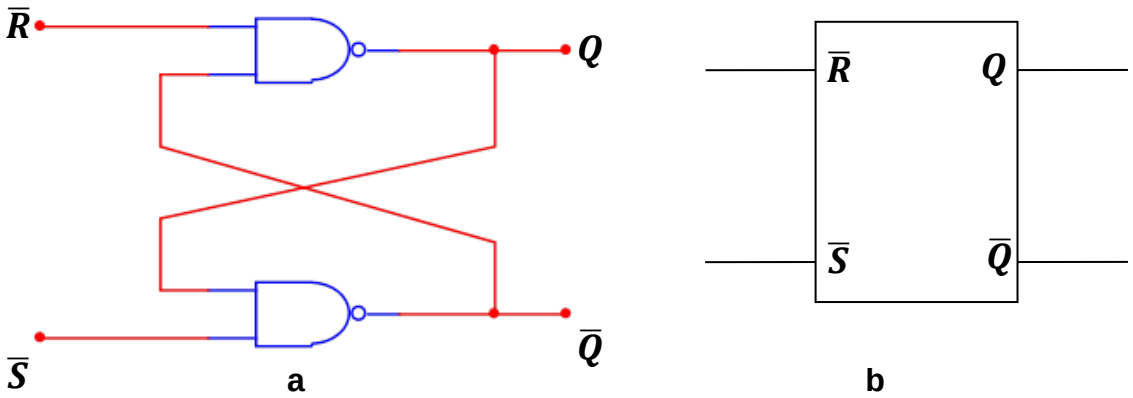


Figura 6.2.2 Latch RS cu porţi NAND (SI-NU)

Pentru a înţelege funcţionarea circuitului se studiază tabela de adevăr al circuitului prezentată mai jos (**Tabelul 6.2.2**).

Tabelul 6.2.2

$\overline{R}_n$	$\overline{S}_n$	Qn+1
1	1	Qn
0	1	0
1	0	1
0	0	X

Indice n – valoare logică prezentă

Indice n+1 – valoare logică viitoare

Cât timp ambele intrări sunt active $\overline{\mathbf{R}} = \overline{\mathbf{S}} = \mathbf{1}$ ieşirile Q şi $\overline{\mathbf{Q}}$ nu îşi schimbă stările logice în care se află (circuitul nu comută).

Când pe intrarea $\overline{S}$ (înscriere) se aplică un impuls pozitiv $\overline{S} = \mathbf{1}$ ieşirea Q trece în 0 logic iar ieşirea complementară $\overline{\mathbf{Q}}$ trece în 1 logic (circuitul trece în starea 0).

Când pe intrarea $\overline{\mathbf{R}}$ (ştergere) se aplică un impuls pozitiv $\overline{\mathbf{R}} = \mathbf{1}$ ieşirea Q trece în 1 logic iar ieşirea complementară $\overline{\mathbf{Q}}$ trece în 0 logic (circuitul trece în starea 1).

Dacă ambele intrări sunt inactive $\overline{\mathbf{R}} = \overline{\mathbf{S}} = \mathbf{0}$ ieşirile Q şi $\overline{\mathbf{Q}}$ se află într-o stare nedeterminată influenţată de procesul tehnologic de construcţie al circuitului.

2. Circuitul basculant bistabil de tip RS SINCRON

În majoritatea aplicaţiilor practice, este necesar ca procesele de comutare să aibă loc numai la anumite momente de timp bine determinate, adică să fie sincronizate cu alte semnale, iar comutarea să se realizeze numai după ce semnalele de intrare au devenit stabile. Pentru a satisface aceste condiţii se utilizează circuitele RS sincrone. Aceste circuite sunt cunoscute şi sub numele de bistabile şi spre deosebire de circuitele RS asincrone sunt prevăzute cu o intrare suplimentară de comandă numită intrare de tact şi pot fi realizate cu 4 porţi SAU-NU (NOR) sau 4 porţi ŞI-NU (NAND). Intrările de control ale circuitului RS sincron, sunt sincronizate cu intrarea de tact şi controlează modul în care se schimbă nivelurile logice ale ieşirilor doar în momentul în care semnalul de tact tranzitează de la un nivel logic la alt nivel logic pe frontul activ al impulsurilor dreptunghiulare aplicate la intrarea de tact (pentru $\overline{\mathbf{CLK}}$ frontul activ este frontul descrescător iar pentru $\mathbf{CLK}$ frontul activ este frontul crescător).

Circuitele basculante (CBB sincrone) comută pe front iar latch-urile (CBB asincrone) comută pe nivel.

În **figura 6.2.3** sunt reprezentate schema logică (a) şi simbolul (b) unui bistabil RS cu porţi NAND.

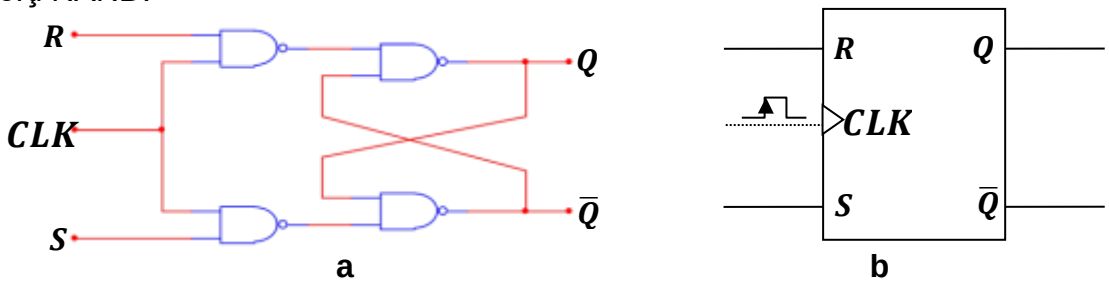


Figura 6.2.3 Bistabil RS cu porţi NAND (SI-NU)

Pentru a înţelege funcţionarea circuitului se studiază tabela de adevăr al circuitului prezentată mai jos (**Tabelul 6.2.3**).

Tabelul 6.2.3

CLK	Rn	Sn	Qn+1
↑	0	0	Qn
↑	1	0	0
↑	0	1	1
↑	1	1	X
0	X	X	Qn
1	0	↓	1
1	↓	0	0

Indice n – valoare logică prezentă

Indice n+1 – valoare logică viitoare

În **figura 6.2.4** sunt reprezentate schema logică (a) şi simbolul (b) unui bistabil RS cu porţi NOR.

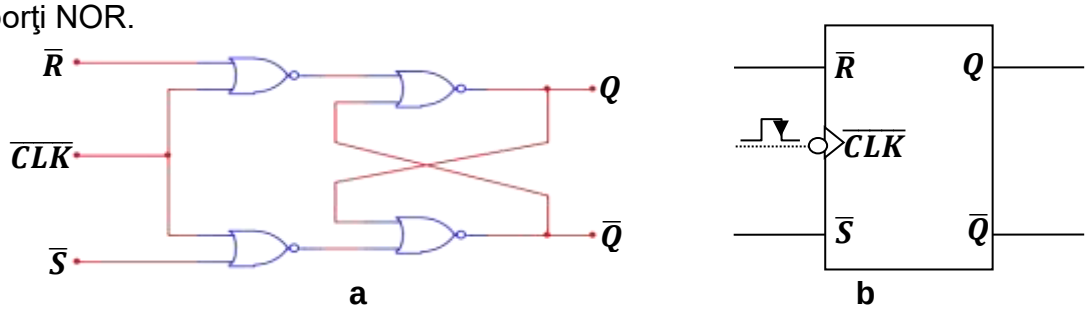


Figura 6.2.4 Bistabil RS cu porţi NOR (SAU-NU)

Pentru a înţelege funcţionarea circuitului se studiază tabela de adevăr al circuitului prezentată mai jos (**Tabelul 6.2.4**).

Tabelul 6.2.4

$\overline{CLK}$	$\overline{R}_n$	$\overline{S}_n$	Qn+1
↓	1	1	Qn
↓	1	0	1
↓	0	1	0
↓	0	0	X
1	X	X	Qn
0	1	↓	1
0	↓	1	0

Indice n – valoare logică prezentă

Indice n+1 – valoare logică viitoare

3. Circuitul basculant bistabil de tip RS MASTER - SLAVE

Acest circuit reprezintă a extensie a circuitului bistabil RS sincron realizat cu porţi NAND, si este format din două bistabile RS sincrone conectate ca în figura 6.2.5.

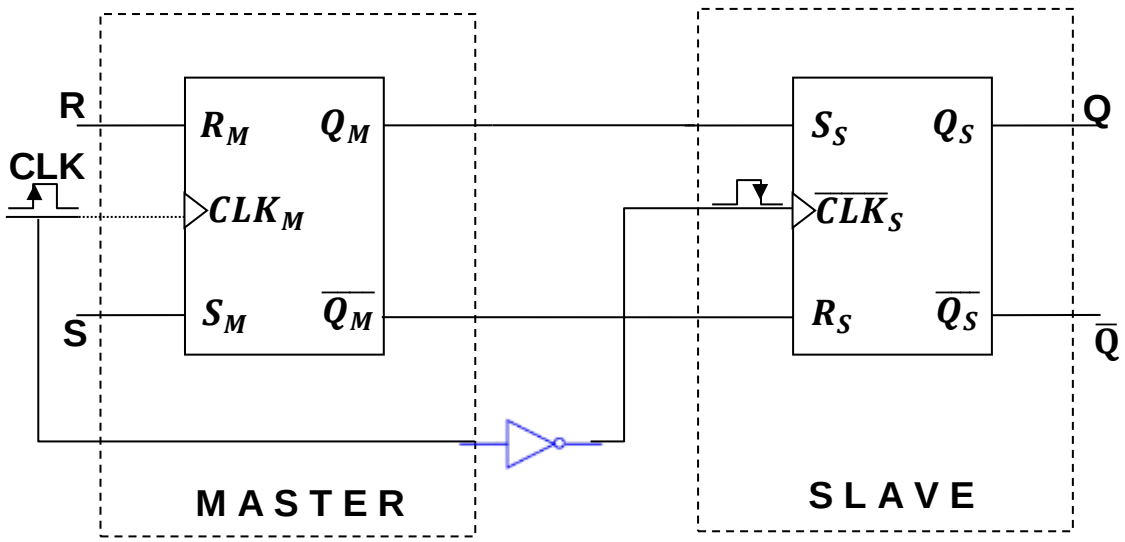


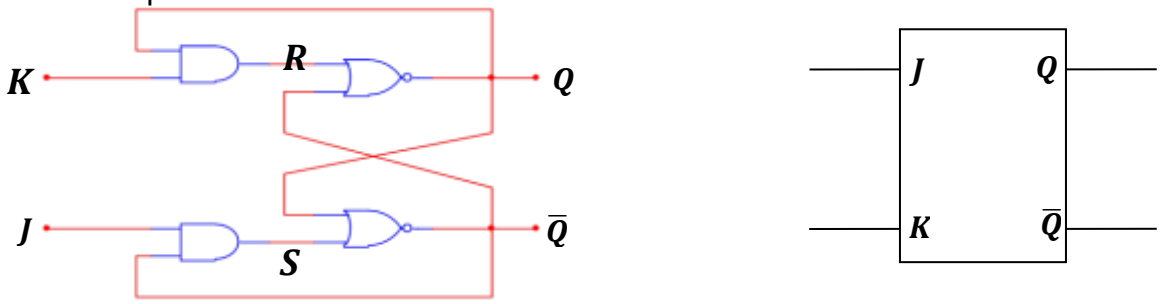
Figura 6.2.5 Bistabil RS de tip Master – Slave

6.2.2 CIRCUITE BASCULANTE BISTABILE DE TIP JK

Aceste circuite elimină starea de nedeterminare a ieşirilor unui circuit basculant când intrările au aceeaşi valoare logică $\mathbf{R} = \mathbf{S} = \mathbf{1}$ sau $\overline{\boldsymbol{R}} = \overline{\boldsymbol{S}} = \mathbf{0}$, deci spre deosebire de circuitele RS admit comenzi simultane la ambele intrări. Bistabilele JK se obţin din bistabilele RS prin introducerea unei bucle de reacţie de la ieşiri la intrări.
Comanda bistabilului J-K se face pe frontul crescător al impulsului de comandă. Deci ieşirea va comuta pe frontul negativ al impulsului de comandă, în funcţie de valorile lui J şi K de pe frontul crescător.

1. Circuitul basculant bistabil de tip JK ASINCRON

Circuitele basculante asincrone de tip JK sunt prevăzute cu 2 intrări J (SET) aducere circuitului din starea de repaus “0” în starea activă “1” şi K (RESET) Ştergerea sau readucerea circuitului din starea activă “1” în starea de repaus “0”, precum şi cu 2 ieşiri complementare Q respectiv $\overline{\mathbf{Q}}$.
În figura 6.2.6 sunt reprezentate schema logică (a) şi simbolul (b) unui bistabil asincron de tip JK.



a b

Figura 6.2.6 Bistabil asincron de tip JK

Pentru a înţelege funcţionarea circuitului se studiază tabela de adevăr al circuitului prezentată mai jos (**Tabelul 6.2.5**).

Tabelul 6.2.5

J	K	Qn+1
0	0	Qn
1	0	1
0	1	0
1	1	$\overline{\mathbf{Q}}_{\mathbf{n}}$ Basculare

Indice n – valoare logică prezentă

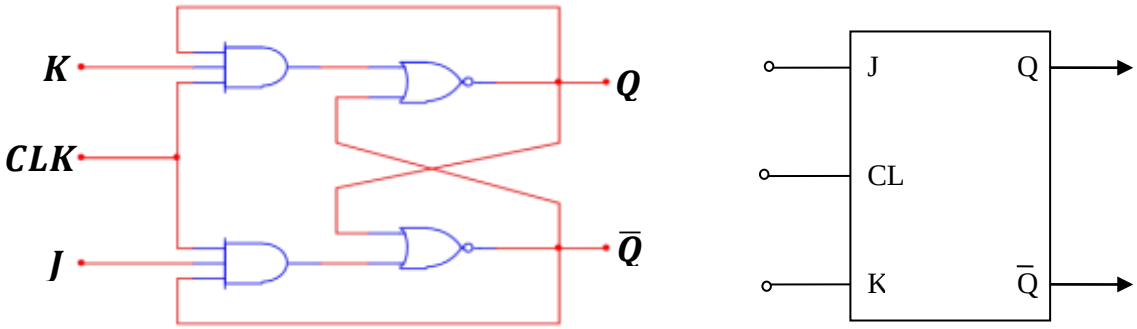
Indice n+1 – valoare logică viitoare

La acest tip de bistabil este necesar ca durata semnalului de comandă să fie mai mare decât timpul de propagare printr-o poartă şi mai mic decât timpul de propagare prin două porţi.

2. Circuitul basculant bistabil de tip JK SINCRON

Circuitele JK sincrone sunt prevăzute cu intrare suplimentară de comandă numită intrare de tact (CLK). Deoarece sunt circuite prevăzute cu reacţie, pentru a nu intra în auto-oscilaţie, impulsul de tact trebuie să fie foarte scurt. Durata impulsului trebuie să fie mai mică decât timpul de propagare a informaţiei de la intrare la ieşire.

În **figura 6.2.7** sunt reprezentate schema logică (a) şi simbolul (b) unui bistabil sincron de tip JK.



a **b**

Figura 6.2.7 Bistabil sincron de tip JK

Pentru a înţelege funcţionarea circuitului se studiază tabela de adevăr al circuitului prezentată mai jos (**Tabelul 6.2.6**).

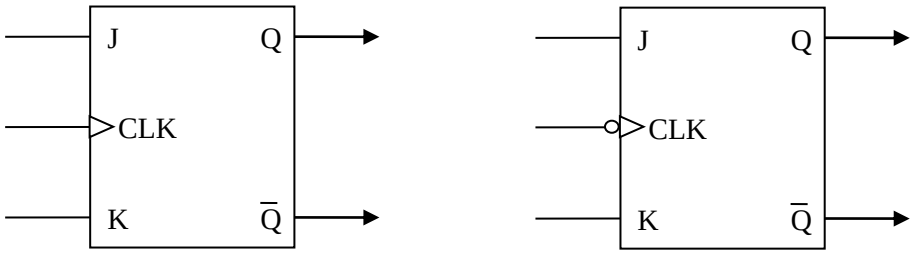
Tabelul 6.2.6

$\overline{CLK}$	J	K	Qn+1
↑	0	0	Qn
↑	1	0	1
↑	0	1	0
↑	1	1	$\overline{Q}_n$
0	X	X	Qn
0	↑	0	1
1	0	↑	0

Indice n – valoare logică prezentă

Indice n+1 – valoare logică viitoare

În **figura 6.2.8** sunt prezentate simbolurile circuitelor JK sincrone cu activare pe front pozitiv (a) şi pe front negativ (b).



a **b**

Figura 6.2.8 Simboluri bistabile sincrone de tip JK

3. Circuitul basculant bistabil de tip JK MASTER – SLAVE

Bistabilul JK Master – Slave este format din două latch-uri RS conectate în serie la care se realizează legături de reacţie de la ieşiri către intrări . Circuitul este prevăzut cu 2 intrări de date J şiK şi o intrare de tact CLK

În **figura 6.2.9** sunt prezentate schema logică (a) şi structura (b) bistabilului.

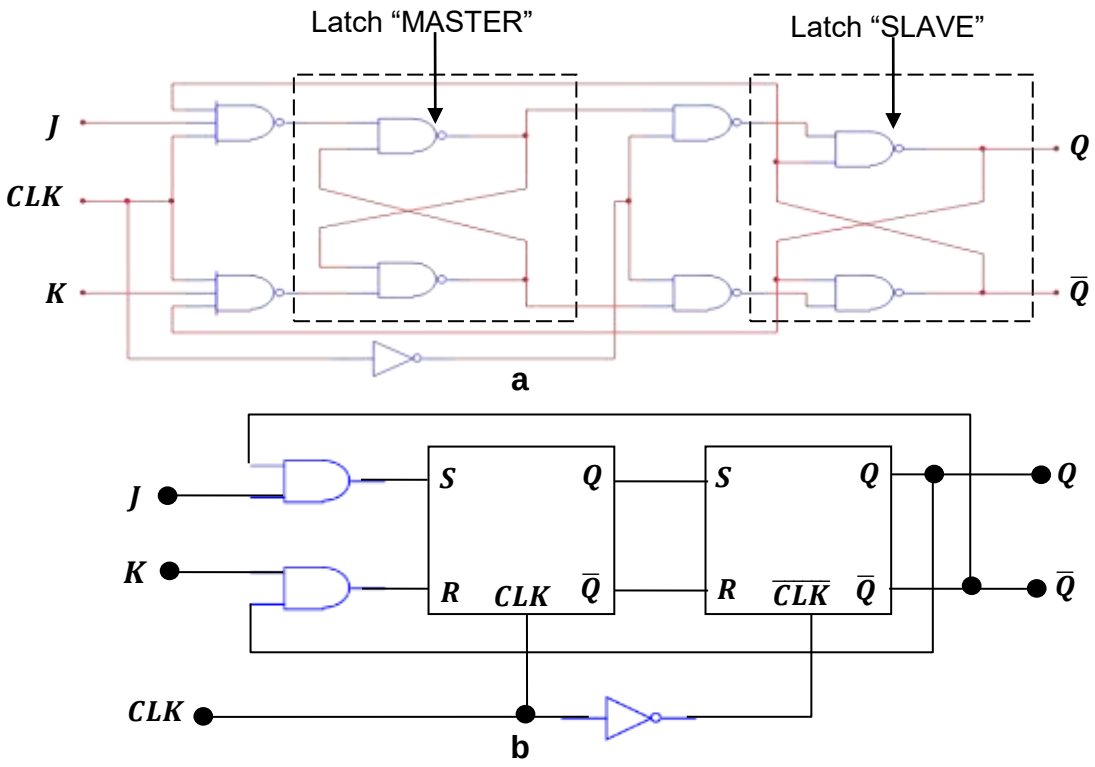


Figura 6.2.9 Schema logică şi structura bistabilului JK Master – Slave

În figura 6.2.10 sunt prezentate 2 exemple de circuite bistabile JK Master-Slave.

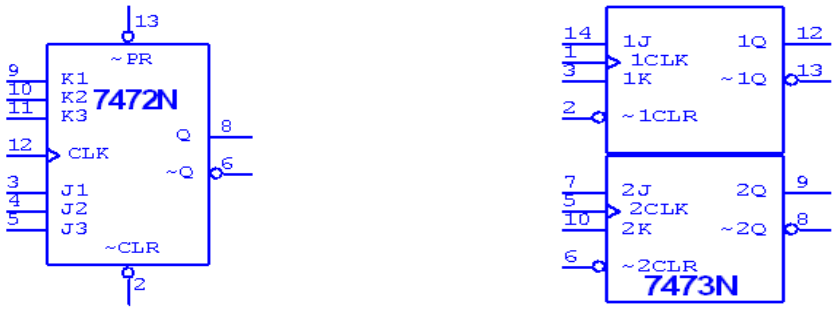


Figura 6.2.10 Exemple de circuite basculante bistabile JK

7472N – este un bistabil JK cu 3 perechi de intrări de date, care comută pe frontul descrescător şi este prevăzut cu 2 intrări asincrone $\overline{\mathbf{PR}}$ (Set) şi $\overline{\mathbf{CLR}}$ (Reset) pentru aducerea circuitului în starea 1 respectiv 0.

7473N – două bistabile JK care comută pe front descrescător, fiecare bistabil este prevăzut cu o intrare asincronă $\overline{\mathbf{CLR}}$ (Reset) pentru aducerea circuitului în 0.

6.2.3 CIRCUITE BASCULANTE BISTABILE DE TIP D

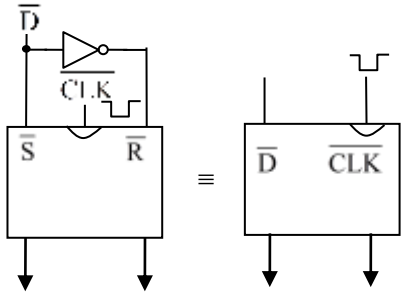
Circuitul basculant bistabil de tip D (Delay) se obţine dintr-un CBB de tip RS sau JK prin conectarea unei porţi inversoare între cele două intrări de date RS sau JK, în scopul eliminării stărilor nedeterminate. Prin ataşarea porţii inversoare între cele 2 intrări, acestea nu mai pot lua simultan valori identice, valorile lor vor fi mereu complementare.

În general un circuit bistabil de tip D este format din:

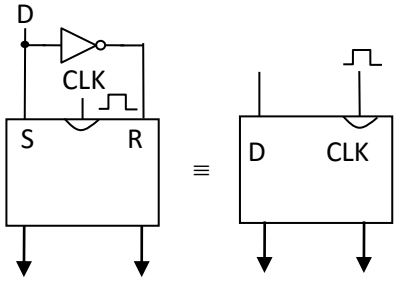
- intrare de date D (Delay)
- intrare de tact (CLK)
- 2 ieşiri complementare $\mathbf{Q}$ şi $\overline{\mathbf{Q}}$
- 2 intrări asincrone, pentru forţarea comutării circuitului într-o anumită stare: 1sau 0
  - Intrarea PR echivalentă cu SET (iniţializare) aduce circuitul în starea 1
  - Intrarea CLR echivalentă cu RESET (ştergere) aduce circuitul în starea 0

Intrările asincrone PR şi CLR sunt specifice CBB de tip D construite în varianta Master – Slave.

Circuitele basculante bistabile de tip D, pot fi realizate în varianta sincronă, asincronă şi Master-Slave.



a) comandat pe palierul inferior al CLK



b) comandat pe palierul superior al CLK

Figura 6.2.11 Circuite basculante bistabile de tip D sincrone

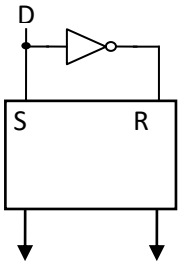


Figura 6.2.12 CBB – D asincron

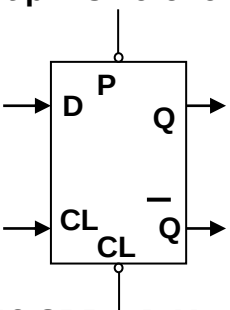


Figura 6.2.13 CBB – D Master-Slave

Circuitele basculante bistabile de tip D se utilizează cel mai frecvent la realizarea registrelor de deplasare serie, paralel, serie-paralel, care se vor studia în ***subcapitolul 6.3.***

6.2.4 CIRCUITE BASCULANTE BISTABILE DE TIP T

Circuitul basculant bistabil de tip T (toggle) reprezintă cel mai simplu automat şi se obţine dintr-un CBB de tip RS sau JK prin conectarea împreună a celor două intrări de date RS sau JK.
Bistabilul de tip T are o singură intrare de date T, o intrare de tact CLK şi două ieşiri complementare $\mathbf{Q}$ şi $\overline{\mathbf{Q}}$.
Familiile curente de circuite integrate nu conţin bistabili de tip T, ei se obţin din CBB J-K de tip Master-Slave prin conectarea intrărilor J şi K împreună. Prin conectarea împreună a intrărilor J şi K, Jn=Kn=Tn, bistabilul basculează dintr-o stare în alta la comanda impulsului de tact CLK.
Bistabilul de tip T, este forţat să funcţioneze doar în 2 situaţii:

- Jn=Kn=Tn = 0 $\Rightarrow \mathbf{Q_{n+1}} = \mathbf{Q_n}$
- Jn=Kn=Tn = 1 $\Rightarrow \mathbf{Q_{n+1}} = \overline{\mathbf{Q}}_\mathbf{n}$

Dacă intrarea bistabilului T este în permanenţă 1 logic, bistabilul basculează în starea opusă la fiecare impuls de tact, ceea ce înseamnă că tot la al doilea impuls revine în aceeaşi stare. Această proprietate recomandă utilizarea bistabilului T ca numărător (divizor) modulo doi, divizarea cu 2 a frecvenţei de pe intrarea de tact (**figura 6.2.14**).

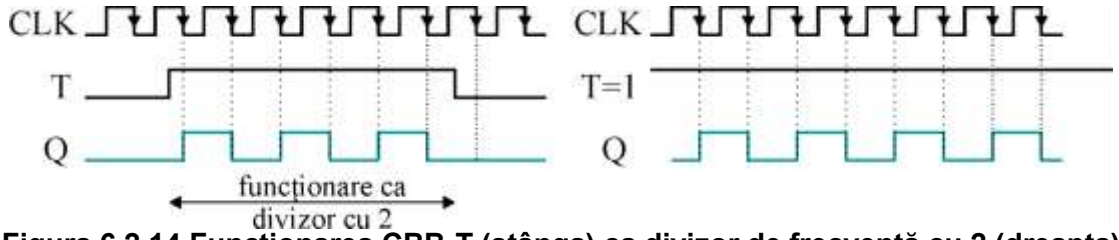


Figura 6.2.14 Funcţionarea CBB-T (stânga) ca divizor de frecvenţă cu 2 (dreapta)

Prin înserierea a n bistabile de tip T se obţine după fiecare bistabil o divizare a frecvenţei cu puterile crescătoare ale lui 2, astfel: 21, 22, 23,.....2n. Aceste circuite numite şi numărătoare se vor studia în ***subcapitolul 6.4***.
Funcţionarea bistabilului de tip T se deduce din **tabelul 6.2.7** şi **tabelul 6.2.8** iar simbolul bistabilului este prezentat în **figura 6.2.15**.

Tabelul 6.2.7

T	$\mathbf{Q_n}$	$\mathbf{Q_{n*1}}$
0	0	0
1	0	1
1	1	0
0	1	1

Tabelul 6.2.8

T	$\mathbf{Q_{n*1}}$
0	$\mathbf{Q_n}$
1	$\overline{\mathbf{Q}}_\mathbf{n}$

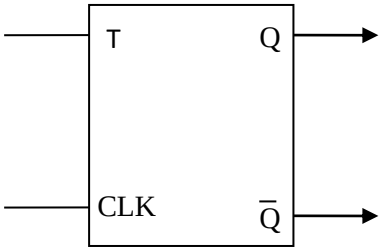


Figura 6.2.15 Simbolul CBB – T

6.3. NUMĂRĂTOARE

Numărătoarele – sunt circuite logice secvenţiale utilizate pentru contorizarea (numărarea şi memorarea) impulsurilor aplicate la intrările acestora. Numărătoarele nu au intrări de date, tranziţiile se efectuează după o anumită regulă într-o anumită ordine, fixate prin construcţia numărătorului, în ritmul unui semnal de tact.

Numărătoarele se realizează cu circuite basculante bistabile (celule de numărare) care stabilesc capacitate de numărare şi porţi logice care stabilesc modul corect în care numărătorul îşi schimbă stările în cadrul procesului de numărare.

Caracteristica principală a unui numărătoare este capacitatea de numărare adică numărul maxim de stări distincte ale numărătorului $\mathbf{N_{max}}$.

Numărul maxim de stări distincte şi stabile ale unui numărător format din n bistabile este $\mathbf{N_{max} = 2^n}$, deci numărătorul este **modulo $\mathbf{2^n}$**.

Deoarece ieşirile circuitelor bistabile indică numărul impulsurilor primite în mod binar, numărătoarele se mai numesc numărătoare binare şi pot fi utilizate şi ca divizoare de frecvenţă.

Numărătoarele binare se clasifică după următoarele criterii:

- ***După modul de conectare a bistabilelor de comandă:***
  - ***numărătoare asincrone*** – bistabilele sunt conectare în serie, intrarea de tact CLK a unui bistabil este conectată la ieşirea Q a bistabilului anterior, basculare a unui bistabil se face numai după basculare a bistabilului anterior;
  - ***numărătoare sincrone*** – bistabilele sunt conectate în paralel, intrările de tact CLK a tuturor bistabilelor sunt conectate împreună, basculare a tuturor bistabililor se face în acelaşi moment.
- ***După sensul numărării***:
  - ***numărătoare directe*** – fiecare impuls prezent la intrarea numărătorului creşte conţinutul acestuia cu o unitate (numără în sens crescător);
  - ***numărătoare inverse*** – fiecare impuls prezent la intrarea numărătorului scade conţinutul acestuia cu o unitate (numără în sens descrescător);
  - ***numărătoare reversibile*** – efectuează numărarea în ambele sensuri în funcţie de comanda primită din exterior.
- ***După codul de numărare:***
  - numărătoare binare – $m=2^n$;
  - numărătoare decadice – $m=10$.

6.3.1 NUMĂRĂTOARE ASINCRONE

Numărătoarele asincrone pot fi realizate cu circuite basculante bistabile asincrone şi sincrone de tip T, care sunt conectate în cascadă (ieşirea fiecărui bistabil este conectată la intrarea de tact al următorului). Bistabilele nu comută simultan la acţionarea unui semnal de tact comun, ci ieşirea unui bistabil comandă comutarea următorului bistabil.

1. NUMĂRĂTORUL ASINCRON BINAR DIRECT

Pentru a înţelege funcţionarea unui numărător asincron se prezintă circuitul integrat 74LS93 care conţine un numărător cu 4 celule basculante bistabile master-slave (acest circuit este echivalent cu CDB 493).

Deoarece are 4 celule bistabile, numărătorul are 16 stări distincte ($m=2^4=16$), deci este un numărător modulo 16.

În **figura 6.3.1** este prezentată schema bloc a capsulei circuitului integrat 74LS93.

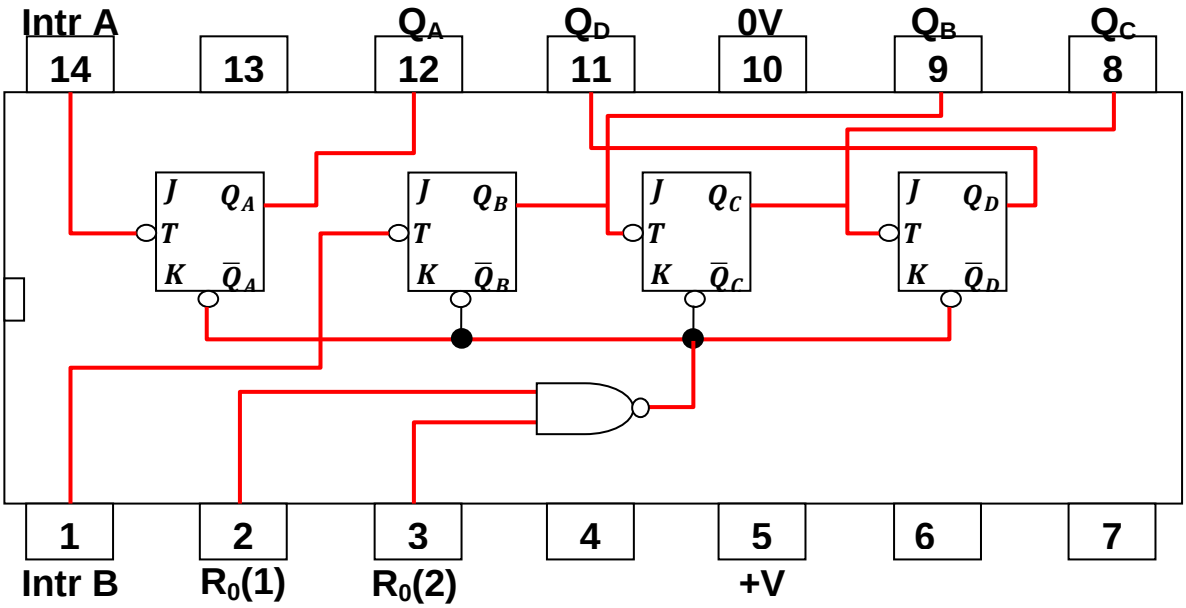


Figura 6.3.1 Numărătorul asincron 74LS93 (CDB 493)

Bistabilii B, C, D sunt interconectaţi intern (în serie) şi formează un divizor cu 8.
Bistabilul A este un divizor cu 2.

Intr A – reprezintă intrarea de tact în divizorul cu 2 (bistabilul A).

Intr B - reprezintă intrarea de tact în divizorul cu 8 (bistabili B, C, D).

R0(1) şi R0(2) – sunt intrări pentru resetarea numărătorului (aducerea la 0). Când ambele intrări sunt în 1 logic numărătorul se resetează şi începe din nou numărarea.

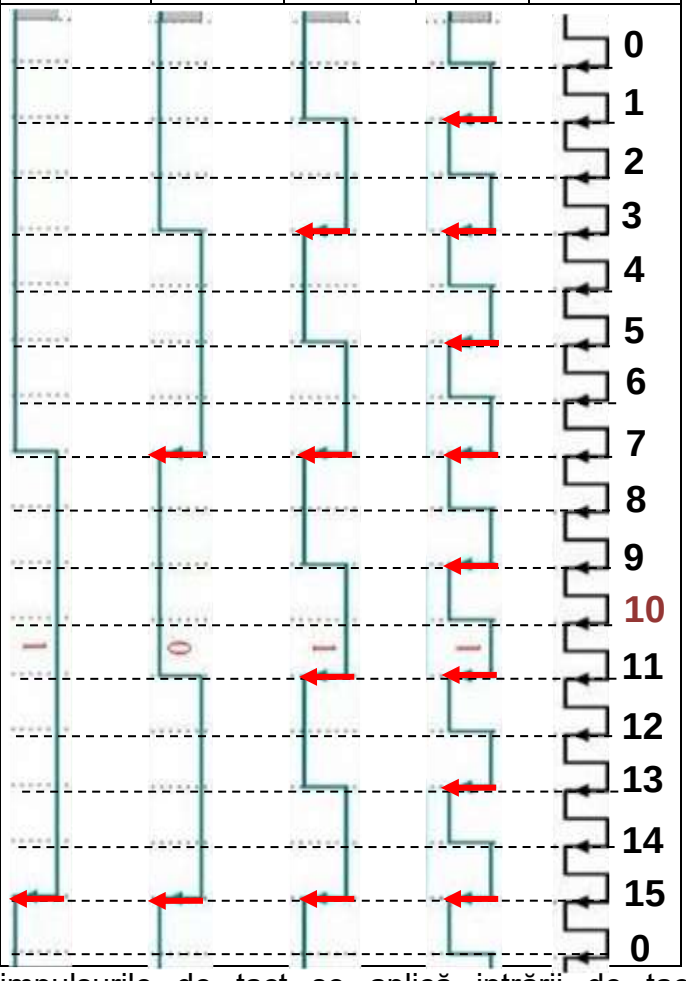
Q_A, Q_B, Q_C, Q_D – ieşirile celulelor bistabile.

Dacă se interconectează extern bistabilul A cu bistabili B, C, D (pin 12 cu pin 1) se obţine un numărător modulo 16 (un divizor prin 16).

Pentru a înţelege funcţionarea circuitului se studiază tabelul de adevăr al circuitului prezentat mai jos (Tabelul 6.3.1)

Tabelul 6.3.1

STĂRI LOGICE					FORME DE UNDĂ				
NR.	IEŞIRI				Q_D	Q_C	Q_B	Q_A	Intrare
	Q_D	Q_C	Q_B	Q_A					
0	0	0	0	0					
1	0	0	0	1					
2	0	0	1	0					
3	0	0	1	1					
4	0	1	0	0					
5	0	1	0	1					
6	0	1	1	0					
7	0	1	1	1					
8	1	0	0	0					
9	1	0	0	1					
10	1	0	1	0					
11	1	0	1	1					
12	1	1	0	0					
13	1	1	0	1					
14	1	1	1	0					
15	1	1	1	1					
16	0	0	0	0					



Pentru realizarea numărătorului, impulsurile de tact se aplică intrării de tact bistabilului asociat bitului de rang inferior Q_A (în acest caz pe Intr A – pin 14).

La fiecare comutare din 1 în 0 (pe frontul descrescător al impulsurilor) a bistabilului Q_A se obţine un front negativ care comandă comutarea bistabilului următor Q_B. Când bistabilul Q_B comută din 1 în 0 se obţine un front negativ care comandă comutarea bistabilului următor Q_C. Când bistabilul Q_C comută din 1 în 0 se obţine un front negativ care comandă comutarea bistabilului următor Q_D.

Exemplu: după comutarea celui de-al 11-lea impuls de tact (notat cu 10) din 1 în 0 ieşirile bistabililor sunt $Q_DQ_CQ_BQ_A$**=1011**, care este tocmai corespondentul binar al numărului zecimal 11.

În **figura 6.3.2** este prezentată schema unei aplicaţii cu numărătorul 74LS93N.

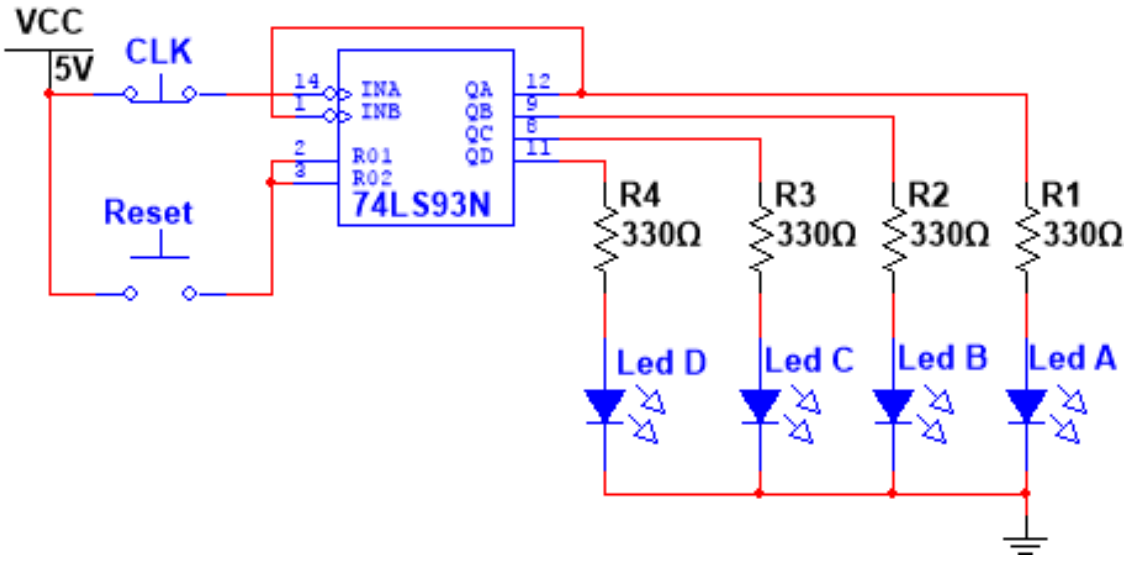


Figura 6.3.2 Aplicaţie cu numărătorul asincron 74LS93

La fiecare activare a butonului ***CLK*** se trimite manual câte un impuls spre numărător. Butonul ***Reset*** se utilizează pentru aducerea la ***0*** a numărătorului (resetare).
Led-urile de la ieşirile numărătorului vor lumina conform **tabelului 6.3.1** în funcţie de numărul impulsului dat de butonul CLK.

Exemplu: la impulsul cu numărul 10 luminează ledurile Led B şi Led D.

Numărătorul asincron 74LS93 poate fi utilizat şi ca divizor de frecvenţă. Dacă la intrarea de tact Intr A este frecvenţa $\boldsymbol{f}$, la ieşirile numărătorului vor fi următoarele frecvenţe:

- La ieşirea $\mathbf{Q_A}$ va fi frecvenţa $\frac{\mathbf{f}}{\mathbf{2}}$
- La ieşirea $\mathbf{Q_B}$ va fi frecvenţa $\frac{\mathbf{f}}{\mathbf{4}}$
- La ieşirea $\mathbf{Q_C}$ va fi frecvenţa $\frac{\mathbf{f}}{\mathbf{8}}$
- La ieşirea $\mathbf{Q_D}$ va fi frecvenţa $\frac{\mathbf{f}}{\mathbf{16}}$ (dacă se conectează pin 12 cu pin 1).

În funcţie de modul de conectare a intrărilor de aducere la 0 a numărătorului $\mathbf{R_0(1)}$ şi $\mathbf{R_0(2)}$ se poate realiza orice divizor printr-un număr întreg cuprins între 1 şi 16.

Exemple:

Pentru a obţine un numărător divizor prin 7, conexiunile se realizează în aşa fel încât în starea 7 cele 2 intrări de aducere la 0 să capete simultan nivelul logic 1. Din tabelul de adevăr se observă ca starea 7 se caracterizează prin nivel logic 1 la ieşirile $\mathbf{Q_A}$, $\mathbf{Q_B}$, $\mathbf{Q_C}$. În această situaţie ieşirea $\mathbf{Q_A}$ se conectează la $\mathbf{R_0(1)}$ iar ieşirile $\mathbf{Q_B}$ şi **Qc** se conectează printr-o poartă ŞI la $\mathbf{R_0(2)}$.
Pentru a obţine un numărător divizor prin 9, ieşirea $\mathbf{Q_A}$ se conectează la $\mathbf{R_0(1)}$ iar ieşirea $\mathbf{Q_D}$ se conectează la $\mathbf{R_0(2)}$.
Pentru a obţine un numărător divizor prin 12, ieşirea $\mathbf{Q_C}$ se conectează la $\mathbf{R_0(1)}$ iar ieşirea $\mathbf{Q_D}$ se conectează la $\mathbf{R_0(2)}$.

2. NUMĂRĂTORUL ASINCRON BINAR INVERS

Numărătorul invers îşi micşorează conţinutul cu câte o unitate la fiecare impuls de tact. În acest scop semnalul de tact (CLK) a bistabilului următor se conectează la ieşirea negată a bistabilului precedent ($\overline{Q}$)(**figura 6.3.3**).

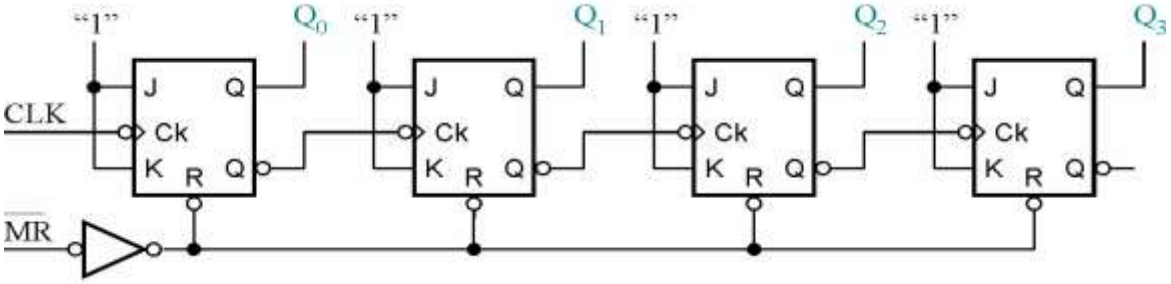


Figura 6.3.3 Numărător binar asincron invers

Numărătorul din **figura 6.2.3** conţine 4 celule bistabile, deci are o capacitate de numărare de 16 impulsuri.

Primul impuls de tact aplicat la intrare basculează toate celulele în starea 1, deci conţinutul numărătorului va fi 1111. La fiecare impuls de tact conţinutul descreşte cu o unitate, astfel că după 16 impulsuri starea numărătorului va fi 0000.

Numărătorul realizează un ciclu de numărare inversă

$15 \rightarrow 14 \rightarrow 13 \rightarrow 12 \rightarrow 11 \rightarrow 10 \rightarrow 9 \rightarrow 8 \rightarrow 7 \rightarrow 6 \rightarrow 5 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 1 \rightarrow 0$

Funcţionarea parţială a numărătorului asincron invers se poate deduce din tabelul 6.3.2

Tabelul 6.3.2

Corespondent zecimal	Intrare tact CLK	IEŞIRI			
		Q_3	**Q_2**	**Q_1**	**Q_0**
0	Valoare iniţială	0	0	0	0
15	1	1	1	1	1
14	2	1	1	1	0
13	3	1	1	0	1
12	4	1	1	0	0
11	5	1	0	1	1
.....	.....	.....	.....	.....	.....
1	15	0	0	0	1

3. NUMĂRĂTORUL ASINCRON DECADIC

Pentru a înţelege funcţionarea unui numărător asincron decadic se prezintă circuitul integrat 74LS90 care conţine un numărător cu 4 celule basculante bistabile master-slave (acest circuit este echivalent cu CDB 490).

În **figura 6.3.4** este prezentată schema bloc a capsulei circuitului integrat 74LS90.

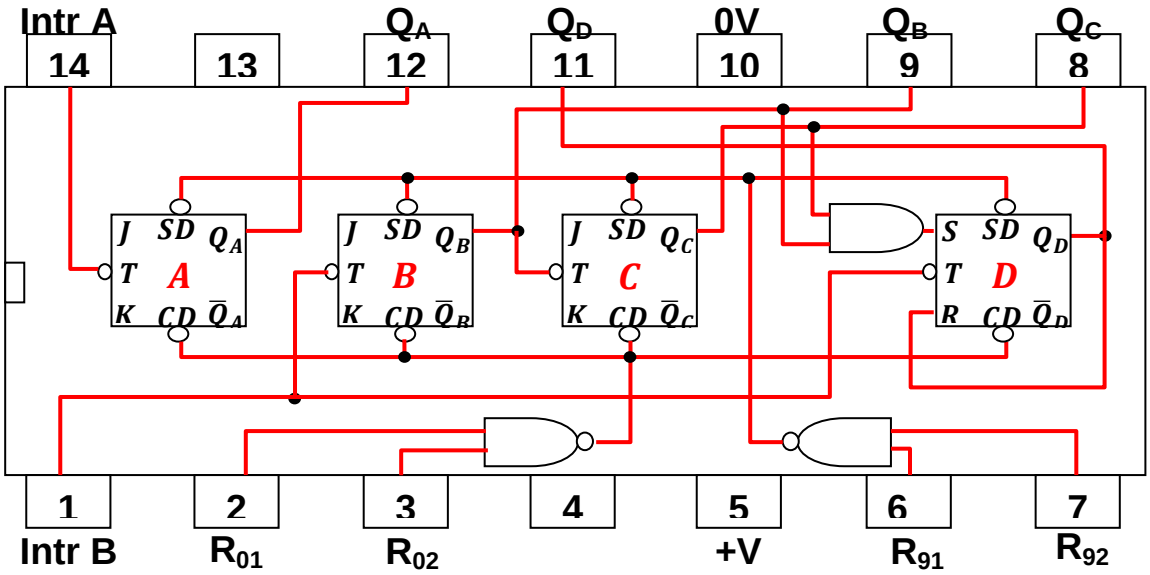


Figura 6.3.4 Numărătorul asincron decadic 74LS90 (CDB 490)

Primul bistabil (A) este un divizor cu 2. Bistabili B, C, D formează un divizor cu 5. Cele două grupe interconectate formează un divizor cu 10 (se conectează ieşirea **Q_A** cu intrarea **Intr B**, pin 12 cu pin 1).

Intr A este intrarea de tact pentru celula divizoare cu 2 iar Intr B este intrarea de tact pentru celula divizoare cu 5.

R_{01} şi **R_{02}** sunt intrări de ştergere, pentru aducerea numărătorului în starea 0.

R_{91} şi **R_{92}** sunt intrări de iniţializare pentru numărare inversă, pentru aducerea numărătorului în starea 9.

Intrările **R_{91}** şi **R_{92}** sunt prioritare faţă de intrările **R_{01}** şi **R_{02}**.

Q_A, Q_B, Q_C, Q_D sunt ieşirile numărătorului.

Numărătorul decadic are 10 stări distincte, deci capacitatea de numărare de 10 impulsuri.

Numărătorul asincron decadic funcţionează la fel cu cel asincron binar până la impulsul al nouălea, când starea circuitului este 1001. La cel de-al 10-lea impuls de tact, datorită modului de interconectare a celor patru bistabile, starea circuitului nu va fi 1010 ci 0000, deci circuitul are 10 stări distincte.

6.3.2 NUMĂRĂTOARE SINCRONE

Această categorie de numărătoare asigură funcţionarea la frecvenţe mult mai mari decât în cazul numărătoarelor asincrone deoarece impulsurile de tact sunt aplicate simultan la toate celulele bistabile care vor comuta în acelaşi timp. În acest mod sunt eliminate întârzierile cumulative datorită comutării succesive a celulelor bistabile. Constructiv sunt mai complicate decât numărătoarele asincrone.

Pentru a înţelege funcţionarea unui numărător sincron se prezintă circuitul integrat 74LS192 care conţine un numărător cu 4 celule basculante bistabile master-slave (acest circuit este echivalent cu CDB 4192).

Circuitul integrat 74LS192 este un numărător sincron decadic, reversibil de 4 biţi cu posibilitate de încărcare paralelă (**figura 6.3.5**).

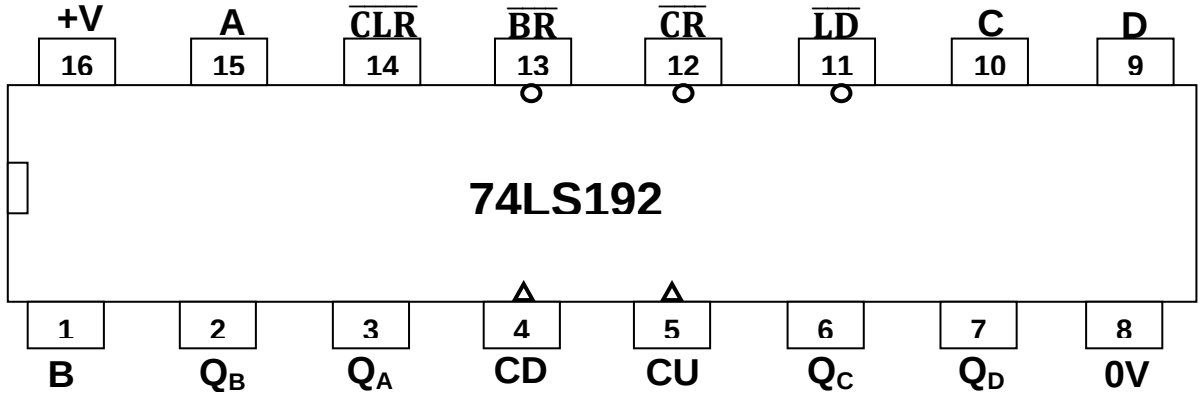


Figura 6.3.5 Numărătorul sincron decadic 74LS192 (CDB 4192)

INTRĂRI:

- Intrări de tact pentru:
  - Numărare directă CU (5);
  - Numărare inversă CD (4);
- Intrări de date (încărcare paralelă):
  - A (15), B (1), C (10), D (9);
- Intrări comandă paralelă:
  - Încărcare date $\overline{\mathbf{LD}}$ (11);
  - Ştergere date $\overline{\mathbf{CLR}}$ (14).

IEŞIRI:

- Ieşiri de date:
  - QA (3), QB (2), QC (6), QD (7);
- Ieşiri caracteristice numărării:
  - Ieşire caracteristică numărării directe – ieşire de transport $\overline{\mathbf{CR}}$ (12)
  - Ieşire caracteristică numărării inverse – ieşire de împrumut $\overline{\mathbf{BR}}$ (13)

În **figura 6.3.6** sunt prezentate formele de undă care descriu funcţionarea numărătorului

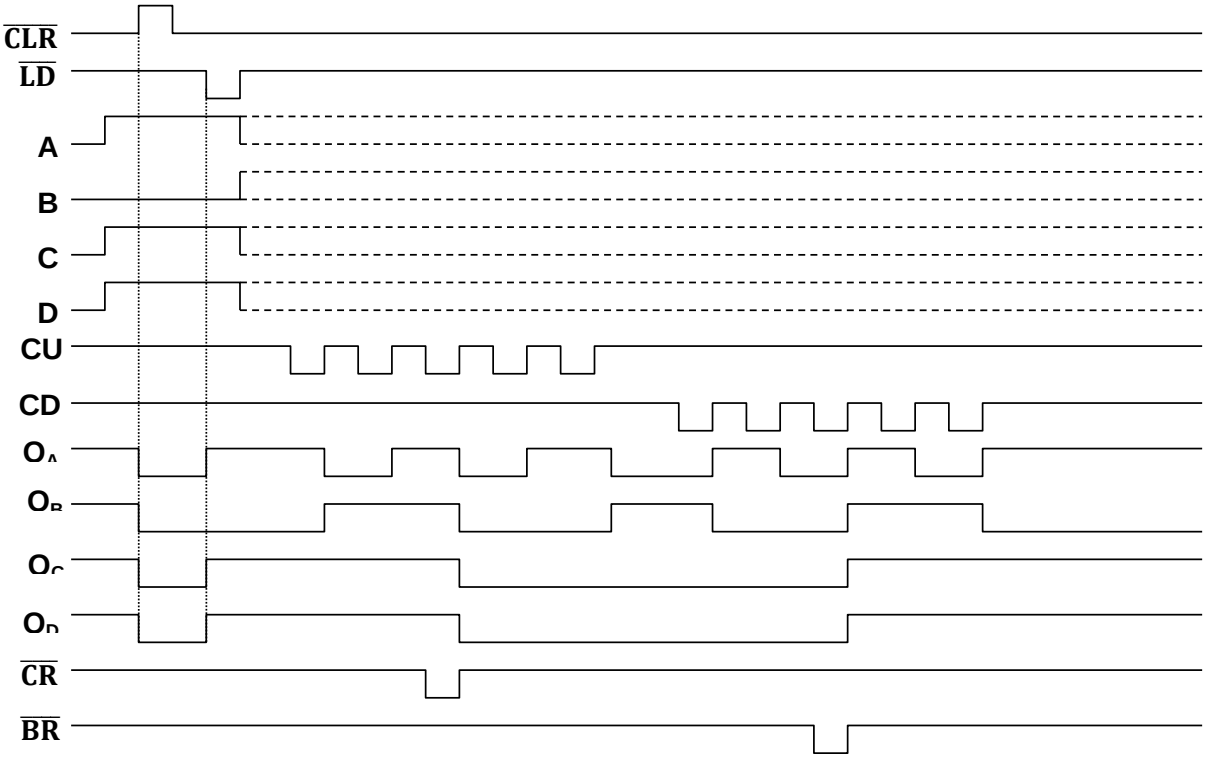


Figura 6.3.6 Forme de undă numărător sincron 74LS192 (CDB 4192)

Sensul de numărare se stabileşte de intrarea pe care se aplică impulsurile de numărat. În acest timp cealaltă intrare de tact care nu se utilizează se va conecta la nivelul 1 logic (+V). Bascularea bistabililor interni are loc pe frontul crescător al semnalului de tact (tranziţia din 0 în 1).

Intrarea $\overline{\mathbf{LD}}$ (Load) se utilizează pentru încărcarea paralelă a datelor iar $\overline{\mathbf{CLR}}$ (Clear) se utilizează pentru ştergerea acestor date. Dacă $\overline{\mathbf{LD}} = \mathbf{0}$ se validează operaţia de încărcare paralelă, independent de semnalul de tact şi de starea numărătorului. Pentru numărare intrarea $\overline{\mathbf{LD}}$ se conectează în 1 logic. Pentru ştergere se aplică un impuls pozitiv, 1 logic, pe intrarea $\overline{\mathbf{CLR}}$.

Pentru conectarea mai multor numărătoare în serie (pentru a stoca un număr mai mare de impulsuri) se utilizează ieşirile $\overline{\mathbf{CR}}$ (Carry) şi $\overline{\mathbf{BR}}$ (Barrow).

$\overline{\mathbf{CR}}$ trece în starea 0 logic când, la numărătoarea directă, numărătorul a atins numărul maxim de impulsuri care poate să le stocheze (1111).

$\overline{\mathbf{BR}}$ trece în starea 0 logic când, la numărătoarea inversă, numărătorul a ajuns la 0000.

O secvenţă de numărare mai scurtă se obţine prin conectarea la intrarea $\overline{\mathbf{LD}}$ a ieşirii $\overline{\mathbf{CR}}$ la numărarea directă sau a ieşirii $\overline{\mathbf{BR}}$ la numărarea inversă.

6.3.3 APLICAŢII ALE NUMĂRĂTOARELOR

În **figura 6.3.8** este prezentată schema unei aplicaţii cu un numărător BCD şi un decodificator 7 segmente realizată cu simulatorul MULTISIM.

U1 (LM 555) – este un generator de impulsuri dreptunghiulare.

Frecvenţa impulsurilor este vată da valorile componentelor R1 – P – C1. Prin modificarea valorii potenţiometrului P se modifică frecvenţa impulsurilor.

Ieşirea generatorului de impulsuri OUT (PIN 3) este conectată la intrarea de tact a numărătorului CLK (PIN 15).

U2 (CD4510) - este un numărător sincron BCD reversibil.

Acest circuit integrat numără impulsurile furnizate de generatorul de impulsuri U1 la intrarea CLK (15), iar rezultatul este furnizat la ieşirile **Q_1, Q_2, Q_3, Q_4** în cod BCD.

În **figura 6.3.7** este prezentată funcţionarea numărătorului sincron BCD reversibil

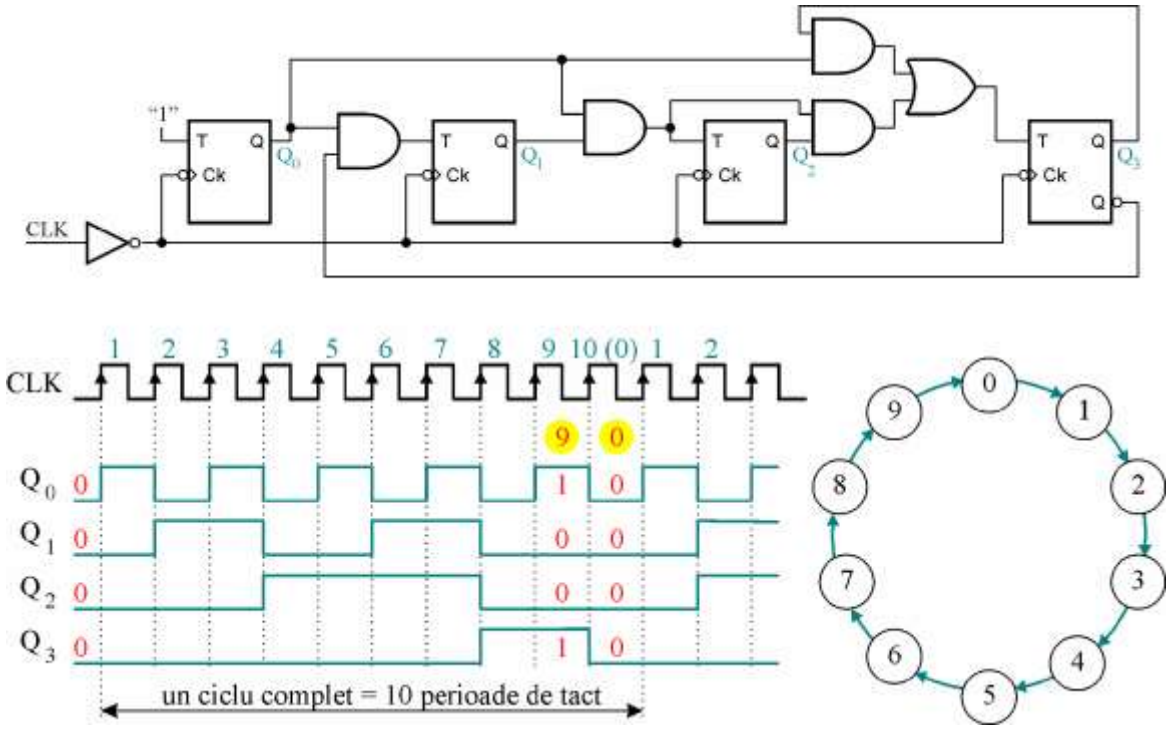


Figura 6.3.7 Numărător sincron BCD

U3 (CD4511) - este un decodificator BCD - 7segmente.

Acesta acceptă la intrare un cod BCD furnizat de numărătorul U2 şi furnizează la ieşire comenzi pentru afişajul 7 segmente .

U4 – este un afişaj 7 segmente cu catodul comun.

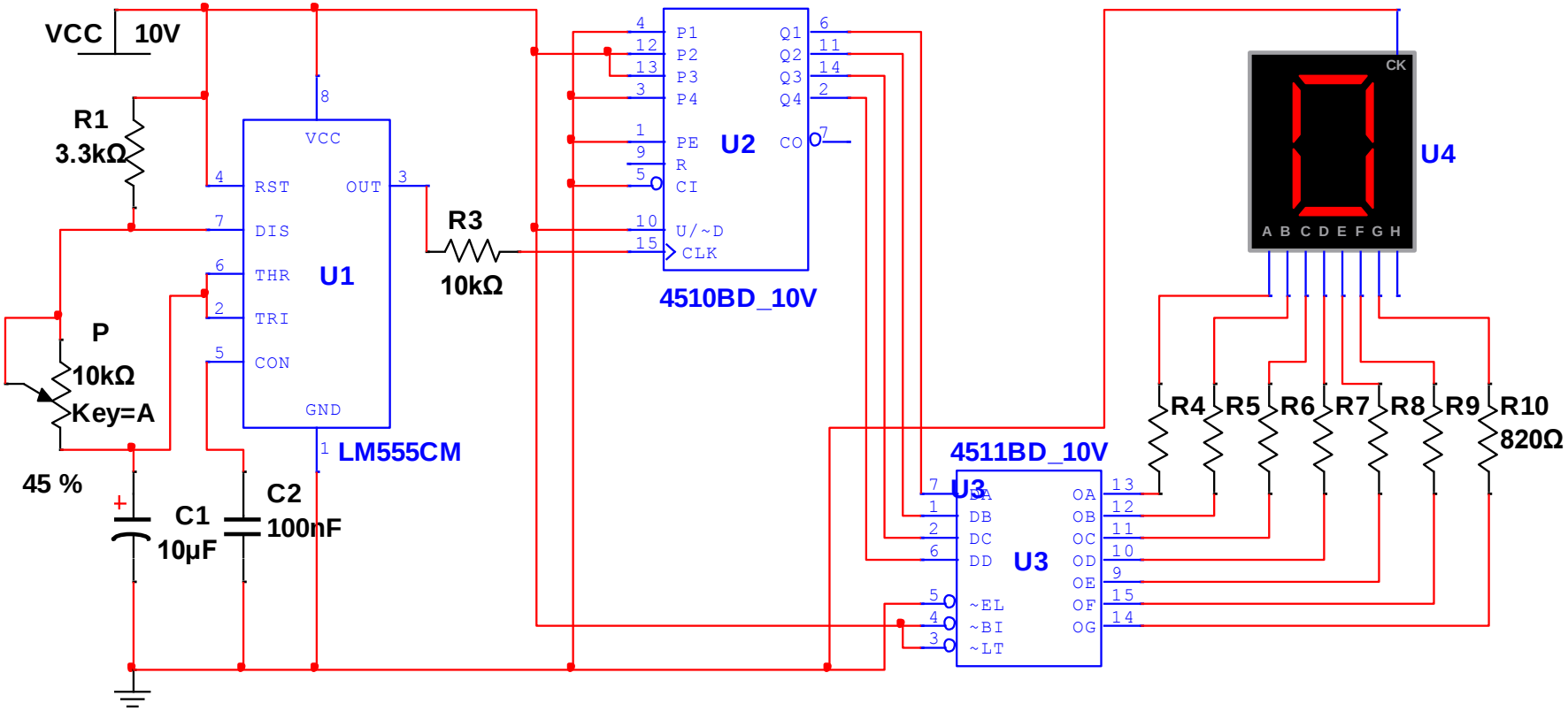


Figura 6.3.8 Numărător BCD cu afişaj 7 segmente

6.4. REGISTRE

Registrele – sunt circuite logice secvenţiale care primesc, stochează şi transferă informaţii sub formă binară. Un registru este format din mai multe celule bistabile de tip RS, JK sau D şi permite memorarea şi/sau deplasarea informaţiei la comanda impulsurilor de tact. Un registru care conţine n celule bistabile are o capacitate de n biţi. Registrele pot fi considerate memorii rapide de mici dimensiuni.

La un registru se definesc următoarele operaţii:

- Înscrierea – introducerea datelor în registru care se poate face:
  - Serial – bit după bit, toţi biţii cuvântului de n biţi;
  - Paralel – cei n biţi se scriu simultan în registru;
- Citirea – extragerea datelor din registru care se poate face:
  - Serial – bit după bit;
  - Paralel – toţi biţii simultan;
- Deplasarea datelor în registru se poate face:
  - Deplasarea la dreapta;
  - Deplasarea la stânga;
  - Deplasarea în ambele sensuri;
- Ştergerea – aducerea tuturor registrelor în starea 0.

După modul de înscriere/ citire se disting patru tipuri de registre:

- registru cu înscriere serie şi citire serie - SISO;
- registru cu înscriere serie şi citire paralel – SIPO;
- registru cu înscriere paralel şi citire serie – PISO;
- registru cu înscriere serie şi citire paralel – PIPO.

Un registru care îndeplineşte două sau mai multe funcţii din cele 4 prezentate mai sus se numeşte registru universal.

În tehnologie TTL se fabrica următoarele tipuri principale de registre:
74LS164, 74LS165, 74LS166, 74LS194, 74LS195, 74LS95, 74LS174, 74LS374, 74LS574, 74LS594, 74LS595.

În tehnologie CMOS se fabrica următoarele tipuri principale de registre:
4006, 4014, 4015, 4021, 4031, 4035, 4042, 4076, 4094, 4517, 4731, 40104

În **tabelul 6.4.1** sunt prezentate principalele tipuri de registre.

Tabelul 6.4.1

TIP	Comută pe	TTL		CMOS		OBSERVAŢII
		Cod	n	Cod	n	
SISO	Front ↑			4006	18	Configurabil 2x4,5,8,9 sau 1x10,12,13,14,16,18
	Front ↑			4031	64	1 registru în capsulă
	Front ↑			4517	64	2 registre în capsulă, prize la 16,32,48,64
	Front ↑			4731	64	4 registre în capsulă
SIPO	Front ↑	74164	8			
	Front ↑			4015	4	2 registre de 4 biţi în capsulă
PIPO	Front ↑	74174	6			
	Front ↑	74374	8			3 stări
	Front ↑	74574	8			Idem 74374, altă dispunere pini
	Front ↑			4042	4	Latch D cu controlul polarităţii tactului
	Front ↑			4076	4	3 stări
PISO	Front ↑	74165	8			Intrări J nK
Combinate	Front ↑	74166	8			PISO, SISO
	Front ↑	74195	8			Intrări J nK
	Front ↑	74594	8			SISO, PIPO, 2 intrări de tact
	Front ↑	74595	8			SISO, PIPO, 2 intrări de tact, 3 stări
	Front ↑	74597	8			PIPO, SIPO, PISO
	Front ↑			4014	8	PISO, SISO
	Front ↑			4021	8	PISO, SISO
	Front ↑			4035	4	PIPO, SISO, bidirecţional J nK
	Front ↑			4094	8	SISO, SIPO, 3 stări
Universale	Front ↓	7495	4			
	Front ↑	74194	4			
	Front ↑			40104	4	3 stări

1. Registru cu înscriere serie şi citire serie (SISO)

Acest tip de registru este format din n bistabile de tip D şi are structura din **figura 6.4.1**. Ieşirea Q a bistabilului k este conectată la intrarea D a bistabilului k+1. Registrul are o singură intrare pentru înscrierea serie şi o singură ieşire pentru citirea serie a datelor.

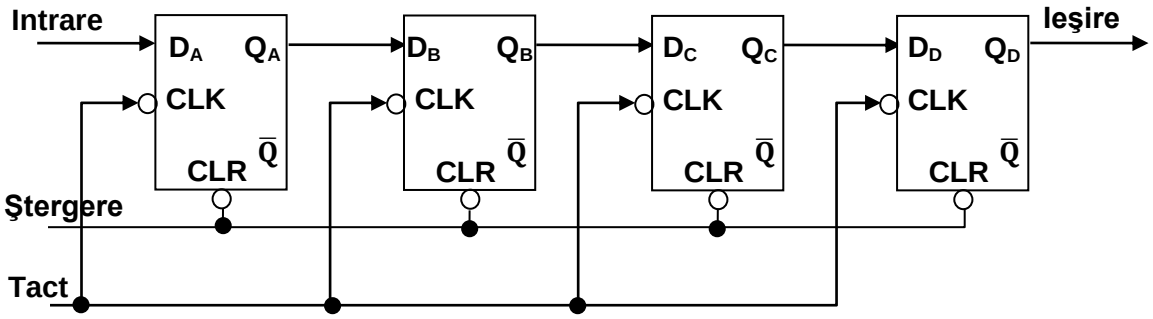


Figura 6.4.1 Schemă principiu registru SISO de 4 biţi

Funcţionarea acestui registru pentru cuvântul 1101 se poate urmări în **tabelul 6.4.2**

Tabelul 6.4.2

Tact	Q_A	Q_B	Q_C	Q_D	
1	**1**	0	0	0	
2	**0**	**1**	0	0	
3	**1**	**0**	**1**	0	
4	**1**	**1**	**0**	**1**	
5	0	**1**	**1**	**0**	**1**
6	0	0	**1**	**1**	**0,1**
7	0	0	0	**1**	**1,0,1**
8	0	0	0	0	**1,1,0,1**

Pentru înscrierea informaţiei în registru, în general nu este obligatorie ştergerea lui, deoarece pachetul de n biţi ce va fi înscris va înlocui informaţia existentă în registru. Datele se înscriu în registru secvenţial la intrarea D a primei celule din stânga. La fiecare impuls de tact datele se deplasează de la stânga spre dreapta. După un număr de impulsuri egal cu numărul de biţi a registrului datele încep să apară la ieşirea registrului în ordinea în care au fost înscrise. În **tabelul 6.4.2** se observă ca după fiecare impuls de tact, biţi cuvântului de intrare se deplasează de la ieşirea primului bistabil **Q_A** la ieşirea ultimului bistabil **Q_D.** După primele 4 impulsuri de tact la ieşirea registrului se află primul bit (din dreapta) al cuvântului de intrare, iar după încă 4 impulsuri la ieşirea registrului se goleşte. Registrul poate fi citit şi paralel dacă ieşirile **Q_A, Q_B** şi **Q_C** sunt accesibile la pinii integratului. Acest tip de registru mai poartă numele de registru de deplasare.

2. Registru cu înscriere serie şi citire paralel (SIPO)

Acest tip de registru este asemănător ca şi structură cu registrul SISO cu deosebirea esenţială că la acest registru sunt accesibile toate ieşirile bistabililor (**figura 6.4.2**).

Acest registru are o singură intrare pentru înscrierea serie a biţilor unui cuvânt şi n ieşiri pentru citirea simultană (paralel) a datelor.

Registrul SIPO mai este prevăzut cu o intrare de citire care comandă citirea simultană a semnalelor de la ieşirile registrului după ce acesta a fost încărcat complet. Informaţiile se păstrează în registru până la resetarea acestuia (ştergere). Utilizarea registrului pentru înscrierea unor date noi se face numai după aducerea tuturor bistabililor în starea 0.

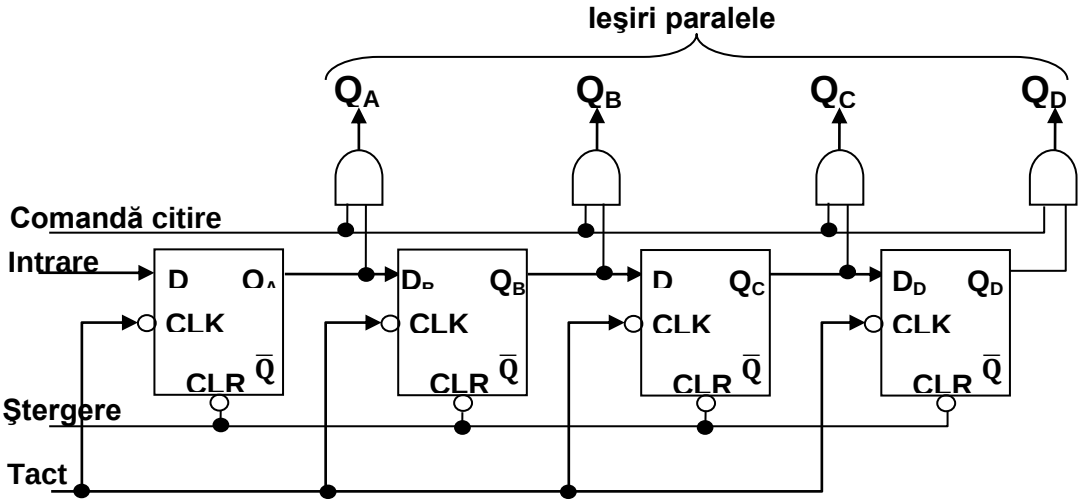


Figura 6.4.2 Schemă principiu registru SIPO de 4 biţi

Funcţionarea acestui registru pentru cuvântul 1101 se poate urmări în tabelul 6.4.3

Tabelul 6.4.3

Tact	Q_A	Q_B	Q_C	Q_D
0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	1	0	1	0
4	1	1	0	1

Informaţia este introdusă în registru la fel ca la registru SISO (bit cu bit, prin deplasarea de la stânga la dreapta a conţinutului pe durata a 4 impulsuri de tact).

Când registrul este complet încărcat se dă comanda de citire şi prin cele 4 porţi ŞI datele sunt livrate simultan la ieşirile paralele ale registrului.

3. Registru cu înscriere paralel şi citire serie (PISO)

Acest tip de registru permite înscrierea paralelă (simultană) a datelor şi citirea bit cu bit a acestora. Registrul are n intrări pentru înscrierea paralel a biţilor informaţiei şi o singură ieşire pentru citirea serie a informaţiei (**figura 6.4.3**).

Acest registru se utilizează în special pentru transformarea transmisiei paralelă a datelor în transmisie serială ce poate fi conectată direct la o linie de comunicaţii sau un computer.

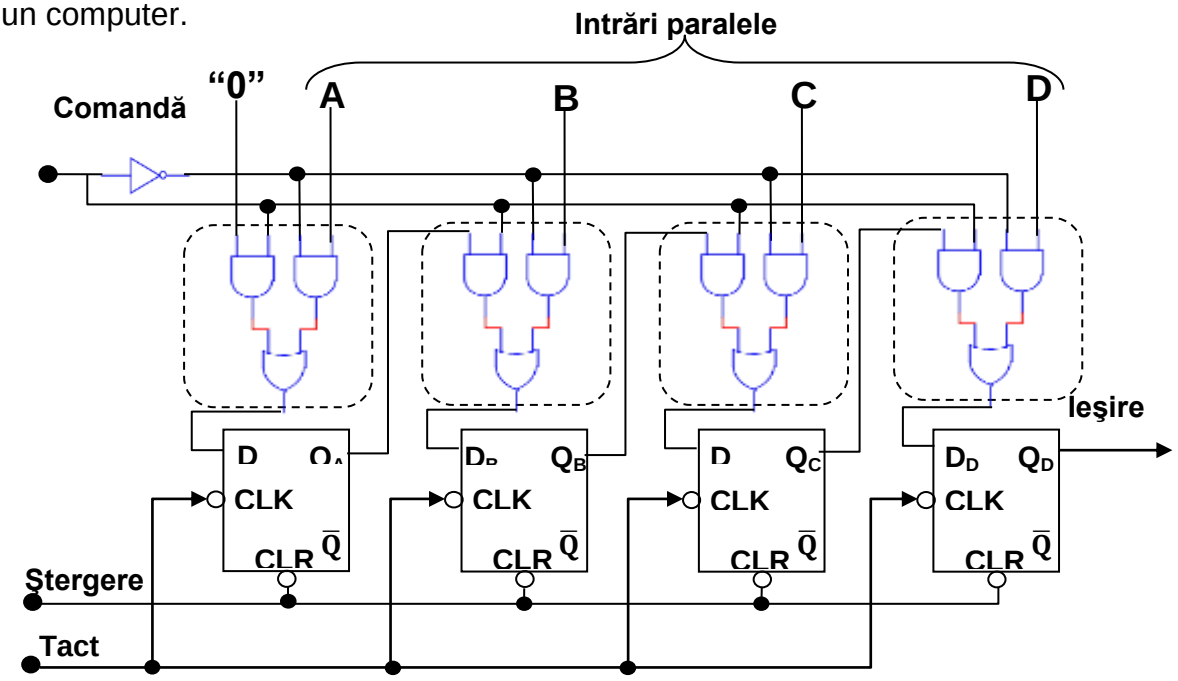


Figura 6.4.3 Schemă principiu registru PISO de 4 biţi

Funcţionarea acestui registru pentru cuvântul 1101 se poate urmări în tabelul **6.4.4**

Tabelul 6.4.4

Tact	Q_A	Q_B	Q_C	Q_D	Ieşire serie
0	0	0	0	0	0
1	**1**	**1**	**0**	**1**	0
2	0	**1**	**1**	**0**	**1**
3	0	0	**1**	**1**	**0,1**
4	0	0	0	**1**	**1,0,1**
5	0	0	0	0	**1,1,0,1**

Pentru înscrierea datelor în registru se activează comanda înscriere. La primul impuls de tact cei 4 biţi de la intrările paralele sunt înscrişi simultan în celulele registrului prin intermediul porţilor ŞI. Citirea se face bit cu bit pe durata a 4 impulsuri de tact conform **tabelului 6.4.4**.

4. Registru cu înscriere paralel şi citire paralel (PIPO)

Acest tip de registru permite înscrierea paralelă (simultană) a datelor şi citirea simultană a acestora. Registrul are n intrări pentru înscrierea paralel a biţilor informaţiei şi o n ieşiri pentru citirea paralel a informaţiei (**figura 6.4.4**).

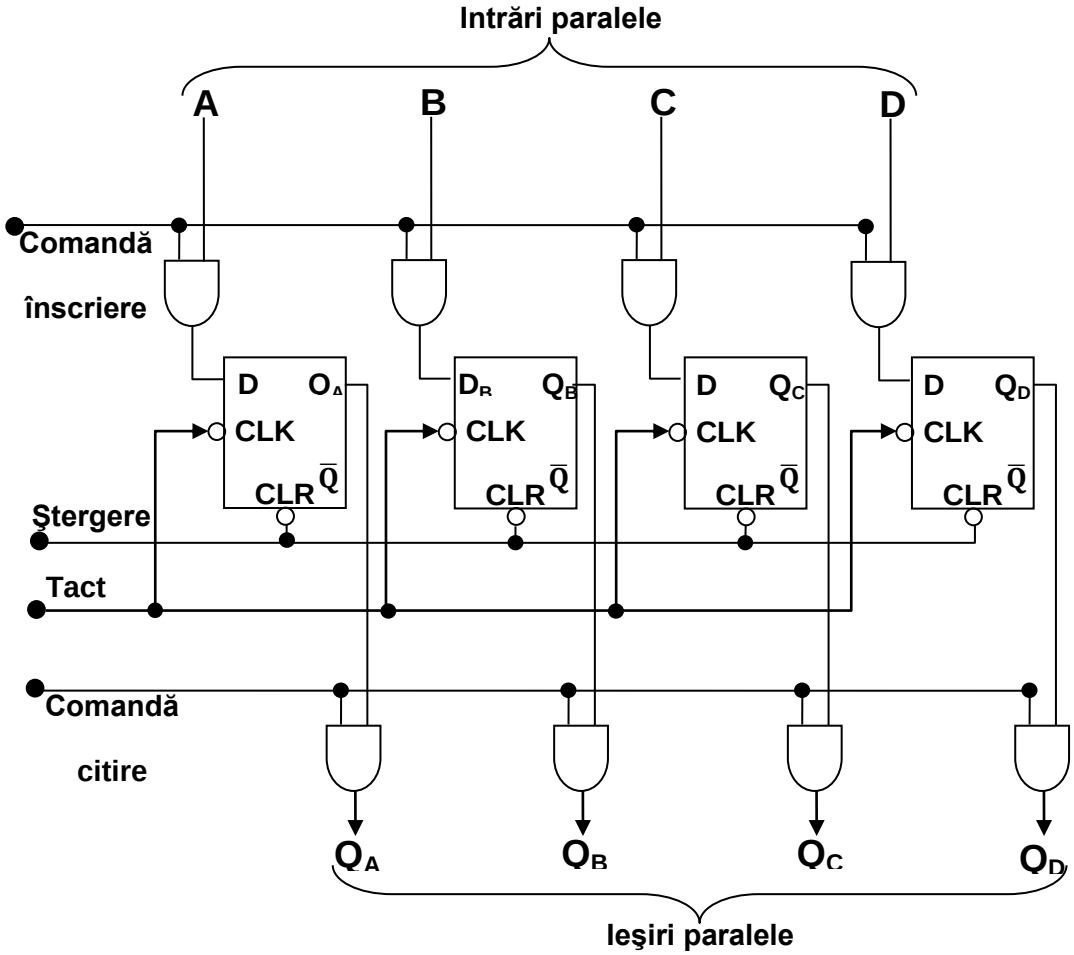


Figura 6.4.4 Schemă principiu registru PIPO de 4 biţi

Când se dă comandă de înscriere, cei 4 biţi a informaţiei (A, B, C,D) sunt introduşi simultan în celulele registrului prin porţile ŞI de intrare, la primul impuls de tact.

Odată înscrisă, informaţia poate rămâne în registru oricât de mult timp.

Când se dă comandă de citire, se extrage informaţia memorată în registru prin intermediul porţilor ŞI de ieşire, astfel încât pe durata unui singur impuls de tact cei 4 biţi a informaţiei (Q_A, Q_B, Q_C, Q_D) sunt extraşi din registru.

5. NUMĂRĂTOARE CU REGISTRU DE DEPLASARE

Un numărător cu registru de deplasare este un registru de deplasare la care i se adaugă un circuit logic combinaţional, obţinându-se un automat de stări cu diagrama de stări ciclică. Spre deosebire de numărătoarele binare, numărătoarele cu registru de deplasare nu numără într-o succesiune binară ascendentă sau descendentă, utilizându-se în aplicaţii de comandă.

Cele mai utilizate numărătoare cu registru de deplasare sunt:

- Numărătorul în inel;
- Numărătorul Johnson.

a. NUMĂRĂTORUL ÎN INEL

Numărătorul utilizează un registru universal cu încărcare şi citire paralel (PIPO), prevăzut cu intrare şi ieşire serială. Pentru a înţelege funcţionarea unui numărător în inel se prezintă o aplicaţie cu registrul 74LS194 (figura 6.4.5)

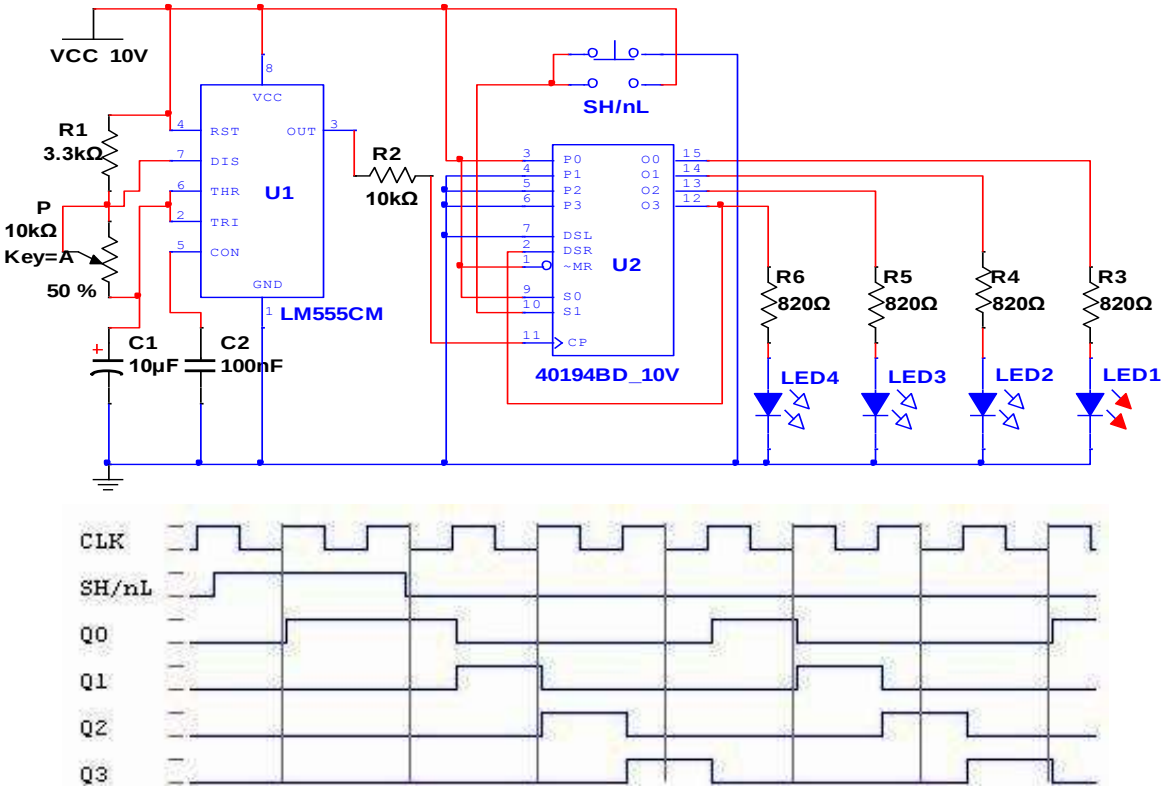


Figura 6.4.4 Numărător în inel pe 4 biţi cu CI 40194 şi diagramele de semnal

Când se activează butonul SH/nL intrarea S1 trece în 1 logic situaţie în care registrul se încarcă paralel (**$Q_3Q_2Q_1Q_0$ = 0001**) – se aprinde LED1. La dezactivarea butonului SH/nL intrarea S1 trece în 0 logic şi sub acţiunea impulsurilor de tact (furnizate de U1-LM555) bitul 1 de la ieşirea **Q_0** se deplasează spre stânga – se aprind succesiv LED-urile 2,3,4 (lumina "curge" de la dreapta spre stânga). După terminarea ciclului începe un nou ciclu identic până la activarea butonului SH/nL când registrul se iniţializează din nou. Circuitul poate fi considerat numărător al impulsurilor de tact aplicate deoarece pentru fiecare impuls de tact dintr-un ciclu starea ieşirilor este distinctă, existând 4 stări distincte.

b. NUMĂRĂTORUL JOHNSON

Numărătorul Johnson se obţine dintr-un registru de deplasare prin conectarea ieşiri **Qn** la intrarea serială printr-o poartă NU. În această situaţie numărul de stări distincte ale unui ciclu complet de funcţionare este 2n. Acest numărător mai este cunoscut şi sub numele de numărător în inel răsucit.

În aplicaţia prezentată între ieşirea $\mathbf{Q_3}$ şi intrarea serială DSR este conectată poarta ŞI – ¼ 4009 (**figura 6.4.5**). Deoarece registrul are 4 biţi, circuitul are 8 stări distincte în cadrul unui ciclu complet, după cum se vede din diagrama din **figura 6.4.5**.

Numărătorul se iniţializează prin aplicarea unui semnal de ştergere ($\overline{\mathbf{MR}} = \mathbf{0}$) care determină $\mathbf{Q_3Q_2Q_1Q_0 = 0000}$.

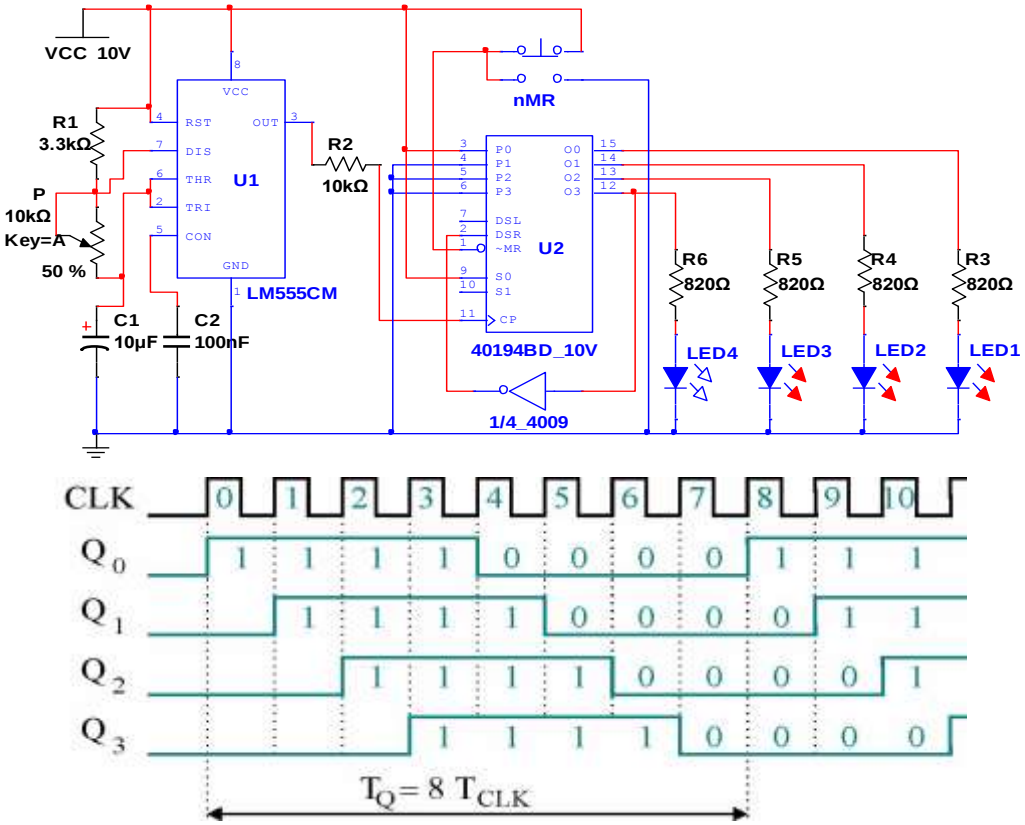
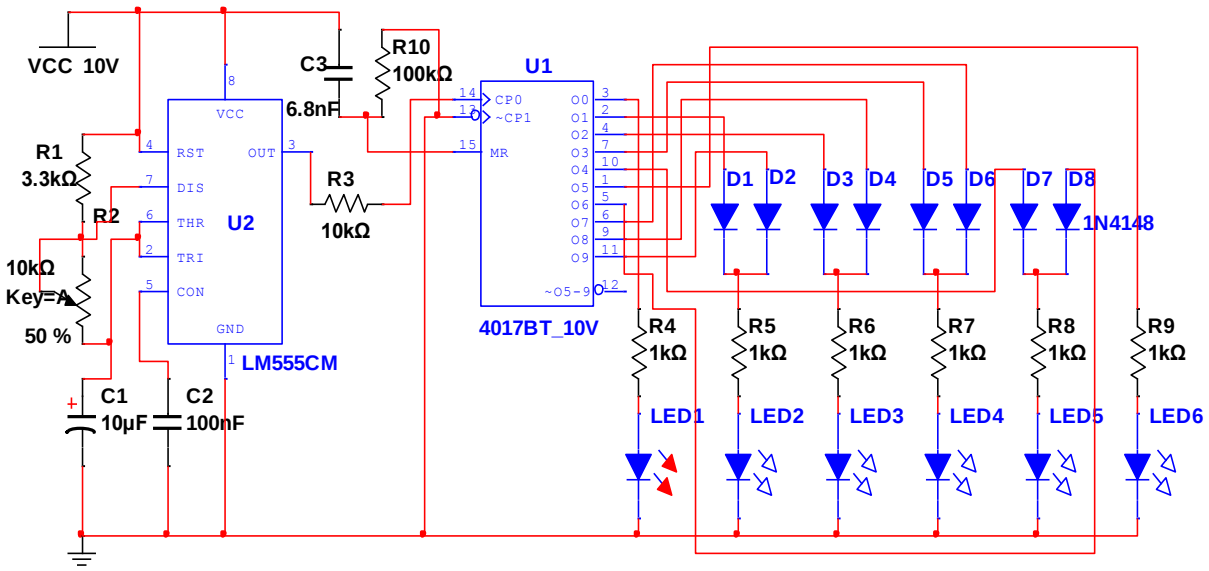


Figura 6.4.5 Numărător Johnson pe 4 biţi cu CI 40194 şi diagramele de semnal

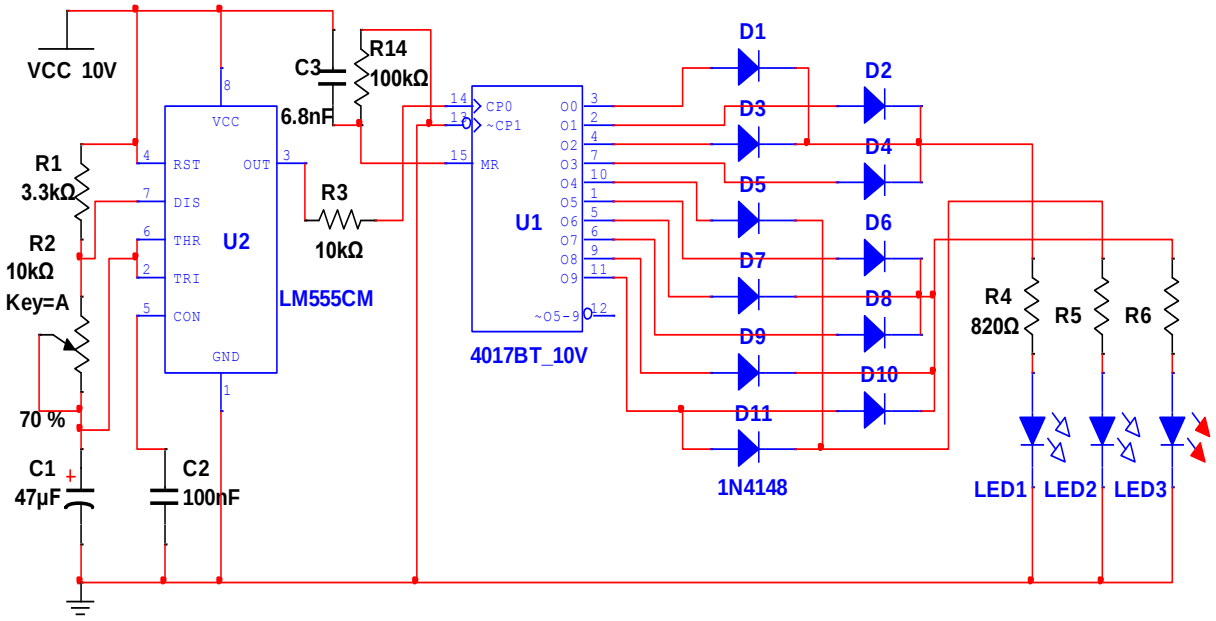
La activarea butonului nMR numărătorul se iniţializează (toate ieşirile trec în 0 logic). Când intrarea $\overline{\mathbf{MR}}$ trece în 1 logic stările logice ale ieşirilor se schimbă la fiecare impuls de tact (CLK1→$Q_3Q_2Q_1Q_0$=0001, CLK2→$Q_3Q_2Q_1Q_0$=0011,..... ,CLK7→$Q_3Q_2Q_1Q_0$=0000).

Led-urile se aprind succesiv de la dreapta spre stânga şi rămân aprinse apoi se sting succesiv în aceeaşi ordine).

În **figura 6.4.6** sunt prezentate 2 aplicaţii cu numărătorul Johnson 4017.



a. Lumină dinamică



b. Semafor

Figura 6.4.6 Aplicaţii cu numărător Johnson 4017

REZUMATUL CAPITOLULUI

- ***Circuitele logice secvenţiale (CLS)*** – sunt circuite logice combinaţionale cu memorie care se caracterizează prin faptul că în fiecare moment starea logică a ieşirilor depind atât de starea logică a intrărilor cât şi de stările logice anterioare ale intrărilor sau ale circuitului.
- circuitele logice secvenţiale se împart în două mari categorii:
  - **circuite secvenţiale asincrone** – starea prezentă a circuitului poate fi modificată în orice moment, ca efect al schimbării nivelelor logice aplicate la intrările principale;
  - **circuite secvenţiale sincrone** - starea prezentă a circuitului poate fi modificată numai la apariţia unui semnal numit semnal de ceas sau tact. Semnalul de ceas este un şir de impulsuri dreptunghiulare care se aplică circuitului printr-o intrare suplimentară numită intrarea semnalului de ceas.
- **Circuitele basculante bistabile (CBB)** – sunt cele mai simple circuite logice secvenţiale, cu două stări stabile, utilizate ca elemente de memorie în circuitele logice secvenţiale complexe în scopul memorării stărilor interne ale acestora.
- Un CBB este prevăzut cu două sau mai multe intrări şi două ieşiri care sunt complementare una faţă de cealaltă şi funcţionează ca o memorie de 1 bit.
- **Circuitele basculante bistabile RS asincrone (latch)** sunt prevăzute cu 2 intrări R (Reset) readucere în 0 sau ştergere şi S (Set) fixare sau înscriere, precum şi cu 2 ieşiri complementare Q respectiv $\overline{Q}$ și pot fi realizate cu 2 porţi SAU-NU (NOR) sau 2 porţi ŞI-NU (NAND)
- **Circuitele basculante bistabile RS sincrone (bistabile)** spre deosebire de cele asincrone sunt prevăzute cu o intrare suplimentară de comandă numită intrare de tact şi pot fi realizate cu 4 porţi SAU-NU (NOR) sau 4 porţi ŞI-NU (NAND).
- **Circuitele basculante bistabile JK** se obţin din bistabilele RS prin introducerea unei bucle de reacţie de la ieşiri la intrări. Aceste circuite elimină starea de nedeterminare a ieşirilor unui circuit basculant când intrările au aceeaşi valoare logică.
- **Circuitul basculant bistabil de tip D (Delay)** se obţine dintr-un CBB de tip RS sau JK prin conectarea unei porţi inversoare între cele două intrări de date RS sau JK, în scopul eliminării stărilor nedeterminate.

- **Circuitul basculant bistabil de tip T (toggle)** reprezintă cel mai simplu automat şi se obţine dintr-un CBB de tip RS sau JK prin conectarea împreună a celor două intrări de date RS sau JK. Bistabilul de tip T are o singură intrare de date T, o intrare de tact CLK şi două ieşiri complementare $\mathbf{Q}$ şi $\overline{\mathbf{Q}}$.
- **Numărătoarele** – sunt circuite logice secvenţiale utilizate pentru contorizarea (numărarea şi memorarea) impulsurilor aplicate la intrările acestora.
- Numărătoarele nu au intrări de date, tranziţiile se efectuează după o anumită regulă într-o anumită ordine, fixate prin construcţia numărătorului, în ritmul unui semnal de tact.
- Numărătoarele se realizează cu circuite basculante bistabile.
- Numărătoarele binare se clasifică după următoarele criterii:
  - ***După modul de conectare a bistabilelor de comandă:***
    - ***numărătoare asincrone*** – bistabilele sunt conectare în serie, intrarea de tact CLK a unui bistabil este conectată la ieşirea Q a bistabilului anterior, basculare a unui bistabil se face numai după basculare a bistabilului anterior;
    - ***numărătoare sincrone*** – bistabilele sunt conectate în paralel, intrările de tact CLK a tuturor bistabilelor sunt conectate împreună, basculare a tuturor bistabililor se face în acelaşi moment;
  - ***După sensul numărării***:
    - ***numărătoare directe*** – fiecare impuls prezent la intrarea numărătorului creşte conţinutul acestuia cu o unitate (numără în sens crescător);
    - ***numărătoare inverse*** – fiecare impuls prezent la intrarea numărătorului scade conţinutul acestuia cu o unitate (numără în sens descrescător);
    - ***numărătoare reversibile*** – efectuează numărarea în ambele sensuri în funcţie de comanda primită din exterior;
  - ***După codul de numărare:***
    - numărătoare binare – $m=2^n$;
    - numărătoare decadice – $m=10$.
- ***Registrele*** – sunt circuite logice secvenţiale care primesc, stochează şi transferă informaţii sub formă binară.
- Un registru este format din mai multe celule bistabile de tip RS, JK sau D şi permite memorarea şi/sau deplasarea informaţiei la comanda impulsurilor de tact.

- Un registru care conţine n celule bistabile are o capacitate de n biţi. Registrele pot fi considerate memorii rapide de mici dimensiuni.
- La un registru se definesc următoarele operaţii:
  - Înscrierea – introducerea datelor în registru;
  - Citirea – extragerea datelor din registru;
  - Deplasarea datelor în registru;
  - Ştergerea – aducerea tuturor registrelor în starea 0.
- După modul de înscriere/ citire se disting patru tipuri de registre:
  - registru cu înscriere serie şi citire serie - SISO;
  - registru cu înscriere serie şi citire paralel – SIPO;
  - registru cu înscriere paralel şi citire serie – PISO;
  - registru cu înscriere serie şi citire paralel – PIPO.
- **Un numărător cu registru de deplasare** este un registru de deplasare la care i se adaugă un circuit logic combinaţional, obţinându-se un automat de stări cu diagrama de stări ciclică.
- **Numărătorul în inel** - utilizează un registru universal cu încărcare şi citire paralel (PIPO), prevăzut cu intrare şi ieşire serială.
- **Numărătorul Johnson** se obţine dintr-un registru de deplasare prin conectarea ieşiri **Qn** la intrarea serială printr-o poartă NU.

6.5 LUCRĂRI DE LABORATOR

LUCRARE DE LABORATOR 6

CIRCUIT BASCULAT BISTABIL DE TIP RS ASINCRON.

- **OBIECTIVE:**
  - Realizarea schemei circuitului basculat bistabil cu simulatorul;
  - Realizarea practică a circuitului basculant bistabil;
  - Realizarea tabelului de adevăr pentru verificarea funcționării corecte a circuitului;

- **RESURSE:**
  - Calculatoare cu soft de simulare a circuitelor electronice;
  - Proiector multimedia;
  - Sursă de tensiune continuă reglabilă;
  - Pistoale de lipit;
  - Accesorii pentru lipit, conductoare;
  - Plăcuțe de lucru;
  - Rezistoare, comutatoare, LED-uri, circuite integrate cu porți logice elementare (NAND, NOR).

- **DESFĂȘURAREA LUCRĂRII:**

1. Realizează cu simulatorul schema electronică din figura de mai jos:

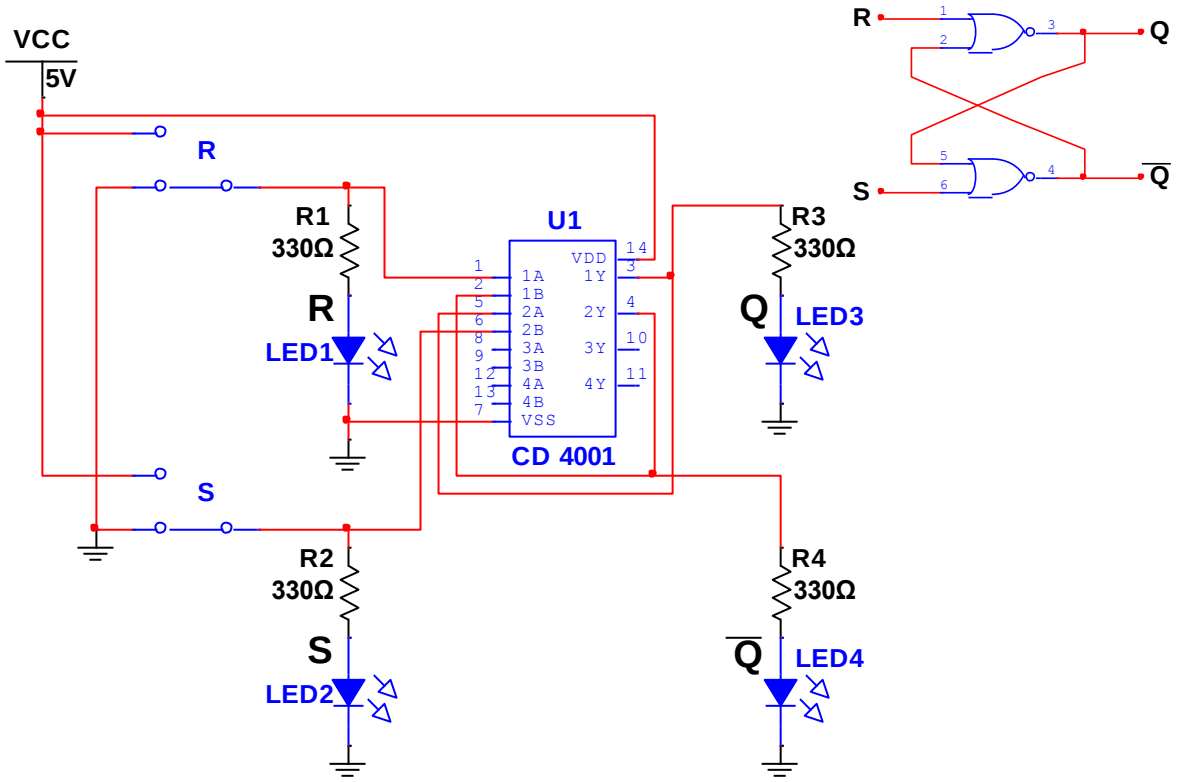


Figura 6.5.1 Circuit basculat bistabil RS asincron cu porți SAU-NU (NOR)

2. Realizează practic, pe plăcuţa de probă montajul corespunzător schemei date.

3. Plasează în soclu de pe placa de probă circuitul integrat (ATENȚIE la poziția CI).

4. Conectează montajul la o sursă de tensiune continuă conform schemei de mai sus, pornește sursa și regleaz-o la valoarea indicată în schemă.

5. Conectează succesiv comutatoarele **R** și **S** la potenţialul **0V** respectiv **5V** conform tabelului de mai jos şi notează în tabel valorile logice ale ieşirilor $\boldsymbol{Q}$ și $\boldsymbol{\overline{Q}}$ în coloanele **NL** (nivel logic).

6. Măsoară cu voltmetrul tensiunile în punctele $R, S, Q, \bar{Q}$ și notează în tabel valorile indicate în coloanele **NT** (nivel tensiune).

Tabel adevăr CBB – RS cu porți SAU-NU

R		S		Q		$\overline{Q}$	
NL	**NT**	**NL**	**NT**	**NL**	**NT**	**NL**	**NT**
0		**0**					
0		**1**					
1		**0**					
1		**1**					

7. Oprește sursa de alimentare și înlocuiește circuitul integrat CD 4001 (4 porți SAU-NU) cu un circuit integrat CI 4011(4 porți ȘI-NU).

8. Conectează succesiv comutatoarele **R** și **S** la potenţialul **0V** respectiv **5V** conform tabelului de mai jos şi notează în tabel valorile logice ale ieşirilor $\boldsymbol{Q}$ și $\boldsymbol{\overline{Q}}$ în coloanele **NL** (nivel logic).

9. Măsoară cu voltmetrul tensiunile în punctele $R, S, Q, \bar{Q}$ și notează în tabel valorile indicate în coloanele **NT** (nivel tensiune).

Tabel adevăr CBB – RS cu porți ȘI-NU

R		S		Q		$\overline{Q}$	
NL	**NT**	**NL**	**NT**	**NL**	**NT**	**NL**	**NT**
0		**0**					
0		**1**					
1		**0**					
1		**1**					

LUCRARE DE LABORATOR 7

CIRCUIT BASCULAT ASTABIL CU PORȚI LOGICE NU (NOT).

- **OBIECTIVE:**
  - Realizarea schemei circuitului basculat astabil cu simulatorul;
  - Realizarea practică a circuitului basculant astabil;
  - Verificarea funcționării circuitului basculat astabil și determinarea frecvenței;

- **RESURSE:**
  - Calculatoare cu soft de simulare a circuitelor electronice;
  - Proiector multimedia;
  - Sursă de tensiune continuă reglabilă, osciloscop cu două spoturi;
  - Pistoale de lipit;
  - Accesorii pentru lipit, conductoare;
  - Plăcuțe de lucru;
  - Rezistoare, comutatoare, LED-uri, circuite integrate cu porți logice inversoare (NOT).

- **DESFĂȘURAREA LUCRĂRII:**

1. Realizează cu simulatorul schemele electronice din figura de mai jos:

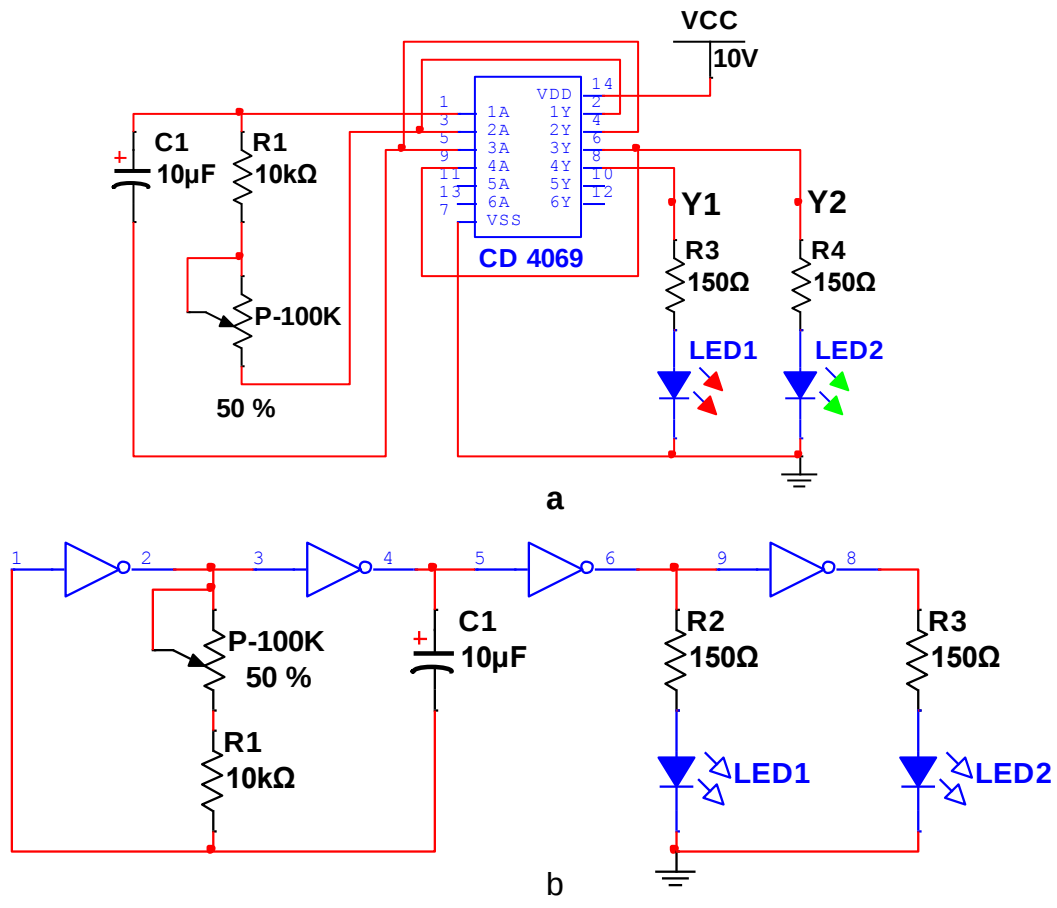


Figura 6.5.2 Circuit basculat astabil cu porți NU (NOT)

2. Realizează practic, pe plăcuța de probă montajul schemei din figura 6.5.2 a.

3. Pentru efectuarea conexiunilor la pinii soclului circuitului integrat urmărește schema din figura 6.5.2 b.

4. Plasează în soclu de pe placa de probă circuitul integrat (ATENȚIE la poziția CI).

5. Conectează montajul la o sursă de tensiune continuă conform schemei din figura 6.5.2 a, pornește sursa și regleaz-o la valoarea indicată în schemă.

6. Reglează potențiometrul **P** la valoarea minimă.

7. Conectează în circuit un osciloscop cu două canale în punctele **Y1** și **Y2**.

8. Reglează potențiometrul **P** spre valoarea maximă (de la 0 la 100 KΩ) și urmărește pe osciloscop modificarea frecvenței.

9. Calculează frecvența când cursorul potențiometrului este în pozițiile extreme (minim și maxim).

P = 0 Ω $\Rightarrow$ f = …….. T = …….

P = 100 KΩ $\Rightarrow$ f = …….. T = …….

LUCRARE DE LABORATOR 8

CIRCUIT BASCULAT MONOSTABIL CU PORȚI LOGICE ȘI-NU (NAND).

- **OBIECTIVE:**
  - Realizarea schemei circuitului basculat monostabil cu simulatorul;
  - Realizarea practică a circuitului basculant monostabil;
  - Verificarea funcționării circuitului basculat monostabil și determinarea frecvenței;

- **RESURSE:**
  - Calculatoare cu soft de simulare a circuitelor electronice;
  - Proiector multimedia;
  - Sursă de tensiune continuă reglabilă, osciloscop;
  - Pistoale de lipit;
  - Accesorii pentru lipit, conductoare;
  - Plăcuțe de lucru;
  - Rezistoare, comutatoare, LED-uri, circuite integrate cu porți logice elementare (NAND, NOR).

- **DESFĂȘURAREA LUCRĂRII:**

1. Realizează cu simulatorul schemele electronice din figura de mai jos:

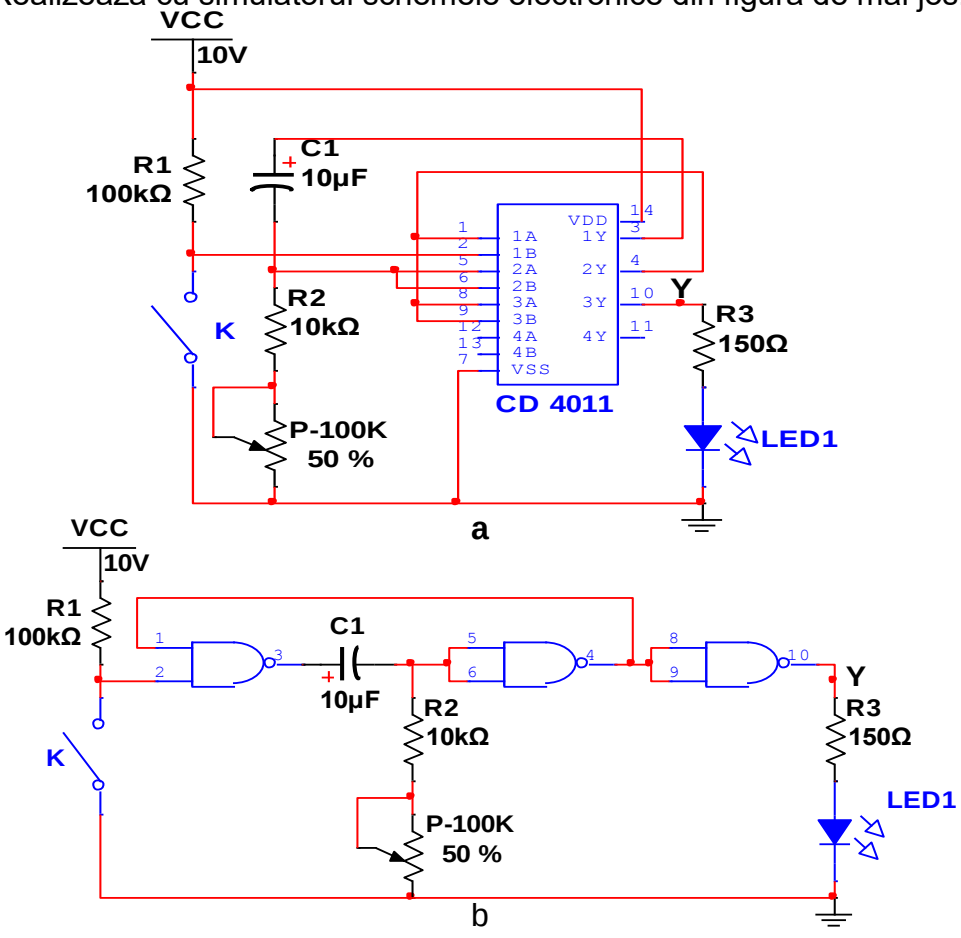


Figura 6.5.3 Circuit basculat monostabil cu porți ȘI-NU (NAND)

2. Realizează practic, pe plăcuța de probă montajul schemei din figura 6.5.3 a.

3. Pentru efectuarea conexiunilor la pinii soclului circuitului integrat urmărește schema din figura 6.5.3 b.

4. Plasează în soclu de pe placa de probă circuitul integrat (ATENȚIE la poziția CI).

5. Conectează montajul la o sursă de tensiune continuă conform schemei din figura 6.5.3 a, pornește sursa și regleaz-o la valoarea indicată în schemă.

6. Reglează potențiometrul **P** la valoarea minimă.

7. Conectează în circuit un osciloscop cu un canal în punctul **Y**.

8. Închide și deschide întrerupătorul K.

9. Vizualizează pe osciloscop și calculează frecvența semnalului în punctul **Y**.

P = 0 KΩ $\Rightarrow$ f = …….. T = …….

10. Reglează potențiometrul **P** la valoarea minimă.

11. Conectează în circuit un osciloscop cu un canal în punctul **Y**.

12. Închide și deschide întrerupătorul K.

13. Vizualizează pe osciloscop și calculează frecvența semnalului în punctul **Y**.

P = 100 KΩ $\Rightarrow$ f = …….. T = …….

LUCRARE DE LABORATOR 9

NUMĂRĂTOARE ASINCRONE

- **OBIECTIVE:**
  - Realizarea schemei unui circuit cu numărător asincron cu simulatorul;
  - Realizarea practică a circuitului cu numărător asincron;
  - Verificarea funcționării numărătorului;
  - Realizarea tabelului de adevăr în funcție de indicațiile LED-urilor de ieșire;

- **RESURSE:**
  - Calculatoare cu soft de simulare a circuitelor electronice;
  - Proiector multimedia;
  - Sursă de tensiune continuă reglabilă;
  - Pistoale de lipit;
  - Accesorii pentru lipit, conductoare;
  - Plăcuțe de lucru;
  - Rezistoare, comutatoare, LED-uri, CI numărătoare.

- **DESFĂȘURAREA LUCRĂRII:**

1. Realizează cu simulatorul schema electronică din figura de mai jos:

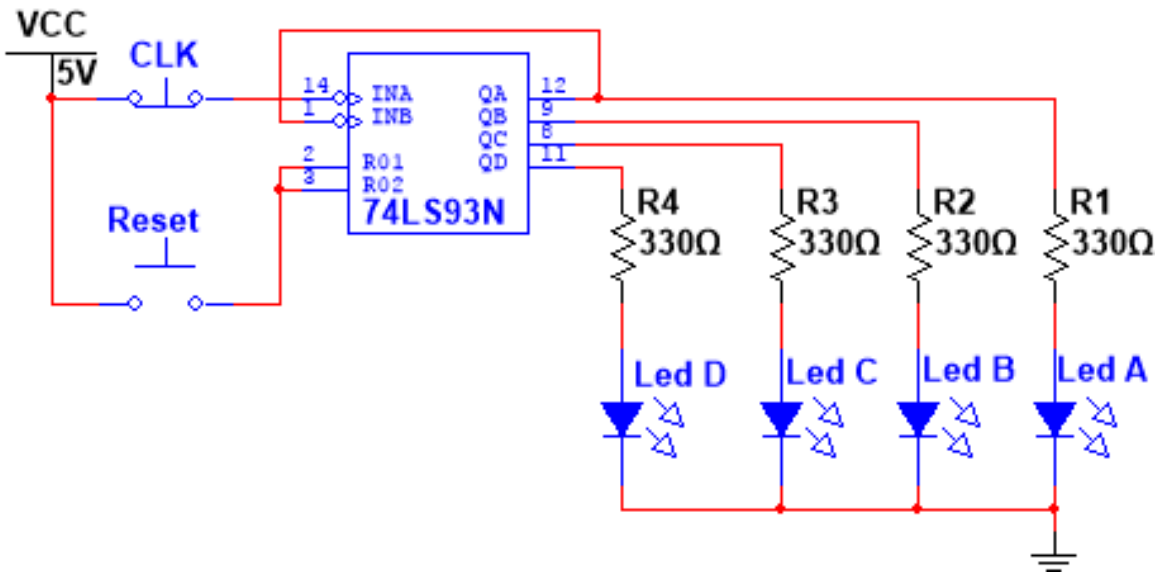


Figura 6.5.4 Aplicaţie cu numărătorul asincron binar 74LS93

2. Realizează practic, pe plăcuța de probă montajul schemei din figura 6.5.4.

ATENȚIE! Pinul 10 al CI se conectează la (-) iar pinul 5 al CI se conectează la (+).

3. Plasează în soclul de pe placa de probă circuitul integrat (ATENȚIE la poziția CI).

4. Conectează montajul la o sursă de tensiune continuă conform schemei din **figura 6.5.4**, pornește sursa și regleaz-o la valoarea indicată în schemă.

5. La fiecare apăsare a butonului cu revenire **CLK** notează în tabelul de adevăr al numărătorului starea LED-urilor (aprins **A** sau stins **S**).

Nr. impuls	Q_D $2^3=8$	Q_C $2^2=4$	Q_B $2^1=2$	Q_A $2^0=1$	Led D	Led C	Led B	Led A
0	0	0	0	0	S	S	S	S
1	0	0	0	1				
2	0	0	1	0				
3	0	0	1	1				
4	0	1	0	0				
5	0	1	0	1				
6	0	1	1	0				
7	0	1	1	1				
8	1	0	0	0				
9	1	0	0	1				
10	1	0	1	0				
11	1	0	1	1				
12	1	1	0	0				
13	1	1	0	1				
14	1	1	1	0				
15	1	1	1	1				
16	0	0	0	0				

LUCRARE DE LABORATOR 10

NUMĂRĂTOARE SINCRONE

- **OBIECTIVE:**
  - Realizarea schemei unui circuit cu numărător sincron cu simulatorul;
  - Realizarea practică a circuitului cu numărător sincron;
  - Verificarea funcționării numărătorului;

- **RESURSE:**
  - Calculatoare cu soft de simulare a circuitelor electronice;
  - Proiector multimedia;
  - Sursă de tensiune continuă reglabilă;
  - Generator de semnal;
  - Pistoale de lipit;
  - Accesorii pentru lipit, conductoare;
  - Plăcuțe de lucru;
  - Rezistoare, afișaj 7 segmente, CI numărătoare și decodificatoare.

- **DESFĂȘURAREA LUCRĂRII:**

1. Realizează cu simulatorul schema electronică din figura de mai jos:

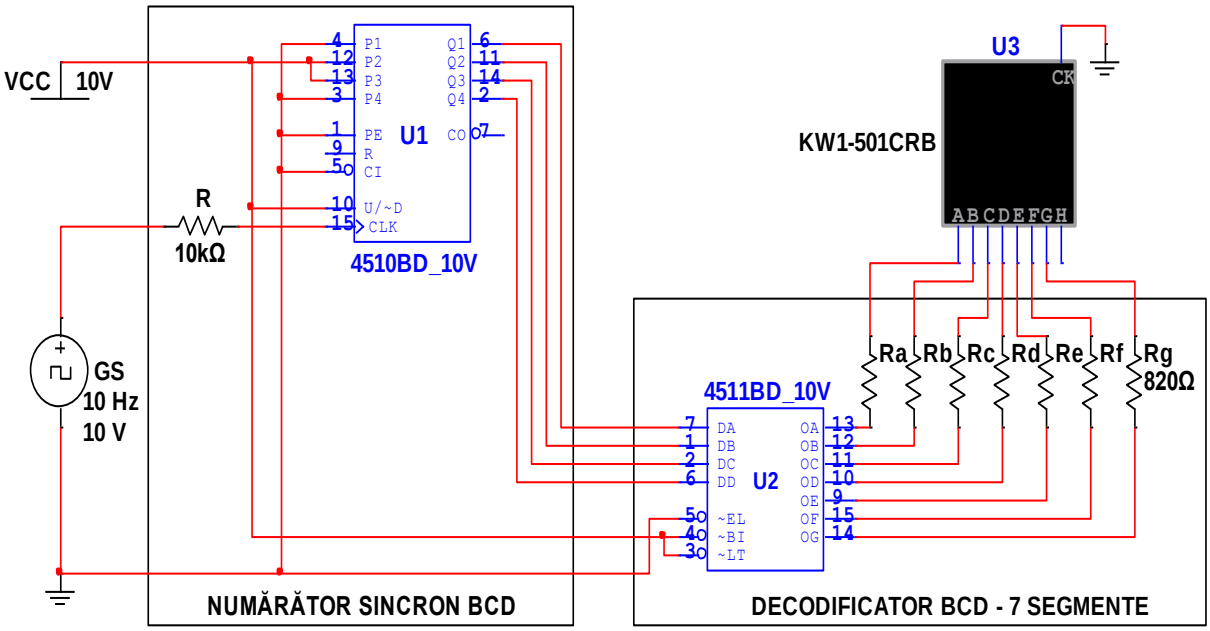


Figura 6.5.5 Aplicaţie cu numărătorul sincron BCD – CD4510

2. Realizează practic, pe o plăcuță de probă montajul schemei **NUMĂRĂTOR SINCRON BCD** din figura 6.5.5.

3. Plasează în soclul de pe placa de probă circuitul integrat (ATENȚIE la poziția CI).

4. Realizează practic, pe o plăcuță de probă montajul schemei **DECODIFICATOR BCD – 7 SEGMENTE** din figura 6.5.5.

5. Plasează în soclul de pe placa de probă circuitul integrat (ATENȚIE la poziția CI).

6. Interconectează cele două montaje conform schemei din figura 6.5.5. și tabelului de mai jos:

CI - 4510	**CI - 4511**
PIN 6	**PIN 7**
PIN 11	**PIN 1**
PIN 14	**PIN 2**
PIN 2	**PIN 6**
PIN 4 + PIN 8	**PIN 5 + PIN 8**
PIN 12 + PIN 16	**PIN 4 + PIN 16**

7. Conectează rezistoarele Ra…Rg de pe montajul decodificatorului la afișaj conform schemei din figura 6.5.5.

8. Conectează sursa de alimentare și generatorul de semnal conform schemei din figura 6.5.5.

9. Pornește generatorul de semnal și realizează următoarele reglaje:

a. Tip semnal – dreptunghiular;
b. Frecvența – 10 Hz;
c. Amplitudinea – 10 V.

10. Pornește sursa de alimentare, regleaz-o la valoarea indicată în schema din figura 6.5.5 și verifică funcționarea corectă a montajului.

LUCRARE DE LABORATOR 11

NUMĂRĂTOARE CU REGISTRU DE DEPLASARE – NUMĂRĂTOR ÎN INEL

- **OBIECTIVE:**
  - Realizarea schemei unui circuit cu numărător în inel cu simulatorul;
  - Realizarea practică a circuitului cu numărător în inel;
  - Verificarea și explicarea funcționării numărătorului;

- **RESURSE:**
  - Calculatoare cu soft de simulare a circuitelor electronice;
  - Proiector multimedia;
  - Sursă de tensiune continuă reglabilă;
  - Generator de semnal;
  - Pistoale de lipit;
  - Accesorii pentru lipit, conductoare;
  - Plăcuțe de lucru;
  - Rezistoare, comutatoare, LED-uri, CI numărătoare.

- **DESFĂȘURAREA LUCRĂRII:**

1. Realizează cu simulatorul schema electronică din figura de mai jos:

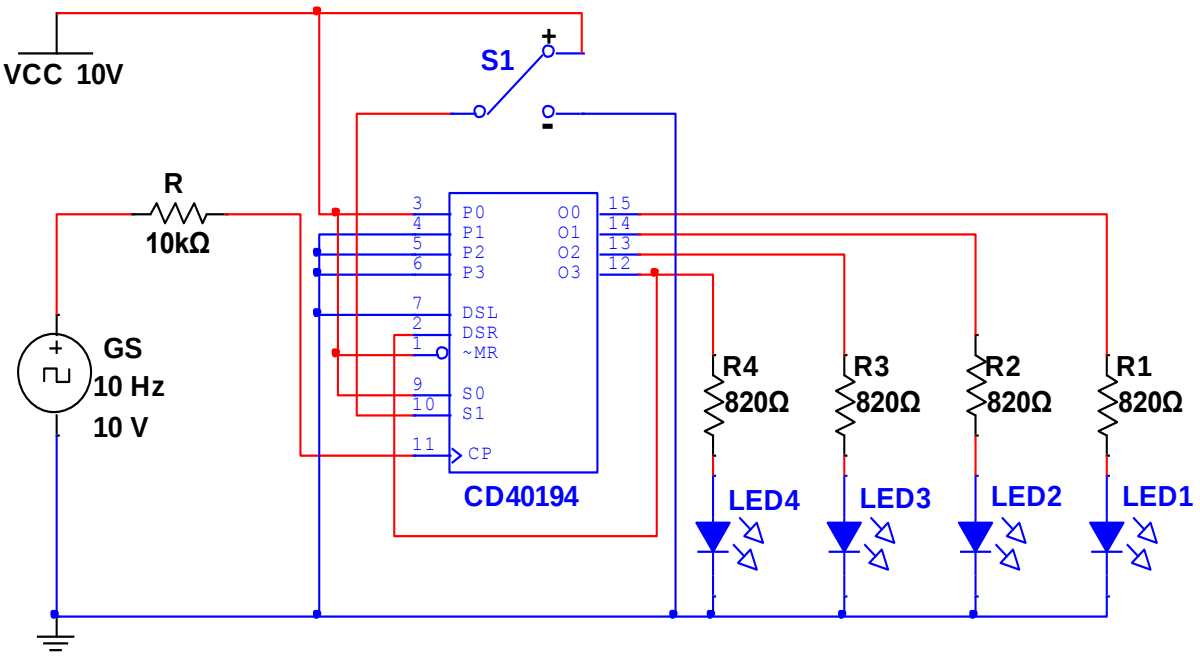


Figura 6.5.6 Aplicaţie cu numărătorul în inel – CD40194

2. Realizează practic, pe o plăcuță de probă montajul schemei din figura 6.5.6.
ATENȚIE! Pinul 8 al CI se conectează la (-) iar pinul 16 al CI se conectează la (+).

3. Plasează în soclu de pe placa de probă circuitul integrat (ATENȚIE la poziția CI).

4. Conectează sursa de alimentare și generatorul de semnal conform schemei din figura 6.5.6.

5. Fixează comutatorul **S1** pe poziția (**-**).

6. Pornește generatorul de semnal și realizează următoarele reglaje:

a. Tip semnal – dreptunghiular;
b. Frecvența – 10 Hz;
c. Amplitudinea – 10 V.

7. Pornește sursa de alimentare, regleaz-o la valoarea indicată în schema din figura 6.5.6.

8. Schimbă poziția comutatorului **S1** de pe (-) pe (+) apoi revin-o cu el în poziția inițială (se dă un impuls pozitiv la intrarea S1 a numărătorului).

9. Verifică funcționare corectă a circuitului urmărind starea led-urilor (led-urile se aprind apoi se sting succesiv de la dreapta spre stânga).

10. Explică funcționarea numărătorului cu registru de deplasare:

……..

……..

……..

LUCRARE DE LABORATOR 12

NUMĂRĂTOARE CU REGISTRU DE DEPLASARE – NUMĂRĂTOR JOHNSON

- **OBIECTIVE:**
  - Realizarea schemei unui circuit cu numărător Johnson cu simulatorul;
  - Realizarea practică a circuitului cu numărător Johnson;
  - Verificarea și explicarea funcționării numărătorului;

- **RESURSE:**
  - Calculatoare cu soft de simulare a circuitelor electronice;
  - Proiector multimedia;
  - Sursă de tensiune continuă reglabilă;
  - Generator de semnal;
  - Pistoale de lipit;
  - Accesorii pentru lipit, conductoare;
  - Plăcuțe de lucru;
  - Rezistoare, LED-uri, CI numărătoare.

- **DESFĂȘURAREA LUCRĂRII:**

1. Realizează cu simulatorul schema electronică din figura de mai jos:

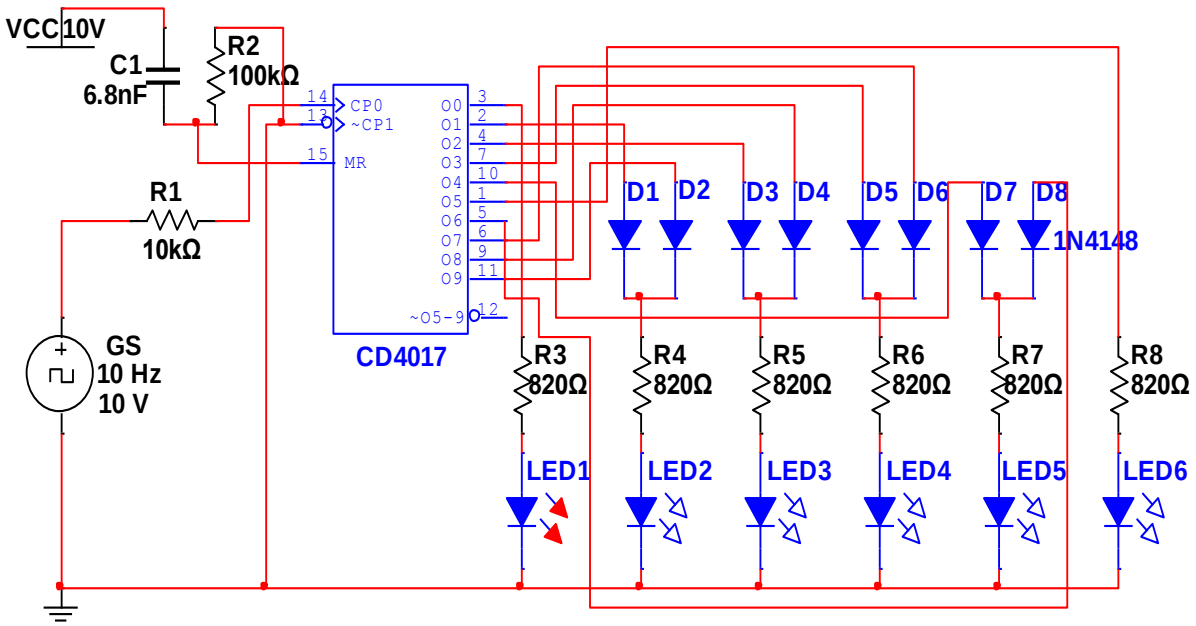


Figura 6.5.7 Aplicaţie cu numărătorul Johnson – CD4017

2. Realizează practic, pe o plăcuță de probă montajul schemei din figura 6.5.7.
ATENȚIE! Pinul 8 al CI se conectează la (-) iar pinul 16 al CI se conectează la (+).

3. Plasează în soclu de pe placa de probă circuitul integrat (ATENȚIE la poziția CI).

4. Conectează sursa de alimentare și generatorul de semnal conform schemei din figura 6.5.7.

5. Pornește generatorul de semnal și realizează următoarele reglaje:

a. Tip semnal – dreptunghiular;
b. Frecvența – 10 Hz;
c. Amplitudinea – 10 V.

6. Pornește sursa de alimentare, regleaz-o la valoarea indicată în schema din figura 6.5.7.

7. Verifică funcționare corectă a circuitului urmărind starea led-urilor (led-urile se aprind apoi se sting succesiv de la stânga spre dreapta apoi de la dreapta spre stânga).

8. Explică funcționarea numărătorului Jonson cu registru de deplasare:

……..

……..

……..

BIBLIOGRAFIE

1. Floyd, T., *Circuite digitale*, Editura Teora, București, 2003
2. Cosma, D., Chivu, A., *Electronică analogică. Electronică digitală - Lucrări practice*, Editura Arves, Craiova, 2005
3. Bițoiu, A., Băluță, G. ș.a., *Practica electronistului amator*, Editura Albatros, București, 1984
4. Cosma, D., Gheață, C., Mușat, C., Chivu, A., *Bazele electronicii digitale – Manual pentru clasa a X-a*, Editura CD Press, București, 2011
5. Găzdaru, C. ș.a., *Îndrumar pentru electroniști*, Editura Tehnică, București, 1986
6. Drăgulescu, N., *Agenda radioelectronistului*, Editura Tehnică, București, 1983
7. Vasilescu, G., *Electronică*, Editura Didactică și Pedagogică, București, 1981
8. http://www.datasheets360.com/
9. http://www.tehnium-azi.ro/page/index
10. http://eprofu.ro/tehnic/materiale-invatare-electronica/
11. http://cndiptfsetic.tvet.ro/
12. http://www.dannicula.ro/ed_ci/

"STUDIAZĂ MAI ÎNTÂI ȘTIINȚA ȘI CONTINUĂ APOI CU PRACTICA NĂSCUTĂ DIN ACEASTĂ ȘTIINȚĂ.” Leonardo da Vinci

ISBN-