

## Capitolul 2

# Circuite logice combinaționale

### P2.1

*Rezolvare:*

Utilizând axioma asociativității asupra operatorului sumă modulo doi, funcția paritate se poate exprima pentru a fi implementabilă cu XOR2 sub următoarele două forme:

$$f = (x_{n-1} \oplus (x_{n-2} \oplus (x_{n-3} \oplus (x_{n-4} \oplus \dots \oplus (x_3 \oplus (x_2 \oplus (x_1 \oplus x_0)))) \dots))))$$

$$f = ((\dots((x_{n-1} \oplus x_{n-2}) \oplus (x_{n-3} \oplus x_{n-4})) \oplus \dots \oplus ((x_3 \oplus x_2) \oplus (x_1 \oplus x_0)) \dots))$$

Prima expresie poate fi mapată pe o rețea serie (rs) iar a doua pe o rețea arbore binar (rb). Adoptând numărul de porți XOR ca măsură pentru dimensiunea rețelei rezultă:  $S(n)_{rs} = n-1$ ,  $S(n)_{rb} = n-1$ , iar adâncimea  $D(n)_{rs} = (n-1)$  niveluri,  $D(n)_{rb} = \lceil \log_2 n \rceil$ . Prin substituția  $XOR2 \rightarrow NXOR2$  se calculează aceeași funcție dacă un număr de  $(n-1)$  biți ai cuvântului, înainte de aplicare la intrarea rețelei, sunt complementați deoarece:  $x_{i+1} \oplus x_i = \overline{x_{i+1} \oplus \overline{x_i}}$ .

### P2.2

*Rezolvare:*

$$\begin{aligned} f_1 &= \overline{(A+B)(C+D)} = \overline{A} \overline{B} + \overline{C} \overline{D} \\ f_2 &= AB \oplus CD = AB(\overline{C} \overline{D}) + (\overline{A} \overline{B})CD = AB(\overline{C} + \overline{D}) + CD(\overline{A} + \overline{B}) = \\ &= AB\overline{C} + AB\overline{D} + \overline{A}CD + \overline{B}CD \\ f_3 &= \overline{ABC} + \overline{BD} \\ f_4 &= \overline{((A \oplus \overline{B})(\overline{C} \oplus D))} = AB + \overline{A} \overline{B} + C\overline{D} + \overline{C}D \end{aligned}$$

Tabelele de adevăr sunt reprezentate în Figura P.2.2-e.

A	B	C	D	$f_1$	$f_2$	$f_3$	$f_4$
0	0	0	0	0	1	0	1
0	0	0	1	0	1	1	1
0	0	1	0	0	1	0	1
0	0	1	1	1	1	1	1
0	1	0	0	0	1	0	0
0	1	0	1	0	0	0	1
0	1	1	0	0	0	1	1
0	1	1	1	1	0	1	0
1	0	0	0	0	1	0	0
1	0	0	1	0	0	1	1
1	0	1	0	0	0	0	1
1	1	1	1	1	0	1	0
1	1	0	0	1	0	0	1
1	1	0	1	1	0	0	1
1	1	1	0	1	0	0	1
1	1	1	1	0	0	0	1

Figura P 2.2

**P2.3***Rezolvare:*

Se vor demonstra analitic următoarele identități:

$$A \oplus B = \overline{A} \oplus \overline{B} = \overline{(A \oplus B)} = \overline{(A \oplus \overline{B})};$$

$$\overline{(A \oplus \overline{B})} = \overline{(\overline{A} \oplus B)} = \overline{(\overline{A} \oplus B)} = (A \oplus \overline{B}).$$

**P2.4***Rezolvare:*

$$\begin{aligned} a) \quad \overline{f} &= \overline{[(\overline{AB})A] \cdot [(\overline{AB})B]} = (AB + \overline{A})(AB + \overline{B}) = ABAB + \overline{A}AB + AB\overline{B} + \overline{A}\overline{B} = \\ &= AB + 0 + 0 + \overline{A}\overline{B} = AB + \overline{A}\overline{B} = \overline{A \oplus B}, \text{ vezi problema 1.6.} \end{aligned}$$

$$\begin{aligned} b) \quad \overline{f} &= \overline{(A + B + \overline{C})(\overline{AB} + \overline{CD}) + \overline{BCD}} = \overline{(A + B + \overline{C})(\overline{AB} + \overline{CD})} + \overline{BCD} = \\ &= \overline{(\overline{A}\overline{B}\overline{C})(\overline{AB}\overline{CD})} + \overline{BCD} = (\overline{A}\overline{B}\overline{C})(\overline{AB}\overline{CD}) + \overline{BCD} = \overline{BCD}. \end{aligned}$$

$$\begin{aligned} c) \quad \overline{f} &= \overline{(ABC + \overline{BCD}) + (\overline{ACD} + \overline{B\overline{C}D} + \overline{BCD})} = \\ &= \overline{(ABC + \overline{BCD})(\overline{ACD} + \overline{B\overline{C}D} + \overline{BCD})} = \\ &= \overline{(ABC + \overline{BCD})(\overline{A} + \overline{C} + \overline{D} + \overline{B\overline{C}D} + \overline{BCD})} = \\ &= \overline{(ABC + \overline{BCD})(\overline{A} + \overline{D}(1 + BC) + \overline{C}(1 + \overline{B}\overline{D}))} = \\ &= \overline{(ABC + \overline{BCD})(\overline{A} + \overline{D} + \overline{C})} = \\ &= \overline{ABC\overline{A} + ABC\overline{D} + ABC\overline{C} + \overline{BCD}\overline{A} + \overline{BCD}\overline{D} + \overline{BCD}\overline{C}} = \\ &= \overline{0 + ABC\overline{D} + 0 + \overline{BCD}\overline{A} + 0 + \overline{BCD}} = \\ &= \overline{ABC\overline{D} + (\overline{A}\overline{BCD} + \overline{BCD})} = \overline{ABC\overline{D} + \overline{BCD}} \end{aligned}$$

**P2.5***Rezolvare:*

a) Implicanți primi:

$$\begin{aligned} \{0, 4\} &\rightarrow \overline{B}\overline{C}; & \{0, 2\} &\rightarrow \overline{A}\overline{C}; & \{2, 3\} &\rightarrow \overline{A}B; \\ \{3, 7\} &\rightarrow BC; & \{4, 5\} &\rightarrow A\overline{B}; & \{5, 7\} &\rightarrow AC; \end{aligned}$$

Există două variante de alegere a implicanților primi prin care sunt acoperiți toți mintermii din diagramă:

$$f = \overline{A}\overline{C} + A\overline{B} + BC \text{ sau } f = \overline{B}\overline{C} + \overline{A}B + AC$$

Forma minimă ca produs de sume rezultă din Figura P2.5-a:

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

Prin desfacerea parantezelor s-au obținut toți implicanții primi de la forma sumă de produse.

b) Pentru forma sumă de produse din Figura P2.5-b rezultă următorii implicanți primi:

$$\begin{aligned} \{0, 1, 4, 5\} &\rightarrow \overline{A}\overline{C}; \\ \{9, 11, 13, 15\} &\rightarrow AD; \\ \{1, 5, 9, 13\} &\rightarrow \overline{C}D. \end{aligned}$$

Dintre acești implicanți primi, numai  $\overline{A}\overline{C}$  și  $AD$  sunt implicanți primi esențiali care trebuie să fie considerați pentru acoperirea tuturor mintermilor din diagrama V-K. Expresia minimă a funcției este:

$$f = \overline{A}\overline{C} + AD$$

Pentru forma produse de sume din Figura P2.5-b rezultă următorii implicanți primi:

$$\begin{aligned} \{2, 3, 6, 7\} &\rightarrow A + \overline{C}; \\ \{8, 10, 12, 14\} &\rightarrow \overline{A} + D; \\ \{2, 6, 10, 14\} &\rightarrow \overline{C} + D. \end{aligned}$$

Dintre acești implicanți primi, numai  $A + \overline{C}$  și  $\overline{C} + D$  sunt implicanți primi esențiali care trebuie considerați pentru acoperirea tuturor maxtermilor din diagrama V-K. Expresia minimă a funcției este:

$$f = (A + \overline{C})(\overline{A} + D) = \overline{A}\overline{C} + AD + \overline{C}D$$

Prin desfacerea parantezelor s-au obținut toți implicanți primi de la forma sumă de produse.

c) Pentru forma sumă de produse din Figura P2.5-c rezultă următorii implicanți primi:

$$\begin{aligned} \{0, 4, 8, 12, 16, 20, 24, 28\} &\rightarrow \overline{D}\overline{E}; \\ \{8, 9, 10, 11, 12, 13, 14, 15\} &\rightarrow \overline{A}B. \end{aligned}$$

Amândoi implicanți primi sunt esențiali, deci acoperirea funcției se realizează prin expresia minimă:

$$f = \overline{D}\overline{E} + \overline{A}B$$



**P2.6**

Rezolvare: e)  $F = \overline{C}\overline{E}$  h)  $F = (A + E)(\overline{A} + C)(\overline{B} + C)$

**P2.7**

Rezolvare: a)  $\overline{A}\overline{B}, \overline{C}D, BD$  b)  $\overline{A}BC, \overline{A}\overline{B}C, CE, B\overline{D}E$

**P2.9**

Rezolvare:

Termenii produs care sunt indiferenți, notați cu  $d$  (*don't care*) pot fi considerați în diagrama V-K fie de valoare 0 fie 1. Se va alege valoarea logică ce va duce la identificarea de implicați primi cu o exprimare cât mai simplă. În diagrama V-K, aceasta înseamnă definirea unor suprafețe cât mai mari și cât mai puține pentru a acoperi căsuțele cu 1.

Pentru funcțiile anterioare se obțin următoarele forme minime: a)  $f = \overline{B}C + AD + \overline{A}B\overline{C}$ ; b)  $f = \overline{C}D + BD$ ; c)  $f = \overline{D}$ ; d)  $f = \overline{C}E + C\overline{E}$ .

**P2.10**

Rezolvare:

O funcție de  $j$  variabile se poate exprima ca o funcție de  $j-1$  variabile și o variabilă reziduu. Variabila reziduu se introduce în expresia funcției reziduu pentru calculul coeficienților. Generarea variabilelor reziduu se poate continua recursiv până când se ajunge la  $j-1$  variabile reziduu și o singură variabilă pentru funcție.

Se va exemplifica pentru funcția de patru variabile:

$$\begin{aligned} f(A, B, C, D) = & (d_0\overline{D} + d_1D)\overline{A}\overline{B}\overline{C} + (d_2\overline{D} + d_3D)\overline{A}\overline{B}C + (d_4\overline{D} + d_5D) \\ & \overline{A}B\overline{C} + (d_6\overline{D} + d_7D)\overline{A}BC + (d_8\overline{D} + d_9D)A\overline{B}\overline{C} + (d_{10}\overline{D} + \\ & + d_{11}D)A\overline{B}C + (d_{12}\overline{D} + d_{13}D)A\overline{B}C + (d_{14}\overline{D} + d_{15}D)ABC \end{aligned}$$

Coeficienții funcției de trei variabile A, B, C se calculează ca o sumă dintre produsul unui coeficient  $d_i$  al funcției de patru variabile cu variabila  $\overline{D}$  și produsul coeficientului următor  $d_{i+1}$  al funcției cu D. Coeficienții  $d_i$  ai funcției de patru variabile se pot deduce fie din tabelul de adevăr, fie din diagrama V-K. Coeficienții funcției de trei variabile calculați din expresiile funcțiilor reziduu pot avea doar valorile: 1, 0, D,  $\overline{D}$ .

De asemenea, pentru funcția de patru variabile, când se introduc în coeficienți două variabile reziduu (C și D), se obține exprimarea:

$$\begin{aligned} f(A, B, C, D) = & (d_0\overline{C}\overline{D} + d_1\overline{C}D + d_2C\overline{D} + d_3CD)\overline{A}\overline{B} + (d_4\overline{C}\overline{D} + d_5\overline{C}D + d_6C\overline{D} + \\ & d_7CD)\overline{A}B + (d_8\overline{C}\overline{D} + d_9\overline{C}D + d_{10}C\overline{D} + d_{11}CD)A\overline{B} + (d_{12}\overline{C}\overline{D} + d_{13}\overline{C}D + d_{14}C\overline{D} + \\ & d_{15}CD)AB \end{aligned}$$

Coeficienții funcției rezultate de două variabile AB au forma:

$$(d_i\overline{C}\overline{D} + d_{i+1}\overline{C}D + d_{i+2}C\overline{D} + d_{i+3}CD)$$

și pot fi calculați ușor ca sumă a patru produse dintre coeficienții consecutivi ai funcției de patru variabile cu mintermii corespunzători ai variabilelor C și D. Acești coeficienți, calculați cu ajutorul unei funcții reziduu, pot avea expresia celor 16 funcții de două variabile dintre care 6 snt valori banale: 0, 1, C,  $\overline{C}$ , D,  $\overline{D}$ .

Introducerea variabilelor reziduu în tabelul de adevăr este prezentată în figura P2.10-a.

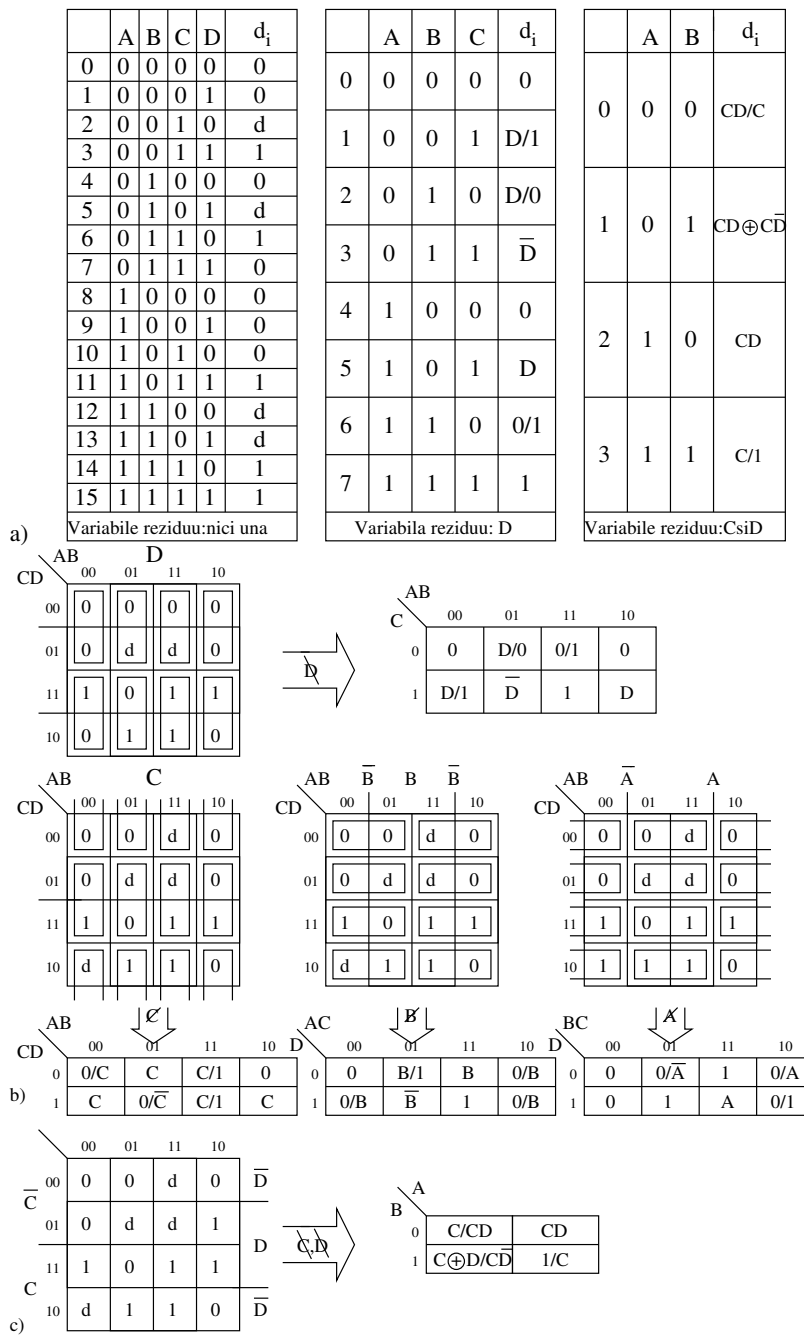


Figura P 2.10

Introducerea unei variabile reziduu (D) în diagramele V-K este prezentată în figura P2.10-b.

$$\begin{aligned}
(0 \cdot \overline{D} + 0 \cdot D) \cdot \overline{A} \overline{B} \overline{C} &= 0 \cdot \overline{A} \overline{B} \overline{C}; \\
(d \cdot \overline{D} + 1 \cdot D) \cdot \overline{A} \overline{B} C &= (D/1) \cdot \overline{A} \overline{B} C; \\
(0 \cdot \overline{D} + d \cdot D) \cdot \overline{A} B \overline{C} &= (D/0) \cdot \overline{A} B \overline{C}; \\
(1 \cdot \overline{D} + 0 \cdot D) \cdot \overline{A} B C &= \overline{D} \cdot \overline{A} B C; \\
(0 \cdot \overline{D} + 0 \cdot D) \cdot A \overline{B} \overline{C} &= 0 \cdot A \overline{B} \overline{C}; \\
(0 \cdot \overline{D} + 1 \cdot D) \cdot A \overline{B} C &= \overline{D} \cdot A \overline{B} C; \\
(d \cdot \overline{D} + d \cdot D) \cdot A B \overline{C} &= (0/1) \cdot A B \overline{C}; \\
(1 \cdot \overline{D} + 1 \cdot D) \cdot A B C &= 1 \cdot A B C;
\end{aligned}$$

Introducerea a două variabile reziduu (C,D) în diagramele V-K este prezentată în figura P2.10-c.

$$\begin{aligned}
(0 \cdot \overline{C} \overline{D} + 0 \cdot \overline{C} D + d \cdot C \overline{D} + 1 \cdot C D) \cdot \overline{A} \overline{B} &= (C/CD) \cdot \overline{A} \overline{B}; \\
(0 \cdot \overline{C} \overline{D} + d \cdot \overline{C} D + 1 \cdot C \overline{D} + 0 \cdot C D) \cdot \overline{A} B &= (C \oplus D/C \overline{D}) \cdot \overline{A} B; \\
(0 \cdot \overline{C} \overline{D} + 0 \cdot \overline{C} D + 0 \cdot C \overline{D} + 1 \cdot C D) \cdot A \overline{B} &= (CD) \cdot A \overline{B}; \\
(d \cdot \overline{C} \overline{D} + d \cdot \overline{C} D + 1 \cdot C \overline{D} + 1 \cdot C D) \cdot A B &= (1/C) \cdot A B;
\end{aligned}$$

## P2.11

*Rezolvare:*

Operația de minimizare pe diagrama V-K se realizează în trei pași.

*Pasul 1:* Se grupează căsuțele care au valori logice 1 și cele care au valori indifferente ( $d$ ) pentru a se exprima cât mai simplu implicanții primi.

*Pasul 2:* Căsuțele cu valoare 1 se consideră a fi cu valoare  $d$ . Se grupează căsuțele care conțin aceeași funcție reziduu și cu unele căsuțe (potrivite) care conțin valori indifferente pentru a exprima cât mai simplu implicanții primi. Implicanții primi rezultați se vor înmulți cu funcția reziduu.

*Pasul 3:* Forma minimă a funcției se obține prin sumarea logică a implicanților primi obținuți de la primul și al doilea pas. Dacă funcțiile reziduu au mai mult de o variabilă, în cazuri particulare, expresia se mai poate reduce prin prelucrări analitice.

Aplicând succesiv cei trei pași pentru fiecare funcție din diagramele date se obțin formele minime prezentate în Figura P2.11.

Se observă că pentru  $F_5$  se obțin două expresii diferite. Acestea se explică prin modul cum se grupează căsuțele în care există valori indifferente  $d$  sau ce valoare se atribuie acesteia. Toate expresiile sunt corecte deoarece pentru acele combinații ale variabilelor, cărora le corespunde  $d$  în diagrama V-K, valoarea funcției este indiferentă.

Funcția	Pasul 1	Pasul 2	Pasul 3 ( forma minima )
$f_1$	AB	AC	AB + AC
$f_2$	—	BC	BC
$f_3$	$\bar{A}B$	$\bar{A}BC$	$\bar{A}B + \bar{A}\bar{B}C$
$f_4$	$\bar{B}$	$\bar{A}C + AD$	$B + \bar{A}C + AD$
$f_5$	A	$\bar{C}D$ sau BD	$A + \bar{C}D$ sau $A + BD$
$f_6$	B + AC	AE + CD	$\bar{B} + AC + AE + CD$
$f_7$	$B\bar{C} + A$	$\bar{B}CD$	$B\bar{C} + A + \bar{B}CD$
$f_8$	AB	BDE + $\bar{A}CF$	$AB + BDE + \bar{A}\bar{C}\bar{E}$
$f_9$	BC	$\bar{A}DE + \bar{A}\bar{C}\bar{E}$	$BC + \bar{A}DE + \bar{A}\bar{C}\bar{E}$
$f_{10}$	AC	$\bar{B}\bar{D}E + ABF$	$AC + \bar{B}\bar{D}E + ABF$

b)

Figura P 2.11

**P2.13***Rezolvare:*

Tabelul de adevăr pentru conversia BCD 2-4-2-1 la matricea de șapte segmente este prezentat în Figura P2.13-a. În mulțimea cuvintelor de intrare nu apar niciodată combinațiile care ar defini mintermi:  $P_2, P_3, P_4, P_5, P_6, P_7, P_8$ , respectiv maxtermii:  $S_2, S_3, S_4, S_5, S_6, S_7$ . Funcțiile care realizează comanda segmentelor matricei, exprimate ca sumă de produse sunt:

$$A = \sum_0^{15} (1, 10) + \sum_0^{15} d(2, 3, 4, 5, 6, 7);$$

$$B = \sum_0^{15} (11, 12) + \sum_0^{15} d(2, 3, 4, 5, 6, 7);$$

$$C = \sum_0^{15} (8) + \sum_0^{15} d(2, 3, 4, 5, 6, 7);$$

$$D = \sum_0^{15} (1, 10, 13) + \sum_0^{15} d(2, 3, 4, 5, 6, 7);$$

$$E = \sum_0^{15} (1, 9, 10, 11, 13, 15) + \sum_0^{15} d(2, 3, 4, 5, 6, 7);$$

$$F = \sum_0^{15} (1, 8, 9, 13) + \sum_0^{15} d(2, 3, 4, 5, 6, 7);$$

$$G = \sum_0^{15} (0, 1, 13) + \sum_0^{15} d(2, 3, 4, 5, 6, 7).$$

Aceleași funcții, exprimate ca produse de sume sunt:

$$A = \prod_0^{15} (0, 8, 9, 11, 12, 13, 14, 15) + \prod_0^{15} d(2, 3, 4, 5, 6, 7);$$

$$B = \prod_0^{15} (0, 1, 8, 9, 10, 13, 14, 15) + \prod_0^{15} d(2, 3, 4, 5, 6, 7);$$

$$C = \prod_0^{15} (0, 1, 9, 10, 11, 12, 13, 14, 15) + \prod_0^{15} d(2, 3, 4, 5, 6, 7);$$

$$D = \prod_0^{15} (0, 8, 9, 11, 12, 14, 15) + \prod_0^{15} d(2, 3, 4, 5, 6, 7);$$

$$E = \prod_0^{15} (0, 8, 12, 14) + \prod_0^{15} d(2, 3, 4, 5, 6, 7);$$

$$F = \prod_0^{15} (0, 10, 11, 12, 14, 15) + \prod_0^{15} d(2, 3, 4, 5, 6, 7);$$

$$G = \prod_0^{15} (8, 9, 10, 11, 12, 14, 15) + \prod_0^{15} d(2, 3, 4, 5, 6, 7).$$

Minimizarea acestor funcții se face pe diagramele V-K de patru variabile prezentate în Figura P2.13-b.

S-au dedus formele minime atât ca produse de sume cât și ca sume de produse. Prin aplicarea dublei negații formei de sumă de produse (obținută din *Forma Normală Disjunctivă* - FND), se poate implementa funcția cu operatori NAND. Similar, prin aplicarea dublei negații formei minime de produs de sume (obținută din *Forma Normală Conjunc-*



CIFRA ZECIM	BCD 2421 INTRARI w x y z	Termeni Min P <sub>i</sub> Max S <sub>i</sub>	IESIRI Codul 7 segmente A B C D E F G	Iesire	Porti NAND suma de produse	Porti NOR produs de sume
0	0 0 0 0	P <sub>0</sub> S <sub>0</sub>	0 0 0 0 0 0 1	A	$\overline{WZ} + \overline{XYZ}$	$\overline{X(Y+Z)(\overline{W}+Z)}$
1	0 0 0 1	P <sub>1</sub> S <sub>1</sub>	1 0 0 1 1 1 1	B	$X\overline{YZ} + \overline{X}YZ$	$(X+Y)(\overline{X+Z})(\overline{Y}+Z)$
2	1 0 0 0	P <sub>8</sub> S <sub>8</sub>	0 0 1 0 0 0 1 0	C	$W\overline{X}YZ$	$\overline{W\overline{X}YZ}$
3	1 0 0 1	P <sub>9</sub> S <sub>9</sub>	0 0 0 0 1 1 0 0	D	$X\overline{YZ} + \overline{WZ} + \overline{X}YZ$	$(\overline{W}+X+Y)(\overline{Y}+\overline{Z})(\overline{X}+\overline{Y})(Y+Z)$
4	1 0 1 0	P <sub>10</sub> S <sub>10</sub>	1 0 0 1 1 1 0 0	E	$Z + \overline{X}Y$	$(\overline{X}+Z)(Y+Z)$
5	1 0 1 1	P <sub>11</sub> S <sub>11</sub>	0 1 0 0 1 0 0 0	F	$W\overline{X}\overline{Y} + \overline{Y}Z$	$\overline{Y}(\overline{X}+Z)(W+Z)$
6	1 1 0 0	P <sub>2</sub> S <sub>2</sub>	0 1 0 0 0 0 0 0	G	$X\overline{YZ} + \overline{W}$	$\overline{Y}(\overline{W}+Z)(\overline{W}+\overline{X})$
7	1 1 0 1	P <sub>3</sub> S <sub>3</sub>	0 0 0 0 1 1 1 1			
8	1 1 1 0	P <sub>4</sub> S <sub>4</sub>	0 0 0 0 0 0 0 0			
9	1 1 1 1	P <sub>5</sub> S <sub>5</sub>	0 0 0 0 0 1 0 0			

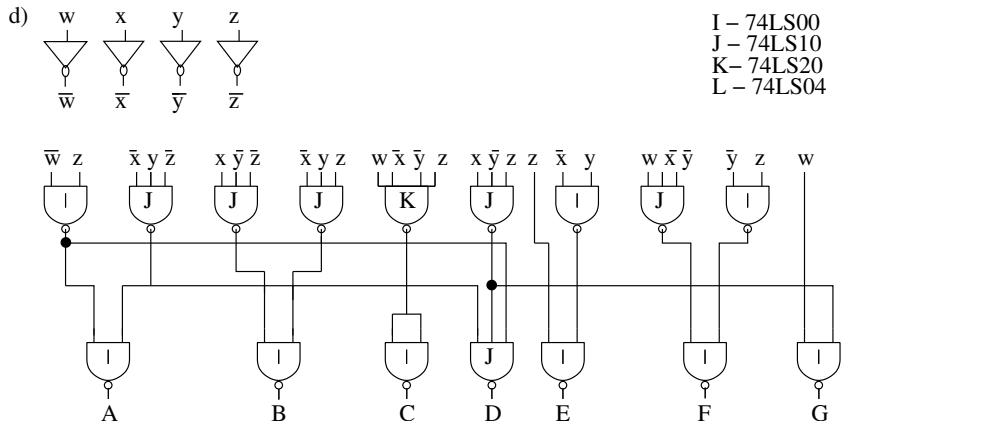
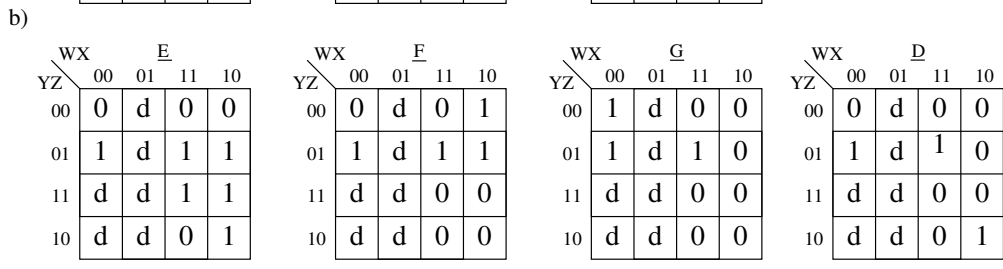
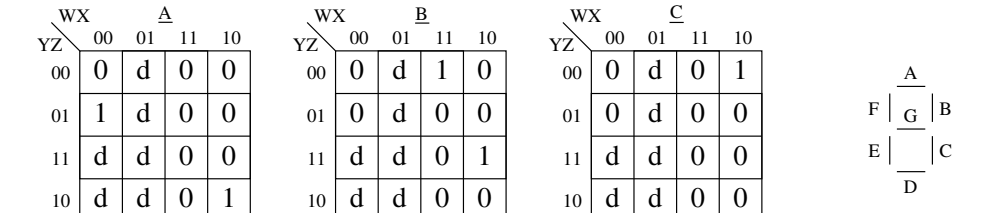


Figura P 2.13

*tivă - FNC*), se poate implementa funcția cu operatori NOR. Expresiile minimizate ale funcțiilor, pentru o implementare cu operatori NAND și pentru o implementare cu porți NOR sunt prezentate în Figura P2.13-c.

Structura cu porți NAND care implementează funcțiile este prezentată în Figura P2.13-d.

### P2.15

*Rezolvare:*

Circuitele AND-OR și NAND-NOR, care calculează termeni produs sunt potențiale de producere de hazard static 1 deoarece ieșirea lor 1 (sinteza formei normale disjunctive a funcției se face pe bază de 1) poate genera un glitch 0 când o variabilă  $x$  comută de la 1 la 0. Hazardul apare dacă, pentru anumite valori constante ale celorlalte variabile, funcția

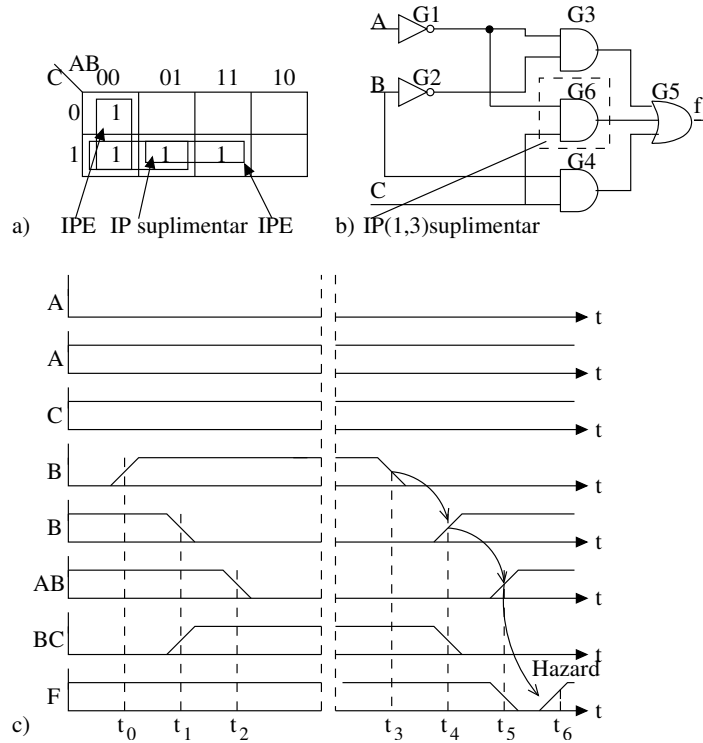


Figura P 2.15

se reduce la:  $x + \overline{x} = 1$ . Pentru circuitul considerat, funcția  $F(A, B, C) = \overline{A}\overline{B} + BC$  se reduce la  $F(0, B, 1) = \overline{B} + B$ , pentru  $A=0$  și  $C=1$ . Ca efect, implementarea propusă va determina apariția unui glitch 0, când B comută de la 1 la 0.

În diagramele de timp din Figura P2.15-c se observă că o comutație de la 0 la 1 a lui B la momentul  $t_0$  nu produce hazard, dar o comutație de la 1 la 0 la momentul  $t_3$  generează un interval  $t_5 - t_6$  când ieșirea devine 0, ( $B + \overline{B} \neq 1$ ). Similar, în diagrama V-K din Figura P2.15-a, când “punctul de funcționare” se deplasează între IPE(0,1) și IPE(3,7), nu apare

hazard deoarece B comută între 0 și 1 dar apare hazard când se deplasează, între IPE(3,7) și IPE(0,1) pentru că B comută de la 1 la 0.

Soluția de eliminare a hazardului este introducerea implicantului prim IP(1,3), “punte“ între cei doi IPE. Acest IP menține funcția la valoarea 1 chiar când B comută de la 1 la 0. Această “punte“ este implementată în Figura P2.15-c (prin introducerea porții  $G_6$ ).

## P2.16

*Rezolvare:*

Circuitele OR-AND și NOR-NOR, care calculează termeni sumă sunt potențiale de producere de hazard static 0 deoarece ieșirea lor 0 (sinteza formei normale conjunctive a funcției se face pe bază de 0) poate genera un glitch la 1 când o variabilă comută de la 0 la 1, dacă pentru anumite valori constante ale celorlalte variabile, funcția se reduce la forma:  $x \cdot \bar{x} = 0$ .

Minimizarea circuitului se realizează cu trei impicanți primi esențiali, prezentați în Figura P2.16.

$$IPE(1, 3, 5, 7) = (A + \bar{D}),$$

$$IPE(10, 11, 14, 15) = \bar{A} + \bar{C},$$

$$IPE(4, 12) = (\bar{B} + C + D).$$

Acestora le corespunde expresia:

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

Când A=0, B=1 și C=0, funcția se reduce la  $F = D\bar{D}$ . La trecerea de la maxtermul 5 ( $A + \bar{B} + C + D$ ), la maxtermul 4 ( $A + \bar{B} + C + \bar{D}$ ) nu se produce hazard, dar la trecerea de la maxtermul 4 la maxtermul 5 se produce hazard deoarece D comută din 0 în 1. Hazardul se elimină prin introducerea implicantului prim  $IP(4, 5) = (\bar{A} + \bar{B} + D)$ .

Între IPE(10,11,14,15) și IPE(4,12), când A=1, B=1 și D=0, funcția se reduce la  $F = C\bar{C}$ . La trecerea de la maxtermul 14 ( $\bar{A} + \bar{B} + \bar{C} + D$ ) la maxtermul 12 ( $\bar{A} + \bar{B} + C + D$ ) nu se produce hazard, dar la trecerea de la maxtermul 12 la maxtermul 14 se produce hazard deoarece variabila C comută din 0 în 1 (Figura P2.16-b). Hazardul se elimină prin introducerea implicantului prim  $IP(12, 14) = (\bar{A} + \bar{B} + D)$ .

Între IPE(10,11,14,15) și IPE(1,3,5,7), când B=1, C=1 și D=1, funcția se reduce la  $F = A\bar{A}$ . La trecerea de la maxtermul 15 ( $\bar{A} + \bar{B} + \bar{C} + \bar{D}$ ) la maxtermul 7 ( $A + \bar{B} + \bar{C} + \bar{D}$ ) nu se produce hazard, dar la trecerea de la maxtermul 7 la maxtermul 15 se produce hazard deoarece variabila comută din 0 în 1. Hazardul se elimină prin introducerea implicantului prim  $IP(3, 7, 11, 15) = \bar{C} + D$ . Aici s-a luat în considerare și comutația de la maxtermul 3 la maxtermul 11.

În Figura P2.16-c este reprezentată implementarea care elimină hazardul static 0 pentru funcția dată.

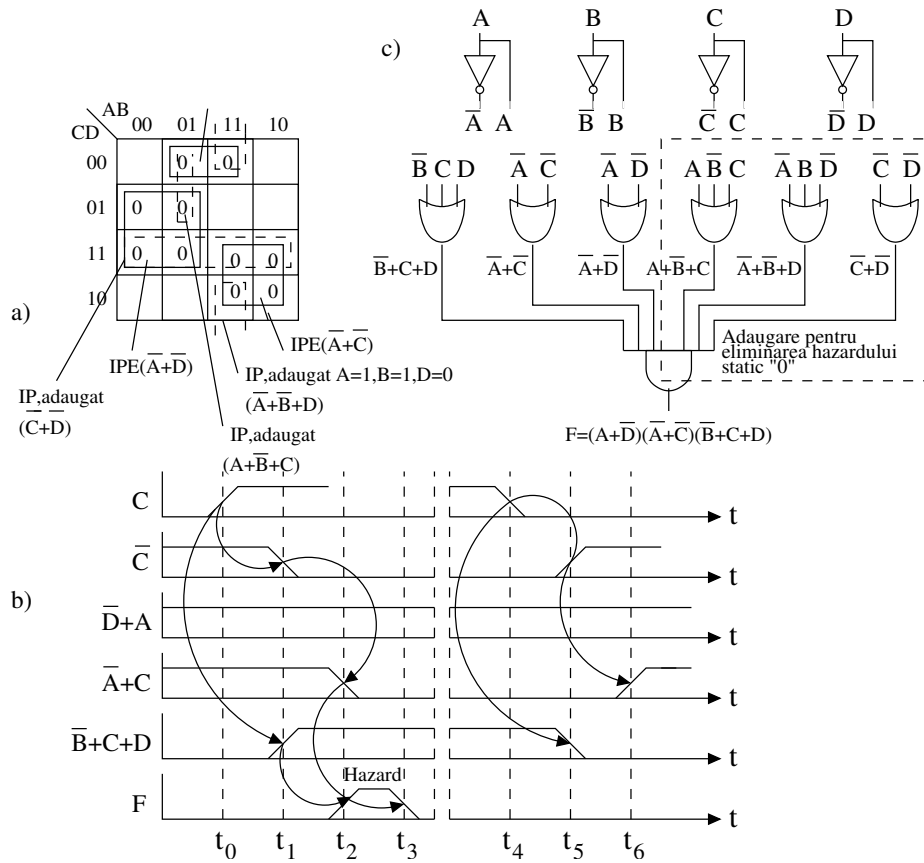


Figura P 2.16

**P2.17***Rezolvare:*

Pentru valori constante ale tuturor variabilelor funcției, în afară de variabila  $x_i$ , dacă funcția de tipul FD poate fi adusă la expresia  $x_i + \bar{x}_i = 1$  iar funcția de tipul FC poate fi adusă la expresia  $x_i \bar{x}_i = 0$  atunci poate apare hazard. Fizic, la comutația variabilei  $x_i$  de la 1 la 0 există un interval, datorat inversorului  $\bar{x}_i$ , când  $x_i + \bar{x}_i = 0$ ! iar pentru comutarea variabilei  $x_i$  de la 0 la 1 există un interval când  $x_i \bar{x}_i = 1$ !

**P2.19***Rezolvare:*

Din structura circuitului se deduce expresia funcției logice:

$$\begin{aligned}
 f &= \overline{\overline{AB} \cdot A \cdot C \cdot \overline{AB}} = \overline{\overline{AB} \cdot A \cdot C} + AB = (AB + \bar{A}) \cdot C + AB = \\
 &= ABC + \bar{A}C + AB = (ABC + AB) + \bar{A}C = AB(C + 1) + \bar{A}C = AB + \bar{A}C.
 \end{aligned}$$

Diagrama V-K din Figura P2.19-a indică faptul că se poate genera hazard static 1. Când se trece de la mintermul 7 ( $A=1, B=1$  și  $C=1$ ), la mintermul 3 ( $A=0, B=1$  și  $C=1$ ), funcția se reduce la expresia  $F = A + \bar{A}$  care ia valoarea nepermisă 0 la comutația variabilei A de la 1 la 0.

Hazardul se poate elimina prin adăugarea implicantului prim BC obținându-se funcția  $F = AB + \bar{A}C + BC$  cu implementarea din Figura P2.19-b.

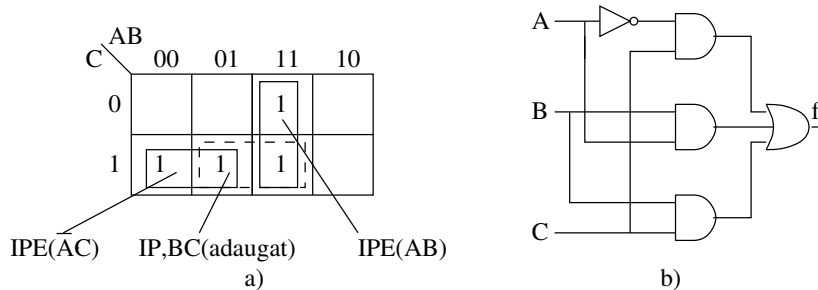


Figura P 2.19

## P2.20

*Rezolvare:*

Tabelul de adevăr al funcției este dat în Figura P2.20-a din care se deduce arborele de decizie binar din Figura P2.20-b. Apoi, aplicând regula de eliminare a nodurilor terminale duplicate, se obține OBDD numai cu două noduri terminale ca în Figura P2.20-c. Continuând cu reducerea, eliminarea nodurilor neterminale duplicate, două din nodurile  $x_1$  având arcele  $low(x_1)$  la nodul terminal 0 și  $high(x_2)$  la nodul terminal pot fi excluse și se obține OBDD din Figura P2.20-d. Și prin ultima transformare, eliminarea nodurilor neterminale care au  $low(x) = high(x)$ , un nod al variabilei  $x_1$  și unul al variabilei  $x_0$  pot fi neglijate și se obține forma redusă de OBDD din Figura P2.20-e.

## P2.21

*Rezolvare:*

Din reprezentarea pe diagrama V-K din figura P2.21-a se deduce că acoperirea ce duce la un număr minim de porți NAND este: BD dar nu ABDC și AC dar nu ABCD deci funcția f se exprimă:

$$f = BD\overline{(ABCD)} + AC\overline{(ABCD)}$$

cu implementarea din figura P2.21-b.

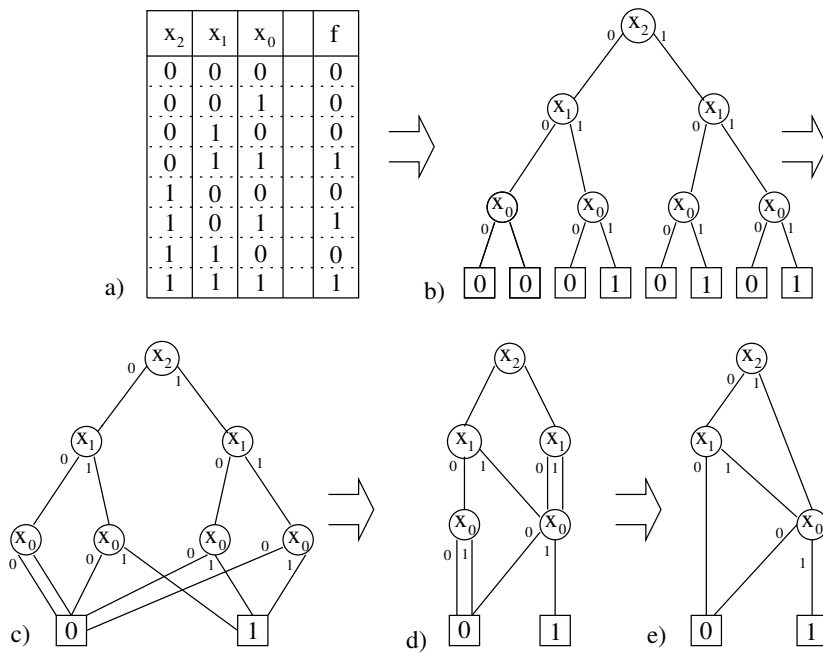


Figura P 2.20

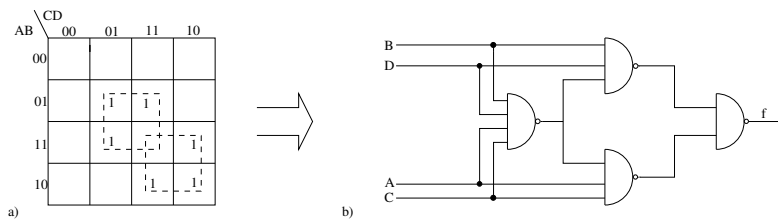


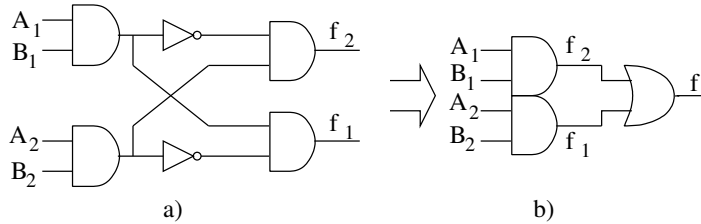
Figura P 2.21

**P2.22***Rezolvare:*

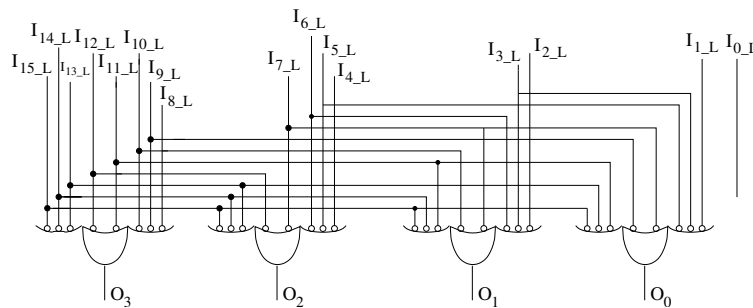
O definiție posibilă pentru DAR este “ $f_1$  este adevărată dacă  $A_1$  și  $B_1$  sunt adevărate DAR fie  $A_2$  sau  $B_2$  sunt false;  $f_2$  este adevărată dacă  $A_2$  și  $B_2$  sunt adevărate DAR fie  $A_1$  sau  $B_1$  sunt false”. Din tabelul de adevăr (de patru variabile), prin minimizare cu diagrama V-K, rezultă

$$f_1 = A_1 B_1 \overline{(A_2 B_2)} \quad ; \quad f_2 = A_2 B_2 \overline{(A_1 B_1)}$$

cu structura din Figura P2.22-a. Pentru funcția  $f$  rezultă implementarea din Figura P2.22-b (analizând tablele de adevăr pentru  $f_1$  și  $f_2$  rezultă că funcția dată poate fi exprimată ca:  $f = f_1 \cup f_2$ ).

**Figura P 2.22****P2.23***Rezolvare:*

Codificatorul este un nivel de OR, în cazul acesta necesită patru porți OR cu 8 intrări (la fiecare poartă se aplică doar 8 din cele 16 intrări  $I_i, i = 0, 1, 2 \dots 15$ ). Deoarece  $A + B = \overline{\overline{A} \cdot \overline{B}}$  adică operatorul OR este echivalent cu operatorul NAND cu intrările negate, implementarea cu 4 porți NAND cu 16 intrări este posibilă, iar semnalele de intrare sunt active în starea low,  $I_{i,L}$ , dar ieșirile sunt active în starea H, ca în Figura P2.23.

**Figura P 2.23**

**P2.24***Rezolvare:*

Circuitul 74XX148, al cărui tabel de adevăr și simbol de reprezentare sunt date în Figura P2.24-a și b, are o intrare de validare  $EI$  și opt intrări ( $I_7, \dots, I_0$ ;  $I_7$  are prioritatea cea mai ridicată) active în stare “low”. Circuitul generează pe ieșirea  $A_{2L}A_{1L}A_{0L}$  numărul intrării de prioritate cea mai ridicată care a fost activată. Numărul intrării este exprimat în binar, în complement față de 1, deoarece ieșirile sunt active în starea “low”. Ieșirea  $GS$  este activată (“0”) când cel puțin o intrare este activată. Ieșirea  $EO$  este activată (“0”) când circuitul este validat de către semnalul  $EIL$  dar nici o intrare nu a fost activată. Semnalul  $EOL$  poate fi utilizat pentru a cascada mai multe circuite, în scopul obținerii unei structuri de codificator cu mai multe intrări.

*Observație:*

Printr-un artificiu de conectare a semnalelor  $I_0, I_1 \dots I_7$  ( $I_7$  fiind evenimentul cu nivelul cel mai ridicat de prioritate), la cele opt intrări în felul următor:  $I_0 \rightarrow 7, I_1 \rightarrow 6 \dots I_7 \rightarrow 0$ , cuvântul de ieșire  $y_{2L}y_{1L}y_{0L}$  va indica în binar (necomplementat) intrarea activată cu prioritatea cea mai scăzută. Aceasta este o modalitate de realizare a unui codificator pentru prioritatea cea mai scăzută.

Utilizarea celui de al doilea codificator prioritar de opt intrări, 74XX148 pentru structurarea codificatorului prioritar de 16 intrări va implica la subcuvântul de ieșire  $y_{2L}y_{1L}y_{0L}$  adăugarea încă a unui bit  $y_{3L}$ . Acest bit  $y_{3L}$  va fi generat doar atunci când există cel puțin o intrare activă în intervalul  $A_{15L} \dots A_{8L}$ , ceea ce determină totdeauna pentru al doilea circuit 74XX148<sub>2</sub> să activeze semnalul  $GS_L$ . Deci, pentru generarea celui de al patrulea bit de ieșire se consideră  $y_{3L} \equiv GS_L$ .

Pentru structurarea serie, figura P2.24-c, al doilea circuit este validat permanent prin conectarea semnalului  $EIL$  la masă. În schimb, primul circuit este necesar să fie validat de către al doilea doar atunci când nici o intrare din intervalul  $A_{15L} \dots A_{8L}$  nu este activată (ieșirea  $EO$  devine activă) ceea ce se poate realiza prin conectarea  $EO \equiv EI$ . Biții  $y_{2L}, y_{1L}, y_{0L}$  ai subcuvântului de ieșire sunt colectați fie de la ieșirile  $O_2, O_1, O_0$  ale primului codificator, când sunt activate intrări numai din intervalul  $A_7 \dots A_0$ , SAU de al doilea codificator, când sunt activate intrări din intervalul  $A_{15} \dots A_8$ . Deci, implementarea se face cu porți NAND (deoarece semnalele de la codificatoarele sunt active în starea low și cuvântul de ieșire este în complement față de unu). La această structurare serie, timpul de propagare fiind prin cele două codificatoarele, viteza circuitului este scăzută. O îmbunătățire se obține prin structurarea paralelă.

Structurarea paralelă este reprezentată în figura P2.24-d. Validarea celor două circuite este permanentă prin legarea intrărilor  $EI$  la masă. Bitul  $y_{3L}$  din cuvântul de ieșire este identic cu semnalul  $GS_L$  de la al doilea circuit 74XX148<sub>2</sub>, la fel ca la structurarea serie. În plus, acest semnal  $GS_L$  este folosit și pentru selectarea subcuvântului  $O_2O_1O_0$  de la primul circuit când sunt activate intrări numai din intervalul  $A_7 \dots A_0$  ( $GS_L = 1$ ) sau de la al doilea circuit când sunt activate intrări și din intervalul  $A_{15} \dots A_8$  ( $GS_L = 0$ ). Implementarea selectării este realizată cu circuitul 74LS157 ( $4 \times MUX2 : 1$ )



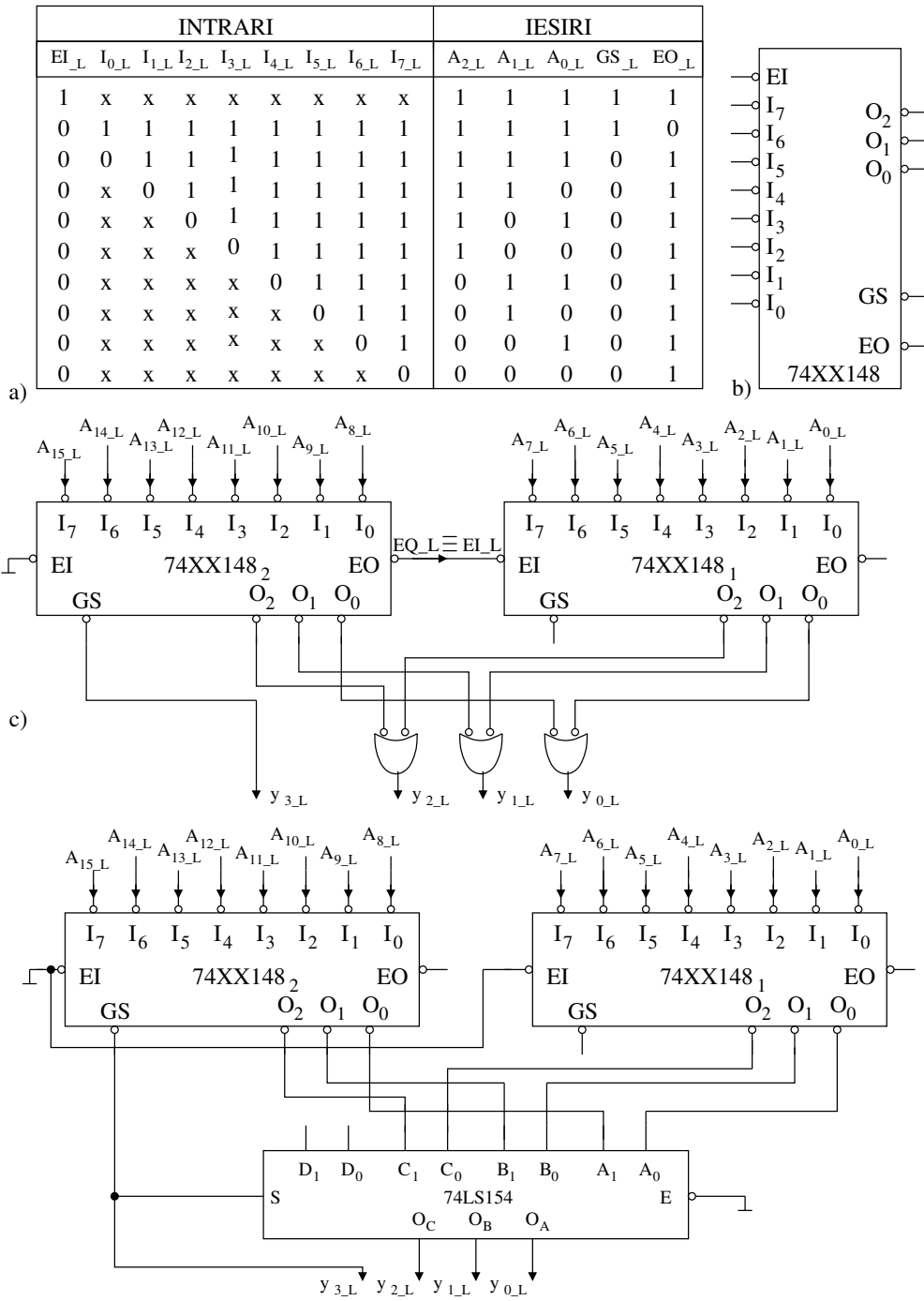


Figura P 2.24

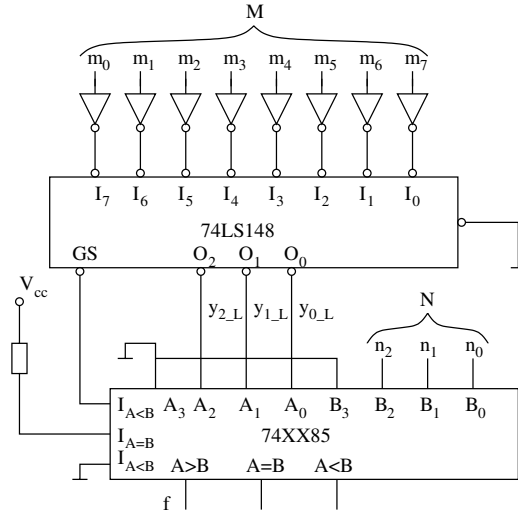
**P2.25***Rezolvare:*

Numărul  $2^N$  se exprimă printr-un cuvânt format de 1 urmat de N zerouri. Numărul binar M este multiplu al numărului  $2^N$  dacă este îndeplinită relația:

$$M = K \cdot 2^N$$

pentru:  $2^N \in \{0, 2, 4, 8, 16, 32, 64, 128\}$ ,  $K \in [0, 255]$  întreg.

Relația este îndeplinită când dintre cei opt biți  $m_7 \dots m_0$  ai cuvântul M, cel puțin ultimii



**Figura P 2.25**

N biți sunt zero. Soluția problemei se reduce la a determina dacă numărul M aplicat pe o intrare are cel puțin cei mai puțin semnificativi N biți egali cu "0" când pe cealaltă intrare se aplică cuvântul N. Determinarea numărului de zerouri din pozițiile cele mai puțin semnificative ale cuvântului M este echivalentă cu aflarea numărului poziției primului bit 1, începând cu  $m_0$ . Implementarea acestei soluții apare ca evidentă cu ajutorul circuitului comparator digital 7485: pe o intrare A a comparatorului se aplică numărul poziției  $m_i$  al primului bit egal cu 1 din cuvântul M, începând cu poziția  $m_0$ , iar pe intrarea B se aplică cuvântul N. La ieșirea  $A > B$  se obține funcția f, așa ca în Figura P2.25.

Circuitul codificator prioritar, dacă se consideră în binar cuvântul de intrare I și cuvântul de ieșire O, implementează relația:

$$O = \lfloor \log_2 I \rfloor$$

Circuitul va genera la ieșire numărul O al poziției activate, cu prioritatea cea mai ridicată în cuvântul de intrare I.

Conform observației de la problema P2.24, inversând ordinea de conectare a biților  $m_7 \dots m_0$  la intrările  $I_7 \dots I_0$  ale codificatorului prioritar 74LS148, acesta va calcula la

ieșire, în cod binar natural și nu complementat, poziția primului bit activ pe intrare cu prioritatea cea mai scăzută, ceea ce este necesar în această problemă. Ieșirile  $O_2, O_1$  și  $O_0$  ale codificatorului se conectează la intrările  $A_2, A_1, A_0$  ale comparatorului digital. Biții cuvântului  $M$  se aplică prin inversoare la intrările comparatorului 7485 deoarece intrările acestuia sunt active în stare “L”.

### P2.27

*Rezolvare:*

Soluția acestei probleme se bazează pe Observația de la problema P2.24. Structura circuitului conține pe două codificatoare prioritare: unul calculează prioritatea cea mai ridicată  $74XX148_1$ , în cod complementar față de 1, iar celălalt  $74XX148_2$  prioritatea cea mai scăzută, Figura P2.27. Când în cuvântul de intrare  $X$  există doar un singur bit zero, bitul  $B_3$  devine 0, deoarece  $B_3 = GS_L = 0$ , prin sumarea celor două cuvinte de cod de trei biți (unul complementat față de 1, iar celălalt necomplementat) se obține pentru  $s_2s_1s_0 = 111 + C_{-1} = 111 + 1 = 000$  plus 1 transfer la  $s_3$ , iar  $s_3 = A_3 + B_3 + 1 = 1$ , deci  $s_3s_2s_1 = 1000$ . În cazul în care nu este activată nici o intrare,  $GS_L = 1$ , cuvintele de cod sunt 111 iar prin sumarea  $1111 + 0111 + 1$  se obține  $s_3s_2s_1s_0 = 0111$  și  $C_3 = 1$ . Când în cuvântul de intrare există mai multe zerouri suma celor două cuvinte de cod este mai mică decât 111 la care adunat  $C_{-1}$  nu va da transfer la  $s_3$  deci ieșirea  $f = s_3$  va fi zero. Deci detectarea singurului zero în cuvântul de intrare se realizează cu ieșirea  $s_3$  iar numărul binar,  $z_2z_1z_0$ , care exprimă poziția zero-ului în acest cuvânt este generat de  $74XX148_2$ .

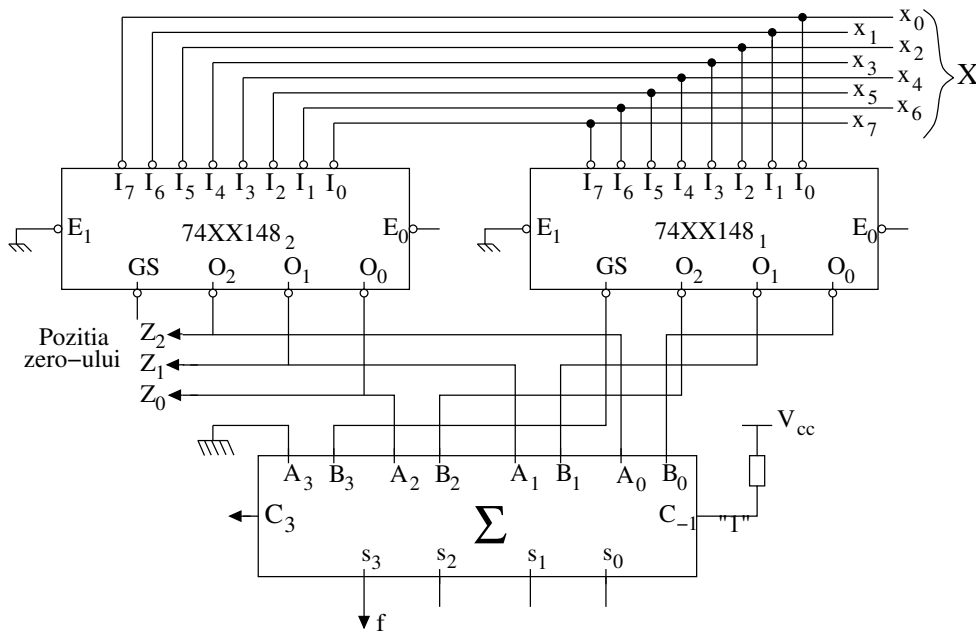


Figura P 2.27

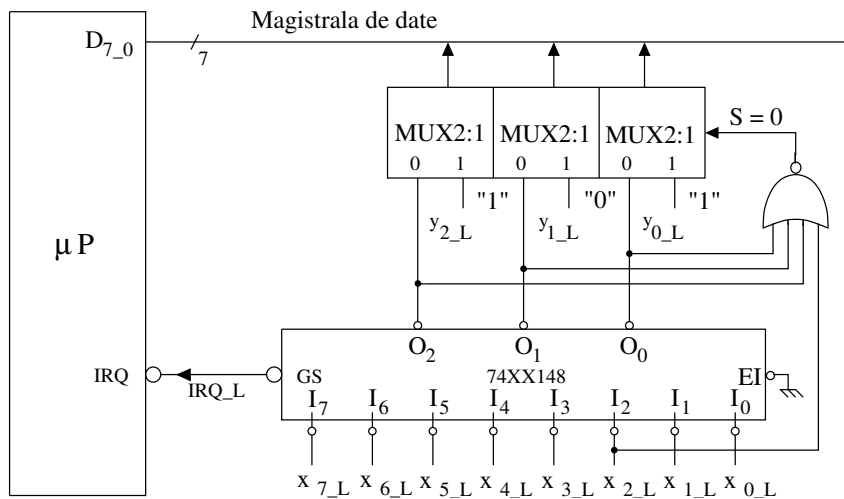
**P2.28***Rezolvare:*

Codarea poziției zeroului celui mai semnificativ pe intrare se poate realiza cu un codificator prioritar (74LS148), vezi Figura P2.24, iar decodarea pentru ieșire se poate realiza cu un circuit decodificator (7442).

**P2.29***Rezolvare:*

Structura circuitului este dată în Figura P.2.29. Codificatorul prioritar 74XX148 generează pe magistrala de date a  $\mu P$  codul perifericului activat  $Y_{2L}Y_{1L}Y_{0L}$  cu prioritatea cea mai ridicată iar semnalul de întrerupere este generat de GS,  $IRQ_L=GS_L$ . Selectarea codului pe magistrala de date se realizează cu  $3 \times MUX2:1$  : 1 din două surse:

- 1- de la ieșirea circuitului 74XX148 și corespunde perifericului activat cu prioritatea cea mai ridicată, pentru  $s=0$ ;
- 2- o valoare cablată 101 când  $x_{7L} = 0$  SI  $x_{2L} = 0$  pentru  $s=1$ . Valoarea  $s=1$  se obține când  $y_{2L} = 0$ ,  $y_{1L} = 0$ ,  $y_{0L} = 0$  SI  $x_{2L} = 0$  adică  $\overline{y_{2L} + y_{1L} + y_{0L} + x_{2L}} = 1$ , ceea ce se realizează cu un NOR4.

**Figura P 2.29****P2.30***Rezolvare:*

Tabelul de adevăr și simbolul de reprezentare pentru circuitul 74XX138 sunt date în Figura P2.30-a și 2.30-b. Pentru un cuvânt de cod cu lungimea de trei biți aplicat pe

intrările A,B,C se activează doar una din ieșirile  $y_{i,L}$ ,  $i = 0,1 \dots 7$ . Aceasta decodificare este validată de conjunția a trei semnale de validare  $G_1, G_{2A,L}$  și  $G_{2B,L}$ , adică  $G_1 \cdot \overline{G_{2A,L}} \cdot \overline{G_{2B,L}} = 1$  ceea ce se realizează când  $G_1 = 1, G_{2A,L} = 0$  și  $G_{2B,L} = 0$ . Structurarea pentru un DCD5:32 este prezentată în Figura P2.30-c.

Circuitele 74XX138 sunt organizate pe două niveluri de decodificare, pe primul nivel un circuit iar pe al doilea nivel patru circuite. La fiecare din intrările A,B,C de pe nivelul al doilea se aplică cei trei biți  $x_2, x_1, x_0$  mai puțin semnificativi din cuvântul de intrare. Primii doi biți mai semnificativi  $x_4$  și  $x_3$  sunt aplicați la decodicatorul din primul nivel și de la acesta sunt utilizate doar patru ieșiri care sunt aplicate ca semnale de validare  $G_{2B,L}$  la cele patru circuite din nivelul doi. Prin acest semnal de validare  $G_{2B,L}$  mulțimea ieșirilor  $y_{31,L}, y_{30,L} \dots y_{1,L}, y_{0,L}$  este împărțită în patru intervale 31-24, 23-16, 15-8, 7-0 și asigurat câte un interval unui circuit decodificator. Precauție trebuie acordată la factorul de încărcare al semnalelor de validare de pe nivelul al doilea.

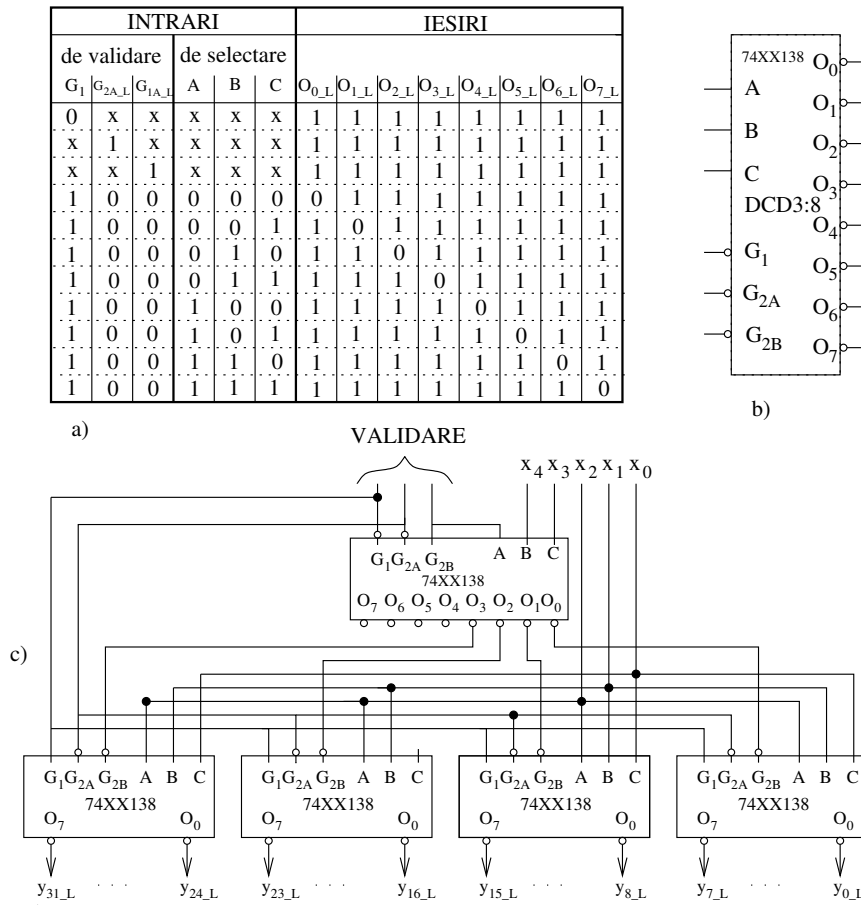
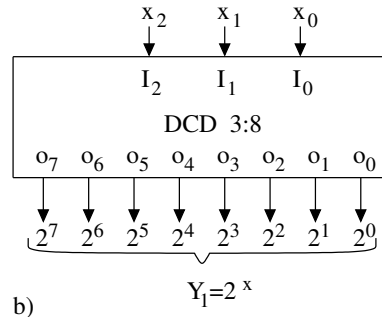


Figura P 2.30

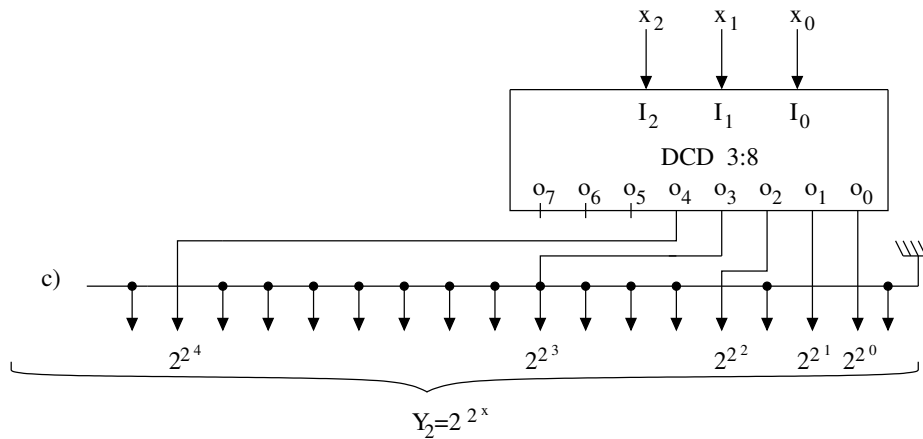
**P2.31***Rezolvare:*

Realizarea funcției  $Y_1$  pe un DCD3:8 este o implementare banală, Figura P2.31-b, deoarece

x	$Y_1=2^x$	$Y_2=2^{2^x}$
0	1	10
1	1 0	100
2	1 00	10000
3	1 000	100000000
4	1 0000	1000000000...0 16 ori
5	1 00000	1000000000...0 32 ori
6	1 000000	1000000000...0 64 ori
7	1 0000000	1000000000...0 128 ori



a)

**Figura P 2.31**

tocmai funcția de exponențiere este realizată de decodificator dacă se consideră cuvintele de intrare și de ieșire ca numere binare.

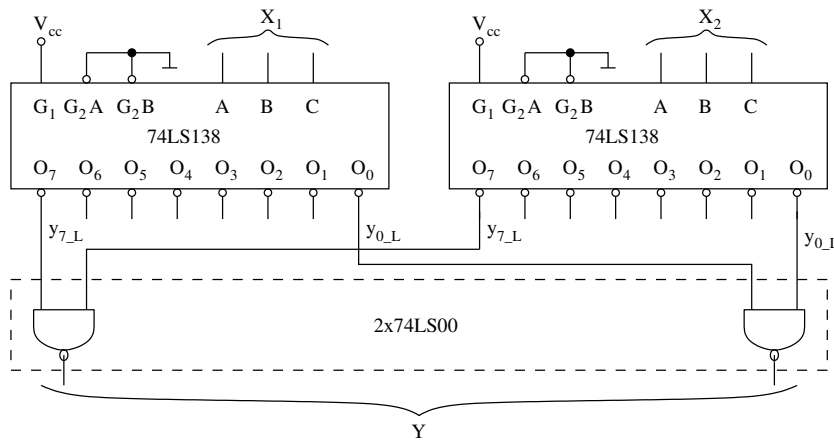
Din tabelul din Figura P2.31-a se observă că pentru funcția  $Y_2$ , ca și pentru funcția  $Y_1$  pe intervalul de definiție  $[000,111]$  se obțin tot opt valori dar cu un număr de zero-uri diferit pentru un același  $x$  și de asemenea când se trece de la  $x$  la  $x+1$  cuvântul  $Y_2$  crește cu mai mult de un zero. Pe baza acestei observații se poate implementa  $Y_2$  în felul următor: cuvântul de ieșire  $Y_2$  se obține cu același circuit ca și  $Y_1$ , dar se intercalează între biții cuvântului  $Y_1$  un număr de biți zero conform valorilor calculate în tabel. Acești biți zero, intercalați în cuvântul de ieșire, se obțin prin calblare la masă, Figura P2.31-c.

**P2.32***Rezolvare:*

Fiecare din operațiile  $2^{x_1}, 2^{x_2}$  poate fi implementată pe câte un circuit 74LS138. Impunând condiția ca  $x_1 \neq x_2$  niciodată ieșirile de același rang nu vor fi activate simultan și de asemenea suma  $2^{x_1} + 2^{x_2}$  nu va depăși  $2^8 - 1$ , adică un număr care este un cuvânt pe opt biți.

Condiția  $x_1 \neq x_2$  va determina ca fiecare bit din cuvântul de ieșire să fie o sumă doar a combinațiilor  $0 + 0 = 0, 0 + 1 = 1$  dar niciodată a combinațiilor  $1 + 1$  deci implementarea adunării se poate face cu o poartă OR și nu cu o poartă XOR. Utilizând circuitul 74LS138 cu ieșirile active în stare zero adunarea anterioară se implementează cu porți NAND cu două intrări  $P_i + P_j = \overline{P_i + P_j} = \overline{P_i} \cdot \overline{P_j} = \overline{y_{i,L}} \cdot \overline{y_{j,L}}$ , deci două circuite 74LS00 ( $4 \times \text{NAND}$  cu două intrări). Rezultă implementarea din Figura P2.32.

Pentru cazul când se elimină restricția  $x_1 \neq x_2$ , pentru sumarea  $2^{x_1} + 2^{x_2}$ , se vor utiliza două circuite sumatoarea 74LS83. Dacă circuitele DCD sunt cu ieșirile active în stare zero, aceste ieșiri trebuie negate înainte de sumare.

**Figura P 2.32****P2.34***Rezolvare:*

Circuitul DCD  $n : 2^n$  produce la ieșire toți cei  $2^n$  mintermi de  $n$  variabile. Din punctul de vedere al implementării unei funcții logice de  $n$  variabile dată sub forma FND (sumă de produse), circuitul DCD  $n : 2^n$  poate fi un suport. Existând generați toți cei  $2^n$  mintermi, deci nivelul ȘI de implementare, mintermi necesari funcției sunt selectați în nivelul SAU care se adaugă în exterior. Deoarece, în general, la ieșirea DCD sunt generați mintermi negați, expresia funcției FND trebuie exprimată în felul următor:

$$\begin{aligned} f(A, B, C) &= \overline{\overline{P_0 + P_1 + P_3 + P_5 + P_7}} = \overline{\overline{P_0} \cdot \overline{P_1} \cdot \overline{P_3} \cdot \overline{P_5} \cdot \overline{P_7}} = \\ &= \overline{y_{0,L} \cdot y_{1,L} \cdot y_{3,L} \cdot y_{5,L} \cdot y_{7,L}} \end{aligned}$$

$\overline{P}_i = y_{i,L}$  pentru:  $i = 0, 1 \dots 7$

Rezultă că în exterior trebuie să se adauge o poartă NAND care să colecteze semnalele

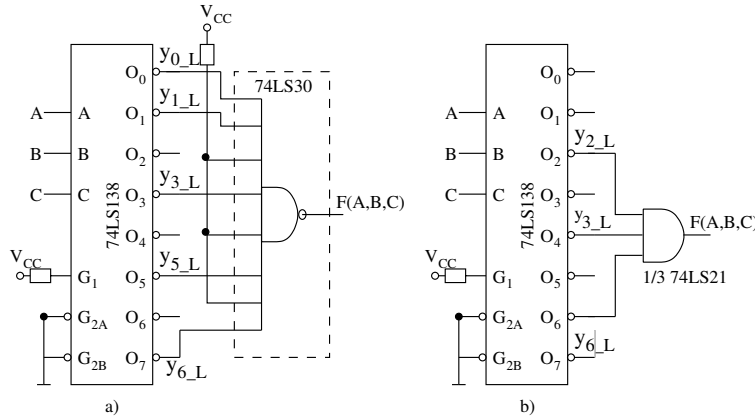


Figura P 2.34

$y_{0,L}, y_{1,L}, y_{3,L}, y_{5,L}$  și  $y_{7,L}$ . Se poate utiliza circuitul 74LS30 care conține o singură poartă NAND cu opt intrări ca în Figura P2.34-a.

Pentru cazul în care funcția are un număr de mintermi mai mare decât  $2^{n-1}$ , se poate face siteza funcției negate  $\overline{f}$ , deoarece acesta se poate exprima cu mai puțini mintermi.

$$\begin{aligned} \overline{f}(A, B, C) &= P_2 + P_4 + P_6 = \overline{\overline{P_2 + P_4 + P_6}} = \overline{\overline{P_2} \cdot \overline{P_4} \cdot \overline{P_6}} \rightarrow \\ &\rightarrow f(A, B, C) = \overline{\overline{P_2} \cdot \overline{P_4} \cdot \overline{P_6}} = y_{2,L} \cdot y_{4,L} \cdot y_{6,L} \end{aligned}$$

În exterior se adaugă o poartă AND cu trei intrări, ceea ce se poate implementa cu 1/3 din circuitul 74LS21 (trei porți AND pe chip) ca în Figura P2.34-b.

Implementarea cu DCD se poate realiza pornind de la forma FNC (produse de sume) a funcției. Aceeași funcție de mai sus poate fi exprimată și transformată în felul următor:

$$f(A, B, C) = \prod(2, 4, 6) = S_2 S_4 S_6 = \overline{P_2} \cdot \overline{P_4} \cdot \overline{P_6} = y_{2,L} \cdot y_{4,L} \cdot y_{6,L}$$

deoarece  $S_i = \overline{P}_i$  pentru  $i = 1, 2 \dots 7$  Rezultă o implementare cu o poartă AND cu trei intrări ca în Figura P2.34-b.

De asemenea, se poate face sinteza funcției negate  $\overline{f}$  exprimate sub forma FNC:

$$\begin{aligned} \overline{f}(A, B, C) &= \prod(0, 1, 3, 5, 7) = S_0 \cdot S_1 \cdot S_3 \cdot S_5 \cdot S_7 = \overline{P_0} \cdot \overline{P_1} \cdot \overline{P_3} \cdot \overline{P_5} \cdot \overline{P_7} \\ f(A, B, C) &= \overline{\overline{P_0} \cdot \overline{P_1} \cdot \overline{P_3} \cdot \overline{P_5} \cdot \overline{P_7}} = y_{0,L} \cdot y_{1,L} \cdot y_{3,L} \cdot y_{5,L} \cdot y_{7,L} \end{aligned}$$

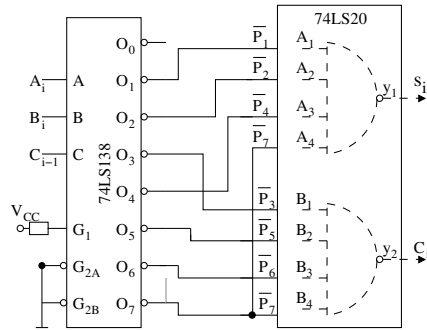
Rezultă implementarea din Figura P2.34-a.

Pentru implementarea unui CLC cu ieșiri multiple pe un singur DCD unii mintermi pot fi utilizați de mai multe porți SAU externe, caz în care se impune verificarea valorii maxime a fan-out-ului pentru semnalele respective.



**P2.36***Rezolvare:*

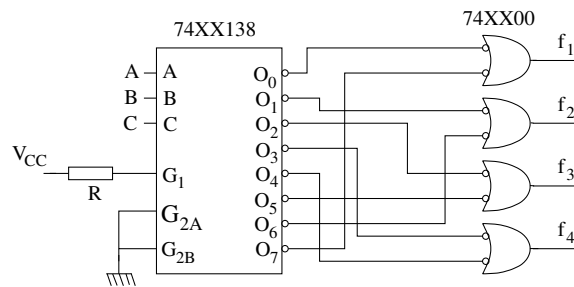
Sumatorul implementat este prezentat în Figura P2.36.

**Figura P 2.36****P2.37***Rezolvare:*

Pentru implementarea unei funcții de  $n$  variabile cu un DCD $n$  :  $2^n$  funcția nu necesită minimizare deoarece în nivelul ȘI sunt produși toți mintermii de  $n$  variabile; în exteriorul decodicatorului pe un nivel SAU trebuie colectați toți mintermii existenți în expresia funcției.

**P2.38***Rezolvare:*

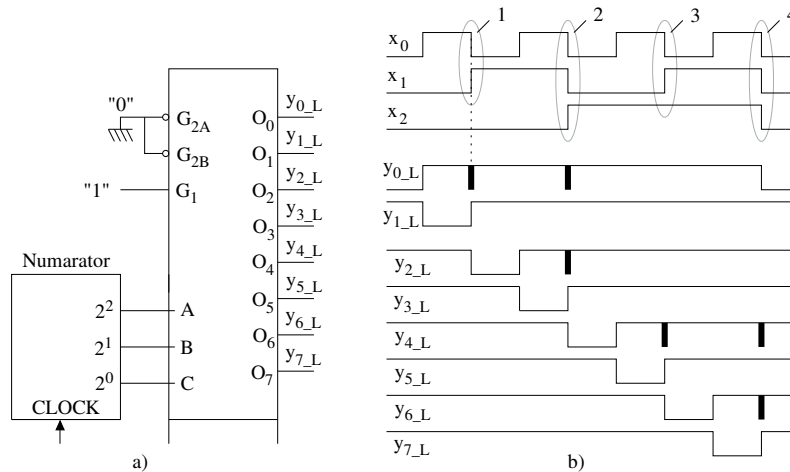
Toți mintermii acestor funcții sunt produși de un circuit 74XX138, DCD3:8 și deoarece fiecare funcție are numai doi mintermi sunt necesare patru porți NAND cu două intrări care se găsesc în circuitul integrat 74XX00. Deci sunt necesare numai două circuite integrate, Figura P2.38.

**Figura P 2.38**

**P2.39***Rezolvare:*

Glitch-urile (hazardul) pe ieșiri pot apare când comutația semnalelor aplicate pe intrările A,B,C, Figura P2.39-a, nu este sincronă. În acest caz semnalele de intrare nu comută sincron deoarece se obțin de la un numărător asincron; datorită întârzierilor din interiorul numărătorului asincron semnalul  $X_1(2^1)$  va avea o întârziere față de comutarea semnalului  $X_0(2^0)$ , de asemenea  $X_2(2^2)$  va avea o întârziere față de  $X_1$ . În punctele notate cu 1,2,3,4, când comută mai mult de un semnal de intrare, Figura P2.39-b, pot apare glitch-uri în semnalele de ieșire  $y_{i,L}$ ,  $i = 0 \dots 7$  ale circuitului decodificator

- În punctul 1 la comutarea de la 001 la 010 poate apare starea tranzitorie la intrare 000 (001  $\rightarrow$  000  $\rightarrow$  010) care va genera un glitch în semnalul  $y_{0,L}$
- În punctul 2 la comutarea de la 011 la 100 pot apare stările tranzitorii la intrare 010 și 000 (011  $\rightarrow$  010  $\rightarrow$  010  $\rightarrow$  100) care vor genera glitch-uri în  $y_{0,L}$  și  $y_{2,L}$
- În punctul 3 la comutarea de la 101 la 110 poate apare starea tranzitorie la intrare 100 (101  $\rightarrow$  100  $\rightarrow$  110) care va genera un glitch în semnalul  $y_{4,L}$
- În punctul 4 la comutarea de la 111 la 000 pot apare stările tranzitorii 110 și 100 (111  $\rightarrow$  110  $\rightarrow$  100  $\rightarrow$  000) care vor genera glitch-uri în semnalul  $y_{6,L}$  și  $y_{4,L}$

**Figura P 2.39**

O modalitate de a elimina glitch-urile în circuitele la care se aplică ieșirile de la decodificator,  $y_{i,L}$ , este strobarea, adică fiecare semnal din acestea este intrarea unei porți AND, iar această poartă este validată, pe o altă intrare, cu un semnal care se aplică numai după trecerea perioadei de timp tranzitoriu. (S-a considerat că în circuitul 74XX138 toate căile de propagare, de la intrările de selectare A,B,C la ieșirile  $O_i$ , au același timp de propagare).

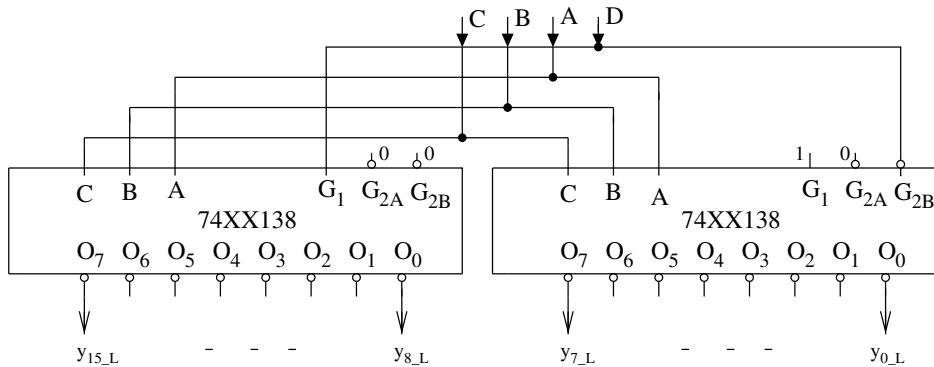
**P2.40***Rezolvare:*

Analizând compunerea semnalului  $s_i = P_1 + P_2 + P_4 + P_7$  se constată că va apărea un impuls parazit "1" în valoarea sa determinat de glitch-ul  $y_4$  din momentul (3) (vezi Figura P2.39-b). Celelalte două momente (2) și (4) reușind doar să extindă, eventual, durata valorii "1" la început respectiv la sfârșit. Pentru semnalul  $C_i = P_3 + P_5 + P_6 + P_7$  glitch-ul din  $y_6$  poate duce la prelungirea duratei valorii de 1 logic, dar nu la o valoare eronată.

Eliminarea glitch-urilor generate pe ieșirile decodificatorului se poate realiza fie prin utilizarea unui numărator sincron dar mai sigur prin activarea (la decodificator) a semnalului compus de validare  $G_1 \cdot \overline{G_{2A}} \cdot \overline{G_{2B}}$  cu o întârziere față de frontul de comutare al clock-ului la numărator astfel încât procesul tranzitoriu să se fi "consumat".

**P2.41***Rezolvare:*

O astfel de organizare, Figura P2.41, este posibilă deoarece circuitul 74XX138 conține atât semnale de validare active în low,  $G_{2B,L}$  și  $G_{2A,L}$  cât și un semnal de validare activ în starea high,  $G_1$ . Aplicând bitul cel mai semnificativ, D, al cuvântului de intrare, D ABC, pe intrarea de validare  $G_{2B}$  al unui circuit și pe intrarea  $G_1$  al celuilalt, atunci ieșirile primului circuit sunt repartizate intervalului 7-0 iar ieșirile celui de al doilea intervalului 15-8.

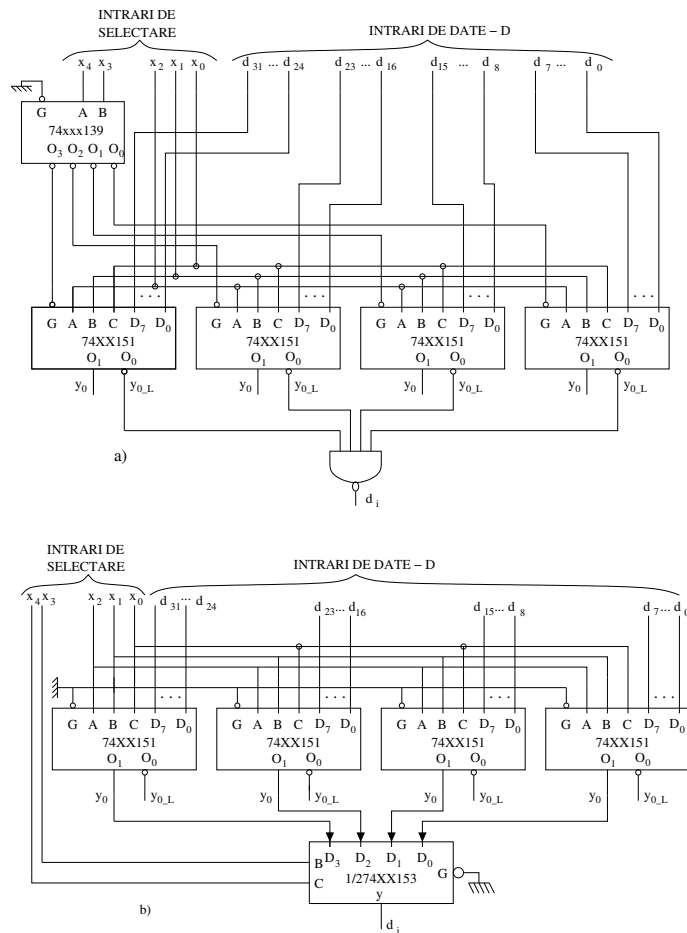
**Figura P 2.41****P2.42***Rezolvare:*

$$f_1 = \sum_0^7 (0, 2, 3, 4); f_2 = \sum_0^7 (0, 1, 5, 7); f_3 = \sum_0^7 (3, 4, 6, 7); f_4 = \sum_0^7 (0, 3, 5, 7)$$

**P2.43***Rezolvare:*

Cuvântul de date  $D$  de pe intrare  $d_{31}-d_0$  este împărțit în patru intervale  $d_{31}-d_{24}$ ,  $d_{23}-d_{16}$ ,  $d_{15}-d_8$ ,  $d_7-d_0$  și fiecare interval de biți de date se aplică unui circuit 74XX151. La toate aceste patru circuite MUX8:1 se aplică pentru selectare cei trei biți mai puțin semnificativi  $x_2, x_1, x_0$  ai cuvântului de selectare. Cei doi biți mai semnificativi  $x_4$  și  $x_3$  se utilizează pentru generarea semnalelor de selectare a datelor din unul din cele patru intervale. În varianta din Figura P2.43-a ieșirile circuitului decodificator 74XX139, aplicate pe intrările de validare al multiplexoarelor, selectează câte un multiplexor pentru fiecare interval de biți de date. Ieșirile  $Y_{0..L}$  ale celor patru multiplexoare sunt colectate într-o poartă NAND, dacă se utilizează ieșirile nenegate este necesară o poartă OR.

În varianta din Figura P2.42 ieșirile celor patru multiplexoare 8:1 sunt selectate la ieșire cu un multiplexor 4:1.

**Figura P 2.43**

### P2.44

*Rezolvare:*

Selectarea unui port la magistrala  $b_3b_2b_1b_0$  se poate realiza cu  $4 \times MUX4 : 1$ . Sunt utilizate circuitele 74XX253, MUX4:1 cu ieșirea TSL, se comandă în stare normală când  $G_L = 0$ , Figura P2.44-a. Porturile  $P_3, P_2, P_1$  și  $P_0$  se selectează respectiv cu următoarele valori ale variabilelor de selectare  $x_1x_0 (= 11, 10, 01, 00)$ . Aceeași funcție de selectare se poate realiza și cu patru buffere TSL neinversoare de patru biți 74XX243, Figura P2.44-b. Comanda (și selectarea) unui buffer în stare normală de funcționare se realizează cu semnalele de ieșire de la un DCD2:4, 74XX139.

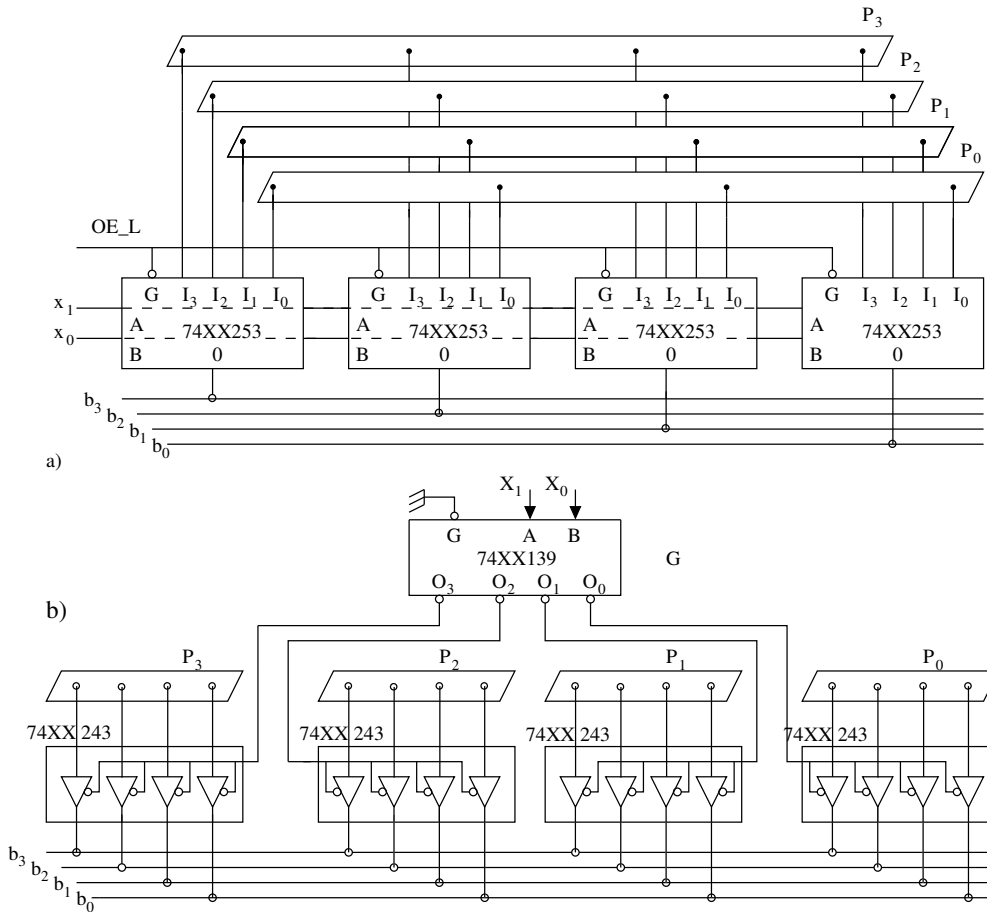


Figura P 2.44

## P2.46

*Rezolvare:*

Implementarea funcției de  $n$  variabile pe un MUX  $2^n : 1$  se realizează prin maparea directă (unu la unu) a tabelului de adevăr sau a diagramei V-K pe intrările de date ale multiplexorului.

Când se realizează o structură de MUX  $2^n : 1$ , din multiplexoare cu un număr de căi mai mic decât  $2^n$ , numărul fiecărei intrări rezultante (numerele înscrise în paranteze în figura P2.46) se obține ca un cuvânt în binar compus din alăturarea biților ce se întâlnesc la parcurgerea tuturor intrărilor de la MUX-urile componente de pe calea rezultantă respectivă. Foarte adesea implementările cu MUX 2:1 pe primul nivel, chiar și pe următoarele niveluri, sunt avantajoase deoarece pentru calculul funcțiilor banale ( $x_0, \bar{x}_0, 0, 1$ ) multiplexoarele 2:1 pot fi eliminate.

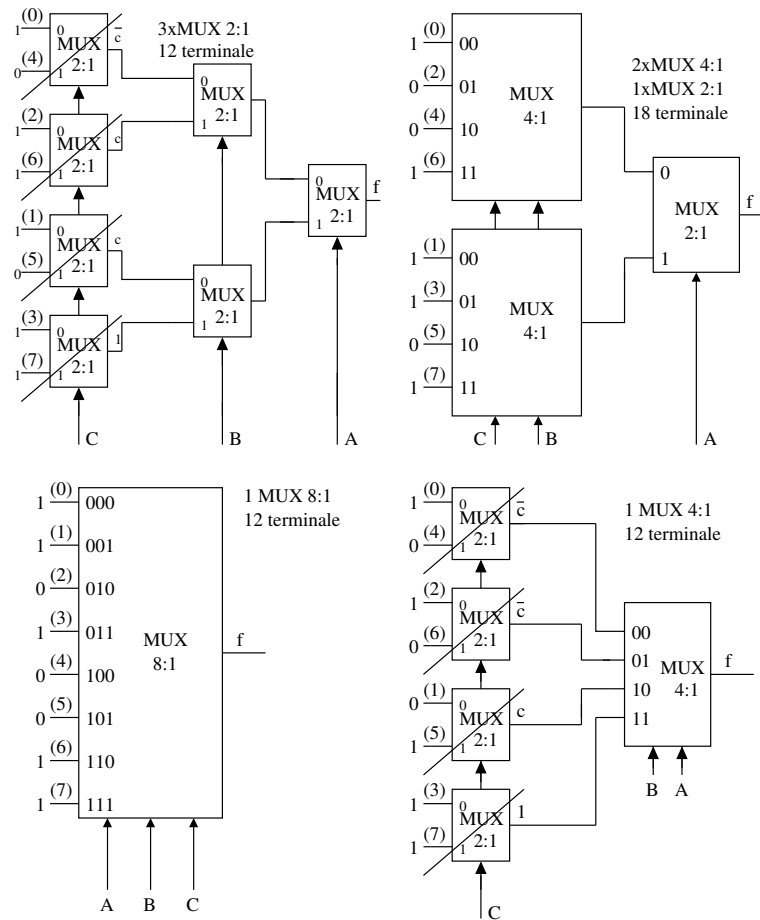
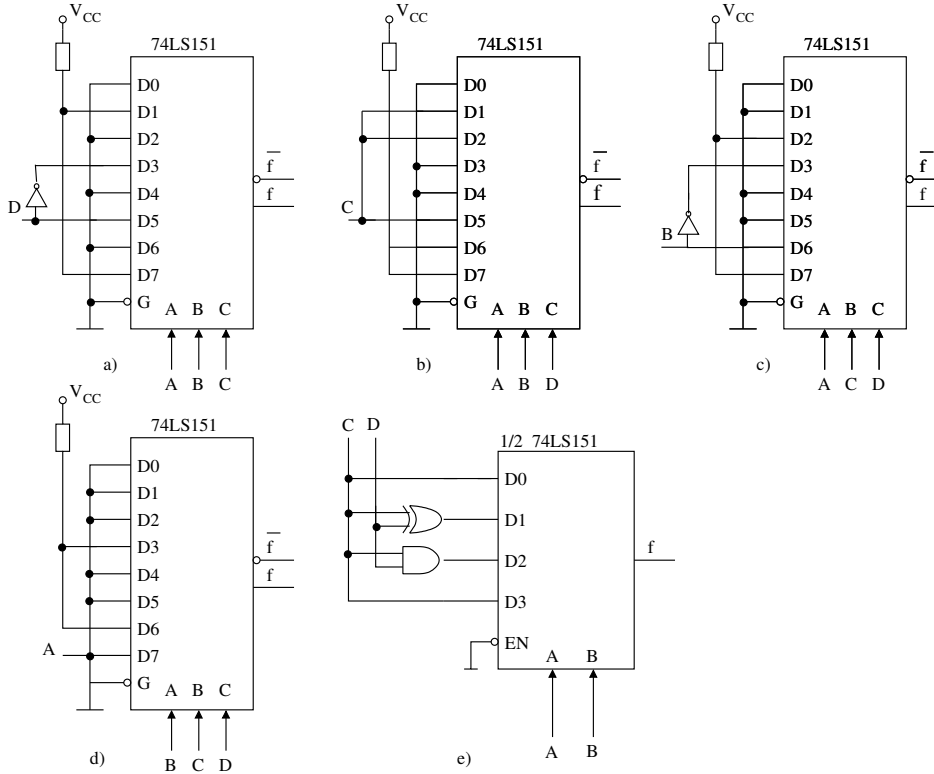


Figura P 2.46

**P2.48** Fie funcția  $f(A, B, C, D) = \sum(3, 6, 11, 14, 15) + \sum d(2, 5, 12, 13)$ . Să se implementeze cu circuitul 74LS151, MUX 8:1.

*Rezolvare:*

Pe circuitul 74LS151 MUX 8:1, ca circuit universal de trei variabile, nu se poate implementa funcția de patru variabile prin maparea directă pe intrări a tabelului de adevăr. În schimb, dacă una din variabile este introdusă în coeficienții funcției, ca variabilă reziduu, rezultă o funcție de trei variabile cu exprimările următoare:



**Figura P 2.48**

$$\begin{aligned}
 f(D, A, B, C) &= 0 \cdot \bar{A}\bar{B}\bar{C} + (D/0) \cdot \bar{A}\bar{B}C + (D/0) \cdot \bar{A}B\bar{C} + D \cdot \bar{A}BC + 0 \cdot \bar{A}B\bar{C} + \\
 &+ D \cdot \bar{A}BC + (0/1) \cdot \bar{A}B\bar{C} + 1 \cdot \bar{A}BC \\
 f(C, A, B, D) &= (0/C) \cdot \bar{A}\bar{B}\bar{D} + C \cdot \bar{A}\bar{B}D + C \cdot \bar{A}B\bar{D} + (0/C) \cdot \bar{A}BD + 0 \cdot \bar{A}B\bar{D} + \\
 &+ C \cdot \bar{A}BD + (C/1) \cdot \bar{A}B\bar{D} + (C/1) \cdot \bar{A}BD \\
 f(B, A, C, D) &= 0 \cdot \bar{A}\bar{C}\bar{D} + (0/B) \cdot \bar{A}\bar{C}D + (B/1) \cdot \bar{A}C\bar{D} + \bar{B} \cdot \bar{A}CD + \\
 &+ (0/B) \cdot \bar{A}C\bar{D} + (0/B) \cdot \bar{A}CD + B \cdot \bar{A}C\bar{D} + 1 \cdot \bar{A}CD \\
 f(A, B, C, D) &= 0 \cdot \bar{B}\bar{C}\bar{D} + 0 \cdot \bar{B}\bar{C}D + (0/\bar{A}) \cdot \bar{B}\bar{C}\bar{D} + 1 \cdot \bar{B}\bar{C}D + (0/A) \cdot \bar{B}\bar{C}\bar{D} + \\
 &+ (0/1) \cdot \bar{B}\bar{C}D + 1 \cdot \bar{B}\bar{C}D + A \cdot \bar{B}CD
 \end{aligned}$$

Pentru aceste exprimări, se obțin implementările din Figura P2.48-a,b,c,d. Aceste implementări sunt recomandate deoarece cei opt coeficienți reziduu care se aplică pe intrările circuitului 74LS151 sunt valori banale care nu trebuie să fie calculate. Introducând două variabile reziduu se obține exprimarea:

$$f(C, D, A, B) = (C/CD) \cdot \overline{A} \overline{B} + (C \oplus C/CD) \cdot \overline{A} B + (CD) \cdot A \overline{B} + (1/C) \cdot AB$$

Implementarea este prezentată în figura P2.48-e, unde s-a utilizat un MUX 4:1. La această implementare sunt necesare anumite calcule, realizate cu porți simple, pentru coeficienții aplicați pe intrările multiplexorului.

### P2.50

*Rezolvare:*

Funcția fiind de cinci variabile, pentru implementare este necesară introducerea în expresia coeficienților a două variabile reziduu. Deoarece în expresia funcției se poate factoriza

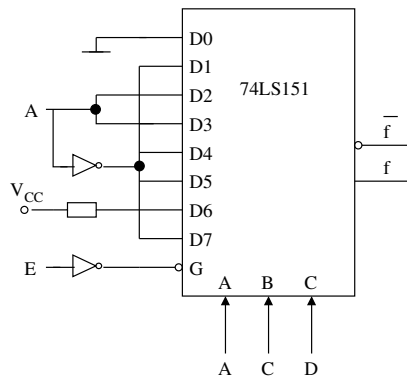


Figura P 2.50

variabila E, aceasta poate fi utilizată pentru comanda validării circuitului, aplicată pe borna G, deci este necesară numai o singură variabilă reziduu. Expresia dată se transformă în felul următor:

$$\begin{aligned} f(A, B, C, D, E) &= E\{\overline{A}[B + \overline{C}D] + A[\overline{B}C + C\overline{D}]\} = \\ &= E\{\overline{A}[B(C + \overline{C})(D + \overline{D}) + \overline{C}D(B + \overline{B})] + A[\overline{B}C(D + \overline{D}) + C\overline{D}(B + \overline{B})]\} = \\ &= E\{\overline{A}[P_1 + P_4 + P_5 + P_7] + A[P_2 + P_3] + P_6 + 0 \cdot P_0\} \end{aligned}$$

Termenii produs sunt calculați numai pentru cele trei variabile B,C,D. Implementarea acestei expresii este reprezentată în Figura P2.50.



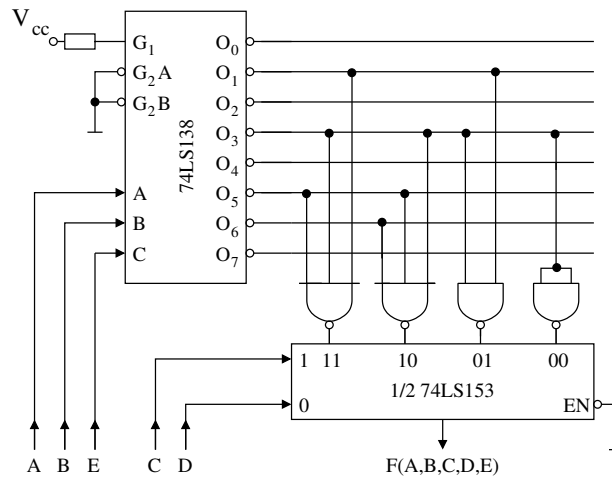
**P2.52**

*Rezolvare:*

Pentru funcțiile de mai multe variabile, la care numărul variabilelor introduse ca variabile reziduu este relativ mare, se poate utiliza pentru calculul coeficienților o structură de DCD căreia i se adaugă nivelul de SAU în exterior. O funcție de  $j$  variabile se implementează cu un MUX cu  $n$  intrări de selectare și mai multe funcții de  $(j-n)$  variabile, care se calculează cu un  $DCD(j-n) : 2^{(j-n)}$ , plus nivelul de SAU exterior, și care, apoi, se aplică pe intrările multiplexorului. În acest sens, pentru implementarea cu circuitul 74LS153, MUX4:1, și cu circuitul 74LS138, DCD3:8, este necesar ca funcția să fie exprimată în funcție de două variabile C și D. Celelalte trei variabile, A,B și E sunt introduse în expresia coeficienților ca variabile reziduu. În plus, expresiile coeficienților trebuie să fie funcții de mintermii  $P_i$ , de cele trei variabile A,B și E, pentru a putea fi calculate de circuitul 74LS138, plus nivelul de SAU exterior. Pentru această implementare expresia inițială a funcției se transformă în felul următor:

$$\begin{aligned}
 f(A, B, C, D, E) &= \bar{A}BE + \bar{A}\bar{B}DE + A\bar{B}CE + AC\bar{D}E = \\
 &= \bar{A}BE(C + \bar{C})(D + \bar{D}) + \bar{A}\bar{B}DE(C + \bar{C}) + A\bar{B}CE(D + \bar{D}) + AC\bar{D}E(B + \bar{B}) = \\
 &= \bar{A}BECD + \bar{A}BE\bar{C}D + \bar{A}BECD + \bar{A}BE\bar{C}\bar{D} + \\
 &+ \bar{A}\bar{B}ECD + \bar{A}\bar{B}E\bar{C}D + A\bar{B}ECD + ABEC\bar{D} + A\bar{B}E\bar{C}\bar{D} = \\
 &= (\bar{A}BE + \bar{A}\bar{B}E + A\bar{B}E) \cdot CD + (\bar{A}BE + \bar{A}\bar{B}E + ABE) \cdot \bar{C}\bar{D} + (\bar{A}BE + \bar{A}\bar{B}E) \cdot \bar{C}D + \\
 &+ (\bar{A}BE) \cdot \bar{C}\bar{D} = \\
 &= (P_3 + P_1 + P_5) \cdot CD + (P_3 + P_5 + P_7) \cdot \bar{C}\bar{D} + (P_3 + P_1) \cdot \bar{C}D + P_3 \cdot \bar{C}\bar{D}
 \end{aligned}$$

Această expresie este implementată în Figura P2.52.



**Figura P 2.52**

## P2.54

*Rezolvare:*

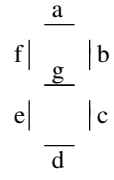
Uzual este circuitul convertor care pentru cele 10 cifre zecimale (0, 1, ..., 8, 9) exprimate în BCD (0000, 0001, ..., 1000, 1001) generează codul binar de 7 segmente pentru comanda afișoarelor compuse din 7 segmente, de data aceasta se cere un circuit convertor implementat în ROM care realizează conversia inversă.

Tabelul de adevăr pentru conversia cuvântului de cod 7 segmente în BCD este dat în Figura P2.54-a care poate fi implementat pe un circuit ROM cu capacitatea de  $2^7 \times 4 = 512$  biți, cuvântul de adresă este  $abcdefg$  iar cel de ieșire  $WXYZ$ . Deoarece la intrare, din cele 128 cuvinte posibile, pot apare doar 10 cuvinte de cod 7 segmente, circuitul ROM este utilizat doar  $(10 : 128)100 = 7,8\%$ .

O prima reducere a capacității circuitului ROM se poate obține în felul următor. Cu ultimii cinci biți  $cdefg$  se pot codifica  $2^5 = 32$  configurații diferite, iar în tabelul din Figura P2.54-a există doar 7 configurații diferite cu acești cinci biți care sunt prezentate în tabelul din Figura P2.54-b. Aceste 7 configurații diferite de cinci biți sunt transcodate într-un cuvânt de cod intermediar de trei biți  $A, B, C$ . Cu această transcodare intermediară plus configurațiile formate cu cei doi biți  $a, b$  din tabelul din Figura P2.54-a se formează un tabel de adevăr, Figura P2.54-c, care are 10 configurații pe intrare dar care nu mai sunt exprimate prin 7 biți ci printr-un cuvânt de 5 biți  $ABCab$ . Implementarea corespunzătoare, Figura P2.54-d, a acestui tabel de adevăr se realizează cu un circuit ROM2 de capacitate  $(32 \times 3)$  biți=96 biți pentru transcodare și un circuit ROM1 de capacitate  $(32 \times 4)$  biți=128 biți, capacitatea memoriei a scăzut față de implementarea anterioară de la 512 la 224 biți. Se mai poate reduce capacitatea ROM dacă se elimină din cele 7 variabile de intrare variabilele care sunt redundante. Sunt redundante acele variabile de intrare care sunt indiferente în definirea ieșirilor  $W, X, Y, Z$  (ieșirile pot fi exprimate și fără acestea), deci pot fi eliminate. Analizând tabelul de adevăr din Figura P2.54-a se constată că se poate elimina fie perechea de variabile  $bc$  fie perechea  $cd$ , cu cele cinci variabile rămase,  $2^5 = 32$ , se pot acoperi cele 10 configurații de intrare distincte ( $32 > 10$ ). Eliminând variabilele  $b$  și  $c$ , se observă că pentru oricare din configurațiile de variabile  $adefgh$  din tabel nu corespund două configurații diferite ale cuvântului  $WXYZ$ . De exemplu, dacă configurația de intrare  $adefg = 11101$ , căruia îi corespunde cuvântul de ieșire 0010 (linia a treia din tabelul din Figura P2.54-e), presupunem, că ar mai fi prezentă și pe linia a cincea din tabel căruia îi corespunde cuvântul de ieșire 0100, atunci în cuvântul de intrare 11101 mai trebuie o variabilă prin care să se facă distincția pentru cele două cuvinte de ieșire 0010 și 0100, deci variabila eliminată nu este redundantă și trebuie păstrată. Tabelul de adevăr în care s-au eliminat variabilele  $b$  și  $c$ , care sunt redundante, este implementat, Figura P2.54-f, pe un circuit ROM de capacitate  $(32 \times 4)$  biți=128 biți.

Se poate merge mai departe cu această reducere prin includerea variabilelor  $d$  și  $g$  într-o funcție logică AND,  $H = g \cdot d$ , astfel numărul variabilelor care se aplică la intrarea unui circuit ROM se reduce la patru  $a, e, f$  și  $H$ , Figura P2.54-g. Implementarea corespunzătoare este prezentată în Figura P2.54-h și necesită un circuit ROM de capacitate  $(16 \times 4)$  biți=64 biți și o poartă AND.

CIFRA ZECIMALA	INTRARI							IESIRI			
	Codul binar fragmente							W	X	Y	Z
	a	b	c	d	e	f	g	(2 <sup>3</sup> )	(2 <sup>2</sup> )	(2 <sup>1</sup> )	(2 <sup>0</sup> )
0	1	1	1	1	1	1	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	0	1
2	1	1	0	1	1	0	1	0	0	1	0
3	1	1	1	1	0	0	1	0	0	1	1
4	0	1	1	0	0	1	1	0	1	0	0
5	1	0	1	1	0	1	1	0	1	0	1
6	0	0	1	1	1	1	1	0	1	1	0
7	1	1	1	0	0	0	0	0	1	1	1
8	1	1	1	1	1	1	1	1	0	0	0
9	1	1	1	0	0	1	1	1	0	0	1



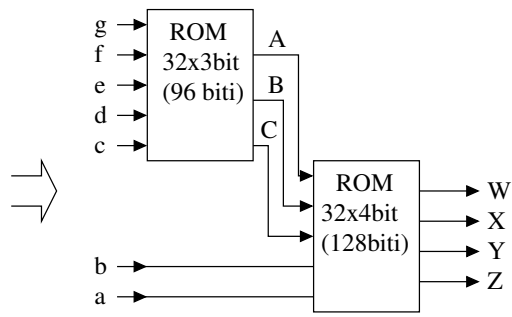
a)

SUBCUVANT de intrare					Corespunde cifrei zecimale	Codul asigurat		
A	B	C	a	b		A	B	C
1	0	0	0	0	2	0	0	0
1	0	0	1	1	7,1	0	0	1
1	0	0	0	1	9,4	0	1	0
1	1	0	1	1	3	0	1	1
1	1	0	1	0	5	1	0	0
1	1	1	1	0	0	1	0	1
1	1	1	1	1	8,6	1	1	0

b)

INTRARI					IESIRI			
A	B	C	a	b	W	X	Y	Z
1	0	1	1	1	0	0	0	0
0	0	1	0	1	0	0	0	1
0	0	0	1	1	0	0	1	0
0	1	1	1	1	0	0	1	1
0	1	0	0	1	0	1	0	0
1	0	0	1	0	0	1	0	1
1	1	0	0	0	0	1	1	0
0	0	1	1	1	0	1	1	1
1	1	0	1	1	1	0	0	0
1	1	0	0	0	1	0	0	1

c)

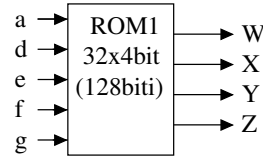


d)

Se continua pe pagina urmatoare

CIFRA ZECI- MALA	INTRARI					IESIRI			
	a	d	e	f	g	W	X	Y	Z
0	1	1	1	1	0	0	0	0	0
1	0	0	0	0	0	0	0	0	1
2	1	1	1	0	1	0	0	1	0
3	1	1	0	0	1	0	0	1	1
4	0	0	0	1	1	0	1	0	0
5	1	1	0	1	1	0	1	0	1
6	0	1	1	1	1	0	1	1	0
7	1	0	0	0	0	0	1	1	1
8	1	1	1	1	1	1	0	0	0
9	1	0	0	1	1	1	0	0	1

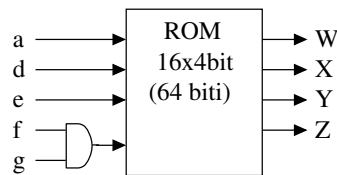
e)



f)

CIFRA ZECI- MALA	INTRARI				IESIRI			
	a	e	f	H(dg)	W	X	Y	Z
0	1	1	1	0	0	0	0	0
1	0	0	0	0	0	0	0	1
2	1	1	0	1	0	0	1	0
3	1	0	0	1	0	0	1	1
4	0	0	1	0	0	1	0	0
5	1	0	1	1	0	1	0	1
6	0	1	1	1	0	1	1	0
7	1	0	0	0	0	1	1	1
8	1	1	1	1	1	0	0	0
9	1	0	1	0	1	0	0	1

g)



h)

Figura P 2.54

## P2.55

*Rezolvare:*

Este necesar un circuit EPROM de capacitate  $2^8 \times 4$  biți=1kbiți. Dacă circuitul ROM în stare neprogramată este “plin“ cu zero-uri în toate locațiile după programare sunt modificate numai locațiile de la adresele indicate în tabelul din Figura P2.55. În cuvântul de ieșire  $O_3O_2O_1O_0$ , citit din ROM, bitul  $O_3 = 1$  indică prezența unui singur bit egal cu zero în cuvântul de intrare/adresă iar biții  $O_2, O_1, O_0$  formează cuvântul care specifică, în binar, poziția în cuvântul de intrare a singurului bit egal cu zero.

INTRARE									IESIRE				
in binar								hexa	zecimal				
x <sub>7</sub>	x <sub>6</sub>	x <sub>5</sub>	x <sub>4</sub>	x <sub>3</sub>	x <sub>2</sub>	x <sub>1</sub>	x <sub>0</sub>			O <sub>3</sub>	O <sub>2</sub>	O <sub>1</sub>	O <sub>0</sub>
0	1	1	1	1	1	1	1	7FH	127	1	0	0	0
1	0	1	1	1	1	1	1	BFH	191	1	0	0	1
1	1	0	1	1	1	1	1	DFH	223	1	0	1	0
1	1	1	0	1	1	1	1	EFH	239	1	0	1	1
1	1	1	1	0	1	1	1	F7H	247	1	1	0	0
1	1	1	1	1	0	1	1	FBH	251	1	1	0	1
1	1	1	1	1	1	0	1	FDH	253	1	1	1	0
1	1	1	1	1	1	1	0	FEH	254	1	1	1	1

Figura P 2.55

**P2.56***Rezolvare:*

Tabelul de adevăr pentru calculul pătratului numerelor binare de trei biți este redat în Figura P2.56-a în care se observă următoarele particularități  $y_1 = 1$  și  $y_0 = A_0$ . Pentru implementare rezultă structura de circuit din Figura P2.56-b, pe baza unui circuit ROM de capacitate  $8 \times 4$  biți=32 biți.

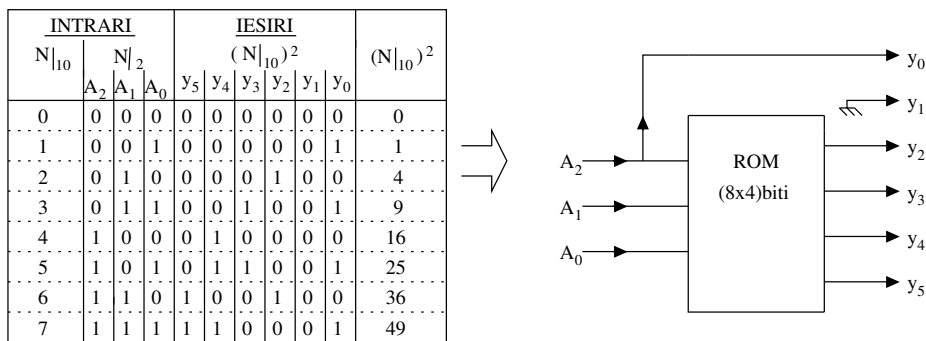


Figura P 2.56

**P2.57***Rezolvare:*

Transformând adresele spațiului de adresare din hexazecimal în binar, pentru o decodificare completă se obțin pentru biții  $A_9 \dots A_0$  valorile prezentate în Figura P2.57-a. Se observă că, pentru toate cele patru adrese biții  $A_9 \dots A_2$  rămân nemodificați în timp ce  $A_1 A_0$  se modifică în funcție de perifericul adresat. Aceasta sugerează soluția utilizării circuitului 74LS139, DCD2:4, la care se aplică biții  $A_1 A_2$  și ale cărui ieșiri selectează perifericele #0,#1#2#3. Coincidența valorilor biților de adresă  $A_9 A_2$ , cu valorile 11110000,

trebuie detectată și aplicată ca un semnal de validare (G) la circuitul 74LS139. Cu ajutorul circuitului 74LS138, DCD3:8, se poate obține activarea ieșirii  $O_0$  când există coincidența biților  $A_9 \dots A_2$  cu următorul cuvânt 11110000, Figura P2.57-b.

O altă variantă de implementare pentru această coincidență este reprezentată în figura P2.57-c utilizând circuitul 74LS682 (circuit comparator de opt biți cu două ieșiri  $P=Q$  și  $P>Q$ , active în L).

Se poate realiza și o decodificare incompletă când nu sunt utilizați toți biții. De exemplu, dacă în Figura P2.57-b se dispune de o poartă NAND numai cu două intrări, bitul  $A_9$  rămâne flotant pentru circuitul de identificare a coincidenței, deci poate lua fie valoarea 0, fie valoarea 1. Rezultă că fiecare periferic va fi decodificat nu pentru o singură adresă transmisă de microprocesor pe magistrala de adresare ci pentru două adrese în felul următor:

#0 → 1C0H și 3C0H;  
 #1 → 1C1H și 3C1H;  
 #2 → 1C2H și 3C2H;  
 #3 → 1C3H și 3C3H;

Pentru implementarea cu comparator, Figura P2.57-c, o astfel de modalitate de decodi-

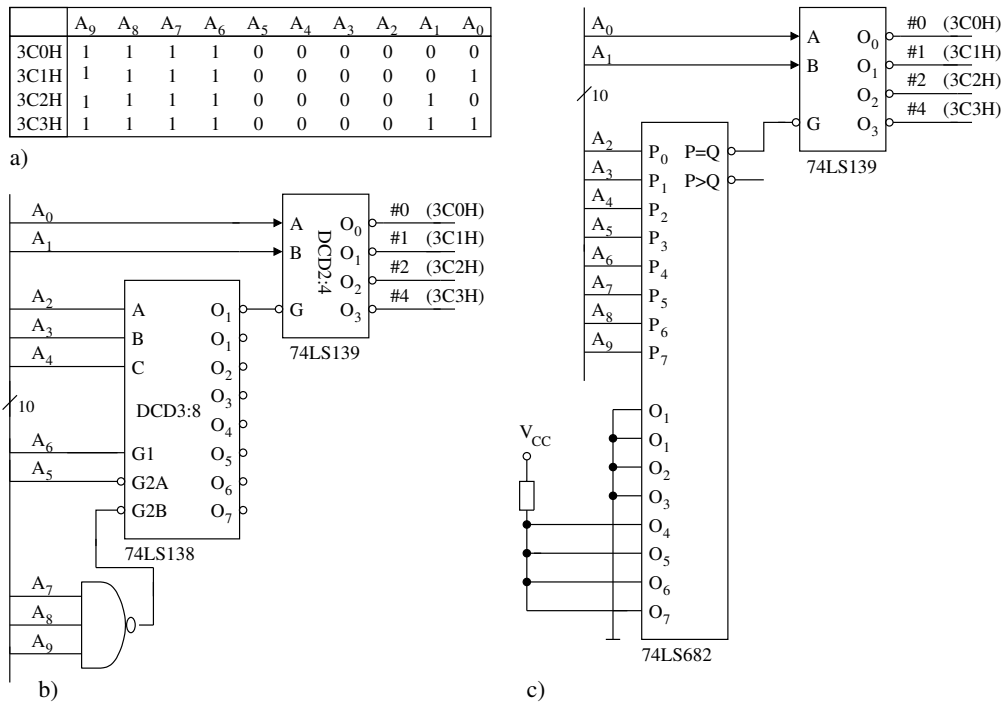


Figura P 2.57

ficare incompletă se obține prin conectarea intrării  $Q_7$  nu la  $V_{CC}$  ci la linia  $A_9$ .

## P2.59

Rezolvare:

Pentru a forma modulul de  $8K \times 16$  biți este necesară o extensie la 16 biți pe ieșire ceea ce înseamnă că sunt necesare câte două circuite 2716 conectate în paralel pentru a obține

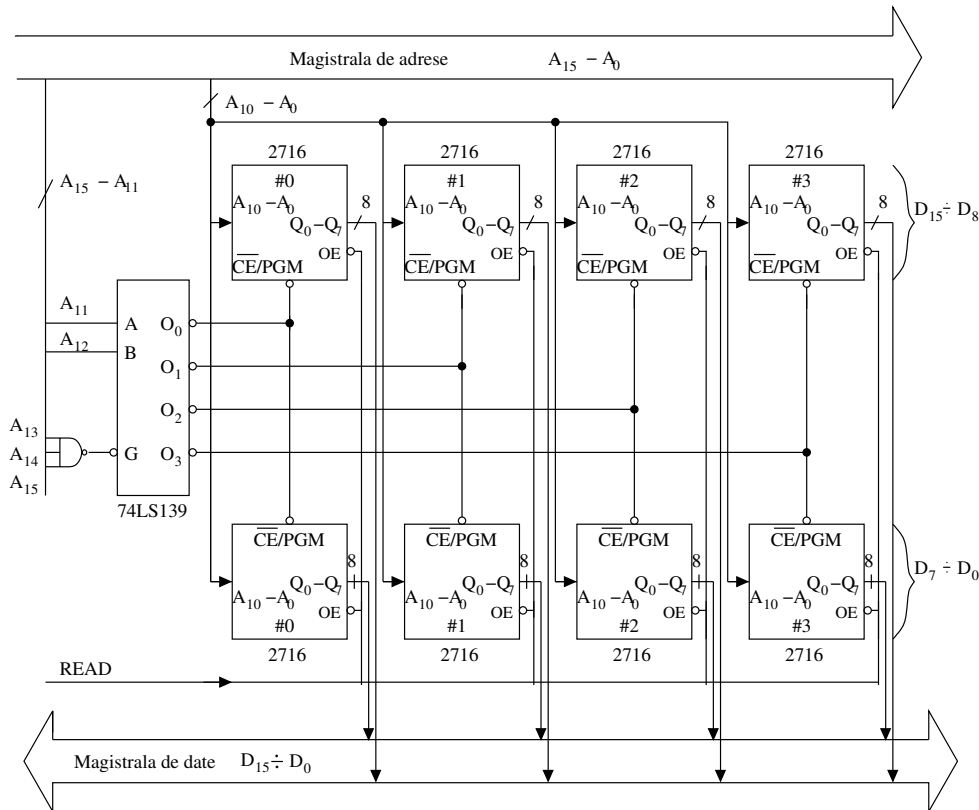


Figura P 2.59

cuvântul de date  $D_{15} \dots D_8, D_7 \dots D_0$  și o extensie pe intrare, Figura P2.59. Extensia pe intrare constă în plasarea pe fiecare subinterval de adresă de  $2K$  (din cei  $8K$  acoperiți de modul) a câte unui grup de două circuite 2716 ( $D_{15} \dots D_8$  și  $D_7 \dots D_0$ ). Deoarece decodificatorul intern al fiecărui circuit decodifică numai 11 ( $A_{10} \dots A_0$ ) biți este necesară extensia decodicatorului în exterior pentru biții de adresă  $A_{12}, A_{11}$ . Extensia externă se realizează cu circuitul 74LS139, DCD 2:4. Pentru a “plasa” acest modul în intervalul de  $8K$  adrese cele mai semnificative din spațiul de  $64K$  este necesar ca cei trei biți cei mai semnificativi din cuvântul de adresă să fie  $A_{15}A_{14}A_{13} = 111$ . Prin conjuncția acestor trei biți se generează semnalul de validare  $G$  al decodicatorului 2:4. Astfel, intervalul de adresare asignat modului este  $E000K \dots FFFFH$  compus din 4 subintervale adiacente care se deduc din mapa memoriei din următorul tabel.

$A_{15}$	$A_{14}$	$A_{13}$	$A_{12}$	$A_{11}$	$A_{10}A_9$	$A_8$	$A_7A_6$	$A_5A_4$	$A_3A_2A_1A_0$		
1	1	1	1	1	1	1	1	1	1	#3 {	
-	-	-	-	-	-	-	-	-	-		FFFF H
0	0	0	0	0	0	0	0	0	0		F800 H
1	1	1	1	0	1	1	1	1	1	#2 {	
-	-	-	-	-	-	-	-	-	-		F7FF H
0	0	0	0	0	0	0	0	0	0		F000 H
1	1	1	0	1	1	1	1	1	1	#1 {	
-	-	-	-	-	-	-	-	-	-		EFFF H
0	0	0	0	0	0	0	0	0	0		E800 H
1	1	1	0	0	1	1	1	1	1	#0 {	
-	-	-	-	-	-	-	-	-	-		E700 H
0	0	0	0	0	0	0	0	0	0		E000 H

**P2.60***Rezolvare:*

Implementarea funcției reprezentate sub forma FCD pe o memorie ROM, se realizează prin înscrierea valorii 1 la adresele pentru care mintermii au valoarea 1 (codul mintermului este considerat ca un cuânt de adresă format din variabilele funcției). Această simplitate apare datorită faptului că ROM-ul produce, pe nivelul de decodificare AND, toți mintermii variabilelor de intrare (adrese). Deci, nu trebuie efectuată nici o minimizare. Rezultă că pe nivelul programabil (OR) se introduce direct tabelul de adevăr al funcției. Implementarea apare mult mai intuitivă dacă funcția este dată într-o diagramă V-K ce poate fi mapată direct pe o structură matriceală de memorie ca în Figurile P2.60-a și b.

Reprezentarea în diagrama V-K poate fi privită ca o sumă de funcții de un număr mai redus de variabile. De exemplu, pentru o funcție de 6 variabile se poate face o descompunere în: două funcții de 5 variabile sau 4 funcții de 4 variabile sau o funcție de 5 variabile și 2 funcții de 4 variabile, etc. În Figura P2.60-c este reprezentată funcția  $f$  ca o sumă logică între o funcție de 5 variabile, una de 4 variabile și două de 3 variabile.

Această descompunere în funcții componente nu este de nici o utilitate pentru o singură funcție, dar este foarte utilă când se dorește implementarea mai multor funcții diferite, de aceleași variabile, pe o aceeași structură ROM. Suprafața ocupată de fiecare funcție pe diagrama V-K este disjunctă față de celelalte suprafețe. Funcția de 5 variabile  $f_1(E, D, C, B, A)$  este înscrisă în intervalul de adrese care se selectează cu valoarea variabilei  $F = 1$ , funcția de 4 variabile  $f_2(D, C, B, A)$  se selectează cu valorile variabilelor  $F = 0$  și  $E = 1$ ; iar funcțiile de 3 variabile  $f_3(C, B, A)$  și  $f_4(C, B, A)$  sunt selectate respectiv prin  $FED = 001$ ,  $FED = 000$ , Figura P2.60-d. Implementarea mai multor funcții diferite, de aceleași variabile, pe aceeași structură ROM, pe intervale disjuncte de adrese, duce la o mai bună utilizare a capacității circuitului dar, evident, funcțiile implementate nu pot fi selectate simultan!



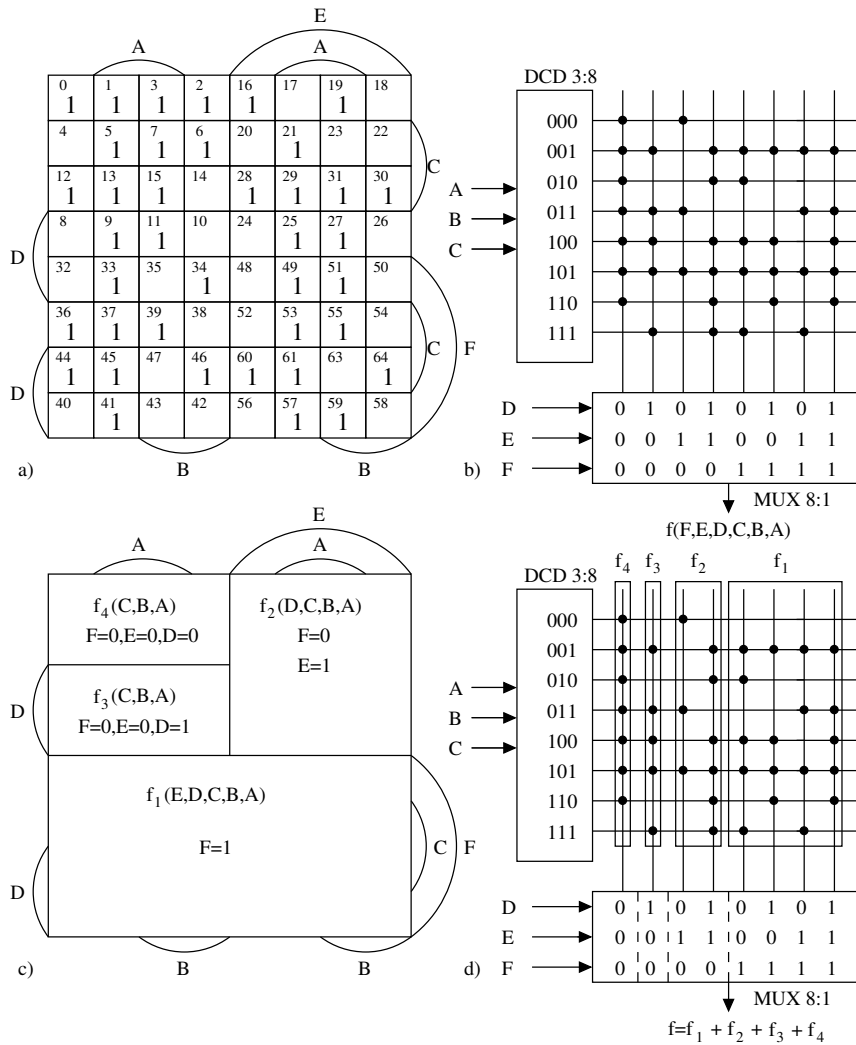


Figura P 2.60

**P2.61***Rezolvare:*

Se reprezintă pe diagrama V-K și apoi se deduc variabilele de selectare (vezi P2.60).

**P2.64***Rezolvare:*

Tabelul de adevăr pentru conversia din BCD în Gray este dat în Figura P2.64-a, iar forma minimă pentru fiecare din biții de ieșire  $W, X, Y, Z$  se obțin cu diagramele V-K din Figura P2.64-b. Formele minime sunt implementate pe o structură generică de PLA în Figura

P2.64-c și pe o structură generică de PAL în Figura P2.64-d. La implementarea pe PAL s-a ținut cont de faptul că fiecare poartă OR poate colecta doar patru termeni produs.

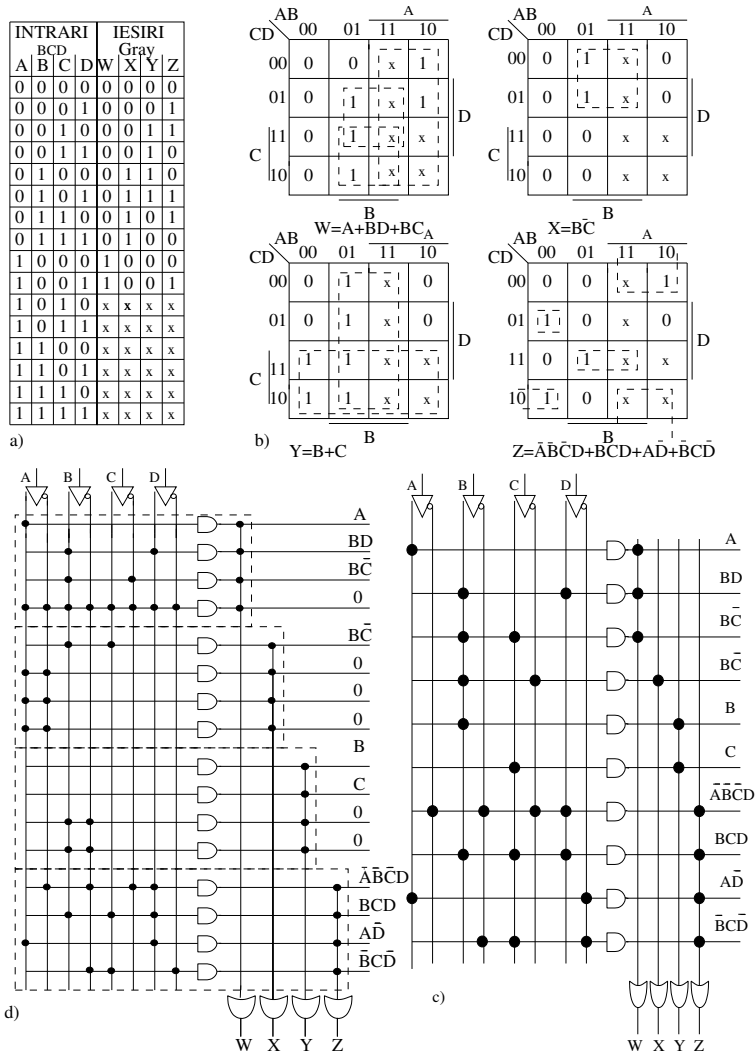


Figura P 2.64

**P2.67***Rezolvare*

O primă variantă de circuit se poate realiza folosind circuitul standard comparator 74XX85. Acest circuit determină pentru două cuvinte A și B de patru biți următoarele relații:  $A < B$ ,  $A = B$  și  $A > B$ . Pentru a putea fi extins la lungimi de cuvânt mai mari de patru biți circuitul posedă trei intrări  $F_i, F_e, F_s$  care indică faptul dacă circuitul repartizat biților cu patru poziții mai spre stânga realizează respectiv relațiile  $A < B$ ,  $A = B$  și  $A > B$ . O variantă de circuit pentru identitatea a două cuvinte de un byte este prezentată în Figura P2.67-a, la care identitatea între semnalele superioare  $A_7 - A_4$  și  $B_7 - B_4$  se transmite ca un semnal  $F_e$  la al doilea circuit 74XX85. Varianta din Figura P2.67-b determină identitatea celor două semnale ale celor două cuvinte A și B cu ajutorul unei porți AND, această variantă este mai rapidă decât prima deoarece cele două circuite 74XX85 lucrează în paralel.

O altă variantă se poate realiza cu porți XNOR, această poartă generează identitatea a doi biți  $f_e$ . Pentru identitatea simultană a patru perechi de biți trebuie să fie adevărată funcția  $f_e = f_{e_3} \cdot f_{e_2} \cdot f_{e_1} \cdot f_{e_0}$  (Figura P2.67-c). (Aceași funcție  $f_e$  poate fi realizată cu patru porți XOR și o poartă NOR cu patru intrări deoarece  $f_e = f_{e_3} \cdot f_{e_2} \cdot f_{e_1} \cdot f_{e_0} = \overline{f_{e_3} + f_{e_2} + f_{e_1} + f_{e_0}}$ ).

Pentru operatorul SAU-EXCLUSIV-NEGAT se poate utiliza circuitul 74XX266 care conține patru porți XNOR cu colectorul în gol care sunt conectate într-o conexiune SI cablat, Figura P2.67-d. Dacă se compară două cuvinte de patru biți ( $m = 4$ ) și la ieșirea porților cu colectorul în gol se comandă o poartă TTL standard ( $N = 1$ ) se poate scrie pentru determinarea intervalului de valori al rezistenței R următoarele relații (marginile de zgomot sunt:  $M_L = 0V$ ;  $M_H = 1V$ ).

$$R_{min} \geq \frac{V_{CCmax} - (V_{ILmax} - M_L)}{I_{OLmax} - N \cdot I_{ILmax}} = \frac{5,5V - (0,8V - 0V)}{8mA - 1 \times 1,6mA} = 734\Omega$$

$$R_{max} \leq \frac{V_{CCmin} - (V_{IHmin} + M_H)}{m \cdot I_{OHmax} + N \cdot I_{IHmax}} = \frac{4,5V - (2V + 1V)}{4 \cdot 0,1mA + 1 \cdot 0,04mA} = \frac{1,5V}{0,404mA} = 3,71k\Omega$$

$$\text{deci } 0,734k\Omega \leq R \leq 3,71k\Omega$$

Dacă se extinde această structură de circuit la cuvinte cu lungimi mai mari scade limita superioară a rezistenței R până se ajunge la condiția limită  $R_{max} = R_{min}$  (datorită faptului că va crește componenta de curent  $m \cdot I_{OH}$  absorbită în starea HZ). Introducând această condiție de limită în relația anterioară, pentru calculul rezistenței  $R_{max}$ , se obține lungimea maximă  $m_{max}$  a cuvintelor care pot fi comparate (se consideră  $M_H = 0,4V$ )

$$\begin{aligned} m_{max} &= \frac{V_{CCmin} - (V_{IHmin} + M_H) - R_{min} \cdot I_{IH}}{R_{min} \cdot I_{OH}} = \\ &= \frac{4,5V - (2V + 0,4V) - 0,734k\Omega \cdot 0,04mA}{0,734k\Omega \cdot 0,1mA} = \frac{2,07V}{0,073V} = 28 \end{aligned}$$

- Varianta din Figura P2.67-e utilizează pentru determinarea relației de identitate  $f_{e_i}$  între doi biți  $A_i, B_i$  un MUX4:1 cu intrările conectate la 1 și 0. Pentru identitatea a două cuvinte de  $m$  biți vor fi utilizate  $m \times$  MUX4:1 a căror ieșiri se colectează într-o poartă AND.

- Cu ajutorul a două  $DCD_{An} : 2^n$  și  $DCD_{Bn} : 2^n$  se poate determina identitatea a două cuvinte. Cele două ieșiri de același rang,  $O_{Ai}$  și  $O_{Bi}$ , se colectează în câte o poartă AND2, cuvintele de comparat se aplică pe intrările decodificatoarelor. Poarta AND2 cu ieșirea activă indică și configurațiile de biți aplicate pe intrări.
- Se poate implementa un circuit de identitate pentru cuvinte de  $n$  biți și pe un EPROM cu  $2^{2n} \times 1$  biți. Numai  $2^n$  locații sunt utilizate din cele  $2^{2n}$ .
- Imaginați și alte variante de circuite.

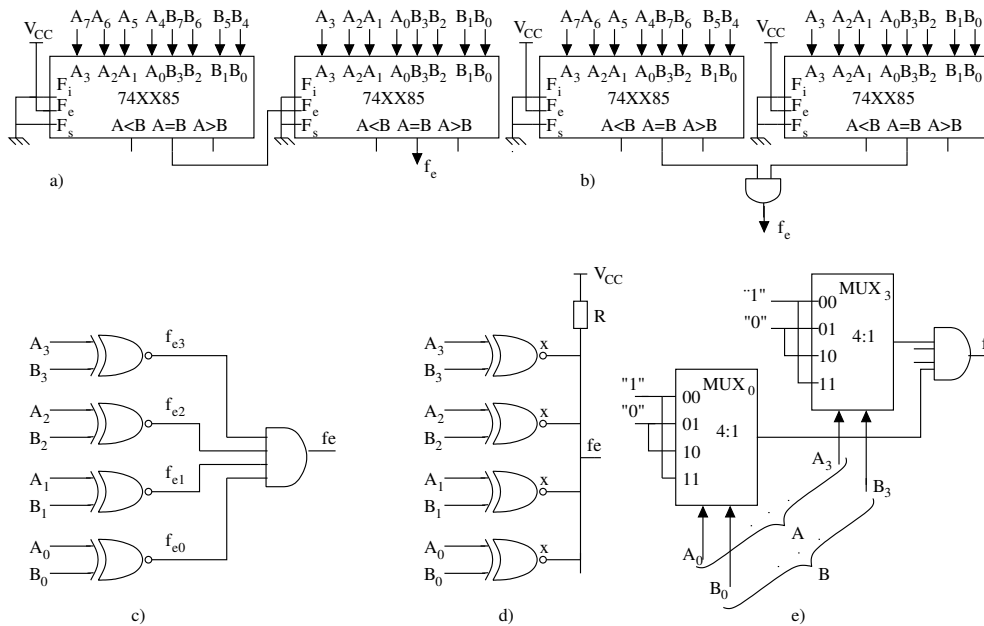


Figura P 2.67

## P2.68

Rezolvare:

- a) Structura de circuit din Figura P2.68-a generează la ieșirile circuitului 74XX85 funcțiile conform funcționării normale ale acestuia pentru  $c=0$  iar când  $c=1$  ieșirile circuitului sunt conform cu cerințele acestei probleme (se poate demonstra această funcționare din analiza valorilor conținute în tabelul din Figura P2.68-b). O altă soluție este obținută prin conectarea pe intrarea A a cuvântului B iar pe intrarea B a cuvântului A.
- b) Relațiile de ordine se obțin cu porțile care procesează semnalele de la ieșirea circuitului 74XX85. Selectarea unei relații de ordine se face cu un MUX8:1, Figura P2.68-c cu ajutorul cuvântului  $c_2c_1c_0$ .

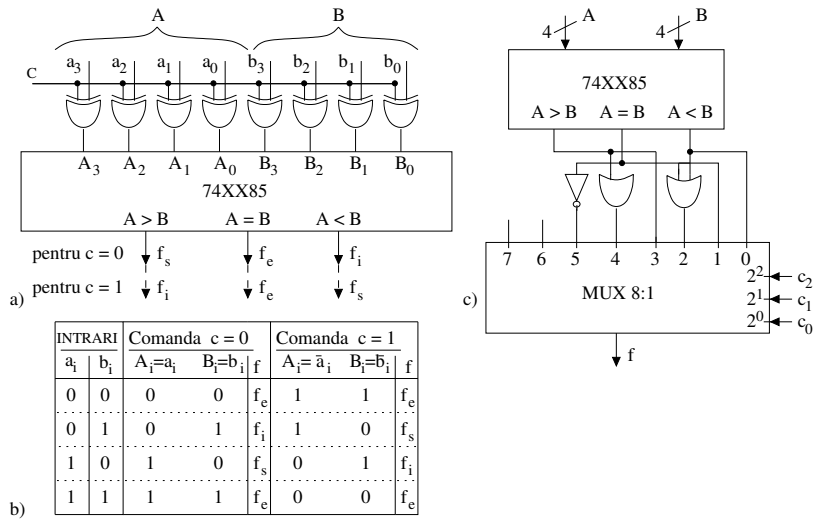


Figura P 2.68

**P2.72***Rezolvare*

Eliminarea unor ranguri (care pot fi alăturate sau nealăturate la variantele de doi biți) pune problema propagării transportului prin acele ranguri. Propagarea unui transport  $C_i$  prin rangul eliminat  $i + 1$  necesită ca variabila propagare  $p_i$  pe rangul  $i + 1$  să fie 1, adică  $A_{i+1} + B_{i+1} = 1$ . Se va exemplifica cu două variante.

Varianta de sumator de trei biți din Figura P2.72-a pentru că  $A_3 = 0, B_3 = 0$ , transportul  $C_2$  nu se va obține la ieșirea  $C_3$  ci la ieșirea  $S_3$ .

Varianta de sumator de doi biți din Figura P2.72-b folosește poziția  $A_0, B_0$  pentru rangul  $2^0$  și poziția  $A_3, B_3$  pentru rangul  $2^3$ , pozițiile  $A_1, B_1$  și  $A_2, B_2$  nu sunt utilizate deci trebuie realizate propagările  $p_1 = 1$  și  $p_2 = 1$  prin aplicarea  $A_2A_1 = 00$  și  $B_2B_1 = 11$ . Biții de sumă  $s_1, s_0$  se obțin la  $S_3$  și  $S_0$  iar transportul  $C_1$  la  $C_3$

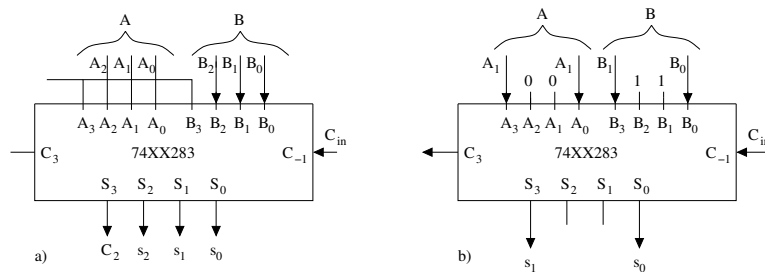


Figura P 2.72

**P2.73***Rezolvare:*

Celula semisumator în bază trei este reprezentată în Figura P2.73-a, are două intrări: cei doi operanzi  $A_i = a_1a_0, B_i = b_1b_0$  și două ieșiri: suma  $s_i = s_1s_0$  și transportul următor  $C_i$  (nu se consideră transportul anterior  $C_{i-1}$ ). Tabelul operației sumă modulo trei este dat în Figura P2.73-b (transportul generat este notat în paranteză) iar tabelul de adevăr pentru celula semisumator este prezentat în Figura P2.73-c. Realizând sinteza și ținând cont de configurațiile operanzilor (11) care nu pot apare pe intrări rezultă circuitul implementat cu porți NAND din Figura P2.73-d.

**P2.74***Rezolvare:*

Celula sumator modulo 3 se obține din două semisumatoare modulo 3 ca în Figura P2.74-a. Pe primul sumator se sumează cei doi operanzi generându-se transportul  $C_{i1}$  și o sumă. Această sumă este adunată, pe un al doilea sumator, cu cuvântul  $0C_{i-1}$  care se aplică ca operandul  $B_i (= 0C_{i-1})$ , se obține valoarea sumei  $s_i$  și un transport  $C_{i2}$ . Transportul  $C_i$  se obține ca o funcție logică OR între  $C_{i1}$  și  $C_{i2}$ . Un sumator modulo  $3^8$  cu transport progresiv se obține prin punerea în paralel a 8 celule sumator ale căror circuite de transfer al transportului se înscriază ca în Figura P2.74-b. Propagarea transportului  $C_{i-1}$  la  $C_i$  pe o celulă se face cu o întârziere de patru niveluri logice. Timpul de propagare al transportului pe sumatorul modulo  $3^8$ , considerând aplicarea simultană a operanzilor  $A_i, B_i (i = 0, 1, \dots, 7)$  și a transportului inițial  $C_{-1}$  este egal cu: 6 niveluri logice pe rangul  $3^0$  și câte patru niveluri logice pe celelalte ranguri, deci în total  $6 + 7 \times 4 = 34$  niveluri logice.

**P2.75***Rezolvare:*

Adunarea a două cuvinte în cod BCD,  $A_i = A_{i3}A_{i2}A_{i1}A_{i0}B_i = B_{i3}B_{i2}B_{i1}B_{i0}$  și transportul anterior  $C_{i-1}$  poate genera un cuvânt  $C_i s_{i3} s_{i2} s_{i1} s_{i0}$  care reprezintă un număr în binar natural în intervalul  $[0,19]$ . Efectuarea adunării BCD pe un sumator binar de patru biți generează un cod corect în BCD numai dacă cuvântul sumă reprezintă în binar un număr în intervalul  $[0,9]$ , și respectiv un cuvânt de cod eronat dacă rezultă un număr binar în intervalul  $[10,19]$ . Codul corectat în BCD, și pentru intervalul  $[10,19]$ , se obține dacă la suma obținută se adună  $6|_{10} = 0110|_2$  sau se scade  $10|_{10} = 1010|_2$ , Figura P2.75-a. Se observă că prin această adunare sau scădere de corecție bitul sumă  $s_{i0}$  nu se modifică. O primă variantă de celulă sumator complet în BCD este reprezentată în Figura P2.75-b. Pentru rangul  $2^0$  biții  $A_{i0}$  și  $B_{i0}$  se sumează pe o celulă  $\sum(3, 2)$ . Iar pentru rangurile  $2^1, 2^2$  și  $2^3$  se realizează o sinteză de subcircuit BCD, care este descris de un tabel de adevăr cu șapte intrări  $A_{i3}, B_{i3}, A_{i2}, B_{i2}, A_{i1}, B_{i1}$  și  $C_{i0}$  și generează patru ieșiri  $s_{i3}, s_{i2}, s_{i1}$  și  $C_i$  (complicat!).

A doua variantă de celulă sumator, Figura P2.75-c utilizează metoda de corecție explicată anterior. Cele două cuvinte  $A_i, B_i$  în cod BCD, plus  $C_{i-1}$  se aplică la primul sumator binar de patru biți rezultând cuvântul sumă  $s_{i3}^* s_{i2}^* s_{i1}^* s_{i0}^*$  și transportul  $C'_i$ . Această sumă este aplicată la al doilea sumator binar de trei biți unde se adună cu valoarea de corecție 0000 sau  $0110|_2 = 6|_{10}$ . Comanda pentru valoarea de corecție este tocmai transportul  $C_i$ . Acest bit de transport se compune din transportul  $C'_i$  care se generează pentru suma în

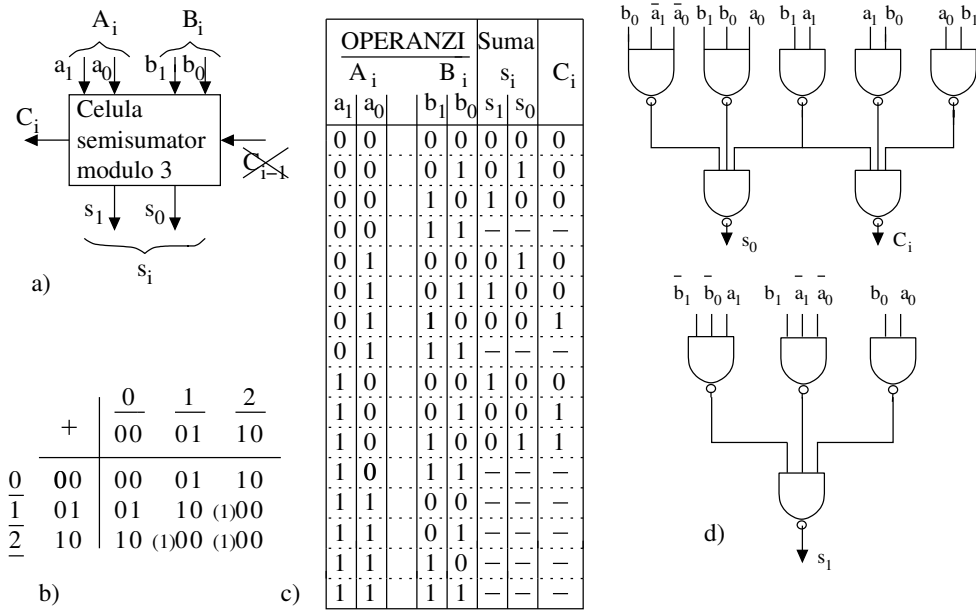


Figura P 2.73

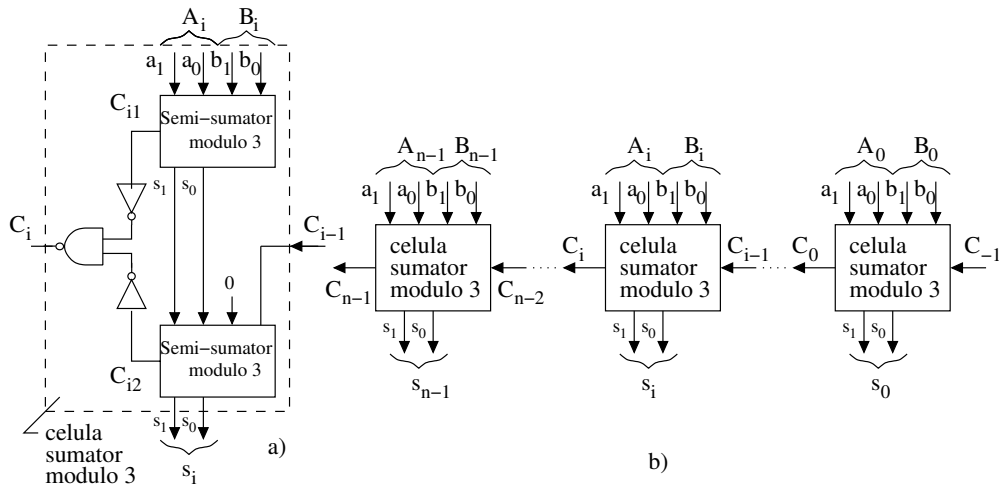


Figura P 2.74

intervalul [16,19] sau când suma este în intervalul [10,15], iar detectarea acestui al doilea interval se face cu relația  $(s_{i3}^* \cdot s_{i2}^*) + (s_{i3}^* \cdot s_{i1}^*)$ .

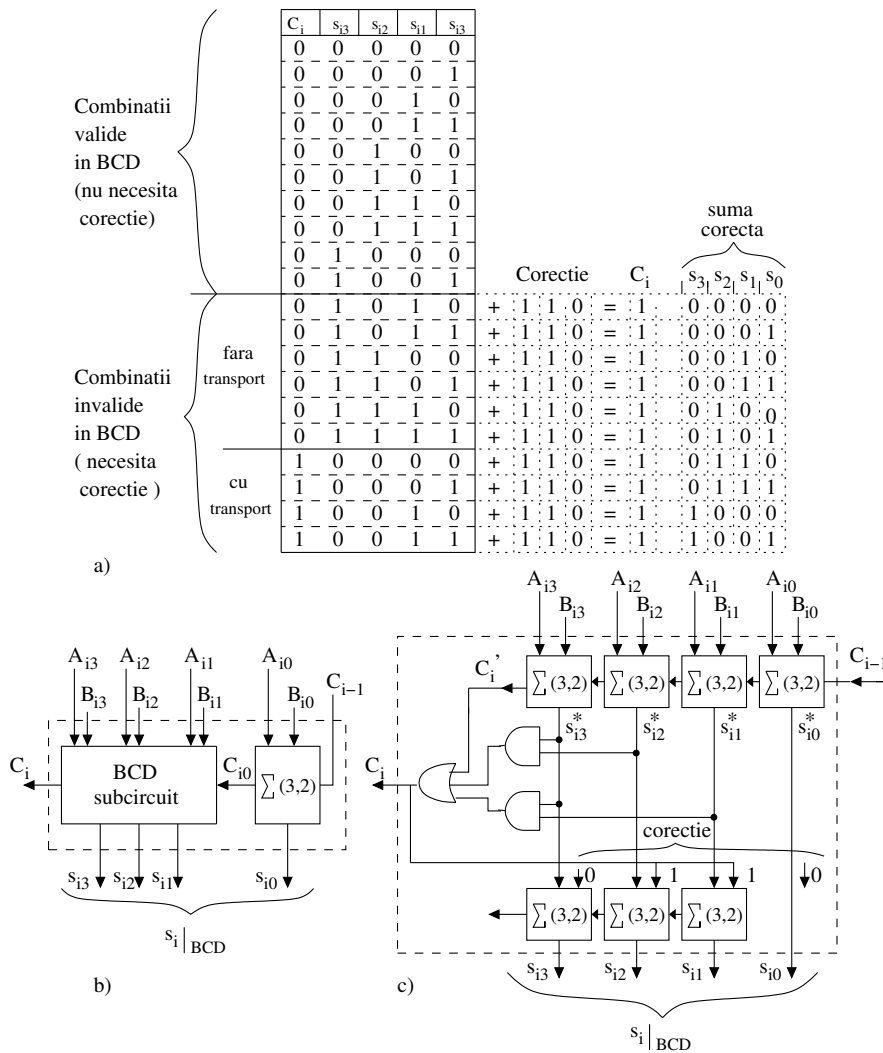


Figura P 2.75

### P2.76

Rezolvare:

O unitate logico-aritmetică pentru un grup de 16 biți poate fi structurată din patru circuite 74LS181, iar în exteriorul acestora se realizează o cale pentru generarea transportului anticipat, pentru fiecare ALU de patru biți, cu un circuit 74LS182, Figura P2.76. Deci,



această structură poate fi o unitate logico-aritmetică pentru cei doi bytes mai puțin semnificativi, adică pentru grupul de biți  $15 \div 0$  din lungimea de 64 de biți. La fel se realizează o astfel de structură pentru fiecare din celelalte trei grupuri  $63 \div 48$ ,  $47 \div 32$  și  $31 \div 16$ . Considerând fiecare grup ca o unitate logico-aritmetică de patru biți care generează semnalele P și G se poate realiza în exteriorul acestor patru grupuri, pe un circuit 74LS181, o cale pentru generarea transportului anticipat pentru fiecare grup. În total sunt necesare  $16 \times 74LS181$  și  $5 \times 74LS182$ . În figură sunt prezentate numai semnalele de control, nu și biții cuvintelor de intrare și de ieșire.

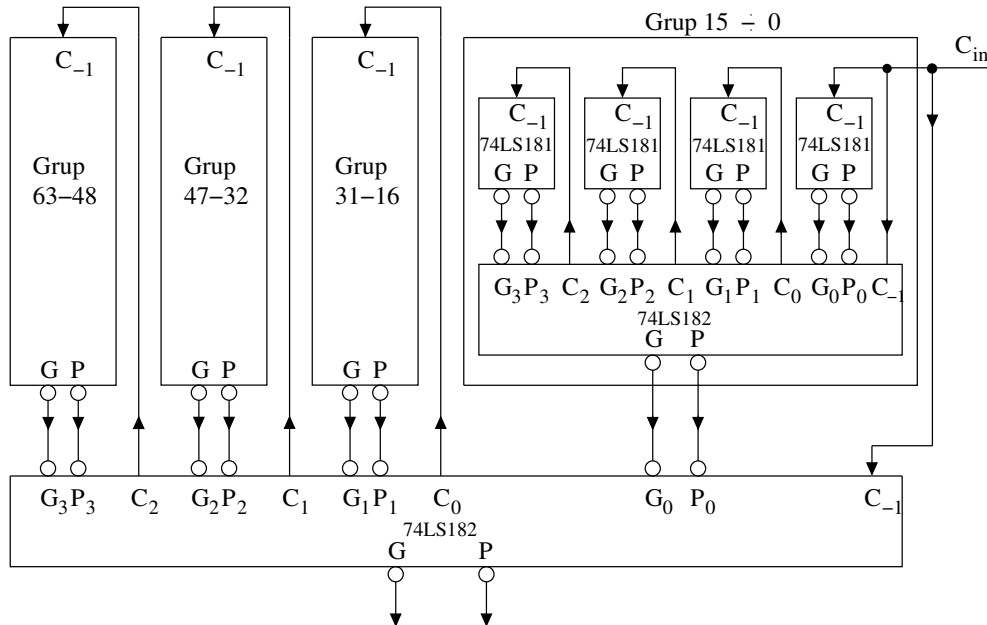


Figura P 2.76

### P2.77

*Rezolvare:*

Dacă la ieșirea unei celule sumator complet  $\sum(3,2)$  se consideră cuvântul  $C_i s_i$ , ca un număr în binar natural, atunci acesta reprezintă numărul de biți care au valoarea 1 în cuvântul de intrare  $A_i B_i C_{i-1}$  (vezi Tabelul 1.6, ultimele trei coloane). Astfel, din cuvântul de intrare se aplică grupuri de câte trei la câte o celulă  $\sum(3,2)$  obținându-se la ieșire  $C_i s_i$  care reprezintă numărul de biți 1 în grupul respectiv; apoi se adună aceste numere pe un sumator cu propagarea transportului, Figura P2.77-a. Pentru cuvântul de un byte, deoarece numărul rezultat (al biților care au valoarea 1) poate fi exprimat în binar cu 4 biți (de exemplu  $8|_2 = 1000$ , pentru un șir de opt biți 1) adunarea se efectuează întâi pe un numărător cu două celule și apoi pe unul cu trei celule, Figura P2.77-b.

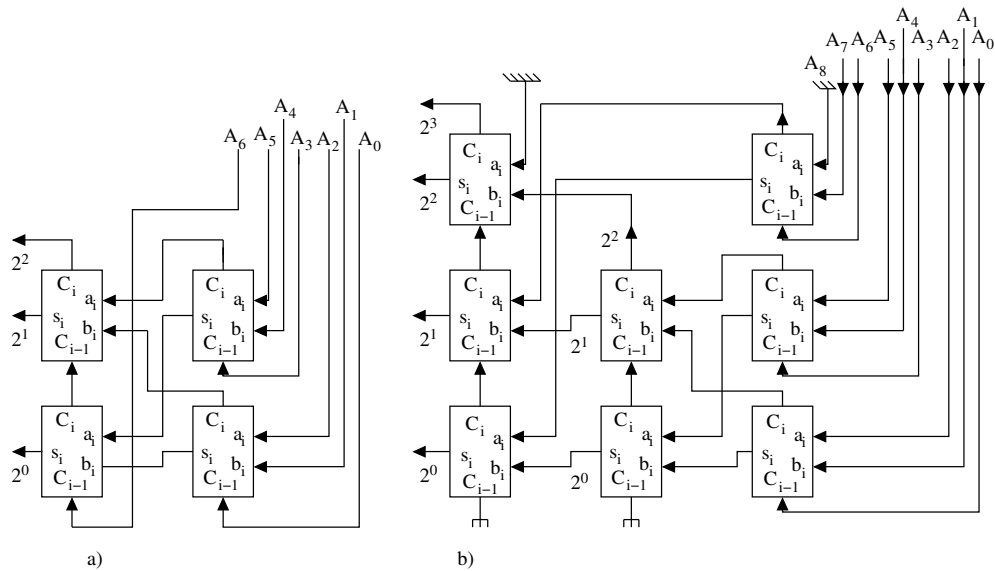


Figura P 2.77

## P2.78

*Rezolvare:*

Pentru incrementare corespunde Figura P2.78-a iar pentru decrementare Figura P2.78-b. Decrementarea poate fi exprimată în felul următor  $A - 1 = A_2A_1A_0 + [-1]_2 = A_2A_1A_0 + 111$ .

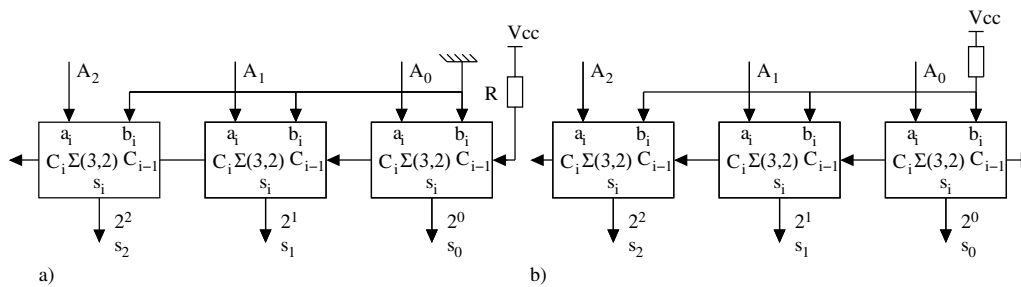


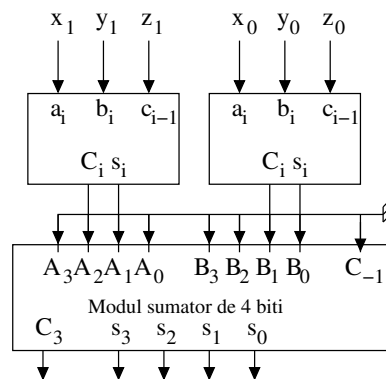
Figura P 2.78

**P2.79***Rezolvare:*

- $C_5 = A_5 \cdot B_5 + C_4 \cdot A_5 + C_4 \cdot B_5;$   
 $C_5 = g_5 + p_5 \cdot g_4 + p_5 \cdot p_4 \cdot g_3 + p_5 \cdot p_4 \cdot p_3 \cdot g_2 + p_5 \cdot p_4 \cdot p_3 \cdot p_2 \cdot g_1 + p_5 \cdot p_4 \cdot p_3 \cdot p_2 \cdot p_1 \cdot g_0$
- Pentru SPT,  $C_5$  se calculează cu 4 porți, iar pentru STA,  $C_5$  se calculează cu 6 porți (vezi expresia anterioară) plus două porți care calculează  $p_i$  și  $p_i$  ( $i = 0, 1, 2, 3, 4, 5$ ) dar care sunt utilizate de mai multe ranguri de sumare.
- SPT:  $3\tau_p$  pentru rangul  $2^0$ ,  $4 \times 2\tau_p$  pentru rangurile  $2^1, 2^2, 2^3, 2^4$  și  $\tau_p$  pentru rangul  $2^5$ , deci  $T_{\Sigma} = 12\tau_p$ .  
 STA: 1 nivel pentru calculul variabilelor intermediare  $p_i, g_i$ ; două niveluri pentru calculul lui  $C_i$ , iar dacă se consideră  $3\tau_p$  pe celula de sumare rezultă  $6\tau_p$ . Creșterea de viteză la STA față de SPT rezultă  $|6\tau_p - 12\tau_p|/12\tau_p \times 100 = 50\%$ .

**P2.80***Rezolvare:*

Deoarece suma maximă poate fi  $9|_{10} = 1001|_2$  sunt necesare două module sumatoare, care în general sunt de patru biți; pe primul modul se sumează  $X = x_1x_0$  cu  $Y = y_1y_0$  iar pe al doilea se sumează  $Z = z_1z_0$  cu  $c_1s_1s_0$ . Se poate elimina un modul sumator de patru biți realizând suma  $X + Y$  pe un sumator cu salvarea transportului, realizat pe două celule sumator complet  $\Sigma(3, 2)$ , după structurarea din Figura P2.80.

**Figura P 2.80****P2.81***Rezolvare:*

Pentru multiplicarea  $x_1x_0 \times y_2y_1y_0$  corespunde circuitul din Figura P2.81-a, iar pentru multiplicarea  $y_2y_1y_0 \times x_1x_0$  corespunde circuitul din Figura P2.81-b.

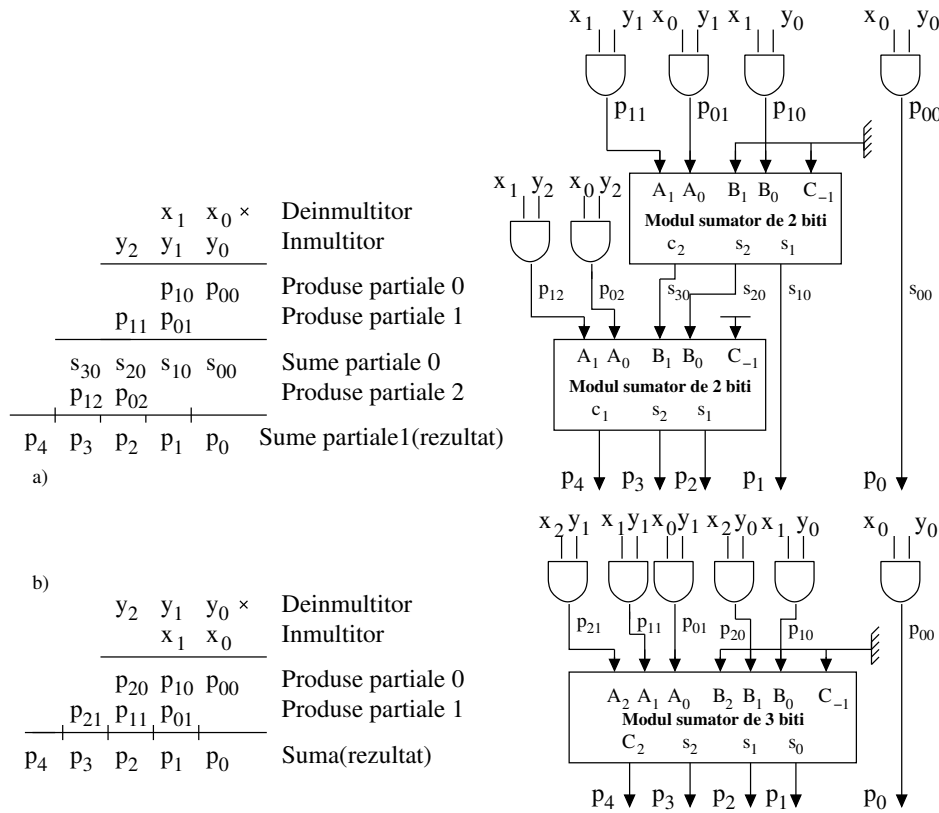


Figura P 2.81

**P2.82***Rezolvare:*

ALUE se structurează ca un selector de date pe bază de multiplexor. Din tabelul din Figura P2.82-a rezultă că două din operații sunt logice: pentru  $S_1S_0 = 00 \rightarrow F = \overline{B}$ ; pentru  $S_1S_0 = 01 \rightarrow A > B$ , care se poate scrie sub forma  $F = \overline{B}A$ , deci pot fi implementate cu porți logice. Celelalte două operații sunt aritmetice și pot fi implementate pe două celule sumator complet  $\sum(3, 2)$ . Celula de scăzător complet se poate obține dintr-o celulă de sumator complet la care se aplică  $\overline{A}_i$  iar bitul de sumă se complementează  $F = \overline{s}_i$ . Toate aceste operații sunt realizate pe circuite separate, Figura P2.82-b iar apoi sunt selectate cu două MUX4:1. Operațiile logice ar necesita doar un singur MUX4:1, deoarece produce doar bitul de funcție F, pe când operațiile aritmetice necesită două MUX4:1, deoarece prin aceste operații aritmetice se generează atât bitul de funcție cât și bitul de transport următor  $C_i$ .



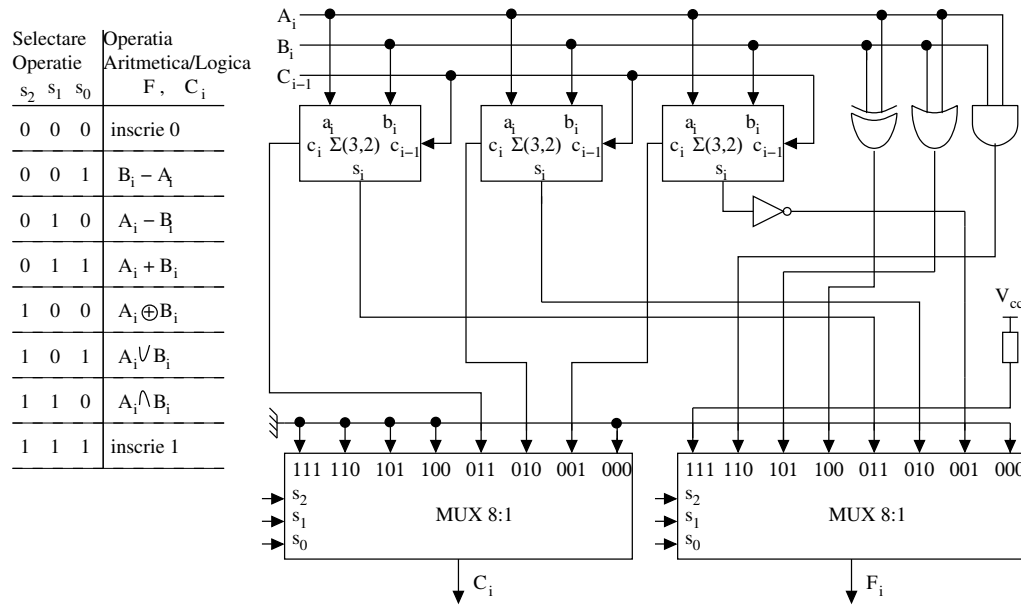


Figura P 2.84