Capitolul 2

Circuite logice combinaţionale

P2.1

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

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

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

Prima expresie poate fi mapată pe o reţea serie (rs) iar a doua pe o reţea arbore binar (rb). Adoptând numărul de porţi XOR ca măsură pentru dimensiunea reţelei 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 substituţia $XOR2 \rightarrow NXOR2$ se calculează aceeaşi funcţie dacă un număr de $(n-1)$ biţi ai cuvântului, înainte de aplicare la intrarea reţelei, sunt complementaţi deoarece: $x_{i+1} \oplus x_i = \overline{x_{i+1} \oplus \overline{x_i}}$.

P2.2

Rezolvare:
$f_1 = \overline{(A+B)(C+D)} = \overline{A}\,\overline{B} + \overline{C}\,\overline{D}$
$f_2 = AB \oplus CD = AB(\overline{CD}) + (\overline{AB})CD = AB(\overline{C} + \overline{D}) + CD(\overline{A} + \overline{B}) =$
$= AB\overline{C} + AB\overline{D} + \overline{A}CD + \overline{B}CD$
$f_3 = \overline{A}BC + \overline{B}D$
$f_4 = \overline{((A \oplus \overline{B})(\overline{C} \oplus D))} = AB + \overline{A}\,\overline{B} + C\overline{D} + \overline{C}D$

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

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

P2.4

Rezolvare:

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

$$b)\quad \overline{f} = \overline{\overline{(A + B + \overline{C})(\overline{AB} + \overline{CD})} + \overline{BCD}} = \overline{(A + B + \overline{C})}\,\overline{(\overline{AB} + \overline{CD})} + \overline{BCD} =$$
$$= (\overline{A}\,\overline{B}\,\overline{\overline{C}})(\overline{\overline{AB}}\,\overline{\overline{CD}}) + \overline{BCD} = (\overline{A}\,\overline{B}C)(ABCD) + \overline{BCD} = \overline{BCD}.$$

$$c)\quad \overline{f} = \overline{\overline{(ABC + B\overline{C}D)} + \overline{(\overline{ACD} + \overline{B}\,\overline{C}\,\overline{D} + BC\overline{D})}} =$$
$$= (ABC + B\overline{C}D)(\overline{ACD} + \overline{B}\,\overline{C}\,\overline{D} + BC\overline{D}) =$$
$$= (ABC + B\overline{C}D)(\overline{A} + \overline{C} + \overline{D} + \overline{B}\,\overline{C}\,\overline{D} + BC\overline{D}) =$$
$$= (ABC + B\overline{C}D)(\overline{A} + \overline{D}(1 + BC) + \overline{C}(1 + \overline{B}\,\overline{D})) =$$
$$= (ABC + B\overline{C}D)(\overline{A} + \overline{D} + \overline{C}) =$$
$$= ABC\overline{A} + ABC\overline{D} + ABC\overline{C} + B\overline{C}D\overline{A} + B\overline{C}D\overline{D} + B\overline{C}D\overline{C} =$$
$$= 0 + ABC\overline{D} + 0 + B\overline{C}D\overline{A} + 0 + B\overline{C}D =$$
$$= ABC\overline{D} + (\overline{A}B\overline{C}D + B\overline{C}D) = ABC\overline{D} + B\overline{C}D$$

P2.5

Rezolvare:

a) Implicanţi primi:

$$\{0,4\} \to \overline{B}\,\overline{C}; \qquad \{0,2\} \to \overline{A}\,\overline{C}; \qquad \{2,3\} \to \overline{A}\,B;$$
$$\{3,7\} \to BC; \qquad \{4,5\} \to A\overline{B}; \qquad \{5,7\} \to AC;$$

Există două variante de alegere a implicanţilor primi prin care sunt acoperiţi toţi 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 obţinut toţii implicanţii primi de la forma sumă de produse.

b) Pentru forma sumă de produse din Figura P2.5-b rezultă următorii implicanţi primi:
$\{0,1,4,5\} \to \overline{A}\,\overline{C}$;
$\{9,11,13,15\} \to AD$;
$\{1,5,9,13\} \to \overline{C}D$.
Dintre aceşti implicanţi primi, numai $\overline{A}\,\overline{C}$ şi AD sunt implicanţi primi esenţiali care trebuie să fie consideraţi pentru acoperirea tuturor mintermilor din diagrama V-K. Expresia minimă a funcţiei este:

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

Pentru forma produse de sume din Figura P2.5-b rezultă următorii implicanţi primi:
$\{2,3,6,7\} \to A + \overline{C}$;
$\{8,10,12,14\} \to \overline{A} + D$;
$\{2,6,10,14\} \to \overline{C} + D$.
Dintre aceşti implicanţi primi, numai $A + \overline{C}$ şi $\overline{C} + D$ sunt implicanţi primi esenţiali care trebuie consideraţi pentru acoperirea tuturor maxtermilor din diagrama V-K. Expresia minimă a funcţiei este:

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

Prin desfacerea parantezelor s-au obţinut toţi implicanţi primi de la forma sumă de produse.

c) Pentru forma sumă de produse din Figura P2.5-c rezultă următorii implicanţi primi:
$\{0,4,8,12,16,20,24,28\} \to \overline{D}\,\overline{E}$;
$\{8,9,10,11,12,13,14,15\} \to \overline{A}B$.
Amândoi implicanţi primi sunt esenţiali, deci acoperirea funcţiei se realizează prin expresia minimă:

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

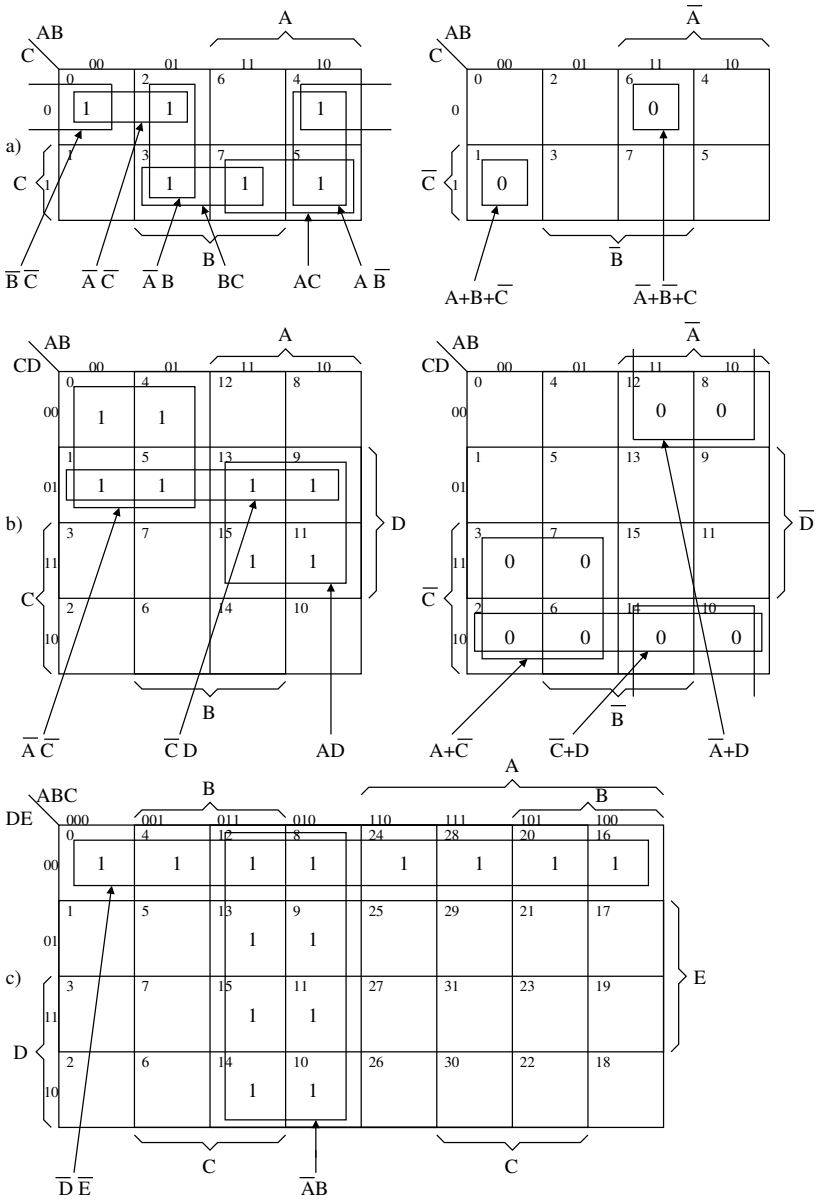

a)
b)
c)
AB
CD
ABC
DE
A
B
C
D
E
$\overline{B}\,\overline{C}$
$\overline{A}\,\overline{C}$
$\overline{A}\,B$
BC
AC
$A\,\overline{B}$
$A+B+\overline{C}$
$\overline{A}+\overline{B}+C$
$\overline{A}\,\overline{C}$
$\overline{C}\,D$
AD
$A+\overline{C}$
$\overline{C}+D$
$\overline{A}+D$
$\overline{D}\,\overline{E}$
$\overline{A}B$

Figura P 2.5

P2.6

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

P2.7

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

P2.9

Rezolvare:

Termenii produs care sunt indiferenţi, notaţi cu *d(don't care)* pot fi consideraţi în diagrama V-K fie de valoare 0 fie 1. Se va alege valoarea logică ce va duce la identificarea de implicanţi primi cu o exprimare cât mai simplă. În diagrama V-K, aceasta înseamnă definirea unor suprafeţe cât mai mari si cât mai puţine pentru a acoperi căsuţele cu 1.
Pentru funcţiile anterioare se obţin următoarele forme minime: $a) \quad f = \overline{B}C + AD + \overline{A}B\overline{C}$; $b) \quad f = \overline{C}D + BD$; $c) \quad f = \overline{D}$; $d) \quad f = \overline{C}E + C\overline{E}$.

P2.10

Rezolvare:

O funcţie de j variabile se poate exprima ca o funcţie de $j-1$ variabile şi o variabilă reziduu. Variabila reziduu se introduce în expresia funcţiei reziduu pentru calculul coeficienţilor. Generarea variabilelor reziduu se poate continua recursiv până când se ajunge la $j-1$ variabile reziduu şi o singură variabilă pentru funcţie.
Se va exemplifica pentru funcţia de patru variabile:

$$\begin{aligned} f(A,B,C,D) &= (d_0\overline{D} + d_1 D)\overline{A}\,\overline{B}\,\overline{C} + (d_2\overline{D} + d_3 D)\overline{A}\,\overline{B}C + (d_4\overline{D} + d_5 D) \\ &\quad \overline{A}B\overline{C} + (d_6\overline{D} + d_7 D)\overline{A}BC + (d_8\overline{D} + d_9 D)A\overline{B}\,\overline{C} + (d_{10}\overline{D} + \\ &+ \quad d_{11}D)A\overline{B}C + (d_{12}\overline{D} + d_{13}D)AB\overline{C} + (d_{14}\overline{D} + d_{15}D)ABC \end{aligned}$$

Coeficienţii funcţiei de trei variabile A, B, C se calculează ca o sumă dintre produsul unui coeficient d_i al funcţiei de patru variabile cu variabila $\overline{D}$ şi produsul coeficientului următor d_{i+1} al funcţiei cu D. Coeficienţii d_i ai funcţiei de patru variabile se pot deduce fie din tabelul de adevăr, fie din diagrama V-K. Coeficienţii funcţiei de trei variabile calculaţi din expresiile funcţiilor reziduu pot avea doar valorile: $1, 0, D, \overline{D}$.
De asemenea, pentru funcţia de patru variabile, când se introduc în coeficienţi două variabile reziduu (C şi D), se obţine exprimarea:
$f(A,B,C,D) = (d_0\overline{C}\,\overline{D} + d_1\overline{C}D + d_2C\overline{D} + d_3CD)\overline{A}\,\overline{B} + (d_4\overline{C}\,\overline{D} + d_5\overline{C}D + d_6C\overline{D} + d_7CD)\overline{A}B + (d_8\overline{C}\,\overline{D} + d_9\overline{C}D + d_{10}C\overline{D} + d_{11}CD)A\overline{B} + (d_{12}\overline{C}\,\overline{D} + d_{13}\overline{C}D + d_{14}C\overline{D} + d_{15}CD)AB$
Coeficienţii funcţiei rezultate de două variabile AB au forma:

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

şi pot fi calculaţi uşor ca sumă a patru produse dintre coficienţii consecutivi ai funcţiei de patru variabile cu mintermii corespunzători ai variabilelor C şi D. Aceşti coeficienţi, calculaţi cu ajutorul unei funcţii reziduu, pot avea expresia celor 16 funcţii de două variabile dintre care 6 snt valori banale: $0, 1, C, \overline{C}, D, \overline{D}$.
Introducerea varaibilelor reziduu în tabelul de adevăr este prezentată în figura P2.10-a.

a)

	A	B	C	D	d_i
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	d
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	d
6	0	1	1	0	1
7	0	1	1	1	0
8	1	0	0	0	0
9	1	0	0	1	0
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	d
13	1	1	0	1	d
14	1	1	1	0	1
15	1	1	1	1	1

Variabile reziduu:nici una

	A	B	C	d_i
0	0	0	0	0
1	0	0	1	D/1
2	0	1	0	D/0
3	0	1	1	$\overline{D}$
4	1	0	0	0
5	1	0	1	D
6	1	1	0	0/1
7	1	1	1	1

Variabila reziduu: D

	A	B	d_i
0	0	0	CD/C
1	0	1	$CD \oplus C\overline{D}$
2	1	0	CD
3	1	1	C/1

Variabile reziduu:CsiD

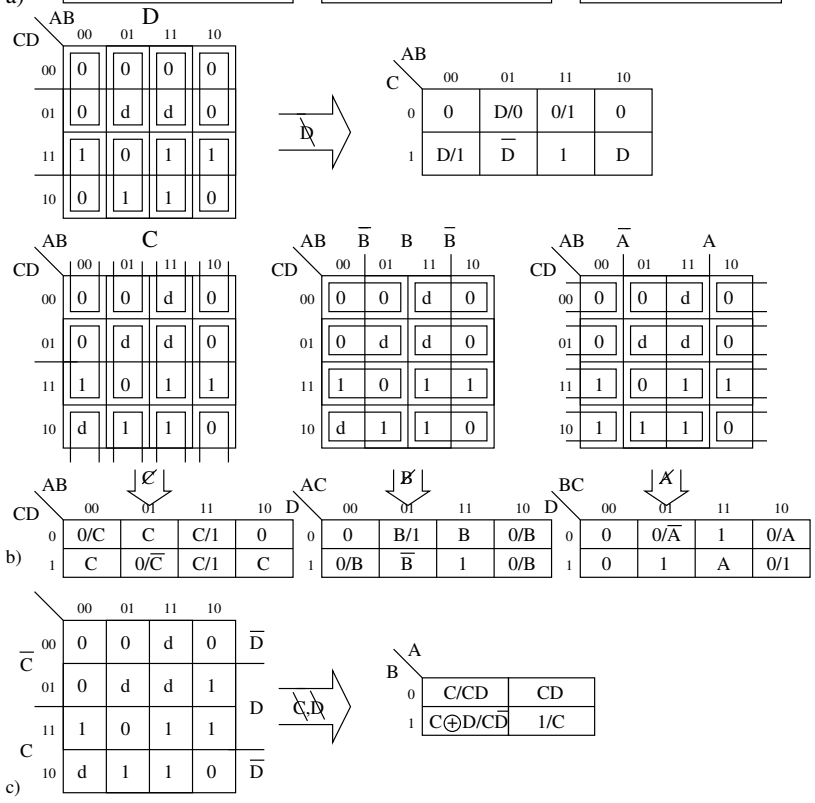


Figura P 2.10

Introducerea unei variabile reziduu (D) în diagramele V-K este prezentată în figura P2.10-b.
$(0 \cdot \overline{D} + 0 \cdot D) \cdot \overline{A}\,\overline{B}\,\overline{C} = 0 \cdot \overline{A}\,\overline{B}\,\overline{C}$;
$(d \cdot \overline{D} + 1 \cdot D) \cdot \overline{A}\,\overline{B}C = (D/1) \cdot \overline{A}\,\overline{B}C$;
$(0 \cdot \overline{D} + d \cdot D) \cdot \overline{A}B\overline{C} = (D/0) \cdot \overline{A}B\overline{C}$;
$(1 \cdot \overline{D} + 0 \cdot D) \cdot \overline{A}BC = \overline{D} \cdot \overline{A}BC$;
$(0 \cdot \overline{D} + 0 \cdot D) \cdot A\overline{B}\,\overline{C} = 0 \cdot A\overline{B}\,\overline{C}$;
$(0 \cdot \overline{D} + 1 \cdot D) \cdot A\overline{B}C = \overline{D} \cdot A\overline{B}C$;
$(d \cdot \overline{D} + d \cdot D) \cdot AB\overline{C} = (0/1) \cdot AB\overline{C}$;
$(1 \cdot \overline{D} + 1 \cdot D) \cdot ABC = 1 \cdot ABC$;

Introducerea a două variabile reziduu (C,D) în diagramele V-K este prezentată in figura P2.10-c.
$(0 \cdot \overline{C}\,\overline{D} + 0 \cdot \overline{C}D + d \cdot C\overline{D} + 1 \cdot CD) \cdot \overline{A}\,\overline{B} = (C/CD) \cdot \overline{A}\,\overline{B}$;
$(0 \cdot \overline{C}\,\overline{D} + d \cdot \overline{C}D + 1 \cdot C\overline{D} + 0 \cdot CD) \cdot \overline{A}B = (C \oplus D/C\overline{D}) \cdot \overline{A}B$;
$(0 \cdot \overline{C}\,\overline{D} + 0 \cdot \overline{C}D + 0 \cdot C\overline{D} + 1 \cdot CD) \cdot A\overline{B} = (CD) \cdot A\overline{B}$;
$(d \cdot \overline{C}\,\overline{D} + d \cdot \overline{C}D + 1 \cdot C\overline{D} + 1 \cdot CD) \cdot AB = (1/C) \cdot AB$;

P2.11

Rezolvare:
Operaţia de minimizare pe diagrama V-K se realizează in trei paşi.
Pasul 1: Se grupează căsuţele care au valori logice 1 şi cele care au valori indiferente *(d)* pentru a se exprima cât mai simplu implicanţii primi.

Pasul 2: Căsuţele cu valoare 1 se consideră a fi cu valoare *d*. Se grupează căsuţele care conţin aceeaşi funcţie reziduu şi cu unele căsuţe (potrivite) care conţin valori indiferente pentru a exprima cât mai simplu implicanţi primi. Implicanţii primi rezultaţi se vor înmulţi cu funcţia reziduu.

Pasul 3: Forma minimă a funcţiei se obţine prin sumarea logică a implicanţilor primi obţinuţi de la primul şi al doilea pas. Dacă funcţiile reziduu au mai mult de o variabilă, în cazuri particulare, expresia se mai poate reduce prin prelucrări analitice.

Aplicând succesiv cei trei paşi pentru fiecare funcţie din diagramele date se obţin formele minime prezentate în Figura P2.11.

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

b)

Functia	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 + 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 + A$\bar{C}$E
f_9	BC	$\bar{A}$DE + A$\bar{C}\bar{E}$	BC + $\bar{A}$DE + A$\bar{C}\bar{E}$
f_{10}	AC	$\bar{B}\bar{D}$E + ABF	AC + $\bar{B}\bar{D}$E + ABF

Figura P 2.11

P2.13

Rezolvare:

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

$A = \sum_0^{15}(1, 10) + \sum_0^{15} d(2, 3, 4, 5, 6, 7)$;
$B = \sum_0^{15}(11, 12) + \sum_0^{15} d(2, 3, 4, 5, 6, 7)$;
$C = \sum_0^{15}(8) + \sum_0^{15} d(2, 3, 4, 5, 6, 7)$;
$D = \sum_0^{15}(1, 10, 13) + \sum_0^{15} d(2, 3, 4, 5, 6, 7)$;
$E = \sum_0^{15}(1, 9, 10, 11, 13, 15) + \sum_0^{15} d(2, 3, 4, 5, 6, 7)$;
$F = \sum_0^{15}(1, 8, 9, 13) + \sum_0^{15} d(2, 3, 4, 5, 6, 7)$;
$G = \sum_0^{15}(0, 1, 13) + \sum_0^{15} d(2, 3, 4, 5, 6, 7)$.

Aceleaşi funcţii, 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 funcţii 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 şi ca sume de produse. Prin aplicarea dublei negaţii formei de sumă de produse (obţinută din *Forma Normală Disjunctivă - FND*), se poate implementa funcţia cu operatori NAND. Similar, prin aplicarea dublei negaţii formei minime de produs de sume (obţinută din *Forma Normală Conjunc-*

a)

CIFRA ZECIM	BCD 2421 INTRARI w x y z	Termeni Min P_i	Termeni Max S_i	IESIRI Codul 7 segmente A B C D E F G
0	0 0 0 0	P_0	S_0	0 0 0 0 0 0 1
1	0 0 0 1	P_1	S_1	1 0 0 1 1 1 1
2	1 0 0 0	P_8	S_8	0 0 1 0 0 1 0
3	1 0 0 1	P_9	S_9	0 0 0 0 1 1 0
4	1 0 1 0	P_{10}	S_{10}	1 0 0 1 1 0 0
5	1 0 1 1	P_{11}	S_{11}	0 1 0 0 1 0 0
6	1 1 0 0	P_{12}	S_{12}	0 1 0 0 0 0 0
7	1 1 0 1	P_{13}	S_{13}	0 0 0 1 1 1 1
8	1 1 1 0	P_{14}	S_{14}	0 0 0 0 0 0 0
9	1 1 1 1	P_{15}	S_{15}	0 0 0 0 1 0 0

c)

Iesire	Porti NAND suma de produse	Porti NOR produs de sume
A	$\overline{W}Z + \overline{X}\overline{Y}Z$	$\overline{X}(Y+Z)(\overline{W}+\overline{Z})$
B	$X\overline{Y}\overline{Z} + \overline{X}YZ$	$(X+Y)(\overline{X}+\overline{Z})(\overline{Y}+Z)$
C	$W\overline{X}\overline{Y}Z$	$W\overline{X}\overline{Y}\overline{Z}$
D	$X\overline{Y}Z + \overline{W}Z + \overline{X}Y\overline{Z}$	$(\overline{W}+X+Y)(\overline{Y}+\overline{Z})$ $(\overline{X}+\overline{Y})(Y+Z)$
E	$Z + \overline{X}Y$	$(\overline{X}+Z)(Y+Z)$
F	$W\overline{X}\overline{Y} + \overline{Y}Z$	$\overline{Y}(\overline{X}+Z)(W+Z)$
G	$X\overline{Y}Z + \overline{W}$	$\overline{Y}(\overline{W}+Z)(\overline{W}+\overline{X})$

b)

A

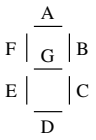
YZ \ WX	00	01	11	10
00	0	d	0	0
01	1	d	0	0
11	d	d	0	0
10	d	d	0	1

B

YZ \ WX	00	01	11	10
00	0	d	1	0
01	0	d	0	0
11	d	d	0	1
10	d	d	0	0

C

YZ \ WX	00	01	11	10
00	0	d	0	1
01	0	d	0	0
11	d	d	0	0
10	d	d	0	0



E

YZ \ WX	00	01	11	10
00	0	d	0	0
01	1	d	1	1
11	d	d	1	1
10	d	d	0	1

F

YZ \ WX	00	01	11	10
00	0	d	0	1
01	1	d	1	1
11	d	d	0	0
10	d	d	0	0

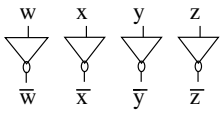
G

YZ \ WX	00	01	11	10
00	1	d	0	0
01	1	d	1	0
11	d	d	0	0
10	d	d	0	0

D

YZ \ WX	00	01	11	10
00	0	d	0	0
01	1	d	1	0
11	d	d	0	0
10	d	d	0	1

d)



I – 74LS00
J – 74LS10
K– 74LS20
L – 74LS04

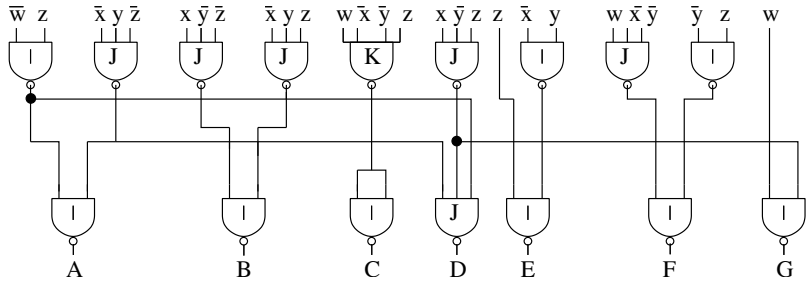


Figura P 2.13

tivă - FNC), se poate implementa funcţia cu operatori NOR. Expresiile minimizate ale funcţiilor, pentru o implementare cu operatori NAND şi pentru o implementare cu porţi NOR sunt prezentate in Figura P2.13-c.
Structura cu porţi NAND care implementează funcţiile este prezentată în Figura P2.13-d.

P2.15

Rezolvare:

Circuitele AND-OR şi NAND-NOR, care calculează termeni produs sunt potenţiale de producere de hazard static 1 deoarece ieşirea lor 1 (sinteza formei normale disjunctive a funcţiei 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, funcţia

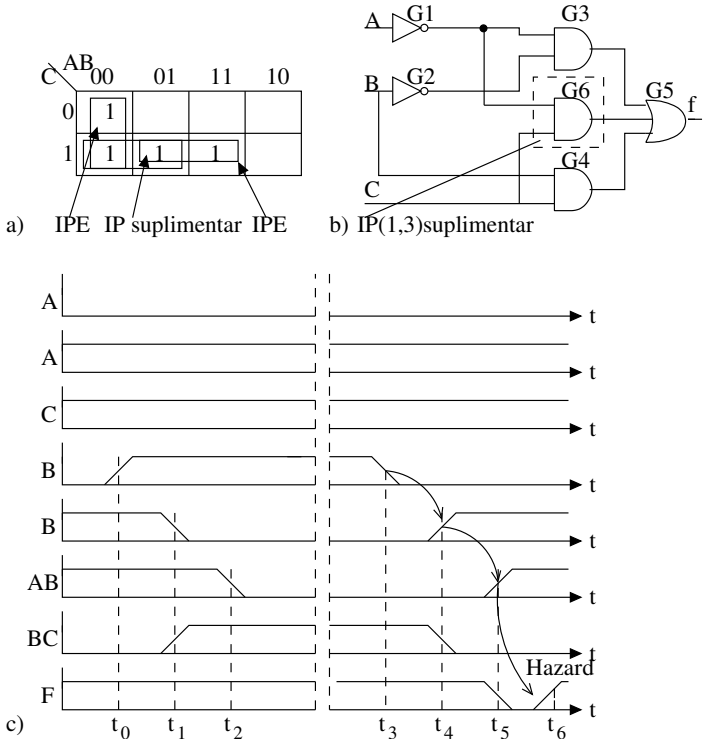


Figura P 2.15

se reduce la: $x + \overline{x} = 1$. Pentru circuitul considerat, funcţia $F(A, B, C) = \overline{A}\,\overline{B} + BC$ se reduce la $F(0, B, 1) = \overline{B} + B$, pentru A=0 şi C=1. Ca efect, implementarea propusă va determina apariţia unui glitch 0, cand B comută de la 1 la 0.
În diagramele de timp din Figura P2.15-c se observă că o comutaţie de la 0 la 1 a lui B la momentrul t_0 nu produce hazard, dar o comutaţie de la 1 la 0 la momentul t_3 generează un interval $t_5 - t_6$ când ieşirea devine 0, $(B + \overline{B} \neq 1)$. Similar, în diagrama V-K din Figura P2.15-a, când "punctul de funcţionare" se deplasează între IPE(0,1) şi IPE(3,7), nu apare

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

Soluţia de eliminare a hazardului este introducerea implicantului prim IP(1,3), "punte" între cei doi IPE. Acest IP menţine funcţia la valoarea 1 chiar când B comută de la 1 la 0. Această "punte" este implementată în Figura P2.15-c (prin introducerea porţii G_6).

P2.16

Rezolvare:

Circuitele OR-AND şi NOR-NOR, care calculează termeni sumă sunt potenţiale de producere de hazard static 0 deoarece ieşirea lor 0 (sinteza formei normale conjunctive a funcţiei 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 celorlate variabile, funcţia se reduce la forma: $x \cdot \overline{x} = 0$.

Miminizarea circuitului se realizează cu trei implicanţi primi esenţiali, prezentaţi în Figura P2.16.
$IPE(1,3,5,7) = (A + \overline{D})$,
$IPE(10,11,14,15) = \overline{A} + \overline{C}$,
$IPE(4,12) = (\overline{B} + C + D)$.
Acestora le corespunde expresia:

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

Când A=0, B=1 şi C=0, funcţia se reduce la $F = D\overline{D}$. La trecerea de la maxtermul 5 $(A + \overline{B} + C + D)$, la maxtermul 4 $(A + \overline{B} + C + \overline{D})$ 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 $IP(4,5) = (\overline{A} + \overline{B} + D)$.
Între IPE(10,11,14,15) şi IPE(4,12), când A=1, B=1 şi D=0, funcţia se reduce la $F = C\overline{C}$. La trecerea de la maxtermul 14 $(\overline{A}+\overline{B}+\overline{C}+D)$ la maxtermul 12 $(\overline{A}+\overline{B}+C+D)$ nu se produce hazard, dar la trecerea de 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 $IP(12,14) = (\overline{A} + \overline{B} + D)$.
Între IPE(10,11,14,15) şi IPE(1,3,5,7), când B=1, C=1 şi D=1, funcţia se reduce la $F = A\overline{A}$. La trecerea de la maxtermul 15 $(\overline{A}+\overline{B}+\overline{C}+\overline{D})$ la maxtermul 7 $(A+\overline{B}+\overline{C}+\overline{D})$ 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 $IP(3,7,11,15) = \overline{C} + D$. Aici s-a luat în considerare şi comutaţia de la maxtermul 3 la maxtermul 11.
În Figura P2.16-c este reprezentată implementarea care elimină hazardul static 0 pentru funcţia dată.

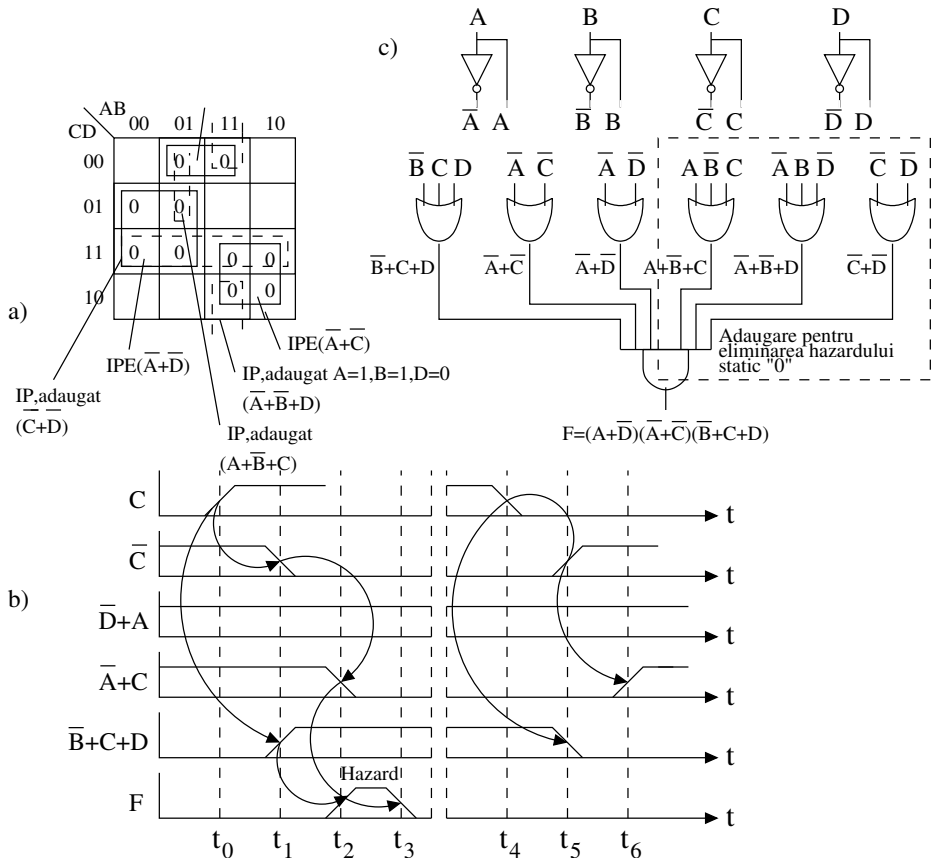


Figura P 2.16

P2.17

Rezolvare:

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

P2.19

Rezolvare:

Din structura circuitului se deduce expresia funcţiei logice:

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

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 şi C=1), la mintermul 3 (A=0, B=1 şi C=1), funcţia se reduce la expresia $F = A + \overline{A}$ care ia valoarea nepermisă 0 la comutaţia variabilei A de la 1 la 0.
Hazardul se poate elimina prin adăugarea implicantului prim BC obţinându-se funcţia $F = AB + \overline{A}C + BC$ cu implementarea din Figura P2.19-b.

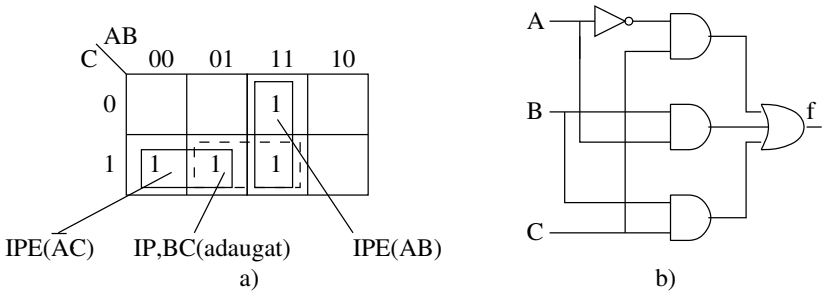


Figura P 2.19

P2.20

Rezolvare:
Tabelul de adevăr al funcţiei 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 obţine 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 şi $high(x_2)$ la nodul terminal pot fi excluse şi se obţine OBDD din Figura P2.20-d. Şi prin ultima transformare, eliminarea nodurilor neterminale care au $low(x) = high(x)$, un nod al variabilei x_1 şi unul al variabilei x_0 pot fi neglijate şi se obţine 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 porţi NAND este: BD dar nu ABDC şi AC dar nu ABCD deci funcţia f se exprimă:

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

cu implementarea din figura P2.21-b.

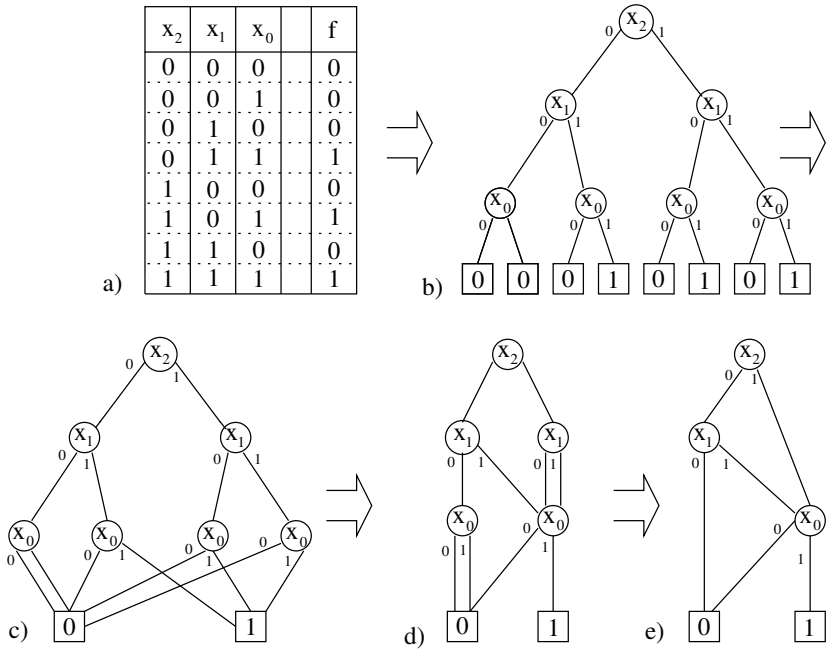


Figura P 2.20

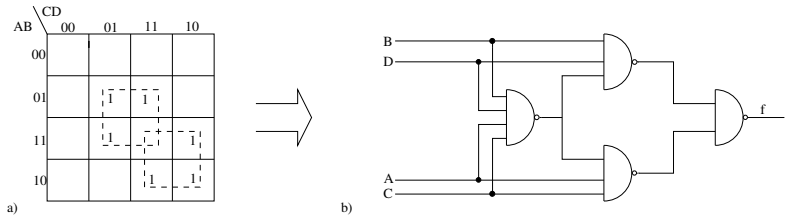


Figura P 2.21

P2.22

Rezolvare:

O definiţie posibilă pentru DAR este "f_1 este adevărată dacă A_1 şi B_1 sunt adevărate DAR fie A_2 sau B_2 sunt false; f_2 este adevărată dacă A_2 şi 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

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

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

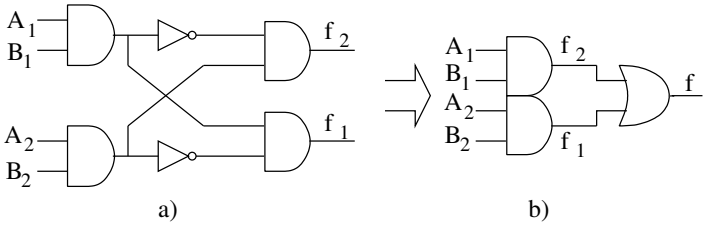


Figura P 2.22

P2.23

Rezolvare:

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

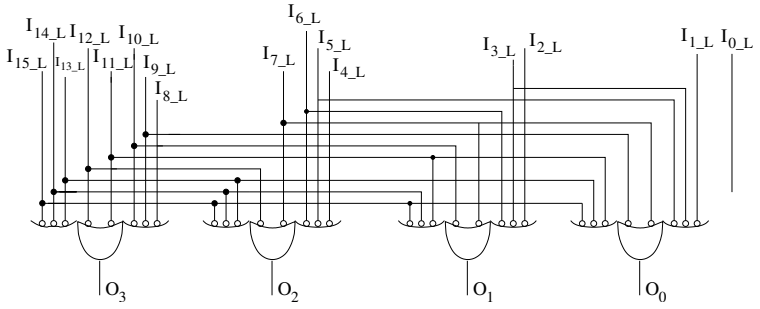


Figura P 2.23

P2.24

Rezolvare:

Circuitul 74XX148, al cărui tabel de adevăr şi simbol de reprezentare sunt date în Figura P2.24-a şi b, are o intrare de validare EI şi opt intrări ($I_7, \ldots, I_0; I_7$ are prioritatea cea mai ridicată) active în stare "low". Circuitul generează pe ieşirea $A_{2\_L}A_{1\_L}A_{0\_L}$ numărul intrării de prioritate cea mai ridicată care a fost activată. Numărul intrării este exprimat în binar, în complement faţă de 1, deoarece ieşirile sunt active în starea "low". Ieşirea GS este activată ("0") când cel puţin o intrare este activată. Ieşirea EO este activată ("0") când circuitul este validat de către semnalul EI_L dar nici o intrare nu a fost activată.
Semanlul EO_L poate fi utilizat pentru a cascada mai multe circuite, în scopul obţinerii unei structuri de codificator cu mai multe intrări.

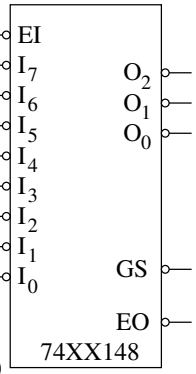
Observaţie:

Printr-un artificiu de conectare a semnalelor $I_0, I_1 \ldots I_7$ (I_7 fiind evenimentul cu nivelul cel mai ridicat de prioritate), la cele opt intrări în felul următor: $I_0 \rightarrow 7, I_1 \rightarrow 6 \ldots I_7 \rightarrow 0$, cuvântul de ieşire $y_{2\_L}y_{1\_L}y_{0\_L}$ 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 ieşire $y_{2\_L}y_{1\_L}y_{0\_L}$ adăugarea încă a unui bit y_{3_L}. Acest bit y_{3_L} va fi generat doar atunci când există cel puţin o intrare activă în intervalul $A_{15\_L} \ldots A_{8\_L}$, ceea ce determină totdeauna pentru al doilea circuit $74XX148_2$ să activeze semnalul GS_L. Deci, pentru generarea celui de al patrulea bit de ieşire se consideră $y_{3\_L} \equiv GS\_L$.
Pentru structurarea serie, figura P2.24-c, al doilea circuit este validat permanent prin conectarea semnalului EI_L la masă. În schimb, primul circuit este necesar să fie validat de către al doilea doar atunci când nici o intrare din intervalul $A_{15\_L} \ldots A_{8\_L}$ nu este activată (ieşirea EO devine activă) ceea ce se poate realiza prin conectarea $EO \equiv EI$. Biţii $y_{2\_L}, y_{1\_L}, y_{0\_L}$ ai subcuvântului de ieşire sunt colectaţi fie de la ieşirile O_2, O_1, O_0 ale primului codificator, când sunt activate intrări numai din intervalul $A_7 \ldots A_0$, SAU de al doilea codificator, când sunt activate intrări din intervalul $A_{15} \ldots A_8$. Deci , inplementarea se face cu porţi NAND (deoarece semnalele de la codificatoare sunt active în starea low şi cuvântul de ieşire este în complement faţa de unu). La această structurare serie, timpul de propagare fiind prin cele două codificatoare, viteza circuitului este scăzută. O îmbunătăţire se obţine prin structurarea paralelă.
Structurarea paralelă este reprezentată in figura P2.24-d. Validarea celor două circuite este permanentă prin legarea intrărilor EI la masă. Bitul y_{3_L} din cuvântul de ieşire este identic cu semnalul GS_L de la al doilea circuit $74XX148_2$, la fel ca la structurarea serie. În plus, acest semnal GS_L este folosit şi pentru selectarea subcuvântului $O_2O_1O_0$ de la primul circuit când sunt activate intrări numai din intervalul $A_7 \ldots A_0 (GS\_L = 1)$ sau de la al doilea circuit când sunt activate intrări şi din intervalul $A_{15} \ldots A_8 (GS\_L = 0)$. Implementarea selectării este realizată cu circuitul 74LS157 ($4 \times MUX2 : 1$)

a)

INTRARI									IESIRI				
$EI_{\_L}$	I_{0_L}	I_{1_L}	I_{2_L}	I_{3_L}	I_{4_L}	I_{5_L}	I_{6_L}	I_{7_L}	A_{2_L}	A_{1_L}	A_{0_L}	$GS_{\_L}$	$EO_{\_L}$
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	0	1	1	1	1	1	1	1	1	1	1	0	1
0	x	0	1	1	1	1	1	1	1	1	0	0	1
0	x	x	0	1	1	1	1	1	1	0	1	0	1
0	x	x	x	0	1	1	1	1	1	0	0	0	1
0	x	x	x	x	0	1	1	1	0	1	1	0	1
0	x	x	x	x	x	0	1	1	0	1	0	0	1
0	x	x	x	x	x	x	0	1	0	0	1	0	1
0	x	x	x	x	x	x	x	0	0	0	0	0	1

b)



c)

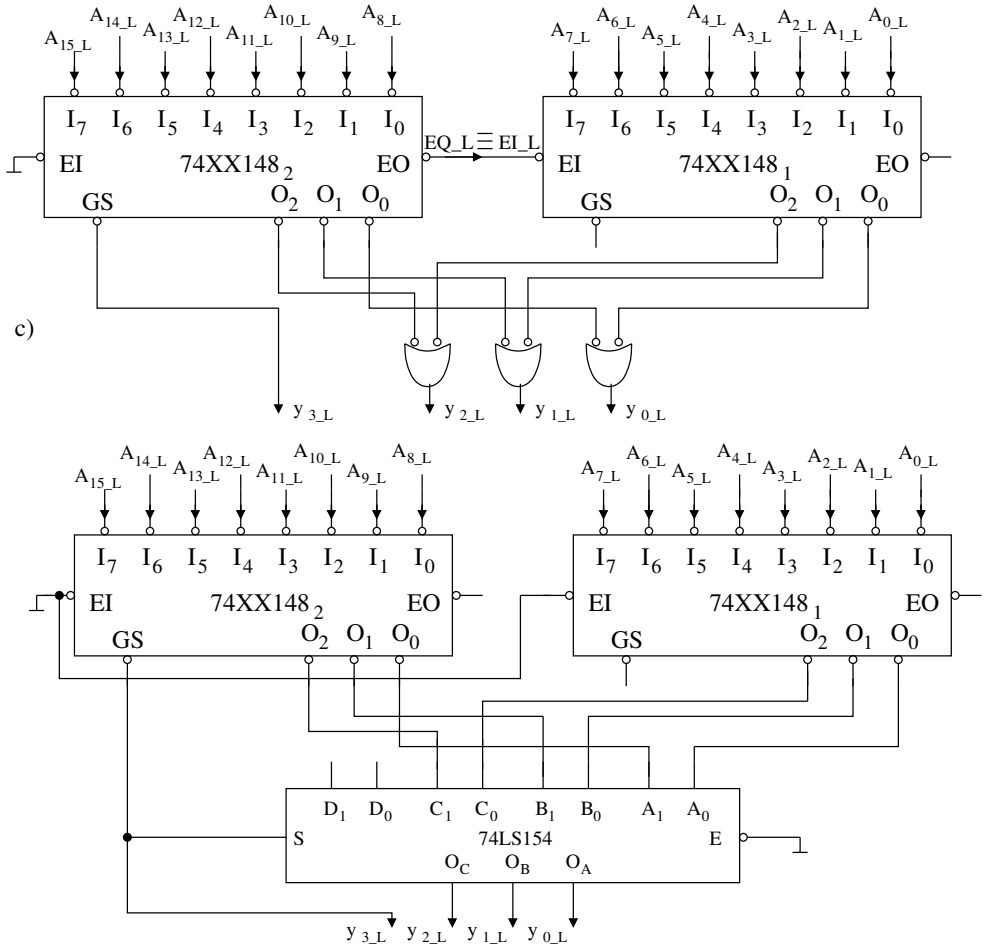


Figura P 2.24

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 relaţia:

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

pentru: $2^N \in \{0, 2, 4, 8, 16, 32, 64, 128\}$, $K \in [0, 255]$ întreg.
Relaţia este îndeplinită când dintre cei opt biţi $m_7 \dots m_0$ ai cuvântul M, cel puţin ultimii

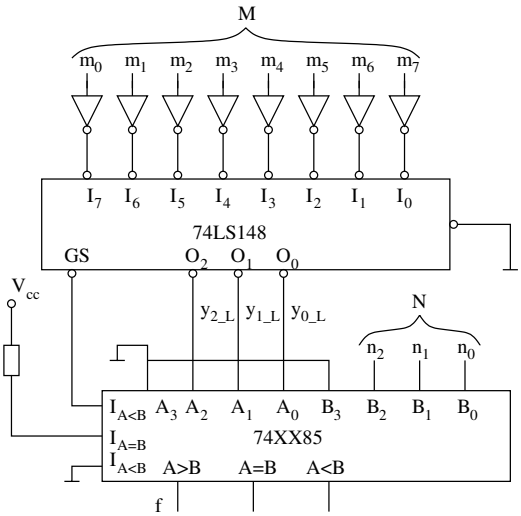


Figura P 2.25

N biţi sunt zero. Soluţia problemei se reduce la a determina dacă numărul M aplicat pe o intrare are cel puţin cei mai puţin semnificativi N biţi egali cu "0" când pe cealaltă intrare se aplică cuvântul N. Determinarea numărului de zerouri din poziţiile cele mai puţin semnificative ale cuvântului M este echivalentă cu aflarea numărului poziţiei primului bit 1, începând cu m_0.Implementarea acestei soluţii apare ca evidentă cu ajutorul circuitului comparator digital 7485: pe o intrare A a comparatorului se aplică numărul poziţiei m_i al primului bit egal cu 1 din cuvântul M, începând cu poziţia m_0, iar pe intrarea B se aplică cuvântul N. La ieşirea $A > B$ se obţine funcţia f, aşa ca în Figura P2.25.
Circuitul codificator prioritar, dacă se consideră în binar cuvântul de intrare I şi cuvântul de ieşire O, implementează relaţia:

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

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

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

ieşire, în cod binar natural şi nu complementat, poziţia primului bit activ pe intrare cu prioritatea cea mai scăzută, ceea ce este necesar în această problemă. Ieşirile O_2, O_1 şi O_0 ale codificatorului se conectează la intrările A_2, A_1, A_0 ale comparatorului digital. Biţii cuvântului M se aplică prin inversoare la intrările comparatorului 7485 deoarece intrările acestuia sunt active în stare "L".

P2.27

Rezolvare:

Soluţia acestei probleme se bazează pe Observaţia de la problema P2.24. Structura circuitului conţine pe două codicatoare prioritare: unul calculează prioritatea cea mai ridicată $74XX148_1$, în cod complementar faţa de 1, iar celalalt $74XX148_2$ 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 biţi (unul complementat faţa de 1, iar celălalt necomplementat) se obţine 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 obţine $s_3s_2s_1s_0 = 0111$ şi $C_3 = 1$. 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 ieşirea $f = s_3$ va fi zero. Deci detectarea singurului zero în cuvântul de intrare se realizează cu ieşirea s_3 iar numărul binar, $z_2z_1z_0$, care exprimă poziţia zero-ului în acest cuvânt este generat de $74XX148_2$.

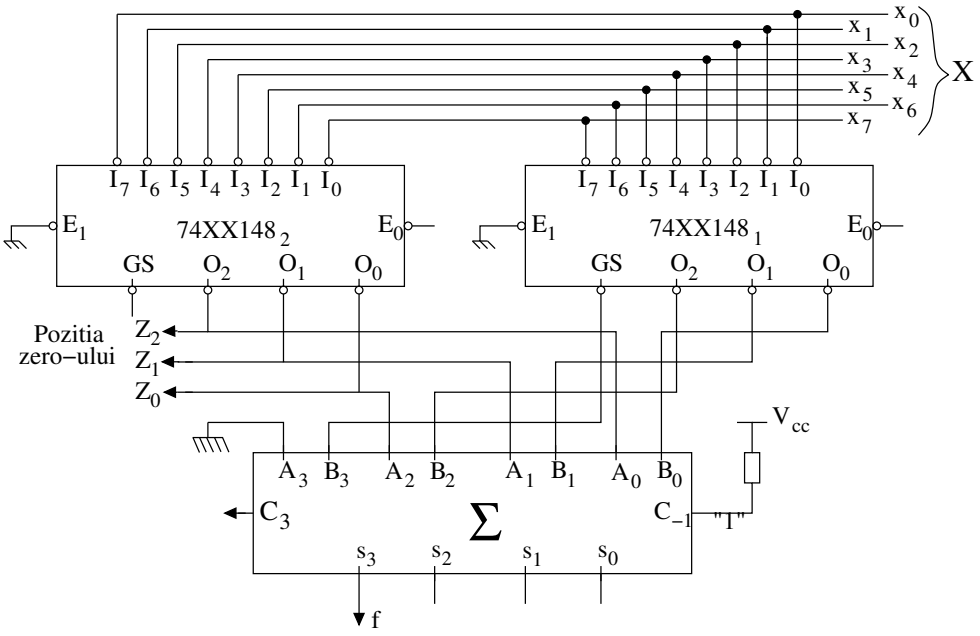


Figura P 2.27

P2.28

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

P2.29

Rezolvare:
Structura circuitului este dată în Figura P2.29. Codificatorul prioritar 74XX148 generează pe magistrala de date a μP codul perifericului activat $Y_{2\_L}Y_{1\_L}Y_{0\_L}$ 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 $3 \times MUX2:1$ din două surse:

1- de la ieşirea circuitului 74XX148 şi corespunde perifericului activat cu prioritatea cea mai ridicată, pentru s=0;

2- o valoare cablată 101 când $x_{7\_L} = 0$ SI $x_{2\_L} = 0$ pentru s=1. Valoarea s=1 se obţine când $y_{2\_L} = 0$, $y_{1\_L} = 0$, $y_{0\_L} = 0$ SI $x_{2\_L} = 0$ adică $\overline{y_{2\_L} + y_{1\_L} + y_{0\_L} + x_{2\_L}} = 1$, ceea ce se realizează cu un NOR4.

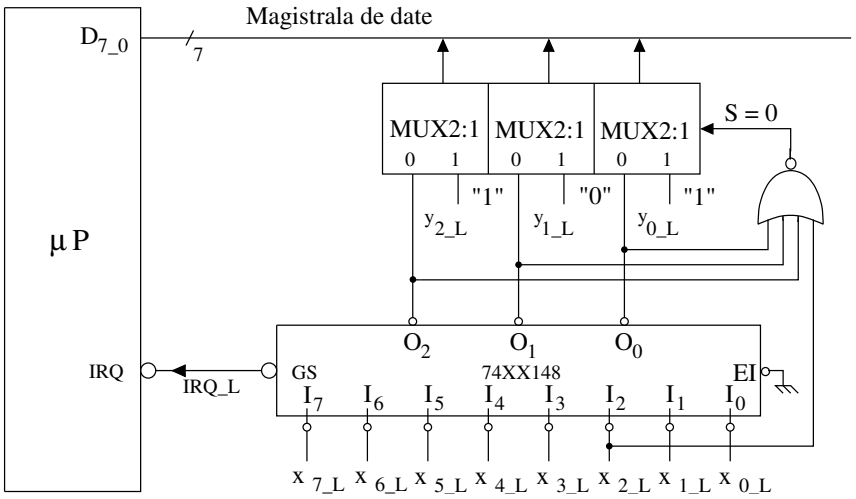


Figura P 2.29

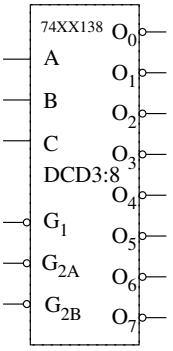
P2.30

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

intrările A,B,C se activează doar una din ieşirile y_{i_L}, $i = 0, 1 \ldots 7$. Aceasta decodificare este validată de conjuncţia a trei semnale de validare G_1, G_{2A_L} şi G_{2B_L}, adică $G_1 \cdot \overline{G}_{2A\_L} \cdot \overline{G}_{2B\_L} = 1$ ceea ce se realizează când $G_1 = 1, G_{2A\_L} = 0$ şi $G_{2B\_L} = 0$. 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 biţi x_2, x_1, x_0 mai puţin semnificativi din cuvântul de intrare. Primii doi biţi mai semnificativi x_4 şi x_3 sunt aplicaţi la decodificatorul din primul nivel şi de la acesta sunt utilizate doar patru ieşiri 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} mulţimea ieşirilor $y_{31\_L}, y_{30\_L} \ldots y_{1\_L}, y_{0\_L}$ este impărţită în patru intervale 31-24, 23-16, 15-8, 7-0 şi asignat câte un interval unui circuit decodificator. Precauţie trebuie acordată la factorul de încărcare al semnalelor de validare de pe nivelul al doilea.

a)

INTRARI						IESIRI							
de validare			de selectare										
G_1	G_{2A_L}	G_{1A_L}	A	B	C	O_{0_L}	O_{1_L}	O_{2_L}	O_{3_L}	O_{4_L}	O_{5_L}	O_{6_L}	O_{7_L}
0	x	x	x	x	x	1	1	1	1	1	1	1	1
x	1	x	x	x	x	1	1	1	1	1	1	1	1
x	x	1	x	x	x	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	0	1	1	0	1	1	1	1	1	1
1	0	0	0	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	0	0	1	0	0	1	1	1	1	0	1	1	1
1	0	0	1	0	1	1	1	1	1	1	0	1	1
1	0	0	1	1	0	1	1	1	1	1	1	0	1
1	0	0	1	1	1	1	1	1	1	1	1	1	0



b)

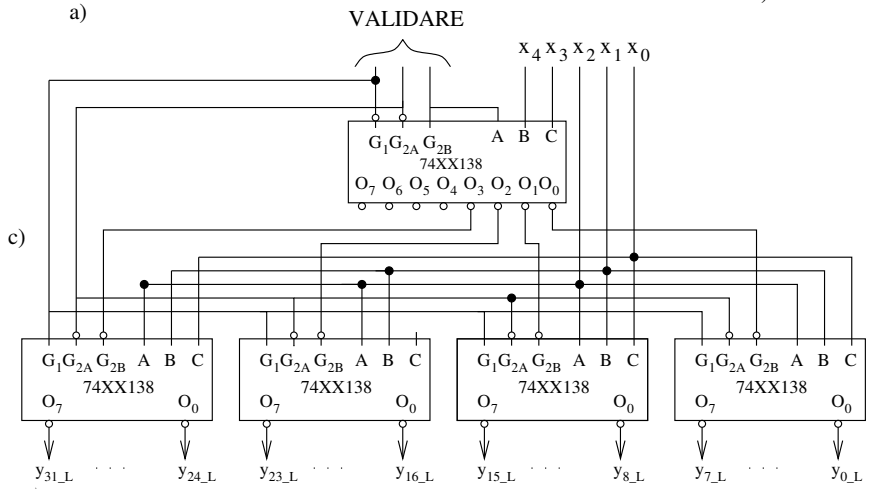


Figura P 2.30

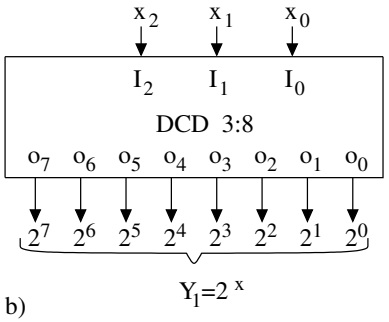
P2.31

Rezolvare:

Realizarea funcţiei 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	1 0
1	1 0	1 0 0
2	1 00	1 0 0 0 0
3	1 000	1 0 0 0 0 0 0 0 0
4	1 0000	1 0 0 0 0 0 0 0 0 . . . 0 (16 ori)
5	1 00000	1 0 0 0 0 0 0 0 0 . . . 0 (32 ori)
6	1 00000	1 0 0 0 0 0 0 0 0 . . . 0 (64 ori)
7	1 0000000	1 0 0 0 0 0 0 0 0 . . . 0 (128 ori)

a)



b)

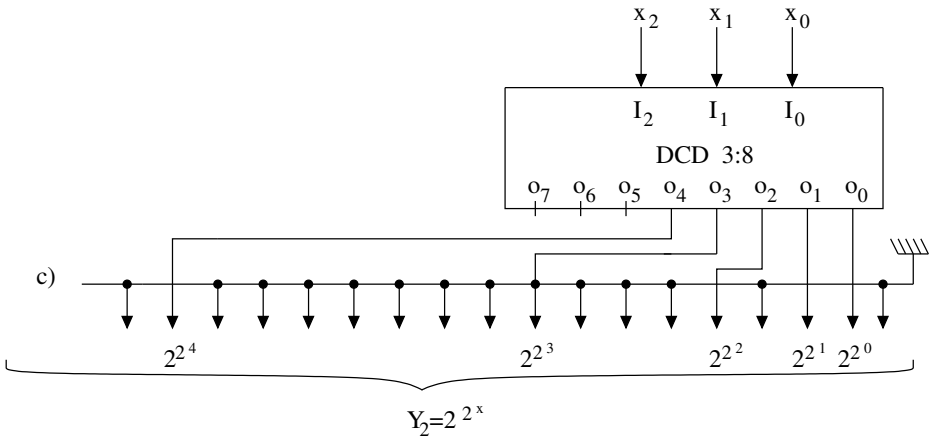


Figura P 2.31

tocmai funcţia de exponenţiere este realizată de decodificator daca se consideră cuvintele de intrare şi de ieşire ca numere binare.

Din tabelul din Figura P2.31-a se observă că pentru funcţia Y_2, ca şi pentru funcţia Y_1 pe intervalul de definiţie [000,111] se obţin tot opt valori dar cu un număr de zero-uri diferit pentru un acelaşi x şi de asemenea când se trece de la x la x+1 cuvântul Y_2 creşte cu mai mult de un zero. Pe baza acestei observaţii se poate implementa Y_2 în felul următor: cuvântul de ieşire Y_2 se obţine cu acelaşi circuit ca şi Y_1, dar se intercalează între biţii cuvântului Y_1 un număr de biţi zero conform valorilor calculate în tabel. Aceşti biţi zero, intercalaţi în cuvântul de ieşire, se obţin prin calblare la masă, Figura P2.31-c.

P2.32

Rezolvare:

Fiecare din operaţiile $2^{x_1}, 2^{x_2}$ poate fi implementată pe câte un circuit 74LS138. Impunând condiţia ca $x_1 \neq x_2$ niciodată ieşirile de acelaşi rang nu vor fi activate simultan şi de asemenea suma $2^{x_1} + 2^{x_2}$ nu va depăşi $2^8 - 1$, adică un număr care este un cuvânt pe opt biţi.

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

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

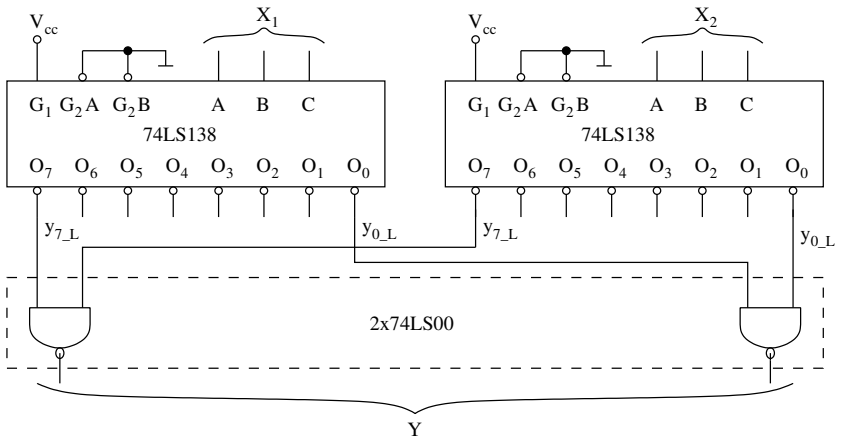


Figura P 2.32

P2.34

Rezolvare:

Circuitul DCD $n : 2^n$ produce la ieşire toţi cei 2^n mintermi de n variabile. Din punctul de vedere al implementării unei funcţii logice de n variabile dată sub forma FND (sumă de produse), circuitul DCD $n : 2^n$ poate fi un suport. Existând generaţi toţi cei 2^n mintermi, deci nivelul ŞI de implementare, mintermii necesari funcţiei sunt selectaţi în nivelul SAU care se adaugă în exterior. Deoarece, în general, la ieşirea DCD sunt generaţi mintermi negaţi, expresia funcţiei 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: $i = 0, 1 \ldots 7$

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

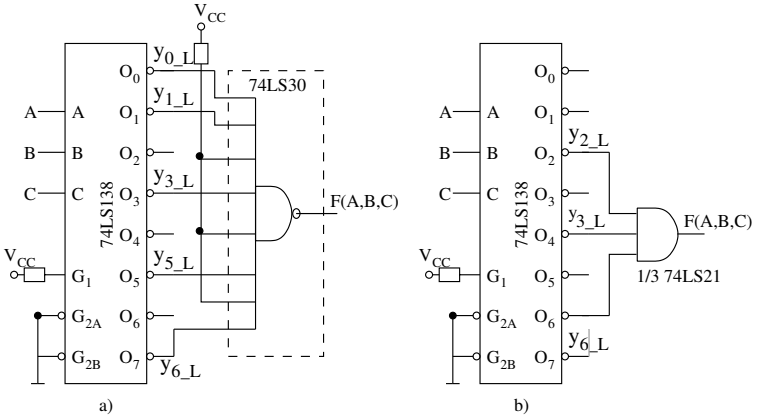


Figura P 2.34

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

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

$$\overline{f}(A,B,C) = P_2 + P_4 + P_6 = \overline{\overline{P_2 + P_4 + P_6}} = \overline{\overline{P_2} \cdot \overline{P_4} \cdot \overline{P_6}} \rightarrow$$

$$\rightarrow f(A,B,C) = \overline{P_2} \cdot \overline{P_4} \cdot \overline{P_6} = y_{2\_L} \cdot y_{4\_L} \cdot y_{6\_L}.$$

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

Implementarea cu DCD se poate realiza pornind de la forma FNC (produse de sume) a funcţiei. Aceeaşi funcţie de mai sus poate fi exprimată şi 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 $i = 1, 2 \ldots 7$ Rezultă o implementare cu o poartă AND cu trei intrări ca în Figura P2.34-b.

De asemenea, se poate face sinteza funcţiei negate f exprimate sub forma FNC:

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

$$f(A,B,C) = \overline{\overline{P_0} \cdot \overline{P_1} \cdot \overline{P_3} \cdot \overline{P_5} \cdot \overline{P_7}} = y_{0\_L} \cdot y_{1\_L} \cdot y_{3\_L} \cdot y_{5\_L} \cdot y_{7\_L}$$

Rezultă implementarea din Figura P2.34-a.

Pentru implementarea unui CLC cu ieşiri multiple pe un singur DCD unii mintermi pot fi utilizaţi de mai multe porţi 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.

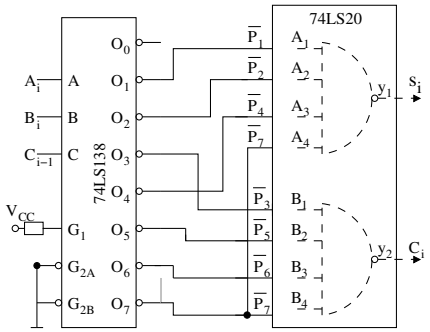


Figura P 2.36

P2.37

Rezolvare:

Pentru implementarea unei funcţii de n variabile cu un DCDn : 2^n funcţia nu necesită minimizare deoarece în nivelul ŞI sunt produşi toţi mintermii de n variabile; în exteriorul decodificatorului pe un nivel SAU trebuie colectaţi toţi mintermii existenţi în expresia funcţiei.

P2.38

Rezolvare:

Toţi mintermii acestor funcţii sunt produşi de un circuit 74XX138, DCD3:8 şi deoarece fiecare funcţie are numai doi mintermi sunt necesare patru porţi NAND cu două intrări care se găsesc în circuitul integrat 74XX00. Deci sunt necesare numai două circuite integrate, Figura P2.38.

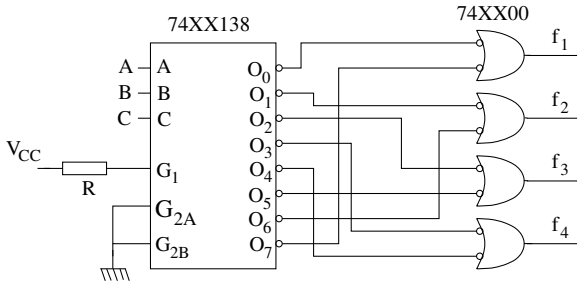


Figura P 2.38

P2.39

Rezolvare:

Glitch-urile (hazardul) pe ieşiri pot apare când comutaţia 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 obţin de la un numărător asincron; datorită întârzierilor din interiorul numărătorului asincron semnalul $X_1(2^1)$ va avea o întârziere faţa de comutarea semnalului $X_0(2^0)$, de asemenea $X_2(2^2)$ va avea o întârziere faţa 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 ieşire $y_{i\_L}, i = 0 \dots 7$ ale circuitului decodificator

- În punctul 1 la comutarea de la 001 la 010 poate apare starea trazitorie la intrare 000 (001 → 000 → 010) care va genera un glitch în semnalul y_{0_L}
- În punctul 2 la comutarea de la 011 la 100 pot apare stările trazitorii la intrare 010 şi 000 (011 → 010 → 010 → 100) care vor genera glitch-uri în y_{0_L} şi y_{2_L}
- În punctul 3 la comutarea de la 101 la 110 poate apare starea trazitorie la intrare 100 (101 → 100 → 110) care va genera un glitch în semnalul y_{4_L}
- În punctul 4 la comutarea de la 111 la 000 pot apare starile trazitorii 110 şi 100 (111 → 110 → 100 → 000) care vor genera glitch-uri în semnalul y_{6_L} şi y_{4_L}

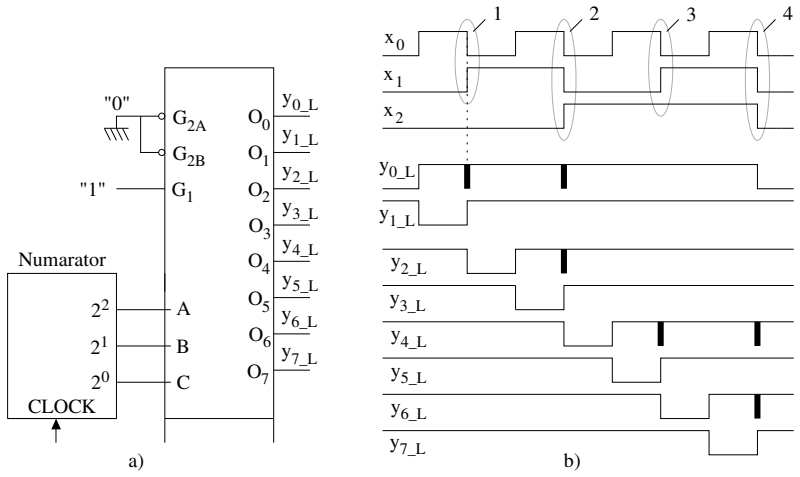


Figura P 2.39

O modalitate de a elimina glitch-urile în circuitele la care se aplică ieşirile de la decodificator, y_{i_L}, este strobarea, adică fiecare semnal din aceastea este intrarea unei porţi 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 ieşirile O_i, au acelaşi 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) şi (4) reuşind doar să extindă, eventual, durata valorii "1" la început respectiv la sfârşit. Pentru semanlul $C_i = P_3 + P_5 + P_6 + P_7$ 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 ieşirile decodificatorului se poate realiza fie prin utilizarea unui numărător 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 faţă de frontul de comutare al clock-ului la numărător astfel încât procesul tranzitoriu să se fi "consumat".

P2.41

Rezolvare:

O astfel de organizare, Figura P2.41, este posibilă deoarece circuitul 74XX138 conţine atât semnale de validare active în low, G_{2B_L} şi G_{2A_L} cât şi 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 şi pe intrarea G_1 al celuilalt, atunci ieşirile primului circuit sunt repartizate intervalului 7-0 iar ieşirile celui de al doilea intervalului 15-8.

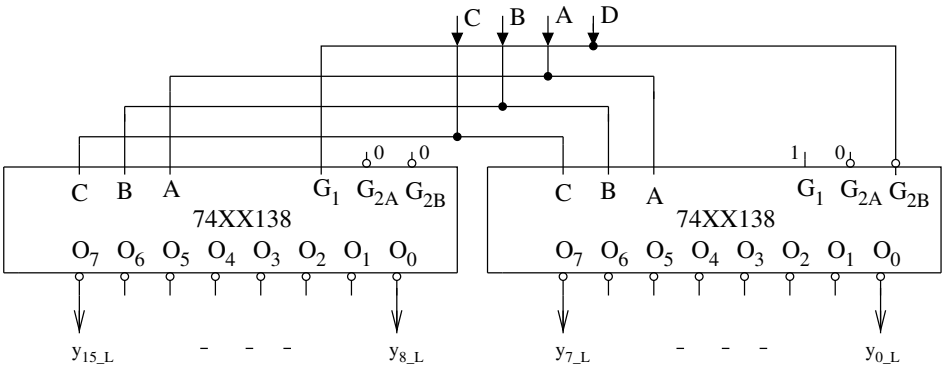


Figura P 2.41

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 împărţit în patru intervale $d_{31}-d_{24}$, $d_{23}-d_{16}$, $d_{15}-d_8$, d_7-d_0 şi fiecare interval de biţi de date se aplică unui circuit 74XX151. La toate aceste patru circuite MUX8:1 se aplică pentru selectare cei trei biţi mai puţini semnificativi x_2, x_1, x_0 ai cuvântului de selectare. Cei doi biţi mai semnificativi x_4 şi x_3 se utilizează pentru generarea semnalelor de selectare a datelor din unul din cele patru intervale. În varianta din Figura P2.43-a ieşirile circuitului decodificator 74XX139, aplicate pe intrările de validare al multiplexoarelor, selectează câte un multiplexor pentru fiecare interval de biţi de date. Ieşirile Y_{0_L} ale celor patru multiplexoare sunt colectate într-o poartă NAND, dacă se utilizează ieşirile nenegate este necesară o poartă OR.
În varianta din Figura P2.42 ieşirile celor patru multiplexoare 8:1 sunt selectate la ieşire cu un multiplexor 4:1.

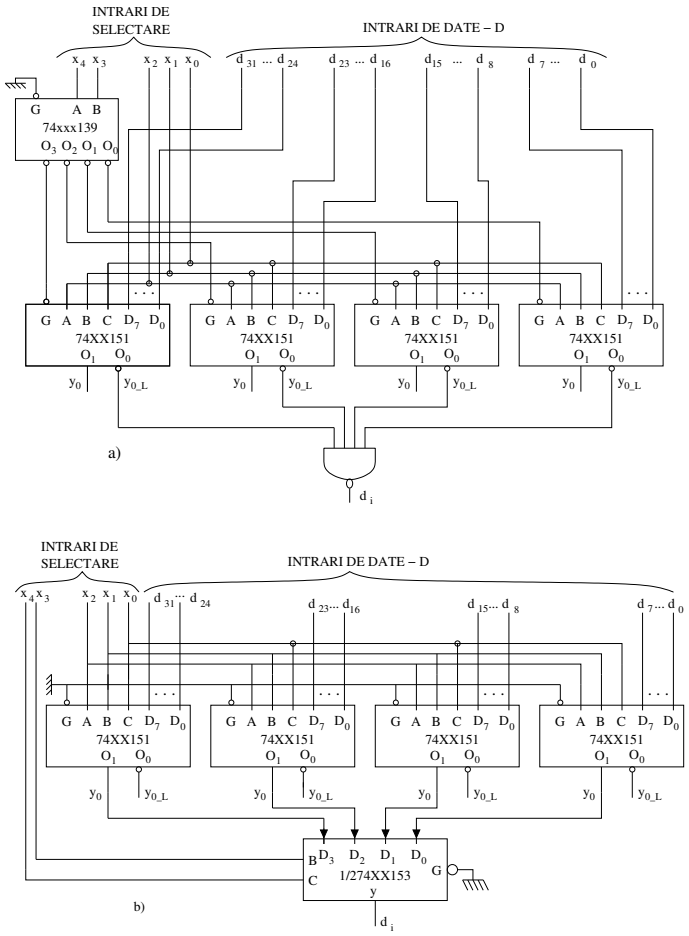


Figura P 2.43

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 ieşirea TSL, se comandă în stare normală când $G_L = 0$, Figura P2.44-a. Porturile P_3, P_2, P_1 şi P_0 se selectează respectiv cu următoarele valori ale variabilelor de selectare $x_1x_0(= 11, 10, 01, 00)$. Aceeaşi funcţie de selectare se poate realiza şi cu patru buffere TSL neinversoare de patru biţi 74XX253, Figura P2.44-b. Comanda (şi selectarea) unui buffer în stare normală de funcţionare se realizează cu semnalele de ieşire de la un DCD2:4, 74XX139.

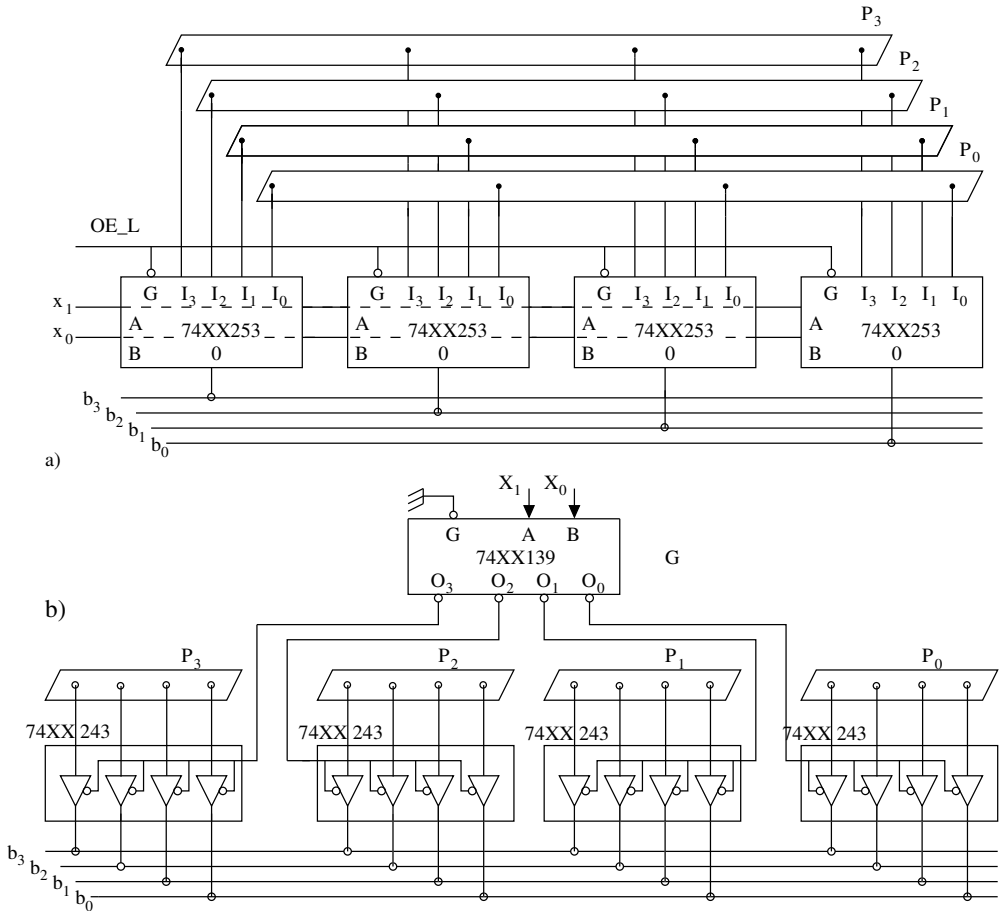


Figura P 2.44

P2.46

Rezolvare:

Implementarea funcţiei de n variabile pe un MUX $2^n : 1$ se realizează prin maparea directă (unu la unu) a tebelului 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 obţine ca un cuvânt în binar compus din alăturarea biţilor 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 şi pe următoarele niveluri, sunt avantajoase deoarece pentru calculul funcţiilor banale $(x_0, \overline{x_0}, 0, 1)$ multiplexoarele 2:1 pot fi eliminate.

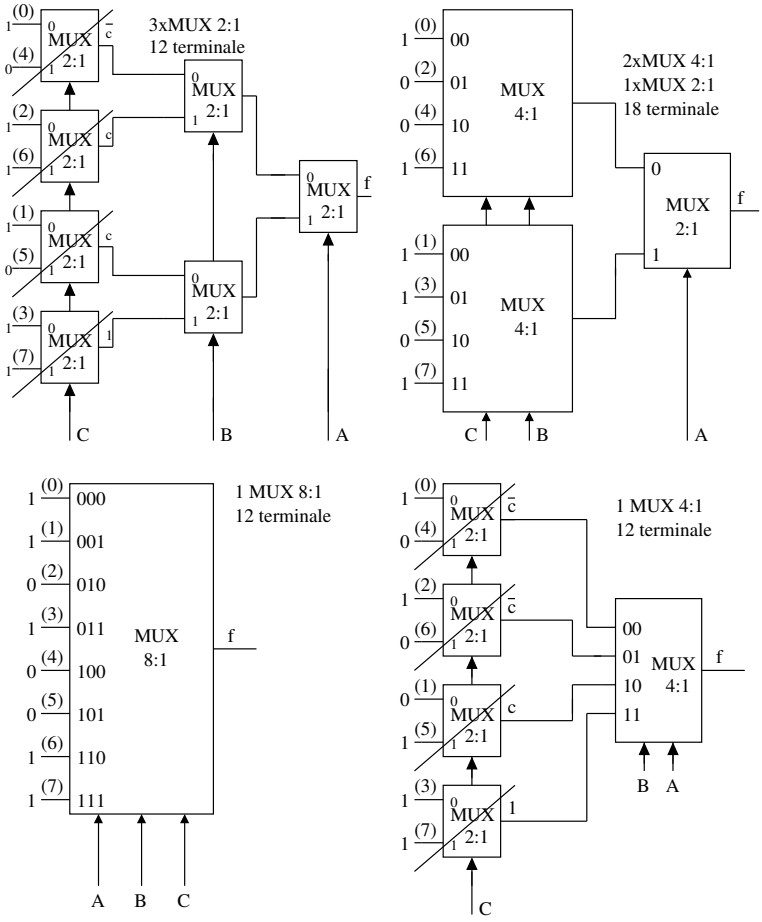


Figura P 2.46

P2.48 Fie funcţia $f(A,B,C,D) = \sum(3,6,11,14,15) + \sum d(2,5,12,13)$. 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 funcţia de patru variabile prin maparea directă pe intrări a tabelului de adevăr. În schimb, dacă una din variabile este introdusă în coeficienţii funcţiei, ca variabilă reziduu, rezultă o funcţie de trei variabile cu exprimările următoare:

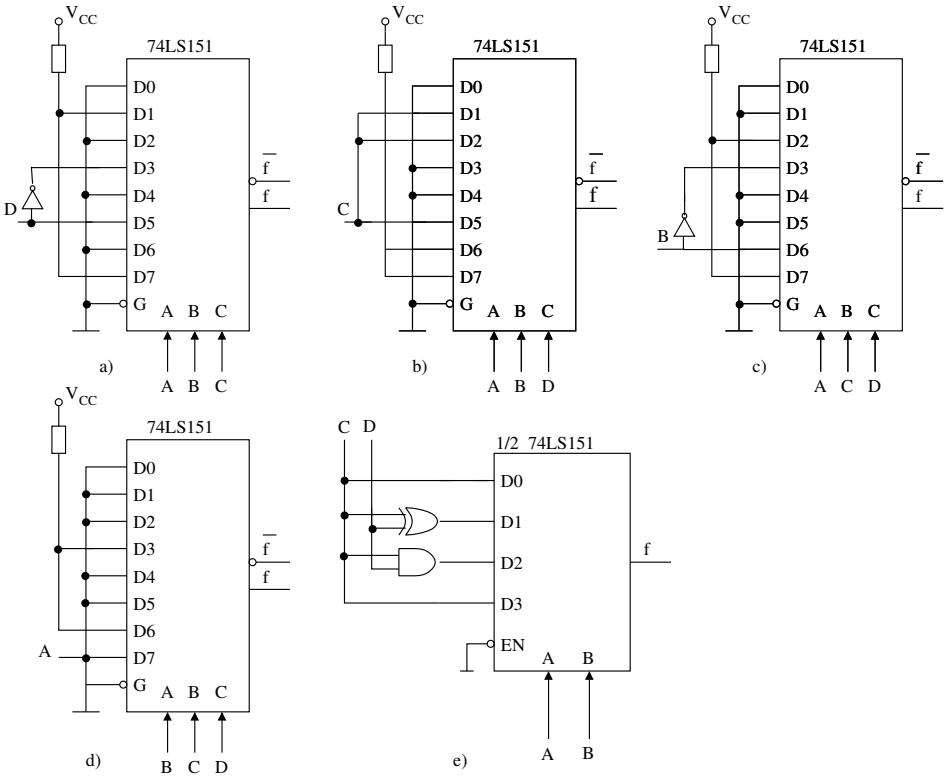


Figura P 2.48

$$
\begin{aligned}
f(D,A,B,C) &= 0\cdot\overline{A}\,\overline{B}\,\overline{C} + (D/0)\cdot\overline{A}\,\overline{B}C + (D/0)\cdot\overline{A}B\overline{C} + D\cdot\overline{A}BC + 0\cdot A\overline{B}\,\overline{C} + \\
&+ D\cdot A\overline{B}C + (0/1)\cdot AB\overline{C} + 1\cdot ABC \\
f(C,A,B,D) &= (0/C)\cdot\overline{A}\,\overline{B}\,\overline{D} + C\cdot\overline{A}\,\overline{B}D + C\cdot\overline{A}B\overline{D} + (0/C)\cdot\overline{A}BD + 0\cdot A\overline{B}\,\overline{D} + \\
&+ C\cdot A\overline{B}D + (C/1)\cdot AB\overline{D} + (C/1)\cdot ABD \\
f(B,A,C,D) &= 0\cdot\overline{A}\,\overline{C}\,\overline{D} + (0/B)\cdot\overline{A}\,\overline{C}D + (B/1)\cdot\overline{A}C\overline{D} + \overline{B}\cdot\overline{A}CD + \\
&+ (0/B)\cdot A\overline{C}\,\overline{D} + (0/B)\cdot A\overline{C}D + B\cdot AC\overline{D} + 1\cdot ACD \\
f(A,B,C,D) &= 0\cdot\overline{B}\,\overline{C}\,\overline{D} + 0\cdot\overline{B}\,\overline{C}D + (0/\overline{A})\cdot\overline{B}C\overline{D} + 1\cdot\overline{B}CD + (0/A)\cdot B\overline{C}\,\overline{D} + \\
&+ (0/1)\cdot B\overline{C}D + 1\cdot BC\overline{D} + A\cdot BCD
\end{aligned}
$$

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

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

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 porţi simple, pentru coeficinţii aplicaţi pe intrările multiplexorului.

P2.50

Rezolvare:

Funcţia fiind de cinci variabile, pentru implementare este necesară introducerea în expresia coeficienţilor a două variabile reziduu. Deoarece în expresia funcţiei se poate factoriza

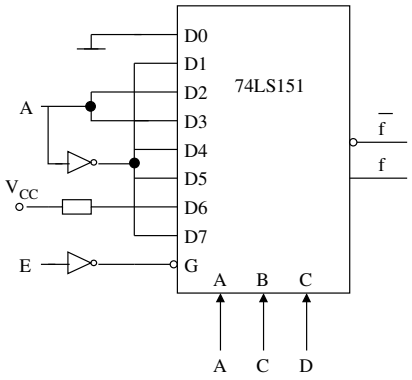


Figura P 2.50

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:

$$f(A,B,C,D,E) = E\{\overline{A}[B + \overline{C}D] + A[\overline{B}C + C\overline{D}]\} =$$

$$= E\{\overline{A}[B(C+\overline{C})(D+\overline{D}) + \overline{C}D(B+\overline{B})] + A[\overline{B}C(D+\overline{D}) + C\overline{D}(B+\overline{B})]\} =$$

$$= E\{\overline{A}[P_1 + P_4 + P_5 + P_7] + A[P_2 + P_3] + P_6 + 0\cdot P_0\}$$

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

P2.52

Rezolvare:

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

$$f(A,B,C,D,E) = \overline{A}BE + \overline{A}\,\overline{B}DE + A\overline{B}CE + AC\overline{D}E =$$

$$= \overline{A}BE(C+\overline{C})(D+\overline{D}) + \overline{A}\,\overline{B}DE(C+\overline{C}) + A\overline{B}CE(D+\overline{D}) + AC\overline{D}E(B+\overline{B}) =$$

$$= \overline{A}BECD + \overline{A}BE\overline{C}D + \overline{A}BEC\overline{D} + \overline{A}BE\overline{C}\,\overline{D} +$$

$$+\overline{A}\,\overline{B}ECD + \overline{A}\,\overline{B}E\overline{C}D + A\overline{B}ECD + ABEC\overline{D} + A\overline{B}EC\overline{D} =$$

$$= (\overline{A}BE + \overline{A}\,\overline{B}E + A\overline{B}E)\cdot CD + (\overline{A}BE + A\overline{B}E + ABE)\cdot C\overline{D} + (\overline{A}BE + \overline{A}\,\overline{B}E)\cdot \overline{C}D +$$

$$+(\overline{A}BE)\cdot \overline{C}\,\overline{D} =$$

$$= (P_3 + P_1 + P_5)\cdot CD + (P_3 + P_5 + P_7)\cdot C\overline{D} + (P_3 + P_1)\cdot \overline{C}D + P_3\cdot \overline{C}\,\overline{D}$$

Această expresie este implementată în Figura P2.52.

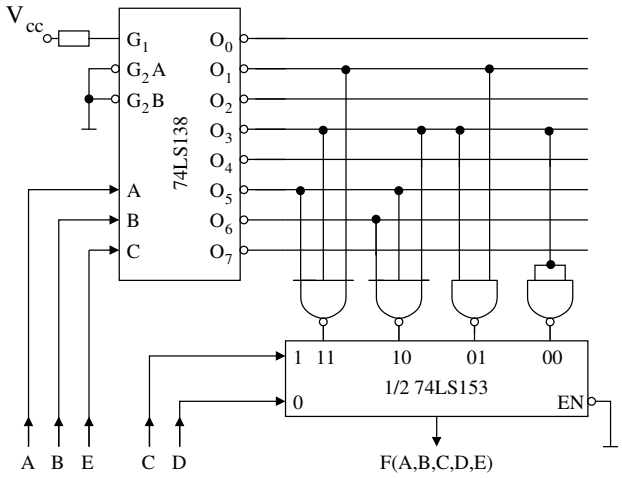


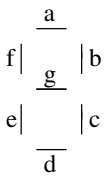
Figura P 2.52

P2.54

Rezolvare:
Uzual este circuitul convertor care pentru cele 10 cifre zecimale $(0, 1, \ldots, 8, 9)$ exprimate în BCD $(0000, 0001, \ldots, 1000, 1001)$ generează codul binar de 7 segmente pentru comanda afişoarelor 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 $2^7 \times 4 = 512$ biţi, cuvântul de adresă este $abcdefg$ iar cel de ieşire $WXYZ$. 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 capacităţii circuitului ROM se poate obţine în felul următor. Cu ultimii cinci biţi $cdefg$ se pot codifica $2^5 = 32$ configuraţii diferite, iar in tabelul din Figura P2.54-a există doar 7 configuraţii diferite cu aceşti cinci biţi care sunt prezentate în tabelul din Figura P2.54-b. Aceste 7 configuraţii diferite de cinci biţi sunt transcodate într-un cuvânt de cod intermediar de trei biţi A, B, C. Cu această transcodare intermediară plus configuraţiile formate cu cei doi biţi a,b din tabelul din Figura P2.54-a se formează un tabel de adevăr, Figura P2.54-c, care are 10 configuraţii pe intrare dar care nu mai sunt exprimate prin 7 biţi ci printr-un cuvânt de 5 biţi $ABCab$. Implementarea corespunzătoare, Figura P2.54-d, a acestui tabel de adevăr se realizează cu un circuit ROM2 de capacitate (32×3) biţi=96 biţi pentru transcodare şi un circuit ROM1 de capacitate (32×4) biţi=128 biţi, capacitatea memoriei a scăzut faţa de implementarea anterioară de la 512 la 224 biţi.
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 ieşirilor W, X, Y, Z (ieşirile pot fi exprimate şi 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 configuraţii de intrare distincte $(32 > 10)$. Eliminând variabilele b şi c, se observă că pentru oricare din configuraţiile de variabile $adefgh$ din tabel nu corespund două configuraţii diferite ale cuvântului $WXYZ$. De exemplu, dacă configuraţia de intrare $adefg = 11101$, căruia îi corespunde cuvântul de ieşire 0010 (linia a treia din tabelul din Figura P2.54-e), presupunem, că ar mai fi prezentă şi pe linia a cincea din tabel căruia îi corespunde cuvântul de ieşire 0100, atunci în cuvântul de intrare 11101 mai trebuie o variabilă prin care să se facă distincţia pentru cele două cuvinte de ieşire 0010 şi 0100, deci variabila eliminată nu este redundantă şi trebuie păstrată. Tabelul de adevăr în care s-au eliminat variabilele b şi c, care sunt redundante, este implementat, Figura P2.54-f, pe un circuit ROM de capacitate (32×4) biţi=128 biţi.
Se poate merge mai departe cu această reducere prin includerea variabilelor d şi g într-o funcţie logică AND, $H = g \cdot d$, astfel numărul variabilelor care se aplică la intrarea unui circuit ROM se reduce la patru a, e, f şi H, Figura P2.54-g. Implementarea corespunzătoare este prezentată în Figura P2.54-h şi necesită un circuit ROM de capacitate (16×4) biţi=64 biţi şi o poartă AND.

CIFRA ZECIMALA	INTRARI: Codul binar fragmente a	b	c	d	e	f	g	IESIRI W (2^3)	X (2^2)	Y (2^1)	Z (2^0)
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



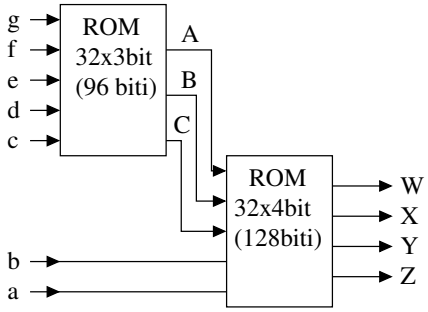
a)

SUBCUVANT de intrare A	B	C	a	b	Corespunde cifrei zecimale	Codul asigurat 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 A	B	C	a	b	IESIRI 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)

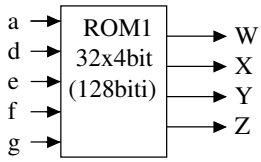


d)

Se continua pe pagina urmatoare

CIFRA ZECI–MALA	INTRARI a	d	e	f	g	IESIRI 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

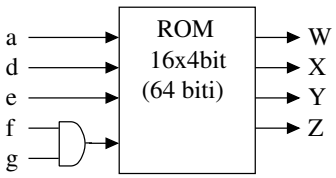
e)



f)

CIFRA ZECI–MALA	INTRARI a	e	f	H(dg)	IESIRI 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 $2^8 \times 4$ biţi=1kbiţi. Dacă circuitul ROM în stare neprogramată este "plin" cu zero-uri în toate locaţiile după progmamare sunt modificate numai locaţiile de la adresele indicate in tabelul din Figura P2.55. În cuvântul de ieşire $O_3O_2O_1O_0$, citit din ROM, bitul $O_3 = 1$ indică prezenţa unui singur bit egal cu zero în cuvântul de intrare/adresă iar biţii O_2, O_1, O_0 formează cuvântul care specifică, în binar, poziţia în cuvântul de intrare a singurului bit egal cu zero.

INTRARE										IESIRE			
in binar								hexa	zecimal				
x_7	x_6	x_5	x_4	x_3	x_2	x_1	x_0			O_3	O_2	O_1	O_0
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 biţi este redat în Figura P2.56-a în care se observă următoarele particularităţi $y_1 = 1$ şi $y_0 = A_0$. Pentru implementare rezultă structura de circuit din Figura P2.56-b, pe baza unui circuit ROM de capacitate 8×4 biţi=32 biţi.

INTRARI				IESIRI						
$N\|_{10}$	$N\|_2$			$(N\|_{10})^2$						$(N\|_{10})^2$
	A_2	A_1	A_0	y_5	y_4	y_3	y_2	y_1	y_0	
0	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	1	1
2	0	1	0	0	0	0	1	0	0	4
3	0	1	1	0	0	1	0	0	1	9
4	1	0	0	0	1	0	0	0	0	16
5	1	0	1	0	1	1	0	0	1	25
6	1	1	0	1	0	0	1	0	0	36
7	1	1	1	1	1	0	0	0	1	49

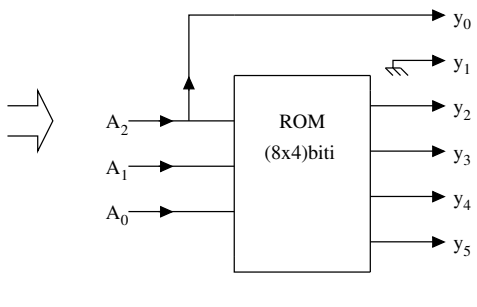


Figura P 2.56

P2.57

Rezolvare:

Transformând adresele spaţiului de adresare din hexazecimal în binar, pentru o decodificare completă se obţin pentru biţii $A_9 \ldots A_0$ valorile prezentate în Figura P2.57-a.

Se observă că, pentru toate cele patru adrese biţii $A_9 \ldots A_2$ rămân nemodificaţi în timp ce A_1A_0 se modifică în funcţie de perifericul adresat. Aceasta sugerează soluţia utilizării circuitului 74LS139, DCD2:4, la care se aplică biţii A_1A_2 şi ale cărui ieşiri selectează perifericele #0,#1#2#3. Coincidenţa valorilor biţilor de adresă A_9A_2, cu valorile 11110000,

trebuie detectată şi aplicată ca un semnal de validare (G) la circuitul 74LS139. Cu ajutorul circuitului 74LS138, DCD3:8, se poate obţine activarea ieşirii O_0 când există coincidenţa biţilor $A_9 \dots A_2$ cu următorul cuvânt 11110000, Figura P2.57-b.
O altă variantă de implementare pentru această coincidenţă este reprezentată în figura P2.57-c utilizând circuitul 74LS682 (circuit comparator de opt biţi cu două ieşiri P=Q şi P>Q, active în L).
Se poate realiza şi o decodificare imcompletă când nu sunt utilizaţi toţi biţi. 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 coincidenţei, 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:

$$\#0 \rightarrow \text{1C0H şi 3C0H;}$$
$$\#1 \rightarrow \text{1C1H şi 3C1H;}$$
$$\#2 \rightarrow \text{1C2H şi 3C2H;}$$
$$\#3 \rightarrow \text{1C3H şi 3C3H;}$$

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

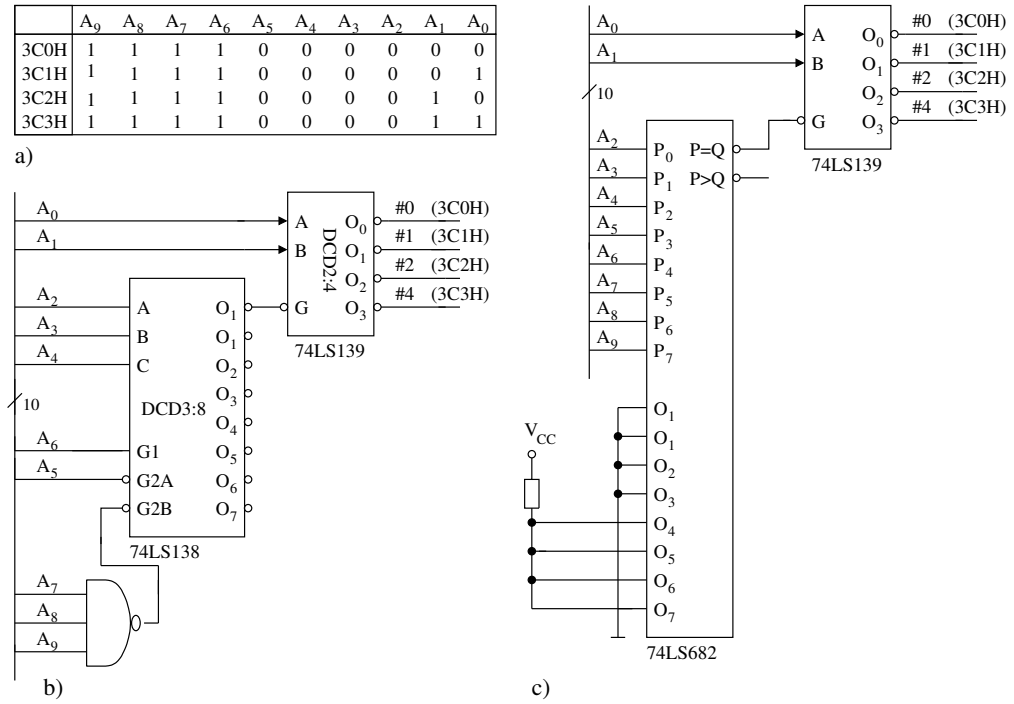


Figura P 2.57

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

P2.59

Rezolvare:

Pentru a forma modulul de $8K \times 16$ biţi este necesară o extensie la 16 biţi pe ieşire ceea ce inseamnă că sunt necesare câte două circuite 2716 conectate în paralel pentru a obţine

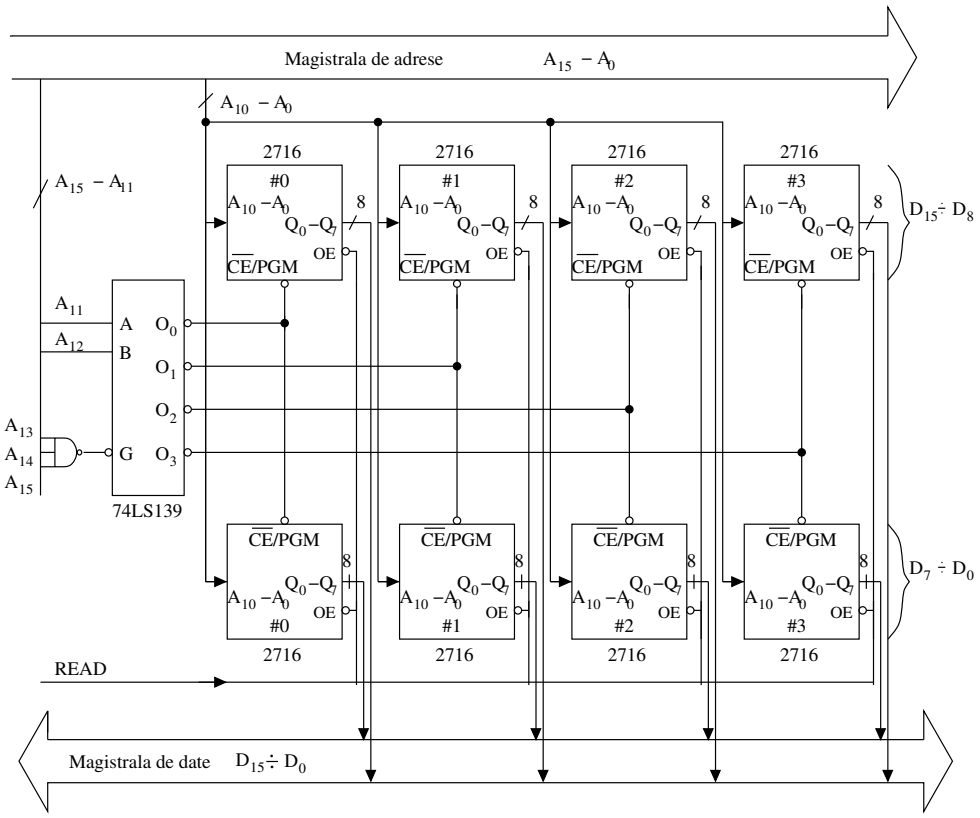


Figura P 2.59

cuvântul de date $D_{15} \ldots D_8, D_7 \ldots D_0$ şi o extensie pe intrare, Figura P2.59. Extensia pe intrare constă în plasarea pe fiecare subinterval de adresă de $2K$ (din cei $8K$ acoperiţi de modul) a câte unui grup de două circuite 2716 ($D_{15} \ldots D_8$ şi $D_7 \ldots D_0$). Deoarece decodificatorul intern al fiecărui circuit decodifică numai 11 ($A_{10} \ldots A_0$) biţi este necesară extensia decodificatorului în exterior pentru biţii 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 spaţiul de $64K$ este necesar ca cei trei biţi cei mai semnificativi din cuvântul de adresă să fie $A_{15}A_{14}A_{13} = 111$. Prin conjuncţia acestor trei biţi se generează semnalul de validare G al decodificatrului 2:4. Astfel, intervalul de adresare asignat modulului este $E000K \ldots FFFFH$ 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_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0		
					1	1	1	1	1	1	1	1	1	1	1		FFFF H
1	1	1	1	1	–	–	–	–	–	–	–	–	–	–	–	#3	– – – –
					0	0	0	0	0	0	0	0	0	0	0		F800 H
					1	1	1	1	1	1	1	1	1	1	1		F7FF H
1	1	1	1	0	–	–	–	–	–	–	–	–	–	–	–	#2	– – – –
					0	0	0	0	0	0	0	0	0	0	0		F000 H
					1	1	1	1	1	1	1	1	1	1	1		EFFF H
1	1	1	0	1	–	–	–	–	–	–	–	–	–	–	–	#1	– – – –
					0	0	0	0	0	0	0	0	0	0	0		E800 H
					1	1	1	1	1	1	1	1	1	1	1		E700 H
1	1	1	0	0	–	–	–	–	–	–	–	–	–	–	–	#0	– – – –
					0	0	0	0	0	0	0	0	0	0	0		E000 H

P2.60

Rezolvare:

Implemetarea funcţiei 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 funcţiei). Această simplitate apare datorită faptului că ROM-ul produce, pe nivelul de decodificare AND, toţi 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 funcţiei. Implementarea apare mult mai intuitivă dacă funcţia este dată într-o diagramă V-K ce poate fi mapată direct pe o structură matriceală de memorie ca în Figurile P2.60-a şi b.

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

Această descompunere în funcţii componente nu este de nici o utilitate pentru o singură funcţie, dar este foarte utilă când se doreşte implementarea mai multor funcţii diferite, de aceleaşi variabile, pe o aceeaşi structură ROM. Suprafaţa ocupată de fiecare funcţie pe diagrama V-K este disjunctă faţă de celelalte suprafeţe. Funcţia de 5 variabile $f_1(E, D, C, B, A)$ este înscrisă în intervalul de adrese care se selectează cu valoarea variabilei $F = 1$, funcţia de 4 variabile $f_2(D, C, B, A)$ se selectează cu valorile variabilelor $F = 0$ şi $E = 1$; iar funcţiile de 3 variabile $f_3(C, B, A)$ şi $f_4(C, B, A)$ sunt selectate respectiv prin $FED = 001, FED = 000$, Figura P2.60-d. Implementarea mai multor funcţii diferite, de aceleaşi variabile, pe aceeaşi structură ROM, pe intervale disjuncte de adrese, duce la o mai bună utilizare a capacităţii circuitului dar, evident, funcţiile implementate nu pot fi selectate simultan!

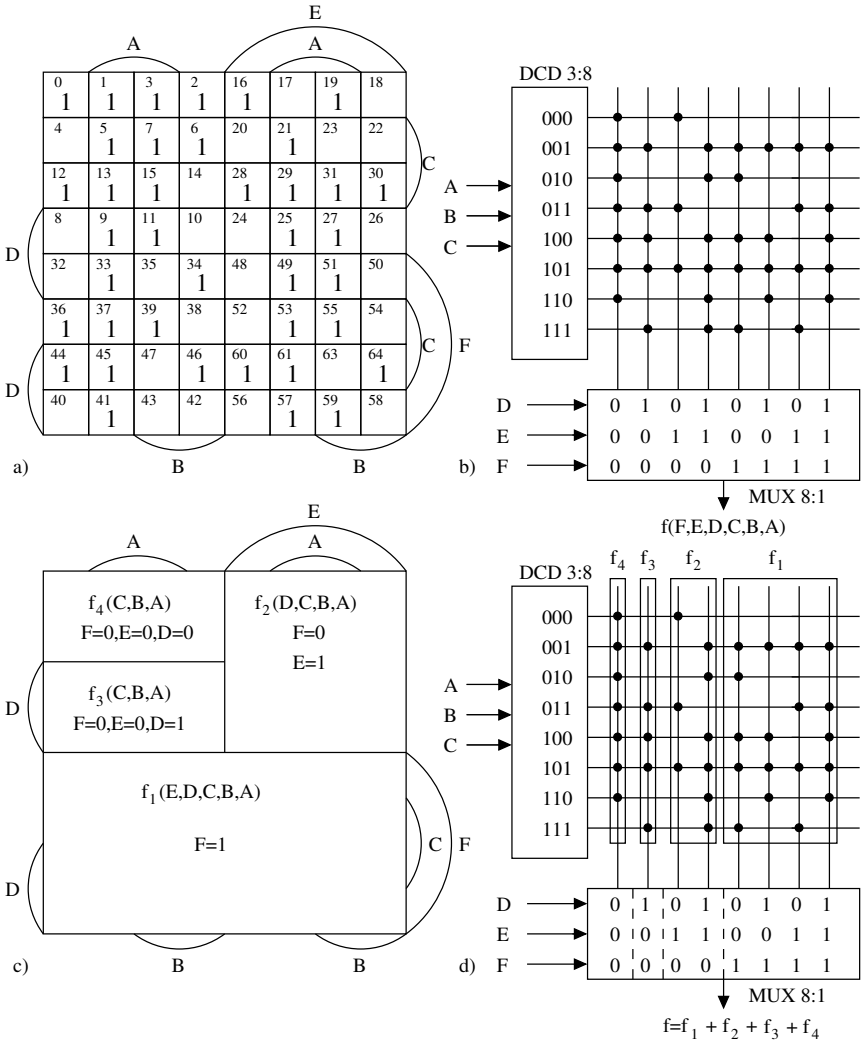


Figura P 2.60

P2.61

Rezolvare:

Se reprezintă pe diagrama V-K şi 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 biţii de ieşire W, X, Y, Z se obţin cu diagramele V-K din Figura P2.64-b. Formele minime sunt implementate pe o structură generică de PLA în Figura

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

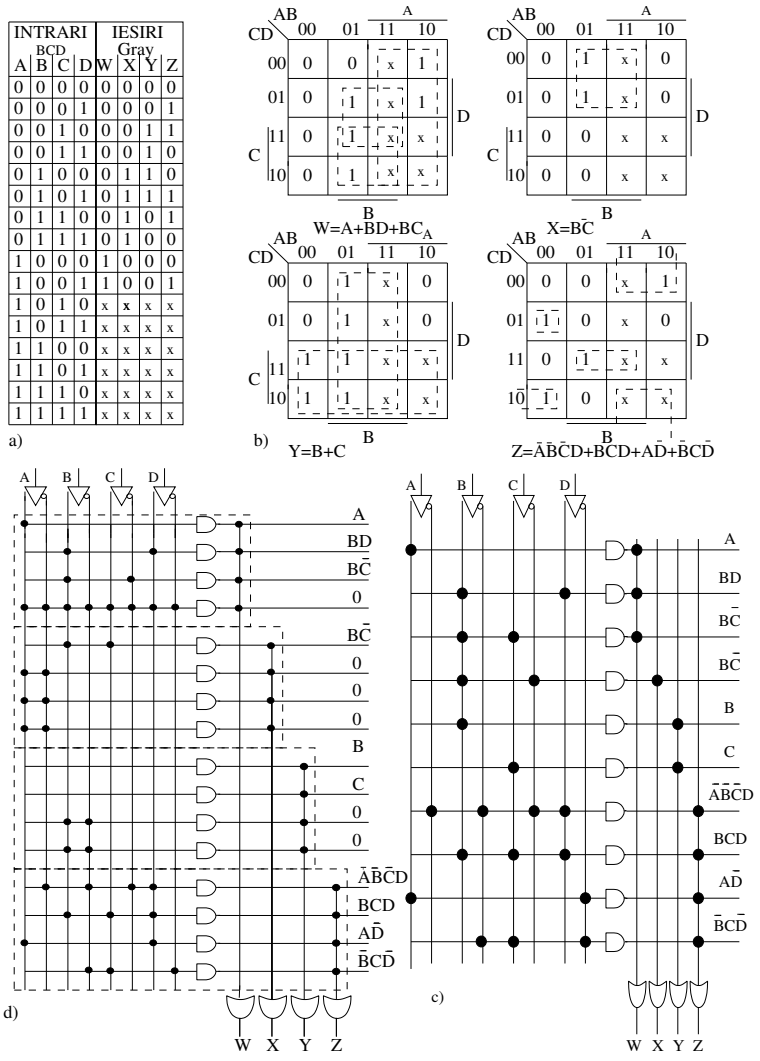


Figura P 2.64

P2.67

Rezolvare

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

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

Pentru operatorul SAU-EXCLUSIV-NEGAT se poate utiliza circuitul 74XX266 care conţine patru porţi XNOR cu colectorul în gol care sunt conectate într-o conexiune SI cablat, Figura P2.67-d. Dacă se compară două cuvinte de patru biţi ($m = 4$) şi la ieşirea porţilor cu colectorul în gol se comandă o poartă TTL standard ($N = 1$) se poate scrie pentru determinarea intervalului de valori al rezistenţei R următoarele relaţii (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 rezistenţei R până se ajunge la condiţia limită $R_{max} = R_{min}$ (datorită faptului că va creşte componenta de curent $m \cdot I_{OH}$ absorbită în starea HZ). Introducând această condiţie de limită în relaţia anterioară, pentru calculul rezistenţei R_{max}, se obţine lungimea maximă m_{max} a cuvintelor care pot fi comparate (se consideră $M_H = 0,4V$)

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

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

- Cu ajutorul a două $DCD_An : 2^n$ şi $DCD_Bn : 2^n$ se poate determina identitatea a două cuvinte. Cele două ieşiri de acelaşi rang, O_{Ai} şi O_{Bi}, se colectează în câte o poartă AND2, cuvintele de comparat se aplică pe intrările decodificatoarelor. Poarta AND2 cu ieşirea activă indică şi configuraţiile de biţi aplicate pe intrări.
- Se poate implementa un circuit de identitate pentru cuvinte de n biţi şi pe un EPROM cu $2^{2n} \times 1$ biţi. Numai 2^n locaţii sunt utilizate din cele 2^{2n}.
- Imaginaţi şi alte variante de circuite.

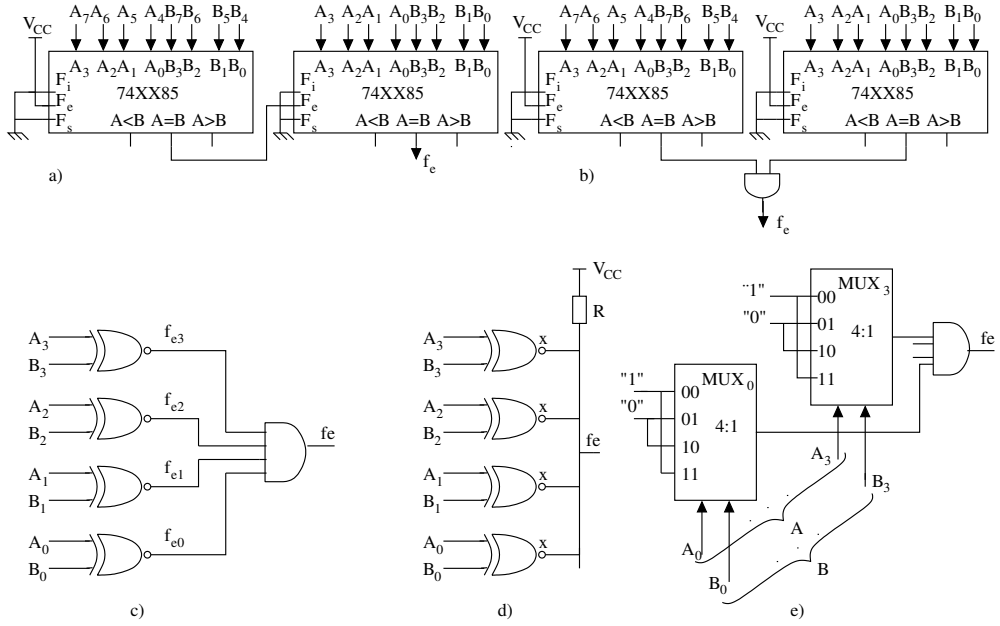


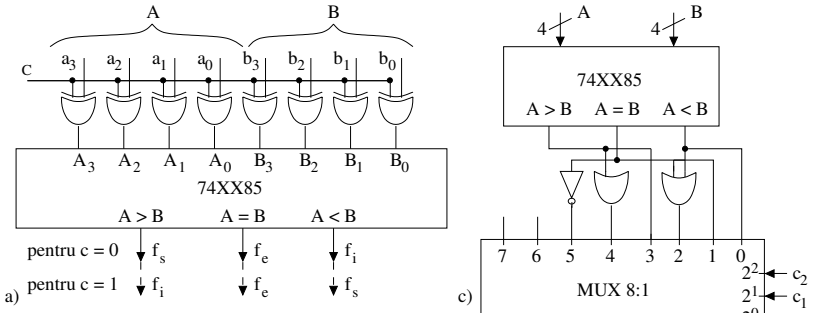
Figura P 2.67

P2.68

Rezolvare:

a) Structura de circuit din Figura P2.68-a generează la ieşirile circuitului 74XX85 funcţiile conform funcţionării normale ale acestuia pentru c=0 iar când c=1 ieşirile circuitului sunt conform cu cerinţele acestei probleme (se poate demonstra această funcţionare din analiza valorilor conţinute în tabelul din Figura P2.68-b.
O altă soluţie este obţinută prin conectarea pe intrarea A a cuvântului B iar pe intrarea B a cuvântului A.

b) Relaţiile de ordine se obţin cu porţile care procesează semnalele de la ieşirea circuitului 74XX85. Selectarea unei relaţii de ordine se face cu un MUX8:1, Figura P2.68-c cu ajutorul cuvântului $c_2c_1c_0$.



b)

INTRARI		Comanda c = 0			Comanda c = 1		
a_i	b_i	$A_i=a_i$	$B_i=b_i$	f	$A_i=\bar{a}_i$	$B_i=\bar{b}_i$	f
0	0	0	0	f_e	1	1	f_e
0	1	0	1	f_i	1	0	f_s
1	0	1	0	f_s	0	1	f_i
1	1	1	1	f_e	0	0	f_e

Figura P 2.68

P2.72

Rezolvare

Eliminarea unor ranguri (care pot fi alăturate sau nealăturate la variantele de doi biţi) 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 biţi din Figura P2.72-a pentru că $A_3 = 0, B_3 = 0$, transportul C_2 nu se va obţine la ieşirea C_3 ci la ieşirea S_3.

Varianta de sumator de doi biţi din Figura P2.72-b foloseşte poziţia A_0, B_0 pentru rangul 2^0 şi poziţia A_3, B_3 pentru rangul 2^3, poziţiile A_1, B_1 şi A_2, B_2 nu sunt utilizate deci trebuie realizate propagările $p_1 = 1$ şi $p_2 = 1$ prin aplicarea $A_2A_1 = 00$ şi $B_2B_1 = 11$. Biţii de sumă s_1, s_0 se obţin la S_3 şi S_0 iar transportul C_1 la C_3

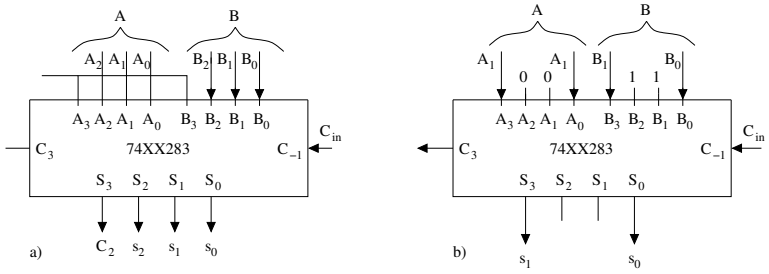


Figura P 2.72

P2.73

Rezolvare:

Celula semisumator în bază trei este reprezentată în Figura P2.73-a, are două intrări: cei doi operanzi $A_i = a_1a_0, B_i = b_1b_0$ şi două ieşiri: suma $s_i = s_1s_0$ şi transportul următor C_i (nu se condideră transportul anterior C_{i-1}). Tabelul operaţiei 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 şi tinând cont de configuraţiile operanzilor (11) care nu pot apare pe intrări rezultă circuitul implementat cu porţi NAND din Figura P2.73-d.

P2.74

Rezolvare:

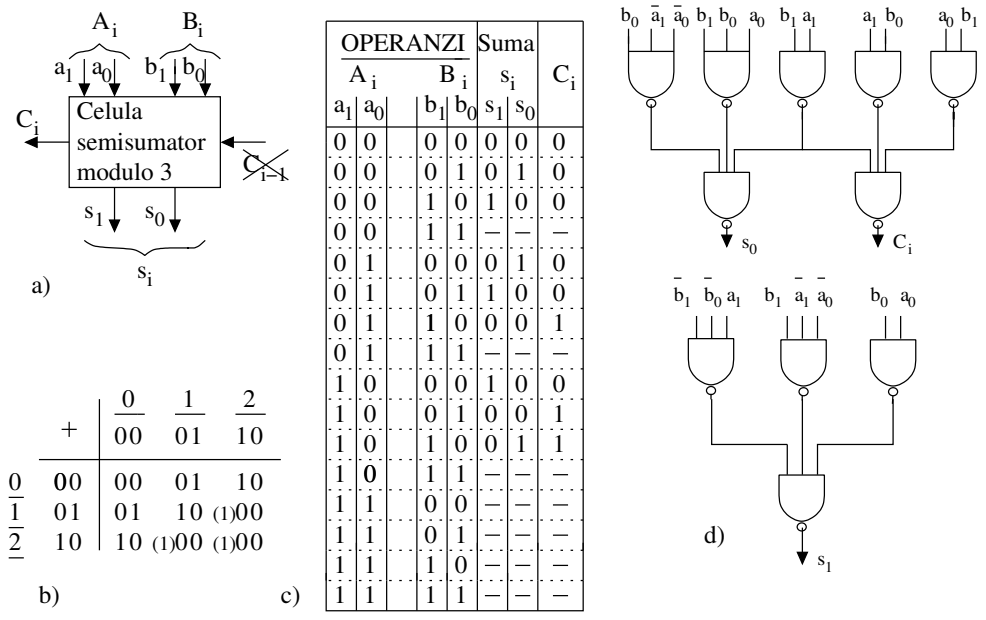
Celula sumator modulo 3 se obţine din două semisumatoare modulo 3 ca în Figura P2.74-a. Pe primul sumator se sumează cei doi operanzi generându-se transportul C_{i1} şi o sumă. Această sumă este adunată, pe un al doilea sumator, cu cuvântul $0C_{i-1}$ care se aplică ca operandul $B_i(= 0C_{i-1})$, se obţine valoarea sumei s_i şi un transport C_{i2}. Transportul C_i se obţine ca o funcţie logică OR între C_{i1} şi C_{i2}. Un sumator modulo 3^8 cu transport progresiv se obţine prin punerea în paralel a 8 celule sumator ale căror circuite de transfer al transportului se înseriază 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 $A_i, B_i(i = 0, 1, \ldots 7)$ şi a transportului iniţial C_{-1} este egal cu: 6 niveluri logice pe rangul 3^0 şi 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, $A_i = A_{i3}A_{i2}A_{i1}A_{i0}B_i = B_{i3}B_{i2}B_{i1}B_{i0}$ şi transportul anterior C_{i-1} poate genera un cuvânt $C_is_{i3}s_{i2}s_{i1}s_{i0}$ care reprezintă un număr în binar natural în intervalul [0,19]. Efectuarea adunării BCD pe un sumator binar de patru biţi generează un cod corect în BCD numai dacă cuvântul sumă reprezintă în binar un număr în intervalul [0,9], şi respectiv un cuvânt de cod eronat dacă rezultă un număr binar în intervalul [10,19]. Codul corectat în BCD, şi pentru intervalul [10,19], se obţine dacă la suma obţinută 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 corecţie 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 biţii A_{i0} şi B_{i0} se sumează pe o celulă $\sum(3, 2)$. Iar pentru rangurile $2^1, 2^2$ şi 2^3 se realizează o sinteză de subcircuit BCD, care este descris de un tabel de adevăr cu şapte intrări $A_{i3}, B_{i3}, A_{i2}, B_{i2}, A_{i1}, B_{i1}$ şi C_{i0} şi generează patru ieşiri s_{i3}, s_{i2}, s_{i1} şi C_i (complicat!).

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



OPERANZI A_i			OPERANZI B_i		Suma s_i		C_i
a_1	a_0		b_1	b_0	s_1	s_0	
0	0		0	0	0	0	0
0	0		0	1	0	1	0
0	0		1	0	1	0	0
0	0		1	1	–	–	–
0	1		0	0	0	1	0
0	1		0	1	1	0	0
0	1		1	0	0	0	1
0	1		1	1	–	–	–
1	0		0	0	1	0	0
1	0		0	1	0	0	1
1	0		1	0	0	1	1
1	0		1	1	–	–	–
1	1		0	0	–	–	–
1	1		0	1	–	–	–
1	1		1	0	–	–	–
1	1		1	1	–	–	–

Figura P 2.73

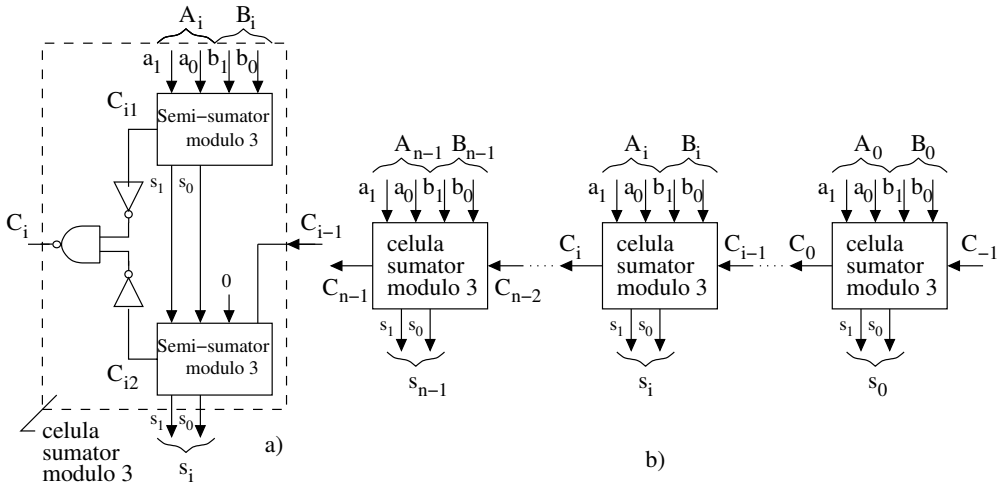


Figura P 2.74

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

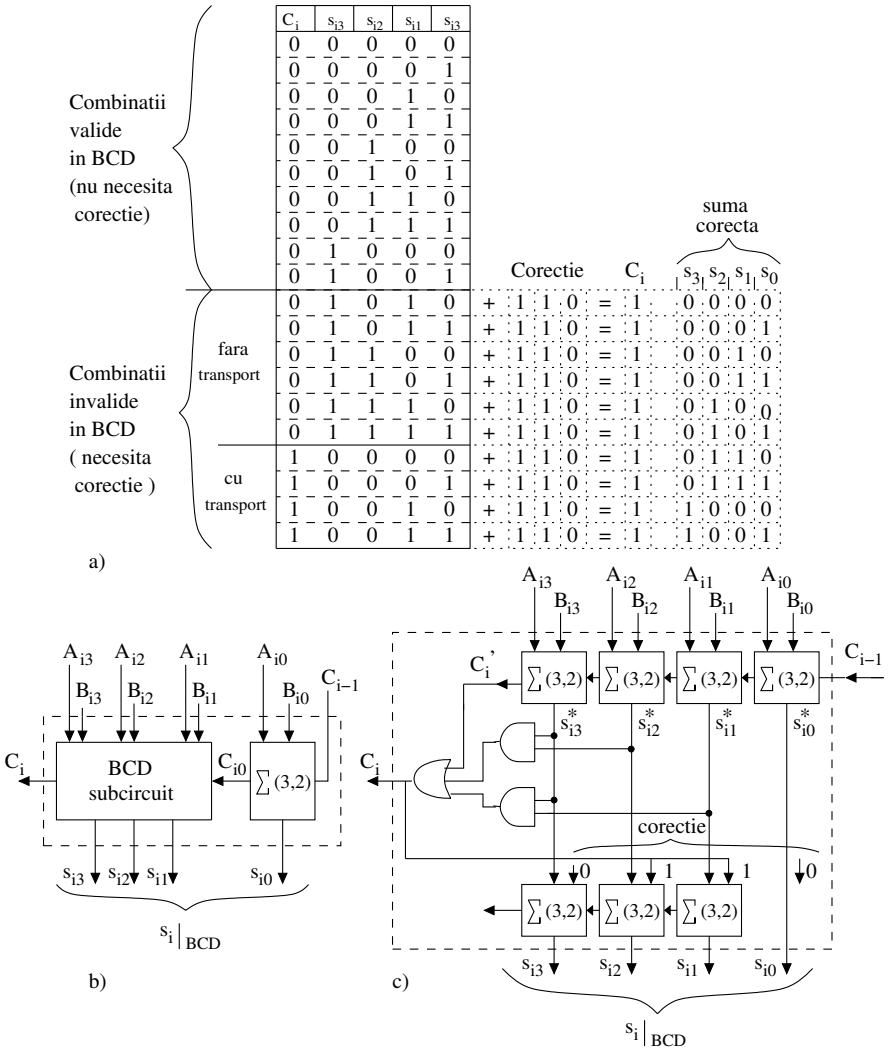


Figura P 2.75

P2.76

Rezolvare:

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

această structură poate fi o unitate logico-aritmetică pentru cei doi bytes mai puţin semnificativi, adică pentru grupul de biţi $15 \div 0$ din lungimea de 64 de biţi. La fel se realizează o astfel de structură pentru fiecare din celelalte trei grupuri $63 \div 48, 47 \div 32$ şi $31 \div 16$. Considerând fiecare grup ca o unitate logico-aritmetică de patru biţi care generează semnalele P şi 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 $16 \times 74LS181$ şi $5 \times 74LS182$. În figură sunt prezentate numai semnalele de control, nu şi biţii cuvintelor de intrare şi de ieşire.

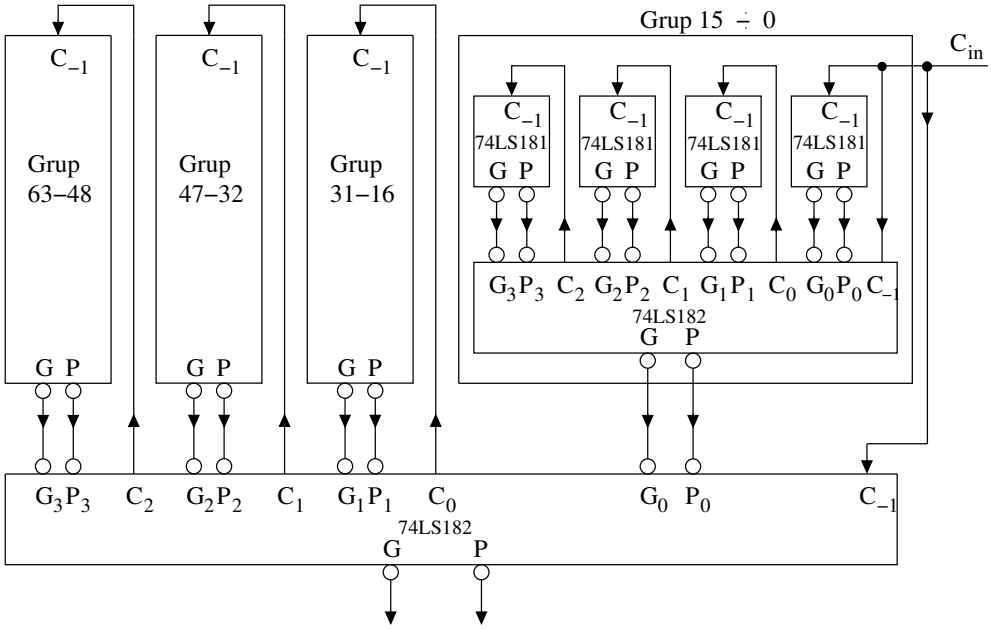


Figura P 2.76

P2.77

Rezolvare:

Dacă la ieşirea 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 biţi 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 $\sum(3,2)$ obţinându-se la ieşire $C_i s_i$ care reprezintă numărul de biţi 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 biţilor care au valoarea 1) poate fi exprimat în binar cu 4 biţi (de exemplu $8|_2 = 1000$, pentru un şir de opt biţi 1) adunarea se efectuează întâi pe un numărător cu două celule şi apoi pe unul cu trei celule, Figura P2.77-b.

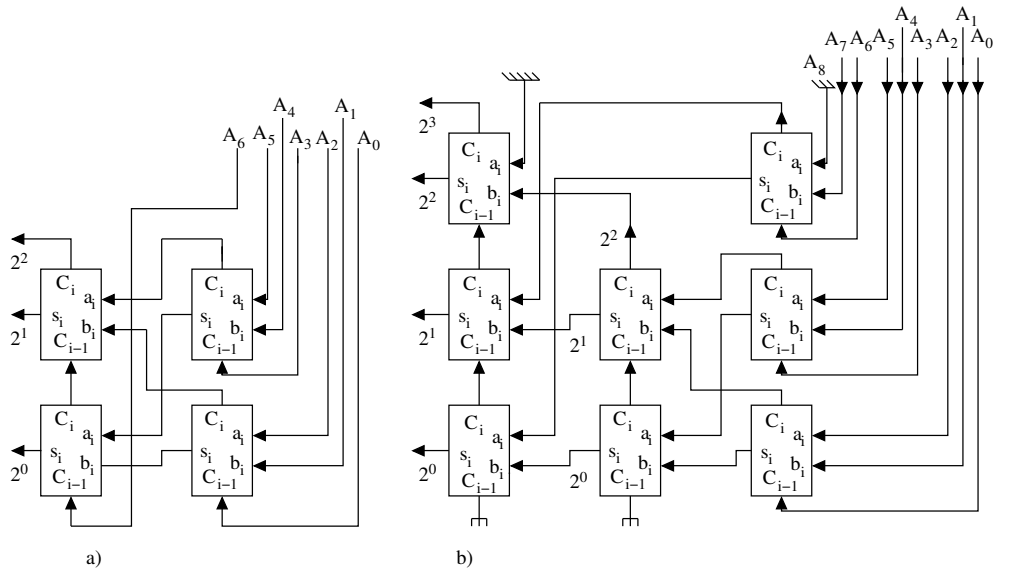


Figura P 2.77

P2.78

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

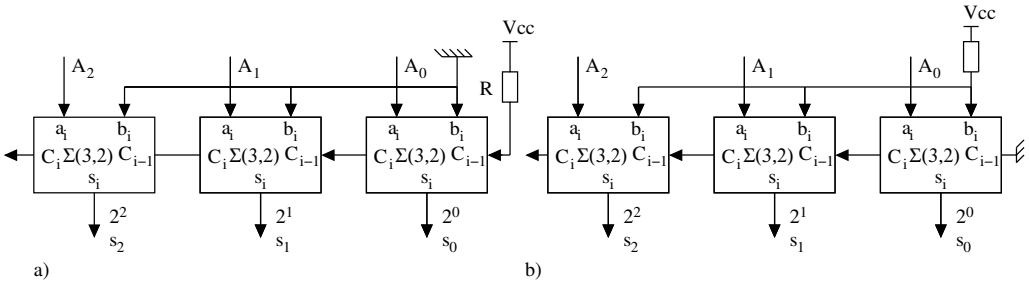


Figura P 2.78

P2.79

Rezolvare:

1. $C_5 = A_5 \cdot B_5 + C_4 \cdot A_5 + C_4 \cdot B_5$;
   $C_5 = g_5 + p_5 \cdot g_4 + p_5 \cdot p_4 \cdot g_3 + p_5 \cdot p_4 \cdot p_3 \cdot g_2 + p_5 \cdot p_4 \cdot p_3 \cdot p_2 \cdot g_1 + p_5 \cdot p_4 \cdot p_3 \cdot p_2 \cdot p_1 \cdot g_0$

2. Pentru SPT, C_5 se calculează cu 4 porţi, iar pentru STA, C_5 se calculează cu 6 porţi (vezi expresia anterioară) plus două porţi care calculează p_i şi $p_i (i = 0, 1, 2, 3, 4, 5)$ dar care sunt utilizate de mai multe ranguri de sumare.

3. SPT: $3\tau_p$ pentru rangul 2^0, $4 \times 2\tau_p$ pentru rangurile $2^1, 2^2, 2^3, 2^4$ şi τ_p pentru rangul 2^5, deci $T_{\sum} = 12\tau_p$.
   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 $6\tau_p$. Creşterea de viteză la STA faţa 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 biţi; 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 biţi realizând suma $X + Y$ pe un sumator cu salvarea transportului, realizat pe două celule sumator complet $\sum(3, 2)$, după structurarea din Figura P2.80.

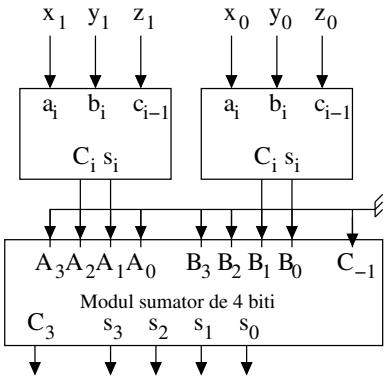


Figura P 2.80

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.

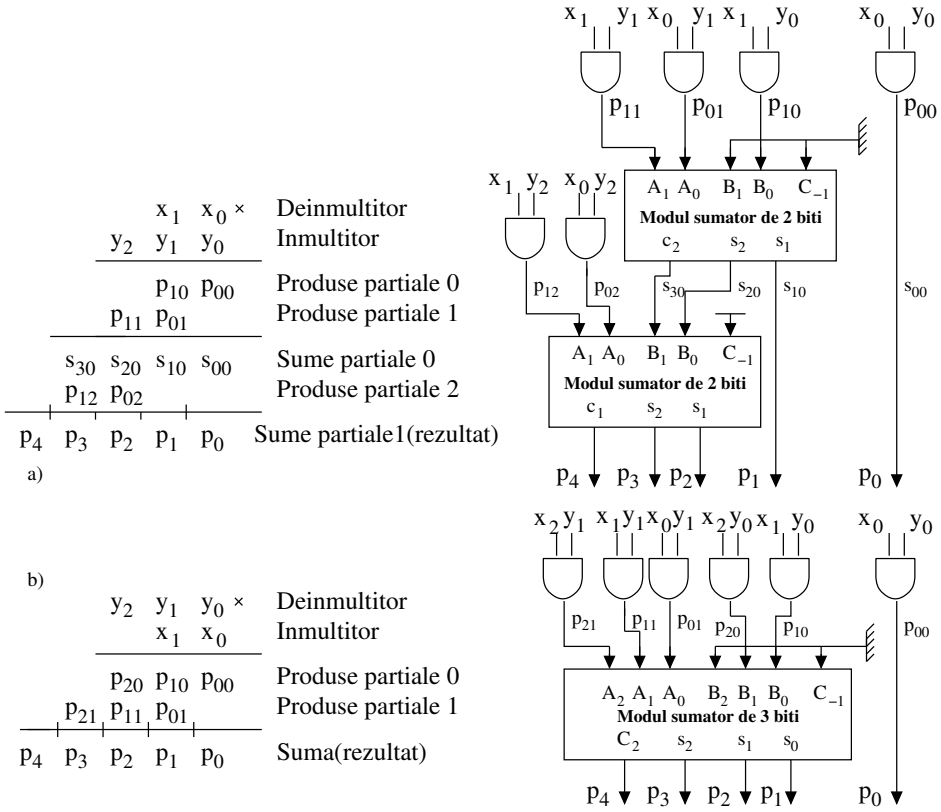


Figura P 2.81

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 operaţii sunt logice: pentru $S_1S_0 = 00 \rightarrow F = \overline{B}$; pentru $S_1S_0 = 01$ $A > B$, care se poate scrie sub forma $F = \overline{B}A$, deci pot fi implementate cu porţi logice. Celelalte două operaţii sunt aritmetice şi pot fi implementate pe două celule sumator complet $\sum(3, 2)$. Celula de scăzător complet se poate obţine 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 operaţii sunt realizate pe circuite separate, Figura P2.82-b iar apoi sunt selectate cu două MUX4:1. Operaţiile logice ar necesita doar un singur MUX4:1, deoarece produce doar bitul de funcţie F, pe când operaţiile aritmetice necesită două MUX4:1, deoarece prin aceste operaţii aritmetice se generează atât bitul de funcţie cât şi bitul de transport următor C_i.

Selectare Operatie s_1 s_0	Operatia Aritmetica/Logica F_i, C_i
0 0	trece B_i
0 1	$A_i > B_i$
1 0	$A_i + B_i + C_i$
1 1	$A_i - B_i - C_{i-1}$

a)

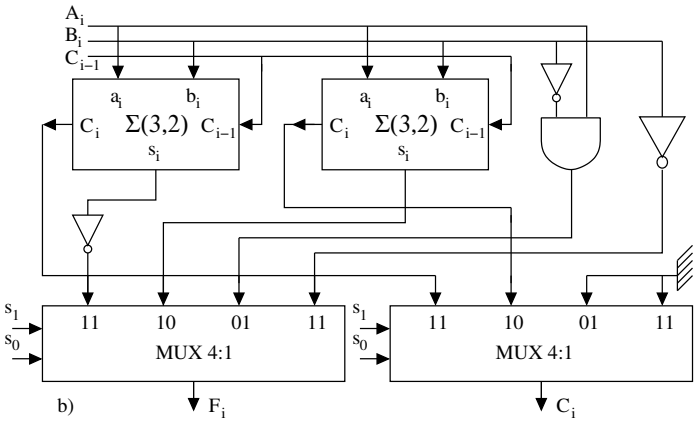


Figura P 2.82

P2.83

Rezolvare:

Modulul ALU de patru biţi este prezentat în Figura P2.83.

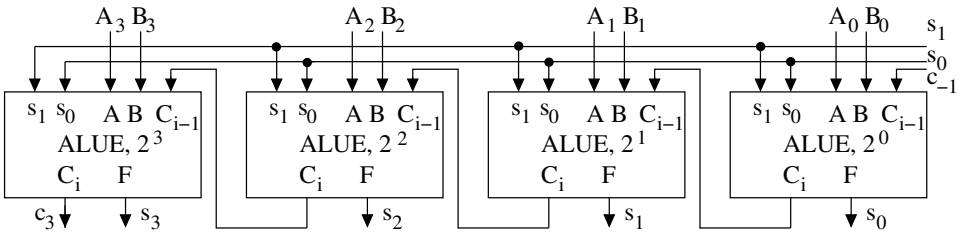


Figura P 2.83

P2.84

Rezolvare:

Fiecare operaţie, în afară de înscrie ieşirea în 0 sau în 1, este realizată pe un circuit separat. Pentru operaţiile logice se utilizează porţile AND, OR şi XOR iar pentru cele aritmetice trei celule sumator complet $\sum(3,2)$. Selectarea se realizează prin două MUX8:1 deoarece la ieşire trebuie generat atât funcţia F cât şi transportul următor C_i (numai pentru operaţiile aritmetice). Selectarea operaţiilor aritmetice se efectuează prin semnalul de selectare $S_2 = 1$ iar cele logice prin $S_2 = 0$, Figura P2.84-a. Structurarea ALUE este prezentată în Figura P2.84-b, se observă că operaţiile logice nu utilizează multiplexorul pentru transport.

Selectare Operatie			Operatia Aritmetica/Logica
s_2	s_1	s_0	F , C_i
0	0	0	inscrie 0
0	0	1	$B_i - A_i$
0	1	0	$A_i - B_i$
0	1	1	$A_i + B_i$
1	0	0	$A_i \oplus B_i$
1	0	1	$A_i \cup B_i$
1	1	0	$A_i \cap B_i$
1	1	1	inscrie 1

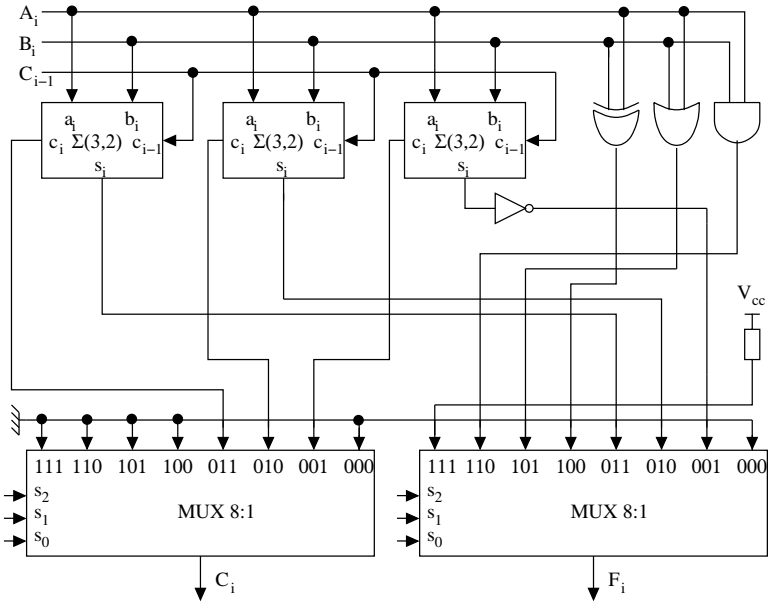


Figura P 2.84