Dan NICULA

ELECTRONICĂ DIGITALĂ
Carte de învăţătură 2.0

Editura Universităţii *TRANSILVANIA* din Braşov
ISBN 978-606-19-0563-8

2015

Lecţia 7

Sinteza circuitelor logice combinaţionale

7.1 Noţiuni teoretice

Orice circuit logic combinaţional poate fi implementat cu porţi NAND pe două nivele logice.

Metodologia de implementare constă în parcurgerea următoarelor etape:

- Se reprezintă funcţia în diagramă V-K.
- Se minimizează funcţia sub formă de sumă de produse (SOP).
- Fiecare produs (AND) se implementează cu o poartă NAND, cu un număr de intrări egal cu numărul de termeni în produs. La intrarea porţilor se aplică variabilele de intrare, negate sau ne-negate. Aceste porţi reprezintă primul nivel de porţi NAND.
- Suma (OR) se implementează cu o poartă NAND cu un număr de intrări egal cu numărul de produse din sumă. La intrările porţii se aplică ieşirile porţilor NAND ale primului nivel logic. Această poartă reprezintă al doilea nivel de porţi NAND.

Ca exemplu, funcţia $F(A,B,C) = \sum(1,3,6,7)$ se minimizează sub forma $F(A,B,C) = A \cdot B + \overline{A} \cdot C$. Implementarea directă (cu porţi AND şi OR) şi transformarea într-o structură pe două nivele de porţi NAND sunt prezentate în figura 7.1.

Justificarea analitică este:
$F(A,B,C) = A \cdot B + \overline{A} \cdot C = \overline{\overline{A \cdot B + \overline{A} \cdot C}} = \overline{\overline{A \cdot B} \cdot \overline{\overline{A} \cdot C}}$

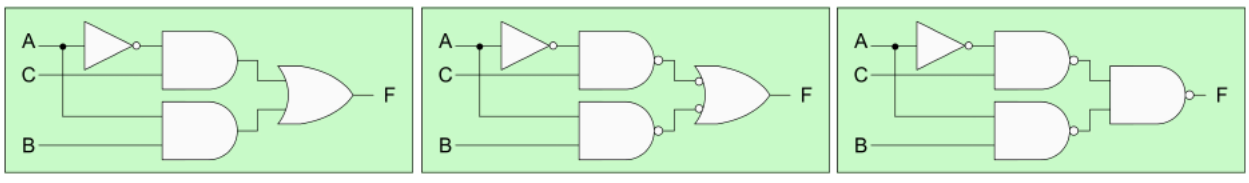

Figura 7.1 Funcţie logică: implementare AND-OR, implementare NAND-NAND.

7.2 Pentru cei ce vor doar să promoveze examenul

1. Minimizaţi funcţiile logice şi implementaţi-le cu porţi NAND pe două nivele logice.

a) $F_a = \sum(3,5,6,7)$
b) $F_b = \sum(0,2,4,6,7)$
c) $F_c = \sum(0,4,6,7,8,14,15)$
d) $F_d = \sum(3,7,9,10,11,15)$
e) $F_e = \sum(0,2,6,8,10,13)$

2. Proiectaţi un circuit de "vot majoritar" cu 3 intrări şi implementaţi-l cu porţi NAND pe două nivele, după minimizarea funcţiei. Ieşirea circuitului are valoarea logică a majorităţii intrărilor.

Soluţie

Tabelul de adevăr al circuitului "vot majoritar" este:

V_2	V_1	V_0	VOT
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Diagrama V-K asociată funcţiei este prezentată în figura 7.2, alături de implementarea cu porţi NAND a funcţiei VOT.

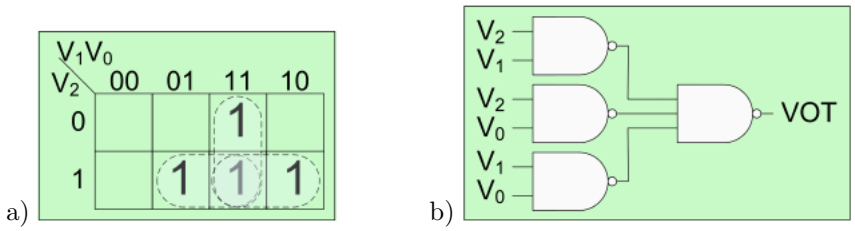

Figura 7.2 Funcţia "vot majoritar": **a)** diagrama V-K, **b)** implementare cu porţi NAND.

Din tabelul de adevăr se determină diagrama V-K şi se minimizează funcţia obţinându-se expresia:
$VOT = V_2 \cdot V_1 + V_2 \cdot V_0 + V_1 \cdot V_0$

Se observă că dacă două din cele trei variabile sunt egale cu 1, există o conjuncţie a două intrări egală cu 1, ceea ce determină $VOT = 1$.

7.3 Pentru cei ce vor să înveţe

1. Minimizaţi funcţiile logice şi implementaţi-le cu porţi NAND pe două nivele logice.

a) $F_a = \sum(0,2,4,7)$
b) $F_b = \sum(1,3,6,7)$
c) $F_c = \sum(1,2,4,7)$
d) $F_d = \sum(1,2,3,4,5)$
e) $F_e = \sum(0,2,5,8,10,13,14)$
f) $F_f = \sum(4,5,6,7,10,11)$
g) $F_g = \sum(2,5,7,9,11,13,15)$
h) $F_h = \sum(4,6,9,11,12,14)$
i) $F_i = \sum(0,3,5,6,9,10,12,15)$

Soluţie

a) Diagrama V-K asociată funcţiei $F_a = \sum(0,2,4,7)$ este prezentată n figura 7.3-a). Forma minimă a funcţiei

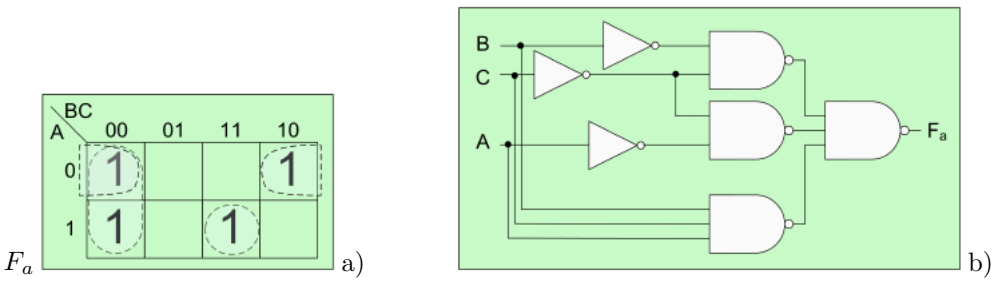

Figura 7.3 F_a, problema 1-a: a) diagrama V-K, b) implementare cu porţi NAND.

este:

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

Implementarea cu porţi NAND este prezentată în figura 7.3-b).

Costul implementării este:
2 porţi $NAND \times 2$
2 porţi $NAND \times 3$
3 porţi $NOT \times 1$
Total: 7 porţi cu un număr total de 13 intrări.

c) Pentru funcţia $F_c = \sum(1, 2, 4, 7)$ diagrama V-K (figura 7.4-a) este "în formă de tablă de şah", forma minimă fiind FCND:

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

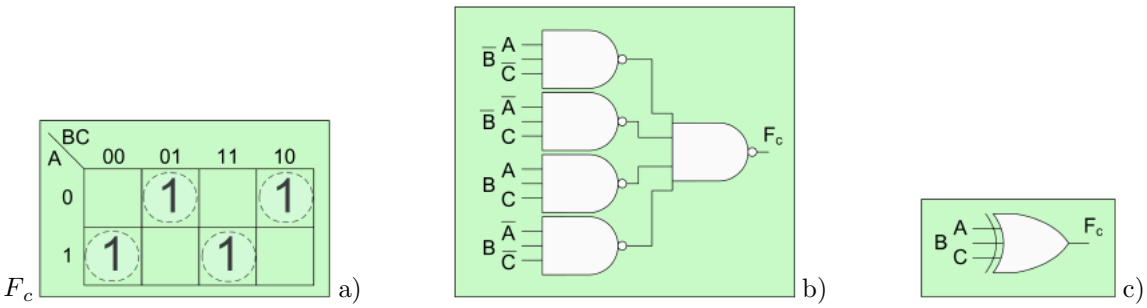

Figura 7.4 F_c, problema 1-c: a) diagrama V-K, b) implementare cu porţi NAND, c) implementare cu porţi XOR.

Se observă că F_c poate fi implementată c o singură poartă XOR de 3 intrări:

$F_c = A \oplus B \oplus C$

e) Diagrama V-K asociată funcţiei $F_e = \sum(0, 2, 5, 8, 10, 13, 14)$ este prezentată n figura 7.5-a).

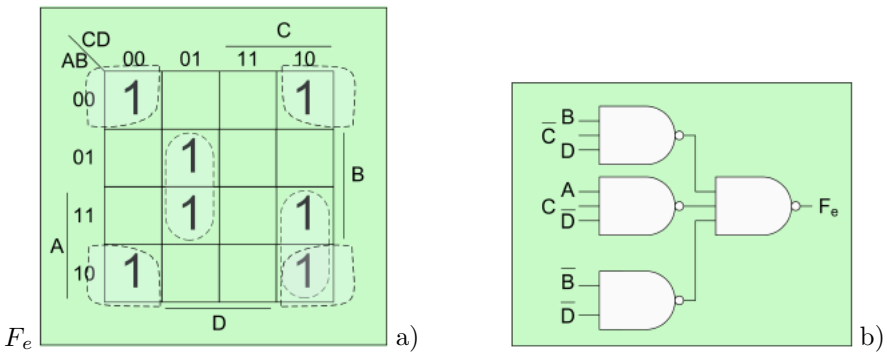

Figura 7.5 F_e, problema 1-e: a) diagrama V-K, b) implementare cu porţi NAND.

Forma minimă a funcţiei este:

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

Implementarea cu porţi NAND este prezentată în figura 7.5-b).

Costul implementării este:
1 poartă $NAND \times 2$
3 porţi $NAND \times 3$
3 porţi $NOT \times 1$
Total: 7 porţi cu un număr total de 14 intrări.

i) Se observă că diagrama V-K asociată funcţiei $F_i = \sum(0, 3, 5, 6, 9, 10, 12, 15)$ este "în formă de tablă de şah" şi se poate implementa cu o poartă XOR sub forma:

$F_i = \overline{A \oplus B \oplus C \oplus D}$

2. Minimizaţi funcţiile logice incomplet definite şi implementaţi-le cu porţi NAND. Evaluaţi "costul" implementărilor. Propuneţi soluţii de eliminare a hazardului combinaţional şi evaluaţi costurile suplimentare ale acestora.

a) $F_a = \sum(1, 3, 5, 9, 14, 15) + d(4, 6, 12, 13)$
b) $F_b = \sum(0, 3, 4, 5, 6, 7, 9, 12, 14, 15) + d(10)$
c) $F_c = \sum(0, 1, 5, 7, 9, 10, 14) + d(3, 8)$
d) $F_d = \sum(2, 6, 7, 9, 10) + d(0, 8)$
e) $F_e = \sum(5, 6, 7, 8, 9, 10, 13, 14, 15) + d(0, 11)$
f) $F_f = \sum(1, 2, 3, 5, 7, 10, 13, 14, 15) + d(6, 8)$

Soluţie

a) Diagrama V-K asociată funcţiei F_a este prezentată în figura 7.6. Funcţia minimizată cu diagrama V-K este:

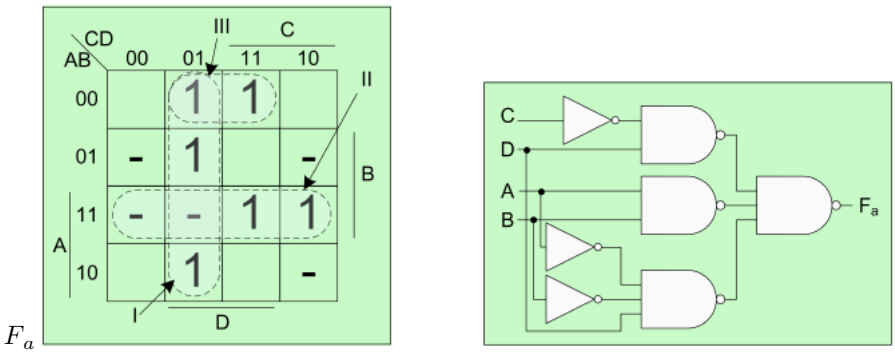

Figura 7.6 F_a, problema 26-a: diagrama V-K şi implementare cu porţi NAND.

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

Pentru a obţine expresia funcţiei doar cu porţi NAND se aplică Teorema lui De Morgan:
$F_a = \overline{\overline{C} \cdot D + A \cdot B + \overline{A} \cdot \overline{B} \cdot D} = \overline{\overline{C} \cdot D} \cdot \overline{A \cdot B} \cdot \overline{\overline{A} \cdot \overline{B} \cdot D}$

Costul implementării este:
2 porţi $NAND \times 2$
2 porţi $NAND \times 3$
3 porţi $NOT \times 1$
Total: 7 porţi cu un număr total de 13 intrări.

Din diagrama V-K se observă că nu există suprafeţe disjuncte adiacente. Deci, funcţia nu are hazard combinaţional.

d) Diagrama V-K asociată funcţiei F_d este prezentată în figura 7.7.

Funcţia minimizată cu diagrama V-K este:
$F_d = I + II + III = \overline{B} \cdot \overline{D} + A \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot C$

Pentru a obţine expresia funcţiei doar cu porţi NAND se aplică Teorema lui DeMorgan:
$F_d = \overline{\overline{B} \cdot \overline{D} + A \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot C} = \overline{\overline{B} \cdot \overline{D}} \cdot \overline{A \cdot \overline{B} \cdot \overline{C}} \cdot \overline{\overline{A} \cdot B \cdot C}$

Costul implementării este:
1 poartă $NAND \times 2$
3 porţi $NAND \times 3$

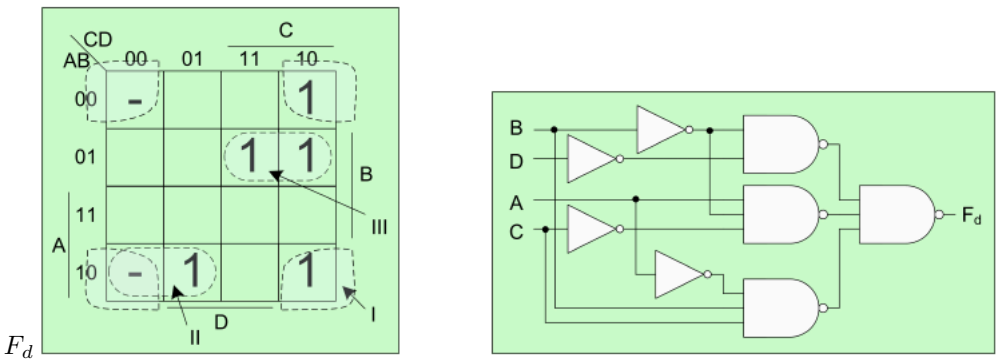

Figura 7.7 F_d, problema 26-d: diagrama V-K şi implementare cu porţi NAND.

4 porţi $NOT \times 1$
Total: 8 porţi cu un număr total de 15 intrări.

Din diagrama V-K se observă că suprafeţele I şi II sunt suprapuse, însă suprafeţele I şi III sunt disjuncte dar adiacente. Rezultă că funcţia poate prezenta hazard combinaţional. Eliminarea hazardului combinaţional se poate face prin adăugarea unei suprafeţe IV prezentată în figura 7.8.

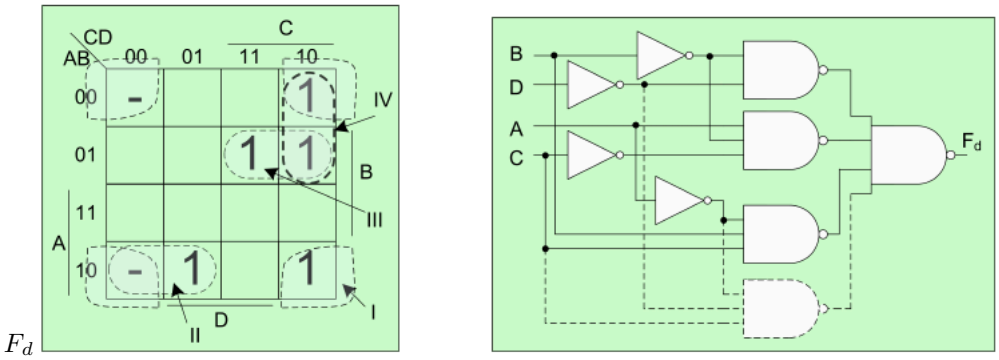

Figura 7.8 F_d, problema 26-d: diagrama V-K şi implementare cu porţi NAND, fără hazard combinaţional.

Funcţia fără hazard combinaţional devine:
$F_d = I + II + III + IV = \overline{B} \cdot \overline{D} + A \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot C + \overline{A} \cdot C \cdot \overline{D}$

3. Să se implementeze funcţia: $F(A, B, C, D) = \prod(1, 3, 4, 5, 7, 10, 11, 12, 14, 15)$ cu două nivele de porţi logice OR-AND fără a produce hazard static.

   Soluţie

   Circuitele combinaţionale implementate pe două nivele logice OR-AND (sau NOR-NOR), implementează pe primul nivel logic implicanţi primi, iar pe al doilea nivel logic se alătură implicanţii primi într-o poartă OR. Acest tip de implementare a funcţiilor minimizate poate produce hazard combinaţional static 0 deoarece ieşirea lor 0 (sinteza formei canonice conjunctive a funcţiei se face pe bază de 0) poate genera un glitch (semnal de scurtă durată) având starea logică 1 în cazul în care o variabilă de intrare comută, dacă pentru anumite valori constante ale celorlalte variabile funcţia se reduce la forma: $X \cdot \overline{X} = 0$.

   Pe baza diagramei V-K prezentate în figura 7.9-a, se obţine expresia minimă în formă canonică conjunctivă: $F = (\overline{A} + \overline{C}) \cdot (A + \overline{D}) \cdot (\overline{B} + C + D)$

4. Implementaţi cu porţi NAND următoarele funcţii cu eliminarea hazardului combinaţional.
   a) $F_a(A, B, C) = \sum(1, 3, 4, 5)$
   b) $F_b(A, B, C) = \sum(1, 2, 3, 5)$
   c) $F_c(A, B, C) = \sum(0, 2, 3, 4, 6)$
   d) $F_d(A, B, C, D) = \sum(0, 1, 2, 5, 6, 7, 15)$
   e) $F_e(A, B, C, D) = \sum(0, 1, 5, 7, 8, 9, 14, 15)$
   f) $F_f(A, B, C, D) = \sum(2, 3, 8, 9, 10, 11)$

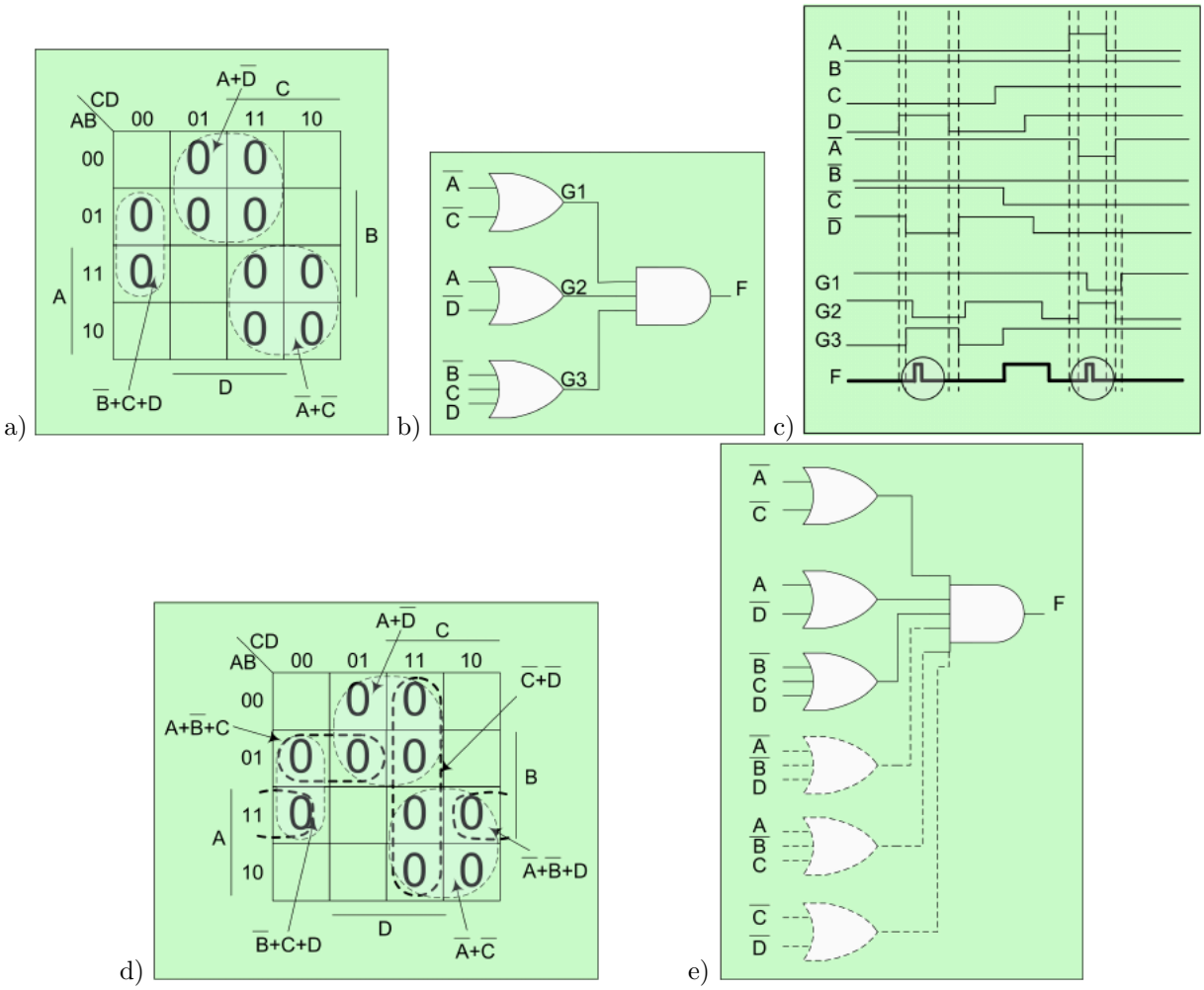

Figura 7.9 a) Diagrama V-K, minimizare cu hazard combinaţional, **b)** Implementare OR-AND, cu hazard combinaţional **c)** Diagramele temporale, **d)** Diagrama V-K, minimizare fără hazard combinaţional, **e)** Implementare OR-AND, fără hazard combinaţional (problema 3).

Soluţie

a) $F_a(A,B,C) = I + II + III = A \cdot \overline{B} + \overline{A} \cdot C + \overline{B} \cdot C = \overline{\overline{(A \cdot \overline{B})} \cdot \overline{(\overline{A} \cdot C)} \cdot \overline{(\overline{B} \cdot C)}}$

5. Proiectaţi un circuit de conversie a cifrelor în baza 10, reprezentate pe 4 biţi BCD (Engl. *"Binary Code Decimal"*), în codurile corespondente pentru comanda unui afişaj cu 7 segmente. Combinaţiile binare care nu au asociat un număr în baza 10 vor determina stingerea tuturor segmentelor. Repetaţi problema cu toate cele 16 combinaţii de intrare. În cazul intrărilor 10-15, segmentelor vor afişa simbolul asociat în baza 16: a, b, c, d, e, f. Comparaţi costurile celor două implementări şi explicaţi diferenţele.

6. Implementaţi următoarea funcţie utilizând porţi XOR şi AND:
$F = A \cdot \overline{B} \cdot C \cdot \overline{D} + \overline{A} \cdot B \cdot C \cdot \overline{D} + A \cdot \overline{B} \cdot \overline{C} \cdot D + \overline{A} \cdot B \cdot \overline{C} \cdot D$

Soluţie

Din diagrama V-K, prezentată în figura 7.11-a reiese că forma canonică a funcţiei este şi formă minimă.
Implementarea cu porţi NAND pe două nivele necesită:
4 porţi NOT şi,
5 porţi NAND x 4
Total: 9 porţi cu 24 de intrări.

Alternativa de implementare se bazează pe observaţia că diagrama V-K are o formă particulară de simetrie, ceea ce sugerează o implementare cu porţi XOR. Prin prelucrări algebrice, se pune funcţia într-o formă simplificată pe baza operatorului XOR:
$F = A \cdot \overline{B} \cdot C \cdot \overline{D} + \overline{A} \cdot B \cdot C \cdot \overline{D} + A \cdot \overline{B} \cdot \overline{C} \cdot D + \overline{A} \cdot B \cdot \overline{C} \cdot D = (A \cdot \overline{B} \cdot C \cdot \overline{D} + A \cdot \overline{B} \cdot \overline{C} \cdot D) + (\overline{A} \cdot B \cdot C \cdot \overline{D} + \overline{A} \cdot B \cdot \overline{C} \cdot D) =$

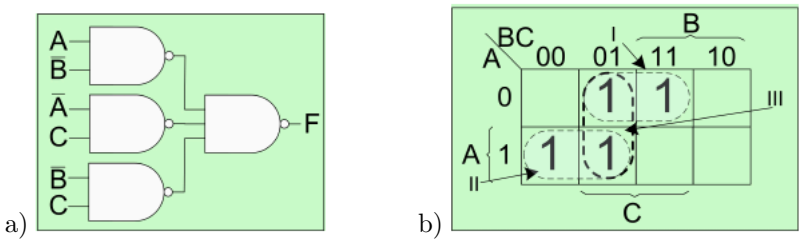

Figura 7.10 Problema 4-a: diagrama V-K şi structura de porţi NAND cu eliminarea hazardului combinaţional.

$A\cdot\overline{B}\cdot(C\cdot\overline{D}+\overline{C}\cdot D)+\overline{A}\cdot B\cdot(C\cdot\overline{D}+\overline{C}\cdot D) = A\cdot\overline{B}\cdot(C\oplus D)+\overline{A}\cdot B\cdot(C\oplus D) = (C\oplus D)\cdot(A\cdot\overline{B}+\overline{A}\cdot B) = (C\oplus D)\cdot(A\oplus B)$
Rezultă circuitul prezentat în figura 7.11-b.

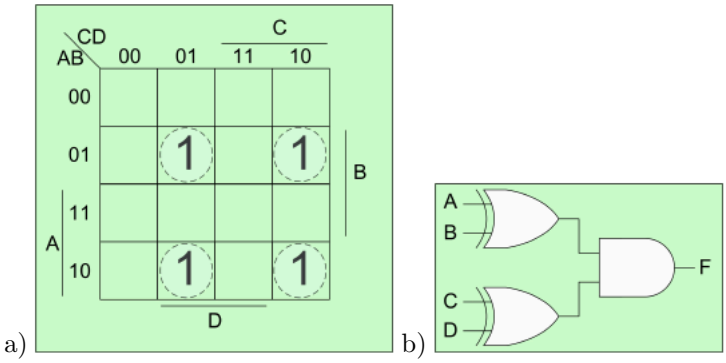

Figura 7.11 a) Diagrama V-K, **b)** circuitul pentru problema 6.

7. Proiectaţi un circuit logic pe două nivele de porţi NAND care să implementeze fără hazard combinaţional funcţia: $F(A,B,C,D) = \sum(0,2,4,5,6,7,8,10,11,15)$.

   Soluţie

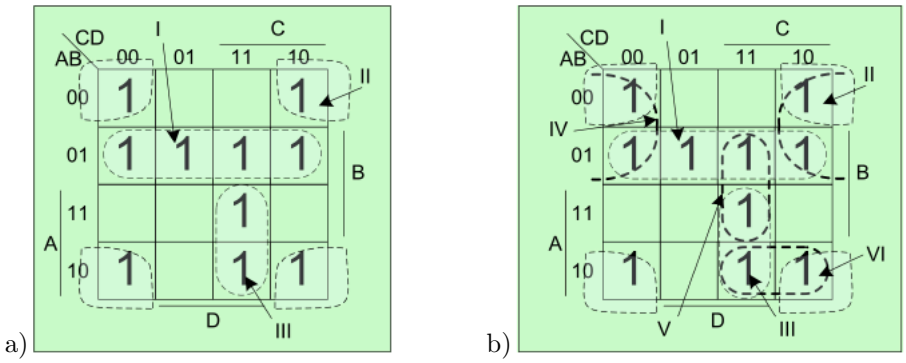

Figura 7.12 Diagrame V-K asociate funcţiei referite la problema 7. Minimizare fără hazard combinaţional.

Diagrama V-K asociată funcţiei este prezentată în figura 7.12-a. Funcţia minimizată este:
$F = I + II + III = \overline{A}\cdot B + \overline{B}\cdot\overline{D} + A\cdot C\cdot D$
Pentru eliminarea hazardului combinaţional mai trebuie definite suprafeţe suplimentare, conform figurii 7.12-b.
$F = I + II + III + IV + V + VI = \overline{A}\cdot B + \overline{B}\cdot\overline{D} + A\cdot C\cdot D + \overline{A}\cdot\overline{D} + B\cdot C\cdot D + A\cdot\overline{B}\cdot C$

Implementarea pe două nivele de porţi NAND se face conform expresiei:
$F = \overline{\overline{A}\cdot B + \overline{B}\cdot\overline{D} + A\cdot C\cdot D + \overline{A}\cdot\overline{D} + B\cdot C\cdot D + A\cdot\overline{B}\cdot C} = \overline{\overline{A}\cdot B\cdot\overline{B}\cdot\overline{D}\cdot\overline{A}\cdot C\cdot D\cdot\overline{A}\cdot\overline{D}\cdot\overline{B}\cdot C\cdot D\cdot\overline{A}\cdot\overline{B}\cdot C}$

Circuitul este prezentat în figura 7.13.

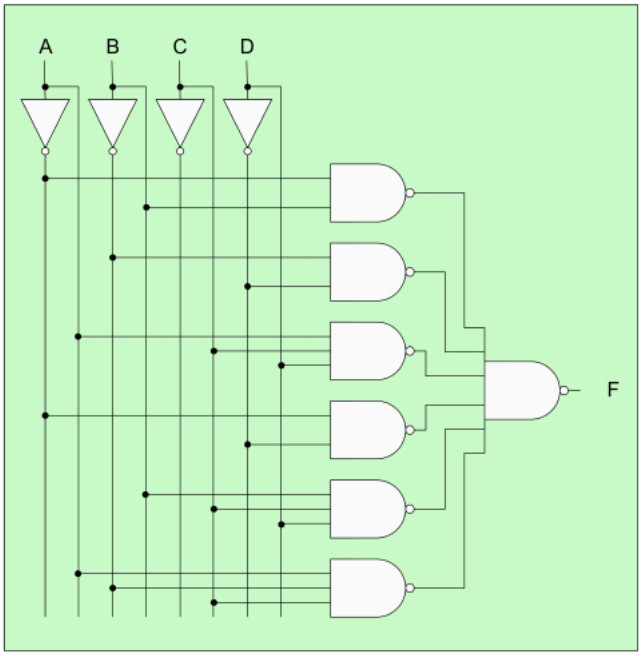

Figura 7.13 Circuit pe două nivele de porţi NAND, implementare a funcţiei referite de problema 7.

7.4 Pentru cei ce vor să devină profesionişti

1. Proiectaţi un circuit generator de paritate de 3 biţi şi un circuit verificator de paritate de 4 biţi, utilizând regula parităţii pare.

   Soluţie

   Generatorul de paritate pară prezintă la ieşire 1 dacă la intrare sunt un număr impar de biţi egali cu 1 şi 0 dacă la intrare sunt un număr par de biţi egali cu 1. Această funcţionalitate este realizată de o poartă logică XOR: $EVEN = I_2 \oplus I_1 \oplus I_0$

   Eroarea de paritate verifică paritatea datelor recepţionate împreună cu bitul de paritate. Dacă suma celor 4 biţi este un număr par, verificarea este corectă, altfel se semnalează eroare de paritate:
   $ERR_{even} = I_2 \oplus I_1 \oplus I_0 \oplus EVEN$

   Circuitele sunt prezentate în figura 7.14.

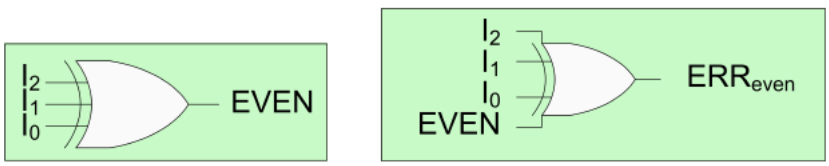

Figura 7.14 Generator şi verificator de paritate pară de 3 biţi.

2. Să se implementeze un circuit logic combinaţional care produce conversia numerelor de la 0 la 15 exprimate în cod binar în reprezentarea acestora în cod Gray.

   Soluţie

   Tabelul de adevăr al convertorului din cod binar în cod Gray este:

Cod Binar				Cod Gray			
B_3	B_2	B_1	B_0	G_3	G_2	G_1	G_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

Diagramele V-K asociate ieşirilor circuitului $G_{[3:0]}(B_{[3:0]})$ sunt prezentate în figura 7.15.

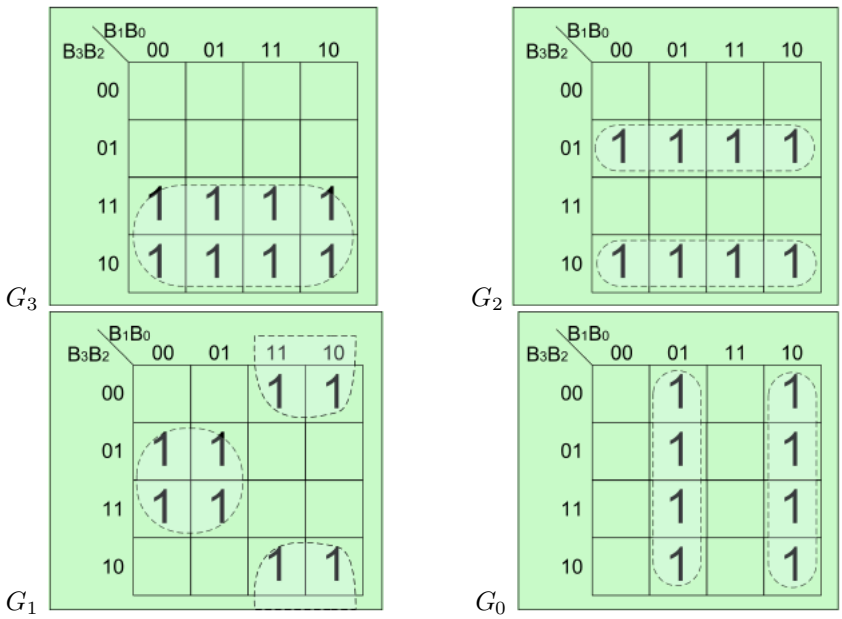

Figura 7.15 Diagramele V-K pentru convertorul binar-Gray pe 4 biţi.

După minimizarea funcţiilor, se obţin ecuaţiile:

$G_3 = B_3$
$G_2 = B_3 \cdot \overline{B_2} + \overline{B_3} \cdot B_2 = B_3 \oplus B_2$
$G_1 = B_2 \cdot \overline{B_1} + \overline{B_2} \cdot B_1 = B_2 \oplus B_1$
$G_0 = B_1 \cdot \overline{B_0} + \overline{B_1} \cdot B_0 = B_1 \oplus B_0$

Circuitul rezultat, implementat cu porţi XOR, este prezentat în figura 7.16. Se observă faptul că se poate obţine o implementare iterativă, posibil de extins pentru conversia din cod binar în cod Gray a numerelor reprezentate pe mai mulţi biţi. Generalizând relaţiile, se pot scrie ecuaţiile conversiei din binar în cod Gray, pentru N biţi astfel:

$G_{N-1} = B_{N-1}$
$G_i = B_{i+1} \oplus B_i$, pentru $\forall i \in [(N-2), ..., 0]$

3. Să se implementeze un circuit logic combinaţional care produce conversia din cod Gray pe 4 biţi în cod binar.

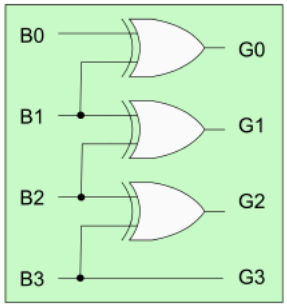

Figura 7.16 Implementarea convertorului numerelor din binar în cod Gray pe 4 biţi, cu porţi XOR.

Soluţie

Tabelul de adevăr al convertorului din cod binar în cod Gray este prezentat la problema 2. Diagramele V-K asociate ieşirilor circuitului $G_{[3:0]}(B_{[3:0]})$ sunt prezentate în figura 7.17.

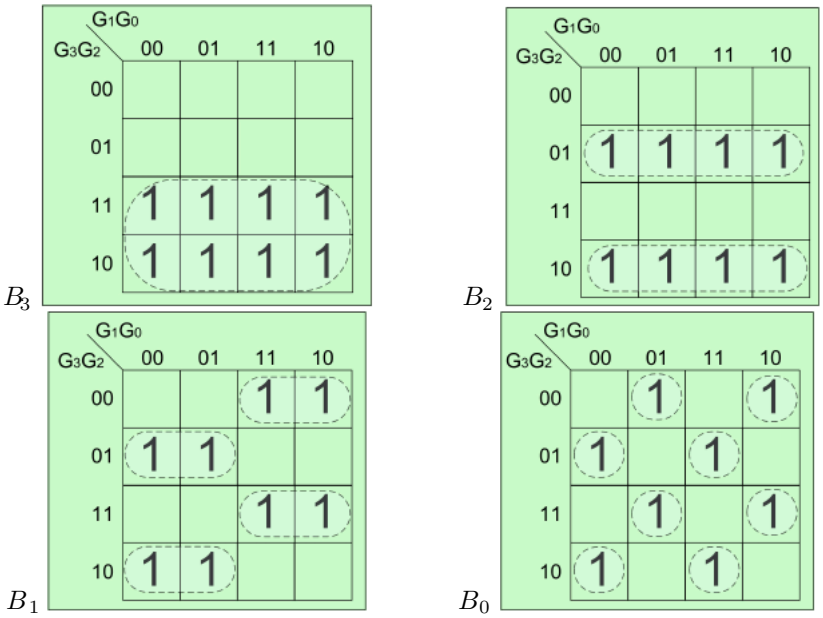

Figura 7.17 Diagramele V-K pentru convertorul din cod Gray pe 4 biţi în cod binar.

După minimizarea funcţiilor, se obţin ecuaţiile:

$B_3 = G_3$
$B_2 = G_3 \cdot \overline{G_2} + \overline{G_3} \cdot G_2 = G_3 \oplus G_2 = B_3 \oplus G_2$
$B_1 = G_3 \cdot G_2 \cdot \overline{G_1} + G_3 \cdot \overline{G_2} \cdot G_1 + \overline{G_3} \cdot G_2 \cdot G_1 + G_3 \cdot G_2 \cdot G_1 = G_3 \oplus G_2 \oplus G_1 = B_2 \oplus G_1$
$B_0 = G_3 \cdot G_2 \cdot G_1 \cdot \overline{G_0} + G_3 \cdot G_2 \cdot \overline{G_1} \cdot G_0 + G_3 \cdot \overline{G_2} \cdot G_1 \cdot G_0 + \overline{G_3} \cdot G_2 \cdot G_1 \cdot G_0 +$
$+ G_3 \cdot \overline{G_2} \cdot \overline{G_1} \cdot \overline{G_0} + \overline{G_3} \cdot G_2 \cdot \overline{G_1} \cdot \overline{G_0} + \overline{G_3} \cdot \overline{G_2} \cdot G_1 \cdot \overline{G_0} + \overline{G_3} \cdot \overline{G_2} \cdot \overline{G_1} \cdot G_0 =$
$= G_3 \oplus G_2 \oplus G_1 \oplus G_0 = B_1 \oplus G_0$

Circuitul rezultat, implementat cu porţi XOR, este prezentat în figura 7.18. Se observă faptul că se poate obţine o implementare iterativă, posibil de extins pentru conversia din cod Gray în cod binar a numerelor reprezentate pe mai mulţi biţi. Generalizând relaţiile, se pot scrie ecuaţiile conversiei din cod Gray în cod binar, pentru N biţi astfel:

$B_{N-1} = G_{N-1}$
$B_i = B_{i+1} \oplus G_i$, pentru $\forall i \in [(N-2), ..., 0]$

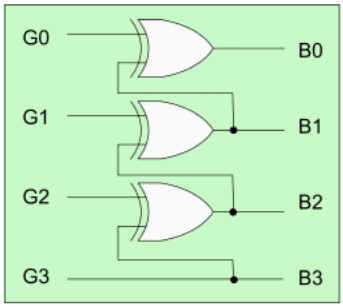

Figura 7.18 Implementarea convertorului din cod Gray pe 4 biţi în binar, cu porţi XOR.