Dan NICULA

ELECTRONICĂ DIGITALĂ
Carte de învăţătură 2.0

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

2015

Lecţia 11

Dispozitive programabile combinaţionale

11.1 Noţiuni teoretice

Dispozitive programabile (Engl. *"PLD = Programmable Logic Devices"*) sunt dispozitive cu structură fixă care prin programare pot determina un circuit combinaţional cu funcţii de transfer configurabile. Programabilitatea trebuie înţeleasă ca fiind modificarea funcţiei de transfer printr-o particularizare explicită a unor resurse. Există dispozitive care se pot particulariza/programa o singură dată (Engl. *"OTP = One Time Programmable")* şi dispozitive programabile de mai multe ori, cu configuraţii diferite, denumite dispozitive re-programabile.

Principiul care stă la baza dispozitivelor combinaţionale programabile constă în implementarea oricărei funcţii logice exprimată în FCND (sumă de produse) pe o structură formată din două nivele logice: nivelul de porţi AND care formează expresiile produs şi nivelul de porţi OR care adună ieşirile porţilor AND pentru a genera funcţiile de ieşire. Structura programabilă este reprezentată în figura 11.1.

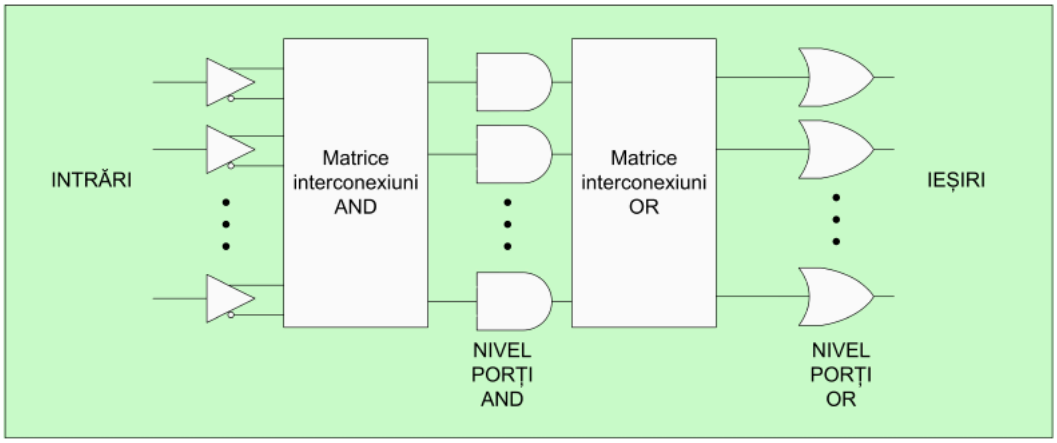

Figura 11.1 Structuri combinaţionale programabile.

În funcţie de programabilitatea sau nu a celor două matrici AND şi OR, dispozitivele programabile se pot clasifica conform tabelului:

Tip dispozitiv	**Matrice AND**	**Matrice OR**
ROM	Fix din fabrică	Programabil de către utilizator
PLA	Programabil de către utilizator	Programabil de către utilizator
PAL	Programabil de către utilizator	Fix din fabrică

Memoria ROM (Read Only Memory) este un circuit logic combinaţional realizat ca un dispozitiv programabil având un nivel AND fix şi complet (în interiorul structurii se generează toţi mintermii asociaţi intrărilor) şi un nivel OR

configurabil. Intrările în ROM sunt considerate *adresele* iar ieşirile sunt considerate *datele* citite din memorie. Întreaga memorie poate fi văzută ca fiind matrice în care datele sunt stocate în locaţii pe baza unei adrese. Simbolul bloc al unei memorii ROM $2^N \times M$ este prezentat în figura 11.2. S-au notat cu N numărul de biţi de adresă (în total 2^N adrese) şi cu M numărul de biţi de date.

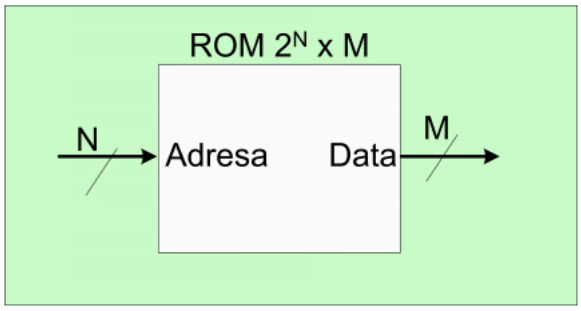

Figura 11.2 Simbol bloc al memoriei ROM $2^N \times M$.

Implementarea unui circuit logic combinaţional cu ROM nu necesită minimizarea funcţiilor. Intrările sunt asociate cu biţii de adresă ai circuitului ROM iar ieşirile circuitului combinaţional se obţin de la datele memoriei ROM. Implementarea constă în realizarea schemei de conectare a memoriei ROM şi a tabelului de programare a acesteia (adrese, date).

Structura ROM prezintă primul nivel AND complet, implementat sub forma unui decodificator care generează toţi mintermii asociaţi intrărilor. Nivelul OR conţine un număr de porţi programabile egal cu numărul de ieşiri. Pe un circuit ROM $2^N \times M$ pot fi implementate orice M funcţii care depind de aceleaşi N intrări.

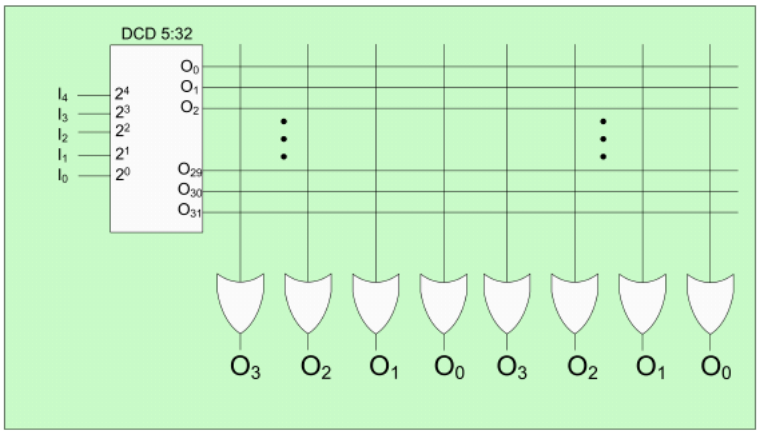

Figura 11.3 Structura ROM $2^5 \times 8$.

Circuitele PLA (Programmable Logic Array) au ambele nivele AND şi OR programabile de către utilizator. Primul nivel, AND, implementează implicanţi primi, iar al doilea nivel, OR, uneşte anumiţi implicanţi primi (configurabili) pentru a realiza anumite funcţii (nu orice funcţii). Implementarea circuitelor logice combinaţionale cu PLA necesită minimizare deoarece acestea nu generează în structura lor decât un număr limitat de implicanţi primi (spre deosebire de memoriile ROM care generează toţi mintermii asociaţi intrărilor). Structura PAL este limitată atât pe nivelul AND cât şi pe nivelul OR.

Figura 11.4 prezintă o structură generică de PAL având 6 porţi AND pe primul nivel logic şi 4 porţi OR pe al doilea nivel logic. Porţile AND au câte 8 intrări, posibil de conectat la cele 4 intrări (negate sau ne-negate). Pe nivelul AND se pot genera 8 implicanţi primi, posibil de utilizat în expresiile tuturor celor 4 funcţii de ieşire.

Porţile OR au câte 6 intrări, posibil de conectat la cele ieşirile celor 6 porti AND de pe primul nivel, realizând astfel funcţii cu maximum 6 implicanţi primi.

Circuitele PAL (Programmable Array Logic) au nivelul AND programabil de către utilizator şi nivelul OR fix. Primul nivel, AND, implementează implicanţi primi, iar al doilea nivel, OR, uneşte implicanţii primi pentru a realiza anumite funcţii (nu orice funcţii). Implementarea circuitelor logice combinaţionale cu PAL necesită minimizare deoarece acestea

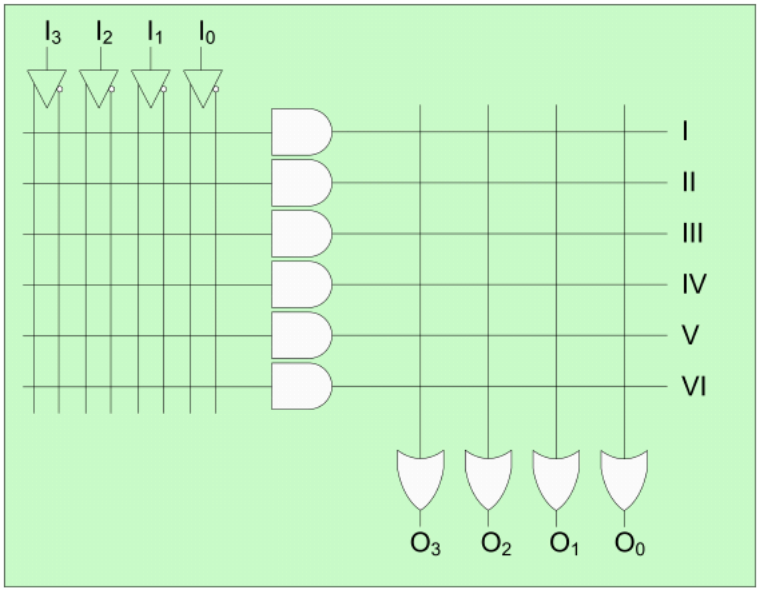

Figura 11.4 Structura PLA cu 4 intrări, 6 porţi AND configurabile pe primul nivel, 4 porţi OR configurabile pe al doilea nivel şi 4 ieşiri.

nu generează în structura lor decât un număr limitat de implicanţi primi (spre deosebire de memoriile ROM care generează toţi mintermii asociaţi intrărilor). Structura PLA este limitată atât pe nivelul AND cât şi pe nivelul OR. Mai mult, faţă de PLA, un circuit PAL nu poate utiliza un implicant prim decât pentru o singură ieşire. Dacă două ieşiri conţin în expresiile lor acelaşi implicant prim, acesta trebuie programat de două ori, odată în structura OR fixă a primei ieşiri şi altă dată în structura OR fixă a celei de-a doua ieşiri.

Figura 11.5 prezintă o structură generică de PAL cu 4 intrări şi 2 ieşiri. Pe nivelul AND, fiecare din cele două ieşiri au asociaţi câte 4 implicanţi primi. Pe nivelul OR, fix, există câte o poartă OR pentru fiecare ieşire.

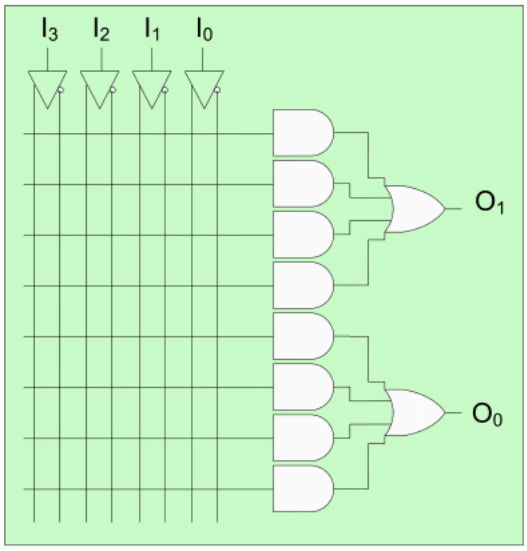

Figura 11.5 Structura PAL cu 4 intrări, 2×4 porţi AND configurabile pe primul nivel, 2 porţi OR fixe pe al doilea nivel şi 2 ieşiri.

11.2 Pentru cei ce vor doar să promoveze examenul

1. Implementaţi cu ROM un circuit logic combinaţional care realizează operaţia de ridicare la putere A^B cu operanzi numere naturale codificate pe câte 2 biţi.

   Soluţie

   Operandul $A \in \{1, 2, 3\}$, operandul $B \in \{0, 1, 2, 3\}$.

   Rezultă că cel mai mare rezultat este $3^3 = 27$, posibil de codificat pe 5 biţi.

   Pentru implementare este necesară o memorie ROM de dimensiune $2^4 \times 5$. Conectarea memoriei ROM este prezentată în figura 11.6.

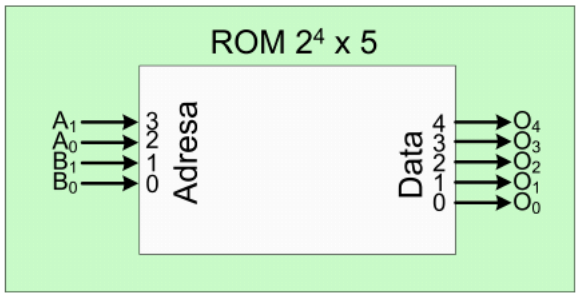

Figura 11.6 Implementarea cu ROM $2^4 \times 5$ a circuitului aritmetic descris la problema 1.

Tabelul cu conţinutul memoriei este următorul:

Numere naturale			Binar			ROM	
A	B	A^B	$A[1:0]$	$B[1:0]$	A^B	Adresa[3:0]	Data[4:0]
0	0	X	00	00	XXXXX	$0000 = 0\|_{10}$	$XXXXX = X\|_{10}$
0	1	X	00	01	XXXXX	$0001 = 1\|_{10}$	$XXXXX = X\|_{10}$
0	2	X	00	10	XXXXX	$0010 = 2\|_{10}$	$XXXXX = X\|_{10}$
0	3	X	00	11	XXXXX	$0011 = 3\|_{10}$	$XXXXX = X\|_{10}$
1	0	1	01	00	00001	$0100 = 4\|_{10}$	$00001 = 1\|_{10}$
1	1	1	01	01	00001	$0101 = 5\|_{10}$	$00001 = 1\|_{10}$
1	2	1	01	10	00001	$0110 = 6\|_{10}$	$00001 = 1\|_{10}$
1	3	1	01	11	00001	$0111 = 7\|_{10}$	$00001 = 1\|_{10}$
2	0	1	10	00	00001	$1000 = 8\|_{10}$	$00001 = 1\|_{10}$
2	1	2	10	01	00010	$1001 = 9\|_{10}$	$00010 = 2\|_{10}$
2	2	4	10	10	00100	$1010 = 10\|_{10}$	$00100 = 4\|_{10}$
2	3	8	10	11	01000	$1011 = 11\|_{10}$	$01000 = 8\|_{10}$
3	0	1	11	00	00001	$1100 = 12\|_{10}$	$00001 = 1\|_{10}$
3	1	3	11	01	00011	$1101 = 13\|_{10}$	$00011 = 3\|_{10}$
3	2	9	11	10	01001	$1110 = 14\|_{10}$	$01001 = 9\|_{10}$
3	3	27	11	11	11011	$1111 = 15\|_{10}$	$11011 = 27\|_{10}$

2. Să se realizeze un modul de memorie ROM $8K \times 16$ utilizând circuite ROM $2K \times 8$.
   Să se identifice chip-ul defect dacă la citirea memorie se determină următoarele erori:
   a) La adresa 666 se citeşte $\#ABCD$ deşi trebuia să se citească $\#ABCC$.
   b) La adresa 6666 se citeşte #7777 deşi trebuia să se citească #6666.
   c) La adresa 4000 se citeşte $\#ABCD$ deşi trebuia să se citească $\#ABCC$.
   d) La adresa 2012 se citeşte $\#DEAD$ deşi trebuia să se citească $\#DEED$.

   Soluţie

   Pentru a realiza modulul de memorie $8K \times 16$ sunt necesare $4 \times 2 = 8$ chip-uri $2K \times 8$. Modulul are 13 biţi de adresă iar chip-ul are doar 11 biţi de adresă. Cei mai puţin semnificativi 11 biţi se conectează pe porturile de adresă ale tuturor chip-urilor, adresând simultan toate chip-urile. Cei mai semnificativi 2 biţi ai adresei modulului selectează chip-ul ale cărui date vor fi prezentate pe ieşirea modulului. Cuvântul de date citite de 16 biţi se obţine din concatenarea a două grupuri de câte 8 biţi generaţi de două chip-uri adiacente.

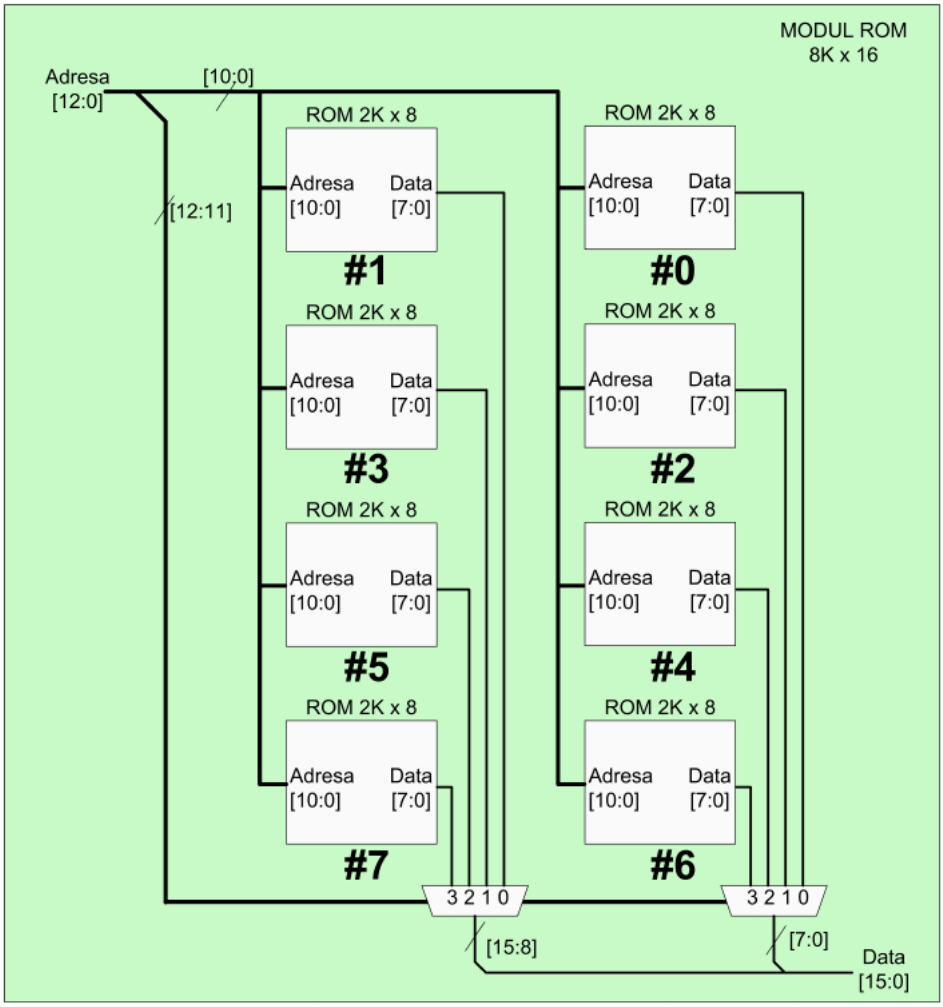

Figura 11.7 Modul ROM $8K \times 16$ implementat cu chip-uri ROM $2K \times 8$.

Pentru a determina rândul chip-ului defect se converteşte în binar adresa şi se deduc valorile celor mai semnificativi doi biţi (care selectează multiplexorul). Circuitul este prezentat în figura 11.7.
a) Adresa $666|_{10} = 0\_0010\_1001\_1010|_2 = 00\_01010011010$, adresează rândul cu chip-urile #1 şi #0.
Diferenţa în conţinutul datelor între $\#ABCD$ şi $\#ABCC$ este la bitul 0, care este stocat în chip-ul din dreapta. Deci, chip-ul defect este chip-ul #0.

3. Să se implementeze cu ROM 64×1 funcţia:
$Y(A,B,C,D,E,F) = \sum(0,1,2,3,4,5,6,9,11,12,13,15,16,19,21,25,26,28,29,30,31,33,34,36,37,39,41,44,45,46,49,51,53,55,57,59,60,61,62)$

   Soluţie

   Implementarea cu ROM a circuitelor logice combinaţionale nu necesită minimizarea funcţiilor. În cazul în care numărul de biţi de adresă pentru ROM este egal cu numărul de intrări ale funcţiei, variabilele funcţiei se conectează direct pe intrările de adrese ale circuitului ROM. Datele citite din ROM reprezintă funcţia. Particularizarea funcţiei constă în conţinutul memoriei ROM, descris printr-un tabel de corespondenţă adresă-date. Circuitul este prezentat în figura 11.8. Tabelul cu conţinutul memoriei ROM conţine 1 la adresele a căror valoare apare în forma canonică a funcţiei Y şi 0 în rest.

4. Care este diferenţa dintre circuitele PAL şi circuitele PLA?

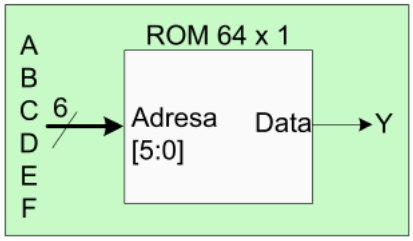

Figura 11.8 Implementare cu ROM 64 × 1, problema 3.

11.3 Pentru cei ce vor să înveţe

1. Să se implementeze pe o structură de circuit PLA generic un convertor de cod din BCD pentru o matrice de afişare pe 7 segmente.

   Soluţie

   Minimizarea corelată este tehnica uzuală pentru obţinerea funcţiilor de implementat pe circuite PLA deoarece aceste circuite au resurse interne limitate pe nivelul de AND programabil. Tabelul de adevăr pentru conversia din cod BCD în 7 segmente este:

A	B	C	D	S_0	S_1	S_2	S_3	S_4	S_5	S_6
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	0	0	1	1
1	0	1	0	x	x	x	x	x	x	x
1	0	1	1	x	x	x	x	x	x	x
1	1	0	0	x	x	x	x	x	x	x
1	1	0	1	x	x	x	x	x	x	x
1	1	1	0	x	x	x	x	x	x	x
1	1	1	1	x	x	x	x	x	x	x

   Utilizând reprezentarea funcţiilor cu diagrame V-K, se obţin formele minime ale celor 7 funcţii de ieşire (figura 11.9):

$$S_0 = A + B \cdot D + C + \overline{B} \cdot \overline{D}$$
$$S_1 = A + \overline{C} \cdot \overline{D} + C \cdot D + \overline{B}$$
$$S_2 = A + B + \overline{C} + D$$
$$S_3 = \overline{B} \cdot \overline{D} + C \cdot \overline{D} + B \cdot \overline{C} \cdot D + \overline{B} \cdot C$$
$$S_4 = \overline{B} \cdot \overline{D} + C \cdot D$$
$$S_5 = A + \overline{C} \cdot \overline{D} + B \cdot \overline{D} + B \cdot \overline{C}$$
$$S_6 = A + C \cdot \overline{D} + B \cdot \overline{C} + \overline{B} \cdot C$$

   Funcţiile minimizate independent conţin 14 termeni produs diferiţi şi necesită, la o implementare cu circuit PLA, 14 porţi programabile AND. Acest număr de porţi AND poate fi redus dacă se minimizează corelat cele 7 funcţii. În figura 11.10 sunt prezentate diagramele V-K cu o minimizare corelată a celor 7 funcţii. S-a căutat să se identifice implicanţi primi, nu neapărat esenţiali, care să fie comuni la cât mai multe diagrame V-K obţinându-se următoarele expresii pentru cele 7 ieşiri:

$$S_0 = I + II + III + IV + V = B \cdot \overline{C} \cdot D + C \cdot D + \overline{B} \cdot \overline{D} + A + B \cdot C \cdot \overline{D}$$
$$S_1 = VI + VII + II + III = \overline{B} \cdot D + \overline{C} \cdot \overline{D} + C \cdot D + \overline{B} \cdot \overline{D}$$
$$S_2 = VI + I + VII + II + V = \overline{B} \cdot D + B \cdot \overline{C} \cdot D + \overline{C} \cdot \overline{D} + C \cdot D + B \cdot C \cdot \overline{D}$$
$$S_3 = VIII + I + III + V = \overline{B} \cdot C + B \cdot \overline{C} \cdot D + \overline{B} \cdot \overline{D} + B \cdot C \cdot \overline{D}$$

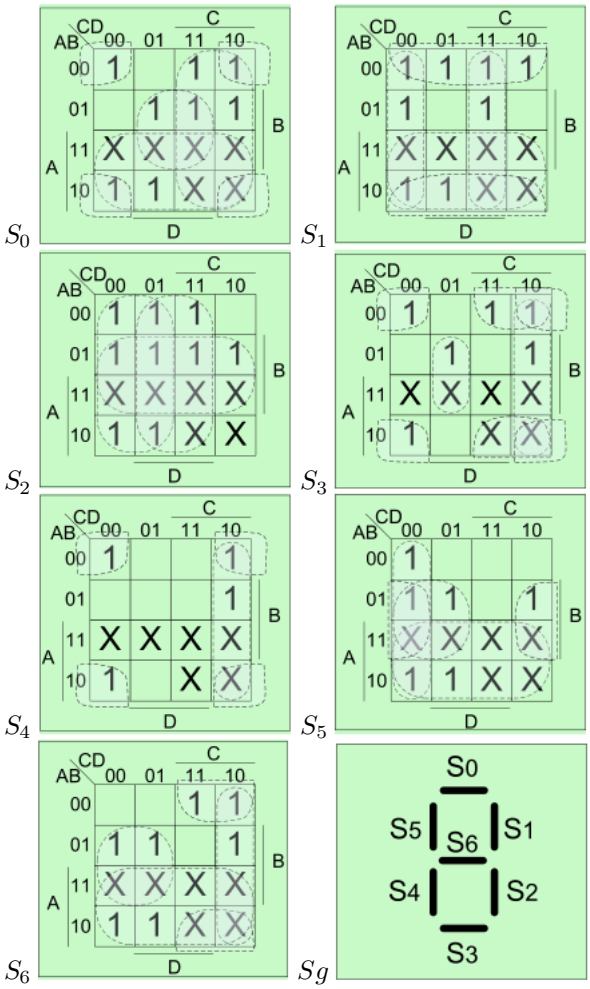

Figura 11.9 Minimizare individuală pe diagramele V-K, problema 1.

$$S_4 = III + V = \overline{B} \cdot \overline{D} + B \cdot C \cdot \overline{D}$$
$$S_5 = IX + VII + IV + V = B \cdot \overline{C} + \overline{C} \cdot \overline{D} + A + B \cdot C \cdot \overline{D}$$
$$S_6 = IX + VIII + IV + V = B \cdot \overline{C} + \overline{B} \cdot C + A + B \cdot C \cdot \overline{D}$$

Funcţiile corelate conţin numai 9 termeni produs diferiţi. Deci, pentru implementare sunt necesare 9 porţi AND programabile. Circuitul rezultat este prezentat în figura 11.11.

2. Să se implementeze cu o memorie ROM un conventor de cod din BCD în 7 segmente. Circuitul are 4 intrări şi 7 ieşiri.

Soluţie

Memoria ROM necesară are dimensiune $2^4 \times 7$. Modul de conectare este prezentat în figura 11.12.

Conţinutul memoriei ROM este prezentat în tabel:

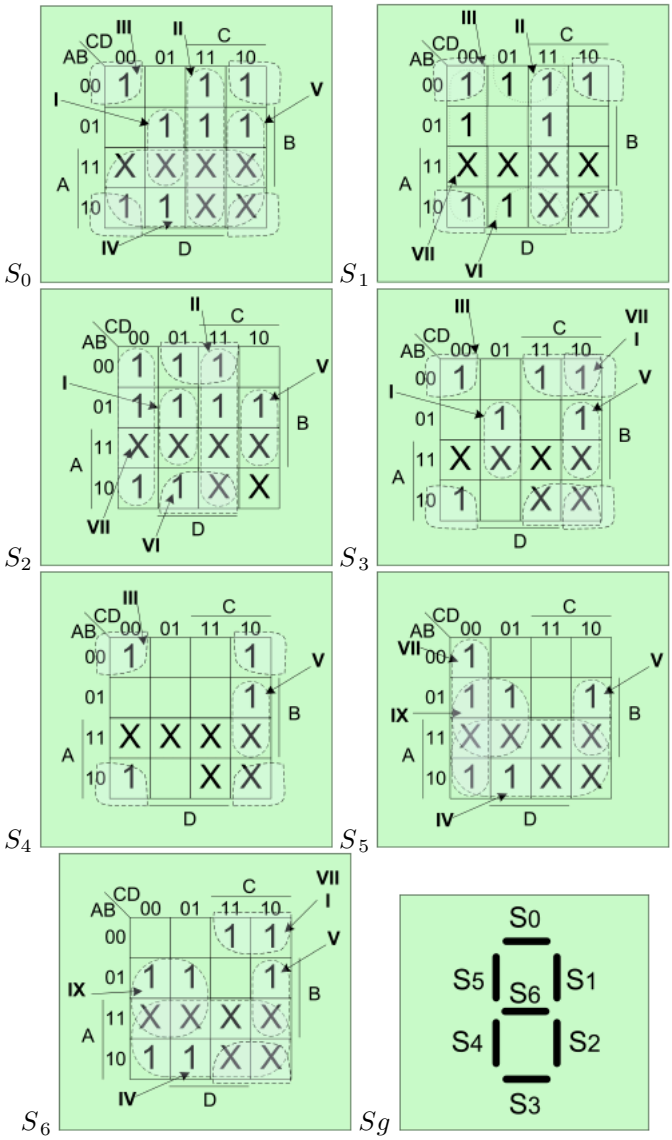

Figura 11.10 Minimizare corelată pe diagramele V-K, problema 1.

Adresă $[3:0] = \{ABCD\}$	Data $[6:0] = S_{0123456}(hex)$
$0000 = 0\|_{10}$	$111\_1110 = 7EH$
$0001 = 1\|_{10}$	$011\_0000 = 30H$
$0010 = 2\|_{10}$	$110\_1101 = 6DH$
$0011 = 3\|_{10}$	$111\_1001 = 79H$
$0100 = 4\|_{10}$	$011\_0011 = 33H$
$0101 = 5\|_{10}$	$101\_1011 = 5BH$
$0110 = 6\|_{10}$	$101\_1111 = 5FH$
$0111 = 7\|_{10}$	$111\_0000 = 70H$
$1000 = 8\|_{10}$	$111\_1111 = 7FH$
$1001 = 9\|_{10}$	$111\_1011 = 7BH$
$1010 = 10\|_{10}$	$XXX\_XXXX = XXH$
$1011 = 11\|_{10}$	$XXX\_XXXX = XXH$
$1100 = 12\|_{10}$	$XXX\_XXXX = XXH$
$1101 = 13\|_{10}$	$XXX\_XXXX = XXH$
$1110 = 14\|_{10}$	$XXX\_XXXX = XXH$
$1111 = 15\|_{10}$	$XXX\_XXXX = XXH$

3. Să se implementeze pe structuri generice PLA şi PAL (cu patru termeni produs pe fiecare poartă OR) un

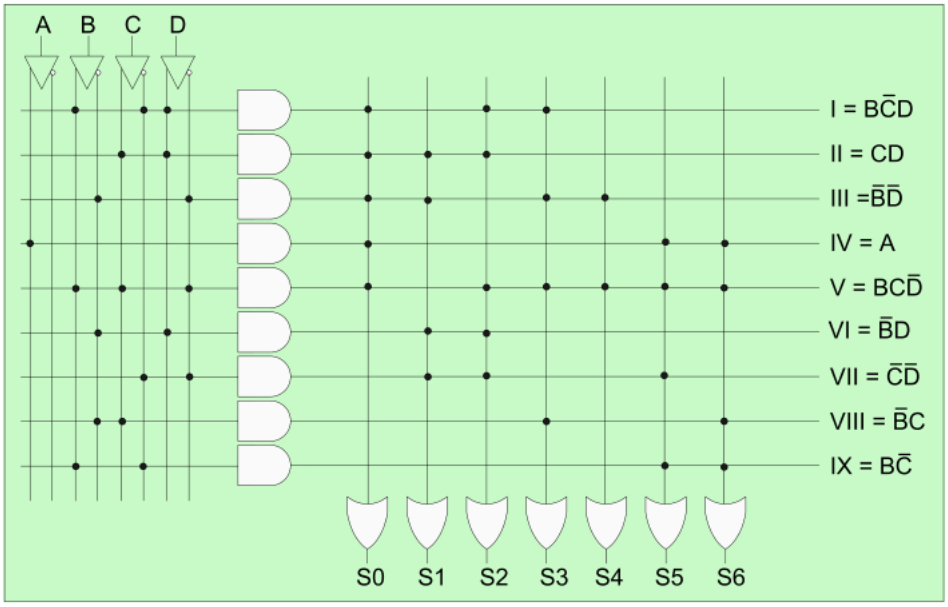

Figura 11.11 Implementarea convertorului din cod BCD în cod matrice cu şapte segmente pe un circuit PLA generic, problema 1.

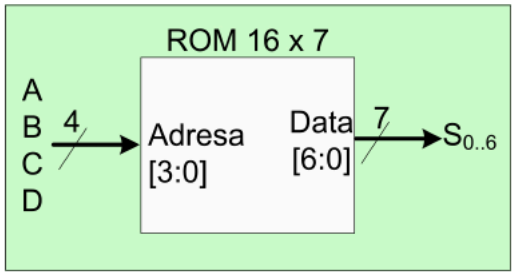

Figura 11.12 Memoria ROM pentru implementarea convertorului din cod BCD în cod matrice cu 7 segmente.

convertor din BCD în cod Gray.

Soluţie

Tabelul de adevăr este:

Intrări BCD				Ieşiri 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	1	1	1	0
0	1	1	0	1	0	1	0
0	1	1	1	1	0	1	1
1	0	0	0	1	0	0	1
1	0	0	1	1	0	0	0
1	0	1	0	x	x	x	x
1	0	1	1	x	x	x	x
1	1	0	0	x	x	x	x
1	1	0	1	x	x	x	x
1	1	1	0	x	x	x	x
1	1	1	1	x	x	x	x

Implementările cu PAL sau PLA necesită exprimarea funcţiilor printr-o formă minimă. Utilizând diagramele V-K, figura 11.13, se obţin formele minime:

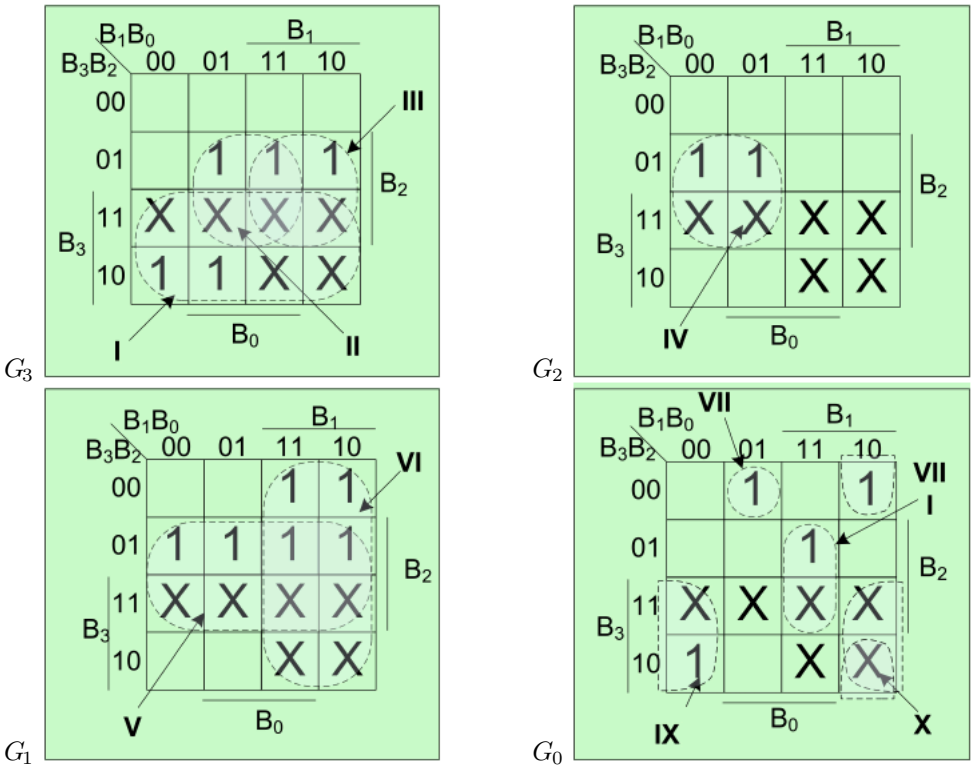

Figura 11.13 Diagramele V-K ale convertorului din cod BCD în cod Gray, problema 3.

$G_3 = I + II + III = B_3 + B_2 \cdot B_0 + B_2 \cdot B_1$
$G_2 = IV = B_2 \cdot \overline{B}_1$
$G_1 = V + VI = B_2 + B_1$
$G_0 = VII + VIII + IX + X = \overline{B}_3 \cdot \overline{B}_2 \cdot \overline{B}_1 \cdot B_0 + B_2 \cdot B_1 \cdot B_0 + B_3 \cdot \overline{B}_0 + \overline{B}_2 \cdot B_1 \cdot \overline{B}_0$

Implementările cu PAL şi PLA sunt prezentate în figurile 11.14-a, b.

4. Să se implementeze următoarele funcţii cu o memorie ROM 64×4:
   $Y_1(F,E,D,C,B,A) = \overline{F} \cdot \overline{D} \cdot \overline{B} + F \cdot D \cdot B \cdot \overline{A} + \overline{F} \cdot E \cdot \overline{D} \cdot \overline{C} \cdot B \cdot \overline{A}$;
   $Y_2(F,E,D,C,B,A) = F \cdot E \cdot \overline{D} \cdot A + F \cdot E \cdot \overline{C} \cdot A + F \cdot E \cdot \overline{D} \cdot \overline{C} \cdot B + F \cdot E \cdot D \cdot C \cdot B \cdot \overline{A}$;
   $Y_3(F,E,D,C,B,A) = F \cdot \overline{E} \cdot \overline{D} \cdot \overline{B} + F \cdot \overline{E} \cdot \overline{D} \cdot C \cdot \overline{A}$;
   $Y_4(F,E,D,C,B,A) = F \cdot \overline{E} \cdot D \cdot A + F \cdot \overline{E} \cdot D \cdot \overline{C} \cdot B + F \cdot \overline{E} \cdot D \cdot C \cdot \overline{B}$.

5. Utilizând circuite ROM de capacitate 256×8, în care sunt înscrise toate rezultatele înmulţirilor dintre numerele reprezentate pe 4 biţi, precum şi circuite sumatoare de lungimi corespunzătoare, să se structureze un multiplicator combinaţional pentru cuvinte de 8 biţi.

6. Să se implementeze pe o structură de circuit PLA generic un comparator digital pentru două cuvinte A şi B de 4 biţi. Circuitul generează la ieşire funcţiile: $A = B$, $A > B$ şi $A < B$. Nu se primesc semnale pentru relaţiile de ordonare de la un modul comparator de rang superior.

7. Implementaţi următoarele funcţii cu:
   a) ROM 16×4;
   b) două circuite ROM 8×4 şi 4 multiplexoare 2:1;
   c) PLA, cu 4 intrări, 4 ieşiri, 6 porţi AND programabile şi 4 porţi OR programabile;
   d) PAL, cu 4 intrări, 4 ieşiri, 3 porţi AND programabile pentru fiecare ieşire.

   $F_1 = A \cdot C + \overline{A} \cdot B \cdot \overline{C} + A \cdot \overline{B} \cdot \overline{C} \cdot \overline{D}$
   $F_2 = A \cdot \overline{C} \cdot \overline{D} + \overline{A} \cdot \overline{B} \cdot C \cdot \overline{D}$

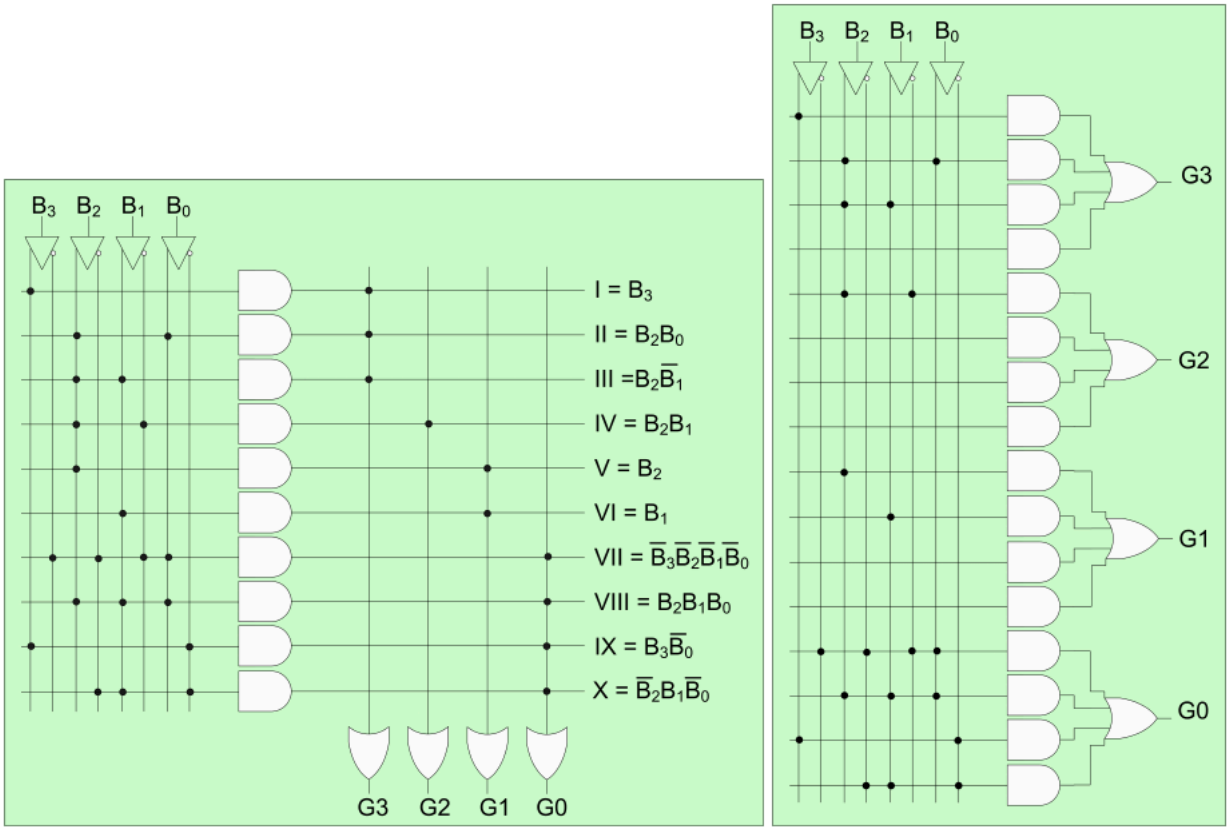

Figura 11.14 Implementarea pe circuite PAL şi PLA generice a convertorului din cod BCD în cod Gray.

$$F_3 = B \cdot \overline{C} \cdot D + A \cdot C \cdot B + \overline{A} \cdot \overline{B} \cdot C \cdot \overline{D} + A \cdot C \cdot \overline{B}$$
$$F_4 = A \cdot \overline{C} \cdot \overline{D} + B \cdot \overline{C} \cdot D + \overline{A} \cdot \overline{B} \cdot C \cdot \overline{D} + A \cdot B \cdot \overline{C}$$

11.4 Pentru cei ce vor să devină profesionişti

1. Se consideră problema: *"Ana are o sumă de bani de la bunica. În fiecare zi cheltuie jumătate plus 1 leu din suma pe care o are, după 3 zile nerămânând nimic. Ce sumă a avut iniţial?".*

   Soluţia problemei generalizate este:

   $$SUMA = 2^{ZILE+1} - 2$$

   Să se implementeze un circuit combinaţional cu ROM $2^5 \times 30$ care să calculeze suma pe baza numărului de zile pentru maximum o lună ($ZILE \in \{1, 31\}$).

   Soluţie

   Rezolvarea problemei se poate face cu un circuit combinaţional care are la intrare numărul de zile (reprezentat pe 5 biţi) şi la ieşire suma, reprezentată pe 32 de biţi. Rezolvarea este evidentă cu un circuit ROM $2^5 \times 32$. Însă, prin observarea valorilor stocate în memorie se remarcă faptul că cei mai puţin semnificativi 2 biţi ai rezultatului au tot timpul aceeaşi valoare "10", deci ar putea să nu mai fie scrişi în memorie:

```
SUMA[1]  = 0000_0000_0000_0000_0000_0000_0000_0010
SUMA[2]  = 0000_0000_0000_0000_0000_0000_0000_0110
SUMA[3]  = 0000_0000_0000_0000_0000_0000_0000_1110
SUMA[4]  = 0000_0000_0000_0000_0000_0000_0001_1110
```

```
...
SUMA[29] = 0011_1111_1111_1111_1111_1111_1111_1110
SUMA[30] = 0111_1111_1111_1111_1111_1111_1111_1110
SUMA[31] = 1111_1111_1111_1111_1111_1111_1111_1110
```

Circuitul este prezentat în figura 11.15 iar tabelul de programare a ROM-ului este prezentat în figura 11.16.

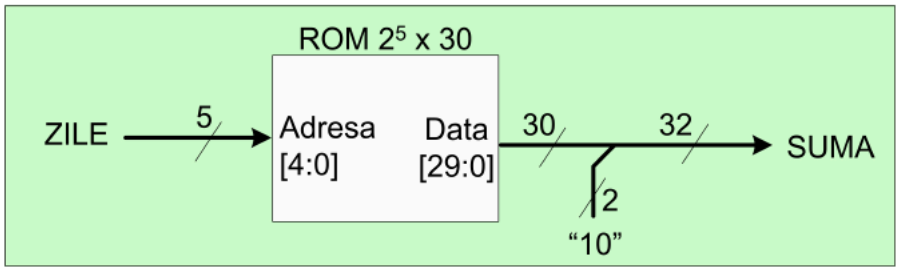

Figura 11.15 Circuit cu ROM care rezolvă *"problema Anei"*.

Adresa[4:0]	Data[31:0]
0	00_0000_0000_0000_0000_0000_0000_0000
1	00_0000_0000_0000_0000_0000_0000_0000
2	00_0000_0000_0000_0000_0000_0000_0001
3	00_0000_0000_0000_0000_0000_0000_0011
4	00_0000_0000_0000_0000_0000_0000_0111
5	00_0000_0000_0000_0000_0000_0000_1111
6	00_0000_0000_0000_0000_0000_0001_1111
7	00_0000_0000_0000_0000_0000_0011_1111
8	00_0000_0000_0000_0000_0000_0111_1111
9	00_0000_0000_0000_0000_0000_1111_1111
10	00_0000_0000_0000_0000_0001_1111_1111
11	00_0000_0000_0000_0000_0011_1111_1111
12	00_0000_0000_0000_0000_0111_1111_1111
13	00_0000_0000_0000_0000_1111_1111_1111
14	00_0000_0000_0000_0001_1111_1111_1111
15	00_0000_0000_0000_0011_1111_1111_1111
16	00_0000_0000_0000_0111_1111_1111_1111
17	00_0000_0000_0000_1111_1111_1111_1111
18	00_0000_0000_0001_1111_1111_1111_1111
19	00_0000_0000_0011_1111_1111_1111_1111
20	00_0000_0000_0111_1111_1111_1111_1111
21	00_0000_0000_1111_1111_1111_1111_1111
22	00_0000_0001_1111_1111_1111_1111_1111
23	00_0000_0011_1111_1111_1111_1111_1111
24	00_0000_0111_1111_1111_1111_1111_1111
25	00_0000_1111_1111_1111_1111_1111_1111
26	00_0001_1111_1111_1111_1111_1111_1111
27	00_0011_1111_1111_1111_1111_1111_1111
28	00_0111_1111_1111_1111_1111_1111_1111
29	00_1111_1111_1111_1111_1111_1111_1111
30	01_1111_1111_1111_1111_1111_1111_1111
31	11_1111_1111_1111_1111_1111_1111_1111

Figura 11.16 Tabel de programare a circuitului ROM care rezolvă *"problema Anei"*.