1.2.4 Sisteme complete de funcţii

Aşa cum s-a mai precizat (v. §1.2.2) funcţiile booleene se pot forma prin superpoziţia funcţiilor elementare. Problema formării funcţiilor booleene poate fi privită şi astfel: să se stabilească un sistem de funcţii care să poată reprezenta orice funcţie boole-ană şi în plus, dacă acest sistem există, să conţină un număr cât mai mic de funcţii. Evident, problema formării funcţiilor booleene pusă în acest mod are o mare importanţă pentru sinteza schemelor logice.

Definiţie. Sistemul de funcţii booleene $(f_1, f_2, \ldots, f_k)$ se numeşte *complet* (*bază* respectiv *sistem funcţional complet de bază*) în clasă R, dacă orice funcţie φ aparţinând lui R poate fi reprezentată prin superpoziţia funcţiilor $(f_1, f_2, \ldots, f_k)$.

În calitate de clasă R se poate considera clasa tuturor funcţiilor booleene care depind de *n* argumente B_2^n. Conform celor arătate în prima parte a subcapitolului 1.2, nu-mărul total al funcţiilor care depind de *n* argumente este egal cu 2^{2^n}. Deci, în clasa B_2^n există un sistem complet compus din cele 2^{2^n} funcţii ale acestei clase, dar acest sistem este trivial. În §1.2.3 s-a stabilit că orice funcţie booleană poate fi reprezentată sub FCD sau FCC. Rezultă că sistemul de funcţii compus din conjuncţie, disjuncţie şi negare constituie un sistem complet de funcţii. Se pune problema dacă sistemul complet de funcţii („+“, „·“, „–“) conţine numărul strict necesar pentru reprezentarea oricărei funcţii booleene. Pentru ca un sistem complet să fie minim (bază minimală) este necesar să satisfacă următoarea definiţie:

Definiţie. Un sistem complet de funcţii $(f_1, f_2, \ldots, f_k)$ este *minim* (*bază minimală*) dacă înlăturând oricare dintre funcţiile aparţinând sistemului acesta devine incomplet.

Completitudinea sistemului de funcţii („+“, „·“, „–“) permite demonstrarea completitudinii oricărui alt sistem de funcţii arbitrar format. Pentru aceasta este suficient să se arate că funcţiile sistemului ales pot reprezenta funcţiile sistemului („+“, „·“, „–“).

Corespunzător celor arătate mai sus, interes practic prezintă evidenţierea următoa-relor teoreme:

Teoremă. Sistemul de funcţii format din conjuncţie şi negaţie („·“, „–“), este un sistem complet în clasa B_2^n.

Teoremă. Sistemul de funcţii format din disjuncţie şi negaţie („+“, „–“), este un sistem complet în clasa B_2^n.

Dacă conform celor două teoreme prezentate , sistemele („·“, „–“) şi („+“, „–“) sunt complete, în acelaşi timp formează baze minimale în raport cu sistemul („+“, „·“, „–“).

Teoremă. Funcţia lui Sheffer formează în clasa B_2^n un sistem complet.

Teoremă. Funcţia lui Pierce formează în clasa B_2^n un sistem complet.

Ultimele două teoreme prezentate sunt deosebit de importante pentru aplicarea practică la sinteza circuitelor logice, permiţând folosirea unui singur tip de circuit pentru materializarea oricărei funcţii booleene. În acest context devine importantă trecerea de la FCD şi FCC la forme cu funcţii Pierce sau Sheffer.Această trecere este cunoscută şi sub denumirea de *implementare.*

1.2.4.1 Implementarea FCD şi FCC cu funcţii Pierce şi Sheffer

Prin analogie cu definirea funcţiilor elementare Pierce şi Sheffer pentru două argumente, se vor defini aceste funcţii pentru *n* argumente folosind tabelul de adevăr:

Tab.1.9 Generalizarea funcţiilor elementare Pierce şi Sheffer.

x_1	x_2	x_3	…	x_{n-1}	x_n	Pierce	Sheffer
0	0	0	…	0	0	1	1
0	0	0	…	0	1	0	1
0	0	0	…	1	0	0	1
…	…	…	…	…	…	…	…
1	1	1	…	0	1	0	1
1	1	1	…	1	0	0	1
1	1	1	…	1	1	0	0

Conform acestui tabel, rezultă:

- funcţia Pierce de *n* argumente: $P_n = x_1 \downarrow x_2 \downarrow \ldots \downarrow x_n = \overline{x_1 + x_2 + \ldots + x_n}$;
- funcţia Sheffer de *n* argumente: $S_n = x_1 \uparrow x_2 \uparrow \ldots \uparrow x_n = \overline{x_1 \cdot x_2 \cdot \ldots \cdot x_n}$.

Pentru $n = 2$ din tabelul de adevăr se obţin funcţiile Pierce şi Sheffer de două argumente, iar pentru $n = 1$ ambele funcţii se transformă în funcţia negaţie: $P_1 = S_1 = \bar{x}$.

Exprimarea prin funcţii Pierce a unei funcţii booleene se obţine considerând FCC a acesteia şi negând de două ori termenii disjunctivi:

$$f^{\mathrm{FCC}}(x_1, x_2, \ldots, x_n) = \prod_0 (x_1^{\bar{a}_1} + x_2^{\bar{a}_2} + \ldots + x_n^{\bar{a}_n}) = \prod_0 \overline{\overline{(x_1^{\bar{a}_1} + x_2^{\bar{a}_2} + \ldots + x_n^{\bar{a}_n})}}. \qquad (1.59)$$

Aplicând relaţiile De Morgan şi ţinând cont de definirea funcţiei Pierce de *n* argumente, se obţine:

$$f(x_1, x_2, \ldots, x_n) = \prod_0 \overline{\overline{x_1^{\bar{a}_1}}\,\overline{x_2^{\bar{a}_2}} \ldots \overline{x_n^{\bar{a}_n}}} = \downarrow_0 (x_1^{\bar{a}_1} \downarrow x_2^{\bar{a}_2} \downarrow \ldots \downarrow x_n^{\bar{a}_n}), \tag{1.60}$$

unde prin $\downarrow_0$ s-a notat faptul că se consideră numai *n*-uplele pe care funcţia le aplică în zero.

Concluzie. Pentru implementarea unei funcţii booleene cu funcţii Pierce se pleacă de la FCC şi se înlocuiesc operaţiile „+“ şi „·“ cu operaţia $\downarrow$.

Având în vedere cele arătate mai sus se poate stabili algoritmul implementării oricărei funcţii booleene cu funcţii NICI plecând de la tabelul de adevăr sau de la diagrama Karnaugh.

Algoritmul 1.3. Implementarea unei funcţii booleene de *n* argumente cu funcţii NICI plecând de la tabelul de adevăr se obţine astfel:

1. Se consideră *n*-uplele pe care funcţia le aplică în 0.

2. Fiecărui *n*-uplu considerat îi corespunde un termen implementat cu funcţii NICI. În aceşti termeni fiecare argument intră ca atare sau negat după cum în combinaţia respectivă are valoarea 0 sau respectiv 1.

3. Toţi termenii obţinuţi la pasul 2 se reunesc prin simbolul funcţiei NICI.

4. Excepţie de la punctul 3 face situaţia când funcţia are un singur *n*-uplu aplicat în 0. În acest caz termenul repectiv se neagă. Explicaţia rezultă din faptul că $x \downarrow x = \bar{x}$.

Tab.1.10 Tabelul de adevăr al funcţiei booleene *f* din exemplu.

x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Exemplu. Să se implementeze cu funcţii NICI funcţia booleană dată prin tabelul de adevăr alăturat.

Conform algoritmului 1.3, se obţine imediat:

$$f(x_1, x_2, x_3) = (x_1 \downarrow x_2 \downarrow x_3) \downarrow (x_1 \downarrow \bar{x}_2 \downarrow \bar{x}_3) \downarrow (\bar{x}_1 \downarrow x_2 \downarrow \bar{x}_3). \tag{1.61}$$

Dacă funcţia ar aplica în 0 numai combinaţia (1,0,1) ar rezulta:

$$f(x_1, x_2, x_3) = \overline{\bar{x}_1 \downarrow x_2 \downarrow \bar{x}_3}. \tag{1.62}$$

Exprimarea prin funcţii Sheffer a unei funcţii booleene se obţine considerând FCD şi negând de două ori termenii conjunctivi:

$$f^{\text{FCD}}(x_1, x_2, \ldots, x_n) = \sum_1 (x_1^{a_1} x_2^{a_2} \ldots x_n^{a_n}) = \sum_1 \overline{\overline{(x_1^{a_1} x_2^{a_2} \ldots x_n^{a_n})}}. \tag{1.63}$$

Aplicând relaţiile De Morgan şi ţinând cont de relaţia de definiţie a funcţiei Sheffer de *n* argumente rezultă:

$$f(x_1, x_2, \ldots, x_n) = \sum_1 \overline{(x_1^{a_1} \uparrow x_2^{a_2} \uparrow \ldots \uparrow x_n^{a_n})} = \uparrow_1 (x_1^{a_1} \uparrow x_2^{a_2} \uparrow \ldots \uparrow x_n^{a_n}). \qquad (1.64)$$

unde, prin $\uparrow_1$ s-a notat faptul că se consideră numai *n*-uplele aplicate de funcţie în 1.

Concluzie. Pentru implementarea cu funcţii Sheffer a unei funcţii booleene se pleacă de la FCD în care se înlocuiesc simbolurile „+“ şi „·“ cu simbolul $\uparrow$.

Având în vedere (1.64) şi algoritmul 1.2 se poate stabili şi în acest caz un algoritm pentru implementarea cu funcţii NUMAI a unei funcţii booleene plecând de la tabelul de adevăr sau diagramă Karnaugh.

Algoritmul 1.4. Implementarea unei funcţii booleene de *n* argumente cu funcţii NUMAI plecând de la tabelul de adevăr se obţine astfel:

1. Se consideră toate *n*-uplele pe care funcţia le aplică în 1.

2. Fiecărui *n*-uplu îi corespunde un termen implementat cu funcţii NUMAI, în care fiecare variabilă se ia ca atare sau negată după cum în combinaţia considerată are valoarea 1 sau respectiv 0.

3. Termenii obţinuţi la pasul 2 se reunesc prin simbolul funcţiei NUMAI.

4. Excepţie de la punctul 3 face cazul când funcţia aplică în 1 numai un *n*-uplu; termenul implementat cu funcţii NUMAI se neagă deoarece $x \uparrow x = \bar{x}$.

Exemplu. Să se implementeze cu funcţii NUMAI funcţia de la exemplul precedent.

Considerând combinaţiile valorilor argumentelor pe care funcţia le aplică în 1 şi ţinând cont de algoritmul 1.4, rezultă:

$$\begin{aligned} f(x_1, x_2, x_3) = {} & (\bar{x}_1 \uparrow \bar{x}_2 \uparrow x_3) \uparrow (\bar{x}_1 \uparrow x_2 \uparrow \bar{x}_3) \uparrow (x_1 \uparrow \bar{x}_2 \uparrow \bar{x}_3) \uparrow \\ & \uparrow (x_1 \uparrow x_2 \uparrow \bar{x}_3) \uparrow (x_1 \uparrow x_2 \uparrow x_3). \end{aligned} \qquad (1.65)$$

Dacă însă funcţia ar aplica în 1 numai combinaţia (0,1,0) atunci:

$$f(x_1, x_2, x_3) = \overline{\bar{x}_1 \uparrow x_2 \uparrow \bar{x}_3}. \qquad (1.66)$$