

Functii logice. Reprezentarea prin circuite logice

Functii logice

Pentru a putea intelege functiile logice care lucreaza in algebrele booleene, trebuie sa dam o definitie pentru notiunea de Algebra Bool si care sunt proprietatile acesteia.

In general, numim algebra booleana o algebra $\mathcal{A} = (A, +, \cdot, \bar{\cdot}, 0, 1)$ de tip $(2, 2, 1, 0, 0)$ care este o latice distributiva, marginita si complementata.

Ea satisface pentru operatiile de “adunare” (disjunctie logica) si “inmultire” (conjunctie logica) proprietatile de asociativitate, comutativitate (definite natural), absorbtie ($x + (xy) = x$, $x(x + y) = x$), distributivitate (si “adunarea” fata de “inmultire”, si “inmultirea” fata de “adunare”) (i.e. $(x(y + z) = xy + xz, x + (yz) = (x + y)(x + z))$), marginire ($0 = \min A$, $1 = \max A$) si complementaritate (i.e. $x + \bar{x} = 1, x\bar{x} = 0$).

Operatiile derivate ce pot fi definite sunt urmatoarele:

- Implicatia: $x \rightarrow y \Leftrightarrow \bar{x} + y$
- Diferenta: $x - y \Leftrightarrow x\bar{y}$
- Echivalenta: $x \leftrightarrow y \Leftrightarrow (x \rightarrow y)(y \rightarrow x)$
- Disjunctia exclusiva (xor): $x(+)y \Leftrightarrow (x - y) + (y - x) = x\bar{y} + \bar{x}y$

In Informatica se lucreaza cu o particularizare a algebrelor booleene la cazul $B_2 := (B_2, +, \cdot, \bar{\cdot}, 0, 1)$, astfel ca putem defini functii booleene de forma

$$f: B_2^k \rightarrow B_2^p$$

cu semnificatia ca functia f primeste ca intrare k argumente, elemente din $\{0, 1\}$, si returneaza p valori, tot din multimea $\{0, 1\}$, fiecare componenta fiind calculata in functie de o expresie booleana.

Functiile booleene pot fi definite informal (dar suficient de precis), printr-o formula sau printr-un tabel. Prima problema care apare este aceea a determinarii formulei functiei in momentul in care aceasta este descrisa intr-unul dintre celelalte doua cazuri.

Obs: Daca functia este descrisa informal, atunci i se poate determina tabelul de valori. Astfel, reducem cele doua cazuri la unul singur, si anume identificarea formulei functiei plecand de la tabelul de valori asociat acestuia.

Forma normala disjunctiva (FND) si forma normala conjunctiva (FNC)

Orice functie booleana poate fi scrisa in oricare dintre formele FND si FNC. Ele sunt definite (pentru particularizarea B_2) astfel:

$$FND(f) := \sum_{\alpha_i \in \{0,1\}, f(\alpha_1, \dots, \alpha_n) = 1, \forall i \in \{1, \dots, n\}} x_i^{\alpha_i}$$

$$FNC(f) := \prod_{\alpha_i \in \{0,1\}, f(\alpha_1, \dots, \alpha_n) = 0, \forall i \in \{1, \dots, n\}} x_i^{\bar{\alpha}_i}$$

Observam ca $FND(f)$ este o suma de produse, astfel ca fiecare produs in parte sa dea 1, iar $FNC(f)$ descrie un produs de sume, astfel ca fiecare suma in parte sa dea 0.

Aplicatii pentru determinarea FND si FNC

Fie $f: B_2^3 \rightarrow B_2^2, f(x, y, z) = (x + y, x(\bar{y} + z))$.

Sa se determine $FND(f_1), FND(f_2), FNC(f_1), FNC(f_2)$ unde $f_1(x, y, z) := x + y, f_2(x, y, z) := (\bar{y} + z)$.

Solutie

Construim tabelul de valori al functiei, care arata astfel:

x	y	z	$f1$	$\bar{y} + z$	$f2$
0	0	0	0	1	0
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	1	1	0
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	1	1	1
1	1	1	1	1	1

Vrem sa reprezentam functia f_1 in FND. Procedam astfel:

- Ne uitam pe coloana asociata functiei f_1 .
- Pentru fiecare linie ce contine valoarea 1, scriem un produs de forma xyz .
- Complementam in cadrul produsului variabilele care, pe linia curenta, au valoarea 0. Exemplu: pe linia a treia din tabel (prima linie pe care $f_1(x,y,z) = 1$), avem $(x, y, z) = (0, 1, 0)$, deci produsul curent va arata $\bar{x}y\bar{z}$ (complementez unde gasesc 0).

Deci:

$$FND(f_1) = \bar{x}y\bar{z} + \bar{x}yz + x\bar{y}\bar{z} + x\bar{y}z + xy\bar{z} + xyz$$

Vrem sa reprezentam functia f_2 in FNC. Procedam astfel:

- Ne uitam pe coloana asociata functiei f_2 .
- Pentru fiecare linie ce contine valoarea 0, scriem o suma de forma $x + y + z$.
- Complementam in cadrul produsului variabilele care, pe linia curenta, au valoarea 1. Exemplu: pe linia a treia din tabel avem $f_2(x,y,z) = 0$ si $(x, y, z) = (0, 1, 0)$, deci suma curenta va fi $x + \bar{y} + z$ (complementez acolo unde gasesc 1).

Deci:

$$FNC(f_2) = (x + y + z)(x + y + \bar{z})(x + \bar{y} + z)(x + \bar{y} + \bar{z})$$

Nu mai este necesara detalierea functiilor $FND(f_2)$ si $FNC(f_1)$.

Concluzie: folosind cele doua forme normale, observam ca putem scrie legea unei functii plecand de la orice alt mod in care aceasta a fost indicata (sau informal, sau prin tabel). Avem urmatorul rezultat:

$$f: B_2^k \rightarrow B_2^p, f \equiv FND(f) \equiv FNC(f).$$

Exercitii (functii date la examenele din anii anteriori):

1. Determinati legea unei functii care, primind ca intrare 3 valori din $\{0,1\}$, returneaza 1 daca secventa data este palindromica.
2. Determinati legea unei functii care, primind ca intrare 3 valori din $\{0,1\}$, returneaza 1 daca numarul format in baza 2 de cei 3 biti este divizibil cu 5.
3. Determinati legea unei functii care, primind ca intrare 3 valori din $\{0,1\}$, returneaza 1 daca numarul format in baza 2 de cei 3 biti este divizor al lui 8.





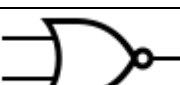
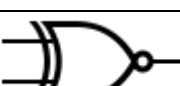

Circuite logice

Fiecare functie booleana poate fi implementata printr-un circuit logic. Studiem acum doar urmatoarele trei tipuri de implementari:

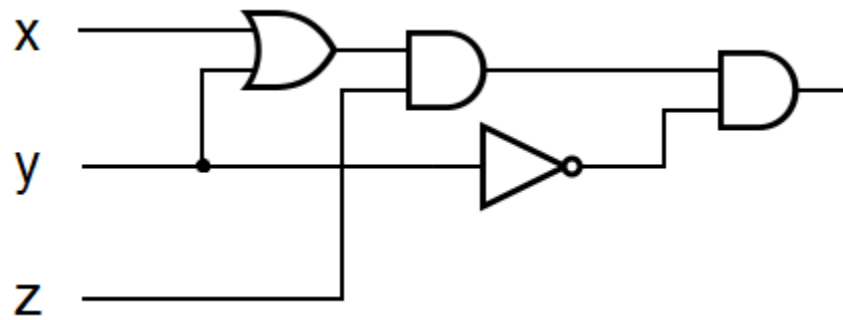
- Implementarea prin porti logice;
- Implementarea PROM;
- Implementarea cu multiplexori si multiplexori elementari.

Implementarea circuitelor prin intermediul portilor logice

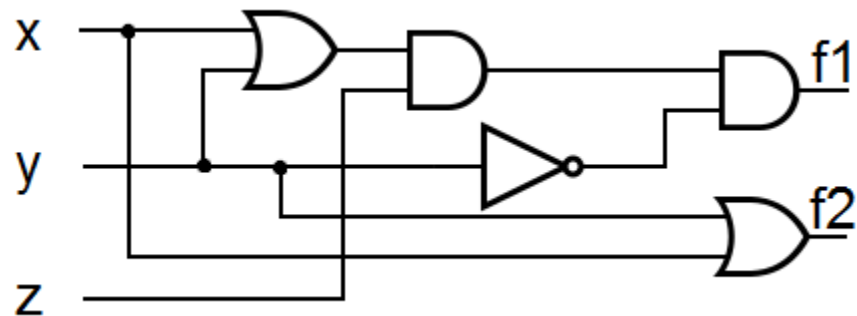
Pentru a putea implementa o functie booleana prin porti logice, trebuie sa stim care sunt acestea:

Nume poarta	Reprezentare
AND	
OR	
XOR	
NAND	
NOR	
NXOR	
NOT	

De exemplu, functia $f: B_2^3 \rightarrow B_2, f(x, y, z) = (x + y)z\bar{y}$ va fi implementata prin porti logice astfel:

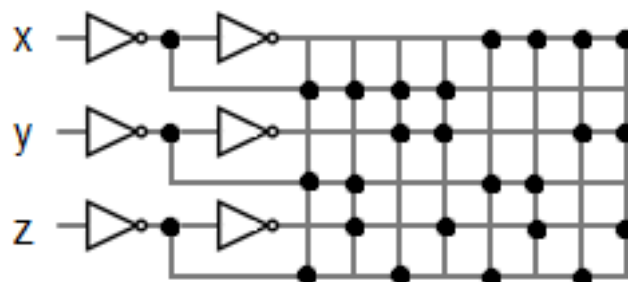


Daca exemplul ar fi fost unul mai complex, de forma $f: B_2^3 \rightarrow B_2^2, f(x, y, z) = ((x + y)z\bar{y}, x + y)$ (obs. ca am pastrat prima componenta egala cu functia implementata anterior prin circuite logice), atunci circuitul logic ar fi avut doua iesiri pentru functia f:



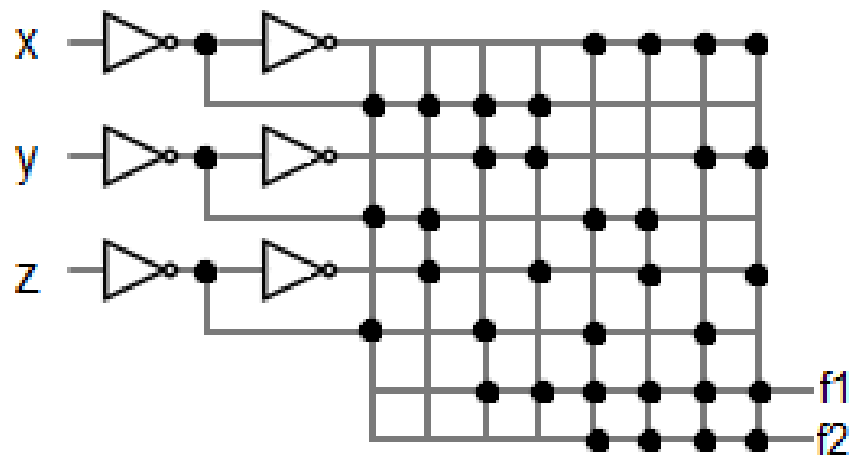
Implementarea PROM (Programmable Read-Only Memory)

(Vedeti detalii in curs) Pentru a construi un circuit PROM, se porneste de la urmatoarea diagrama: (obs. constructia ei, si anume faptul ca fiecarei variabile ii corespund cate doua linii: pe prima linie se marcheaza valorile 1, pe a doua linie se marcheaza valorile 0; ordinea in care sunt trecute corespunde ordinii lexicografice, analog tabelului de valori).



In continuare, circuitul se prelungeste in partea de jos si se marcheaza cu puncte liniile pe care functia obtine valoarea 1. Pentru $f: B_2^3 \rightarrow B_2^2, f(x, y, z) = (x + y, x(\bar{y} + z))$ (cea careia, mai sus, i-am facut tabelul logic), implementarea PROM va arata astfel: (amintim, mai jos, si tabelul)

x	y	z	$f1$	$\bar{y} + z$	$f2$
0	0	0	0	1	0
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	1	1	0
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	1	1	1
1	1	1	1	1	1



Explicatie:

- Pentru fiecare variabila am folosit de doua ori poarta logica NOT, astfel ca prima linie contine valorile "true" (1), iar pe cea de-a doua linie am pastrat valorile "false" (0).
- Completarea a fost facuta astfel: pentru x – 4 de 0, 4 de 1 (4 buline pe linia a doua, 4 buline pe prima linie); pentru y – alternant 2 de 0, 2 de 1; pentru z – alternant un 1, un 0.
- Pentru calculul functiilor am prelungit liniile (care semnifica, in circuite, fire, iar bulinele/punctele sunt intersectiile cu contact) si am marcat doar unde f1 sau f2 aveau si in tabel valoarea 1. Prima linie verticala corespunde primei intrari din tabelul de valori (toate

variabilele sunt setate pe 0, f_1 si f_2 sunt si ele 0. Alt exemplu: a treia linie verticala corespunde celei de-a treia linii din tabel – $x = 0, y = 1, z = 0, f_1 = 1$ si $f_2 = 0$).

Implementarea cu multiplexori si multiplexori elementari

Un multiplexor (multiplexer) cu selector pe n biti ($MUX_n, n \geq 1$), este un comutator de tip “many into one”, care poate conecta o intrare selectabila printr-un cod numeric la o iesire unica.

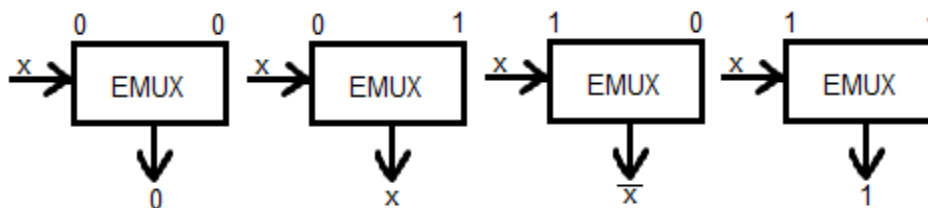
Obs: multiplexorul si PROM-ul studiat anterior sunt sisteme digitale de tip 0-DS (detalii la curs).

1. Multiplexorii elementari (EMUX). Simbolizare si iesiri furnizate

Un multiplexor elementar este un multiplexor care are doua intrari, ambele din $\{0, 1\}$; in functie de cele 4 combinatii ale acestor doua intrari, se pot primi urmatoarele semnale:

- Bitul x intra in multiplexorul cu intrarile $\{0, 0\}$, iar multiplexorul furnizeaza 0.
- Bitul x intra in multiplexorul cu intrarile $\{0, 1\}$, iar multiplexorul furnizeaza x .
- Bitul x intra in multiplexorul cu intrarile $\{1, 0\}$, iar multiplexorul furnizeaza \bar{x} .
- Bitul x intra in multiplexorul cu intrarile $\{1, 1\}$, iar multiplexorul furnizeaza 1.

Simbolizari:



2. Implementarea efectiva a unei functii booleene prin multiplexori si multiplexori elementari

Fie aceeasi functie $f: B_2^3 \rightarrow B_2^2, f(x, y, z) = (x + y, x(\bar{y} + z))$, insa pentru care vom implementa doar componenta $f_2: f_1(x, y, z) = x + y$.

Descriere:

- Functia f_1 are ca intrare 4 multiplexori elementari, valorile acestora fiind, in ordine, valorile functiei f_1 : (0, 0, 1, 1, 1, 1, 1, 1): astfel, primul multiplexor are intrarea (0, 0), al doilea (1, 1), al treilea (1, 1), al patrulea (1, 1).

- Ordinea in care bitii x, y, z "intra" in multiplexori este de la z (cel mai putin semnificativ) spre x. Astfel ca, la primul pas, variabila z este intrare pentru fiecare dintre cei 4 multiplexori:
 - z cu (0, 0) furnizeaza 0.
 - z cu (1, 1) furnizeaza 1.
 - z cu (1, 1) furnizeaza 1.
 - z cu (1, 1) furnizeaza 1.
- In urma pasului anterior am obtinut, din 4 multiplexori, doar 2, in care "intra" bitul y. Avem:
 - y cu (0, 1) furnizeaza y.
 - y cu (1, 1) furnizeaza 1.
- La ultimul pas, bitul x intra intr-un singur multiplexor ale carui intrari sunt (y, 1), ceea ce inseamna ca functia depinde exclusiv de valorile bitilor x si y (ceea ce este evident, formula de calcul fiind disjunctia logica intre x si y).

Reprezentare grafica:

