3 BAZELE LOGICE ALE CALCULATOARELOR

3.1 Algebra de comutatie

Definitie. Algebra de comutatie (algebra booleana) este o structura algebrica $A = (\mathbf{B}, \mathbf{O})$, unde $\mathbf{B} = \{0, 1\}$ este multimea constantelor de comutatie, iar $\mathbf{O} = \{\text{SI, SAU, NU}\}$ este multimea operatorilor de comutatie, definiti astfel:

$$\forall x, y \in \mathbf{B} \quad x\,SI\,y = \begin{cases} 1 & daca\ x = 1\ si\ y = 1 \\ 0 & daca\ x = 0\ sau\ y = 0 \end{cases}$$

$$\forall x, y \in \mathbf{B} \quad x\,SAU\,y = \begin{cases} 1 & daca\ x = 1\ sau\ y = 1 \\ 0 & daca\ x = 0\ si\ y = 0 \end{cases}$$

$$\forall x \in \mathbf{B} \quad NU\,x = \begin{cases} 1 & daca\ x = 0 \\ 0 & daca\ x = 1 \end{cases}$$

Pentru operatorii din **O** se pot utiliza si alte notatii echivalente, astfel:

-SI: AND, $\cap$, $\bullet$ (produs logic);

-SAU: OR, $\cup$, $+$ (suma logica);

-NU: NOT, #, $\overline{\ }$ (negatia).

In continuare in acest capitol se va utiliza ultima notatie pentru fiecare operator.

O expresie de comutatie este o combinatie a unui numar finit de variabile de comutatie, constante de comutatie si operatori de comutatie, definita astfel:

-orice constanta sau variabila de comutatie este o expresie de comutatie;

-daca t_1 si t_2 sunt expresii de comutatie atunci $t_1 \bullet t_2$, $t_1 + t_2$, $\bar{t}_1$ si (t_1) sunt expresii de comutatie.

Ordinea de evaluare a unei expresii de comutatie se bazeaza pe urmatoarele reguli:

-daca exista paranteze, evaluarea se face din interior spre exterior;

-ordinea descrescatoare a prioritatilor operatorilor este: NU (cel mai prioritar), SI, SAU (cel mai putin prioritar);

-pentru prioritati egale evaluarea se face de la stanga spre dreapta.

Proprietati ale expresiilor de comutatie

Pe baza definitie algebrei de comutatie se pot stabili o serie de proprietati utile in prelucrarea expresiilor de comutatie:

1a) $x + 0 = x$ (0 este element neutru pentru SAU)
1b) $x \bullet 1 = x$ (1 este element neutru pentru SI)
2a) $x + 1 = 1$
2b) $x \bullet 0 = 0$
3a) $x + \overline{x} = 1$
3b) $x \cdot \overline{x} = 0$
4) $\overline{\overline{x}} = x$ (dubla negatie)
5a) $x + x = x$
5b) $x \bullet x = x$
6a) $x \bullet (y + z) = x \bullet y + x \bullet z$ (distributivitatea)
6b) $x + (y \bullet z) = (x + y) \bullet (x + z)$
7a) $x + y = y + x$ (comutativitatea)
7b) $x \bullet y = y \bullet x$
8a) $(x + y) + z = x + (y + z)$ (asociativitatea)
8b) $(x \bullet y) \bullet z = x \bullet (y \bullet z)$

Proprietatile 1-8 se pot demonstra prin inductie perfecta: se dau variabilelor toate valorile posibile si se verifica rezultatul.

Exemplu. Se va demonstra proprietatea 3a) $x + \overline{x} = 1$. Astfel:

$daca \;\; x = 0 \Rightarrow 0 + \overline{0} = 0 + 1 = 1$

$daca \;\; x = 1 \Rightarrow 1 + \overline{1} = 1 + 0 = 1$

deci proprietatea este adevarata.

Pornind de la aceste proprietati se pot deduce si alte proprietati importante in prelucrarea expresiilor de comutatie, cum sunt:

9a) $x + x \bullet y = x$ (absorbtia)
9b) $x \bullet (x + y) = x$

10a) $\overline{(x_1 + x_2 + ... + x_n)} = \overline{x_1} \cdot \overline{x_2} \cdot ... \cdot \overline{x_n}$ (De Morgan)

10b) $\overline{(x_1 \cdot x_2 \cdot ... \cdot x_n)} = \overline{x_1} + \overline{x_2} + ... + \overline{x_n}$

Exemplu. Se va demonstra proprietatea 9a) $x + x \bullet y = x$. Astfel:

$$x + x \bullet y = x \bullet 1 + x \bullet y = x \bullet (1 + y) = x \bullet 1 = x$$
$$\text{(1b)} \qquad \text{(6a)} \qquad \text{(2a)} \quad \text{(1b)}$$

Deoarece costul implementarii in hardware a unei expresii de comutatie este direct proportional cu complexitatea acesteia, apare necesitatea simplificarii. Simplificarea expresiilor de comutatie se poate face fie pe baza proprietatilor enuntate mai sus, fie pe baza unor metode sistematice cum sunt metoda diagramelor Karnaugh, algoritmul Quine McCluskey si altele.

Functii de comutatie

O functie de comutatie de n variabile este o functie $f : \mathbf{B}^n \rightarrow \mathbf{B}$. Functia negata (complementara) a unei functii f, notata cu $\bar{f}$ este acea functie care are valoarea 1 / 0 pentru toate combinatiile de valori ale variabilelor pentru care functia f ia valoarea 0 / 1.

Specificarea unei functii de comutatie de n variabile se poate face in principal prin tabela de adevar sau prin expresia de comutatie. Tabela de adevar specifica valorile functiei pentru toate cele 2^n combinatii de valori ale variabilelor.

Exemplu. Se considera functia $f : \mathbf{B}^3 \rightarrow \mathbf{B}$, data prin tabela de adevar:

x	y	z	f
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

Pornind de la tabela de adevar se poate obtine expresia de comutatie a functiei astfel: expresia se poate scrie sub forma unei sume de produse, cate un termen al sumei pentru fiecare valoare 1 a functiei (deci in cazul

exemplului expresia contine trei termeni). In cadrul fiecarui termen se scrie un produs al tuturor variabilelor functiei, fiecare variabila reprezentandu-se in forma directa daca variabila are valoarea 1 in combinatia respectiva de valori din tabela sau in forma complementata (negata) daca variabila are valoarea 0. Se obtine:

$$f(x, y, z) = \bar{x} \cdot \bar{y} \cdot \bar{z} + \bar{x} \cdot \bar{y} \cdot z + x \cdot \bar{y} \cdot z$$

O astfel de expresie suma de produse in care fiecare produs contine toate variabilele in forma directa sau negata se numeste forma canonica suma de mintermeni (un mintermen este un produs al tuturor variabilelor in forma directa sau negata). Forma canonica suma de mintermeni este unica pentru o functie data, dar in general, o functie de comutatie poate fi exprimata prin mai multe expresii de comutatie echivalente. De interes este expresia minima (suma de produse), care pentru exemplul considerat se poate obtine aplicand proprietatile de mai sus, astfel:

$$f(x, y, z) = \bar{x} \cdot \bar{y} \cdot \bar{z} + \bar{x} \cdot \bar{y} \cdot z + x \cdot \bar{y} \cdot z = \bar{x} \cdot \bar{y} \cdot \bar{z} + \bar{x} \cdot \bar{y} \cdot z + \bar{x} \cdot \bar{y} \cdot z + x \cdot \bar{y} \cdot z =$$
$$= \bar{x} \cdot \bar{y} \cdot (\bar{z} + z) + (\bar{x} + x) \cdot \bar{y} \cdot z = \bar{x} \cdot \bar{y} \cdot 1 + 1 \cdot \bar{y} \cdot z = \bar{x} \cdot \bar{y} + \bar{y} \cdot z$$

Observatii.

1) Orice functie de comutatie de n variabile se poate reprezenta in mod unic printr-o expresie forma canonica suma de mintermeni.

2) Pentru n variabile exista 2^n mintermeni.

3) Pentru o combinatie de valori ale celor n variabile un singur mintermen are valoarea 1, toti ceilalti mintermeni au valoarea 0.

4) Produsul a doi mintermeni diferiti este 1.

3.2 Circuite logice combinationale

Un circuit logic primeste la intrare mai multe semnale si genereaza de asemenea mai multe semnale. Circuitele logice se pot clasifica in doua mari categorii:

-circuite logice combinationale: valorile semnalelor de iesire la un moment dat depind numai de valorile semnalelor de intrare la momentul respectiv;

-circuite logice secventiale: valorile semnalelor de iesire la un moment dat depind de valorile semnalelor de intrare la momentul respectiv

dar si de valori ale semnalelor de intrare la momente anterioare de timp (”istoria circuitului”).

Schemele logice se implementeaza cu ajutorul circuitelor logice. Pe masura dezvoltarii tehnologice au fost realizate circuite tot mai complexe, cu un numar din ce in ce mai mare de componente elementare (tranzistoare, rezistente, capacitati, trasee de conexiune,...). In functie de gradul de integrare a acestor componente in cadrul circuitelor se disting urmatoarele categorii:

-circuite SSI (Small Scale Integration), integrate pe scara redusa, cu un numar mic (zeci sau sute) de componente elementare intr-un cip. Exemple: porti logice si circuite basculante bistabile;

-circuite MSI (Medium Scale Integration), integrate pe scara medie, cu un numar mediu de componente, de ordinul cateva mii. Exemple: decodificatoare, multiplexoare, sumatoare, registre, numaratoare;

-circuite LSI (Large Scale Integration), integrate pe scara larga, cu multe componente (zeci de mii sau sute de mii) elementare intr-un singur cip. Exemple: PLA, ROM, RAM (circuitele mai vechi);

-circuite VLSI (Very Large Scale Integration), integrate pe scara foarte larga, cu un numar foarte mare (milioane sau zeci de milioane) de componente elementare. Exemple: memorii, microprocesoare, circuite speciale.

Un circuit logic combinational poate fi reprezentat astfel (fig.3.2.1):

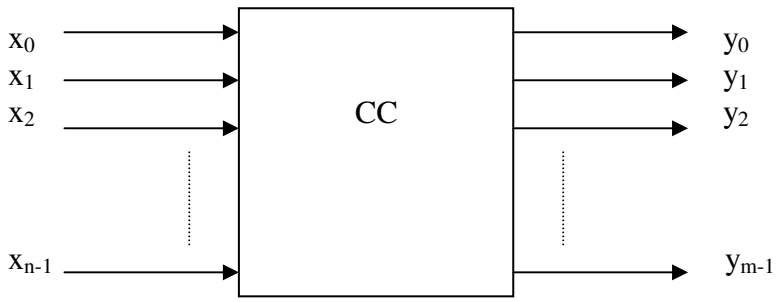


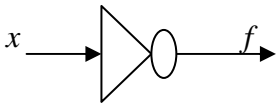
Fig.3.2.1 Circuit logic combinational.

Circuitul are *n* intrari si *m* iesiri. Functionarea unui astfel de circuit poate fi descrisa prin *m* functii de comutatie de cate *n* variabile fiecare, functii corespunzatoare iesirilor. Cele mai utilizate circuite combinationale sunt portile logice, decodificatoarele, multiplexoarele, sumatoarele, PLA-urile si memoriile ROM.

Porti logice

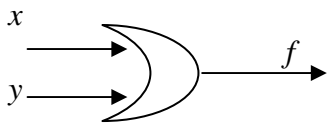
Portile logice sunt cele mai simple circuite combinationale. Acestea implementeaza direct functii de comutatie de baza. In continuare sunt prezentate pe scurt cateva din principalele tipuri de porti logice (simbolul de reprezentare si functia implementata):

Poarta inversoare



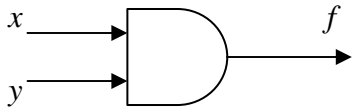
$f(x) = \overline{x}$

Poarta SAU (OR) cu 2 intrari



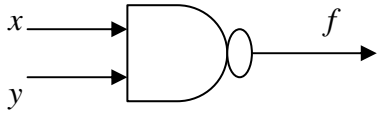
$f(x, y) = x + y$

Poarta SI (AND) cu 2 intrari



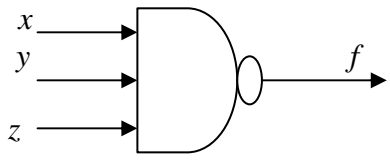
$f(x, y) = x \cdot y$

Poarta SI-NU (NAND) cu 2 intrari



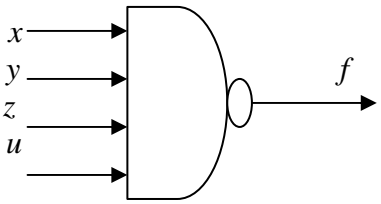
$f(x, y) = \overline{x \cdot y} = \overline{x} + \overline{y}$

Poarta SI-NU (NAND) cu 3 intrari



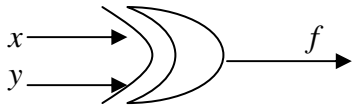
$$f(x,y,z) = \overline{x \cdot y \cdot z} = \bar{x} + \bar{y} + \bar{z}$$

Poarta SI-NU (NAND) cu 4 intrari



$$f(x,y,z,u) = \overline{x \cdot y \cdot z \cdot u} = \bar{x} + \bar{y} + \bar{z} + \bar{u}$$

Poarta SAU-EXCLUSIV (XOR) cu 2 intrari



$$f(x,y) = x \oplus y = \bar{x} \cdot y + x \cdot \bar{y}$$

Portile logice se pot utiliza pentru implementarea de functii de comutatie si realizarea de scheme logice combinationale.

Exemplu. Sa se implementeze cu porti logice functia de comutatie:

$$f(x,y,z) = \bar{x} \cdot \bar{y} \cdot z + \bar{x} \cdot y \cdot \bar{z} + x \cdot \bar{y} \cdot \bar{z} + x \cdot y \cdot z$$

Functia nu poate fi simplificata ca suma de produse si deci va fi implementata chiar expresia de comutatie data (fig.3.2.2):

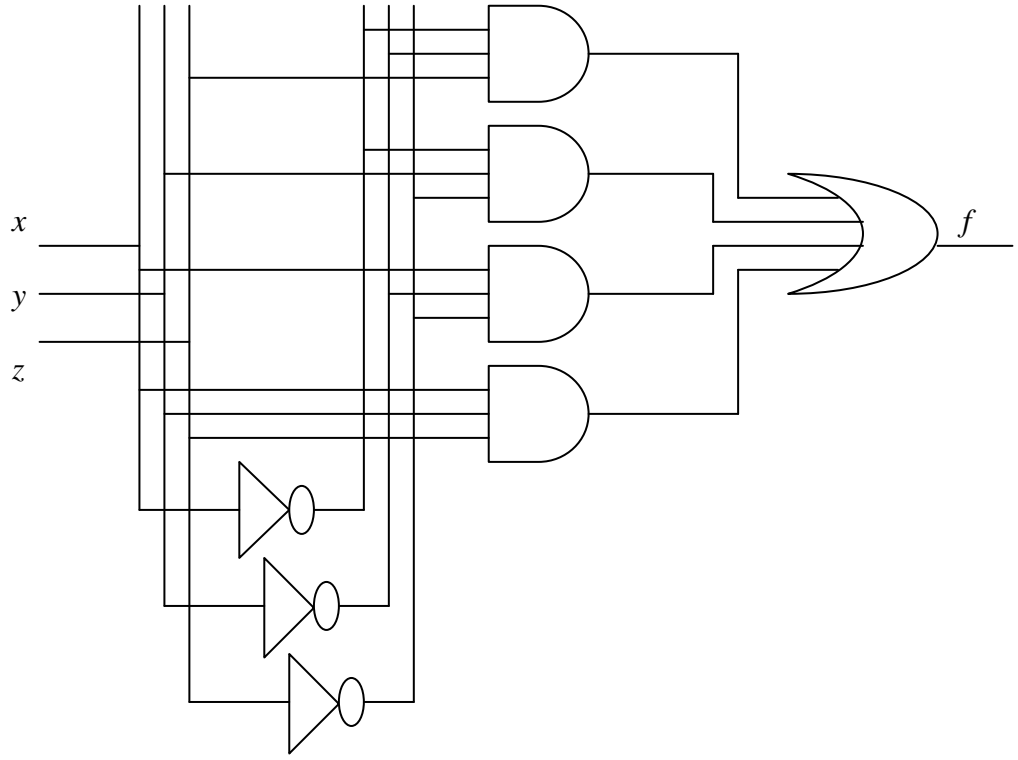


Fig.3.2.2 Implementarea functiei f cu porti logice.

Dar functia din acest exemplu se mai poate scrie sub forma:

$$f(x,y,z) = \bar{x}\cdot\bar{y}\cdot z + \bar{x}\cdot y\cdot\bar{z} + x\cdot\bar{y}\cdot\bar{z} + x\cdot y\cdot z =$$
$$= \bar{x}\cdot(\bar{y}\cdot z + y\cdot\bar{z}) + x\cdot(\bar{y}\cdot\bar{z} + y\cdot z) = \bar{x}\cdot(y\oplus z) + x\cdot\overline{(y\oplus z)} =$$
$$= x\oplus y\oplus z$$

In acest caz schema poate fi implementata cu porti SAU-EXCLUSIV (fig.3.2.3):

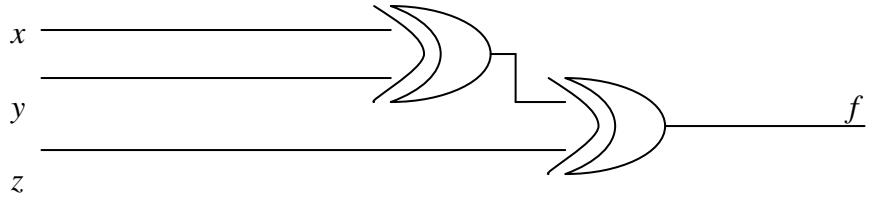


Fig.3.2.3 Implementarea functiei f cu porti SAU-Exclusiv.

Decodificatoare

Un decodificator *n:m* (fig.3.2.4) este un circuit logic combinational cu n intrari x_0, x_1, ..., x_{n-1}si $m=2^n$ iesiri y_0, y_1, ... , y_{m-1}.

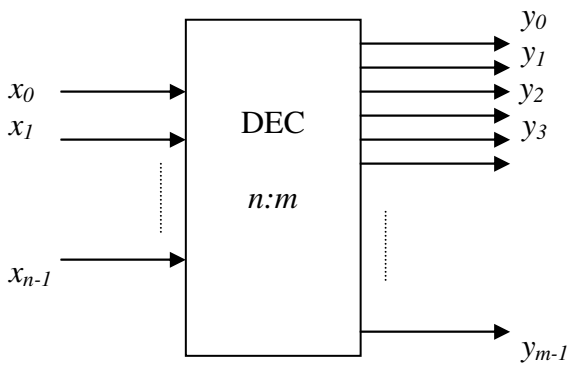


Fig.3.2.4 Decodificator.

La un moment dat este activa o singura iesire si anume iesirea avand codul (numarul de ordine) reprezentat de intrari. Functionarea decodificatorului este prezentata in tabela urmatoare.

$x_{n-1}\ x_{n-2}...x_1\ x_0$	y_{m-1}	y_{m-2}	.....	y_1	y_0
0 0 ... 0 0	0	0	...	0	1
0 0 ... 0 1	0	0	...	1	0
.....	...	...	...	...	...
1 1 ... 1 0	0	1	...	0	0
1 1 ... 1 1	1	0	...	0	0

Se observa ca pentru oricare combinatie de valori ale variabilelor de intrare x_0, x_1, ..., x_{n-1} o singura iesire este activa (are valoarea 1 logic), toate celelalte iesiri sunt inactive (au valoarea 0 logic).

Exista diferite tipuri de decodificatoare, de exemplu cu 2 intrari si 4 iesiri (decodificator 2:4), cu 3 intrari si 8 iesiri (decodificator 3:8), etc. De asemenea exista si decodificatoare avand iesirile active pe 0 (reprezentate in cadrul simbolului prin cerculete pe iesiri), deci la un moment dat o singura iesire are valoarea 0, toate celelalte iesiri au valoarea 1.

Functia principala a unui decodificator este sa sesizeze furnizarea oricarei valori binare pe cele n intrari (sa decodifice), activand iesirea corespunzatoare combinatiei respective. In continuare este prezentata o

aplicatie practica in care un decodificator 2:4 permite selectarea unui modul de memorie din patru module, pe baza a doi biti de adresa (fig.3.2.5). In acest fel se activeaza un singur modul de memorie care va realiza operatia de citire sau scriere.

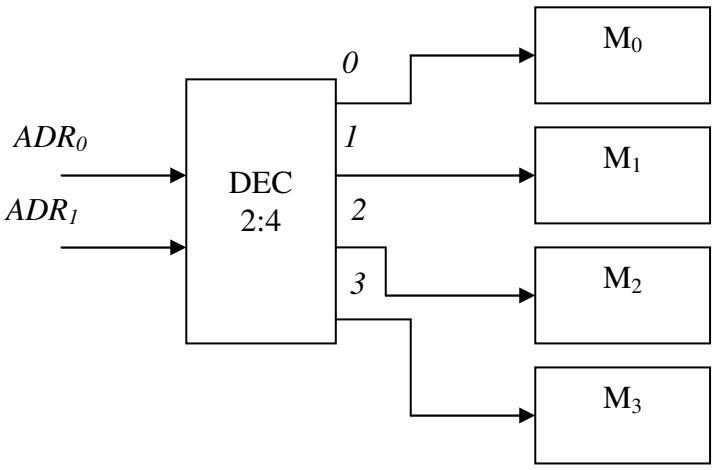


Fig.3.2.5 Selectarea modulelor de memorie.

Decodificatoarele pot fi utlizate si pentru implementarea de diferite expresii de comutatie.

Exemplu. Sa se implementeze cu decodificator functia:

$$f(x, y, z) = \overline{x} \cdot \overline{y} \cdot z + \overline{x} \cdot y \cdot \overline{z} + x \cdot \overline{y} \cdot \overline{z} + x \cdot y \cdot z$$

Este necesar un decodificator 3:8, functia avand 3 variabile (fig.3.2.6). Fiecare iesire a decodificatorului este asociata cu cate un mintermen posibil care se poate exprima prin cele trei variabile. Pentru implementarea functiei se vor selecta iesirile decodificatorului corespunzand mintermenilor functiei (este necesar ca functia sa fie exprimata ca suma de mintermeni).

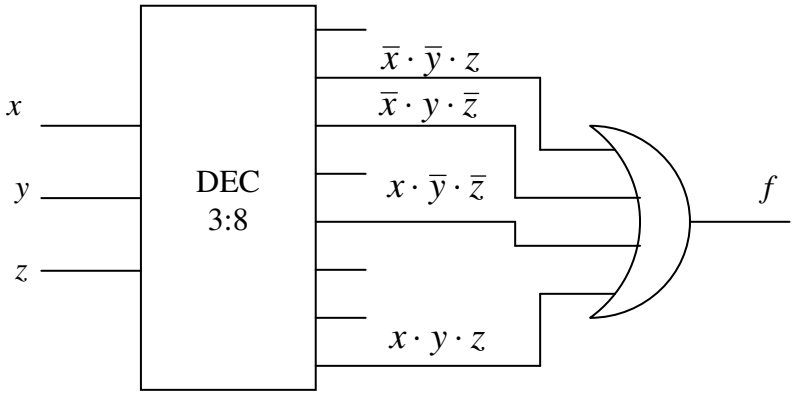


Fig.3.2.6 Implementarea functiei f cu decodificator.

Multiplexoare

Un multiplexor *m*:1(fig.3.2.7) este un circuit logic combinational cu *n* intrari de selectie (comanda) s_0, s_1, ... , s_{n-1}, $m=2^n$ intrari de date x_0, x_1, ..., x_{m-1} si o singura iesire *y* (eventual o a doua iesire, complementara, negata primei iesiri). Nivelul logic de la iesirea multiplexorului este dat de intrarea avand codul reprezentat de intrarile de selectie.

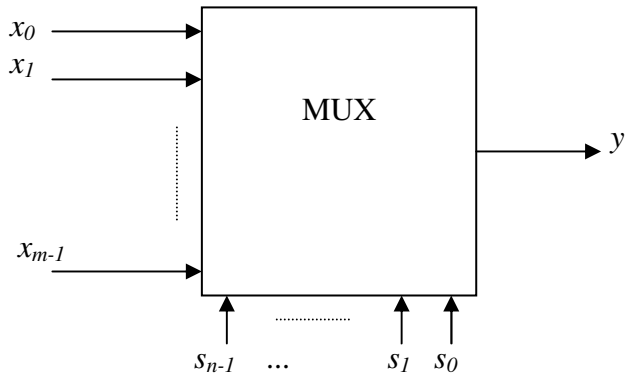


Fig.3.2.7 Multiplexor.

Functionarea unui multiplexor este prezentata in tabela urmatoare, in care s-au considerat toate combinatiile de valori ale variabilelor de selectie.

s_{n-1} .... s_1, s_0	y
0 .... 0 0	x_0
0 .... 0 1	x_1
.....	...
1 .... 1 1	x_{m-1}

Fiecare valoare a functiei de iesire este data de valoarea intrarii de date care a fost selectata prin combinatia respectiva a variabilelor de selectie.

In practica se utilizeaza diferite tipuri de multiplexoare: cu doua intrari de date si o singura intrare de selectie (multiplexor 2:1), cu patru intrari de date si doua intrari de selectie (multiplexor 4:1), cu opt intrari de date si trei intrari de selectie (multiplexor 8:1), etc.

Multiplexorul permite furnizarea unei anumite informatii din mai multe posibile, selectia realizandu-se pe baza unui cod (furnizat pe intrarile de selectie). In continuare este prezentata o aplicatie practica in care un microprocesor si un dispozitiv DMA fac acces la un modul de memorie, cu adrese diferite pe *w* biti (fig.3.2.8). La un moment dat are acces un singur

dispozitiv, iar adresa sa este furnizata memoriei prin intermediul unui bloc de *w* multiplexoare 2:1, comandate prin acelasi semnal de selectie (se presupune ca pentru *s*=0 are acces microprocesorul si pentru *s*=1 are acces modulul DMA).

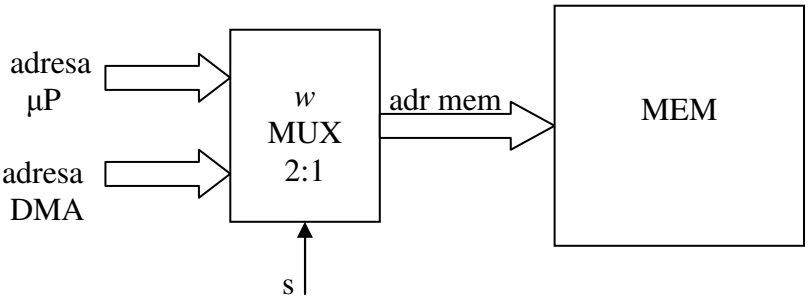


Fig.3.2.8 Multiplexarea datelor la scrierea in memorie.

Circuitele multiplexoare pot fi utilizate si pentru implementarea de diferite functii de comutatie.

Exemplu. Sa se implementeze cu multiplexor functia:

$$f(x, y, z) = \overline{x} \cdot \overline{y} \cdot z + \overline{x} \cdot y \cdot \overline{z} + x \cdot \overline{y} \cdot \overline{z} + x \cdot y \cdot z$$

Este necesar un multiplexor cu 2 intrari de selectie (numar de variabile ale functiei – 1) pe care se conecteaza doua dintre variabile (de exemplu, *y* si *z*), iar pe intrarile de date se conecteaza cea de a treia variabila in forma directa sau negata, sau eventual pot fi conectate pe anumite intrari si constante 0 sau 1 (fig.3.2.9).

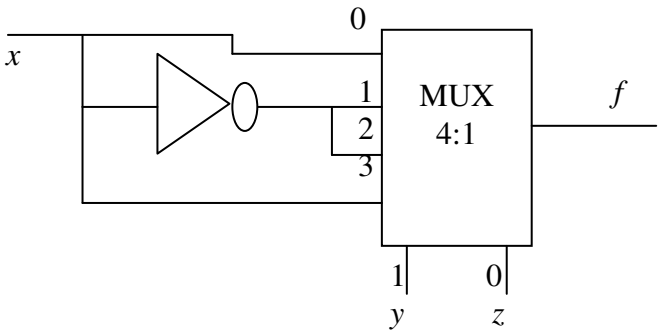


Fig.3.2.9 Implementarea functiei f cu multiplexor.

Sumatoare

Sumatorul este un circuit logic combinational care permite insumarea a doi operanzi fiecare reprezentat pe *n* biti. Sumatorul este un element de baza al oricarei unitati aritmetice - logice, iar de performantele sale depind performantele intregii unitati. In practica se utilizeaza diferite tipuri de sumatoare, insa cel mai simplu de realizat este sumatorul cu transport succesiv.

Blocul de baza al unui sumator cu transport succesiv este sumatorul elementar complet (fig3.2.10), care pentru o pereche de biti de acelasi rang din cei doi operanzi x_i, y_i si pentru transportul de la rangul precedent t_{i-1}, realizeaza suma lor si furnizeaza bitul corespunzator al sumei s_i si transportul catre rangul urmator t_i.

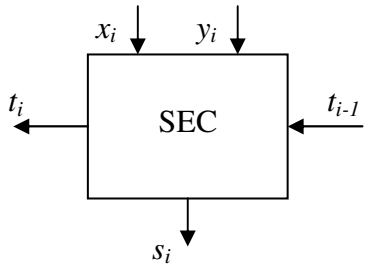


Fig.3.2.10 Sumator elementar complet.

Functionarea sumatorului elementar complet este prezentata prin tabela urmatoare:

x_i	y_i	t_{i-1}	s_i	t_i
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Din aceasta tabela se pot deduce ecuatiile de comutatie ale iesirilor sumatorului elementar complet:

$$s_i = x_i \oplus y_i \oplus t_{i-1}$$
$$t_i = x_i \cdot y_i + x_i \cdot t_{i-1} + y_i \cdot t_{i-1}$$

Pentru realizarea unui sumator pe *n* biti se conecteaza in cascada *n* astfel de sumatoare elementare complete, ca in schema urmatoare (fig.3.2.11):

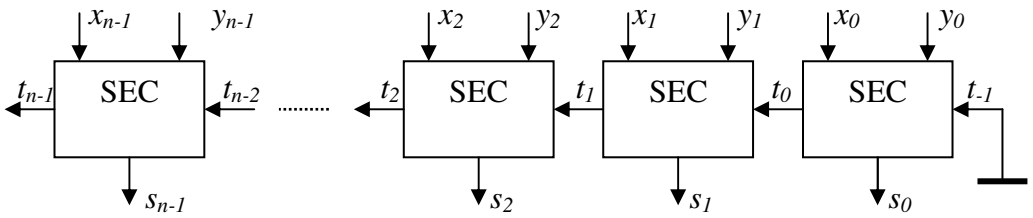


Fig.3.2.11 Sumator cu transport succesiv pe n biti.

Intrarea de transport pentru primul sumator (de rang 0) se conecteaza la 0 logic (masa electrica). Suma se obtine pe n biti: s_0, s_1, ... , s_{n-1}. Iesirea de transport de la ultimul sumator elementar complet reprezinta iesirea generala de transport a sumatorului. Se observa in schema ca un transport poate fi propagat de la un etaj la etajul urmator, de unde si denumirea acestui tip de sumator.

PLA

Un circuit PLA (Programmable Logic Array) cu *n* intrari si *m* iesiri poate fi programat pentru implementarea mai multor functii de comutatie, deci a unei scheme logice combinationale. Circuitul contine intern doua niveluri de porti logice si o serie de contacte care pot fi programate inchise sau deschise. Pe primul nivel se gasesc porti SI cu 2*n* intrari, la fiecare poarta se poate furniza, prin intermediul contactelor programabile, orice combinatie de variabile de intrare in forma directa sau negata, generand astfel orice produs al variabilelor de intrare. Pe al doilea nivel se gasesc porti SAU la care tot prin intermediul contactelor se pot furniza oricare produse de pe primul nivel, generand astfel orice suma logica a acestor produse. Iesirile portilor SAU pot fi furnizate in exteriorul circuitului PLA fie in forma directa, fie in forma complementata (negata), de asemenea prin intermediul unor contacte.

Exemplu. Sa se implementeze cu PLA urmatoarele doua functii de comutatie:

$$f(x,y,z) = \bar{x}\cdot\bar{y}\cdot\bar{z} + x\cdot\bar{y}\cdot\bar{z}$$
$$g(x,y,z) = \bar{x}\cdot\bar{y}\cdot z + \bar{x}\cdot y\cdot\bar{z} + \bar{x}\cdot y\cdot z + x\cdot\bar{y}\cdot\bar{z} + x\cdot\bar{y}\cdot z + \bar{x}\cdot y\cdot\bar{z}$$

In locul functiei *g* se va implementa:

$$\bar{g}(x,y,z) = \bar{x}\cdot\bar{y}\cdot\bar{z} + x\cdot y\cdot z$$

iar la iesire se va furniza semnalul corespunzator prin inversor, deci se genereaza de fapt functia *g*. Este necesar un PLA (fig.3.2.12) cu cel putin trei intrari (trei variabile), doua iesiri (doua functii), trei porti SI pe primul nivel (trei produse distincte la cele doua functii si doua porti SAU pe al doilea nivel (doua functii). Se obtine schema:

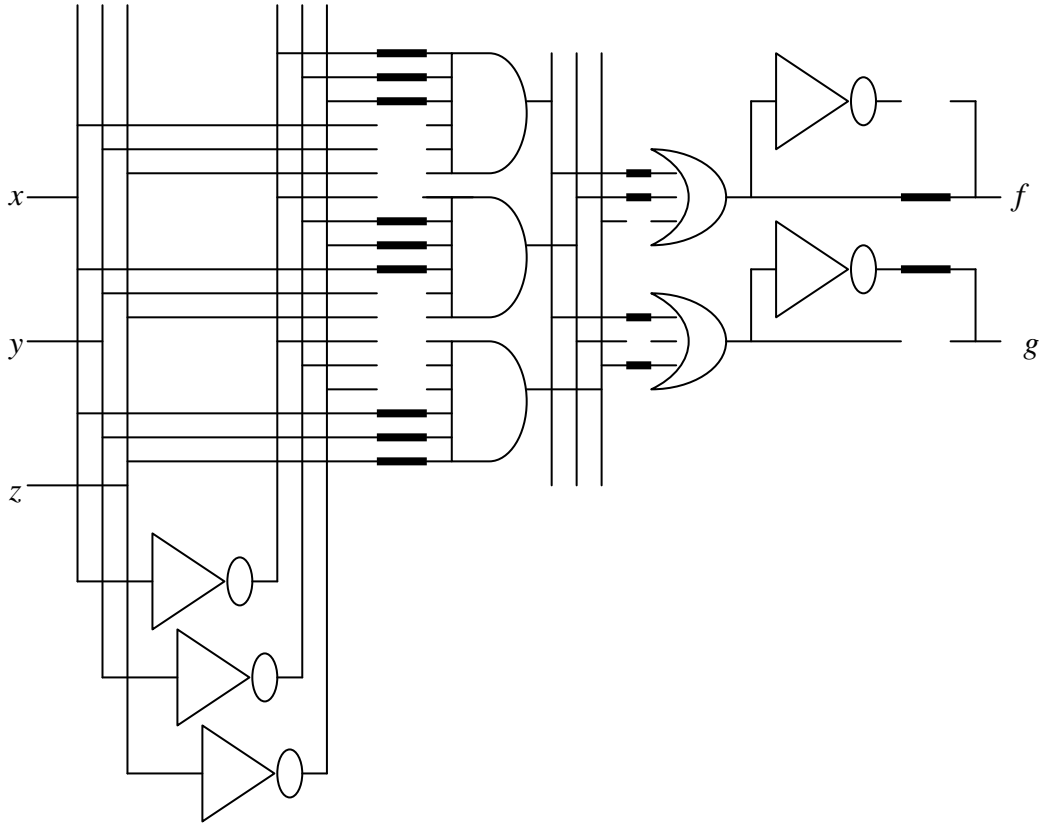


Fig.3.2.12 Implementarea functiilor f si g cu PLA.

3.3 Circuite logice secventiale

Un circuit logic secvential (CLS) cu *n* intrari si *m* iesiri (fig.3.3.1), contine o logica combinationala (CLC) si un set de elemente de memorare *Δt*. Iesirile depind de valorile furnizate la intrare la momentul respectiv de timp si de valorile stocate in elementele de memorare (starea circuitului).

Starea circuitului depinde de valori care s-au furnizat pe intrari la momente anterioare de timp (depinde de "istoria circuitului"). Starea circuitului este reprezentata de iesirile elementelor de memorare (q_0, q_1, ... , q_{p-1}). Elementele de memorare sunt comandate tot prin intermediul logicii combinationale de semnalele de control c_0, c_1, ... , c_{r-1}.

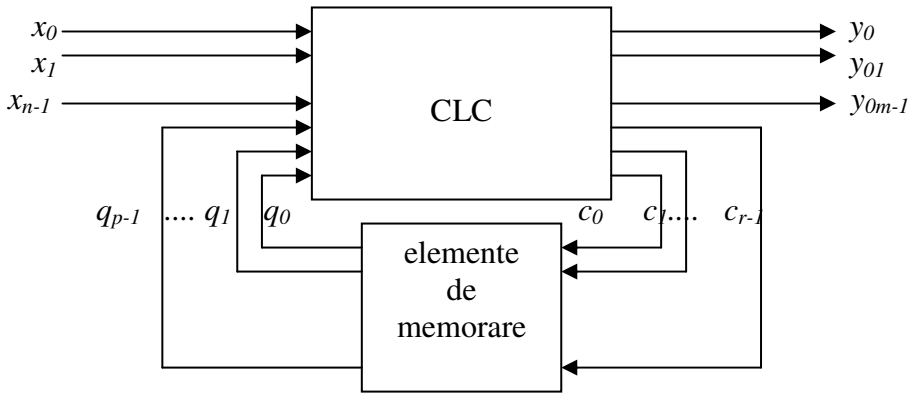


Fig.3.3.1 Circuit logic secvential.

Un CLS poate fi descris prin:

$M = (\mathbf{I}, \mathbf{O}, \mathbf{S}, f, g)$

unde:

$\mathbf{I} \subseteq \mathbf{B}^n$ este multimea valorilor semnalelor de intrare;

$\mathbf{O} \subseteq \mathbf{B}^m$ este multimea valorilor semnalelor de iesire;

$\mathbf{S} \subseteq \mathbf{B}^p$ este multimea starilor;

$f : \mathbf{S}^*\mathbf{I} \rightarrow \mathbf{O}$ este functia care determina valorile iesirilor;

$g : \mathbf{S}^*\mathbf{I} \rightarrow \mathbf{S}$ este functia care determina starea urmatoare.

Functionarea unui CLS se poate descrie prin graful starilor, un graf orientat in care fiecare nod reprezinta o stare, iar fiecare arc reprezinta o tranzitie dintr-o stare in starea urmatoare. Pe marginea fiecarui arc se inscriu valorile semnalelor de intrare si cele ale semnalelor de iesire.

Exemplu. Se considera circuitul logic secvential (fig.3.3.2) cu un singur semnal de intrare, un semnal de iesire si un element de memorare, reprezentat prin urmatorul graf:

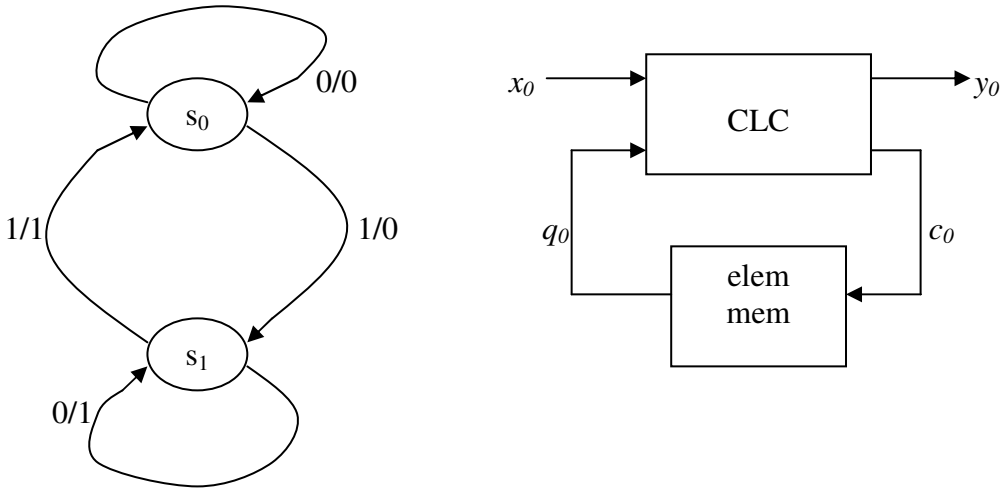


Fig.3.3.2 Circuit logic secvential simplu.

Pentru acest circuit se pot determina urmatoarele marimi:

I = B

O = B

S = $\{s_0, s_1\}$

$f(s_0,0) = 0$ $\quad g(s_0,0) = s_0$

$f(s_0,1) = 0$ $\quad g(s_0,1) = s_1$

$f(s_1,0) = 1$ $\quad g(s_1,0) = s_1$

$f(s_1,1) = 1$ $\quad g(s_1,1) = s_0$

Exista doua mari categorii de circuite logice secventiale:

-CLS sincron: schimbarea starii circuitului se produce la momente de timp bine determinate, controlate de un semnal de sincronizare (de ceas);

-CLS asincron: schimbarea starii circuitului se face la momente de timp oarecare, prin schimbarea valorilor semnalelor de intrare.

Chiar daca CLS-urile asincrone sunt mai rapide, in practica se prefera utilizarea CLS-urilor sincrone, caci acestea sunt mai stabile in functionare si exista metode sistematice de proiectare, cum este de exemplu metoda ASM (Algorithmic State Machine).

Principalele categorii de circuite logice secventiale sunt: circuitele basculante bistabile, registrele, numaratoarele, PLD-urile, memoriile.

Circuite basculante bistabile.

Circuitele basculante bistabile (CBB) sunt circuitele logice secventiale de baza, cu ajutorul lor se pot realiza circuite mai complexe. Denumirea lor provine de la faptul ca un astfel de circuit poate avea numai doua stari stabile: o stare in care furnizeaza 0 logic la iesire si o alta stare in care furnizeaza 1 logic la iesire. Astfel, prin intermediul starii, bistabilul memoreaza un bit de informatie (0 sau 1). Schimbarea starii se face cu ajutorul semnalelor de comanda. Principalele tipuri de bistabili sunt bistabilul de tip D si bistabilul de tip JK.

Bistabilul D

Un bistabil de tip D (fig.3.3.3) are o intrare de date D, o intrare de ceas CLK (Clock), doua intrari de comanda prioritare (asincrone) PR# (Preset) si CL# (Clear), care se mai numesc SET si RESET, o iesire Q si iesirea negata Q# (s-a notat prin # semnalul respectiv negat, complementat).

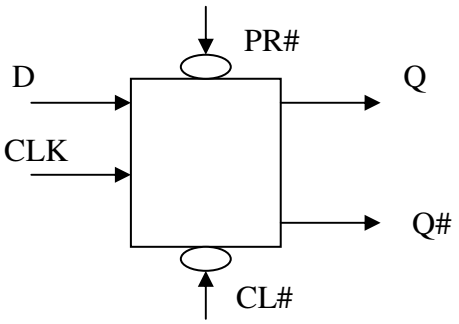


Fig.3.3.3 Bistabil de tip D.

Intrarea de ceas, asa cum a fost reprezentata in simbolul asociat, fara cerculet pe intrare, se considera activa pe front pozitiv, deci o tranzitie din 0 in 1 a semnalului de pe intrarea de ceas va schimba starea circuitului (memora o noua informatie). Functionarea circuitului este prezentata in tabela urmatoare:

D^t	PR#	CL#	Q^{t+1}
0	1	1	0
1	1	1	1
X	0	1	1
X	1	0	0
X	0	0	?

Primele doua linii din tabela corespund functionarii sincrone a circuitului, cand semnalele de comanda prioritare sunt inactive. Ultimele trei linii din tabela specifica functionarea asincrona, la activarea semnalelor de comanda prioritare. Din ultima linie a tabelei rezulta faptul ca este interzisa activarea simultana a celor doua intrari de comanda prioritare, caci in acest caz starea urmatoare a circuitului nu este determinata (?).

Bistabilul JK

Un bistabil de tip JK (fig.3.3.4) are doua intrari de date J si K, o intrare de ceas CLK (Clock), doua intrari de comanda prioritare (asincrone) PR# (Preset) si CL# (Clear), o iesire Q si iesirea negata Q#.

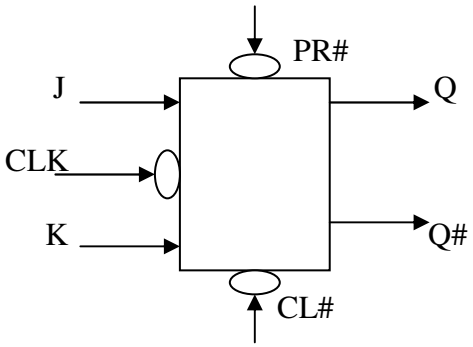


Fig.3.3.4 Bistabil de tip JK.

Intrarea de ceas, asa cum a fost reprezentata in simbolul asociat, cu cerculet pe intrare, se considera activa pe front negativ, deci o tranzitie din 1 in 0 a semnalului de pe intrarea de ceas va schimba starea circuitului (va memora o noua informatie). Functionarea circuitului este prezentata in tabela urmatoare:

J^t	K^t	PR#	CL#	Q^{t+1}
0	0	1	1	Q^t
0	1	1	1	0
1	0	1	1	1
1	1	1	1	Q^t#
X	X	0	1	1
X	X	1	0	0
X	X	0	0	?

Primele patru linii din tabela corespund functionarii sincrone a circuitului. Se observa ca pentru J=0 si K=0 starea nu se schimba, pentru J=1 si K=0 circuitul memoreaza Q=1, pentru J=0 si K=1 memoreaza Q=0, indiferent de starea anterioara, iar pentru J=1 si K=1 starea circuitului se complementeaza. Ultimele trei linii din tabela specifica functionarea asincrona, la activarea semnalelor de comanda prioritare. Din ultima linie a tabelei rezulta din nou faptul ca este interzisa activarea simultana a celor doua intrari de comanda prioritare, caci in acest caz starea urmatoare a circuitului nu este determinata (?).

Registre

Un registru este un set de bistabili comandati simultan si memoreaza *n* biti de informatie, cate un bit in fiecare bistabil. In functie de modul de introducere si extragere a informatiei memorate, exista urmatoarele tipuri de registre:

1) *Registru cu introducere paralela si extragere paralela a informatiei*. Un astfel de registru (fig.3.3.5) dispune de un semnal de comanda de incarcare (LOAD). Pe frontul activ al acestui semnal informatia (cuvantul de *n* biti) de la intrare se va incarca in registru si va fi furnizata la iesiri.

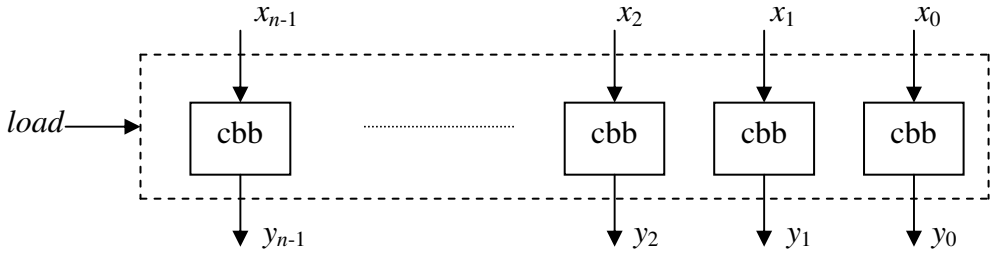


Figura 3.3.5 Registru cu incarcare paralela si extragere paralela.

Functia principala a unui astfel de registru este de memorare (temporara) paralela a informatiei.

2) *Registru cu incarcare paralela si extragere seriala a informatiei*. Circuitul (fig.3.3.6) dispune de un semnal care comanda incarcarea paralela(LOAD) si un semnal care comanda deplasarea (SHIFT). La fiecare front activ al semnalului de la intrarea SHIFT informatia se deplaseaza cu o pozitie, astfel incat la iesirea y_{n-1} este disponibil bitul urmator.

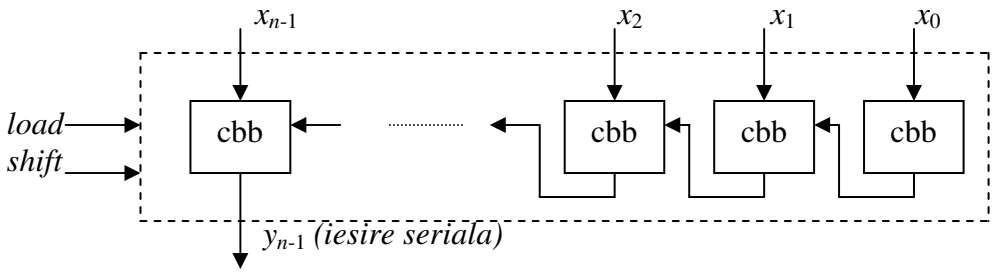


Figura 3.3.6 Registru cu incarcare paralela si extragere seriala.

Functia principala a acestui registru este conversia informatiei din format serial in format paralel (de exemplu la transmisia seriala din cadrul interfetei seriale a calculatorului).

3) *Registru cu incarcare seriala si extragere paralela a informatiei*. Registrul (fig.3.3.7) are un singur semnal de comanda SHIFT, care comanda deplasarea in registru. La fiecare front activ, informatia se deplaseaza o pozitie, iar bitul furnizat la intrarea seriala x_0 se va incarca in bistabilul corespunzator. In orice moment informatia poate fi extrasa in paralel de la iesiri.

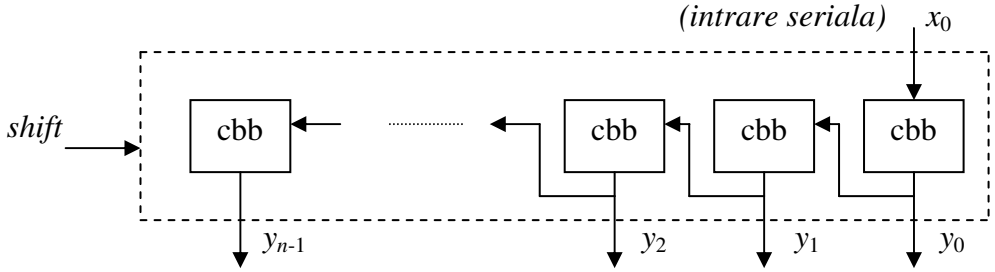


Figura 3.3.7 Registru cu incarcare seriala si extragere paralela.

Functia principala este de conversie din format serial in format paralel (exemplu de utilizare: in interfata seriala a calculatorului pentru receptie).

4) *Registru cu incarcare seriala si extragere seriala*. Este un caz particular al tipului 3, cand informatia este extrasa numai printr-o singura iesire, opusa intrarii seriale (fig.3.3.8).

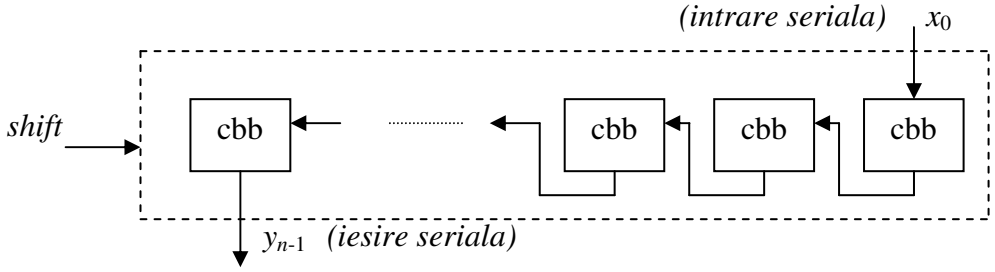


Figura 3.3.8 Registru cu incarcare seriala si extragere seriala.

Acest registru poate sa introduca o intarziere *d*t oricat de mare si precis controlata in calea unui semnal logic.

Numaratoare

Un numarator este un circuit secvential sincron care contorizeaza impulsurile aplicate pe o intrare de ceas, modificandu-si starea dupa fiecare impuls pentru a furniza valoarea numarata pana in momentul respectiv. Exista o varietate larga de numaratoare: numaratoare cu incrementare (care numara inainte), numaratoare cu decrementare (care numara inapoi), numaratoare cu presetare (care se pot initiliza cu o anumita valoare), numaratoare reversibile (care pot numara inainte sau inapoi), numaratoare modulo *n* (in principiu se poate proiecta un numarator pentru a numara modulo *n*, cu orice *n* natural).

Un numarator poate fi cel mai bine reprezentat prin graful starilor, in care nu se testeaza nici un semnal de intrare, din orice stare se va trece la sosirea impulsului de numarat intr-o anumita stare urmatoare.

Exemplu. Numarator cu incrementare modulo 8 (fig.3.3.9).

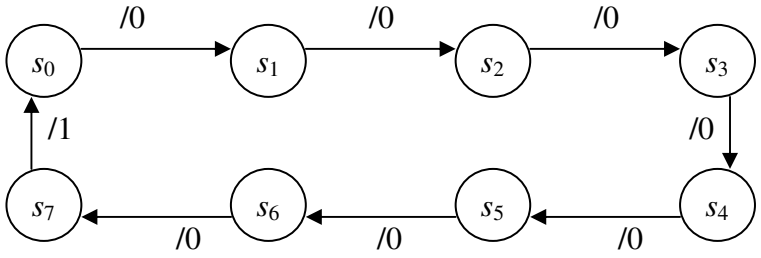


Fig.3.3.9 Numarator modulo 8.

Se poate considera la numarator ca semnal de iesire semnalul corespunzator transportului care se activeaza cand numaratorul trece de la valoarea maxima numarata la valoarea zero.

I = Ø

O = **B** (semnal de transport)

S = $\{s_0, s_1, ..., s_7\}$

$$g(s_k) = \begin{cases} s_{k+1} & daca \;\; k < 7 \\ s_0 & daca \;\; k = 7 \end{cases}$$

$$f(s_k) = \begin{cases} 0 & daca \;\; k < 7 \\ 1 & daca \;\; k = 7 \end{cases}$$

Circuite de memorie

Circuitele de memorie se impart in doua mari categorii:

-RAM (Random Access Memory) sunt memorii care permit citire si scriere; sunt volatile (la oprirea tensiunii de alimentare informatia memorata se pierde);

-ROM (Read Only Memory) sunt memorii fixe care permit numai operatia de citire, iar la intreruperea tensiunii de alimentare informatia nu se pierde.

La randul lor, memoriile RAM sunt de doua mari categorii: memorii RAM statice (SRAM) si memorii RAM dinamice (DRAM).

SRAM

Celula de baza la un circuit de memorie SRAM este asemanatoare unui circuit basculant bistabil. De aici rezulta o serie de proprietati:

-consum semnificativ de putere (curent electric): pentru memorarea informatiei celula SRAM consuma permanent curent in vederea pastrarii starii;

-celula SRAM ocupa o suprafata relativ mare pe pastila de siliciu, de aceea circuitele SRAM au in general capacitati de memorare mai mici decat circuitele DRAM de acelasi nivel tehnologic;

-circuitele SRAM se pot interfata usor intr-un sistem numeric.

Simbolul pentru reprezentarea unei memorii SRAM este prezentat in figura urmatoare (fig.3.3.10):

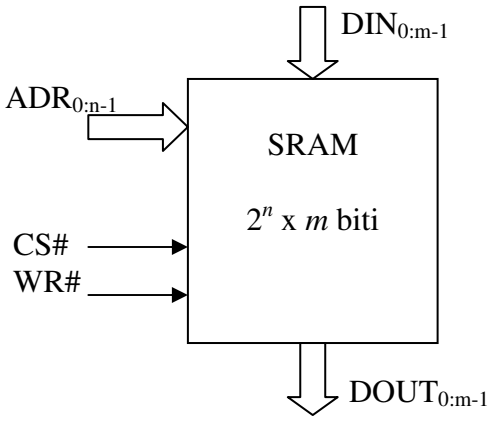


Fig.3.3.10 Circuit SRAM.

unde,

$DIN_{0:m-1}$ reprezinta liniile de date de intrare (*m* biti);

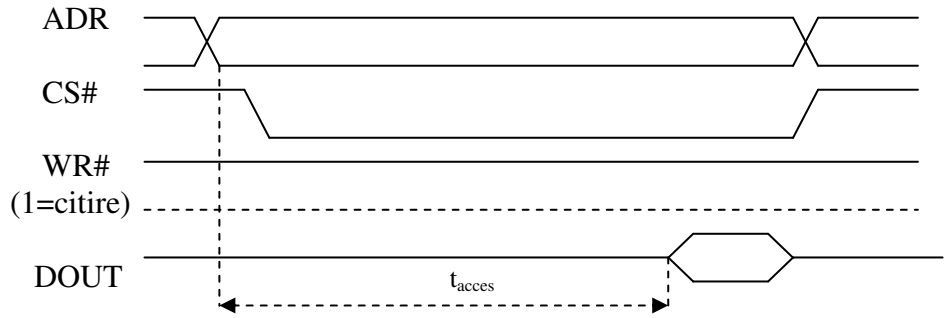
$DOUT_{0:m-1}$ sunt liniile de date de iesire;

CS# (Chip Select) este un semnal de selectie a circuitului pentru realizarea unei operatii de citire sau scriere, acest semnal fiind activ la majoritatea circuitelor pe nivel logic coborat (0 logic);

WR# (Write) este un semnal de comanda care specifica operatia de realizat, citire (WR = 1) sau scriere (WR =0).

Pentru a ilustra modul de functionare a circuitului SRAM se vor considera diagramele de semnale (fig.3.3.11) pentru o operatie de citire si o operatie de scriere.

Ciclu de citire



Ciclu de scriere

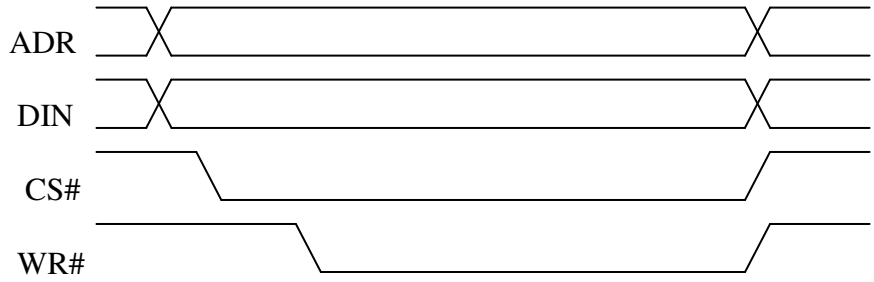


Fig.3.3.11 Citire si scriere la un circuit SRAM.

DRAM

Celula de baza a unui circuit DRAM este reprezentata printr-o capacitate realizata cu tranzistoare. Informatia memorata este sarcina electrica stocata pe aceasta capacitate. Fiind o capacitate reala, exista curent de scurgere, sarcina electrica scade in timp si de aceea este necesar ca la anumite intervale de timp sa se efectueze o operatie de reimprospatare, de

regenerare a sarcinilor electrice din circuitul de memorie ("refresh"). Caracteristicile principale ale unui circuit DRAM sunt:

-consum mic de putere (curent electric): circuitul consuma semnificativ numai in timpul operatiilor de citire, scriere si reimprospatare, in rest consumul este nesemnificativ;

-capacitatea corespunzatoare celulei de baza ocupa o suprafata mica si deci densitatea de memorare (capacitatea) unui circuit DRAM este in principiu mai mare decat acea a unui SRAM;

-interfatarea intr-un sistem numeric este mai complicata, datorita circuitelor suplimentare necesare operatiei de reimprospatare.

Simbolul pentru reprezentarea unei memorii DRAM este prezentat in figura urmatoare (fig.3.3.12):

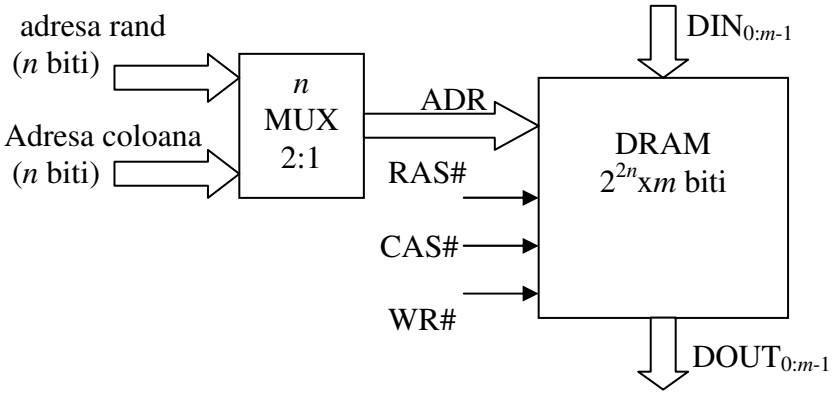


Fig.3.3.12 Circuit DRAM.

unde,

$DIN_{0:m-1}$ reprezinta liniile de date de intrare (*m* biti);

$DOUT_{0:m-1}$ sunt liniile de date de iesire;

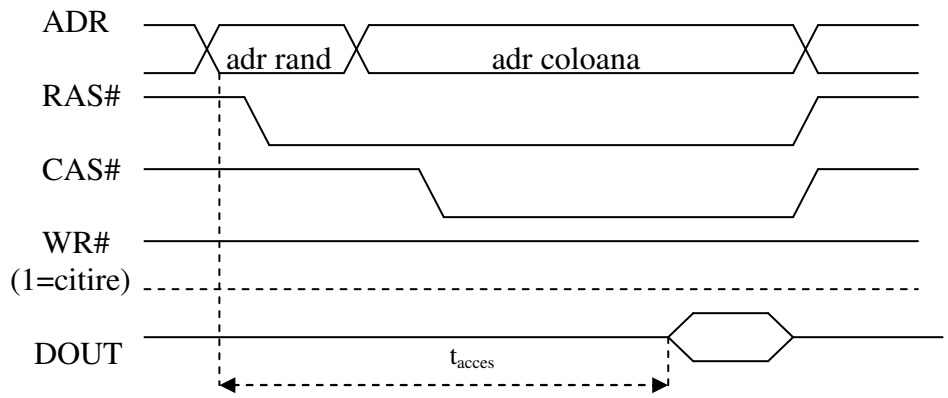
-circuitul are doua semnale de selectie RAS# (Row Adresse Strobe) semnal de selectie a adresei de rand si CAS# (Column Adresse Strobe) semnal de selectie a adresei de coloana, aceste semnale fiind active la majoritatea circuitelor pe nivel logic coborat (0 logic);

WR# (Write) este un semnal de comanda care specifica operatia de realizat, citire (WR# = 1) sau scriere (WR# =0).

De remarcat faptul ca multiplexarea adresei de rand cu adresa de coloana se realizeaza extern, printr-un bloc de multiplexoare 2:1.

Pentru a ilustra modul de functionare a circuitului DRAM se vor considera diagramele de semnale pentru o operatie de citire si o operatie de scriere (fig.3.3.13).

Ciclu de citire



Ciclu de scriere

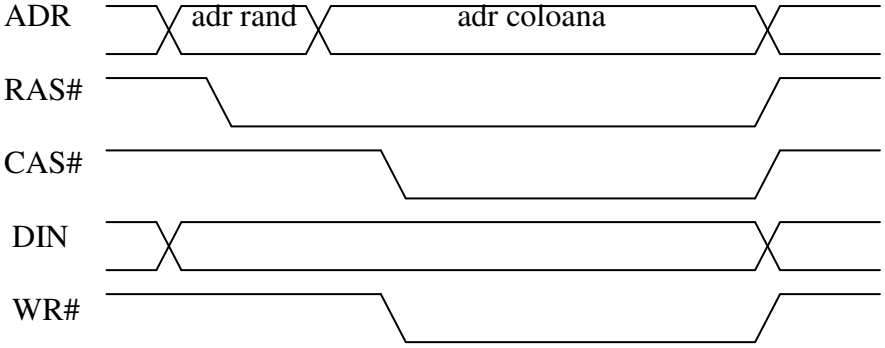


Fig.3.3.13 Citire si scriere la un circuit DRAM.

Exista diferite tipuri de memorii DRAM:

FPM DRAM (Fast Page Mode DRAM) este o memorie rapida in mod pagina, in raport cu DRAM conventional. Timpul de acces pentru al doilea cuvant si urmatoarele devine mai scurt prin faptul ca adresa de rand se transmite o singura data la inceputul ciclului si apoi se transmit numai adrese de coloana.

EDO DRAM (Extended Data Output DRAM) este o memorie la care ciclul de citire este mai scurt. Circuitul este compatibil la pini cu FPM DRAM.

SDRAM (Synchronous DRAM) opereaza la frecvente mai mari decat EDO DRAM, in sincronism cu un ceas de intrare. Controlul circuitului se face prin comenzi codificate cu ajutorul semnalelor de intrare de comanda (exemple de comenzi: activare, citire, scriere, preincarcare,...). Intern

circuitul este organizat pe mai multe bancuri, controlul realizandu-se la nivel de banc.

DDR SDRAM (Double Data Rate SDRAM) utilizeaza doua faze ale aceluiasi semnal de ceas de sincronizare. Furnizeaza date citite pe ambele fronturi ale ceasului, dublandu-si astfel rata de iesire (posibil datorita controlului la nivel de banc). Suporta numai modul rafala (”burst”).

Cu circuite de memorie au fost realizate placute care pot fi utilizate direct intr-un calculator:

SIMM (Single Inline Memory Module) cu 72 de pini, care se pot instala in perechi;

DIMM (Dual Inline Memory Module) pentru sistemele incepand cu Pentium II, avand 168 de pini si capacitate sporita.

Memorii ROM

Memoriile ROM (Read Only Memory) permit numai operatii de citire (fig.3.3.14). Sunt memorii nevolatile, deci la intreruperea tensiunii de alimentare informatia se pastreaza, aceasta fiind din nou disponibila la reluarea alimentarii.

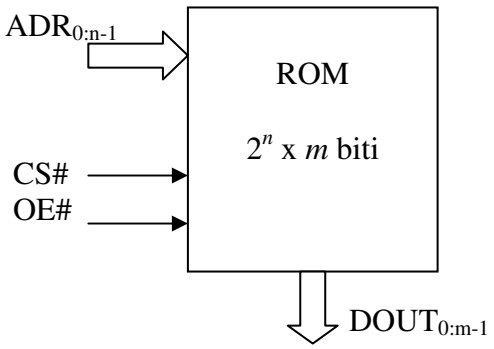


Fig.3.3.14 Memorie ROM.

Circuitele de memorie ROM sunt circuite logice combinationale. Un astfel de circuit consta dintr-o matrice de contacte, care unele sunt deschise, iar celelalte sunt inchise (fig.3.3.15). Pentru exemplificare se considera in continuare o memorie ROM cu 4 cuvinte de 4 biti.

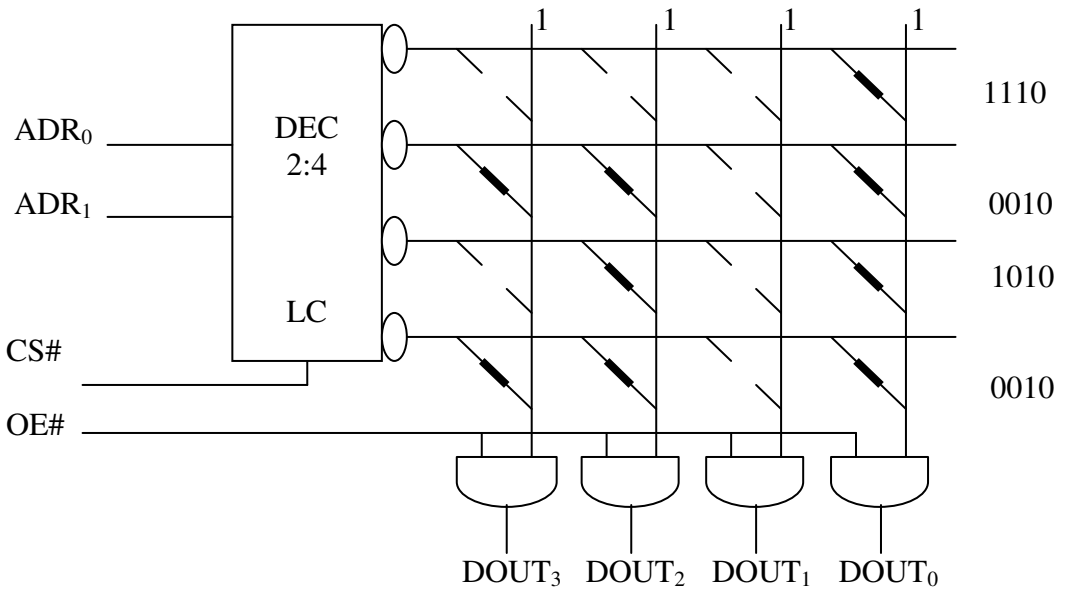


Fig.3.3.15 Structura interna a unei memorii ROM cu 4 cuvinte de 4 biti.

Intern circuitul contine un decodificator pentru biti de adresa. Iesirile decodificatorului sunt active pe nivel logic coborat (cerculete pe iesiri). Traseele verticale sunt conectate la 1 logic. Pentru o anumita adresa, trece in 0 numai iesirea corespunzatoare a decodificatorului, celelalte iesiri sunt inactive (1 logic si nu influenteaza liniile verticale). Liniile verticale care au contacte inchise fata de linia orizontala activa, vor fi ”trase” la 0 logic (deci in aceste pozitii s-au memorat biti 0), iar liniile verticale avand contacte deschise raman neinfluentate, deci la 1 logic (in aceste pozitii s-au memorat biti 1). In cadrul schemei sunt specificate valorile memorate de ROM.

Memoriile ROM se pot clasifica in urmatoarele categorii:

-ROM propriu-zis: continutul memoriei este introdus de la fabricatie, pentru produse in serie mare;

-PROM (Programmable ROM): continutul este introdus o singura data de catre utilizator cu ajutorul unui dispozitiv numit programator de PROM-uri;

-EPROM (Erasable PROM): continutul memoriei poate fi introdus si sters de mai multe ori. Stergerea se face cu ajutorul unei lampi cu ultraviolete, iar programarea cu ajutorul programatorului de PROM-uri;

-EAPROM (Electricaly Alterable PROM) si EEPROM (Electricaly Eraseble PROM) pot fi sterse electric si reprogramate, circuitele EEPROM chiar fara a fi scoase din dispozitivul in care functioneaza;

-memoria FLASH este asemanatoare cu EEPROM, insa are capacitate mult mai mare, iar stergerea si reprogramarea se face pe blocuri.

PGA

Circuitele PGA (Programmable Gate Array) sunt circuite de mare densitate care se pot configura de catre utilizator. Schemele sunt editate pe calculator, cu ajutorul unui editor de scheme, sunt prelucrate si convertite apoi in informatii de codificare, care sunt furnizate direct sau prin intermediul unei memorii PROM circuitului PGA. Acesta este astfel configurat sa contina chiar schemele logice editate. Utilizarea circuitelor PGA face ca activitatea de proiectare si testare sa devina deosebit de comoda, iar intervalul de timp intre doua variante succesive ale unui proiect sa scada foarte mult. In continuare se va considera familia de circuite PGA de la firma Xilinx, numite circuite LCA (Logic cell Array). Un astfel de circuit (fig.3.3.16) contine o matrice de blocuri logice interne numite CLB-uri (Configurable Logic Block), avand pe granita blocuri de intrare / iesire (IOB – Input Output Block), toate interconectate.

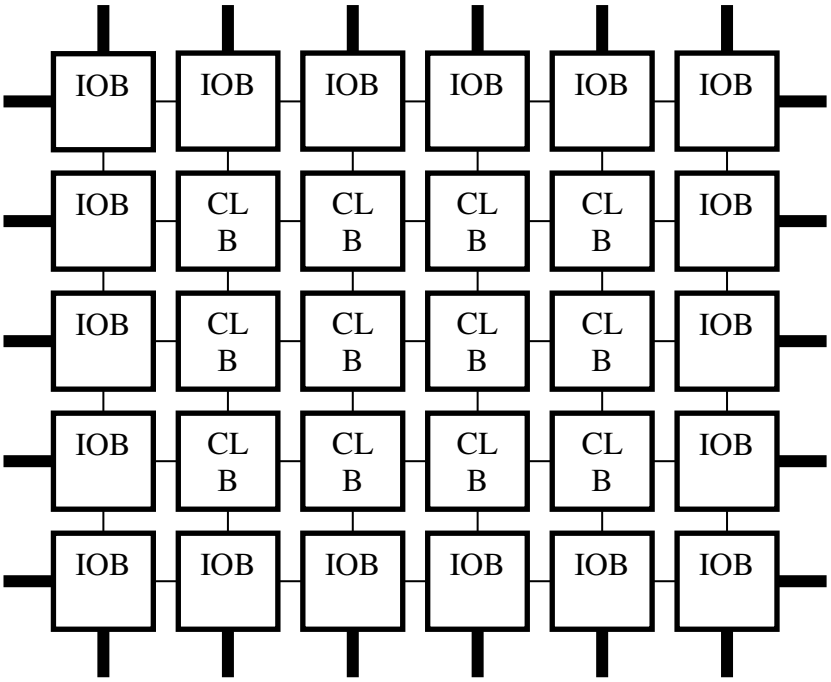


Fig.3.3.16 Structura interna a unui circuit LCA.

Cu linii ingrosate s-au reprezentat pinii externi ai circuitului. Blocurile CLB contin circuite basculante bistabile si logica combinationala, pentru implementarea atat a schemelor logice combinationale cat si a celor secventiale. Blocurile IOB permit transferul semnalelor de intrare, iesire sau bidirectionale si contin logica pentru stabilirea tipului de iesire: ieisre normala, iesire ”open colector”, iesire ”three state”, etc. Toate blocurile interne dispun de semnale comune de initializare (RESET) si sincronizare (CLOCK).