Cursul 3

2.3.1 Minimizarea funcţiilor logice folosind metoda de minimizare Quine-Mc Cluskey

Dacă funcţia care urmează a fi minimizată depinde de mai mult de 5 variabile, aplicarea metodei Veitch-Karnaugh devine greoaie. Pentru minimizarea acestor funcţii este mai utilă metoda Quine-Mc Cluskey. În cadrul acestei metode vor fi parcurse următoarele 2 etape :

- se determină mulţimea implicanţilor primi
- se determină acoperirea minimă a funcţiei.

Pentru determinarea mulţimii implicanţilor primi vom folosi următorul algoritm :

- funcţia care urmează a fi minimizată va fi scrisă în formă canonică disjunctivă. Fiecare produs logic standard va fi reprezentat printr-un număr binar ;
- produsele logice standard vor fi ordonate în grupe în funcţie de numărul variabilelor nenegate din expresia lor, fiecărui produs asociindu-se câte un număr binar care corespunde valorii sale zecimale;
- se determină implicanţii primi prin comparaţii succesive ale produselor logice.

Pentru determinarea acoperirii minime a funcţiei se completează un tabel al implicanţilor primi pe linii fiind trecuţi implicanţii primi ai funcţiei iar pe coloane produsele logice standard din expresia canonică a funcţiei care urmează să fie minimizată.Pentru a obţine expresia minimă a funcţiei vor fi parcurşi următorii paşi:

- în tabelul implicanţilor primi se caută implicanţii primi esenţiali aceştia fiind cei care acoperă un produs logic standard neacoperit de alţi implicanţi;
- dintre implicanţii primi rămaşi vor fi luaţi în considerare în forma minimă a funcţiei doar aceia care asigură o acoperire mai bună a produselor logice standard rămase neacoperite după determinarea implicanţilor primi esenţiali.

În cazul în care funcţia este incomplet definită, produsele logice standard pentru care funcţia este nedefinită vor fi luate în considerare doar în prima parte a algoritmului respectiv în cea de determinare a implicanţilor primi.
Pentru ilustrarea metodei de minimizare ne vom folosi de un exemplu concret.

Vom considera funcţia f definită ca în diagrama Veitch-Karnaugh din figura 2.9. Pornind de la această diagramă vom scrie funcţia în formă canonică disjunctivă.Pentru funcţia f aceasta este :

$$f(a,b,c,d)= \sum p_i \quad \text{cu } i \in K \text{ iar combinaţii indiferente } p_j \text{ cu } j\in N.$$

1	0	0	0
1	1	1	0
*	0	0	1
1	*	0	*

Figura 2.9

K = {0,4,5,7,8,14} , N = {9,10,12}

Având în vedere că deja a fost făcută asocierea produs – număr binar vom trece la completarea tabelului subcuburilor 0 dimensionale ca cel din figura 2.10

TABELUL SUBCUBURILOR 0 DIMENSIONALE

GRUPA	INDICI	REPREZ. BIN.	
0	0	0000	√
1	4	0100	√
	8	1000	√
2	5	0101	√
	9	1001	√
	10	1010	√
	12	1100	√
3	7	0111	√
	14	1110	√

Figura 2.10

În urma comparării succesive ale produselor logice din tabelul din figura 2.10 s-a obţinut cel din figura 2.11. Datorită faptului că toate produsele au intrat în combinaţiile din tabelul subcuburilor 1 dimensionale nu avem deocamdată nici un implicant prim.

TABELUL SUBCUBURILOR 1 DIMENSIONALE

GRUPA	INDICI	REPREZ. BIN.	
0	0,4	0*00	√
0	0,8	*000	√
1	4,5	010*	
	4,12	*100	√
	8,9	100*	
	8,10	10*0	√
	8,12	1*00	√
2	5,7	01*1	
1	10,14	1*10	√
	12,14	11*0	√

Figura 2.11

Din nou folosindu-ne de aceleaşi comparaţii succesive obţinem tabelul din figura 2.12. De această dată , în tabloul subcuburilor 1 dimensionale prin comparare au rămas ca implicanţi primi perechile (4,5), (5,7) respectiv (8,9).

TABELUL SUBCUBURILOR 2 DIMENSIONALE

GRUPA	INDICI	REPREZ. BIN.
0	0,4,8,12	**00
1	8,10,12,14	1**0

Figura 2.12

Din ultimul tabel nu mai pot fi făcute alte grupări, deci vom trece la completarea tabelului implicanţilor primi.

Nr crt.	Implicanţi primi	Indici	Produse standard					
			p_0	p_4	p_5	p_7	p_8	p_{14}
1	010*	4,5		*	*			
2	01*1	5,7			*	*		
3	100*	8,9					*	
4	**00	0,4,8,12	*	*			*	
5	1**0	8,10,12,14					*	*

Figura 2.13

În tabelul implicanţilor primi observăm că produsele logice p_0 şi p_{14} sunt acoperite doar de implicanţii primi cu indicii (0,4,8,12) respectiv (8,10,12,14).Şi produsul p_7 e acoperit doar de implicantul prim cu indicii (5,7). Din acest motiv aceştia sunt implicanţi primi esenţiali. Ei mai acoperă totodată şi produsele p_4 , p_8 p_5.Având în vedere că au fost acoperite toate produsele logice rezultă că am obţinut forma minimă a funcţiei care este:

$$f = \bar{a}bd + \bar{c}\bar{d} + a\bar{d}$$

Scrierea formei normale disjunctive din tabel se face în felul următor : se trec în sumă doar implicanţii primi esenţiali (şi eventual alţi implicanţi primi dacă cei esenţiali nu acoperă tot) trecându-se doar acele variabile care au valoarea 0 (în formă negată) respectiv 1(în formă nenegată).Dacă avem * înseamnă că variabila respectivă va lipsi în forma normală disjunctivă din produsul respectiv .

CAP. III NOŢIUNI DE TEORIA AUTOMATELOR CU STĂRI FINITE

3.1 Introducere. Definiţii

Circuitele de comutaţie constituie componentele de bază în proiectarea echipamentelor de conducere moderne. Studiul sistemelor digitale realizate cu aceste circuite se bazează pe modelul lor matematic, automatul finit, care face obiectul teoriei automatelor cu stări finite.

<u>Observaţie:</u> notiunea de automat finit, fiind o noţiune abstractă, se aplică atât sistemelor fizice cât şi celor informaţionale.

3.1.1. Moduri de reprezentare a automatelor finite

Un automat finit interacţionează cu mediul prin aceea că la un anumit moment "t" i se aplică un semnal de intrare, iar ca răspuns el oferă la momentul "t+Δt" un semnal de ieşire.

Se numeşte automat finit un cvintuplu ordonat

A=(U,X,Y,f,g)

în care U,X,Y,f si g sunt:

U – mulţimea finită a semnalelor de intrare;

X – mulţimea finită a stărilor; un element $x_i \in X$ poartă denumirea de stare;

Y – mulţimea finită a semnalelor de ieşire;

$f : X \times U \rightarrow X$ poartă numele de funcţie de tranziţie şi precizează starea în care ajunge automatul în cazul aplicării unei intrări (starea viitoare);

$g : X \times U \rightarrow Y$ poartă numele de funcţie de ieşire şi precizează ieşirea pe care o va "oferi" automatul în cazul aplicării unei intrări (ieşirea viitoare).

Fizic, automatul finit A definit mai sus poate fi interpretat ca un dispozitiv a cărui intrare, ieşire şi stare la momentul “t” sunt notate cu u(t), y(t) si x(t) (fig.3.1). Aceste variabile sunt definite numai pentru valori discrete, prin convenţie numere întregi, ale lui “t” şi primesc valori în mulţimile U, Y şi X. Aplicând la intrarea automatului A o secvenţă de intrare de lungime arbitrară “p” se va obţine o secvenţă de stări şi o secvenţă de ieşiri de aceeaşi lungime.

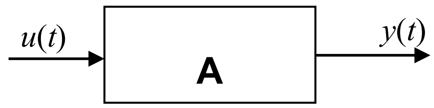


Figura 3.1

În principiu se consideră că automatele finite sunt de tip Mealy şi de tip Moore. Definiţia automatului de tip Mealy a fost dată mai sus. Automatele de tip Moore diferă de cele de tip Mealy prin aceea că funcţia de ieşire precizează, pentru automatele Moore, ieşirea pe care o oferă automatul funcţie de starea în care se află, adică

$$g : X \to Y$$

Studiul automatelor finite se face în general pe reprezentări ale acestora. Cele mai utilizate reprezentări sunt algoritmice, prin grafuri, prin tabele sau prin organigrame funcţionale. Aceste moduri de reprezentare vor fi ilustrate pentru automatele A1 de tip Mealy si A2 de tip Moore, definite în continuare:

$$U = \{u_1, u_2\} \qquad U = \{u_1, u_2\}$$
$$X = \{x_1, x_2, x_3\} \qquad X = \{x_1, x_2, x_3, x_4\}$$
$$Y = \{y_1, y_2\} \qquad Y = \{y_1, y_2\}$$

$$\begin{array}{ll} f(x_1,u_1) = x_2 & g(x_1,u_1) = y_1 \\ f(x_1,u_2) = x_3 & g(x_1,u_2) = y_2 \\ f(x_2,u_1) = x_1 & g(x_2,u_1) = y_2 \\ f(x_2,u_2) = x_3 & g(x_2,u_2) = y_1 \\ f(x_3,u_1) = x_2 & g(x_3,u_1) = y_2 \\ f(x_3,u_2) = x_3 & g(x_3,u_2) = y_1 \end{array} \qquad \begin{array}{ll} f(x_1,u_1) = x_2 & f(x_4,u_1) = x_3 \\ f(x_1,u_2) = x_3 & f(x_4,u_2) = x_1 \\ f(x_2,u_1) = x_4 & g(x_1) = y_1 \\ f(x_2,u_2) = x_3 & g(x_2) = y_2 \\ f(x_3,u_1) = x_2 & g(x_3) = y_2 \\ f(x_3,u_2) = x_4 & g(x_4) = y_1 \end{array}$$

La reprezentarea automatelor finite prin grafuri se respectă următoarele reguli:

- fiecărei stări $x_i \in X$ i se acordă un nod din graf;

- fiecărei tranziţii din starea prezentă x_i în starea viitoare x_j i se asociază un arc care uneşte nodurile corespunzătoare.

În cazul unui automat de tip Mealy fiecărei tranziţii i se asociază ieşirea corespunzătoare, iar în cazul unui automat de tip Moore ieşirea este asociată intrării. În fig. 3.2 este reprezentat graful automatului A1 de tip Mealy, iar în fig. 3.3 graful automatului A2 de tip Moore.

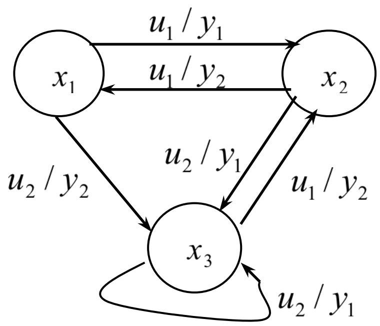
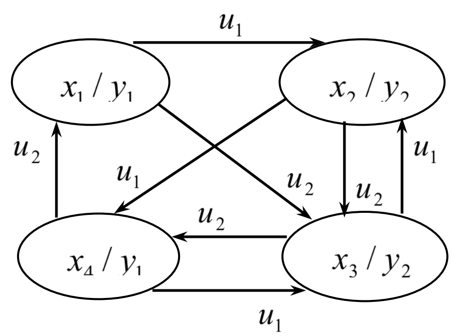


Figura 3.2



Conversia unui automat Mealy într-un automat Moore se face înlocuind fiecare stare a automatului Mealy cu atâtea stări câte ieşiri diferite sunt asociate tranziţiilor care intră în acea stare.

Conversia unui automat Moore în automat Mealy se face asociind fiecărei tranziţii din automatul Moore ieşirea corespunzătoare stării la care duce tranziţia.

Aplicând aceste reguli automatelor reprezentate în fig. 3.2 şi fig. 3.3 se obţin automatele din fig. 3.4 şi respectiv fig. 3.5.

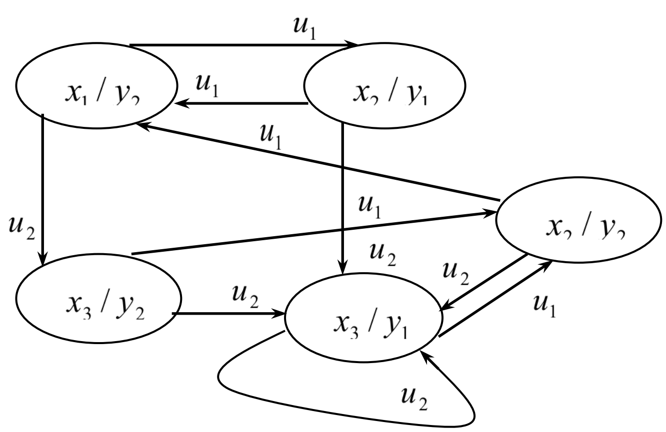
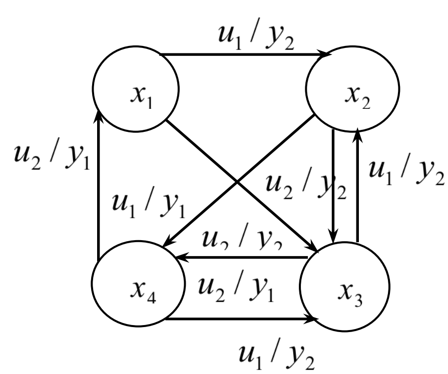


Figura 3.4



La reprezentarea automatelor prin tabele, liniile tabelului corespund stărilor prezente ale automatului, iar coloanele corespund semnalelor de intrare. Dacă x_i este o stare a automatului, iar u_j unul din semnalele de intrare, la intersecţia liniei "i" cu coloana "j" în tabel se trece funcţia de tranziţie $f(x_i, u_j)$. Dacă automatul este de tip Mealy în acelaşi loc se trece şi funcţia de ieşire $g(x_i, u_j)$, iar dacă automatul este de tip Moore în tabel se introduce o coloană suplimentară în care se trece funcţia $g(x_i)$.

STARE PREZENTĂ \ STARE VIITOARE	u_1	u_2
x_1	x_2 / y_1	x_3 / y_2
x_2	x_1 / y_2	x_3 / y_1
x_3	x_2 / y_2	x_3 / y_1

Tabelul 3.1

STARE PREZENTĂ \ STARE VIITOARE	u_1	u_2	$g(x_i)$
x_1	x_2	x_3	y_1
x_2	x_4	x_3	y_2
x_3	x_2	x_4	y_2
x_4	x_3	x_1	y_1

Tabelul 3.2

Automatelor din fig. 3.2 şi fig. 3.3 le corespund tabelele 3.1 şi respectiv 3.2.

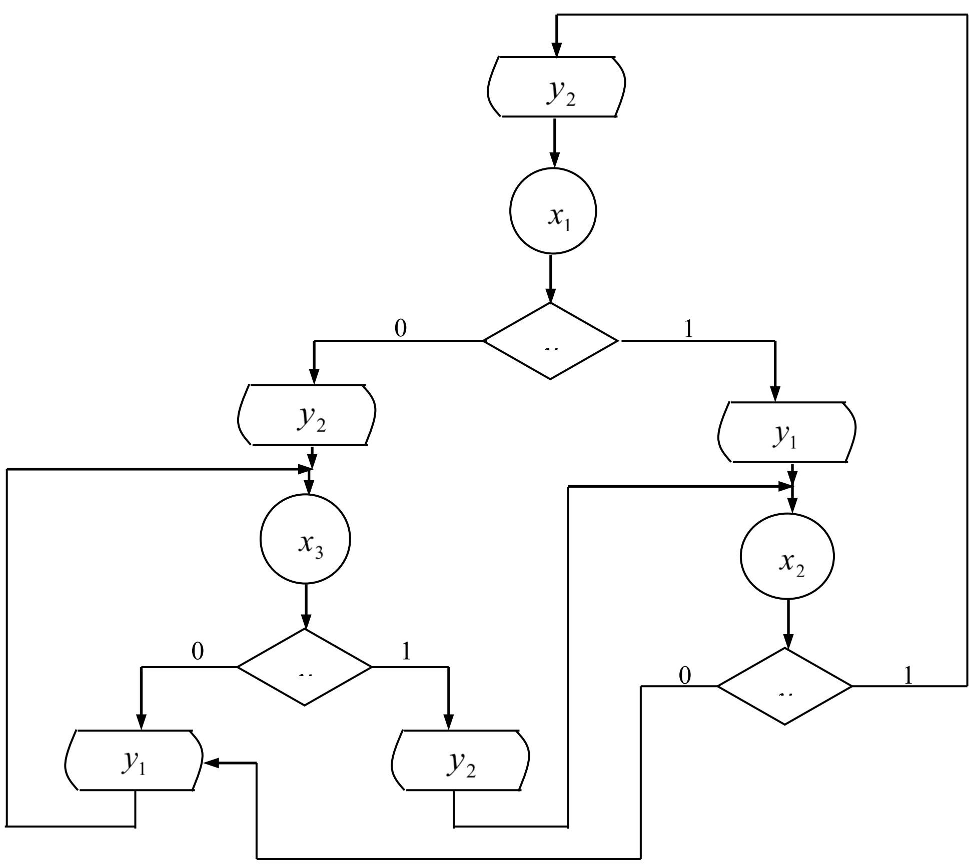


Figura 3.6

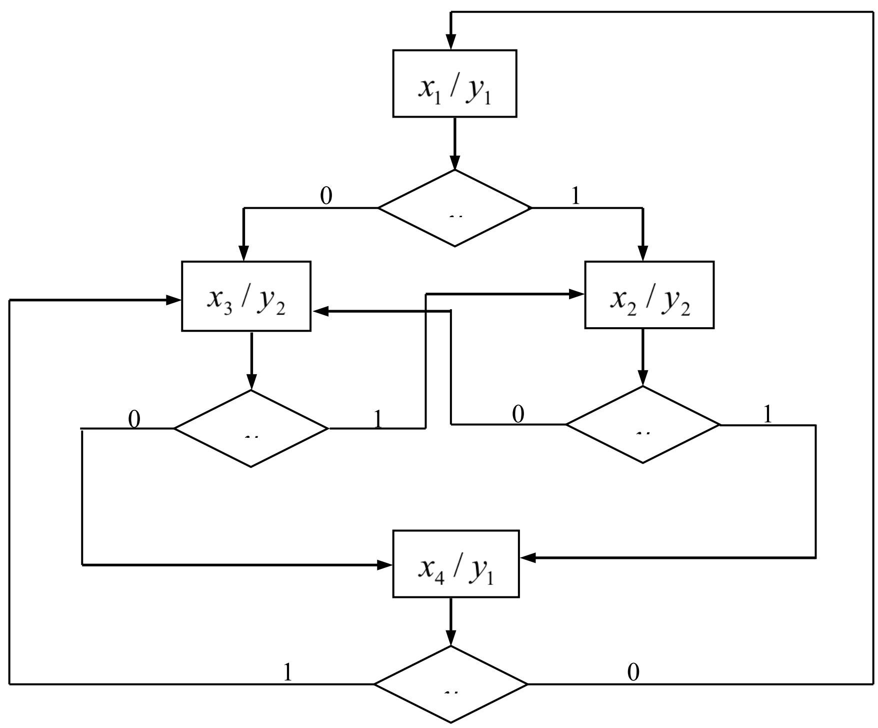


Figura 3.7

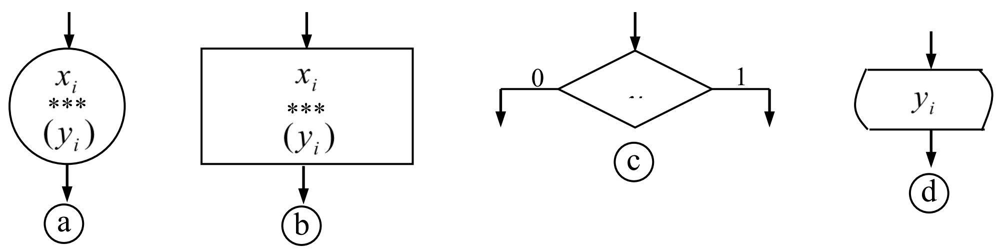


Figura 3.8

Cele două moduri de reprezentare exemplificate, deşi prezintă avantajul unei transpuneri apropiate de conceptul de automat finit, sunt mai greu de aplicat în cazul automatelor cu un număr mare de stări şi de intrări. Pentru acestea se poate utiliza o metodă de transpunere directă, rapidă şi intuitivă a condiţiilor de funcţionare ce trebuie îndeplinite de un automat, numită metoda organigramei funcţionale.

În fig.3.6 şi respectiv 3.7 sunt prezentate, deşi metoda nu este reprezentativă pentru automatele cu un număr mic de stări şi de intrări, organigramele funcţionale ale automatelor din fig.3.2 şi respectiv 3.3, cu observaţia că cele două variabile de intrare u_1 şi u_2 au fost codificate cu o singură variabilă u. Astfel u=1 reprezintă pe u_1, iar u=0 reprezintă pe u_2.

La construirea organigramelor funcţionale se folosesc simbolurile din fig.3.8.

În fig.3.8a şi b sunt reprezentate simbolurile pentru stările automatului. Simbolurile cuprind în mod curent indicativul stării, codul stării şi, dacă este cazul, ieşirea generată în starea respectivă.

În fig.3.8c este reprezentat blocul de decizie ce indică modul de evoluţie al automatului sub influenţa mărimilor de intrare care se aplică acestuia.

Ieşirea este definită, în general, în asociere cu starea automatului. Ea poate fi definită însă şi cu ajutorul blocului din fig.3.8d. În unele situaţii însă, atunci când funcţia de ieşire este generată în urma unor factori de decizie fără a fi corelate cu o stare a automatului, simbolul din fig.3.8d se impune.

<u>Observaţie:</u> Reprezentarea cu ajutorul grafurilor se numeşte, în general, reprezentare cu diagrame de stare.

3.1.2 Clasificări ale automatelor cu stări finite

Din punct de vedere abstract automatele finite pot fi clasificate în două mari categorii: automate combinaţionale şi automate secvenţiale. Această clasificare utilizată cu rezultate bune foarte mult timp a devenit prea "generală" odată cu "explozia tehnologică".

Criteriul de clasificare ce va fi considerat în continuare presupune:

- un automat de ordinul "n+1" poate fi generat prin interconectarea unor automate de ordin inferior, din care cel puţin unu este un automat de ordin "n" conectat într-o configuraţie ce presupune o buclă de reacţie;
- automatul de ordinul zero este reprezentat de un automat combinaţional caracterizat prin absenţa variabilelor de stare, ieşirea fiind definită ca o simplă transformare combinaţională a intrării.

Această clasificare este deschisă în sensul că se poate completa în funcţie de evoluţiile tehnologice ulterioare.

<u>Observaţie:</u> interconectarea în serie, paralel sau serie/paralel a unor automate de ordin "n" conduce la generarea unor automate tot de ordin "n".

<u>Automatul de ordinul zero</u>, numit şi automat combinaţional, are schema bloc şi organigrama funcţională prezentate în fig. 3.9. Automatul de ordinul zero conţine o singură stare internă, iar variabilele de ieşire depind numai de variabilele de intrare. Modelul fizic al automatului de ordinul zero îl constituie circuitul logic combinaţional.

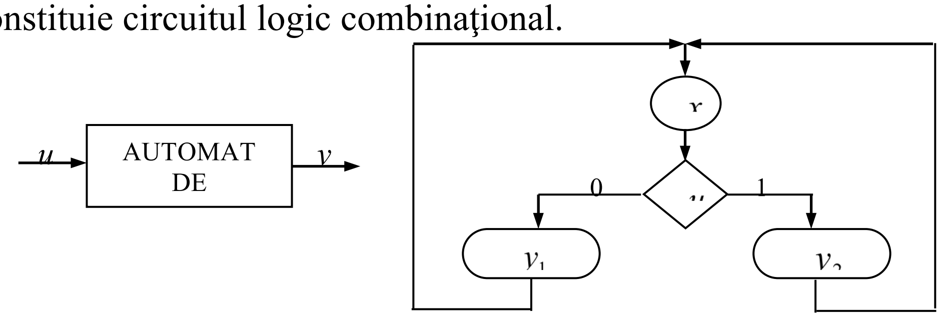


Figura 3.9

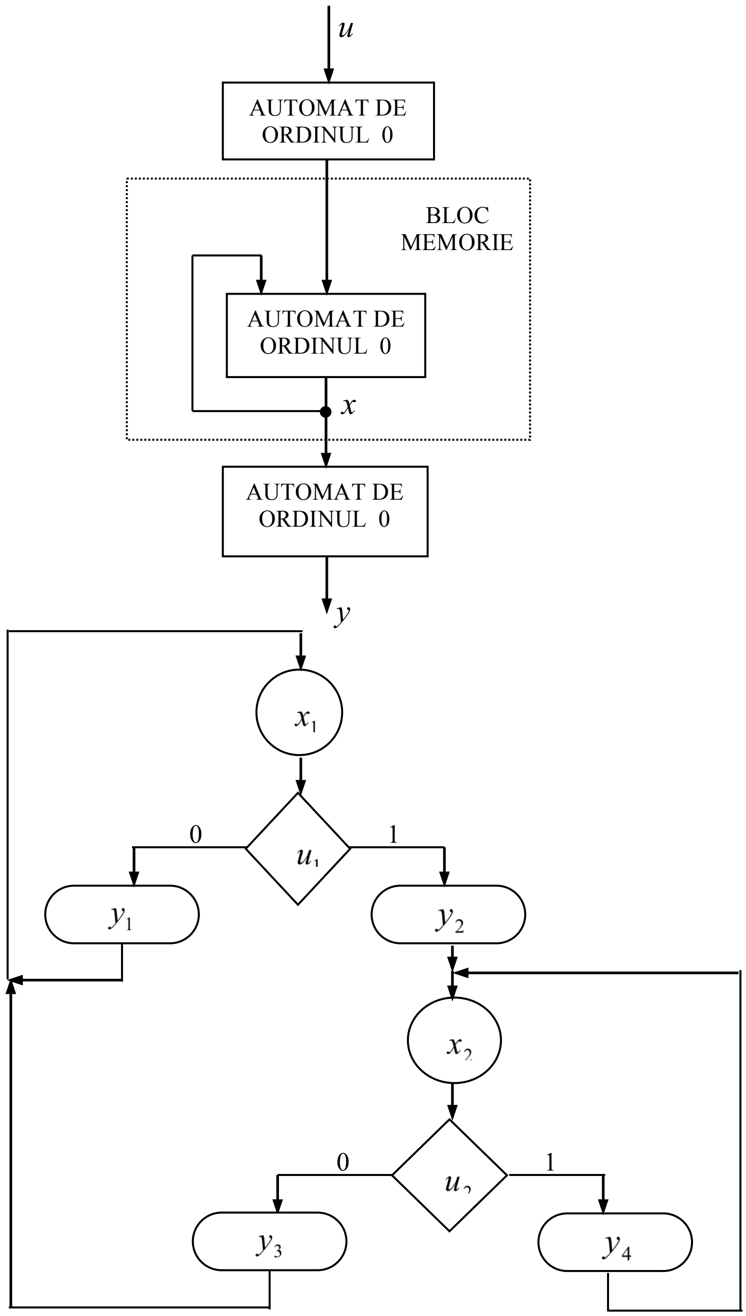


Figura 3.10

Automatul de ordinul 1 se obţine prin introducerea unei reacţii într-o structură de ordinul zero. El reprezintă cea mai simpla formă de automat secvenţial în care starea este generată de un bloc de memorie aflat într-o configuraţie fără buclă de reacţie. Semnalele de la intarea şi ieşirea blocului de memorie pot fi prelucrate de către două automate de ordinul zero. Cel mai simplu automat de ordinul 1 este modelat fizic de un circuit basculant bistabil de tip RS realizat cu două porţi logice SI-NU şi care îndeplineste funcţia de bloc de memorie.

Schema bloc şi organigrama funcţională a unui automat de ordinul 1 sunt reprezentate în fig.3.10.

Automatele de ordinul 1 prezintă o anumită autonomie faţă de evoluţia intrărilor. Această autonomie este însă limitată astfel încât aceste automate, deşi depăşesc nivelul automatelor combinaţionale, nu sunt totuşi automate secvenţiale propriu-zise şi deci justifică încadrarea lor într-o clasă separată.

Automatul de ordinul doi este automatul secvenţial tipic şi se obţine introducând o reacţie într-o structură de ordinul 1. Aceste automate prezintă o autonomie parţială, la limită chiar totală, faţă de evoluţia intrărilor. O secvenţă aplicată la intrarea unui automat de ordinul doi va genera la ieşire un răspuns parţial dependent de secvenţa de intrare şi puternic dependent de secvenţele anterioare aplicate la intrare ce se reflectă prin intermediul "stării prezente" a automatului. Mai mult, în paralel cu evoluţia ieşirilor, la aceste automate mai întâlnim o evoluţie în spaţiul stărilor care le conferă autonomia ce le deosebeşte de automatele de ordinul unu. Schema bloc generală a unui automat de ordinul doi este dată în fig. 3.11 şi ea reprezintă schema tip a unui automat Mealy. Acest automat se caracterizează prin faptul că funcţiile de tranziţie şi de ieşire sunt definite atât pe baza stării prezente cât şi prin variabilele de intrare.

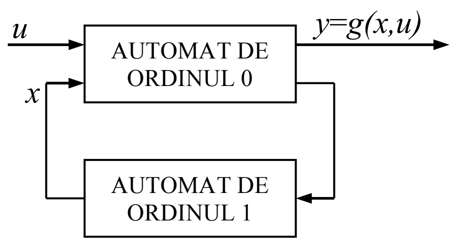


Figura 3.11

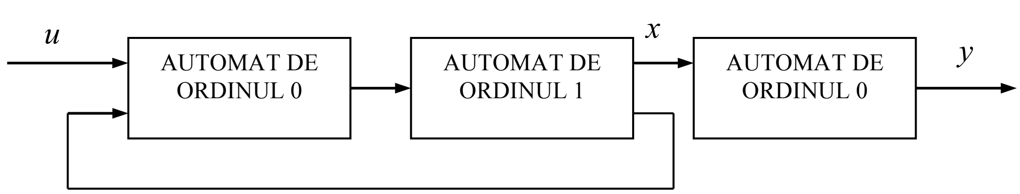


Figura 3.12

O structură particulară de automat de ordinul doi prezintă automatele de tip Moore a căror schemă bloc este redată în fig. 3.12. La acest tip de automat ieşirea se obţine în funcţie de starea prezentă:

$$y = g(x)$$

Un caz particular de automat de tip Moore este automatul având schema bloc din fig.3.13 ce se caracterizează prin faptul că nu este activat de variabile de intrare autentice, evoluţia dintr-o stare în alta realizându-se pe baza unui semnal de tact.

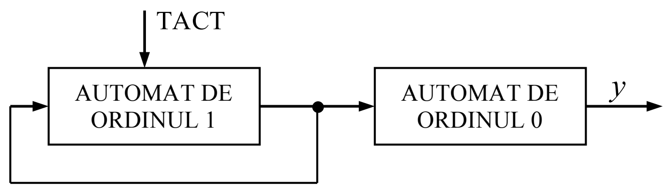


Figura 3.13

Automatele de ordinul doi sunt modelate fizic de către circuitele basculante bistabile de tip J-K, numărătoare şi extensiile lor serie şi paralel.

Automatul de ordinul trei se obţine prin introducerea unei reacţii într-o structură de ordinul doi. Schemele bloc pricipiale ale automatului de ordinul trei sunt prezentate în fig.3.14 a,b,c.

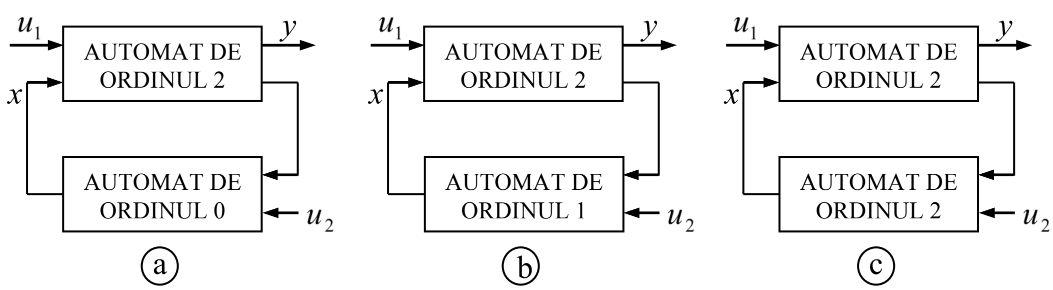


Figura 3.14

Automatul din fig. 3.14a este cel mai simplu dintre automatele de ordinul trei. El este modelat fizic de către structurile realizate cu circuite basculante bistabile de tip J-K sau cu numărătoare, în jurul cărora este realizată o reacţie cu porţi logice.

Automatul din fig. 3.14b conţine în bucla sa de reacţie un automat de ordinul unu materializat în general printr-un circuit de memorie.

Structura de automat de ordinul trei cea mai evoluată este cea din fig. 3.14c. Ea este materializată prin structuri microprogramabile în care unul din automate controlează activitatea celuilalt prin intermediul unui microprogram. O structură microprogramabilă constituie forma cea mai simplă a unui procesor.

Automatele de ordinul patru (şi mai mare de patru) se pot obţine introducând o reacţie într-o structură de ordinul trei. Fizic ele sunt materializate de către sistemele programabile.

Diversitatea acestora cât şi evoluţia lor dinamică justifică utilizarea acestei metode de clasificare deschisă a automatelor finite.