UNIUNEA EUROPEANĂ

GUVERNUL ROMÂNIEI

Instrumente Structurale 2007-2013

Platformă de e-learning și curriculă e-content pentru învățământul superior tehnic

Proiectarea Logică

11. Sinteza circuitelor seceventiale complexe

SINTEZA CIRCUITELOR SECVENTIALE COMPLEXE

Pasul 1. *Înţelegerea problemei*
Un automat finit este deseori descris comportamental prin anumite specificaţii. Este important să puteţi înţelege şi aceste descrieri într-o manieră neambiguă. Pentru numărătoare, este suficient de simplu să enumeraţi secvenţe. Pentru automate finite, încercaţi să daţi câteva valori pentru intrări, pentru a fi sigur că înţelegeţi condiţiile în care sunt generate ieşirile.

Pasul 2. *Obţinerea reprezentării abstracte a automatului*
Odată ce aţi înţeles problema, trebuie să o transformaţi într-o formă uşor de manipulat de către procedurile de implementare a automatelor finite. Diagrama de stări este o posibilitate. Alte reprezentări include maşinile algoritmice şi specificaţiile în limbaje de descriere hardware.

Pasul 3. *Minimizarea numărului de stări*
Pasul 2 în care se obţine reprezentarea abstractă, rezultă de multe ori în a avea prea multe stări. Anumite drumuri prin stările maşinii pot fi eliminate deoarece comportamentul intrărilor/ieşirilor poate fi echivalat de alte drumuri. Acest pas nu este necesar în proiectarea numărătoarelor simple.

Pasul 4. *Atribuirea stărilor*
La numărătoare, starea şi ieşirea erau identice şi nu am avut nevoie de codificarea unei anumite stări în mod particular. În general pentru automatele finite, acest lucru nu mai este valabil. Ieşirile sunt derivate din biţii stocaţi în stările bistabilului (şi din intrări) şi o bună alegere a codificării stării duce de obicei la o implementarea mai simplă.

Pasul 5. *Alegerea tipului bistabilului necesar implementării*
Acest pas este identic cu decizia din procedura de proiectare a numărătorului. Bistabili JK reduc numărul de porţi, în defavoarea numărului de conexiuni. Bistabilii D simplifică procesul de implementare.

Pasul 6. *Implementarea automatului finit*
Etapa finală este regăsită şi în procedeul de proiectare al numărătoarelor. Folosirea ecuaţiilor booleene sau a diagramelor Karnaugh, produc minimizarea şi implementarea.

În acest curs ne vom concentra pe primi 2 paşi ai procesului de proiectare. Paşii de la 3 la 6 vor fi explicaţi într-un curs viitor.

Un automat simplu

Pentru a ilustra procedeul de proiectare, se va parcurge implementarea unui automat finit care controlează funcţionarea unui automat.
Automatul eliberează un pahar de cafea după inserarea a 1,5 lei. Aparatul are un singur orificiu pentru monede şi bancnote, acceptă numai monede 50 de bani şi bancnote de 1 leu, câte una la o inserare. Un senzor mecanic indică dacă s-au introdus

50 de bani sau 1 leu. Ieşirea maşinii determină eliberarea unui singur pahar de cafea odată.

O ultimă specificaţie: Se va proiecta maşina astfel încât să nu dea restul. Un client ce va introduce 2 bancnote de 1 leu, va pierde 50 bani.

Înţelegerea problemei. Primul pas în proiectarea automatelor finite este să înţelegem cerinţele problemei. Începem prin a desena o diagramă bloc pentru a înţelege intrările şi ieşirile. Figura 9 este un bun exemplu în acest sens. Semnalul 0,50 este asertat pentru o perioadă de ceas când se introduce o monedă de 50 de bani. 1,0 este asertat când este introdusă o bancnota de 1 leu. Automatul asertează *Cafea* pentru o perioadă de ceas când s-au adunat cel puţin 1,5 lei de la ultima iniţializare (*Ini*).

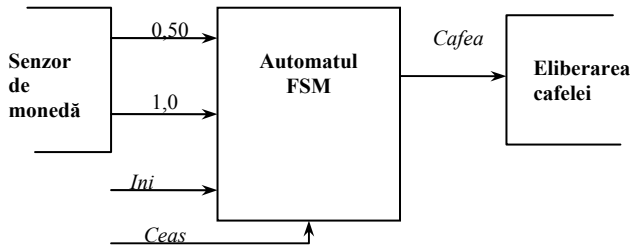


Figura 9. Diagrama bloc pentru maşina de pahare cu cafea

Specificaţiile pot să nu definească complet comportamentul maşinii. De exemplu, ce se întâmplă dacă cineva introduce o monedă de 1 ban? Sau, ce se întâmplă după ce paharul de cafea a fost eliberat clientului? Câteodată trebuie să facem anumite presupuneri. Pentru prima întrebare, se presupune că senzorul returnează orice monedă pe care nu o recunoaşte, lăsând semnalele 0,50 şi 1,0 neasertate. Pentru a doua întrebare, se presupune că automatul se re-iniţializează singur după ce paharul de cafea este eliberat.

Reprezentarea abstractă Odată înţeles comportamentul, este timpul să transformăm specificaţiile într-o reprezentare abstractă. Un mod corect de începere a acestui lucru este acela prin care se enumera secvenţele posibile de intrări sau configuraţiile sistemului. Acestea vor ajuta la definirea stărilor maşinii.

Pentru această problemă, nu este prea dificil să se enumere toate secvenţele posibile de intrări ce duc la eliberarea paharului de cafea:

- 3 monede de 50 de bani în secvenţa : 0,50, 0,50, 0,50;
- 2 monede de 50 de bani şi 1 bancnotă de 1 leu în secvenţa: 0,50, 0,50, 1,0;
- 1 monedă de 50 de bani şi o bancnota de 1 leu în secvenţa: 0,50, 1,0;
- 2 bancnote de 1 leu în secvenţa: 1,0, 1,0.

Aceste posibilităţi pot fi reprezentate printr-o diagramă de stări ca în fig. 10. De exemplu, automatul va trece prin stările S_0, S_1, S_3, S_7 dacă secvenţa de intrări este 3 monede de 50 de bani.

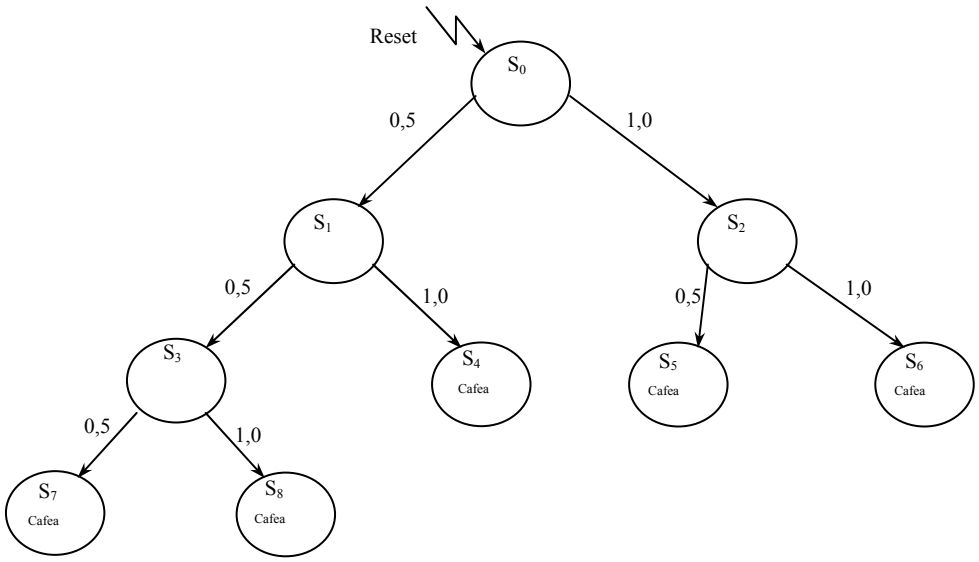


Figura 10. *Diagrama stărilor pentru maşina de vândut cafea.*

Pentru a păstra diagrama de stări simplă şi lizibilă, se vor include numai tranziţiile care cauzează explicit schimbarea stării. De exemplu, în starea S_0, dacă nici 0,5 şi nici 1,0 nu sunt asertate, se presupune ca automatul rămâne în starea S_0 (specificaţia face să se presupună că 0,5 şi 1,0 nu pot fi asertate în acelaşi timp). De asemenea, se include ieşirea *Cafea* numai în stările în care aceasta este *asertată* şi se consideră implicit *neasertată* în celelalte.

Minimizarea numărului de stări Această reprezentare cu 9 stări nu este cea optimă posibilă. De exemplu, stările S_4, S_5, S_6, S_8 au comportament identic şi pot fi combinate într-o singură stare.

Pentru a reduce numărul de stări şi mai mult, putem să gândim fiecare stare ca fiind suma acumulată până în acel punct. De exemplu nu ar trebui să conteze că la starea reprezentând 1.0 lei s-a ajuns cu 2 monede de 50 de bani sau cu o bancnotă având valoarea 1 leu.

Diagrama de stare ce rezultă este reprezentată în fig. 11. Putem reprezenta comportamentul cu numai 4 stări, comparativ cu 9, care erau în fig. 10. De asemenea, o altă utilitate a reprezentării minimale poate fi observată la tranziţia din starea 1,0 lei la starea 1,5 lei. Se interpretează notaţia "*ND*" asociată acestei tranziţii ca fiind "se trece în starea 1,5 lei dacă N sau D sunt asertate".

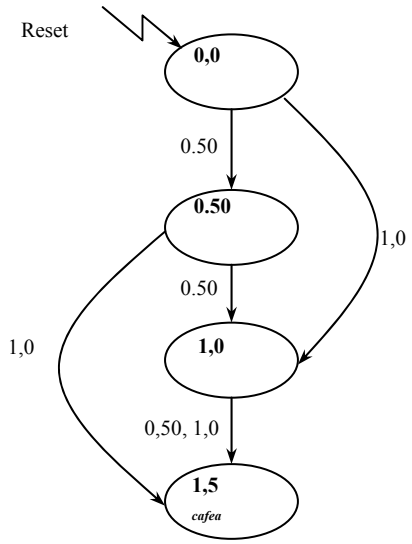


Figura 11. *Diagrama minimizată a stărilor.*

În următorul capitol, vom examina metodele formale de găsire a diagramei de stări minimale. Procesul de minimizare a diagramei asociate automatului se numește minimizarea numărului de stări.

Codificarea stărilor În acest moment avem un automat finit, cu număr minim de stări, dar încă la un nivel simbolic. Se remarcă în figura 12 tabelul tranziţiilor de stare. Următoarea etapă este *codificarea.*

Modul de codificare al stărilor poate avea un efect important asupra "*volumului*"de circuite necesar implementării automatului. Un mod natural de codificare ar fi cel cu 2 biţi: starea 0 lei ca fiind 00, starea 50 bani ca fiind 01, starea corespunzătoare valorii 1,0 lei ca fiind 10, şi starea aferentă valorii1,5 lei ca fiind 11. O codificare mai puţin evidentă ar putea duce la o configuraţie hardware mai simplă. Tabelul tranzițiilor de stări codificate este reprezentat în figura 13.

Starea curentă	Intrări		Starea următoare	Ieșire
	D	N		Cafea
0 lei	0	0	0 lei	0
	0	1	50 bani	0
	1	0	1,0 leu	0
	1	1	*X*	X
50 bani	0	0	50 bani	0
	0	1	1,0 leu	0
	1	0	1,5 lei	0
	1	1	*X*	X
1,0 lei	0	0	1,0 leu	0
	0	1	1,5 lei	0
	1	0	1,5 lei	0
	1	1	*X*	X
1,5 lei	*X*	*X*	1,5 lei	1

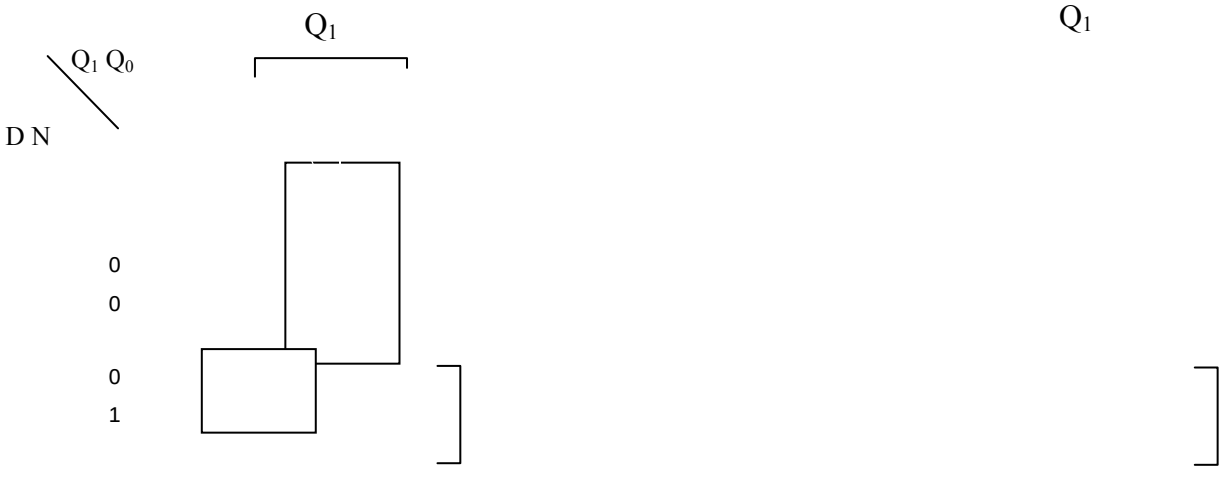
Figura 12. *Tabelul tranziţiilor de stare.*

Starea curentă		Intrări		Starea următoare		Ieşire
Q_1	Q_0	D	N	D_1	D_0	Cafea
0	0	0	0	0	0	0
		0	1	0	1	0
		1	0	1	0	0
		1	1	*X*	*X*	*X*
0	1	0	0	0	1	0
		0	1	1	0	0
		1	0	1	1	0
		1	1	*X*	*X*	*X*
1	0	0	0	1	0	0
		0	1	1	1	0
		1	0	1	1	0
		1	1	*X*	*X*	*X*
1	1	*X*	*X*	1	1	1
				1	1	1
				1	1	1
				X	*X*	*X*

Figura13. *Tabelul tranziţiilor stărilor codificate.*

Implementarea Următorul pas este implementarea tabelului tranziţiilor de stare după alegerea prealabilă a elementelor de memorie, stocare. Se va urmări implementarea cu ajutorul bistabililor *JK* şi *D*. Diagramele Karnaugh pentru implementare cu bistabili *D* sunt reprezentate în figura 14. Au fost completate direct din tabelul stărilor codificate. Ecuaţiile de minimizare pentru intrările şi ieşirea circuitului sunt:

$$D_1 = Q_1 + D + Q_0 * N$$
$$D_0 = N * \vec{Q}_0 + Q_0 * \bar{N} + Q_1 * N + Q_1 * D$$
$$CAFEA = Q_1 * Q_0$$



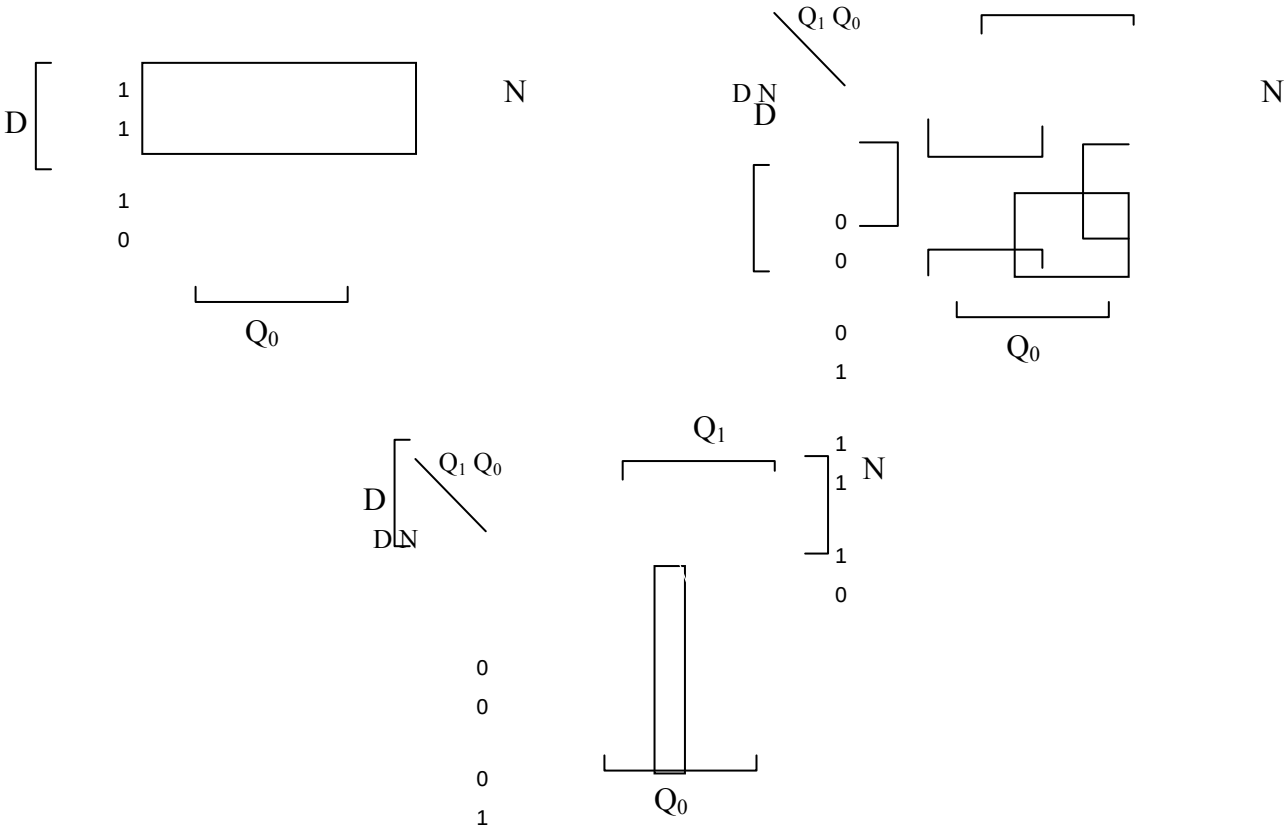


Figura 14. *Diagramele Karnaugh pentru bistabili D.*

Logica implementării este reprezentată în figura 15. Se utilizează 8 porţi şi 2 bistabili.

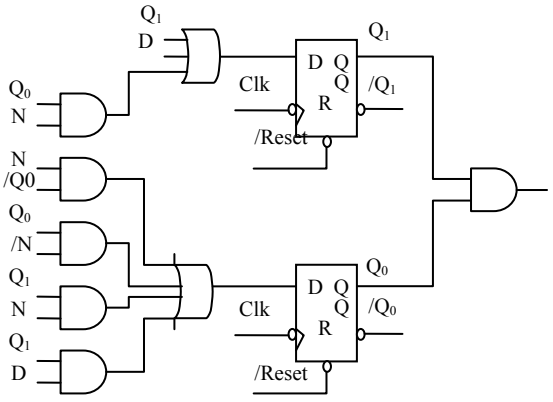


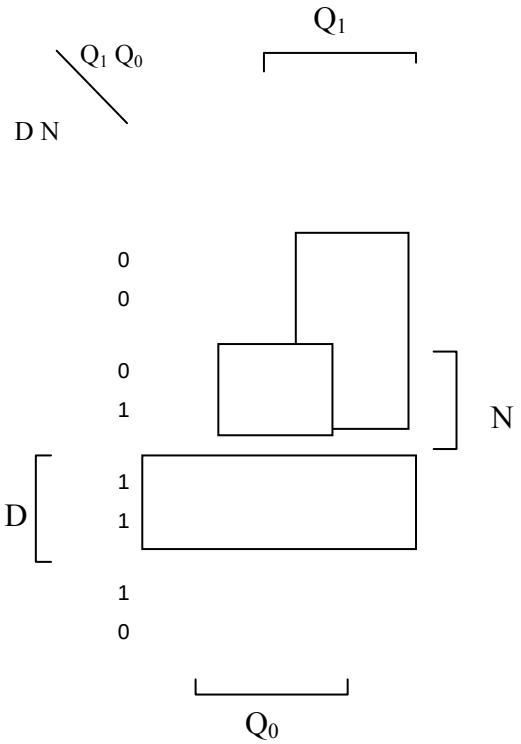
Figura 15. *Logica implementării cu bistabili D.*

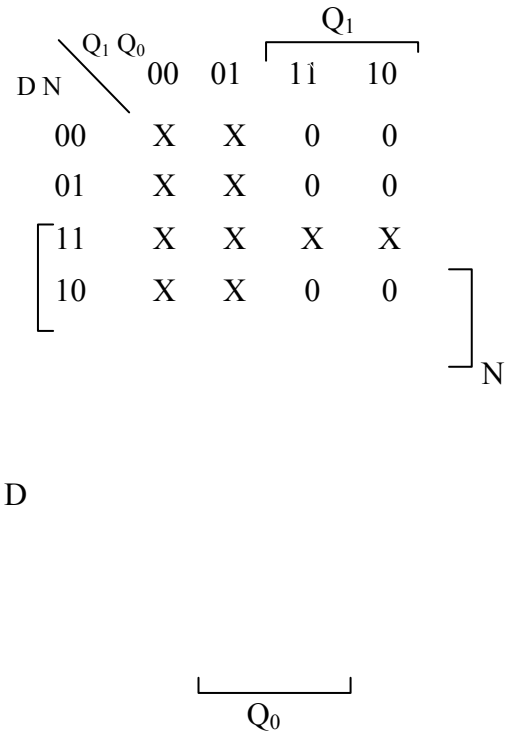
Pentru a implementa automatul cu bistabili *JK*, trebuie să se refacă funcţiile de tranziţie de stare. Tabelul refăcut al tranzițiilor de stare pentru implementarea cu bistabili *JK* este reprezentat mai jos. Ecuaţiile minimizate arată astfel:

$$J_1 = D + Q_0 * N \qquad K_1 = 0$$
$$J_0 = \overline{Q_0} * N + Q_1 * D \qquad K_0 = \overline{Q_1} * N \tag{17}$$

Figura 18 reprezintă logica implementării utilizând bistabili *JK*. Se observă reducerea numărului de circuite: 7 porţi şi 2 bistabili

Starea curentă		Intrări		Starea următoare		J_1	K_1	J_0	K_0
Q_1	Q_0	D	N	D_1	D_0				
0	0	0	0	0	0	0	X	0	X
		0	1	0	1	0	X	1	X
		1	0	1	0	1	X	0	X
		1	1	X	X	X	X	X	X
0	1	0	0	0	1	0	X	X	0
		0	1	1	0	1	X	X	1
		1	0	1	1	1	X	X	0
		1	1	X	X	X	X	X	X
1	0	0	0	1	0	X	0	0	X
		0	1	1	1	X	0	1	X
		1	0	1	1	X	0	1	X
		1	1	X	X	X	X	X	X
1	1	0	0	1	1	X	0	X	0
		0	1	1	1	X	0	X	0
		1	0	1	1	X	0	X	0
		1	1	X	X	X	X	X	X





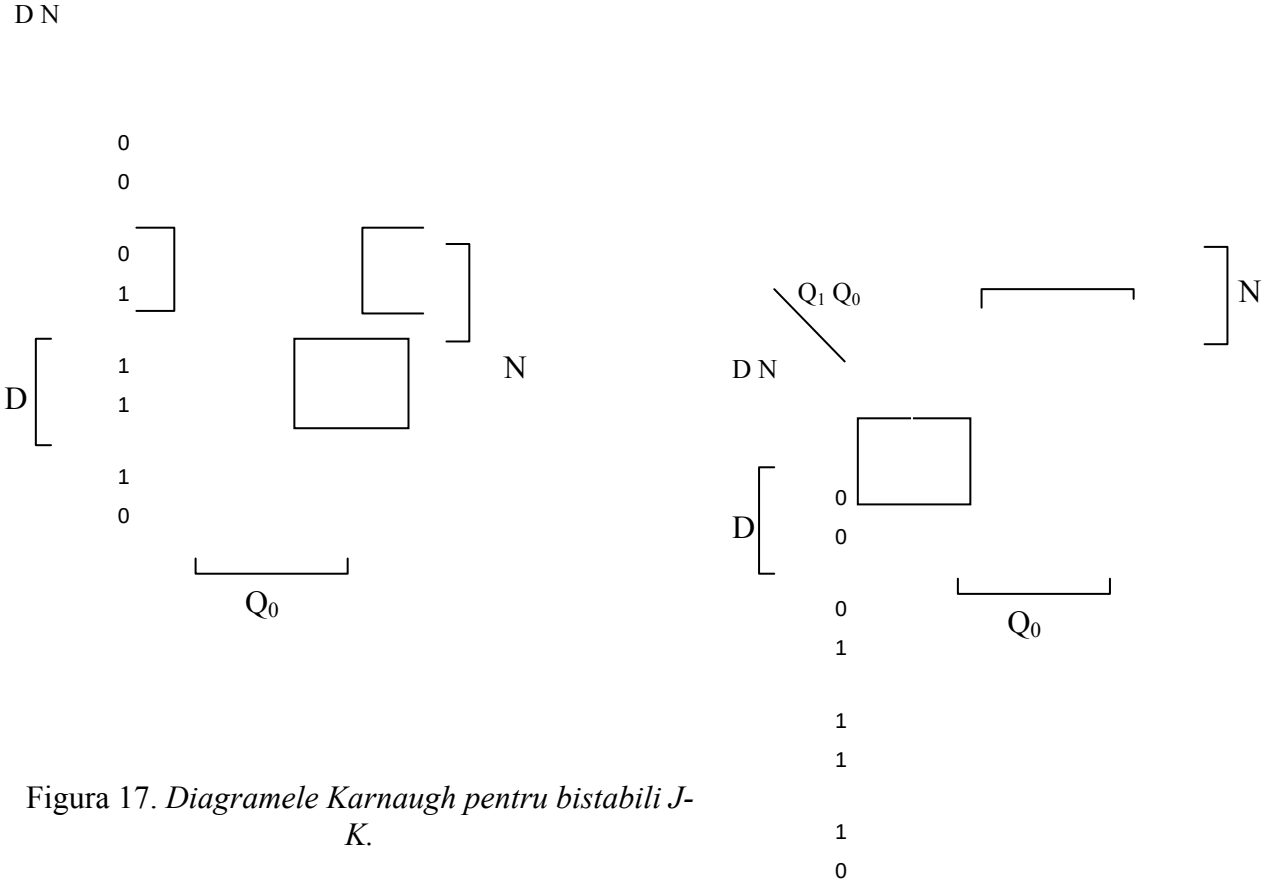


Figura 17. *Diagramele Karnaugh pentru bistabili J-K.*

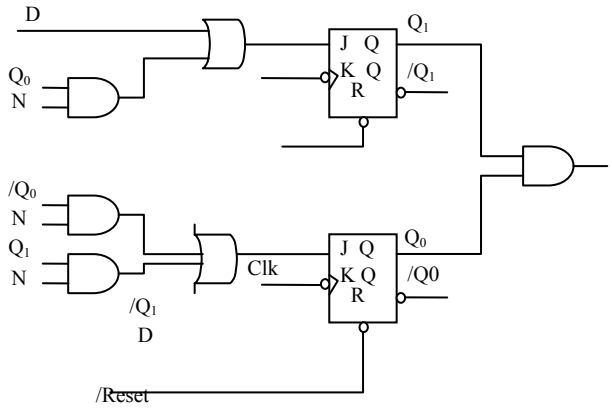


Figura 18. *Logica implementării cu bistabili J-K.*

Concluzii. S-a descris pe scurt procesul de proiectare al automatului finit şi a fost ilustrat cu ajutorul unui automat simplu. Începând cu considerarea cerinţelor, s-a descris automatul într-un mod formal. În acest caz s-au folosit diagramele de stări.

Cum există mai mult decât o diagramă care să conducă la acelaşi comportament intrare/ieşire este important să se găsească o descriere cu cât mai puţine stări. Acest lucru reduce de obicei complexitatea implementării automatului finit. De exemplu, diagrama de stare din fig. 10 conţine 9 stări şi necesită 4 bistabili pentru implementare. Diagrama minimizată din fig. 11 are 4 stări şi poate fi implementată cu numai 2 bistabili.

Odată ce s-a obţinut o descriere minimală, următorul pas este găsirea codificării potrivite a stărilor. Alegerea potrivită poate reduce logica implementării pentru funcţiile de ieşire. În exemplu nostru s-a folosit o codificare evidentă.

Ultimul pas este alegerea tipului de bistabili pentru registrul de stare. În exemplul considerat, implementarea cu bistabili *D* a fost mai uşor de realizat din punct de vedere al calculelor necesare. Nu a fost nevoie să se refacă diagramele corespunzătoare intrărilor, dar s-au folosit mai multe porţi decât la implementarea cu bistabile *JK*.