PROIECTAREA AUTOMATELOR FINITE

Introducere

În acest capitol se începe examinarea celui mai important tip de circuit secvenţial: automatul cu (număr de) stări finite. Acest tip de circuit secvenţial este numit aşa datorită logicii secvenţiale care îl implementează şi care poate avea doar un număr finit de stări. Numărătoarele prezentate anterior sunt exemple simple de automate finite. Ieşirile acestora sunt identice cu stările şi nu se poate modifica semnificativ, pe durata funcţionării, secvenţa de stări ce urmează să fie urmată în cadrul mersului maşinii.

Mai general, ieşirile şi starea următoarea a unui automat finit sunt funcţii logice combinaţionale având ca parametrii intrările şi starea curentă. Starea viitoare depinde de liniile de intrare, implicând astfel, un comportament mult mai complex comparativ cu cel al numărătoarelor. Automatele (maşinile) cu stări finite sunt critice pentru realizarea controlului şi logicii decizionale din sistemele digitale.

În consideraţiile următoare se vor extinde metodele introduse în proiectarea numărătoarelor pentru cazul mai general al automatelor finite. În rândurile care vor urma se vor aborda:

- Metode de descriere comportamentală a automatelor finite. Acestea incluzând notaţii, diagrame de stări şi tabele de stări.
- Tehnici de implementare specifice automatelor cu stări finite.

1. Conceptul maşinii cu stări

Se va demara abordarea maşinilor cu un număr finit de stări printr-un exemplu de funcţionare a cărei ieşire depinde de intrările precedente. Se va parcurge integral procesul de transformare al specificaţiilor maşinii cu stări, printr-o secvenţă de reprezentări echivalente, rezultând în final implementarea prin porţi şi bistabili a respectivei maşini cu stări.

1.1. Circuitele de paritate ori imparitate

Se consideră proiectarea unui circuit logic care determina paritatea apariţiilor valorii „1” dintr-un şir, dintr-o serie de valori, „0” ori „1” ale liniei de intrare. Dacă circuitul are ca ieşire ‘1’ când se detectează un număr par de valori ‘1’ la intrare, se va spune că acesta *verifică paritatea*, iar atunci când care ieşirea ia valoarea ‘1’ pentru un număr impar de intrări ‘1’, acest circuit se va spune că *verifică imparitatea*. Circuitul în cauză se dovedeşte a fi secvenţial, în natura funcţionării sale, deoarece ieşirea curentă depinde de valorile precedente ale liniei de intrare.

Diagrama stărilor. Primul pas în procesul de proiectare este determinarea unei diagrame de stări care să descrie comportarea circuitului. Nu este greu să observam că circuitul poate fi intr-una din cele două stări posibile: poate fi un număr par sau impar de valori ‘1’ de la pornirea circuitului. Când se primește o valoare ‘1’ la intrare, circuitul își schimbă starea. Dacă a fost citit un număr impar de valori ‘1’ până în momentul curent, spre exemplu, iar intrarea curentă este ‘1’ atunci vor fi un număr par de intrări ‘1’. Dacă intrarea este ‘0’, se păstrează aceiaşi stare.

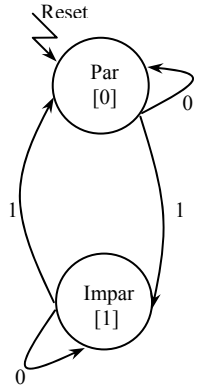

Figura 1 Exemplul unei diagrame de stări.

Diagrama de stări este prezentată în figura 1. Cele două stări posibile ale circuitului vor fi numite *Par* şi *Impar*. Ieşirile circuitului sunt asociate explicit cu stările acestuia şi sunt notate între paranteze drepte. Când se determină un număr impar de ‘1’, ieşirea este ‘1’, altfel este ‘0’. Se asociază valoarea intrării tranziţiei dintre stări.

Starea prezentă	Intrare	Următoarea stare	Ieşirea
Par	0	Par	0
Par	1	Impar	0
Impar	0	Impar	1
Impar	1	Par	1

Figura 2. Tabelul tranziţiilor stărilor.

Tabelul tranziţiilor de stare. O reformulare a diagramei de stări, este tabelul tranziţiilor de stare, reprezentat în figura 2. Se asociază nume simbolice explicite intrărilor, ieşirilor şi stărilor. Circuitul căutat nu este încă implementabil. Sunt necesare asocieri constituite din codificări binare pentru fiecare stare, pentru intrările şi ieşirile din acest tabel.

Starea curentă	Intrarea actuală	Starea următoare	Ieșirea
0	0	0	0
0	1	1	0
1	0	1	1
1	1	0	1

Figura 3. Tabelul codificării stărilor

În figura 3 este prezentată reprezentarea codificată, numită şi tabelul de codificare al stărilor. S-a asociat codul ‘0’ stării par şi ‘1’ stării impar.

Funcţia stării următoare şi funcţia ieşirii. În acest moment funcţiile stării următoare (S^+ ori S^{t+1}) şi liniei de ieşire (*Ieşirea*) sunt exprimate ca funcţii logice având ca parametrii starea prezentă (S) şi intrarea automatului (IA). Se pot scrie aceste funcţii ca fiind:

$$S^+ = S \oplus IA$$

$$Ieşirea = S$$

Implementarea. Acum este posibil să se implementeze circuitul. Starea unui automat este implementată prin bistabili. Deoarece sunt doar două stări, în cazul de faţă, se va putea implementa circuitul printr-un singur bistabil. Funcţia *stării_următoare* (S^+) va determina valoarea liniei (liniilor) intrării în bistabil.

Se poate realiza o implementare a acestui circuit utilizând un singur bistabil D, aşa cum se poate urmări în figura 4 (a). Poarta XOR determină valoarea liniei de intrare D a bistabilului, în funcţie de starea curentă şi de linia de intrare IA.

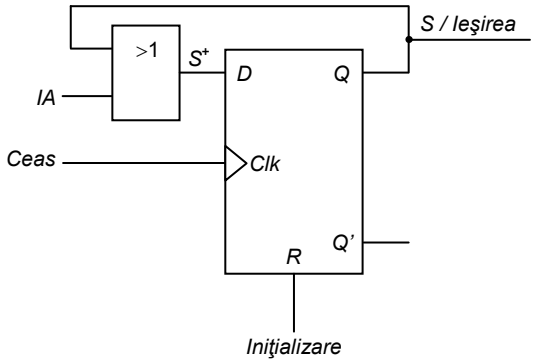

Figura 4 (a). Implementarea automatului printr-un bistabil *D*.

O privire mai atentă asupra tabelei tranziţiilor de stare, ar putea să sugereze şi o implementarea alternativă. Când intrarea este ‘0’, circuitul rămâne în aceiaşi stare. Când

este ‘1’, starea se schimbă. O implementare bazată pe un bistabil T este înfăţişată în figura 4 (b). Această implementare este mai simplă, deoarece poarta XOR este implicită. Adeseori, printr-o alegere corespunzătoare a tipului de bistabil, se poate simplifica implementarea.

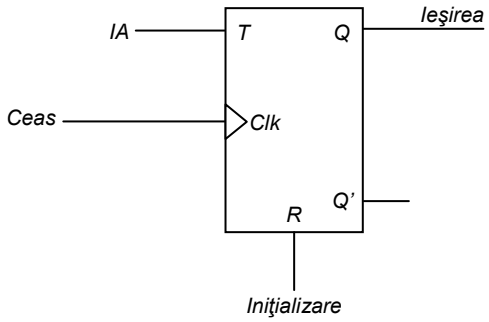

Figura 4 (b). Implementarea automatului printr-un bistabil *T*.

Figura 5 prezintă comportamentul în timp al automatului, pentru un şir de valori ale liniei de intrare de forma 100110101110.

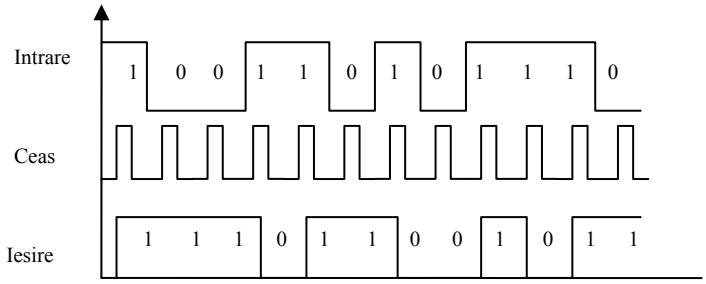

Figura 5. Formele de undă pentru intrare/ieşire.

Fiecare valoare a liniei de intrare este procesată pe frontul crescător al impulsului de ceas, pentru că registrul de stare este implementat pe un bistabil activat pe front crescător (aşa cum se poate urmări în figura 5).

Ieşirea se modifică imediat după modificarea nivelului de tensiune al liniei de ceas. Ar trebui să fie evident că ieşirea este ‘1’ când şirul de intrare conţine un număr impar de 1 şi este ‘0’ altfel.

1.2. Sincronizarea automatelor finite

În proiectarea automatelor, maşinilor cu stări finite (MSF), se va urma o metodologie strictă de proiectare sincronă. Aceasta înseamnă că se va activa schimbarea stării printr-un semnal de referinţă global, numit ceas. Este important să se înţeleagă când anume sunt considerate şi preluate valorile liniilor de intrare, când se instaurează starea următoare şi când este transmisă, în concordanţă cu semnalul de ceas, valoarea ieşirii.

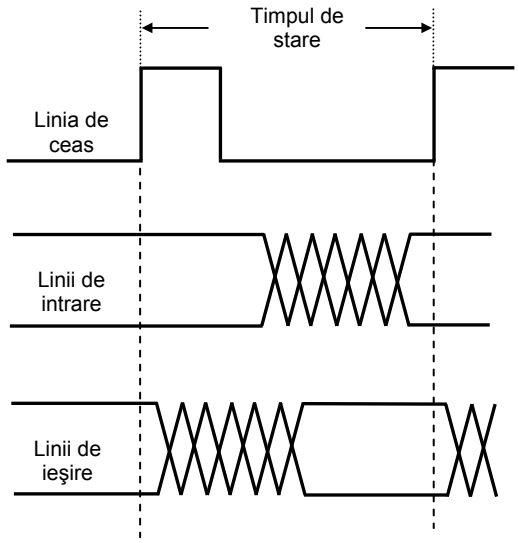

Figura 6. Modul de funcţionare al unei maşini cu stări acţionate pe frontul pozitiv al liniei de ceas.

Timpul stării. Se defineşte *timpul stării* ca fiind timpul dintre două evenimente succesive asociate liniei ceasului.

Pentru sistemele acţionate pe frontul semnalelor de ceas, aceste evenimente sunt frontul crescător şi frontul descrescător ale ceasului. În cazul unui sistem cu stări finite, acţionat pe frontul crescător al semnalului de ceas, timpul de stare este măsurat între două fronturi crescătoare consecutive. Similar, pentru un automat cu stări finite acţionat pe frontul descrescător al semnalului de ceas, timpul de stare este măsurat între două fronturi descrescătoare consecutive.

Ca răspuns la un eveniment asociat ceasului, starea şi ieşirea unui automat cu stări finite se modifică conform stării actuale şi liniei (liniilor) de intrare. Pentru mai multă siguranţă, şi din considerente impuse de întârzierile de propagare dar şi de timpul de pregătire necesar logicii care determină starea următoare, valorile liniilor de intrare trebuie să fie stabile înainte să se producă evenimentului asociat ceasului.

După o întârziere corespunzătoare propagării semnalelor, automatul cu stări finite va migra în starea următoare, iar noile valori ale liniilor de ieşire se stabilizează. Pentru o maşină cu stări finite implementată cu bistabile acţionate pe frontul crescător (pozitiv) se

poate vedea, în figura 6, o ilustrare a schimbării stărilor, a eşantionării valorilor liniilor de intrare şi a schimbării valorilor liniilor de ieşire.

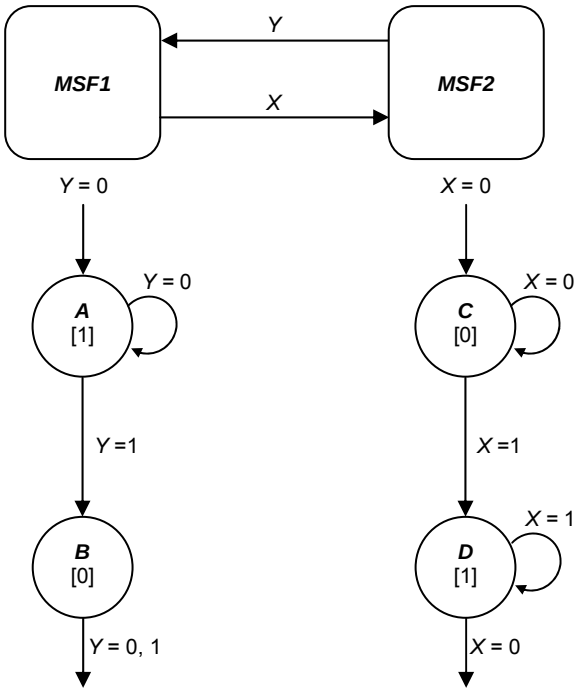

Figura 7. Schiţă a unui mod de comunicare dintre două maşini cu stări.

Validarea ieşirii. O linie de ieşire nu are valori stabile, valide, înainte de evenimentul asociat liniei de ceas (încheierea frontului activ al impulsului de ceas) deoarece liniile de intrare sunt eşantionate, preluate, chiar înaintea frontului activ.

Ca un exemplu de comportament temporal al unui automat cu stări finite, se prezintă sumar în figura 7 diagramele de stări a două maşini cu stări care au semnalul de ceas comun şi pot comunica prin semnalele *X* şi *Y*.

Se presupune că ambele automate sunt active pe frontul crescător al impulsului de ceas şi că ieşirea uneia dintre maşini este intrare pentru cealaltă. Interacţiunea dintre aceste două maşini cu stări este detaliată prin diagramele de timp din figura 8.

Pentru început automatul FSM_1 (maşina 1), se află în prima perioadă (indexul 0) a impulsului de ceas, urmând să intre în starea *A*, având acum $X = 0$, linia sa de ieşire (figura 8, linia FSM_1).

Cealaltă maşină FSM_2 (maşina 2) a intrat în starea C iar linia sa de ieşire *Y* are valoarea 0 (figura 8, linia FSM_2).

După al doilea impuls de ceas FSM_1 e în starea A şi asertează linia sa de ieşire X, $X = 1$. A doua maşină FSM_2 este în starea C cu ieşirea *Y* neasertată ($Y = 0$). Pe frontul crescător al celei de-a treia perioade de ceas, maşina FSM_1 rămâne în starea A având intrarea tot 0.

Automatul FSM_2 intră în starea D, şi asertează pe linia de ieşire *Y* ($Y = 1$), dar prea târziu pentru a influenţa schimbarea stării din FSM_1. Aceasta se întâmplă pentru că valoarea intrării de dinainte de front, este cea care contează.

Acum, pentru că linia de Y este 1, prima maşină FSM_1 intră în starea B la următorul front crescător. În această stare, va avea ieşirea X =0, dar prea târziu pentru a afecta modificările stării automatului FSM_2, care va rămâne în starea D.

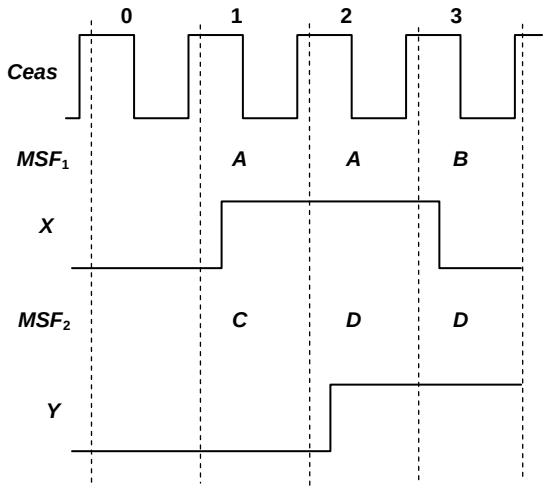

Figura 8. Schimbările stărilor şi valorilor liniilor de ieşire pentru modul de comunicare dintre maşinile cu stări.

2. Modul de abordare al proiectării maşinilor cu un număr finit de stări

Procedeul de proiectare, anterior prezentat, al numărătoarelor sincrone constituie un nucleu al unui procedeu mai general, aplicabil unei largi categorii de maşini cu număr finit de stări.

2.1. Procedeul de proiectare al unui automat cu un număr finit de stări

Pasul 1. *Înţelegerea problemei*

Un automat finit este deseori descris comportamental, prin anumite specificaţii dar şi descrieri. Este important să se poată înţelege aceste descrieri chiar şi atunci când acestea sunt enunţate într-o manieră cvasi-ambiguă. Pentru numărătoare, este suficient de simplu să se listeze, ori sa să se enumere, secvenţa respectivă de stări.

În cazul automatelor cu stări finite este, adesea, util să se încerce verificarea funcţionării acestora prin considerarea unui subset potrivit al mulţimii valorilor liniilor de intrare, în scopul dobândirii asigurării că s-au înţeles corect şi complet condiţiile în care sunt generate valorile liniilor de ieşire.

Pasul 2. *Obţinerea reprezentării abstracte a automatului*

Odată ce s-a înţeles complet şi corect tema de proiectare, aceasta trebuie să fie transformată într-o formă uşor de utilizat prin procedeele de implementare ale automatelor finite. Diagrama de stări este, în acest sens o posibilitate simplă şi adesea întâlnită. Alte reprezentări cuprind maşinile algoritmice şi specificaţiile în limbaje de descriere hardware.

Pasul 3. Minimizarea numărului de stări

Reprezentarea obţinută în cadrul pasului anterior poate avea, de multe ori, un număr prea mare de stări. Anumite căi între liniile de intrare şi de ieşire, trecând printre stările maşinii, pot fi eliminate deoarece, în virtutea comportamentul dintre liniile de intrare şi cele de ieşire, se pot stabili alte drumuri echivalente. Acest pas este tipic maşinilor cu stări finite, nefiind necesar în proiectarea numărătoarelor obişnuite.

Pasul 4. Atribuirea stărilor

Atunci când se proiectează numărătoarele, între starea numărătorului şi ieşirea acestuia există o legătură directă şi nu necesară codificarea vreunei stări anumite, în mod particular. În cazul automatelor finite, acest lucru nu mai este valabil, în general. Valorile liniilor de ieşire sunt deduse din valorile binare stocate în stările bistabililor, pe de-o parte şi din valorile liniilor de intrare. O alegere potrivită a codificării stării poate conduce, de regulă, la o implementarea mai simplă, posibil cu costuri mai mici.

Pasul 5. Alegerea tipului bistabilului necesar implementării

Acest pas este apropiat celui de la numărătoare. Astfel, bistabilele *JK* reduc numărul de porţi, în defavoarea numărului de conexiuni iar bistabilele *D* simplifică procesul de implementare.

Pasul 6. Implementarea automatelor finite

Aceasta este etapa finală, care se regăseşte şi în procedeul de proiectare al numărătoarelor. Prin folosirea ecuaţiilor booleene sau a diagramelor Karnaugh, se realizează minimizarea şi implementarea.

Această secţiune a cursului se focalizează asupra primilor doi paşi ai procesului de proiectare. Paşii ceilalţi vor fi abordaţi într-o abordare viitoare.

2.2. Un automat simplu

Pentru ilustrarea procedeului de proiectare, se va parcurge implementarea unui automat finit care controlează funcţionarea unui automat.
Acest automat eliberează un pahar de cafea după introducerea sumei de 1,5 lei. Aparatul are o singură deschidere care slujeşte atât pentru introducerea monedelor cât şi a bancnotelor. Sunt acceptate numai monede 50 de bani şi bancnote de 1 leu, câte una la o introducere. Un senzor mecanic determină dacă s-a introdus o monedă de 50 de bani sau

o bancnotă de 1 leu şi automatul afişează valoarea recunoscută. Ieşirea maşinii determină eliberarea unui singur pahar de cafea odată.

O ultimă specificaţie: maşina nu oferă restul. Un client care va introduce două bancnote de 1 leu, va pierde 50 bani.

Înţelegerea problemei. Primul pas în proiectarea automatelor finite este să se înţeleagă care sunt cerinţele problemei. O bună practică este să se înceapă prin desenul unei diagrame bloc astfel încât să se înţeleagă determinarea ieşirilor în raport cu ieşirile. În figura 9 se prezintă un exemplu, în acest sens. Semnalul 0,50 este asertat pentru o perioadă de ceas după ce se introduce o monedă de 50 de bani. Semnalul 1,0 este asertat după ce se introduce o bancnota de 1 leu. Automatul asertează *Cafea* pentru o perioadă de ceas atunci când s-au adunat cel puţin 1,5 lei de la ultima inițializare (*Ini*).

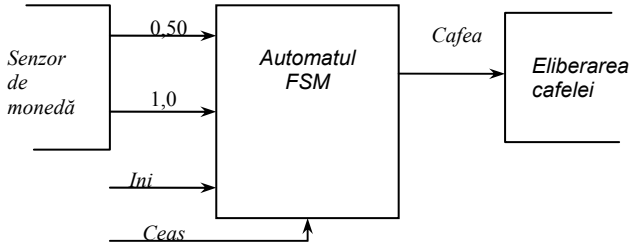

Figura 9. Diagrama bloc pentru maşina de pahare cu cafea

Specificaţiile pot să nu definească complet comportamentul maşinii. Astfel, spre exemplu, se pot ridica următoarele întrebări:

- 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 făcute anumite presupuneri.
Pentru prima întrebare, se presupune că senzorul returnează orice monedă pe care nu o recunoaşte, menţinând semnalele 0,50 şi 1,0 neasertate.
Privitor la cea de-a doua întrebare, se presupune că automatul se re-inițializează singur după ce paharul de cafea a fost eliberat.

Reprezentarea abstractă. Odată înţeles comportamentul, este timpul să se transforme specificaţiile primite 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 care duc la eliberarea paharului de cafea:

- 3 monede de 50 de bani în secvenţa : 0,5, 0,5, 0,5;
- 2 monede de 50 de bani şi 1 bancnotă de 1 leu în secvenţa: 0,5, 0,5, 1;
- 1 monedă de 50 de bani şi o bancnotă de 1 leu în secvenţa: 0,5, 1;
- 2 bancnote de 1 leu în secvența: 1, 1.

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.

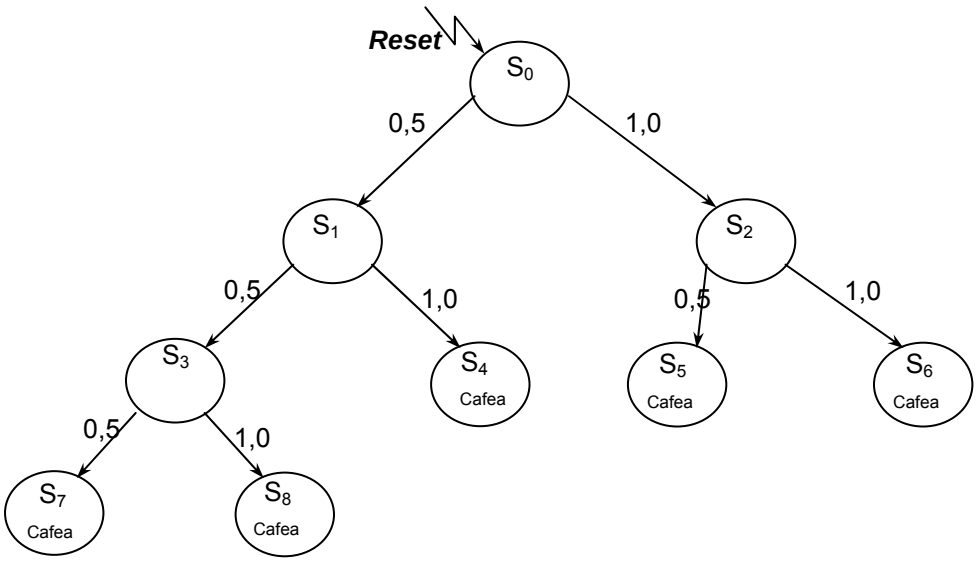

Figura 10. Diagrama stărilor pentru maşina de vândut cafea.

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 aşa cum a fost enunţată 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 una optimă posibilă. Stările S_4, S_5, S_6, S_8 au comportament identic şi pot fi combinate într-o singură stare, spre exemplu.

Pentru reducerea numărul de stări şi mai mult, se poate să se conceapă fiecare stare ca fiind reprezentată prin suma acumulată până în acea stare (respectiv, nod din graful stărilor). Ar trebui să nu aibă importanţă faptul că s-a ajuns în starea corespunzătoare sumei 1.0 lei prin introducerea a două monede de 50 de bani sau prin introducerea unei singure bancnote având valoarea 1 leu.

Diagrama de stări corespunzătoare, care rezultă, este reprezentată în figura 11. Se poate reprezenta comportamentul acestei maşini de vândut cafea prin numai 4 stări, comparativ cu 9, câte erau în figura 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 "1,50 1,0" asociată acestei tranziţii ca fiind "*se trece în starea 1,5 lei dacă 1,50 sau 1,0 sunt asertate*".

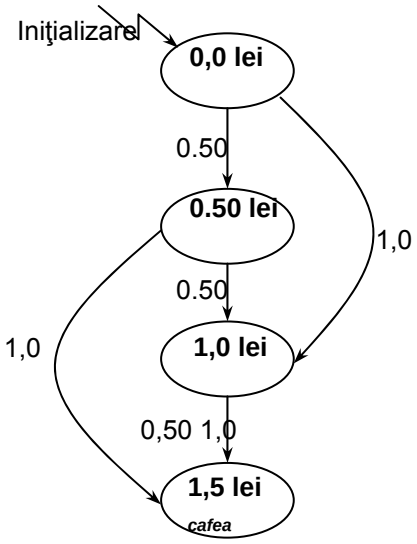

Figura 11. Diagrama minimizată a stărilor.

Într-o următoare abordare, vor fi considerate metodele formale de stabilire a diagramei de stări minimale. Procedeul de minimizare a diagramei asociate automatului poartă denumirea generică *minimizarea numărului de stări*.

Codificarea stărilor. În acest moment s-a determinat un automat finit, cu număr minim de stări, dar aflat încă la un nivel simbolic. În figura 12 este prezentat tabelul tranziţiilor de stare. Următoarea etapă este *codificarea.*

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

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 sumei 50 bani ca fiind 01, starea corespunzătoare

valorii 1,0 lei ca fiind 10, şi starea aferentă valorii 1,5 lei ca fiind 11. O codificare mai puţin evidentă posibil să producă o configuraţie hardware mai simplă. Tabelul tranziţiilor de stări codificate este reprezentat în figura 13.

Starea curentă		Intrări		Starea următoare		Ieșire
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	0	0	1	1	1
		0	0	1	1	1
		1	0	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 *D* şi *JK*. Pentru intrările etichetate până acum prin 1,5 şi respectiv 0,5 s-a utilizat în continuare notaţia *L* (Leu) şi respectiv *B* (Bani). Ecuaţiile minimizate, în cazul bistabililor *D,* sunt:

$$D_1 = Q_1 + L + Q_0 * B$$
$$D_0 = B * Q_0 ' + Q_0 * B ' + Q_1 * B + Q_1 * L$$
$$CAFEA = Q_1 * Q_0$$