LABORATOR 5

Proiectarea în VHDL a automatelor finite

1. Introducere teoretică

Primul pas în proiectarea unui automat finit (maşină cu stări finite – sau Finite State Machine -FSM) este luarea unei decizii asupra tipului de circuit care va fi proiectat: va fi un automat Moore sau unul Mealy. În continuare se vor prezenta pe scurt aceste noțiuni pentru a putea ajunge la obținerea unui tabel de tranziție a stărilor. *Prin definiție, ieşirile unei maşini Moore depind numai de starea curentă a automatului şi sunt independente de semnalele de intrare. La o maşină Mealy, ieşirile depind atât de starea curentă cât şi de intrări*. Din punct de vedere funcțional, este întotdeauna posibil să se implementeze orice specificație fie sub forma unei maşini Moore fie sub forma unei maşini Mealy. Diferența importantă apare în sincronizarea ieşirilor. Există trei efecte practice care trebuie luate în considerare:

1. La o maşină Moore, ieşirile îşi stabilesc valorile finale mai târziu (cu câteva întârzieri de porți logice) decât apariția frontului activ al semnalului de ceas. Ieşirile sunt constante pentru restul perioadei curente de ceas, chiar dacă intrările se schimbă în acest timp. Totuşi, pentru că ieşirile sunt independente de intrările curente, orice efect pe care intrările l-ar putea induce este amânat până la următoarea perioadă de ceas. Un avantaj al unui automat Moore este acela că izolează ieşirile de intrări.

2. La o maşină Mealy, pentru că ieşirile depind de intrări, o ieşire se poate schimbe chiar ín mijlocul unei perioade de ceas dacă se modifică valoarea unei intrări. Acest lucru permite unei maşini Mealy să răspundă cu o perioadă de timp mai repede decât o maşină Moore la schimbările intrărilor, dar permite ca ieşirile să urmărească tranzițiile false ale intrărilor, de exemplu, zgomotul pe liniile de intrare poate fi transferat la ieşiri.

3. O maşină Moore poate avea mai multe stări decât maşina Mealy echivalentă.

O anumită specificație poate impune o structură Moore sau una Mealy, sau poate permite proiectarea ambelor forme. În mod frecvent, alegerea este impusă de experiența proiectantului.

1.1 Construirea tabelului de tranziție a stărilor

După ce a fost selectat un automat Moore sau Mealy ca dispozitiv țintă, putem începe construirea tabelului de stări. Această activitate impune folosirea talentului creativ al proiectantului. Există foarte multe potențiale tabele de stări care să îndeplinească specificațiile

impuse. Deşi nu există un algoritm formal pentru construirea unui astfel de tabel, proiectanții vor avea o mai mare şansă de succes dacă urmează o metodologie structurată.

Deseori, proiectanții folosesc două unelte pentru a simplifica activitatea de construire a unui tabel de stări: diagramele de stări şi listele de tranziție. Vom ilustra în continuare ambele tehnici. O diagramă de stări oferă o descriere grafică, uşor de înțeles, a operării unui automat finit, dar se limitează la un număr relativ mic de dispozitive. Tehnica listelor de tranziție se foloseşte atunci când problema este prea complexă pentru a putea fi construită o diagramă de stări. Exemple de liste de tranziții sunt prezentate în exemplelel din figurile 1.a şi 2.a.

1.2 Obținerea unei diagrame de stări

Diagrama de stări este o reprezentară vizuală care ajută la înțelegerea relațiilor dintre stările circuitului. Dacă se dă o secvențp de intrări, va fi relativ uşor să se obțină secvența corespunzătoare de tranziții de stare.

Pe măsură ce se dezvoltă descrieri ale stărilor, este convenabil să se deseneze o diagramă de stări prin care să se vizualizeze funcționarea circuitului. Diagrama de stări este un graf orientat care are ca noduri stările circuitului. In interiorul fiecărui nod apare numele stării respective. Pentru o maşină Moore, în interiorul nodului se trec şi valorile ieşirilor. Tranziția de la o stare X la o stare Y este indicată printr-un arc orientat de la nodul X la nodul Y. Fiecare arc este etichetat cu condițiile de intrare care determină tranziția de stare respectivă, care apare simultan cu frontul crescător (sau descrescător) al semnalului de ceas. Dacă diagrama de stări se construieşte pe măsură ce se scriu descrierile stărilor, ea devine un ajutor important în procesul de proiectare. Exemple de diagrame de stări sunt cele din figurile 1.b şi 2.b.

Trebuie să se analizeze fiecare stare nouă pentru a determina tranzițiile necesare pentru toate combinațiile posibile de intrare. Aceste tranziții sunt adăugate pe diagrama de stări sub forma unor noi arce între noduri existente sau adăugând noduri noi dacă este nevoie. Acest proces continuă până când sunt analizate toate nodurile.

Pentru a construi o astfel de diagramă, se începe cu o stare care este uşor de descris prin cuvinte. Dacă există o stare *reset*, aceasta reprezintă o bună stare de început. Se recomandă scrierea unor descrieri complete, în cuvinte, pentru fiecare stare care este creată. Această modalitate permite referirea ulterioară, pe măsură ce proiectul avansează, şi oferă o informație importantă pentru documentarea proiectului final. Procesul folosit este iterativ. Adică, descrierea unei stări poate să apară necesară în timp ce proiectul evoluează, iar descrierea finală poate fi diferită de aceea scrisă mai înainte. Totuşi, este posibil să se revizuiască descrierea unei stări, scrisă mai înainte, pentru a respecta noile condiții, în loc de a crea o stare complet nouă.

Principiul excluderii mutuale - În procesul de proiectare, pentru a verifica diagramele de tranziție şi pentru a corecta eventualele erori, se foloseşte principiul excluderii mutuale. Adică, expresiile logice cu care sunt etichetate arcele care pornesc din orice nod trebuie să fie mutual exclusive. Aceasta înseamnă că, două expresii de pe arce care pornesc din acelaşi nod

nu pot fi adevărate simultan. în cazul în care două asemenea expresii ar fi adevărate simultan, malina ar încerca să comute în două stări diferite, ceea nu se poate întâmpla.

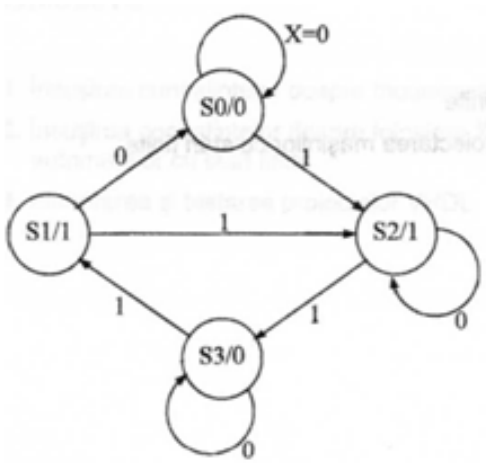
Testul pentru excluderea mutuală este: funcția AND între oricare două expresii de pe arce care pornesc din acelaşi nod trebuie să fie **0.**

2. Modelarea VHDL a automatelor finite

În Figura 1 sunt prezentate specificațiile unui automat finit de tip Moore. Automatul are o intrare (X), patru stări (SO, SI, S2, S3) şi o ieşire (Z). în Figura 2 este prezentat modelul VHDL al acestei maşini. Descrierea include două procese: un process defineşte partea combinațională (instrucțiune *case* pentru asignarea stărilor). Al doilea process defineşte elementele sincrone ale automatului (registru de stare).

Stare curentă	Stare următoare		Ieşire
	X=0	X=1	Z
S0	S0	S2	0
S1	S0	S2	1
S2	S2	S3	1
S3	S3	S1	0

a) Tabelul de tranziție



b) Graful de tranziție a stărilor

Fig. 1 Specificațiile unui automat Moore.

```
entity MOORE is
port  (X,  CLOCK:  in std_logic;
          Z:  out std_logic );
end MOORE;

architecture BEHAVIOR of MOORE is
type STATE_TYPE is   (S0,  S1,  …);
signal CURRENT_STATE,  NEXT_STATE:  STATE_TYPE;
begin
     -- Proces pentru secțiunea combinationala
COMBIN: process   (CURRENT_STATE,  X)
begin
case CURRENT_STATE is

when S0 =>
```

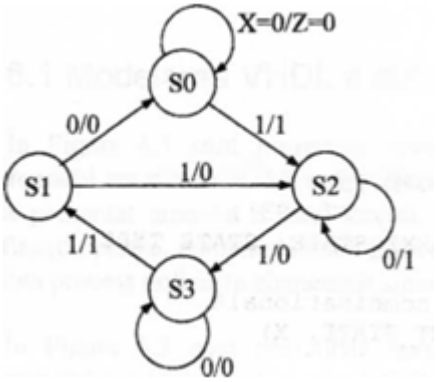
```
     Z<= … ;
   if X = …     then
          NEXT_STATE <= S0;
     else
          NEXT_STATE <= …;
end if;

when S1 =>
          Z <= '1' ;
     if X = …   then
          NEXT_STATE <= …     ;
     else
          NEXT_STATE <= …     ;
end if;
      -- toate stările curente


end case;
end process COMBIN;
--   Proces pentru secţiunea secvenţiala  (bistabile)
SYNCH: process begin
wait until CLOCK'event     and CLOCK =… ;
  CURRENT_STATE <= NEXT_STATE;
  end process SYNCH;

end architecture;
```

Fig. 2 Modelul VHDL al automatului Moore

În Figura 3 sunt prezentate specificațiile unui automat finit de tip Mealy. Automatul are aceleaşi elemente definitorii ca şi automatul Moore: o intrare (X), patru stări (SO, SI, S2, S3) şi o ieşire (Z). În Figura 4 este prezentat modelul acestei maşini care include aceleaşi două procese ca mai înainte: un process defineşte partea combinațională a automatului (instrucțiune case pentru asignarea stărilor). Al doilea process defineşte elementele sincrone ale automatului (registrul de stare).

Stare curentă	Stare următoare		Ieşire Z	
	X=0	X=1	X=0	X=1
S0	S0	S2	0	1
S1	S0	S2	0	0
S2	S2	S3	1	0
S3	S3	S1	0	1

a) Tabelul de tranziție



b) Graful de tranziție a stărilor

Fig. 3 Specificațiile unui automat Mealy

```
entity Mealy is
port  (X,  CLOCK:  in std_logic;
       Z:  out .... );
end Mealy;

architecture BEHAVIOR of Mealy is
type STATE_TYPE is   (S0, ....;
signal CURRENT_STATE,  NEXT_STATE:  STATE_TYPE;
begin
  -- Proces pentru secțiunea combinationala
COMBIN: process   (....) begin
case CURRENT_STATE is

when S0 =>
     if X = ... then
       Z<= '0';
            NEXT_STATE <= ...;
  else
       Z<= ...
            NEXT_STATE <= ...;
  end if;

when S1 => ...

   --toate starile curente

end case;
end process COMBIN;
--     Proces pentru secțiunea secvențiala  (bistabile)
```

```
   SYNCH: process begin
   wait until .... ;
     CURRENT_STATE <= ...;
     end process SYNCH;

   end architecture;
```

Fig. 4 Modelul VHDL al automatului Mealy

1. Completați modelele din Figura 2 şi din Figura 4 (două proiecte), compilați programele şi simulați funcționarea celor două automate.
2. Elaborați un raport scris care să conțină modelele VHDL complete (comentate) şi formele de undă care demonstrează funcționarea corectă a automatelor în toate cazurile posibile (conform grafului sau tabelului de tranziție).