Proiectarea subsistemelor numerice VLSI, circuite numerice fundamentale.

Probleme principale:

- gasirea unor celule de baza care pot implement functiile primitive de baz ale sistemului;

Celula/celulele de baza dicteaza:

- geometria celulei: dimensiunile si factorul de forma;
- un set de criterii/reguli de sincronizare.

Geometria celulei si criteriile de sincronizare sunt utilizate pentru specificarea cerintelor privind circuitul de comanda; acesta inconjoara aria constituita din celulele de baza, care asigura prelucrarea semnalelor.

Dupa amplasarea circuitelor de comanda, rezulta un subsistem, un modul functional, cu o interfata binedefinita privind:

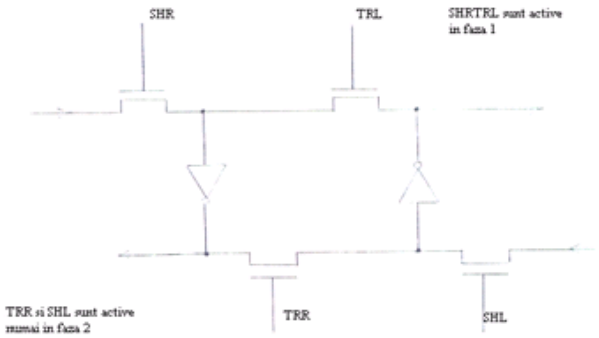
- specificatiile functionale;
- specificatiile geometrice;
- specificatiile de I/E, pentru date;
- specificatiile de comanda si sincronizare.

Setul de cerinte de sincronizare are in vedere:

- intrarile de date;
- iesiruile de date;
- intrarile de comanda

Exemplu: Proiectarea unei memorii de tip stiva)LIFO)

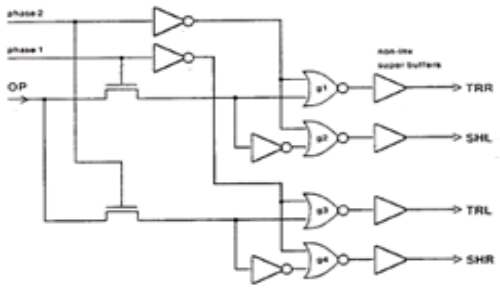
Celula de baza este capabila sa realizeze operatiile:Push, Pop, Nop, in conditiile utilizarii unui ceas bifazic si al unui cod de operatie de un bit.



Randul de sus va fi activat pe durata fazei1, iar ransul de jos pe durata fazei2. Fluxul de comenzi este vertical, pe trasee metalice, iar fluxul de date este orizontal pe linii de difuzie.

- Deplasare dreapta: PUSH:
  faza1: SHR=1 si faza2: TRR=1;
- Deplasare stanga: POP:
  faza2: SHL=1 si faza1: TRL=1;
- Memorare: NOP:
- faza1: TRL=1 si faza2: TRR=1;

Unitatea de Comanda:

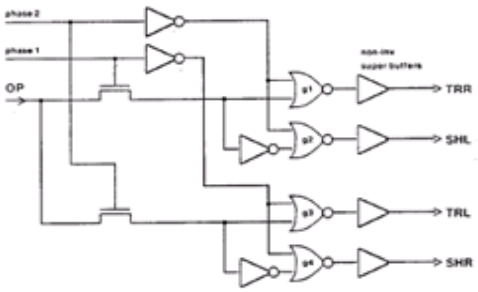


Schema unitatii de comanda la nivelul tranzistoarelor si al portilor logice.

Randul de sus va fi activat pe durata fazei1, iar ransul de jos pe durata fazei2. Fluxul de comenzi este vertical, pe trasee metalice, iar fluxul de date este orizontal pe linii de difuzie.

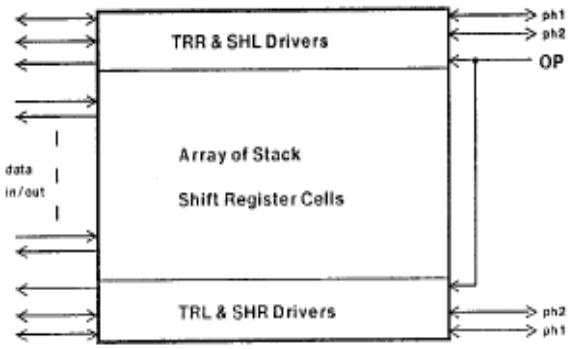
- Deplasare dreapta: PUSH:
  faza1: SHR=1 si faza2: TRR=1;
- Deplasare stanga: POP:
  faza2: SHL=1 si faza1: TRL=1;
- Memorare: NOP:
- faza1: TRL=1 si faza2: TRR=1;

Unitatea de Comanda:



Schema unitatii de comanda la nivelul tranzistoarelor si al portilor logice.

Schema bloc a unei celule de memorie stiva, vazuta ca modul:



ph1= faza1

ph2= faza2

Proiectarea logicii combinationale si secventiale.

Pentru *logica combinationala* sunt trei clase de probleme:

Prima clasa: se cere o cantitate mica de logica.

A doua clasa: implementarea unor functii logice complexe, pentru care trebuie sa se conceapa metode de structurare topologica.

A treia clasa: Retele Logice Programabile. (Programmable Logic Arrays-PLA).

Toate aceste cazuri sunt tratate in module separate.

Cazul comenzii unui circuit format dintr-o singura celula, care poate fi multiplicata.

Se folosesc procedeele proiectarii logice traditionale, cu circuite NAND, NOR. Prin inspectie, fara a mai folosi procedeele formale de minimizare, se genereaza schema circuitului.

In principiu, utilizarea portilor statice nu este recomandata deoarece:

- nu conduce la forme regulate;
- nu asigura aria minima;
- conduce la o putere disipata relativ mare;
- nu realizeaza maximizarea numarului de functii logice pe unitatea de arie.

In cazul primei clase de probleme se recomanda utilizarea tranzistoarelor de trecere.

Celelalte doua cazuri vor fi tratate in module separate: Circuite TALLY, Weinberger, PLA

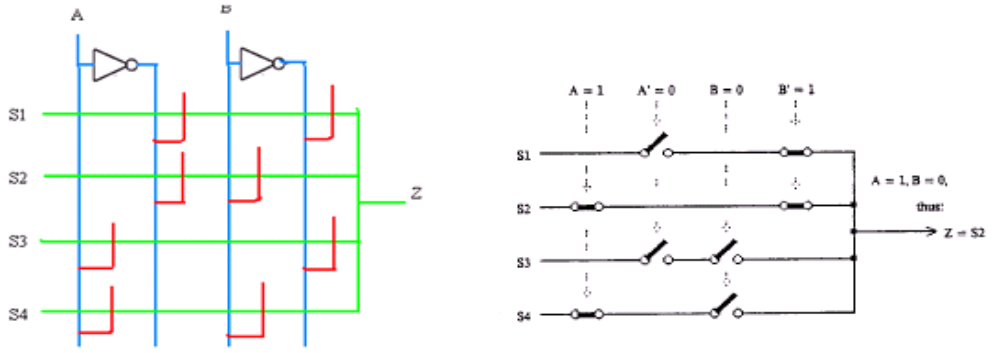
Logica secventiala necesita, pe langa retelele combinationale si elemente de memorare a starilor automatului.

In functie de modul de implementare a logicii: statica s-au dinamica, se vor folosi bistabile/registre statice sau dinamice, cat si sisteme de sincronizare (ceas) adecvate, de tip monofazic sau polifazic.

Abordarea problematicii proiectarii logicii secventiale se va face in modulele: PLA, automate finite, circuite de ceas programabil si registre statice/dinamice.

Utilizarea retelelor formate din tranzistoare de trecere.

In figura de mai jos se prezinta structura unui selector/multiplexor cu 4 intrari de date si cu 2 intrari de selectie. Structura este prezentata la nivelul diagramelor mixte de bare colorate, cat si la nivelul contactelor de relee:



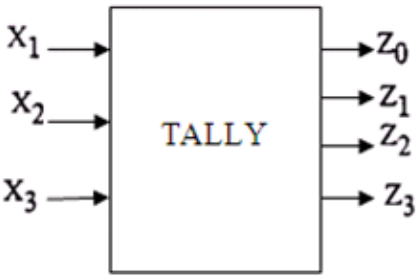
Structura realizeaza oricare functie logica de 2 variabile Z= f(A,B), prin selecatrea adecvata a intrarilor S_1, S_2, S_3, S_4:

$$Z= S_1.\overline{A}.\overline{B} + S_2.A.\overline{B} + S_3.\overline{A}.B + S_4.A.B$$

A doua clasa: implementarea unor functii logice complexe, pentru care trebuie sa se conceapa metode de structurare topologica.

Exemplu particular.

Fie circuitul TALLY (recuperat din domeniul centralelor telefonice cu relee electromagnetice), cu n intrari si n+1 iesiri, ca in figura de mai jos, pentru n=3:



Circuitul TALLY indica la cele n+1 iesiri, cate din cele n intrari sunt simultan activate, adica au valoarea logica 1. Pe aceste considerente au fost elaborat sistemul de functii logice de mai jos:

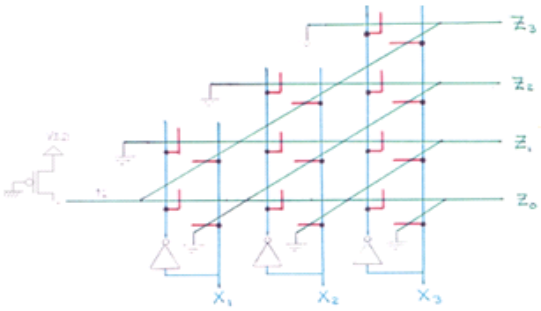
$$z_0 = \overline{x_1}\,\overline{x_2}\,\overline{x_3}$$

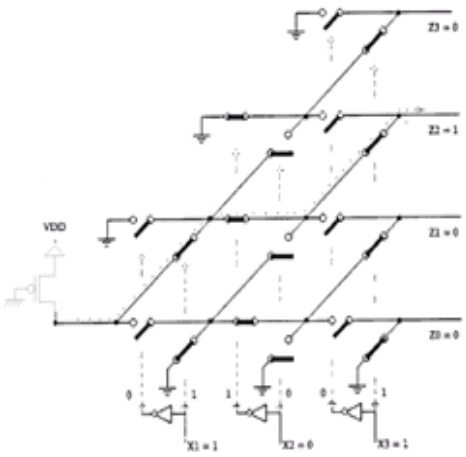
$$z_1 = x_1\overline{x_2}\,\overline{x_3} + \overline{x_1}\,x_2\,\overline{x_3} + \overline{x_1}\,\overline{x_2}\,x_3$$

$$z_2 = x_1 x_2\,\overline{x_3} + x_1\,\overline{x_2}\,x_3 + \overline{x_1}\,x_2\,x_3$$

$$z_3 = x_1 x_2 x_3$$

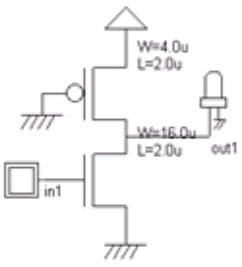
Diagrama mixta de bare colorate, care reprezinta o prima aproximatie a reprezentarii mastilor, pentru circuitul TALLY, cat si structura la nivelul contactelor de relee sunt date mai jos:



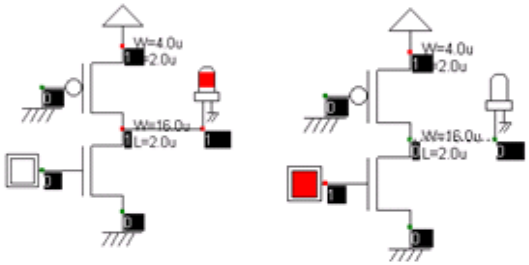


Dupa cum se constata, implementarea conduce, sub aspect topologic, la structuri regulate. De observat ca structura se bazeaza pe logica pseudoNMOS, in care se utilizeaza, in calitate de tranzistor "trage sus" un tranzistor PMOS, cu poarta legata la GND, ceea ce face ca sa fie in continuu deschis. Partea "trage jos"a structurii este constituita dintr-o retea de tranzistoare NMOS, la grilele carora se aplica semnalele de intrare, care se prelucreaza.

In figurile de mai jos se prezinta un exemplu elementar de inversor realizat in logica pseudoNMOS:



Schema inversorului pseudoNMOS



(a) in1=0, out1=1 (b) in1=1,out1=0

Rezultatele simularii logice

Pentru ca schema sa functioneze corect este necesara alegerea unui raport convenabil al geometriilor canalelor:

$$(Wpu/Lpu)/(Wpd/Lpd)= Zpu/Zpd \approx 2/8$$

Retele de tip Weinberger (RW).

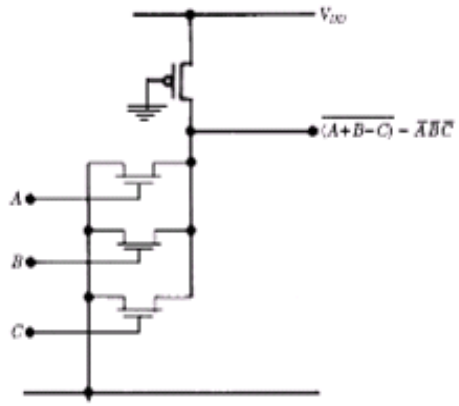
Reprezinta o abordare structurata care simplifica proiectarea structurilor VLSI si care mareste densitatea acestora.

RW sunt create prin plasarea tranzistoarelor pe chip intr-o maniera geometrica regulata. Pentru a cabla impreuna dispozitivele se folosec sabloane de interconectare verticala si orizontala. Se pot genera circuite complexe (NMOS sau pseudo NMOS) utilizand porti de tip NOR. RW sunt candidate ideale pentru generarea automata a mastilor.

Structurarea retelelor Weinberger.

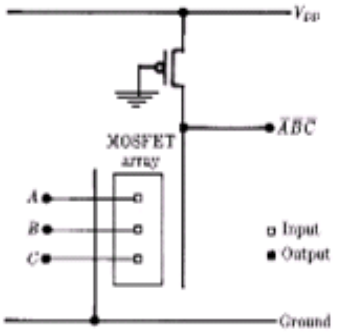
Exemplu de reducere la poarta NOR pentru structurarea RW:

$$F = \overline{(A + B + C)}$$



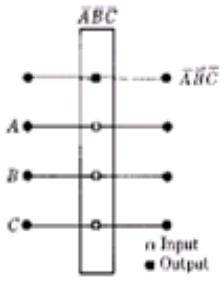
Circuitul de baza

(a)



Circuitul simplificat

(b)

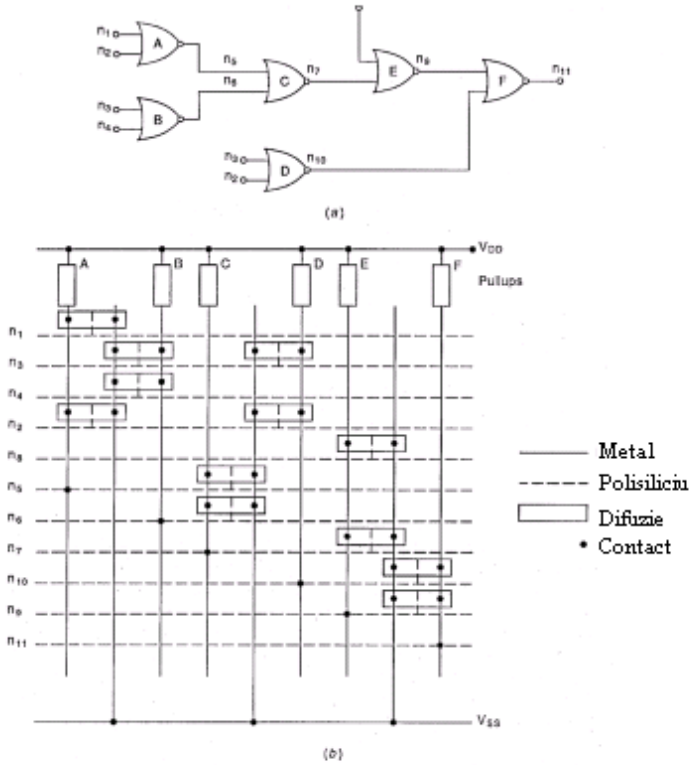


Reprezentarea redusă la nivel de poarta

(c)

Patratele albe = conexiuni de intrare
Patratele negre = conexiuni de iesire

Implementarea cu ajutorul unei retele Weinberger a unei functii logice care este realizata cu porti NOR:



Structura retelei Weinberger. (a) circuitul, (b) planul

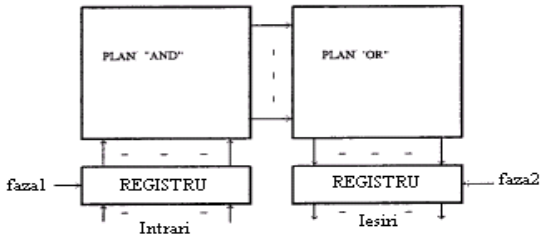
A treia clasa: Retele Logice Programabile. (Programmable Logic Arrays-PLA)

O functie logica complexa trebuie implementata fara a cunoaste aplicarea/maparea ei exacta intr-o structura regulata, de exemplu: logica combinationala plasata intre un registru sursa si un registru destinatie.

Retelele programabile permit maparea functiilor logice neregulate in structuri regulate. Functiile logice pot fi modificate substantial fara schimbari ale proiectului sau ale mastii PLA.

Utilizarea memoriilor pentru stocarea tabelelor de adevar nu este convenabila deoarece ocupa un volum foarte mare.

Schema bloc a unui PLA este data mai jos:



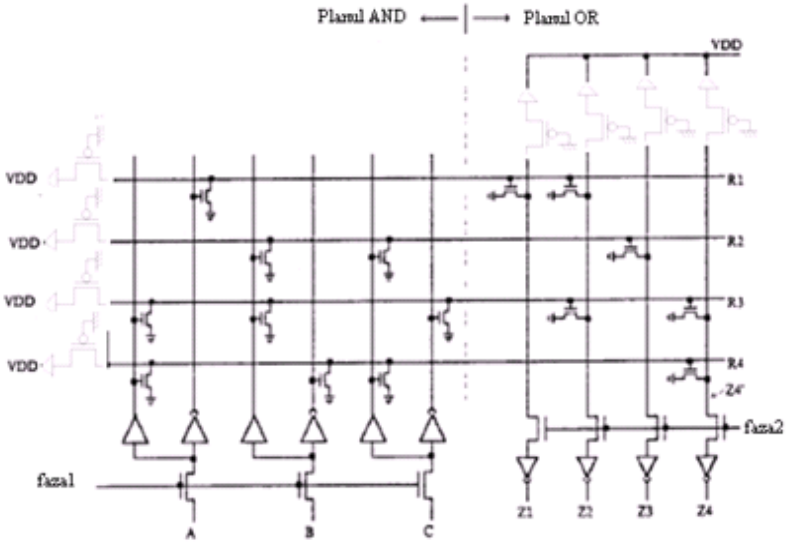
Ariile "AND"/SI si "OR" /SAU sunt realizate cu circuite NOR

Pentru exemplificarea implementarii unui sistem de functii logice, cu ajutorul unei PLA, se considera urmatorul exemplu, plecand de la sistemul de mai jos:

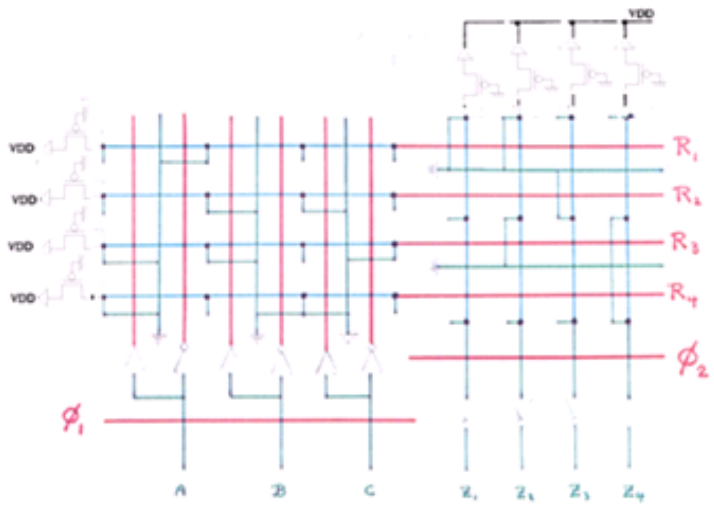
$$Z1 = A$$
$$Z2 = A + \overline{A}.\overline{B}.C$$
$$Z3 = \overline{B}.\overline{C}$$
$$Z4 = \overline{A}.\overline{B}.C + \overline{A}B\overline{C}$$

care are urmatorii implicanti primi:

$$A,\ \overline{B}.\overline{C},\ \overline{A}.\overline{B}.C,\ \overline{A}B\overline{C}$$



Structurii PLA, de mai sus, ii corespunde urmatoarea diagrama de bare colorate, care sugereaza si planul mastilor, in mare masura.



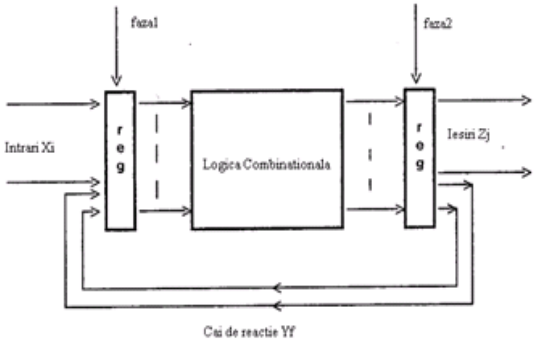
O retea logica programabila necesita existenta unei linii de elemente de circuit numai pentru acele produse, care intra in forma canonica a functiei.

Dimensiunile su forma generala depind de urmatorii parametri:

- numarul de intrari;
- numarul termenilor de tip produs;
- numarul de iesiri;
- rezolutia procesului λ.

Automate Finite.

Automatele finite se realizeaza plasand o retea logica programabila intre doua registre, in conditiile existentei unei reactii de la intrare la iesire:



Semnalele de intrare X, impreuna cu semnalele de stare Y sunt stocate in registrul de intrare reg, pe durata fazei1, a ceasului bifazic. Semnalele se propaga prin Reteaua Combinationala pana la intrarea registrului de iesire, care le stocheaza pe durata fazei2, a ceasului bifazic.

Se mentioneaza ca s-a presupus existenta unui ceas bifazic, ale carui faze nu se suprapun.

Trebuie avuta in vedere, la calculul perioadei ceasului, intarzierea in Reteaua Logica Programabila. Numarul liniilor din reactie determina numarul starilor.

Automatul este sincron si este caracterizat prin urmatoarele ecuatii:

$$Y[(k+1).T] = f[Y(k.T), X(k.T)],$$

$$Z[k.T] = g[Y(k.T), X(k.T)],$$

pentru conditiile initiale date: Y[0] si X[0].

Daca bucla de reactie ar fi permanent activa, automatul s-ar comporta asincron. Automatele sincrone functioneaza corect daca intarzierile prin circuitele de reactie sunt suficient de scurte in raport cu perioada ceasului.

Exemplu: Sinteza unui automat cu:

- 3 intrari: X_0, X_1, X_2;
- 5 iesiri: $Z_0, \ldots\ldots Z_4$;
- 4 stari: A, B, C, D codificate cu ajutorul a doua variabile de stare: Y_0 si Y_1, dupa cum urmeaza:

Y_0	Y_1	Stare
0	0	A
0	1	B
1	1	C
1	0	D

Automatul functioneaza dupa urmatoarea diagrama de tranzitii a starilor:

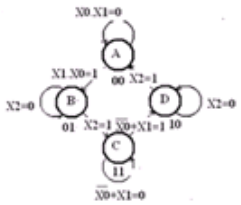
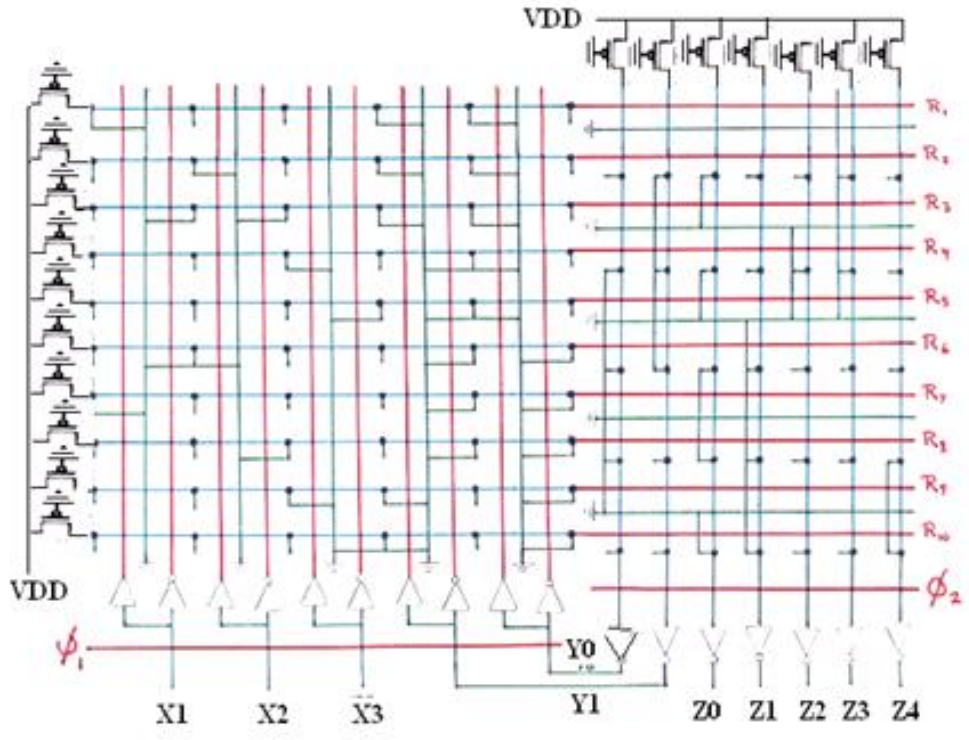


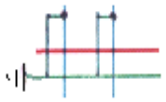
Tabela de tranzitii pentru stari si tabela de iesiri:

Intrari			Stari prezente		Stari viitoare		Iesiri					
X0	X1	X2	Y0	Y1	Y0	Y1	Z0	Z1	Z2	Z3	Z4	R
0	x	x	0	0	0	0	0	0	0	1	0	R1
x	0	x	0	0	0	0	0	0	0	1	0	R2
1	1	x	0	0	0	1	1	0	0	1	0	R3
x	x	0	0	1	0	1	0	0	1	1	0	R4
x	x	1	0	1	1	1	1	0	1	1	0	R5
1	0	x	1	1	1	1	0	1	0	0	0	R6
0	x	x	1	1	1	0	1	1	0	0	0	R7
x	1	x	1	1	1	0	1	1	0	0	0	R8
x	x	0	1	0	1	0	0	1	0	0	1	R9
x	x	1	1	0	0	0	1	1	0	0	0	R10



Reguli pentru programarea retelei logice:

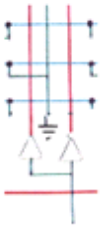
- **Regula 1** (programarea ariei OR): pentru fiecare 1 logic din tabela pentru starile urmatoare si pentru iesiri, se va trasa o cale de difuzie (verde) intre linia verticala de metal (albastru), corespunzatoare iesirii, peste traseul orizontal de polisiliciu (rosu) si traseul orizontal de difuzie (verde), ca fi figura alaturata:



- **Regula 2** (programarea ariei AND): pentru fiecare 1 logic din tabela pentru intrari si starile prezente se traseaza o cale de difuzie (verde) de linia de metal (albastra) corespunzatoare termenului produs, peste intrarea inversata (traseu de polisiliciu – rosu), la traseul vertical de difuzie (verde), conectat la masa:

X2

– **Regula 3** (programarea ariei AND): pentru fiecare 0 logic din tabela pentru intrari si starile prezente se traseaza o cale orizontala de difuzie (verde), de la linia de metal (albastra) corespunzatoare termenului produs, la masa, peste intrarea directa (traseu de polisiliciu – rosu), la traseul vertical de difuzie (verde) conectat la masa:

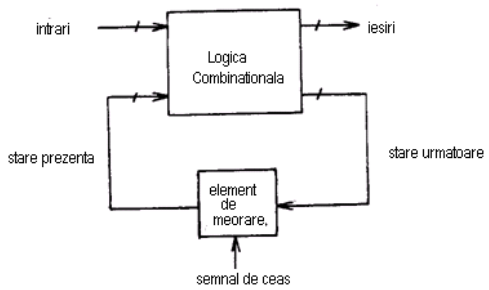


X3

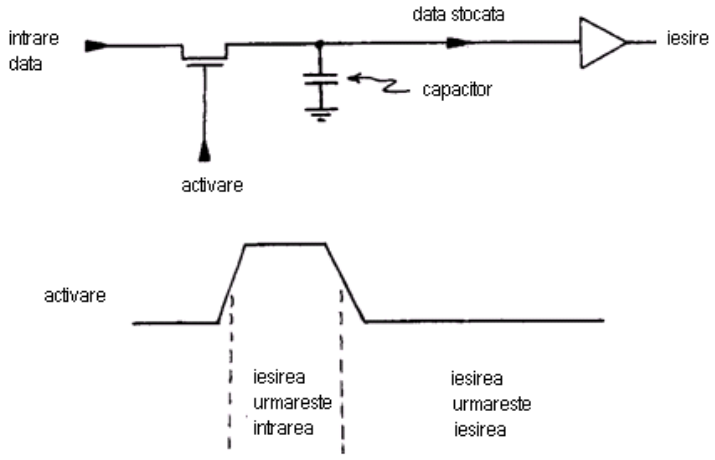
Automatul studiat mai sus se poate implementa pe o arie de $(150\ \lambda)^2$. El este format din peste 150 de tranzistoare.

Circuite de ceas programabil.

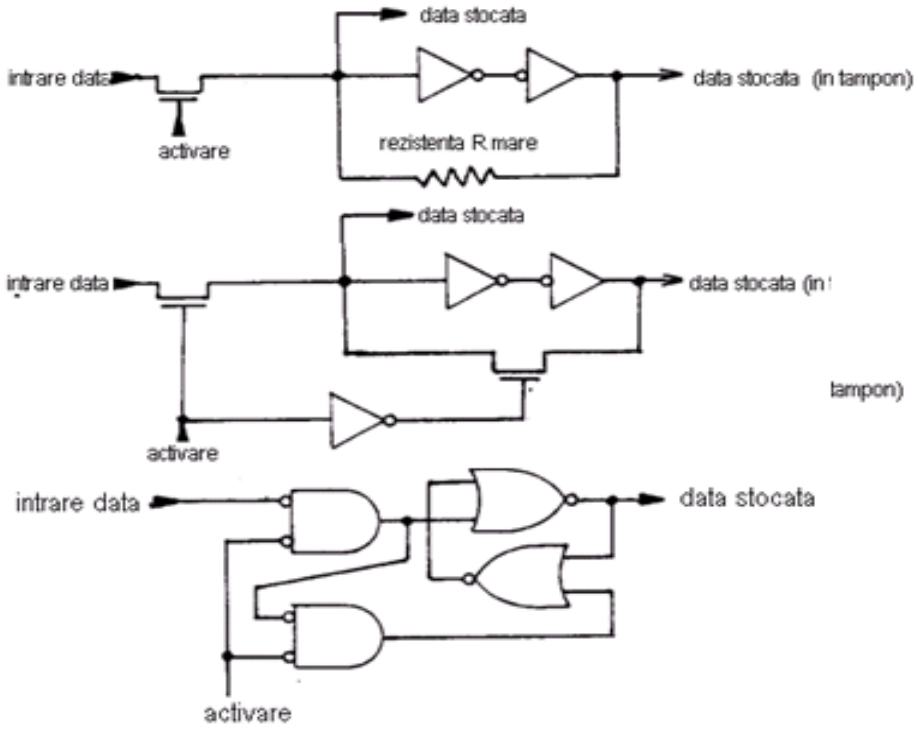
In sistemele numerice sincrone, secventa si timpul sunt conectate cu ajutorul semnalului de ceas, disponibil in sistem. Ceasul reprezinta o referinta globala a secventei si, de asemenea, o referinta globala de timp. Ca referinta de secventa, tranzitiile sale urmaresc scopul logic de a defini instantele succesive la care pot aparea modificari ale starilor. In calitate de referinta de timp, ceasul, perioada sau intervalul fix/variabil, intre tranzitiile ceasului de a tine seama de intarzierile in portile sau firele din caile dintre iesirile si intrarile elementelor sincronizate (bistabile-registre). Modelul unui sistem sincron este dat mai jos:



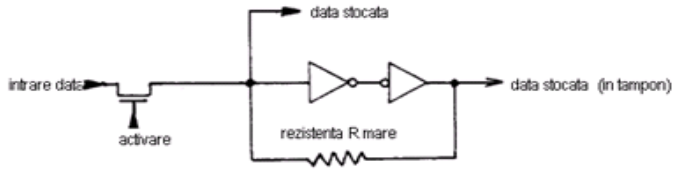
Registru dinamic de 1 bit.

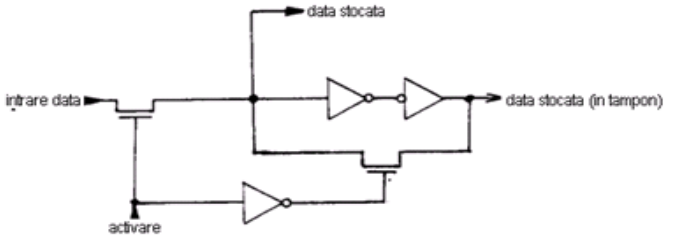


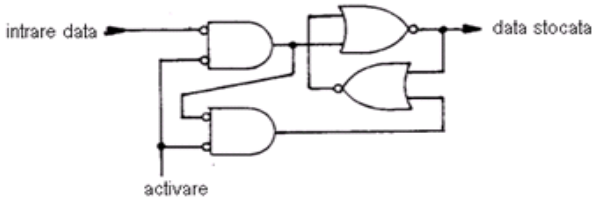
Registru static de 1 bit:

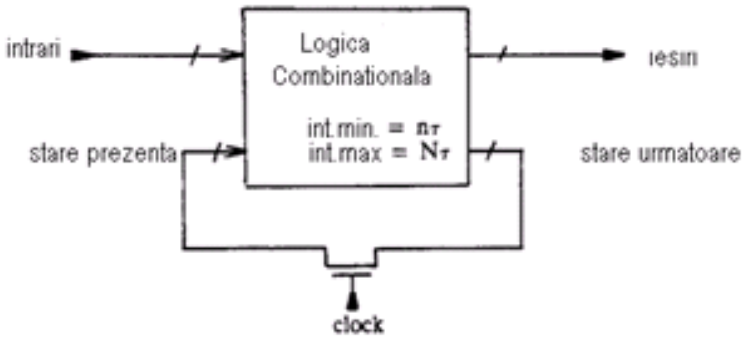


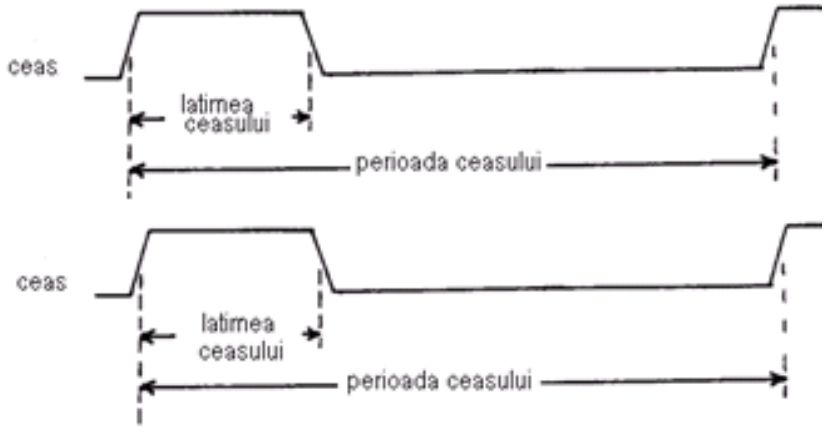
Registru static de 1 bit:





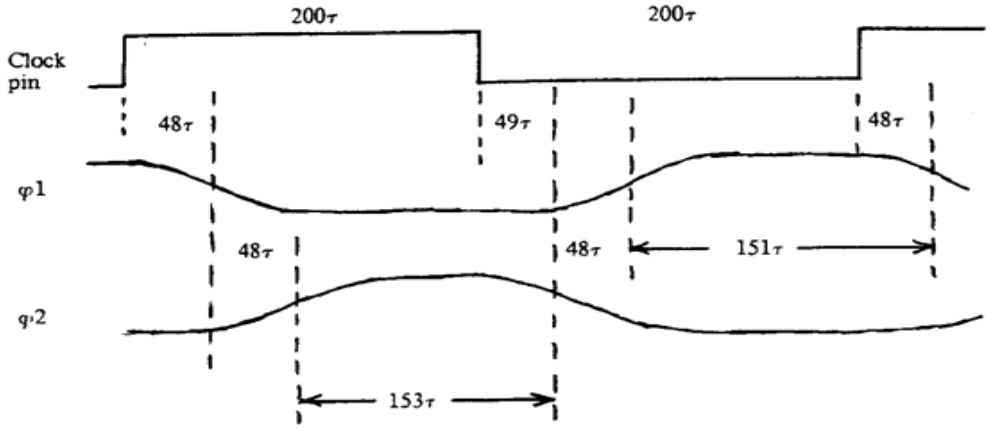




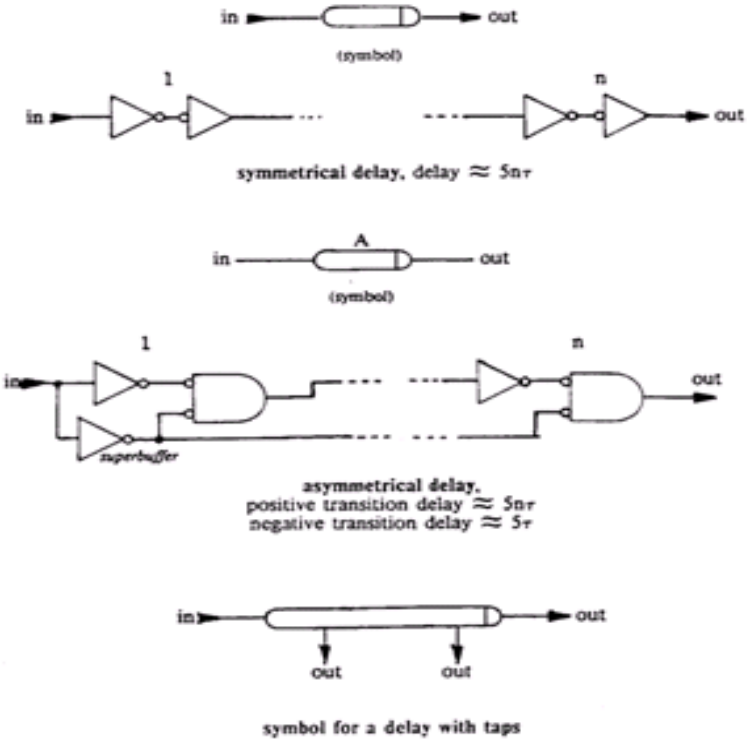


$$(R_{on}C_{in})_{max} < \begin{matrix}\text{latimea}\\ \text{ceasului}\end{matrix} < n_T + (R_{on}C_{in})_{min}$$

$$N_T < \begin{matrix}\text{perioada}\\ \text{ceasului}\end{matrix} < \begin{matrix}\text{perioada de}\\ \text{reimprospatare}\end{matrix}$$



Formele de unde ale ceasului bifazic



Intarzieri