

CALCULATOARE NUMERICE

Calculatoare 2 2024-2025 Sem. 2

CURS 5 Sapt 5 27 martie 2025 12:00-14:00

serban@upit.ro

Regulament disciplina

Nota finala este formata din activitatile:

- Laborator 30%
- Lucrare control (midterm) 20%
- Examen 50%
- Bonus – prezenta activa curs 10%

Conditii pentru promovare:

- Nota de la laborator trebuie sa fie minim 5 (prezenta obligatorie la toate sedintele de laborator);
- Nota la lucrarea de control trebuie sa fie minim 5;
- Nota de la examen trebuie sa fie minim 5.

In cazul reluarii disciplinei intr-un alt an universitar, activitatile nepromovate trebuie parcurse din nou.

Continut disciplina

Structuri digitale folosite în calculatoare numerice

Memorii semiconductoare in calculatoare numerice

Memoria cache

Memoria virtuală

Adresarea memoriei in sistemele de calcul

Bibliografie

1. **David PATTERSON, John HENNESSY** *Computer Organization and Design The Hardware/Software Interface*, 5th ed., Morgan Kaufmann Elsevier 2012 (a se vedea și traducerea în lb. română David A. Patterson, John L. Hennessy *Organizarea și proiectarea calculatoarelor, Interfața hardware/software*; Editura All, București, 2002);
2. John L. HENNESSY, David A. PATTERSON *Computer Architecture, A Quantitative Approach*, 5th ed., Morgan Kaufmann Publishers, Inc, San Francisco, 2012;
3. Bruce JACOB, Spencer NG, David WANG *Memory Systems Cache, DRAM, Disk* Morgan Kaufmann Elsevier 2008;
4. Miles MURDOCCA, Vincent HEURING *Principles of Computer Architecture*, Prentice Hill, 1999;
5. Jim HANDY *The Cache Memory*, 2nd ed., Academic Press Elsevier, 1998;
6. **Andrew S. TANENBAUM, Todd AUSTIN** *Structured Computer Organization*, 6th ed., Prentice-Hall, Inc., 2013 (a se vedea și traducerea în lb. română Andrew S. Tanenbaum *Organizarea Structurată a Calculatoarelor*, Agora, Tg. Mureș, 2004);
7. **William STALLINGS** *Computer Organization and Architecture: Designing for Performance*, 9th edition, Prentice-Hall Inc., 2013;
8. Vincent HEURING, Harry JORDAN *Computer Systems Design and Architecture*, 2nd ed., Person Prentice Hill, 2007;
9. Sajjan G. SHIVA *Computer Organization, Design, and Architecture*, 4th ed., CRC Press, Taylor & Francis Group, Boca Raton, USA, 2008;
10. Carl HAMACHER, Zvonko VRANESIC, Safwat ZAKY, Naraig MANJIKIAN *Computer organization and embedded systems*, 6th edition McGraw Hill, 2012

AUTOMATE

Automat – o structura digitala care poate evolua intr-o multime de stari logice, cu trecerea dintr-o stare anterioara intr-o stare urmatoare, in conformitate cu un algoritm bine stabilit, tinand seama de valorile logice ale unor semnale de intrare si cu generarea unor semnale de iesire.

Un automat este un cvintuplu: $A = (S, I, O, f, g)$ unde

S – multimea finita a starilor posibile

I – multimea finita a semnalelor de intrare

O – multimea finita a semnalelor de iesire

f – multimea functiilor de tranzitie care precizeaza starile viitoare in care ajunge automatul ca urmare a aplicarii unui semnal de intrare ($f : S \times I \rightarrow S$)

g – multimea functiilor de iesire care precizeaza semnalul de iesire generat in cazul aplicarii unui semnal de intrare ($g : S \times I \rightarrow O$)

Tipuri de automate: cu stari finite de tip Mealy, Moore, microprogramate

AUTOMATE MICROPROGRAMATE

Un Automat Microprogramat evolueaza pe baza unui Microprogram scris intr-o memorie (ROM) format din Microinstrucțiuni. Fiecarei stari din evolutia automatului ii corespunde o microinstrucțiune prin care se rezolva mai multe sarcini: selectarea si testarea unui semnal de intrare specific starii respective, generarea corespunzatoare a unui semnal de iesire (comanda), pregatirea trecerii in starea urmatoare, corespunzatoare algoritmului implementat.

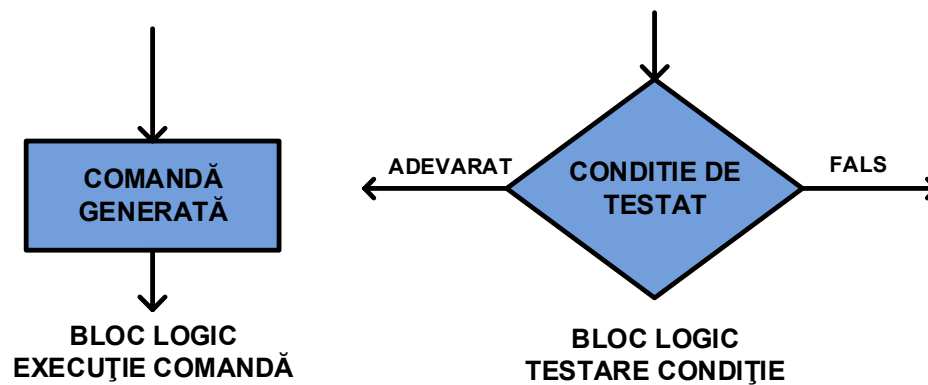
2 tipuri de automate microprogramate:

- cu format fix al microinstrucțiunilor
- cu format variabil al microinstrucțiunilor

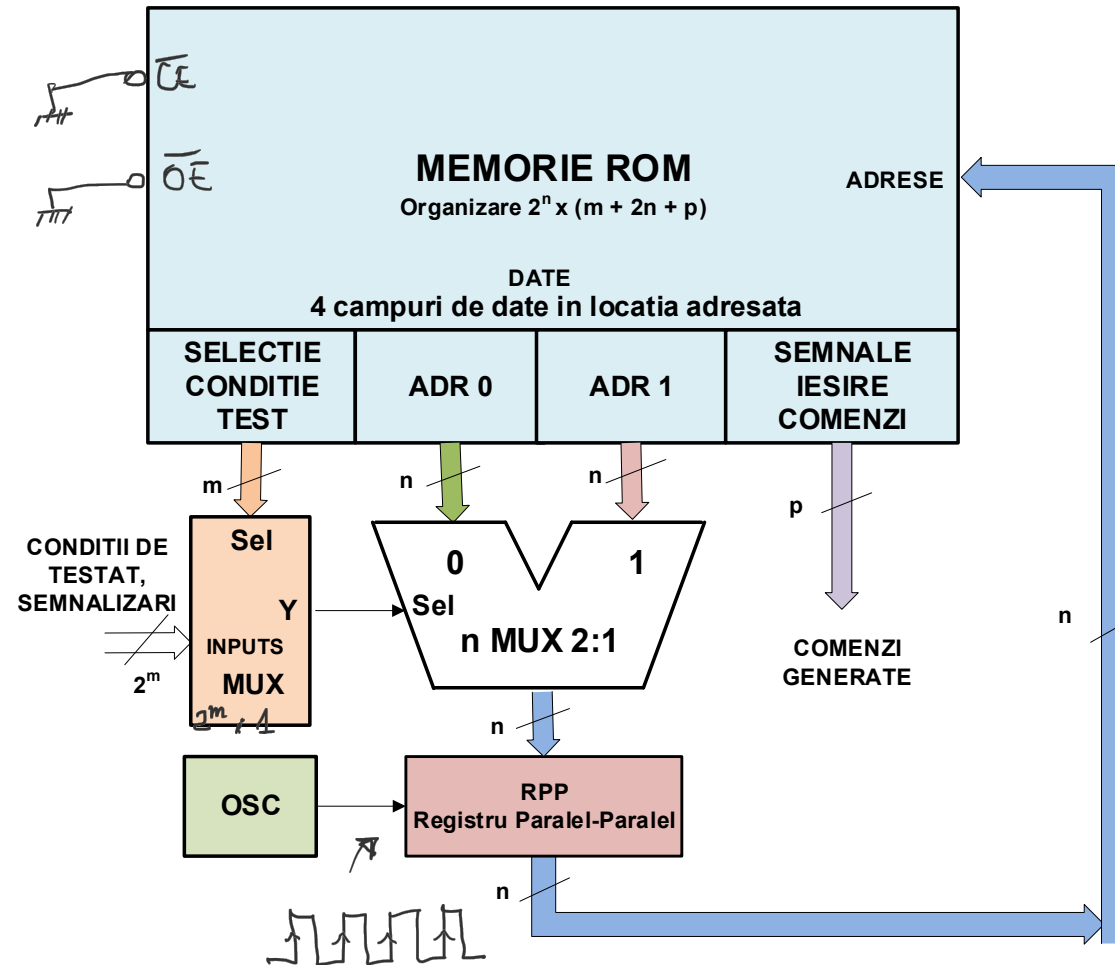
Functionarea unui automat se poate descrie prin mai multe forme:

- Limbaj natural;
- Diagrame de semnale;
- Diagrame de tranzii a starilor si iesirilor;
- Scheme logice (organigrame).

Blocuri care apar in scheme logice (organigrame)



Automat microprogramat cu format fix al microinstrucțiunilor



Caracteristici ale automatului microprogramat cu format fix al microinstrucțiunilor

Fiecare locație de memorie din memoria de microprograme corespunde unui bloc logic din organigrama. Fiecare bloc logic din organigrama reprezintă o stare în funcționarea automatului microprogramat.

Conținutul unei locații de memorie a automatului este format din mai multe campuri de date, care păstrează aceeași dimensiune, indiferent de blocul logic corespondent din organigrama.

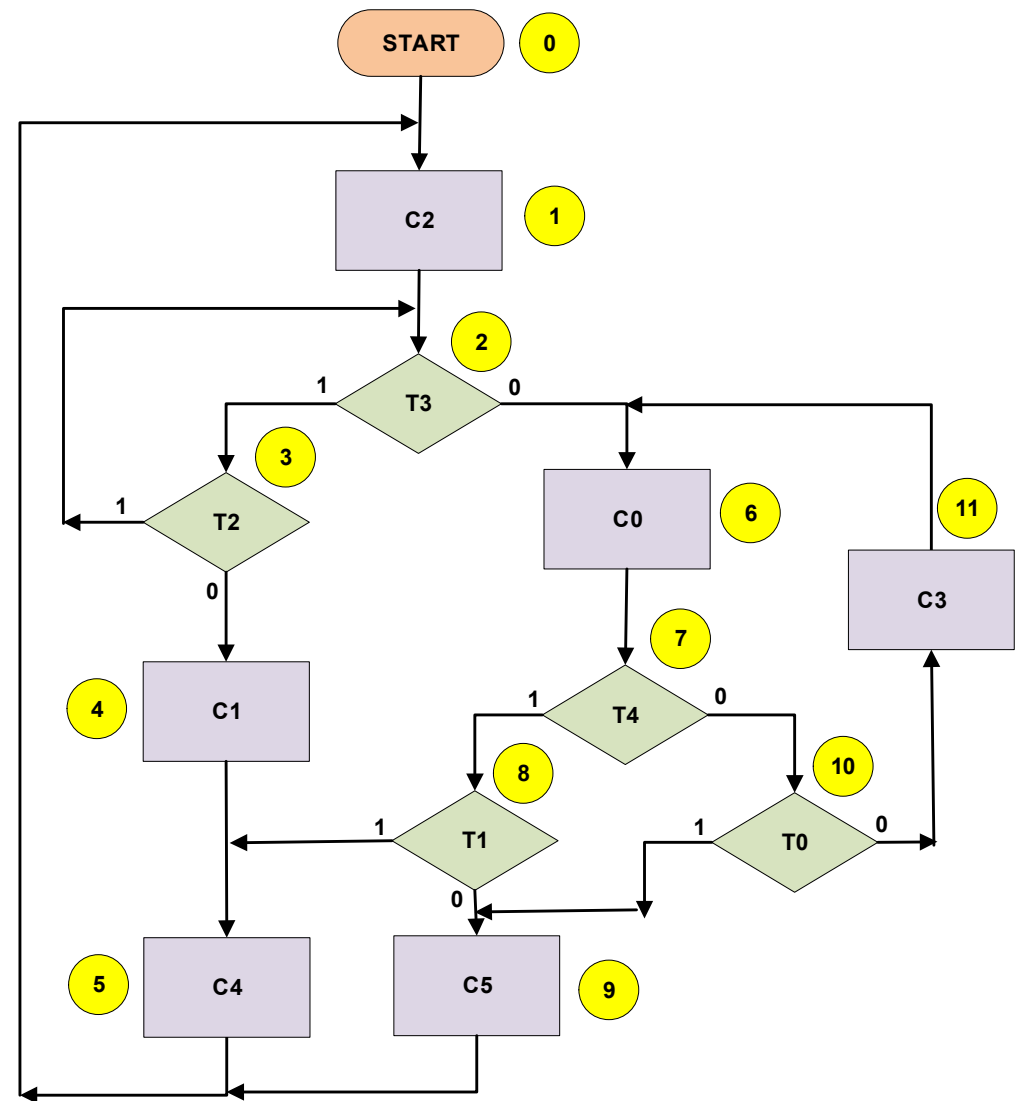
Semnificația și dimensiunea fiecărui camp dintr-o locație se păstrează pentru toate blocurile logice din organigrama (format fix al microinstrucțiunii).

La blocurile logice de tip *Generare Comanda* (reprezentate prin dreptunghi în organigrama), campurile de date *ADR0* și *ADR1* coincid (generează o aceeași adresă către memoria ROM), iar conținutul campului *Selectie Conditie Test* nu contează (don't care); campul de date *Semnale Iesire Comenzi* se completează conform organigramei.

La blocurile logice *Testare Conditie*, campurile de date *Selectie Conditie Test*, respectiv *ADR0* și *ADR1* se corelează cu organigrama; campul de date *Semnale Iesire Comenzi* se completează astfel încât comenzile de ieșire să nu fie acționate.

Aplicatie

Sa se realizeze automatul microprogramat cu format fix al microinstruciunilor care implementeaza urmatoarea schema logica (organigrama).



Implementarea ceruta de problema necesita raspunsul la 2 chestiuni:

- schema hardware a automatului; schema hardware este deja fixata, dar este nevoie de dimensionarea componentelor utilizate;
- continutul memoriei ROM din automat.

Metodologie de rezolvare:

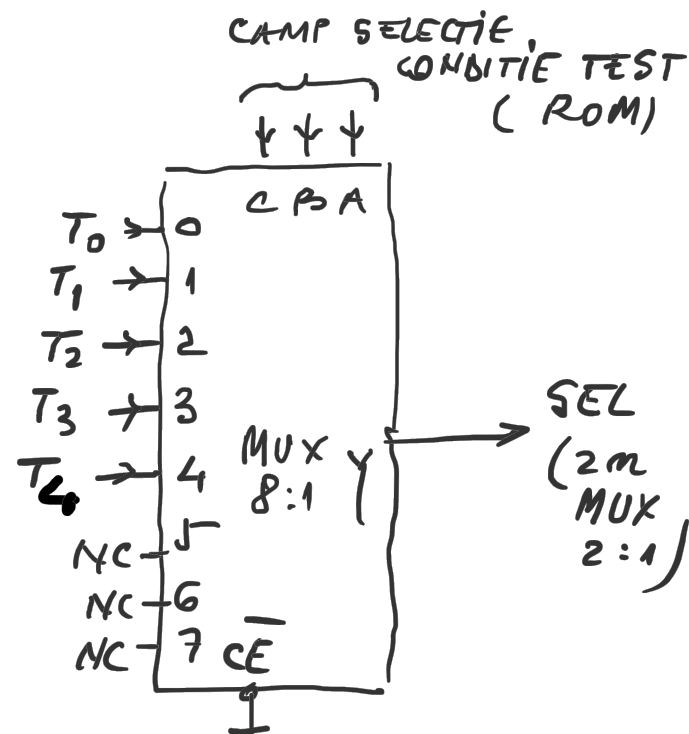
- 1).** Identificarea multimii semnalelor de intrare, dimensionarea MUX-ului care testeaza aceste semnale si alocarea lor pe intrarile MUX-ului de tip $2^m:1$;
- 2).** Identificarea multimii semnalelor de iesire (p);
- 3).** Identificarea multimii starilor prin care trece automatul (fiecarui bloc logic din organigrama ii este corespondenta o stare), de unde rezulta n , codificarea acestora, dimensionarea memoriei ROM $2^n \times (m + 2n + p)$ si a celor $nMUX2:1$ si alocarea iesirilor memoriei ROM; implementarea cu memoriei cu circuite ROM tipizate;
- 4).** Stabilirea continutului memoriei ROM.

SOLUTIE

1). Sunt 5 semnale de intrare care apar in schema logica (T_0, T_1, T_2, T_3, T_4) si care trebuie testate de automat. Rezulta necesitatea unui MUX cu 8 intrari, dintre care 5 vor fi folosite pentru semnalele mentionate anterior, iar 3 raman neconectate (NC).

La MUX-ul 8:1 vom avea 3 semnale de selectie ($m=3$) notate C, B si A. Alocam semnalele de intrare conform urmatorului tabel:

Intrare MUX 8:1	Semnal alocat	Selectie CBA
0	T0	000
1	T1	001
2	T2	010
3	T3	011
4	T4	100
5	NC	101
6	NC	110
7	NC	111



2). Semnalele de iesire care apar in schema logica sunt in numar de 6 ($C0, C1, C2, C3, C4, C5$). Rezulta $p=6$. Vom considera starea activa a semnelor de iesire ca fiind 1-logic.

3). Numarul de stari prin care trece automatul este de 12. O solutie de codificare a starilor este prezentata in schema logica. Numarul de stari este numarul de locatii din memorie folosite in implementare. Memoria ROM trebuie sa aiba minimal 16 locatii (cea mai mica putere intrega a lui 2 care acopera numarul de stari folosit). Rezulta $n=4$ (numarul de linii de adrese al memoriei ROM; le notam $A3, A2, A1, A0$). Numarul de iesiri al memoriei ROM este: $m + 2n + p = 3 + 8 + 6 = 17$ (linii de date, notate $D16 - D0$). Vor fi necesare $n = 4$ MUX-uri de tip 2:1 (notate MUX3, MUX2, MUX1, MUX0 conform structurii HW a automatului).

Alocarea celor 17 linii de date, $D16-D0$, pentru memoria ROM este prezentata in tabelul:

	Selectie semnale intrare testate MUX8:1 (camp 3)			Adresa ADR0 – adresa urmatoare pentru semnal testat cu valoarea 0-logic (camp 2)				Adresa ADR1 urmatoare pentru semnal testat cu valoarea 1-logic (camp 1)				Semnale de iesire generate Comenzi (camp 0)					
Linie date ROM	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
Semnal conectat	C	B	A	In 0 MUX 3	In 0 MUX 2	In 0 MUX 1	In 0 MUX 0	In 1 MUX 3	In 1 MUX 2	In 1 MUX 1	In 1 MUX 0	C5	C4	C3	C2	C1	C0

x-don't care

END 20 mar 2025

[illegible]

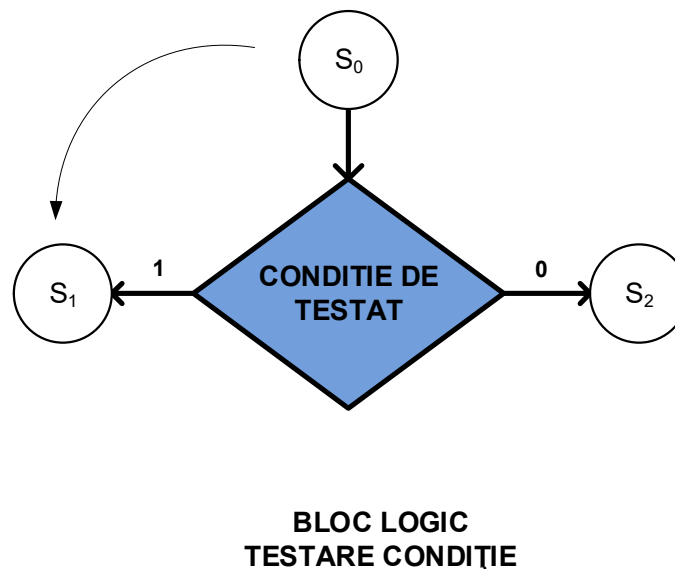
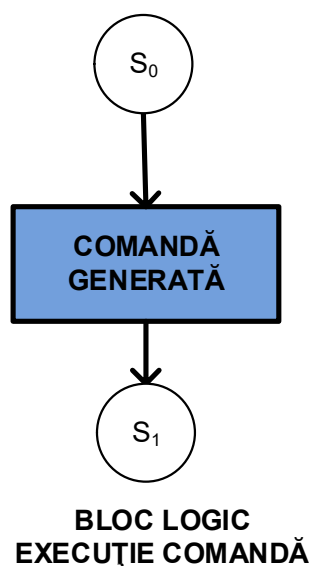
Automat microprogramat cu format variabil al microinstrucțiunilor

Automatul cu format variabil al microinstrucțiunilor urmărește optimizarea folosirii memoriei care intra in structura acestuia.

Optimizarea se bazează pe o codare specifică a stărilor pe care un automat le parcurge în cadrul unei scheme logice (organigrama), funcție de blocul logic traversat.

Astfel, la executia unui bloc logic de tip *Executie comandă*, se urmărește, pe cât este posibil, o codare a stărilor care să se obțină prin incrementare ($S_1 = S_0 + 1$), unde S_1 este starea următoare, iar S_0 este starea anterioară.

La executia unui bloc logic de tip *Testare condiție* codare va urmări ca pentru ramura de tip 1 - adevărat - starea următoare S_1 să fie obținută din starea precedentă S_0 prin incrementare ($S_1 = S_0 + 1$), iar pentru ramura de tip 0 – fals, starea următoare S_2 nu este necesar să aibă o legătură directă cu starea anterioară S_0 .



Automat microprogramat cu format variabil al microinstrucțiunilor (cont)

Starile prin care trece automatul sunt de fapt adresele aplicate memoriei de microprograme. Fiecare stare este asociata unui bloc logic din organigrama. Continuturile adresate in fiecare stare se constituie in microinstrucțiunile prin care se efectueaza functionalitatile din blocurile logice ale organigramei (executia comenzilor, testarea semnalelor de intrare).

La automatul cu format fix al microinstrucțiunilor, adresarea memoriei de microprograme se face folosind un RPP (registru paralel-paralel) care permite doar functia de incarcare paralela, fiind actionat de semnalul de ceas in ritmul caruia se parcurge organigrama.

La automatul cu format variabil al microinstrucțiunilor, implementarea hardware se bazeaza pe utilizarea unui circuit prin care se adreseaza memoria de microprograme si care permite functia de incrementare, respectiv pe cea de incarcare paralela.

Un astfel de circuit este un numerator binar presetabil.

Acest circuit trebuie sa fie si sincron pentru ambele functii (numarare – incrementare, respectiv incarcare paralela), astfel incat starile prin care trece automatul sa fie sincrone cu semnalul de ceas aplicat numeratorului si in ritmul caruia se parcurge organigrama.

Dubla functionalitate a circuitului numerator presetabil permite optimizarea memoriei de microprograme a automatului, dar implica si necesitatea unei codari a starilor parcurse prin care sa se urmareasca parcurgerea acestora prin incrementare.

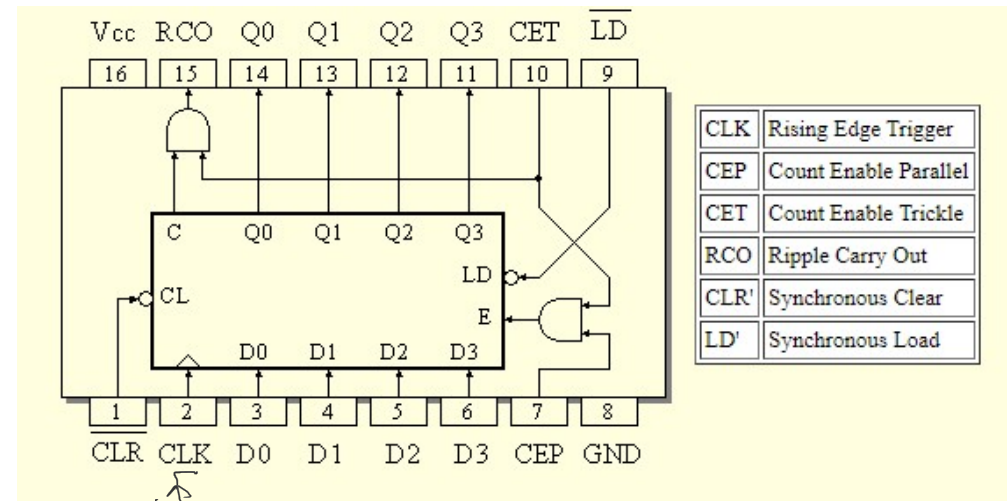
Exemple de circuite de tip numarator binar presetabil sincron

Circuitul 74163

Circuitul 74163 este un numarator binar pe 4 biti, cu functia de numarare catre inainte (incrementare), respectiv cu functia de incarcare paralela (presetare) sincrona, precum si cu functia de stergere sincrona. Cele 3 functii mentionate se realizeaza pe frontal pozitiv al semnalului de ceas (CLK).

Din tabelul de adevar al circuitului se constata ca:

- functia de stergere are prioritatea maxima (se realizeaza la activarea semnalului LD, indiferent de starea semnalelor LD, respectiv CEP si CET);
- functia de presetare are urmatoarea prioritate (se realizeaza doar daca CLR este inactiv si indiferent de starea semnalelor CEP si CET);
- functia de numarare are prioritatea minimala (se realizeaza doar daca semnalele CLR si LD sunt inactive, respectiv daca CEP si CET sunt active)



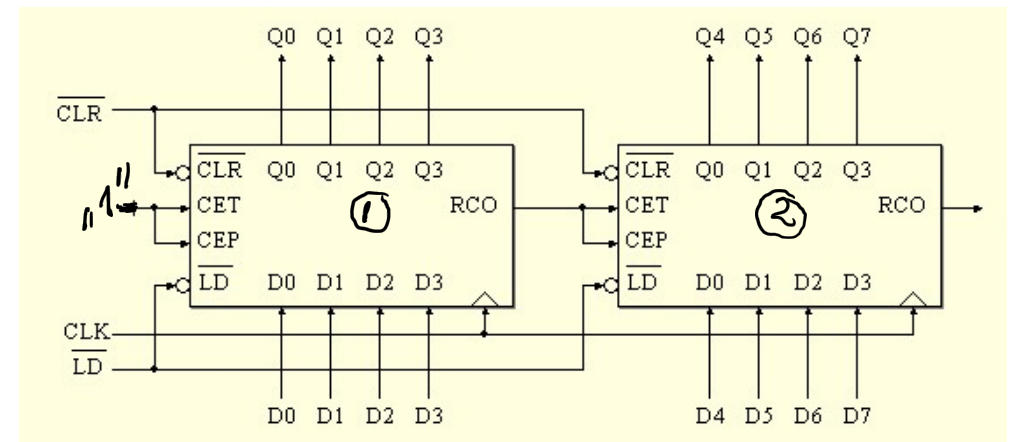
Input					Output			Operation
CLR'	LD'	CLK	Enable		Q(t-1)	Q(t)	RCO	
			CEP	CET	Q3 - Q0	Q3 - Q0		
H	H	↑	H	H	< 15	Q(t-1) + 1	L	Count Up
H	H	↑	H	H	HHHL	HHHH	H	Count Up & Carry Out
L	X	↑	X	X	-	LLLL	L	Synchronous Clear
H	L	↑	X	X	-	D3 - D0	-	Load Data
H	H	-	L	H	HHHH	HHHH	H	Hold but Carry Out
H	H	-	X	L	-	-	L	Hold

Circuitul 74163 permite realizarea de extinsii de numarare pe mai mult de 4 biti (multipli de 4 biti), care pe ansamblu au toate cele 3 functii mentionate sincrone (stergere – sau aducere la zero, incarcare paralela si numarare catre inainte).

Cel 3 functii se realizeaza pe frontul crescator al semnalului de ceas aplicat (CLK).

Se aplica acelasi semnal de ceas in mod comun circuitelor 74163 cu care se realizeaza extensia.

In figura se exemplifica o extensie pe 8 biti cu 2 circuite 74163.



Circuitul 74169

Circuitul 74169 este un numărător binar pe 4 biți, cu funcția de numărare către înainte, respectiv înapoi (reversibil), cu funcția de încărcare paralelă (presetare) sincronă.

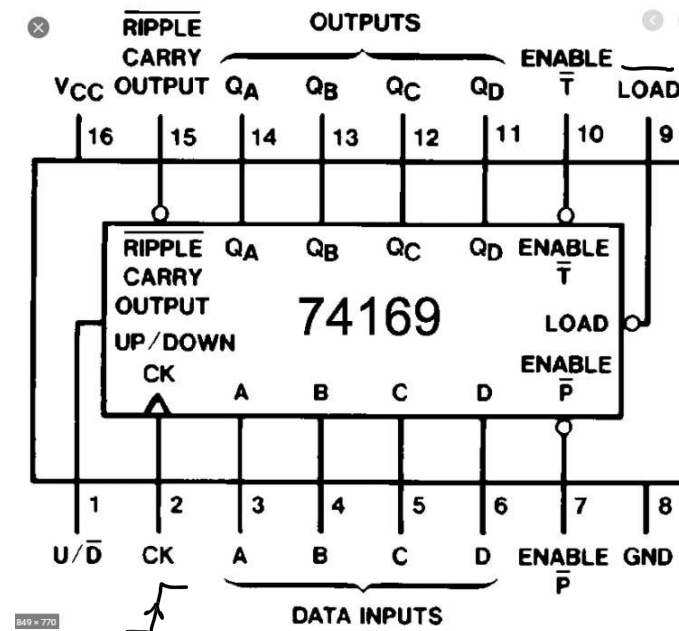
Cele 2 funcții menționate se realizează pe frontal pozitiv al semnalului de ceas (CLK).

Funcția de încărcare paralelă este prioritară față de funcția de numărare.

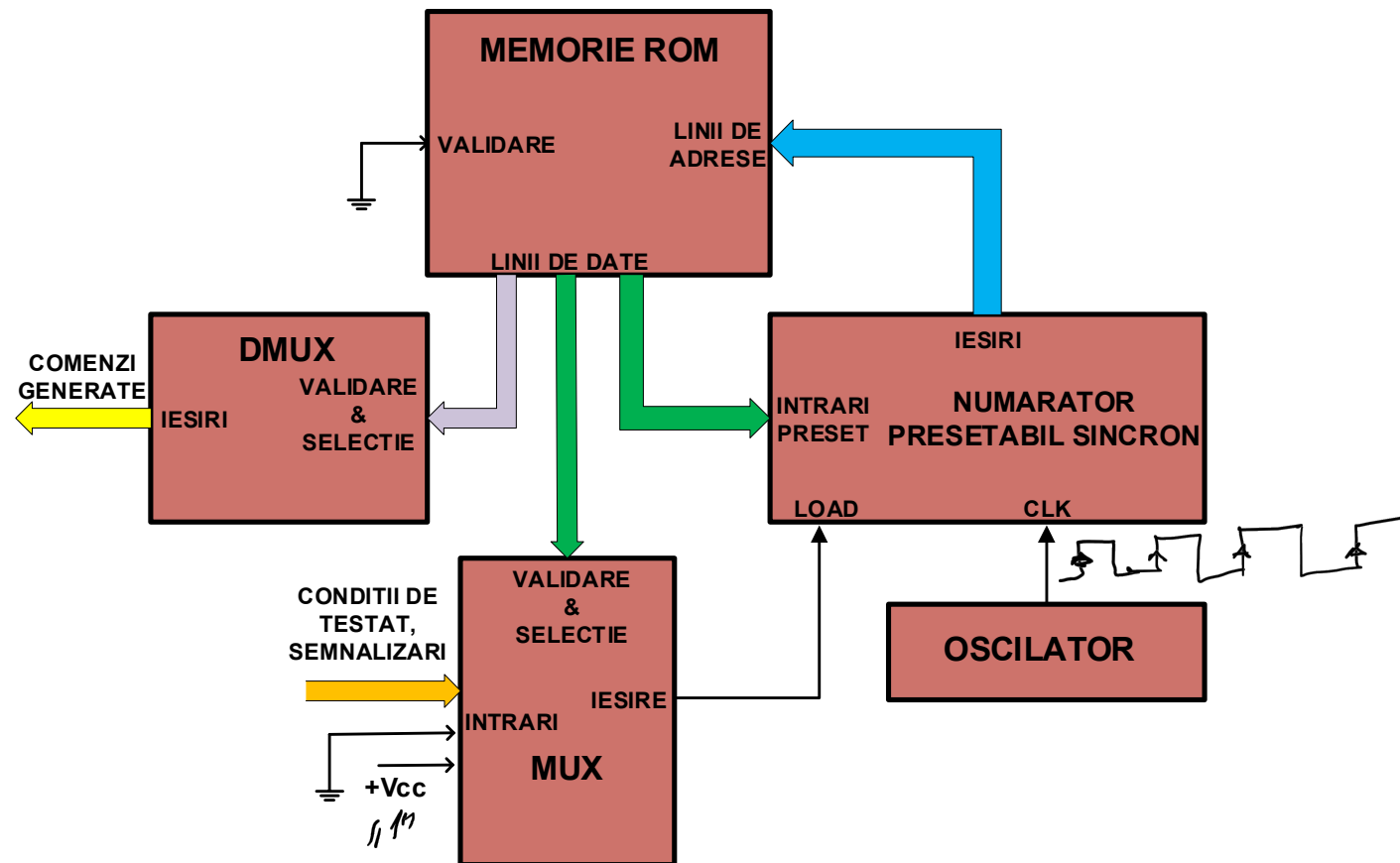
Circuitul poate realiza extensii sincrone cu funcțiile de numărare și încărcare paralelă.

Circuitul nu dispune de funcția de ștergere (clear – sau aducere la zero).

Circuitul dispune de intrarea Up/Down (U/D) prin care se stabilește sensul de numărare.



Schema bloc a automatului microprogramat cu format variabil al microinstrucțiunilor



Explicatii pentru schema bloc a automatului microprogramat cu format variabil al microinstructiunilor

Fiecare bloc logic din organigrama are asociata o stare, identificata prin adresa locatiei respective de memorie.

Orice bloc logic din organigrama se executa prin folosirea continutului unei locatii de memorie (microinstructiune).

Cum in organigrame apar 2 tipuri de blocuri logice, pentru fiecare dintre acestea va corespunde cate un format al microinstructiunii prin care se executa, campurile de biti continuti fiind diferiti in fiecare format – de unde denumirea automatului.

Microinstructiunea prin care se efectueaza blocurile logice de tip *Testare conditie*, contine 2 campuri de biti: *campul de biti care codeaza intrarea testata*, respectiv *campul de biti care reprezinta adresa de salt*, daca conditia testata indica efectuarea unui salt. Saltul se realizeaza in hardware utilizand functia de presetare a numaratorului. Campul de biti care codeaza intrarea testata este conectat la intrarile de selectie al unui MUX caruia i se aplica semnalele de ce trebuie testate. In plus, acest MUX prezinta 2 intrari suplimentare – una conectata permanent in 0-logic prin care se permite efectuarea de salturi neconditionate (prin functia de preset la numarator), iar o a 2-a conectata permanent in 1-logic care se utilizeaza cand trebuie efectuate blocuri logice de tip *Executie comanda* (prin functia de incrementare la numarator).

Dimensiunea *campului de biti care codeaza intrarea testata* este dat de relatia: $\log_2(\text{nr semnale de intrare testate}+2)$.

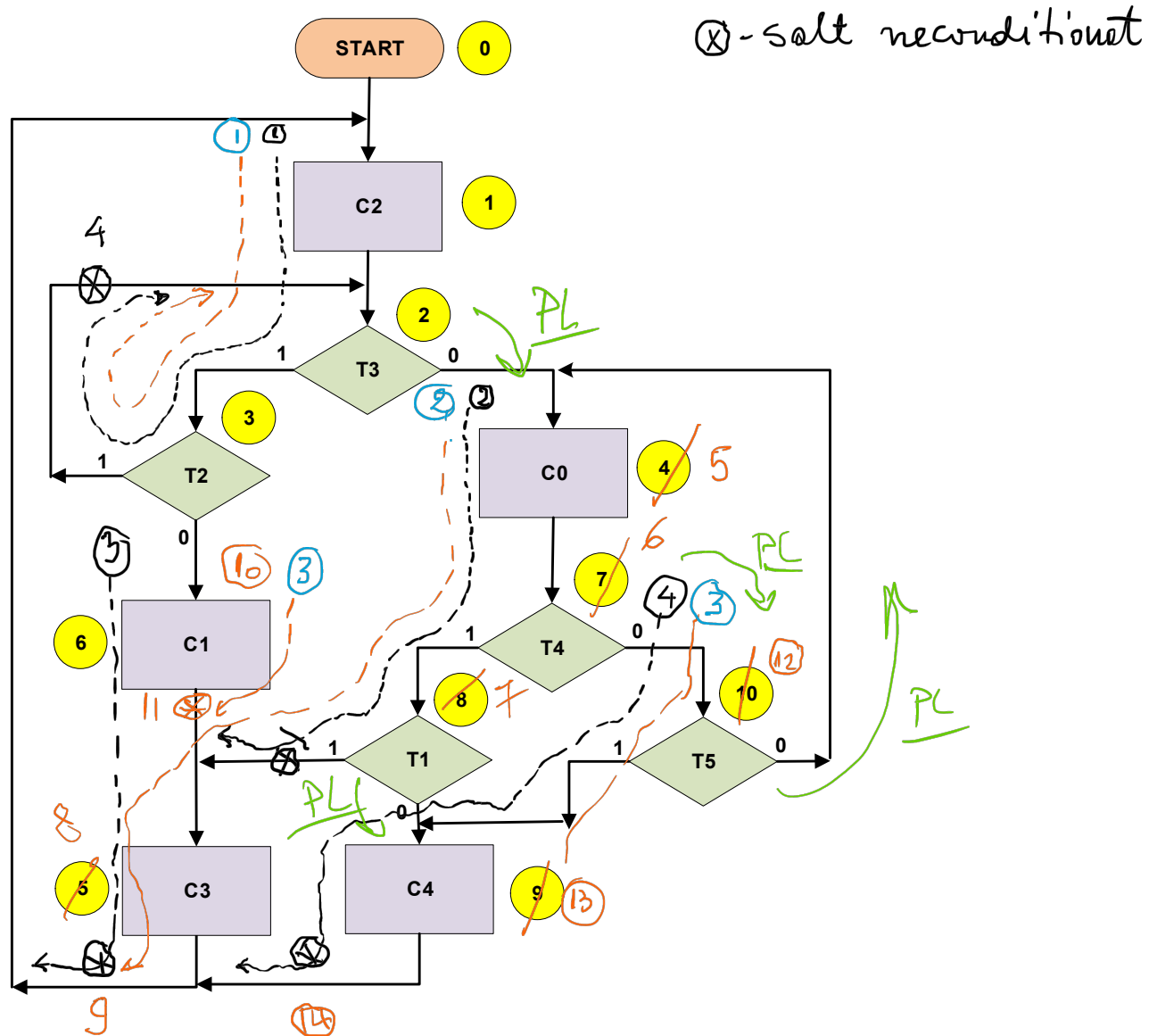
Dimensiunea *campului care reprezinta adresa de salt* este dat de relatia: $\log_2(\text{nr stari prin care se trece in organigrama})$.

In microinstructiunea prin care se efectueaza blocurile logice de tip *Executie comanda*, se regaseste un *camp de biti care codeaza iesirile ce trebuie executate*, componenta hardware utilizata pentru executia comenzilor fiind un DMUX, respectiv un camp de biti prin care se activeaza DMUX-ul si, respectiv forteaza prin MUX efectuarea unei operatii de incrementare la nivelul numaratorului (selecteaza intrarea conectata in 1-logic a acestuia).

END 20 mar 2025

Problema

Sa se implementeze prin folosirea unui automat microprogramat cu format variabil al microinstruciunilor organigrama din figura.



Implementarea ceruta de problema necesita raspunsul la 2 chestiuni:

- schema hardware a automatului – este deja fixata, dar daca situatia impune, este necesara modificarea stucturii automatului si a dimensiunii componentelor utilizate; este nevoie de precizarea alocarii (asignarii) semnalelor de intrare si cele de iesire;
- continutul memoriei ROM din automat.

Metodologie de rezolvare:

- 1).** Identificarea multimii semnalelor de intrare si alocarea lor pe intrarile MUX-ului; MUX-ul trebuie sa aiba suplimentar inca 2 intrari (una conectata la GND – pentru efectuarea salturilor neconditionate prin functia de incarcare paralela a numaratorului 74163, iar alta intrare conectata in 1-logic – pentru ca la efectuarea comenzilor, numaratorul 74163 sa efectueze functia de numarare);
- 2).** Identificarea multimii semnalelor de iesire si alocarea lor pe iesirile DMUX-ului;
- 3).** Identificarea multimii starilor prin care trece automatul (fiecarui bloc logic din organigrama ii este corespondenta o stare) si o prima codificare a acestora pe organigrama;
- 4).** Recodificarea starilor prin care trece automatul, cu respectarea urmatoarelor reguli:
 - identificarea traseelor din organigrama la parcurgerea carora numaratorul executa functia de numarare; toate blocurile logice din organigrama trebuie parcurse doar o singura data;
 - identificarea locurilor din organigrama unde se impune efectuarea de salturi neconditionate si introducerea de stari suplimentare pe respectivele pozitii;
 - renumerotarea starilor din organigrama pentru a tine seama de cele 2 reguli anterioare.
- 5).** Stabilirea continutului memoriei ROM.

FORMATUL MICROINSTRUCTIUNILOR

D7	D6	D5	D4	D3	D2	D1	D0
1	1	1	Com2	Com1	Com0	x	x

Microinstructiunea pentru executia comenzilor (tip M1)

D7	D6	D5	D4	D3	D2	D1	D0
Tst2	Tst1	Tst0	x	Adr3	Adr2	Adr1	Adr0

Microinstructiunea pentru testarea semnalelor de intrare (tip M2)

SOLUTIE

1). Sunt 5 semnale de intrare care apar in schema logica (*T1, T2, T3, T4, T5*) si care trebuie testate de automat. Rezulta necesitatea unui MUX cu 8 intrari, dintre care 5 vor fi folosite pentru semnalele mentionate anterior.

O intrare va fi conectata la GND – pentru efectuarea salturilor neconditionate prin functia de incarcare paralela a numaratorului 74163.

O alta intrare va fi conectata in 1-logic – pentru ca la efectuarea comenzilor, numaratorul 74163 sa efectueze functia de numarare.

O ultima intrare ramane neconectata (NC).

La MUX-ul 8:1 vom avea 3 semnale de selectie notate C, B si A. Alocam semnalele de intrare conform urmatorului tabel:

Intrare MUX 8:1	Semnal alocat	Selectie CBA
0	GND	000
1	T1	001
2	T2	010
3	T3	011
4	T4	100
5	T5	101
6	NC	110
7	1-logic	111

2). Sunt 5 semnale de iesire care apar in schema logica (*C0*, *C1*, *C2*, *C3*, *C4*) si care trebuie generate de automat. Folosim un DMUX 1:8 cu 3 semnale de selectie notate C, B, A. Alocam semnalele de iesire conform urmatorului tabel:

Iesire DMUX 1:8	Semnal alocat	Selectie CBA
0	C0	000
1	C1	001
2	C2	010
3	C3	011
4	C4	100
5	NC	101
6	NC	110
7	NC	111

3). In organigrama data sunt 11 stari. Numerotarea initiala a acestora este de la 0 pana la 10.

4). Identificam pe organigrama 4 trasee care pot fi parcurse prin numarare.

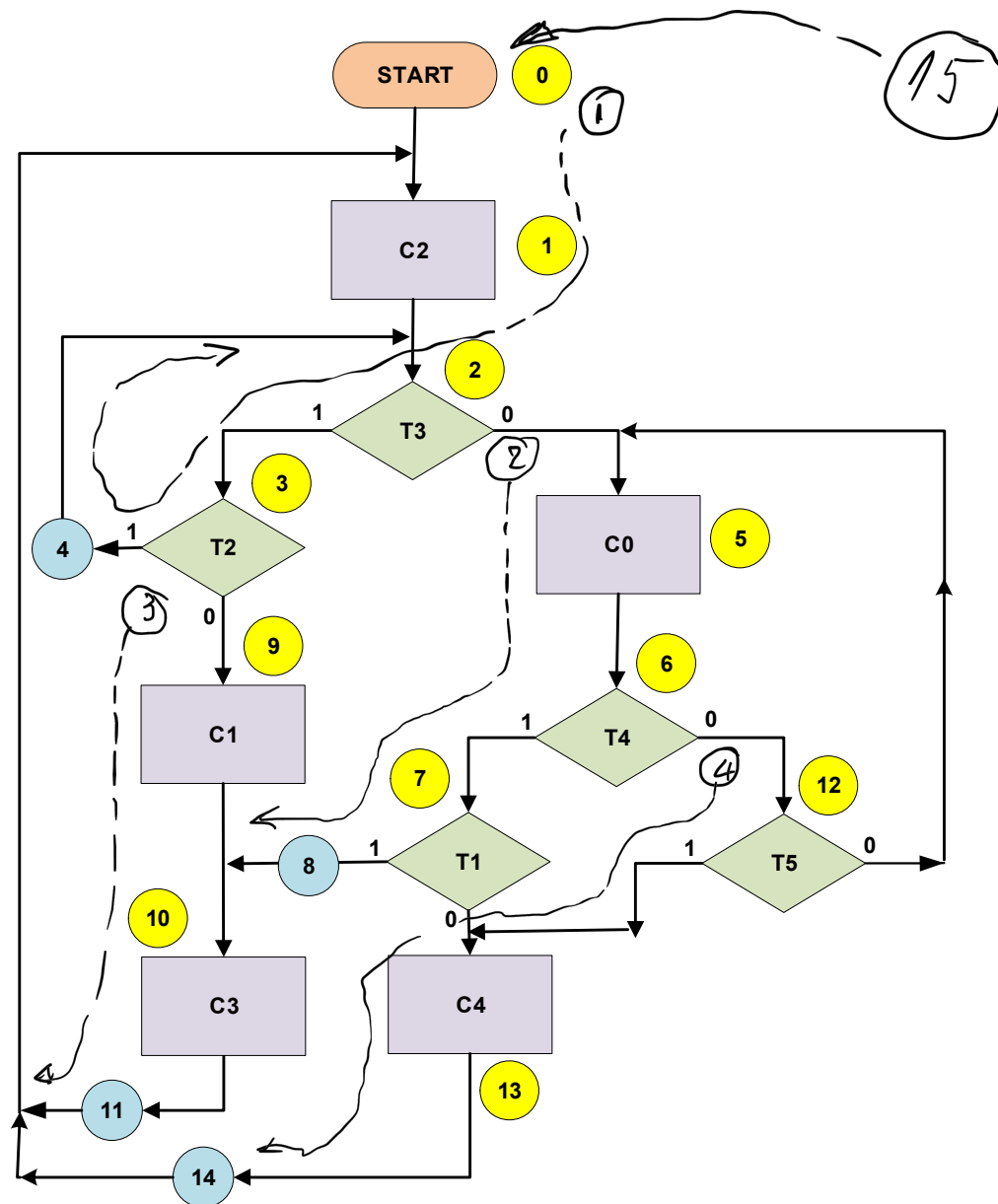
Identificam 4 pozitii pe organigrama unde se impune efectuarea de salturi neconditionate.

In aceste 4 pozitii introducem stari suplimentare.

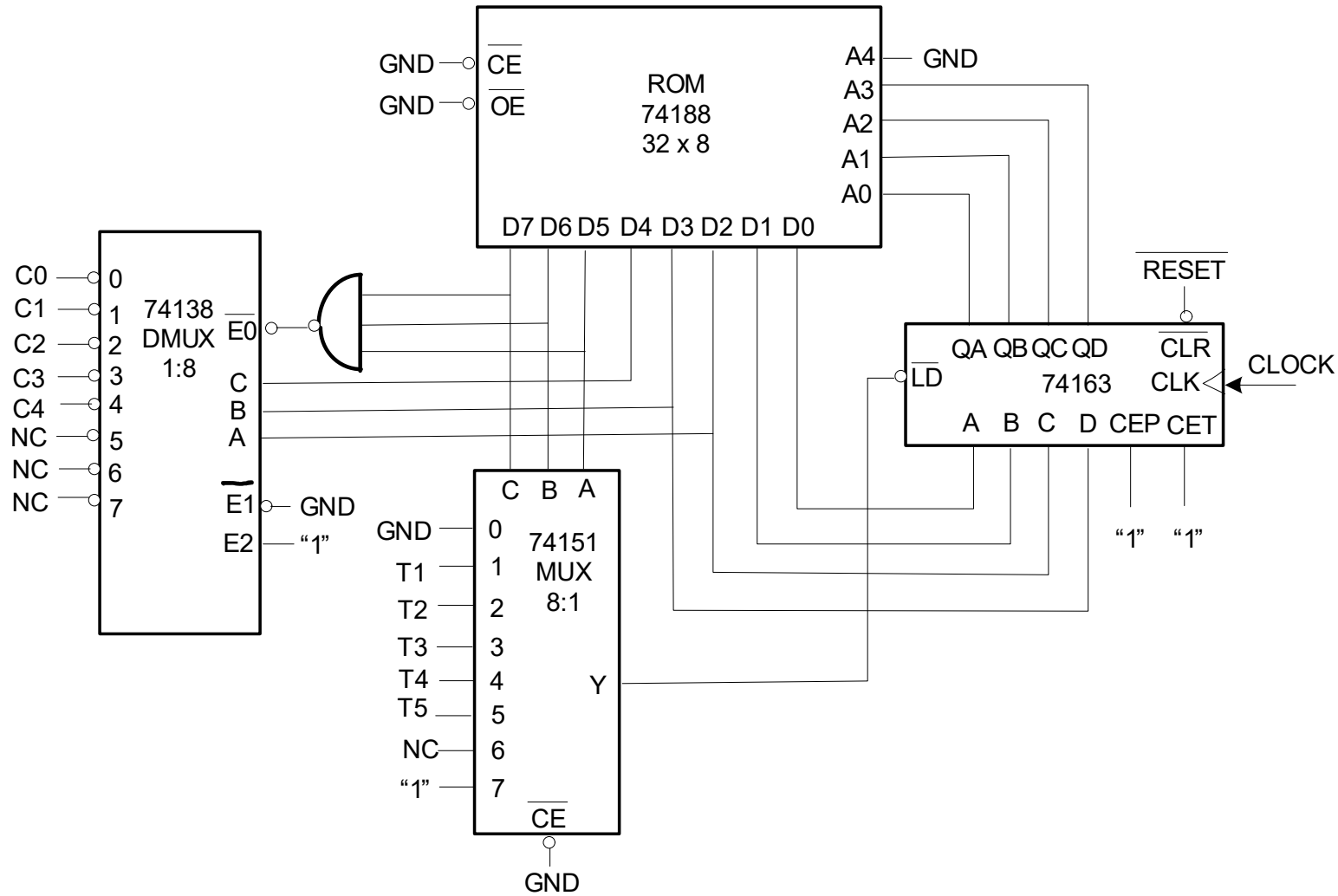
Renumerotam starile prin care trece automatul, tinand seama de mentiunile anterioare.

Apar in total 15 stari, pe care le renumerotam (de la 0 pana la 14).

Organigrama obtinuta prin introducerea de stari suplimentare si renumerotarea acestora



Structura concreta a unui
automat microprogramat
cu format variabil al
microinstruciunilor



Alocare liniilor de date din memoria ROM

Linie date ROM	D7	D6	D5	D4	D3	D2	D1	D0
Semnificatie	1	1	1	Com2	Com1	Com0	x	x
Semnal generat	1	1	1	C (DMUX)	B (DMUX)	A (DMUX)	x	x

Microinstruciunea pentru executia comenzilor (tip M1)

Linie date ROM	D7	D6	D5	D4	D3	D2	D1	D0
Semnificatie	Tst2	Tst1	Tst0	x	Adr3	Adr2	Adr1	Adr0
Semnal generat	C (MUX)	B (MUX)	A (MUX)	x	Adr3	Adr2	Adr1	Adr0

Microinstruciunea pentru testarea semnalelor de intrare (tip M2)

4). Stabilirea continutului memoriei ROM. Se intocmeste tabelul urmator pentru continutul memoriei ROM:

Stare	ADRESE					DATE (Microinstructiuni)								OBS
	A4	A3	A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0	
0	0	0	0	0	0	1	1	1	1	1	1	1	1	M1 (C7)
1	0	0	0	0	1	1	1	1	0	1	0	1	1	M1 (C2)
2	0	0	0	1	0	0	1	1	0	0	1	0	1	M2 (T3)
3	0	0	0	1	1	0	1	0	0	1	0	0	1	M2 (T2)
4	0	0	1	0	0	0	0	0	0	0	0	1	0	M2 (2)
5	0	0	1	0	1	1	1	1	0	0	0	1	1	M1 (C0)
6	0	0	1	1	0	1	0	0	0	1	1	0	0	M2 (T4)
7	0	0	1	1	1	0	0	1	0	1	1	0	1	M2 (T1)
8	0	1	0	0	0	0	0	0	0	1	0	1	0	M2 (10)
9	0	1	0	0	1	1	1	1	0	0	1	1	1	M1 (C1)
10	0	1	0	1	0	1	1	1	0	1	1	1	1	M1 (C3)
11	0	1	0	1	1	0	0	0	0	0	0	0	1	M2 (1)
12	0	1	1	0	0	1	0	1	0	0	1	0	1	M2 (T5)
13	0	1	1	0	1	1	1	1	1	0	0	1	1	M1 (C4)
14	0	1	1	1	0	0	0	0	0	0	0	0	1	M2 (1)
15	0	1	1	1	1	0	0	0	0	0	0	0	0	M2 (0)

1 - C7
2 - C2
3 - T3
4 - T2
5 - (2)
6 - C0
7 - T4
8 - T1
9 - (10)
10 - C1
11 - C3
12 - (1)
13 - T5
14 - (1)
15 - (0)

Concluzii

Automatul prezentat poate implementa organigrame (scheme logice) cu urmatoarele caracteristici:

- Numar de total de stari logice 16
- Numar de intrari care pot fi testate 6
- Numar de iesiri 8

Circuite de memorie semiconductoare

Exista doua tipuri mari de memorii: ROM (read only memory) si RAM (random access memory/ read-write memory).

Exista doua mari tehnologii prin care circuitele de memorie se pot realiza:

→ Tehnologia bipolară – foloseste tranzistoare npn, pnp

→ Tehnologia unipolară – foloseste tranzistoare nMOS, pMOS (CMOS)

Circuitele de memorie realizate in tehnologie bipolară se caracterizeaza prin urmatoarele:

→ Timp de raspuns / acces foarte scazut (sunt foarte rapide);

→ Densitate mica in integrare;

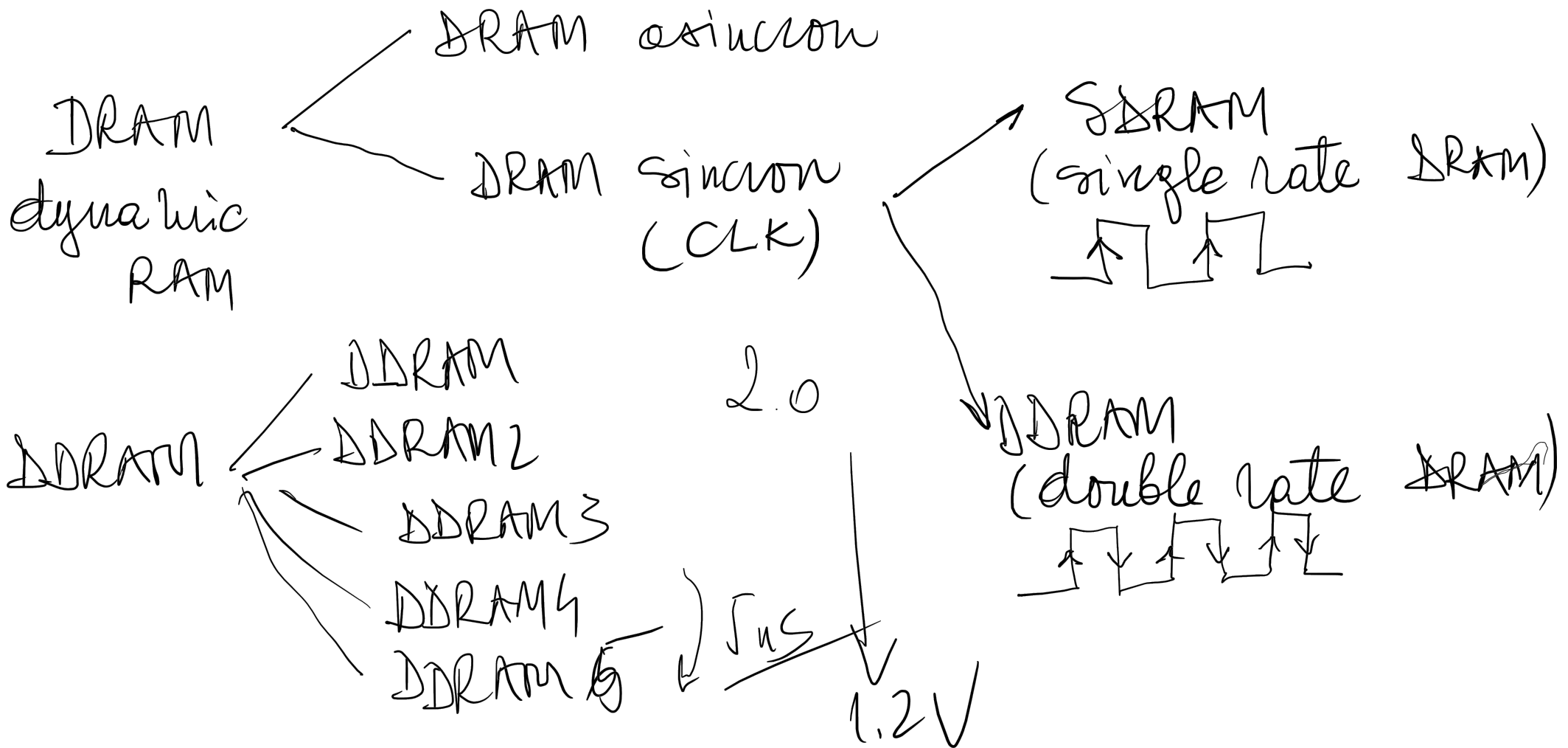
→ Consum energetic ridicat.

Circuitele de memorie unipolare prezinta urmatoarele caracteristici:

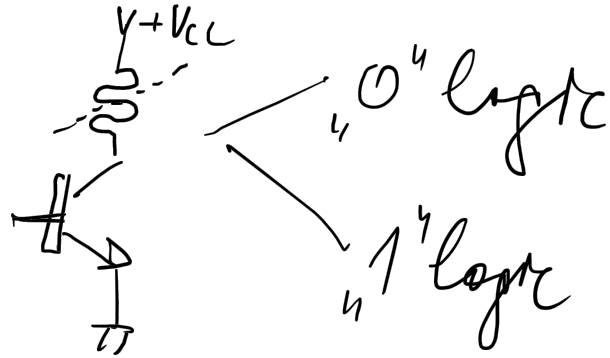
→ Densitate de integrare foarte mare, prezinta capacitate foarte mari;

→ Consum energetic redus;

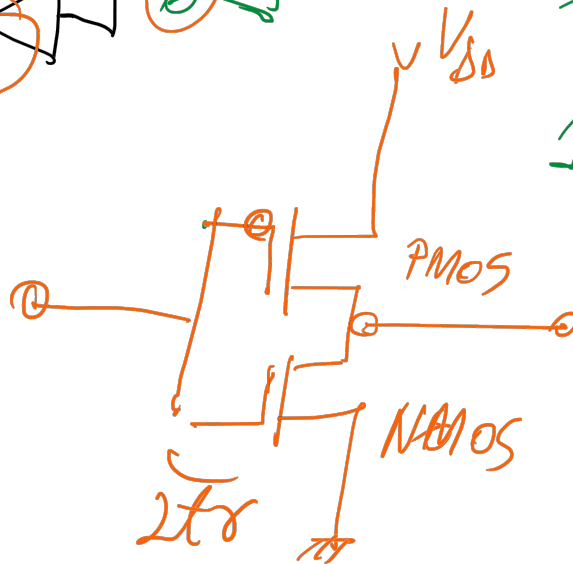
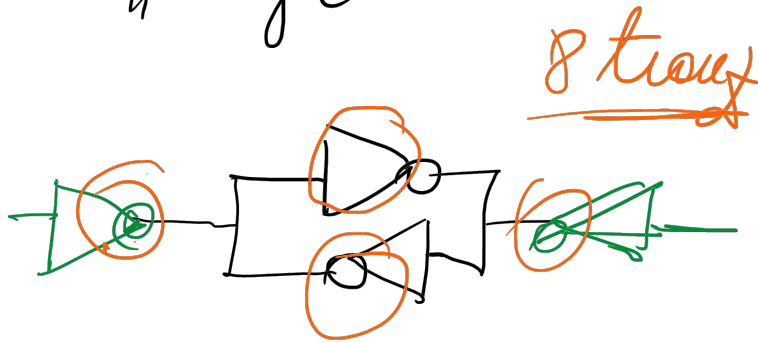
→ Timpul de acces este crescut (nu sunt foarte rapide).



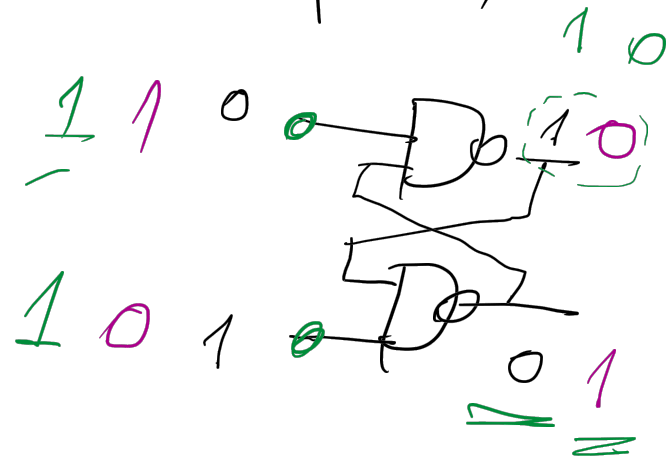
Cellular memory in ROM



SRAM (unipolar)



RAM { SRAM
(unipolar) DRAM



TIPURI DE MEMORII ROM UNIPOLARE

1. PROM (programmable ROM / OTP (one time programming)) - nu admit stergeri;
2. EPROM (erasable PROM) – admit stergere si se pot reprograma. Cipul prezinta o fereasta de sticla transparenta, stergerea se face prin expunere la raze ultraviolete. Numarul de cicli de stergere se apropie de 50;
3. EEPROM (electrically EPROM/ flash ROM) - memorii care se sterg electric. Numarul de stergeri urca peste 10000. Admit programare inCircuit.
4. Serial EEPROM adresa se da bit dupa bit. Numarul de stergeri urca la 1000000.

TIPURI DE MEMORII RAM

Memorii RAM statice – SRAM (Static RAM) – bipolare, unipolare

Memorii RAM dinamice – DRAM – unipolara

DRAM – 2 tipuri – asincrone, sincrone

**DRAM sincrone: SDRAM (single data rate DRAM - operatii pe un singur front al CLK)
 DDRAM (double data rate DRAM – operatii pe ambele fronturi ale CLK)**

DDRAM: DDRAM 2, 3, 4, 5, 6

END 27 mar 2025