**Ministerul Educației, Culturii și Cercetării**

**Universitatea Tehnică a Moldovei**

[**Departamentul Informatică şi Ingineria Sistemelor**](http://utm.md/subdiviziuni-universitare/facultati/facultatea-calculatoare-informatica-si-microelectronica/catedra-calculatoare/)

# Raport

**la proiectul de an la disciplina:**

**Proiectarea dispozitivelor numerice**

## Tema: Dispozitive numerice

**A elaborat :** stud. gr. CR-221 FR Serba Cristina

**Conducătorul de proiect:**  lect. univ., **Țurcan Ana**

**Admis la susținere \_\_\_\_\_\_\_\_\_\_\_ “\_\_\_\_\_\_\_\_\_\_\_”**

**Susținerea “\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_”**

**Nota “\_\_\_\_\_\_\_”**

**2024**

Cuprins

Contents

[Raport 1](#_Toc178340976)

[Tema: Dispozitive numerice 1](#_Toc178340977)

[Introducere 3](#_Toc178340978)

[Capitolul I 4](#_Toc178340979)

[1.1 Date generale despre regiștre și numărătoare 4](#_Toc178340980)

[1.2 Descrierea regiștrilor numărător Registrele 4](#_Toc178340981)

[1.3 Sinteza elementului multifuncțional 6](#_Toc178340982)

[1.3.1 Tabelul de tranziție 6](#_Toc178340983)

[1.3.2 Minimizarea funcțiilor 9](#_Toc178340984)

[Capitolul II Sinteza automatului de comandă 14](#_Toc178340985)

# Introducere

Dispozitivele numerice sunt o prezență cotidiană atît în structura calculatoarelor numerice, cît și în componența diferitor altor sisteme orientate spre procesarea rapidă a datelor.[1] Pentru a implementa astfel de dispozitive, specialistul trebuie să dispune de o mulțime de cunoștințe în domeniul algebrei booleane, funcționării dispozitivelor numerice și mecanismele care sunt utilizate în obținerea dispozitivelor numerice.

În primul capitol a memoriului curent este prezentat ciclul de implementare a elementului multifuncțional. În acesta este inclus tabelul de tranziții și mimizarea funcțiilor obținute, cît și schema logică a elementului elaborat.

În capitolul numărul doi, se vorbește de implementarea și sinteza unui automat de comandă. În exemplul dat, se creează un automat de comandă care ne va efectua împărțirea a două numere în virgulă mobilă. Sinteza se începe cu implementarea algoritmului, după care se construiește schema bloc marcată. Pentru lucrarea dată, a fost ales automatul de tip Moore, unde semnalul de comandă depinde doar de starea curentă a automatului, iar starea următoare va depinde atât de starea precedentă căt și de condițiile logice. După elaborarea schemei bloc marcate, realizăm graful de tranziție al automatului, pentru a vizualiza mai bine tranzițiile stărilor în dependență de condiții. La următoarea etapă am elaborat tabelul de tranziție al stărilor și am finalizat cu crearea schemei funcționale a automatului.

În ultimul capitol, este descrisă implementarea automatului operațional în baza schemei bloc a algoritmului și schemei bloc marcate. Aici vom vorbi despre componentele de care avem nevoie, și vom reprezenta doar schematic cum ar arăta un automat în realitate în baza unui exemplu.

# Capitolul I

**Sinteza registrului numărător invers modulo 23, cu deplasare la stânga**

## 1.1 Date generale despre regiștre și numărătoare

## 1.2 Descrierea regiștrilor numărător Registrele

Registrele se includ în categoria elementelor funcţionale secvenţiale şi sunt destinate memorării şi procesării cuvintelor binare. Componenta de bază a oricărui registru sunt bistabilele.

Structura generală a unui registru este prezentată în figura 3.5. şi este constituită din ***n*** bistabile (în cazul de faţă de tip *D*), avînd un semnal de ceas CLK comun pentru toate bistabilele (B). Intrarea

de ştergere *CLR*, activă pe zero logic, este prezentă la majoritatea registrelor şi permite resetarea celor ***n*** bistabile. Intrările *S1* şi *S0* comandă cele ***n*** comutatoare logice (CM), asigurînd astfel selecţia regimului de lucru al registrului. Comutatoarele, în dependenţă de codul de selecţie, por asigura conectarea intrărilor bistabilelor în trei moduri: la ieşirea Bi din stînga, din dreapta sau la intrarea de date *D*. În dependenţă de conectarea intrărilor bistabilelor, registrul poate încărca un cuvînt binar în cod paralel sau succesiv (Figura 1).

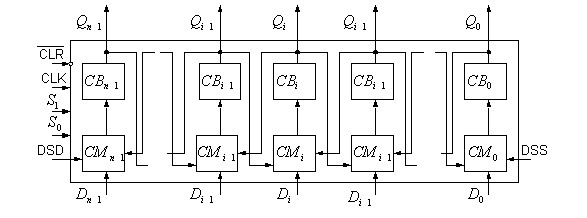


Figura 1. Structura generală a unui registru

**Număratoarele**

O clasă importantă de circuite logice secvenţiale o reprezintă numărătoarele (contoarele). Destinaţia numărătorului este înregistrarea numărului de impulsuri aplicate la intrarea lui şi divizarea frecvenţei acestor impulsuri. Componenta de bază a numărătoarelor sunt bistabilele. Un numărător are ***M***stări distincte. Tranziţia între stările succesive se produce în urma impulsurilor aplicate la intrarea numărătorului. După aplicarea unui număr de ***M*** impulsuri, numărătorul revine în starea iniţială (de exemplu în starea zero). Un astfel de circuit reprezintă un numărător modulo ***M***. Relaţia dintre modulo ***M*** şi numărul de bistabile ***n,*** ce intră în componenţa numărătorului este următoarea: ***n =[log2M].*** Dacă numărul ***n*** nu este întreg, atunci el se rotunjeşte pînă la cel mai apropiat număr întreg mai mare decît cel obţinut.

Numărătoarele se clasifică după două criterii: în dependenţă de ordinea numărării şi în dependenţă de faptul cum îşi schimbă starea bistabilele, Dacă numărătorul realizează o succesiune de stări codificate în ordine crescătoare, atunci el se numeştenumărător direct, iar dacă succesiunea stărilor este în ordine descrescătoare, atunci el se numeşte numărătorinvers. Numărătorul care realizează atît numărarea directă, cît şi cea inversă se numeşte reversibil. Numărătoarele se divizează în sincrone sau asincrone în dependenţă de faptul cum îşi schimbă starea bistabilele ce intră în componenţa lor. Dacă bistabilele îşi schimbă starea concomitent, atunci numărătoarele se

numesc sincrone, iar dacă starea lor se schimbă succesiv numărătoarele se numesc asincrone.

Pentru un numărător modulo 23 vor fi necesare cinci bistabile JK.

La sinteza unui numărător modulo ***M***, din cele **2n** stări posibile un număr de **2n-*M*** sunt stări

neutilizate (ilegale) şi se consideră ca nedeterminate în diagrama Veitch -Karnaugh. Se poate întîmpla, însă, ca numărătorul la pornire sau sub influenţa unor semnale parazitare să nimerească în una din aceste stări. Dacă după cîteva tacturi numărătorul poate ajunge într-o stare legală, atunci funcţionarea lui de mai departe este corectă. Însă se poate întîmpla ca numărătorul să nu poată ieşi din stările ilegale, decît numai printr-o nouă pornire. Pentru a evita aceste cazuri, sinteza numărătorului trebuie efectuată în aşa fel ca în tabelul de tranziţie să se facă tranziţia spre starea iniţială din oricare din stările ilegale la următorul impuls de numărare.

Mai jos este prezentată sinteza unui numărător direct, sincron modulo 23 în baza bistabilelor *JK*. Numărul de bistabile necesar realizării acestui numărător este *n=[log223]=5*. În tabelul de tranziţie aceste bistabile sunt notate respectiv: *Q1, Q2, Q3, Q4, Q5.*

## 1.3 Sinteza elementului multifuncțional

### 1.3.1 Tabelul de tranziție

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | t | | | | | t+1 | | | | | JK | | | | |
| **Q1** | **Q2** | **Q3** | **Q4** | **Q5** | **Q1** | **Q2** | **Q3** | **Q4** | **Q5** | **J1K1** | **J2K2** | **J3K3** | **J4K4** | **J5K5** |
| **0** | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 \* | 0\* | 1\* | 1\* | 0\* |
| **1** | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0\* | 0\* | 0\* | 0\* | \*1 |
| **2** | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0\* | 0\* | 0\* | \*1 | 1\* |
| **3** | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0\* | 0\* | 0\* | \*0 | \*1 |
| **4** | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0\* | 0\* | \*1 | 1\* | 1\* |
| **5** | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0\* | 0\* | \*0 | 0\* | \*1 |
| **6** | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0\* | 0\* | \*0 | \*1 | 1\* |
| **7** | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0\* | 0\* | \*0 | \*0 | \*1 |
| **8** | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0\* | \*1 | 1\* | 1\* | 1\* |
| **9** | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0\* | \*0 | 0\* | 0\* | \*1 |
| **10** | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0\* | \*0 | 0\* | \*1 | 1\* |
| **11** | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0\* | \*0 | 0\* | \*0 | \*1 |
| **12** | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0\* | \*0 | \*1 | 1\* | 1\* |
| **13** | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0\* | \*0 | \*0 | 0\* | \*1 |
| **14** | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0\* | \*0 | \*0 | \*1 | 1\* |
| **15** | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0\* | \*0 | \*0 | \*0 | \*1 |
| **16** | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | \*1 | 1\* | 1\* | 1\* | 1\* |
| **17** | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | \*0 | 0\* | 0\* | 0\* | \*1 |
| **18** | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | \*0 | 0\* | 0\* | \*1 | 1\* |
| **19** | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | \*0 | 0\* | 0\* | \*0 | \*1 |
| **20** | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | \*0 | 0\* | \*1 | 1\* | 1\* |
| **21** | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | \*0 | 0\* | \*0 | 0\* | \*1 |
| **22** | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | \*0 | 0\* | \*0 | \*1 | 1\* |
| **23** | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | \*1 | 0\* | \*1 | \*1 | \*1 |
| **24** | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | \*1 | \*1 | 0\* | 0\* | 0\* |
| **25** | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | \*1 | \*1 | 0\* | 0\* | \*1 |
| **26** | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | \*1 | \*1 | 0\* | \*1 | 0\* |
| **27** | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | \*1 | \*1 | 0\* | \*1 | \*1 |
| **28** | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | \*1 | \*1 | \*1 | 0\* | 0\* |
| **29** | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | \*1 | \*1 | \*1 | 0\* | \*1 |
| **30** | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | \*1 | \*1 | \*1 | \*1 | 0\* |
| **31** | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | \*1 | \*1 | \*1 | \*1 | \*1 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | t | | | | | t+1 | | | | | JK | | | | |
| **Q1** | **Q2** | **Q3** | **Q4** | **Q5** | **Q1** | **Q2** | **Q3** | **Q4** | **Q5** | **J1K1** | **J2K2** | **J3K3** | **J4K4** | **J5K5** |
| **32** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0\* | 0\* | 0\* | 0\* | 0\* |
| **33** | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0\* | 0\* | 0\* | 1\* | \*1 |
| **34** | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0\* | 0\* | 1\* | \*1 | 0\* |
| **35** | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0\* | 0\* | 1\* | \*0 | \*1 |
| **36** | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0\* | 1\* | \*1 | 0\* | 0\* |
| **37** | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0\* | 1\* | \*1 | 1\* | \*1 |
| **38** | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0\* | 1\* | \*0 | \*1 | 0\* |
| **39** | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0\* | 1\* | \*0 | \*0 | \*1 |
| **40** | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1\* | \*1 | 0\* | 0\* | 0\* |
| **41** | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1\* | \*1 | 0\* | 1\* | \*1 |
| **42** | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1\* | \*1 | 1\* | \*1 | 0\* |
| **43** | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1\* | \*1 | 1\* | \*0 | \*1 |
| **44** | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1\* | \*0 | \*1 | 0\* | 0\* |
| **45** | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1\* | \*0 | \*1 | 1\* | \*1 |
| **46** | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1\* | \*0 | \*0 | \*1 | 0\* |
| **47** | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1\* | \*0 | \*0 | \*0 | \*1 |
| **48** | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | \*1 | 0\* | 0\* | 0\* | 1\* |
| **49** | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | \*1 | 0\* | 0\* | 1\* | \*0 |
| **50** | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | \*1 | 0\* | 1\* | \*1 | 1\* |
| **51** | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | \*1 | 0\* | 1\* | \*0 | \*0 |
| **52** | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | \*1 | 1\* | \*1 | 0\* | 1\* |
| **53** | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | \*1 | 1\* | \*1 | 1\* | \*0 |
| **54** | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | \*1 | 1\* | \*0 | \*1 | 1\* |
| **55** | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | \*1 | 1\* | \*0 | \*0 | \*0 |
| **56** | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | \*0 | \*1 | 0\* | 0\* | 1\* |
| **57** | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | \*0 | \*1 | 0\* | 1\* | \*0 |
| **58** | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | \*0 | \*1 | 1\* | \*1 | 1\* |
| **59** | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | \*0 | \*1 | 1\* | \*0 | \*0 |
| **60** | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | \*0 | \*0 | \*1 | 0\* | 1\* |
| **61** | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | \*0 | \*0 | \*1 | 1\* | \*0 |
| **62** | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | \*0 | \*0 | \*0 | \*1 | 1\* |
| **63** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | \*0 | \*0 | \*0 | \*0 | \*0 |

Tabelul de tranziție a bistabilului JK:

|  |  |  |  |
| --- | --- | --- | --- |
| **(Q)** | **(Q+1)** | **J** | **K** |
| 0 | 0 | 0 | X |
| 0 | 1 | 1 | X |
| 1 | 0 | X | 1 |
| 1 | 1 | X | 0 |

### 1.3.2 Minimizarea funcțiilor

Diagramele Veitch-Karnaugh pentru minimizarea funcţiilor de instalare a bistabilelor:



















|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | K5 | |  |  |  |  |  |  | |  | | --- | |  | |  |
|  | 0 0 0 | 0 0 1 | 0 1 1 | 0 1 0 | 1 1 0 | 1 1 1 | 1 0 1 | 1 0 0 |
| 0 0 0 | \* | \* | \* | \* | \* | \* | \* | \* |
| 0 0 1 | 1 | 1 | 1 | 1 |  |  | 1 | 1 |
| 0 1 1 | 1 | 1 | 1 | 1 |  |  | 1 | 1 |
| 0 1 0 | \* | \* | \* | \* | \* | \* | \* | \* |
| 1 1 0 | \* | \* | \* | \* | \* | \* | \* | \* |
| 1 1 1 | 1 | 1 | 1 | 1 |  |  | 1 | 1 |
| 1 0 1 | 1 | 1 | 1 | 1 |  |  | 1 | 1 |
| 1 0 0 | \* | \* | \* | \* | \* | \* | \* | \* |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

### 1.3.3 Schema registrului numărător - Anexa 1

Schema registrului numărător invers, cu deplasare la stânga, modulo 23 obținut printr-un set debistabile JK și elemente logice ȘI-NU.

# Capitolul II Sinteza automatului de comandă

## 2.1 Date generale despre automate

Orice unitate a unui calculator numeric are două componente distincte,care la mod generalizat se împart în automatul de comandă şi automatul operaţional (în continuare A.O.). A.O. este acea componentă a unităţii de calcul care este destinată îndeplinirii tuturor operaţiilor logice de transfer,de memorare ş.a. din calculator,iar auotmatul de comandă (în continuare A.C.) este acea parte componentă a unităţii de calcul care stabileşte pentru A.O. în ce ordine şi-n care operaţii trebuie îndeplinite pentru a atinge scopul scondat.

În componenţa unităţilor de comandă intră două componente de bază:

1) un circuit logic combinaţional care asigură examinarea informaţiei în baza căreia se emit aşa numitele semnale de comandă pentru a îndeplini operaţiile necesare;

2) un grup de bistabile care determină starea automatului de comandă şi în baza acestei stări se stabilleţâşte în fiecare moment de timp care semnale de comandă trebuie transmise automatului operaţional şi care semnale de comandă sunt necesare pentru a asigura tranziţia A.C. dintr-o stare în alta.

Pentru îndeplinirea fiecărei operaţii este necesară elaborarea algoritmului care stabileşte succesiunea de microoperaţii care trebuie îndeplinite de A.O. Această succesiune de microoperaţii formează microprogramul de realizare a microoperaţiei respective, microprogramul se împarte în microinstrucţiuni.O microinsrucţiune include cîteva microoperaţii care se îndeplinesc în una din stările A.C. (într-un tact de calcul). Pentru îndeplinirea fiecărei microoperaţii A.C. trebuie să genereze un semnal de comandă, iar A.O. în baza acestui semnal de comandă trebuie să îndeplinească microoperaţia respectivă.

## 2.2 Descrierea algoritmului pentru împărțirea în virgulă mobilă

Primul aspect pe care trebuie să înțelegem este cum putem să reprezentăm numerele cu virgulă în formă binară. Ca și în cazul numerelor întregi, valorile fracționare se reprezintă pe un anumit număr de biți.

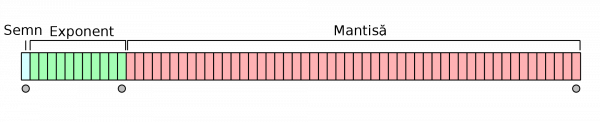
Principalele obiective ale unei reprezentări pe un anumit număr de biți sunt:

* posibilitatea de a reprezenta cât mai multe valori intre valoarea minimă și valoarea maximă
* o precizie cât mai bună a valorilor (numărul maxim de cifre după virgulă)

Pentru a reprezenta valorile fracționare vom folosi **Reprezentarea în virgulă mobilă** (**Floating Point Representation**). În această reprezentare, numerele au următoarea structură:

[http://elf.cs.pub.ro/asm/wiki/_media/laboratoare/lab12-equation1.png?w=500&tok=a07ca1](http://elf.cs.pub.ro/asm/wiki/_detail/laboratoare/lab12-equation1.png?id=laboratoare%3Alaborator-12)

După cum putem observa mai sus, valorile trebuie transformate astfel încât partea întreagă să fie 1. Această formă poartă numele de formă normală, iar operația de transformare în această formă poartă numele de **normalizare**.

În forma binară, valorile se reprezintă astfel:[](http://elf.cs.pub.ro/asm/wiki/_detail/laboratoare/2000px-ieee_754_double_floating_point_format.png?id=laboratoare%3Alaborator-12)

**Semnul** este dat de primul bit din reprezentarea binară: 1 = negativ, 0 = pozitiv.  
**Mantisa** dă partea fracționară a numărului în forma normală. Numărul de biți pe care mantisa este reprezentată dă precizia maximă a reprezentării.  
**Baza** este de obicei 2, 10 sau 16 și este dată de standardul de reprezentare ales.  
**Exponentul** dă valoarea la care este ridicată baza și numărul de biți pe care este reprezentat dă valorile valorile maxime și minime ce pot fi reprezentate.

Împărţirea a două numere în virgulă mobilă normalizate va produce întotdeauna un cât normalizat. Din acest motiv, spre deosebire de celelalte operaţii în virgulă mobilă, rezultatul operaţiei de împărţire nu necesită normalizare.

### 2.2.1 Exemplu de împărțire în virgulă mobilă

X = -4,562 Y = 26,85

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Sg | 16 | 8 | 4 | 2 | 1 |  |  |  |
| -4 | 1. |  |  | **1** | **0** | **0** |  |  |  |
| 26 | 0. | **1** | **1** | **0** | **1** | **0** |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| 0,562: | 0.562\*2 | 1.124 | **1** |  | 0,85: | 0.85\*2 | 1.7 | **1** |  |
|  | 0.124\*2 | 0.248 | **0** |  |  | 0.7\*2 | 1.4 | **1** |  |
|  | 0.248\*2 | 0.496 | **0** |  |  | 0.4\*2 | 0.8 | **0** |  |
|  | 0.496\*2 | 0.992 | **0** |  |  | 0.8\*2 | 1.6 | **1** |  |
|  | 0.992\*2 | 1.984 | **1** |  |  | 0.6\*2 | 1.2 | **1** |  |
|  | 0.984\*2 | 1.968 | **1** |  |  | 0.2\*2 | 0.4 | **0** |  |
|  | 0,968\*2 | 1.936 | **1** |  |  | 0.4\*2 | 0.8 | **0** |  |
|  | 0,936\*2 | 1.872 | **1** |  |  | 0.8\*2 | 1.6 | **1** |  |
|  | 0.872\*2 | 1.744 | **1** |  |  | 0.6\*2 | 1.2 | **1** |  |
|  | 0.744\*2 | 1.488 | **1** |  |  | 0.2\*2 | 0.4 | **0** |  |
|  | 0.488\*2 | 0.976 | **0** |  |  | 0.4\*2 | 0.8 | **0** |  |
|  | 0.976\*2 | 1.952 | **1** |  |  |  |  |  |  |

X = 1.100,100011111101

Y = 0.11010,1101100110

Mx =1.100100011111101 Ex = 0.00000011

My = 0.110101101100110 Ey = 0.00000101

#### Algoritmul nr. 2 de împărțire

1. Verificăm la 0 pe My

My ≠ 0

1. SgMx + SgMy = SqMz

1+0 = 1

1. |Mx| = 0.011011100000011

|My| = 0.110101101100110

-|My| = 1.001010010011010

1. Ex – Ey = Ez

0.00000011+

1.11111011

---------------

1.111111110

1. Mx / My

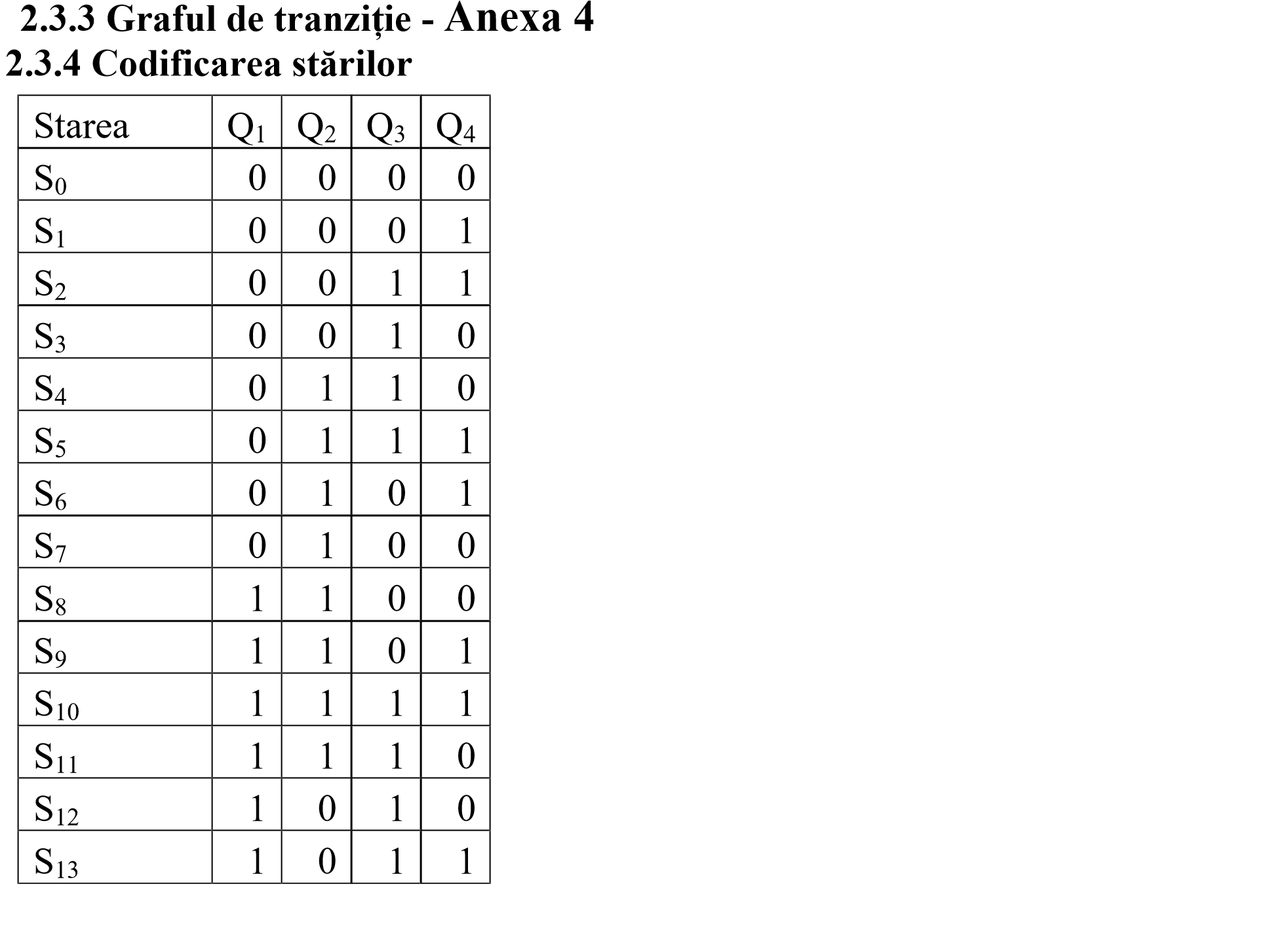
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0  1 | 0  0 | 1  0 | 1  1 | 0  0 | 1  1 | 1  0 | 1  0 | 0  1 | 0  0 | 0  0 | 0  1 | 0  1 | 0  0 | 1  1 | 0  1 | |Mx|  -|My| |
| 0 | 1  0  0 | 1  1  0 | 0  1  1 | 0  0  1 | 1  1  0 | 0  0  1 | 1  1  1 | 1  1  1 | 1  0  0 | 0  1  0 | 0  1  0 | 1  0  0 | 1  0  0 | 1  1  0 | 0  1  1 | 1  0  1 | R < 0  |My|  Restabilim |
| 0  1 | 1  0 | 1  0 | 0  1 | 1  0 | 1  1 | 1  0 | 0  0 | 0  1 | 0  0 | 0  0 | 0  1 | 0  1 | 1  0 | 1  1 | 0  0 | Deplasam  -|My| |
| 1 | 0  0  1 | 0  0  0 | 0  0  0 | 0  0  1 | 0  0  0 | 0  1  1 | 1  0  0 | 0  1  0 | 1  0  1 | 0  0  0 | 0  1  0 | 1  1  1 | 1  1  1 | 1  0  0 | 0  0  1 | 0  0  1 | R > 0  Deplasam  -|My| |
| 0 | 1  0  0 | 0  1  0 | 0  1  0 | 1  0  0 | 1  1  0 | 0  0  1 | 0  1  0 | 1  1  0 | 1  0  0 | 1  1  0 | 0  1  1 | 1  0  1 | 0  0  1 | 0  1  0 | 1  1  0 | 1  0  1 | R < 0  |My|  Restabilim |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | Deplasam |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Rezultat : -|mz|=1.00100000110110 mz= 1.11011111001010= 14282\*26

#### 2.3 Etapele de sinteză a algoritmului

**2.3.1 Schema bloc a algoritmului - Anexa 2**

**2.3.2 Schema bloc marcată - Anexa 3**

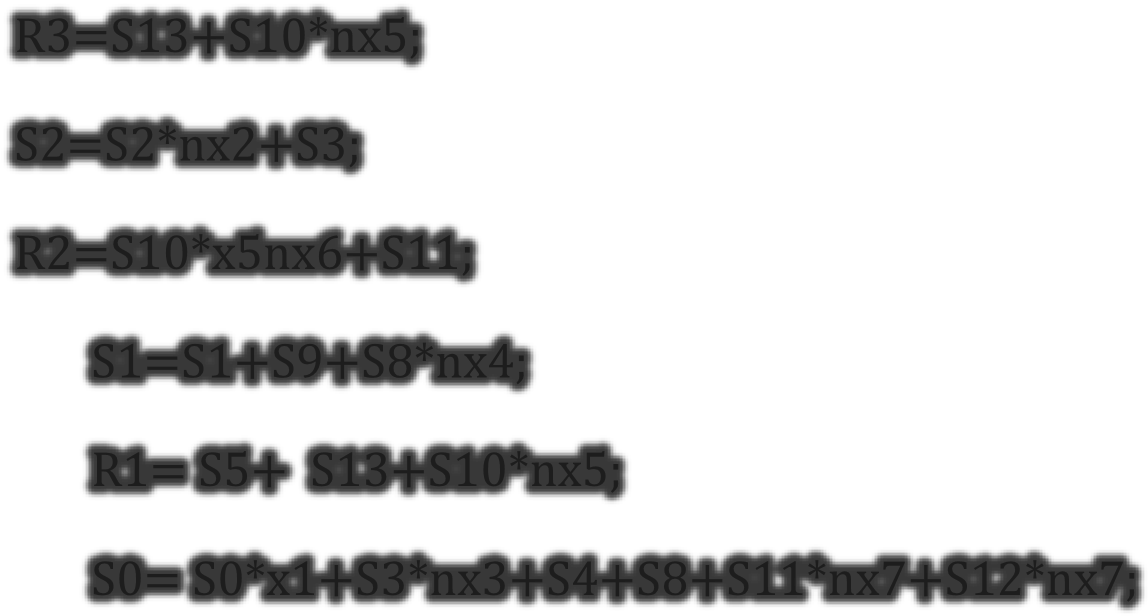


##### **2.3.5 Tabela de tranziție**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **starea, st** |  | **codul st** | | | **starea**  **st+1** | **codul st+1** | | |  | **conditia** | **st de**  **comanda** | **semnal de stare** | | | |
|  | **Q**  **3** | **Q**  **2** | **Q**  **1** | **Q0** |  | **Q3** | **Q2** | **Q**  **1** | **Q**  **0** | **Xi** | **Yi** | **Q**  **3** | **Q2** | **Q1** | **Q0** |
| S0 | 0 | 0 | 0 | 0 | S0 | 0 | 0 | 0 | 0 | nx1 | - | - | - | - | - |
| S0 | 0 | 0 | 0 | 0 | S1 | 0 | 0 | 0 | 1 | x1 | Y1,Y2,Y3,Y 4,Y5,y6,y7 | - | - | - | S0 |
| S1 | 0 | 0 | 0 | 1 | S2 | 0 | 0 | 1 | 1 | 1 | y8 | - | - | S1 | - |
| S2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| S2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| S2 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| S3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| S3 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| S4 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| S5 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| S6 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| S7 | 0 | 1 | 0 | 0 | S8 | 1 | 1 | 0 | 0 | 1 | y14 | S3 | - | - | - |
| S8 | 1 | 1 | 0 | 0 | S10 | 1 | 1 | 1 | 1 | nx4 | Y16,Y17 | - | - | S1 | S0 |
| S8 | 1 | 1 | 0 | 0 | S9 | 1 | 1 | 0 | 1 | x4 | y15 | - | - | - | S0 |
| S9 | 1 | 1 | 0 | 1 | S10 | 1 | 1 | 1 | 1 | 1 | Y16,Y17 | - | - | S1 | - |
| S10 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| S10 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| S10 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| S10 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| S11 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| S11 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| S12 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| S13 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Funcțiile pentru automatul de comandă:

S3=S7;



R0=S2\*x2+S2\*nx2x3+S6+S10\*x5x6+S10\*x5nx6x7+S13;

**2.3.6 Expresiile logice și schema automatului de comandă - Anexa 5 Capitolul III**

**Sinteza automatului operational**

**3.1 Date generale**

**3.2 Alegerea componentelor**

Orice unitate a unui calculator numeric are două componente distincte,care la mod generalizat se împart în automatul de comandă şi automatul operaţional (în continuare A.O.).

A.O. este acea componentă a unităţii de calcul care este destinată îndeplinirii tuturor operaţiilor logice de transfer,de memorare ş.a. din calculator,iar auotmatul de comandă (în continuare A.C.) este acea parte componentă a unităţii de calcul care stabileşte pentru A.O. în ce ordine şi-n care operaţii trebuie îndeplinite pentru a atinge scopul scondat.

În componenţa A.O. intră: elementele functionale care asigură memoria datelor,adică registrele,numărătoarele; elementele funcţionale ce asigură procesarea datelor, adică sumatorile,circuite cu operaţii logice.

În componenţa unităţilor de comandă intră două componente de bază:Conținutul a fost șters intenționat

1. un circuit logic combinaţional care asigură examinarea informaţiei în baza căreia se emit aşa numitele semnale de comandă pentru a îndeplini operaţiile necesare;
2. un grup de bistabile care determină starea automatului de comandă şi în baza acestei stări se stabilleţâşte în fiecare moment de timp care semnale de comandă trebuie transmise automatului operaţional şi care semnale de comandă sunt necesare pentru a asigura tranziţia A.C. dintr-o stare în alta.

Pentru îndeplinirea fiecărei operaţii este necesară elaborarea algoritmului care stabileşte succesiunea de microoperaţii care trebuie îndeplinite de A.O. Această succesiune de microoperaţii formează microprogramul de realizare a microoperaţiei respective, microprogramul se împarte în microinstrucţiuni.O microinsrucţiune include cîteva microoperaţii care se îndeplinesc în una din stările A.C. (într-un tact de calcul). Pentru îndeplinirea fiecărei microoperaţii A.C. trebuie să genereze un semnal de comandă, iar A.O. în baza acestui semnal de comandă trebuie să îndeplinească microoperaţia respectivă.

Pentru a efectua sinteza A.O. la început se stabileşte cîte variabile iniţiale se vor prelucra în acest automat şi pentru fiecare dintre ele se rezervează câte un registru în care ele vor fi păstrate. Dacă pentru a îndeplini operaţia respectivă este necesar un proces ciclic atunci în componenţa A.O. trebuie inclus minimum un numărător cu ajutorul căreia se va stabili numărul de iteraţii necesare. Dacă pentru a realiza operaţia sunt necesare operaţii aritmetice, în componenţa A.O. trebuie inclus un sumator. Dacă pentru a realiza operaţii sunt necesare microoperaţii de decodificare sau codificare a unor cuvinte în componenţa A.O. trebuie inclus un

codificator sau un decodificator. În cazul cînd pentru a realiza algoritmi sunt necesare operaţii de comparare atunci în componenţa A.O. se includ operatoare sau în unele cazuri se includ operatoare, sau în unele cazuri se includ circuite de îndeplinire a operaţiilor logice.

Conținutul a fost șters intenționat

În cazul cînd către una şi aceeaşi componentă a A.O. datele se transmit din mai multe surse atunci în structura A.O. trebuie incluse multiplexoarele care realizează transmiterea datelor din sursa necesară în fiecare tact de calculator. Î afară de registre în care se păstrează variabilele iniţiale, operanzii în componenţa A.O. mai trebuie inclus unul sau mai multe registre pentru păstrarea datelor intermediare. Toate cele enumerate mai sus formează etapa de stabilire a componentelor A.O.

A doua etapă este stabilirea legăturilor între componentele A.O. Legăturile se stabilesc astfel: dacă este necesară îndeplinirea operaţiilor de adunare, de înmulţire şi puteri a conţinutelor unor registre, atunci ieşirea acestor registre se comuteză la intrările sumatorului în aşa mod ca ele să poată fi recepţionate.

Ieşirea sumatorului trebuie să fie conectată la intrările acelor componente a A.O. cu care trebuie să fie transmis rezultatul obţinut. În multe cazuri sumatorul şi un registru sunt unul faţă de altul şi sursă şi destinator deaceea sunt necesare aceste conexiuni.

Conexiunile necesare în A.O. se efectuează în paralel cu elaborarea algoritmului de funcţionare a acestui automat.

**3.3 Schema de structură - Anexa 6**

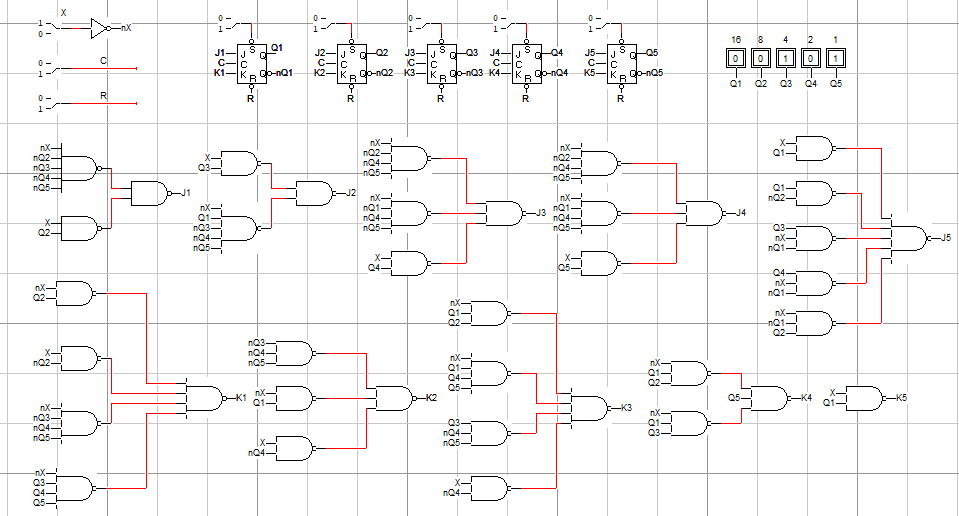
**Concluzie:**

În cadrul proiectului dat am studiat registrele, numărătorele, automatele de comanda și operaționale. Datorită faptului că am făcut de la 0 tot necesarului(tablelele de tranziție, grafurile, schemele bloc) am putut realiza cum arată întreg procesu, care este rolul registrilor, bistabilelor, a tuturor elementelor in circuitele finale.

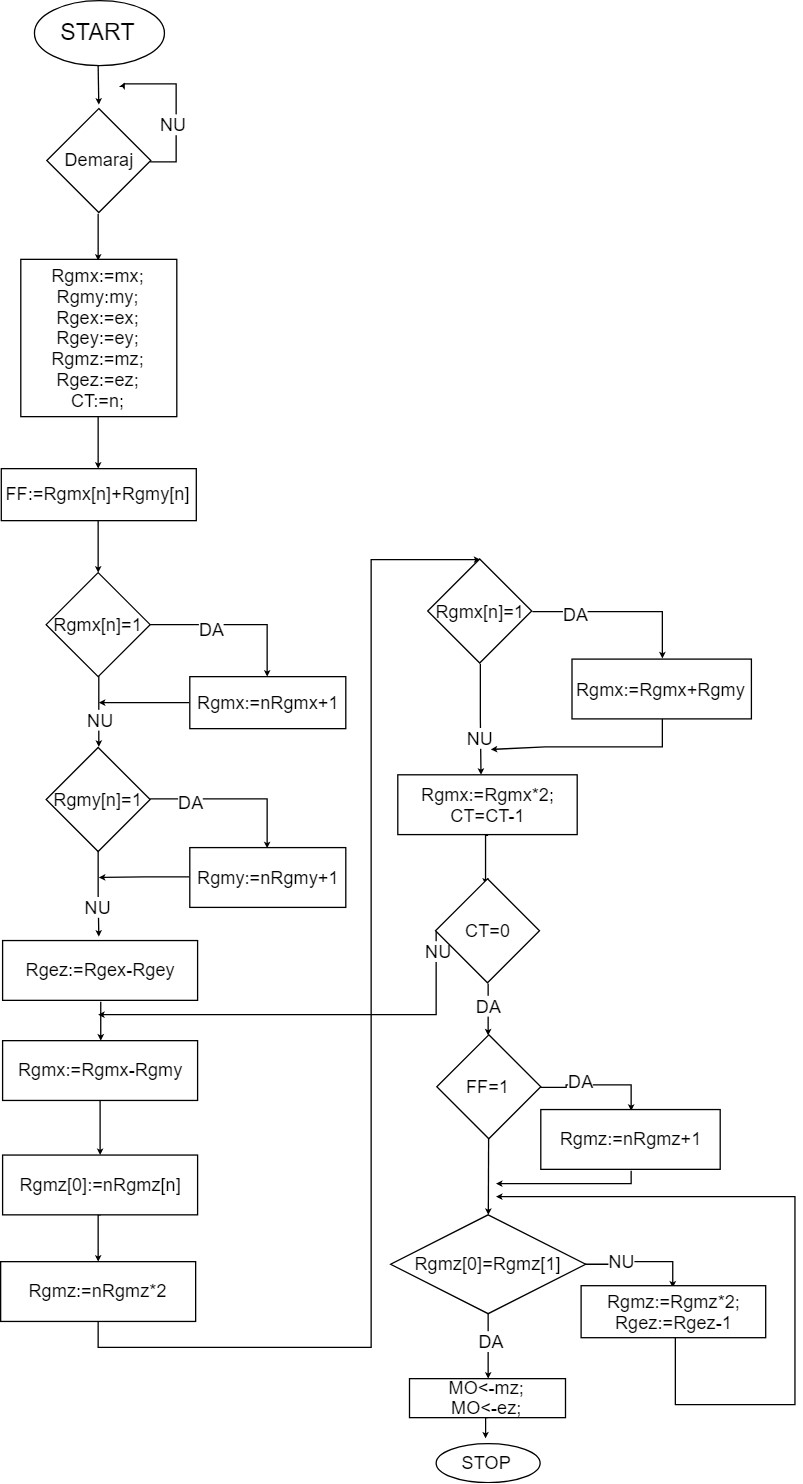
**Bibliografie**

<https://regielive.net/cursuri/calculatoare/automate-de-comanda-si-automate-operationale-114860.html>

**Anexa 1**



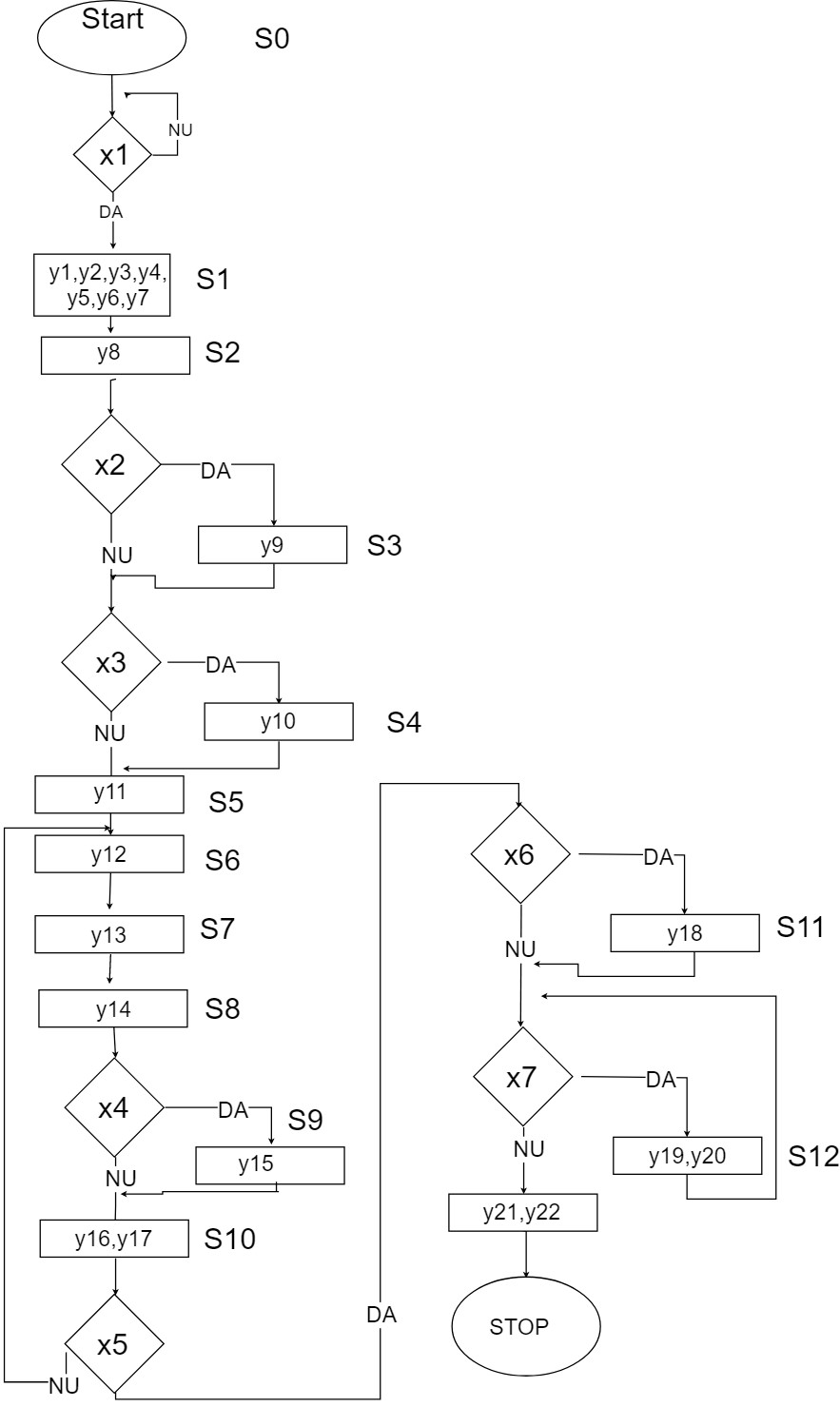
**Anexa 2**



C

t

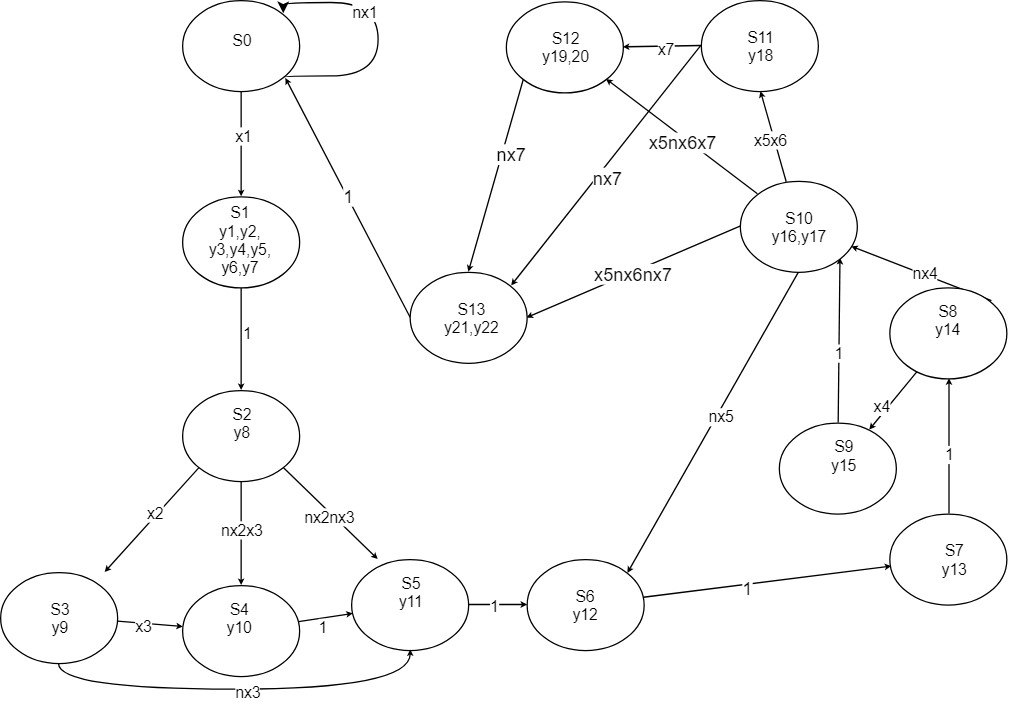
**Anexa 3**



C

t

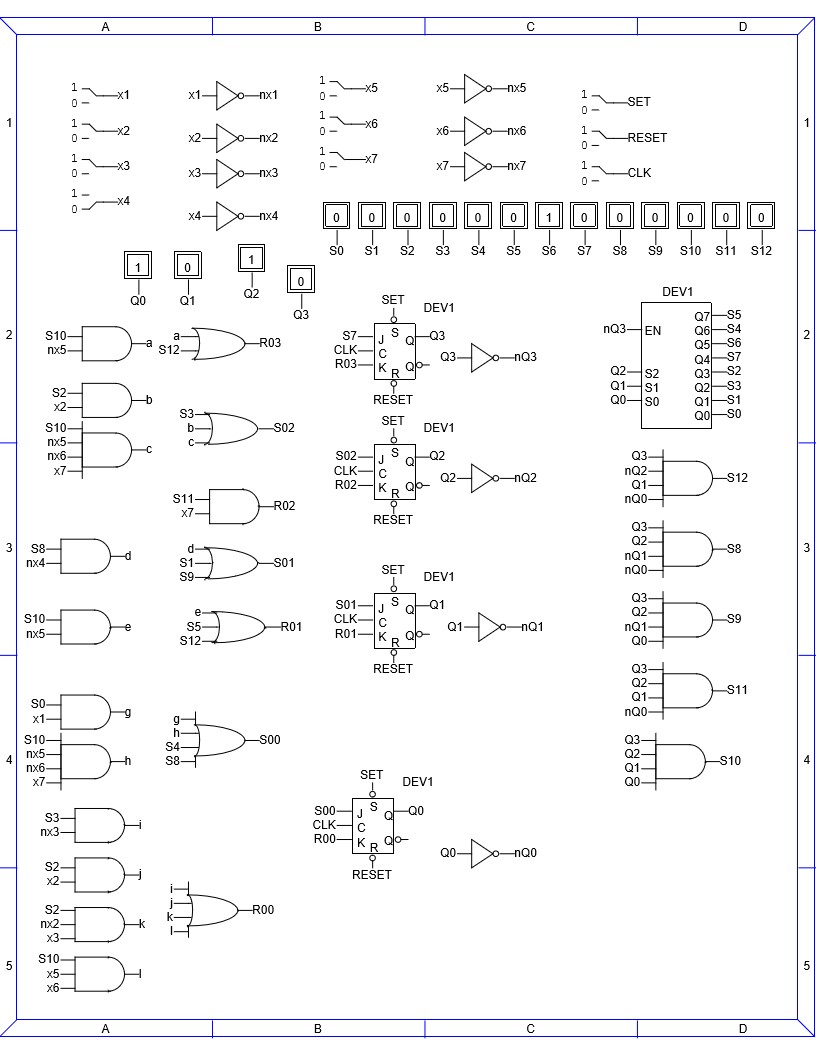
**Anexa 4**



C

t

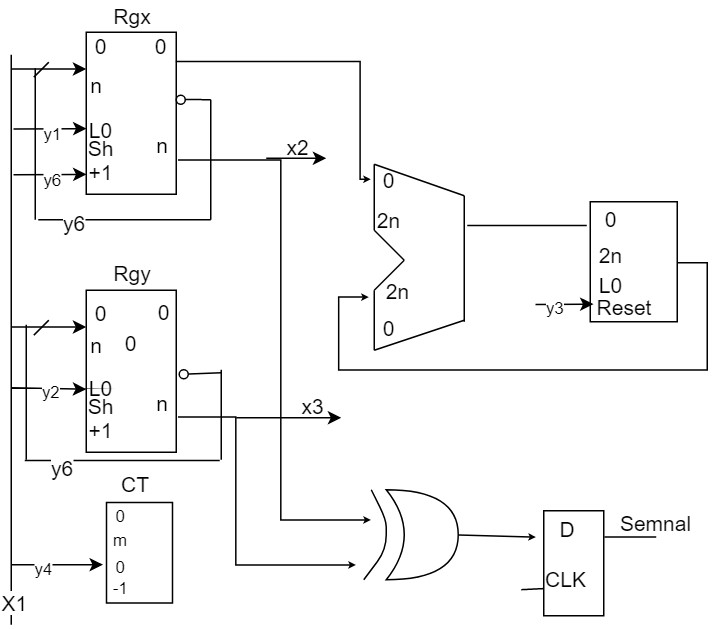
**Anexa 5**



C

t

**Anexa 6**



C

t