AL. ROGOJAN

METODA PENTRU SINTEZA SCHEMEI LOGICE A UNUI CALCULATOR NUMERIC

TEZA PENTRU OBTINEREA TITLULUI
DE DOCTOR INGINER

PARTEA 1

1974
INSTITUTUL POLITEHNIC TRAIAN VUIA
TIMISOARA

PREFATA

Prezenta teză de doctorat, "Metodă pentru sinteza schemei logice a unui calculator numeric", a fost elaborată pe parcursul lucrărilor de proiectare şi construcţie a calculatorului CETA, dat în exploatare în aprilie 1973 la I.P.Timişoara.

Pentru participarea la marele volum de muncă cerut de realizarea calculatorului, realizare care a permis verificarea metodei, aduc aici mulţumiri asist.inginer Mircea Vlăduţiu, pentru elaborarea complexelor scheme de aşezare pe plăci a circuitelor logice ale blocului de comandă al calculatorului şi pentru întocmirea tabelelor generale de cablaj. Mulţumesc tehnicianului Constantin Nănasi pentru importantele contribuţii aduse la realizarea practică a calculatorului CETA. Aduc mulţumiri fiicei mele Dorina şi Veturiei Borcean pentru faptul că în proiectul lor de diplomă, elaborat în vederea obţinerii titlului de inginer, au aplicat, prima oară, metoda de sinteză a schemei logice a unui calculator numeric, care face obiectul prezentei teze, demonstrînd şi prin aceasta utilitatea ei.

Deasemenea mulţumesc călduros conducerii Ministerului educaţiei şi învăţămîntului şi conducerii Institutului Politehnic Timişoara, care mi-au asigurat condiţii optime pentru realizarea calculatorului.

Timişoara, 4 august 1974 Autorul

CUPRINSUL
Partea 1

Capitolul 1

INTRODUCERE

Procesele care se desfăşoară într-un calculator numeric sînt foarte complexe şi, în consecinţă, schemele logice ale acestora sînt afectate şi ele, la rîndul lor, de aceeaşi complexitate.

Existenţa unei metode sistematice de sinteză a schemei logice, bazată pe o viziune clară asupra tuturor proceselor, care au loc în calculator, permite obţinerea unei scheme logice raţionale, care duce la o realizare mai economică şi mai sigură în funcţionare. Pentru acest motiv, în decursul timpului, au existat străduanii cu privire la elaborarea unei atari metode.

Au apărut astfel o seamă de lucrări, care au jalonat drumul proiectării. Numărul de lucrări, care se ocupă de proiectarea schemei logice de ansamblu a calculatorului, în care se include neapărat proiectarea circuitelor de comandă, sînt relativ puţine. Cele mai multe lucrări referitoare la proiectarea calculatoarelor tratează algebra booleană şi procedee de proiectare a schemelor elementelor logice, care intervin într-un calculator. Prin "elemente" înţelegem aici părţi din calculator cu scheme logice limitate, cum sînt: registrele, numărătoarele, decodificatoarele, sumatoarele etc. Ori adevărata complexitate începe atunci cînd, cu ajutorul acestor elemente, vrem să realizăm schema logică a calculatorului, care trebuie să fie în măsură să execute complicata listă de instrucţii.

Lucrările care se ocupă de această problemă, devenite clasice, sînt prezentate în lista bibliografică dela sfîrşitul acestei lucrări. Lista nu conţine poziţii numeroase, deoarece ne-am limitat la acele lucrări, care

au,fie o legătură directă cu proiectarea schemei logice de ansamblu a calculatorului, fie că ajung în contact cu ea.

In cele ce urmează vom trece sumar în revistă lucrări, care ni se par mai proeminenete şi care nu lipsesc din listele bibliografice a articolelor sau tratatelor din literatura de specialitate, ce se referă la proiectarea schemei logice a unui calculator numeric.

Vom aminti în primul rînd "Preliminary Discussion of the Logical Design of an Electronic Computing Instrument" de A.W.Burks, H.H.Goldstine, J.von Neumann(1947). Această lucrare, care a apărut în perioada de început a dezvoltării calculatoarelor, ridică probleme de organizare a structurii unui calculator şi tratează o seamă de algoritmi pentru operaţiile aritmetice ce se efectuează cu ajutorul calculatorului. Ea nu atinge problema metodei de proiectare a schemei logice. Am prezentat-o aici pentru că ea este lucrarea, devenită clasică, care deschide calea spre abordarea însăşi a problemelor de acest gen.

O altă lucrare, care aduce elemente foarte utile în privinţa proiectării schemei logice a unui calculator numeric este: "Design of a Digital Computer by Boolean Algebra" de R.C.Jeffrey şi I.S.Reed (1952). Conţinutul acestei lucrări a fost reluat în tratatul:"Digital Computer Design Fundamentals" de Y.Chu (1962). Calculatorul proiectat este foarte simplu pe deoparte, iar pe de altă parte structura lui este foarte diferită de cea a calculatoarelor reale. In lucrare se fac o seamă de precizări cu privire la fazele proiectării unui calculator. Astfel cele trei faze sînt: proiectarea structurii, proiectarea logică a blocurilor din structură şi proiectarea circuitelor. La rîndul său proiectarea logică

cuprinde şi ea trei etape: proiectarea funcţională, proiectarea simbolicică şi proiectarea detaliată. Proiectarea este dusă pînă la schemele logice de detaliu, dar exemplul foarte simplu, care e dat, nu îngăduie dezvoltarea metodei aşa fel, încît ea să corespundă proiectării calculatoarelor, substanţial mai complexe, din zilele noastre.

In lucrarea "The Logical Design of an Idealized General-purpose Computer" de A.W.Burks şi I.M.Copi (1956) se descrie proiectarea logică a unui calculator abstractizat, foarte simplu.

Simplitatea exemplului nu permite nici aici dezvoltarea metodei spre una adaptată la forma şi complexitatea calculatoarelor moderne. Totuşi, structura calculatorului idealizat, are blocuri care se adaptează mai bine calculatoarelor din zilele noastre.

Tratatul privind proiectarea logică a calculatoarelor numerice, rămas clasic pînă în zilele noastre, este "Logical Design of Digital Computers" de M.Phister (1957). Cu toate că în exemplul său de calcul, tratează un calculator serie, prevăzut cu o memorie cu tambur magnetic şi cu registre de circulaţie pe tambur, metoda de proiectare este bine închegată. Se stabileşte succesiunea proiectării plecînd de la lista de instrucţii la structură, procedură mai raţională decît cea inversă, practicată în lucrările precedente. Pentru scrierea ecuaţiilor logice Phister utilizează tabele, chiar dacă ele nu sînt intocmite pe instrucţii. Si aici calculatorul luat ca exemplu este foarte simplu, ceea ce limitează dezvoltarea metodei.

Lucrările "O sinteze upravlenia electronnoi ţifrovoi matematicescoi maşinî" şi "O sinteze upravlenia aritmeticeşcogo ustroistva electronnoi matematicescoi maşinî" de N.G.Bruevici (1961) prezintă proiectarea logică a

blocului de comandă al unui calculator numeric, utilizînd un anumit gen de tabele, menite să sistematizeze scrierea ecuaţiilor logice.

In lucrările "Eine Metode zum Entwurf komplexer Schaltwerke unter Verwendung spezieller Ablauf Diagramme" şi "Zur Sistematik von Mikroprogramm Werkstrukturen" de S.Wendt, prezintă interes faptul că în ele se formulează mai clar noţiunea de bloc cu borne de intrare şi ieşire, decît în alte lucrări cu privire la proiectarea de scheme logice.

Ne limităm aici cu prezentarea bibliografiei, deoarece prin aceste cîteva lucrări s-au atins punctele care formează trăsătura esenţială a metodelor de proiectare de pînă acum.

Toate aceste metode sînt, mai mult sau mai puţin, unilaterale; nu sînt suficient de sistematice şi, mai ales, nu permit cuprinderea tuturor proceselor de comandă, fie că sînt automate sau sînt declangate manual, într-o formă unică, corespunzătoare naturii lor comune.

Lipsa, pînă la ora actuală, a unei metode de descriere sistematică a funcţionării calculatoarelor numerice şi de proiectare a schemei lor logice face ca, în majoritatea covărşitoare a manualelor despre calculatoare numerice, să nici nu se încerce a se aborda problemaa cea mai complexă a proiectării dispozitivului de comandă. De aici derivă dificultăţile foarte mari de a urmări descrierile funcţionării calculatoarelor numerice.

Prezenta teză de doctorat umple acest gol prin elaborarea unei metode clare şi sistematice de descriere şi proiectare a schemei logice a unui calculator numeric. Pentru ilustrarea metodei, în lucrare se dă un exemplu complet de proiectare a unui calculator paralel real şi nu idealizat. Descrierea şi proiectarea pot fi

urmărite ugor, motiv pentru care metoda, pe lîngă valoarea ei în domeniul proiectării, are şi o importantă valoare didactică.

Capitolul 2

SINTEZA SCHEMEI LOGICE A UNUI CALCULATOR NUMERIC

Un calculator numeric este format din mai multe blocuri, între care circulă informaţie. Procesul de trecere a informaţiei dintr-un bloc în altul, precum şi modul de transformare a acestei informaţii, în timpul trecerii, este dirijat de către circuite, care produc impulsuri de comandă. Acestea din urmă formează, la rîndul lor, şi ele, un bloc al circuitelor de comandă (BCC). Blocurile calculatorului ca şi procesul de circulaţie a informaţiei, a semnalelor de comandă, sînt foarte complexe de aceea proiectarea schemei logice a blocurilor, din care este format un calculator, pune probleme dificile.

Vom prezenta aici o metodă sistematică, cu caracter de generalitate, de descriere şi proiectare, elaborată de autor, care permite atît înţelegerea relativ uşoară a funcţionării, cît şi proiectarea sistematică a schemei logice a calculatorului. Cînd vorbim de proiectare folosim şi termenul de " descriere " deoarece acestea sînt inseparabil legate una de alta.

Metoda a fost elaborată în anul 1969 şi, începînd cu anul 1970, ea se predă studenţilor de la

Secţia de calculatoare a Institutului politehnic Timişoara.

Metoda de proiectare

Proiectarea unui calculator în general constă din trei părţi, anume: proiectarea sistemului, care cuprinde determinarea caracteristicilor generale ale calculatorului; proiectarea logică, care determină schema logică a lui şi, în sfîrşit, proiectarea circuitelor, cu ajutorul cărora va fi construit. Aici ne limităm la proiectarea schemei logice a unui calculator căruia i-au fost fixate, în prealabil, caracteristicile generale.

Pe parcurs se va arăta că metoda de proiectare se poate utiliza atît la calculatoarele sincrone sau asincrone, cît şi la cele paralele sau serie.

Metoda constă în următoarele:

Se alcătuieşte lista de instrucţii a calculatorului. La această operaţie se ţine seama de tipul de probleme care se vor rezolva cu ajutorul calculatorului, de mărimea lui, de uşurinţa programării. Soluţia finală va fi desigur un compromis între uşurinţa programării şi complexitatea schemei calculatorului.

Odată lista de instrucţii fixată se va descrie prin cuvinte fiecare instrucţie cît mai în detaliu, arătîndu-se limpede ce anume operaţii trebuie să fie executate prin instrucţia respectivă. Se va arăta totodată ce anume se păstrează şi ce anume se pierde din informaţia care intră în joc.

Fixăm acum modul de organizare al executării instrucţiilor. Astfel instrucţiile vor fi executate în mai multe faze. Prin faze înţelegem aici un interval de timp, care durează cît ciclul calculatorului -

ciclu a cărui durată se va preciza mai jos - în care apare o secvenţă determinată de impulsuri de tact.Secvenţa de impulsuri determină o secvenţă de microoperaţii caracteristică pentru faza parcursă.Putem avea spre exemplu fazele ADUCERE, INDIRECT şi EXECUTIE. In faza ADUCERE impulsurile de tact,pe care le vom nota cu i, vor provoca secvenţa de microoperaţii prin care se aduce din memorie instrucţia,ce urmează a fi executată,în registrul de instrucţii.Tot în această fază se execută şi microoperaţiile de efectuare a instrucţiei,pentru acele instrucţii care se execută într-o singură fază.Faza INDIRECT conţine microoperaţiile care se efectuează atunci cînd în instrucţia scoasă în faza de ADUCERE se arată că adresa conţinută în instrucţie nu este adresa unui operand ci adresa adresei unui operand.In faza EXECUTIE se efectuează secvenţa de microoperaţii în care are loc îndeplinirea unei instrucţii pentru care sînt necesare mai multe faze,cum ar fi în cazul adunării,cînd în faza de EXECUTIE trebuie să scoatem din memorie operandul şi apoi să efectuăm adunarea.

Pentru a se putea face,în calculator,deosebire între o instrucţie şi alta,precum şi pentru a putea şti care sînt microoperaţiile care trebuiesc executate,e necesar să se precizeze formatul cuvîntului instrucţie şi semnificaţia fiecărui bit din cuvînt. Se vor considera pe deoparte instrucţiile cu referire la memorie,iar pe pe de alta grupele de instrucţii microprogramate. La instrucţiile care fac parte din grupe nu e nevoie să se extragă informaţie din memorie şi în consecinţă partea de adresă poate fi folosită în scopul microprogramării mai multor instrucţii de bază.Astfel de instrucţii microprogramate se pot execu-

ta mai multe în timpul fazei unice de ADUCERE.La instrucţiile microprogramate trebuie de asemenea precizat care din impulsurile i efectuează diversele instrucţii microprogramate (primele impulsuri i sînt necesare pentru aducerea din memorie a grupului de instrucţii), precum şi restricţiile care rezultă la microprogramare. Spre exemplu nu se pot executa simultan instrucţii microprogramate,care se referă la unul şi acelaşi registru.

Calculatorul,pentru a putea executa instrucţiile, trebuie să fie alcătuit din diverse blocuri.Precizăm acum toate blocurile din care este format calculatorul. Aceste blocuri,vor fi spre exemplu: registrul acumulator, registrul instrucţiei,registrul numărător de adrese,generatbrul de impulsuri de orologiu,generatorul de faze,memoria,dispozitivele periferice.La aceste părţi mai adăugăm încă una (încă un bloc) alcătuită din circuitele logice în care se formează semnalele de comandă.Menţionăm aici,pentru a evita confuziile, că dispozitivul de comandă al calculatorului nu este format numai din blocul circuitelor de comandă (BCC),amintit imediat mai sus,ci şi din alte blocuri cum sînt: registrul instrucţiei (RI), numărătorul de adrese (NA), descifratorul codului operaţiei şi altele.Blocurile, a căror schemă nu este încă cunoscută,se reprezintă simbolic,spre exemplu,prin dreptunghiuri.

Dreptunghiurile,în afară de cel care prezintă BCC, vor fi prevăzute cu borne la care sosesc semnale de comandă de la BCC, precum şi cu borne prin care blocurile trimit semnale de informare către BCC.Aceste borne se prevăd,într-o primă etapă,pe baza cunoştinţelor noastre despre funcţionarea unor asemenea blocuri.După cum se va vedea,aceste borne pot fi prevăzute destul de corect în această etapă.Evident că BCC

va cumula toate bornele tuturor blocurilor,dar într-un sens invers.Intre blocuri circulă informaţie.In consecinţă,vor fi prevăzute la fiecare bloc (în afară de BCC) bornele corespunzătoare pentru trecerea informaţiei dintr-un bloc în altul.Intrucît vor apărea foarte multe borne e necesar să fie notate mnemonic pentru a li-se putea cunoaşte, uşor,funcţia.

Pentru precizarea lucrurilor,este util în această etapă a se alcătui o schemă monofilară a circulaţiei informaţiei între diversele blocuri.

Bineînţeles că,atunci cînd descriem diversele blocuri,trebuie să ne fixăm şi asupra comportării lor în timp.Aceasta se poate face plecînd de la memorie. Se fixează ciclul memoriei şi timpul de acces, adică intervalul de timp după care,din momentul primirii comenzii "citeşte", informaţia ajunge în registrul de memorie (RM). In funcţie de ciclul memoriei se fixează ciclul calculatorului,care trebuie să fie cu ceva mai mare decît ciclul memoriei.Prin ciclul calculatorului înţelegem durata unei secvenţe de impulsuri date de generatorul de orologiu.După cum am văzut anumite instrucţii pot fi efectuate într-un singur ciclu de calculator,iar altele în mai multe cicluri resp. faze . Numărul acestor faze,cum am arătat,se fixează în prealabil, în funcţie de instrucţiile pe care dorim să le executăm.Necesitatea alegerii ciclului calculatorului cu ceva mai lung decît al memoriei derivă din faptul că o nouă adresare la memorie nu se poate face decît după ce memoria şi-a încheiat ciclul propriu.In timpul ciclului memoriei nu este îngăduit a se schimba conţinutul registrului de adrese,deoarece în acest caz reînscrierea informaţiei în memoria cu citire distructivă s-ar putea face eronat.

De asemenea trebuie să avem stabilit timpul maxim în care se execută diversele microoperaţii, spre exemplu, timpul maxim în care se face o adunare în acumulator, etc.

Intrucît numărul de impulsuri, care formează o secvenţă a generatorului de orologiu, secvenţă care durează cît ciclul calculatorului, depinde şi de durata diverselor microoperaţii, acum putem să precizăm atît numărul impulsurilor cît şi intervalul de timp dintre ele. Vom avea impulsuri ascuţite i, precum şi impulsuri late I, acestea din urmă fiind necesare pentru a putea realiza, în condiţii bune, funcţii de coincidenţă. Intr-o secvenţă de impulsuri avem acelaşi număr de impulsuri ascuţite şi late, cele ascuţite fiind plasate, în timp, la mijlocul celor late. Mai precizăm că toate impulsurile dintr-o secvenţă apar la ieşiri diferite ale generatorului de orologiu.

Cu aceste elemente, acum, pornim la sinteza circuitelor din blocul circuitelor de comandă. Pentru aceasta alcătuim nişte tabele, care reprezintă descrierea în detaliu a fiecărei faze ce intră în componenţa instrucţiilor. Alcătuim tabele nu numai pentru diversele faze ale instrucţiilor, dar şi pentru comenzile manuale efectuate de la pupitrul de comandă. Tabelul are cîte o coloană pentru fiecare impuls dat de generatorul de orologiu. In coloana respectivă se trec, prin reprezentări mnemonice, bornele din blocurile calculatorului, la care este dus impulsul corespunzător al generatorului de orologiu. Tot aici se trec toate condiţionările pe care trebuie să le satisfacă impulsul de comandă rezultat din impulsul generatorului de orologiu. In coloana ultimului impuls al fazei se formează, în funcţie de condiţiile existente, un impuls

care stabileşte faza următoare pe care o va executa calculatorul.Acest impuls este dus la intrarea potrivită a generatorului de faze,care, la rîndul lui,ajunge în starea corespunzătoare fazei dorite.Prin urmare, în tabel sînt trecute secvenţial toate microoperaţiile din care va fi formată faza respectivă a instrucţiei. In partea de jos a tabelului se dau explicaţii asupra diverselor microoperaţii.

Cu ocazia întocmirii tabelelor se completează, dacă este necesar,bornele blocurilor pentru ca ele să poată îndeplini toate funcţiile cerute de executarea instrucţiilor.In această fază de lucru am putea constata că este necesar să prevedem un nou bloc, a cărui necesitate nu am observat-o de la început. In timpul întocmirii tabelelor facem deci o permanentă îmbunătăţire a formei prevăzute iniţial pentru blocuri. In ultimă analiză am putea chiar să precizăm felul blocurilor şi bornele lor abia pe parcursul întocmirii tabelelor.

Tabelele reprezintă acum descrierea completă a modului de funcţionare al calculatorului şi putem trece la scrierea ecuaţiilor logice după datele din tabele,care apoi vor duce la sinteza schemei blocului circuitelor de comandă.

Scrierea ecuaţiilor se face în felul următor:

In tabelelse găsesc funcţii (variabile) care au fost asociate bornelor blocurilor.Acestea reprezintă funcţii logice de intrare.Luăm o astfel de funcţie logică şi scriem ecuaţia booleană care îi corespunde, parcurgînd absolut toate tabelele.Bineînţeles că termenii funcţiei,corespunzători diferitelor tabele, se leagă între ei prin operaţia logică SAU.Facem această operaţie pentru toate funcţiile rezultate din tabele.

Putem scrie funcţiile şi numai pentru un grup de tabele,dacă aceasta ni se pare util,urmînd apoi,ca,în final,să unim expresiile obţinute pentru aceeaşi funcţie legîndu-le prin operaţia SAU.La scrierea ecuaţiilor va trebui să fim atenţi să introducem în funcţiile respective şi variabilele care corespund instrucţiei,respectiv fazei executate,ceea ce va asigura ca,pentru anumită instrucţie şi o anumită fază,să se obţină o secvenţă de microoperaţii precis determinată.Evident,pentru uşurarea scrierii ecuaţiilor ne putem servi de cunoscutele tabele de combinaţii.

Avînd acum la dispoziţie toate ecuaţiile de intrare,procedăm la minimizarea lor şi apoi la executarea sintezei schemei blocului circuitelor de comandă.

In continuare ne ocupăm,pe rînd,de fiecare bloc despre care ştim acum exact,după prezenţa bornelor,ce fel de operaţii trebuie să îndeplinească.Prin procedee cunoscute se elaborează schemele logice ale blocurilor, operaţie cu care încheiem proiectarea schemei logice a calculatorului.

Metoda prezentată,de proiectare a schemei logice a unui calculator numeric,are pe de-o parte,marele avantaj că priveşte diversele părţi ale calculatorului ca blocuri înzestrate cu borne de intrare şi ieşire şi în consecinţă schema logică a blocului însăşi este evitată în faza proiectării schemei logice a blocului cel mai complex,acel al circuitelor de comandă,iar pe de altă parte utilizează tabele astfel alcătuite încît ele permit o descriere exactă,uşor de urmărit,şi care oferă o privire de ansamblu asupra funcţionării calculatorului.

Toate părţile schemei calculatorului în care se produc comenzi (funcţii de comandă) sînt descrise,sub

aspectul funcţionării,unitar,în acelaşi grup de tabele.

Un alt avantaj al metodei este caracterul ei de generalitate.Astfel ea poate fi folosită şi în cazul proiectării schemei logice a unui calculator sincron, asincron, paralel, serie, microprogramat.Spre exemplu în cazul proiectării schemei logice a unui calculator asincron,în tabelele care descriu instrucţiile,în loc de impulsurile i,date de generatorul de orologiu,apar impulsurile pe care blocurile,care au executat microoperaţiile le generează la terminarea microoperaţiilor.

In cazul proiectării unui calculator cu dispozitiv de comandă microprogramat,instrucţiile se realizează prin microprograme formate din microinstrucţii. In acest caz, se formează microprogramele pentru fiecare instrucţie,iar pentru microinstrucţii se întocmesc tabelele de descriere.

Metoda îngăduie ca blocurile să fie privite ca părţi formate şi ele, la rîndul lor, din subblocuri care posedă o schemă proprie de comandă,adică o schemă numită "de comandă locală" spre deosebire de cea care comandă blocurile şi care e numită "schemă de comandă centrală".

Metoda prezintă flexibilitate.Astfel putem începe proiectarea plecînd de la blocuri existente.Spre exemplu avem proiectat deja dispozitivul aritmetic pe baza unor algoritmi privind efectuarea operaţiilor aritmetice.In acest caz însă pot apărea limitări în privinţa instrucţiilor,fie necesitatea de a se recurge la completarea dispozitivului aritmetic.Cel mai logic mers al proiectării este însă acela care porneşte de la lista de instrucţii.De fapt lista de instrucţii reprezintă punerea problemei,ea arată în amănunt ce anume trebuie să îndeplinească calculatorul,deci,într-un fel,reprezintă enunţul problemei.

Pentru uşurinţa urmăririi proiectării dăm mai jos, într-o exprimare succintă, succesiunea etapelor:

1. - Se elaborează lista de instrucţii a calculatorului şi a comenzilor date manual prin chei dela P.C. Se stabileşte denumirea mnemonică. Se descrie pe scurt fiecare instrucţie şi funcţiunea fiecărei chei.
2. - Se organizează execuţarea instrucţiilor şi se fixează numărul de faze.
3. - Se stabileşte formatul cuvîntului instrucţie; se stabilesc grupele de instrucţii microprogramate, se precizează semnificaţie biţilor din cuvîntul instrucţie. De asemenea se precizează restricţiile privind executarea instrucţiilor microprogramate.
4. - Se descrie în detaliu instrucţia utilizînd semnificaţia biţilor. Se face de fapt încă o descriere pentru proiectare. Se descrie în detaliu modul cum se execută fiecare comandă dată manual dela P.C.
5. - Se stabilesc blocurile din care este alcătuit calculatorul şi se reprezintă simbolic aceste blocuri prin dreptunghiuri.
6. - Se prevăd borne de intrare şi ieşire la blocuri şi se notează cu inscripţii mnemonice.
7. - Se stabileşte circulaţia informaţiei în calculator.
8. - Se fixează ciclul memoriei şi, în funcţie de el, ciclul calculatorului.
9. - Se stabileşte durata diverselor microoperaţii, în funcţie de genul circuitelor cu care dorim să construim calculatorul.
10. - Se stabileşte numărul de impulsuri i, care formează o secvenţă a generatorului de orologiu (un ciclu al calculatorului, o fază) şi se precizează intervalul dintre impulsuri.
11. - Se alcătuiesc tabele pentru toate fazele din

care este formată fiecare instrucţie în parte,precum şi pentru fiecare cheie de comandă de pe pupitrul de comandă. In tabele se arată ce microoperaţie declanşează fiecare impuls i precum şi condiţionările necesare.

12. - Se completează blocurile cu borne suplimentare de intrare sau ieşire, dacă din descrierea din tabele rezultă că acest lucru este necesar.

13. - Se scriu ecuaţiile logice după datele din tabele,ecuaţii care descriu semnalele de comandă ce ajung la diversele borne ale blocurilor.

14. - Se minimizează ecuaţiile.

15. - Se face,pe baza ecuaţiilor minimizate, sinteza schemei blocului circuitelor de comandă.

16. - Se proiectează,pe rînd,schemele logice a fiecărui bloc din care este format calculatorul,plecînd de la bornele de intrare şi ieşire a blocurilor, care arată exact ce operaţii trebuie să îndeplinească fiecare bloc. Pentru acestea se folosesc procedee cunoscute.

La elaborarea diverselor etape în ordinea numerotării putem reveni şi putem face completări, după ce am obţinut informaţii prin elaborarea unei etape cu număr de ordine mai mare.Spre exemplu după ce am fixat lista de instrucţii şi am descris pe scurt fiecare instrucţie (pct.1),într-o altă etapă (pct.4) descriem din nou instrucţiile, mai în detaliu,folosind datele obţinute prin elaborarea etapelor pct.2 şi 3.

Tabelele se întocmesc pe baza descrierilor în detaliu a instrucţiilor efectuate la pct.4. Cele 4 puncte cu privire la instrucţii (1-4) se întrepătrund.Numerotarea nu înseamnă că elaborarea completă se face strict în ordinea crescătoare a numerelor. Adică după ce am elaborat un punct putem să ne întoarcem şi să completăm elaborarea altuia parcurs mai înainte.

Analizînd cele de mai sus se vede că sinteza schemei logice a unui calculator numeric poate fi împărţită în 4 părţi anume: partea I (pct.1-4) care se ocupă de descrierea instrucţiilor; partea II (pct.5-10) care se ocupă de blocurile ce alcătuiesc calculatorul; partea III (pct.11-15) care se ocupă de întocmirea tabelelor, scrierea ecuaţiilor şi sinteza schemei blocului circuitelor de comandă; partea IV (pct.16) care se ocupă cu stabilirea schemei logice a fiecărui bloc, din care este format calculatorul.

In cele ce urmează se va proiecta schema logică a calculatorului CETA cu ajutorul metodei descrisă mai sus.

CARACTERISTICILE GENERALE ALE CALCULATORULUI CETA

Pe baza unor studii prealabile se precizează următoarele:

CETA (Calculator Electronic Tranzistorizat Automat) va fi un calculator universal, binar, paralel, cu virgulă fixă, care va lucra cu cuvinte lungi de 24 biţi. Reprezentarea numerelor se va face în memorie, în semn-mărime, în acumulator, în complement modificat faţă de doi. Instrucţiile calculatorului vor fi cu o singură adresă.

Memoria cu care va lucra va fi realizată cu miezuri de ferită şi va avea un ciclu de 7,5 microsecunde.

Dispozitivul aritmetic va avea timpul de adunare maxim de 2 microsecunde.

Dispozitivul de comandă al calculatorului va fi astfel proiectat, încît să permită ca instrucţiile fără referire la memorie să poate fi microprogramate, prin utilizarea biţilor din cîmpul de adresă al instrucţiilor cu referire la memorie.

Calculatorul va avea posibilitate de adresare indirectă, precum şi posibilităţi de întrerupere la cererea perifericelor.

Calculatorul va fi realizat numai cu componente autohtone şi va avea o viteză medie de 150-200.000 instrucţiuni pe secundă pe program.

Instrucţiile calculatorului CETA

Lista de instrucţii a calculatorului corespunde listelor de instrucţii care se utilizează în prezent la calculatoarele mici. Unele dintre instrucţiile cuprinse în listă sînt complexe şi de mare eficacitate. De acest fel sînt, spre exemplu, instrucţiile "măreşte cu unu şi omite dacă e zero" sau "compară cu conţinutul registrului B şi omite la inegalitate". Cum calculatorul va fi utilizat în învăţămînt, lista a fost extinsă la 84 instrucţii de bază, spre a se putea face diverse experienţe de programare.

Instrucţiile sînt împărţite pe grupe. Astfel avem:

- 22 instrucţii cu referire la memorie (IRM)
- 42 instrucţii cu referire la registre (IRR), care la rîndul lor sînt subîmpărţite în două grupe, anume: grupa de deplasări şi rotiri (GDR) formată din 14 instrucţii şi grupa de modificări şi omiteri (GMO) formată din 28 instrucţii.
- 2o instrucţii de introducere-extragere (IIE)

In cele ce urmează se vor prezenta grupele de instrucţii pe rînd, precizînd tot odată organizarea şi codificarea cuvîntului instrucţie, precum şi restricţiile care apar la microprogramarea instrucţiilor grupelor GDR şi GMO.

Instrucţiile RM, în număr de 22, sînt de mai multe tipuri. Astfel avem:

- Instrucţii de încărcare prin care se aduce într-un anumit registru un cuvînt din memorie, de la adresa specificată în cîmpul adresei din instrucţie.
- Instrucţii de memorare prin care un cuvînt, dintr-un anumit registru, se duce în memorie la adresa specificată în cîmpul adresei din instrucţie.

- Instrucţii aritmetice prin care se efectuează operaţii aritmetice între un număr aflat într-unul din registrele dispozitivului aritmetic şi un al doilea număr aflat în memorie la adresa specificată în cîmpul adresei din instrucţie.

- Instrucţii logice prin care se efectuează operaţii logice între biţii unui număr aflat în registrul acumulator şi biţii unui al doilea număr aflat în memorie la adresa specificată în cîmpul adresei din instrucţie.

- Instrucţii de salt necondiţionat prin care se obţine ca instrucţia următoare să se ia din memorie de la adresa specificată în cîmpul adresei din instrucţia de salt şi nu de la adresa rezultată prin mărirea cu unu a adresei curente.

- Instrucţii speciale. Acestea sînt "măreşte cu unu şi omite dacă e zero" şi "compară cu conţinutul lui B şi omite la inegalitate". Se vede că aceste instrucţii verifică existenţa unor condiţii anumite, care dacă sînt îndeplinite se omite instrucţia următoare. Aceasta înseamnă că în loc să luăm instrucţia următoare de la adresa obţinută prin mărirea cu unu a adresei curente, o luăm de la adresa rezultată din mărirea cu doi a adresei curente.

Instrucţiile cu referire la memorie au nevoie, pentru a fi executate, de informaţie din memorie şi în consecinţă conţin, pe lîngă codul operaţiei format din 5 biţi şi un cîmp pentru bitul de adresare indirectă (Iα). Organizarea cuvîntului instrucţie se vede în fig.2.1, denumirea instrucţiilor în tabelul 2.1, iar codul operaţiei corespunzător fiecărei instrucţii în tabelul 2.2.

INSTRUCȚIILE CU REFERIRE LA MEMORIE (IRM)

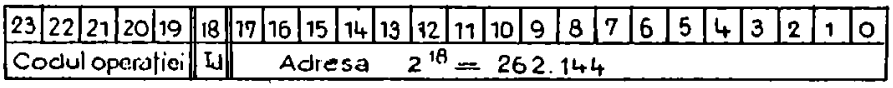


FIG. 2.1

TABELUL 2.1

Nr. crt.	DENUMIREA MNEMONICĂ A INSTR.	CONȚINUTUL INSTRUCȚIEI
1	INB	Încarcă req. B
2	INR	Încarcă req. R
3	INQ	Încarcă req. Q
4	INA	Încarcă req. A
5	IDA	Încarcă deîmpărțitul în req. A
6	MEQ	Memorează conținutul req. Q
7	MG2	Memorează conținutul lui G2
8	MEA	Memorează conținutul req. A
9	MPQ	Memorează produsul din req. Q
10	MCQ	Memorează câtul din req. Q
11	MRA	Memorează rezultatul adunării sau scăderii din req. A
12	ADA	Adună la conținutul req. A
13	SCA	Scade din conținutul req. A
14	INM	Înmulțește
15	IMP	Împarte
16	SAI	SAU INCL cu conținutul req. A
17	SAE	SAU EXCL cu conținutul req. A
18	ȘIA	Și cu conținutul req. A
19	SLT	Salt
20	SSP	Salt la subprogram
21	MOZ	Mărește cu unu și omite dacă e zero
22	CBO	Compară cu conținutul req. B și omite la inegalitate

INSTRUCȚIILE CU REFERIRE LA MEMORIE (IRM)

TABELUL 2.2

Nr. crt.	DENUMIREA MNEMONICĂ A INSTR.	CODUL OPERAȚIEI				
1	INB	0	0	0	0	1
2	INR	0	0	0	1	0
3	INQ	0	0	0	1	1
4	INA	0	0	1	0	0
5	IDA	0	0	1	0	1
6	MEQ	0	0	1	1	0
7	MG2	0	0	1	1	1
8	MEA	0	1	0	0	0
9	MDQ	0	1	0	0	1
10	MCQ	0	1	0	1	0
11	MRA	0	1	0	1	1
12	ADA	0	1	1	0	0
13	SCA	0	1	1	0	1
14	INM	0	1	1	1	0
15	IMP	0	1	1	1	1
16	SAI	1	0	0	0	0
17	SAE	1	0	0	0	1
18	ȘIA	1	0	0	1	0
19	SLT	1	0	0	1	1
20	SSP	1	0	1	0	0
21	MOZ	1	0	1	0	1
22	CBO	1	0	1	1	0
23	GDR	1	1	0	0	0
24	GMO	1	1	0	0	1
25	IIE	1	1	0	1	1
26		1	1	0	1	1
27		1	1	1	0	0
28		1	1	1	0	1
29		1	1	1	1	0
30		1	1	1	1	1

Tot în tabelul 2.2 se vede că am atribuit cîte un cod al operaţiei şi pentru instrucţiile microprogramate din grupa de deplasări-rotiri (GDR) şi din grupa modificări-omiteri (GMO), precum şi pentru instrucţiile de introducere-extragere (IIE). Despre aceste instrucţii vom vorbi în cele ce urmează.

Instrucţiile cu referire la registre (IRR)

Cum aceste instrucţii nu au nevoie de informaţie din memorie, pentru a fi executate, şi în consecinţă cîmpul utilizat la IRM pentru adresă este disponibil, îl vom utiliza pentru a indica, prin biţii din care e format, efectuarea unor operaţii (instrucţii). Astfel de operaţii (instrucţii) pot fi mai multe şi se vor executa succesiv. Se obişnuieşte să se numească acest gen de operaţii: instrucţii microprogramate, prin analogie cu instrucţiile obţinute prin metoda propriu zisă de microprogramare.

Aceste instrucţii microprogramate, după cum am văzut, le vom împărţi în alte două grupe: grupa de modificări şi omiteri (GMO), formată din 28 instrucţii şi grupa de deplasări şi rotiri, formată din 14 instrucţii. Fiecăreia din grupe i-se ataşează un cod de operaţie propriu, pe care îl vom numi cod de grupă şi e format din 5 biţi; care este acelaşi pentru toate instrucţiile grupei.

Microprogramarea acestor instrucţii este supusă unor restricţii. Astfel nu pot fi microprogramate în acelaşi cuvînt instrucţii contradictorii, spre exemplu, deplasarea simultană la stînga şi la dreapta a conţinutului aceluiaşi registru. In general nu pot fi microprogramate în acelaşi cuvînt instrucţii, care într-un fel sau altul se perturbă reciproc. Instrucţiile microprogramate se vor executa în 3 timpi, la impulsurile de

tact i_4, i_5 şi i_6. Impulsurile de tact i_0-i_3 sînt utilizate pentru aducerea din memorie a cuvîntului instrucţie, iar ultimul impuls din secvenţa de tact, i_7, pentru îndreptarea calculatorului spre faza care urmează.

Ordinea în care se pot executa diversele instrucţii, ordine fixată de succesiunea impulsurilor de tact i_4, i_5 şi i_6, este determinată de necesităţi de programare. Astfel, în tabelul 2.7 instrucţia AZB (adă la zero B) trebuie să se execute înaintea instrucţiei AINB (conţinutul lui A se trece în B), motiv pentru care AZB se execută de către i_4, iar AINB de către i_5. Instrucţiile care se pot executa simultan, şi în anumite cazuri e util să se execute simultan, sînt efectuate de către acelaşi impuls de tact. Aceste instrucţii, în mod obligatoriu, în timpul efectuării trebuie să nu se perturbe reciproc.

Pentru a ţine seama de restricţii şi a face microprogramarea corect trebuie să luăm numai o singură instrucţie dintr-o coloană din tabelele de restricţii. Se vede în tabele că pentru a asigura această corectitudine a fost necesară împărţirea tabelelor şi pe orizontală. Facem în totdeauna microprogramarea păstrîndu-ne în cadrul aceluiaşi compartiment pe orizontală.

Un important avantaj al acestui mod de organizare este faptul că putem alcătui cuvinte instrucţii cu un conţinut foarte variat, deci realizăm multe instrucţii complexe. Un alt avantaj este creşterea vitezei prin mai buna folosire a timpului. Astfel la GMO putem microprograma, la limită, într-un singur cuvînt instrucţie, care se execută într-o singură fază (8 microsec.) 13 instrucţii de bază.

Instrucţiile microprogramate din grupa de deplasări şi rotiri (GDR)

Această grupă conţine următoarele tipuri de instrucţii:

- Instrucţii de deplasare aritmetică la stînga şi la dreapta a conţinutului registrelor A şi Q. Deplasarea, pentru simplificarea circuitelor calculatorului, se face numai cu cîte un rang.

- Instrucţii de rotire la stînga şi la dreapta asupra conţinutului registrelor A şi Q. Prin rotire se înţelege o deplasare a conţinutului unui registru la care un capăt (ieşirea) este legată cu celălalt capăt (intrarea). Se efectuează prin urmare un proces de circulaţie a informaţiei. Aici, în cazul rotirii la stînga, pe lîngă instrucţiile care rotesc cu un singur rang, avem şi instrucţii care rotesc cu două ranguri.

- Instrucţii de omitere la care condiţia omiterii este existenţa unui număr par în registrul A respectiv Q. Aceste instrucţii intervin adesea împreună cu instrucţiile de deplasare şi rotire.

- Instrucţiile AZL şi NOP. Instrucţia AZL intervine adesea împreună cu instrucţiile de deplasare şi rotire şi e util să poată fi microprogramată în acelaşi cuvînt cu acestea. Instrucţia AZL este prevăzută şi în GMO. Rolul instrucţiei NOP se va preciza mai tîrziu la descrierea instrucţiilor.

In fig 2.2 se arată formatul instrucţiilor din GDR. In tabelul 2.3 este prezentată lista instrucţiilor. Din tabelul 2.4 rezultă modul în care se poate face corect microprogramarea. Menţionăm că în acest tabel nu este introdusă instrucţia NOP, care prin faptul că determină lipsa oricărei operaţii, se execută întodeauna

INSTRUCȚIILE MICROPROGRAMATE DIN GRUPA DE DEPLASĂRI ȘI ROTIRI (GDR)

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	0	A/a																		

FIG. 2.2

TABELUL 2.3

Nr. crt.	DENUMIREA MNEMONICĂ A INSTR.	CONȚINUTUL INSTRUCȚIEI
1	NOP	Nici o operație
2	AZL	Adă la zero L
3	OAP	Omite dacă A este par
4	OQP	Omite dacă Q este par
5	DSA	Depl. aritmetic la stga cu un rang reg A
6	DSQ	Depl. aritmetic la stga cu un rang reg Q
7	DDA	Depl. aritmetic la dr. cu un rang reg. A
8	DDQ	Depl. aritmetic la dr. cu un rang reg. Q
9	RSA	Rot. stga cu un rang reg. A+L
10	RSQ	Rot. stga cu un rang. reg. Q
11	RDA	Rot. dr. cu un rang. reg. A+L
12	RDQ	Rot. dr cu un rang. reg. Q
13	RSA2	Rot. stga cu 2 ranguri reg. A+L
14	RSQ2	Rot. stga cu 2 ranguri reg. Q

Instrucții de microprogramare la instrucțiile GDR

Tabelul 2.4

Col 1	col2	col3	col4
i_4	i_5	i_5	i_6
DSA	AZL	OAP	DSA
DDA			DDA
RSA			RSA
RDA			RDA
DSA	—	—	DSA
DDA			DDA
RSA			RSA
RDA			RDA
RSA2			RSA2
DSQ	AZL	OQP	DSQ
DDQ			DDQ
RSQ			RSQ
RDQ			RDQ
DSQ	AZL	—	DSQ
DDQ			DDQ
RSQ			RSQ
RDQ			RDQ
RSQ2			RSQ2

Obs. Instrucțiile RSQ2 din col 1 sînt executate cu impulsurile i_4 și i_5, iar aceleași instrucții din col 4 cu impulsurile i_6 și i_7.

INSTRUCȚIILE MICROPROGRAMATE DIN GRUPA DE DEPLASĂRI ȘI ROTIRI (GDR)

TABELUL 2.5

Nr. crt	DENUMIREA MNEMONICĂ A INSTR.	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		1	1	0	0	0	A/Q																		
1	NOP	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	AZL	1	1	0	0	0								1											
3	OAP	1	1	0	0	0	1									1									
4	OQP	1	1	0	0	0	0									1									
5	DSA	1	1	0	0	0	1	1										1							
6	DSQ	1	1	0	0	0	0	1										1							
7	DDA	1	1	0	0	0	1		1										1						
8	DDQ	1	1	0	0	0	0		1										1						
9	RSA	1	1	0	0	0	1			1										1					
10	RSQ	1	1	0	0	0	0			1										1					
11	RDA	1	1	0	0	0	1				1										1				
12	RDQ	1	1	0	0	0	0				1										1				
13	RSA2	1	1	0	0	0	1					1										1			
14	RSQ2	1	1	0	0	0	0					1										1			

singură.In tabelul 2.4 se arată şi impulsurile i de către care se execută diferitele instrucţii.Deasemenea se vede din tabel că instrucţiile de deplasare şi rotire se pot executa atît i_4 cît şi la i_6, adică în acelaşi cuvînt instrucţie se pot microprograma de două ori instrucţiile de deplasare şi rotire,avînd între ele instrucţia AZL şi sau instrucţiile de omitere. Aceste posibilităţi uşurează programarea.(Vezi obs.din tab.2.4).

In tabelul 2.5 se prezintă codificarea biţilor cuvîntului instrucţie pentru fiecare din cele 14 instrucţii.Aici bitul 18 este utilizat pentru a indica la care dintre cele două registre A respectiv Q se referă instrucţia.Anume cînd bitul 18 este unu instrucţia se referă la registrul A, iar cînd este zero se referă la registrul Q.

Instrucţiile microprogramate din grupa de modificări şi omiteri (GMO)

Această grupă conţine următoarele tipuri de instrucţii:

- Instrucţii de aducere în starea unu sau zero a unui anumit registru.
- Instrucţiuni pentru complementarea conţinutului unui registru.
- Instrucţiuni pentru adunarea unităţii la conţinutul unui registru sau chiar a conţinutului unui registru la conţinutul acumulatorului (ADMR).
- Instrucţiuni pentru trecerea conţinutului unui registru în alt registru.
- Instrucţiuni de omitere condiţionată şi necondiţionată.

In fig.2.3 se arată formatul instrucţiunilor din GMO. In tabelul 2.6 este prezentată lista instruc-

INSTRUCTIILE MICROPROGRAMATE DIN GRUPA DE MODIFICĂRI ŞI OMITERI (GMO)

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	1	COL 1		COL 2		COL 3		4	COL 5		6	COL 7		COL 8		9	10	11	12		

FIG. 2.3

TABELUL 2.6

Nr. crt.	DENUM. MNEMON. A INSTR.	CONŢINUTUL INSTRUCŢIEI
1.	AZA	Adă la zero A
2.	AZB	Adă la zero B
3.	AZQ	Adă la zero Q
4.	AZL	Adă la zero L
5.	G1 IN Q	Conţinutul lui G1 se trece în Q
6.	AZMR	Adă la zero mărimea lui R
7.	AZSR	Adă la zero semnul lui R
8.	AUB	Adă la unu B
9.	AUL	Adă la unu L
10.	CML	Complementează L
11.	ADMR	Adună partea de mărime a numărului din R la cont. lui A
12	CMR	Complementează partea de mărime a lui R faţă de 2 în A
13.	RTP	Rotunjeşte produsul
14.	AD1SnA	Adună unu în rangul semnului lui A
15.	AINB	Conţinutul lui A se trece în B
16.	NA IN Q	Conţinutul lui NA se trece în Q
17.	B IN R	Conţinutul lui B se trece în R
18.	Q IN R	Conţinutul lui Q se trece în R
19.	Q IN B	Conţinutul lui Q se trece în B
20.	AINQ	Conţinutul lui A se trece în Q
21.	MUA	Măreşte cu unu A
22	OLZ	Omite dacă L=0
23.	OA+	Omite dacă semnul lui A e zero
24.	OAP	Omite dacă A e par
25	OAZ	Omite daca A e zero
26.	OSI	Omite cu sens invers
27.	OND	Omite neconditionat
28	ADSnR	Adună semnul reg. R la A

ţiilor.Din tabelul 2.7 rezultă modul cum se poate face corect microprogramarea.Aici se arată şi impulsurile i de către care se execută diferitele instrucţii.

In tabelul 2.8 se prezintă codificarea biţilor cuvîntului instrucţie pentru fiecare din cele 28 instrucţii.Din acest tabel se vede că datorită numărului mare de instrucţii (28) nu putem microprograma fiecare instrucţie cu ajutorul unui singur bit, ci sîntem obligaţi să îmbinăm mai mulţi biţi (vezi diversele coloane) şi să folosim descifratoare astfel încît cu 3 biţi îmbinaţi (vezi col.8) putem microprograma 2^3=8 instrucţii.Aceasta este microprogramarea pe verticală.

Instrucţiile de introducere-extragere (IIE)

Instrucţiile IIE, în număr de 2o, conţin,în formatul cuvîntului instrucţie,un cîmp pentru codul de grupă, un cîmp pentru codul de selecţie,care precizează periferiCul cu care se lucrează şi un cîmp pentru codificarea instrucţiilor.De fapt din cele 2o instrucţii lo sînt instrucţii cu referire la registre şi numai lo sînt instrucţii propriu zise de introducere şi extragere. Am introdus cele zece instrucţii cu referire la registre aici pentru a utiliza mai bine posibilităţile de codificare din cuvîntul instrucţie,care numai cu instrucţiile IIE nu ar fi fost bine utilizat.

Intre aceste instrucţii găsim următoarele tipuri:

- Instrucţii MACRO. S-a prevăzut posibilitatea codificării unui mare număr de macroinstrucţii.
- Instrucţia de oprire a calculatorului
- Instrucţii care se referă la depăşirea capacităţii registrelor. Acestea conţin şi două instrucţii de omitere în funcţie de starea bistabilului DCR.
- Instrucţii pentru generatoarele de cuvinte
- Instrucţii pentru pornirea şi oprirea perife-

RESTRICȚII DE MICROPROGRAMARE LA INSTRUCȚIILE *(GMO)

TABELUL 2.7

	1	2	3	4	5	6	7	8	9	10	11	12	13
	i_4	i_4	i_4	i_4	i_4	i_4	i_4	i_5	i_5	i_5	i_5	i_6	—
1	AZQ	AZB	AZA	G1 INQ	AZSR	AZMR	AZL	A IN B	OA+	OAP	—	OAZ	OSI
			RJP		CMR		CML						
		AUB					AUL	A IN Q					
			AD1SnA		ADSnR	ADMR	OZL	NA IN Q					
2	— " —							—	OA+	OAP	MUA	OAZ	OSI
3	— " —							B IN R	OA+	OAP	MUA	OAZ	OSI
								Q IN R					
								Q IN B					

↑ Nu se pot executa simultan căci sînt ocupate SCI

*) Instrucția CND nu este trecută în tabel deoarece ea se execută cînd avem microprogramată OSI și nu avem microprogramată nici o altă instr. de omitere.

Obs. Întrucît la efectuarea instrucțiilor CMR, RJP și MUA intervine adunarea unui unu de rcmps este necesar să acordăm timpul de cel puțin 2 microsecunde necesar ptr. efectuarea ei, deaceea nu vom microprograma după ele (cu i_5 sau i_6) instrucții care fac uz de rezultatul dat de cele menționate, deoarece el nu este disponibil. Astfel s. ex. după RJP nu se poate microprograma nici o instrucție legată de reg. A.

INSTRUCTIILE MICROPROGRAMATE DIN GRUPA DE MODIFICĂRI ŞI OMITERI (GMO)

TABELUL 2.8

Nr. crt.	DENUMIREA MNEMONICĂ A INSTR	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		1	1	0	0	1	COL 1		COL 2		COL 3		4	COL 5		6	COL 7		COL 8			9	10	11	12
1	AZA	1	1	0	0	1					0	1													
2	AZB	1	1	0	0	1			0	1															
3	AZQ	1	1	0	0	1	0	1																	
4	AZL	1	1	0	0	1											0	1							
5	G1 in Q	1	1	0	0	1	1	0																	
6	AZMR	1	1	0	0	1								0	1										
7	AZSR	1	1	0	0	1							1												
8	AUB	1	1	0	0	1			1	0															
9	AUL	1	1	0	0	1											1	0							
10	CML	1	1	0	0	1											1	1							
11	ADMR	1	1	0	0	1								1	0										
12	CMR	1	1	0	0	1								1	1										
13	RJP	1	1	0	0	1					1	0													
14	ADI SnA	1	1	0	0	1					1	1													
15	A IN B	1	1	0	0	1													0	0	1				
16	NA in Q	1	1	0	0	1													0	1	0				
17	B IN R	1	1	0	0	1													0	1	1				
18	Q IN R	1	1	0	0	1													1	0	0				
19	Q IN B	1	1	0	0	1													1	0	1				
20	A IN Q	1	1	0	0	1													1	1	0				
21	MUA	1	1	0	0	1																	1		
22	OLZ	1	1	0	0	1										1									
23	OA+	1	1	0	0	1													1	1	1				
24	OAP	1	1	0	0	1																1			
25	OAZ	1	1	0	0	1																		1	
26	OSI	1	1	0	0	1																			1
27	OND	1	1	0	0	1										0			OM=0			0		0	1
28	ADSnR	1	1	0	0	1	1	1																	

ricelor

- Instrucţii cu privire la fanioane. Aici avem trei instrucţii de omitere în funcţie de starea fanioanelor.

- Instrucţii de introducere şi extragere a unui caracter.

In fig.2.4 se arată formatul instrucţiilor IIE. In tabelul 2.9 este prezentată lista instrucţiilor.

In tabelul 2.lo se prezintă codificarea biţilor instrucţiei.

Descrierea în detaliu a instrucţiilor

Intr-o primă etapă descrierea instrucţiilor va fi făcută mai sumar în forma necesară utilizatorului, apoi după ce s-au făcut o seamă de precizări asupra blocurilor,din care este format calculatorul,se revine şi se face o descriere foarte în detaliu. Această a doua descriere corespunde necesităţilor constructorului.In această descriere,pentru constructor,ne vom referi la algoritmul utilizat şi îl vom dezvolta,acolo unde e cazul,şi ne vom referi deasemenea,la toate microoperaţiile din care este formată instrucţia.

In cele ce urmează,acolo unde o singură descriere scurtă serveşte atît utilizatorului cît şi constructorului,vom rămîne la aceasta. In descrierea instrucţiilor,la instrucţiile mai complicate,se va indica şi cîte un exemplu asupra modului de utilizare al instrucţiei.

Am fi putut utiliza o descriere a instrucţiilor prin organigrame; nu facem acest lucru deoarece limbajul curent oferă mai multe posibilităţi de descriere, iar pe de altă parte,vom utiliza,ulterior,descrierea prin tabele,pe baza cărora se pot scrie foarte uşor ecuaţiile logice necesare.

INSTRUCȚIUNILE DE ÎNTRODUCERE ȘI EXTRAGERE (IIE)

23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	1	1	*	CODUL INSTRUCȚIEI												CODUL DE SELECȚIE					

FIG 2.4

TABELUL 2.9

Nr. crt.	DENUMIREA MNEMONICĂ A INSTR.	CONȚINUTUL INSTRUCȚIEI
1	MAC	Macroinstrucție
2	OPR	Oprește calculatorul
3	AZDC	Adă la zero bistabilul DCR
4	AUDC	Adă la unu bistabilul DCR
5	ODCZ	Omite dacă bistabilul DCR e zero
6	ODCU	Omite dacă bistabilul DCR e unu
7	G1 IN R	Conținutul lui G1 se trece în R
8	G1 SAU R	Conținutul lui G1 se trece în R prin funcția SAU
9	G2 ÎN B	Conținutul lui G2 se trece în B
10	AZQ	Adă la zero conținutul registrului Q
11	PORP	Pornește dispozitivul periferic
12	OPRP	Oprește dispozitivul periferic
13	AZF	Adă la zero fanionul
14	OFA	Omite dacă fanionul A este la unu
15	OFB	Omite dacă fanionul B este la unu
16	INT	Introdu un caracter
17	EXT	Extrage un caracter
18	OTS	Omite la tensiune scăzută
19	AITR	Autorizează întreruperea
20	DITR	Dezautorizează întreruperea

INSTRUCȚIUNILE DE ÎNTRODUCERE ȘI EXTRAGERE (IIE)

TABELUL **2.1o**

Nr crt.	DENUMIREA MNEMONICĂ A INSTR.	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		1	1	0	1	1	*													CODUL SELECȚIEI					
1	MAC	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	Nu contează					
2	OPR	1	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0						
3	AZDC	1	1	0	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0						
4	AUDC	1	1	0	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0						
5	ODCZ	1	1	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0						
6	ODCU	1	1	0	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0						
7	G1 IN R	1	1	0	1	1	1	0	0	0	0	0	1	0	0	0	0	0	0						
8	G1 SAU R	1	1	0	1	1	1	0	0	0	0	0	0	1	0	0	0	0	0						
9	G2 IN B	1	1	0	1	1	1	0	0	0	0	0	0	0	1	0	0	0	0						
10	AZQ	1	1	0	1	1	1	0	0	0	0	0	0	0	0	1	0	0	0						
11	PORP	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	1	0	0						
12	OPRP	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0						
13	AZF	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1						
14	OFA	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1						
15	OFB	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	1	1	0						
16	INT	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	1	1	1						
17	EXT	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	1	0	1						
18	OTS	1	1	0	1	1	1	0	0	0	0	0	0	1	1	0	0	0	0	Nu contează					
19	AITR	1	1	0	1	1	1	0	0	0	0	0	0	0	1	1	0	0	0						
20	DITR	1	1	0	1	1	1	0	0	0	0	0	0	1	0	1	0	0	0						

* Dacă bitul 18 este zero avem o macroinstrucție, iardacă este unu avem o instrucție de introducere - extragere

I. INSTRUCTII CU REFERIRE LA MEMORIE

1. INB Incarcă registrul B. Registrul B este adus la zero, iar apoi este încărcat cu conţinutul locului adresat.Conţinutul celulei din memorie rămîne nemodificat.

2. INR Incarcă registrul R. Registrul R este adus la zero,iar apoi este încărcat cu conţinutul locului adresat.Conţinutul celulei din memorie rămîne nemodificat.

3. INQ Incarcă registrul Q cu înmulţitorul. Registrul Q este adus la zero, iar apoi este încărcat cu conţinutul locului adresat.Totodată semnul înmulţitorului este primit şi în S_nR. Conţinutul celulei din memorie rămîne nemodificat.

Obs. Numărul cu care este încărcat Q este de obicei înmulţitorul.Introducerea semnului şi în SnR se face în scopul pregătirii formării semnului produsului.

4. INA Incarcă registrul A în vederea adunării sau scăderii. Registrele R şi A sînt aduse la zero.Conţinutul locului adresat este încărcat în R, de unde,dacă semnul lui este pozitiv,se trimite nemodificat în A, iar dacă semnul lui e negativ se trimite complementat faţă de 2 în A. Conţinutul celulei din memorie rămîne nemodificat.

Obs. Timpul în care se execută instrucţia INA include şi timpul de adunare a lui 1, ($2\mu S$) pentru formarea complementului, în consecinţă e necesar să ţinem seama de acest lucru.Instrucţia poate fi folosită şi pentru memorarea restului la împărţire.

5. IDA Incarcă deîmpărţitul în registrul A. Registrele R şi A sînt aduse la zero.Conţinutul locului adresat e adus în R. Mărimea numărului din R (fără semn) se

predă nemodificată în A. Conţinutul celulei din memorie rămîne nemodificat.

6. MEQ Memorează conţinutul registrului Q.

Conţinutul registrului Q

este introdus în memorie (memorat) la locul adresat. Conţinutul precedent al celulei memoriei se pierde.

7. MG2 Memorează conţinutul generatorului de cuvinte G2. Conţinutul generatorului de cuvinte G2 este introdus în memorie (memorat) la locul adresat. Conţinutul precedent al celulei memoriei se pierde.

8. MEA Memorează conţinutul registrului A. Conţinutul registrului A este introdus de memorie (memorat) la locul adresat. Conţinutul precedent al celulei memoriei se pierde; conţinutul registrului A rămîne neschimbat.

9. MPQ Memorează produsul din Q (partea mai puţin semnificativă).Conţinutul registrelor A şi Q - în care după înmulţire se află produsul, cu partea mai semnificativă în A şi partea mai puţin semnificativă în Q - împreună se deplasează cu un rang la dreapta, după care conţinutul registrului Q este introdus în memorie (memorat) la locul adresat.Conţinutul precedent al celulei memoriei se pierde; conţinutul iniţial al registrului Q rămîne în registrul Q dar e deplasat cu un rang la dreapta.

Obs. Inainte de a efectua instrucţia MPQ trebuie memorată partea c.m.s. a produsului din A, deoarece cînd se efectuează instrucţia MPQ conţinutul registrului A se altereză cu ocazia deplasării spre dreapta,cînd se face şi o adunare a conţinutului registrului R la conţinutul registrului A.Alterarea se produce datorită faptului că pentru a avea legătură între A şi Q, spre a putea deplasa împreună conţinutul celor două registre,trebuie să avem semnal la

borna LEG ceace, cînd se dă impuls de deplasare la dreapta,permite şi efectuarea adunării conţinutului registrului R la conţinutul registrului A, dacă este îndeplinită condiţia ca în rangul c.m.p.s. a lui Q să se afle unu.

lo. MCQ Memorează cîtul din registrul Q. Conţinutul părţii de mărime a registrului Q şi a părţii de semn a registrului R - în care după împărţire se află partea de mărime respectiv partea de semn a cîtului - sînt introduse în memorie (se memorează) la locul adresat.Conţinutul precedent al celulei memoriei se pierde; conţinutul registrului Q şi R rămîne neschimbat.

11. MRA Memorează rezultatul adunării sau scăderii din registrul A. Registrul R se aduce la zero, apoi conţinutul registrului A se introduce în R, după care registrul A se aduce la zero.Acum conţinutul registrului R,dacă semnul lui este pozitiv,se trimite nemodificat în A, iar dacă semnul lui e negativ,se trimite complementat faţă de 2 în A. Conţinutul părţii de mărime a registrului A şi conţinutul părţii de semn a registrului R sînt introduse în memorie (se memorează) la locul adresat.Conţinutul precedent al celulei memoriei se pierde; **conţinutul ultim al registrului A şi R rămîne neschimbat.**

Obs. Prin această instrucţie conţinutul registrului A este complementat faţă de 2 pentru a fi introdus în memorie în reprezentarea semn-mărime, adică în reprezentarea în care se păstrează în memorie. Timpul în care se execută instrucţia include şi cele 2 microsecunde necesare adunării lui unu de rangul c.m.p.s.,cînd se face complementarea faţă de doi.

12. ADA Adună la conţinutul registruluiA. Regis-

trul R este adus la zero.Conţinutul locului adresat este încărcat în R,de unde,dacă semnul lui este pozitiv,se trimite nemodificat în A, iar dacă semnul lui este negativ,se trimite complementat faţă de doi. Se verifică apoi depăşirea capacităţii registrelor.Dacă a apărut DCR se aduce la unu bistabilul DCR şi se aprinde becul DCR pe pupitrul de comandă.Rezultatul adunării se află în registrul A.Conţinutul celulei din memorie rămîne nemodificat.

Obs. Timpul în care se efectuează instrucţia ADA include şi timpul maxim de adunare de două microsecunde.

13. SCA Scade din conţinutul registrului A.Partea de mărime a registrului R se aduce la zero, iar partea de semn se aduce la unu.Conţinutul locului adresat este încărcat în R, de unde, dacă semnul lui (conţinutul lui SnR) este pozitiv, se trimite nemodificat în A, iar dacă semnul lui e negativ,se trimite complementat faţă de doi în A. Se verifică apoi depăşirea capacităţii registrelor.Dacă a apărut DCR se aduce la unu bistabilul DCR şi se aprinde becul DCR de pe pupitrul de comandă.Rezultatul scăderii se află în registrul A. Conţinutul celulei din memorie rămîne nemodificat.

Obs.-Timpul în care se efectuează instrucţia SCA include şi timpul maxim de adunare de 2 microsecunde.

- Se observă că singura modificare în instrucţia de scădere faţă de cea de adunare este aducerea la început,a părţii semnului din R la unu ceace provoacă apoi, cînd se încarcă R, schimbarea semnului numărului adus din memorie în R.

14. INM Inmulţeşte. Inainte de efectuarea in-

structiei INM se aduce,cu ajutorul instrucţiei INQ, înmulţitorul în registrul Q. Prin instrucţia INM se aduce conţinutul locului adresat, care este deînmulţitul, în registrul R. Se face înmulţirea numerelor din registrele Q şi R în reprezentare semn mărime.Rezultatul obţinut se află în registrul A cu partea cea mai semnificativă şi în registrul Q cu partea cea mai puţin semnificativă.Conţinutul registrului R se pierde.Conţinutul celulei din memorie rămîne nemodificat.

Registrul R este adus la zero.Conţinutul loçului adresat,care este deînmulţitul,este încărcat în R. Se deplasează spre dreapta cu un rang registrul A şi Q după care, dacă cifra cu care se înmulţeşte a înmulţitorului este unu se adună deînmulţitul, iardacă cifra înmulţitorului este zero se adună zero. Totodată se introduce unitatea în numărătorul paşilor. Se efectueazăoperaţia de deplasare şi adunare de 24 ori. Semnul din partea de semn a registrului R se trece în partea de semn a registrului A. Acum semnul produsului se află în partea de semn a registrului A, iar mărimea produsului, cu partea cea mai semnificativă, în partea de mărime a registrului A, pe cînd partea mai puţin semnificativă a produşului se găseşte în partea de mărime a registrului Q. La începutul înmulţirii în registrul Q se află înmulţitorul, încărcat cu ajutorul instrucţiei INQ. Conţinutul celulei din memorie rămîne neschimbat. Vezi algoritmul la descrierea dispozitivului aritmetic.

15. IMP Imparte. Inainte de efectuarea instrucţiei IMP se aduce,cu ajutorul instrucţiei IDA, deîmpărţitul în registrul A.Prin instrucţia IMP se aduce conţinutul locului adresat, care este împărţitorul în

registrul R. Se verifică dacă împărţitorul este sau nu mai mare ca deîmpărţitul.In cazul cînd împărţitorul este mai mic sau egal cu deîmpărţitul se aduce la unu bistabilul DCR, se reface deîmpărţitul,după care împărţirea se opreşte şi se trece la instrucţia următoare.Aceasta poate fi "omite dacă DCR e zero", urmată la rîndul său de instrucţia SSD (salt la subprogram).Dacă împărţitorul este mai mare decît deîmpărţitul se efectuează împărţirea, în reprezentare semn-mărime, între deîmpărţitul din A şi împărţitul din R. La sfîrşit partea de mărime a cîtului se află în Q, iar partea de semn în partea de semn a registrului R. Conţinutul registrului R se pierde.Conţinutul celulei din memorie rămîne nemodificat.

Partea mărimii registrului R se aduce la zero (în partea de semn se află semnul deîmpărţitului introdus prin instrucţia IDA plasată în program înaintea instrucţiei IMP). Conţinutul locului adresat,care este împărţitorul, este încărcat în R. Se verifică prin scădere dacă deîmpărţitul este egal sau mai mare ca împărţitorul. In cazul că este egal sau mai mare se aduce bistabilul DCR la unu, se reface deîmpărţitul, după care împărţirea se opreşte şi se trece la instrucţia următoare. Aceasta poate fi "omite dacă DCR e zero", urmată la rîndul său de instrucţia "salt la subprogram".Subprogramul la care se face saltul se ocupă de DCR. Subprogramul poate să tipărească indicarea unei erori, să acţioneze asupra împărţitorului sau deîmpărţitorului, sau să execute alte corecţii matematice şi să refacă împărţirea). In cazul cînd deîmpărţitul este mai mic decît împărţitorul se continuă împărţirea.Prima cifră a cîtului

este zero şi nu se introduce. Se deplasează cu un rang spre stînga conţinutului registrului A şi al registrul Q; se introduce unu în numărătorul paşilor; se adună sau scade, în funcţie de semnul restului deplasat, împărţitorul aflat în R la conţinutul registrului A; se introduce în Q cifra cîtului. Microoperaţiile începînd cu deplasarea cu un rang spre stînga a registrului A şi Q se execută încă de 23 ori. După acestea cîtul se găseşte în Q, iar restul înmulţit cu 2^{24} se află în A. Conţinutul celulei din memorie rămîne nemodificat. Vezi algoritmul la descrierea dispozitivului aritmetic.

Descrierea schemei logice pentru repetarea impulsurilor la INM şi IMP

După cum se va vedea mai tîrziu la descrierea prin tabele a instrucţiilor, în cazul instrucţiei INM se vor executa de 24 ori, în faza EXECUTIE 1 (Fz.3), impulsurile i_4-i_6 împreună cu microoperaţiile respective, iar în cazul instrucţiei IMP se vor executa de 24 ori, în faza EXECUTIE 2 (Fz.4), impulsurile i_0-i_3 împreună cu microoperaţiile respective. In acest scop va fi nevoie de un circuit bistabil pentru "înmulţire şi împărţire" notat cu BII şi de un numărător N 24, care numără pînă la 24. Circuitul BII va fi adus la unu în cazul INM la Fz3.i_6 şi cînd încă N24 nu a ajuns în starea 24, adică avem $\overline{N24}$=1. Primul impuls de tact care apare după aducerea lui BII în starea unu şi pe care îl notăm cu i, va fi trimis la A(4)N8 în loc de la +1N8. Tot acest impuls va aduce pe BII la zero. In felul acesta N8 reia numărarea dela i_4 şi astfel cînd N24 a numărat 24 impulsuri tactele i_4-i_6 sînt executate de 24 ori. In mod asemănător se petrec lucrurile şi cînd avem instrucţia

IMP cu deosebirea că BII este adus acum la unu de i_3, iar primul impuls de tact care apare, i, aduce pe N8 în starea zero corespunzător impulsului i_0. Astfel se execută de 24 ori tactele i_0-i_3 în timpul lui F_{z4}.

Se vor prevedea deasemenea circuite care stabilesc starea inițială a tuturor elementelor schemei.

Se dau mai jos ecuațiile logice care corespund descrierii de mai sus.

1. $A(1)BII = INM.\overline{N24}.Fz3.i_6 + IMP.\overline{N24}.Fz4.i_3$
2. $A(o)BII = BII.i + i_{ST\ INIT}$
3. $A(4)N8 = INM.BII.i$
4. $A(o)N8 = IMP.BII.i + i_{ST\ INIT}$
5. $+1N8 = \overline{BII}.i$

Impulsul de stare inițială $i_{ST\ INIT}$ va fi dat de o cheie de pe pupitrul de comandă, care se acționează manual.

16. SAI SAU inclusiv cu conținutul registrului A. Registrul R este adus la zero. Conținutul registrului A este dus în R. Se primește în R conținutul locului adresat. In R se formează astfel funcția SAU inclusiv între cele două numere. Simultan se aduce la zero A. Conținutul registrului R care este funcția SAU inclusiv între biții conținutului inițial al registrului A și ai conținutul locului adresat se duce în A. Conținutul registrului A și R se pierde, iar conținutul locului adresat rămîne neschimbat.

17. SAE SAU exclusiv cu conținutul registrului A.

Registrul R este adus la zero. Conținutul locului adresat este încărcat în R. Conținutul registrului R este trimis nemodificat în registrul A, la care se blochează simultan propagarea transferului. In registrul

A se formează SAU EXCLUSIV între biţii corespunzători a celor două numere.Conţinutul registrului A şi R se pierde.Conţinutul celulei din memorie rămîne nemodificat.

Obs. Adunînd două numere X respectiv Y fără să permitem propagarea transferului obţinem X $\overline{\vee}$ Y.

Obs. Trimiterea conţinutului nemodificat al registrului R în A se face prin impulsuri la bornele -MR şi +SnR. In timpul instrucţiei SEA cele două bistabile Sn 1 A şi Sn 2 A au totdeauna acelaşi conţinut.

18. ŞIA Si cu conţinutul registrului A. Registrele R şi Q se aduc la zero. Conţinutul registrului A se trimite în Q. Conţinutul locului adresat se aduce simultan în R şi Q. Conţinutul lui R se trimite nemodificat în registrul A, la care se blochează simultan propagarea transferului.Registrul R se aduce la zero.Conţinutul registrului Q se trimite în R.Conţinutul registrului R se trimite nemodificat în A, la care se blochează simultan propagarea transferului. Acum în A se află rezultatul operaţiei SI între biţii corespunzători numărului aflat iniţial în registrul A, şi numărului aflat la locul adresat.Conţinutul registrelor A şi R se pierde.Conţinutul celulei din memorie rămîne nemodificat.

Obs. Operaţia SI între biţii a două numere X şi Y se face pe baza algoritmului:

$$X \,.\, Y = (X \overline{\vee} Y) \oplus (X \vee Y)$$

care este demonstrat în tabelul de mai jos.

X	Y	$X\bar{V}Y$	XVY	$(X\bar{V}Y) \oplus (XVY)$
o	o	o	o	o
o	1	1	1	o
1	o	1	1	o
1	1	o	1	1

Acest algoritm elaborat de autor nu cere adunări cu propagarea transferului,prin urmare este rapid.

19. SLT Salt. Conţinutul părţii de adresă a registrului RI este dus în registrul NA. Datorită acestui fapt instrucţia următoare va fi adusă din memorie, din celula a•cărei adresă a fost înscrisă în partea de adresă a instrucţiei,adresă care se află acum în NA. Conţinutul iniţial al registrului NA se pierde. **Conţinutul registrului A nu este modificat în timpul executării acestei instrucţiuni.**

2o SSP Salt la subprogram. Conţinutul registrului NA este mărit cu unu, se formează P+1 (P este adresa la care se află instrucţia SSP în programul principal), apoi P+1 este memorat la adresa Y (Y este adresa specifică în partea de adresă a instrucţiei SSP).Se aduce la zero NA.Conţinutul părţii de adresă a registrului RI (care este Y) se trimite în NA. Se măreşte cu unu conţinutul registrului NA şi se formează astfel Y+1. Conţinutul registrului A nu este modificat în timpul acestei instrucţii.

Obs. La sfîrşitul instrucţiei,la adresa Y în memorie,se află adresa P+1 **a instrucţiei din programul** principal la care se revine cînd s-a terminat subprogramul, iar în NA se află adresa Y+1 la care se începe efectiv subprogramul,adică instrucţia care urmează

imediat după instrucţia SSP.

Prin urmare această instrucţie "memorează" adresa instrucţiei din programul principal la care trebuie să se revină după terminarea subprogramului.Această revenire se face printr-o instrucţie de salt (SLT) cu adresare indirectă la adresa Y a subprogramului.In consecinţă instrucţia care urmează după SLT este acea din programul principal aflată la adresa P+1.

21. MOZ Măreşte cu unu şi omite dacă e zero.Prin această instrucţie se adună unu la conţinutul locului adresat din memorie.Dacă prin aceasta se obţine rezultat zero instrucţia următoare este omisă.Dacă rezultatul nu e zero următoarea instrucţie care se execută este acea care urmează normal în program. Valoarea mărită cu unu este dusă în fiecare caz înapoi în celula din memorie de unde a fost luată valoarea înainte de a fi mărită.

Registrul B se aduce la zero.Conţinutul lui A se duce în B. Registrele R şi A se aduc la zero.Conţinutul locului adresat este încărcat în R,de unde, se trimite în A, nemodificat dacă e pozitiv, sau complementat faţă de doi dacă e negativ.

Se adună un unu de rangul c.m.p.s.,prin aducerea unui impuls la +1A.

Dacă ZA=1, înseamnă că conţinutul lui A a devenit zero şi se adaugă unu la NA (ceea ce înseamnă că se omite instrucţia următoare), dacă ZA=0, înseamnă că conţinutul lui A nu e zero şi conţinutul lui NA se lasă neschimbat.

Acum se memorează rezultatul adunării la aceiaşi adresă,procedînd identic ca la instrucţia MRA adică transformînd conţinutul lui A, din reprezentarea în

complement de doi în reprezentarea semn-mărime,după care se duce în memorie.Se procedează în felul următor. Registrul R se aduce la zero,după care conţinutul registrului A se introduce în R. Se aduce apoi la zero registrul A. Acum conţinutul registrului R, dacă semnul lui este pozitiv,se trimite nemodificat în A,iar dacă semnul lui e negativ, se trimite complementat faţă de doi în A. Conţinutul părţii de mărime a registrului A şi conţinutul părţii de semn a registrului R sînt introduse în memorie la locul adresat.In celula de memorie apare conţinutul precedent mărit cu unu. Numărul care a fost iniţial în registrul A şi care se află acum în registrul B este readus prin registrul R înapoi în registrul A.

Obs. Instrucţia MOZ este utilizată pentru numărarea unor operaţii şi oprirea lor după ce au fost efectuate de un anumit număr de ori.Această proprietate din urmă se datoreşte faptului că,pe lîngă mărirea cu unu,instrucţia permite ieşirea din bucla de program, prin posibilitatea ei de a provoca omiterea,cînd rezultatul măririi cu unu duce la rezultatul zero.Foarte comod se utilizează instrucţia în felul următor. Intr-o anumită celulă din memorie numită acum"contor" se introduce un număr negativ,care în valoare absolută arată de cîte ori trebuie făcută o anumită operaţie.Prin mărirea cu unu la fiecare operaţie a conţinutului contorului se ajunge ca,atunci cînd numărul cerut de operaţii s-a terminat,rezultatul măririi cu unu să fie zero,ceace provoacă omiterea şi ieşirea din bucla de program.

Spre exemplu dorim să adunăm la conţinutul acumulatorului numărul 12 consecutiv de patru ori.

Putem realiza aceasta cu ajutorul următoarei secvenţe:

```
1  ADA  5
2  MOZ  6
3  SLT  1
4  MRA  7
5  12
6  -4
7  Rezultatul
```

Instrucţia MOZ poate fi utilizată şi la modificarea adreselor.Cum în acest caz prin mărirea ca unu numărul creşte în valoare absolută,proprietatea de omitere la zero nu poate fi folosită şi avem nevoie de o altă instrucţie care să permită ieşirea din bucla.Aceasta va fi CBO.

22. CBO Compară cu conţinutul registrului B şi omite la inegalitate. Prin această instrucţie se compară conţinutul locului adresat din memorie cu conţinutul registrului B.Dacă cele două cuvinte de cîte 24 biţi sînt diferite,instrucţia următoare este omisă.Dacă cele două cuvinte sînt identice următoarea instrucţie care se execută este aceea care urmează normal în program.La sfîrşitul executării instrucţiei conţinutul registrului A, a registrului B şi a locului adresat din memorie rămîn nemodificate.

Registrele R şi Q sînt aduse la zero.Conţinutul lui A este trecut în Q (pentru a nu se pierde),se aduce la zero A. Conţinutul locului adresat este încărcat în R, iar de aici este dus **nemodificat** în A. Se aduce la zero R, apoi se trece în el conţinutul lui B. Conţinutul registrului R este trimis nemodificat în registrul A, la care, simultan, se blochează propagarea transferului.

Dacă în A nu avem zero (ZA=0) înseamnă că numerele au fost inegale şi se măreşte conţinutul lui NA cu unu,ceace înseamnă că instrucţia următoare va fi omisă.Dacă în A avem zero (ZA=1) înseamnă că numere au fost egale şi conţinutul lui NA se lasă nemodificat.

Se aduce R şi A la zero şi se trimite, prin R, conţinutul lui Q în A. In felul acesta se reface conţinutul iniţial al lui A.

Obs. Semnalul ZA (zero în A) se obţine la ieşirea unor circuite logice SI, care cuprind toate rangurile registrului A. Semnalul ZA este unu cînd conţinutul lui A este zero.

Obs. Instrucţia CBO este utilizată în programe în care e necesară modificarea unor adrese,modificare care se face cu ajutorul instrucţiei MOZ, iar CBO serveşte la ieşirea din bucla programului.Dacă numărul din locaţia din memorie care serveşte drept "contor" (vezi instr.MOZ) este pozitiv (reprezintă o adresă) la mărirea cu unu prin instr.MOZ nu se merge spre zero deci omiterea la zero din MOZ nu e utilă.In acest caz,dacă spre exemplu dorim să adunăm un şir de numere aflate la adresele 26-3o, introducem adresa limită (3o) în registrul B şi formăm următoarea secvenţă de instrucţii:

```
1  INB     8
2  ADA  I  7
3  CBO     7
4  SLT  Adresă care face ieşirea din buclă
5  MOZ     7
6  SLT     2
7  Rezervată pentru adresa numărului (la început conţine 26)
8  3o
```

Se vede din secvenţă că verificarea ajungerii la adresa 3o cu CBO se face înainte de a face o nouă mărire cu unu prin MOZ, ceace înseamnă că ieşim din buclă cînd la adresa 7 avem 3o. Secvenţa de mai sus justifică motivul pentru care la instrucţia CBO omiterea se face la inegalitate şi nu la egalitate.

II. INSTRUCTIUNI CU REFERIRE LA REGISTRE

A. Grupul de deplasări - rotiri (GDR)

23. NOP Nici o operaţie. Nu se efectuează nici o operaţie afară de adăugarea unui unu la NA pentru formarea adresei instrucţiei următoare.

Un exemplu de utilizare a acestei instrucţii este următorul: Dacă dorim ca efectul de omitere din instrucţia "măreşte cu unu şi omite dacă e zero" (MOZ) să fie eliminat, este suficient să întroducem în program, imediat după instrucţia MOZ, instrucţia NoP. Este evident că se obţine efectul dorit. Printre altele instrucţia NoP poate fi utilizată şi atunci cînd dorim să obţinem o întîrziere în desfăşurarea programului.

24. AZL Adă la zero conţinutul registrului L. Se aduce la zero conţinutul bistabilului de legătura L care este identic cu Sn2A, prin aducerea unui impuls la borna A(o)Sn2A.

25. OAP Omite dacă conţinutul lui A e par. In cazul cînd rangul cel mai puţin semnificativ (OA) al registrului A este zero, adică numărul aflat în A este par, se omite instrucţia următoare; în caz contrar instrucţia la care trece calculatorul este aceea care urmează imediat în program.

26. OQP Omite dacă conţinutul lui Q este par. In cazul cînd rangul cel mai puţin semnificativ (OQ) al registrului Q este zero, adică numărul aflat în Q este par, se omite instrucţia următoare; în caz contrar instrucţia la care trece calculatorul este aceea care urmează imediat în program.

27. DSA Deplasează aritmetic la stînga cu un rang conţinutul registrului A. Se trimite un impuls la borna St A. Prin aceasta mărimea pătrunde în ran-

gul semnului Sn1A, iar conţinutul din Sn1A pătrunde în Sn2A. Cifrele noi ce pătrund prin dreapta sînt zerouri şi pătrund în RgCA.Conţinutul lui Sn2A se pierde.

28. DSQ Deplasează la stînga cu un rang conţinutul registrului Q. Se trimite un impuls la borna StQ. Prin aceasta mărimea pătrunde în rangul semnului SnQ. Cifrele noi ce pătrund prin dreapta sînt zerouri şi pătrund în RgCQ. Conţinutul lui SnQ se pierde.

29. DDA Deplasează aritmetic la dreapta cu un rang conţinutul registrului A. Se trimite un impuls la borna Dr.A. Prin aceasta partea de semn pătrunde în partea de mărime.Partea de semn rămîne neschimbată.Deci pentru semn pozitiv (zero) pătrund zerouri în partea de mărime, iar pentru semn negativ (unu) pătrund cifre de unu.Conţinutul rangului 0A pătrunde în RgCA. Conţinutul lui RgCA se pierde.

3o. DDQ Deplasează la dreapta cu un rang conţinutul registrului Q. Se trimite un impuls la borna DrQ şi simultan semnal de potenţial la borna DrTQ (dreapta tot Q).Prin aceasta toţi biţii se deplasează spre dreapta.Bitul 0Q pătrunde în RgCQ.Conţinutul acestuia din.urmă se pierde.In SnQ intră zero dacă la borna "Pm prin stînga" era zero.Dacă această bornă se află la unu în momentul apariţiei impulsului DrQ, în SnQ pătrunde unu.

31. RSA Roteşte la stînga cu un rang conţinutul registrului A+L. Se trimite un semnal de potenţial la borna Rot StA şi în timpul acestuia se trimite un impuls la borna StA.Prin aceasta conţinutul părţii de mărime pătrunde în Sn1A, iar conţinutul acestuia în Sn2A (L).Conţinutul lui Sn2A(L) pătrunde în RgCA, iar conţinutul lui RgCA pătrunde în 0A ş.a.m.d.

32. RSQ Roteşte la stînga cu un rang conţinutul registrului Q. Se trimite un semnal de potenţial la borna Rot StQ şi în timpul acestuia se trimite un impuls la borna StQ.Prin aceasta conţinutul părţii de mărime pătrunde în SnQ. Conţinutul lui SnQ pătrunde în RgCQ, iar conţinutul lui RgCQ pătrunde în OQ ş.a. m.d.

33. RDA Roteşte la dreapta cu un rang conţinutul registrului A+L. Se trimite un semnal de potenţial la borna Rot DrA. In timpul acestuia se trimite un impuls la borna DrA. Prin acestea conţinutul lui Sn1A pătrunde în partea de mărime,iar a lui Sn2A în Sn1A.Conţinutul lui OA pătrunde în RgCA, iar conţinutul acestuia din urmă pătrunde în Sn2A ş.a.m.d.

34. RDQ Roteşte la dreapta cu un rang conţinutul registrului Q. Se trimite cîte un semnal de potenţial la bornele "Rot DrQ" şi DrTQ. In timpul acestora se trimite un impuls la borna DrQ.Prin acestea conţinutul semnului pătrunde în partea de mărime,conţinutul lui OQ pătrunde în RgCQ, iar conţinutul acestuia din urmă pătrunde în SnQ ş.a.m.d.

Obs. In timpul acestei instrucţii borna "Pm prin stga" trebuie să se afle în starea zero.

35. RSA2 Roteşte la stînga cu două ranguri conţinutul registrului A+L. Se trimite de două ori la rînd semnal de potenţial la borna "Rot StA" şi tot odată cîte un impuls la borna StA. Prin aceasta se efectuează de două ori rotirea care a fost descrisă la RSA.

36. RSQ2 Roteşte la stînga cu două ranguri conţinutul registrului Q. Se trimite de două ori la rînd semnal de potenţial la borna "Rot StQ" şi tot odată cîte un impuls la borna StQ. Prin aceasta se efectuea-

ză de două ori la rînd rotirea care a fost descrisă la instrucţia RSQ.

B.Grupul pentru modificări şi omiteri (GMO)

37. AZA Adă la zero conţinutul registrului A.Se trimite impuls la borna A(o)A.

38. AZB Adă la zero conţinutul registrului B.Se trimite impuls la borna A(o)B.

39. AZQ Adă la zero conţinutul registrului Q.Se trimite impuls la borna A(o)Q.

40. AZL Adă la zero conţinutul registrului L.Se trimite impuls la borna A(o)Sn2A.

41. Gl IN Q Conţinutul lui Gl se trece în Q . Se trimite impuls la borna A(o)RM.

42. AZMR Adă la zero partea de mărime a conţinutului registrului R. Se trimite impuls la borna A(o)MR.

43. AZSR Adă la zero semnul din registrul R. Se trimite impuls la borna A(o)SnR.

44. AUB Adă la unu conţinutul registrului B. Se trimite impuls la borna A(1)B.

45. AUL Adă la unu conţinutul registrului L. Se trimite impuls la borna A(1)Sn2A.

46. CML Complementează conţinutul registrului L. Se trimite impuls la borna: Com.Sn2A.

47. ADMR Adună partea de mărime a numărului din registrul R la conţinutul registrului A. Se trimite un impuls la borna +MR.

48. CMR Complementează partea de mărime a conţinutului registrului R faţă de doi în A. Se trimite un impuls la borna -MR.

49. RJP Rotunjeşte produsul sau "adă la unu rangul complementar al registrului A (AURgCA)".Se trimite impuls la borna +1RgCA.

ză de două ori la rînd rotirea care a fost descrisă la instrucţia RSQ.

B.Grupul pentru modificări şi omiteri (GMO)

37. AZA Adă la zero conţinutul registrului A.Se trimite impuls la borna A(o)A.

38. AZB Adă la zero conţinutul registrului B.Se trimite impuls la borna A(o)B.

39. AZQ Adă la zero conţinutul registrului Q.Se trimite impuls la borna A(o)Q.

4o. AZL Adă la zero conţinutul registrului L.Se trimite impuls la borna A(o)Sn2A.

41. Gl IN Q Conţinutul lui Gl se trece în Q . Se trimite impuls la borna A(o)RM.

42. AZMR Adă la zero partea de mărime a conţinutului registrului R. Se trimite impuls la borna A(o)MR.

43. AZSR Adă la zero semnul din registrul R. Se trimite impuls la borna A(o)SnR.

44. AUB Adă la unu conţinutul registrului B. Se trimite impuls la borna A(1)B.

45. AUL Adă la unu conţinutul registrului L. Se trimite impuls la borna A(1)Sn2A.

46. CML Complementează conţinutul registrului L. Se trimite impuls la borna: Com.Sn2A.

47. ADMR Adună partea de mărime a numărului din registrul R la conţinutul registrului A. Se trimite un impuls la borna +MR.

48. CMR Complementează partea de mărime a conţinutului registrului R faţă de doi în A. Se trimite un impuls la borna -MR.

49. RJP Rotunjeşte produsul sau "adă la unu rangul complementar al registrului A (AURgCA)".Se trimite impuls la borna +1RgCA.

obs. de la nr.51).

57. MUA Măreşte cu unu conţinutul registrului A.
Se trimite impuls la borna +1A.

Obs. La efectuarea acestei instrucţii poate apărea depăşirea capacităţii registrului, prin urmare în program se va prevedea, după ea, o instituţie pentru verificarea depăşirii.

58. ADSnR Adună semnul registrului R la A.
Se trimite impuls la borna +SnR.

Obs. Această instrucţie împreună cu ADMR serveşte la introducerea unui număr nemodificat din R în A.

In continuare se vor descrie instrucţiile de omitere din GMO. Avînd în vedere că între instrucţiile de omitere din această grupă am prevăzut legături mai complexe vom trata pe larg algoritmii utilizaţi şi vom scrie relaţiile logice necesare. Datorită acestui fapt va trebui să precizăm şi valoarea biţilor din registrul RI cu ajutorul cărora se microprogramează executarea acestor instrucţii.

Descrierea instrucţiilor de omitere din GMO

Modul de efectuare al instrucţiilor din GMO depinde de conţinutul rangului zero (BO) din registrul RI. Cînd BO=0 înseamnă că omiterea se va face la sensul direct, iar dacă BO=1 omiterea se va face luînd condiţiile de omitere în sens invers (avem -OSI). La descrierea fiecărei instrucţii de omitere din GMO vom arăta particularităţile corespunzătoare.

59. OLZ Omite instrucţia următoare dacă conţinutul bistabilului L este zero

Instrucţia se efectuează la impuls de tact i_4. Executarea ei este influenţată de faptul dacă avem sau nu OSI.

Dacă executăm grupul de instrucţii, modificări şi omiteri (GMO=1) şi sîntem în tactul 4 (i_4=1) şi avem microprogramată instrucţia OLZ (în RI avem B9=1) şi nu avem OSI (în RI avem BO=0) şi conţinutul lui L este zero (L=0), atunci se aduce bistabilul de omitere la unu (A(1)BOM=1).

Sau dacă situaţia este ca mai sus dar avem OSI (în RI avem BO=1) şi conţinutul lui L este unu (L=1) atunci se aduce BOM la unu (A(1)BOM=1).

Alte instrucţii din GMO pot modifica starea bistabilului L dar testarea conţinutului lui în vederea executării instrucţiei OLZ este terminată înainte de modificare.

Cele de mai sus pot fi cuprinse în următorul tabel de combinaţii.

GMO	i_4	M9 (OLZ)	BO (OSI)	L	A(1)BOM
1	1	1	o	o	1
1	1	1	1	1	1

Din tabel se poate scrie ecuaţia.

$$(\bar{L}.\bar{BO}+L.BO).i_4.GMO.B9 = A(1)BOM \qquad (1)$$

60. OA+ Omite instrucţia următoare dacă bitul de semn (SnlA) al registrului A este zero; adică omite dacă conţinutul lui A este pozitiv.

Instrucţia se efectuează la impulsul de tact i_5. Executarea ei este influenţată de faptul dacă avem sau nu OSI.

Dacă executăm grupul de instrucţii modificări şi omiteri (GMO=1) şi sîntem în tactul 5(i_5=1) şi avem microprogramată instrucţia OA+ (în RI avem la descifratorul biţilor 4,5 şi 6 ieş. OM=1) şi nu avem OSI (în RI avem BO=0) şi în bistabilul semnului registrului A avem zero (SnA=0), atunci se aduce bistabilul de omitere la unu (A(1)BOM=1).

Sau dacă situaţia este ca mai sus dar avem OSI

(în RI avem BO=1) şi în bistabilul semnului registrului A avem unu (SnlA=1) şi instrucţia OAP, care se execută tot la i_5, nu este microprogramată (în RI avem B3=o) atunci se aduce BOM la unu (A(1)BOM=1).

Obs. Necesitatea ca OAP să nu fie microprogramată (B3=o) atunci se aduce BOM la unu (A(1)BOM=1).

Obs. Necesitatea ca OAP să nu fie microprogramată (B3=o), cînd avem OSI, rezultă dintr-o particularitate, ce va fi lămurită mai tîrziu, a microprogramării simultane a instrucţiilor OA+ şi OAP, care ambele se efectuează la **impulsul** de sincronizare i_5.

Cele de mai sus pot fi cuprinse în următorul tabel de combinaţii.

GMO	i_5	BO (OSI)	OM (OA+)	SnA	B3 (OAP)	OA	A(1)BOM
1	1	o	1	o	-	-	1
1	1	1	1	1	o	-	1

Din tabel se poate scrie ecuaţia:

$$(\overline{BO}.\overline{SnA}+BO.SnA.\overline{B_3}).i_5.GMO.OM = A(1)BOM \qquad (2')$$

61. OAP Omite instrucţia următoare dacă bitul cel mai puţin semnificativ (BO) al registrului A este zero; adică omite dacă în A avem număr par. Instrucţia se efectuează la impulsul de tact i_5. Executarea ei este influenţată de faptul dacă avem sau nu OSI.

Dacă executăm grupul de instr. modificări şi emiteri (GMO=1) şi sîntem în tactul 5(i_5=1) şi avem microprogramată instrucţia OAP (în RI avem B3=1) şi nu avem OSI (în RI avem BO=o) şi în bistabilul rangului cel mai puţin semnificativ al registrului A avem zero(OA=0),

atunci se aduce bistabilul de omitere la unu (A(1)BOM=1).

Sau dacă situaţia este ca mai sus dar avem OSI (în RI avem BO=1) şi în bistabilul rangului cel mai puţin semnificativ al registrului A avem unu (OA=1)şi instrucţia OA+, care se execută tot la i_5, nu este microprogramată (în RI avem la descifratorul biţilor ieşirea OM=0), atunci se aduce BOM la unu (A(1)BOM=1).

Obs. Necesitatea ca OA+ să nu fie microprogramată (în RI avem OM=0) cînd avem OSI rezultă dintr-o particularitate,ce va fi lămurită mai tîrziu,a microprogramării simultane a instrucţiilor OA+ şi OAP,care ambele se efectuează la impulsul de sincronizare i_5.

Cele de mai sus pot fi cuprinse în următorul tabel de combinaţii.

GMO	i_5	BO (OSI)	OM (OA+)	SnA	B3 (oAP)	OA	A(1)BOM
1	1	0	-	-	1	0	1
1	1	1	0	-	1	1	1

Din tabel se poate scrie ecuaţia:

$$(BO\cdot\overline{OA}+BO\cdot\overline{OA}\cdot OM\cdot i_5\cdot GMO\cdot B3 = A(1)BOM \qquad (2'')$$

Condiţie specială privind efectuarea simultană a instrucţiilor OA+ şi OAP, care uşurează efectuarea unor programe.

Enunţul condiţiei: Dacă microprogramăm simultan OA+ şi OAP şi avem OSI atunci omiterea apare numai dacă în bistabilul semnului şi în ultimul bistabil al registrului A avem unu.

Astfel dacă executăm grupul de instrucţii modificări şi omiteri (GMO=1) şi sîntem în tactul 5(i_5=1) şi avem OSI (în RI avem BO=1) şi avem microprogramate

instrucţia OA+ (în RI avem OM=1) şi instrucţia OAP (în RI avem B3=1) şi în bistabilul semnului registrului A avem unu (SnlA=1) şi în bistabilul rangului cel mai puţin semnificativ al registrului A avem unu (OA=1), atunci se aduce BOM la unu (A(1)BOM=1).

Aceasta înseamnă că omiterea are loc cînd numărul din A este negativ şi se termină cu unu (e impar).

Prin formularea acestei condiţii speciale se explică observaţiile făcute la descrierea instrucţiunilor OA+ şi OAP.

Cele de mai sus pot fi cuprinse în următorul tabel de combinaţii:

GMO	i_5	BO (OSI)	OM (OA+)	SnA	B3 (OAP)	OA	A(1)BOM
1	1	1	1	1	1	1	1

Din tabel se poate scrie ecuaţia:

$$SnA.OA.BO.B3.OM.GMO.i_5 = A(1)BOM \quad (2''')$$

Unificăm relaţiile (2')(2") şi (2"'),care se execută toate la **impulsul de sincronizare i_5 şi grupînd** după BO şi $\overline{BO}$ avem:

$$[(SnA.OM+OA.B3).i_5.BO+(SnA.OM.B3+OA.OM.B3+SnA.OA.OM.B3).i_5.\overline{BO}].\ GMO = A(1)BOM \quad (2)$$

62. OAZ Omite instrucţia următoare dacă în registrul A avem zero, adică fiecare bit este zero. Instrucţia se efectuează la impulsul de tact i_6.Executarea ei este influenţată de faptul dacă avem sau nu OSI.

Dacă executăm grupul de instrucţiuni modificări şi omiteri (GMO=1) şi sîntem în tactul i_6(i_6=1) şi avem microprogramată instrucţia OAZ (în RI avem B1=1) şi nu

avem OSI (în RI avem BO=o) şi conţinutul registrului A este zero (ZA=1), atunci se aduce bistabilul de omitere la unu (A(1)BOM=1).

Sau dacă situaţia este ca mai sus dar avem OSI (în RI avem BO=1) şi conţinutul registrului A nu este zero (ZA=o), atunci se aduce BOM la unu (A(1)BOM=1).

Cele de mai sus pot fi cuprinse în următorul tabel de combinaţii.

GMO	i_6	BO (OSI)	Bl (OAZ)	ZA	A(1)BOM
1	1	o	1	1	1
1	1	1	1	o	1

Din tabel se poate scrie ecuaţia:

$$(ZA.\overline{BO}+\overline{ZA}.BO).i_6.GMO.Bl = A(1)BOM \qquad (3')$$

63 OSI Omite în sens invers, adică inversează condiţia de omitere pentru oricare din instrucţiile de omitere din GMO, respectiv omite dacă condiţia de nu zero este îndeplinită (condiţia de omitere pentru fiecare din instrucţiile precedente OLZ, OA+, OAP, OAZ, cînd nu avem OSI este condiţie de zero).

S-a văzut la instrucţiile precedente felul cum acţionează OSI şi au fost scrise ecuaţiile corespunzătoare.

Folosind OSI putem efectua şi o instrucţie de omitere necondiţionată.

64. OND Omite necondiţionat. Dacă executăm grupul de instrucţii modificări şi omiteri (GMO=1) şi sîntem în tactul 6 (i_6=1) şi avem OSI (în RI avem BO=1) şi nu avem microprogramată nici o instrucţie

de omitere (în RI avem B9=o, OM=o, B3=o, B1=o),atunci se aduce bistabilul de omitere la unu (A(1)BOM=1).

Cele de mai sus pot fi prinse în următorul tabel de combinaţii.

GMO	i_6	$\overline{BO}$ (OSI)	B9 (OLZ)	OM (OA+)	B3 OAP	B1 (OAZ)	A(1)BOM
1	1	1	o	o	o	o	1

Din tabel se poate scrie ecuaţia:

$$\overline{B9}.\overline{OM}.\overline{B3}.\overline{B1}.i_6.BO.GMO = A(1)BOM \quad (3'')$$

Unificăm relaţiile (3') şi (3"), care se execută ambele la impulsul de tact i_6 şi grupînd după BO în $\overline{BO}$ avem:

$$[ZA\cdot B1\cdot i_6\cdot\overline{BO}+(\overline{ZA}\cdot B1+\overline{B9}\cdot\overline{OM}\cdot\overline{B3}\cdot\overline{B1})\cdot i_6\cdot BO]\cdot GMO = A(1)BOM \quad (3)$$

Dacă unificăm ecuaţiile (1), (2) şi (3) şi facem ca impulsurile de tact să apară ca factori comuni obţinem:

$$A(1)BOM=\{(\overline{L}\cdot\overline{BO}+L\cdot BO)\cdot B9\cdot i_4+[(SnA\cdot OM+\overline{OA}\cdot B3)\cdot\overline{BO}+(SnA\cdot OM\cdot\overline{B3}+OA\cdot OM\cdot B3+SnA\cdot OA\cdot OM\cdot B3)\cdot BO]\cdot i_5+[ZA\cdot B1\cdot\overline{BO}+(\overline{ZA}\cdot B1+\overline{B9}\cdot\overline{OM}\cdot\overline{B3}\cdot\overline{B1})\cdot BO]\cdot i_6\}\cdot GMO \quad (4)$$

Variabilele care reprezintă impulsurile de tact i_4, i_5 şi i_6 apar fizic într-un interval de timp foarte scurt - sînt impulsuri ascuţite - pe cînd celelalte variabilele sînt de durată mai lungă.

Obs. generală privind efectuarea instrucţiilor de omitere din GMO (vezi ecuaţiile (1),(2) şi (3)).

Cînd avem în cuvîntul GMO microprogramate mai multe instrucţii de omitere (la impulsurile de tact i_4, i_5, i_6) şi nu avem OSI, omiterea apare atunci cînd condiţia de zero este îndeplinită pentru cel puţin una din instrucţii.

Cînd avem OSI lucrurile se petrec la fel doar că avem condiţia de nu zero. In acest caz avem însă o singură excepţie la i_5 (vezi ec.(2)). Cele două instrucţii OA+ şi OAP, care se execută la i_5, dacă sînt microprogramate simultan, provoacă omitere (la i_5) numai dacă simultan este îndeplinită condiţia de **nu** zero la ambele (aşa am scris ecuaţiile, căci am dorit să avem o astfel de posibilitate, ceace înseamnă că se omite numai dacă numărul din A este negativ şi impar). Bineînţeles că dacă avem şi alte instrucţii de omitere microprogramate, OLZ sau/şi OAZ omiterea se face corespunzător, în funcţie şi de ele. Astfel s. ex. dacă la OLZ este îndeplinită condiţia de **nu** zero, iar la OA+ şi OAP (excepţie) nu este, omiterea apare. Toate acestea se văd foarte uşor din ecuaţiile (1), (2) şi (3).

Se vede că prezenţa unui bistabil de omitere BOM permite ca îndeplinirea condiţiei, la o singură instrucţie de omitere microprogramată, sau la mai multe, să producă omiterea. Indiferent cîte semnale **A(1)BOM**, dela unu în sus, sosesc, unul după altul, la intrarea de unu a bistabilului BOM, el trece şi rămîne în starea unu. Această stare este testată cu impulsul de tact i_7 şi dacă avem BOM=1 se adaugă unu în NA, ceace provoacă omiterea **instrucţiei** următoare.

Minimizarea relaţiilor pentru semnalul A(1)BOM.

Se vede din relaţiile (1), (2) şi (3) că acestea exprimă, toate la un loc, o singură funcţie **A(1)BOM dată în rela-**

ţia (4).Această funcţie va apărea în tabelele pentru descrierea instrucţiei GMO.Pentru a simplifica scrierea în tabele,acolo ne vom referi la funcţiile minimizate corespunzătoare ecuaţiilor (1),(2) şi (3) de mai sus.La sfîrşit ecuaţia unificată şi minimizată va servi la sinteza schemei pentru producerea semnalului (funcţiei),care ajunge la intrarea de aducere la unu a bistabilului de omitere (A(1)BOM).

Minimizarea o vom efectua separat la cele trei ecuaţii (1),(2),(3) şi anume vom căuta ca cele două părţi a fiecărei relaţii legate de BO respectiv $\overline{BO}$ să **conţină** acelaşi factor într-o parte nenegat,iar în cealaltă negat.Aceasta,sperăm că va duce la simplificarea schemei deoarece pentru realizarea factorului negat nu avem nevoie decît de un circuit de negare în care se introduce factorul nenegat.

Ecuaţia (1)

$$A(1)BOM = (\overline{L}\cdot\overline{BO} + L\cdot BO)\cdot i_4\cdot B9\cdot GMO = [\overline{L}\cdot B9\cdot i_4\cdot\overline{BO} +$$

$$+ (L + \overline{B9} + \overline{i_4})\cdot B9\cdot i_4\cdot BO]\cdot GMO = [\underbrace{\overline{L}\cdot B9\cdot i_4}_{X'}\cdot\overline{BO} + (\underbrace{\overline{\overline{L}\cdot B9\cdot i_4}}_{\overline{X'}})\cdot$$

$$\cdot \underbrace{B9\cdot i_4}_{Y'}\cdot BO]\cdot GMO \qquad (5)$$

Ecuaţia (2)

$$A(1)BOM = [(\overline{SnA}\cdot OM + \overline{OA}\cdot B3)\cdot i_5\cdot\overline{BO} + (SnA\cdot OA\cdot OM\cdot B3 +$$

$$+ SnA\cdot OM\cdot\overline{B3} + OA\cdot\overline{OM}\cdot B3)\, i_5\cdot BO]\cdot GMO$$

Luăm numai partea din ecuaţie care conţine pe BO şi o transformăm urmărind acelaşi ţintă ca mai sus

$$(SnA\cdot OA\cdot OM\cdot B3 + SnA\cdot OM\cdot\overline{B3} + OA\cdot\overline{OM}\cdot B3)\cdot i_5\cdot BO =$$

$$= (SnA\cdot OA\cdot B3 + SnA\cdot OM\cdot B3 + OM\cdot B3\cdot OA)\cdot i_5\cdot BO =$$

$$= (SnA\cdot OA\cdot B3 + SnA\cdot OA\cdot B3\cdot OM + SnA\cdot OM\cdot\overline{B3} + SnA\cdot OM\cdot\overline{B3}\cdot$$

$$\cdot OA + \overline{OM}\cdot B3\cdot OA)\cdot i_5\cdot BO = (SnA\cdot OA\cdot B3 + SnA\cdot OA\cdot OM +$$

$$+ SnA\cdot OM\cdot\overline{B3} + OA\cdot\overline{OM}\cdot B3)\cdot i_1\cdot BO =$$

$$= (\overline{SnA}.OA.B3.i_5+\overline{SnA}.OA.OM.i_5+\overline{SnA}.OM.\overline{B3}.i_5 +$$

$$+ OA.\overline{OM}.B3.i_5).BO =$$

$$= \left[\overline{SnA}(OA.B3.i_5+OA.OM.i_5+OM.\overline{B3}.i_5)+OA.\overline{OM}.B3.i_5\right].BO =$$

$$= (\overline{SnA}+\overline{OM}+\overline{i_5})(OA.B3.i_5+OA.OM.i_5+OM.\overline{B3}.i_5).BO =$$

$$= (\overline{SnA}+\overline{OB}+\overline{i_5})(OA+\overline{B3}.\overline{i_5}).(B3.i_5+OM.i_5).BO =$$

$$\overline{(\overline{SnA}.OM.i_5+\overline{OA}.B3.i_5)}.(B3.i_5+OM.i_5).BO$$

Introducînd expresia nouă în ecuaţia (2) şi înmulţind în prima paranteză cu i_5, obţinem:

$$A(1)BOM=\left[\underbrace{(\overline{SnA}.OM.i_5+OA.B3.i_5)}_{X''}.BO+\underbrace{(SnA.OM.i_5+OA.}_{\overline{X''}}\right. \quad (6)$$

$$\left.\underbrace{.B3.i_5)}_{\overline{X''}}.\underbrace{(B3.i_5+OM.i_5)}_{Y''}BO\right] GMO$$

Ecuaţia (3)

$$A(1)BOM=\left[ZA.B1.i6.\overline{BO}+(\overline{ZA}.B1+\overline{B9}.\overline{OM}.\overline{B3}.\overline{BI}).i_6.BO\right] .$$

$$. GMO =$$

$$= \left[ZA.B1.i_6.\overline{BO}+(\overline{ZA}+\overline{B1}+\overline{i_6}).(B1+\overline{B9}.\overline{OM}.\overline{B3}.\overline{BI}).i_6.BO\right] .$$

$$. GMO =$$

$$=\left[\underbrace{ZA.B1.i_6.\overline{BO}}_{X'''}+\underbrace{\overline{ZA}.B1.i_6}_{\overline{X'''}}.\underbrace{(B1+\overline{B9}.\overline{OM}.\overline{B3}.\overline{BI}).i_6.BO}_{Y'''}\right].GMO \quad (7)$$

Ecuaţiile(5),(6) şi (7) reprezintă termeni ai aceleiaşi funcţii A(1)BOM, prin urmare le putem unifica legînd partea dreaptă prin operaţii SAU. Avînd în vedere că am dat tuturor celor trei ecuaţii aceeaşi formă,unificarea o putem face ca mai jos.

$$X = X'+X''+X''' = \overline{L}.B9.i_4+\overline{SnA}.OM.i_5+\overline{OA}.B3.i_5+ZA.B1.i_6 \quad (8)$$

$$Y=Y'+Y''+Y'''=B9\cdot i_4+B3\cdot i_4+OM\cdot i_5+B1\cdot i_6+\overline{B9}\cdot\overline{OM}\cdot\overline{B3}\cdot$$
$$\cdot\overline{B1}\cdot i_6 \qquad (9)$$

$$A(1)BOM=(X\cdot\overline{BO}+\overline{X}\cdot Y\cdot BO)\cdot GMO \qquad (1o)$$

Ecuaţiile (8),(9) şi (1o), înainte de a fi utilizate la sinteza schemei circuitelor care formează funcţia A(1)BOM, trebuie supuse unei analize spre a vedea dacă întîrzierile,care apar în mod inevitabil la trecerea semnalelor prin circuite,nu împiedică formarea corectă a unor termeni.Se vede din relaţii că în urma prezenţei variabilelor i în termenii funcţiilor X şi Y, acestea sînt de durată scurtă (o,2 microsec.) ca şi variabilele i.Prin urmare în ec.(1o) termenul $\overline{X}.Y.BO$ este format din doi factori de durată scurtă $\overline{X}$ respectiv Y şi de unul de durată lungă BO şi există deci pericolul ca cei doi factori să nu apară în acelaşi timp datorită întîrzierii la formarea lor prin număr de nivele (etaje) diferite şi deci termenul $\overline{X}.Y.BO$ să nu se formeze corect.Astfel dacă considerăm situaţia la impulsul de tact i_5 avem:

$$X_{i5}=\overline{SnA}\cdot OM\cdot i_5+\overline{OA}\cdot B3\cdot i_5$$

$$Y_{i5}=OM\cdot i_5+B3\cdot i_5$$

La sinteza schemei putem forma $OM.i_5$ respectiv $B3.i_5$ din ecuaţia pentru Y_{i5} numai o singură dată în scopul de utiliza rezultatul la formarea lui Xi_5. Ţinînd seama că noi avem nevoie de $\overline{X}_{i5}$, se vede că formarea acestuia din urmă necesită patru nivele pe cînd formarea lui Yi_5 necesită numai două.Aceasta creează pericolul de a nu obţine coincidenţă pe o durată su-

ficient de îndelungată,sau chiar să nu coincidă de loc, perioadele de apariţie a lui Xi_5 respectiv Yi_5 şi deci termenul se poate forma eronat.

Pentru a evita astfel de neajunsuri,renunţăm la ecuaţiile (8),(9) şi (1o) şi utilizăm la sinteza schemei ecuaţia (4) în care variabilele de durată scurtă apar ca factor comun,deci o singură dată,şi în consecinţă nu se formează funcţii logice SI între factori de scurtă durată.Prin aceasta numărul de circuite creşte cu puţin.Dealtfel asupra criteriilor de minimizare a ecuaţiilor logice vom mai reveni.

Instrucţiile de introducere-extragere (IIE)

După cum am văzut,în grupa de instrucţii de introducere-extragere,am creat şi posibilitatea codificării unui mare număr de macroinstrucţii,prin aceea că am prevăzut un cuvînt instrucţie denumit MAC, prin care 18 biţi pot fi utilizaţi pentru codificarea macroinstrucţiilor.Aceste instrucţii pot fi introduse numai atunci cînd există sisteme de operare corespunzătoare. In acest caz codul MAC conduce mai întîi calculatorul la un program special.Acesta "ştie" unde se găseşte subprogramul care reprezintă macroinstrucţia şi ia măsurile pentru introducerea lui în programul principal. Deoarece MAC poate fi folosită numai cînd este elaborat sistemul de operare corespunzător,noi nu vom număra această instrucţie printre instrucţiile de bază ale calculatorului.

65. OPR Opreşte calculatorul. Calculatorul se consideră oprit cînd impulsurile de oroloziu (H) nu mai ajung la generatorul impulsurilor i, acces care este permis sau oprit de către bistabilul de pornire-oprire (BPO). Deci instrucţia se execută prin aceea

că se trimite impuls la borna A(o)BPO.

66. AZDC Adă la zero bistabilul DCR. Se trimite impuls la borna A(o)DCR.

67. AUDC Adă la unu bistabilul DCR. Se trimite impuls la borna A(1)DCR.

68. ODCZ Omite dacă bistabilul DCR se află în starea zero. Dacă conţinutul bistabilului DCR este zero se trimite pe lîngă impulsul obişnuit încă un impuls la borna +1NA.

69. ODCU Omite dacă bistabilul DCR se află în starea zero. Dacă conţinutul bistabilului DCR este unu se trimite pe lîngă impulsul obişnuit încă un impuls la borna +1NA.

70. G1 IN R Conţinutul registrului cu comutatoare G1 se trece în registrul R. Se aduce mai întîi la zero registrul R prin trimiterea unui impuls la bornele A(o)SmR şi A(O)MR, iar apoi se trece conţinutul lui G1 în R prin trimiterea unui semnal de potenţial la borna PmR şi a unui impuls la borna PdG1.

71. G1 SAU R Conţinutul registrului cu comutatoare G1 se trece în registrul R prin funcţia SAU. In cursul acestei instrucţii conţinutul lui G1 se trece în R efectuîndu-se între biţii de acelaşi rang a celor două cuvinte funcţia SAU. Rezultatul se păstrează în R.

Obs. Modul de execuţie al acestei instrucţii se deosebeşte de cel al instrucţiei precedente G1 IN R numai prin faptul că la instrucţia G1 SAU R nu se aduce conţinutul registrului R la zero înainte de a se trimite cuvîntul din G1 în R.

72. G2 IN B Conţinutul registrului cu comutatoare G2 se trece în registrul B. Se aduce mai întîi la zero registrul B prin trimiterea unui impuls la

borna A(o) B, iar apoi se trece conţinutul lui G2 în B prin trimiterea unui semnal de potenţial la borna PmB şi a unui impuls la borna = PdG2.

73. AZQ Adă la zero conţinutul registrului Q

Se trimite impuls la borna A(o)Q.

Obs. Această instrucţie poate fi microprogramată împreună cu instrucţie INT (introdu) în scopul de a se oferi programatorului mai multe posibilităţi în privinţa introducerii. Astfel dacă asamblăm cuvîntul în Q atunci AZQ nu e necesară, dar dacă îl asamblăm în alt registru ea e necesară.

Instrucţia AZQ se execută independent de valoarea codului de selecţie pe cînd INT este dependentă de acest cod.

74. PORP Porneşte dispozitivul periferic

Această instrucţie permite pornirea de către calculator a dispozitivului periferic al cărui cod de selecţie este în cuvîntul instrucţie, dacă perifericul este înscris înzestrat cu circuitele necesare unei astfel de porniri. In caz contrar pornirea se face manual. Executarea instrucţiei se face în felul următor: Se trimite impuls sau semnal de potenţial (impuls de durată mai lungă) la borna POR P a dispozitivului periferic al cărui cod este înscris în cuvîntul instrucţie.

75. OPR P Opreşte dispozitivul periferic

Această instrucţie permite oprirea de către calculator a dispozitivului periferic al cărui cod de selecţie este înscris în cuvîntul instrucţie, dacă perifericul este înzestrat cu circuitele necesare unei astfel de opriri. In caz contrar oprirea se face manual. Executarea instucţiei se face în felul următor: Se trimite impul sau semnal de potenţial (impuls de durată mai lungă) la borna OPRP a dispozitivului periferic al cărui cod

este înscris în cuvîntul instrucţie.

75. AZF Adă la zero fanionul de introducere. In circuitele de interfaţă pentru fiecare dispozitiv periferic care poate fi de introducere sau de extragere este prevăzut unul sau două circuite bistabile numite fiecare fanion (FANA, FANB) care sînt aduse în starea unu atunci cînd periferical este gata pentru ca un fragment de informaţie (s.ex. un caracter) să poată fi introdus sau extras în respectiv din calculator, sau atunci cînd un alt proces este terminat. Această instrucţie se execută prin aceea că se trimite şi simultan un impuls la bornele A(o)FANA şi A(o)FANB a dispozitivului periferic al cărui cod de selecţie se află înscris în cuvîntul instrucţie.

77. OFA Omite dacă fanionul A se află în starea unu. Dacă bistabilul fanion (FANA) al dispozitivului periferic al cărui cod de selecţie se află înscris în cuvîntul instrucţie, se află în starea unu, se trimite, pe lîngă impulsul obişnuit, încă un impuls la borna +1NA.

78. OFB Omite dacă fanionul B se află în starea unu. Dacă bistabilul fanion (FANB) al dispozitivului periferic al cărui cod de selecţie se află înscris în cuvîntul instrucţie, se trimite, pe lîngă impulsul obişnuit, încă un impuls la borna +1NA.

79. INT Introdu un caracter. Din exterior, adică din dispozitivele periferice, informaţia sub formă de caracter se introduce prin intermediul şinelor codului informaţiei (SCI) în registrul Q, care are posibilităţi de deplasare a conţinutului spre stînga şi dreapta. Aceasta e necesar în scopul unei aranjări a informaţiei pentru formarea unui cuvînt.

Această instrucţie se execută în felul următor: Se trimite un semnal de potenţial la borna PmQ şi simultan cu acesta se trimite un impuls la borna INTROD a dispozitivului periferic al cărui cod de selecţie se află înscris în cuvîntul instrucţie.

Obs. Dacă ar fi necesară o instroducere într-un alt registru s.ex. în RM, se poate prevedea, relativ uşor, o astfel de instrucţie de introducere. Vezi şi instrucţia 73.

80. EXT Extrage un caracter

Informaţia care se extrage din calculator se află în rangurile curs ale registrului A. Ea se trimite la un dispozitiv periferic. Extragerea informaţiei din registrul A este legată de faptul ca în unele cazuri informaţia care se extrage reprezintă rezultatul unor calcule prealabile, rezultat care se află în registrul A.

Această instrucţie se execută în felul următor: Se trimite mai întîi un impuls la borna t_1 a dispozitivului periferic al cărui cod de selecţie se află înscris în cuvîntul instrucţie. Se trimite apoi un semnal de potenţial la borna PmRB (primeşte în registrul RB al interfeţei) şi simultan cu aceasta se trimite un impuls la borna t_2. Ambele se trimit la dispozitivul periferic al cărui cod de selecţie se află înscris în cuvîntul instrucţie. Tot simultan cu precedentele se trimite un impuls la borna PdA, a registrului A.

81. OTS Omite la tensiune scăzută.

Este testat fanionul pentru tensiune scăzută a reţelei şi dacă conţinutul acestuia este unu (s-a detectat o cădere a tensiunii reţelei de alimentare), conţinutul lui NA este mărit cu unu astfel încît instrucţia următoare este omisă.

Obs Prin omitere putem trece la un subprogram.

Durata subprogramului nu va fi mai lungă de 1 ns

Obs. Cînd tensiunea rețelei a revenit, un circuit de repornire aduce la zero FANTS,**și repornește pro**gramul.O cheie de repornire (RESTART),plasată pe modulul de detectare a căderii tensiunii,autorizează sau dezautorizează restabilirea automată a funcționării.Cu cheia în poziția de autorizare se aduce la zero RA imediat și se produce un semnal care simulează semnalul de**ST INIȚ**cu 2oo ms după ce condițiile tensiunii sînt satisfăcătoare. RA fiind la zero funcționarea începe cu executarea instrucției dela adresa oooo. Această instrucție trebuie să fie SSP (salt la subprogram) la adresa de început a unui subprogram, care restabilește conținutul registrelor active și**a lui** NA la starea de dinainte de întreruperea la căderea tensiunii.Intîrzierea de 2oo ms asigură ca dispozitivele mecanice încete,cum este teleimprimatorul,să-și fi terminat anumite operații înainte ca programul să fie reluat.Simularea funcției manuale **STINIT** determină calculatorul să genereze un impuls de putere care aduce la zero elementele de comandă interne,precum și registrele dispozitivelor de I/E. Cînd cheia RESTART se află în poziția de dezautorizare, FANTS este adus la zero, dar pornirea calculatorului trebuie făcută manual, ceace este posibil după aducerea în stare de funcționare a dispozitivelor periferice,sau cînd se reia dela început programul întrerupt.

82. AITR Autorizează întreruperea

Instrucțiunea autorizează calculatorul să răspundă la cerere de întrerupere a programului **în curs. Dacă** atunci cînd se execută această instrucție, întreruperea este dezautorizată,calculatorul execută instrucția următoare,după care este autorizată întreruperea.

Instrucţia care se execută înainte de a fi autorizată întreruperea permite ieşirea din subprogramul de întrerupere înainte de a îngădui apariţia unei noi întreruperi.Această instrucţie nu are nici un efect asupra condiţiilor circuitelor de întrerupere daçă ea este executată cînd întreruperea este autorizată.

83. DITR **Dezautorizează** întreruperea

Instrucţiunea nu permite calculatorului (îl dezautorizează) să răspundă la o cerere de întrerupere,în scopul de a împiedica o întrerupere a programului curent.

Descrierea comenzilor date manual dela pupitrul de comandă (PU) cu ajutorul cheilor

Am arătat la prezentarea generală a metodei de proiectare că odată cu descrierea instrucţiilor vom descrie şi funcţionarea cheilor de pe pupitrul de comandă, cu ajutorul cărora se generează manual comenzi similare cu acelea care apar în cazul executării instrucţiilor. Aceasta permite o descriere unitară a tuturor elementelor care dau comenzi.

Pentru a putea urmări mai bine descrierea funcţionării cheilor vom prezenta mai întîi,pe scurt,pupitrul de comandă.Aceasta conţine cele două registre cu comutatoare,numite şi generatoare de cuvinte G1 şi G2, a căror conţinut,înscris manual prin aşezarea comutatoarelor,poate fi trimis în registrul R sau în registrul RM prin instrucţiile văzute: G1 IN R, G1 IN RM, G1 SAU R, G1 SAU RM. După cum se va arăta mai jos conţinutul celor două registre poate fi trimis şi în alte registre (NA,RA) şi în memorie,cu ajutorul unor chei aflate pe P.C. Tot pe P.C. se găseşte şi întrerupătorul RETEA,cu ajutorul căruia se conectează calculatorul la reţeaua de tensiune alternativă,Cînd sursele stabilizate ating tensiunile normale se ilumineză pe PC becuri corespun-

zătoare fiecărei tensiuni.Conţinutul memoriei nu este afectat cînd se conectează respectiv deconectează calculatorul la reţea,dar conţinutul registrelor se pierde.După conectare conţinutul registrelor este aleator. In continuare se descriu comenzile date de cheile de pe P.C.

1. Cheia STARE INITIALA. Este o cheie momentană, adică ea face un contact atîta vreme cît este ţinută într-o anumită poziţie, după care contactul se întrerupe.Cheia are rolul de a aduce calculatorul în starea iniţială.Cheia acţionează un generator de impulsuri singulare (GS), care dă la ieşirea lui un impuls format $i_{ST\ IN}$. Cheia se acţionează numai cînd calculatorul este conectat la reţea,dar este OPRIT. Impulsul $I_{ST\ IN}$ este trimis la borna f_{z1} şi aduce prin aceasta calculatorul în faza ADUCERE.Acelaşi impuls mai este trimis la următoarele borne: A(o)BPO,A(o)BPOR,A(o)BOPR1, A(o)BOPR2, A(o)BMEM, A(o)BINAD, A(o)BVIZ, A(o)BUNCICL, A(o)BINCA, A(7)N8, A(o)BII, prin care se obţine aducerea la zero a bistabilelor corespunzătoare.

2. Cheia PORNESTE, Este o cheie momentană utilizată pentru pornirea calculatorului, care este deja conectat la reţea. După acţionarea cheii se aprinde indicatorul luminos. Acesta se stinge cînd se manipulează cheia OPRESTE sau cînd se execută instrucţia OPR (opreşte calculatorul).

Cheia acţionează un GS, care dă la ieşirea lui un impuls format i_{POR}. Cheia se acţionează numai cînd calculatorul este conectat la reţea dar este OPRIT. Impulsul i_{PCR} este trimis la borna A(1)BPOR a unui bistabil BPOR ataşat acestei chei, care memorează comanda şi care are rolul de a permite o sincronizare cu impulsurile de orologiu.Acum dacă BPOR=1,

primul impuls de orologiu,pe care îl notăm cu i,va fi trimis la borna A(1)BPO ceace aduce bistabilul de pornire-oprire (BPO) în starea unu.Prin aceasta se porneşte calculatorul şi în aceleaşi condiţii impulsul i este trimis la borna A(o)BPOR prin care BPOR este adus la zero.

Dacă nu am fi ataşat la cheie bistabilul BPOR şi deci impulsul i_{POR} ar fi trimis direct la borna A(1)BPO s-ar putea întîmpla următoarele: Dacă prin i_{POR} bistabilul BPO ar fi adus la unu chiar în timpul apariţiei unui impuls de tact i, acesta ar putea trece prin poarta deschisă încă incomplet către BPO cu amplitudine mai mică decît cea normală şi cum acesta este dus la mai multe intrări (+1N8, Δ , circ.care face funcţia SI) ar putea ca pe unele să le comande, iar pe altele nu, ceace ar duce la eroare.Introducem bistabilul BPOR, care este adus la unu de i_{POR}. Dacă basculareа lui se face în timpul apariţiei unui impuls i, şi acesta trecînd prin poarta SI deschisă de BPOR=1 (parţial) o bţine o amplitudine insuficientă lucrurile s-ar petrece fără aroare dacă acest impuls ar fi dus numai la borna A(1)BPO căci în cazul cînd nu ar avea amplitudine suficientă pentru basculareа lui BPO,aceasta s-ar face la impulsul i următor,acum poarta SI fiind deja deschisă.Vom vedea că la cele mai multe chei aşa se petrec lucrurile.

In cazul nostru impulsul ieşit din poarţa SI deschisă mai trebuie să meargă şi la A(o)BPOR. Se pot întîmpla următoarele cazuri anormale: BPO să fie dus la unu,iar BPOR să nu fie adus la zero,în care caz calculatorul porneşte şi următorul impuls i va aduce la zero pe BPOR.Sau se poate întîmpla ca BPO să nu fie adus la unu,iar BPOR să fie adus la zero.In acest

caz calculatorul nu porneşte,ceace se observă după becul de pe PC şi se manipulează din nou cheia. Se vede deci că erori nu pot apărea.

3. Cheia OPREŞTE. Este o cheie momentană folosită pentru a opri funcţionarea calculatorului la sfîrşitul fazei, în timpul efectuării căreia a fost manipulată cheia. Cînd calculatorul este oprit indicatorul luminos OPRIT este aprins.

Cheia acţionează un GS, care dă la ieşire un impuls format i_{OPR}. Cheia se acţionează numai cînd calculatorul este PORNIT. Impulsul i_{OPR} este trimis la borna A(1)BOPR1 a unui bistabil BOPR1 ataşat cheii, care memorează comanda. Primul impuls de orologiu i care urmează după ce BOPR1 a fost adus la 1 aduce la unu pe BOPR2 (necesar din motive de sincronizare).La sfîrşitul fazei în care s-a dat comandă (cu i_7),dacă avem BOPR2=1 atunci impulsul i_7 se trimite la borna A(o)BPO, prin care se opreşte calculatorul, şi la borna A(o)BOPR1, prin care se şterge memorarea în BOPR1.In privinţa sincronizării vezi obs. de la cheia nr.2.

4. Cheia MEMOREAZA. Este o,cheie momentană folosită pentru memorarea conţinutului generatorului de cuvinte G1 în memorie la adresa specificată în registrul NA. După acţionarea cheii MEMOREAZA conţinutul registrului NA este mărit în mod automat cu unu, în scopul memorării cuvintelor la adrese consecutive din memorie. Cuvîntul memorat rămîne în RM şi este vizualizat prin becurile ataşate acestui registru. La sfîrşitul operaţiei de memorare se stabileşte faza ADUCERE.

Cheia acţionează un GS, care dă la ieşire un impuls format i_{MEM}. Cheia se acţionează numai cînd calculatorul este OPRIT. Impulsul i_{MEM} este trimis la borna A(1)BMEM a unui bistabil BMEM ataşat cheii,care memorează comanda,şi la fz5, prin care generatorul de faze stabileşte Fz5 în care se execută microoperaţiile celor mai multe chei. Faza Fz5 este necesa-

ră pentru a putea obţine microoperaţiile provocate de chei în mod clar şi cu circuite logice mai simple. Dacă BMEM=1 atunci primul impuls de orologiu,pe care îl notăm cu i, va fi trimis la borna A(1)BPO prin care se porneşte calculatorul.Menţionăm că întrucît înainte calculatorul a fost OPRIT, el începe să funcţioneze începînd cu impulsul i_0. Acum calculatorul execută aceleaşi microoperaţii ca şi în cazul instrucţiei MG1 în faza EXECUTIE (FZ3) cu deosebirea că,la sfîrşitul fazei,i_6 se trimite la borna +1NA, iar i_7 la bornele fz1, A(o)BMEM şi A(o)BPO prin care se măreşte adresa cu unu,se stabileşte faza ADUCERE, se şterge memorarea comenzii şi se OPREŞTE calculatorul.

In privinţa sincronizării vezi obs. de la cheia nr.2.

5. Cheia INCARCA ADRESA. Este o cheie momentană folosită pentru trecerea conţinutului generatorului de cuvinte G1 în registrul NA. In acest caz conţinutul lui G1 este o adresă.La sfîrşitul operaţiei INCARCA ADRESA se stabileşte faza ADUCERE.

Cheia acţionează un GS, care dă la ieşire un impuls format i_{INAD}. Cheia se acţionează numai cînd calculatorul este OPRIT. Impulsul i_{INAD} este trimis la borna A(1)BINAD a unui bistabil BINAD, ataşat cheii, care memorează comanda precum şi la fz5. Acum dacă BINAD=1 atunci primul impuls de orologiu,pe care îl notăm cu i, este trimis la borna A(1)BPO, ceace porneşte calculatorul.Acesta porneşte începînd cu i_0.Cu ajutorul secvenţei de impulsuri i_0-i_7 se aduce la zero NA, se trimite conţinutul lui G1 în NA, iar la sfîrşit se stabileşte faza ADUCERE, se aduce la zero BINAD, stergîndu-se astfel memorarea comenzii şi se aduce la zero B PO, oprindu-se astfel calculatorul.Vezi obs.

dela cheia nr.2.

6. Cheia VIZUALIZEAZA. Este o cheie momentană care se foloseşte pentru a vizualiza în registrul RM conţinutul locaţiei din memorie a cărei adresă se află înscrisă în registrul NA.După manipularea cheii VIZUALIZEAZA, conţinutul registrului NA este automat mărit cu unu, astfel încît conţinutul locaţiei consecutive din memorie este vizualizat simplu prin repetarea manipulării acestei chei.(Astfel registrul NA indică o adresă cu unu mai mare decît aceea a conţinutului vizualizat în RM).După mărirea cu unu a conţinutului registrului NA se stabileşte faza ADUCERE.

Cheia acţionează un GS, care dă la ieşire un impuls format i_{VIZ}. Cheia se acţionează numai cînd calculatorul este OPRIT. Impulsul i_{VIZ} este trimis la borna A(1)BVIZ a unui bistabil BVIZ, ataşat cheii, care memorează comanda,precum şi la fz5.Acum dacă BVIZ=1 atunci primul impuls de orologiu,pe care îl notăm cu i, este trimis la borna A(1)BPO, ceace porneşte calculatorul.Acesta porneşte începînd cu i_0.Cu ajutorul secvenţei de impulsuri i_0-i_7 se citeşte cuvîntul din memorie dela adresa specificată în NA şi se primeşte în RM. Apoi se măreşte cu unu conţinutul lui NA, iar la sfîrşitul secvenţei se trimite impuls la bornele A(o)BVIZ, A(o)BPO-şi la fz1, prin care se sterge memorarea comenzii,se opreşte calculatorul şi se stabileşte faza ADUCERE.Vezi obs. dela nr.2.

7. Cheia UN CICLU. Este o cheie momentană folosită pentru executarea unei singure FAZE a calculatorului de fiecare dată cînd cheia este manipulată.

Cheia acţionează un GS, care dă la ieşire un impuls format $i_{UN\ CICL}$. Cheia se acţionează numai cînd calculatorul este OPRIT. Impulsul $i_{UN\ CICL}$ este

trimis la borna A(1) BUNCICL,a unui bistabil BUNCICL ataşat cheii, care memorează comanda. Acum dacă BUNCICL=1 atunci primul impuls de orologiu, pe care îl notăm cu i, este trimis la borna A(1)BPO, ceace porneşte calculatorul. Calculatorul execută faza prevăzută a instrucţiei troduse,iar la terminarea ei i_7 este trimis la borna A(o)BUNCICL, prin care se şterge memorarea comenzii şi la borna A(o)BPO prin care se OPRESTE calculatorul.Vezi obs. de la cheia nr.2.

8. Cheia INCARCA A. Este o cheie momentană care se foloseşte pentru a trece în registrul A cuvîntul înscris manual în Gl.

Cheia acţionează un GS, care dă la ieşire un impuls format $i^{\bullet}_{INCA}$. Cheia se acţionează numai cînd calculatorul este OPRIT. Impulsul i_{INCA} este trimis la borna A(1)BINCA, a unui bistabil BINCA ataşat cheii, care memorează comanda precum şi la fz5. Acum dacă BINCA=1 atunci primul impuls de orologiu, pe care îl notăm cu i, este trimis la borna A(1)BPO, ceace porneşte calculatorul. Calculatorul care începe secvenţa cu i_o execută operaţii asemănătoare cu cele de la instrucţia RM Nr.4 (încarcă A). Diferă faptul că în cazul nostru cuvîntul care se încarcă A nu se ia din memorie, ci din Gl. De asemenea la sfîrşitul secvenţei trimitem impulsul i_7 la bornele A(o)B INCA, A(o)BPO şi la borna fzl, prin care se şterge memorarea comenzii, se OPRESTE calculatorul şi se stabileşte faza ADUCERE. Vezi obs. de la cheia nr.2.

9. Cheia UNPAS. Este o cheie cu 3 poziţii, anume: "normal" (NORM), "un pas" (UNP) şi "numărătorul N8 acţionat manual" (N8M). Ea este necesară cînd se verifică corecta funcţionare a calculatorului, respectiv cînd

se caută deranjamente.

Cu ajutorul ei se poate instala situaţia normală, (NORM), cînd impulsurile date de generatorul de tact H pătrund în generatorul impulsurilor de orologiu, pentru a se transforma în impulsuri i.

Intr-o altă poziţie (UNP) cheia deconectează generatorul de tact H şi conectează la generatorul de impulsuri de orologiu (G_1) un generator singular (GR), pe a cărui buton, dacă apăsăm, trimitem cîte un singur impuls (g.), care se transformă în impulsuri i. In felul acesta putem urmări microoperaţiile ce se execută la oricare din impulsurile i.

In a treia poziţie (N8M) generatorul singular este conectat la intrarea de numărare (+1 N8) a numărătorului N8, ceea ce ne permite să-l aducem în oricare stare. Astfel putem executa microoperaţiile de la oricare impuls i, fără să trebuiască să executăm microoperaţiile corespunzătoare impulsurilor precedente.

Aceste posibilităţi sînt de mare utilitate cu ocazia punerii la punct a calculatorului şi a exploatării lui.

Manipularea cheii se va face numai cînd calculatorul este oprit.

lo. Cheile Fz1...Fz6. Aceste chei în număr de 6, corespunzătoare celor 6 faze, acţionează fiecare din ele cîte un GS. Cheile se manipulează numai cînd calculatorul este oprit. Impulsul dat de generatorul singular instalează faza corespunzătoare.

Cheile servesc la înlăturarea deranjamentelor. Cu ajutorul acestora instalăm faza dorită, iar cu ajutorul cheii UNPAS obţinem impulsul de orologiu dorit şi putem astfel genera microoperaţiile, care se efectuează la o anumită fază şi la un anumit impuls i al unei instrucţii.

11. Cheia AZNA. Este o cheie momentană care acţionează un GS, cu ajutorul căreia se aduce la zero conţinutul lui NA.Serveşte la manevrele ce se fac de la pupitrul de comandă.

12. Cheia PREINTRODUCERE. Este o cheie momentană, care serveşte la înscrierea în memorie a programului de preintroducere.

In calculator informaţia se introduce de pe banda perforată.Conţinutulbenzii perforate se introduce cu ajutorul unui program.Acest program la rîndul său trebuie introdus printr-un alt mijloc decît banda perforată.El se poate introduce cu ajutorul generatoarelor de cuvinte şi a cheilor INAD şi MEM, de pe pupitru de comandă.Cum acest lucru este incomod, se utilizează o memorie fixă,în care se înscrie programul de preintroducere şi care,atunci cînd se apasă pe cheia PREINT,este introdus automat în memoria operativă.

Cheia actionează un GS.Ea se acţionează numai cînd calculatorul este oprit.La apăsarea cheii un bistabil BPRI este adus în starea 1 memorînd comanda.Totodată se stabileşte Fz5.Primul impuls de orologiu i,care apare după ce BPRI a ajuns la 1, aduce la unu BPO,prin care se porneşte calculatorul.Microoperaţiile pe care le execută calculatorul acum sînt cele corespunzătoare cheii INAD, prin care se duce,dela adresa zero a memoriei fixe,în NA, adresa la care se înscrie,în memoria operativă,prima instrucţie a programului PREINT.In ciclurile următoare se vor efectua microoperaţii corespunzătoare cheii MEM,prin care se vor memora cuvintele memoriei fixe,de la adrese succesive,în memoria operativă,la adrese succesive.Bineînţeles că pe parcurs se va schimba la fiecare ciclu adresa memoriei fixe.La sfîrşit se va reveni la adresa zero a memoriei fixe şi prin acelaşi proces,ca şi la început,se va reintroduce în NA adresă primei instrucţii a programului de PREINT, care acum este necesară aici pentru a putea

porni calculatorul astfel încît să execute programul, începînd de la prima instrucţie.După introducerea în NA a acestei adrese calculatorul se opreşte.

13. Faza INTRERUPERE (Fz6)

Calculatorul CETA va avea şi posibilităţi de întrerupere a programului de către dispozitivele periferice,în scopul de a se putea cîştiga timp atunci cînd calculatorul lucrează cu acestea.Deasemenea se recurge la posibilităţile de întrerupere şi atunci cînd tensiunea reţelei de alimentare scade sub limita admisă pentru funcţionare, spre a salva conţinutul registrelor active şi apoi a putea reîncepe funcţionarea fără erori,de acolo de unde a fost întreruptă.Pentru efectuarea întreruperii prevedem o fază specială numită faza INTRERUPERE, pe care o tratăm aici la capitolul cheilor pentru motivul că declanşarea microoperaţiilor acestei faze se face dinafară şi similar cu procesele ce au loc cînd se dau comenzi prin chei.

Inainte de a descrie care sînt microoperaţiile efectuate în faza INTRERUPERE să descriem procedeele de introducere şi extragere a informaţiei în calculatorul CETA. Introducerea şi extragerea se pot face:

a. <u>Prin program la iniţiativa calculatorului</u>. Acesta îşi întrerupe programul principal şi face un salt la un subprogram corespunzător,în care intră pe lîngă alte instrucţii şi instrucţiile de introducere-extragere.Subprogramul poate fi de introducere,respectiv extragere şi fiecare subprogram de acest gen deserveşte un anumit periferic.In timpul executării unui astfel de subprogram calculatorul, care funcţionează mult mai repede decît periferical, aşteaptă pînă cînd periferical pregăteşte informaţia de transferat.Aşteptarea se efectuează printr-o buclă de aşteptare realizată în cadrul subprogramului.Se vede că în acest caz se pierde foarte mult timp.

b. Prin program la inițiativa perifericuluI, cînd se face uz de posibilitățile de întrerupere. In acest caz programul principal se întrerupe numai pe timpul cît se face efectiv schimbul de informație. Spre exemplu cînd se lucrează cu CIT TEL, între introducerea, respectiv extragerea a două caractere succesive trece un timp de 150 ms. In acest interval calculatorul este ocupat foarte puțin cu programul de introducere, anume pe parcursul fazei de INTRERUPERE și a cîteva instrucții, după care revine la programul principal pe care îl continuă și abia cînd CIT a terminat pregătirea unui nou caracter se întrerupe din nou programul principal ș.a.m.d.

Pentru ca să putem lucra în acest fel este deci nevoie ca periferical să aibă posibilitatea de întrerupere a programului principal, ceace se va întîmpla prin instalarea fazei de INTRERUPERE atunci cînd periferical anunță printr-un fanion că a pregătit un caracter și îl poate transmite. Inafară de aceasta subprogramul de I/E va trebui să fie întocmit corespunzător. Astfel în subprogram după instrucția INT sau EXT trebuie prevăzută instrucția de salt cu adresare indirectă la adresa zero (SLTI ooo), prin care revenim la programul principal. Inainte de instrucția SLTI ooo se va introduce instrucția de autorizare a întreruperii (AITR), pentru a pregăti lucrurile în vederea unei noi întreruperi. Astfel dispozitivul periferic prin circuitele de întrerupere cere întreruperea programului principal la fiecare caracter pregătit, pînă ce periferical se oprește, odată cu trimiterea transferului de informație.

Descriem acum în detaliu cum anume se produce întreruperea.

Cererea de întrerupere va fi formulată cu ajutorul unui fanion pe care perifericul, sau dispozitivul de detectare a tensiunii scăzute, îl va aduce în starea unu. Se va realiza o bornă denumită FANG la ieşirea unui circuit SAU, la a cărui intrări se aduc ieşirile fanioanelor tuturor perifericilor. Calculatorul va testa la sfîrşitul fiecărui instrucţii (cu ajutorul impulsului i_7) starea bornei FANG. Dacă avem FANG=1, adică avem cerere de întrerupere, şi avem autorizată întreruperea (BITR=1), i_7 va fi trimis la generatorul de faze (GF) pe care îl va aduce în faza INTRERUPERE (Fz6). Totodată va fi dezautorizată întreruperea (cu i_7 se va aduce la zero BITR). In cursul acestei faze Fz6 conţinutul lui NA, în care este deja formată adresa instrucţiei următoare, prin aceia că i s-a adăugat o unitate, va fi memorat la adresa 0000 din memorie; apoi va fi adus la zero NA şi se va adăuga unu la NA, pentru a se forma adresa 0001, iar cu ultimul impuls al fazei (i_7) se va instala Fz1 (ADUCERE). Acum calculatorul va extrage din memorie instrucţia aflată la adresa 0001, care este prima instrucţie a subprogramului de verificare a stări fanionului pentru tensiune scăzută (FANTS). Facă FANTS se află în starea unu se continuă subprogramul cu ajutorul căruia se efectuează memorarea conţinutului tuturor registrelor active. Dacă din contra FANTS se află în starea zero se trece la un alt subprogram, cu ajutorul căruia se identifică dispozitivul periferic care a produs întreruperea.

Obs. Instrucţia AITR aduce bistabilul BITR în starea unu la sfîrşitul ei, cu ajutorul impulsului i_7. Cum tot cu i_7, la sfîrşitul instrucţiei, se cercetează dacă avem cerere de întrerupere, înseamnă că existenţa unei astfel de cereri nu poate fi luată în considerare

pentru că testarea făcîndu-se cu acelaşi impuls i_7 cu care BITR este adus la unu în momentul testării BITR se mai află încă în starea zero.Prin urmare faza de INTRERUPERE va fi instalată abia la sfîrşitul următoarei instrucţii din subprogram.In consecinţă instrucţia AITR are efect întîrziat cu timpul cît durează o instrucţie.

Pentru lămurirea completă a procesului vezi instrucţiile: omite la tensiune scăzută (OTS),autorizează întreruperea (AITR) şi dezautorizează întreruperea (DITR).

Prin urmare pentru introducerea posibilităţilor de întrerupere este,în primul rînd,necesar ca la sfîrşitul fiecărei instrucţii să se testeze starea faniomului FANG şi a bistabilului pentru întrerupere BITR şi apoi să se instaleze,dacă sînt îndeplinite condiţiile, faza INTRERUPERE.

In faza INTRERUPERE se vor face următoarele: Conţinutul numărătorului de adrese NA,care reprezintă adresa instrucţiei din programul principal la care trebuie să ne întoarcem după terminarea întreruperii,îl introducem la adresa 0000 din memorie. Se aduce la zero NA şi se adaugă un unu la conţinutul lui,formîndu-se astfel adresa 0001. Aceasta din urmă este de fapt adresa la care începe efectiv subprogramul care deserveşte întreruperea.După aceasta se revine la faza ADUCERE.

Obs. Cu privire la sincronizare

Intrucît apariţia semnalului FANG nu este sincronizată cu impulsurile de tact ale calculatorului, el poate apărea întîmplător chiar în timpul unu impuls i_7 de la sfîrşitul unei instrucţii.Atunci impul-

sul de comandă,rezultat la ieşirea circuitelor logice necesare,poate fi mai mic ca amplitudine decît unul normal datorită faptului că circuitul SI nu a fost deschis complet. Cum el trebuie să efectueze atît instalarea fazei Fz6 (i7 → fz6),cît şi dezautorizarea prin aducerea bistabilului BITR în starea zero (i7 → A(o)BITR) s-ar putea ca el să îndeplinească numai una din aceste funcţii, ceea ce ar duce la eroare. De aceea se va introduce un bistabil BFANG care, în funcţie de semnalul FANG, va fi adus în starea unu cu ajutorul impulsului i_6. In acest caz impulsul de la ieşirea circuitului logic execută o singură microoperaţie anume A(1)BFANG. Prin urmare o amplitudine insuficientă nu provoacă eroare ci numai întîrzierea detecţării cererii de întrerupere cu durata unei instrucţii. In acest caz cu ajutorul lui i_7, printre altele va trebui să aducem la zero BFANG.

Capitolul 3

BLOCURILE DIN CARE E FORMAT CALCULATORUL SI BORNELE LOR DE INTRARE SI IESIRE

In cele ce urmează se prezintă blocurile din care este format calculatorul.Tipul şi numărul lor rezultă din descrierea instrucţiilor.Cu ajutorul blocurilor trebuie să putem realiza toate microoperaţiile din care sînt formate instrucţiile. Aşa cum s-a stabilit,blocurile vor fi prezentate aici numai sub formă de dreptunghiuri cu borne, respectiv se vor mai desena explicit, în schemă,circuitele SI prin care se introduce sau se extrage informaţia din diversele registre.In cadrul dreptunghiului, la registre, se scrie numărul de ranguri.

Calculatorul CETA este format din următoarele blocuri:

1. Registrul R şi registrul A, fig.3.1. Registrul R primeşte informaţie prin SCI2 mai ales din memorie,dar poate primi şi din diverse registre.Din R informaţia nu se poate preda decît în registrul A, care este registrul acumulator, în care se fac adunările. După semnul numărului din R, acesta poate fi predat complementat sau necomplementat în A. Registrul A are posibilităţi de deplasare spre stînga şi spre dreapta. El poate preda informaţie în SCIl. Cele două registre vor fi înzestrate cu un mare număr de borne,corespunzător operaţiilor aritmetice şi logice,care se execută cu ajutorul lor şi care rezultă din descrierea pentru constructor a instrucţiilor.

Registrul A mai este folosit şi pentru extragerea informaţiei din calculator.Astfel se pot trimite spre exterior cinci din rangurile cele mai semnificative.

2. Registrul Q, fig.3.2. Acest registru se utilizează mai ales la efectuarea operaţiilor de înmulţire

i împărţire. Astfel aici se păstrează înmulţitorul, respectiv cîtul. Registrul se mai utilizează şi la primirea informaţiei din exterior, anume în cinci rânguri in cele mai puţin semnificative. El are posibilităţi e deplasare la stînga şi la dreapta. El poate primi din SCI2 şi poate preda în SCI1. Registrul Q este deasemea înzestrat cu un mare număr de borne pentru executaea microoperaţiilor necesare la efectuarea instrucţii-

3. Registrul B, fig.3.3. Acesta este utilizat ca memorie rapidă pentru un singur cuvînt, care ajută la efectuarea anumitor instrucţii. El poate primi informaţie din SCI2 şi poate preda în SCI1.

4. Registrul DCR, fig.3.4. Acesta serveşte pentru indicarea apariţiei unei depăşiri, la efectuarea unor operaţii aritmetice.

5. Memoria (M) şi registrul de adrese al ei (RA),

3.5. Memoria are rolul de a păstra informaţie în ea, atît din instrucţiile şi datele care compun programe- le trebuie să permită la o comandă CITIRE, venită de la dispozitivul de comandă al calculatorului, aducerea registrul memoriei (RM), a informaţiei memorată într-o locaţie a cărei adresă se găseşte înscrisă în registrul adresei (RA). La o comandă SCRIE, venită tot de la dispozitivul de comandă al calculatorului, trebuie să se memoreze cuvîntul aflat în RM într-o locaţie a cărei adrese se află înscrisă în RA.

Registrul RA poate primi informaţie din SCI2.

6. Registrul memoriei (RM), fig.3.6. Acesta primeşte cuvîntul citit din memorie în vederea reînscrierii lui, întrucît memoria fiind cu miezuri de ferită, este cu citire distructivă. Simultan cu introducerea cuvîntului citit în RM, el se duce şi într-un alt registru.

Tot în RM se înscrie şi cuvîntul care trebuie memorat. Registrul poate primi informaţie din SCI2 şi poate preda informaţie în SCI1.

7. Dispozitivele de introducere-extragere, fig.3.7. pot fi, spre exemplu, un cititor de bandă perforată pentru introducere şi un teleimprimator pentru extragere.

Calculatorul CIFA va avea şase canale de introducere-extragere, prin urmare se vor putea conecta încă patru dispozitive pe lîngă cele două indicate mai sus. Dispozitivele de introducere-extragere se mai numesc şi dispozitive periferice. Fiecare periferic se conectează la calculator prin intermediul unor, aşa numite, circuite de interfaţă, care au rolul de a adapta semnalele periferalului la cerinţele calculatorului. Circuitele de interfaţă primesc informaţie din SCI2 şi trimit informaţie în SCI1. Circuitele de interfaţă primesc deasemenea semnale de comandă, cu ocazia executării instrucţiilor de introducere şi extragere şi sînt prevăzute, în acest scop, cu bornele necesare.

In cazul cititorului de bandă perforată, rolul circuitelor de interfaţă este acela de a transforma semnalele primite în serie de la cititor, în semnale trimise în paralel în calculator. Circuitele de interfaţă pentru teleimprimator au rolul de a transforma informaţia primită în paralel de la calculator, în informaţie serie trimisă la teleimprimator. In cazul cititorului de bandă perforată şi al teleimprimatorului, circuitele de interfaţă au fost îmbinate într-un singur ansamblu.

Toate circuitele de interfaţă posedă unul sau două circuite bistabile denumite fanioane (FANA, FANB), care sînt aduse în starea unu de către însuşi dispozitivul periferic şi indică, astfel, calculatorului că este gata să dea, sau să primească informaţii. Pentru periferic ul

cu nr.1 funcţionale se vor nota FANA 1 respectiv FANB 1, iar pentru periferiul cu nr.2, FANA 2 respectiv FANB 2. Menţionăm că atunci cînd avem un singur fanion la un anumit periferic cu nr.n el va fi notat cu FANA n.

8. Memoria fixă MF pentru introducerea programului de preintroducere, fig.3.8. Memoria se compune dintr-un numărător NAB, în care se formează adresele; un decodificator D; matricea memoriei fixe MMF, care formează memoria propriu zisă şi dintr-un circuit logic SI, cu ajutorul căruia se detectează momentul cînd se ajunge la ultima adresă.

In memoria fixă se păstrează, cum am văzut, programul de preintroducere, de unde, în mod foarte simplu, prin apăsarea pe cheia PRI, se introduce în memoria operativă.

9. Registrul instrucţiei (RI) şi decodificatorul codului operaţiei (DCO), fig.3.9. RI primeşte din memorie instrucţia, care urmează a fi executată şi care se află memorată la adresa înscrisă în registrul numărător de adrese (NA).Registrul are o parte de cod al operaţiei, care decodificat, cu ajutorul unui decodificator face ca la ieşirea acestuia să apară potenţial ridicat la bara corespunzătoare instrucţiei. Registrul mai are un bit I_a, care arată, dacă este unu, că adresarea se face indirect. O a treia parte reprezintă cîmpul adresei, în care se înscrie adresa operandului, în cazul instrucţiilor cu referire la memorie, sau biţii care indică instrucţiile de bază, în cazul instrucţiilor microprogramate.

Registrul poate primi informaţie din SCI2 şi poate preda informaţie din partea de adresă în SCI1.

Decodificatorul codului operaţiei (DCO) este format dintr-o matrice decodificatoare şi are rolul de a decodifica codul operaţiei din instrucţie în aşa fel, încît

pentru fiecare cod adus la intrarea decodificatorului, să obţinem semnal de unu, la o singură ieşire a sa. Aceste semnale servesc la formarea unor funcţii logice, care dirijează semnalele de comandă la bornele blocurilor, pentru declanşarea microoperaţiilor ce alcătuiesc instrucţia al cărui cod a fost decodificat.

lo. Numărătorul de adrese (NA), fig.3.lo, care are proprietăţi de numărare în sus. In el se formează adresa instrucţiei următoare, prin adăugarea unei unităţi la adresa curentă. NA poate primi informaţie din SCI2 şi poate da informaţie în SCIl.

ll. Generatorul impulsurilor de orologiu (Gl), bistabilul pornire-oprire (BPO), bistabilul frmulţire-împărţire (BFl), numărătorul pînă la 24 (N24), generatorul impulsurilor de tact h (H) şi generatorul de impulsuri singulare g (GS), fig.3.ll. Generatorul GI produce impulsurile de orologiu în număr de opt ($i_0 \ldots i_7$), respectiv $I_0 \ldots I_7$. Aceste impulsuri, care apar succesiv şi la ieşiri diferite, sînt duse prin diverse circuite logice la bornele unde declanşează microoperaţiile cuprinse în diversele faze, care, la rîndul lor, formează instrucţiile. El se compune dintr-un numărător (N8), o matrice decodificatoare cu diode, 8 circuite SI şi o linie de întîrziere Δ.

Generatorul H generează impulsurile de tact h, care se transformă în impulsuri de orologiu.

Generatorul singular GS serveşte la obţinerea de impulsuri g, date manual, din care, prin circuite logice se formează impulsuri IH, care apar unul cîte unul.

Bistabilul BPO serveşte la pornirea şi oprirea calculatorului, prin aceeaşi permite, respectiv nu permite, impulsurilor generatorului de tact să ajungă la generatorul impulsurilor de orologiu.

Bistabilul BII serveşte la repetarea unor impulsuri de orologiu, repetare necesară la înmulţire şi împărţire.

Numărătorul N24 serveşte la numărarea grupurilor de microoperaţii, ce se efectuează de 24 de ori la instrucţia de înmulţire, respectiv împărţire.

12. Generatorul de faze (GF), fig.3.12, care are rolul de a furniza şase semnale, corespunzătoare celor şase faze despre care am vorbit (ADC, IND, EXC1, EXC2, CHE1, INTRERUPERE). Generatorul este format din 3 circuite bistabile şi un decodificator. Cînd se aduce impuls la una din intrări ($fz_1 ... fz_6$) apare potenţial ridicat la ieşirea corespunzătoare ($Fz_1 ... Fz_6$).

13. Generatoarele de cuvinte (G1,G2), fig.3.13. Acestea sînt de fapt registre formate din comutatoare, plasate pe pupitrul de comandă. Prin aşezarea manuală a comutatoarelor putem înscrie 0 sau 1 în fiecare rang. Fiecare generator de cuvinte este format din cîte 24 de comutatoare, adică 24 de biţi. Cuvintele formate manual în generatoarele G1 şi G2, pot fi trimise fie manual, prin manipularea cheilor de pe pupitru de comandă, fie automat, prin instrucţii, în diverse registre sau în memorie. Trimiterea se face prin SGI1.

14. Bistabilele şi comutatoarele cheilor, fig.3.14-3.19. Bistabilele servesc pentru memorarea comenzii. Ele sînt aduse în starea 1 cu ajutorul cîte unui generator singular. Bistabilele cheilor sînt: BINAD, BINCA, BMEM, BOPR1, BOPR2, BPOR, BUNCICL, BVIZ, BPRI şi BMF. Cheia BMP este formată dintr-o cheie telefonică cu trei poziţii. Cheia STINIT, cheile $Fz_1 ... Fz_6$ şi cheia AZNA au cu bistabile ci numai cîte un generator singular.

15. Alte bistabile care fac parte din calculator sînt: bistabilul pentru omiteri BOM, bistabilul ataşat terminalului general BFANG şi bistabilul necesar proce-

nelor de întrerupere, fig.3.26-3.28.

16. Blocul circuitelor de comandă BCC, fig.3.29. El va asigura dirijarea semnalelor de comandă i_0-i_7 în așa fel, încît să declanșeze microoperațiile necesare efectuării instrucțiilor și comenzilor date de cheile de pe pupitrul de comandă. Schema lui va rezulta prin sinteză pe baza ecuațiilor ce se vor scrie. BCC posedă, înșumate, bornele tuturor celorlalte blocuri, ce primesc semnale de comandă, bineînțeles într-o situație inversă, deoarece bornelor de intrare din blocuri corespund borne de ieșire la BCC și invers.

17. Blocul șinelor pentru circulația informației (SCI), fig.3.3o. Cînd vorbim de circulația informației ne referim la cuvintele, care reprezintă instrucții sau date și nu la impulsurile de comandă.

In CETA circulația informației se face prin două șine ale codului informației (SCI1 și SCI2). Informația dintr-un registru poate ajunge, printr-un număr de circuite SI egal cu numărul de ranguri ale registrului, în SCI1. Circuitele SI au cîte o intrare legată la ieșirile de 1 ale bistabilelor, iar o altă intrare a fiecărui circuit este legată la o bară comună, la care se aduce un impuls ascuțit (s.ex.PdRI). Prin urmare în SCI1, care au același număr de bare cîte sînt și bistabilele, apare impuls acolo unde bistabilul corespunzător a fost pe unu, și nu apare impuls acolo unde bistabilul a fost pe zero. Fiecare rang al șinelor SCI1, și avem 24 de astfel de ranguri, se compune de fapt dintr-un circuit SAU, cu atîtea intrări cîte registre au legătură cu SCI1. Vom avea prin urmare 24 circuite SAU. Ieșirea fiecărui circuit SAU formează cîte o bară din cele 24 bare SCI2.

Din barele SCI2 informația poate trece în oricare registru, prin intermediul unor circuite SI, în număr egal cu al barelor SCI2, respectiv al bistabilelor re-

gistrului. Una din intrările circuitelor SI este legată la o bară comună la care se dă un impuls lat pentru primirea informaţiei în registru (spre ex.PmRI).

In fig.3.3o se arată schema circulaţiei informaţiei în CETA. Aici noi am prezentat un singur rang din cele 24 ale şinelor SCI1 respectiv SCI2. Se înţelege deci că în realitate, în locul unui circuit din figura 3.3o există atîtea, cîte ranguri are registrul corespunzător, în cele mai multe cazuri 24 de ranguri.

In fig.3.3o nu sînt prinse generatorul de impulsuri de orologiu, generatorul de faze, numărătorul N24 şi în general circuitele de comandă, deoarece în ele nu circulă informaţie sub formă de cuvinte ci ele furnizează impulsuri, care comandă circulaţie informaţiei formată din cuvinte..

Se vede acum uşor că pentru a trimite informaţie dintr-un registru, spre ex.B, în altul, spre ex.R, va trebui să dăm un impuls ascuţit de predare la bara PdB şi un impuls lat de primire la bara PmR, avînd grijă ca primul să se afle în intervalul de timp cît durează cel de al doilea. In felul acesta informaţia din registrul B trece, sub forma de prezenţă de impuls, respectiv lipsă de impuls, în SCI1, iar de aici, prin circuitele SAU, în barele SCI2 şi prin circuitele SI în R.

Pentru a înţelege mai uşor conectarea şinelor codului informaţiei, se prezintă în fig.3.31 schema acestora în perspectivă. Sînt prezentate numai două registre B şi Q. Pentru simplificarea figurii, acestea sînt considerate că au, în loc de 24 de biţi, numai 3 biţi. Prin urmare în realitate avem 24 circuite SAU pentru SCI1 şi 24 fire pentru SCI2. Numărul intrărilor fiecărui circuit SAU este determinat de numărul de elemente care trimit informaţie în şinele respective SCI1.

Am văzut la începutul cursului, că un calculator numeric este format din următoarele părţi: dispozitivul aritmetic, memoria, dispozitivele de introducere-extragere şi dispozitivul de comandă. Să vedem acum felul cum se repartizează blocurile din care e format calculatorul, în cazul nostru CIFA, pe aceste părţi.

La dispozitivul aritmetic aparţin: Registrul R, registrul A, registrul Q şi bistabilul DCR. Deasemenea putem considera că la dispozitivul aritmetic aparţine şi registrul B, deoarece el ajută la efectuarea operaţiilor aritmetice şi logice.

La memorie aparţin: Memoria propriu zisă (M), registrul adresei (RA) şi registrul memoriei (RM).

La dispozitivele de introducere-extragere aparţin toate perifericele cu circuitele de interfaţă aferente.

La dispozitivul de comandă aparţin: registrul instrucţiei (RI), numărătorul de adrese (NA), generatoarele de cuvinte (G1 şi G2), generatorul impulsurilor de orologiu (GI), bistabilul pornire-oprire (BPO), bistabilul înmulţire-împărţire (BII), numărătorul N24 şi generatorul de faze (GF). Deasemenea la dispozitivul de comandă aparţin toate elementele prezentate la punctele 14 şi 15. Tot la dispozitivul de comandă aparţine aşa numitul bloc al circuitelor de comandă (BCC), care conţine toate circuitele logice, prin care impulsurile de orologiu se trimit la bornele corespunzătoare pentru realizarea succesiunii de microoperaţii necesare executării diverselor faze, respectiv instrucţii. BCC este partea cea mai complexă a calculatorului şi i se va da mare atenţie la proiectare. BCC în final va rezulta din ecuaţiile scrise.

Sinele circulaţiei informaţiei, cum rezultă din însăşi denumirea lor, fac legătura între diversele blocuri,

prin ele trecînd informaţia sub formă de cuvinte (nu de semnale de comandă) de la un bloc la altul.

Menţionăm aici că şinele pentru circulaţia informaţiei se pot realiza şi altfel decît le-am realizat pentru CETA. In fig.3.32 se vede o astfel de realizare. In acest caz, se prevăd şine de la ieşirea fiecărui registru sau bloc, din care dorim să trimitem informaţie în alte registre sau blocuri. In general, la o astfel de schemă, numărul circuitelor utilizate este mai mic. Impulsurile cu care se predă informaţia dintr-un registru în altul sînt ascuţite şi sînt notate astfel încît indică registrul din care în care se predă. Astfel PdA/RM înseamnă că se predă din A în RM.

Noi am ales pentru CETA schema din fig.3.30, deoarece ea se pretează mai bine la introducerea de instrucţii noi, întrucît, dacă vrem să introducem un nou transfer de informaţie, nu sînt necesare circuite noi, ci numai semnale de comandă corespunzătoare.

In fig.3.33 este prezentată notarea rangurilor registrelor şi se indică deasemenea care din rangurile acestora sînt legate la SCI1 respectiv SCI2. Aceasta din urmă se face prin cifrele aflate dedesubtul dreptunghiului care reprezintă registrul.

Din fig.3.33 se vede că NA are 16 ranguri, ceea ce înseamnă că s-ar putea face referiri directe la adresele dintr-o memorie cu o capacitate de 2^{16}=65.536 cuvinte. Registrul de adrese RA are numărul de ranguri corespunzător capacităţii memoriei utilizate.

Tot din fig.3.33 se vede că rangurile c.m.p.s. ale tuturor registrelor sînt notate cu "0".

In cele ce urmează se prezintă schemele bloc, din care e format calculatorul CETA.

1 REGISTRUL R și REGISTRUL A

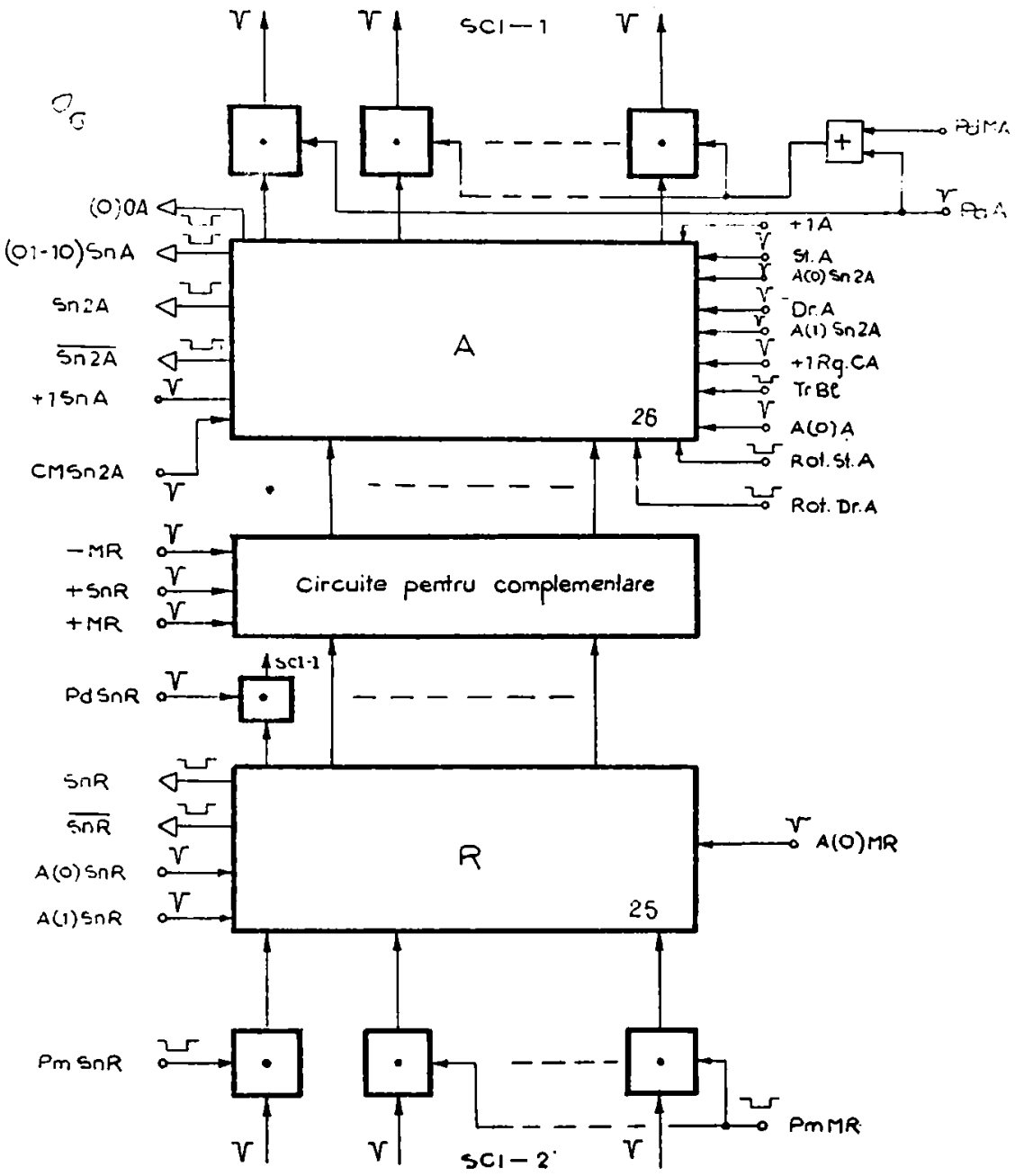


Fig. 3.1

2 REGISTRUL Q

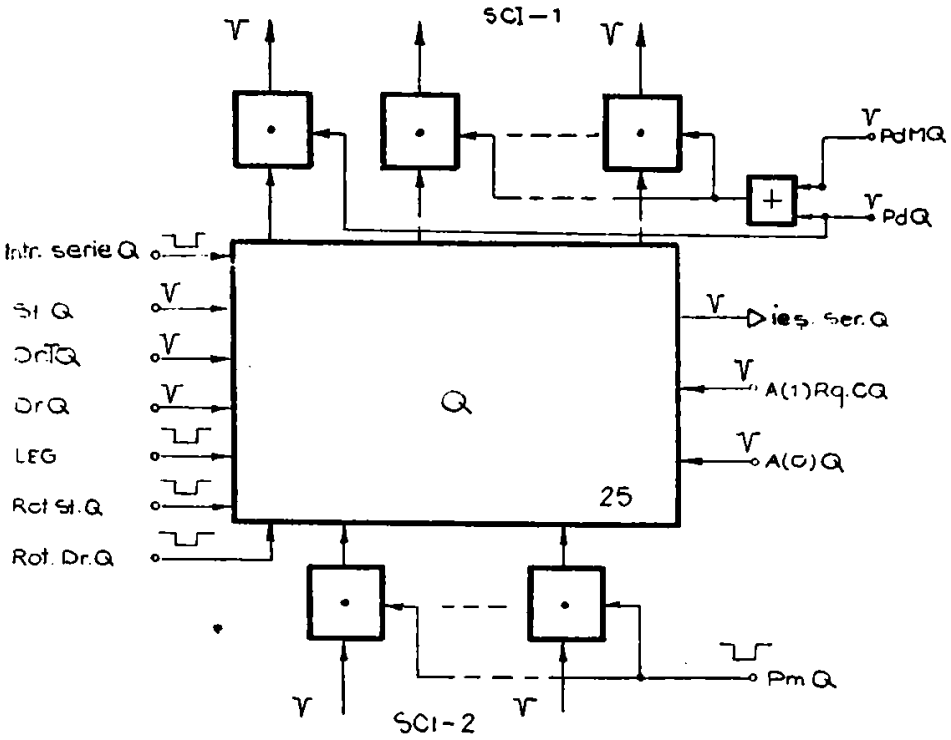


Fig. 3. 2

3 REGISTRUL B

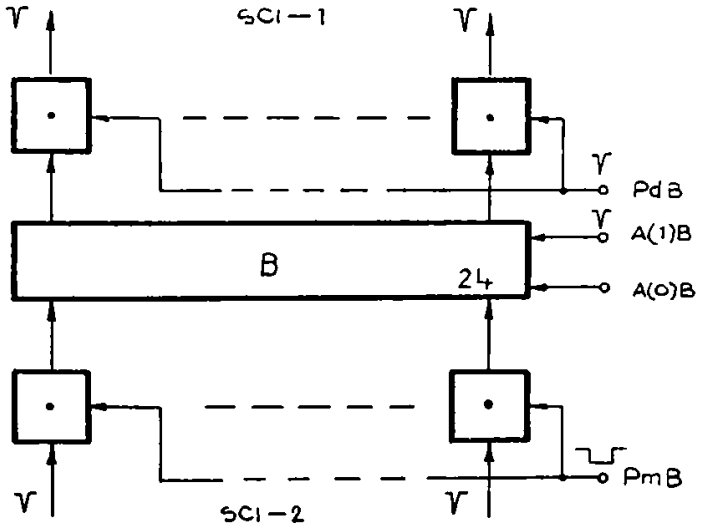


Fig. 3. 3

4. BISTABILUL DCR

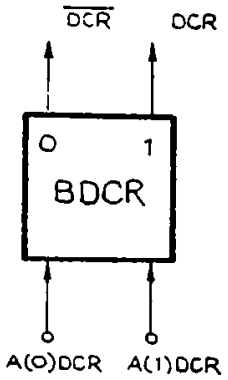


Fig. 3.4

5. REGISTRUL ADRESEI (RA) ȘI MEMORIA (M)

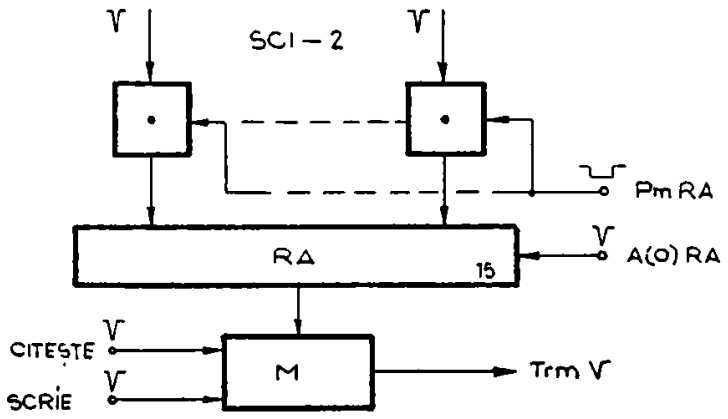


Fig 3.5

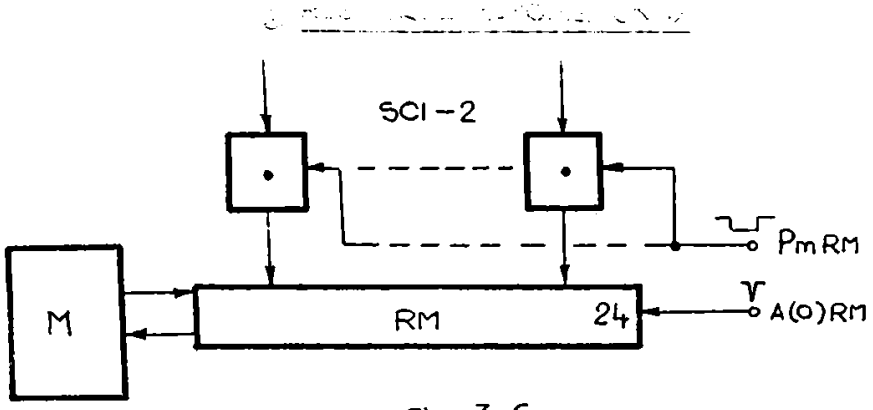
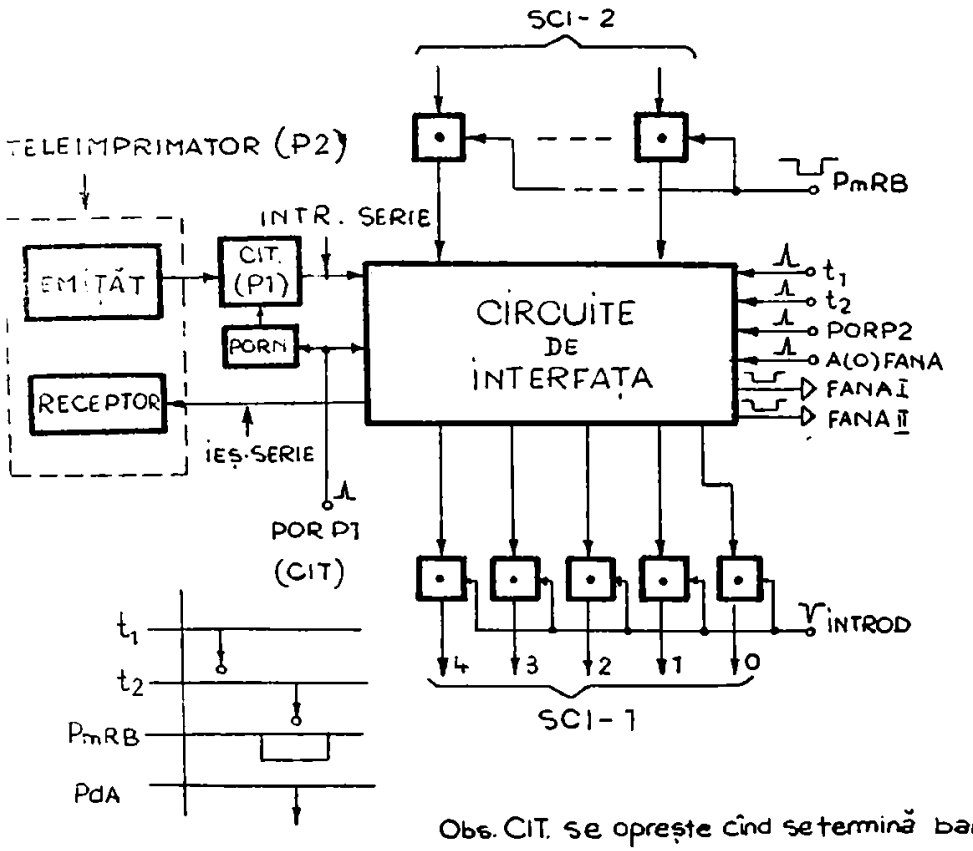


Fig. 3.6

7. DISPOZITIVELE DE ÎNTRODUCERE–EXTRAGERE



Obs. CIT. se oprește când se termină banda

Fig. 3.7

8. MEMORIA FIXĂ PENTRU INTRODUCEREA PROGRAMULUI DE PREINTRODUCERE

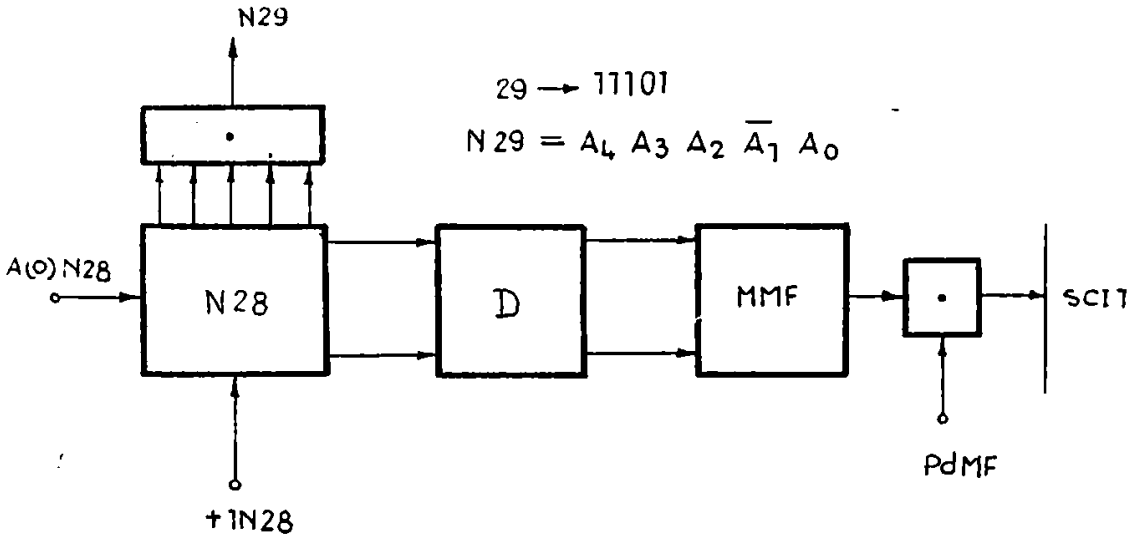


Fig. 3.8

9. REGISTRUL INSTRUCȚIEI (RI) și DECODIFICATORUL CODULUI OPERAȚIEI

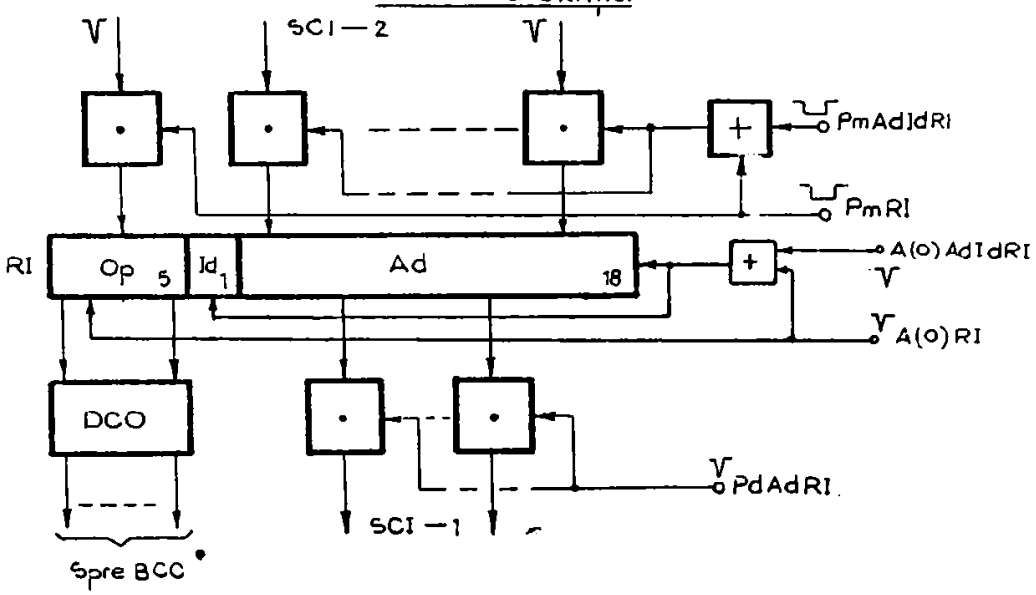


Fig. 3.9

10. NUMĂRĂTORUL DE ADRESE (NA)

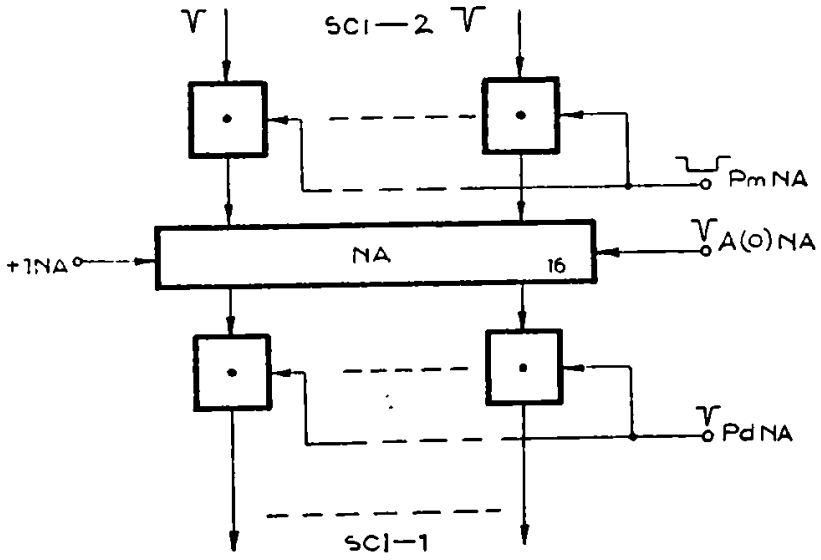


Fig. 3.10

GENERATORUL IMPULSURILOR i (GI) BISTABILUL PORNIRE OPRIRE (BPO), BISTABILUL ÎNMULȚIRE - ÎMPARTIRE (BII) NUMĂRATORUL PÎNA LA 24 (N24), GENERATORUL DE IMPULSURI DE OROLOGIU (H) ȘI GENERATORUL DE IMPULSURI SINGULARE (GS).

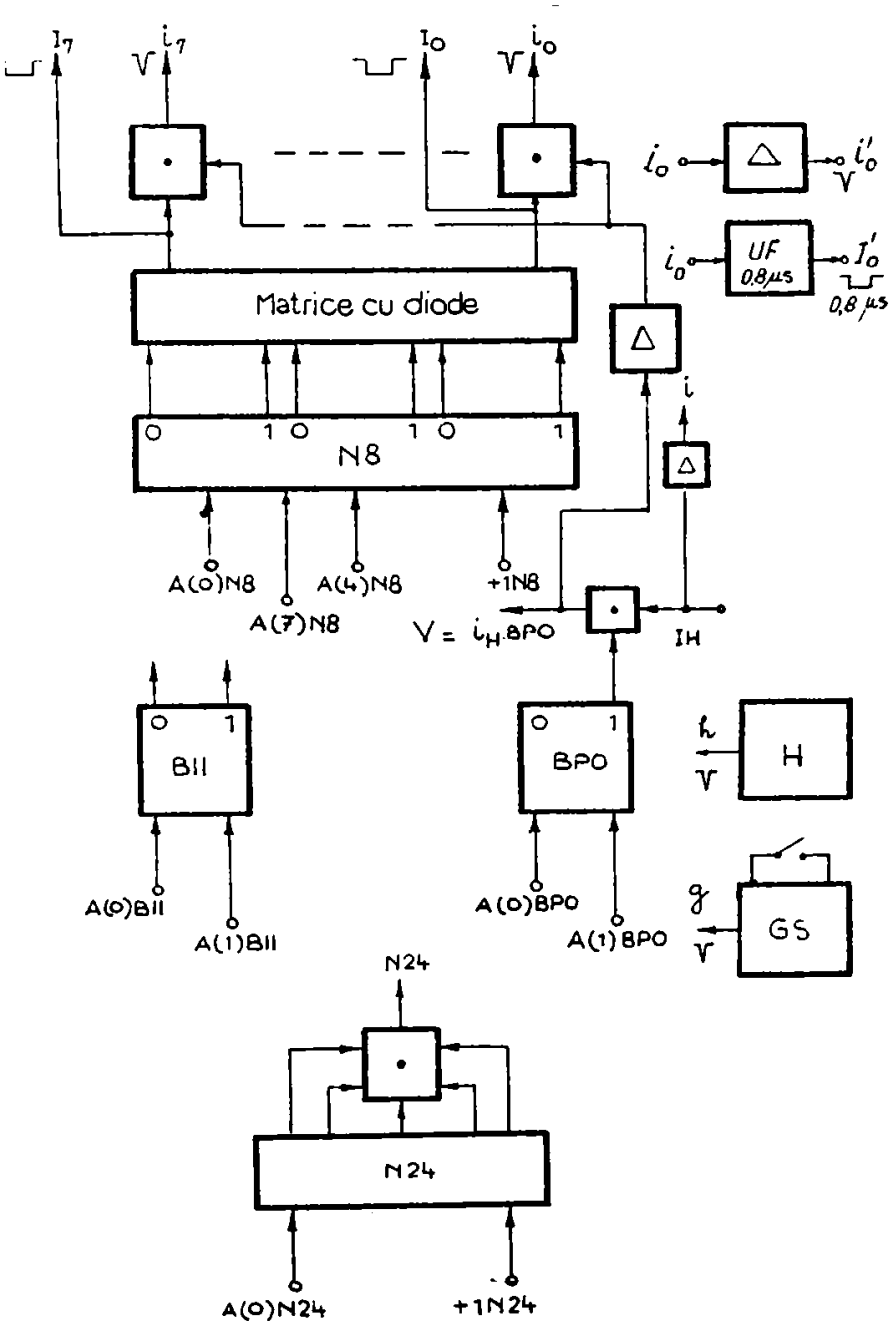


Fig. 3.11

12 SCHEMA GENERATORULUI DE FAZE GF, (Adc, Ind, Exc1, Exc2, CHEI, Intrerupere)

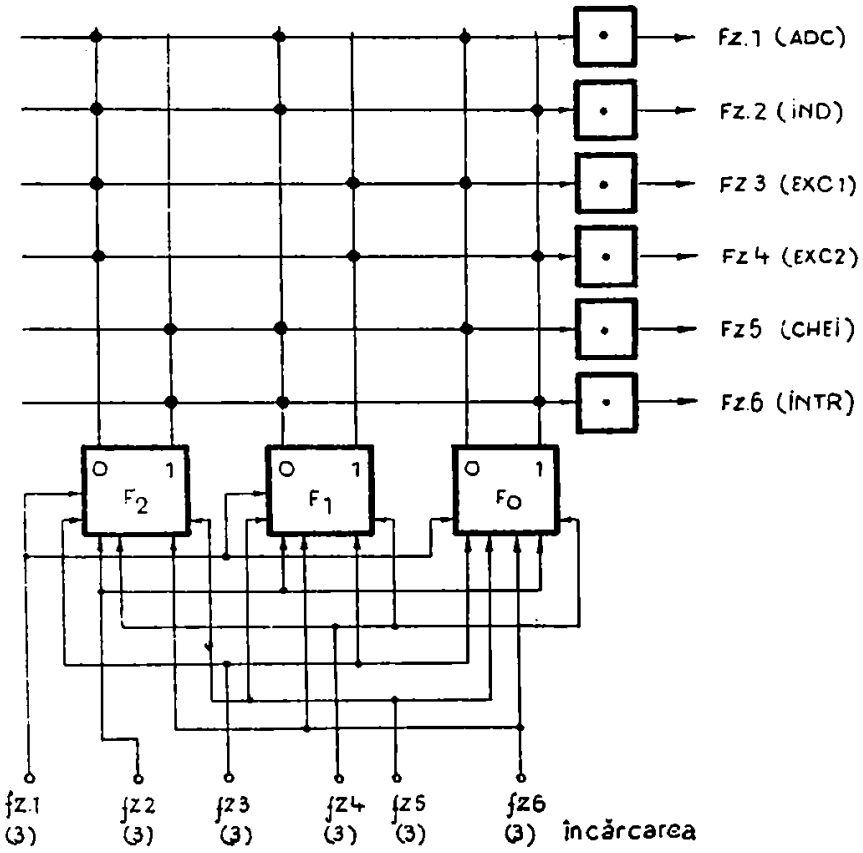
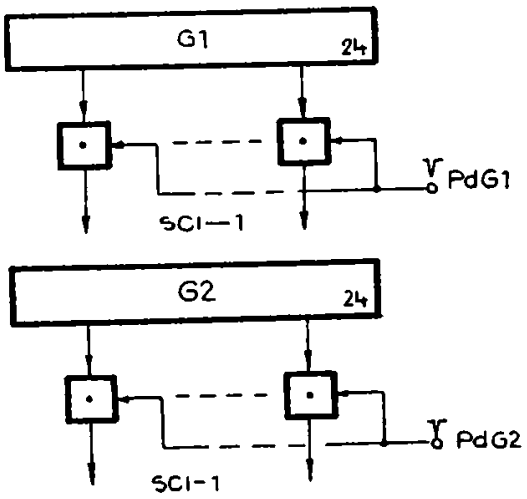


Fig. 3.12.

13 GENERATOARELE DE CUVINTE (G1 și G2)



14. BISTABILELE, COMUTATOARELE ŞI GENERATOARELE SINGULARE ALE CHEILOR

cheia ST INIT

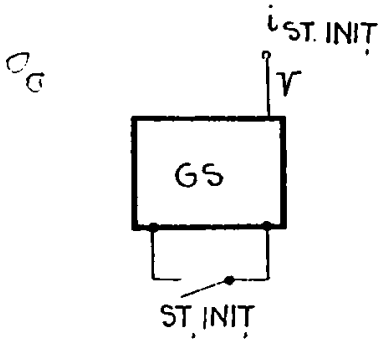


Fig. 3.14

cheia POR

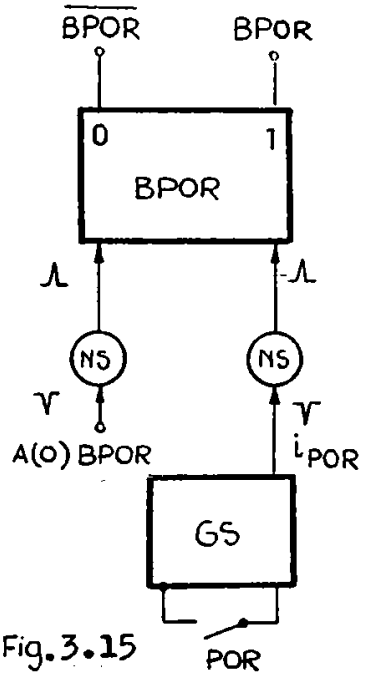


Fig. 3.15

cheia OPR

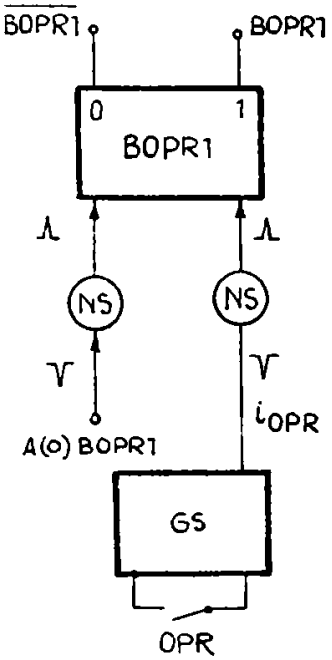
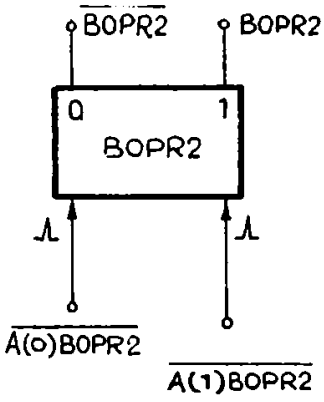


Fig. 3.16



cheia MEM

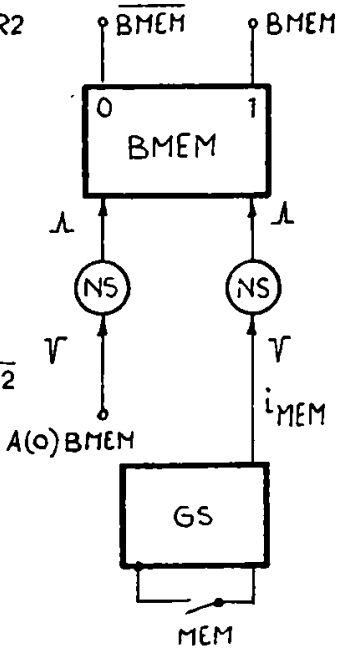


Fig. 3.17

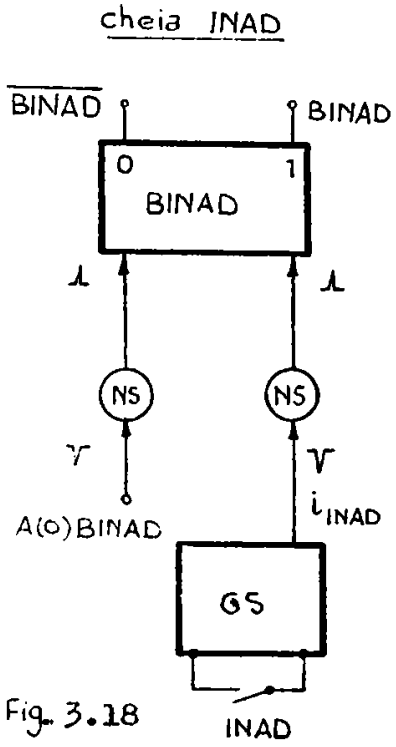


Fig. 3.18

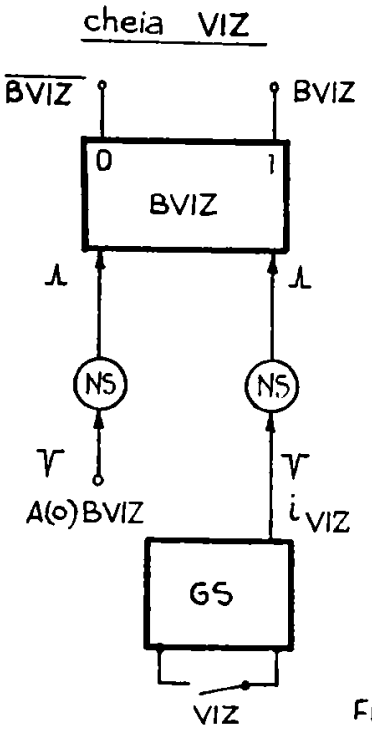


Fig.3.19

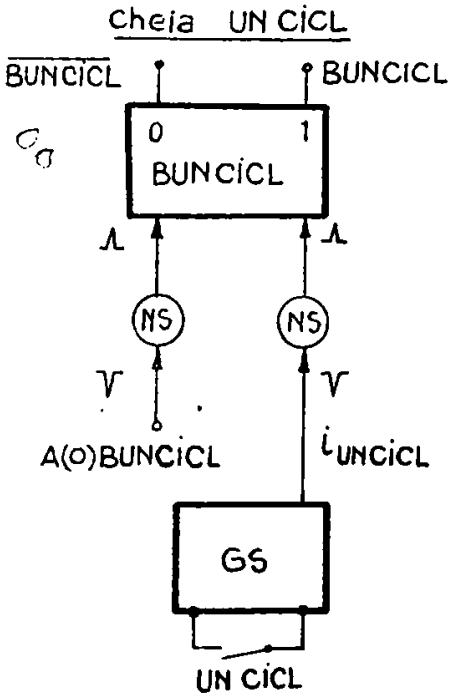


Fig.3.20

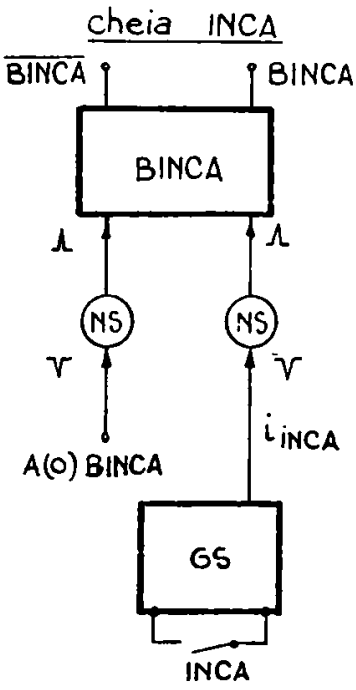


Fig.3.21

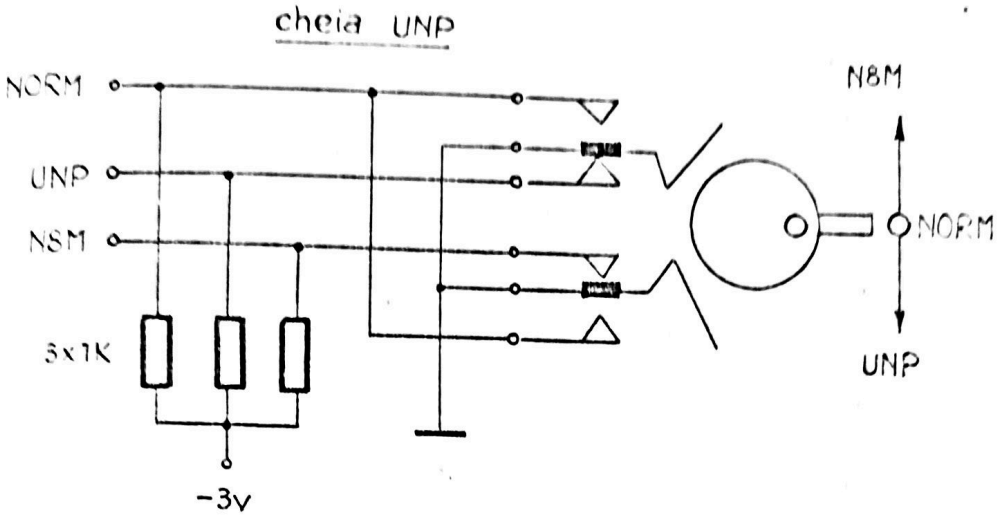


Fig. 3.22

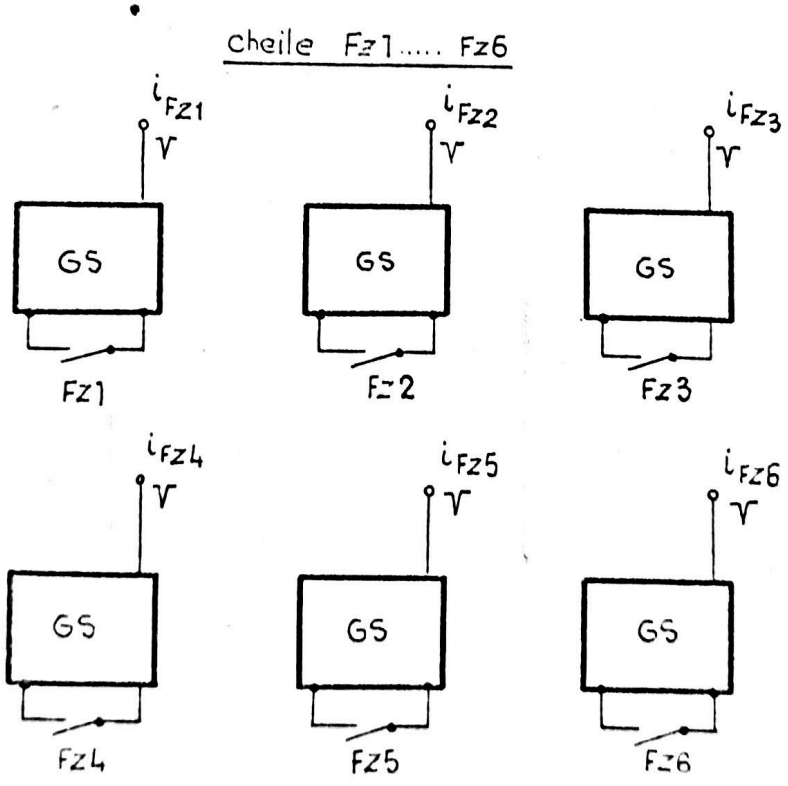


Fig. 3.23

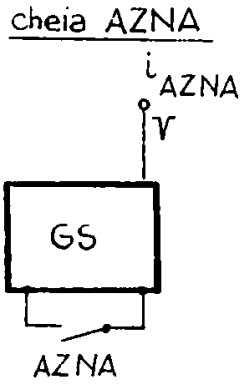


Fig. 3.24

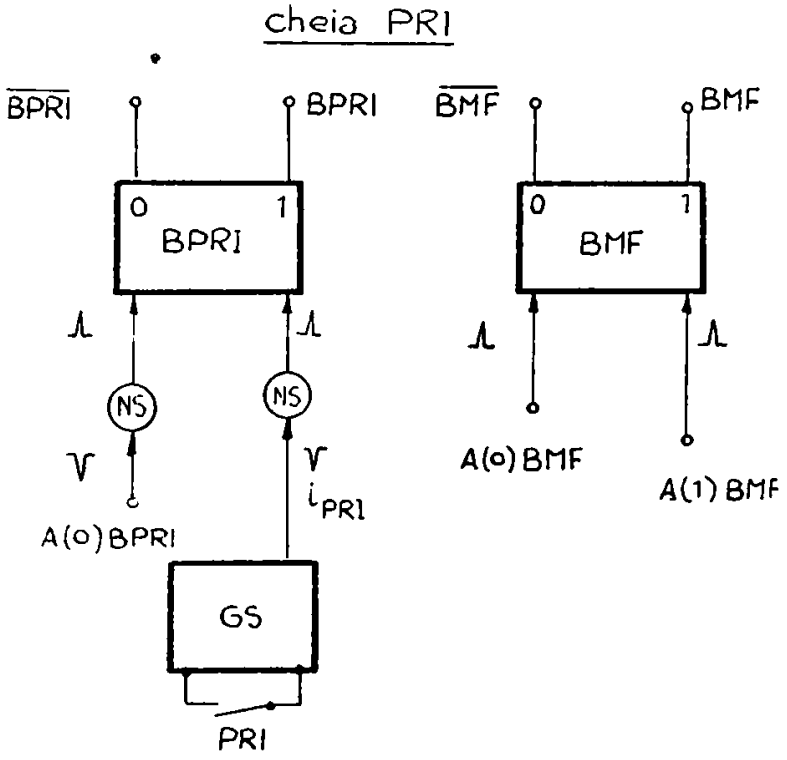


Fig. 3.25

15. ALTE BISTABILE

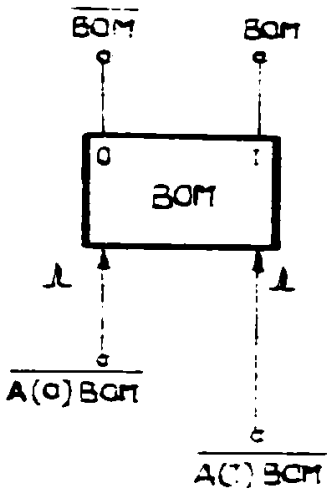


Fig. 3.26

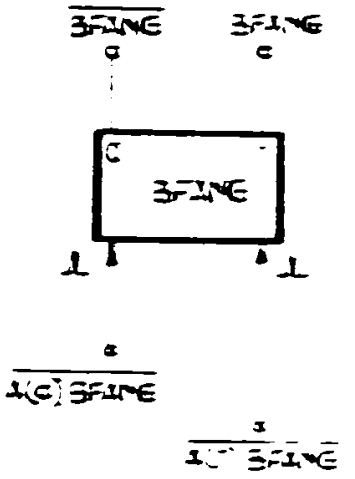


Fig. 3.27

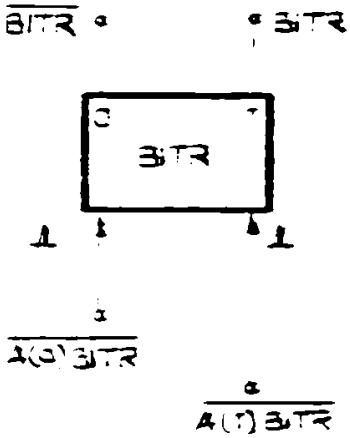


Fig. 3.28

16. BLOCUL CIRCUITELOR DE COMANDĂ

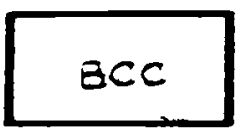


Fig. 3.29

17. Sinele circulatiei informatiei (SCI

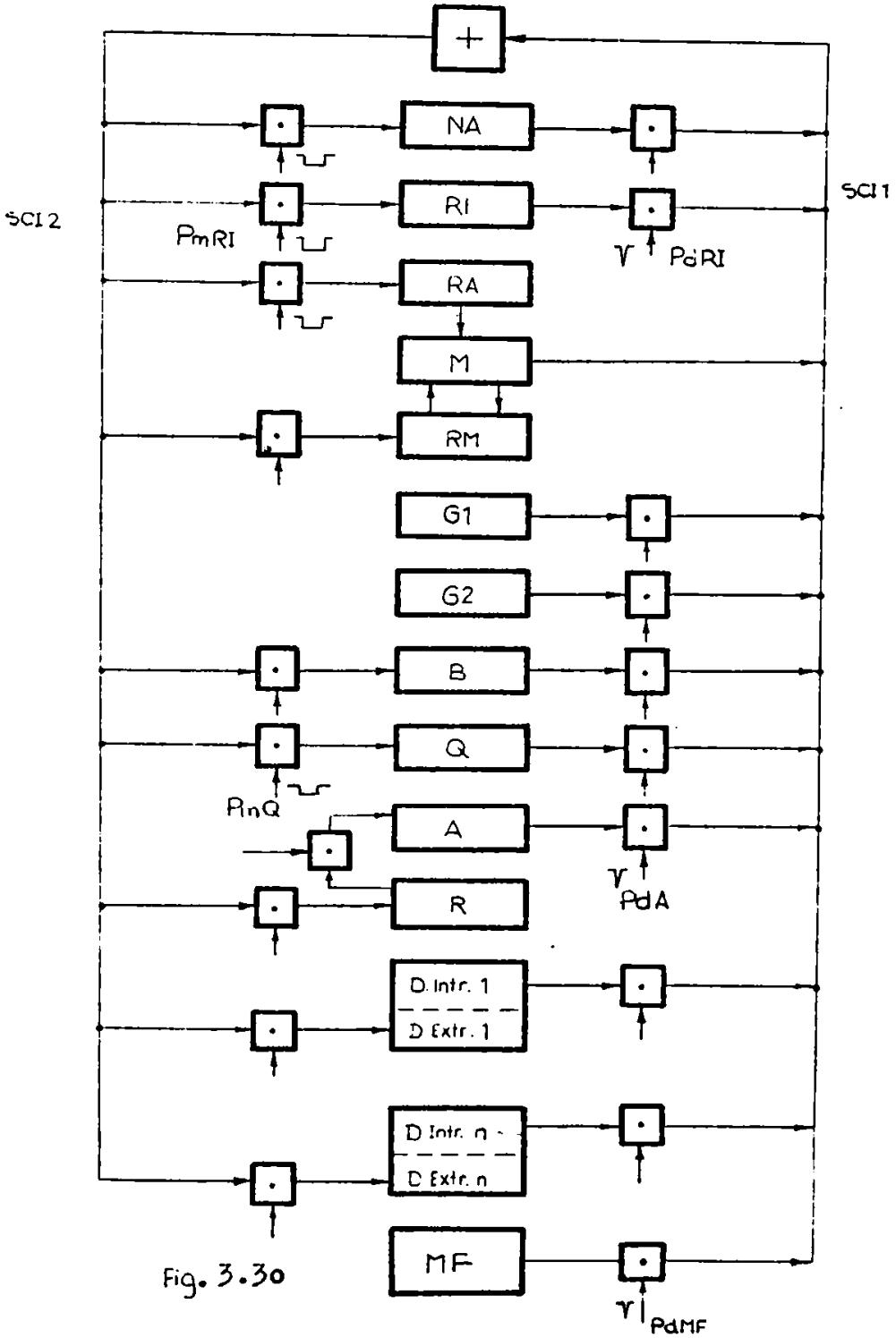


Fig. 3.30

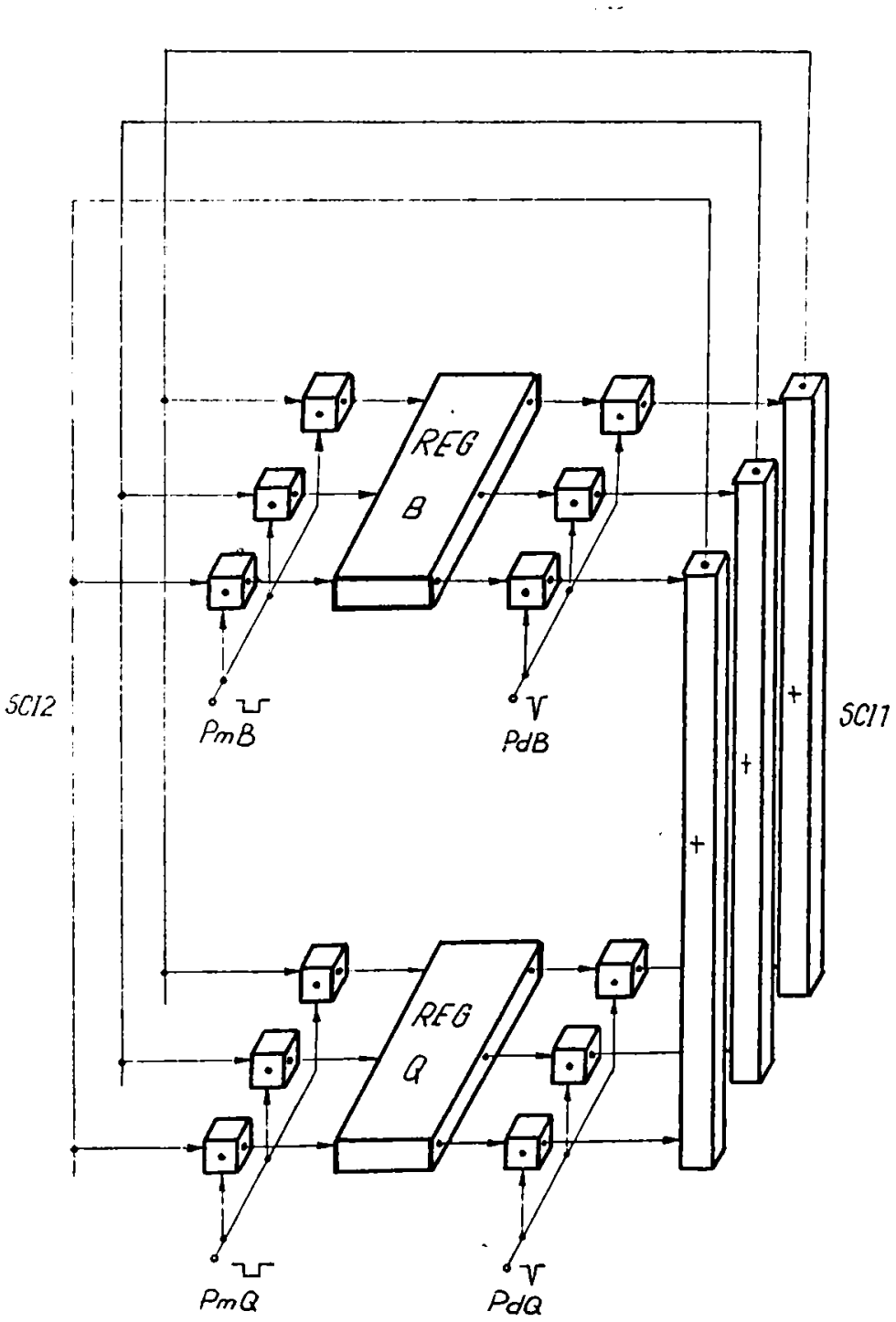
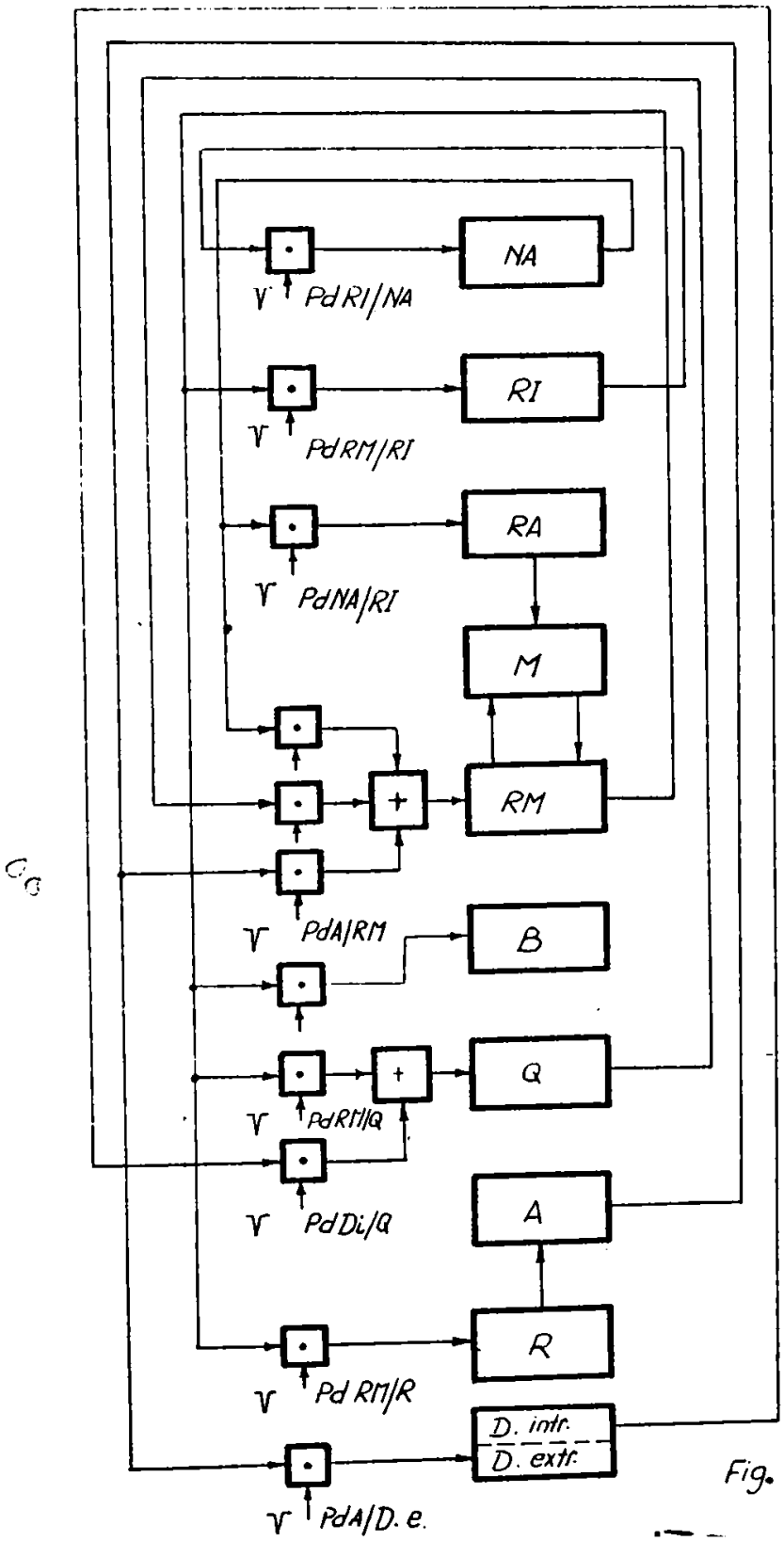


Fig. 3.31



Obs Schema serveste numai pt. ilustrarea metodei

Fig. 3.32

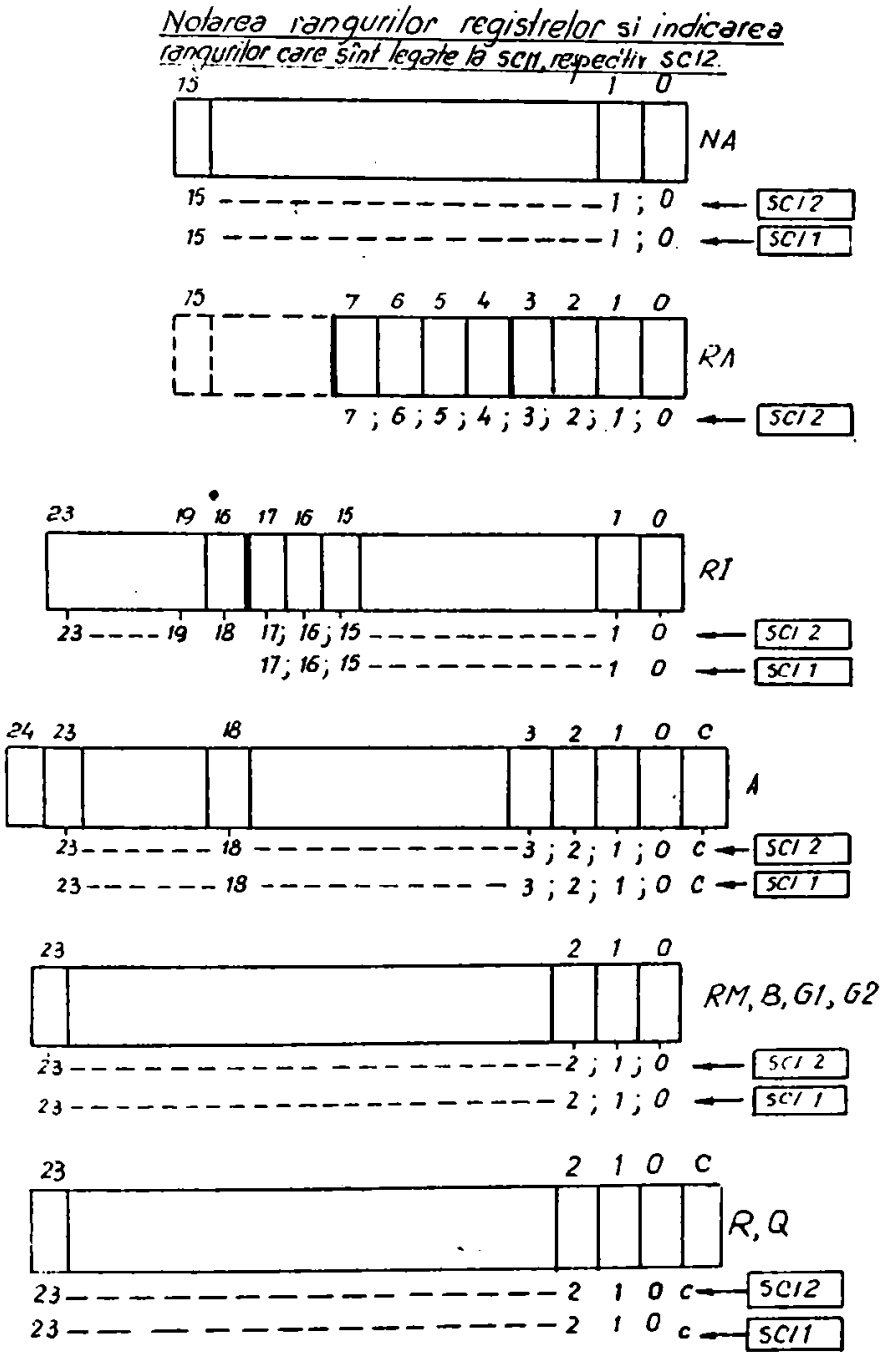

Notarea rangurilor registrelor si indicarea rangurilor care sînt legate la SCI1, respectiv SCI2.
NA
RA
RI
A
RM, B, G1, G2
R, Q
SCI 2
SCI 1
SCI2
SCI1

Fig. 3.33

Capitolul 4

DESCRIEREA TUTUROR FAZELOR INSTRUCTIILOR SI A CHEILOR DE PE PUPITRUL DE COMANDA, CU AJUTORUL TABELELOR

Stabilirea unor intervale de timp importante pentru funcţionarea calculatorului

Inainte de a trece la întocmirea tabelelor de ere, să stabilim unele intervale de timp necesare cînd alcătuim tabelele.

Ciclul memoriei este intervalul de timp, care trece momentul cînd se dă o comandă de scriere sau citi pînă în momentul cînd astfel de comenzi pot fi din nou. În cazul nostru memoria este dată şi ea ciclu de 7,5 microsecunde.

Ciclul calculatorului este intervalul de timp, în e desfăşoară o secvenţă de impulsuri de orologiu. um am văzut ciclul calculatorului trebuie să fie re sau egal cu ciclul memoriei. In felul acesta în unui ciclu al calculatorului se poate face o sincriere în, sau citire din memorie.

Ţinînd seama de circuitele logice care vor fi folosi care vor avea o frecvenţă de 2 megaherţi - e înseamnă că se poate lucra cu ele la impulsuri t care au o frecvenţă pînă la 2 megaherţi - şi l seama de faptul, că o seamă de instrucţii sînt . de complexe, stabilim ca într-o secvenţă de inşi, respectiv într-un ciclu al calculatorului, să 3 impulsuri de orologiu. Prin urmare intervalul de între ele va fi de o microsecundă. Mai menţionăm ata unui impuls ascuţit, necesar pentru comanda lelor, va fi de 0,2 microsecunde.

În figura 4.1, se arată diagrama de timp a impulsurilor de orologiu. Se vede de aici că avem impulsuri ascuţite i (cu durata de 0,2 microsecunde) şi impulsuri late I (cu durata de 1 microsecundă). Impulsurile i cad, în timp, la mijlocul duratei impulsurilor late I. Atît impulsurile i, cît şi impulsurile I, apar fiecare la ieşiri diferite şi la intervale de timp de 1 microsecundă. Generatorul impulsurilor de orologiu va avea deci, după cum s-a arătat şi cu ocazia prezentării blocurilor calculatorului (fig.3.11), opt ieşiri pentru impulsurile $i_0 \ldots i_7$ şi alte 8 ieşiri pentru impulsurile $I_0 \ldots I_7$.

Motivul pentru care am plasat, în timp, impulsurile $i_0 \ldots i_7$ la mijlocul duratei impulsurilor $I_0 \ldots I_7$ reiese din cele de mai jos.

Să presupunem că dorim să trecem informaţia din registrul NA în registrul RA. Pentru aceasta trebuie să avem funcţia PdNA, care este reprezentată de un impuls ascuţit, şi funcţia PmRA, care e reprezentată de un impuls lat (vezi fig.3.30). Pentru ca transferul informaţiei să se efectueze corect este necesar ca cele două funcţii să coincidă, ceace înseamnă că impulsurile care ar reprezenta informaţia din NA trebuie să găsească deschise porţile de intrare în RA. Să presupunem că expresiile logice a celor două funcţii sînt:

$$PdNA = i_0 . Fz1 . IRMlo + i_1 . Fz3 . SSP + \ldots$$

$$PmRA = I_0 . Fz1 . IRMlo + I_0 . Fz3 . IRM5 + \ldots$$

Mai presupunem că avem IRMlo = 1 şi Fz1 = 1. Aceasta înseamnă că atunci cînd apar i_0 şi I_0, adică avem i_0=1 şi I_0=1 funcţiile PdNA şi PmRA devin fiecare egale cu 1, datorită faptului că primii termeni ai lor sînt egali cu 1. Funcţiile PdNA şi PmRA sînt executate de scheme logice care conţin funcţii SI-SAU realizate cu două

etaje de circuite logice. Formarea funcţiilor este afectată de întîrzieri, datorită timpului finit de comutare a circuitelor logice. Cum întîrzierile diverselor circuite nu sînt identice înseamnă că funcţia PdNA, fig. 4.3, poate fi întîrziată cu τ_1, faţă de i_0, iar funcţia PmRA cu τ_2 faţă de I_0. După cum avem $\tau_1 > \tau_2$ sau $\tau_1 < \tau_2$, una din funcţii apare înaintea celeilalte. Ca să avem totuşi coincidenţa necesară, funcţia PmRA este realizată cu ajutorul impulsului lat I_0. Se vede că în cazul nostru toleranţa cea mai mare cu privire la diferenţele de întîrzieri a circuitelor, aşa fel încît coincidenţa să fie totuşi asigurată, se poate admite cînd impulsurile $i_0 \ldots i_7$ cad, în timp, la mijlocul duratei impulsurilor $I_0 \ldots I_7$.

In fig.4.3 este prezentată diagrama de timp a impulsurilor din memorie, împreună cu impulsurile de ceas i_0, i'_0, I_0 şi I'_0, care comandă microoperaţii legate de memorie.

Impulsurile i'_0 şi I'_0 sînt utilizate în scopul de a mări viteza de executare a instrucţiilor, în felul următor. Comanda de citire (CIT) sau de scriere (SCR) se va forma cu ajutorul impulsului i_0, spre exemplu în faza Fzl a unei anumite instrucţii. Citirea trebuie să se facă de la o anumită adresă din memorie. Codul acestuia se află la dispoziţie într-un registru de unde ea trebuie dusă în registrul de adrese al memoriei (RA). Pentru aceasta trebuie mai întîi să aducem la zero RA, ceea ce realizăm cu ajutorul unei funcţii formată cu I_0. După liniştirea fenomenelor tranzitorii provocate de aducerea la zero, ducem adresa, spre exemplu, din NA, în RA făcînd funcţiile PdNA şi PmRA. Acestea le-am putea realiza cu i_1 respectiv I_1, ceace înseamnă că trebuie să aşteptăm o microsecundă. Dar aducerea la zero a lui RA

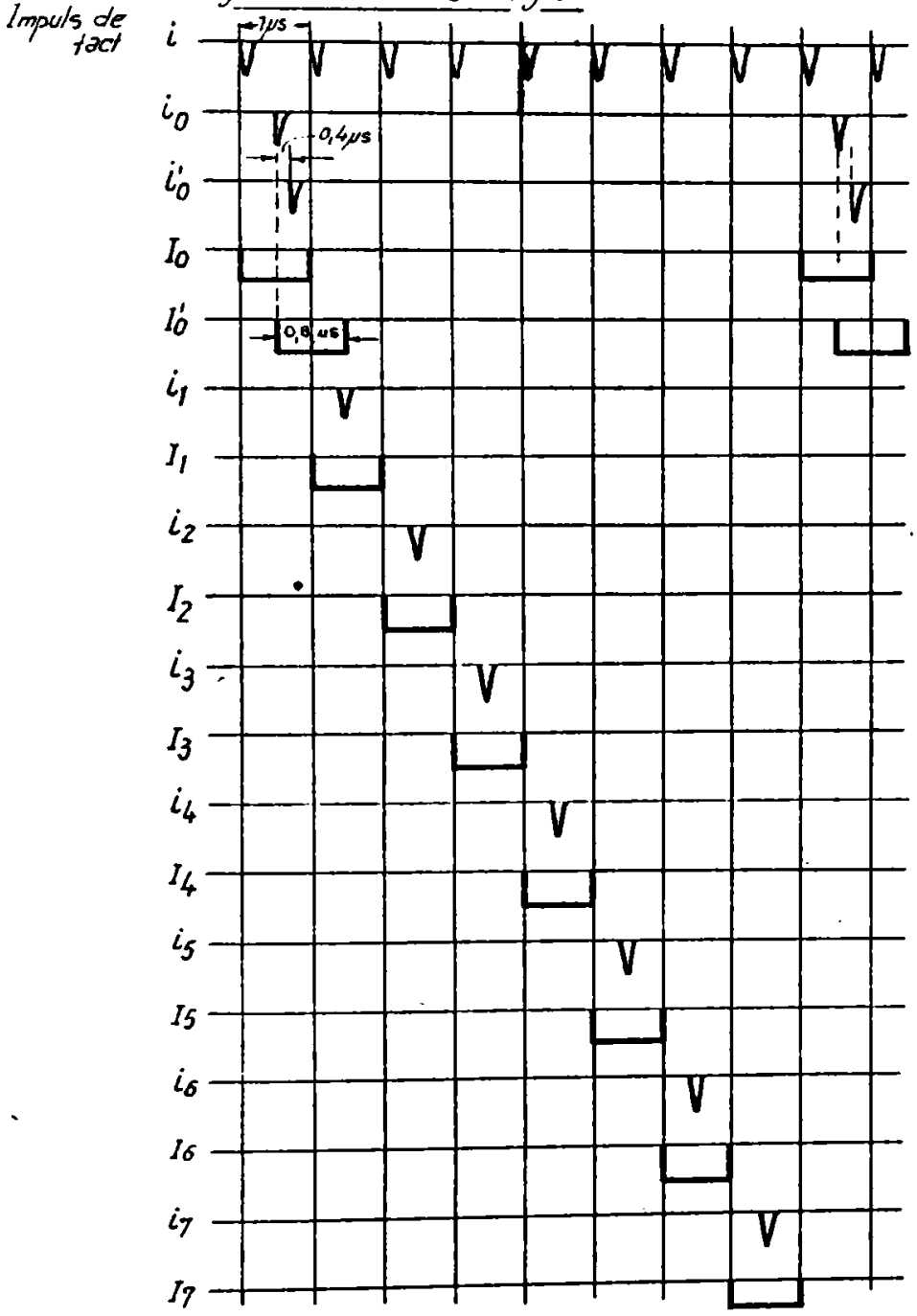
nu durează mai mult de 0,3-0,4 microsecunde, iar restul timpului e pierdut.

Pentru a economisi acest timp formăm impulsul i'_0, întîrziat cu 0,4 microsecunde față de i_0, și impulsul I'_0, întîrziat cu 0,5 microsecunde față de I_0 și lat de numai 0,2 microsecunde. Cu i'_0 formăm funcția PdNA și cu I'_0 funcția PmRA. In felul acesta putem aduce mai devreme adresa în RA și deci putem declanșa mai devreme ciclul de semiexcitare de citire în memorie, ceace face ca informația citită să fie accesibilă după timpul de 3 microsecunde (timpul de acces). Altfel timpul de acces ar fi rezultat mai lung.

Această metodă o vom folosi, după cum se va vedea, cu ocazia descrierii fazelor instrucțiilor cu ajutorul tabelelor. Bineînțeles trebuie să fim atenți ca atunci cînd formăm cu i'_0 și I'_0 funcțiile PdNA și PmRA să nu formăm funcții similare de predare sau primire pentru alte registre cu ajutorul impulsurilor i_0 și I_0, deoarece, avînd în vedere suprapunerea parțială în timp a impulsurilor I_0 și I'_0, precum și apropierea impulsurilor i_0 și i'_0, informația aflată în alt registru decît NA ar putea să intre în RA, sau cele două genuri de informație să se amestece în SCI, ceace ar duce la erori. Se va vedea că la descrierea prin tabele acest lucru se evită.

Informația citită din memorie va fi trimisă în registrul memoriei (RM), dar simultan va fi trimisă și în ținele codului informației (SCIl) în scopul de a putea ajunge direct în registrul de destinație, fără să treacă prin RM. Aceasta duce la un cîștig de timp, suplimentar, adică la mărirea din nou a vitezei de lucru. Tot din diagramă se vede că pentru a se putea înscrie corect un cuvînt de memorie, el trebuie să se găsească

Diagrama de timp a impulsurilor i date de generatorul de orologiu



Obs. In calculator se folosesc și impulsuri înfirziate față de imp. C_C i_0-i_7. Aceste impulsuri sînt notate cu $i'_0 \cdots i'_7$, și I'_0. In fig. din motive de economie de spațiu se prezintă numai impulsul i'_0 și I'_0

Fig. 4.1

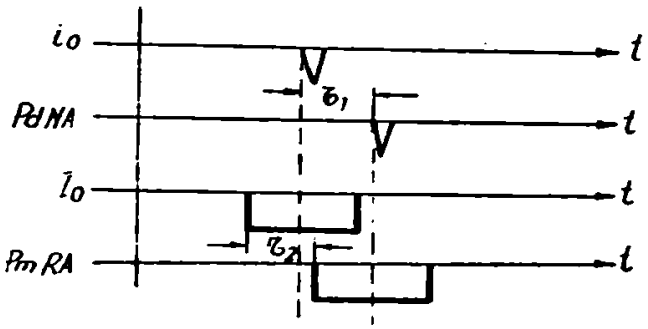


Fig. 4.2

Diagrama de timp a impulsurilor din memorie împreună cu cifrele impulsuri de orologiu

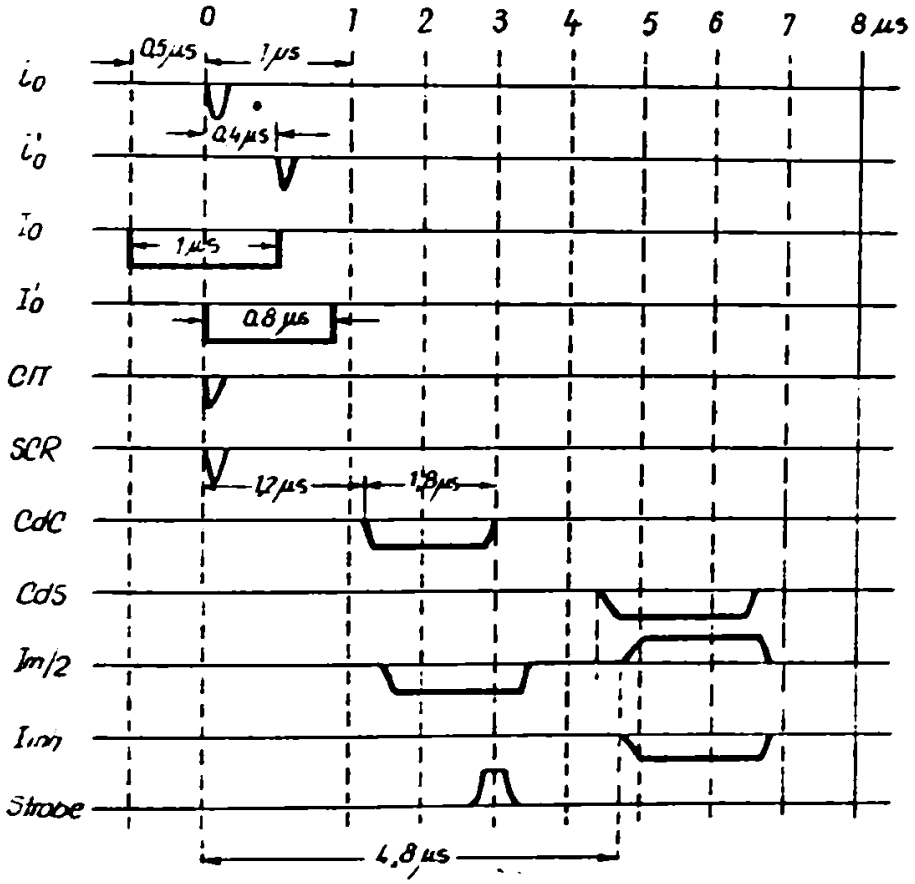


Fig. 4.3

comenzii de scriere.In acest caz impulsul de semiexcitare de scrie, care apare după 4,8 microsecunde de la darea comenzii, găseşte informaţia în RM. In diagramă se arată şi momentele cînd apar impulsurile de orologiu i respectiv I în raport cu impulsurile din memorie. Deasemenea sînt arătate şi forma impulsurilor CdC şi CdS, care comandă apariţia curenţilor de semiexcitare de citire şi scriere $I_m/2$. Tot în diagramă este prezentat impulsul de„strobe"care deschide amplificatorul de citire în momentul cel mai favorabil citirii, adică atunci cînd raportul semnal-zgomot este maxim.

In cazul nostru mai cunoaştem şi timpul maxim de adunare, care poate fi atins, în cazul cel mai defavorabil,de către registrul acumulator A. Anume el este $t_{ad.max}$=2 microsecunde. Acest timp este bine corelat cu ciclul memoriei şi al calculatorului. Va trebui deci să avem grijă ca din momentul dării comenzii, pentru funcţionarea acumulatorului, să lăsăm un interval de timp de cel puţin 2 microsecunde pînă în momentul utilizării rezultatului obţinut în acumulator.

Intocmirea tabelelor de descriere

După cum am văzut la prezentarea metodei de sinteză a schemei logice a unui calculator numeric, procesele care se petrec în timpul efectuării unei instrucţii, sau al unei comenzi date manual prin cheile de pe pupitru de comandă, vor fi descrise, în mod concis şi riguros, cu ajutorul unor tabele. Tabelele sînt întocmite pe instrucţii şi pe faze. La CETA avem 6 faze:

1. Faza ADUCERE (Fz1), în timpul căreia se aduce instrucţia din memorie în registrul instrucţiei. Primele patru impulsuri i_0 la i_3, cu care se face aducerea, execută aceleaşi microoperaţii absolut la toate instrucţiile. Unele instrucţii, cum sînt cele cu referire la

registre (microprogramate), şi cele de introducere extragere, se execută în întregime în această fază.

2. Faza INDIRECT (Fz2), în care se execută microoperaţii necesare atunci cînd prin bitul Id, din instrucţie, arătăm că adresa trecută în cîmpul de adrese al instrucţiei nu este adresa operandului, ci adresa adresei operandului.

3. Faza EXECUTIE (Fz3), în care au loc microoperaţiile necesare pentru terminarea instrucţiilor, care nu se pot termina în faza de ADUCERE. Aceste instrucţii sînt cele cu referire la memorie.

4. Faza EXECUTIE SUPLIMENTARA (Fz4), în care se execută microoperaţiile pentru instrucţiile mai lungi cum sînt IMP, MOZ, şi CBO, care nu pot fi terminate nici în faza EXECUTIE.

5. Faza pentru CHEI (Fz5), în care se execută microoperaţiile comandate manual prin apăsarea cheilor de pe pupitrul de comandă.

6. Faza INTRERUPERE (Fz6), în care se execută microoperaţiile necesare atunci cînd un echipament periferic solicită întreruperea, iar calculatorul o acceptă.

In tabele în capul coloanelor se trec impulsurile de orologiu $i_0 \ldots i_7$ (respectiv $I_0 \ldots I_7$). In coloanele corespunzătoare acestor impulsuri se arată prin săgeţi unde trebuiesc ele duse, pentru a efectua microoperaţiile necesare în faza respectivă. Toate aceste indicaţii se fac prin notaţii mnemonice, care reprezintă de fapt un limbaj simbolic foarte concis. In partea de jos a coloanelor se fac o seamă de comentarii asupra microoperaţiilor.

Completarea tabelelor se face pornind de la descrierea, pentru constructor, a instrucţiilor, descriere care are la bază algoritmul adoptat pentru execuţia

instrucţiei respective.

Se va da o atenţie deosebită respectării intervalelor de timp precizate înainte astfel:

- Intervalul de timp între două impulsuri de orologiu i este de 1 microsecundă.

- Timpul maxim de adunare este de 2 microsecunde.

- Ciclul calculatorului (durata secvenţei de opt impulsuri i) este de opt microsecunde şi este cu ceva mai mare decît ciclul memoriei (7,5 microsecunde).

- Timpul de acces al memoriei este de 3 microsecunde. Se va avea grijă de faptul că, pentru a obţine viteză mai mare, cuvîntul citit din memorie se va primi direct în registrul de destinaţie, simultan cu primirea lui în registrul memoriei (RM).

- Cuvîntul, care se memorează, trebuie să fie adus în RM cel mai tîrziu cu impulsul de orologiu i_4, atunci cînd comandă de scriere a fost dată cu impulsul i_0 al aceleaşi faze.

In tabele se va arăta întotdeauna denumirea mnemonică a instrucţiei şi fazele din care este formată instrucţia. Denumirea mnemonică va fi ulterior, la scrierea ecuaţiilor, folosită ca variabilă, rezultată din decodificarea codului operaţiei.

In cazul instrucţiilor de referire la registre, tabelele capătă o formă diferită de cele pentru instrucţiile cu referire la memorie. Aceasta rezultă din faptul că instrucţiile cu referire la registre se execută cu microoperaţii declanşate de un singur impuls de orologiu şi numai în caz excepţional de către două impulsuri.

La întocmirea tabelelor, corespunzătoare comenzilor date manual prin chei, se va ţine seamă că, la apăsarea pe o cheie, se pune în funcţiune un generator de impulsuri singulare, care furnizează la ieşirea sa un singur impuls format, negativ, adică cu o lăţime de o,2

microsecunde şi o amplitudine de cca 3V (de la -o,5V la -3,5V).

Avînd în vedere că impulsurile singulare apar asincron faţă de impulsurile de tact (care devin apoi impulsuri de orologiu), se va da o atenţie deosebită aspectelor de sincronizare necesare. Deasemenea se va da atenţie fazei Fz3, care are rolul de a asigura, ca atunci cînd apăsăm cheia, să se efectueze numai microoperaţiile necesare comenzii cerute. Menţionăm aici că în unele cazuri impulsurile i_0-i_7 nu sînt suficiente. In aceste cazuri se crează impulsuri i obţinute prin trecerea impulsurilor i_0-i_7 prin linii de întîrziere. Obţinem astfel, spre exemplu din i_2 un impuls i'_2 întîrziat faţă de primul spre exemplu cu o.3 microsecunde. In felul acesta mai putem, în intervalul de timp dintre două impulsuri i învecinate, să realizăm două microoperaţii succesive, bineînţeles dacă acestea pot fi executate în timpul mai scurt decît o microsecundă, care ne stă la dispoziţie.

In cele ce urmează se prezintă tabelele.

INSTR. RM | TOATE INSTR. RM AFARĂ DE SLT | (ADUCERE, INDIRECT) CETA

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE Fz.1	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i'_0 \rightarrow$ PcNA $I'_6 \rightarrow$ PmRA	$i_1 \rightarrow$ A(0)RI	—	$I_3 \rightarrow$ PmRI	←	—	$i_6 \rightarrow +1$ NA	Dacă $\overline{Id} = 1$ $i_7 \rightarrow fz_3$ Dacă $Id = 1$ $i_7 \rightarrow fz_2$
	- se dă cda. CITEȘTE - se aduce la zero RA și RM - se primește adresa din NA în RA	- Se aduce la zero RI pt. a primi instrucțiunea citită din M	Nu se folosește	Se primește în RI instr. citită, după trecerea timpului de acces de 3 μs	Nu se folosește	Nu se folosește	Se formează adresa instr. următoare prin adunarea lui 1 la NA	Se trece în cazul adresării dir. la Fz3 (EXEC) iar în cazul ad. indir. la fz.2 (INDIR.)
INDIRECT Fz.2	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i'_0 \rightarrow$ PcAdRI $I'_6 \rightarrow$ PmRA	$i_1 \rightarrow$ A(0)AdIdRI	—	$I_3 \rightarrow$ PmAdRI	—	—	—	idem
	- Se dă cda. CITEȘTE - se aduce la zero RA și RM - Se primește în RA adresa din partea de ad. a lui RI	- se aduce la zero partea de adresă și partea bitului pt. adresare indirectă a reg. RI	Nu se folosește	Se primește noua ad. din M în AdRI și bitul Id în Id RI	Nu se folosește	Nu se folosește	Nu se folosește	idem

INSTR. RM Nr 1 [ÎNCARCĂ B] [ÎNB] (ADUCERE, INDIR, EXECUȚIE)

CETA

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE Fz.1			La fel ca	la celelalte	instrucț. RM			
INDIRECT Fz.2			La fel ca	la celelalte	instrucții RM			
EXECUȚIE Fz.3 ÎNB	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i'_0 \rightarrow$ PdAdRI $i'_0 \rightarrow$ Pm RA	$i_1 \rightarrow$ A(0) B	—	$I_3 \rightarrow$ Pm B	—	—	Dacă FANG = 1 atunci $i_6 \rightarrow$ A(1)BFANG	Dacă BFANG·BITR=1 atunci $i_7 \rightarrow$ fz1
	-Se dă cda. CITEȘTE -Se aduce la zero RA și RM -Se primește în RA adresa din partea de adresă a lui RI	se aduce la zero B	Nu se folosește	Se primește cuvîntul CITIT din M în B	Nu se folosește	Nu se folosește	Dacă avem o cerere de întrer. atunci, în scopul realiz. unei sincronizări, se aduce bistabilul BFANG la unu. Vezi explicații în vol. 1, pag. 303	Dacă BFANG·BITR=1 atunci $i_7 \rightarrow$ fz6 $i_7 \rightarrow$ A(0)BITR $i_7 \rightarrow$ A(0)BFANG Dacă nu există cerere de într. sau într. nu este autoriz. se trece la Fz1, iar dacă avem cerere s. autoriz. se trece la Fz6, se dezautoriz. într. și se aduce la zero BFANG

INSTR. RM N=2 [INCARCĂ R] [INR] (ADUCERE, IND.R, EXECUȚIE)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE FZ.1			La fel ca	la celelalte	instrucțiiRM			
INDIRECT FZ.2			La fel ca	la celelalte	instrucții	RM		
EXECUȚIE FZ.3 INR	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i'_0 \rightarrow$ RADR $i'_0 \rightarrow$ RmRA -Se dă cda. CITEȘTE -Se aduce la zero RA și RM -Se primește în RA adresa din partea de adresă a lui RI	$i_1 \rightarrow$ A(0)SnR $i_1 \rightarrow$ A(0)MR Se aduce la zero R.	Nu se folosește	$I_3 \rightarrow$ RmSnR $I_3 \rightarrow$ RmMR Se primește în R cuvântul citit din M	—	—	Idem ca la IRM1, FZ3	Idem ca la IRM1, FZ3

CETA

INSTR. RN = 3 ÎNCARCĂ CUÎNMULTITORUL-Q [illegible] (ADUCERE, INDIR, EXECUȚIE)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE Fz.1			La fel ca	la celelalte	instrucții	RM		
INDIRECT Fz.2			la fel ca	la celelalte	instrucții	RM		
EXECUȚIE Fz.3 ÎNQ	i_0 → CITEȘTE i_0 → A(P)RA i_0 → A(C)RM i'_0 → RADRI i''_0 → PmRA	i_1 → A(C)Q i_1 → A(C)SnR	—	I_3 → PmQ I_3 → PmSnR	—	—	Idem ca la IRM1, Fz.3	Idem ca la IRM', Fz.3
	-Se dă cda CITEȘTE -Se aduce la zero RA și RM -Se primește în RA adresa din partea de adresă a lui RI	Se aduce la zero Q	Nu se folosește	Se primește în Q cuvîntul citit din M Simultan se primește semnul cuv și în SnR în vederea formării semnului produsului	Nu se folosește	Nu se folosește	Idem	Idem

II STR RM NL4 [INCARCA A] [INA] (ADUCERE, INDIR., EXECUTIE)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE Fz.1			La fel ca	la celelalte	instrucții	RM		
INDIRECT Fz.2			La fel ca	la celelalte	instrucții	RM		
EXECUȚIE Fz.3 INA	i_0 → CITEȘTE 0 → A(0)RA 0 → A(0)RM 0 → BAdRI 0 → PmRA	i_1 → A(0)SnR i_1 → A(0)MR i_1 → A(0)A	—	I_3 → BnSnR I_3 → PmMR	Dacă avem SnR = 1 atunci i_4 → +MR Dacă avem SnR = 1 atunci i_4 → -MR i_4 → 1SnA	—	Idem ca la IRM1, Fz.3	Idem ca la IRM1, Fz.3
	-Se dă cda. CITEȘTE -Se aduce la zero RA și RM. -Se primește adresa numărului din partea de adresă a lui R1 în RA	Se aduce la zero R și A.	Nu se folosește	Se primește numărul și semnul său din M în R	Se adună numărul aflat în R la zero aflat în A. Astfel dacă numărul e negativ apare sub formă de complement față de 2 în A.	Nu se folosește Obs. E necesar un tact liber pentru a asigura timpul de 2 μs. pentru adunarea lui 1 la formarea complementului.	Idem	Idem

CETA

INSTR. RM Nr 5. ÎNCARCĂ DEÎMPĂRȚITUL ÎN A [ ] (ADUCERE, INDIR, EXECUȚIE)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE Fz.1		La	fel ca la	celelalte	instrucții	RM		
INDIRECT Fz.2		La	fel ca la	celelalte	instrucții	RM		
EXECUȚIE Fz.3 IDA	i_0→citește i_0→A(0)RA i_0→A(0)RM i_0→RBADRI i_0→PmRA -Se dă cdă pentru citirea numărului. -Se aduce la zero RA și RM. -Se primește în RA adresa din partea de adresă a lui RI	i_1→A(0)SnR i_1→A(0)MR i_1→A(0)A Se aduce la zero R și A	— Nu se folosește	i_3→PmSnR i_3→PmMR Se primește deîmpărțitul și semnul său din MînR	i_4→MR Se trece mărimea deîmpărțitului (fără semn) din R în A	— Nu se folosește	Idem ca la IRM1, Fz.3 Idem	Idem ca la IRM1, fz.3 Idem

CETA

INSTR RM NR 6 MEMOREAZĂ Q (MEQ) (ADUCERE, INDIR, EXECUTĂ)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE Fz.1		La fel ca la		celelalte	instrucții	RM		
INDIRECT Fz. 2		La	fel ca la	celelalte	instrucții	RM		
EXECUȚIE Fz. 3 MEQ	$i_0 \rightarrow$ SCRIE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i_0 \rightarrow$ RADRI $I_0' \rightarrow$ PmRA	—	$I_2 \rightarrow$ PmRM $i_2 \rightarrow$ PdQ	—	—	—	Identica la IRM1, Fz.3	Identica la IRM1, Fz.3
	- Se dă cda. SCRIE - Se aduce la zero RA și RM - Se primește în RA adresa din partea de adresă a lui RI	Nu se folosește	Conținutul lui Q este primit în RM în vederea memorării	Nu se folosește	Nu se folosește	Nu se folosește	Idem	Idem

CETA

INSTR. RM Nr. 7. MEMOREAZĂ GR. DE CUV. 2 (MG2) (ADUCERE, INDIR, EXECUȚIE)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE Fz. 1		La	fel ca la	celelalte	instrucț	RM		
INDIRECT Fz 2		La	fel ca la	celelalte	instrucții	RM		
EXECUȚIE Fz. 3 MG 2	$i_0 \rightarrow$ SCRIE $i_0 \rightarrow$ A(0) RA $i_0 \rightarrow$ A(0) RM $i'_0 \rightarrow$ PdAdRI $I'_6 \rightarrow$ Pm RA	—	$I_2 \rightarrow$ Pm RM $i_2 \rightarrow$ PdG2	—	—	—	Idem ca la IRM1, Fz.3	Idem ca la IRM1, Fz.3
	- Se dă cda. SCRIE - Se aduce la zero RA și RM. - Se primește în RA adresa din partea de adresă a lui RI.	Nu se folosește	Continutul lui G2 este primit în RM în vederea memorării	Nu se folosește	Nu se folosește	Nu se folosește	Idem	Idem

CETA

INSTR. RM NR.8 MEMOREAZĂ A (REA) (ADUCERE, INDIRECT, EXECUȚIE)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE FZ.1		La	Ieșl ca la	celelalte	instrucții	RM		
INDIRECT FZ.2		La	Ieșl ca la	celelalte	instrucții	RM		
EXECUȚIE FZ.3 MEA	$i_0 \rightarrow$ SCRIE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i_0 \rightarrow$ PdADRI $i_0 \rightarrow$ PmRA -Se dă cda. SCRIE -Se aduce la zero RA și RM -Se primește în RA adresa din parte de ieșire a lui R...	— Nu se folosește	$i_2 \rightarrow$ PmRM $i_2 \rightarrow$ PdA Conținutul lui A este primit în RM în vederea memorării	— Nu se folosește	— Nu se folosește	— Nu se folosește	Idem ca la IRM1, FZ.3 Idem	Idem ca la IRM1, FZ.3 Idem

CETA

INSTR. RM Nr 9 MEMOREAZA PRODUSUL DIN Q MEQ (ADUCERE, INDIR. EXECUTIE)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE Fz.1		La	fel ca la	celelalte	instrucții	RM		
INDIRECT Fz.2.		La	fel ca la	celelalte	instrucții	RM		
EXECUȚIE Fz.3 MPQ	$i_0 \rightarrow$ SCRIE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i'_0 \rightarrow$ Pd AdRI $I'_0 \rightarrow$ Pm RA	$i_1 \rightarrow$ DrA, DrQ $I_1 \rightarrow$ LEG	$I_2 \rightarrow$ Pm RM $i_2 \rightarrow$ Pd Q	—	—	—	Idem ca și IRM1, Fz.3	Idem ca la IRM1, Fz.3
	- Se dă cda. SCRIE - Se aduce la zero RA și RM. - Se primește în RA adresa din partea de adresă a lui RI.	Q și A se deplasează cu un rang la dreapta pentru a avea partea m.p.s a produsului plasală corect în Q	Continutul lui Q este primit în RM în vederea memorării.	Nu se folosește	Nu se folosește	Nu se folosește	Idem	Idem

CETA

INSTR RM NR 10 MEMOREAZĂ CÎTUL DIN MQ (MCQ) (ADUCERE INDIR., EXECUȚIE)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE Fz. 1		La	fel ca la	celelalte	instrucții	RM		
INDIRECT Fz. 2		La	fel ca la	celelalte	instrucții	RM		
EXECUȚIE Fz. 3 MCQ	$i_0 \rightarrow$ SCRIE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i_0 \rightarrow$ Pd AdRI $I_0 \rightarrow$ Pm RA	—	$I_2 \rightarrow$ Pm RM $i_2 \rightarrow$ Pd MQ $i_2 \rightarrow$ Pd SnR	—	—	—	Idem ca la IRM1, Fz.3	Idem ca la IRM1, Fz.3
	- Se dă cda. SCRIE - Se aduce la zero RA și RM. Se primește în RA adresa din partea de acresă a lui RI.	Nu se folosește	Continutul părții de mărime a lui Q și continutul părții de semn a lui R sînt primite în RM în vederea memorării.	Nu se folosește	Nu se folosește	Nu se folosește	Idem	Idem

INSTR. GRUP. NR. 11 | MEMOREAZĂ REZULTATUL ADUNĂRII SAU SCĂDERII | MRA | CETA (ADUCERE, INDIR, EXECUȚIE)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE Fz.1		La	fel ca la	celelalte	instrucții	RM		
INDIRECT Fz.2		La	fel ca la	celelalte	instrucții	RM		
EXECUȚIE Fz.3 MRA	$i_0 \rightarrow$ SCRIE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i_0 \rightarrow$ A(0)SnR $i_0 \rightarrow$ A(0)MR $i'_0 \rightarrow$ PdAdRI $I6 \rightarrow$ PmRA	$I_1 \rightarrow$ PmSnR $I_1 \rightarrow$ PmMR $i_1 \rightarrow$ PdA $i'_1 \rightarrow$ A(0)A	-Dacă avem: SnR = 1 atunci $i_2 \rightarrow +$MR -Dacă avem: SnR = 1, atunci $i_2 \rightarrow -$MR	—	$I_4 \rightarrow$ PmRM $i_4 \rightarrow$ PdMA $i_4 \rightarrow$ PdSnR	$i_5 \rightarrow$ A(0)DCR	Idem ca la IRM1, Fz.3	-Dacă avem (01-10)SnA=1 atunci $i_7 \rightarrow$ A(1)DCR Idem ca la IRM1, Fz.3
	- Se dă cda. SCRIE - Se aduce la zero RA și RM - Se aduce la zero R - Se primește în RA adresa din partea de adresă a lui RI	-Conținutul lui A este primit în R -Cu ajutorul lui i'_1 întârziată de i_1, se aduce la zero A. Obs. Aducerea la zero cu i_2 e prea târzie.	- Conținutul lui R se trece în A necomplementat dacă e + sau complementat dacă e -. Se recomplementează numărul dacă e neg. pt. a fi dus în mărire reprezentarea semn-mărime.	Nu se folosește Obs. Trebuie lăsat liber pt. a obține timpul de 2 μs necesar adunării lui 1 la complementare.	Conținutul părții de mă-ime a lui A și al părții de semn a lui R se primește în RM pt. a memoral Obs. Pt. a fi memorată o informație trebuie să fie în RM cel târziu cu 4,5 μs după i_0	-Se aduce la zero bistabilul DCR	Idem	- Se aduce la 1 bistab. DCR dacă avem depășire. - Idem ca la IRM1, Fz 3

INSTR RM NR 12 ADUNĂ LA A [ADA] (ADUCERE, INDIR., EXECUȚIE) CETA

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE FZ.1		La	fel ca la	celelalte	instrucții	RM		
INDIRECT FZ.2		La	fel ca la	celelalte	instrucții	RM		
EXECUȚIE FZ.3' ADA	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow A(0)RA$ $i_0 \rightarrow A(0)RM$ $i'_0 \rightarrow PdAdRI$ $I'_0 \rightarrow Pm\,RA$	$i_1 \rightarrow A(0)SnR$ $i_1 \rightarrow A(0)MR$	—	$I_3 \rightarrow Pm\,SnR$ $I_3 \rightarrow Pm\,MR$	-Dacă avem $\overline{SnR}=1$ atunci $i_4 \rightarrow +MR$ Dacă avem $SnR=1$ atunci $i_4 \Rightarrow :MR_{SnA}$	$i_5 \rightarrow A(0)DCR$	Idem ca la IRM1, Fz 3	-Dacă avem (01-10) $Sn\,A=1$ atunci $i_7 \rightarrow A(1)DCR$ Idem ca la IRM1, Fz.3
	-Se dă cda. pentru citirea operand. -Se aduce la zero RA și RM -Se primește în RA adresa operandului din partea de adresă a lui RI.	-Se aduce la zero R	Nu se folosește	-Se primește operandul și semnul lui din M în R.	-Se efectuează adunarea numărului aflat în R cu cel aflat în A	-Se aduce la zero bistabilul DCR	Idem	-Se aduce la 1 bistab. DCR dacă avem depășire -Idem ca la IRM1, Fz.3

CETA

INSTR RM Nr 13 SCADE DIN A [SCA] (ADUCERE, INDIR, EXECUTIE)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE FZ.1		La	fel ca la	celelalte	instructii	RM		
INDIRECT FZ.2		La	fel ca la	celelalte	instructii	RM		
EXECUTIE FZ.3 SCA	$i_0 \rightarrow$ CITESTE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i_0' \rightarrow$ PE AdR $I_0 \rightarrow$ Pm RA	$i_1 \rightarrow$ A(1)SnR $i_1 \rightarrow$ A(0)MR	—	$I_3 \rightarrow$ Pm SnR $I_3 \rightarrow$ Pm MR	- Dacă avem SnR = 1 atunci $i_4 \rightarrow$ + MR Dacă avem SnR = 1 atunci $i_4 \rightarrow$ − MR, $i_4 \rightarrow$ + 1 SnA	$i_5 \rightarrow$ A(0) DCR	Idem ca la IRM1, FZ.3	Dacă avem (01-10) SnR = 1 atunci $i_7 \rightarrow$ A(1) DCR Idem ca la IRM1, FZ.3
	- Se ca coa. pentru citirea operandului. - Se aduce la zero RA și RM - Se primește în RA adresa operandului din partea de adr. a lui R:	- Se aduce la 1 partea de semn a lui R, ceace va duce la schimb. semnului nr corespunz. scăderii - Se aduce part. de mărime a lui R la zero	Nu se folosește	Se primește operandul și semnul lui din M în R	Se efectuează adunarea num. cu semn schimbat aflat acum în R cu cel aflat în A.	- Se aduce la zero bistabilul DCR	Idem	Se aduce la 1 bistab. DCR dacă avem depășire. Idem ca la IRM1, FZ 3

INSTR. RM Nr. 14 INMULȚEȘTE INM (ADUCERE, INDIR, EXECUȚIE) CETA 20

FAZA	i_0	i_1	i_2	i_3	i_4	i_5 —	i_6	i_7
ADUCERE Fz 1		La	fel ca la	celelalte	instrucții	RM		
INDIRECT Fz. 2		La	fel ca la	celelalte	instrucții	RM		
EXECUȚIE Fz. 3 INM	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i'_0 \rightarrow$ PdAdRI $I'_0 \rightarrow$ PmRA	$i_1 \rightarrow$ A(0)MR $i_1 \rightarrow$ A(0)A	$i_2 \rightarrow$ A(0)N24	$I_3 \rightarrow$ PmSnR $I_3 \rightarrow$ PmMR	$i_4 \rightarrow$ DrA, DrQ $i_4 \rightarrow$ +1N24 $I_4 \rightarrow$ LEG	—	- Dacă avem $\overline{N24}$ = - atunci $i_6 \rightarrow$ A(1) BII Idem ca la IRM 1, Fz. 3	$i_7 \rightarrow$ SnR Idem ca la IRM 1, Fz. 3
	- Se dă cdă pt. citirea deînmultitului. - Se aduce la zero RA și RM - Se primește în RA adresa deînmultitului din partea de adresă a lui RI.	- Se aduce la zero A și partea de mâine a lui R. Obs. În SnR se găsește adus prin instr. INQ semnul înmulțitorului.	- Se aduce la zero numărătorul N 24	- Se primește în R deînmultitul citit din M. Cu această ocazie se formează semnul produsului în SnR căci aici se află, adus înainte, semnul înmulțitorului prin INQ.	- Se dă cda de depl. și adunare a prod. partial. - Se introd. 1 în N24 - Se face leg. între reg. A și Q Cele de mai sus se repetă de 24 ori prin revenirea de 23 ori de la i_6 la i_4.	Nu se folosește Obs. Tactul liber asigură cele 2,5 μs necesare la efectuarea depl. la Dr. și a adunării ce intervine la INM.	- Se trece de la i_6 înapoi la i_4 în funcție de starea lui N24. Aceleași microoper. care aparțin imp. i_4 - i_6 se efectuează de 24 ori. - Idem ca la IRM 1, Fz. 3 *).	- Semnul produsului se duce din SnR în SnA. În A avem partea c m.s. a produsului. Idem ca la IRM 1, Fz. 3.

*) Se va urmări descrierea repetării impulsurilor.

CETA

INSTR. RM Nr. 15 IMPARTE IMP (ADUCERE, INDIR, EXECUTIE SUPL)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE Fz.1		La fel ca la		celelalte	instrucț.	RM		
INDIRECT Fz. 2		La fel ca la		celelalte	instrucții	RM		
EXECUȚIE Fz.3 IMP	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i'_0 \rightarrow$ PdAdRI $I'_0 \rightarrow$ PmRA – Se dă cda. pt. citirea împărțitorului. – Se aduce la zero RA și RM. – Se primește în RA adresa împărțitorului din partea de adresă a lui RI.	$i_1 \rightarrow$ A(0)MR – Se aduce la zero partea de mărime a lui R. Obs. In SnR se găsește, adus prin instr. IDA, semnul deîmpărțitului.	— Nu se folosește	$I_3 \rightarrow$ PmSnR $I_3 \rightarrow$ PmMR – Se primește în R împărțitorul citit din M. Cu această ocazie se formează semnul câtului în SnR, căci, aici se află, adus dinainte cu IDA semnul deîmpărțitului.	$i_4 \rightarrow -$MR $i_4 \rightarrow +1$SnA – Se scade împărțitorul din deîmpărțit pt. a verifica existența condiției de efectuare a împărțirii, adică deîmpărțitul să fie mai mic ca împărțitorul.	$i_5 \rightarrow$ A(0)N24 $i_5 \rightarrow$ A(0)DCR $i_5 \rightarrow$ A(0) Q – Se aduce la zero N24 – Se aduce la zero DCR – Se aduce la zero reg. Q Tactul e necesar și ptr. asigurarea timpului de 2μs ptr. efectuarea scăderii	– Dacă avem $\overline{(01-10)SnA}=1$ și $\overline{Sn2A}=1$ atunci $i_6 \rightarrow$ A(1)DCR – Dacă deîmpărțitul este mai mare sau egal cu împărțitorul împărțirea nu se poate efectua și se aduce DCR în starea 1. *)	– Daca avem DCR = 1 atunci $i_7 \rightarrow$ fz.1 – Dacă avem $\overline{DCR}=1$ atunci $i_7 \rightarrow$ fz.4 – Daca avem 1 în DCR se întrerupe instr. IMP și se trece la instrucția următoare, care poate fi omisă la DCR urmată de SSP – Dacă nu avem depășire se trece la Fz.4 și se continuă IMP

*). Deîmpărțitul este mai mare sau egal cu împărțitorul când în Sn1A și Sn2A avem aceeași cifră adică, $\overline{(01-10)SnA}=1$ și când această cifră este 0, adică $\overline{Sn2A}=1$. De fapt prima condiție există totdeauna.

CETA

INSTR. RM Nr 15 (continuare) IMPARTE IMP (ADUCERE, INDR, EXECUTIE, EXECUTIE SUP.)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
EXECUTIE SUPLIMENTARĂ FZ 4 IMP	$i_0 \rightarrow$ SfA $i_0 \rightarrow$ SfQ $i_0 \rightarrow +1$ N24	-Dacă avem Sn 2A = 1 atunci $i_1 \rightarrow +$MR -Dacă avem Sn 2A = 1 atunci $i_1 \rightarrow -$MR $i_1 \rightarrow +$1SnA	—	-Dacă avem $\overline{Sn\,2A} = 1$ atunci $i_3 \rightarrow$ A(0)RgQ -Dacă avem N24 = 1 atunci $i_3 \rightarrow$ A(0)3II	—	—	Idem ca la IRM1, Fz.3	Idem ca la IRM1, Fz.3
	-Se deplasează la stînga cu un rang registrele A și Q. -Se adaugă 1 în NUM. N24	-Se face, conform cu algoritmul de împărțire, adunarea sau scăderea împărțitului la deîmpărțitul deplasat cu un rang la stînga	Nu se folosește -Timpul e necesar pt. a asigura intervalul de 2 μs pentru efectuarea adunării sau scăderii	-Se formează în conformitate în algoritmul, cifra cîtului și se introduce în RgCQ -Se aduce la 1 biștabilul înm-împ. prin care împ. $i_0 \div i_3$ vor fi parcurse de 24 ori *)	Nu se folosește	Nu se folosește	Idem ca la IRM1, Fz.3	Idem ca la IRM1, Fz.3

*) Se va urmări descrierea repetării impulsurilor.

CETA

INSTR RM NR 16 SAU INCLUSIV CU A [SAI] (ADUCERE, IND R, EXECUTIE)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE Fz.1		La	fel ca la	celelalte	instrucții	RM		
INDIRECT Fz.2		La	fel ca la	celelalte	instrucții	RM		
EXECUȚIE Fz.3 SAI	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i'_0 \rightarrow$ Pd AdRI $I'_0 \rightarrow$ Pm RA	$i_1 \rightarrow$ A(0)SnR $i_1 \rightarrow$ A(0)MR	$I_2 \rightarrow$ Pm SnR $I_2 \rightarrow$ Pm MR $i_2 \rightarrow$ Pd A	$I_3 \rightarrow$ Pm SnR $I_3 \rightarrow$ Pm MR $i_3 \rightarrow$ A(0) A	$i_4 \rightarrow +$ SnR $i_4 \rightarrow +$ MR	—	Idem ca la IRM1, Fz 3	Idem ca la IRM1, Fz. 3
	- Se dă cda CITEȘTE - se aduce la zero RA și RM. - se primește în RA adresa din partea de adresă a lui RI.	- Se aduce la zero reg. R	- Se duce în reg. R conținutul registrului A.	- Se primește în R conținutul locului adresat și se formează și funcția SAU INCL - Se aduce la zero A	- Rezultatul obținut în urma formării funcției SAU INCL - se trece în A	Nu se folosește	Idem ca la IRM1, Fz.3	Idem ca la IRM1 Fz.3

CETA

INSTR RM Nr 17 SAU EXCLUSIV CU A [SAE] (ADUCERE, INDIR, EXECUTE)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE Fz. 1		La	fel ca la	celelalte	instructii	RM		
INDIRECT Fz. 2		La	fel ca la	celelalte	instructii	RM		
EXECUTIE Fz. 3 SAE	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i_0 \rightarrow$ PdAdRI $i_0 \rightarrow$ PmRA	$i_1 \rightarrow$ A(0)SnR $i_1 \rightarrow$ A(0)MR	—	$I_3 \rightarrow$ PmSnR $I_3 \rightarrow$ PmMR	$I_4 \rightarrow$ Tr. Bl $I_4 \rightarrow +$ SnR $i_4 \rightarrow +$ MR	—	Idem ca la IRM1, FZ3	Idem ca la IRM1, FZ.3
	- Se da cda. CITEȘTE - Se aduce la zero RA și RM - Se primește în RA adresa din partea de adresă a lui RI	- Se acuce la zero registrul R	Nu se folosește	- Se primește în R conținutul locului adresat.	- Se blochează transferul. - se trimite în A numărul aflat în R, în condițiile transf. blocat, cînd în A se formează SAU EXCL.	Nu se folosește	Idem ca la IRM1, Fz 3	Idem ca la IRM1, Fz. 3

INSTR. RM Nr. 18 ȘI CU A ȘI A (ADUCERE INDIR, EXECUȚIE)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE FZ 1		La	fel ca la	celelalte	instrucții	RM		
INDIRECT FZ 2		La	fel ca la	celelalte	instrucții	RM		
EXECUȚIE FZ 3 ȘI A	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i'_0 \rightarrow$ PdAdRI $I6 \rightarrow$ Pm RA	$i_1 \rightarrow$ A(0)SnR $i_1 \rightarrow$ A(0)MR $i_1 \rightarrow$ A(0)Q	$I_2 \rightarrow$ PmQ $i_2 \rightarrow$ PdA	$I_3 \rightarrow$ PmSnR $I_3 \rightarrow$ PmMR $I_3 \rightarrow$ PmQ	$I_4 \rightarrow$ TrBl $i_4 \rightarrow +$ SnR $i_4 \rightarrow +$ MR	$i_5 \rightarrow$ A(0)SnR $i_5 \rightarrow$ A(0)MR	$I_6 \rightarrow$ PmSnR $I_6 \rightarrow$ PmMR $i_6 \rightarrow$ PdQ Idem ca la IRM1, Fz 3	$I_7 \rightarrow$ TrBl $i_7 \rightarrow +$ SnR $i_7 \rightarrow +$ MR Idem ca la IRM1, Fz 3
	- Se dă cda. CITEȘTE - Se aduce la zero RA și RM. - Se primește în RA adresa din partea de adresă a lui RI.	Se aduce la zero reg R și reg Q	Conținutul reg.A se trimite în reg.Q. Deci X se află în Q și în A.	Conținutul locului adresat Y se aduce simultan în reg.R și Q. Astfel în reg. Q se formează funcția SAU INC. între X și Y, iar în R se află Y.	-Se blochează transferul în reg. A și se trimite aici, nemodificat, conținutul reg. R. Astfel se formează în A funcția SAU EXCL între X și Y	-Se aduce la zero reg. R.	- Conținutul reg.G se trimite în R. Astfel avem în A $X\bar{\vee}Y$, iar în R $X \vee Y$ *) - Idem ca la IRM1, Fz.3	-Se blochează trans. în reg A și se trimite aici, nemodificat, conținutul reg. R. Astfel se formează în A funcția X Y vezi descr. instr. -Idem ca la IRM1, Fz 3

*) Prin simbolul $\bar{\vee}$ am notat funcția SAU EXCL.

INSTR RM Nr 19 [SALT] [SLT] (ADUCERE, INDIRECT) CETA .

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE FZ 1	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i_6 \rightarrow$ PdNA $I'_0 \rightarrow$ PmRA	$i_1 \rightarrow$ A(0)RI	—	$I_3 \rightarrow$ PmRI	—	$i_5 \rightarrow$ A(0)NA	$I_6 \rightarrow$ PmNA $i_6 \rightarrow$ PdAdRI Idem ca la IRM1. fz. 3	Dacă $\overline{Id} = 1$ și dacă $\overline{BFANG} \cdot BITR = 1$ atunci $i_7 \rightarrow fz1$
	– Se dă cda CITEȘTE – Se aduce la zero RA și RM – Se primește în RA adr. din NA	– Se aduce la zero RI pt. a putea primi instr. citită din M.	Nu se folosește	– Se primește în RI instrucția citită, după trecerea timpului de acces de 3 μs.	Nu se folosește	– Se aduce la zero NA	– Partea de adresă a instrucției din RI se trece în NA. – Idem ca la IRM1, FZ3	Dacă $\overline{Id} = 1$ și dacă BFANG · BITR = 1 atunci $i_7 \rightarrow$;26 $i_7 \rightarrow$ A(0)BITR $i_7 \rightarrow$ A(0)BFANG
INDIRECT Fz. 2 SLT	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i_6 \rightarrow$ PdNA $I_6 \rightarrow$ PmRA	$i_1 \rightarrow$ A(0)AdIdRI	—	$I_3 \rightarrow$ PmAdIdRI	—	Idem	Idem	Dacă Id = 1 atunci $i_7 \rightarrow fz2$
	– Se dă cda CITEȘTE – Se aduce la zero RA și RM – Se primește în RA adr. din NA	– Se aduce la zero partea de adresă și Id din RI	Nu se folosește	– Se primește în partea de adresă și în partea lo.	Nu se folosește	Idem	Idem	Idem ca la IRM1, FZ3 la care se adaugă *)

Obs: în faza de ADUCERE primele 4 impulsuri $i_0 \div i_3$ execută la fiecare instrucție aceleași microoperații, care corespund aducerii instrucției din M în RI.

*) Microoperațiile coresp. lui i_7 în FZ2 sînt aceleași cași în FZ1. Aceasta face ca funcționarea să fie următoarea: Dacă în timpul lui FZ1 avem cerere de întrerupere dar avem și adresare indirectă (Id=1), nu se ia în considerare cererea de întrerupere ci se trece la FZ2 Abia la sfîrșitul adresărilor indirecte (Id=0) se ia în considerare, în timpul lui FZ2, cererea de întrerupere. Faptul că la i_6 se execută atît la FZ1 cît și la FZ2, microoperația de la IRM1, Fz 3 nu jenează.

CETA

INSTR. RM. Nr. 20 **SALT LA SUBPROGRAM [SSP]** (ADUCERE, INDIRECT, EXECUTIE)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE FZ.1	...	La fel ca la celelalte instructii RM						
INDIRECT FZ. 2	La fel ca la celelalte instructii RM							
EXECUTIE FZ.3 SSP	$i_0 \rightarrow$ SCRIE $i_0 \rightarrow$ A(0) RA $i_0 \rightarrow$ A(0) RM $i'_0 \rightarrow$ PdAdRI $I'_0 \rightarrow$ Pm RA	$I_1 \rightarrow$ Pm RM $i_1 \rightarrow$ Pd NA	$I_2 \rightarrow$ A(0) NA	$I_3 \rightarrow$ Pm NA $i_3 \rightarrow$ PdAdRI	—	—	$i_6 \rightarrow$ NA Idem ca la IRM1, FZ3	Idem ca la IRM1, FZ3
	- Se dă com. SCRIE - Se aduce la zero RA si RM - Se primeste în RA adr Y din partea de adresă a lui RI	- Continutul lui NA, care este adresa de întoarcere P+1, se trimite în RM pentru a fi înscrisă în M la adr. Y	- Se aduce la zero NA	- Se primeste în NA adresa Y din partea de adresă a lui RI	Nu se foloseste	Nu se foloseste	- Se formează Y+1 la care începe efectiv subprogramul.	Idem ca la IRM1, FZ3
	Obs In M la adresa Y am introdus adresa din programul principal la care trebuie să revenim pentru a relua programul din locul întrerupt. La sfîrsitul subprogramului avem instr. SALT la Y cu adresare indirectă. Y este adresa specificată în SSP si este adresa primei instrucții a subprogramului. La această adresă se înmag. adresa P+1 din prog. principal, la care trebuie să revenim.							

CETA

INSTR. RM Nr. 21 MAREȘTE ȘI OMITE DACĂ E ZERO [MOZ] (ADUCERE, INDIR, EXECUȚIE, EX. SUPL)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE FZ 1	La fel ca la celelalte instrucții RM							
INDIRECT FZ 2	La fel ca la celelalte instrucții RM							
EXECUȚIE FZ 3 MOZ	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow A(0)RA$ $i_0 \rightarrow A(0)RM$ $i_0 \rightarrow A(0)B$ $I'_0 \rightarrow$ Rd AdRI $I'_0 \rightarrow PmRA$	$I_1 \rightarrow PmB$ $i_1 \rightarrow PdA$	$i_2 \rightarrow A(0)SnR$ $i_2 \rightarrow A(0)MR$ $i_2 \rightarrow A(0)A$	$I_3 \rightarrow PmSnR$ $I_3 \rightarrow PmMR$	Dacă avem $\overline{SnR} = 1$ atunci $i_4 \rightarrow +MR$ Dacă avem $SnR = 1$ atunci $i_4 \rightarrow -MR$ $i_4 \rightarrow +1SnA$	$i'_5 \rightarrow +1A$	—	$i_7 \rightarrow A(0)SnR$ $i_7 \rightarrow A(0)MR$ $i_7 \rightarrow fz4$
	— Se dă cda CITEȘTE — Se aduce la zero RA și RM — Se aduce la zero reg. B — Se primește în RA adresa din partea de adresă a lui RI	— Se primește în reg. B conținutul reg. A în scopul de a nu se pierde acest conținut cînd A este utilizat pt. efectuarea de operații	— Se aduce la zero reg. R și A în vederea introducerii în A a conținutului locului adresat	— Se primește în reg. R conținutul locului adresat	— Se adună numărul aflat în R la zero aflat în A. Astfel dacă numărul este negativ apare sub formă de complement de 2 în A.	— Se adună unu de rangul c.m.p.s. la conținutul lui A.	— Interval necesar propagării transferului.	— Se aduce la zero reg. R — Se trece la FZ 4

INSTR RM Nr 21 (continuare) MĂREȘTE ȘI OMITE DACĂ E ZERO (ADUCERE, IND R EXEC EXEC SUPL)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
EXECUȚIE SUPLIMENTARĂ FZ.4 MOZ	$I_0 \rightarrow Pm\,SnR$ $I_0 \rightarrow Pm\,MR$ $i_0 \rightarrow PdA$ $i_0 \rightarrow SCRIE$ $i_0 \rightarrow A(0)RM$ Dacă avem ZA = 1 atunci $i_0 \rightarrow +1\,NA$	$i_1 \rightarrow A(0)A$	Dacă avem: $\overline{SnR} = 1$ atunci $i_2 \rightarrow +MR$ Dacă avem: SnR = 1 atunci $i_2 \rightarrow -MR$ $i_2 \rightarrow +1\,SnA$	—	$I_4 \rightarrow Pm\,RM$ $i_4 \rightarrow PdA$ $i_4 \rightarrow A(0)\,SnR$ $i_4 \rightarrow A(0)\,MR$	$i_5 \rightarrow A(0)A$ $I_5 \rightarrow Pm\,SnR$ $I_5 \rightarrow Pm\,MR$ $i_5 \rightarrow PdB$	$i_5 \rightarrow +SnR$ $i_5 \rightarrow +MR$ Idem ca la IRM1, FZ3	Idem ca la IRM1, FZ3
	— Numărul din A, la care s-a adăugat unu, este dus în R în vederea recomplementării spre a fi exprimat în reprez. semn-mărime. — Se dă comanda scrie în M la adresa de la care s-a scris și care se află în RA*	— Se aduce la zero A.	— Dacă numărul este pozitiv se introduce în A nemodificat — Dacă este negativ se complementează.	— Asigură timpul de 2 μs necesar la adunarea care intervine la complementare.	— Numărul din A, acum reprezentat în semn-mărime, este înscris în RM pentru a fi memorat. — Se aduce la zero R. Obs. Acum numărul din A este cel adus din M, mărit cu o unitate.	— Se aduce la zero A — Se primește din B în R numărul care a fost inițial în A	— Numărul care a fost inițial în A se trimite din nou în A astfel conținutul lui A nu se pierde. — Idem ca la IRM1, FZ3	— Idem ca la RM1, FZ3

*) Pentru a putea fi înscris corect în M un cuvînt aflat în RM, el trebuie să fie adus în RM cel mai tîrziu la i_4 (cda. SCRIE se dă la i_0).

— Dacă după adunarea unității la conținutul lui A, cu i_5 din Fz3, acesta devine zero, se adaugă 1 în NA și deci următoarea instrucție se omite

CETA

INSTR. RM Nr 22 COMPARĂ CU B SI OPRTE LA INEGALITATE CBO (ADUCERE, INDIRECT, EXECUȚIE, EXECUȚIE)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ADUCERE FZ.1	La fel ca la celelalte instrucții RM							
INDIRECT FZ.2	La fel ca la celelalte instrucții RM							
EXECUȚIE FZ. 3 CBO	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow$ A(0) RA $i_0 \rightarrow$ A(0) RM $i'_0 \rightarrow$ RAdR1 $I'_0 \rightarrow$ Rm RA – Se dă cda CITEȘTE – Se aduce la zero RA ș RM – Se primește în RA adresa din partea de acresă a lui RI.	$i_1 \rightarrow$ A(0) SnR $i_1 \rightarrow$ A(0) MR $i_1 \rightarrow$ A(0) Q – Se aduc la zero R ș Q	$i_2 \rightarrow$ PmQ $i_2 \rightarrow$ PdA – Conținutul reg A se trece în Q în scopul păstrării numărului ellat inițial în A	$I_3 \rightarrow$ Rm SnR $I_3 \rightarrow$ Rm MR $i_3 \rightarrow$ A(0) A – Conținutul locului adresat se primește în R. El reprezintă unul din numerele ce se compară. – Se aduce la zero A.	$i_4 \rightarrow$ +SnR $i_4 \rightarrow$ +MR – Se trece nemodificat în A numărul din R.	$i_5 \rightarrow$ A(0) SnR $i_5 \rightarrow$ A(0) MR – Se aduce la zero R.	$I_6 \rightarrow$ Rm SnR $I_6 \rightarrow$ Rm MR $i_6 \rightarrow$ PdB – Se predă numărul din B în R. Acesta este celălalt număr care se compară.	$i_7 \rightarrow$ fz 4 – Se trece la FZ 4

CETA

INSTRUCȚIUNEA Nr 22 (continuare) COMPARĂ CU SUBSTRACȚIE LA INEGALITATE (CEEJ) (ADUCERE, INDICATOR EXEC, EXEC SUPL.)

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
EXECUȚIE SUPLIMENTARĂ FZ.4 CBO	$I_0 \rightarrow$ Tr B $i_0 \rightarrow +$SnR $i_0 \rightarrow +$MR		$i_2 \rightarrow$ A(0) SnR $i_2 \rightarrow$ A(0) MR Dacă în A nu avem zero, adică $\overline{ZA}=1$ atunci $i_2 \rightarrow +1$ NA	$I_3 \rightarrow$ Pm SnR $I_3 \rightarrow$ Pm MR $i_3 \rightarrow$ Pd Q	$i_4 \rightarrow$ A(0) A	$i_5 \rightarrow +$SnR $i_5 \rightarrow +$MR	Idem ca la IRM1, FZ3	Idem ca la IRM1, FZ3
	- Se blochează transferul - Se trimite în A numărul aflat în R, în concluzia transferului blocat, cînd în A se formează funcția sau EXCL	Nu se folosește	- Se aduce la zero R. - Dacă în A nu avem zero, ceace înseamnă că numerele nu sînt egale, se mărește cu unu conținutul lui NA, ceace provoacă omiterea	- Numărul din Q, care era inițial în A se trece în R	- Se aduce la zero A.	- Numărul din R se trece nemodificat în A. Astfel în A ajunge numărul care a fost inițial în A.	Idem ca la IRM1, FZ3	Idem ca la IRM1, FZ3

INSTR. RR | GRUP DEPLASĂRI ROTIRI | GDR | NOP, DSA, DSQ, DDA, DDQ, RSQ, RDQ, RSQ2, AZL, OQP

(ADUCERE) RSA, RDA, RSA2, OAP

FAZĂ	i_0	i_1	i_2	i_3	i_4		i_5	i_6		i_7
ADUCERE Fz1 GDR	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i'_0 \rightarrow$ PUNA $i'_0 \rightarrow$ PmRA	$i_1 \rightarrow$ A(0)RI	–	$I_3 \rightarrow$ PmRI	5. DSA Dacă B17=1 și B18=1 atunci $i_4 \rightarrow$ St A 6. DSQ Dacă B17=1 și $\overline{B18}$=1 atunci $i_4 \rightarrow$ St Q 7. DDA Dacă B16=1 și B18=1 atunci $i_4 \rightarrow$ DrA 8. DDQ Dacă B16=1 și $\overline{B18}$=1 atunci $I_4 \rightarrow$ DrTQ, $i_4 \rightarrow$ DrQ 9. RSA Dacă B15=1 și B18=1 atunci $I_4 \rightarrow$ Rot St A, $i_4 \rightarrow$ St A 10. RSQ Dacă B15=1 și B18=1 atunci $I_4 \rightarrow$ Rot StQ, $i_4 \rightarrow$ St.Q	11. RDA Dacă B14=1 și B18=1 atunci $I_4 \rightarrow$ Rot DrA, $i_4 \rightarrow$ DrA 12. RDQ Dacă B14=1 și $\overline{B18}$=1 atunci $I_4 \rightarrow$ Rot DrQ, $I_4 \rightarrow$ DrTQ, $i_4 \rightarrow$ DrQ 13. RSA2 Dacă B13=1 și B18=1 atunci $I_4 \rightarrow$ Rot St A | $I_5 \rightarrow$ Rot St A $i_4 \rightarrow$ St A | $i_5 \rightarrow$ St A 14. RSQ2 Dacă B13=1 și $\overline{B18}$=1 atunci $I_4 \rightarrow$ Rot StQ | $I_5 \rightarrow$ Rot StQ $i_4 \rightarrow$ StQ | $i_5 \rightarrow$ StQ	2. AZL Dacă B11=1 atunci $i_5 \rightarrow$ A(0)L 3. OAP Dacă B9=1 și B18=1 și OA=0 atunci $i_5 \rightarrow$ +1NA 4. OQP Dacă B9=1 și B18=1 și OQ=0 atunci $i_5 \rightarrow$ +1NA	5. DSA Dacă B7=1 și B18=1 atunci $i_6 \rightarrow$ St A 6. DSQ Dacă B7=1 și $\overline{B18}$=1 atunci $i_6 \rightarrow$ St Q 7. DDA Dacă B6=1 și B18=1 atunci $i_6 \rightarrow$ DrA 8. DDQ Dacă B6=1 și $\overline{B18}$=1 atunci $I_6 \rightarrow$ DrTQ, $i_6 \rightarrow$ DrQ 9. RSA Dacă B5=1 și B18=1 atunci $I_6 \rightarrow$ Rot St A, $i_6 \rightarrow$ St A $i_6 \rightarrow$ +1NA Idem ca la IRM1, Fz3	10 RSQ Dacă B5=1 și B18=1 atunci $I_6 \rightarrow$ Rot StQ, $i_6 \rightarrow$ St Q 11. RDA Dacă B4=1 și B18=1 atunci $I_6 \rightarrow$ Rot DrA, $i_6 \rightarrow$ DrA 12. RDQ Dacă B4=1 și $\overline{B18}$=1 atunci $I_6 \rightarrow$ Rot DrQ, $I_6 \rightarrow$ DrTQ, $I_6 \rightarrow$ DrQ 13. RSA2 Dacă B3=1 și B18=1 atunci $I_6 \rightarrow$ Rot St A | $I_7 \rightarrow$ Rot St A $i_6 \rightarrow$ St A | $i_7 \rightarrow$ St A 14. RSQ2 Dacă B3=1 și $\overline{B18}$=1 atunci $I_6 \rightarrow$ Rot StQ | $I_7 \rightarrow$ Rot StQ $i_6 \rightarrow$ StQ | $i_7 \rightarrow$ StQ	Idem ca la IRM1, Fz3

Obs. Instr. Nr.1 NOP se execută cînd toți biții B0-B18 sînt zero, adică nu este microprogr. nici o altă instrucție

INSTR. R.R | GRUPA MODIFICĂRI OMITERI | GMO | (ADUCERE) | CETA

FAZA	i_0	i_1	i_2	i_3	i_4		i_5		i_6	i_7
ADUCERE FZ 1 GMO	i_0 → CITEȘTE i_0 → A(0)RA i_0 → A(0)RM i_6 → PdNA i_6 → PmRA i_0 → A(0)BOM	i_1 → A(0)RI	—	I_3 → PmRI	1AZA Dacă $\overline{B14}$ = 1 și B13 = 1 atunci i_4 → A(0)A 2. AZB Dacă $\overline{B15}$ = 1 și B15 = 1 atunci i_4 → A(0)B 3. AZQ Dacă $\overline{B16}$ = 1 și B17 = 1 atunci i_4 → A(0)Q 4. AZL Dacă $\overline{B8}$ = 1 și B7 = 1 atunci i_4 → A(0)L 5. G1 IN Q Dacă B18 = 1 și $\overline{B17}$ = 1 atunci I_4 → PmQ i_4 → PdG1 6. AZMR Dacă $\overline{B11}$ = 1 și B10 = 1 atunci i_4 → A(0)MR	7 AZSR Dacă B12 = 1 atunci i_4 → A(0)SnR 8. AUB Dacă B16 = 1 și B15 = 1 atunci i_4 → A(1)B 9. AUL Dacă B8 = 1 și $\overline{B7}$ = 1 atunci i_4 → A(1)Sn2A 10. CML Dacă B8 = 1 și B7 = 1 atunci i_4 → CMSn2A 11. ADMR Dacă B11 = 1 și B10 = 1 atunci i_4 → +MR 12. CMR Dacă B11 = 1 și B10 = 1 atunci i_4 → −MR 13. AD1RqCA RJP Dacă B14 = 1 și $\overline{B13}$ = 1 atunci i_4 → +1RqCA 14. AD1SnA Dacă B14 = 1 și B13 = 1 atunci i_4 → +1SnA 22. OLZ Vezi ec(2) de descr GMO 28. AD1SnR Dacă B18 = 1 și B17 = 1 atunci i_4 → +1SnR	15. A IN B Dacă col 8 este 001 atunci I_5 → PmB i_6 → PdA 16. NA IN Q Dacă col 8 este 010 atunci I_5 → PmQ i_5 → PdNA 17. B IN R Dacă col 8 este 011 atunci I_5 → PmSnR I_5 → Pm MR i_5 → Pd B 18. Q IN R Dacă col 8 este 100 atunci I_5 → PmSnR I_5 → PmMR i_5 → PdQ 19. Q IN B Dacă col.8 este 101 atunci I_5 → PmB i_5 → PdQ	20. A IN Q Dacă col 8 este 110 atunci I_5 → PmQ i_5 → PdA 21 MUA Dacă B2 = 1 atunci i_5 → +1A 23. OA − Vezi ec. (3) de descr GMO 24. OA P vezi ec. (3) de descr GMO	25. OAZ Vez ec. de descr GMO 26. OS! Vezi ec(4) de descr a GMO i_6 → +1NA 27. OND Vezi ec. (5) de descr GMO Idem ca la IRM1, FZ3	Dacă avem BOM = 1 atunci i_7 → +1NA Idem ce la IRM1, FZ3

Obs. Instrucțiile de omitere se îndeplinesc în funcție de condițiile din ec GMO(2), (3) și (4). În cazul îndeplinirii condițiilor impulsului corespunzător i se duce la A(1) BOM

INSTR. IE | INTRODUCERE | EXTRAGERE | IIE | (ADUCERE) | a0 | CETA

FAZA	i_0	i_1	i_2	i_3	Dacă se începe cu diferențe mai jos se execută microoper. de la imp. i_4, i_5, i_6, i_7	i_4	i_5	i_6	i_7
ADUCERE Fz1 IIE	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i'_0 \rightarrow$ PdNA $I'_0 \rightarrow$ PmRA	$i_1 \rightarrow$ A(0)RI	—	$I_3 \rightarrow$ PmRI	2. OPR IIE = 1 și B18=1 și B17 = 1 și biții de la B16 – B6=0	—	—	$i_5 \rightarrow$ +1NA	Idem ca la IRM1, Fz 3 $i_7 \rightarrow$ Fz1 $i_7 \rightarrow$ A(0)BPC
					3. AZDC IIE=1 și B18=1 și B16=1 și restul biților de la B17 – B6 = 0	$i_4 \rightarrow$ A(0) DCR	—	Idem ca la IRM1, Fz 3	
					4. AUDC IIE=1 și B18=1 și B15=1 și restul biților de la B17 – B6=0	$i_4 \rightarrow$ A(1) DCR	—		Idem ca la IRM1, Fz 3
					5. ODCZ IIE=1 și B18=1 și B14=1 și restul biților de la B17 – B6=0	Dacă avem $\overline{DCR}=1$ atunci $i_4 \rightarrow$ +1 NA	—		
					6. ODCU IIE=1 și B18=1 și B13=1 și restul biților de la B17 – B6 = 0	Dacă avem DCR=1 atunci $i_4 \rightarrow$ +1NA	—		

FAZA	i_0	i_1	i_2	i_3	Dacă se începe cond. de mai jos se exec microoper. de la i_4-i_7	i_4	i_5	i_6	i_7
ADUCERE FZ1 IIE					7. G1 IN R IIE=1 și B18=1 și B12=1 și restul biților de la B17-B6=0	$i_4 \rightarrow$ A(0)SnR $i_4 \rightarrow$ A(0)MR	$I_5 \rightarrow$ PmSnR $I_5 \rightarrow$ PmMR $i_5 \rightarrow$ PdG1		
					8 G1 SAUR IIE=1 și B18=1 și B11=1 și $\overline{B10}$=1 și $\overline{B9}$=1 și restul biților de la B17-B6=0	—	$I_5 \rightarrow$ PmSnR $I_5 \rightarrow$ PmMR $i_5 \rightarrow$ PdG1		
					9. G2 IN B IIE=1 și B18=1 și $\overline{B11}$=1 și B10=1 și $\overline{B9}$=1 și restul biților de la B17-B6=0	$i_4 \rightarrow$ A(0)B	$I_5 \rightarrow$ PmB $i_5 \rightarrow$ PdG2		
					10. AZQ IIE=1 și B18=1 și $\overline{B11}$=1 și $\overline{B10}$=1 și B9=1 și restul biților de la B17-B6=0	$i_4 \rightarrow$ A(0)Q			
					11. PORP IIE=1 și B18=1 și B8=1 și $\overline{B7}$=1 și $\overline{B6}$=1 restul biților de la B17-B5=0	$i_4 \rightarrow$ PORP a perif. a cărui COD DE SEL se află înscris în instr.	—		
					12. OPRP IIE=1 și B18=1 și $\overline{B8}$=1 și B7=1 și $\overline{B6}$=1 restul biților de la B17-B6=0	$i_4 \rightarrow$ OPRP a perif. a cărui COD de SEL se află înscris în instr.	—		

INSTR IE (continuare) | INTRODUCERE – EXTRAGERE IIE (ADUCERE) | CETA

FAZA	i_0	i_1	i_2	i_3	Dacă se începe/nesc cond. cem. și jos se exec. microop. de la i_4 la i_7	i_4	i_5	i_6	i_7
ADUCERE FZ1 IIE					13. AZF IIE=1 și B18=1 și $\overline{B8}$=1 și $\overline{B7}$=1 și B6=1 și restul biților de la B17–B6=0	i_4 → A(D)FANIB al periferic. al cărui COD de SEL se află în scris în instr.	—		
					14. OFA IIE=1 și B18=1 și $\overline{B8}$=1 și B7=1 și B6=1 și restul biților de la B17–B6=0	Dacă avem FAN A=1 al periferic. al cărui COD de SEL este în instr. i_4 → atunci → INA	—		
					15. OFB IIE=1 și B18=1 și B8=1 și B7=1 și B6=1 și restul biților de la B17–B6=0	Dacă avem FANB=1 al periferic. al cărui COD de SEL este în instr. atunci i_4 → INA	—		
					16. INT IIE=1 și B18=1 și B8=1 și B7=1 și B6=1 și restul biților de la B17–B5=0	—	i_5 → PmQ i_5 → INTROD. al perif. al cărui COD de SEL este în instr.		
					17. EXT IIE=1 și B18=1 și B8=1 și $\overline{B7}$=1 și B5=1 și restul biților de la B17–B6=0	i_4 → i_1 al periferic. al cărui COD de SEL se află în instrucție.	i_5 → t_2 i_5 → PmRB ambele la sf. al cărui COD de SEL se află în instrucție. i_5 → PcA		

INSTR I.E (continuare) INTRODUCERE-EXTRAGERE IIE (ADUCERE) CETA

FAZA	i_0	i_1	i_2	i_3	Dacă se îndeplinesc cond. de mai jos se exec. microop. de la i_4 la i_7	i_4	i_5	i_6	i_7
ADUCERE Fz.1 IIE					18 OTS IIE=1 și B18=1 și B11=1 și B10=1 și $\overline{B9}$=1 și restul biților de la B17 la B6=0	Dacă avem: FANTS=1 atunci i_4 → +1NA	—		
					19. AIRT IIE=1 și B18=1 și $\overline{B11}$=1 și B10=1 și B9=1 și restul biților de la B17 — B6=0	—	—		Idem ca la IRM1, Fz 3 i_7 → A(1)BITR
					20. DITR IIE=1 și B18=1 și B11=1 și $\overline{B10}$=1 și B9=1 și restul biților de la B17-B6=0	—	—		Idem ca la IRM1, Fz. 3 i_7 → A(0)BITR

1. CHEIA | STAREA INITIALĂ | CETA

	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
Cheia actionează un GS. Cheia se actionează numai cînd calculatorul este OPRIT.								
$i_{STIN} \rightarrow$ fz1								
$i_{STIN} \rightarrow$ A(0) BPO								
$i_{STIN} \rightarrow$ A(0) BPOR								
$i_{STIN} \rightarrow$ A(0) BOPR1								
$i_{STIN} \rightarrow$ A(0) BOPR2								
$i_{STIN} \rightarrow$ A(0) BMEM								
$i_{STIN} \rightarrow$ A(0) BINAD								
$i_{STIN} \rightarrow$ A(0) BVIZ								
$i_{STIN} \rightarrow$ A(0) BUNCICL								
$i_{STIN} \rightarrow$ A(0) BINCA								
$i_{STIN} \rightarrow$ A(7) N8								
$i_{STIN} \rightarrow$ A(0) BII								
$i_{STIN} \rightarrow$ A(0) BITR								
$i_{STIN} \rightarrow$ A(0) BFANG								
$i_{STIN} \rightarrow$ A(0) BPREINT								

	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
Cheia acționează un GS. Cheia se acționează numai cînd calculatorul este OPRIT. Cheia acționează un bistabil pt. a se putea sincroniza pornirea calc. cu imp. de tact. $i_{POR} \rightarrow$ A(1) BPOR Dacă BPOR = 1 $i \rightarrow$ A(1) BPO $i \rightarrow$ A(0) BPOR	Dacă nu ar fi prevăzut BPOR și imp. i_{POR} ar fi trimis direct la BPO pentru a-l aduce la unu s-ar putea întîmpla următoarele: Dacă prin i_{POR} bistabilul BPO ar fi adus la unu chiar în timpul apariției unui imp. de tact. i, acesta ar putea trece cu amplitudine mai mică și cum acesta merge la mai multe intrări (+1N8, Δ și la o poartă ȘI) ar putea pe unele să le comande, iar pe altele nu, ceace duce la eroare. Dacă avem BPOR chiar dacă basculeaza lui se produce în timpul unui impuls i, acesta la ieșirea din circ. Și va trebui să aducă la unu BPO și trebuie să facă numai acest lucru. Dacă ar fi redus ca amplit. și nu aduce la unu pe BPO, nu se întîmplă nimic; următorul i îl va aduce sigur. (vezi paragr. despre sincroniz. vol I pag. 301). Lucru similar se petrece și la A(1) BPO resp. A(0) BPOR. Aici nu beneficiem de ceace am descris mai sus, dar aici nu apar erori. Astfel dacă A(1) BPO este comandat, iar A(0) BPOR nu, atunci aceasta din urmă va fi comandat cu imp. următor. Dacă lucrurile stau invers înseamnă că nu a pornit calculatorul și mai apăsăm odată pe buton. Dacă vrem să pornim calculatorul, și acesta să înceapă funcționarea de la F_{Z1} trebuie înainte de cheia PORNEȘTE să fie apăsată cheia STARE INIȚIALĂ. $i - \Delta\, i_H$ Obs. Impulsul i rezultă din i_H și este întîrziat față de acesta pentru a cădea în același moment cu impulsurile $i_0 \ldots i_7$							

3. CHEIA OPRIȘTEI

CETA

	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
Cheia acționează un GS. Cheia se acționează numai când calculatorul este PORNIT Cheia acționează un bistabil. i_{OPR} → A(1) BOPR1 Dacă BOPR1=1 atunci i → A(1) BOPR 2		Bistabilul BOPR2 este introdus din motive de sincronizare. Dacă am avea un singur bistabil și el ar fi pascalat în timpul lui i_7 s-ar putea ca A(0) BPO să fie acționat, iar A(0) BOPR să nu fie, sau invers. În acest caz apar dificultăți. Toate microop. corespunzătoare fazei existente se execută. Pe linică acestea se mai execută cele din prezentul tabel. Calculatorul se oprește la sfîrșitul fazei în timpul căreia a fost manipulată cheia.						Dacă avem BOPR2=1 atunci i_7 → A(0)BPO i_7 → A(0)BOPR1 i_7 → A(0)BOPR2

CFA

4. CHEIA [MEMOREAZĂ]

	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
Cheia acționează un GS. Cheia se acționează numai cînd calculatorul este CERIT. Cheia acționează un bistabil $i_{MEM} \rightarrow A(1)BMEM$ $i_{MEM} \rightarrow fz5$ Dacă BMEM = 1 atunci $-i \rightarrow A(1)$ BPO	$i_0 \rightarrow$ SCRIE $i_0 \rightarrow A(0)RA$ $i_0 \rightarrow A(0)RM$ $i_0 \rightarrow PdNA$ $i_0 \rightarrow PmRA$		$i_2 \rightarrow PmRM$ $i_2 \rightarrow PcSI$				$i_6 \rightarrow +1NA$	$i \rightarrow A(0)BMEM$ $i \rightarrow A(0)BPO$ $i_7 \rightarrow jz1$

Cînd scriem ecuațiile:

La funcția A(1) BPO nu este necesar să apară fz5 BMEM & factorj suficient BMEM deoarece cînd apare BMEM în mod obligatoriu anulează și fz5.

Execuțarea succesiunii de microoperații din acest label va fi asigurată de prezența funcției fz5 BMEM.

5. CHEIA [ÎNCARCĂ ADRESA] CETA

	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
Cheia acționează un GS. Cheia se acționează numai când calc. este OPRIT. Cheia acționează un bistabil $i_{INAD} \rightarrow A(1)BINAD$ $i_{INAD} \rightarrow fz5$ Dacă BINAD=1 atunci $i \rightarrow A(1)BPO$	$i_0 \rightarrow A(0)NA$	—	$I_2 \rightarrow P_m.NA$ $i_2 \rightarrow REG1$	—	—	—	—	$i_7 \rightarrow A(0)B.NAD$ $i_7 \rightarrow A(0)BPO$ $i_7 \rightarrow fz1$

Efectul succesiunii de microoperații va fi asigurată de prezența funcției FZ5 · BINAD.

6. CHEIA VIZUALIZEAZĂ CEFA

	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
Cheia acționează un GS. Cheia se acționează numai cînd calculatorul este OPRIT Cheia acționează un bistabil. $i_{viz.} \rightarrow$ A(1) BViz $i_{viz.} \rightarrow fz5.$ Dacă Bviz = 1 atunci $i \rightarrow$ A(1) BPO	$i_0 \rightarrow$ CITEȘTE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i_0 \rightarrow$ RNA $I_0 \rightarrow$ PmRA	—	—	—	—	—	$i_6 \rightarrow$ +1 NA	$i_7 \rightarrow$ A(0) BViz $i_7 \rightarrow$ A(0) BPO $i_7 \rightarrow fz1$

Execuarea succes. de microoperații, va fi asigurată de prezența funcției Fz5. BViz.

7. CHEIA UN CICLU CETA

	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
Cheia acționează un GS. Cheia se acționează numai când calc. este OPRIT. Cheia acționează un basculabil. $i_{UNCICL} \longrightarrow$ A(1) BUNCICL Dacă BUNCICL=1 atunci $i \rightarrow$ A(1) BPO Obs. Când calc este oprit cu ajutorul cheii OPREȘTE, el rămâne la sfârșitul unei faze. Prin urmare cheia UNCICLU pornește calc. începând cu i_0	—	—	—	—	—	—	—	$i_7 \rightarrow$ A(0) BUNCICL $i_7 \rightarrow$ A(0) BPO
	Efect. succes. Ce microoperații este asigurată de FZ.5 BUNCICL.							

8 CHEIA ÎNCARCĂ A

CETA

	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
Cheia actionează un GS. Cheia se actionează numai cînd calc este OPRIT Cheia actionează un bistab.(: $i_{\overline{INCA}} \rightarrow$ A(1) BINCA $i_{\overline{INCA}} \rightarrow$ fz5 Dacă BINCA = 1 atunci $i \rightarrow$ A(1) BPO obs. Vezi INSTR. RM Nr.4 ÎNCARCĂ A	—	$i_1 \rightarrow$ A(0)SnR $i_1 \rightarrow$ A(0)MR $i_1 \rightarrow$ A(0)A	—	$I_3 \rightarrow$ PmSnR $I_3 \rightarrow$ PmMR $i_3 \rightarrow$ PHG1	Dacă avem $\overline{SnR}=1$ atunci $i_4 \rightarrow$ +MR Dacă avem SnR = 1 atunci $i_4 \rightarrow$ −MR $i_4 \rightarrow$ +1SnA	—	—	$i_7 \rightarrow$ A(0) BINCA $i_7 \rightarrow$ A(0) BPO $i_7 \rightarrow$ fz1

Cu ajutorul cheii se încarcă în A conținutul lui G1. În G1 numărul se scrie în reprez. semn-mărime, iar în A ajunge în reprez. în compl. de 2 modif.

Efect. succes. de microoperații este asigurată de FZ.Ș BINCA

9. CHEIA UN PAS, NORM, N8M

	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7

Este o cheie cu 3 poziții și 3 ieșiri ca mai jos. La cheie avem asociat un generator singular care dă la ieșire impulsul g. Comportarea cheii în funcție de poziție e dată în tabel.

Poziția	Valoarea variabilei la ieșirea		
	NORM	UNP	N8M
NORM	1	0	0
UNP	0	1	0
N8M	0	0	1

1. Dacă avem NORM = 1 atunci $h \longrightarrow i_H$
2. Dacă avem UNP = 1, atunci $g \longrightarrow i_H$
3. Dacă avem N8M = 1 atunci $g \longrightarrow +1N8$

10. CHEILE FZ1 ---- FZ6

-	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7

Prin manipularea oricărei chei FZ1 ... FZ6 se acționează un GS, care stabilește la generatorul de faze GF faza corespunzătoare. Cheile se acționează cînd calc. este oprit.

$i_{FZ1} \longrightarrow fz1$ $\qquad$ $i_{FZ4} \longrightarrow fz4$

$i_{FZ2} \longrightarrow fz2$ $\qquad$ $i_{FZ5} \longrightarrow fz5$

$i_{FZ3} \longrightarrow fz3$ $\qquad$ $i_{FZ6} \longrightarrow fz6$

11. CHEIA AZNA

Prin manipularea cheii AZNA se acționează un GS, care aduce la zero conținutul lui NA. Cheia se acționează cînd calculatorul este oprit.

$i_{AZNA} \longrightarrow A(0)\ NA$

12. CHEIA PRE INTRODUCERE PRI

	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
Cheia acționează un GS. Cheia se acționează numai cînd calculatorul este oprit. Cheia acționează un bistabil. $i_{PRI} \rightarrow$ A(1)BPRI $i_{PRI} \rightarrow$ A(0)BMF $i_{PRI} \rightarrow$ fz5 Dacă BPRI=1 atunci $i \rightarrow$ A(1)BPO	$i_0 \rightarrow$ A(0)N28 $i_0 \rightarrow$ A(0)NA	—	$I_2 \rightarrow$ Pm NA $i_2 \rightarrow$ Pd MF	$i_3 \rightarrow$ +1N28	—	—	—	Dacă avem BPRI=1 atunci $i_7 \rightarrow$ A(0)BPRI Dacă avem BMF=0 atunci $i_7 \rightarrow$ A(1)BMF Dacă avem BMF=1 atunci $i_7 \rightarrow$ A(0)BMF $i_7 \rightarrow$ A(0)BPO $i_7 \rightarrow$ fz1

$A(0)NA = BPRI \cdot Fz5 \cdot i_0$

$Pm\,NA = BPRI \cdot Fz5 \cdot I_2$

$Pd\,MF = BPRI \cdot Fz5 \cdot i_2$

$+1N28 = BPRI \cdot Fz5 \cdot i_3$

$A(0)N28 = BPRI \cdot Fz5 \cdot i_0$

$A(0)BPRI = i_7 \cdot BPRI$

$A(1)BMF = BPRI \cdot Fz5 \cdot \overline{BMF} \cdot i_7$

$A(0)BMF = BPRI \cdot Fz5 \cdot BMF \cdot i_7$

$A(0)BPO = BPRI \cdot Fz5 \cdot BMF \cdot i_7$

$fz1 = BPRI \cdot Fz5 \cdot BMF \cdot i_7$

12. (continuare) cheia PRI (continuare)

	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
Dacă avem BMF = 1 BPRI = 0 Fz5 = 1	$i_0 \rightarrow$ SCRIE $i_0 \rightarrow$ A(0)RA $i_0 \rightarrow$ A(0)RM $i'_0 \rightarrow$ PdNA $I'_0 \rightarrow$ PmRA	—	$I_2 \rightarrow$ PmRM $i_2 \rightarrow$ PdMF	$i_3 \rightarrow$ +1N28	—	—	$i_6 \rightarrow$ +1NA	Dacă avem N29 = 1 atunci $i_7 \rightarrow$ A(1)BPRI

SCRIE = $BMF \cdot \overline{BPRI} \cdot Fz5 \cdot i_0$

A(0)RA = $BMF \cdot \overline{BPRI} \cdot Fz5 \cdot i_0$

A(0)RM = $BMF \cdot \overline{BPRI} \cdot Fz5 \cdot i_0$

Pd NA = $BMF \cdot \overline{BPRI} \cdot Fz5 \cdot i'_0$

Pm RA = $BMF \cdot \overline{BPRI} \cdot Fz5 \cdot I'_0$

Pm RM = $BMF \cdot \overline{BPRI} \cdot Fz5 \cdot I_2$

PdMF = $BMF \cdot \overline{BPRI} \cdot Fz5 \cdot i_2$

+1N28 = $BMF \cdot \overline{BPRI} \cdot Fz5 \cdot i_3$

+1NA = $BMF \cdot \overline{BPRI} \cdot Fz5 \cdot i_6$

A(1)BPRI = $BMF \cdot \overline{BPRI} \cdot Fz5 \cdot N29 \cdot i_7$

13. FAZA ÎNTRERUPERE Fz6 CETA

FAZA	i_0	i_1	i_2	i_3	i_4	i_5	i_6	i_7
ÎNTRERUPERE Fz6	$i_0 \to$ SCRIE $i_0 \to A(0)RA$ $i_0 \to A(0)RM$	—	$I_2 \to P_m RM$ $i_2 \to P_d NA$	—	—	$i_5 \to A(0)NA$	$i_5 \to +1NA$	$i_7 \to fz1$
	-Se dă comanda scrie și se aduc la zero RA și RM	Nu se folosește	-Se trece în RM conținutul lui NA în vederea înscrierii lui la adresa 0000 din M	Nu se folosește	Nu se folosește	-Se aduce la zero conținutul lui NA	-Se mărește cu unu conținutul lui NA care astfel devine 1, în vederea luării instrucției următoare ce la adresa 1 din M	-Se trece la faza ADUCERE a unei instrucții care va urma.

Fz6 se descrie la cap. CHEI, pentru că apariția și executarea ei este întru totul similară cu apariția și executarea unor comenzi determinate de manipularea unor chei.

Capitolul 5

SCRIEREA ECUATIILOR LOGICE ALE FUNCTIILOR DIVERSELOR SEMNALE, PE BAZA TABELELOR DE DESCRIERE A INSTRUCTIILOR SI A COMENZILOR EFECTUATE DE CHEI.STABILIREA INCARCARILOR CIRCUITELOR

In tabelele pentru descrierea instrucţiilor şi a comenzilor date de chei, se arată la ce bornă a blocurilor calculatorului trebuie să ajungă impulsurile de orologiu i_0-i_7, pentru a declanşa microoperaţiile, care compun o anumită fază a unei instrucţii,ceace revine la asigurarea unui conţinut precis de microoperaţii fiecărei faze.

Pe baza acestor date din tabele, trebuie acum să elaborăm o schemă logică prin care impulsurile de orologiu i_0-i_7 să ajungă,la bornele arătate în tabel,numai în cazul instrucţiei, respectiv fazei dorite. Sinteza schemei logice o vom face pe baza unor ecuaţii scrise după tabelele de descriere.

Pentru a putea preciza conţinutul de microoperaţii al fiecărei faze, respectiv al fiecărei instrucţii, e necesar să avem, din punct de vedere logic, variabile, care indică instrucţia, respectiv faza. Astfel precizarea conţinutului, în microoperaţii, a unei instrucţii, este realizată, spre exemplu, la instrucţiile cu referire la memorie, prin codul operaţiei, care se află cuprins în cuvîntul instrucţie. Cînd instrucţia este dusă în registrul instrucţiei, biţii codului operaţiei devin variabile la intrarea decodificatorului codului operaţiei.

Decodificatorul decodifică şi, la ieşirea lui, vom avea potenţial corespunzător lui "unu" la o singură

ieşire, acela care corespunde codului de la intrare. Ieşirea aceasta, pe care o notăm mnemonic, reprezintă chiar variabila, care se introduce în ecuaţiile logice şi care asigură executarea numai a fazelor din care este compusă instrucţia. Ea asigură, totodată, fazelor un conţinut precis în microoperaţii.

In tabelele noastre această variabilă este notată în dreptunghiul punctat al fiecărui tabel. Notaţia este mnemonică pentru ca, din prezentarea ei, să ne putem da seama, printr-o simplă asociaţie, de conţinutul ei. Astfel pentru instrucţia "Memorează conţinutul registrului A", variabila mnemonică este MEA.

Intrucît o instrucţie poate fi formată din mai multe faze, înseamnă că noi avem nevoie de variabile, care arată ce fază, din cele 6, execută calculatorul, pentru instrucţia în curs. Astfel dacă dorim ca o anumită microoperaţie să se execute la un anumit impuls i_0-i_7 într-o anumită fază, a unei anumite instrucţii, vom introduce în ecuaţia corespunzătoare toate aceste mărimi de condiţionare. Spre exemplu dorim ca la instrucţia cu referire la memorie nr.4 "Incarcă A",(INA), în faza de EXECUTIE (Fz3) să se execute, cu impulsul i_1, microoperaţia prin care se aduce la 0 conţinutul registrului A adică A(o)A, (vezi tabelul instrucţiei nr.4) se va scrie ecuaţia:

$$A(o)A = INA.Fz_3.i_1$$

Se vede acum că funcţia (microoperaţia) A(o)A se va executa numai cînd vom avea INA=1, adică avem în registrul instrucţiei codul operaţiei instrucţiei "Incarcă A" şi deci la ieşirea lui, notată cu INA, avem potenţial corespunzător lui "unu" şi cînd avem Fz_3=1,adică generatorul de faze are potenţial corespunzător lui "unu" la ieşirea Fz_3, şi cînd i_1=1, adică atunci cînd

apare impulsul de orologiu i_1.

Evident poate fi necesar ca funcţia A(o)A să fie executată şi cînd se efectuează o altă instrucţie, respectiv o altă fază. Spre exemplu, mai putem avea ecuaţia:

$$A(o)A = MOZ.Fz_4.i_1$$

După cum se vede această ecuaţie corespunde instrucţiei "Măreşte cu 1 şi omite dacă e O", care este instrucţia IRM nr.21.

Prin urmare ecuaţia pentru funcţia de aducere la O a registrului A va arăta în felul următor:

$$A(o)A = INA.Fz_3.i_1 + MOZ.Fz_4.i_1 + \ldots$$

Desigur funcţia A(o)A mai poate avea şi alţi termeni, corespunzători altor instrucţii, respectiv faze. Toţi aceşti termeni, rezultaţi după parcurgerea tuturor tabelelor de descriere, vor fi legaţi între ei prin funcţia SAU şi dau astfel forma finală a funcţiei A(o)A. Cînd căutăm o anumită funcţie în tabel, pe măsură ce scriem termenii corespunzători în ecuaţii, facem un semn în tabel pentru a şti, atunci cînd se face un control, că acel termen a fost deja scris.

Procedăm la fel pentru toate funcţiile aflate în tabele, obţinînd astfel 102 ecuaţii.

Scrierea ecuaţiilor se face cu foarte multă atenţie. Pentru o uşoară găsire a ecuaţiilor, şi funcţiilor, acestea se aranjează în ordine alfabetică. In cadrul ecuaţiei, termenii pot fi aranjaţi în ordinea impulsurilor de orologiu, iar termenii care conţin acelaşi impuls de orologiu, pot fi aranjaţi în ordinea fazelor. In felul acesta se obţine tabelul I cu ecuaţiile principale a tuturor semnalelor (funcţiilor) de comandă, care intervin în calculator.

Cu ocazia scrierii ecuaţiilor verificăm, tot timpul, şi corectitudinea tabelelor de descriere, care trebuie să asigure, după cum s-a precizat mai sus, pentru fiecare instrucţie şi fază, secvenţa precis determinată de microoperaţii, şi care trebuie să se execute numai cînd ne aflăm în cursul instrucţiei şi fazei corespunzătoare.

Se vede din tabelul I că anumite ecuaţii sînt scrise concentrat, dar în continuare ele sînt explicitate. Spre exemplu: ecuaţia nr.18 se referă la aducerea la zero a fanioanelor celor 6 periferice prevăzute; periferice, care, fiecare din ele, are atribuit cîte un cod de selecţie. In continuarea acestei ecuaţii sînt scrise toate ecuaţiile rezultate din ea, precum şi ecuaţiile corespunzătoare codurilor de selecţie.

Mai sînt şi alte variabile, care intră în ecuaţii, care sînt de fapt variabile compuse şi în astfel de cazuri întotdeauna în continuarea ecuaţiei, în care apare, se arată explicitarea corespunzătoare. Spre exemplu în ecuaţia nr.36 (A(1)BOM) apar variabilele compuse OM, respectiv ZA, pentru care se arată ecuaţiile logice corespunzătoare.

Adesea în ecuaţii, un anumit număr de factori, care aparţin unui termen al ecuaţiei, sînt prinşi sub o acoladă, deasupra căreia se găseşte denumirea mnemonică a instrucţiei de bază, care este microprogramată prin factorii de sub acoladă.

In ecuaţii se găsesc anumite variabile, cum sînt de ex. IRM6, IRM12, MIC, etc., care sînt variabile compuse, rezultate din prinderea împreună prin funcţia SAU, a unor variabile ce reprezintă semnale (mnemonice) pentru diferite instrucţii. Alcătuirea acestor variabile compuse, este arătată în tabelul II. Prin acest pro-

cedeu se obţine o oarecare simplificare a ecuaţiilor principale.

Acum, în cele 102 ecuaţii principale se găseşte descris riguros absolut întregul proces de comandă al calculatorului, adică ele descriu blocul de comandă (BCC) al calculatorului.

De fapt singurele ecuaţii care lipsesc sînt acelea care corespund decodificatorului codului operaţiei. Acestea nu au mai fost scrise, deoarece ele sînt extrem de uşor de scris şi ele ne arată cum se obţin variabilele, care definesc diversele instrucţii, la ieşirea acestui decodificator, cînd considerăm ca variabile de intrare biţii din care este format codul de operaţie corespunzător. Spre exemplu codul de operaţie pentru instrucţia "adună la conţinutul lui A" este 01100 şi el se află plasat în biţii B23, B22, B21, B20 şi B19 ai registrului instrucţiei (RI). In acest caz ecuaţia care corespunde funcţiei la ieşire a decodificatorului codului operaţiei este:

$$ADA = \overline{B23} \cdot B22 \cdot B21 \cdot \overline{B20} \cdot \overline{B19}$$

In mod identic se pot scrie şi celelalte ecuaţii corespunzătoare celorlalte instrucţii, iar în final, pe baza acestor ecuaţii, se face sinteza schemei decodificatorului codului operaţiei.

Este uşor de văzut că prin procedeul nostru, de a utiliza direct variabila mnemonică, am obţinut o scriere mai concisă a ecuaţiilor, care descriu întregul proces de comandă al calculatorului.

Inainte de a trece la sinteza schemei blocului circuitelor de comandă (BCC), care se face pe baza celor 102 ecuaţii, trebuie să procedăm la minimizarea acestor ecuaţii. Minimizarea va fi îmbinată cu determinarea încărcării circuitelor logice.

Pentru efectuarea minimizării trebuie să precizăm criteriile, după care ne conducem cînd o efectuăm.

Calculatorul CETA este realizat cu circuite logice NU-SI (NS). Numărul de intrări la aceste circuite poate fi mărit prin adăugarea cîte unei diode pentru fiecare intrare în plus. Un circuit NU-SI este compus dintr-un tranzistor, 3 diode cu siliciu, 4 diode cu germaniu şi 3 rezistenţe. Se vede deci că, din consideraţii economice, trebuie să urmărim să avem un număr cît mai mic de circuite NU-SI, ceiace va duce desigur la un număr mai mare de intrări. Aceasta nu ne deranjează avînd în vedere că ele necesită numai cîte o diodă de fiecare. Procedînd astfel mai avem şi avantajul că în felul acesta obţinem mai puţine etaje la formarea funcţiilor logice şi în consecinţă întîrzieri reduse.

Reducerea numărului de intrări se face prin darea de factori comuni în ecuaţii. Cum pe noi nu ne interesează acest lucru vom lăsa, în esenţă, ecuaţiile în forma în care au fost scrise iniţial; vom căuta însă ca termenii identici din diferitele ecuaţii să fie realizaţi o singură dată. De altfel în forma în care sînt scrise ecuaţiile din tabelul I, ele pot fi realizate foarte uşor cu circuitele NU-SI prin 2 etaje. Astfel fiecare termen dintr-o ecuaţie se realizează cu cîte un circuit NU-SI, care are atîtea intrări cîţi factori are termenul respectiv. Ieşirile acestor circuite NU-SI sînt apoi introduse într-un alt circuit NU-SI, care, datorită inversiunii, formează funcţia SAU a termenilor, adică ne furnizează la ieşirea,funcţie dorită.

Pentru găsirea termenilor identici,sau cum îi mai numim "semnale comune" (SC) alcătuim un nou tip de tabel, anume tabelul III, care se întocmeşte cîte unul pentru fiecare impuls de orologiu i sau i'.

Tabelul III are atîtea coloane cîte faze avem. Ele permit acum să trecem în tabel toţi termenii din cele 102 ecuaţii, care conţin în ei impulsul de orologiu căruia îi este destinat tabelul. Termenii sînt acum distribuiţi şi după faze, pe coloane. Astfel în coloanele respective trecem numai factorii din termeni, care au mai rămas, după scoaterea impulsului de orologiu i şi a fazei. In dreptul factorilor, care au rămas, trecem în paranteză şi numărul ecuaţiei căreia aparţine termenul. Dacă termenul din ecuaţie este format numai din factorii i şi Fz atunci în locul respectiv în tabelul III se trece 1. Se vede că acest tabel poate fi uşor întocmit pe baza tabelului I. In felul acesta reuşim să grupăm pe coloane un număr relativ mic de termeni şi putem observa foarte uşor semnalele comune SC (termenii identici). Cercetăm acum coloanele şi dacă găsim termeni identici tăiem cu linii poziţiile corespunzătoare, iar termenul respectiv îl scriem o singură dată în partea de jos a coloanei, scriind lîngă el, în paranteză, numărul ecuaţiilor în care acesta să găseşte, corespunzător poziţiile tăiate.

Rezultă bineînţeles mai multe tabele III, dar noi prezentăm numai unul drept model.

Intocmim acum tabelul IV în care trecem în ordine alfabetică toate semnalele comune rezultate din tabelele III.

Se vede uşor că pentru obţinerea sintezei schemei BCC va trebui să facem sinteza corespunzătoare ecuaţiilor din tabelele I, II şi IV, care reprezintă forma minimizată.

O problemă importantă, ce trebuieşte rezolvată atunci cînd se proiectează schema logică a unui calculator, este acea a <u>stabilirii încărcărilor diverselor</u>

circuite logice, care produc diferitele variabile şi funcţii logice.

Stabilirea încărcărilor este necesară, deoarece numai aşa putem şti ce putere trebuie să aibă circuitele cu care formăm impulsurile respective. Astfel un circuit NS3 poate comanda 1o intrări, iar un circuit NS6 poate comanda 2o intrări.Se poate întîmpla ca numărul de intrări, care trebuiesc comandate să depăşească chiar numărul de intrări ce pot fi comandate de un circuit NS6. In acest caz intrările ce trebuiesc comandate se repartizează pe mai multe circuite NS6, circuite care sînt comandate pe la intrare,toate,cu acelaşi semnal.

In cele ce urmează vom analiza problema încărcărilor în cazul circuitelor aflate în blocul circuitelor de comandă, respectiv a încărcării provocate de BCC asupra circuitelor din alte blocuri. Problema se pune însă similar şi în cazul proiectării schemelor celorlalte blocuri ale calculatorului.

La stabilirea acestor încărcări, în cazul BCC,vom pleca de la ecuaţiile logice cuprinse în tabelul I şi în tabelul II. In aceste tabele sînt cuprinse absolut toate variabile şi funcţiile,care apar în complexul proces de comandă al calculatorului.

Să luăm spre exemplu funcţia:

$$PdB = i_5.Fz_1.GMO.B4.B5.B6+i_5.Fz_4.MOZ+i_6.Fz_3.CBO+ \ldots$$

Dacă facem sinteza schemei logice, care corespunde acestei ecuaţii, utilizînd circuite NU-SI (NS), obţinem fig.5.1

In ecuaţia de mai sus avem 11 variabile distincte dintre care una (i_5) apare de două ori. Aceste variabile, în schema logică a calculatorului, sînt reprezentate de ieşirile unor circuite. Astfel variabilele i_5, i_6

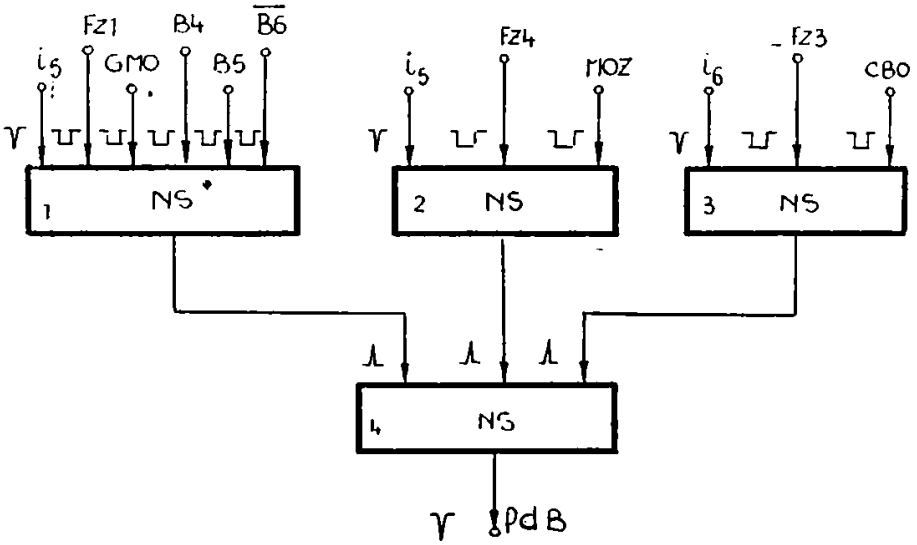

Fz1
B4
$\overline{B6}$
i_5
GM0
B5
1
NS
Fz4
i_5
M0Z
2
NS
Fz3
i_6
CB0
3
NS
4
NS
pd B

Fig.5.1

le găsim la ieşirile corespunzătoare a unor circuite NS din generatorul de impulsuri de orologiu; variabilele GMO, MOZ, le găsim la ieşirile corespunzătoare a unor circuite NS din decodificatorul codului operaţiei; variabilele Fz_1, Fz_3, Fz_4, le găsim la ieşirile unor circuite NS din generatorul de faze; variabilele B4, B5, B6 le găsim la ieşirile unor circuite NS din registrul instrucţiei, ş.a.m.d.

Oricare dintre aceste variabile poate apărea şi în alte funcţii. Dacă, spre exemplu, variabila Fz_1 mai apare încă de 30 ori în alte funcţii, înseamnă că circuitul NS din GF la ieşirea căruia avem reprezentată variabila Fz_1 trebuie să fie în măsură să alimenteze 31 de intrări. Cum fiecare intrare consumă un anumit curent, circuitul NS respectiv va trebuie să poată furniza de 31 ori acest curent. El va fi un circuit NS de putere. Aceeaşi situaţie apare la ieşirile circuitelor care produc oricare dintre variabile.

Inafară de aceasta unii termeni ai ecuaţiilor se repetă în alte funcţii, dar pentru simplificare noi îi producem o singură dată şi îi numim "semnale comune" (SC). Astfel, ca exemplu, putem considera că termenul $i_5.Fz_1.GMO.B4.B5.B6$ este un "semnal comun" şi apare în încă 9 funcţii. Aceasta înseamnă că circuitul care îl formează, (circuitul NS cu nr.1 din fig.5.1), trebuie să poată alimenta 10 intrări.

Funcţia însăşi, în cazul nostru PdB, apare la ieşirea unui circuit NS. Semnalul care reprezintă această funcţie este dus la o bară care uneşte intrările a 24 circuite NS, prin urmare circuitul la ieşirea căruia se formează funcţia PdB trebuie să poată alimenta 24 de intrări.

Dacă am utiliza peste tot circuite, care pot alimenta, spre exemplu, 100 intrări, atunci nu ne-am mai

pune problema găsirii încărcărilor. Dar o astfel de soluţie este neeconomică pentrucă un circuit, care poate alimenta 100 de intrări, este mult mai scump decît unul care poate alimenta 3 intrări, ori într-o schemă logică deobicei circuitele, care trebuie să alimenteze multe intrări nu sînt prea multe, din contră, foarte multe sînt acelea care alimentează puţine intrări.

Dacă utilizăm circuite, la care conectăm încărcări mai mari decît acelea pentru care au fost dimensionate, apar întîrzieri la trecerea semnalelor, sau semnalul la ieşire nu se mai formează. Dacă la ieşirea unui circuit logic se conectează un număr de intrări, care corespunde puterii sale, dar aceste intrări sînt legate prin fire lungi, atunci circuitul trebuie să fie de putere mai mare deoarece capacităţile parazite introduse, de firele lungi, reprezintă o încărcare suplimentară.

Se vede că alegerea circuitelor sub raportul încărcării lor este o problemă foarte importantă. Fără rezolvarea ei corectă, calculatorul nu poate funcţiona sau va funcţiona defectuos.

Rezolvarea problemei se poate face numai dacă cunoaştem încărcările, care revin fiecărui circuit logic. Va trebui deci să elaborăm o metodă cu ajutorul căreia să stabilim încărcările. Metoda trebuie să fie sistematică pentru ca, ţinînd seama de numărul mare de circuite care intervin într-un calculator şi de complexitatea schemei calculatorului, să nu se strecoare scăpări. In acest scop vom utiliza mai departe tabele în care vom cuprinde toate variabilele şi funcţiile precum şi încărcările circuitelor corespunzătoare.

Astfel pentru găsirea încărcărilor circuitelor care produc impulsurile de orologiu i şi semnalele de fază Fz ne vom folosi de tabelul III, după cum urmează: Intrucît semnalul comun după tăierea din coloană a fost

scris o singură dată în partea de jos a tabelului III, înseamnă că numărînd acum pozițiile rămase, atît cele din partea de sus cît și cele din partea de jos a coloanelor, vom găsi numărul de intrări, care trebuie să fie alimentate de circuitul, care produce impulsul de orologiu, căruia îi este destinat tabelul, adică obținem încărcarea acestui circuit. Numărul care reprezintă încărcarea, îl scriem în paranteză sub impulsul de orologiu al tabelelor III.

Numărăm acum termenii netăiați pe fiecare coloană și aflăm încărcarea pentru circuitul care produce semnalul corespunzător. Bineînțeles că încărcarea totală pentru circuitele care produc de exemplu semnalul Fz_1 va rezulta din adunarea încărcărilor corespunzătoare din fiecare tabel pentru impulsurile de orologiu.

Ca tabel ajutător întocmim tabelul V. Acesta este întocmit pe baza tabelului II. Astfel tabelul V va conține variabilele, care apar în tabelul II împreună cu încărcările lor. Urmărirea acestor tabele face de prisos comentariile.

Pentru a găsi acum încărcările circuitelor, care produc fiecare variabilă, alcătuim tabelul VI. Acesta, în prima coloană, conține toate variabilele, care intervin în ecuații, în ordine alfabetică.

Vom rezerva în tabelul VI o coloană divizată în 29 subcoloane corespunzătoare încărcărilor rezultate în tabel III.

Intr-o altă coloană a tabelului VI, vom introduce încărcările rezultate din tabelul V, adică acelea care corespund variabilelor, din tab.II, variabile care pot apărea cu încărcări și în tab.III.

Noi am mai văzut că din necesitățile de micșorare a numărului de circuite logice am căutat termenii iden-

tici respectiv semnalele comune (SC) în diverse ecuaţii, în scopul de a-i forma prin circuite logice o singură dată. Ieşirile circuitelor logice care formează semnalele comune vor avea şi ele diferite încărcări. Aceste încărcări vor fi trecute într-o coloană distinctă chiar în tab.IV în care se află înşiruite semnalele comune.

Dacă analizăm atent, se vede că încărcările circuitelor, care formează variabilele, şi care intră ca factori în semnalele comune, au fost deja prinse în prima coloană a tabelului VI şi anume pe baza datelor culese din tabelul III.

In ceea ce priveşte funcţiile, care comandă borne ale altor blocuri, le vom stabili încărcările atunci cînd blocurile logice respective vor avea schemele logice proiectate şi le vom trece într-un tabel aparte, tabelul VII. Spre exemplu, încărcarea circuitului care produce funcţia A(o)A este 1, pentru motivul că în schema registrului A există circuitele de putere necesare pentru comandareatuturor rangurilor. Prin urmare numai acesta din urmă produce încărcare pentru circuitul care formează funcţie A(o)A. Nu acelaşi lucru se petrece spre exemplu cu circuitul care produce funcţia -MR, care are o încărcare de 24, deoarece el comandă direct 24 intrări.

Incărcările din tabele sînt acelea provocate de BCC în alte blocuri, sau a circuitelor din BCC provocate de alte blocuri.

Desigur cînd elaborăm schemele diverselor părţi ale calculatorului avem de a face şi cu alte circuite decît acelea care produc la ieşire variabilele din tab.VI sau VII, spre exemplu în fig.5.1, circuitul NS nr.2 sau nr.3. Incărcările acestora sînt minime şi foarte uşor de stabilit, ca urmare nu le trecem în

tabele, deoarece acestea s-ar complica mult, în mod inutil.

Cea mai mare parte din circuitele, care produc variabile, se află nu în BCC ci în alt bloc. In aceasta din urmă circuitul trebuie de asemenea să comande anumite intrări. Vom avea grijă ca atunci cînd stabilim încărcarea unor circuite din aceste blocuri să ţinem seama atît de încărcarea produsă în blocul din care face parte cît şi de acea din tabelul VI, provocată de BCC. Incărcările corespunzătoare pot fi notate pe schemă la ieşirea circuitului respectiv. Spre exemplu, variabila B3 este formată de ieşirea de 1 a rangului 3 a registrului instrucţiei (RI). Această ieşire este încărcată cu o intrare corespunzătoare legării sale printr-un circuit SI la SCI1 şi cu 11 intrări în BCC. Prin urmare încărcarea totală va fi 1+11=12.

Tot timpul trebuie să acordăm atenţie stabilirii încărcărilor cerute pentru ca funcţionarea calculatorului construit să fie sigură.

Rezumat cu privire la conţinutul celor 7 tabele

Tab.I

Conţine ecuaţiile scrise pe baza descrierii prin tabele a fazelor instrucţiilor.

Pe baza acestui tabel se face sinteza schemelor BCC.

Tab.II

Conţine ecuaţiile variabilelor compuse formate din mnemonicile instrucţiilor.

Pe baza acestui tabel se face sinteza schemelor care produc variabilele compuse. Acestea fac parte şi ele din BCC.

Tab.III

Sînt în număr de 29. Cu ajutorul lor se face o analiză a ecuaţiilor din tab.I sub aspectul găsirii "semnalelor comune" şi a încărcării circuitelor care produc diversele variabile.

Tabelul serveşte la alcătuirea tabelului VI, în vederea stabilirii încărcărilor.

Tab.IV

Conţine termenii denumiţi "semnale comune", stabiliţi pe baza tabelelor III, şi încălcările circuitelor care produc aceste semnale comune.

Tabelul serveşte la stabilirea încărcărilor circuitelor respective.

Tab.V

Conţine variabilele care intră în variabilele compuse din tabelul II şi încărcările circuitelor care produc aceste variabile.

Tabelul serveşte la alcătuirea tabelului VI în vederea stabilirii încărcărilor. E nevoie de tabelul V pentru că încărcările circuitelor pot proveni atît de la formarea variabilelor compuse cît şi a altor funcţii cuprinse în tabelul III.

Tab.VI

Conţine variabilele din ecuaţiile tabelului I, afară de semnalele comune şi cele 102 funcţii descrise de ecuaţiile din tabelul I, precum şi încărcările circuitelor care produc variabilele.

El este alcătuit pe baza tabelului III şi V şi serveşte la stabilirea încărcărilor circuitelor.

Tab.VII

Conţine cele 102 funcţii din tabelul I şi încărcările circuitelor la ieşirea cărora obţinem funcţia. El se alcătuieşte ţinînd seama de încărcările pe care le introduc bornele de intrare din celelalte blocuri (afară

de BCC), comandate de circuitele din BCC, care produc funcţiile.

Tabelul serveşte la stabilirea încărcării circuitelor.

Prin urmare, în final, încărcările circuitelor vor fi găsite în tab.IV, VI şi VII.

Aceste încărcări servesc atît pentru dimensionarea circuitelor logice din BCC, cum sînt cele care produc funcţiile (tab.IV, VI şi VII) cît şi pentru dimensionarea circuitelor din alte blocuri (tab.VI). In acest din urmă caz încărcarea luată din tab.VI se adaugă la încărcarea produsă în blocul respectiv pentru a determina încărcarea totală a circuitului considerat.

In cele ce urmează se prezintă tabelele I, II, IV şi V complet, iar celelalte, pentru economie de spaţiu, numai parţial.

scrise pe baza tabelelor de descriere a instructiilor si a comenzilor efectuate de chei.

Ec. 1.

$$A(0)A = i_1 \cdot Fz_3 \cdot IDA + i_1 \cdot fz_3 INA + i_1 \cdot Fz_3 \cdot INM + i_1 Fz_4 \cdot MOZ +$$
$$+ i_1 \cdot Fz_5 \cdot BINCA +$$
$$+ i_1' \cdot Fz_3 \cdot MRA +$$
$$+ i_2 \cdot Fz_3 \cdot MOZ +$$
$$+ i_3 \cdot Fz_3 \cdot CBO + i_3 \cdot Fz_3 \cdot SAI +$$
$$+ i_4 \cdot Fz_1 \cdot GMO \cdot \overbrace{B13 \cdot \overline{B14}}^{AZA} + i_4 \cdot Fz_4 \cdot CBO +$$
$$+ i_5 \cdot Fz_4 \cdot MOZ$$

Ec. 2.

$$A(0)AdIdRI = i_1 \cdot Fz_2 \cdot IRM12$$

Ec. 3.

$$A(0)B = i_0 \cdot Fz_3 \cdot MOZ +$$
$$+ i_1 \cdot Fz_3 \cdot INB + \overbrace{AZB}^{} + i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B9} \cdot B10 \cdot \overline{B11} \cdot B18}^{G2INB}$$
$$+ i_4 \cdot Fz_1 \cdot GMO \cdot B15 \cdot \overline{B16}$$

Ec 4

$$A(0)BOM = i_0 \cdot Fz_1$$

Ec. 5.

$$A(0)BFANG = i_7 \cdot Fz_1 \cdot MIC \cdot BFANG \cdot BITR + i_7 \cdot Fz_1 \cdot SLT \cdot BFANG \cdot BITR \cdot \overline{Id} +$$
$$+ i_7 \cdot Fz_2 \cdot SLT \cdot BFANG \cdot BITR \cdot \overline{Id} + i_7 \cdot Fz_3 \cdot IRM5 \cdot BFANG \cdot BITR +$$
$$+ i_7 \cdot Fz_3 \cdot IRM7 \cdot BFANG \cdot BITR + i_7 \cdot Fz_3 \cdot IRM10 \cdot BFANG \cdot BITR +$$
$$+ i_7 \cdot Fz_4 \cdot IRM6 \cdot BFANG \cdot BITR +$$
$$+ i_{ST\ INIT}$$

Ec. 6

$$A(0)BII = i_H \cdot BII \cdot BPO +$$
$$+ i_{ST.\ INIT}$$

Ec. 7

$$A(o)BINAD = i_7 \cdot Fz_5 \cdot BINAD + i_{ST\ INIT}$$

Ec 8

$$A(o)BINCA = i_7 \cdot Fz_5 \cdot BINCA + i_{ST\ INIT}$$

Ec. 9

$$A(o)BITR = i_7 \cdot Fz_1 \cdot IIE \cdot \overbrace{B9 \cdot \overline{B10} \cdot B11 \cdot B8}^{BITR} + i_7 Fz_1 \cdot MIC \cdot BFANG \cdot BITR + i_7 \cdot Fz_1 \cdot SLT \cdot BFANG \cdot BITR \cdot \overline{Id} + i_7 \cdot Fz_2 \cdot SLT \cdot BFANG \cdot BITR \cdot \overline{Id} + i_7 \cdot Fz_3 \cdot IRM5 \cdot BFANG \cdot BITR + i_7 \cdot Fz_3 \cdot IRM7 \cdot BFANG \cdot BITR + i_7 \cdot Fz_3 \cdot IRM10 \cdot BFANG \cdot BITR + i_7 \cdot Fz_4 \cdot IRM6 \cdot BFANG \cdot BITR + i_{ST\ INIT}$$

Ec. 10a

$$A(o)BMEM = i_7 \cdot Fz_5 \cdot BMEM + i_{ST\ INIT}$$

Ec. 10b

$$A(o)BMF = \overbrace{i_7 \cdot Fz5 \cdot BMF \cdot BPRI}^{SC75}$$

Ec. 11

$$A(o)BOPR1 = i_7 \cdot BOPR2 + i_{ST\ INIT}$$

Ec. 12

$$A(o)BOPR2 = i_7 \cdot BOPR2$$

Ec. 13

$$A(o)BPO = i_7 \cdot Fz_1 \cdot IIE \cdot \overbrace{B17 \cdot B18}^{OPR} + i_7 \cdot Fz_5 \cdot BCH + \overbrace{i_7 \cdot BOPR2}^{SC54} + i_7 \cdot BUNCICL + \underbrace{i_7 \cdot Fz5 \cdot BMF \cdot BPRI}_{SC75} + \underbrace{i_{ST\ INIT}}_{SC71} + SC54'$$

Ec. 14 a

$$A(0)\,BPOR = \overbrace{i \cdot BPOR}^{SC1} + \underbrace{i_{ST\,INIT}}_{SC71}$$

Ec. 14b

$$A(0)\,BPRI = i_7 \cdot BPRI$$

Ec. 15

$$A(0)\,BUNCICL = \overbrace{i_7 \cdot BUNCICL}^{SC54'} + \underbrace{i_{ST\,INIT}}_{SC71}$$

Ec. 16

$$A(0)\,BVIZ = \overbrace{i_7 \cdot Fz_5 \cdot BVIZ}^{SC67} + \underbrace{i_{ST\,INIT}}_{SC71}$$

Ec. 17

$$A(0)\,DCR = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B16 \cdot B18}^{AZDC} + i_5 \cdot Fz_3 \cdot ADA + \underbrace{i_5 \cdot Fz_3 \cdot IMP}_{SC41'} + i_5 \cdot Fz_3 \cdot MRA + i_5 Fz_3 \cdot SCA$$

Ec. 18

$$A(0)\,FAN(1 \div 6) = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot \overline{B8} \cdot B18}^{AZF} \cdot CS(1 \div 6)$$

Ac. ecuație se defalcă în următoarele:

a. $A(0)\,FANA1 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot \overline{B8} \cdot B18}^{AZF} \cdot CS1$

b. $A(0)\,FANA2 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot \overline{B8} \cdot B18}^{AZF} \cdot CS2$

c. $A(0)\,FANA3 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot \overline{B8} \cdot B18}^{AZF} \cdot CS3$

d. $A(0)\,FANA4 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot \overline{B8} \cdot B18}^{AZF} \cdot CS4$

e. $A(0)\,FANA5 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot \overline{B8} \cdot B18}^{AZF} \cdot CS5$

f. $A(0)\,FANA6 = i_4 \cdot Fz_1 \cdot IIE \cdot B6 \cdot \overline{B7} \cdot \overline{B8} \cdot B18 \cdot CS6$

g. $A(0)\,FANB1 = A(0)\,FANA1$

h. $A(0)\,FANB2 = A(0)\,FANA2$

i. $A(0)\,FANB3 = A(0)\,FANA3$

j. $A(0)\,FANB4 = A(0)\,FANA4$

k. $A(0)\,FANB5 = A(0)\,FANA5$

l. $A(0)\,FANB6 = A(0)\,FANA6$

$CS1 = Bo \cdot \overline{B_1} \cdot \overline{B_2}$ $\quad CS2 = \overline{Bo} \cdot B_1 \cdot \overline{B_2}$ $\quad CS3 = Bo \cdot B_1 \cdot \overline{B_2}$

$CS4 = \overline{Bo} \cdot \overline{B_1} \cdot B_2$ $\quad CS5 = Bo \cdot \overline{B_1} \cdot B_2$ $\quad CS6 = \overline{Bo} \cdot B_1 \cdot B_2$

Ec. 19

$$A(0)MR = \overbrace{i_0 \cdot Fz_3 \cdot MRA}^{SC5} + \overbrace{i_1 \cdot Fz_3 \cdot IRM1}^{SC15} + \overbrace{i_1 \cdot Fz_5 \cdot BINCA}^{SC16} + \overbrace{i_2 \cdot Fz_3 \cdot MOZ}^{SC18} + \overbrace{i_2 \cdot Fz_1 \cdot CBO}^{SC20} + \overbrace{i_4 \cdot Fz_1 \cdot GMO \cdot \overbrace{B10 \cdot \overline{B11}}^{AZMR} + i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B12 \cdot B18}^{G1INR}}^{SC28} + \overbrace{i_4 \cdot Fz_4 \cdot MOZ}^{SC30} + \overbrace{i_5 \cdot Fz_3 \cdot CRO}^{SC41} + \overbrace{i_5 \cdot Fz_3 \cdot SIA}^{SC42} + \overbrace{i_7 \cdot Fz_3 \cdot MOZ}^{SC61}$$

Ec. 20

$$A(0)NA = i_0 \cdot Fz_5 \cdot BINAD + i_0 \cdot Fz5 \cdot BPRI + i_2 \cdot Fz_3 \cdot SSP + i_5 \cdot Fz_1 \cdot SLT + i_5 \cdot Fz_2 \cdot SLT + i_5 \cdot Fz_8 + i_{AZNA}$$

Ec. 21a

$$A(0)N24 = i_2 \cdot Fz_3 \cdot INM + \overbrace{i_5 \cdot Fz_3 \cdot IMP}^{SC41'}$$

Ec. 21b

$$A(0)N28 = i_7 \cdot FZ5 \cdot BMF \cdot \overline{BPRI} \cdot N28$$

Ec. 22

$$A(0)N8 = i_H \cdot IMP \cdot B11 \cdot BPO$$

Ec. 23

$$A(0)Q = i_1 \cdot Fz_3 \cdot CBO + \overbrace{i_1 \cdot Fz_3 \cdot INQ}^{SC13} + i_1 \cdot Fz_3 \cdot SIA + i_4 \cdot Fz_1 \cdot GMO \cdot \overbrace{B17 \cdot \overline{B18}}^{AZQ} + i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B9 \cdot \overline{B10} \cdot \overline{B11} \cdot B18}^{AZQ} + \underbrace{i_5 \cdot Fz_3 \cdot IMP}_{SC41'}$$

Ec. 24

$$A(0)RA = \overbrace{i_0 \cdot Fz_1}^{SC2} + \overbrace{i_0 Fz_2 \cdot IRM12}^{SC3} + \overbrace{i_0 Fz_5 \cdot IRM11}^{SC4} + \overbrace{i_0 \cdot Fz5 \cdot BMF \cdot \overline{BPRI}}^{SC74} + \underbrace{i_0 Fz_5 \cdot BMEM}_{SC9} + \underbrace{i_0 \cdot Fz_5 \cdot BVIZ}_{SC10} + \underbrace{i_0 \cdot Fz_6}_{SC11}$$

Ec. 25

$$A(0)RI = i_1 \cdot Fz_1$$

Ec. 26

$$A(0)RM = \overbrace{i_0 \cdot Fz_1}^{SC2} + \overbrace{i_0 \cdot Fz_2 \cdot IRM12}^{SC3} + \overbrace{i_0 \cdot Fz_3 IRM11}^{SC4} + i_0 \cdot Fz_5 \cdot BMF \cdot BPRI +$$
$$+ \overbrace{i_0 \cdot Fz_4 \cdot MOZ}^{SC8} + \overbrace{i_0 \cdot Fz_5 \cdot BMEM}^{SC9} + \overbrace{i_0 \cdot Fz_5 \cdot BVIZ}^{SC10} + \overbrace{i_0 \cdot Fz_6}^{SC1}$$

Ec. 27

$$A(0)SnR = \overbrace{i_0 \cdot Fz_3 \cdot MRA}^{SC5} +$$
$$+ \overbrace{i_1 \cdot Fz_3 \cdot INQ}^{SC13} + i_1 Fz_3 \cdot IRM2 + \overbrace{i_1 \cdot Fz_5 \cdot BINCA}^{SC15} +$$
$$+ \overbrace{i_2 \cdot Fz_3 \cdot MOZ}^{SC18} + \overbrace{i_2 \cdot Fz_4 \cdot CBO}^{SC20} + SC28$$
$$+ i_4 \cdot Fz_1 \cdot GMO \cdot \overbrace{\overline{B12}}^{AZSR} + \overbrace{i_4 \cdot Fz_1 \cdot IIE \cdot \overline{B12} \cdot B18}^{G1INR} +$$
$$+ \overbrace{i_4 \cdot Fz_4 \cdot MOZ}^{SC39} +$$
$$+ \overbrace{i_5 \cdot Fz_3 \cdot CBO}^{SC41} + \overbrace{i_5 \cdot Fz_3 \cdot SIA}^{SC42} +$$
$$+ \underbrace{i_7 \cdot Fz_3 \cdot MOZ}_{SC67}$$

Ec. 28

$$A(0)Sn2A = i_4 \cdot Fz_1 \cdot GMO \cdot \overbrace{\overline{B7} \cdot \overline{B8}}^{AZL} +$$
$$+ i_5 \cdot Fz_1 \cdot GDR \cdot \overbrace{\overline{B11}}^{AZL}$$

Ec. 29

$$A(1)B = i_4 \cdot Fz_1 \cdot GMO \cdot \overbrace{\overline{B15} \cdot B16}^{AUB}$$

Ec. 30

$$A(1)BFANG = i_6 \cdot Fz_1 \cdot MIC \cdot FANG + i_6 \cdot Fz_1 \cdot SLT \cdot FANG +$$
$$i_6 \cdot Fz_2 \cdot SLT \cdot FANG + i_6 \cdot Fz_3 \cdot IRM5 \cdot FANG +$$
$$i_6 \cdot Fz_3 \cdot IRM7 \cdot FANG + i_6 \cdot Fz_3 \cdot IRM10 \cdot FANG +$$
$$i_6 \cdot Fz_4 \cdot IRM8 \cdot FANG$$
$$FANG = FANA1 + FANA2 + FANA4 +$$
$$+ FANA4 + FANA5 + FANA6$$

Ec. 31

$$A(1)BII = i_3 \cdot Fz_4 \cdot IMP \cdot \overline{N24} + i_6 \cdot Fz_3 \cdot INM \cdot \overline{N24}$$

$$N24 = \overline{A_0} \cdot \overline{A_1} \cdot \overline{A_2} \cdot A_3 \cdot A_4$$

Ec. 32

$$A(1)BINAD = i_{INAD}$$

Ec 33

$$A(1)BINCA = i_{INCA}$$

Ec. 34

$$A(1)BITR = i_7 Fz_1 \cdot IIE \cdot \overbrace{B9 \cdot B10 \cdot \overline{B11} \cdot B18}^{AITR}$$

Ec. 35 a

$$A(1)BMEM = i_{MEM}$$

Ec. 35b

$$A(1)BMF = i_7 \cdot Fz_6 \cdot \overline{BMF} \cdot BPRI$$

Ec. 36

$$A(1)BOM = i_4 \cdot Fz_1 \cdot GMO \cdot BO \cdot B9 \cdot Sn2A + i_4 \cdot Fz_1 \cdot GMO \cdot \overline{BO} \cdot B9 \cdot \overline{Sn2A} +$$
$$+ i_5 \cdot Fz_1 \cdot GMO \cdot BO \cdot B3 \cdot OM \cdot SnA \cdot 0A + i_5 \cdot Fz_1 \cdot GMO \cdot BO \cdot B3 \cdot \overline{OM} \cdot 0A +$$
$$+ i_5 \cdot Fz_1 \cdot GMO \cdot BO \cdot \overline{B3} \cdot OM \cdot SnA + i_5 \cdot Fz_1 \cdot GMO \cdot \overline{BO} \cdot B3 \cdot \overline{0A} +$$
$$+ i_5 \cdot Fz_1 \cdot GMO \cdot \overline{BO} \cdot OM \cdot \overline{SnA} +$$
$$+ i_6 \cdot Fz_1 \cdot GMO \cdot BO \cdot B1 \cdot \overline{ZA} + i_6 \cdot Fz_1 \cdot GMO \cdot BO \cdot \overline{B1} \cdot \overline{B3} \cdot \overline{B9} \cdot \overline{OM} +$$
$$+ i_6 \cdot Fz_1 \cdot GMO \cdot \overline{BO} \cdot B1 \cdot ZA$$

Obs. $OM = B4 \cdot B5 \cdot B6$ (când OM=1 este microprogramată instr. OA+)
$ZA = \overline{Rg \cdot CA} \cdot \overline{OA} \cdot \overline{1A} \cdots \overline{21A} \cdot \overline{22A}$; $\overline{OM} = B_4 \cdot B_5 \cdot B6$

Ec. 37

$$A(1)BOPR1 = i_{OPR}$$

Ec. 38

$$A(1)BOPR2 = i \cdot BOPR1 \qquad i = \Delta i_H$$

Ec. 39

$$A(1)BPO = \overbrace{i \cdot BPOR + i \cdot BCH}^{\text{SC1}} + i\,BUNCICL \;;\; i = \Delta i_H$$

Ec. 40a

$$A(1)BPOR = i_{POR}$$

Ec. 40b

$$A(1)BPRI = i_7 \cdot FZ5 \cdot BMF \cdot \overline{BPRI} \cdot N28$$

Ec. 41

$$A(1)BUNCICL = i_{UNCICL}$$

Ec. 42

$$A(1)BVIZ = i_{VIZ}$$

Ec. 43

$$A(1)DCR = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B15} \cdot \overline{B18}}^{\text{AUDC}} +$$
$$+ i'_6 \cdot Fz_3 \cdot IMP \cdot \overline{Sn2A} \cdot \overline{(01-10)SnA} +$$
$$+ i_7 \cdot Fz_3 \cdot ADA \cdot (01-10)SnA + i_7 \cdot Fz_3 \cdot MRA \cdot (01-10)SnA +$$
$$+ i_7 \cdot Fz_3 \cdot SCA \cdot (01-10)SnA$$

Ec. 44

$$A(1)Rg.CQ = i'_3\, Fz_4 \cdot IMP \cdot \overline{Sn2A}$$

Ec. 45

$$A(1)SnR = i_7 Fz_3 SCA$$

Ec. 48

$$A(1)Sn2A = i_4 \cdot Fz_1 \cdot GMO \cdot \overbrace{\overline{B7} \cdot B8}^{\text{AUL}}$$

Ec. 47a

$$A(4)N8 = i_H \cdot INM \cdot BII \cdot BPO$$

Ec. 47b

$$A(7)N8 = i_{ST\ INIT}$$

Ec. 48

$$CIT = \overbrace{i_0 \cdot Fz_1}^{SC2} + \overbrace{i_0 \cdot Fz_2 \cdot IRM12}^{SC3} + i_0 \cdot Fz_3 \cdot IRM4 + i_0 \cdot Fz_3 \cdot IRM7 + \underbrace{+ i_0 \cdot Fz_5 \cdot BVIZ}_{SC10}$$

Ec. 49

$$CMSn2A = i_4 \cdot Fz_1 \cdot GMO \cdot \overbrace{B7 \cdot B8}^{CML}$$

Ec. 50

$$DrA = i_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B14 \cdot B18}^{RDA} + i_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B16 \cdot B18}^{DDA} + + i_6 \cdot Fz_1 \cdot GDR \cdot \overbrace{B4 \cdot B18}^{RDA} + i_6 \cdot Fz_1 \cdot GDR \cdot \overbrace{B6 \cdot B18}^{DDA}$$

Ec 51

$$DrA,DrQ = i_1 \cdot Fz_3 \cdot MPQ + \underbrace{+ i_4 \cdot Fz_3 \cdot INM}_{SC33}$$

Ec. 52 a

$$DrQ = i_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B14 \cdot B18}^{RDQ} + i_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B16 \cdot B18}^{DDQ} + + i_6 \cdot Fz_1 \cdot GDR \cdot \overbrace{B4 \cdot B18}^{RDQ} + i_6 \cdot Fz_1 \cdot GDR \cdot \overbrace{B6 \cdot B18}^{DDQ}$$

Ec. 52 b

$$DrTQ = I_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B14 \cdot B18}^{RDQ} + I_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B16 \cdot B18}^{DDQ} +$$

Ec. 53

$$+ I_6 \cdot Fz_1 \cdot GDR \cdot B4 \cdot B18 + I_6 \cdot Fz_1 \cdot GDR \cdot B6 \cdot B18$$

$$Fz_1 = i_4 \cdot Fz_1 \cdot MIC \cdot \overline{BFANG} \cdot \overline{BITR} + i_7 \cdot Fz_1 \cdot SLT \cdot \overline{BFANG} \cdot \overline{BITR} \cdot \overline{Id} +$$

$$+ i_7 \cdot Fz_2 \cdot SLT \cdot \overline{BFANG} \cdot \overline{BITR} \cdot \overline{Id} + i_7 \cdot Fz_3 \cdot IMP \cdot DCR +$$

$$+ i_7 \cdot Fz_3 \cdot IRM5 \cdot \overline{BFANG} \cdot \overline{BITR} + i_7 \cdot Fz_3 \cdot IRM7 \cdot \overline{BFANG} \cdot \overline{BITR} +$$

$$+ i_7 \cdot Fz_3 \cdot IRM10 \cdot \overline{BFANG} \cdot \overline{BITR} + i_7 \cdot Fz_4 IRM8 \cdot \overline{BFANG} \cdot \overline{BITR} +$$

$$+ \overbrace{i_7 \cdot Fz_5 \cdot BINAD}^{SC64} + \overbrace{i_7 \cdot Fz_5 \cdot BINCA}^{SC65} + \overbrace{i_7 \cdot Fz_5 \cdot BMEM}^{SC66} + \overbrace{i_7 \cdot Fz_5 \cdot BVIZ}^{SC67} + i_7 \cdot Fz_6 + \overbrace{i_7 \cdot Fz_5 \cdot BMF \cdot BPRI}^{SC75} +$$

$$+ \underbrace{i_{STINIT}}_{SC71} + i_{FZ1}$$

Ec. 54

$$\underline{fz_2} = i_7 \cdot Fz_1 \cdot IRM12 \cdot Id + i_7 \cdot Fz_2 \cdot IRM12 \cdot Id + i_{FZ2}$$

Ec. 55

$$\underline{fz_3} = i_7 \cdot Fz_1 \cdot IRM11 \cdot \overline{Id} + i_7 \cdot Fz_2 \cdot IRM11 \cdot \overline{Id} + i_{FZ3}$$

Ec. 56

$$\underline{fz_4} = i_7 \cdot Fz_3 \cdot CBO + i_7 \cdot Fz_3 \cdot IMP \cdot \overline{DCR} + \overbrace{i_7 \cdot Fz_3 \cdot MOZ}^{SC61} + i_{FZ4}$$

Ec. 57

$$\underline{fz_5} = \overbrace{i_{INAD}}^{SC68} + \overbrace{i_{INCA}}^{SC69} + \overbrace{i_{MEM}}^{SC70} + \overbrace{i_{VIZ}}^{SC73} + i_{FZ5}$$

Ec. 58

$$\underline{fz_6} = \overbrace{i_7 \cdot Fz_1 \cdot MIC \cdot BFANG \cdot BITR}^{SC56} + \overbrace{i_7 \cdot Fz_1 \cdot SLT \cdot BFANG \cdot BITR \cdot \overline{Id}}^{SC55} + \overbrace{i_7 \cdot Fz_2 \cdot SLT \cdot BFANG \cdot BITR \cdot \overline{Id}}^{SC57} + \overbrace{i_7 \cdot Fz_3 \cdot IRM5 \cdot BFANG \cdot BITR}^{SC58} + \overbrace{i_7 \cdot Fz_3 \cdot IRM7 \cdot BFANG \cdot BITR}^{SC59} + \overbrace{i_7 \cdot Fz_3 \cdot IRM10 \cdot BFANG \cdot BITR}^{SC60} + \overbrace{i_7 \cdot Fz_4 \cdot IRM6 \cdot BFANG \cdot BITR}^{SC63} + i_{FZ6}$$

Ec. 59

$$\underline{INTROD(1 \div 6)} = i_5 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot B7 \cdot B8 \cdot B18}^{INT} \cdot CS(1 \div 6)$$

Ac ecuație se desfalcă înurmătoarele:

a. $INTROD1 = i_5 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot B7 \cdot B8 \cdot B18}^{INT} \cdot CS1$

b. $INTROD3 = i_5 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot B7 \cdot B8 \cdot B18}^{INT} \cdot CS3$

c. $INTROD5 = i_5 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot B7 \cdot B8 \cdot B18}^{INT} \cdot CS5$

$CS1 = Bo \cdot \overline{B1} \cdot \overline{B2} \qquad CS3 = Bo \cdot B1 \cdot \overline{B2} \qquad CS5 = Bo \cdot \overline{B1} \cdot B2$

Ec. 60

$$\underline{LEG} = I_1 \cdot Fz_3 \cdot MPQ + \\ + I_4 \cdot Fz_3 \cdot INM$$

Ec. 61

$$\underline{+MR} = \overbrace{i_0 \cdot Fz_4 \cdot CBO}^{SC6} + \\ + i_1 \cdot Fz_4 \cdot IMP \cdot Sn2A + \\ + i_2 \cdot Fz_3 \cdot MRA \cdot \overline{SnR} + i_2 \cdot Fz_4 \cdot MOZ \cdot \overline{SnR} + \\ + i_4 \cdot Fz_1 \cdot GMO \cdot \overbrace{\overline{B10}\,\overline{B11}}^{ADMR} + i_4 \cdot Fz_3 \cdot ADA \cdot \overline{SnR} + \\ + \overbrace{i_4 \cdot Fz_3 \cdot CBO}^{SC30} + i_4 \cdot Fz_3 \cdot IDA + i_4 \cdot Fz_3 \cdot INA \cdot \overline{SnR} + \\ + i_4 \cdot Fz_3 \cdot MOZ \cdot \overline{SnR} + i_4 \cdot Fz_3 \cdot SCA \cdot \overline{SnR} + \overbrace{i_4 \cdot Fz_3 \cdot SAE}^{SC36} + \\ + \overbrace{i_4 \cdot Fz_3 \cdot SAI}^{SC35} + \overbrace{i_4 \cdot Fz_3 \cdot SIA}^{SC38} + i_4 \cdot Fz_5 \cdot BINCA \cdot \overline{SnR} + \\ + \overbrace{i_5 \cdot Fz_4 \cdot CBO}^{SC43} + \\ + \overbrace{i_6 \cdot Fz_4 \cdot MOZ}^{SC51} + \\ + \underbrace{i_7 \cdot Fz_3 \cdot SIA}_{SC62}$$

Ec. 62

$$\underline{-MR} = \overbrace{i_1 \cdot Fz_4 \cdot IMP \cdot \overline{Sn2A}}^{SC14} + \\ + i_2 \cdot Fz_3 \cdot MRA \cdot SnR + i_2 \overbrace{Fz_4 \cdot MOZ \cdot SnR}^{SC21} + \\ + i_4 \cdot Fz_1 \cdot GMO \cdot \overbrace{B10 \cdot B11}^{CMR} + \overbrace{i_4 \cdot Fz_3 \cdot ADA \cdot SnR}^{SC29} + \\ + \overbrace{i_4 \cdot Fz_3 \cdot IMP}^{SC31} + \overbrace{i_4 \cdot Fz_3 \cdot INA \cdot SnR}^{SC32} + \overbrace{i_4 \cdot Fz_3 \cdot MOZ \cdot SnR}^{SC34} + \\ + \overbrace{i_4 \cdot Fz_3 \cdot SCA \cdot SnR}^{SC37} + \overbrace{i_4 \cdot Fz_5 \cdot BINCA \cdot SnR}^{SC40}$$

Ec. 63

$$\underline{OPRP(1 \div 6)} = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6} \cdot B7 \cdot \overline{B8} \cdot B18}^{OPRP} \cdot CS(1 \div 6)$$

Ac. ecuatie se defalcă în următoarele:

a. $OPRP1 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6} \cdot B7 \cdot \overline{B8} \cdot B18}^{OPRP} \cdot CS1$

b. $OPRP2 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6} \cdot B7 \cdot \overline{B8} \cdot B18}^{OPRP} \cdot CS2$

c. $OPRP3 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6} \cdot B7 \cdot \overline{B8} \cdot B18}^{OPRP} \cdot CS3$

d. $OPRP4 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6} \cdot B7 \cdot \overline{B8} \cdot B18}^{OPRP} \cdot CS4$

e. $OPRP5 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6} \cdot B7 \cdot \overline{B8} \cdot B18}^{OPRP} \cdot CS5$

f. $OPRP6 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6} \cdot B7 \cdot \overline{B8} \cdot B18}^{OPRP} \cdot CS6$

$CS1 = B0 \cdot \overline{B1} \cdot \overline{B2}$ $\quad CS2 = \overline{B0} \cdot B1 \cdot \overline{B2}$ $\quad CS3 = B0 \cdot B1 \cdot \overline{B2}$

$CS4 = \overline{B0} \cdot \overline{B1} \cdot B2$ $\quad CS5 = B0 \cdot \overline{B1} \cdot B2$ $\quad CS6 = \overline{B0} \cdot B1 \cdot B2$

Ec 64

$$PdA = \overbrace{i_0 \cdot Fz_4 \cdot MOZ}^{SC8} +$$
$$+ i_1 \cdot Fz_3 \cdot MOZ + i_1 \cdot Fz_3 \cdot MRA +$$
$$+ i_2 \cdot Fz_3 \cdot CBO + i_2 \cdot Fz_3 \cdot MEA + i_2 \cdot Fz_3 \cdot SAI + i_2 \cdot Fz_3 \cdot SIA +$$
$$+ \overbrace{i_4 \cdot Fz_4 \cdot MOZ}^{SC39} +$$
$$+ i_5 \cdot Fz_1 \cdot GMO \cdot \overbrace{B4 \cdot B5 \cdot B6}^{AINB} + i_5 \cdot Fz_1 \cdot GMO \cdot \overbrace{\overline{B4} \cdot B5 \cdot B6}^{AINQ} +$$
$$+ i_5 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot B8 \cdot B18}^{EXT}$$

Ec 65

$$Pd.AdRI = i'_0 \cdot Fz_2 \cdot IRM11 + i''_0 \cdot Fz_3 \cdot IRM11 +$$
$$+ i_3 \cdot Fz_3 \cdot SSP +$$
$$+ i_6 \cdot Fz_1 \cdot SLT + i_6 \cdot Fz_2 \cdot SLT$$

Ec. 66

$$PdB = i_5 \cdot Fz_1 \cdot GMO \cdot \overbrace{B4 \cdot B5 \cdot \overline{B6}}^{BINR} + \overbrace{i_5 \cdot Fz_4 \cdot MOZ}^{SC44} +$$
$$+ i_6 \cdot Fz_3 \cdot CBO$$

Ec 67

$$PdG1 = i_2 \cdot Fz_5 \cdot BINAD + i_2 \cdot Fz_5 \cdot BMEM + \overbrace{\quad}^{G1INQ}$$
$$+ i_3 \cdot Fz_5 \cdot BINCA + i_4 \cdot FZ1 \cdot GMO \cdot \overbrace{\overline{B17} \cdot B18}$$
$$+ i_5 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B9} \cdot \overline{B10} \cdot B11 \cdot B18}^{G1SAUR} + i_5 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B12} \cdot B18}^{G1INR}$$

Ec 68

$$PdG2 = i_2 \cdot FE_3 \cdot MG2 + i_5 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B9} \cdot B10 \cdot \overline{B11} \cdot B18}^{G2INB}$$

Ec. 69a

$$PdMA = i_4 \cdot Fz_3 \cdot MRA$$

Ec 69b

$$PdMF = i_2 \cdot Fz5 \cdot BPRI + i_2 \cdot FZ5 \cdot BMF \cdot \overline{BPRI}$$

Ec 69c

$$PdMQ = \overbrace{i_2 \cdot Fz_3 \cdot MCQ}^{SC17}$$

Ec. 70

$$PdNA = i'_0 \cdot Fz_1 + i'_0 \cdot Fz_2 \cdot SLT + i'_0 \cdot Fz_3 \cdot BMEM + i'_0 \cdot FZ5 \cdot BViz + i'_0 \cdot FZ5 \cdot BMF \cdot \overline{BPRT} +$$
$$+ i_1 \cdot Fz_3 \cdot SSP + \overbrace{\quad\quad}^{NAINQ}$$
$$+ i_5 \cdot Fz_1 \cdot GMO \cdot \overline{B4} \cdot B5 \cdot \overline{B6}$$

Ec. 71

$$PdQ = i_2 \cdot Fz_3 \cdot MG1 + i_2 \cdot Fz_3 \cdot MPQ +$$
$$+ i_3 \cdot Fz_4 \cdot CBO + \overbrace{\quad\quad}^{QINR}$$
$$+ i_5 \cdot Fz_1 \cdot GMO \cdot \overline{B4} \cdot \overline{B5} \cdot B6 + i_5 \cdot FZ1 \cdot GMO \cdot \overbrace{B4 \cdot \overline{B5} \cdot B6}^{QINB} +$$
$$+ i_6 \cdot Fz_3 \cdot SIA$$

Ec 72

A fost suprimată în urma unor modificări

Ec. 73

$$PdSnR = \overbrace{i_2 \cdot Fz_3 \cdot MCQ}^{SC17} + \overbrace{i_4 \cdot Fz_3 \cdot MRA}^{SC34'}$$

Ec 74

$$PmAdIdRI = I_3 \cdot Fz_2 \cdot IRM12$$

Ec. 75

$$PmB = I_1 \cdot Fz_3 \cdot MOZ + \overbrace{\quad\quad\quad\quad\quad\quad}^{SC46}$$
$$+ I_3 \cdot Fz_3 \cdot INB + \overbrace{\quad\quad}^{QINB}$$
$$+ I_5 \cdot Fz_1 \cdot GMO \cdot \overbrace{B4 \cdot \overline{B5} \cdot \overline{B6}}^{AINB} + I_5 \cdot FZ1 \cdot GMO \cdot \overbrace{B4 \cdot \overline{B5} \cdot B6}^{} +$$
$$+ I_5 \cdot FZ1 \cdot IIE \cdot \overbrace{\overline{B9} \cdot B10 \cdot \overline{B11} \cdot B18}^{G2INB}$$

Ec. 76

$$PmMR = \overbrace{I_0 \cdot Fz_4 \cdot MOZ}^{SC12} + \overbrace{I_1 \cdot Fz_3 \cdot MRA}^{SC16} + \overbrace{I_2 \cdot Fz_3 \cdot SAI}^{SC22} + \overbrace{I_3 \cdot Fz_3 \cdot IRM4}^{SC24} + \overbrace{I_3 \cdot Fz_3 \cdot IRM8}^{SC25} + \overbrace{I_3 \cdot Fz_4 \cdot CBO}^{SC26} + \overbrace{I_3 \cdot Fz_5 \cdot BINCA}^{SC27} + \overbrace{I_5 \cdot Fz_1 \cdot GMO \cdot \overline{B4} \cdot B5 \cdot \overline{B6}}^{SC43\ BINR} + \overbrace{I_5 \cdot Fz_1 \cdot GMO \cdot \overline{B4} \cdot \overline{B5} \cdot B6}^{SC47\ BINR} + \overbrace{I_5 \cdot Fz_1 \cdot IIE \cdot \overline{B9} \cdot \overline{B10} \cdot B11 \cdot B18}^{SC48\ G1SAUR} + \overbrace{I_5 \cdot Fz_1 \cdot IIE \cdot \overline{B12} \cdot B18}^{SC49\ G1INR} + \overbrace{I_5 \cdot Fz_4 \cdot MOZ}^{SC30} + \overbrace{I_6 \cdot Fz_3 \cdot CBO}^{SC52} + \overbrace{I_6 \cdot Fz_3 \cdot SIA}^{SC53}$$

Ec. 77

$$PmNA = I_2 \cdot Fz_5 \cdot BINAD + I_2 \cdot Fz_5 \cdot BPRI + I_3 \cdot Fz_3 \cdot SSP + I_6 \cdot Fz_1 \cdot SLT + I_6 \cdot Fz_2 \cdot SLT$$

Ec. 78

$$PmQ = I_2 \cdot Fz_3 \cdot CBO + I_2 \cdot Fz_3 \cdot SIA + I_3 \cdot Fz_3 \cdot INQ + I_3 \cdot Fz_3 \cdot SIA + I_4 \cdot Fz_1 \cdot GMO \cdot \overbrace{\overline{B17} \cdot B18}^{G1INQ} + I_5 \cdot Fz_1 \cdot GMO \cdot \overline{B4} \cdot B5 \cdot B6 + I_5 \cdot Fz_1 \cdot IIE \cdot B6 \cdot B7 \cdot B8 \cdot B18 + I_5 \cdot Fz_1 \cdot GMO \cdot \overbrace{\overline{B4} \cdot B5 \cdot \overline{B6}}^{NAINQ}$$

Ec. 79

$$PmRA = I_0' \cdot Fz_1 + I_0' \cdot Fz_2 \cdot IRM12 + I_0' \cdot Fz_3 \cdot IRM11 + I_0' \cdot Fz_5 \cdot BMEM + I_0' \cdot Fz_5 \cdot BMF \cdot \overline{BPRI} + I_0' \cdot Fz_5 \cdot BVIZ$$

Ec. 80

$$\underline{PmRB\ (1\div6)} = I_5 \cdot Fz_1 \cdot IIE \overbrace{B6 \cdot \overline{B7} \cdot B8 \cdot B18}^{EXT} \cdot CS\ (1\div6)$$

Ac. ecuație se defalcă în următoarele:

a $PmRB2 = I_5 \cdot Fz_1 \cdot IIE \overbrace{B6 \cdot \overline{B7} \cdot B8 \cdot B18}^{EXT} \cdot CS2$

b $PmRB4 = I_5 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot B8 \cdot B18}^{EXT} \cdot CS4$

c. $PmRB6 = I_5 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot B8 \cdot B18}^{EXT} \cdot CS6$

$CS2 = \overline{B0} \cdot B1 \cdot \overline{B2} \qquad CS4 = \overline{B0} \cdot \overline{B1} \cdot B2 \qquad CS6 = \overline{B0} \cdot B1 \cdot B2$

Ec. 81

$$\underline{PmRI} = I_3 \cdot Fz_1$$

Ec. 82.

$$\underline{PmRM} = I_1 \cdot Fz_3 \cdot SSP +$$
$$+ I_2 \cdot Fz_3 \cdot IRM9 + I_2 \cdot Fz_5 \cdot BMEM + I_2 \cdot Fz_6 + I_2 \cdot Fz_5 \cdot BMF \cdot \overline{BPRI} +$$
$$+ I_4 \cdot Fz_3 \cdot MRA + I_4 \cdot Fz_4 \cdot MOZ +$$

Ec. 83

$$\underline{PmSnR} = \overbrace{I_0 \cdot Fz_4 \cdot MOZ}^{SC12} +$$
$$+ \overbrace{I_1 \cdot Fz_3 \cdot MRA}^{SC16} +$$
$$+ \overbrace{I_2 \cdot Fz_3 \cdot SAI}^{SC22} + \overbrace{I_3 \cdot Fz_3 \cdot INQ}^{SC23} + \overbrace{I_3 \cdot Fz_3 \cdot IRM4}^{SC24} + \overbrace{I_3 \cdot Fz_3 \cdot IRM8}^{SC26} + \overbrace{I_3 \cdot Fz_4 \cdot CBO}^{SC26} +$$
$$+ \overbrace{I_3 \cdot Fz_5 \cdot BINCA}^{SC27} + \overbrace{I_5 \cdot Fz_1 \cdot GMO \cdot B4 \cdot B5 \cdot \overline{B6}}^{BINR} +$$
$$+ \underbrace{I_5 \cdot Fz_1 \cdot GMO \cdot \overline{B4} \cdot \overline{B5} \cdot B6}_{SC45} + \underbrace{I_5 \cdot Fz_1 \cdot IIE \cdot \overline{B9} \cdot \overline{B10} \cdot B11 \cdot B18}_{SC48} +$$
$$+ \underbrace{I_5 \cdot Fz_1 \cdot IIE \cdot \overline{B12} \cdot B18}_{SC49} + \overbrace{I_5 \cdot Fz_4 \cdot MOZ}^{SC30} +$$
$$+ \underbrace{I_6 \cdot Fz_3 \cdot CBO}_{SC52} + \underbrace{I_6 \cdot Fz_3 \cdot SIA}_{SC53}$$

Ec. 84

$$PORP(1\div6) = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6}\cdot\overline{B7}\cdot B8\cdot B18}^{PORP} \cdot CS(1\div6)$$

Ac. ecuație se defalcă în următoarele:

a. $PORP1 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6}\cdot\overline{B7}\cdot B8\cdot B18}^{PORP} \cdot CS1$

b. $PORP2 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6}\cdot\overline{B7}\cdot B8\cdot B18}^{PORP} \cdot CS2$

c. $PORP3 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6}\cdot\overline{B7}\cdot B8\cdot B18}^{PORP} \cdot CS3$

d. $PORP4 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6}\cdot\overline{B7}\cdot B8\cdot B18}^{PORP} \cdot CS4$

e. $PORP5 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6}\cdot\overline{B7}\cdot B8\cdot B18}^{PORP} \cdot CS5$

f. $PORP6 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6}\cdot\overline{B7}\cdot B8\cdot B18}^{PORP} \cdot CS6$

$CS1 = B0\cdot\overline{B1}\cdot\overline{B2}$ $\quad CS2 = \overline{B0}\cdot B1\cdot\overline{B2}$ $\quad CS3 = B0\cdot B1\cdot\overline{B2}$

$CS4 = \overline{B0}\cdot\overline{B1}\cdot B2$ $\quad CS5 = B0\cdot\overline{B1}\cdot B2$ $\quad CS6 = \overline{B0}\cdot B1\cdot B2$

Ec 85

$$Rot.DrA = I_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B14\cdot B18}^{RDA} + I_6 \cdot Fz_1 \cdot GDR \cdot \overbrace{B4\cdot B18}^{RDA'}$$

Ec. 86

$$Rot.DrQ = I_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B14\cdot\overline{B18}}^{RDQ} + I_6 \cdot Fz_1 \cdot GDR \cdot \overbrace{B4\cdot\overline{B18}}^{RDQ}$$

Ec. 87

$$Rot.StA = I_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B13\cdot B18}^{RSA2} + I_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B15\cdot B18}^{RSA} + I_5 \cdot Fz_1 \cdot GDR \cdot B13\cdot B18 + I_6 \cdot Fz_1 \cdot GDR \cdot \overbrace{B3\cdot B18}^{RSA2} + I_6 \cdot Fz_1 \cdot GDR \cdot \overbrace{B5\cdot B18}^{RSA} + I_7 \cdot Fz_1 \cdot GDR \cdot \overbrace{B3\cdot B18}^{RSA2}$$

Ec. 88

$$Rot\ StQ = I_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B13\cdot\overline{B18}}^{RSQ2} + I_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B15\cdot\overline{B18}}^{RSQ} + I_5 \cdot Fz_1 \cdot GDR \cdot \overbrace{B13\cdot\overline{B18}}^{RSQ2} + I_6 \cdot Fz_1 \cdot GDR \cdot \overbrace{B3\cdot B18}^{RSQ2} + I_6 \cdot Fz_1 \cdot GDR \cdot \overbrace{B5\cdot\overline{B18}}^{RSQ} + I_7 \cdot Fz_1 \cdot GDR \cdot \overbrace{B3\cdot\overline{B18}}^{RSQ2}$$

Ec. 89

$$SCR = i_0 \cdot Fz_3 \cdot IRM10 + i_0 \cdot Fz_3 \cdot SSP + \overbrace{i_0 \cdot Fz_4 \cdot MOZ}^{SC8} + \overbrace{i_0 \cdot Fz_5 \cdot BMEM}^{SC9} + \overbrace{i_0 \cdot Fz_6}^{SC11} + \overbrace{i_0 \cdot Fz_5 \cdot BMF \cdot \overline{BPRI}}^{SC74}$$

Ec. 90

$$+SnR = \overbrace{i_0 \cdot Fz_4 \cdot CBO}^{SC6} + \overbrace{i_4 \cdot Fz_1 \cdot GMO \cdot B17 \cdot B18}^{ADSnR} + \overbrace{i_4 \cdot Fz_3 \cdot CBO}^{SC30} + \overbrace{i_4 \cdot Fz_3 \cdot SAI}^{SC35} + \overbrace{i_4 \cdot Fz_3 \cdot SAE}^{SC36} + \overbrace{i_4 \cdot Fz_3 \cdot SIA}^{SC38} +$$
$$+ \overbrace{i_5 \cdot Fz_4 \cdot CBO}^{SC43} +$$
$$+ \overbrace{i_6 \cdot Fz_4 \cdot MOZ}^{SC51} +$$
$$+ i_7 \cdot Fz_3 \cdot INM + \overbrace{i_7 \cdot Fz_3 \cdot SIA}^{SC62}$$

Ec. 91

$$StA = \overbrace{i_0 \cdot Fz_4 \cdot IMP}^{SC7} \cdot$$
$$+ i_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B13 \cdot B18}^{RSA2} + i_4 i_1 GDR \cdot \overbrace{B15 \cdot B18}^{RSA} + i_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B17 \cdot B18}^{DSA} +$$
$$+ i_5 \cdot Fz_1 \cdot GDR \cdot \overbrace{B13 \cdot B18}^{RSA2} +$$
$$+ i_6 \cdot Fz_1 \cdot GDR \cdot \overbrace{B3 \cdot B18}^{RSA2} + i_6 \cdot Fz_1 \cdot GDR \cdot \overbrace{B5 \cdot B18}^{RSA} + i_6 \cdot Fz_1 \cdot GDR \cdot \overbrace{B7 \cdot B18}^{DSA} +$$
$$+ i_7 \cdot Fz_1 \cdot GDR \cdot \overbrace{B3 \cdot B18}^{RSA2}$$

Ec. 92

$$StQ = \overbrace{i_0 \cdot Fz_4 \cdot IMP}^{SC7} +$$
$$+ i_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B13 \cdot B18}^{RSQ2} + i_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B15 \cdot B18}^{RSQ} + i_4 \cdot Fz_1 \cdot GDR \cdot \overbrace{B17 \cdot B18}^{DSQ} +$$
$$+ i_5 \cdot Fz_1 \cdot GDR \cdot \overbrace{B13 \cdot B18}^{RSQ2} +$$
$$+ i_6 \cdot Fz_1 \cdot GDR \cdot \overbrace{B3 \cdot B18}^{RSQ2} + i_6 \cdot Fz_1 \cdot GDR \cdot \overbrace{B5 \cdot B18}^{RSQ} + i_6 \cdot Fz_1 \cdot GDR \cdot \overbrace{B7 \cdot B18}^{DSQ} +$$
$$+ i_7 \cdot Fz_1 \cdot GDR \cdot \overbrace{B3 \cdot B18}^{RSQ2}$$

Ec. 93

$$TrBf = Io \cdot Fz_4 \cdot CBO +$$
$$+ I_4 \cdot Fz_3 \cdot SAE + I_4 \cdot Fz_3 \cdot SIA +$$
$$+ I_7 \cdot Fz_3 \cdot SIA$$

Ec. 94

$$t_1(1\div6) = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot B8 \cdot B18}^{EXT} \cdot CS\,(1\div6)$$

Ac. ecuație se defalcă în următoarele:

$$a\,(t_1)2 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot B8 \cdot B18}^{EXT} \cdot CS2$$

$$b\,(t_1)4 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot B8 \cdot B18}^{EXT} \cdot CS4$$

$$c\,(t_1)6 = i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot B8 \cdot B18}^{EXT} \cdot CS6$$

$CS2 = \overline{B_0} \cdot B_1 \cdot \overline{B_2}$ $\quad CS4 = \overline{B0} \cdot \overline{B_1} \cdot B2$ $\quad CS6 = \overline{B0} \cdot B1 \cdot B2$

Ec. 95

$$t_2(1\div6) = i_5 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot B8 \cdot B18}^{EXT} \cdot CS\,(1\div6)$$

Ac. ecuație se defalcă în următoarele:

$$a.(t_2)2 = i_5 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot B8 \cdot B18}^{EXT} \cdot CS2$$

$$b(t_2)4 = i_5 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot B8 \cdot B18}^{EXT} \cdot CS4$$

$$c(t_2)6 = i_5 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot \overline{B7} \cdot B8 \cdot B18}^{EXT} \cdot CS6$$

$CS2 = \overline{B_0} \cdot B_1 \cdot \overline{B_2}$ $\quad CS4 = \overline{B_0} \cdot \overline{B_1} \cdot B_2$ $\quad CS6 = \overline{B_0} \cdot B_1 \cdot B_2$

Ec. 96

$$V = i_H \cdot BPO$$

Ec. 97

$$\dot{1}1A = i_5 \cdot Fz_1 \cdot GMO \cdot \overbrace{\overline{B2}}^{MUA} + i'_5 \cdot Fz_3 \cdot MOZ$$

Ec. 98

$$\begin{aligned}
+1NA = {} & i_0 \cdot Fz_4 \cdot MOZ \cdot ZA + \\
& + i_2 \cdot Fz_4 \cdot CBO \cdot \overline{ZA} + \\
& + i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot B7 \cdot \overline{B8} \cdot B18}^{OFA} \cdot CS1 \cdot FANA1 + i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot B7 \cdot \overline{B8} \cdot B18}^{OFA} \cdot CS2 \cdot FANA2 + \\
& + i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot B7 \cdot \overline{B8} \cdot B18}^{OFA} \cdot CS3 \cdot FANA3 + i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot B7 \cdot \overline{B8} \cdot B18}^{OFA} \cdot CS4 \cdot FANA4 + \\
& + i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot B7 \cdot B8 \cdot B18}^{OFA} \cdot CS5 \cdot FANA5 + i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot B7 \cdot B8 \cdot B18}^{OFA} \cdot CS6 \cdot FANA6 + \\
& + i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6} \cdot B7 \cdot B8 \cdot B18}^{OFB} \cdot CS1 \cdot FANB1 + i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6} \cdot B7 \cdot B8 \cdot B18}^{OFB} \cdot CS2 \cdot FANB2 +
\end{aligned}$$

$$+\ i_4 \cdot Fz_1 \cdot IIF \cdot \overbrace{\overline{B6}\ B7\ B8\ B18}^{OFB} \cdot CS3 \cdot FANB3 + i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6}\ B7 \cdot B8 \cdot B18}^{OFB} \cdot CS4 \cdot FANB4 +$$

$$+\ i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B6} \cdot B7 \cdot B8 \cdot B18}^{OFB} \cdot CS5 \cdot FANB5 + i_4 Fz_1 \cdot IIE \cdot \overbrace{B6 \cdot B7 \cdot B8 \cdot B18}^{OFA} \cdot CS6 \cdot FANB6 +$$

$$+\ i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{\overline{B9} \cdot B10 \cdot B11 \cdot B18}^{OTS} \cdot FANT5 + i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B13 \cdot B18}^{ODCU} \cdot DCR +$$

$$+\ i_4 \cdot Fz_1 \cdot IIE \cdot \overbrace{B11 \cdot B18}^{ODCZ} \cdot \overline{DCR} +$$

$$+\ i_5 \cdot Fz_1 \cdot GDR \cdot \overbrace{B9 \cdot B18 \cdot \overline{OA}}^{OAP} + i_5 \cdot Fz_1 \cdot GDR \cdot \overbrace{B9 \cdot \overline{B18} \cdot \overline{OQ}}^{OQP} +$$

$$+\ i_6 \cdot Fz_1 \cdot IRM1 + i_6 \cdot Fz_1 \cdot MIC + i_6 \cdot Fz_3 \cdot SSP + i_8 \cdot Fz_5 \cdot BMEM + i_6 \cdot Fz_5 \cdot BMF \cdot \overline{BPRI} +$$

$$+\ i_6 \cdot Fz_5 \cdot BVIZ + i_6 \cdot Fz_6 +$$

$$+\ i_4 \cdot Fz_1 \cdot GMO \cdot BOM +$$

$CS1 = \overline{B_0} \cdot \overline{B_1} \cdot \overline{B_2}$ $\quad CS2 = \overline{B_0} \cdot B_1 \cdot \overline{B_2}$ $\quad CS3 = B_0 \cdot B_1 \cdot \overline{B_2}$

$CS4 = \overline{B_0} \cdot \overline{B_1} \cdot B_2$ $\quad CS5 = B_0 \cdot \overline{B_1} \cdot B_2$ $\quad CS6 = \overline{B_0} \cdot B_1 \cdot B_2$

Ec 99a $\quad ZA = \overline{0A} \cdot \overline{1A} \cdot \ldots \cdot \overline{22A} \cdot \overline{23A}$

$$+1N24 = \overbrace{i_0 \cdot Fz_4 \cdot IMP}^{SC7} + \overbrace{i_4 \cdot Fz_3 \cdot INM}^{SC33} +$$

Ec. 99b

$$+1N28 = i_3 \cdot Fz5 \cdot BPRI + i_3 \cdot Fz5 \cdot BMF \cdot \overline{BPRI}$$

Ec. 100

$$+1N8 = \overbrace{i_H \cdot \overline{B11} \cdot BPO}^{ADN8} + g \cdot N8M \quad \text{unde} \quad i_H = h \cdot NORM + g \cdot UNP \ ^{x)}$$

EC. 101

$$+1Rg\ CA = i_4 \cdot Fz_1 \cdot GMO \cdot \overbrace{B13 \cdot B14}^{RTP}$$

Ec. 102

$$+1SnA = \overbrace{i_1 \cdot Fz_4 \cdot IMP \cdot Sn2A}^{SC14} +$$
$$+ \underbrace{i_2 \cdot Fz_4 \cdot MOZ \cdot SnR}_{SC21} +$$
$$+ i_4 \cdot Fz_1 \cdot GMO \cdot \overbrace{B13 \cdot B14}^{AD1SnA} + \underbrace{i_4 \cdot Fz_3 \cdot ADA \cdot SnR}_{SC29} + \underbrace{i_4 \cdot Fz_3 \cdot IMP}_{SC31} +$$
$$+ \overbrace{i_4 \cdot Fz_3 \cdot INA \cdot SnR}^{SC32} + \overbrace{i_4 \cdot Fz_3 \cdot MOZ \cdot SnR}^{SC34} + \underbrace{i_4 \cdot Fz_3 \cdot SCA \cdot SnR}_{SC37} +$$
$$+ \overbrace{i_4 \cdot Fz_5 \cdot BINCA \cdot SnR}^{SC40}$$

x) - Semnalele N8M, NORM și UNP se realizează manual cu ajutorul unui comutator cu 3 poziții (vezi sinteza schemelor BCC)
- g este impulsul la ieșirea GS cînd se apasă pe butonul lui.

TABELUL II

Ecuațiile logice ce privesc mnemonicele instrucțiilor

$$BCH = BINAD + BINCA + BMEM + BVIZ$$

$$IRM1 = ADA + CBO + IDA + IMP + INA + INM + INR + SAI + SAE + SCA + SIA$$

$$IRM2 = ADA + CBO + IDA + INA + INR + SAI + SAE + SIA$$

$$IRM3 = ADA + CBO + IMP + INM + MOZ + SAI + SAE + SCA + SSP + SIA$$

$$IRM4 = ADA + CBO + IMP + INM + MOZ + SAI + SAE + SCA + SIA$$

$$IRM5 = ADA + INM + SAI + SAE + SCA + SSP + SIA$$

$$IRM6 = CBO + IMP + MOZ$$

$$IRM7 = IDA + INA + INB + INQ + INR$$

$$IRM8 = IDA + INA + INR$$

$$IRM9 = MCQ + MEA + MEQ + MG2 + MPQ$$

$$IRM10 = MCQ + MEA + MEQ + MG2 + MPQ + MRA$$

$$IRM11 = IRM3 + IRM7 + IRM10$$

$$IRM12 = IRM3 + IRM7 + IRM10 + SLT$$

$$MIC = GDR + GMO + IIE$$

TABELLA III

f_{Z1}	f_{Z2}	f_{Z3}	f_{Z4}	f_{Z5}	f_{Z6}
GNO (4)	IRM12 (24)	NOZ (3)	MOZ (26)	BINAD (20)	4 (24)
1 (24)	IRM12 (26)	MRA (19)	CBO (61)	BMEM (24)	1 (25)
1 (26)	IRM12 (48)	IRM11 (24)	MOZ (64)	BVIZ (24)	1 (89)
4 (48)	$i_0 \cdot f_{Z2} \cdot$ IRM12 (24,26,48)	IRM11 (26)	MOZ (89)	BMEM (26)	$i_0 \cdot f_{Z5}$ (24,26,89)
$i_0 \cdot f_{Z1}$ (24,26,48)		MRA (27)	CBO (90)	BVIZ (26)	
		IRM4 (48)	IMP (91)	BVIZ (48)	
		IRM7 (48)	IMP (92)	BMEM (89)	
		IRM10 (89)	IMP (99)	$i_0 \cdot f_{Z5} \cdot$ BMEM (24,26,89)	
		SSP (89)	$i_0 \cdot f_{Z4}$ CBO (61,90)	$i_0 \cdot f_{Z5} \cdot$ BVIZ (24,26,48)	
		$i_0 \cdot f_{Z3} \cdot$ IRM11 (24,26)			
		$i_0 \cdot f_{Z3} \cdot$ MRA (19 27)	$i_0 \cdot f_{Z4} \cdot$ IMP (91,92,99)		
			$i_0 \cdot f_{Z4} \cdot$ MOZ (26,64,89)		

i_0 (17)

TABELUL IV

Nr. crt.	Semnal comun (SC)	Ecuația în care apare	Încărcare	Observații
[1]	[2]	[3]	[4]	[5]
1	$i \cdot BPOR$	14, 39	2	
2	$i_0 \cdot Fz1$	24, 26, 48	3	
3	$i_0 \cdot Fz2 \cdot IRM12$	24, 26, 48	3	
4	$i_0 \cdot Fz3 \cdot IRM11$	24, 26	2	
5	$i_0 \cdot Fz3 \cdot MRA$	19, 27	2	
6	$i_0 \cdot Fz4 \cdot CBO$	61, 90	2	
7	$i_0 \cdot Fz4 \cdot IMP$	91, 92, 99	3	
8	$i_0 \cdot Fz4 \cdot MOZ$	26, 64, 89	3	
9	$i_0 \cdot Fz5 \cdot BMEM$	24, 26, 89	3	
10	$i_0 \cdot Fz5 \cdot BVIZ$	24, 26, 48	3	
11	$i_0 \cdot Fz6 \cdot$	24, 26 89	3	
12	$i_0 \cdot Fz4 \cdot MOZ$	16, 83	2	
13	$i_1 \cdot Fz3 \cdot INQ$	23, 27	2	
14	$i_1 \cdot Fz4 \cdot IMP \cdot \overline{Sn2A}$	62, 102	2	
15	$i_1 \cdot Fz5 \cdot BINCA$	1, 19, 27	3	
16	$I_1 \cdot Fz3 \cdot MRA$	76, 83	2	
17	$i_2 \cdot Fz3 \cdot MCQ$	69, 73	2	
18	$i_2 \cdot Fz3 \cdot MOZ$	1, 19, 27	3	
19	$i_2 \cdot Fz3 \cdot MRA \cdot SnR$	62	2	modificat
20	$i_2 \cdot Fz4 \cdot CBO$	19, 27	2	
21	$i_2 \cdot Fz4 \cdot MOZ \cdot SnR$	62, 102	2	
22	$I_2 \cdot Fz3 \cdot SAI$	76, 83	2	
23	$I_3 \cdot Fz3 \cdot INQ$	78, 83	2	
24	$I_3 \cdot Fz3 \cdot IRM4$	76, 83	2	

[1]	[2]	[3]	[4]	[5]
25	$I_3 \cdot Fz_3 \cdot IRM8$	76, 83	2	
26	$I_3 \cdot Fz_4 \cdot CBO$	76, 83	2	
27	$I_3 \cdot Fz_5 \cdot BINCA$	76, 83	2	
28	$i_4 \cdot Fz_4 \cdot IIE \cdot B12 \cdot B18$	19, 27	2	
29	$i_4 \cdot Fz_3 \cdot AIA \cdot SnR$	62, 102	2	
30	$i_4 \cdot Fz_3 \cdot CBO$	61, 90	2	
31	$i_4 \cdot Fz_3 \cdot IMP$	62, 102	2	
32	$i_4 \cdot Fz_3 \cdot INA \cdot SnR$	62, 102	2	
33	$i_4 \cdot Fz_3 \cdot INM$	51, 98	2	
34	$i_4 \cdot Fz_3 \cdot MOZ \cdot SnR$	62, 102	2	
34'	$i_4 \cdot Fz_3 \cdot MRA$	69a, 73	2	
35	$i_4 \cdot Fz_3 \cdot SAI$	61, 90	2	
36	$i_4 \cdot Fz_3 \cdot SAE$	61, 90	2	
37	$i_4 \cdot Fz_3 \cdot SCA \cdot SnR$	62, 102	2	
38	$i_4 \cdot Fz_3 \cdot SIA$	61, 90	2	
39	$i_4 \cdot Fz_4 \cdot MOZ$	19, 27, 64	3	
40	$i_4 \cdot Fz_5 \cdot BINCA \cdot SnR$	62, 102	2	
40'	$I_4 \cdot Fz_1 \cdot GDR \cdot B14 \cdot \overline{B18}$	52b, 86	2	
41	$i_5 \cdot Fz_3 \cdot CBO$	19, 27	2	
41'	$i_5 \cdot Fz_3 \cdot IMP$	17, 21, 23	3	
42	$i_5 \cdot Fz_3 \cdot SIA$	19, 27	2	
43	$i_5 \cdot Fz_4 \cdot CBO$	61, 90	2	
44	$i_5 \cdot Fz_4 \cdot MOZ$	1, 66	2	
45	$I_5 \cdot Fz_1 \cdot GMO \cdot B4 \cdot B5 \cdot \overline{B6}$	76, 83	2	
46	$I_5 \cdot Fz_1 \cdot GMO \cdot B4 \cdot \overline{B5} \cdot B6$	76, 83	2	
47	$I_5 \cdot Fz_1 \cdot GMO \cdot \overline{B4} \cdot \overline{B5} \cdot B6$	76, 83	2	

[1]	[2]	[3]	[4]	[5]
48	$I_5 \cdot Fz1 \cdot IIE \cdot \overline{B9} \cdot \overline{B10} \cdot B11 \overline{B18}$	76, 83	2	
49	$I_5 \cdot Fz1 \cdot IIE \cdot B12 \cdot B18$	76, 83	2	
50	$I_5 \cdot Fz4 \cdot MOZ$	76, 83	2	
51	$i_6 \cdot Fz4 \cdot MOZ$	61, 90	2	
51'	$I_6 \cdot I_6 \cdot Fz1 \cdot GDR \cdot B4 \cdot \overline{B18}$	526, 86	2	
53	$I_6 \cdot Fz3 \cdot SIA$	76, 83	2	
54	$i_7 \cdot BDPRC$	11, 12, 13	3	
54'	$i_7 \cdot BUNCICL$	13, 15	2	
55	$i_7 \cdot Fz1 \cdot SLT \cdot BFANG \cdot BITR \cdot \overline{Id}$	5, 9, 58	3	
56	$i_7 \cdot Fz1 \cdot MIC \cdot BFANG \cdot BITR$	5, 9, 58	3	
57	$i_7 \cdot Fz2 \cdot SIT \cdot BFANG \cdot BITR \cdot \overline{Id}$	5, 9, 58	3	
58	$i_7 \cdot Fz3 \cdot IRM5 \cdot BFANG \cdot BITR$	5, 9, 58	3	
59	$i_7 \cdot Fz3 \cdot IRM7 \cdot BFANG \cdot BITR$	5, 9, 58	3	
60	$i_7 \cdot Fz3 \cdot IRM10 \cdot BFANG \cdot BITR$	5, 9, 58	3	
61	$i_7 \cdot Fz3 \cdot MOZ$	19, 27, 56	3	
62	$i_7 \cdot Fz3 \cdot SIA$	61, 90	2	
63	$i_7 \cdot Fz4 \cdot IRM6 \cdot BFANG \cdot BITR$	5, 9, 58	3	
64	$i_7 \cdot Fz5 \cdot BINAD$	7, 53	2	
65	$i_7 \cdot Fz5 \cdot BINCA$	8, 53	2	
66	$i_7 \cdot Fz5 \cdot BMEM$	10, 53	2	
67	$i_7 \cdot Fz5 \cdot BVIZ$	16, 53	2	
68	$\overline{iINAD}$	32, 57	2	
69	$\overline{iINCA}$	33, 67	2	
70	$\overline{iMEM}$	35, 57	2	
71	$i_7 \overline{SIINIT}$	5,6,7,8,9,10,11 13,14,15,16,476,53	13	-
72	$\overline{iUNCICL}$	1	2	modificat

[1]	[2]	[3]	[4]	[5]
73	$i \cdot VIZ$	42, 57	2	
74	$i_0 \cdot fzs \cdot BMF \cdot \overline{BPRI}$ *)	24, 26, 89	3	
75	$i_7 \cdot fzs \cdot BMF \cdot BPRI$	10b, 13, 53	3	
76	$i_7 \cdot fzs \cdot BMF \cdot \overline{BPRI}$	21b, 40b	2	

Avem s. ex. $SC\ 67 = i_7 \cdot fzs \cdot Bviz$

*) SC Nr. 72 ÷ 75 nu sînt înscrise în tabel în ordinea alfabetică deoarece au fost introduse mai tîrziu, după întocmirea tabelului

TABELUL V

Acest tabel conține variabilele care intră în variabilele compuse și încărcările circuitelor care produc variabilele.

Nr. crt.	Variabila	Variabila compusă din care face parte	Încărc. circ. care produce variabila.	Unde e formată var.		Observații
				Placa	Borna	
1.	$\overline{ADA}$	IRM1; IRM3; IRM5; IRM2; IRM4	5	16 J	a5	
2.	$\overline{BINAD}$	BCH;	1	2 J	—	
3.	$\overline{BINCA}$	BCH;	1	1 J	a4	
4.	$\overline{BMEM}$	BCH;	1	2 J	—	
5.	$\overline{BUNCICL}$	BCH;	1	2 J	—	
6.	$\overline{BVIZ}$	BCH;	1	1 J	a5	
7	$\overline{CBO}$	IRM1; IRM3; IRM6; IRM2; IRM4	5	19 J	a3	
8.	$\overline{GDR}$	MIC;	1	19 J	a2	
9.	$\overline{GMO}$	MIC;	1	18 J	—	
10	$\overline{IDA}$	IRM1; IRM7; IRM2; IRM8	4	15 J	a4	
11.	$\overline{IIE}$	MIC	1	18 J	—	
12	$\overline{IMP}$	IRM1; IRM4; IRM3; IRM6;	4	16 J	a2	
13	$\overline{INA}$	IRM1; IRM7; IRM2, IRM8	4	15 J	a5	
14.	$\overline{INB}$	IRM7	1	15 J	a8	
15.	$\overline{INM}$	IRM1; IRM4 IRM3; IRM5	4	16 J	a3	
16.	$\overline{INQ}$	IRM7	1	15 J	a6	
17.	$\overline{INR}$	IRM1; IRM7 IRM2; IRM8	4	15 J	a7	

Nr. crt.	Variabila	Variabila compusă din care fac parte	Incărc. circ. care prod. variabila.	Unde e formată variab.		Observații
				Placa	Borna	
18	$\overline{MCQ}$	IRM9; IRM10;	2	16J	a7	
19	$\overline{MEA}$	IRM9; IRM10;	2	15J	a1	
20	$\overline{MG1}$	IRM9, IRM10;	2	15J	a3	
21	$\overline{MG2}$	IRM9, IRM10;	2	15J	a2	
22	$\overline{MOZ}$	IRM3; IRM6; IRM4;	3	19J	a4	
23	$\overline{MPQ}$	IRM9 IRM10	2	16J	a8	
24	$\overline{MRA}$	IRM10	1	16J	a6	
25	$\overline{SAI}$	IRM1; IRM3, IRM5; IRM2; IRM4	5	16J	a1	
26	$\overline{SAE}$	IRM1; IRM3; IRM5; IRM2; IRM4	5	19J	a8	
27	$\overline{SCA}$	IRM1; IRM4 IRM3; IRM5	4	16J	64	
28	$\overline{SLT}$	IRM12;	1	19J	a6	
29	$\overline{SSP}$	IRM3; IRM6;	2	19J	a5	
30	$\overline{SIA}$	IRM1; IRM3, IRM5; IRM2; IRM4	5	19J	a7	

Obs. La coloana „Placa" se trece indicativul cu care se notează placa cu conector pe care se află borna la care ajunge variabila. In coloana „Borna" se arată indicativul cu care se notează borna conectorului la care ajunge variabila.

TABELUL VI

Acest tabel conține încărcările însumate acircuitelor care produc variabilele și care sînt cuprinse în tabelele III și V

Nr. crt.	Variabila	Încărcarea rezultală din tabelul III nr. 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	Încărc. rezultă-e din tab. III	Încărc. rezultă. din tab. V.	Suma încăr-cărilor	Unde e formată variabila: Placa	Borna
1.	ADA	–	–	–	–	–	–	–	–	–	–	–	–	2	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	4	–	4	16J	05
2.	$\overline{ADA}$	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	5	5	16J	25
3.	BCH	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	1	–	1	2J	52
195	OA	–	–	–	–	–	–	–	–	–	–	–	–	–	2	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	2	–	2	DA/26J	02
196	$\overline{OA}$	–	–	–	–	–	–	–	–	–	–	–	–	–	2	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	2	–	2	DA/26J	01
197	$\overline{OQ}$	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	1	–	1	DA/26S	02

Obs. Vezi obs. de la tab. V

TABELUL VII

Acest tabel conține încărcările circuitelor care produc cele 102 funcții din tabelul I

Nr. crt.	Funcția	Încărcarea	Unde e formată funcția.		Obs.
			Placa	Borna	
1	A(0)A	1	25S	a9	
2.	A(0)AdIdRI	19	8 J	b4	
3.	A(0)B	1	24S	a9	
4	A(0)BOM	1	11S	b8	

99.	+ 1N24	10	8S	b8	
100.	+ 1NS	6	7S	c4	
101.	+ 1Rg CA	1	27S	a9	
102	+ 1SmA	1	26S	a9	

Obs. Vezi obs de la tab. V.

CAPITOLUL 6

SINTEZA SCHEMELOR BCC

Se face întîi sinteza circuitelor care produc funcţiile din tabelul I, utilizîndu-se variabilele comune şi semnalele comune. La sfîrşit se vor face şi schemele care produc anumite variabile comune cum sînt OM sau ZA, în conformitate cu ecuaţiile arătate tot în tabelul I, la locurile unde ele apar în ecuaţii. La ieşiri se vor nota în cercuri încărcările luate din tabelul VII.

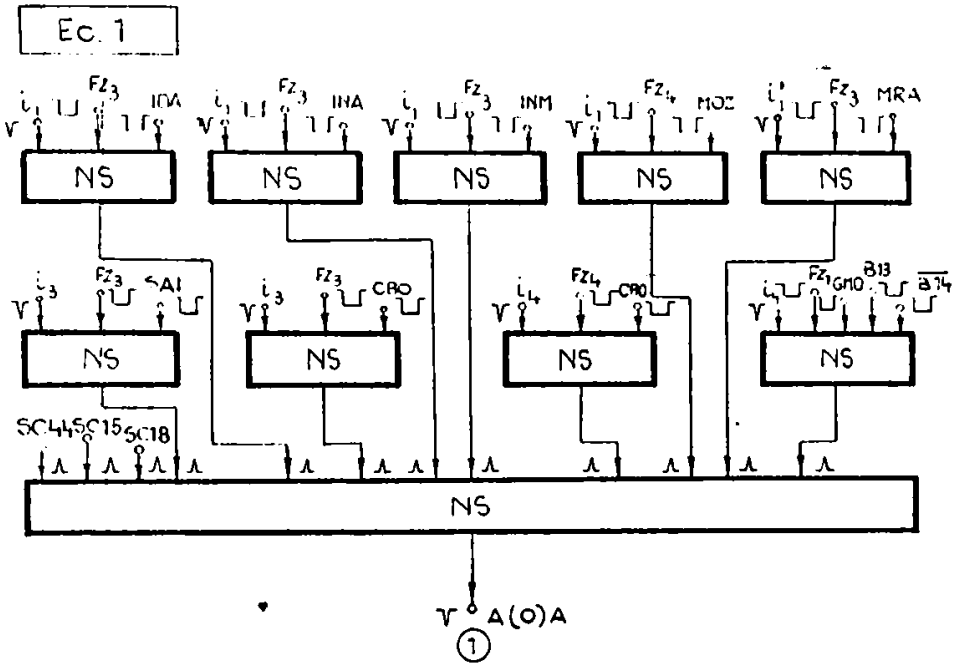
Se va face acum sinteza schemelor care produc variabilele comune, după tabelul II. La ieşirile circuitelor corespunzătoare se vor nota în cerc încărcările luate din tabelul VI.

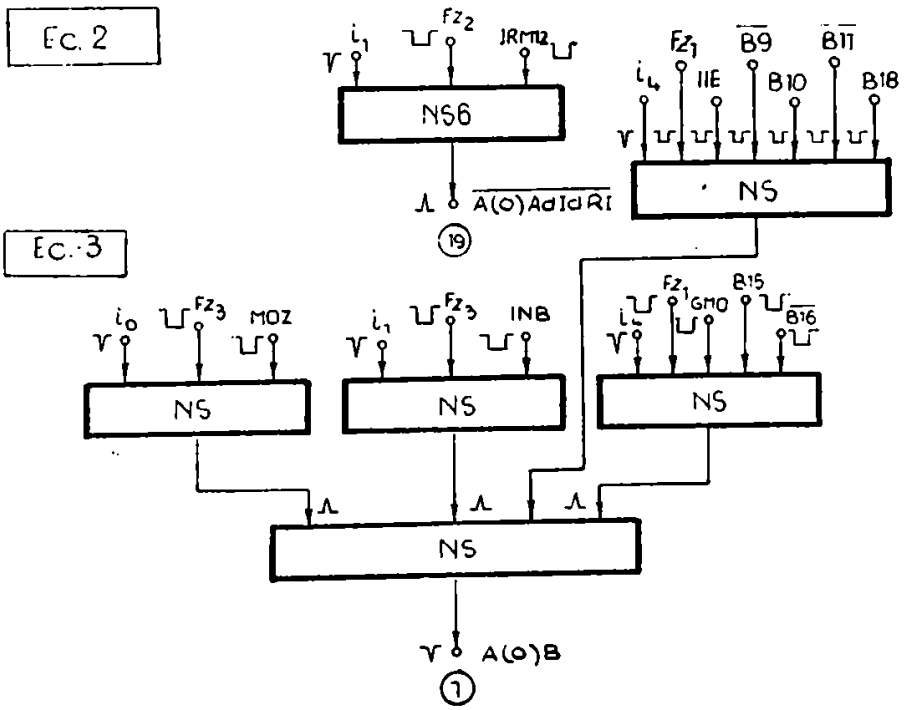
Se va face apoi sinteza circuitelor care produc semnalele comune pe baza tabelului IV şi se vor nota şi aici la ieşiri încărcările în cercuri luate din tab. IV.

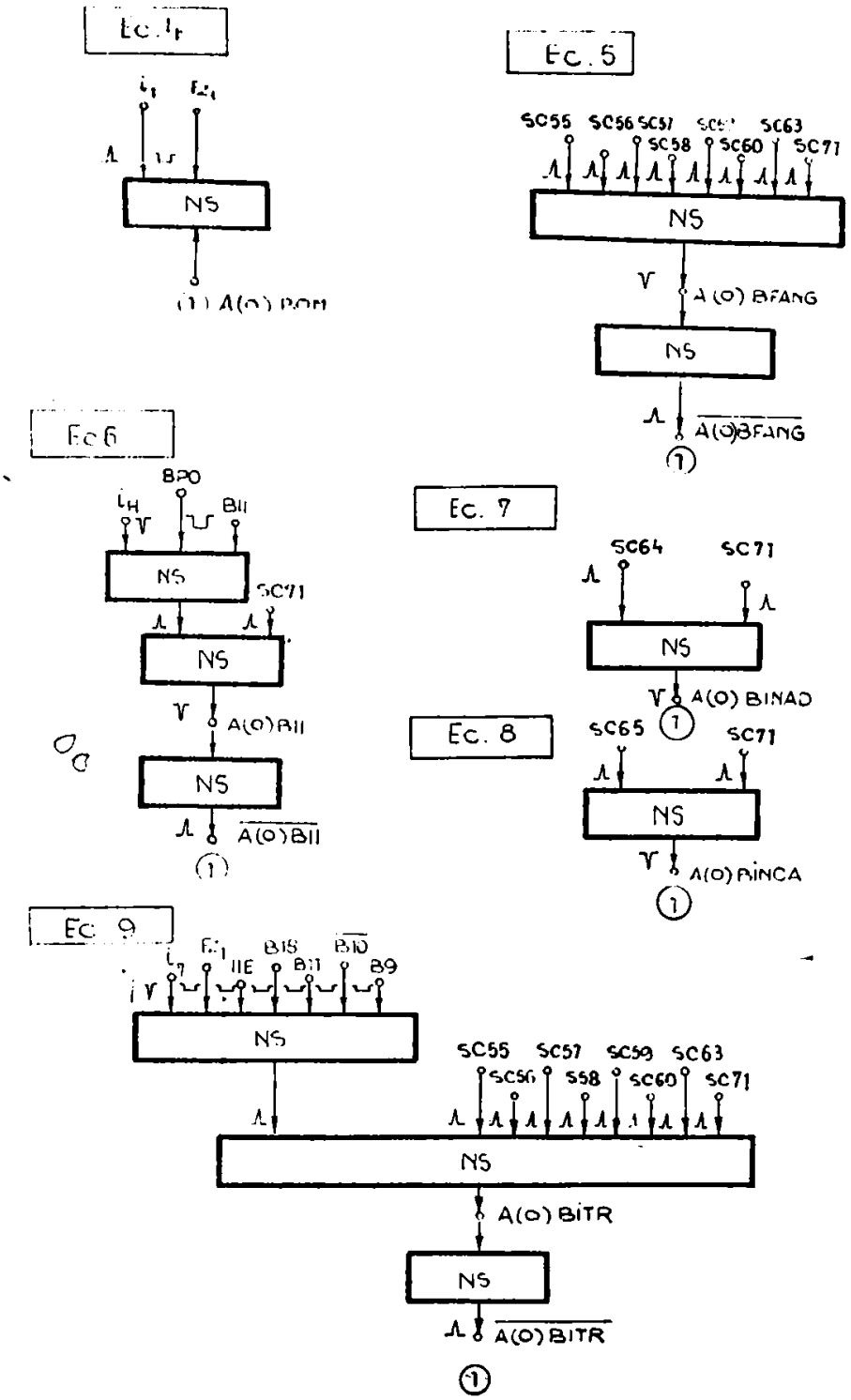
Avem deci realizată sinteza tuturor schemelor din care se compune BCC.

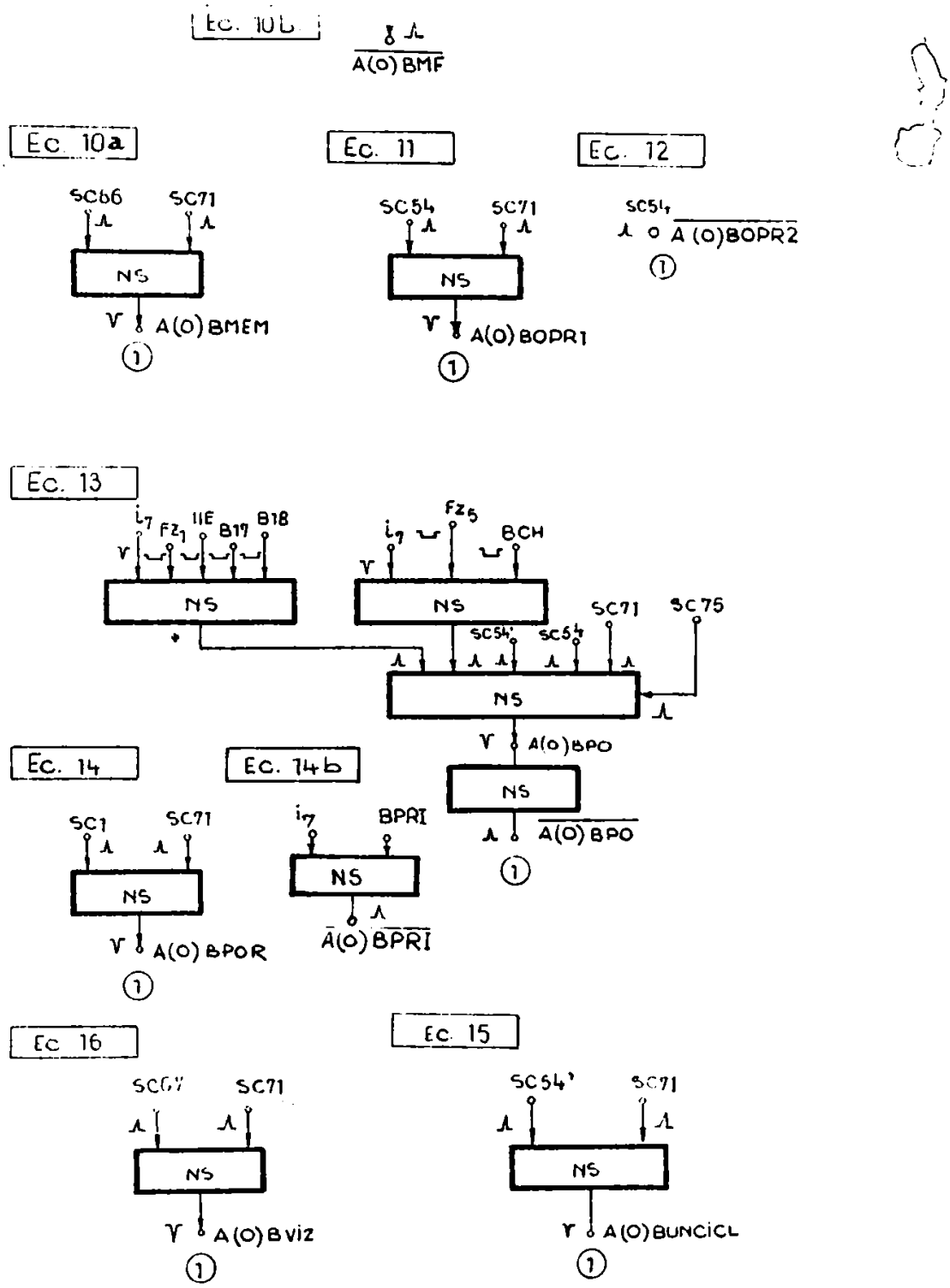
Ne rămîne acum numai elaborarea schemelor celorlalte blocuri înafară de BCC, din care este format calculatorul.

Sinteza schemelor BCC pe baza ecuoatiilor din tabelul I

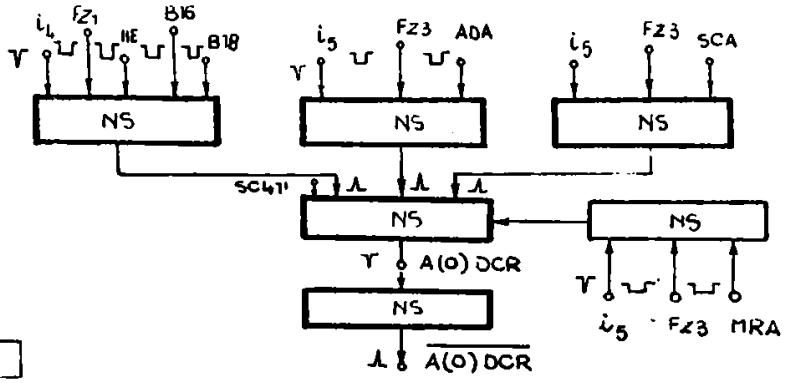




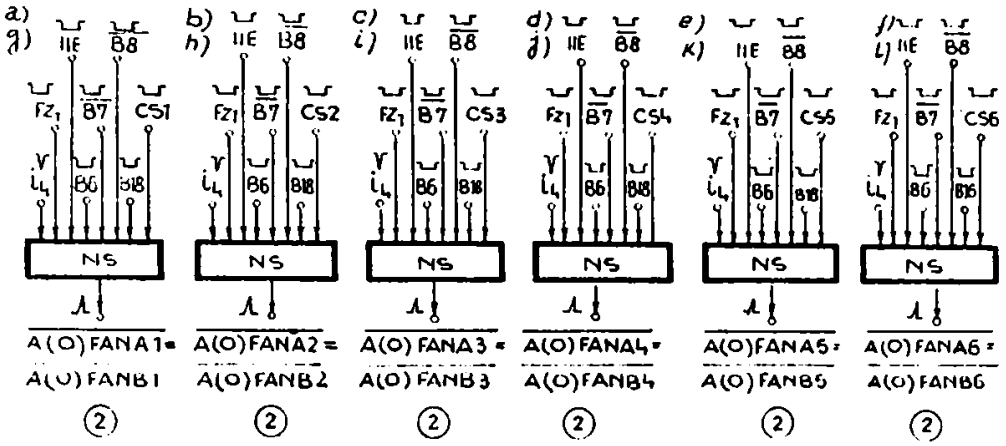

Ec. 4
NS
A(0) BOM
Ec. 5
SC55 SC56 SC57 SC58 SC60 SC63 SC71
NS
A(0) BFANG
NS
A(0)BFANG
Ec 6
BP0
BII
NS
SC71
NS
A(0)BII
NS
A(0)BII
Ec. 7
SC64
SC71
NS
A(0) BINAD
Ec. 8
SC65
SC71
NS
A(0) BINCA
Ec. 9
BIS
BII
B9
NS
SC55 SC57 SC59 SC63
SC56 S58 SC60 SC71
NS
A(0) BITR
NS
A(0) BITR


Ec. 10b
A(0) BMF
Ec. 10a
SC66
SC71
NS
A(0) BMEM
Ec. 11
SC54
SC71
NS
A(0) BOPR1
Ec. 12
SC54
A (0)BOPR2
Ec. 13
i7
FZ1
IIE
B19
B18
NS
FZ5
i7
BCH
NS
SC71
SC75
SC54'
SC54
NS
A(0)BPO
Ec. 14
Ec. 14b
SC1
SC71
NS
A(0) BPOR
i7
BPRI
NS
A(0) BPRI
NS
A(0) BPO
Ec 16
Ec. 15
SC67
SC71
NS
A(0) BVIZ
SC54'
SC71
NS
A(0) BUNCICL

Ec. 17

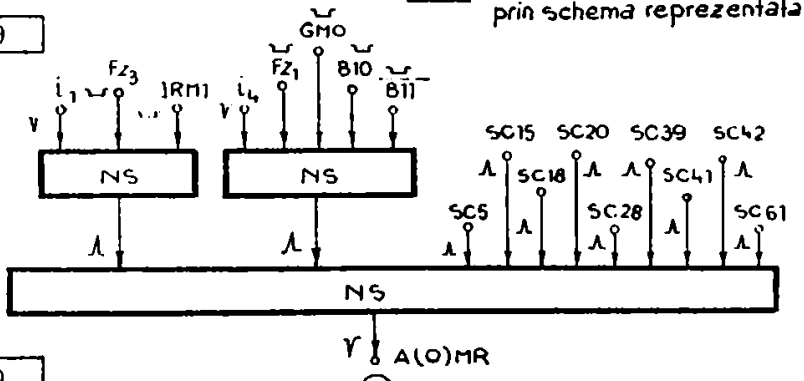


Ec. 18

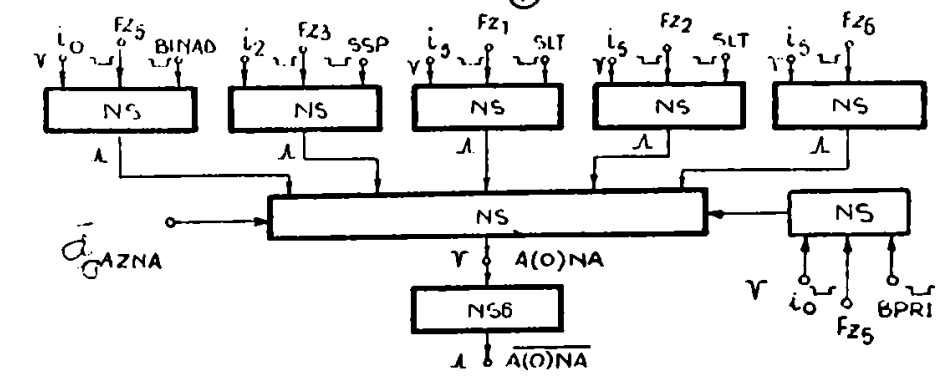


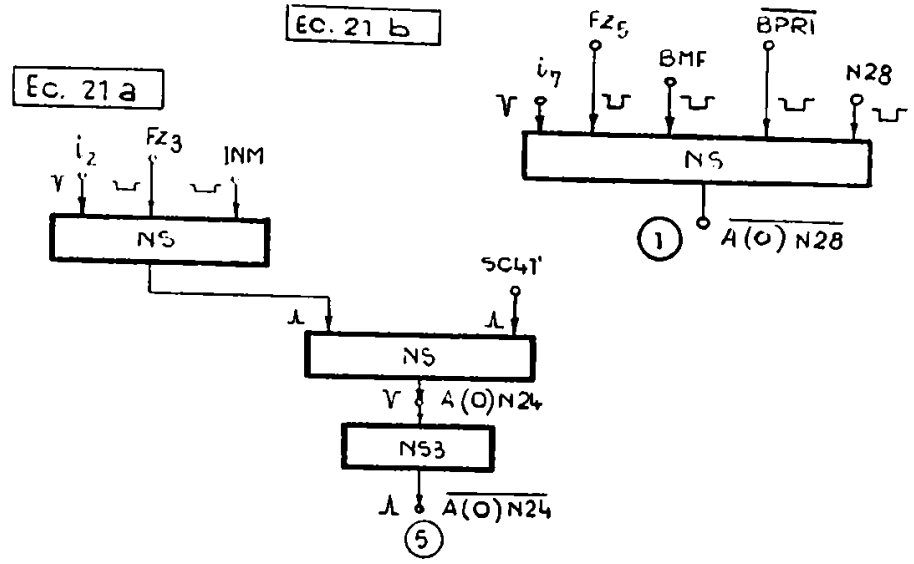
Obs: Semnalele CS1÷CS6 sînt realizate prin schema reprezentată la pag. 158

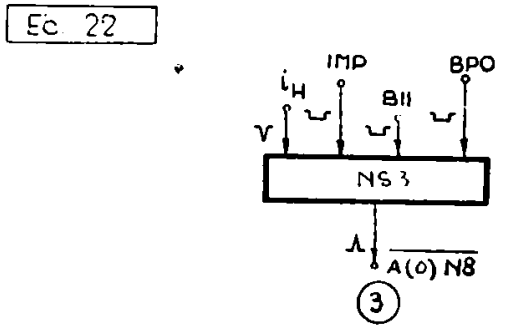
Ec. 19

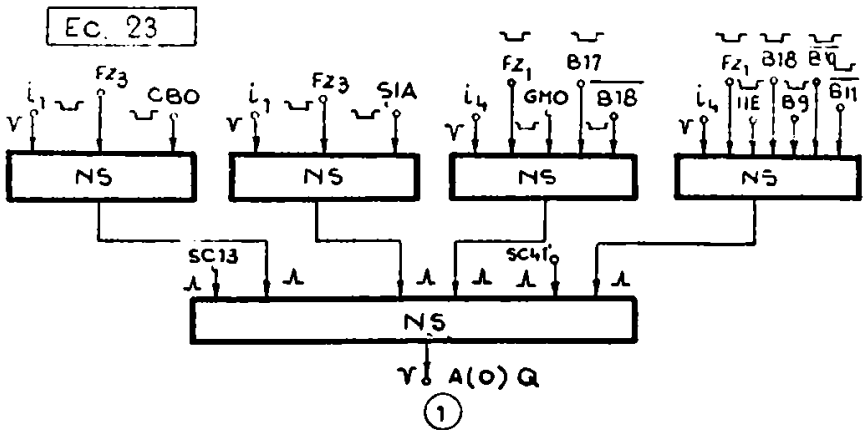


Ec. 20

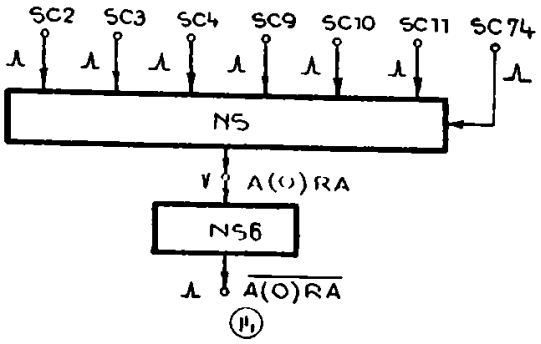



Ec. 21 b
Ec. 21 a
i_2
Fz_3
INM
NS
i_7
Fz_5
BMF
$\overline{BPRI}$
N28
NS
1
$\overline{A(0)\,N28}$
SC41'
NS
A(0)N24
NS3
$\overline{A(0)N24}$
5

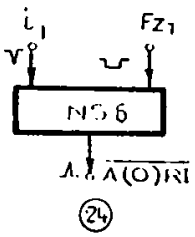

Ec 22
i_H
IMP
B11
BPO
NS 3
$\overline{A(0)\,N8}$
3


Ec. 23
i_1
Fz_3
CBO
NS
i_1
Fz_3
SIA
NS
i_4
Fz_1
GMO
B17
$\overline{B18}$
NS
i_4
Fz_1
IIE
B18
B9
$\overline{B11}$
NS
SC13
SC41'
NS
A(0) Q
1

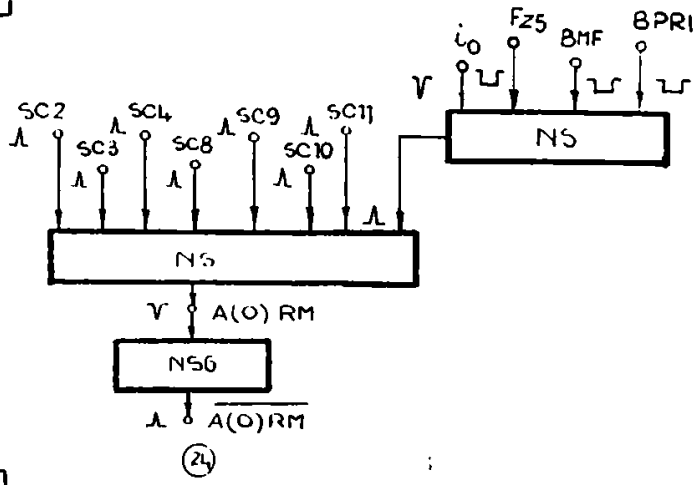
Ec. 24



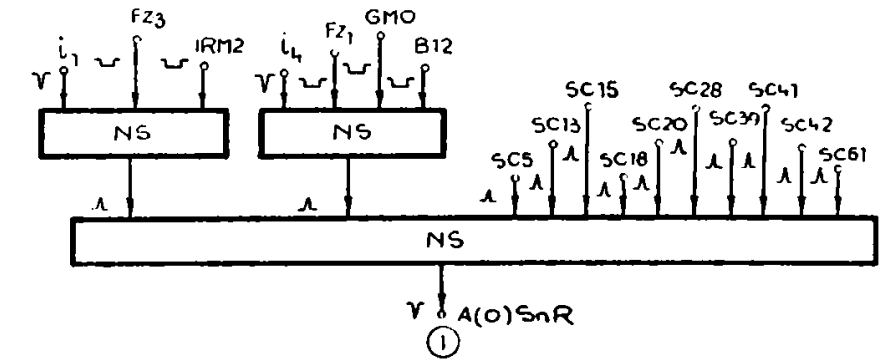
Ec. 25



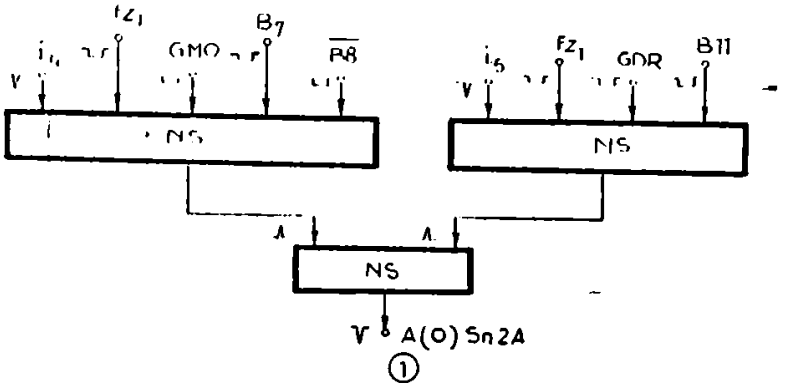
Ec. 26



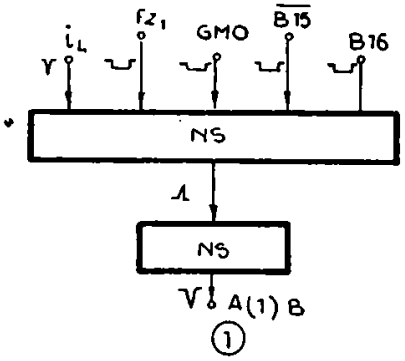
Ec. 27



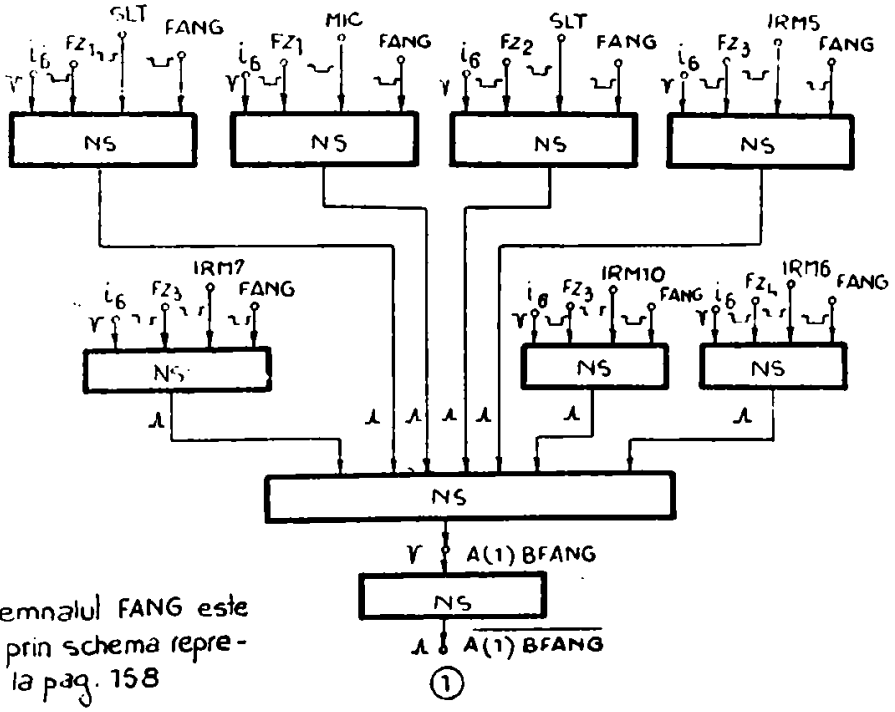
Ec 28



Ec 29

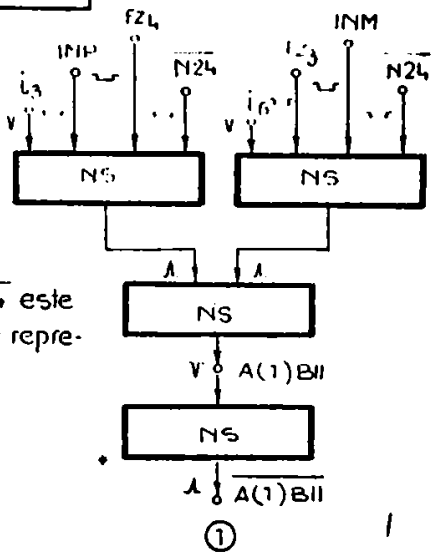


Ec 30



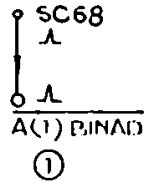
Obs. Semnalul FANG este realizat prin schema reprezentată la pag. 158

Ec 31

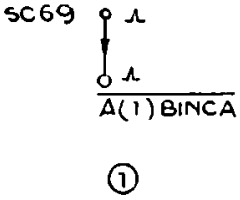


Obs. Semnalul $\overline{N24}$ este realizat prin schema reprezentată la pag. 159

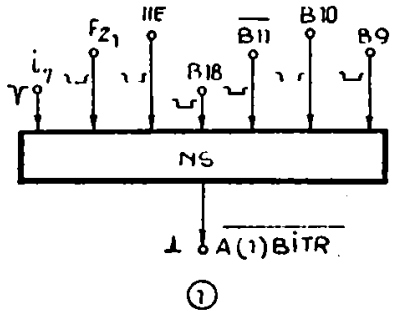
Ec. 32



Ec. 33

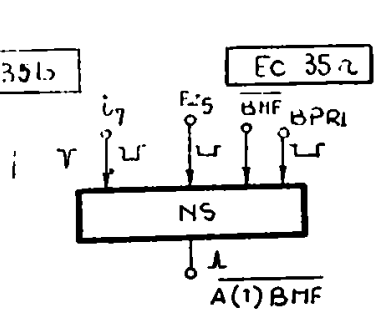


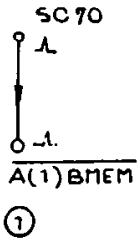
Ec 34

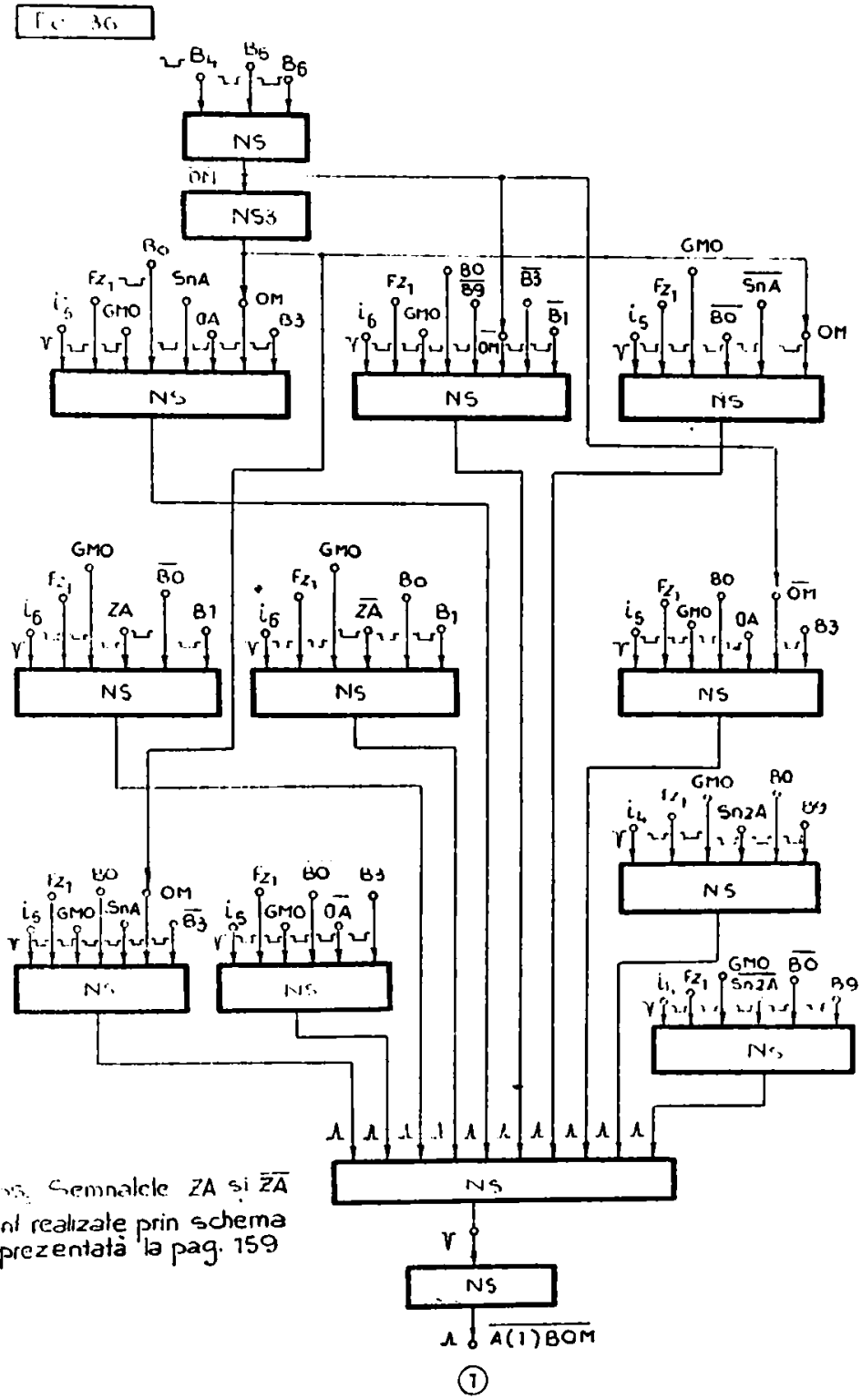


Ec. 35b

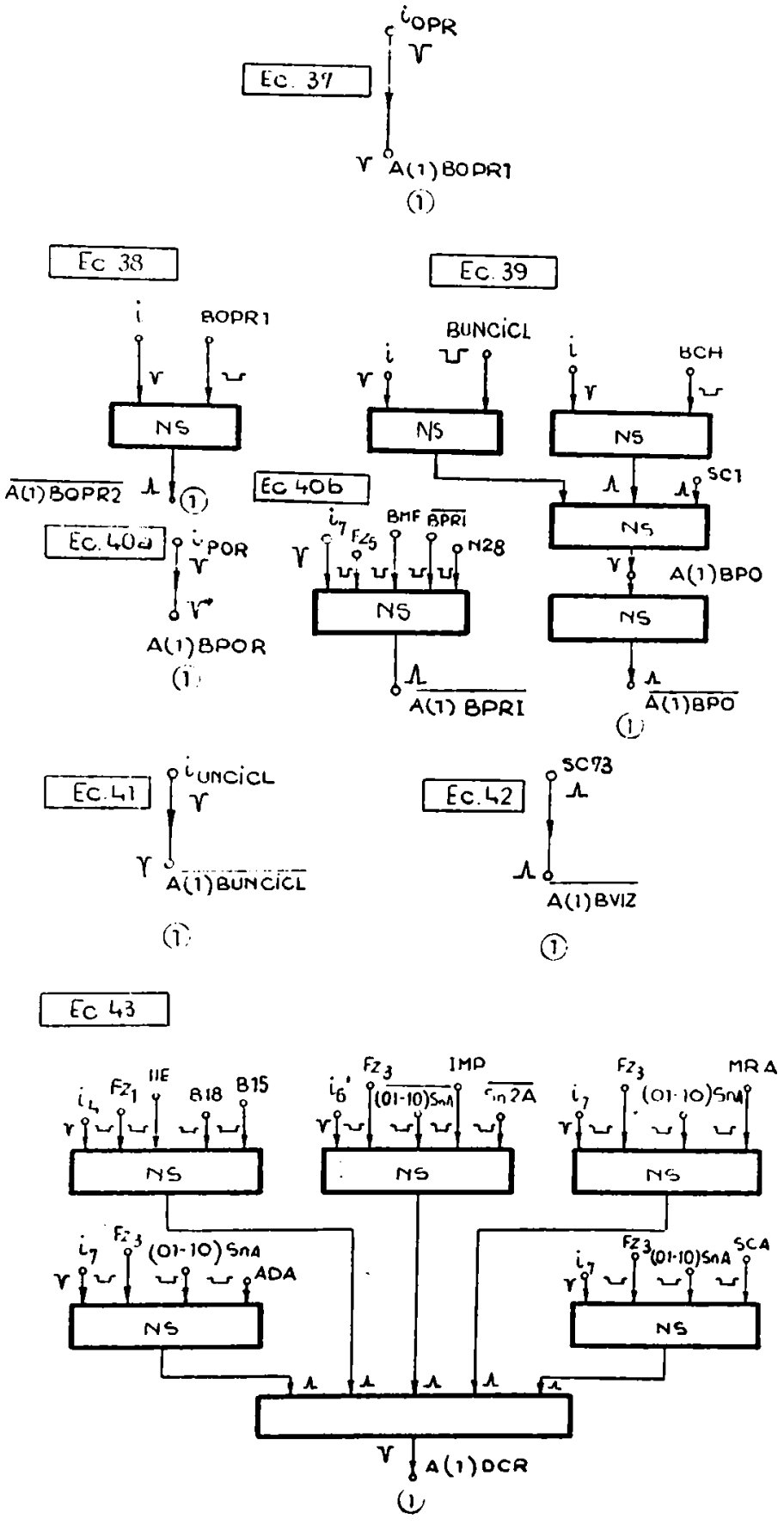
Ec. 35a



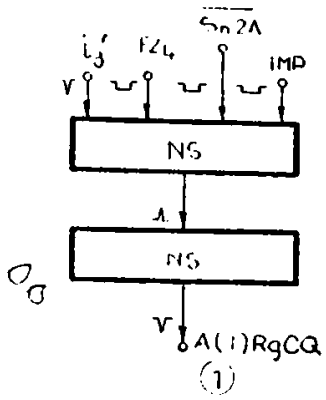




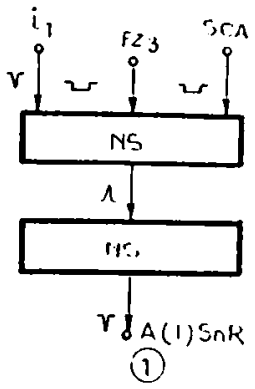
Obs. Semnalele ZA și $\overline{ZA}$ sînt realizate prin schema reprezentată la pag. 159


Ec. 37
iOPR
A(1)BOPR1
(1)
Ec 38
BOPR1
NS
A(1)BOPR2
(1)
Ec. 39
BUNCICL
BCH
NS
NS
SC1
Ec 40b
BMF
BPRI
FZ5
N28
NS
A(1)BPO
NS
A(1)BPRI
A(1)BPO
(1)
Ec. 40a
iPOR
A(1)BPOR
(1)
Ec. 41
iUNCICL
A(1)BUNCICL
(1)
Ec. 42
SC73
A(1)BVIZ
(1)
Ec 43
IIE
FZ1
B18
B15
FZ3
IMP
(01-10)SnM
FZ3
MRA
(01-10)SnM
FZ3
(01-10)SnA
ADA
FZ3
(01-10)SnA
SCA
NS
A(1)DCR
(1)

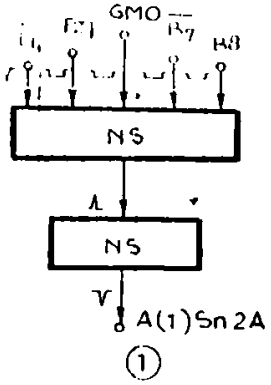
Ec 44



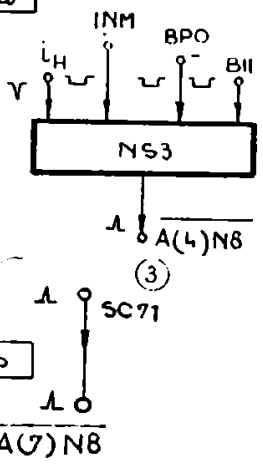
Ec 45



Ec. 46



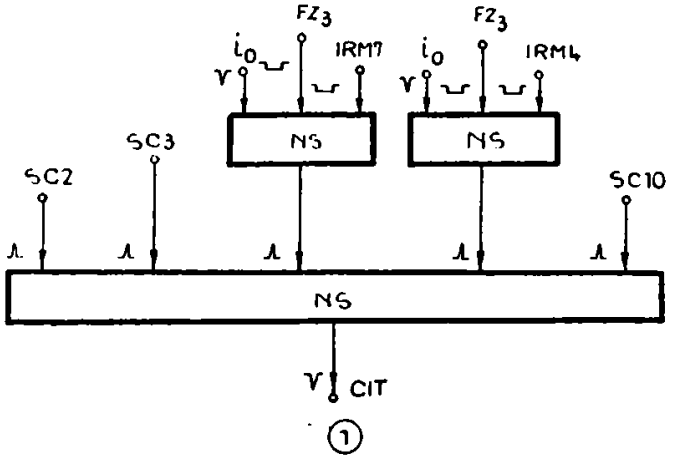
Ec. 47a

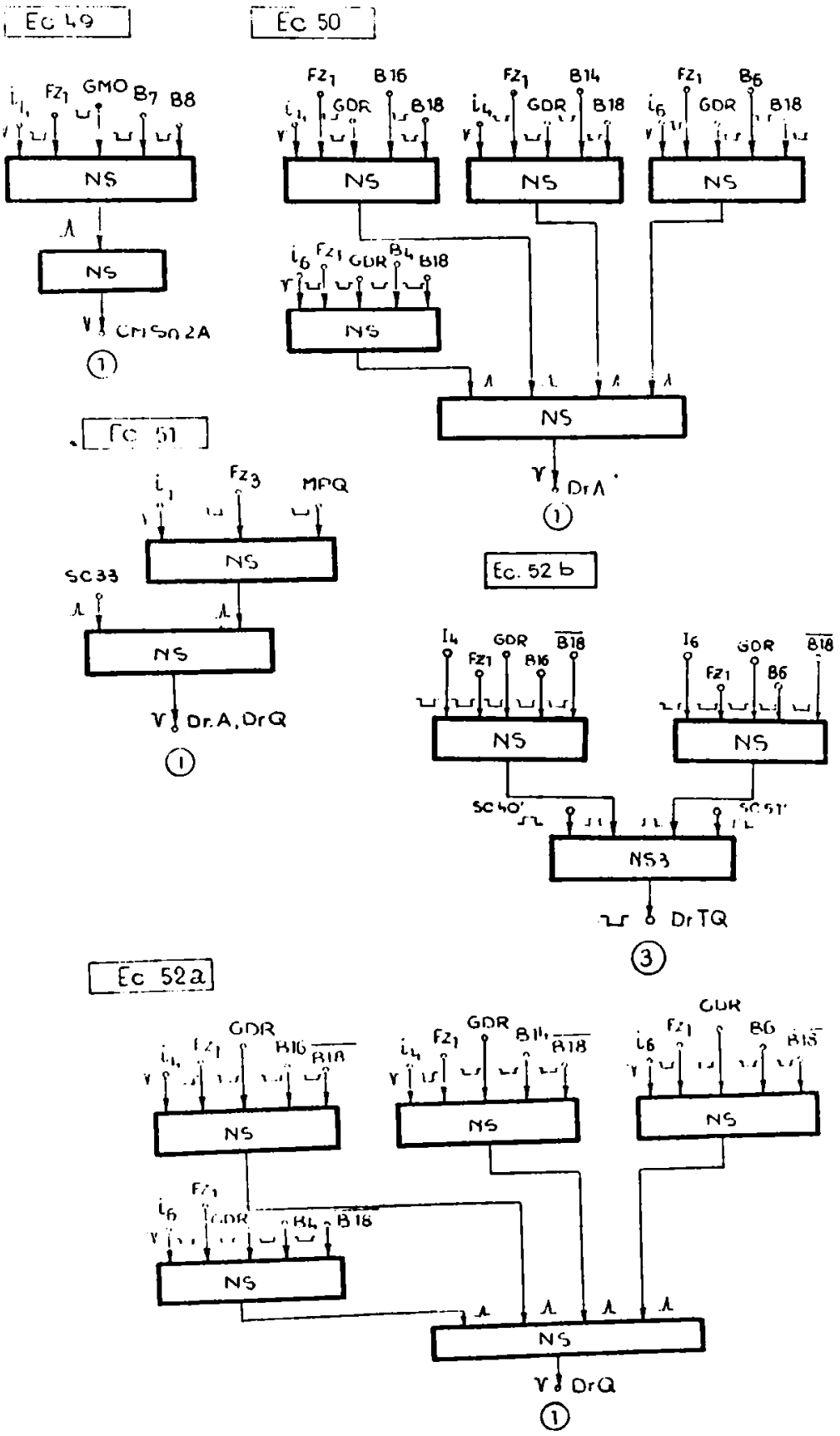


Ec. 47b

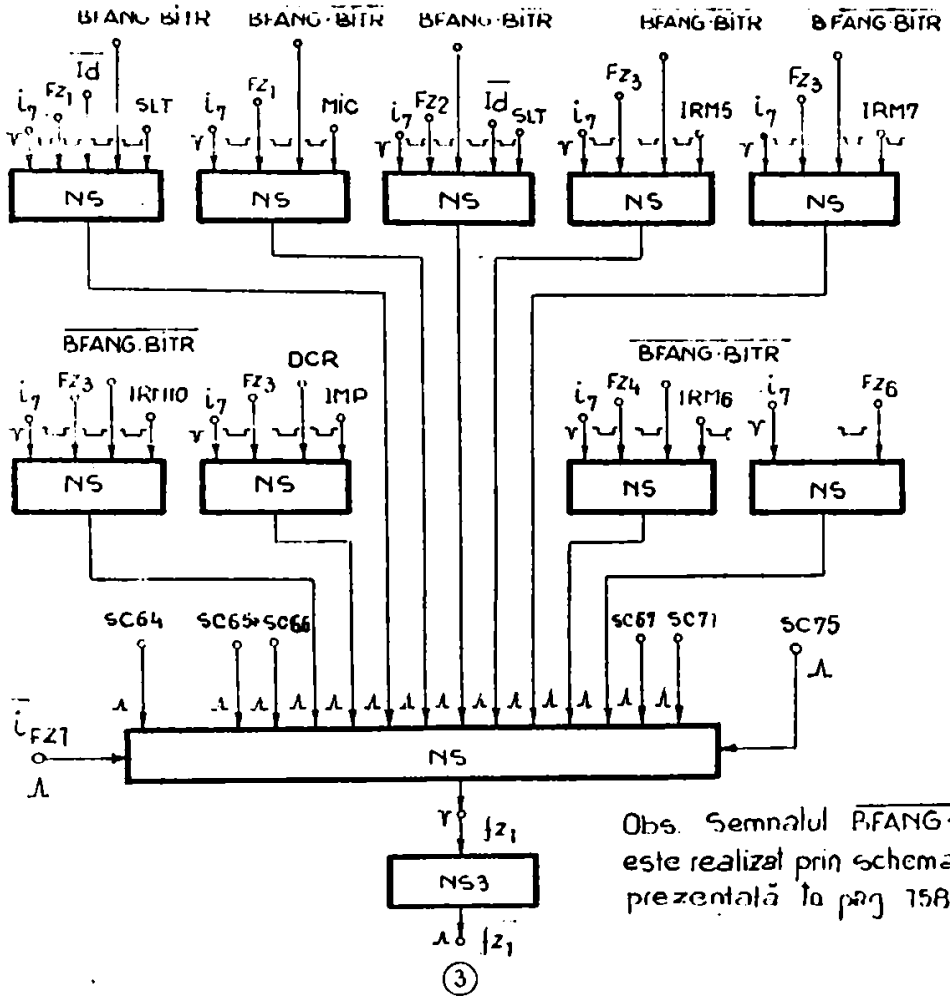
$\overline{A(7)N8}$

Ec. 48




Ec 49
Ec 50
Ec 51
Ec. 52 b
Ec 52a
NS
NS3
GDR
MPQ
SC33
SC40'
SC51'
DrA
DrA, DrQ
DrTQ
DrQ
CMSn2A

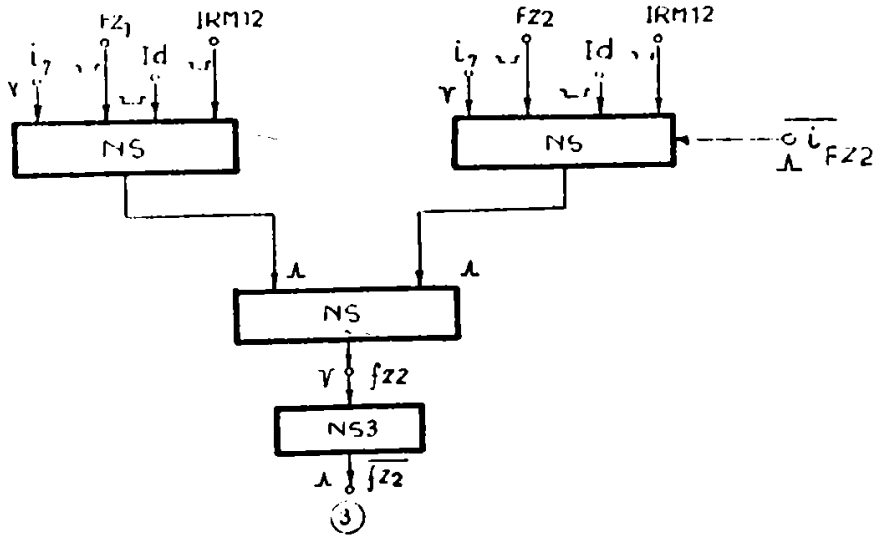
Ec 53



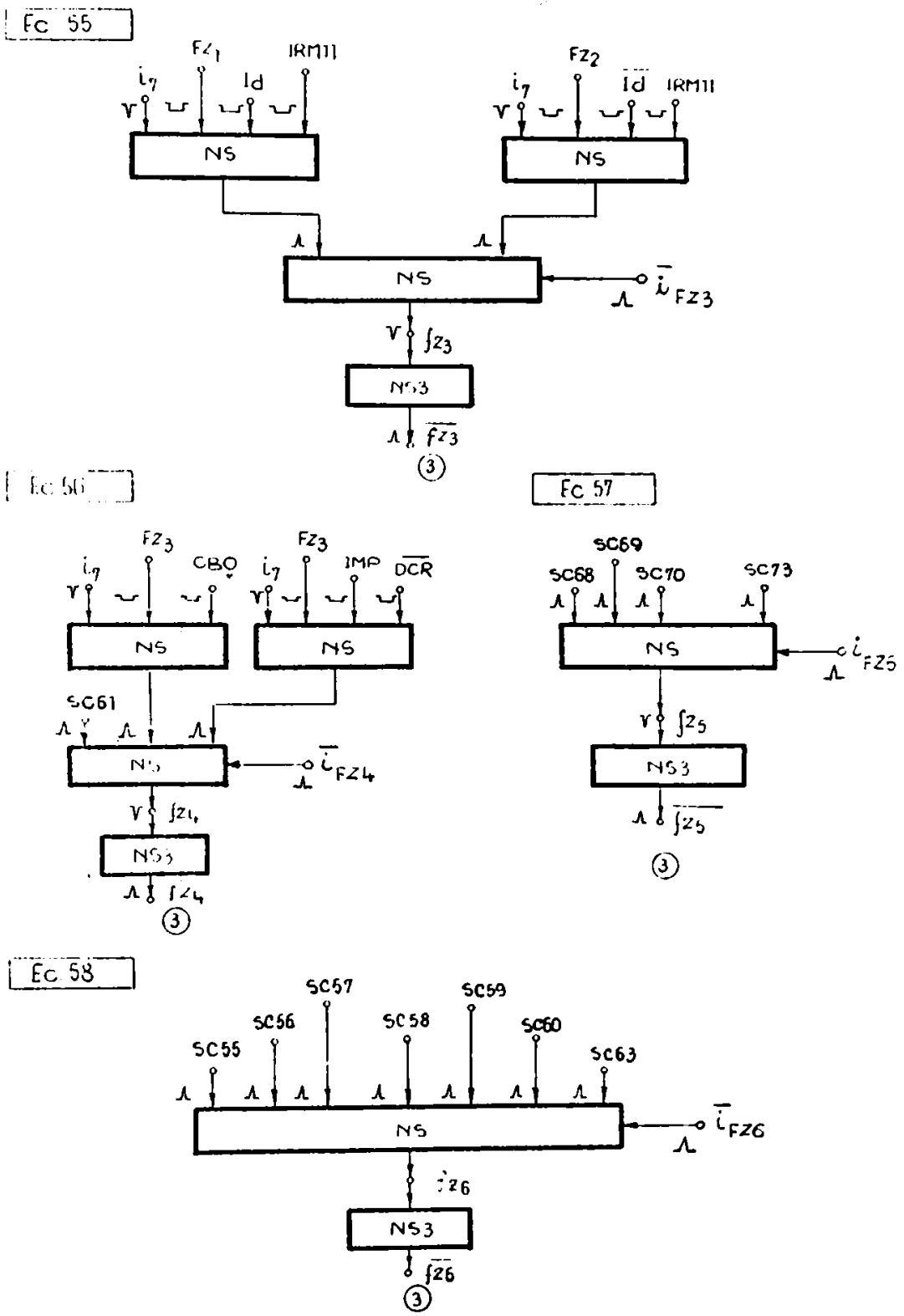
Obs. Semnalul $\overline{\text{BFANG·BITR}}$ este realizat prin schema reprezentată la pag 158

③

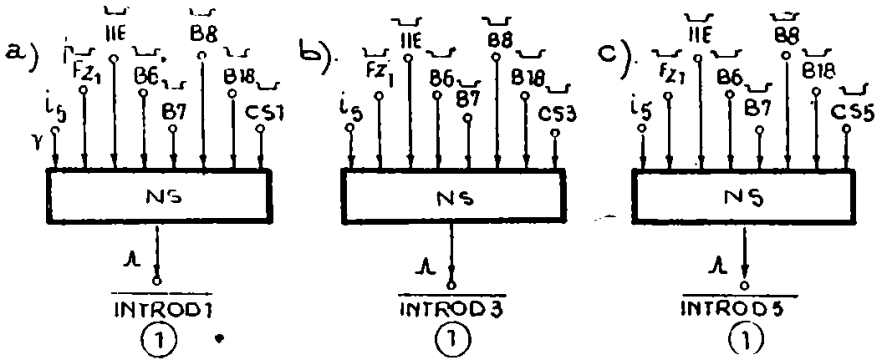
Ec 54



③

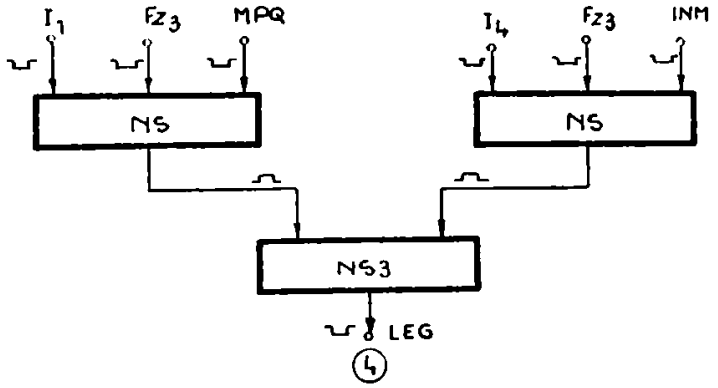

Fc 55
FZ1
IRM11
i7
Id
NS
FZ2
i7
Id
IRM11
NS
NS
$\bar{i}_{FZ3}$
fz3
NS3
$\overline{fz3}$
3
Fc 56
FZ3
CBO
i7
FZ3
IMP
DCR
i7
NS
NS
SC61
NS
$\bar{i}_{FZ4}$
fz4
NS3
fz4
3
Fc 57
SC69
SC68
SC70
SC73
NS
i_{FZ5}
fz5
NS3
$\overline{fz5}$
3
Fc 58
SC57
SC56
SC55
SC58
SC59
SC60
SC63
NS
$\bar{i}_{FZ6}$
fz6
NS3
$\overline{fz6}$
3

Ec. 59

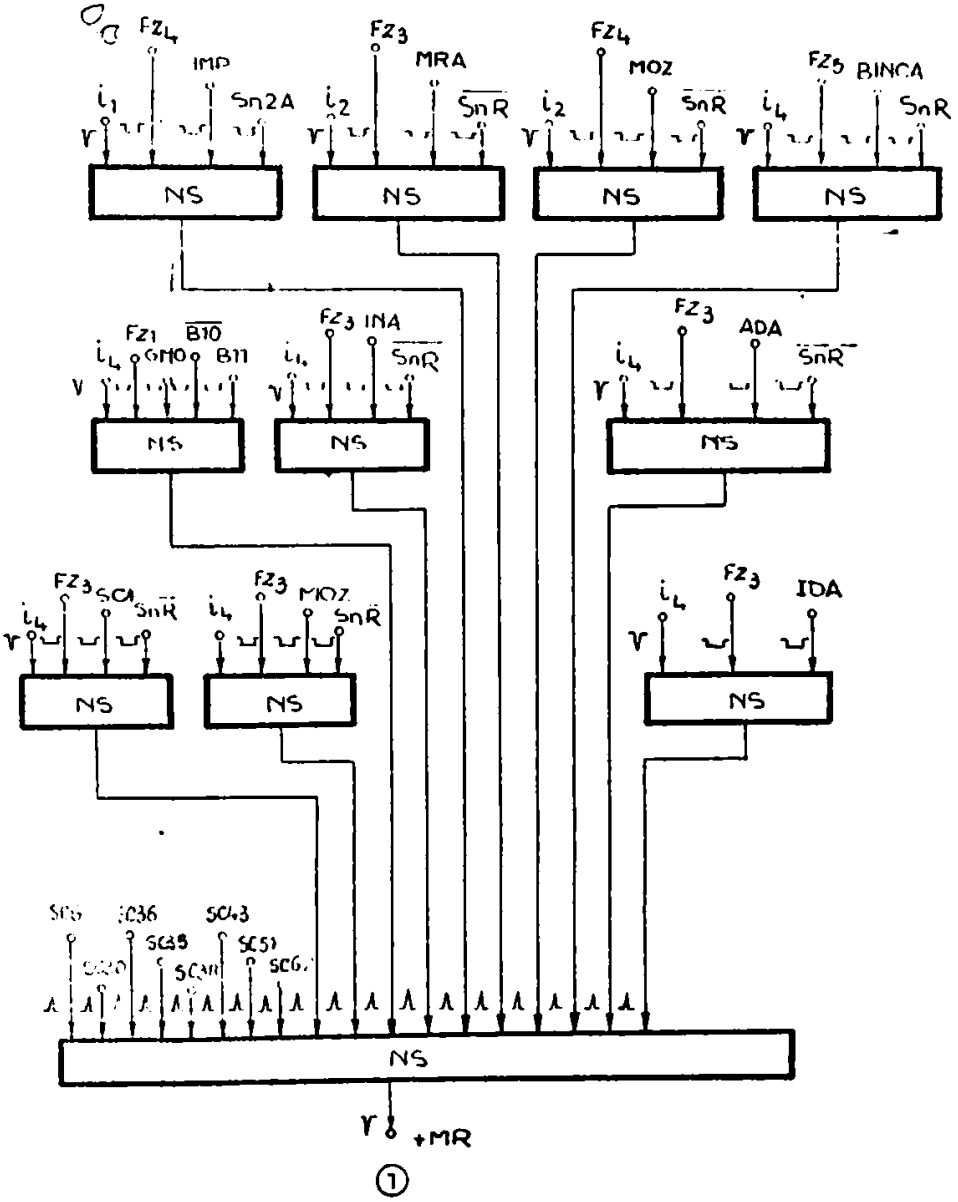


Obs. Semnalele CS1, CS3, si CS5 sint realizate prin schema reprezentată la pag. 158

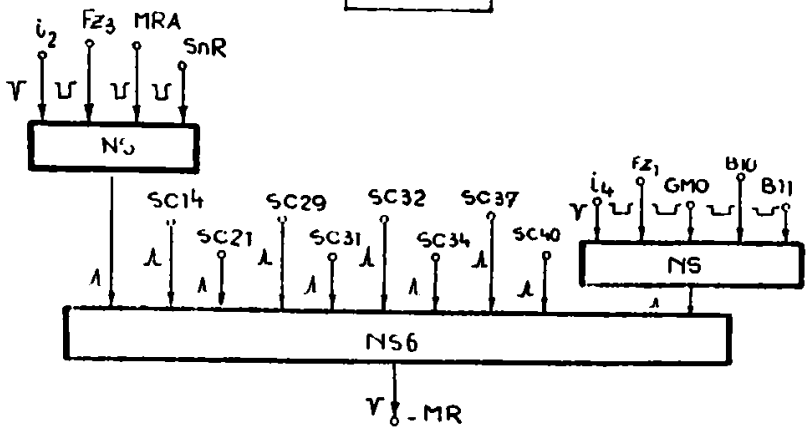
Ec. 60



Fc 61

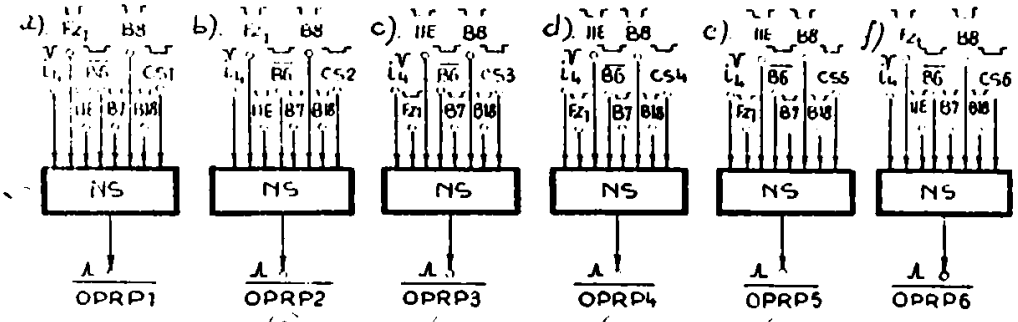

FZ4
IMP
Sn2A
FZ3
MRA
SnR
FZ4
MOZ
SnR
FZ5
BINCA
SnR
NS
NS
NS
NS
FZ1
B10
B11
FZ3
INA
SnR
FZ3
ADA
SnR
NS
NS
NS
FZ3
SCA
SnR
FZ3
MOZ
SnR
FZ3
IDA
NS
NS
NS
SC36
SC43
SC51
NS
+MR
1

Ec 62



(24)

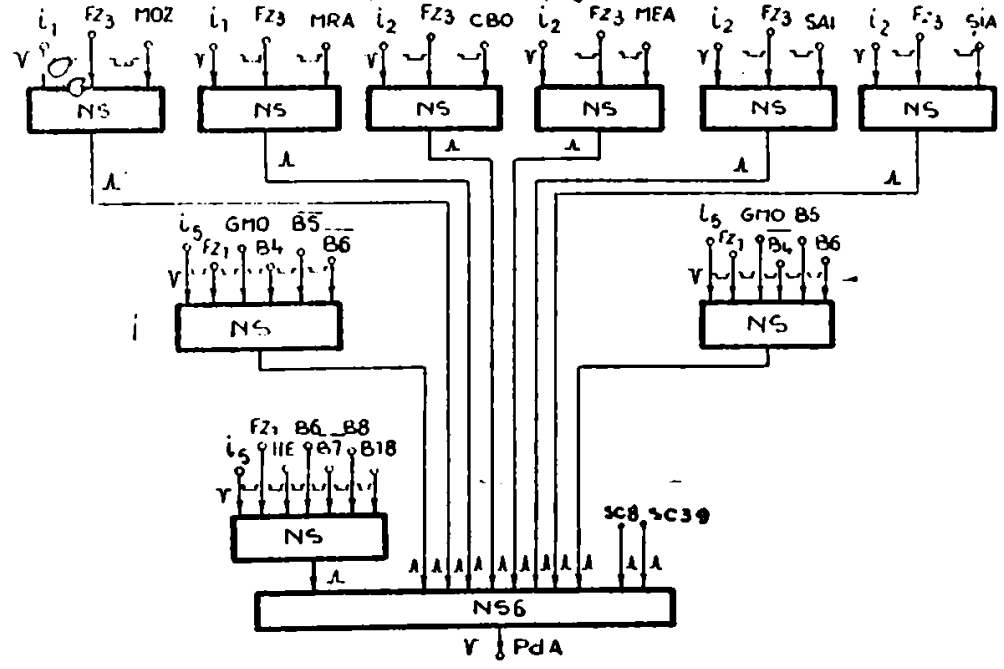
Ec.63



(1) (1) (1) (1) (1) (1)

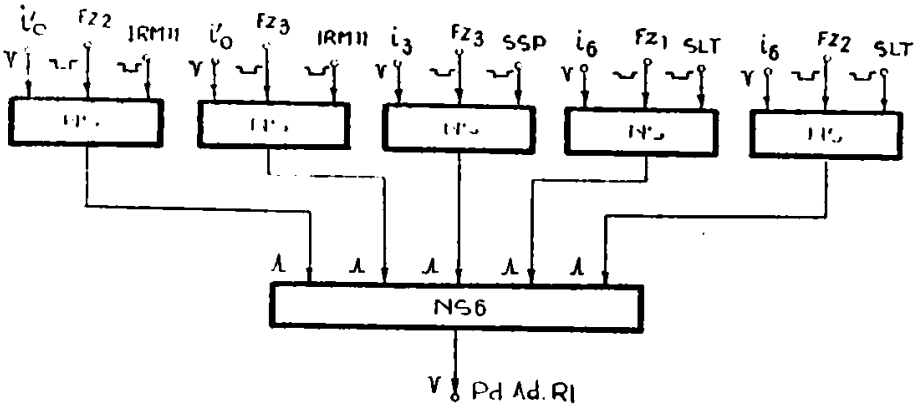
Ec 64

Obs. Semnalele CS1÷CS6 sînt realizate prin schema reprezentată la pag. 158



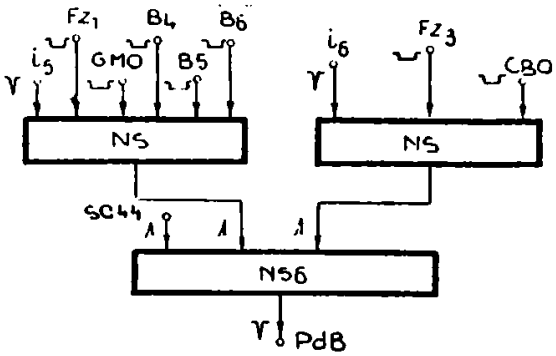
(24)

Ec 65



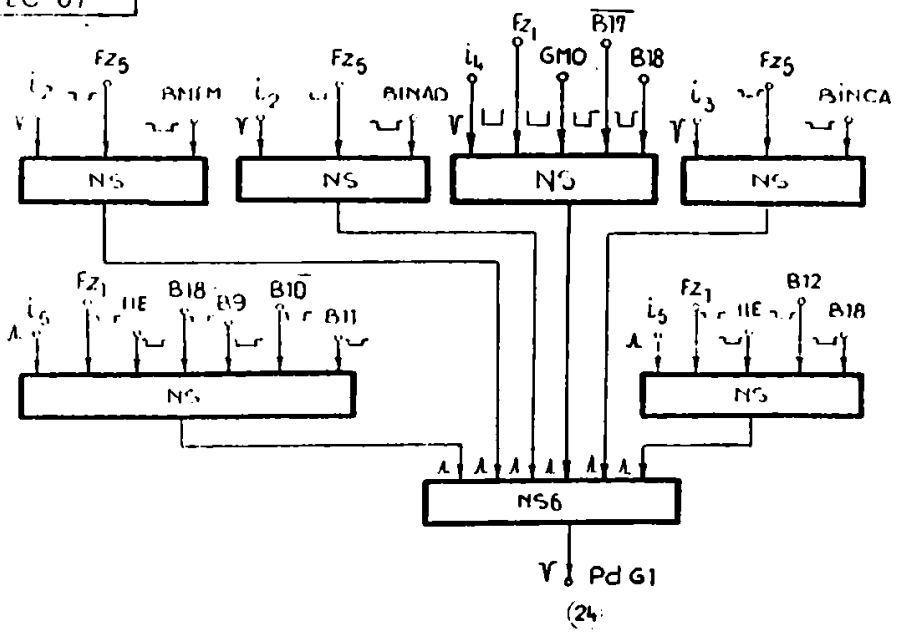
(18)

Ec 66

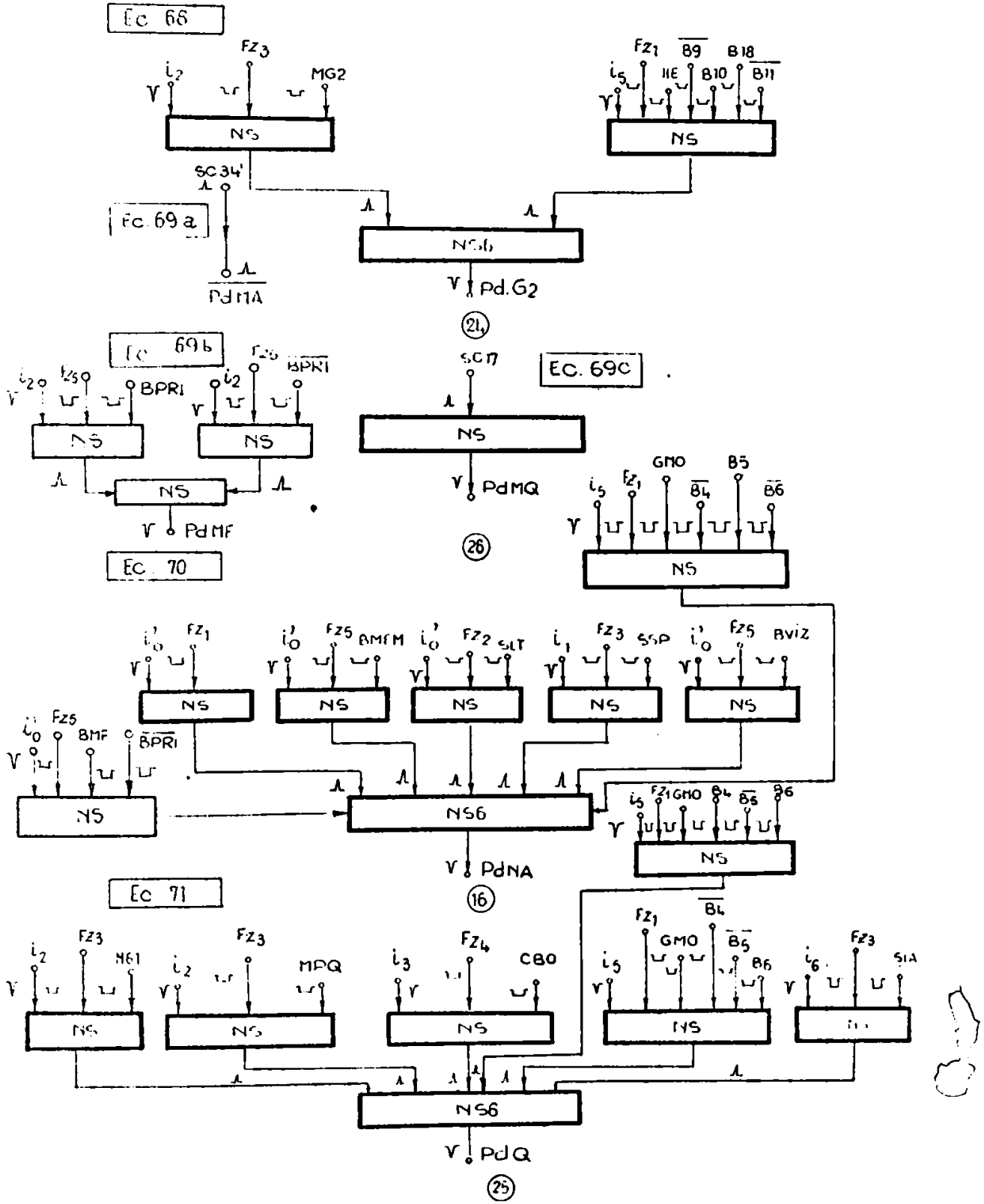


(24)

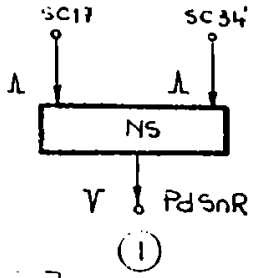
Ec 67



(24)


Ec 68
Fz3
i2
MG2
NS
Fz1 B9 B18
i5 11E B10 B11
NS
SC34'
Ec 69a
NS6
PdMA
Pd.G2
21
Ec 69b
BPRI
NS
SC17
Ec. 69c
PdMQ
26
GM0 B5
B4 B6
PdMF
Ec 70
Fz1
Fz5 BMFM
Fz2 SLT
Fz3 SSP
Fz5 BVIZ
BMF
BPRI
NS6
PdNA
16
Ec 71
Fz3
NG1
MPQ
Fz4
CBO
GM0
B4
B5
B6
SIA
NS6
PdQ
25

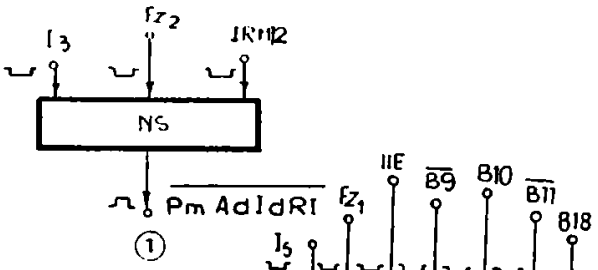
Ec. 73



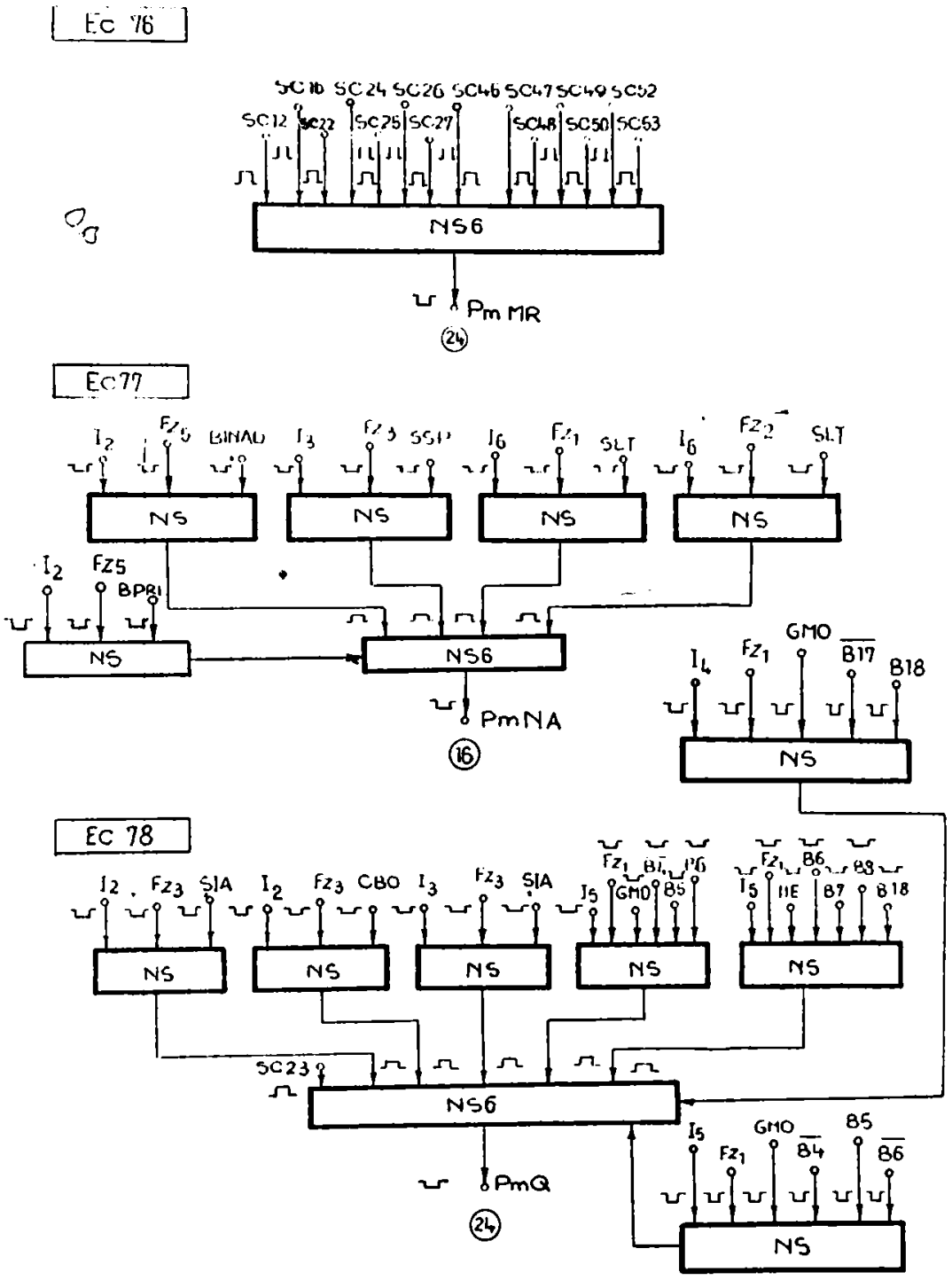
Ec. 72.

Ec. 72. a fost suprimată în urma unor modificări.

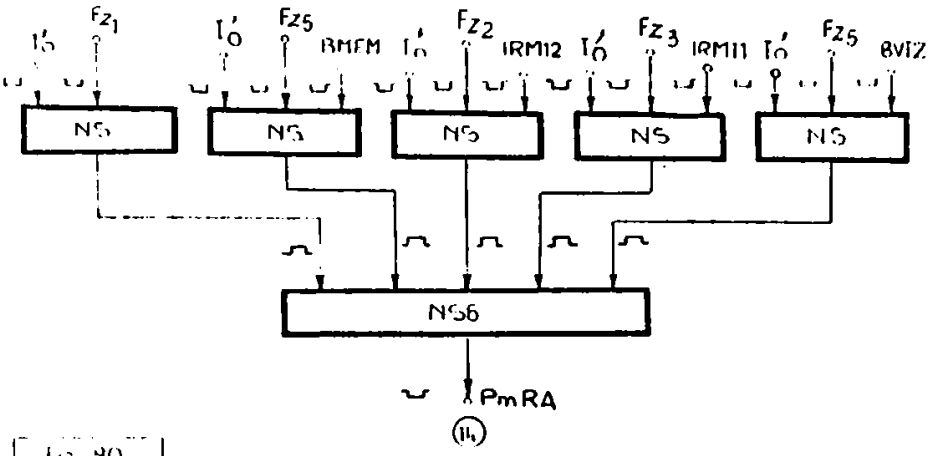
Ec. 74



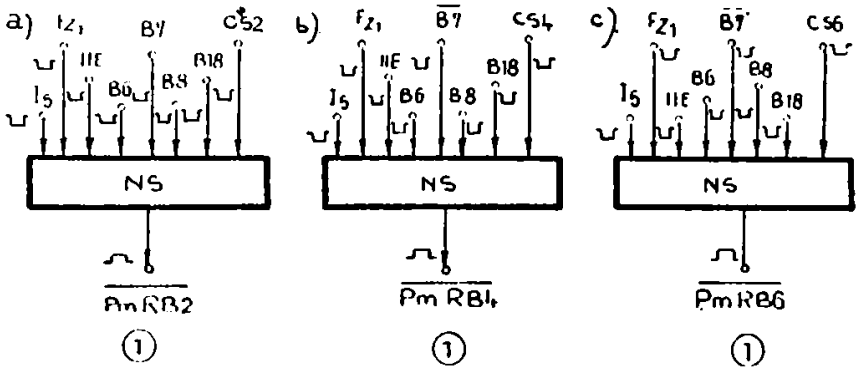
Ec. 75


Ec 76
SC10 SC24 SC26 SC46 SC47 SC49 SC52
SC12 SC22 SC25 SC27 SC48 SC50 SC53
NS6
Pm MR
24
Ec77
I2 Fz5 BINAU I3 Fz3 SSP I6 Fz1 SLT I6 Fz2 SLT
NS NS NS NS
I2 FZ5 BPBI
NS
NS6
PmNA
16
I4 Fz1 GMO B17 B18
NS
EC 78
I2 Fz3 SIA I2 Fz3 CBO I3 Fz3 SIA I5 Fz1 B7 I6 GMO B5 I5 Fz1 B6 B8 IIE B7 B18
NS NS NS NS NS
SC23
NS6
PmQ
24
I5 Fz1 GMO B4 B5 B6
NS

Ec. 79

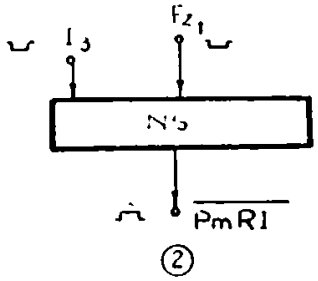


Ec. 80

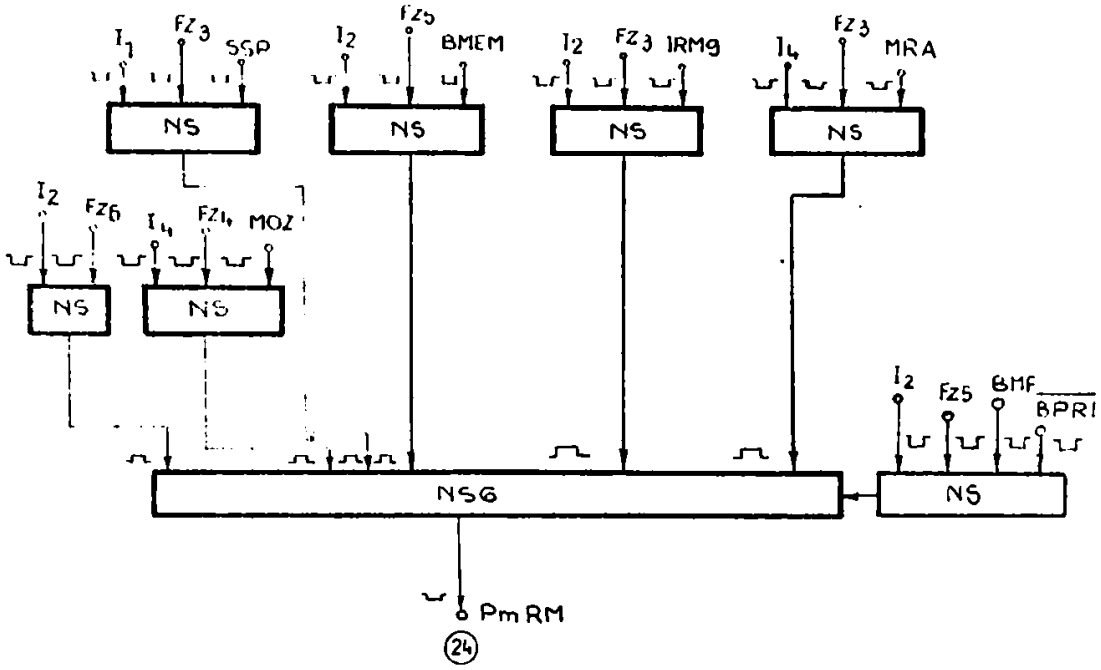


Ec. 81

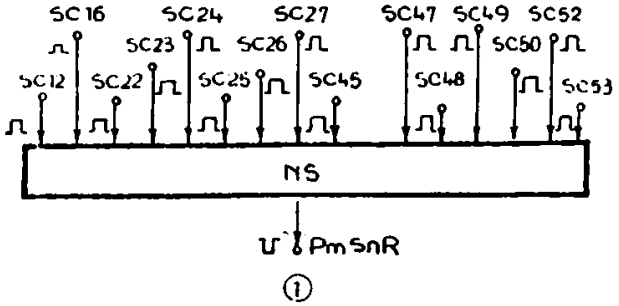
Obs. Semnalele CS2, CS4 și CS6 sînt realizate prin schema reprezentată la pag. 158



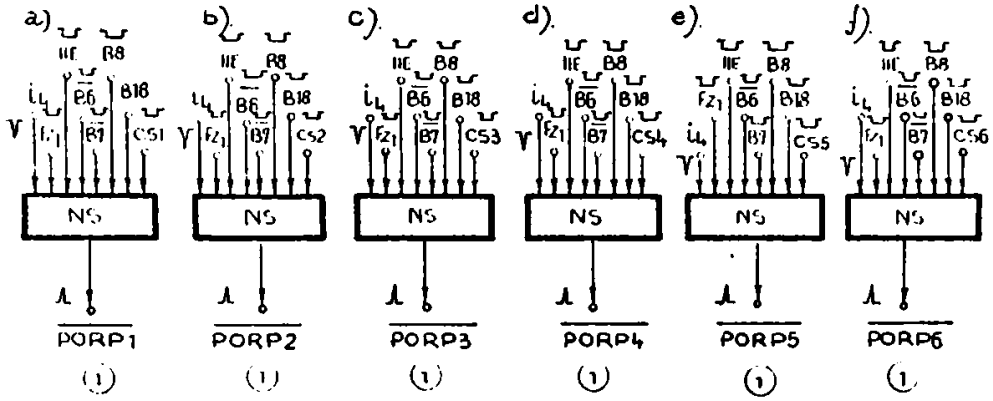
Ec 82



Ec 83

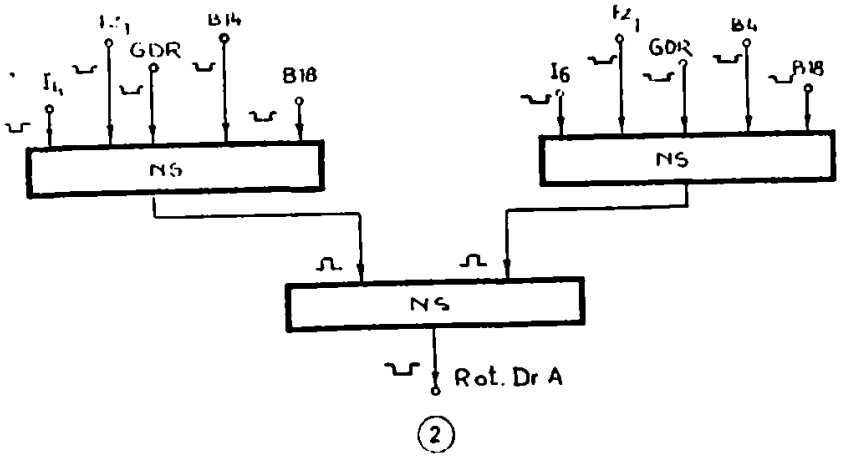


Ec. 84

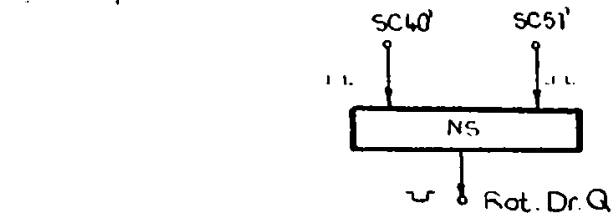


Obs. semnalele CS1 ÷ CS6 sînt realizate prin schema reprezentată la pag. 158

Ec 85

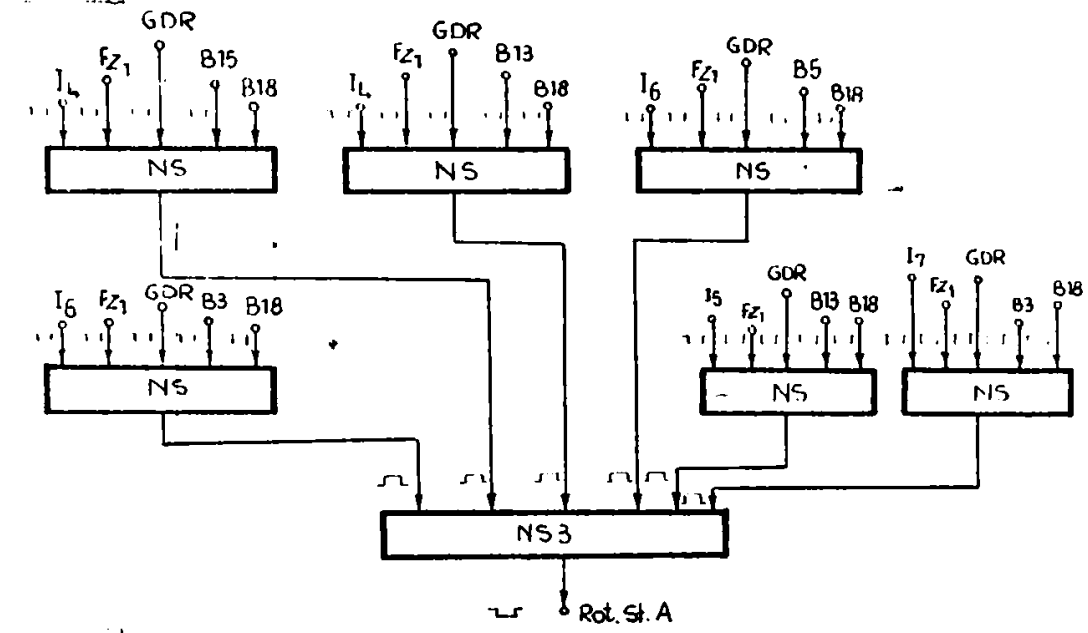


Ec 86



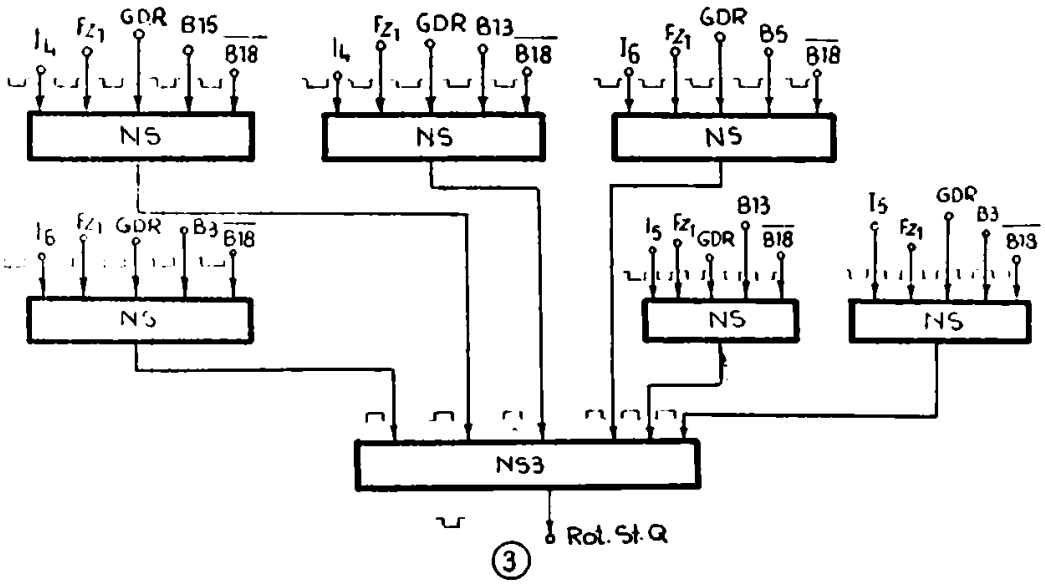
(1)

Ec 87

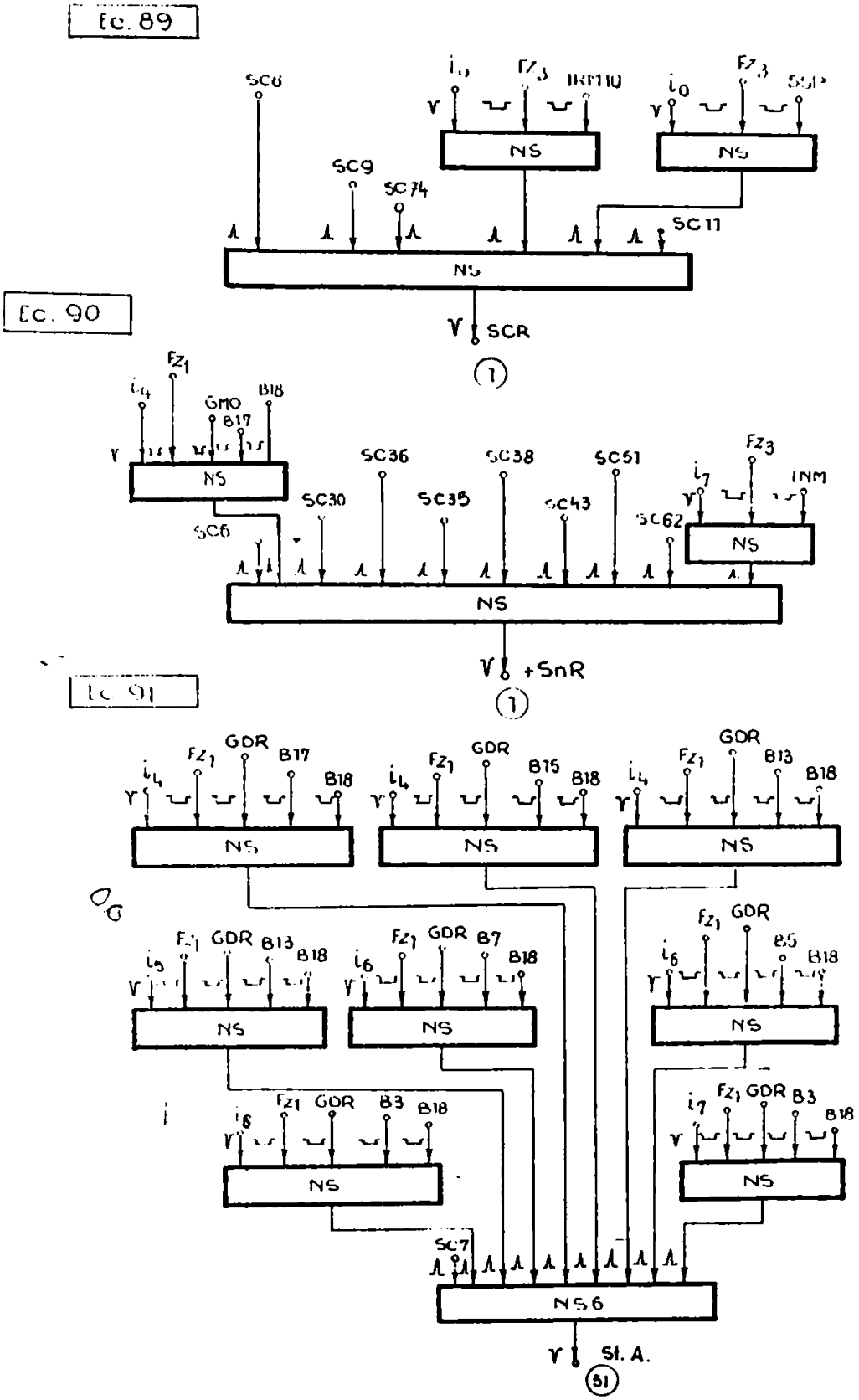


③

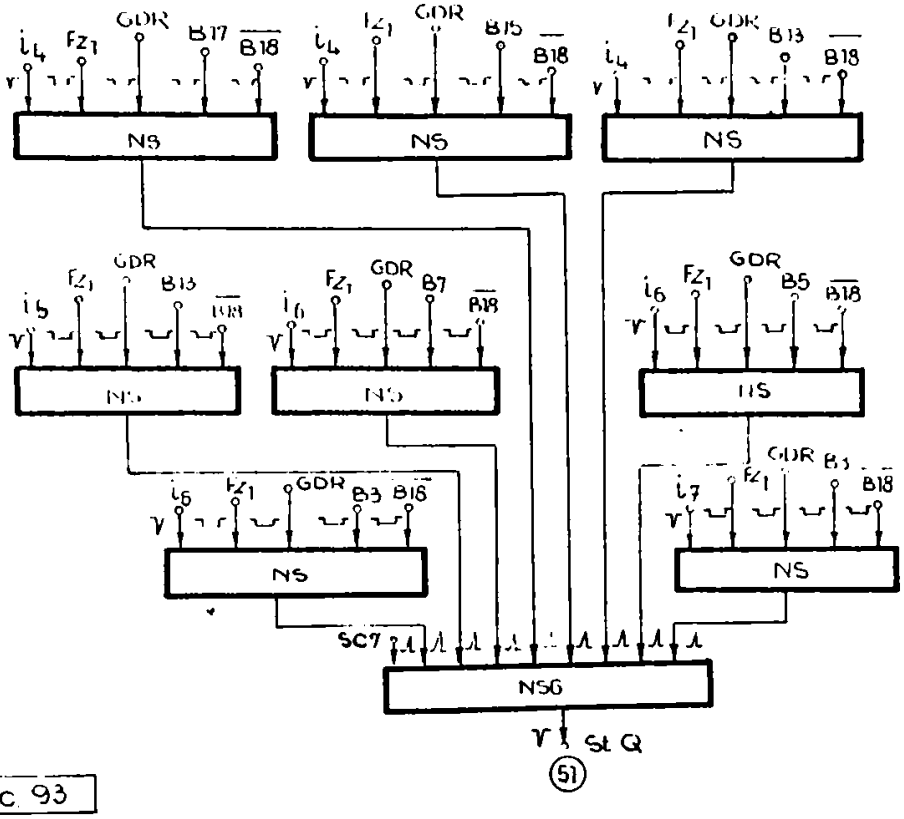
Ec. 88



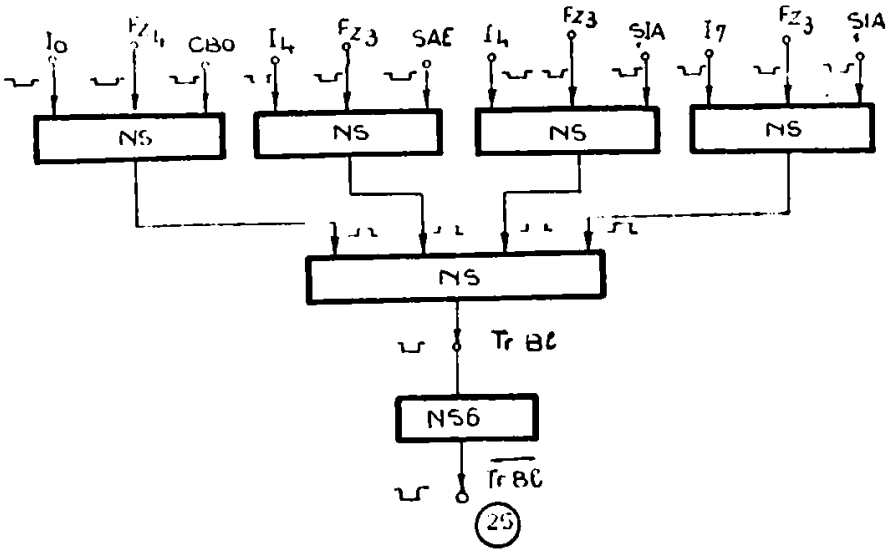
③


Ec. 89
SC8
SC9
SC74
SC11
NS
SCR
Ec. 90
SC6
SC30
SC36
SC35
SC38
SC43
SC51
SC62
INM
+SnR
Ec. 91
GDR
B17
B18
B15
B13
B7
B5
B3
SC7
NS6
St. A.
51

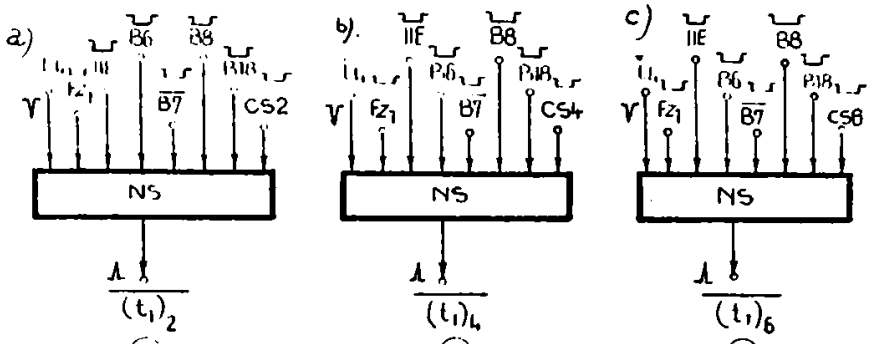
Ec. 92



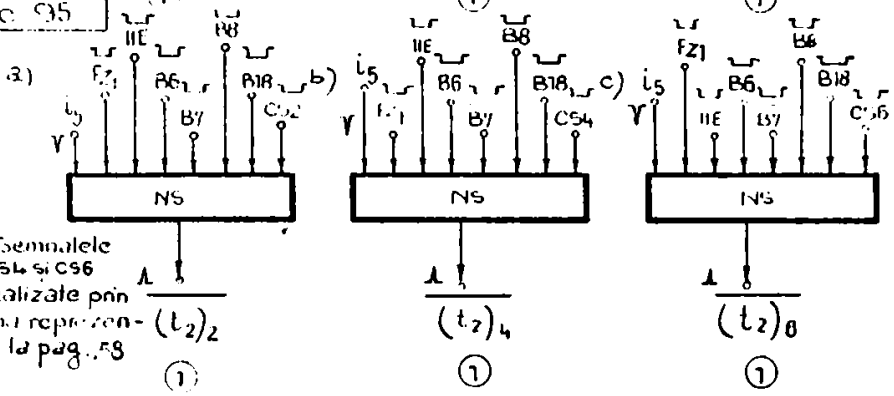
Ec. 93



Ec. 94

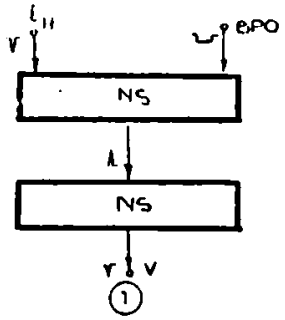


Ec. 95

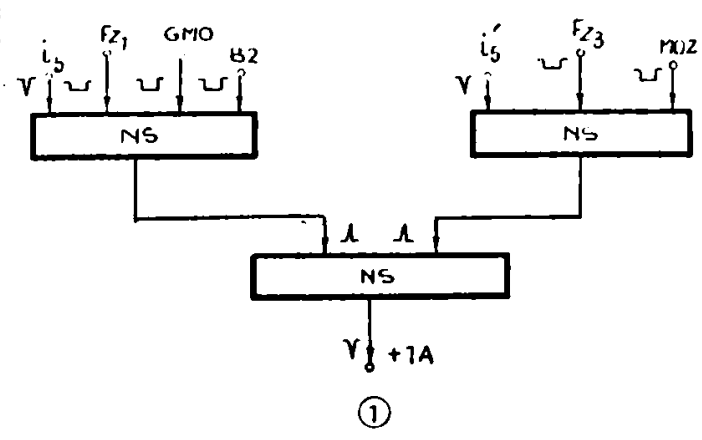


Obs. Semnalele CS2, CS4 și CS6 sînt realizate prin schema reprezentată la pag..58

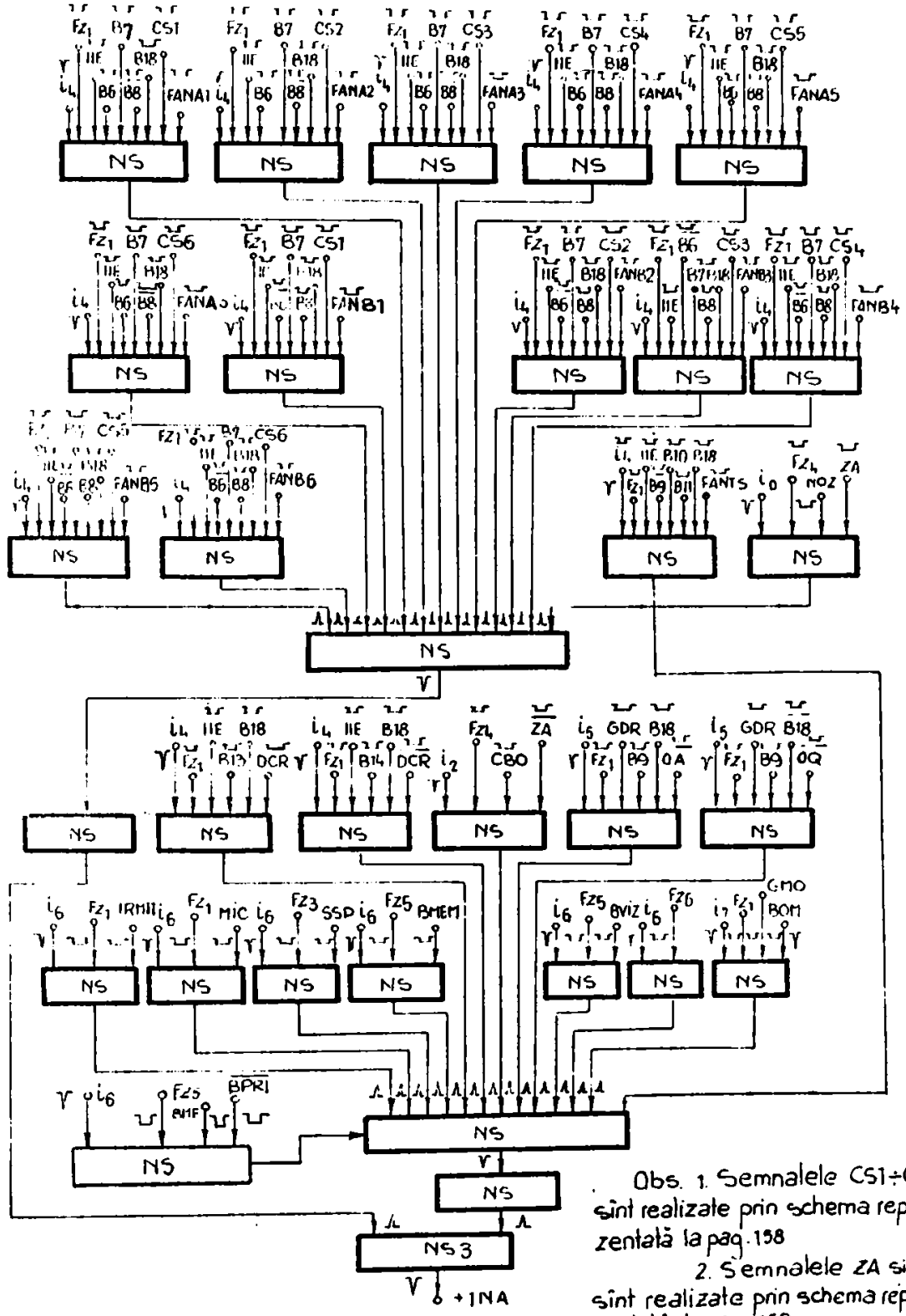
Ec. 96



Ec. 97



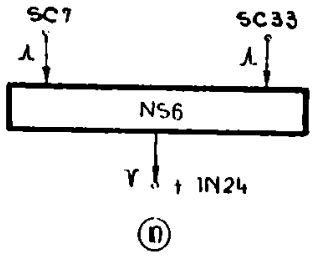
Ec. 98



Obs. 1. Semnalele CS1÷CS6 sînt realizate prin schema reprezentată la pag. 158

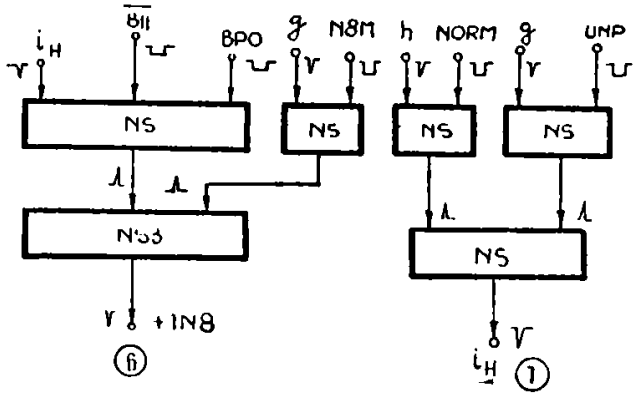
2. Semnalele ZA și $\overline{ZA}$ sînt realizate prin schema reprezentată la pag. 159

Fc 99

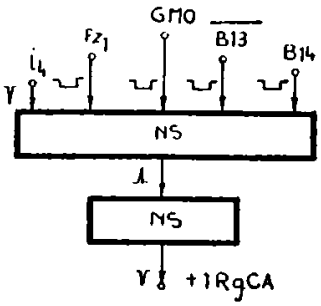


(n)

Fc 100

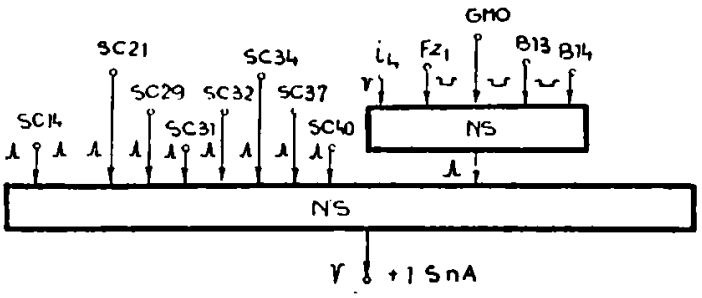


Fc 101



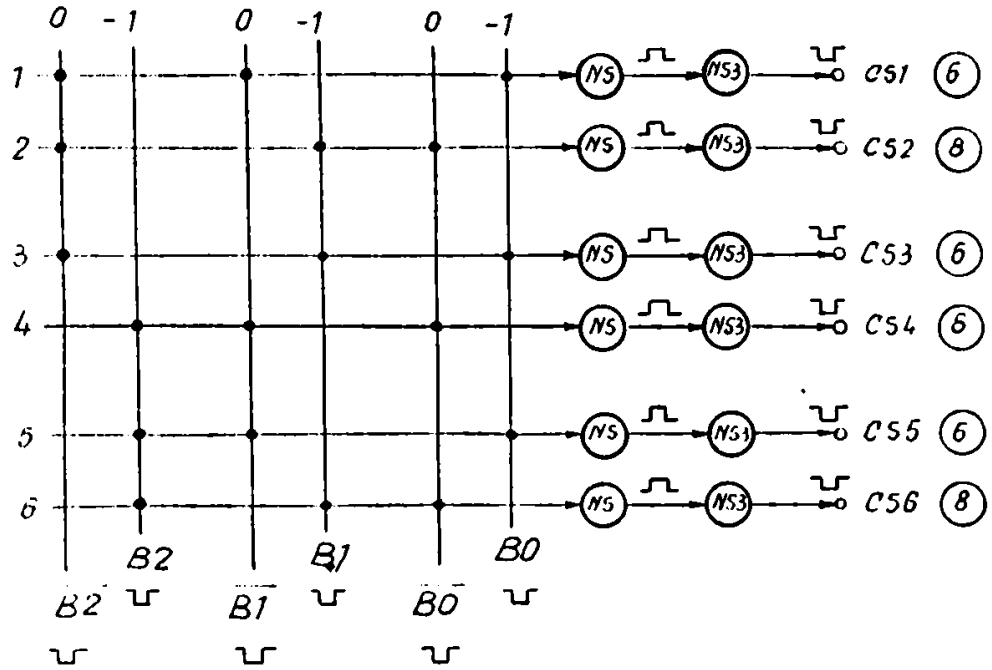
(1)

Fc 102

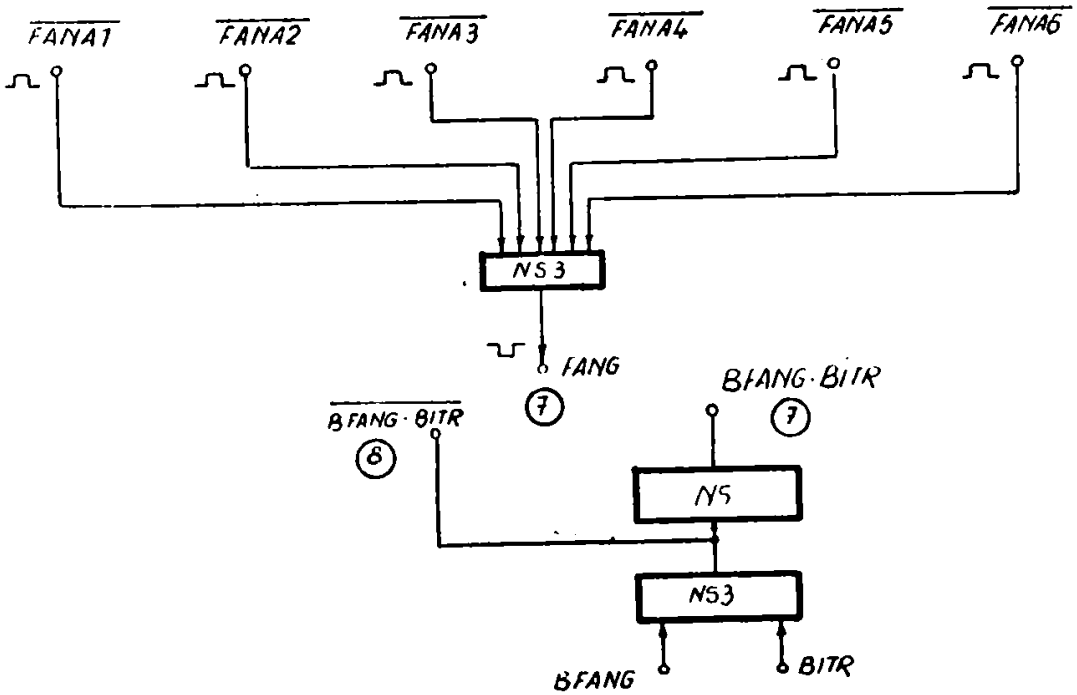


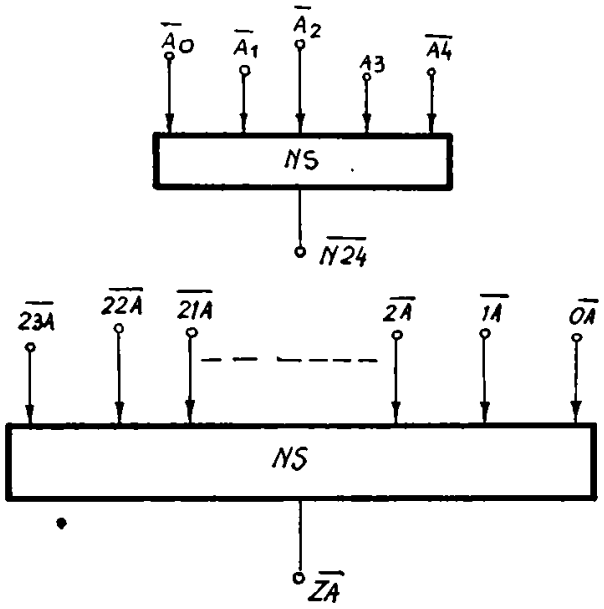
(1)

Schema descifratorului codului de selecție

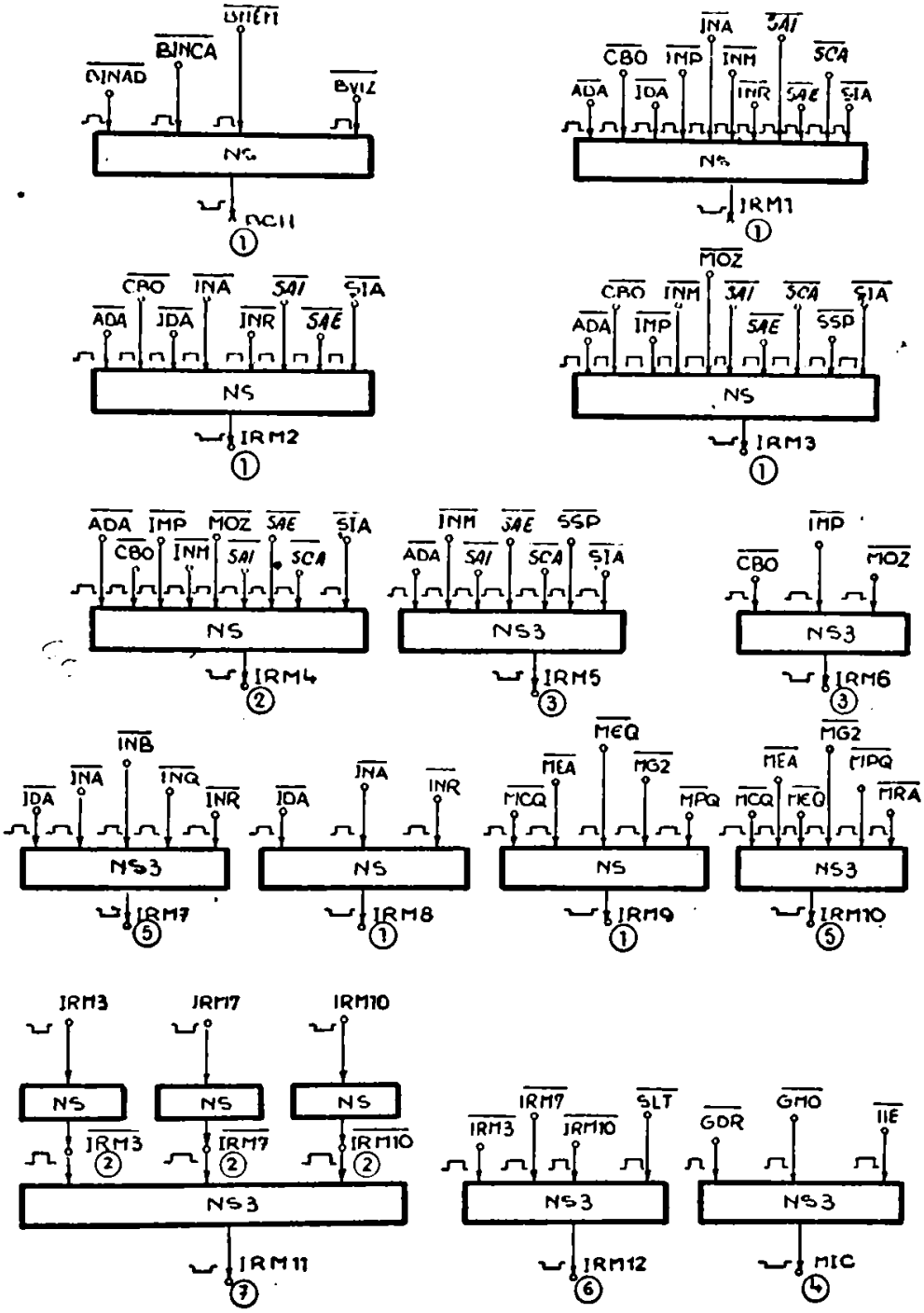


Schema logică pentru formarea semnalului FANG

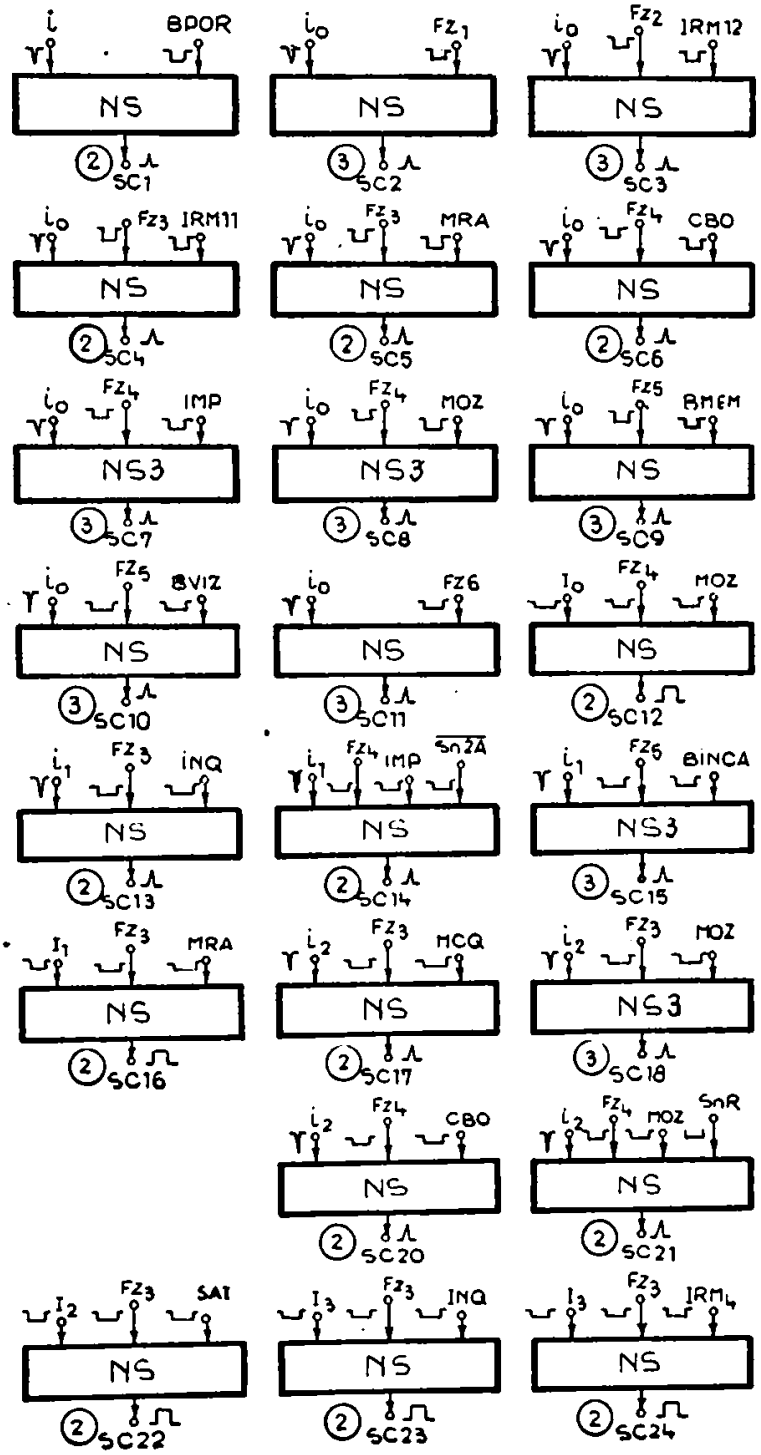


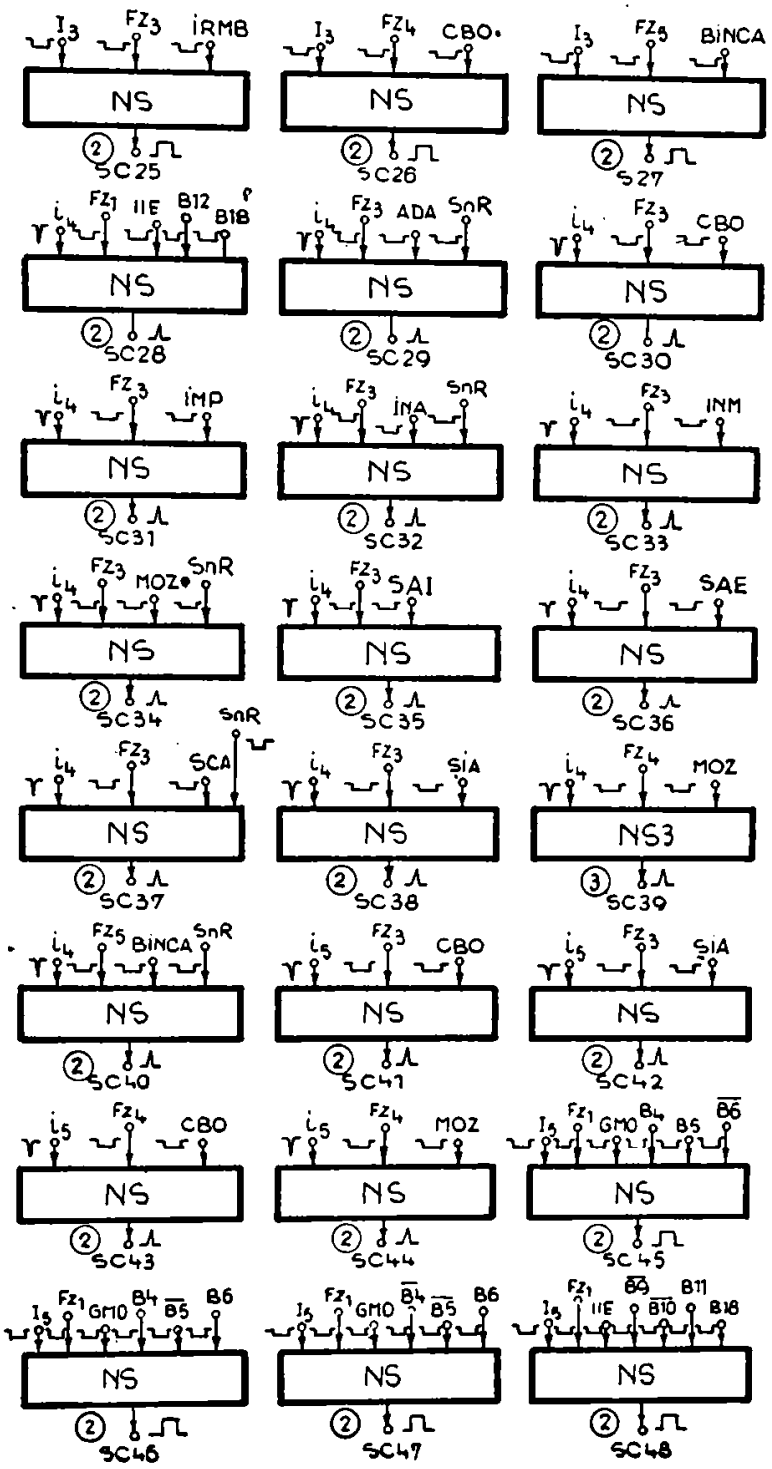

$\overline{A_0}$
$\overline{A_1}$
$\overline{A_2}$
A_3
$\overline{A_4}$
NS
$\overline{N24}$
$\overline{23A}$
$\overline{22A}$
$\overline{21A}$
$\overline{2A}$
$\overline{1A}$
$\overline{0A}$
NS
$\overline{ZA}$

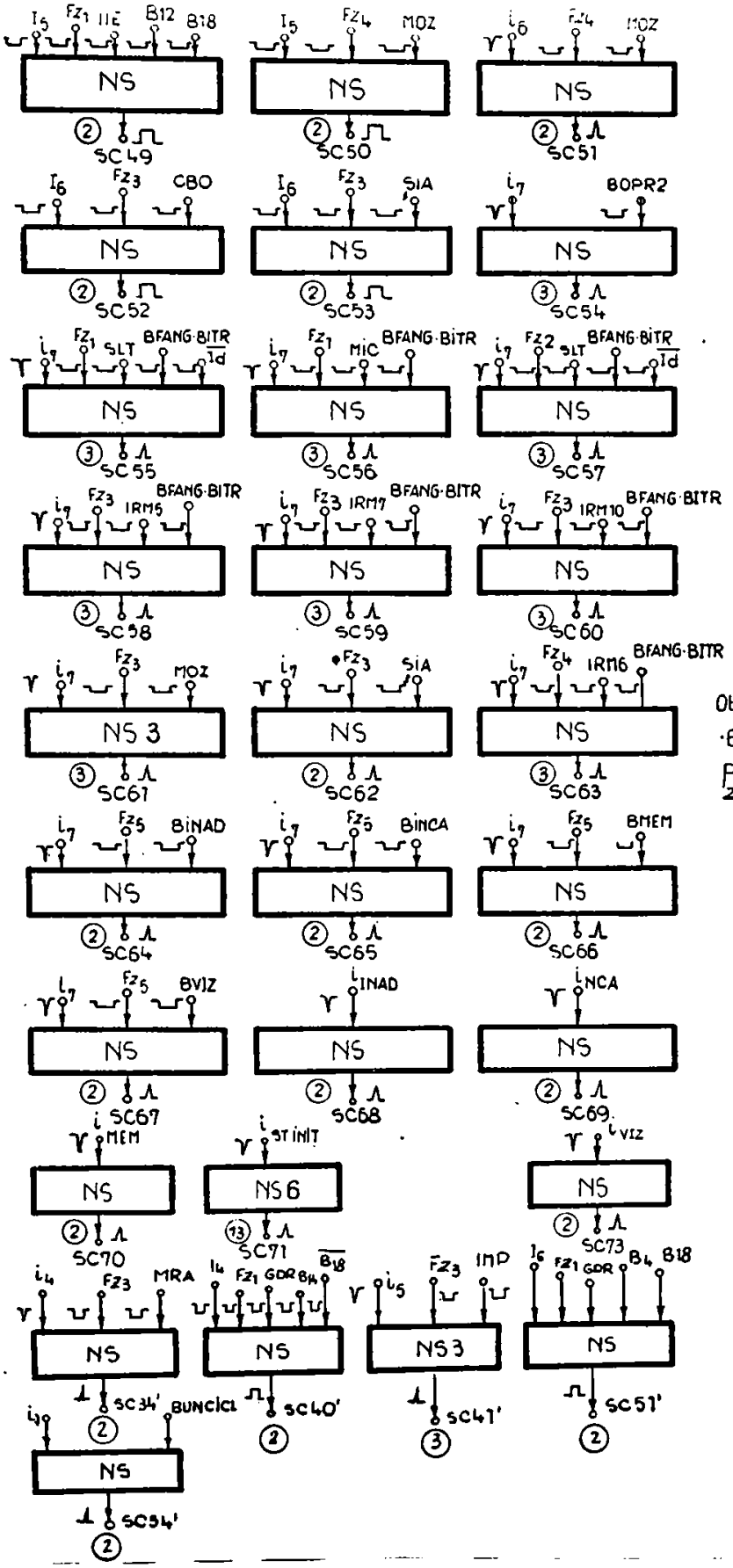
Sinteza schemelor pe baza tabelului II



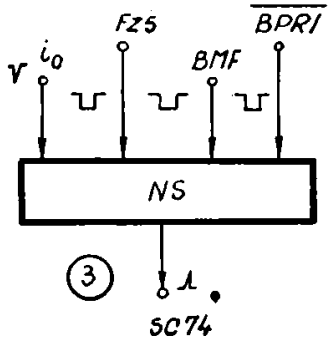
Sinteza schemelor semnalelor comune (SC) pe baza tabelului IV

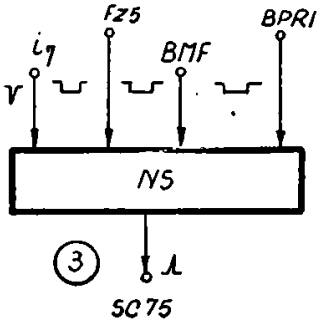


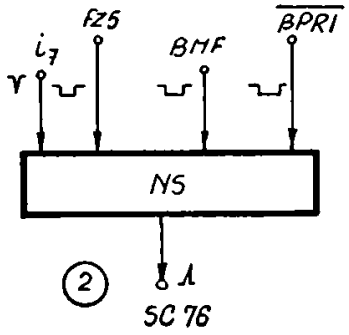

I3 FZ3 IRMB NS ② SC25
I3 FZ4 CBO NS ② SC26
I3 FZ5 BINCA NS ② SC27
i4 FZ1 IIE B12 B18 NS ② SC28
i4 FZ3 ADA SnR NS ② SC29
i4 FZ3 CBO NS ② SC30
i4 FZ3 IMP NS ② SC31
i4 FZ3 INA SnR NS ② SC32
i4 FZ3 INM NS ② SC33
i4 FZ3 MOZ SnR NS ② SC34
i4 FZ3 SAI NS ② SC35
i4 FZ3 SAE NS ② SC36
i4 FZ3 SCA SnR NS ② SC37
i4 FZ3 SIA NS ② SC38
i4 FZ4 MOZ NS3 ③ SC39
i4 FZ5 BINCA SnR NS ② SC40
i5 FZ3 CBO NS ② SC41
i5 FZ3 SIA NS ② SC42
i5 FZ4 CBO NS ② SC43
i5 FZ4 MOZ NS ② SC44
I5 FZ1 GM0 B4 B5 B6 NS ② SC45
I5 FZ1 GM0 B4 B5 B6 NS ② SC46
I5 FZ1 GM0 B4 B5 B6 NS ② SC47
I5 FZ1 IIE B9 B10 B11 B18 NS ② SC48



Obs. Semnalul BFANG·BITR se realizează prin schema reprezentată la pag. 158


FZ5
BPRI
BMF
i_0
NS
3
SC74


FZ5
BPRI
BMF
i_7
NS
3
SC75


FZ5
BPRI
BMF
i_7
NS
2
SC76