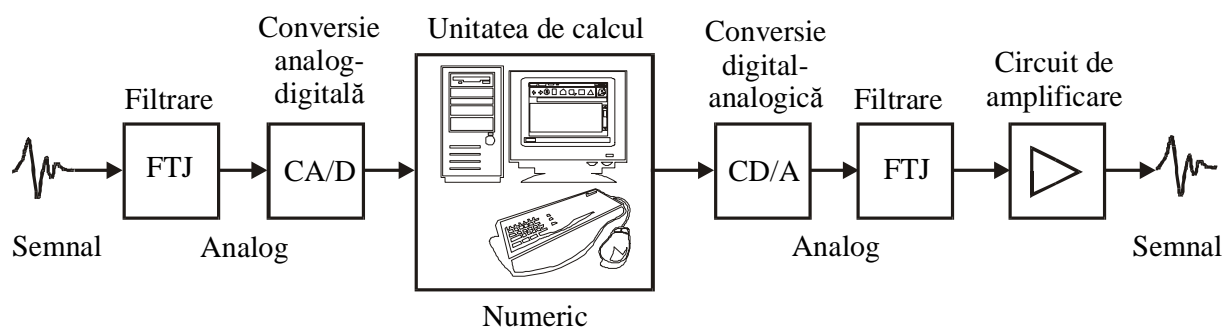


COSTIN ȘTEFĂNESCU

NICOLAE CUPCEA

ELECTRONICĂ APLICATĂ - SISTEME INTELIGENTE HARDWARE- SOFTWARE DE MĂSURARE ȘI CONTROL



București 2003

CUPRINS

1. Elemente introductive referitoare la conducerea proceselor industriale din perspectiva sistemelor inteligente hardware-software de măsurare și control	1
1.1 Introducere.....	1
2. Sisteme de achiziție și prelucrare a datelor	5
2.1 Noțiuni generale	5
2.2 Sisteme de achiziții de date. Arhitectură. Principalele tipuri de resurse utilizate în cadrul sistemelor de achiziții de date	5
2.2.1 Multiplexoare analogice utilizate în sisteme de achiziții de date	6
2.2.2 Circuite de eșantionare-memorare utilizate în sisteme de achiziții de date	9
2.2.3 Circuite pentru conversia datelor utilizate în sisteme de achiziții de date: convertoare analog-digitale și digital-analogice	14
2.2.3.1 Converteoare digital-analogice. Scheme de principiu	19
2.2.3.2 Converteoare analog-digitale. Scheme de principiu	25
2.3 Interfețe specializate de comunicație	35
2.3.1 Comunicația de tip serial. Protocoale de transmisie serială a datelor	36
2.3.1.1 Interfața RS-232	36
2.3.1.2 Interfața I ² C	40
2.3.1.2.1 Specificațiile interfeței I ² C	41
2.3.1.2.2 Conceptul de magistrală I ² C	42
2.3.1.2.3 Transferurile pe magistrala I ² C	43
2.3.1.2.3.1 Transferurile de date pe magistrală	44
2.3.1.2.4 Arbitrarea priorităților și generarea ceasului	45
2.3.1.3 Interfața USB	46
2.3.2 Comunicația de tip paralel. Protocoale de transmisie paralelă a datelor	48
2.3.2.1 Interfața HPIB	49
2.3.2.1.1 Structura bus-ului HPIB	50
2.3.2.2 Interfața Centronics	53
2.3.2.2.1 Protocolul de comunicație Centronics-Handshake	55
3. Tipuri de sisteme de achiziții de date	57
3.1 Sistem de achiziții de date cu multiplexare temporală	57
3.2 Sistem de achiziție sincronă de date	61
3.3 Sistem rapid de achiziții de date	67
3.4 Unitatea centrală de comandă	68
3.5 Sisteme de achiziție de date cu microprocesor	69
3.5.1 Unități centrale de prelucrare tradiționale	69
3.5.2 Procesoare de semnal: DSP	75
3.5.2.1 Arhitectura unui procesor de semnal	76

3.5.2.2	Portul serial sincron al familiei DSP TMS320C2xx.....	81
3.5.2.3	Portul serial asincron al familiei DSP TMS320C2xx	82
4.	Considerații generale asupra instrumentației virtuale.....	85
4.1	Instrumente virtuale.....	85
4.2	Interfața calculator - proces de măsurare sau control	90
4.3	Software pentru instrumentație virtuală.....	92
4.3.1	Alegerea platformei software: Unix sau Windows?	94
4.4	Particularități ale instrumentației virtuale.....	95
4.5	Noi instrumente DAQ specializate extind noțiunea de instrument virtual.....	97
4.5.1	Transferul de date în bus-ul PCI	98
4.5.2	Implementarea DMA pe placa de tip PCI Bus Master. Chip-ul ASIC MITE.	99
4.5.3	Windows NT 4.0 aduce îmbunătățiri importante pentru utilizatorii de instrumentație virtuală.....	101
4.5.4	Terenul este pregătit pentru noile instrumente DAQ	103
4.5.4.1	Tehnici de eșantionare utilizate în osciloscoapele numerice	104
4.5.5	DAQScope	105
4.5.5.1	De ce este importantă mărimea memoriei și viteza de transfer DMA la un osciloscop?	107
4.5.6	DAQArb	108
4.5.7	DAQMeter.....	109
4.6	Software specializat pentru achiziția datelor	110
4.6.1	Software pentru achiziția de date	111
4.6.2	Detalii privind cerințele impuse unui pachet software pentru măsurări electrice	115
4.6.3	SCPI (Standard Commands for Programmable Instrumentations). ..	119
5.	Prezentarea microcontrollerului 80C552 (PHILIPS).....	125
5.1	Arhitectura hardware a microcontroller-ului 80C552.....	125
5.1.1	Memoria internă a microcontroller-ului 80C552.....	125
5.1.1.1	Memoria de program (Program Memory)	125
5.1.1.2	Memoria de date (Data Memory).....	126
5.1.1.3	Registreele cu funcții speciale	127
5.1.2	Structura și lucrul cu porturile de intrare-ieșire	130
5.1.2.1	Programarea și utilizarea temporizatoarelor	132
5.1.2.2	Interfața serială SIO ₀	134
5.1.2.3	Ieșirile modulate în durată.....	135
5.1.2.4	Secțiunea analogică a microcontrollerului.....	137
5.1.2.5	Măsurarea intervalelor de timp prin utilizarea registrelor de captare a evenimentelor	142
5.2	Prezentarea setului de instrucțiuni al microcontroller-ului 80C51	143
5.3	Sistem de dezvoltare cu microcontroller 80C552	167
5.3.1	Domeniul de aplicabilitate	172

5.3.2 Detalierea resurselor sistemului	172
5.3.2.1 Unitatea Centrală de Prelucrare	172
5.3.2.2 Interfața cu Procesul Controlat.....	172
5.3.2.3 Interfața cu Operatorul	180
5.3.2.4 Interfața cu un Sistem de Calcul	183
5.3.3 Resurse software, utilizare	183
5.3.4 Rutine de bază pentru manipularea resurselor sistemului.....	185
6. Sistem universal, modular, de achiziții de date.....	197
6.1 Mărimi de intrare în sistemul de achiziții de date	197
6.2 Specificațiile de proiectare ale sistemului de achiziții de date	197
6.3 Interfața specializată de achiziții de date a sistemului	200
6.3.1 Interfața de achiziții de date propriu-zisă.....	201
6.3.1.1 Blocul de adaptare a semnalelor analogice	201
6.3.1.2 Blocul filtrelor antirepliare.....	204
6.3.1.3 Blocul circuitelor de eșantionare-memorare suplimentare	207
6.3.1.4 Blocul convertoarelor analog-digitale.....	209
6.3.1.4.1 Descrierea funcțională a blocului de conversie analog- digitală din cadrul interfeței specializate de achiziții de date	209
6.3.1.5 Blocul de conversie digital-analogică.....	217
6.4 Unitatea centrală de prelucrare locală cu microcontroller 80C552	221
6.4.1 Descrierea funcțională a UCPL.....	221
6.4.2 Resursele unității centrale de prelucrare locale a sistemului de achiziții de date.....	221
6.5 Interfațarea unității centrale de prelucrare, cu microcontroller 80C552, cu un sistem hardware extern (interfața de achiziții de date)	226
6.5.1 Modalități de cuplare a unității centrale de prelucrare cu un dispozitiv hardware extern	226
6.5.2 Descriere funcțională a ansamblului unitate centrală de prelucrare locală - interfața specializată de achiziții de date.....	227
6.6 Estimarea erorilor ce se manifestă în cadrul sistemului de achiziții.....	229
6.6.1 Estimarea erorilor software	229
6.6.2 Estimarea erorilor hardware	231
7. Software de analiză a semnalelor electrice	235
7.1 Considerații generale asupra instrumentelor software de analiză a semnalelor electrice.....	235
7.2 Platforma HP VEE pentru Windows. Prezentarea analizorului ESA.....	236
7.3 Implementarea analizorului ESA	239
7.3.1 Blocul de prelucrare a fișierului de date de intrare	241
7.3.2 Blocurile pentru controlul dispozitivelor de afișare.....	243
7.3.3 Dispozitivele de afișare de tip oscilograf	246
7.3.4 Blocul de afișare sub formă digitală a valorilor minime/maxime ale semnalelor de intrare	247

7.4 Interfațarea instrumentului virtual de analiză a semnalelor electrice, ESA, cu interfața specializată de achiziții de date	248
7.5 Detalii suplimentare privind implementarea instrumentului virtual ESA	250
7.6 Resurse suplimentare ale analizorului de semnale electrice, ESA	250
7.6.1 Analizorul Fourier	254
7.7 Testarea instrumentului virtual ESA și rezultate experimentale.....	257
7.8 Listingul programului de achiziție de date.....	258
8. Sistem cu microcontroller pentru măsurarea și controlul temperaturii.....	271
8.1 Specificațiile de proiectare ale sistemului pentru măsurarea și controlul temperaturii	271
8.2 Descrierea funcțională a sistemului de măsurare a temperaturii	272
8.2.1 Blocul de măsurare a temperaturii	272
8.2.1.1 Traductorul de temperatură	273
8.2.1.1.1 Conectarea traductorului de temperatură	273
8.2.1.1.2 Senzorul de temperatură.....	273
8.2.1.1.3 Modulul surselor de tensiune de referință.....	275
8.2.1.1.4 Senzor de temperatură cu ieșire unificată în curent	278
8.2.2 Utilizarea resurselor unității centrale de prelucrare	279
8.2.3 Etajele de ieșire pentru comanda elementelor de execuție	283
8.3 Software de analiză a rezultatelor	286
8.3.1 Implementarea analizorului PROTERM.....	288
8.4 Listingul aplicației de măsurare și control a temperaturii.....	289
9. Implementarea hardware-software a unui instrument de vizualizare a semnalelor (EASY SCOPE).....	299
9.1 Descrierea funcțională a osciloscopului digital EASY SCOPE	299
9.2 Prezentarea circuitului PIC16C71 (Microchip)	303
9.2.1 Descriere generală a PIC16C71	304
9.2.2 Prezentare arhitecturală	306
9.2.2.1 Ceasul de sistem / Ciclul instrucțiune	308
9.2.2.2 Fluxul de execuție al instrucțiunii / Pipeline-ing	308
9.2.3 Organizarea memoriei de program (Program Memory)	309
9.2.4 Organizarea memoriei de date (Data Memory)	310
9.2.4.1 Registrul STATUS	310
9.2.5 Porturi I/O	312
9.2.5.1 Registrele PORTA și TRISA	312
9.2.5.2 Registrele PORTB și TRISB.....	313
9.2.6 Considerații de programare I/O.....	315
9.2.6.1 Porturi I/O bidireționale	315
9.2.6.2 Operații succesive asupra porturilor I/O	316
9.2.7 Modulul de conversie analog-digitală.....	316
9.2.7.1 Specificații pentru achiziția A/D.....	319

9.2.7.2 Particularități de utilizare a circuitului de conversie analog-digitală a microcontroller-ului PIC16C71	320
9.3 Caracteristici constructive. Testare și calibrare EASY SCOPE	327
9.4 Descrierea meniului aplicației EASY SCOPE	329
9.5 Codul sursă în limbaj de asamblare al microcontroller-ului PIC16C71 pentru osciloscopul digital EASY SCOPE	331
9.6 Codul sursă în limbaj C pentru osciloscopul digital EASY SCOPE	338
10. Traductor inteligent pentru măsurarea nivelului	349
10.1 Prezentarea hardware a traductorului inteligent de nivel	350
10.2 Programul de aplicații NIV.ASM	354
10.2.1 Descrierea programului de aplicație NIV.ASM	356
10.2.1.1 Secțiunea de inițializare și declarativă	356
10.2.1.2 Secțiunea de programare a parametrilor de funcționare	358
10.2.1.3 Secțiunea de măsurare propriu-zisă	361
10.3 Programul de aplicații NIVOK.ASM	366
10.3.1 Secțiunea de inițializare și declarativă	366
10.3.2 Secțiunea de calibrare a generatorului de curent	366
10.3.3 Secțiunea de programare a parametrilor de funcționare	367
10.3.4 Secțiunea de măsurare propriu-zisă	369
10.4 Listingul programului de aplicație NIVOK.ASM	372

1. ELEMENTE INTRODUCTIVE REFERITOARE LA CONDUCEREA PROCESELOR INDUSTRIALE DIN PERSPECTIVA SISTEMELOR INTELIGENTE HARDWARE-SOFTWARE DE MĂSURARE ȘI CONTROL

1.1 INTRODUCERE

Sugestiv, conducerea proceselor industriale, poate fi reprezentată printr-o piramidă împărțită pe mai multe niveluri (fig. 1.1).

Supravegherea se găsește în “piramida conducerii proceselor” pe nivelul al treilea, alături de conducerea procesului, ceea ce arată că, practic, ele nu pot fi separate.

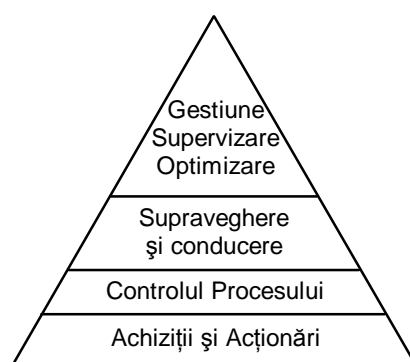


Fig. 1.1 Nivelurile de conducere a proceselor industriale.

Domeniul supravegherii proceselor industriale este destul de vast. Acesta conține aplicații începând cu simpla achiziție de date și până la prelucrări foarte complexe:

- analize statistice;
- gestiunea elaborării alarmelor;
- ghid operator;
- supravegherea acțiunilor de conducere ale operatorilor;
- identificări de parametri și simulări;
- supravegherea dinamică a răspunsului procesului, etc.

La baza “piramidei” se situează operațiunile de achiziție din proces a mărimilor de intrare și de transmitere către procesul supravegheat a comenzilor de acționare.

Funcțiile de bază ale unei aplicații de *supraveghere* a unui proces sunt:

- comunicația cu procesul;

- semnalizarea;
- comunicația cu programele utilizate pentru prelucrarea datelor;
- interfațarea om-mașină;
- gestiunea alarmelor;
- gestiunea rapoartelor.

Conceptul de *aplicație în timp real* poate fi definită astfel:

Aplicația în timp real, este acea aplicație care realizează un sistem informatic al cărui comportament este condiționat de evoluția dinamică a stării procesului la care este conectat. Acest sistem informațional este menit să urmărească sau să conducă procesul, respectând condițiile de timp stabilite. Deci, timpul real este o noțiune care marchează de fapt conceptul de timp de reacție relativ la dinamica procesului pe care sistemul informatic îl conduce (supraveghează).

Supravegherea în timp real a unui proces este o etapă necesară pentru trecerea la pasul următor: conducerea procesului.

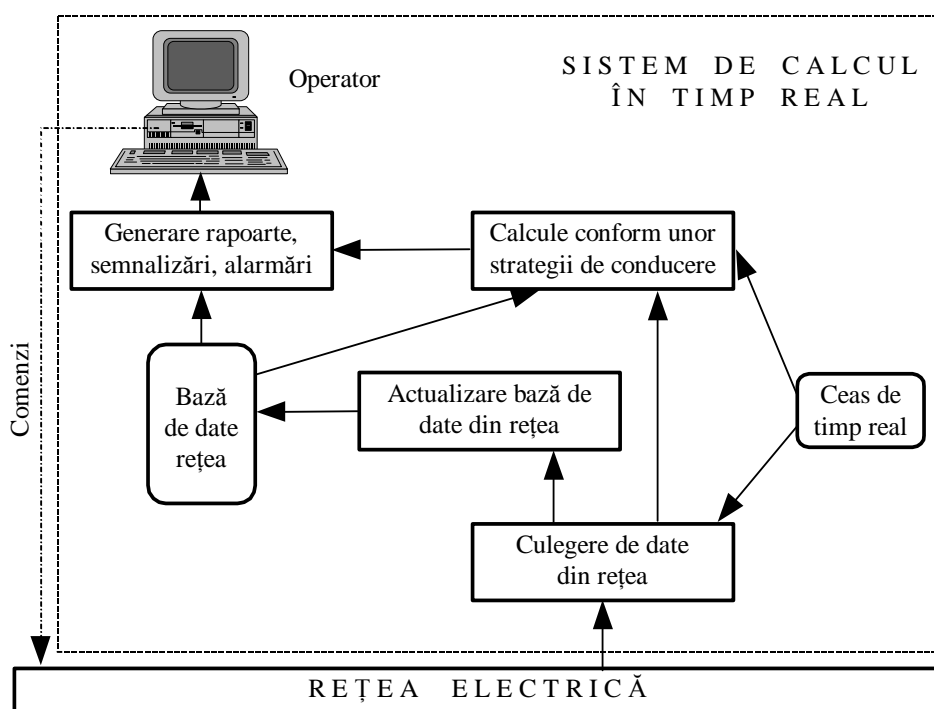


Fig. 1.2 Schema unui sistem de achiziție și calcul, în timp real, pentru supravegherea unei rețele electrice.

Un sistem în timp real este sistemul de automatizare complexă cu ajutorul calculatorului a unor probleme de decizie, mai ales cu caracter operativ, în care timpul de răspuns este suficient de redus pentru a putea influența în mod

semnificativ și pozitiv evoluția obiectivului condus.

În fig. 1.2 este prezentată schema simplificată a unui sistem de achiziție și prelucrare a datelor în timp real, destinat supravegherii proceselor dintr-o rețea electrică, care realizează:

- culegerea de date;
- actualizarea bazei de date;
- calcule conform unor strategii de conducere;
- supravegherea și corectarea *on-line* a regimului.

Un sistem de achiziție de date și control a unui proces industrial, asociat cu un microsistem de calcul, se comportă ca un sistem *intelligent* (care poate lua decizii bazate pe informații anterioare, prelucrează informația, efectuează calcule, după care, pe baza rezultatelor obținute, adoptă o decizie, din mai multe soluții posibile).

Sistemele de achiziție de date asociate cu microsistemele de calcul, în timp real, au ca principale avantaje:

- flexibilitatea și adaptabilitatea la o mare varietate de situații;
- creșterea gradului de automatizare al unor operații;
- mărirea preciziei măsurărilor;
- fiabilitate bună (număr redus de componente, posibilitatea de autotestare datorită programelor încorporate);
- miniaturizarea echipamentelor;
- posibilitatea prelucrării complexe a datelor din proces;
- simplificarea proiectării electrice și tehnologice datorită existenței familiilor de componente ce permit interconectări standard.

2. SISTEME DE ACHIZIȚIE ȘI PRELUCRARE A DATELOR

2.1 NOȚIUNI GENERALE

Ca rezultat al răspândirii pe scară largă, în ultimul timp, a calculatoarelor personale și a perfecționării lor continue, marile firme producătoare de sisteme de măsurare au căutat să realizeze echipamente care să utilizeze calculatorul personal pentru:

- achiziția de date din sistemele industriale;
- reglajul și supravegherea unor parametri sau instalații (proces);
- realizarea unor aparate de măsurare cu performanțe ridicate.

În prezent, resursele calculatorului personal sunt utilizate pentru a efectua sarcini cum ar fi: comanda, gestiunea, prelucrarea și afișarea datelor care altfel ar fi preluate de un microprocesor, plasat în interiorul instrumentului.

Instrumentul de măsurare comunică cu PC-ul prin intermediul unei interfețe care are în mod obligatoriu un convertor analog-digital. Instrumentul de măsurare poate fi redus la o simplă cartelă de achiziții de date pentru măsurători.

În momentul de față, prin intermediul tastaturii calculatorului se poate comanda instrumentul de măsurare, iar pe display pot fi vizualizate rezultatele măsurătorilor, sub formă numerică sau sub formă grafică.

Aceste rezultate apar ca urmare a prelucrării datelor brute obținute de la instrumentul de măsurare de către calculator, la cererea utilizatorului. De aici, rezultă aparate cu preț de cost mult mai scăzut.

2.2 SISTEME DE ACHIZIȚII DE DATE. ARHITECTURĂ. PRINCIPALELE TIPURI DE RESURSE UTILIZATE ÎN CADRUL SISTEMELOR DE ACHIZIȚII DE DATE

Un sistem de achiziție de date cu n canale de intrare poate fi realizat în următoarele trei configurații:

- *sistem cu multiplexare temporală;*
- *sistem cu achiziție sincronă de date;*
- *sistem rapid de achiziție de date.*

Un sistem de achiziție de date utilizat pentru achiziția și prelucrarea datelor într-un sistem (fig. 2.1) este compus din următoarele module funcționale principale:

1. convertoare de intrare;

2. circuite de multiplexare analogică;
3. circuite de eșantionare-memorare (E/M);
4. circuite pentru conversia datelor - convertoare analog-digitale (CA/D) și digital-analogice (CD/A);
5. registre tampon (buffer-e);
6. unitatea centrală de prelucrare (μP);
7. interfața de interconectare cu calculatorul personal.

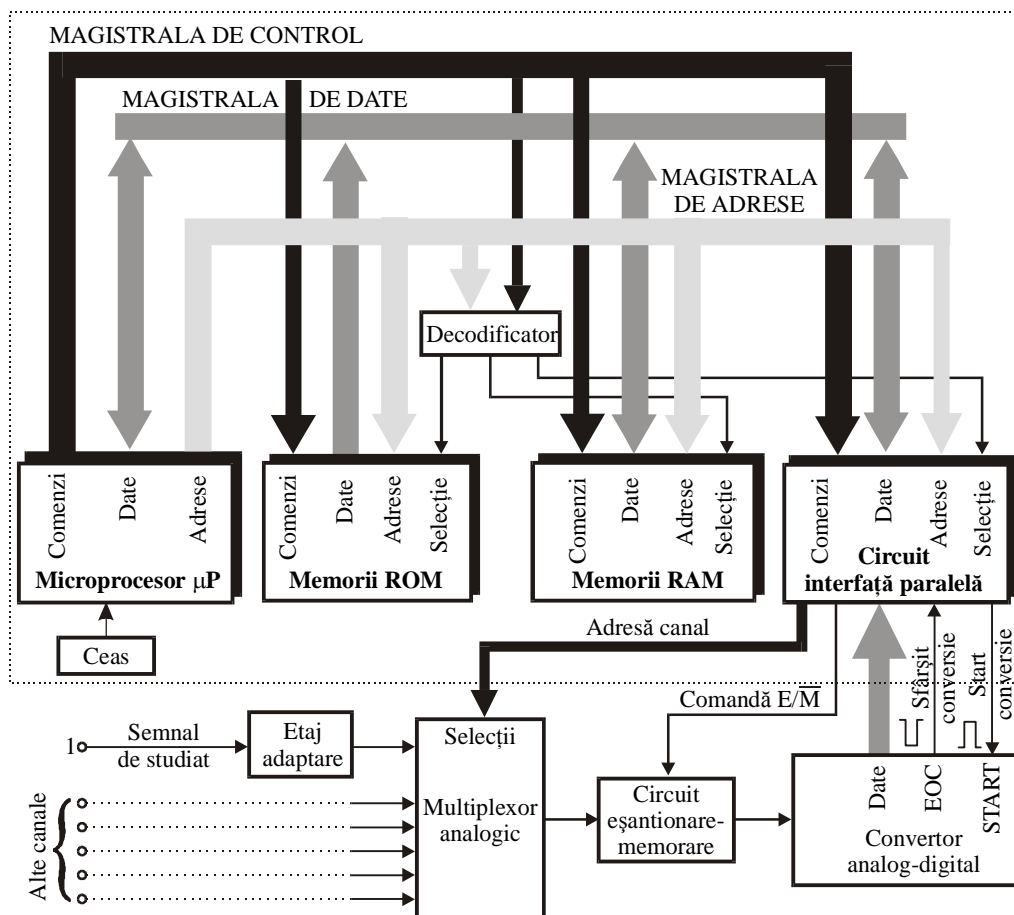


Fig. 2.1 Structura generală a unui sistem de achiziții de date.

În continuare vor fi prezentate aspectele esențiale, parametrii caracteristici și vor fi enumerate recomandări de proiectare ale acestor componente de bază din cadrul sistemelor de achiziții de date.

2.2.1 MULTIPLEXOARE ANALOGICE UTILIZATE ÎN SISTEME DE ACHIZIȚII DE DATE

În multe situații este necesar să fie transmise mai multe informații pe același canal; cum acest lucru nu se poate face simultan, se recurge la o partajare în timp a canalului, denumită *multiplexare*. Operația inversă se numește

demultiplexare. Operația de multiplexare/demultiplexare analogică necesită dispozitive de comutare care să direcționeze semnalul util pe un canal dorit. În varianta sa cea mai simplă, un multiplexor analogic poate fi asimilat cu un comutator rotativ cu $k = 2^n$ poziții sau cu un ansamblu de $k = 2^n$ comutatoare, dintre care numai unul este închis, în timp ce toate celelalte sunt deschise, comandat de un sistem logic care permite cuplarea uneia din intrări la ieșire (fig. 2.2). Deoarece comutatoarele sunt bilaterale, rezultă că un multiplexor analogic poate fi utilizat și ca demultiplexor analogic, prin simpla schimbare a sensului.

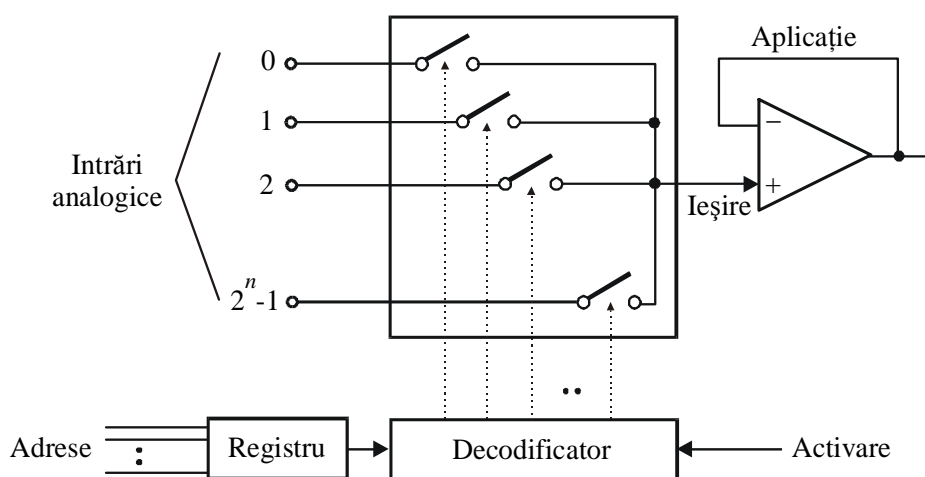


Fig. 2.2 Structura unui multiplexor analogic.

Parametrii multiplexoarelor/demultiplexoarelor analogice sunt:

- **rezistența în starea deschis (off):** R_{off} [M Ω];
- **rezistența în starea închis (on):** R_{on} [Ω];
- **curentul de pierderi în starea deschis:** I_{off} [nA, μ A, mA];
- **timpul de comutare directă (închidere):** t_{on} [ns, μ s]. Este definit ca intervalul de timp de la aplicarea comenzii de închidere până ce semnalul de ieșire atinge o valoare egală cu cea de la intrare (cu o precizie impusă, de exemplu 1%);
- **timpul de comutare inversă (deschidere):** t_{off} [ns, μ s]. Este definit ca intervalul de timp de la aplicarea comenzii de deschidere până la reducerea curentului la valoarea curentului de pierderi, I_{off} , la valoarea specificată în catalog;
- **banda de frecvențe:** B.

Multiplexorul analogic permite utilizarea unui singur convertor analog-digital pentru mai multe canale analogice de intrare (sisteme de achiziții de date cu multiplexare temporală). Utilizarea multiplexoarelor reprezintă o soluție economic viabilă și în cazul semnalelor de intrare de nivel redus, pentru care

multiplexarea se realizează cu costuri ridicate.

Elementul principal al multiplexoarelor analogice îl constituie elementul de comutare, care poate fi realizat în mai multe variante constructive:

- cu relee obișnuite;
- cu relee cu mercur;
- cu relee *reed*;
- cu elemente semiconductoare (tranzistoare bipolare, diode Schottky, tranzistoare TEC-J, tranzistoare CMOS).

Primele trei variante constructive, utilizând elemente electromecanice, conduc la investiții inițiale reduse, compensate însă de costuri ridicate de exploatare, fiabilitate și durată de funcționare reduse. De aceea, utilizarea lor este recomandabilă doar în situațiile în care este nevoie să fie multiplexate semnale cu nivele mari.

Fiecare tip constructiv de multiplexoare analogice, realizat cu elemente semiconductoare, sunt caracterizate de unele performanțe notabile, dar și de inconveniente mai mult sau mai puțin surmontabile. Astfel:

- comutatoarele cu diode rapide au timp de comutație de valori foarte reduse (≤ 1 ns), însă rezistențele reziduale (în stare închisă, respectiv deschisă) R_{on} și R_{off} au valori neperformante, în comparație cu alte tipuri;
- comutatoarele cu tranzistoare bipolare au timpi de comutație mici și rezistențe reziduale R_{on} de valori reduse, necesită curenți de comandă importanți, dar R_{off} are o valoare relativ mică, ceea ce conduce la o “transparență” mare a comutatorului;
- comutatoarele cu tranzistoare cu efect de câmp TEC-J au rezistența R_{on} de ordinul zecilor de ohmi, timpi de comutație medii, însă necesită circuite de comandă complicate (translatoare de nivel pentru compatibilizarea comenzilor);
- comutatoarele cu tranzistoare complementare CMOS sunt cele mai avantajoase și cele mai folosite. Ele sunt caracterizate prin timpi de comutație satisfăcători, rezistența R_{on} de valoare relativ mică și R_{off} de valoare ridicată. În același timp ele pot fi comandate foarte simplu, iar “transparența” crește doar la frecvențe înalte ($10^5 \div 10^8$ Hz).

În prezent, datorită evoluției explozive a tehnologiei dispozitivelor semiconductoare CMOS, au fost realizate multiplexoare analogice ce pot fi direct interfațate cu un microprocesor. Acestea dispun de un registru ce poate memora adresa de canal prin executarea unei instrucțiuni de scriere la adresa specifică alocată multiplexorului. De asemenea, majoritatea multiplexoarelor analogice realizate în tehnologie CMOS sunt caracterizate de protecția dispozitivului la aplicarea unor supratensiuni pe intrări, cu valori de 5-6 ori mai mari decât semnalele manipulate în funcționare normală. Protecțiile sunt active

pentru canale în stare *on* sau *off* în cazul dispozitivelor în stare de funcționare (alimentate), și chiar pentru circuite nealimentate. Un canal deschis, căruia i se aplică o supratensiune, este comutat automat în stare *off*, realizând protecția etajelor electronice conectate la ieșirea multiplexorului. Un exemplu tipic de astfel de multiplexor analogic interfațabil și cu protecție la aplicarea de supratensiuni accidentale pe intrări este circuitul MAX368, produs de firma Maxim.

2.2.2 CIRCUITE DE EȘANTIONARE-MEMORARE UTILIZATE ÎN SISTEME DE ACHIZIȚII DE DATE

Un circuit de eșantionare-memorare realizează prelevarea, la un moment dat, a valorii unui semnal analogic și memorarea analogică a acesteia (fig. 2.3a).

În modul de lucru “eșantionare” (sau *urmărire*), determinat de nivelul logic “1” al semnalului de comandă E/\overline{M} , circuitul de eșantionare-memorare funcționează ca repetor. În modul de lucru “memorare” (sau *menținere*), determinat de nivelul logic “0” al semnalului de comandă E/\overline{M} , circuitul de eșantionare-memorare funcționează ca o *memorie analogică*, memorând la bornele unei capacități semnalul de intrare eșantionat anterior (fig. 2.3b).

Circuitele de eșantionare-memorare se utilizează în sistemele de achiziție și distribuție de date. Astfel, într-un sistem de achiziții de date, ieșirea circuitului de eșantionare-memorare este conectată la intrarea convertorului analog-digital (CA/D). În intervalul corespunzător efectuării unei conversii analog-numerice, circuitul de eșantionare-memorare este comandat în stare de memorare pentru a menține constantă tensiunea la intrarea convertorului analog-digital. Se obține astfel mărirea valorii limitei superioare a domeniului de frecvențe ale semnalului de intrare pentru care CA/D poate fi utilizat la rezoluția maximă (specificată de numărul de biți ai rezultatului conversiei). Acest deziderat este realizat dacă tensiunea de la intrarea convertorului analog-digital nu se modifică, pe durata efectuării conversiei, cu mai mult de $\pm 1/2$ LSB. În sistemele de distribuție a datelor, circuitele de eșantionare-memorare sunt utilizate pentru reconstituirea semnalelor multiplexate în timp.

Circuitele de eșantionare-memorare sunt caracterizate de o serie de parametri (fig. 2.4), grupați în mai multe caracteristici:

- **caracteristici de urmărire** (fig. 2.4a):
 - *eroarea staționară* - reprezintă abaterea de la amplificarea unitară sau de la cea specificată prin datele de catalog;

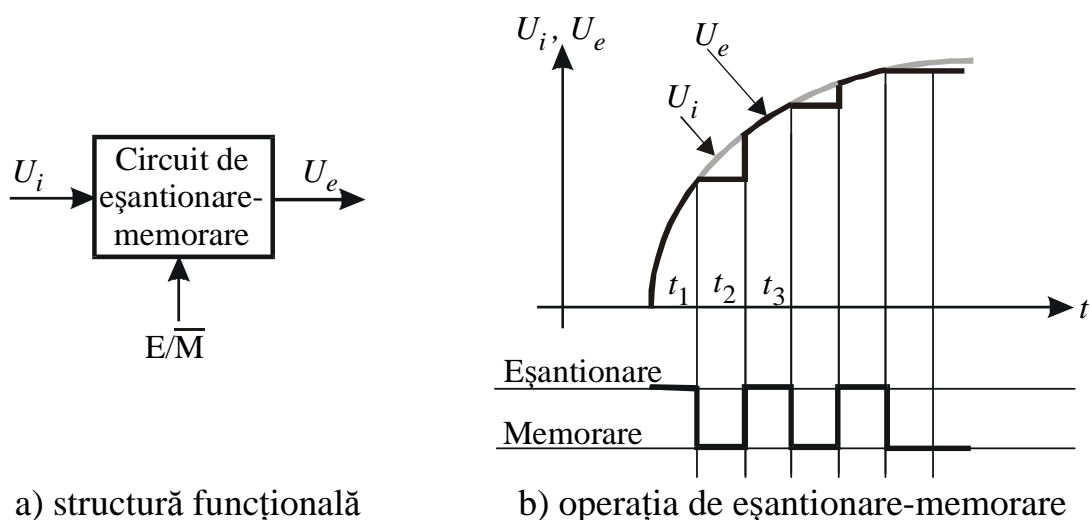


Fig. 2.3 Circuit de eșantionare-memorare.

- *eroarea de decalaj* - reprezintă valoarea ieșirii pentru o tensiune de intrare nulă;

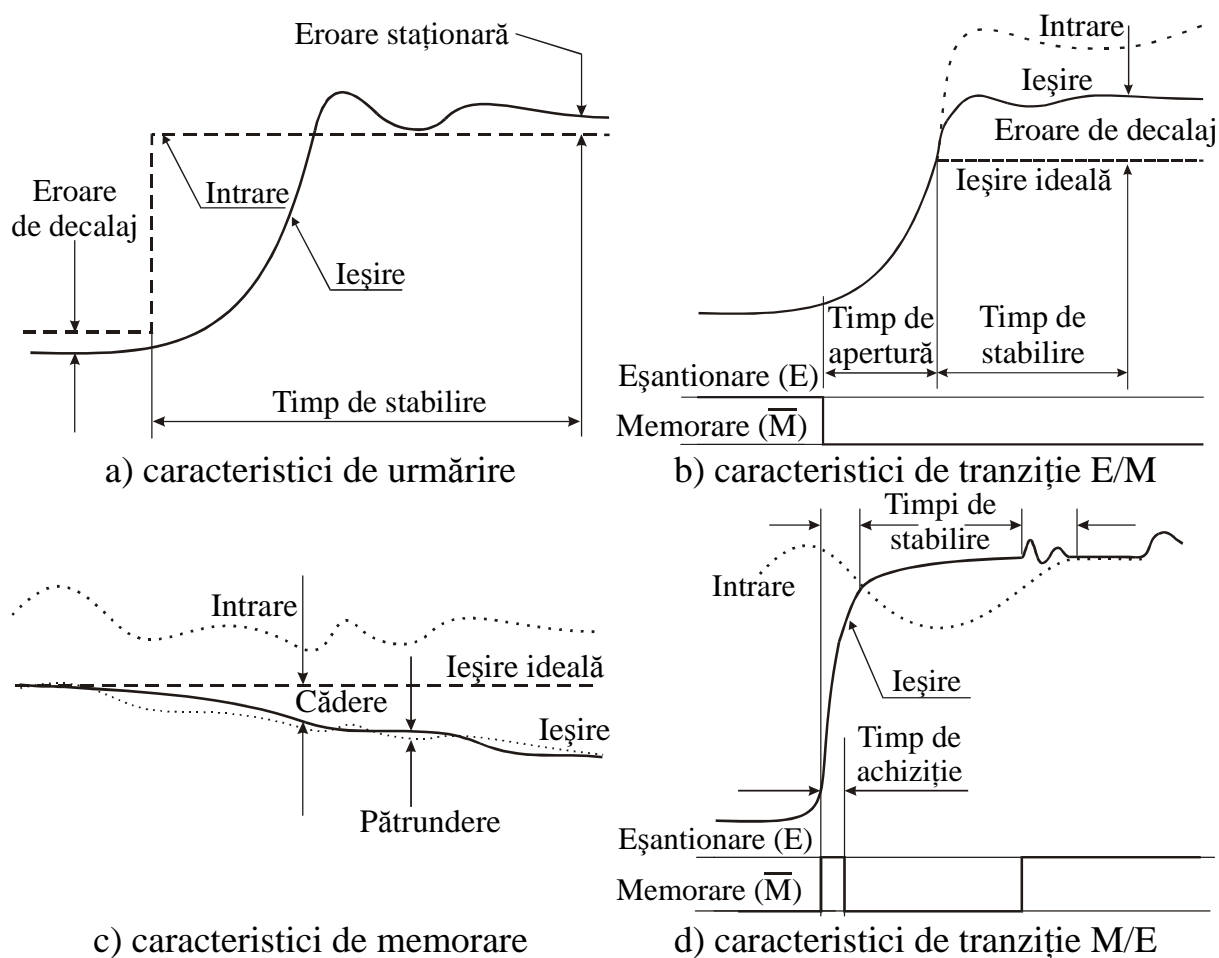


Fig. 2.4 Erori ale circuitelor de eșantionare-memorare.

- *timpul de stabilire* - reprezintă intervalul de timp necesar pentru

atingerea valorii dorite a ieșirii, cu o toleranță maximă specificată;

- **caracteristici de tranziție eșantionare-memorare** (fig. 2.4b):
 - *timpul de apertură* - reprezintă intervalul de timp dintre comanda de memorare și momentul efectiv al comutării circuitului în regim de memorare;
 - *incertitudinea timpului de apertură* - reprezintă variația timpului de deschidere a comutatorului regimului de eșantionare-memorare, după primirea comenzii de memorare;
 - *eroarea de decalaj la memorare* - este determinată, în principal, de comutarea târzie a circuitului de memorare și a regimului tranzitoriu de încărcare a condensatorului de memorare;
- **caracteristici de memorare** (fig. 2.4c):
 - *căderea* - reprezintă tendința de scădere a nivelului de la ieșire față de cel ideal, datorită descărcării condensatorului de memorare;
 - *pătrunderea* - caracterizează influența intrării asupra ieșirii, datorată imperfecțiunilor circuitelor de comutare analogică;
- **caracteristici de comutare memorare-eșantionare** (fig. 2.4d):
 - *timpul de achiziție* - reprezintă intervalul minim necesar de eșantionare, pentru a se obține o tensiune de ieșire dorită, egală cu semnalul aplicat la intrare cu o toleranță dată. Acest parametru depinde aproape liniar de valoarea capacității de memorare;
 - *timpul de stabilire la tranziția memorare-eșantionare* - reprezintă intervalul de timp dintre comutarea propriu-zisă și atingerea unei valori a ieșirii corespunzătoare intrării, cu o toleranță maximă specificată.

Uzual, în cadrul sistemelor de achiziții de date sunt utilizate:

- circuite de eșantionare-memorare în buclă de reacție;
- circuite de eșantionare-memorare cu integrare.

În timpul operației de eșantionare, bucla de reacție negativă din fig. 2.5 permite eliminarea erorii de mod comun și a erorii de offset, ieșirea fiind forțată să urmărească intrarea. Ca efect, tensiunea la bornele capacității de memorare C , pe durata cât comutatorul K este închis, este egală cu:

$$U_e = U_i \cdot \frac{A}{A-1} \quad (2.1)$$

în care A reprezintă amplificarea în buclă deschisă a amplificatorului operațional $A1$ (de valoare foarte mare). Se obține astfel egalitatea între U_e și U_i .

Precizia ridicată este obținută în detrimentul rapidității, deoarece pe durata regimului de memorare amplificatorul $A1$ este saturat, întoarcerea la funcționarea liniară, pentru operația de eșantionare, determină creșterea timpului de achiziție, care poate atinge mai multe zeci de μs .

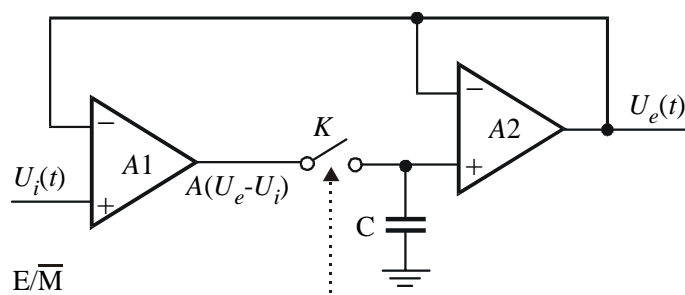


Fig. 2.5 Circuit de eșantionare-memorare cu buclă de reacție negativă.

În fig. 2.6 sunt prezentate două variante de circuit de eșantionare-memorare cu integrare. În montajul din fig. 2.6a capacitatea de memorare C este izolată în raport cu masa circuitului, iar comutatorul K funcționează în comutație de curent, comanda fiind simplificată. Ca și în cazul precedent, primul amplificator este saturat pe durata regimului de memorare. Evitarea saturării ieșirii amplificatorului $A1$ este ilustrată în schema din fig. 2.6b.

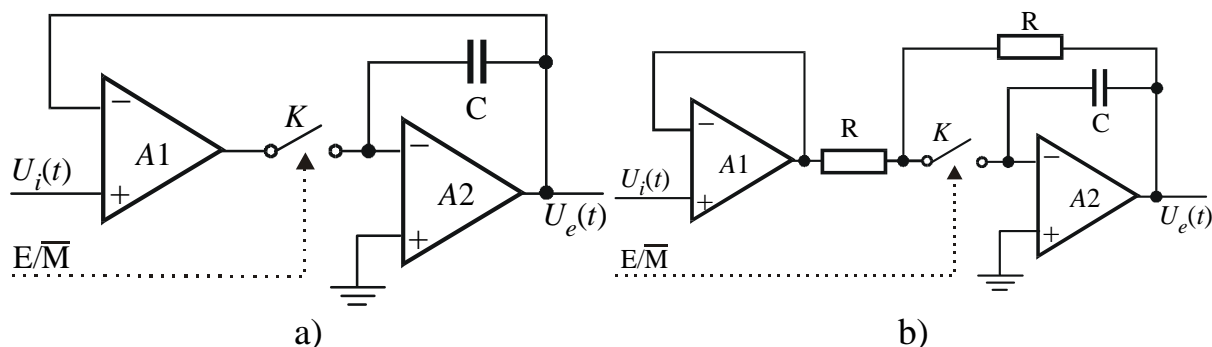


Fig. 2.6 Circuite de eșantionare-memorare cu integrare.

În tabelul 2.1 sunt prezentate câteva tipuri de circuite de eșantionare și memorare, produse de firme cum ar fi: Analog Devices, National, Burr-Brown și Datel-Intersil.

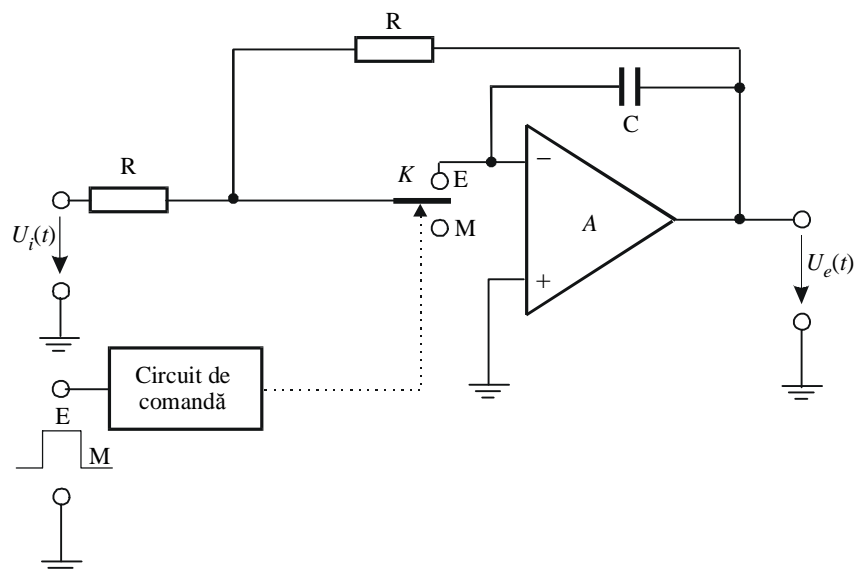
Circuitele de eșantionare-memorare disponibile la momentul actual acoperă o paletă largă și diversă din punct de vedere al performanțelor, la cele două extreme aflându-se, pe de o parte, circuitele de eșantionare-memorare rapide, dar cu o exactitate scăzută, respectiv cele lente, dar cu exactitate bună în ceea ce privește deriva, decalajul etc.

Aplicațiile ce necesită utilizarea circuitelor de eșantionare-memorare acoperă și ele o paletă largă de frecvențe și viteze de variație a semnalelor de eșantionat. În general, semnalele rapid variabile nu necesită o precizie deosebită, din această cauză, în regim tranzitoriu, viteza constituie parametrul principal, ceea ce înseamnă timpi de achiziție și de stabilire mici. O situație mai dificilă este atunci când se cere o viteză ridicată de eșantionare și, în același timp, o precizie bună.

Tabelul 2.1 Principalele caracteristici ale unor circuite de eșantionare și memorare

Tipul	Timpul de achiziție	Precizia	Timpul de apertură	Timpul de stabilire	Tehnologie; Particularități
AD582	6 μ s 25 μ s	0,10 % 0,01 %	150 ns	0,5 μ s	monolitică, uz comun
AD583	4 μ s 5 μ s	0,10 % 0,01 %	50 ns	-	monolitică, rapidă
LF398	4 μ s 6 μ s	0,10 % 0,01 %	150 ns	0,8 μ s	monolitică, uz comun
SHC298	9 μ s 10 μ s	0,10 % 0,01 %	200 ns	1,5 μ s	monolitică, uz comun
AD346	2 μ s	0,01 %	60 ns	0,5 μ s	hibridă, condensator de memorare intern
SHC85	4 μ s	0,01 %	25 ns	0,5 μ s	hibridă, condensator de memorare intern, timp ridicat de reținere a tensiunii
HTS0025	20 ns	0,01 %	20 ns	30 ns	hibridă, extrem de rapidă

Pentru semnale caracterizate printr-o viteză de variație mai scăzută, se aleg circuite de eșantionare-memorare cu performanțe satisfăcătoare de viteză, dar cu performanțe bune în ceea ce privește dispersia la deschidere, deriva de zero și rata de descărcare a condensatorului de memorare.

**Fig. 2.7** Circuit de eșantionare memorare cu blocare.

Pentru memorarea valorilor semnalelor, în vederea conversiei analog-digitale, cea mai des utilizată metodă de eșantionare este eșantionarea prin blocare. În această metodă, valoarea semnalului este memorată din primul

moment al eșantionării.

Un exemplu de circuit care utilizează această metodă de eșantionare, este circuitul de eșantionare-memorare cu integrare, varianta inversoare, a cărei schemă funcțională este prezentată în fig. 2.7.

2.2.3 CIRCUITE PENTRU CONVERSIA DATELOR UTILIZATE ÎN SISTEME DE ACHIZIȚII DE DATE: CONVERTOARE ANALOG-DIGITALE ȘI DIGITAL-ANALOGICE

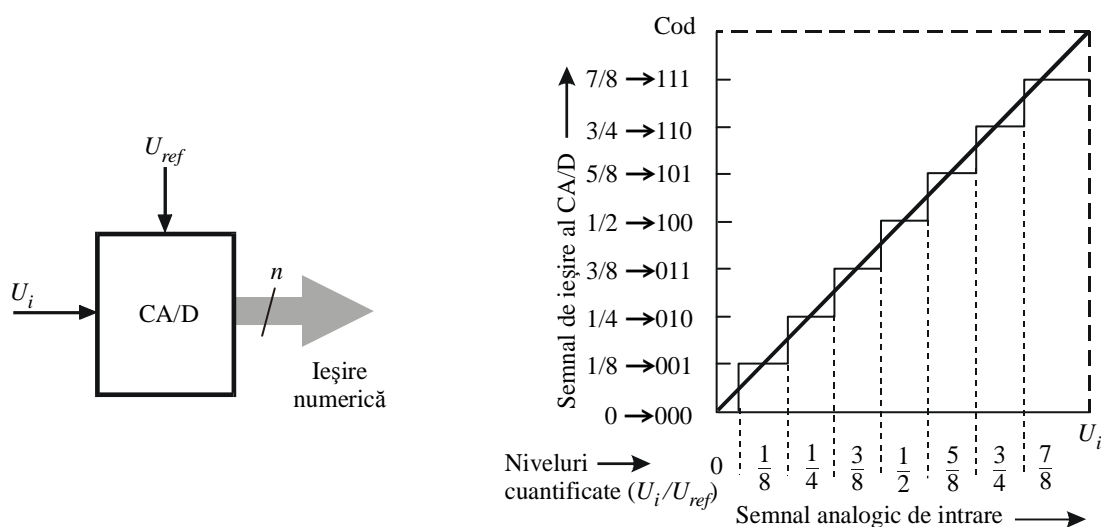
Conversia datelor reprezintă principala operație realizată în cadrul sistemelor de achiziție și reprezintă transformarea semnalelor din formă analogică în formă digitală sau invers.

Convertorul analog-digital reprezintă componenta principală a oricărui sistem de achiziții de date. Acesta realizează transformarea tensiunii analogice de la intrare într-un cod numeric binar (fig. 2.8a).

Acest rezultat reprezintă cea mai bună aproximație numerică a tensiunii de la intrare. Măsura acestei aproximații este reprezentată de numărul de biți ai rezultatului conversiei.

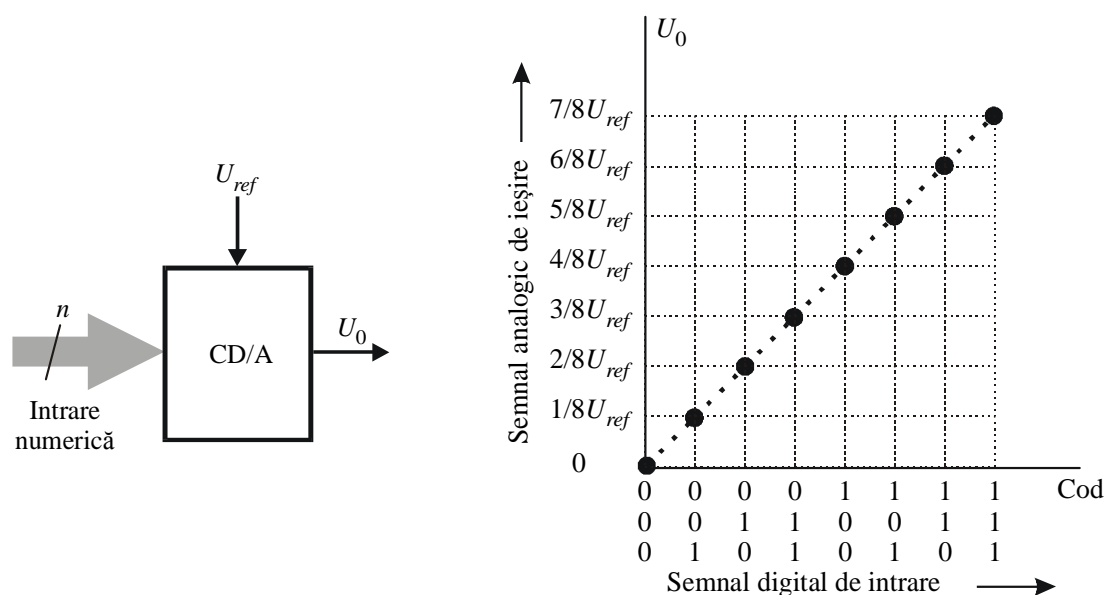
Într-un sens mai larg, procesul de conversie analog-digitală poate fi considerat ca o plasare a mărimii de intrare într-un *interval de cuantizare*, obținut prin divizarea intervalului de variație a acesteia într-un număr de clase egale.

Atunci când mărimea exprimată numeric la intrare este transformată în mărime analogică la ieșire se realizează o conversie digital-analogică (fig. 2.8c).



a) structura funcțională a CA/D

b) caracteristica de transfer ideală a CA/D



c) structura funcțională a CD/A d) caracteristica de transfer ideală a CD/A

Fig. 2.8 Convertoare analog-digitale și digital analogice: reprezentare funcțională și caracteristică ideală de transfer.

Circuitele de conversie a datelor utilizate în cadrul sistemelor de achiziții de date sunt caracterizate printr-o serie de parametri, cum ar fi:

- **gama de variație a intrării** (pentru CA/D) sau **a ieșirii** (pentru CD/A) (*domeniul de lucru*), reprezentând domeniul maxim de variație a mărimii analogice (de obicei tensiune) și exprimată în unități absolute (V, mV, mA) sau relative (dB);
- **caracteristica de transfer**, reprezentând dependența mărimii de la ieșirea convertorului față de mărimea de intrare; pentru un convertor analog-digital caracteristica de transfer ideală este o funcție scară (fig. 2.8b) iar pentru un convertor digital-analogic este un set de puncte dispuse pe o dreaptă (fig. 2.8d);
- **rezoluția** reprezintă numărul total de coduri distincte de ieșire ale convertorului analog-digital, respectiv numărul total de nivele de ieșire pentru un convertor digital-analogic. Uzual, rezoluția se exprimă în biți, în procente din valoarea domeniului de lucru, sau în număr de nivele de cuantificare (CA/D) sau de ieșire (CD/A). Rezoluția teoretică a unui convertor de N biți este 2^N ; rezoluția reală poate fi însă mai mică, datorită erorilor. Acest parametru important al convertoarelor se determină ca reprezentând valoarea variației minime a mărimii de intrare ce provoacă modificarea a două coduri consecutive de ieșire (CA/D), respectiv valoarea variației minime a mărimii analogice de la ieșire (CD/A). Rezoluția poate fi prezentată ca fiind $\frac{1}{2^N}$ din domeniul de lucru.

Acest parametru nu trebuie considerat ca o performanță specifică a convertorului, ci un parametru de proiectare. Plecând de la o aplicație concretă, pentru care se impune prelevarea unei mărimi cu o precizie dată, se poate determina rezoluția minimă a convertorului ce va fi folosit;

- **timpul de stabilire** caracterizează viteza de răspuns a circuitului și reprezintă timpul scurs între aplicarea unui semnal de intrare de tip treaptă ideală și până la obținerea ieșirii dorite cu o aproximație specificată (de regulă $\pm 1/2$ LSB) (fig. 2.9). Timpul de stabilire include mai multe intervale de timp specifice, cum ar fi: *timpul de propagare* t_p (până la începerea unui efect observabil la ieșire), *timpul de creștere* t_c (până la prima atingere a nivelului de ieșire dorit), *timpul de restabilire* t_r (după supracreșterea ieșirii) și *timpul de relaxare liniară* t_a (amortizarea eventualului răspuns oscilant). Este un parametru specific convertoarelor digital-analogice și se exprimă în unități de timp, indicând și limitele intervalului de aproximație în jurul ieșirii specificate;
- **timpul de conversie** , t_{CONV} , reprezintă intervalul de timp necesar unui convertor să obțină mărimea de ieșire pornind de la o mărime de intrare dată (timpul necesar obținerii codului numeric de ieșire corespunzător mărimii analogice de intrare). Variația tensiunii de intrare, pe parcursul procesului de conversie, introduce o eroare în valoarea semnalului de ieșire. În cazul convertoarelor digital-analogice acest timp poate fi considerat a fi chiar timpul de stabilire;
- **timpul de revenire (relaxare)** , t_{rev} , reprezintă timpul necesar unui convertor pentru a putea opera din nou corect;
- **rata de conversie** este o măsură a vitezei convertorului și este definită de inversul sumei timpilor de conversie și de revenire:

$$R_{\text{CONV}} = \frac{1}{t_{\text{CONV}} + t_{\text{rev}}} \quad (2.2)$$

În majoritatea situațiilor, timpul de revenire este mult mai mic decât timpul de conversie, astfel încât rata de conversie poate fi aproximată doar ca invers al timpului de conversie. În cazul convertoarelor rapide și foarte rapide, timpul de revenire trebuie luat în calcul pentru estimarea ratei de conversie;

- **timpul de conversie pe bit** este timpul echivalent de generare a unui bit (parametru caracteristic pentru convertoare analog-digitale secvențiale);
- **viteza de variație a ieșirii (SR Slew-Rate)** a unui convertor D/A reprezintă o caracterizare a intervalului de timp necesar ieșirii să

execute excursia maximă în cadrul domeniului de variație.

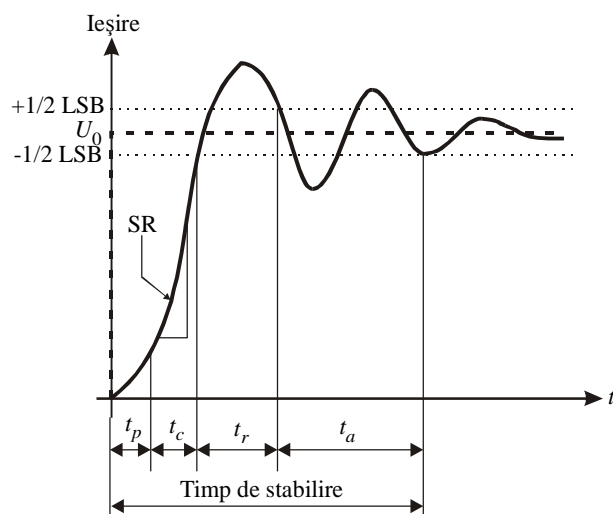


Fig. 2.9 Timpul de stabilire.

Conversia analog-digitală este caracterizată în sine prin *eroarea de cuantizare*. Datorită formei caracteristicii de transfer (în scară), a codificării unice a unui întreg interval de cuantizare, apare o incertitudine de $\pm 1/2$ LSB, nulă la mijlocul intervalului și maximă la ambele capete. Influența erorii de cuantizare poate fi diminuată prin mărirea numărului de biți ai codului de ieșire a convertorului.

Fiecare cuantă (mărime a intervalului) a unei astfel de divizări reprezintă o valoare a mărimii analogice, pe care se disting nivelurile semnalului de intrare, prezentate prin două combinații de coduri învecinate. Această cuantă poartă denumirea de bitul cel mai puțin semnificativ (LSB).

Astfel:

$$q = \text{LSB} = \frac{U_{i \max}}{2^N} \quad (2.3)$$

unde: q este cuanta, iar $U_{i \max}$ gama de variație a semnalului analogic de intrare.

Caracteristicile reale ale circuitelor de conversie a datelor pot diferi de caracteristicile sale ideale (fig. 2.10). Caracteristica de transfer a convertorului analog-digital poate fi translatată în raport cu cea ideală (fig 2.10a). Această eroare se numește *eroare de decalaj (offset)* și se poate pune în evidență aplicând la intrare o mărime nulă și măsurând ieșirea.

Eroarea determinată de modificarea pantei caracteristicii de transfer reale față de cea ideală, eroarea inițială fiind nulă, se numește *eroare de amplificare (de gamă)* (fig. 2.10b). Pentru majoritatea CA/D erorile de decalaj și de amplificare sunt mici și pot fi complet eliminate prin reglaj prealabil.

Mai dificil de eliminat sunt erorile legate de neliniaritatea caracteristicilor de transfer, care nu pot fi înlăturate prin reglare prealabilă. Convertoarele analog-digitale sunt caracterizate de două tipuri de neliniarități: cea integrală, respectiv cea diferențială:

- *neliniaritatea integrală* definește gradul în care caracteristica de transfer a unui convertor se abate de la forma teoretică (ideală) de dreaptă, considerând erori de decalaj și de amplificare nule (fig. 2.10c);

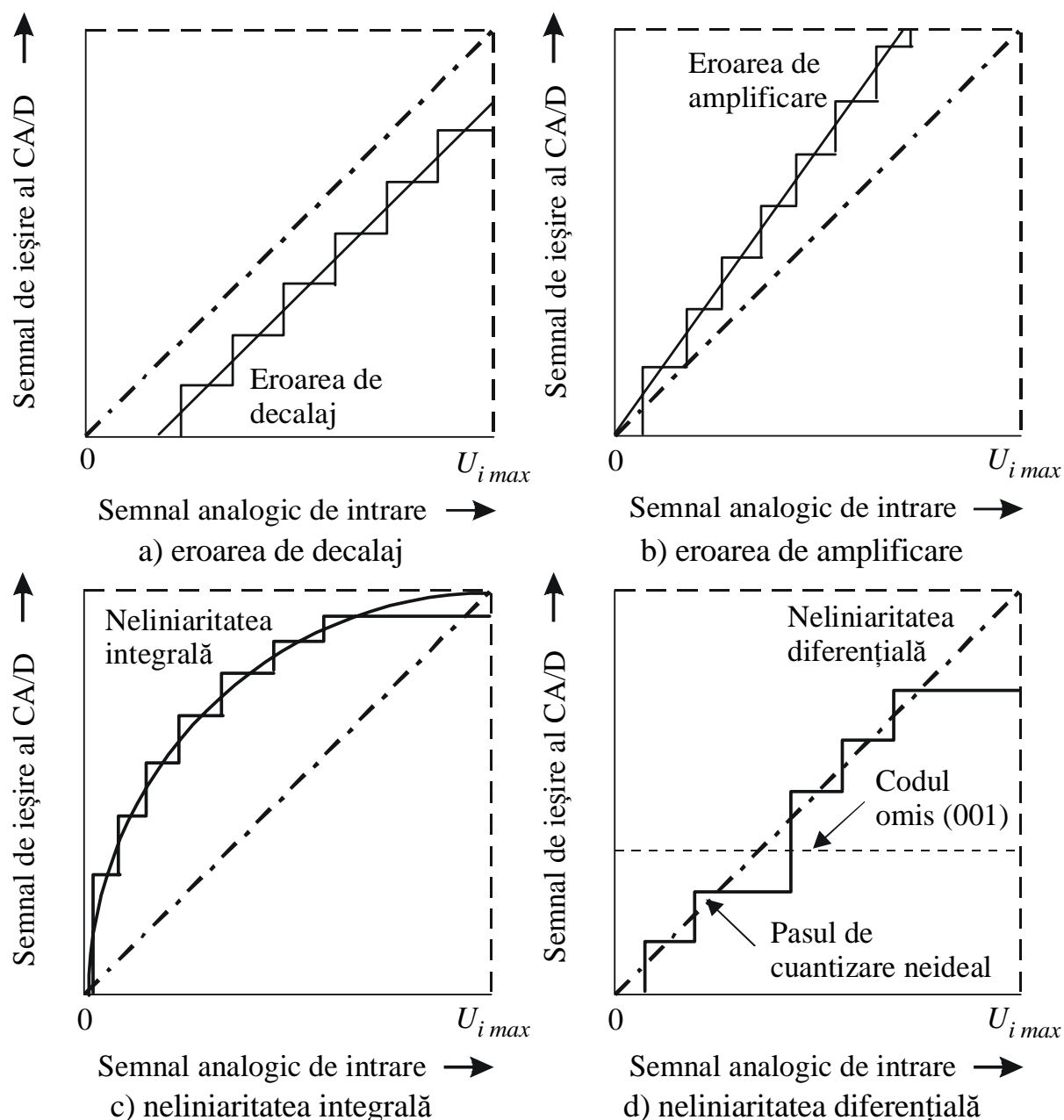


Fig. 2.10 Erorile convertoarelor analog-digitale.

- *neliniaritatea diferențială* caracterizează uniformitatea intervalelor de cuantizare ale unui convertor analog-digital. Dacă neliniaritatea diferențială depășește 1 LSB, aceasta conduce la o comportare nemonotonă a caracteristicii de transfer (în semnalul numeric de ieșire poate lipsi una din combinațiile de cod - fig 2.10d). Neliniaritatea diferențială este afectată de metoda de conversie; ea tinde să fie maximă atunci când convertorul trece prin toate intervalele de cuantizare secvențial.

Precizia reprezintă capacitatea circuitelor de conversie de a respecta cu strictețe caracteristica de transfer ideală, reflectând capacitatea convertoarelor de

a nu fi afectate de erori sistematice și aleatoare. **Precizia absolută** caracterizează funcționarea unui convertor în ansamblu, reflectând orice anomalie a caracteristicii de transfer. **Precizia relativă** este influențată doar de liniaritatea caracteristicii de transfer.

Precizia totală de conversie se realizează numai în cazul când această eroare nu depășește rezoluția convertorului. Astfel, pentru un convertor cu rezoluția de N biți, caracterizat de timpul de conversie t_{CONV} , este necesară îndeplinirea următoarei condiții:

$$\left(\frac{dU_i}{dt} \right)_{\text{max}} \leq \frac{U_{i \text{ max}}}{2^N \cdot t_{\text{CONV}}} \quad (2.4)$$

2.2.3.1 CONVERTOARE DIGITAL-ANALOGICE. SCHEME DE PRINCIPIU

Conversia datelor presupune ca oricărei mărimi analogice să i se asocieze o reprezentare numerică corespunzătoare; codurile utilizate pot fi ponderate sau neponderate, prezentând avantajul unei exprimări naturale și compatibilitate cu circuitele de calcul numeric. În cazul unui cod ponderat, o cifră din cadrul unui număr are semnificația valorii sale propriu-zise, cât și a ponderii datorate poziției sale în cadrul numărului. Conversia digital-analogică presupune transformarea valorii și ponderii cifrelor numărului într-o mărime de ieșire analogică corespunzătoare (tensiune sau curent).

Considerând un număr întreg binar de N biți, de forma:

$$\overline{B_{N-1}B_{N-2}...B_1B_0} = \sum_{i=0}^{N-1} B_i \cdot 2^i \quad (2.5)$$

ponderea cifrei B_{i-1} (ce ocupă poziția i începând cu LSB) este 2^{i-1} ; așadar ponderea sa crește de la dreapta spre stânga de la valoarea 1 (ponderea LSB) la valoarea 2^{N-1} (ponderea MSB). Aceleași observații sunt valabile și pentru un număr subunitar de N biți, de forma:

$$\overline{B_1B_2...B_{N-1}B_N} = \sum_{i=1}^N B_i \cdot 2^{-i} \quad (2.6)$$

Procesul de conversie digital-analogică poate fi considerat similar cu procesul de transformare a unui număr din sistemul de numerație binar în sistemul de numerație zecimal: se asociază fiecărei cifre binare “1” o anumită valoare a unei mărimi electrice care se însumează ponderat conform rangului pe care îl ocupă în cadrul reprezentării numerice. Deoarece ponderea cifrelor descrește cu factori de forma 2^{-i} , o soluție simplă pentru realizarea operației de ponderare ar consta în utilizarea unor rețele rezistive divizoare, cu mai multe noduri, având între noduri consecutive un raport de divizare de $1/2$. Majoritatea

convertoarelor digital-analogice moderne folosesc scheme cu sumare de curenți, care sunt mai stabile, mai rapide și mai ușor de realizat. Schema bloc a unui astfel de convertor este prezentată în fig. 2.11.

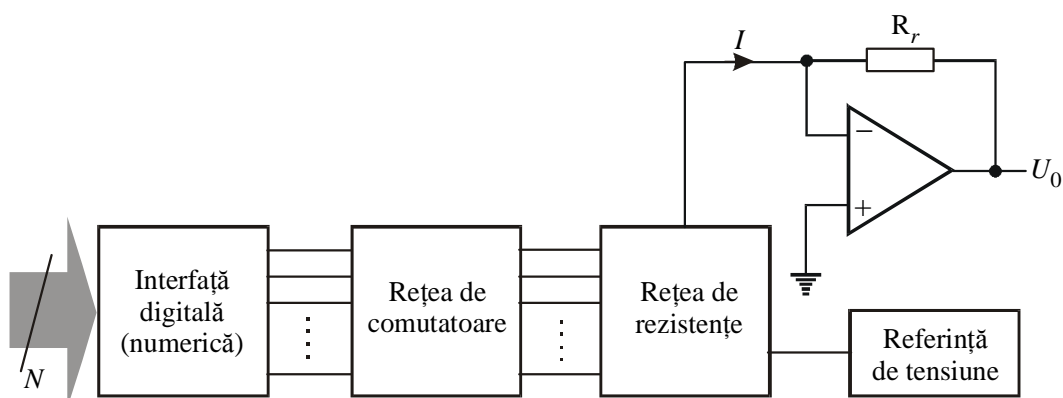


Fig. 2.11 Schema bloc simplificată a unui convertor digital-analogic.

Interfața digitală (numerică) asigură compatibilitatea convertorului cu semnale TTL/CMOS și produce semnale de comandă pentru o rețea de comutatoare analogice. Aceste comutatoare controlează curenții aplicați unei rețele rezistive de precizie, care realizează ponderarea lor, pentru a obține, prin sumare, valoarea analogică corespunzătoare. Valorile curenților care circulă prin rețea sunt determinate de valorile rezistențelor ce compun rețeaua și de mărimea (tensiune sau curent) de referință ce intră în compunerea convertorului. Ieșirea poate fi constituită chiar de suma curenților din rețea sau de o tensiune obținută prin transformarea curent-tensiune.

Convertorul prezentat anterior funcționează în permanență: la fiecare modificare a intrării, ieșirea reacționează corespunzător. Dacă se dorește menținerea valorii analogice de ieșire și în absența unei mărimi de intrare valide, se poate recurge la memorarea acesteia într-un registru încărcat adecvat, doar la momentele de timp la care se dorește modificarea ieșirii.

Pentru implementarea convertoarelor digital-analogice, așa cum a fost precizat anterior, metoda consacrată constă în utilizarea rețelelor rezistive.

Convertoarele digital-analogice cu rețele ponderate binar (fig. 2.12a) conțin un grup de rezistențe de valori $R_i = 2^i \cdot R$, $i = 1, \bar{N}$, conectate împreună la una dintre extremități. Numărul rezistențelor din rețea este determinat de numărul de biți N ai cuvântului de intrare. Fiecare intrare logică, B_i , comandă câte un comutator analogic, K_i , ce conectează câte o rezistență a rețelei la sursa de tensiune de referință, U_{ref} , generând un curent I_i . Tensiunea de ieșire poate fi calculată conform relației:

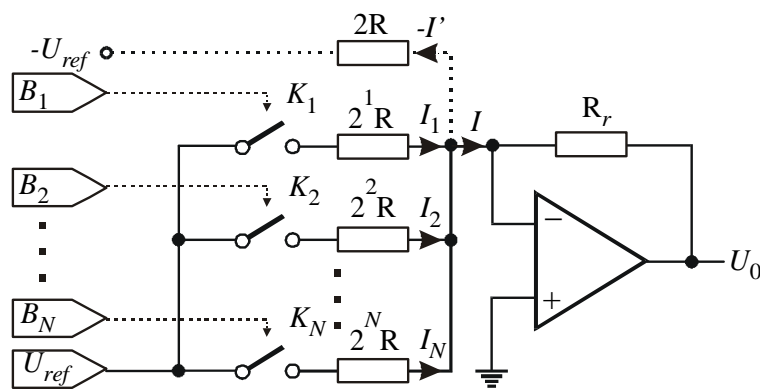
$$U_0 = R_r \cdot \sum_{i=1}^N B_i \cdot \frac{U_{ref}}{2^i \cdot R} = U_{ref} \cdot \frac{R_r}{R} \cdot \sum_{i=1}^N \frac{B_i}{2^i} = \frac{U_{ref}}{2^N} \cdot \frac{R_r}{R} \cdot \sum_{i=0}^{N-1} B_i \cdot 2^i \quad (2.7)$$

Expresia (2.7) arată că mărimea de ieșire este o fracțiune din mărimea de referință U_{ref} și proporțională cu numărul aplicat la intrare. Convertorul prezentat funcționează doar unipolar. Pentru o funcționare bipolară, schema se modifică aducând în nodul de sumare a curenților un curent egal cu jumătate din valoarea corespunzătoare capătului de gamă (fig. 2.12a cu linie punctată).

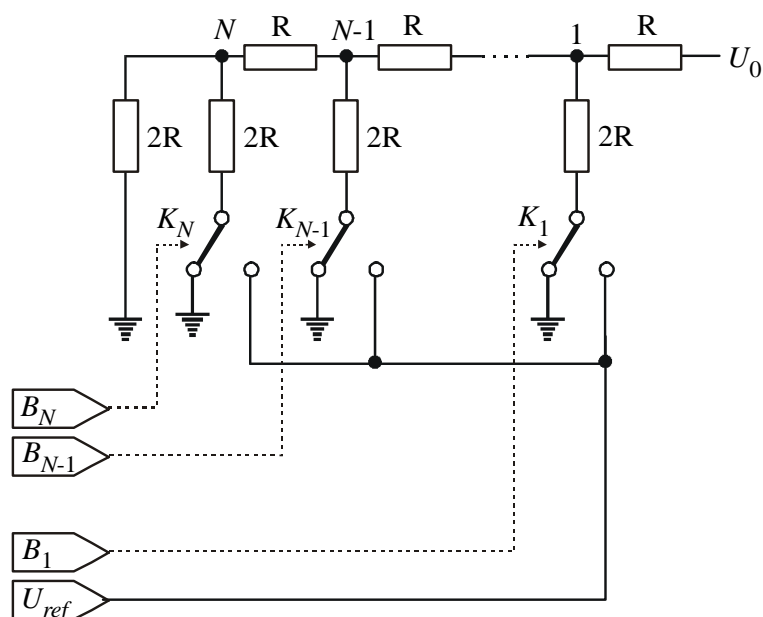
Convertoarele digital-analogice bazate pe acest principiu se numesc și *convertoare digital-analogice cu curenți ponderați*, deoarece schema utilizează sumarea unor astfel de curenți.

Simplitatea structurii prezentate în fig. 2.12a trebuie pusă în balanță cu inconvenientul major al stabilității și preciziei. Deoarece legea de variație a rezistențelor rețelei este exponențială, la un număr mare de biți, valorile lor se distribuie pe un interval foarte mare. Acest lucru face dificilă realizarea lor cu precizii ridicate și cu caracteristici de temperatură identice.

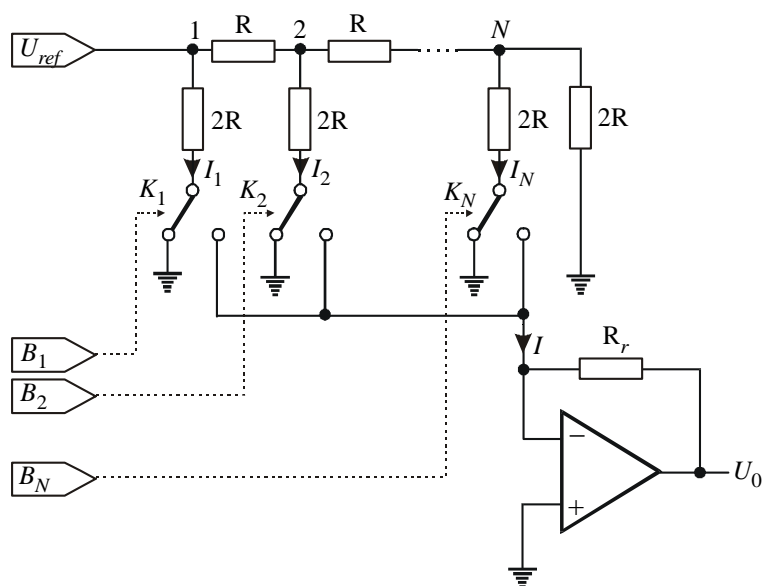
Un alt tip de rețele rezistive utilizate pe scară largă în construcția convertoarelor digital-analogice, cât și în alte circuite de instrumentație (convertoare analog-digitale, amplificatoare și atenuatoare programabile, etc) sunt rețelele rezistive R-2R. Schema unei astfel de rețele care permite o rezoluție de N biți (fig. 2.12b) prezintă caracteristicile unei legări în cascadă de divizoare cu 2, comandate fiecare de câte un bit al cuvântului de la intrare. Rețeaua rezistivă conține rezistențe de valoare R conectate în serie și rezistențe de valoare $2R$ conectate în paralel. Fiecărui bit B_i al cuvântului de intrare îi este asociat câte un comutator cu două poziții, care conectează terminalele rezistențelor $2R$ la masă ($B_i = "0"$) sau la referință ($B_i = "1"$). Comanda poate fi făcută în tensiune sau în curent. Rezistența de valoare $2R$ conectată în permanență la masă are rolul ca rezistența echivalentă a circuitului măsurată între bornele de ieșire să fie întotdeauna R .



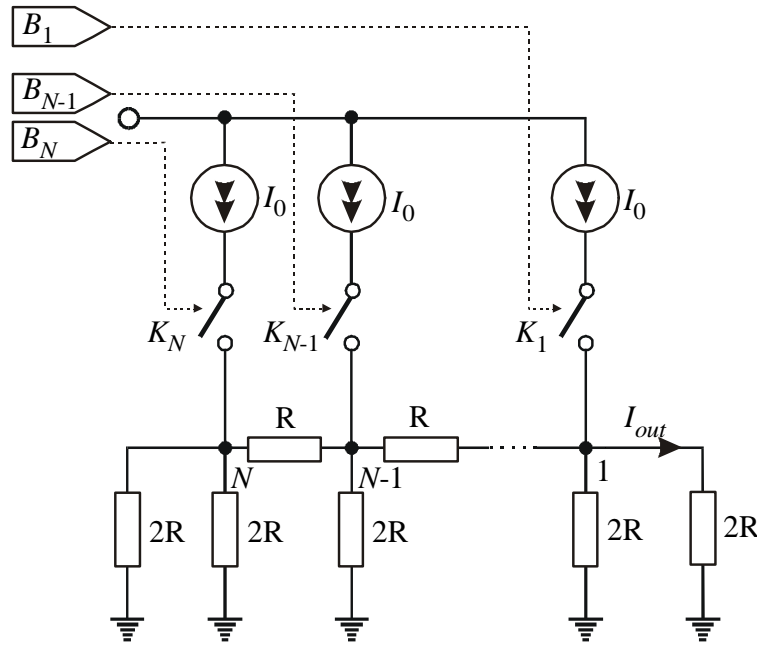
a) Convertor digital-analogic unipolar/bipolar cu rețea ponderată binar.



b) Structura rețelei rezistive R-2R.



c) Convertor digital-analogic cu rețea R-2R inversată.



d) Rețea R-2R comandată cu generatoare de curent.

Fig. 2.12 Principii de implementare ale convertoarelor digital-analogice.

În cazul structurii din fig. 2.12b, tensiunea de la ieșire este descrisă prin relația:

$$U_0 = U_{\text{ref}} \cdot \sum_{i=1}^N \frac{B_i}{2^i} = \frac{U_{\text{ref}}}{2^N} \cdot \sum_{i=0}^{N-1} B_i \cdot 2^i \quad (2.8)$$

Rețelele rezistive R-2R comandate în tensiune sunt simple și ieftine, dar au o liniaritate relativ redusă datorită comportării comutatoarelor analogice. Comutatoarele analogice CMOS au o rezistență R_{on} variabilă, dependentă de tensiunea drenă-sursă a tranzistorului cu efect de câmp. Rezistențele comutatoarelor se sumează cu cele ale rețelei, determinând erori de neliniaritate prin modificarea factorilor de divizare de la o celulă la cealaltă.

O variantă a conversiei digital-analogice utilizează conexiunea inversă, schimbând rolul ieșirii cu cel al intrării (fig. 2.12c). În această situație, comutatoarele se găsesc practic la același potențial, iar rezistențele rețelei sunt parcurse de curenți de valori constante, de tip $\frac{U_{\text{ref}}}{2^i \cdot R}$, $i = \overline{1, N}$, indiferent de poziția acestora. Valorile logice ale biților cuvântului de intrare comandă poziția comutatoarelor; acestea determină sumarea componentei i de curent în nodul de intrare al convertorului curent-tensiune sau conectarea acestei componente la masă. Dezavantajul major al acestei structuri este reprezentat de valorile relativ mari ale timpului de stabilire, datorate sumării capacităților parazite.

Tensiunea de ieșire a acestui tip de convertor digital-analogic este caracterizată de expresia (2.7).

Rețelele R-2R pot fi comandate direct în curent, folosind generatoare de

curent comutate (fig. 2.12d). Această schemă este frecvent utilizată în cadrul convertoarelor digital-analogice moderne, datorită performanțelor sale de viteză.

Din punct de vedere al interfațării convertoarelor digital-analogice în cadrul sistemelor de achiziții de date, trebuie remarcat că circuitele din prima generație foloseau o interfață minimală cu circuitele numerice, cu rolul de adaptare a nivelelor logice de comandă. Circuitele din generațiile recente conțin integrate unul sau mai multe convertoare digital-analogice, registre de memorare a cuvântului numeric de intrare și o logică de comandă. Aceste resurse permit interfațarea simplă și unitară cu microcontroller-e și microprocesoare a căror magistrală de date diferă ca dimensiune de lungimea cuvântului de comandă, respectiv programarea convertoarelor multiple (fig. 2.13).

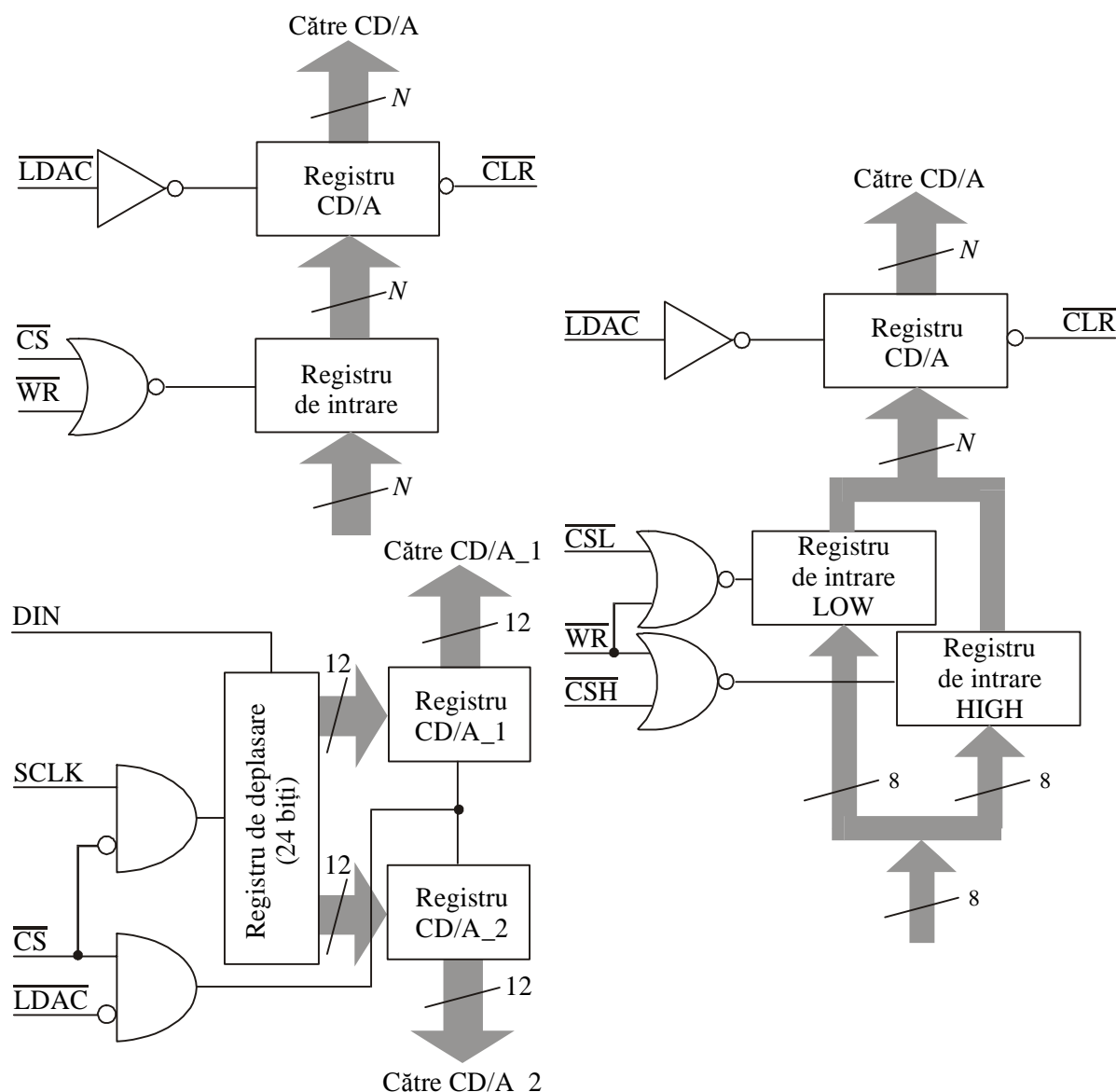


Fig. 2.13 Interfațarea convertoarelor digital-analogice cu magistrala unui microprocesor.

Logica de comandă permite memorarea întării digitale sub forme diferite (12 biți - un singur ciclu de programare, 8+4 biți - două cicluri de programare, intrare serială), precum și selectarea modului de lucru (ieșire unipolară sau bipolară).

O categorie aparte de convertoare digital-analogice o constituie circuitele de conversie cu transformare intermediară în timp. Aceste circuite folosesc transformarea mărimii numerice într-o mărime intermediară (durată sau frecvență), ce produce ulterior mărimea de ieșire analogică proporțională printr-un procedeu oarecare (filtrare, transfer de sarcină și integrare, etc). În această categorie pot fi încadrate convertoarele digital-analogice cu modulație în durată a impulsurilor (conținute în structura majorității microcontroller-elor moderne și putând fi utilizate în aplicații în care nu sunt necesare performanțe deosebite) și convertoarele digital-analogice cu transformare frecvență-tensiune.

2.2.3.2 CONVERTOARE ANALOG-DIGITALE. SCHEME DE PRINCIPIU

Convertorul analog-digital realizează transformarea mărimii analogice de la intrare într-o mărime numerică la ieșire. Generalizând, procesul de conversie analog-digitală poate fi considerat ca o plasare a mărimii de intrare într-un *interval de cuantizare*, obținut prin divizarea intervalului de variație a acesteia într-un număr de clase egale. Prima operație definește aspectul temporal al conversiei, în timp ce a doua operație definește chiar modul de obținere a echivalentului numeric al mărimii analogice.

Convertoarele analog-digitale sunt realizate pe baza unor soluții principiale extrem de diverse, fiecare dintre acestea prezentând atât avantaje, cât și dezavantaje. Până în acest moment nu s-a găsit un principiu de funcționare care să asigure simultan obținerea ieftină de rezoluții mari, viteze ridicate, erori de neliniaritate foarte reduse, etc. De aceea, alegerea unui anumit tip de convertor analog-digital se face în funcție de cerințele aplicației, urmărind obținerea performanțelor dorite cu efort material minim.

Convertoare analog-digitale paralele. Ideea simplă a inversării procedurii de conversie digital-analogică cu ponderarea controlată numeric a unei mărimi de referință, conduce la folosirea comparării mărimii de intrare cu un șir de valori de referință (reprezentând limitele intervalelor de cuantizare), pentru obținerea conversiei analog-digitale. Tensiunea de referință este aplicată unei rețele rezistive de precizie, astfel încât fracțiunea din tensiunea de referință aplicată intrării inversoare a fiecărui comparator să fie cu un LSB mai mare decât cea aplicată comparatorului de pe rangul anterior. Comparatoarele realizează atribuirea mărimii de intrare (de pe intrările neinversoare) unui interval de cuantizare; toate comparatoarele ale căror referință este mai mică

decât tensiunea de intrare produc un nivel logic “1” la ieșire, celelalte comparatoare vor furniza la ieșire nivele logice “0”. Ieșirile rețelei de $2^N - 1$ comparatoare sunt aplicate unui codificator logic cu priorități care are rolul de a furniza la ieșire codul numeric dorit (fig. 2.14).

Acest tip de convertor obține biții cuvântului de ieșire simultan și independent de valoarea sau polaritatea intrării; de aici, denumirea de convertor ***analog-digital paralel*** sau ***flash***. Numărul mic de operații, precum și simplitatea lor, determină viteza foarte ridicată a acestui tip de convertor. Principalul său dezavantaj constă în rezoluția limitată, datorată creșterii exponențiale a numărului de comparatoare odată cu creșterea numărului de biți de ieșire. Este utilizat în conversia rapidă a semnalelor video (televiziune, radar), cât și ca subansamblu în implementarea altor tipuri de convertoare rapide.

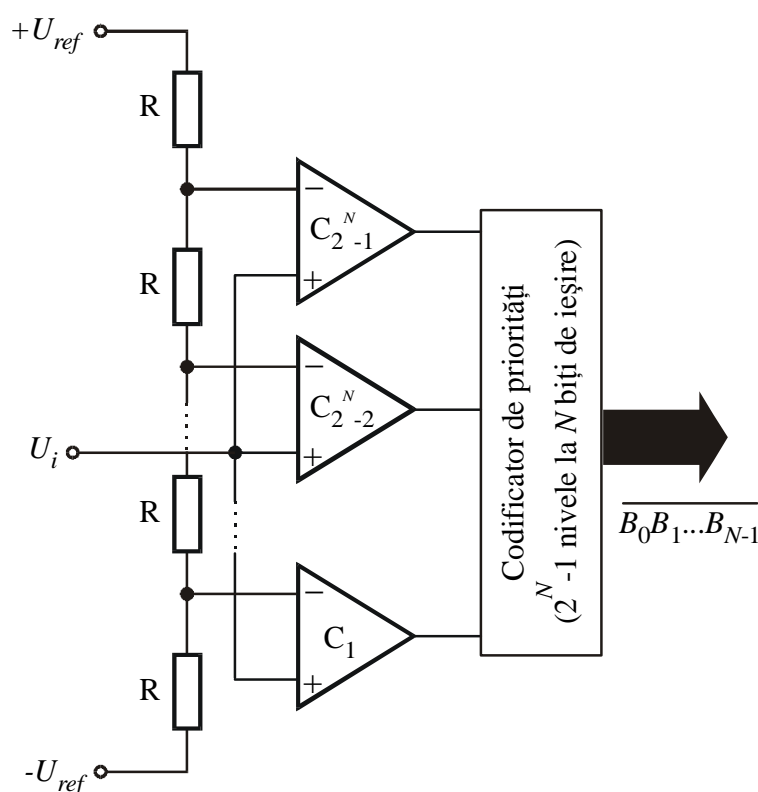


Fig. 2.14 Convertor analog-digital paralel (*flash*).

Convertoare analog-digitale analog-seriale și digital-paralele. O altă soluție de obținere a unor convertoare rapide constă în cascada unor celule elementare de conversie ce conțin amplificatoare și comparatoare (fig. 2.15).

Celula elementară (fig. 2.15a) conține două amplificatoare diferențiale cu amplificarea egală cu 2, ce produc ieșiri cu tendințe de variații contrare; ieșirile amplificatoarelor diferențiale sunt selectate cu ajutorul unui multiplexor analogic comandat de ieșirea unui comparator având tensiunea de prag egală cu jumătatea tensiunii de referință. Această tensiune de prag este obținută prin

divizarea cu 2 (cu ajutorul unui divizor rezistiv de precizie) a tensiunii de referință a convertorului. Caracteristica de tranfer a circuitului elementar este prezentată în fig. 2.15b.

În aceste condiții, se poate preciza expresia ieșirii analogice:

$$U_0 = \begin{cases} 2 \cdot U_i, & 0 \leq U_i < U_{ref}/2 \\ 2 \cdot (U_{ref} - U_i), & U_{ref}/2 \leq U_i < U_{ref} \end{cases} \quad (2.9)$$

și cea a ieșirii logice:

$$B = \begin{cases} 0, & 0 \leq U_i < U_{ref}/2 \\ 1, & U_{ref}/2 \leq U_i < U_{ref} \end{cases} \quad (2.10)$$

Lanțul de celule, realizat prin înserierea analogică, este prezentat în fig. 2.15c.

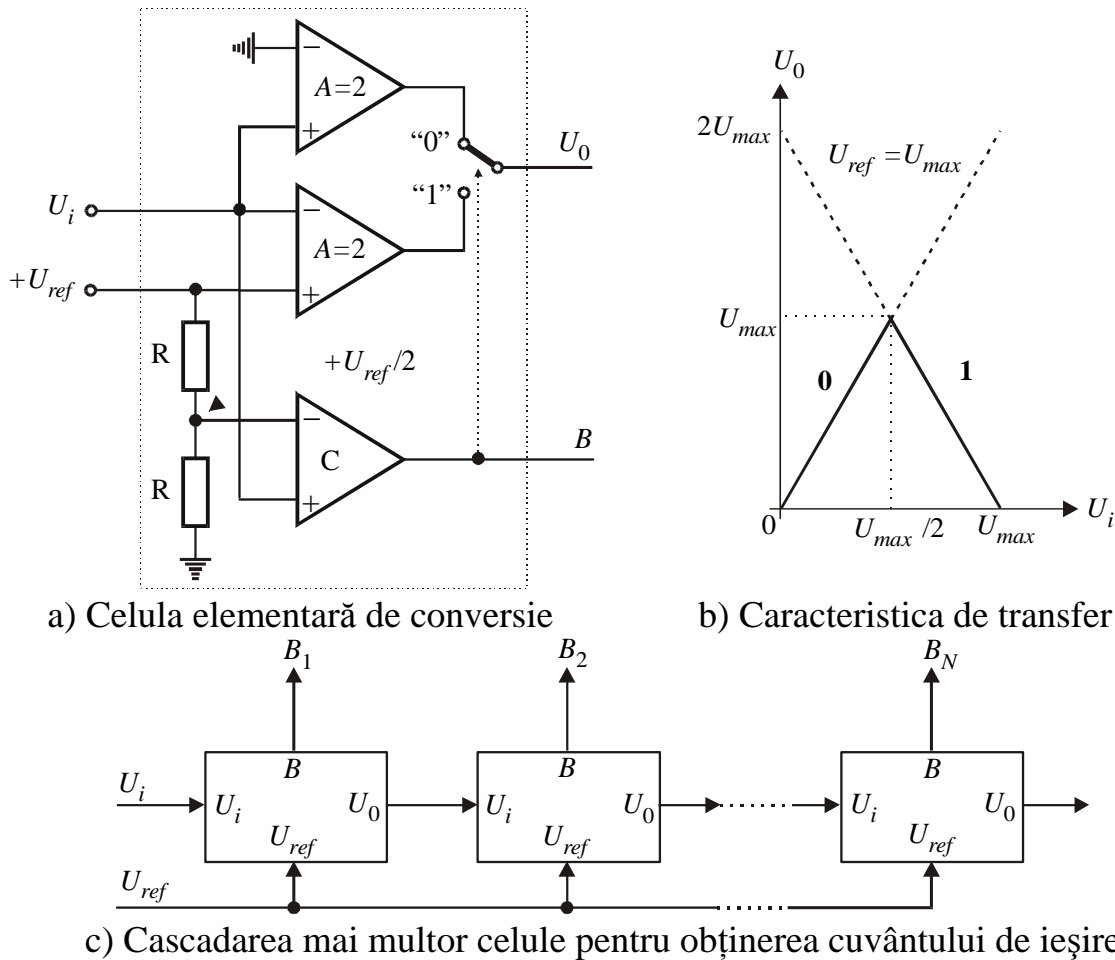


Fig. 2.15 Convertor analog-digital analog-serial și digital-paralel.

La aplicarea semnalului analogic la intrarea lanțului, acesta se va propaga rapid de la un etaj la altul producând într-un timp foarte scurt ieșiri stabile atât pe liniile analogice, cât și pe cele digitale. Este necesar, ca pe durata conversiei, valoarea analogică aplicată la intrarea lanțului să fie stabilă. Acest deziderat

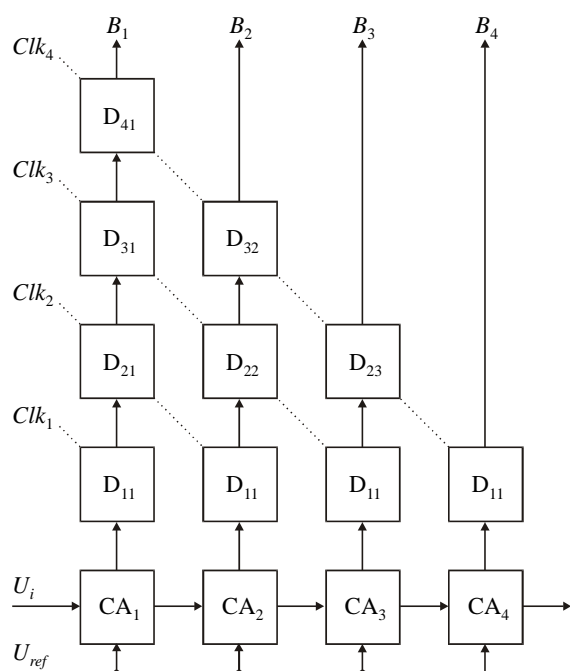


Fig. 2.16 Convertor analog-digital rapid cu timp de conversie egal cu timpul de obținere a unui bit.

poate fi realizat prin utilizarea unui circuit de eșantionare-memorare.

Timpul complet de conversie este determinat de întârzierea globală prin celulele lanțului. Cu toate acestea, fiecare bit poate fi memorat imediat ce este obținut, astfel încât o nouă conversie poate fi demarată după obținerea primului bit. Utilizând acest principiu, rata de conversie este determinată de timpul de obținere a unui singur bit. Acest tip de convertor permite, cu o schemă numerică adecvată (pipeline pe fiecare bit - fig. 2.16), obținerea unui cuvânt la rata de conversie a unei singure celule.

Structurile descrise anterior își găsesc aplicații în domeniul convertoarelor video utilizate în achiziții de imagine. În practică, numărul de celule

ce pot fi cascadeate este limitat din considerente tehnologice. Ca urmare, acest tip de convertor se utilizează în scheme mixte, împreună cu cele paralele (convertoare analog-digitale analog-seriale și digital-paralele pentru rangurile superioare și convertoarele analog-digitale paralele pentru rangurile inferioare).

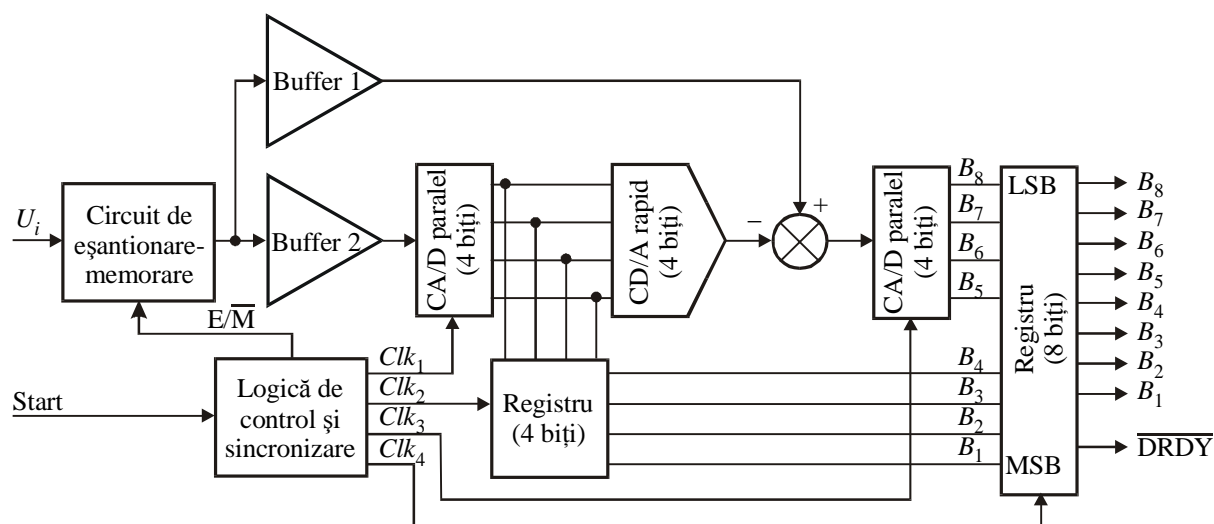
În momentul de față se constată o revenire spectaculoasă a ideii expuse, odată cu dezvoltarea arhitecturilor sistolice de prelucrare a semnalelor.

Convertoare analog-digitale serie-paralel. O soluție de compromis, care poate fi exploatată foarte eficient la obținerea unor rezoluții și viteze ridicate, este utilizarea tehnicii cu corecție de subdomeniu.

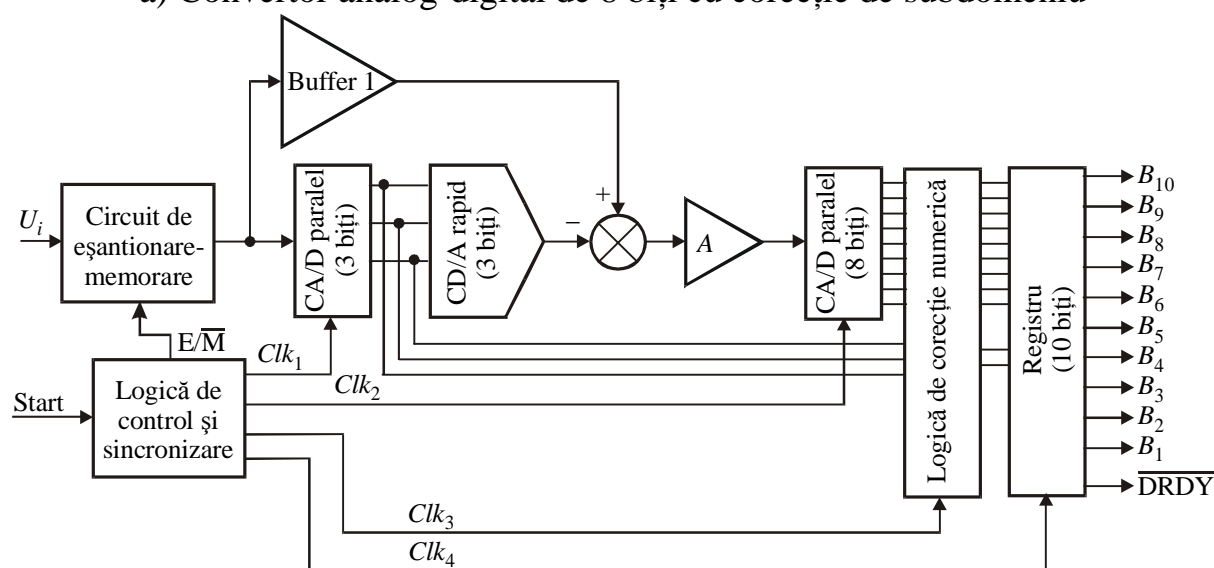
Un convertor analog-digital de N biți (număr par), funcționând pe principiul amintit, folosește două convertoare analog-digitale de $N/2$ biți care vor furniza mai întâi partea mai semnificativă a rezultatului și apoi partea mai puțin semnificativă.

În fig. 2.17a este prezentat un convertor analog-digital rapid de 8 biți. Un ciclu de conversie al acestui convertor poate fi rezumat astfel:

- se furnizează din exterior comanda de “start conversie”, care inițializează logica internă de sincronizare;
- circuitul de eșantionare-memorare este comandat în starea de memorare ($E/\overline{M} = 0$);
- se activează funcționarea primului convertor analog-digital paralel, activând semnalul Clk_1 ;



a) Convertor analog-digital de 8 biți cu corecție de subdomeniu



b) Convertor analog-digital de 10 biți cu corecție numerică de subdomeniu

Fig. 2.17 Convertoare serie-paralel.

- după încheierea conversiei, se încarcă registrul intermediar cu cei mai semnificativi 4 biți ai valorii analogice convertite, activând semnalul Clk_2 ;
- aceeași valoare numerică se aplică și convertorului digital-analogic de precizie. Acest convertor va produce la ieșire o valoare analogică foarte apropiată de cea a intrării, mai puțin eroarea de cuantizare. După expirarea timpului de stabilire, se activează funcționarea celui de-al doilea convertor analog-digital paralel, activând semnalul Clk_3 . Acest convertor primește ca semnal analogic de intrare rezultatul diferenței dintre tensiunea de intrare și versiunea sa cuantizată (de la ieșirea CD/A);

- la sfârșitul conversiei, se poate încărca, activând semnalul Clk_4 , registrul de 8 biți de la ieșire atât cu cei mai puțin semnificativi biți abia obținuți, cât și cu biții cei mai semnificativi memorați în registrul intermediar;
- după încărcarea registrului de ieșire, se poate activa semnalul $\overline{\text{DRDY}}$, semnalizând faptul că este disponibil un nou rezultat al conversiei.

Convertorul cu corecție de subdomeniu este cunoscut și sub denumirea de convertor analog-digital serie-paralel și reprezintă una dintre soluțiile de compromis între cost și performanțe. Cu toate acestea, liniaritatea diferențială este scăzută, mai ales la tranziția de la bitul $N/2$ la bitul $(N/2)+1$; această eroare poate depăși cu ușurință 1 LSB și ca urmare poate provoca omiterea unor coduri și abateri de la monotonie. Problema poate fi rezolvată cu ajutorul unei tehnici de conversie analog-digitală paralelă, numită *corecție numerică de subdomeniu* (fig. 2.17b). Convertoarele ce folosesc corecția numerică de subdomeniu au o arhitectură similară cu cea prezentată anterior, dar semnalul analogic este cuantizat suplimentar; rezoluția astfel obținută este utilizată în cadrul unui circuit numeric de corecție a erorilor incrementale, erori inerente convertoarelor analog-digitale cu corecție de subdomeniu ce folosesc tehnologii uzuale.

În figura 2.17b este prezentat un convertor analog-digital de 10 biți. Cei mai semnificativi 3 biți sunt obținuți cu un convertor A/D paralel; ei sunt introduși într-un convertor D/A de 3 biți cu precizie de 12 biți, pentru a putea păstra precizia ieșirii corespunzătoare rezoluției de 10 biți. Diferența dintre valoarea intrării și valoarea corespunzătoare ieșirii convertorului D/A este amplificată și aplicată la intrarea celui de-al doilea convertor A/D paralel de 8 biți, cu ajutorul căruia se obțin biții mai puțin semnificativi. După cum se poate observa, acest convertor produce un bit suplimentar, folosit pentru corecția numerică de subdomeniu. Această corecție contribuie substanțial la îmbunătățirea liniarității.

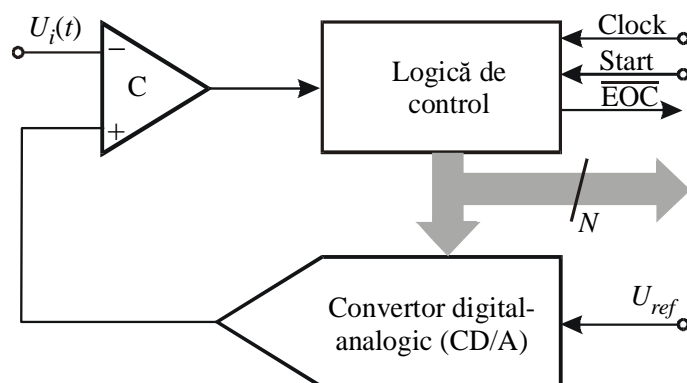
Convertoarele analog-digitale serie-paralel sunt frecvent utilizate în sistemele de achiziție a semnalelor video.

Convertoare analog-digitale cu reacție. Deși simple ca principiu, convertoarele analog-digitale paralele sunt limitate ca rezoluție datorită complexităților tehnologice (numărul mare de comparatoare determină creșterea dimensiunilor fizice, puterea disipată și prețul de cost). Ideea comparării mărimii analogice de intrare cu un set de valori de referință este aplicabilă, într-o variantă mai economică, secvențial, în cadrul convertoarelor analog-digitale cu reacție. Cu un singur comparator, un convertor digital-analogic destinat generării treptelor de referință și o logică secvențială (numărător/registru) care generează numeric limitele intervalelor de cuantizare se obține un convertor analog-digital cu reacție.

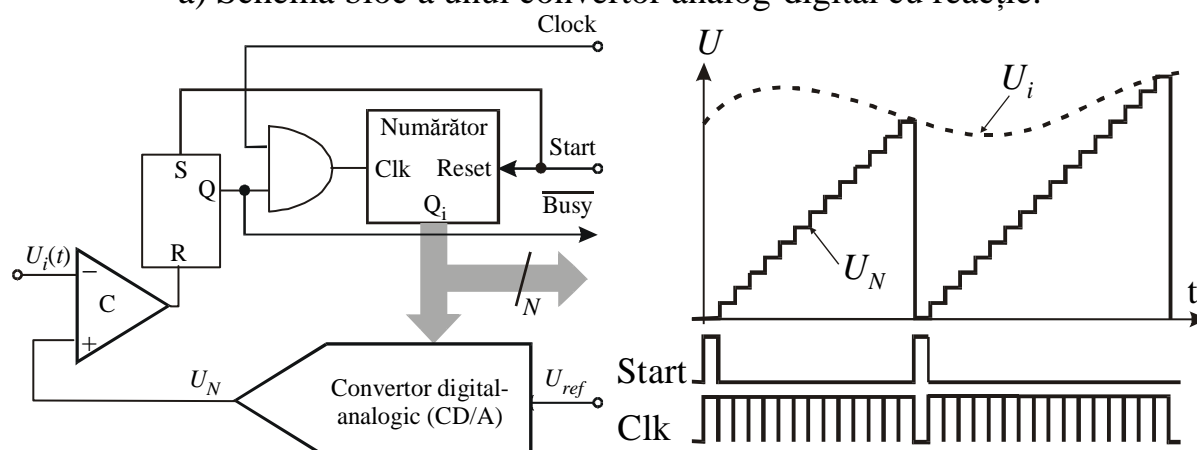
Mărimea analogică de intrare este comparată cu mărimea de referință

generată de ansamblul convertor digital-analogic-logică de control; funcție de rezultatul comparării, logica de control decide următoarea valoare logică pe care o va produce în pasul următor (fig. 2.18a). Algoritmul de conversie poate fi implementat în mai multe variante, din care rezultă și tipurile convertoarelor A/D cu reacție:

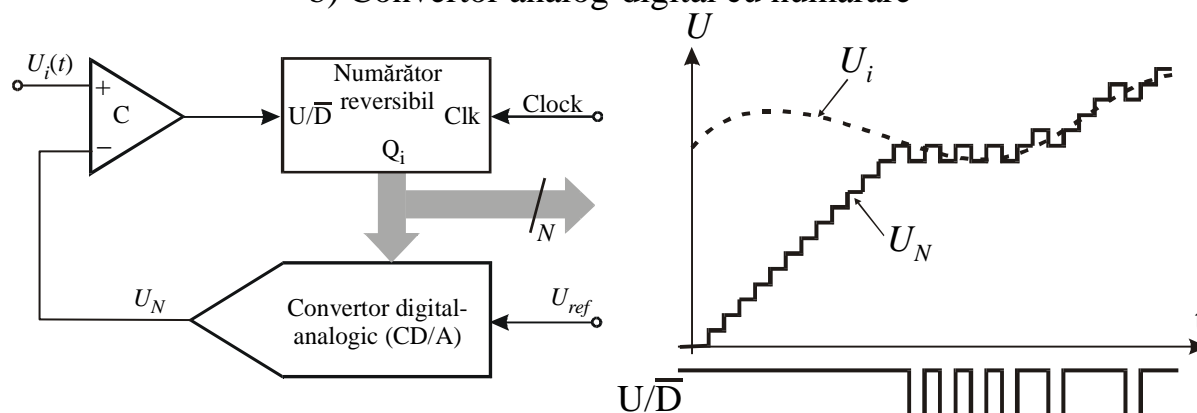
- convertor analog-digital cu numărare (fig. 2.18b);
- convertor analog-digital cu urmărire (fig. 2.18c);
- convertor analog-digital cu aproximații succesive (fig. 2.18d).



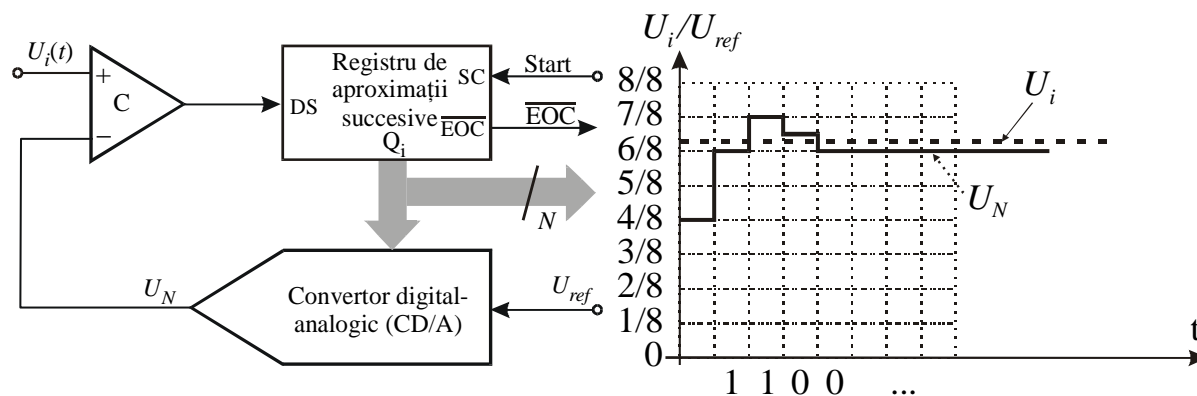
a) Schema bloc a unui convertor analog-digital cu reacție.



b) Convertor analog-digital cu numărare



c) Convertor analog-digital cu urmărire



d) Convertor analog-digital cu aproximații succesive

Fig. 2.18 Convertoare analog-digitale cu reacție.

Convertorul A/D cu numărare folosește cel mai simplu algoritm de generare a treptelor de referință: parcurgerea lor consecutivă (numărare), de la limita inferioară a gamei de lucru și până la depășirea valorii analogice de la intrare (fig. 2.18b). Logica de control are la bază un numărător, inițializat la începutul fiecărui ciclu de conversie; numărul de biți ai acestuia este egal cu rezoluția convertorului D/A și a circuitului de conversie realizat. Semnalul “Start” determină *reset*-area numărătorului și validează intrarea de ceas a numărătorului prin *set*-area bistabilului de tip RS; în același timp, convertorul D/A produce o tensiune U_N la limita inferioară a domeniului de lucru. Dispunând de semnal de ceas, numărătorul începe să se incrementeze, crescând și tensiunea de referință U_N treptă cu treaptă. La atingerea valorii semnalului de intrare, comparatorul își schimbă starea de la ieșire, *reset*-ează bistabilul și oprește ceasul de numărare, finalizând procesul de conversie. Ieșirea acestui bistabil poate fi utilizată drept semnal “*conversie în curs de desfășurare*”.

Se poate observa ușor că durata conversiei nu este constantă, ea depinzând de valoarea mărimii analogice aplicate la intrare. Deși timpul de conversie poate fi redus prin creșterea frecvenței ceasului, limita sa superioară este determinată de timpul de propagare pentru numărător și circuitele porții, de timpul de stabilire al CD/A și al comparatorului.

Deși avantajul major al acestei structuri rezidă în simplitatea sa, acest convertor cu reacție e caracterizat de un timp de conversie ridicat, dependent de valoarea intrării, precum și de o rejecție slabă a perturbațiilor (determinată de variația impedanței de intrare).

Înlocuind în schema precedentă numărătorul cu incrementare cu unul reversibil (cu incrementare/decrementare) și comandând sensul de numărare în funcție de rezultatul comparării mărimii de intrare cu treptele de referință, se obține un *convertor analog-digital cu urmărire* (funcționare continuă) (fig. 2.18c). Ieșirea comparatorului reprezintă, de fapt, codificarea pe un bit a tendinței de variație a semnalului de intrare. Dacă semnalul de intrare este relativ constant, după egalizarea semnalului U_N cu mărimea de la intrare, ieșirea

comparatorului va oscila, odată cu U_N , eroarea conversiei fiind $\pm 0,5$ LSB. Valoarea numerică corespunzătoare intrării va fi oricare dintre stările numărătorului reversibil (aproximație prin lipsă sau adaos).

Problema fundamentală a acestor două tipuri de convertoare A/D cu reacție constă în posibilitatea apariției distorsiunilor de neurmărire, cauzate de viteza constantă de incrementare/decrementare a numărătorului (limitează viteza de variație a semnalelor aplicate la intrare). În practică, banda de frecvențe a semnalului de intrare este limitată la valori de ordinul câtorva kHz.

Înlocuind numărătorul din bucla de reacție a convertorului cu un registru de deplasare special, denumit registru de aproximații succesive, se determină eliminarea dezavantajelor menționate anterior. Se obține, astfel, un *convertor analog-digital cu aproximații succesive*.

În fig 2.18d este prezentată schema funcțională a convertorului analog-digital cu aproximații succesive pentru $N=3$ și se prezintă principiul lui de funcționare.

Conversia începe cu inițializarea la valoarea “1” a bitului celui mai semnificativ (MSB) în cadrul registrului de aproximații succesive. Aceasta corespunde primei evaluări a valorii semnalului de intrare cu jumătatea valorii domeniului de intrare. Se compară semnalul de ieșire al CD/A corespunzător acestei valori cu tensiunea de intrare și se comandă de *reset*-area valorii bitului celui mai semnificativ dacă evaluarea primară depășește valoarea semnalului de intrare; în caz contrar aceeași valoare este validată și este memorată. În tactul următor controlerul fixează valoarea “1” pentru următorul bit și, din nivelul semnalului de intrare, comparatorul “decide” memorarea sau reset-area stării acestui rang. Conversia continuă în mod similar, până se evaluează bitul cel mai puțin semnificativ (LSB). În acest moment, cuvântul conținut în registrul de aproximații succesive (transferat și în registrul de ieșire) reprezintă cea mai bună aproximație numerică a semnalului analogic de intrare. Dacă datele se obțin direct de la ieșirea registrului de aproximații succesive, trebuie menționat că acestea devin stabile doar după sfârșitul conversiei (în rest ele reproduc procesul de aproximare); în consecință, logica externă trebuie adaptată în mod corespunzător.

În metoda de conversie bazată pe aproximații succesive, semnalul de ieșire al CD/A crește neliniar până la nivelul semnalului de intrare pe perioada a N tacte (pentru convertorul cu rezoluția de N biți). Ca rezultat, procesul de conversie durează un timp considerabil mai redus și, în plus, timpul de conversie este constant și nu depinde de nivelul, semnul sau modulul de variație a semnalului de la intrare.

Metoda aproximațiilor succesive este cea mai răspândită metodă de conversie analog-digitală pentru convertoarele de uz general, cu rate de conversie medii și ridicate (timpuri de conversie cuprinși între 1 și 25 μ s).

Convertoare analog-digitale cu transformare tensiune-timp. Aceste

tipuri de convertoare realizează transformarea mărimii analogice de intrare într-un interval de timp proporțional, care este măsurat numeric. Din această categorie fac parte:

- convertorul analog-digital cu generator de rampă;
- convertorul analog-digital cu integrare în dublă pantă;
- convertorul analog-digital cu integrare în mai multe rampe.

Convertoarele analog-digitale cu integrare în dublă pantă sunt caracterizate de o precizie și o liniaritate excelente, o bună rejecție a semnalelor parazite (datorită integrării), în pofida timpului de conversie de valoare mare. Majoritatea circuitelor sunt monolitice, realizate în tehnologie CMOS, fiind extrem de răspândite în echipamente de măsurare numerice clasice (aparate portabile, de tablou sau de laborator).

Trebuie menționat faptul că majoritatea convertoarelor analog-digitale de generație recentă dispun de o interfață specializată, versatilă cu microprocesoare pe 8 sau 16 biți, ceea ce simplifică mult interfațarea acestor circuite în cadrul sistemelor inteligente de achiziții de date.

Firmele producătoare de convertoare analog-digitale oferă dispozitive cu o paletă largă de performanțe. Metoda de conversie utilizată (cu aproximații succesive, cu integrare cu dublă pantă, conversie paralelă, etc) și tehnologia de realizare a schemei (monolitică, hibridă sau modul) determină caracteristicile esențiale ale convertoarelor analog-digitale - rapiditatea, rezoluția, prețul.

Paleta de variație a unor caracteristici ale CA/D, realizate în practică, sunt prezentate în fig 2.19.

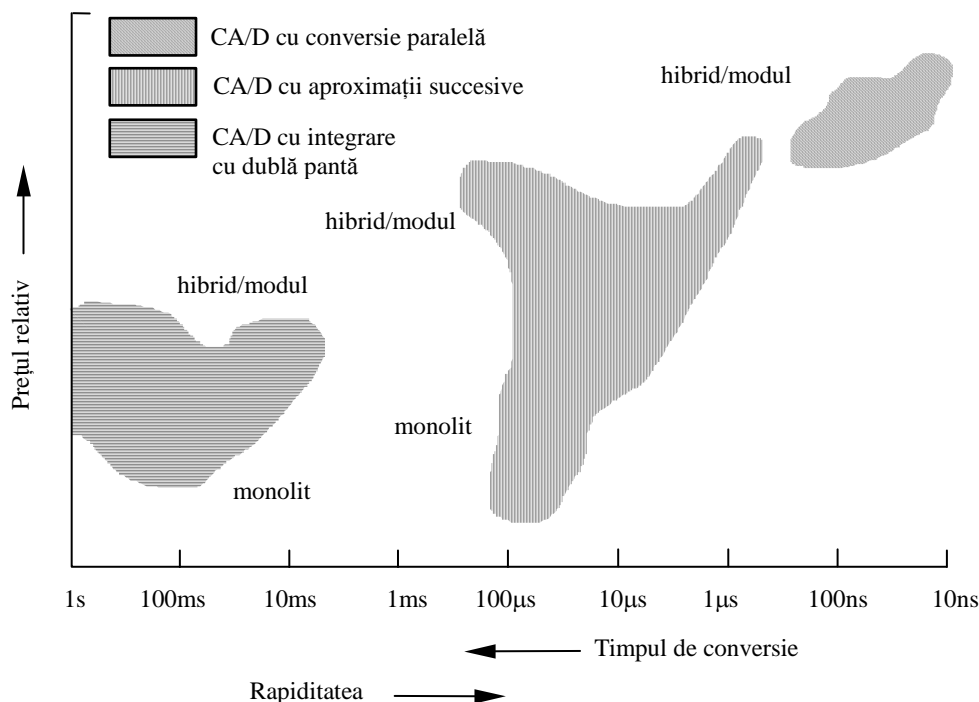


Fig. 2.19 Paleta caracteristicilor CA/D produse în serie.

Tabelul 2.2 Principalele caracteristici ale unor convertoare analog-digitale.

Tipul	Rezoluția	Metoda de conversie	Timpul de conversie	Tensiunea de alimentare	Tehnologia de realizare
ADC0804	8	cu aproximații succesive	100 μ s	+5V	monolitică
AD7574	8	cu aproximații succesive	15 μ s	+5V	monolitică
AD570	8	cu aproximații succesive	25 μ s	+5V, -15V	monolitică
TSC7109	12	cu integrare dublă pantă	33 ms	+5V	monolitică
ADC0808	8	cu aproximații succesive	100 μ s	+5V	monolitică
AD5010	6	paralelă	10 ns	\pm 5V	monolitică
AD579	10	cu aproximații succesive	2,2 μ s	+5V, \pm 15V	hibridă
AD574	12	cu aproximații succesive	25 μ s	+5V, \pm 15V	hibridă
ADC868	12	cu aproximații succesive	0,5 μ s	+5V, \pm 15V	hibridă
HS9516	16	cu aproximații succesive	100 μ s	+5V, \pm 15V	hibridă
ADC71	16	cu aproximații succesive	50 μ s	+5V, \pm 15V	hibridă

În tabelul 2.2 sunt prezentate sintetic principalele caracteristici ale unor CA/D uzuale, realizate de firmele National Intersil, Analog Device, Teledyne, Texas Instruments și Hybrid System.

Se constată o varietate mai largă a CA/D cu aproximații succesive, utilizate în majoritatea cazurilor în cadrul proceselor care necesită conversia analog-digitală. Cele mai ieftine sunt convertoarele analog-digital monolitice. Aceste CA/D sunt realizate în tehnologie bipolară și CMOS.

Convertorul analog-digital optim pentru măsurători și achiziții de date în rețele electroenergetice este CA/D cu aproximații succesive, care asigură viteze bune de conversie, precizie ridicată, rezoluția fiind un compromis între viteză și precizie.

2.3 INTERFEȚE SPECIALIZATE DE COMUNICAȚIE

Tehnicile de măsurare pot fi implementate la nivel fizic prin blocuri funcționale cu destinație precisă (aparatele de măsurare) sau prin module care pot realiza funcții multiple (eșantionare, conversie, memorare) și a căror selecție este făcută de o unitate centrală (eventual PC). În cazul aparatelor de măsurare

numerice, dotarea acestora cu interfețe de comunicație (serială sau paralelă) permite interconectarea lor cu unități de calcul puternice și, deci, lărgirea considerabilă a ariei funcțiilor ce pot fi efectuate de sistemul astfel realizat.

Aparatele numerice memorează datele sub formă de caractere reprezentate adesea pe 8, 16 sau 32 de biți. Biții care formează un caracter se pot transmite la distanță către un alt sistem numeric fie prin transmiterea simultană a câte 8 biți (comunicație paralelă), fie prin transmiterea succesivă a biților care formează un caracter (comunicație serială). În primul caz, se utilizează 8 linii de date și alte linii (conductoare) pentru semnalul de referință (GND) și cele de control al comunicației. În al doilea caz, informația prezentă de obicei sub formă paralelă este apelată de un registru de deplasare paralel-serie, comandat de un semnal de tact, transmisă printr-o singură pereche de conductoare și apoi, la recepție, reconstituită în format paralel prin intermediul registrului de deplasare serie-paralel.

2.3.1 COMUNICAȚIA DE TIP SERIAL. PROTOCOALE DE TRANSMISIE SERIALĂ A DATELOR

Interfața serială este un sistem de comunicație numerică introdus ca urmare a necesității de a controla un ansamblu tehnic cu elemente dispersate pe suprafețe mari. PC-urile sunt dotate cu mai multe porturi seriale (de obicei, două), utilizate, în cea mai mare parte, pentru comanda plotter-elor, a imprimantelor seriale și a unor mouse-uri. De asemenea, această interfață este folosită pentru comunicația cu PC-ul și de către dispozitive speciale, cum ar fi programatoarele EPROM și PAL, emulatoarele, controller-ele logice programabile sau anumite interfețe de achiziție de date.

Achiziția datelor se efectuează prin executarea unui program de achiziție de către calculatorul care asigură comanda mijlocului de măsurare, transferul datelor într-un fișier de date și prelucrarea lor imediată sau ulterioară. Denumirea **RS-232** (mai exact, **RS-232C**) corespunde normei americane a interfeței seriale, normă propusă inițial în 1960 și devenită variantă standard în 1969, apoi remodificată în 1987. Denumirea **V24** este o prescurtare a normei franceze (și recomandată CEI). În principiu, ambele norme sunt identice.

În prezent există și module dedicate comunicației seriale performante, cum este **RS-485** (de tip plug-in) pentru care se poate asigura comunicația până la distanța de 1,2 km, cu o viteză maximă de transfer de 100 kHz .

2.3.1.1 INTERFAȚA RS-232

Numeroase aparate utilizează conectarea la calculator prin intermediul

interfeței seriale RS-232. Norma clasifică aparatele în două categorii:

1. **DTE** (*Data Terminal Equipments*) – categorie din care fac parte PC-ul, tastatura etc.
- și
2. **DCE** (*Data Communication Equipments*) – modem-urile, aparatele de măsurare etc.

Modul de conectare poate să difere de la un aparat la altul. În principiu, se poate conecta **numai un singur aparat** la o interfață serială. Programarea modului de comunicație poate fi, de asemenea, foarte diferită. De aceea, nu se poate vorbi de un standard. În forma minimală, o conexiune serială RS-232 se compune din numai 3 conductoare:

1. **RXD** (**Receive Data**), conductorul pentru semnalul de recepție;
2. **TXD** (**Transmit Data**), conductorul pentru semnalul de emisie;
3. **GND** (**Ground**), conductorul de masă.

Modul de legare a conductoarelor RXD și TXD la portul calculatorului depinde de aparatul utilizat.

Siguranța în transmisia datelor poate fi mai mare dacă se introduce o comunicație de tip *handshaking*. În acest caz se folosesc (fig. 3.3), pe lângă semnalele **RXD** și **TXD** (semnale de date), și semnalele **RTS** (**Request To Send**) și **CTS** (**Clear To Send**).

RTS (cerere de emisie) și **CTS** (autorizare de emisie) sunt semnale care girează funcționarea half-duplex (HDX) - de exemplu, a liniei telefonice.

Calculatorul semnalizează modem-ului prin **RTS** că are un caracter de transmis; transmisia este posibilă numai atunci când semnalul **CTS** este primit de calculator. O siguranță superioară în transmisia datelor se obține prin utilizarea semnalelor **DTR** (**Data Terminal Ready**) și **DSR** (**Data Set Ready**). Prin aceste semnale receptorul este anunțat că emițătorul este pregătit să trimită datele. Astfel, **DTR** poate fi perceput ca un semnal de **BUSY** pentru receptor.

Siguranța unei transmisii este determinată de lungimea cablurilor de legătură (maximum 2 x 15=30 m), nivelul de tensiune al semnalelor și viteza de transmisie.

Nivelele de tensiune pentru interfața **RS-232** sunt:

- HIGH: -15 V (-25 V);
- LOW: +15 V (+25 V).

Intervalul de la -3 V la +3 V nu este permis (fig. 2.21).

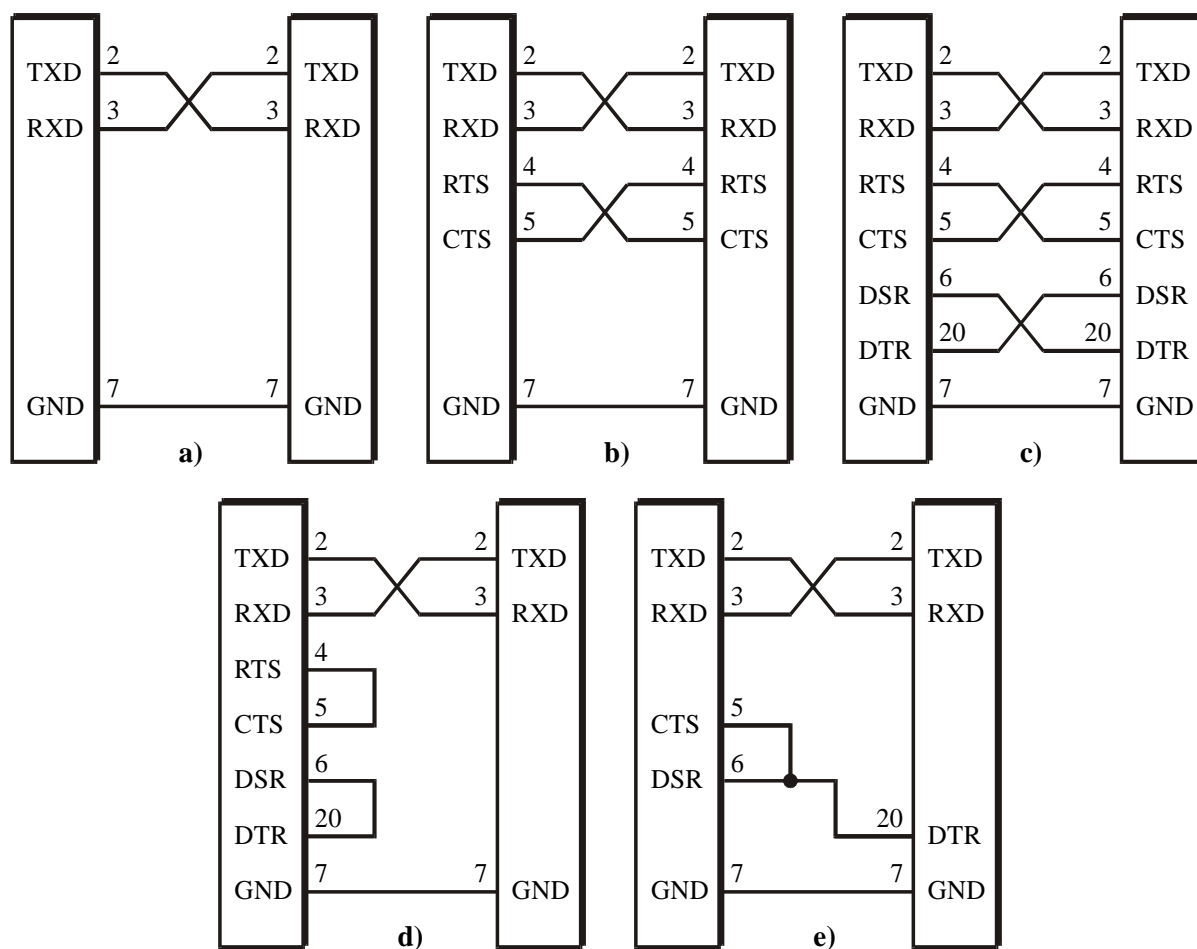


Fig. 2.20 Tipuri de conexiuni utilizate în interfațarea serială: a) varianta minimală; b) varianta handshake; c) handshake cu confirmare DTR și DSR; d) transmisie cu punte pe semnalele de handshake; e) conectarea unui plotter.

Viteza de transmisie este dată în **BAUD**¹. O altă unitate uzuală în cazul transmisiilor este **BPS (Bits Per Second)**. În cazul comunicației seriale între două echipamente, exprimarea vitezei de transmisie în **BAUD** și **BPS** este identică. În cazul conectării lor prin intermediul modemurilor, însă, acest lucru nu mai este valabil. Valorile uzuale pentru viteza de transmisie (*Baudrate*) sunt date mai jos:

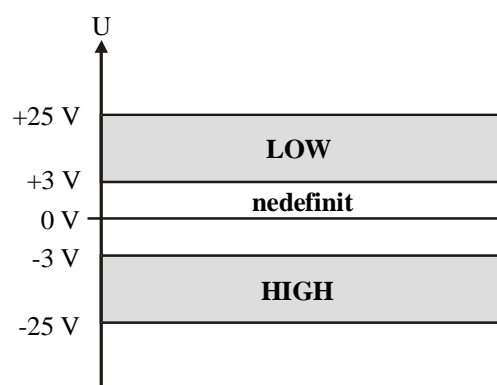


Fig. 2.21 Nivelele de tensiune pentru portul serial.

¹ BAUD este unitatea de măsurare a numărului de schimbări pe secundă ale stării unei linii, denumită după **Jean Maurice Emile Baudot**, un fost ofițer al Serviciului Francez de Telegrafie. El a proiectat, la sfârșitul secolului al XIX-lea, primul cod pe 5 biți pentru reprezentarea unitară a caracterelor alfabetului.

50 110 300 600 1200 2400 4800 9600 19200 38400

Formatul de transmisie al datelor este descris prin următorii parametri:

- Baudrate (viteza de transmisie);
- Startbit (bitul² de start);
- Numărul de biți de date;
- Paritatea;
- Numărul de biți de stop.

Prin intermediul biților de start și stop se determină începutul, respectiv sfârșitul secvenței de date transmisă. Numărul de biți de date este, de obicei, 7 sau 8. Prin intermediul testului de paritate se pot evidenția eventualele erori de transmisie. În acest sens, există trei posibilități de detecție:

- **No Parity**: Nu se face nici un test de paritate;
- **Even parity** (paritate pară): Emițătorul numără toți biții de date care au valoarea “1” și setează bitul de paritate cu “1”, dacă suma a fost impară, și cu “0”, dacă suma a fost pară. Receptorul adună biții de date cu valoarea bitului de paritate. Suma este **totdeauna** (în cazul unei transmisii corecte) **pară**; în caz contrar, a survenit o eroare la transmisia datelor;
- **Odd Parity** (paritate impară): Metoda corespunde testului de paritate pară, cu deosebirea că suma biților de date și a celui de paritate este **totdeauna** (la emițător) **impară**.

În fig. 2.22 se descrie procesul de transmitere a caracterului “2” (în reprezentarea binară corespunzătoare codului ASCII) cu protocolul “1 bit de start, 7 biți de date, 2 biți de stop, paritate impară”.

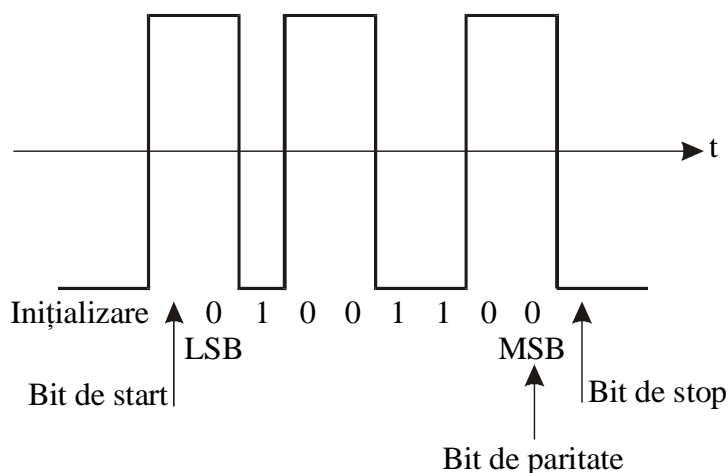


Fig. 2.22 Semnalele corespunzătoare transmisiei caracterului “2”.

² Termenul bit a apărut pentru prima dată scris cu sensul utilizat astăzi în informatică în anul 1949, ales de **John Tuckey** care s-a decis (în timp ce lua prânzul) pentru o variantă mai comodă decât denumirile de **bigit** sau **binit**.

2.3.1.2 INTERFAȚA I²C

Pentru a exploata similaritățile care există în proiectele și echipamentele proiectate de diverși utilizatori, ca și pentru maximizarea eficienței *hardware*-ului și pentru simplificarea proiectării circuitelor, a fost dezvoltată o magistrală bidirecțională pe două fire, cu scopul eficientizării controlului interconectării circuitelor integrate. Această magistrală se numește **INTER IC** sau **I²C**. În prezent, această magistrală permite cuplarea a mai mult de 150 de tipuri de circuite integrate, realizate în tehnologie CMOS sau bipolară, realizând funcții în domeniul controlului inteligent, a circuitelor integrate de uz dedicat (*driver*-e pentru afișaje cu cristale lichide, porturi de intrare-ieșire, memorii RAM și EEPROM, convertoare) și a circuitelor orientate pe aplicații (procesare de semnale pentru sisteme radio și video, generatoare DTFM pentru telefonie, etc.). Toate circuitele compatibile **I²C** încorporează o interfață care permite intercomunicația rapidă prin intermediul acestui tip de magistrală.

Dintre caracteristicile generale ale magistralei **I²C** putem menționa:

- magistrala conține doar două linii: o linie serială de date (**SDA**) și o linie de ceas serial (**SCL**);
- fiecare dispozitiv conectat la magistrală este adresabil prin *software*, având o adresă unică; pe magistrala **I²C** se manifestă, la orice moment de timp, o relație de tip master-slave;
- magistrala **I²C** este o magistrală multi-master, incluzând detecția conflictelor și arbitrarea acesteia, pentru a preveni alterarea informației dacă două sau mai multe dispozitive master inițiază transferuri simultane;
- transferurile bidirecționale de date, cu lungimi de 8 biți, pot fi efectuate cu rate de transfer de 100 kbiți pe secundă, în modul standard, sau cu maxim 400 kbiți pe secundă, în modul rapid;
- rejectarea impulsurilor scurte, parazite, de pe magistrală, este asigurată de circuitele de filtrare implementate în fiecare dispozitiv cuplat la magistrală. Rejecția acestor impulsuri asigură păstrarea integrității datelor;
- numărul de dispozitive cuplabile pe aceeași magistrală **I²C** este limitat doar de capacitatea maximă suportată de aceasta și care este de 400 pF.

Circuitele integrate compatibile cu magistrala **I²C** permit dezvoltarea rapidă a proiectării de la o schemă bloc funcțională la prototip, asigurând proiectanților o serie întregă de avantaje:

- structura extrem de simplă a magistralei (2 fire) minimizează interconexiunile cu exteriorul;
- protocolul complet integrat al magistralei **I²C** elimină folosirea decodificatoarelor de adrese și a unei logici externe, suplimentare;

- capacitățile de multimaster ale magistralei I^2C permit testarea rapidă și alinierea utilizatorilor, prin utilizarea unor conexiuni externe, la un sistem de calcul;
- disponibilitatea circuitelor integrate I^2C sub amprente de tip **SO** (Small Outline), **VSO** (Very Small Outline) și **DIL** (Dual In Line) reduce necesitățile de spațiu.

2.3.1.2.1 SPECIFICAȚIILE INTERFEȚEI I^2C

Pentru aplicații de control industrial pe 8 biți, care necesită utilizarea unor microcontroller-e, pot fi stabilite a priori anumite criterii de proiectare:

- un astfel de sistem este compus din cel puțin un microcontroller și din alte dispozitive periferice, ca de pildă memorii și circuite de intrare-ieșire (fig. 2.23);
- criteriul principal de proiectare constă în minimizarea costului de interconectare a diferitelor dispozitive din componența sistemului;
- un sistem care asigură o funcție de reglare (control) într-un proces nu necesită rate mari ale transferurilor de date;
- eficiența globală a sistemului depinde de natura circuitelor utilizate și de structura magistralei de interconectare a acestora.

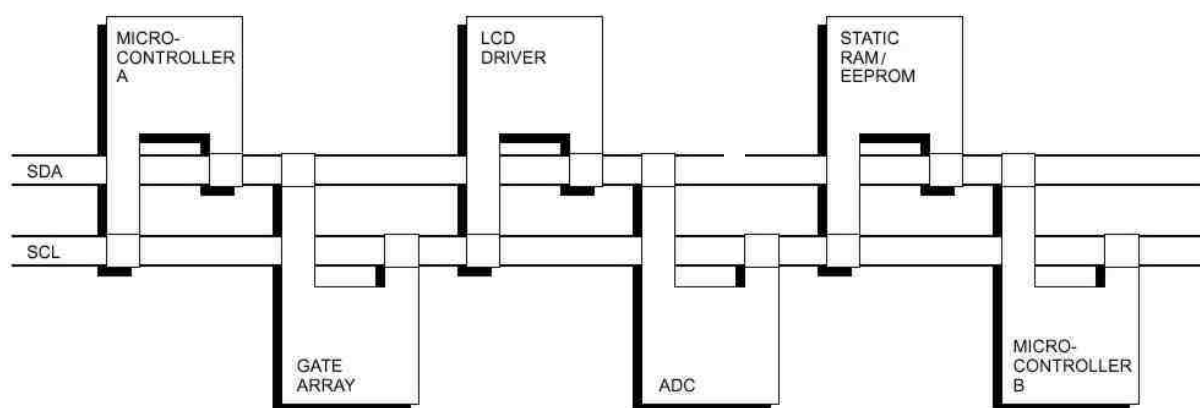


Fig. 2.23 Exemplu de sistem organizat în jurul magistralei I^2C .

Pentru a satisface aceste criterii, este necesară utilizarea unei magistrale seriale, care deși nu permite rate de transfer a informațiilor atât de mari ca o magistrală de interconectare de tip paralel, asigură minimizarea numărului firelor și pinilor de interconectare între diversele circuite utilizate în proiect.

Dispozitivele care intercomunică prin intermediul unei magistrale seriale necesită utilizarea unor protocoale care au rolul de a elimina erorile, pierderile de informații și conflictele pe magistrală și de asemenea, posibilitatea ca unele

dispozitive rapide să poată comunica cu dispozitive lente. Este necesar ca sistemul să poată funcționa independent de numărul de dispozitive înglobate în structura sa, sau cu alte cuvinte, adăugarea de dispozitive în structura sistemului să nu afecteze funcționarea acestuia.

2.3.1.2.2 CONCEPTUL DE MAGISTRALĂ I²C

Magistrala I²C permite cuplarea unor circuite compatibile în structura sistemului, indiferent de tehnologia de fabricație a acestora: NMOS, CMOS sau bipolară. Magistrala constă în două linii: o linie serială de date (**SDA**) și o linie de ceas serial (**SCL**), ce manipulează informațiile între oricare două dispozitive cuplate la magistrală. Orice dispozitiv este recunoscut prin intermediul unei adrese unice asociate, indiferent dacă este vorba de un microprocesor, display cu cristale lichide, interfață de tastatură, etc., și poate funcționa ca emițător sau receptor, depinzând de funcția realizată de acesta (fig. 2.24). O clasificare suplimentară a dispozitivelor cuplate la magistrala I²C constă în dispozitive *master*, respectiv *slave*. Un dispozitiv *master* este acela care poate iniția un transfer de date pe magistrală și care generează semnalul de ceas ce coordonează transferul. În tot acest timp, orice alt dispozitiv adresat este privit ca *slave*.

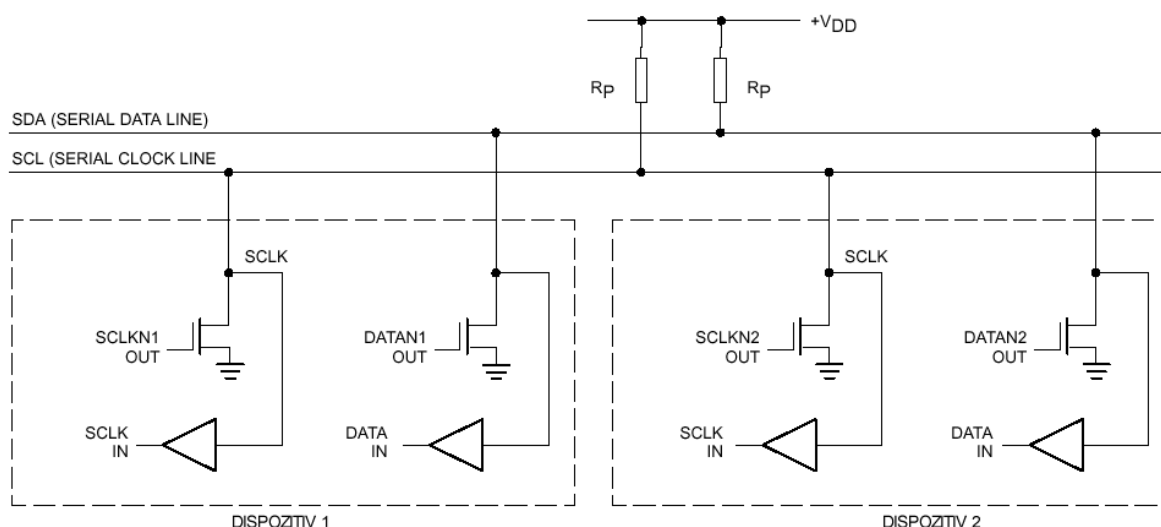


Fig. 2.24 Interconectarea a două dispozitive pe magistrala I²C.

Magistrala I²C este o magistrală de tip *multi-master*. Aceasta înseamnă că mai multe dispozitive care pot controla magistrala pot fi cuplate la aceasta. Posibilitatea de a cupla mai mult de un dispozitiv *master* la magistrală înseamnă că mai mult de un singur dispozitiv poate încerca să inițieze un transfer pe magistrală, la același moment de timp. Pentru a se evita această situație de incertitudine, a fost elaborată o procedură de arbitraj a priorităților, bazată pe

conectarea de tip ȘI-cablat a tuturor dispozitivelor la magistrală. Semnalele de ceas pe durata arbitrării de priorități reprezintă rezultatul sincronizării semnalelor de ceas generate de cele două dispozitive *master* prin utilizarea funcției de tip ȘI-cablat a liniilor **SCL**. Generarea semnalelor de ceas pe magistrală revine întotdeauna în sarcina dispozitivelor *master*; fiecare dispozitiv *master* generează propriul său semnal de ceas pe durata transferului de date pe magistrala sistemului. Semnalele de ceas de pe magistrală pot fi doar alterate numai dacă un dispozitiv *slave* lent forțează linia de ceas la nivel logic LOW sau de un alt dispozitiv *master*, pe durata arbitrării priorităților.

Ambele linii, **SDA** și **SCL**, sunt bidirecționale și conectate printr-o rezistență de “pull-up” la tensiunea de alimentare. Atunci când magistrala este liberă, ambele linii sunt în starea HIGH. Etajul de ieșire al dispozitivului conectat la magistrală trebuie să fie de tip *open-drain* sau *open-collector* pentru a se realiza funcția ȘI-cablat. Pe magistrala **I²C**, transferurile de date pot fi efectuate cu rate de maxim 100 kbiți/s în modul standard sau maxim 400 kbiți/s în modul rapid. Numărul de dispozitive cuplabile la magistrala **I²C** este limitat doar de încărcarea capacitivă (maxim 400 pF) a liniilor magistralei.

2.3.1.2.3 TRANSFERURILE PE MAGISTRALA **I²C**

Datorită diversității tehnologiilor de implementare a circuitelor cuplabile la liniile interfeței **I²C**, nivelele logice nu sunt fixate și depind de valoarea tensiunii de alimentare V_{DD} . Pentru transferul fiecărui bit este generat câte un impuls de ceas.

a) Validitatea datelor

Datele vehiculate pe linia **SDA** trebuie să fie stabile pe durata HIGH a impulsului de ceas. Modificările stării liniei **SDA** trebuie să se producă doar atunci când semnalul de ceas este LOW.

b) Condițiile START și STOP

Printre procedurile implementate pe magistrala **I²C**, situații de excepție sunt considerate condițiile de START și STOP.

O tranziție din starea HIGH în starea LOW a liniei **SDA**, pe durata căreia linia **SCL** este HIGH, este interpretată ca o condiție de START.

O tranziție din starea LOW în starea HIGH a liniei **SDA**, pe durata căreia linia **SCL** este HIGH, este interpretată ca o condiție de STOP.

Condițiile de START și de STOP sunt generate întotdeauna de un dispozitiv *master*. După generarea unei condiții de START se consideră că magistrala este ocupată. Magistrala este considerată din nou ca fiind neutilizată după apariția unei condiții de STOP.

Detectarea condițiilor de START și de STOP de către dispozitivele *slave* cuplate la magistrală este foarte facilă dacă acestea înglobează *hardware*-ul specializat de interfațare. Pentru dispozitivele care nu dispun de acest *hardware* specializat, se impune ca linia **SDA** să fie eșantionată de două ori pe durata unei perioade de ceas, pentru ca această tranziție să poată fi detectată.

2.3.1.2.3.1 TRANSFERURILE DE DATE PE MAGISTRALĂ

a) Transferurile de date sub formă de cuvânt

Orice cuvânt de date transferat pe magistrală trebuie să aibă lungimea de 8 biți. În schimb, numărul de octeți ce pot fi transferați pe linia **SDA** este practic nelimitat. Fiecare octet transferat trebuie să fie urmat de un bit de confirmare (*acknowledge*). Transferurile de date încep întotdeauna cu bitul cel mai semnificativ al octetului respectiv. Dacă un dispozitiv receptor nu poate accepta un alt octet de date înainte de a realiza o funcție specială cum ar fi de pildă tratarea unei întreruperi interne, acesta poate forța linia de ceas, **SCL**, la nivel LOW pentru a face ca emițătorul să intre în stare de WAIT. Transferul de date poate continua atunci când receptorul eliberează linia **SCL**.

În anumite cazuri, este posibilă utilizarea unui alt format pentru transferul pe magistrală. Un mesaj care începe cu o astfel de adresă poate fi terminat prin utilizarea unei condiții de STOP, chiar în timpul transmiterii unui octet de informație. În această situație nu se generează bitul de confirmare.

b) Bitul de confirmare

Transferurile de date cu confirmare sunt obligatorii pentru a se asigura integritatea datelor pe magistrală. Semnalul de ceas asociat bitului de confirmare este generat de dispozitivul *master*. Pe durata acestui impuls de ceas, dispozitivul emitent eliberează linia **SDA** (nivelul acesteia este HIGH).

Dispozitivul de recepție trebuie să forțeze linia **SDA** la nivel coborât pe durata impulsului de ceas de confirmare, astfel acest nivel coborât să rămână stabil pe durata HIGH a impulsului de ceas de confirmare.

În mod obișnuit, un dispozitiv ce realizează funcția de recepție mesaj trebuie să emită câte un semnal de confirmare după fiecare octet recepționat. Atunci când un dispozitiv *slave* cu funcție de recepție nu confirmă adresa asociată (de exemplu, acest dispozitiv nu este capabil să răspundă deoarece efectuează un set de operații în timp real), linia de date trebuie lăsată neutilizată (HIGH) de către dispozitivul *slave*. În această situație, dispozitivul *master* poate genera o condiție de STOP pentru a termina transferul. Dacă dispozitivul *slave* ce realizează funcția de recepție confirmă adresa asociată dar în procesul de transfer ulterior nu mai poate recepționa octeți, este, de asemenea, necesar ca dispozitivul *master* să termine transferul. Acest fapt este indicat prin faptul că

receptorul nu confirmă recepționarea următorului octet, lasă linia **SDA** pe nivel HIGH, iar dispozitivul master generează condiția de STOP.

Dacă în procesul de transfer este implicat un dispozitiv *master* ce realizează funcția de recepție, acest dispozitiv trebuie să semnalizeze sfârșitul transferului prin neconfirmarea ultimului octet recepționat de la *slave*. Dispozitivul *slave* trebuie să elibereze linia **SDA** pentru ca dispozitivul *master* să poată transmite o condiție de STOP.

2.3.1.2.4 ARBITRAREA PRIORITĂȚILOR ȘI GENERAREA CEASULUI

a) Sincronizarea pe magistrala I^2C

Toate dispozitivele *master* generează propriul semnal de ceas pe linia **SCL** pentru a transmite mesaje pe magistrala I^2C . Datele sunt valide doar pe durata HIGH a impulsurilor de ceas. Prezența unui semnal de ceas pe magistrală este necesară pentru procedura de arbitrare bit cu bit.

Sincronizarea ceasului este asigurată prin utilizarea conexiunii de tip ȘI-cablat a interfețelor de magistrală la linia **SCL**. Aceasta înseamnă că o tranziție din HIGH în LOW pe linia **SCL** va determina dispozitivele cuplate la magistrală să își înceapă procesul de contorizare a perioadelor LOW odată ce semnalul de ceas al unui dispozitiv a devenit LOW și se va menține linia **SCL** în această stare până semnalul de ceas devine din nou HIGH. Totuși, tranziția din starea LOW în starea HIGH nu va determina schimbarea stării liniei de ceas dacă un alt semnal de ceas cuplat la linia de ceas a magistralei se află în stare LOW. Durata cât timp linia **SCL** va fi menținută în stare LOW va fi determinată de dispozitivul care este caracterizat de cea mai mare durată a nivelului coborât al ceasului. Celelalte dispozitive, caracterizate de o durată mai mică a palierului stării LOW a semnalului de ceas, trec în stare de WAIT cu semnalul de ceas la nivel ridicat.

Atunci când, toate dispozitivele, implicate în procesul de comunicare pe magistrală, și-au încheiat contorizarea perioadei LOW a semnalului de ceas, linia respectivă va fi eliberată și va trece în stare HIGH. În acest mod, nu vor mai exista diferențe între semnalele de ceas ale dispozitivelor și starea liniei de ceas a magistralei, toate dispozitivele începându-și contorizarea duratelor HIGH ale semnalelor de ceas. Primul dispozitiv care își încheie perioada HIGH a semnalului de ceas va forța linia **SCL** din nou la nivel LOW.

Semnalul de ceas de pe linia **SCL** este astfel sincronizat, având durata de nivel coborât determinată de dispozitivul caracterizat de cea mai lungă perioadă LOW a semnalului de ceas și durata de nivel ridicat determinată de dispozitivul caracterizat de cea mai scurtă perioadă HIGH a semnalului de ceas.

b) Arbitrarea priorităților

Un dispozitiv **master** poate iniția un transfer de date doar dacă magistrala este liberă. Două sau mai multe dispozitive **master** de magistrală pot genera o condiție de START pe durata timpului de HOLD din condiția de START. Arbitrarea are loc prin intermediul liniei de date, **SDA**, pe durata cât linia de ceas, **SCL**, este pe nivel HIGH. Astfel, unul dintre dispozitivele **master** transmite un nivel HIGH pe magistrală, în timp ce celălalt, care transmite un nivel LOW, își va dezactiva etajul de ieșire deoarece nivelul logic de pe magistrală nu corespunde cu nivelul logic transmis de către acesta. Arbitrarea poate continua pentru mai mulți biți. Prima etapă constă în compararea biților de adresă. Dacă două dispozitive **master** încearcă să adreseze același dispozitiv **slave**, arbitrarea continuă cu compararea datelor. Deoarece adresele și datele sunt utilizate pentru arbitrarea magistralei, se constată că nu există pierderi de informație pe liniile magistralei pe durata acestui proces.

Un dispozitiv **master** care pierde arbitrarea poate genera impulsuri de ceas până la încheierea procesului de transmitere a octetului în cursul căruia a pierdut arbitrarea.

2.3.1.3 INTERFAȚA USB

Interfața USB (Universal Serial Bus) a fost proiectată cu scopul de a simplifica procedura de conectare a perifericelor la un PC, crescând viteza de transmisie prin intermediul unei comunicații de tip serial până la valori de 12 Mbit/s. Faptul că necesită o conectare mai facilă impune însă utilizarea unui protocol mai complex, pentru păstrarea eficienței și transparenței față de utilizator.

USB este deja recomandat pentru noua generație de PC-uri compatibile IBM de către PC'98 System Design Guide și este, deja inclus ca driver în sistemul de operare Windows 98.

Suportul hardware constă dintr-o conexiune pe patru conductoare, dintre care două sunt pentru alimentare (V_{bus}) respectiv masă (GND) iar celelalte două pentru transferul de date (D+ și D-). Prin intermediul USB se pot conecta **simultan** la un PC până la 126 de periferice cu avantajul suplimentar al reducerii costului și al spațiului alocat plăcii de bază a PC-ului (PCB) prin eliminarea necesității unui port suplimentar “tradițional” cum sunt cele ale tastaturii și/sau porturile seriale clasice. Bineînțeles că marele avantaj rezidă în costul scăzut al USB și în viteza (12 Mbit/s în așa-numitul “full-speed mode”) care permite transferul în timp real al semnalelor de voce sau video comprimat.

La sfârșitul anului 1999 a fost lansată oficial varianta USB2.0 care permite transferuri de până la 120, respectiv 240 Mbit/s. În cele ce urmează se prezintă protocolul USB 1.1, pe baza unui exemplu de implementare hardware: *Infineon Technologies C541 embedded USB microcontroller*.

Arhitectura USB se compune din trei elemente principale (fig. 2.25) - gazda (*host*), conectorii (*hubs*) și perifericele (*devices*). Conexiunea utilizează topologia “tiered-star” și poate fi structurată în nivele, deci poate avea până la 5 distribuitoare (*hub tiers*). În mod uzual, controller-ul gazdă (*host controller*) și *hub*-ul principal (de *root*) sunt implementate pe un același chip pe placa de bază a PC-ului. Controller-ul gazdă controlează transmisiile prin sistemul USB. Există două tipuri de *host controllers*: OHCI (**O**pen-**H**ost **C**ontroller **I**nterface) și UHCI (**U**niversal **C**ontroller **H**ost **I**nterface). Din punct de vedere al aplicațiilor, OHCI poate gestiona multiple tranzacții pentru un anumit periferic *End Point* (EP) într-un interval de 1 ms. Pe de altă parte, UHCI permite câte o tranzacție pentru fiecare EP în fiecare cadru de aplicație (*frame*). Pachetele software ale echipamentelor USB trebuie să fie capabile să gestioneze comunicația cu fiecare dintre aceste tipuri de controller-e.

Un distribuitor principal acționează ca un port care se atașează echipamentului USB (fig. 2.25), permițând multiple conexiuni la sistemul USB și detectează momentele când echipamentele sunt conectate sau deconectate de la sistem. De asemenea, el transmite mai departe traficul pe *bus* între portul trece-sus (*upstream*) și porturile trece-jos adiacente (*downstream*).

Fiecare echipament dotat cu USB are alocate numere EP. Numărul EP0 este rezervat pentru configurarea echipamentelor de către gazdă. El asigură un punct de comunicație către gazdă prin intermediul descriptorilor EP. Descriptorii EP comunică atributele echipamentelor și caracteristicile acestora gazdei. În conformitate cu aceste informații, gazda configurează echipamentul și-i alocă driver-ul software corespunzător (USB client software).

Celelalte EP pot fi considerate ca o funcție a echipamentelor și pot fi configurate separat pentru unul dintre tipurile de transfer pentru a comunica cu gazda. De exemplu, o aplicație de tastatură, care se clasifică în standardul USB “**H**uman **I**nterface **D**evice”, HID, folosește EP0 pentru configurarea echipamentului (tastaturii) și poate folosi EP1 ca un transfer pe întreruperi pentru trimiterea datelor (*key-scanned data*) către gazdă.

USB suportă 4 tipuri de transfer de date:

- *Control transfer* - comenzi de cereri de transfer de la gazdă către echipament;
- *Interrupt transfer* - transfer de date de la un *interrupt driver device* către gazdă;
- *Bulk transfer* - transferul unei cantități mari de date;
- *Isochronous transfer* - pentru aplicații care necesită rate de transfer constante.

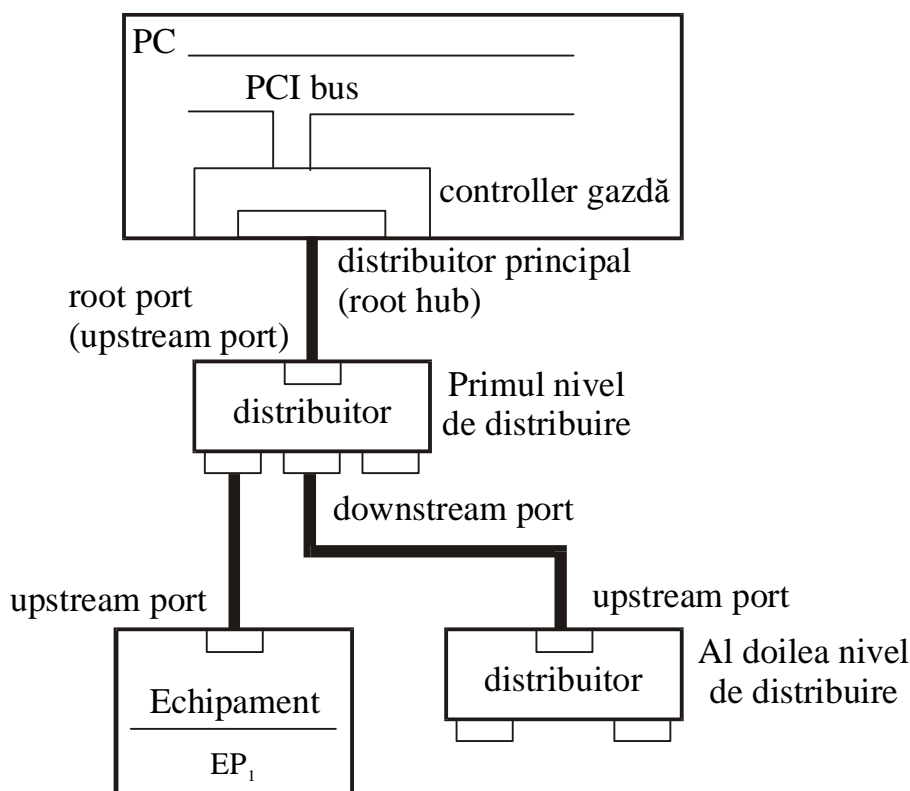


Fig. 2.25 Structura generală a unei interfețe USB.

2.3.2 COMUNICAȚIA DE TIP PARALEL. PROTOCOALE DE TRANSMISIE PARALELĂ A DATELOR

Pentru sistemele de măsurare ce utilizează aparate inteligente conduse de calculator, comunicația paralelă este cea mai indicată, asigurând viteze mari de comunicație, drept pentru care au fost realizate standarde internaționale la care s-au aliniat majoritatea constructorilor de aparate inteligente dotate cu microprocesoare.

Comunicația paralelă este utilizată și pentru alte aparate cuplate la calculator, cum ar fi: imprimante, plotter-e, dispozitive de memorie externă etc.

Până în anii '60 au existat numai aparate de măsurare cu comandă manuală și de-abia o dată cu apariția aparatelor numerice, în deceniile următoare, s-au proiectat primele interfețe cu rolul, la început, de a permite cuplarea mai multor aparate de măsurare între ele. În anii '70 s-a pus problema standardizării interfețelor, prima soluție constituind-o interfața **RS-232** pentru interconectarea calculatoarelor ca și a perifericelor la acestea. Încă din 1965, însă, firma Hewlett-Packard lucra la definirea unui concept de interfață **HPIB** (Hewlett Packard Interface Bus), din care a decurs apoi norma internațională **IEC 625-1**, adoptată în 1976.

2.3.2.1 INTERFAȚA HPIB

BUS-ul IEC 625 utilizează transmisia asincronă ceea ce înseamnă că viteza de comunicație este determinată de aparatul cel mai lent din sistem. Acesta este numai aparent un dezavantaj, deoarece timpul de măsurare al aparatelor este de obicei mult mai mare decât timpul necesar comunicației.

Se obțin astfel viteze de 2000-3000 kBaud ceea ce nu este deloc puțin în comparație cu comunicația serială prin RS-232 ce poate asigura maximum 38,4 kBaud.

BUS-ul IEC pentru sistemele de măsurare este cunoscut sub mai multe denumiri și variante, diferențele dintre acestea fiind însă minime. Astfel, între HPIB și GPIB, realizate după standardul american IEEE-488 și, respectiv, standardul internațional IEC-625, diferența este la conectarea în cuple și numărul de pini ai acestora. Prescurtările au următoarele semnificații:

- HPIB: Hewlett Packard Interface Bus
- GPIB: General Purpose Interface Bus

Conectorul utilizat de bus-ul HPIB este redat în fig.3.9. Acest conector are 24 de pini care sunt alocați conform standardului pentru intrări-ieșiri de date și comenzi și care vor fi explicați în continuare.

Pentru realizarea unui sistem automat de măsurare prin interfața HPIB este necesar un echipament de calcul (PC), care să posede implementată pe magistrala proprie placa de interfața pentru acest Bus, iar aparatele utilizate trebuie să fie prevăzute de asemenea cu această interfață.

Sistemul poate fi format din **maximum 15 aparate** ce pot fi conectate la calculator în două moduri: în stea (fig. 2.27a) sau în serie (fig. 2.27b).

Legăturile între aparate trebuie să fie cât mai scurte și să nu depășească lungimea de 2 m.

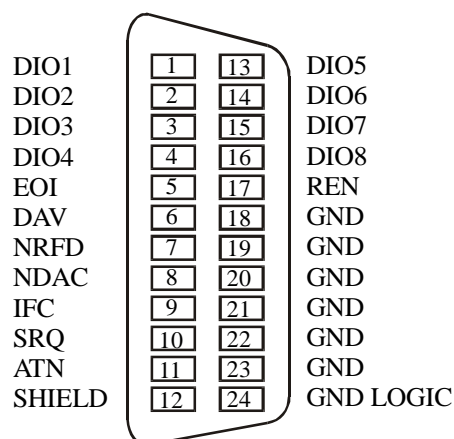


Fig. 2.26 Conectorul HPIB.

Legătura în stea asigură o configurație mai avantajoasă asigurând distanțe minime între aparate și o viteză de comunicație mai mare, de aceea este mai recomandată.

Legătura în serie permite o dispersare mai mare a aparatelor, dar o conectare imperfectă la una dintre cuple poate crea probleme de reflexii pe cabluri sau chiar întreruperea comunicației între aparate. De asemenea, viteza de comunicație este mai mică, datorită traseelor mai lungi.

2.3.2.1.1 STRUCTURA BUS-ULUI HPIB

Aparatele compatibile cu protocolul HPIB și dotate, astfel, cu interfața corespunzătoare, pot avea în sistem trei funcții:

- **ascultătorii** (listeners), care primesc datele atunci când sunt adresați. Pot fi activi și mai mulți ascultători simultan;
- **vorbitorii** (talkers), care emit date atunci când sunt adresați. Un singur vorbitor poate fi activ la un moment dat pe bus;
- **controller-ele**, (controllers) care adresează aparatele legate la bus, fie că este vorba de ascultători, fie că este vorba de vorbitori, și trimit instrucțiuni speciale și semnale de comandă.

Pentru a comanda în mod eficient, controller-ul trebuie să poată asculta și vorbi în mod egal. Într-un sistem de măsurare automatizat se pot distinge trei tipuri de sarcini:

- selecția unui aparat;
- transferul informațiilor;
- gestiunea transferului de informații.

Noțiunea de gestiune a transferului de informații este aici foarte importantă, deoarece un aparat “vorbitor” (care poate fi controller-ul, când el emite adrese, sau un aparat selecționat când el primește date) nu poate efectua un nou transfer decât atunci când el este sigur că aparatele vizate au primit în condiții bune informația precedentă.

Sistemul HPIB este constituit din ansamblul elementelor funcționale electrice și mecanice ale unei interfețe conforme cu norma IEEE-488. Cablul utilizat pentru conectarea aparatelor are 24 de conductoare, din care 16 sunt repartizate în trei grupe:

- transferul informațiilor 8 linii;
- gestiunea transferului de informații 3 linii;
- gestiunea generală 5 linii.

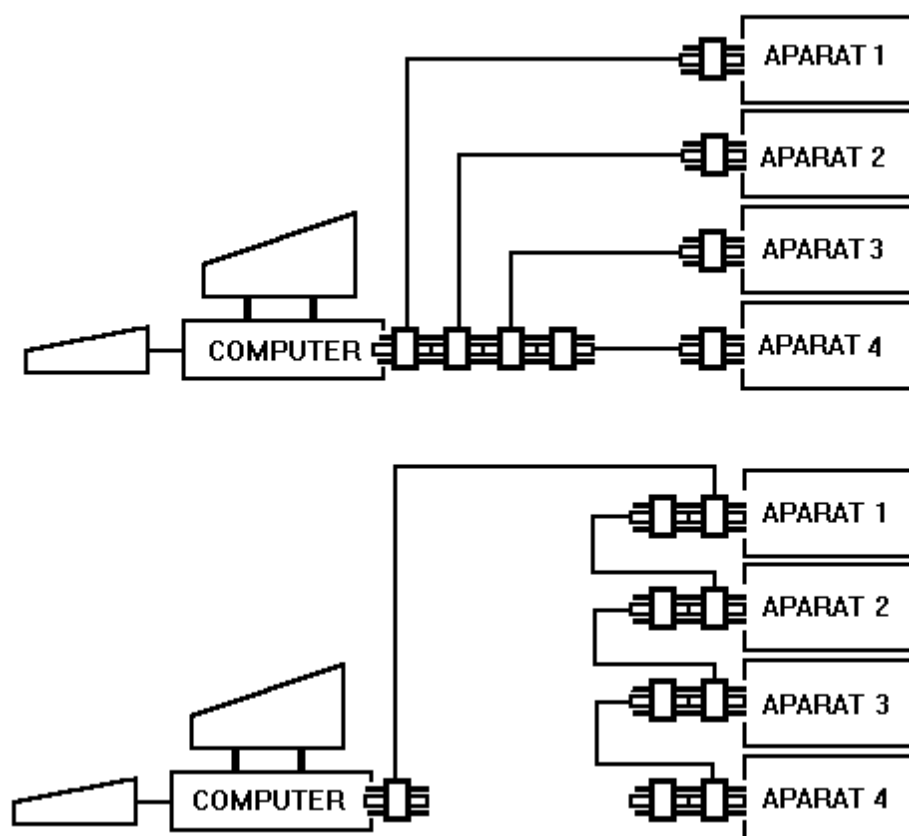


Fig. 2.27 Conectarea aparatelor la calculator prin bus-ul HP-IB.

Linii de transfer de date (DIO1÷DIO8) sunt rezervate intrărilor și ieșirilor. Ele sunt utilizate pentru măsurări, instrucțiuni de programare cuvinte de stare, adrese, instrucțiuni de interfațare. Un octet de date este format din 8 biți transmiși în paralel. Un mesaj poate să cuprindă mai mulți octeți; acești octeți sunt transmiși atunci în serie. Viteza de transfer poate atinge 1 Moctet/secundă pentru o comunicație bidirecțională asincronă.

Linii de gestiune a transferului sunt destinate să gireze transferul fiecărui octet de la un aparat emițător spre unul sau mai multe aparate receptoare și asigură protocolul de tip “handshake” ori de câte ori informațiile sunt transmise pe cele 8 linii ale bus-ului de date. Aceste linii sunt:

- **DAV (DAta Valid)**; această linie informează că datele prezentate pe bus de la un emițător sunt valide; ele sunt comandate de emițător;
- **NRFD (Not Ready For Data)**; această linie este acționată de controler în modul “comandă” (ATN=1) și de receptor în modul “date” (ATN=0);
- **NDAC (Not Data ACcepted)**; atunci când NDAC=1, datele nu sunt acceptate de receptor, sau în modul “comandă”, comanda nu este acceptată de aparatul conectat la bus.

Linii de gestiune generală, fiecare din cele cinci linii ale acestei grupe având o funcție de comandă specifică între controler și alte aparate ale

sistemului:

- **ATN (ATteNtion)**; permite controlerului să indice instrumentelor că instrucțiuni și adrese sau date sunt prezente pe bus. Atunci când $ATN=1$ numai emițătorul și receptorul adresați sunt vizați. În acest caz, pe bus-ul de date este trimis codul adresei emițătorului. Dacă $ATN=0$, codul reprezintă date. Toate aparatele trebuie să “privească” în orice moment la această linie și atunci când pe ea se produce o schimbare, ele trebuie să răspundă într-un interval de max. 200 ns.
- **IFC (InterFace Clear)**; această linie poate fi pusă pe “1” (sau “adevărat”) numai de către controler înainte de a pune interfețele conectate la bus într-o stare inactivă. Toate operațiunile în curs sunt atunci oprite pentru a permite repornirea de la o situație neutră și uniformă înaintea tuturor operațiilor.
- **SRQ (Service ReQuest)**; această linie este activată de toate aparatele care au de cerut un “serviciu” controller-ului. Această cerere poate întrerupe o operațiune în curs. Atunci când mai multe aparate cer SRQ în același moment, controller-ul trebuie să efectueze o căutare pentru a depista aparatele respective și natura serviciului cerut. Această căutare se poate efectua fie “în serie”, fie “în paralel”.
- **EOI (End Or Identify)**; dacă $ATN=0$ (liniile DIO sunt pe modul “date”), această linie este activată (adică $EOI=1$ sau “adevărat”) de un emițător semnificând faptul că octetul în curs este ultimul transmis pentru operație. Dacă $ATN=1$ (liniile DIO sunt în modul “comandă interfață”), controller-ul activează linia EOI într-o căutare paralelă.
- **REN (Remote ENable)**; această linie este activată numai de controler pentru a comuta un aparat de la poziția “comandă de pe panoul frontal” (sau comandă locală) pe poziția “comandă de la distanță”. Atunci când $REN=0$, aparatul revine la modul de comandă locală.

Se disting, astfel, două tipuri de mesaje :

- comenzi generale, date de controler și destinate interfețelor încorporate aparatelor, acestea fiind mesaje de interfață.
- datele destinate aparatelor de măsurare înainte de a le plasa într-o stare particulară sau de primire a rezultatului obținut de la o măsurare. Datele sunt furnizate de un emițător care poate fi controller-ul; acestea sunt “mesaje pentru aparate”.

În acest fel se delimitează zonele de intervenție ale normei IEEE-488 care acționează prin comenzi asupra interfeței și prin mesaje asupra aparatelor.

Mesajele pentru aparate

Un mesaj pe cele 8 linii DIO este un mesaj de aparat (sau de date) dacă linia $ATN=0$. Aceste date sunt emise de aparatul adresat ca vorbitor și primite

de aparatele adresate ca ascultători sub controlul procedurii de tip “*handshake*”.

Ele pot fi:

a) date de intrare:

- date de comandă, de exemplu instrucțiuni ale programului pentru un aparat particular;
- date pentru afișare sau stocare.

b) date de ieșire:

- datele unui rezultat al măsurării;
- informații de stare a aparatului.

Între aparatele conectate la sistemul de comunicare, este necesar să existe convenții asupra codificării datelor. Se poate remarca faptul că norma definește numai modul transferului de date, dar nu și conținutul lor; acesta depinde de aparatul utilizat; la ora actuală, convenția cea mai răspândită este codul ASCII.

Mesaje de interfață

Se disting, ca și în cazul mesajelor de aparat, mesaje de interfață multifilară și unifilară. Mesajele unifilare ATN, IFC și REN care comandă anumite funcții ale aparatelor conectate la bus-ul HPIB au fost deja definite anterior. Așa cum s-a arătat, este vorba de comenzi emise de controler interfețelor aparatelor. Un mesaj multifilar pe linia DIO este considerat ca o comandă dacă linia ATN = 1. El este codificat prin 7 biți trimiși pe liniile DIO1÷DIO7.

2.3.2.2 INTERFAȚA CENTRONICS

Interfața dezvoltată de firma **Centronics** în scopul, inițial, al transmiterii comenzilor către imprimante, nu este (încă) normată, cu toate că este utilizată de toate PC-urile. Modul de lucru este caracterizat de un transfer paralel al datelor, distanța maximă dintre echipamentele interconectate prin această interfață fiind de 8 m, datorită limitărilor privind distorsionarea semnalelor cauzată de capacitatea lineică a conductoarelor. O soluție este dispunerea conductoarelor de semnal alături de conductoarele de masă și răsucirea acestora (*twisted-pair*), dar mulți producători de imprimante recomandă o distanță maximă între PC și acestea de 3m. Viteza de transfer a datelor prin intermediul acestei interfețe este dependentă de hardware. Ea poate, teoretic, să aibă valori de peste 1 MByte/s, dar pentru aceasta se impune o distanță maximă între echipamentele interconectate de 1 m. Interfața utilizează nivele de tensiune TTL, ceea ce facilitează utilizarea ei în diferite aplicații.

În cele ce urmează se va face o descriere a semnalelor specifice interfeței

(fig. 2.28):

- **Strobe** (activ LOW): această linie este activată de către calculator, atunci când se dorește transferul datelor către exterior (imprimanta, eventual);

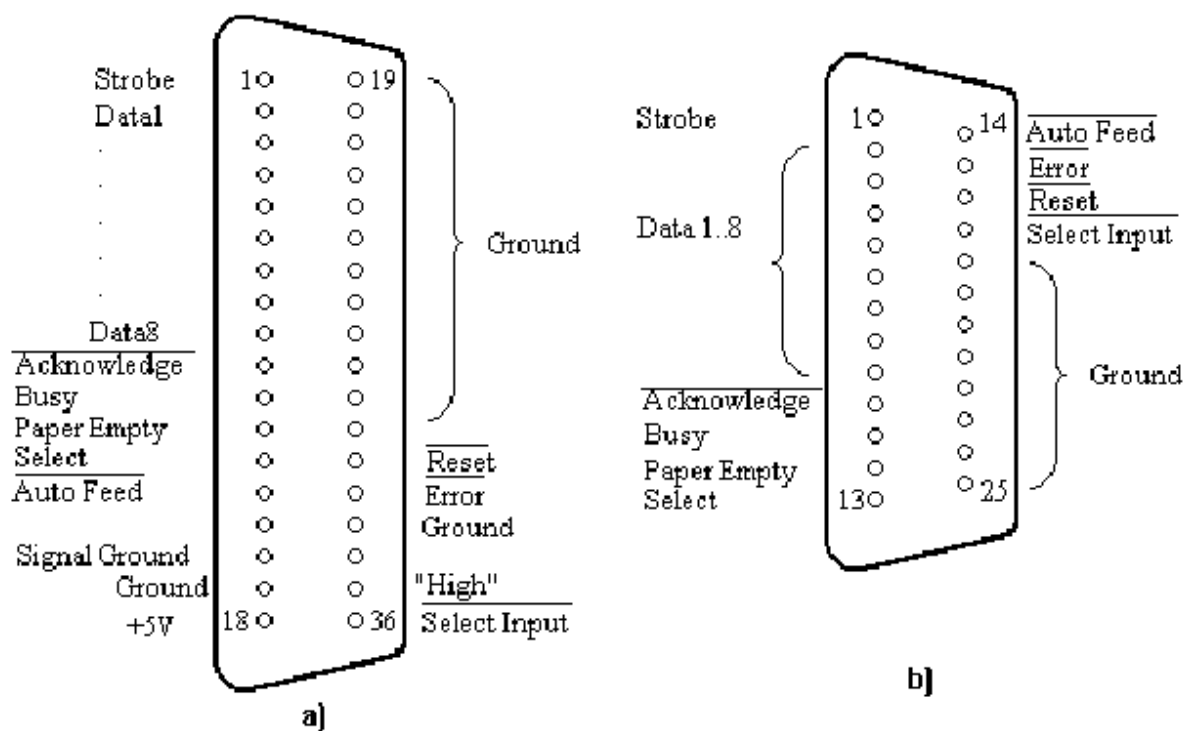


Fig. 2.28 Dispunerea pinilor portului Centronics:

a) conector cu 36 de pini (Amphenol-seria 57); b) conector cu 25 de pini (Subminiatur-D)

- **Data 1 ÷ Data 8:** linii de date
- **Acknowledge** (activ LOW): atunci când echipamentul exterior (imprimanta) a preluat datele transmise, transmite un semnal de înștiințare cu durată de 30 μ s;
- **Busy:** apariția unei erori în timp ce imprimanta preia datele, este în procesul de imprimare sau în starea off-line, determină activarea acestui semnal.
- **Paper Empty:** acest semnal este activ până când senzorul va detecta, din nou, prezența colilor de hârtie.
- **Select:** cu ajutorul acestui semnal imprimanta anunță faptul că este accesată și activă.

Următoarele semnale nu aparțin, de fapt, standardului (în sensul de variantă inițială a interfeței):

- **Autofeed** (activ LOW): o dată cu activarea acestei linii, imprimanta inserează la sfârșitul fiecărei linii câte un salt la linie nouă;

- **Reset** (activ LOW): cu semnalul trimis pe această linie, imprimanta se va seta într-o stare predefinită;
- **Error** (activ LOW): acest semnal se activează o dată cu apariția unei erori sau atunci când imprimanta este în starea off-line;
- **Select Input** (activ LOW): transmiterea unui semnal pe această linie determină selectarea imprimantei.

2.3.2.2.1 PROTOCOLUL DE COMUNICAȚIE CENTRONICS-HANDSHAKE

În fig. 2.29 este reprezentată diagrama de semnale corespunzătoare transmiterii datelor prin interfața Centronics. Procedeul *handshake* este caracterizat de următoarea succesiune a semnalelor: după cel puțin 500 ns de la transmiterea datelor pe liniile de date (8) ale magistralei, aparatul emițător (aici, PC-ul) va transmite un semnal de preluare (*Strobe*) de durată minimă de 500 ns. După alte cel mult 500 ns, imprimanta semnalizează prin intermediul semnalului de *Busy*, faptul că este în procesul de preluare a datelor primite. Acest semnal poate persista mai mult timp dacă, de exemplu, buffer-ul imprimantei este plin și trebuie așteptată efectuarea imprimării pentru golirea lui și preluarea unui nou set de date. Apoi (după cel mult 10μs de la inactivarea semnalului de *Busy*), devine activ semnalul de *Acknowledge* (varianta Centronics). În varianta Epson, acest semnal apare cu aproximativ 7 μs înainte de frontul descrescător al semnalului *Busy*.

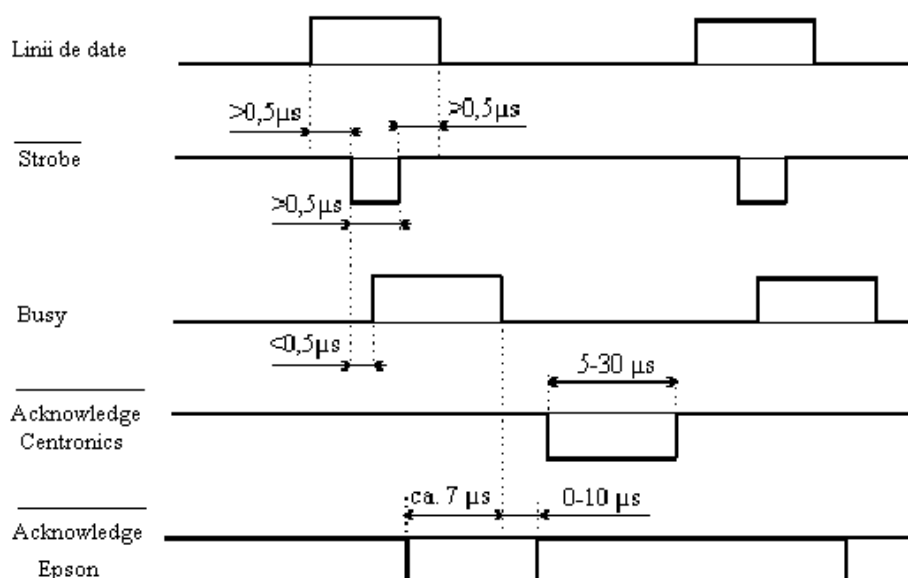


Fig. 2.29 Diagrama semnalelor pentru protocolul de comunicație pentru interfața Centronics.

În cazul în care se utilizează imprimanta conectată la portul serial al calculatorului, sau atunci când există mai multe porturi paralele (extrem de rar) sau dacă pur și simplu nu se conectează nici o imprimantă, atunci portul paralel disponibil poate fi utilizat pentru comanda circuitelor de măsurare, reglare și comandă. În cazul PC-urilor, interfața paralelă este accesată prin intermediul următoarelor *adrese de port*:

- LPT1: 3BC H ÷ 3BE H
- LPT2: 378 H ÷ 37A H
- LPT3: 278 H ÷ 27A H

3. TIPURI DE SISTEME DE ACHIZIȚII DE DATE

3.1 SISTEM DE ACHIZIȚII DE DATE CU MULTIPLEXARE TEMPORALĂ

Cea mai simplă structură de sistem de achiziții de date (fig. 3.1) presupune utilizarea *multiplexării temporale*. Diferitele semnale analogice de pe cele n canale sunt multiplexate la intrarea circuitului de eșantionare - memorare care reține, de fiecare dată, valoarea unui eșantion, în vederea conversiei.

În acest proces, *circuitul de eșantionare-memorare* realizează o dublă funcție:

- menține constant (în limite de cel mult $\pm 0,5$ **LSB** - **Least Significant Bit** - bitul cel mai puțin semnificativ) semnalul achiziționat la intrarea convertorului analog-digital;
- permite o utilizare cât mai eficientă a timpului de achiziție, comutarea următorului canal putând avea loc pe durata cât circuitul de eșantionare-memorare se găsește în starea de memorare și tensiunea sa de ieșire este supusă procesului de conversie analog-digitală.

Multiplexorul analogic permite utilizarea unui singur convertor analog-digital, pentru n canale analogice; de regulă n este de forma 2^k . Multiplexorul este o componentă electronică, conținând $n = 2^k$ comutatoare analogice, ale căror ieșiri sunt conectate împreună, pentru a furniza ieșirea unică a multiplexorului; numărul de comutatoare determină numărul de intrări ale multiplexorului. Comanda de închidere și de deschidere a comutatoarelor analogice este efectuată prin intermediul a $\log_2 n = k$ intrări de selecție.

Când dinamica procesului de colectare a datelor permite acest lucru, se poate mări numărul de canale analogice de intrare, care se pot conecta, prin multiplexare, la același lanț de conversie, folosind *structuri de tip arbore de multiplexoare*, deoarece multiplexoarele semiconductoare analogice sunt disponibile doar în combinații **2:1, 4:1, 8:1, 16:1**.

Din fig. 2.1, se remarcă *funcțiunile unității centrale*. Aceasta trebuie să asigure:

- semnalul de comandă a circuitului de eșantionare-memorare;
- semnalul de inițiere a conversiei (**START**) a convertorului analog-digital; la sfârșitul fiecărei conversii, convertorul **CA/D** furnizează un semnal, **EOC** (**End of Conversion** - sfârșitul conversiei), pentru a semnala în exterior că ieșirea numerică este disponibilă și stabilă;
- semnalele de selecție a canalului pentru multiplexor, adresarea canalelor de intrare putând fi făcută fie secvențial, fie aleatoriu.

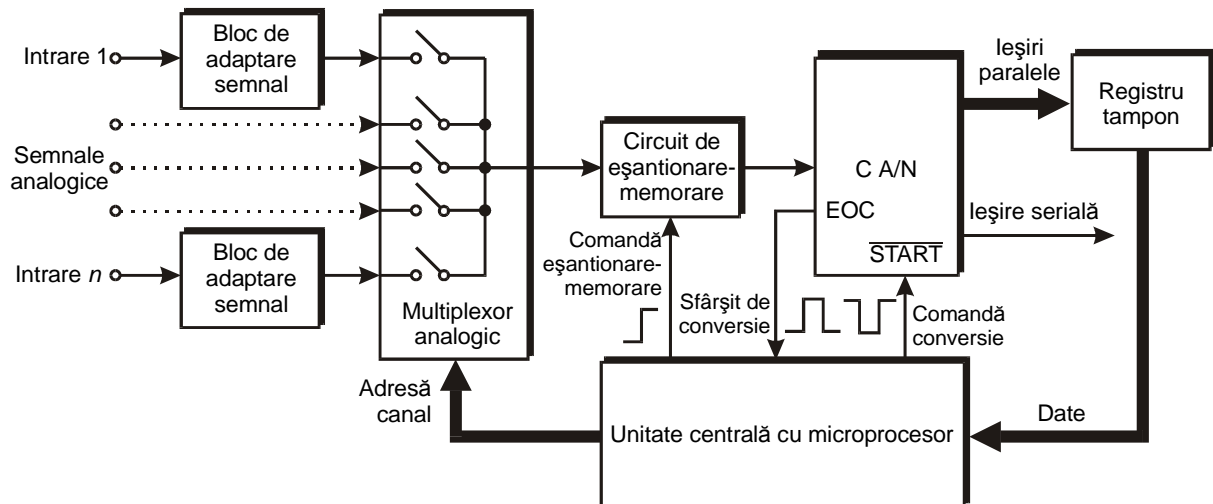


Fig. 3.1 Sistem de achiziții de date cu multiplexare temporală.

Timpul de achiziție, pentru un canal i , T_{ACH}^i poate fi determinat astfel:

$$T_{ACH}^i = t_{E/M} + t_{MUX} + t_C + t_{MEM}; i = 1 \div n \quad (3.1)$$

în care semnificația mărimilor ce intervin este următoarea:

- $t_{E/M}$ - timpul de eșantionare-memorare, compus din timpul de comandă și timpul de achiziție;
- t_{MUX} - timpul de multiplexare, compus din timpul de comandă și din timpul de stabilire al multiplexorului;
- t_C - timpul conversie, compus din timpul de comandă și din timpul de conversie analog-digitală;
- t_{MEM} - timpul memorare a rezultatelor, rezultat ca suma timpilor de execuție a unei instrucțiuni de citire a rezultatelor de la convertorul analog-digital și a unei instrucțiuni de scriere a rezultatelor în memorie.

Pentru n canale de intrare, timpul de eșantionare, T_e , se calculează astfel:

$$T_e = \sum_{i=1}^n T_{ACH}^i = n \cdot (t_{E/M} + t_{MUX} + t_C + t_{MEM}) \quad (3.2)$$

Sistemul cu multiplexor analogic, care permite accesul secvențial al semnalelor, are dezavantajul unei viteze reduse de măsurare, soluția fiind, în schimb, cea mai ieftină.

Frecvența maximă de eșantionare, f_e , a acestui sistem de achiziții de date rezultă:

$$f_e = \frac{1}{T_e} \quad (3.3)$$

de valoare relativ mică, ceea ce conduce la observația, conform teoremei eșantionării a lui Shannon:

$$f_e \geq 2 \cdot f_{\max} \quad (3.4)$$

că *această arhitectură de sistem de achiziții de date nu poate fi folosită în mod eficient pentru monitorizarea unor semnale de frecvență mare, sau rapid variabile în timp.*

Trebuie, însă, precizat faptul că, de obicei, unitatea centrală nu este proprie sistemului de achiziții de date, ci este mult mai corect să vorbim de o interfață de achiziții de date compatibilă cu un sistem de calcul compatibil **IBM - PC XT/AT**, acest sistem constituind unitatea centrală de prelucrare.

Un exemplu reprezentativ de *sistem de achiziții de date cu multiplexare temporală* este constituit de sistemul **DAS 1600**, produs de firma **Keithley**.

Acest sistem de achiziții de date se compune din:

- *interfață de achiziții de date;*
- *microsistem de calcul*, compatibil **IBM PC/AT**, organizat în jurul unei magistrale de tip **ISA**³.

Schema bloc a interfeței de achiziții de date **DAS 1600** este prezentată în fig. 3.2.

Interfața **DAS 1600** permite achiziția a **16** semnale de intrare analogice nediferențiale, sau a **8** semnale analogice de intrare complet diferențiale, în gama $\pm 5\text{V}$, frecvența maximă de eșantionare pentru un canal fiind de **100 kHz**.

Selecția modului de lucru (nediferențial sau diferențial) se face printr-o configurare *hardware*. Semnalele de selecție a canalului curent eșantionat sunt asigurate de o logică de comandă și incrementare, constituită dintr-un numărător și circuite auxiliare. Această logică permite atât baleierea secvențială, cât și aleatoare, a canalelor de intrare.

Ieșirea *multiplexorului analogic* este aplicată unui *amplificator de instrumentație*, a cărui amplificare este selectabilă *software*, funcție de nivelul semnalului de intrare. Amplificarea este comandată prin intermediul unui cuvânt de doi biți, astfel încât la ieșirea amplificatorului de instrumentație, nivelul semnalului să fie adus în gama $\pm 5\text{ V}$. Nivelul optim al amplificării este stabilit prin utilizarea unui *registru de stare*, registru care conține și informații despre canalul de intrare selectat, starea procesului de conversie, etc. De asemenea, un circuit auxiliar permite selectarea polarității semnalului de intrare.

Conversia analog-digitală este realizată pe **12** biți, rezoluția asigurată fiind de **2,44 mV**.

Interfața **DAS 1600** dispune de *două convertoare digital-analogice* cu multiplicare, pe **12** biți, a căror tensiune de referință și polaritate a ieșirii pot fi selectate *hardware*. Tensiunea de referință poate fi furnizată fie de o sursă de referință de **-10 V** sau **-5 V**, fie de o tensiune externă.

Transferul datelor, sub formă numerică în complement față de doi, către sistemul de calcul se face prin intermediul unei interfețe **DMA**⁴, al cărei nivel de

³ **ISA** - Industrial Standard Architecture - arhitectură industrială standard

⁴ **DMA** - Direct Memory Access - logică de acces direct la memorie.

prioritate poate fi configurat *hardware*.

Subsistemul numeric al interfeței de achiziții de date DAS 1600 este divizat în trei părți importante:

- *logica de control*, care include:
 - *registrul de date* al convertorului analog-digital și al multiplexorului analogic de intrare;
 - *logica de comandă și incrementare* a multiplexorului analogic;
 - *registrul de stare*;
 - *registrul de control* (comandă);
 - *logica de selecție a modului de trigger*-are a procesului de achiziție;
 - *logica de ceas* cu frecvența de 10 MHz;
 - *logica de selecție și decodificare*;
 - *logica de gestionare a întreruperilor*;
 - *logica de comandă a canalului DMA*,
are rolul de a gestiona desfășurarea procesului de achiziție;
- *interfețele programabile de intrare-ieșire*, constituite din:
 - *trei contoare programabile* de **16** biți, organizate într-un circuit **8254** și folosite pentru selectarea frecvenței de eșantionare, efectuarea unui număr prestabilit de eșantioane;
 - *trei porturi paralele, bidirecționale*, dispunând de **8** biți, organizate într-un circuit **8255** și folosite ca linii digitale de intrare-ieșire;
 - *un registru de intrare* de **4** biți, folosit pentru sincronizarea externă a comenzilor de achiziție;
 - *un registru de ieșire* de 4 biți;
- *logica de interfață cu magistrala sistemului de calcul*, ce înglobează circuite de tip tampon pentru liniile de date.

Pentru alimentare, interfața **DAS 1600** necesită o singură tensiune de **+5V**. Celelalte tensiuni, **±15 V**, necesare funcționării sunt asigurate de un convertor curent continuu - curent continuu, cu care este echipată interfața. Tensiunea de referință, cu valoarea de **-5V**, este asigurată de sursa internă, implementată în convertorul analog-digital.

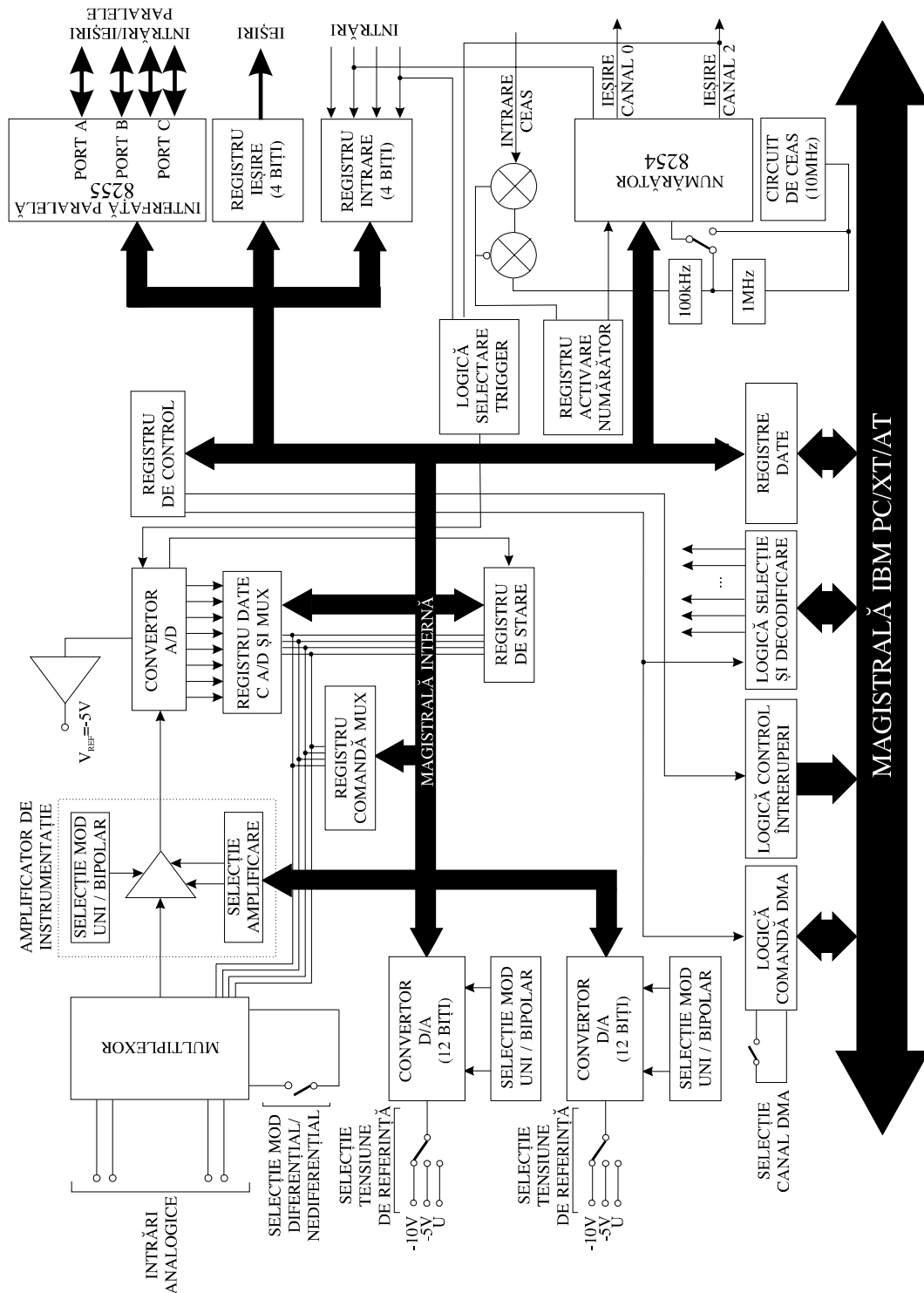


Fig. 3.2 Schema bloc a interfeței de achiziții de date DAS-1600.

3.2 SISTEM DE ACHIZIȚIE SINCRONĂ DE DATE

În arhitectura sistemului de achiziții de date, reprezentat în fig. 3.3, se remarcă amplasarea, pe fiecare canal de intrare, a câte unui circuit de eșantionare-

memorare, **E/M**, în amonte față de multiplexor. Comanda pentru trecerea în stare de memorare este dată simultan, pentru toate circuitele **E/M**, după care ieșirile acestora sunt multiplexate la intrarea convertorului **CA/D**. Multiplexarea canalelor analogice de intrare se poate face fie secvențial, fie aleator.

Deoarece timpul de așteptare, în vederea conectării la intrarea convertorului **CA/D**, poate fi destul de lung, circuitele de **E/M** trebuie să prezinte o rată redusă de alterare a tensiunii memorate.

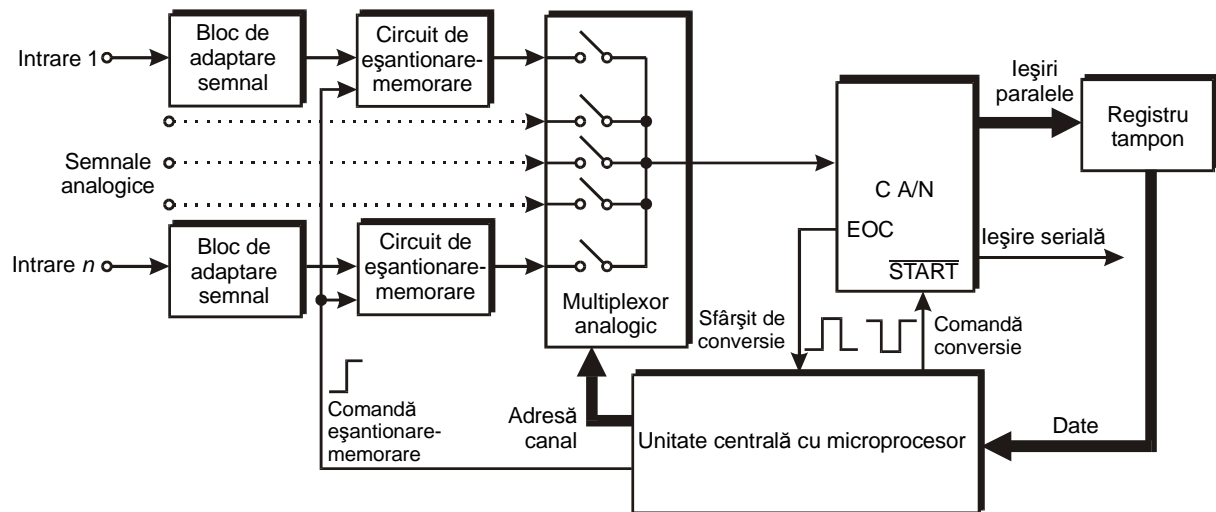


Fig. 3.3 - Sistem de achiziție sincronă de date.

Timpul de eșantionare, T_e , pentru n canale analogice de intrare rezultă:

$$T_e = \sum_{i=1}^n T_{ACH}^i = t_{E/M} + n \cdot (t_{MUX} + t_C + t_{MEM}) \quad (3.15)$$

Se constată scăderea timpului de eșantionare în raport cu acela corespunzător arhitecturii cu multiplexare temporală, însă, cu toate acestea T_e , timpul de eșantionare, depinde încă, în mod dramatic, de numărul de canale de intrare, cu care este prevăzut sistemul de achiziții de date.

Ca exemplu, în continuare este prezentată arhitectura interfeței de achiziții de date **EISA-2000**, produsă de firma **National Instruments**.

Acest *sistem rapid de achiziții de date*, produs de firma **National Instruments**, este realizat sub forma unei plăci echipate cu un conector compatibil cu magistrala **EISA**⁵ a sistemelor de calcul compatibile **PC**. Interfața dispune de *patru canale analogice de intrare*, fiecare echipat cu propriul circuit de eșantionare-memorare. Viteza maximă de eșantionare este:

- **1 MHz**, în cazul eșantionării pe un singur canal;
- **500 kHz**, în cazul eșantionării simultane a două canale;

⁵ **EISA** - Extended Industrial Standard Architecture - arhitectură industrială standard extinsă.

- **250 kHz**, în cazul eșantionării simultane a celor patru canale analogice de intrare.

Procesul de achiziție a datelor, pentru fiecare canal analogic de intrare, poate fi declanșat cu ajutorul unor circuite de tip *trigger*⁶ de tip *software*, analogic (nivelul analogic de *trigger*-are este programabil pe **10** biți, în intervalul (-5,12...+5,12)V, cu posibilitatea de alegere a polarității) sau digital. Deoarece întregul proces de configurare și de calibrare este controlat prin program, nu este necesară intervenția asupra sistemului de calcul în care a fost instalată interfața de achiziții.

Interfața rapidă de achiziții de date **EISA-A2000** utilizează sistemul de integrare în timp real **RTSI**⁷, elaborat de firma **National Instruments**, pentru a sincroniza funcționarea mai multor plăci de achiziții de date, de tip **EISA** sau **AT**, instalate într-un sistem de calcul. Prin utilizarea unui *controller DMA*, interfața **EISA-A2000** poate transfera datele direct în memoria sistemului de calcul în modul *burst* (“rafală”) cu rate de maxim **16,5** Mocteți/secundă.

Ca domenii generale de aplicații, interfața **EISA-A2000** poate fi utilizată în aplicații de laborator sau industriale. Cele patru intrări analogice rapide, cu rezoluție de **12** biți, permit utilizarea interfeței pentru analiza, de înaltă precizie, de semnal, pentru analiza regimurilor tranzitorii, etc. Eșantionarea sincronă multicanal este deosebit de utilă în cazul analizei fazei a mai multor semnale.

STRUCTURA HARDWARE A INTERFEȚEI DE ACHIZIȚII DE DATE EISA-A2000

În fig. 3.4 este prezentată schema bloc a interfeței de achiziții de date **EISA-A2000**.

Circuitele analogice de intrare: Circuitele de eșantionare-memorare și convertorul analog-digital, cu rezoluție de **12** biți, permit digitizarea unui eșantion o dată la o microsecundă. Banda maximă de frecvență a semnalelor de intrare este pentru semnal mic de **4** Mhz, iar de semnal mare de **1,1** MHz. Gama tensiunilor de intrare acceptate de sistem este de **±5V**. Selecția tipului de semnal de intrare (**AC**⁸/**DC**⁹) se face prin program. În cazul opțiunii **AC**, intrările analogice asigură o rejecție a semnalelor continue de **±30V**. Toate canalele analogice de intrare sunt echipate cu circuite de protecție la supratensiuni de intrare de până la **±30V**, atât în condiții de funcționare a interfeței, cât și dacă interfața nu este alimentată.

Interfața de achiziții **EISA-A2000** asigură o rezoluție de **12** biți, ceea ce este echivalent cu o rezoluție analogică de **2,44mV**. Este posibilă creșterea

⁶ **Trigger** - circuit de declanșare.

⁷ **RTSI** - **Real-Time System Integration** - interfață de integrare sistem, în timp real.

⁸ **AC** - **Alternative Current** - semnal variabil în timp după o lege sinusoidală.

⁹ **DC** - **Direct Current** - semnal continuu, invariabil în timp.

rezoluției efective peste **12** biți, folosind generatorul *Gauss Dither*, implementat pe placă, și medierea eşantioanelor achiziționate. Rezoluția este automat crescută la **16** biți, rezultatul fiind reprezentat în complement față de doi.

Circuitele de autocalibrare: Interfața **EISA-A2000** conține un nucleu *software* de autocalibrare a circuitelor de intrare analogice. Sursa de referință internă asigură o înaltă precizie și stabilitate în timp și cu temperatura. Sunt, de asemenea, disponibile circuite pentru reglajul de offset pentru fiecare canal, asigurând scăderea erorii de offset la mai puțin de **0,25 LSB**. Procesul de calibrare nu necesită conexiuni exterioare. Datele necesare pentru efectuarea autocalibrării sunt memorate într-un circuit de memorie **E²PROM**, amplasat pe placă.

Circuitele analogice și digitale de trigger-are: **EISA-A2000** dispune atât de circuite analogice, cât și digitale de *trigger*-are pentru declanșarea procesului de achiziții de date. Există trei moduri pentru declanșarea convertorului analog-digital, **CA/D**:

- în *modul analogic de trigger*-are, procesul de achiziție este demarat de un semnal analogic de intrare, atunci când atât nivelul semnalului, cât și polaritatea acestuia, corespund valorilor programate. Un convertor digital-analog generează tensiunea de prag, cu o rezoluție de **10 mV**. Se compară nivelul tensiunii de intrare cu această tensiune de prag programată; conversia este declanșată la egalitatea celor două tensiuni și este, de asemenea, îndeplinită condiția de polaritate. Semnalul de *trigger* analogic poate fi selectat de la oricare dintre cele patru canale analogice de intrare sau de la o intrare externă;
- în *modul digital de trigger*-are, există opțiunea de a selecta prin program care dintre fronturile semnalului, crescător sau scăzător, va declanșa procesul de achiziție;
- în *modul software de trigger*-are, procesul de achiziții de date este declanșat prin program.

Circuitele de secvențializare și de comandă a conversiei prezintă trei moduri de *trigger*-are pentru procesul de achiziții de date: *modul de lucru cu pretrigger*-are, *modul de lucru cu întârziere față de semnalul de trigger*-are și *modul de lucru cu posttrigger*-are.

În *modul de lucru cu pretrigger*-are, procesul de achiziție este declanșat prin *software*; acesta continuă, depunând rezultatele într-un *buffer* circular, până când se primește un semnal de trigger analogic sau digital.

În cel de-al doilea mod de lucru, se poate selecta, prin program, un timp de întârziere, pe durata căruia procesul de achiziție este inhibat, de la primirea unui semnal *trigger* analogic sau digital.

În *modul de lucru posttrigger*, achiziția începe după ce interfața **EISA-A2000** primește un semnal *trigger* analogic, digital sau *software*.

Prin combinarea modurilor de lucru, se poate achiziționa un număr pre-

programat de eşantioane, înainte sau după îndeplinirea unei condiții de *trigger*-are.

Logica de comandă și timing a achiziției: Aceste circuite generează semnalele de secvențializare (*timing*) și de comandă a procesului de achiziție. *Timing*-ul de conversie multiplă analog-digitală este comandat fie de un circuit de tip numărător, implementat pe interfața **EISA-A2000**, fie de un ceas extern de eşantionare.

Circuitul de numărare, pe **16** biți, cu care este echipată placa, generează *timing*-ul necesar pentru conversia analog-digitală. Acest circuit dispune de baze de timp, selectabile *software*, de **200ns**, **1μs**, **10μs**, **100μs**, **1ms** și **10ms**. Intervalul minim de eşantionare, în cazul operării monocanal, este de **1μs**. Dacă este necesar în cadrul aplicației, numărătorul intern poate fi înlocuit cu un semnal de ceas de eşantionare extern. Numărul de eşantioane prelevate în cadrul aplicației, pe fiecare canal, este monitorizat de un numărător pe **32** de biți, care oprește procesul de achiziție la atingerea numărului preprogramat de eşantioane.

Interfața de magistrală RTSI: **EISA-A2000** este interfațată cu magistrala **RTSI** National Instruments printr-un comutator, de fapt o rețea bidirecțională de porți de comutare de tip *crossbar*. Liniile magistralei **RTSI** includ semnalele externe de *trigger*, semnalul de ceas extern de eşantionare și semnale de intrare-ieșire de numărare. Folosind aceste linii de semnal, mai multe interfețe **EISA-A2000** pot fi sincronizate să achiziționeze sincron semnalele analogice de intrare.

Interfața de magistrală EISA: Interfața de achiziții de date este complet compatibilă cu magistrala **EISA**, putând manipula liniile de cerere **DMA** de pe această magistrală. Un circuit *buffer*, cu capacitate de **512** cuvinte, previne pierderea de informații în condițiile în care datele nu pot fi transferate imediat în memorie. Astfel, mai multe interfețe **EISA-A2000**, cuplate în sistemul de calcul, pot funcționa la întreaga capacitate.

Interfața cu magistrala **EISA** dispune, de asemenea, de linii de întrerupere, logică **DMA**, registre de comandă și de stare. Această interfață este astfel proiectată încât permite ca mai multe sisteme **EISA-A2000** să poată partaja magistrala sistemului de calcul, prin transferarea datelor în “rafală” cu rate extrem de ridicate, lăsând suficiente posibilități de servire a celorlalte resurse ale sistemului.

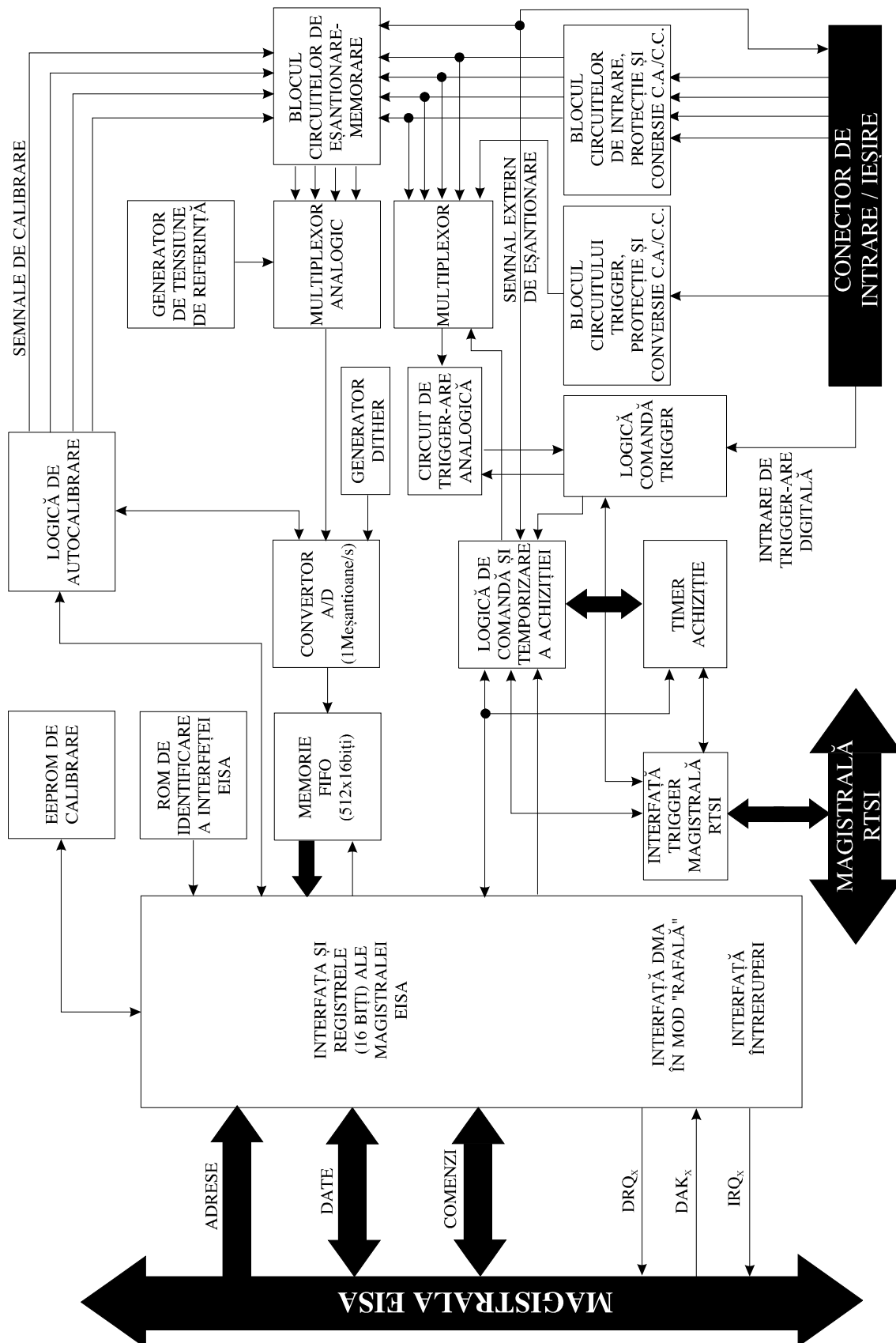


Fig. 3.4 Schema bloc a interfeței de achiziții de date EISA-2000.

3.3 SISTEM RAPID DE ACHIZIȚII DE DATE

În cazul în care semnalele de intrare evoluează rapid în timp, cele două arhitecturi precedente, care se bazează pe un singur **CA/D** pentru conversie, nu mai sunt utilizabile.

În consecință, pentru mărirea vitezei de măsurare, se utilizează arhitectura de sistem prezentată în fig. 3.5, care are câte un convertor **CA/D**, pentru fiecare canal, precedat de elemente de eșantionare - memorare, **E/M**.

Structura sistemului, așa cum este prezentată în fig. 3.5, în care cele n intrări analogice sunt conectate împreună, permite folosirea *tehnicii de supraeșantionare* pentru achiziția semnalului de intrare. În această configurație, *viteza de achiziție este practic multiplicată de n ori față de aceea care se obține prin eșantionarea secvențială*.

Informațiile, de la ieșirile convertoarelor analog-digitale, sunt aplicate unui **multiplexor numeric**, care selectează datele primite și le transmite secvențial pe magistrala sistemului de calcul. Trebuie precizat că marea majoritate a convertoarelor analog-digitale realizate în momentul actual sunt astfel concepute încât înglobează *circuite de interfață cu un microprocesor pe 8/16 biți*. Aceste circuite de interfață constau în implementarea internă a unor registre cu ieșiri cu trei stări, pentru preluarea rezultatelor și a unor semnale de dialog specifice magistralei unui microprocesor. Echiparea ieșirilor circuitului de conversie cu registre cu trei stări permite eliminarea multiplexorului numeric din structura prezentată anterior, legarea mai multor ieșiri cu trei stări la aceeași linie de date a magistralei sistemului implementând, de fapt, un multiplexor cablat.

Timpul de eșantionare, T_e , pentru n canale analogice de intrare, caracteristic acestei arhitecturi de sistem de achiziții este:

$$T_e = \sum_{i=1}^n T_{ACH}^i = t_{E/M} + t_C + n \cdot (t_{MUX} + t_{MEM}) \quad (3.6)$$

Trebuie să menționăm că, pentru această arhitectură de sistem de achiziții de date, timpul elementar de multiplexare, t_{MUX} , este corespunzător unui **multiplexor numeric** și este considerabil mai redus în comparație cu situațiile anterioare, deoarece timpul de stabilire al multiplexorului numeric este cu circa trei ordine de mărime mai mic decât cel al unui multiplexor analogic.

Avantajele acestei structuri de sistem de achiziții sunt următoarele:

- pot fi utilizate convertoare **CA/D** mai lente, și deci mai ieftine, chiar dacă se dorește o viteză mare de achiziție;
- prin conversia locală sub formă numerică, se asigură o bună imunitate la perturbații;
- posibilitatea separării galvanice a unei surse de semnal, împreună cu convertorul **CA/D** aferent, față de restul sistemului.

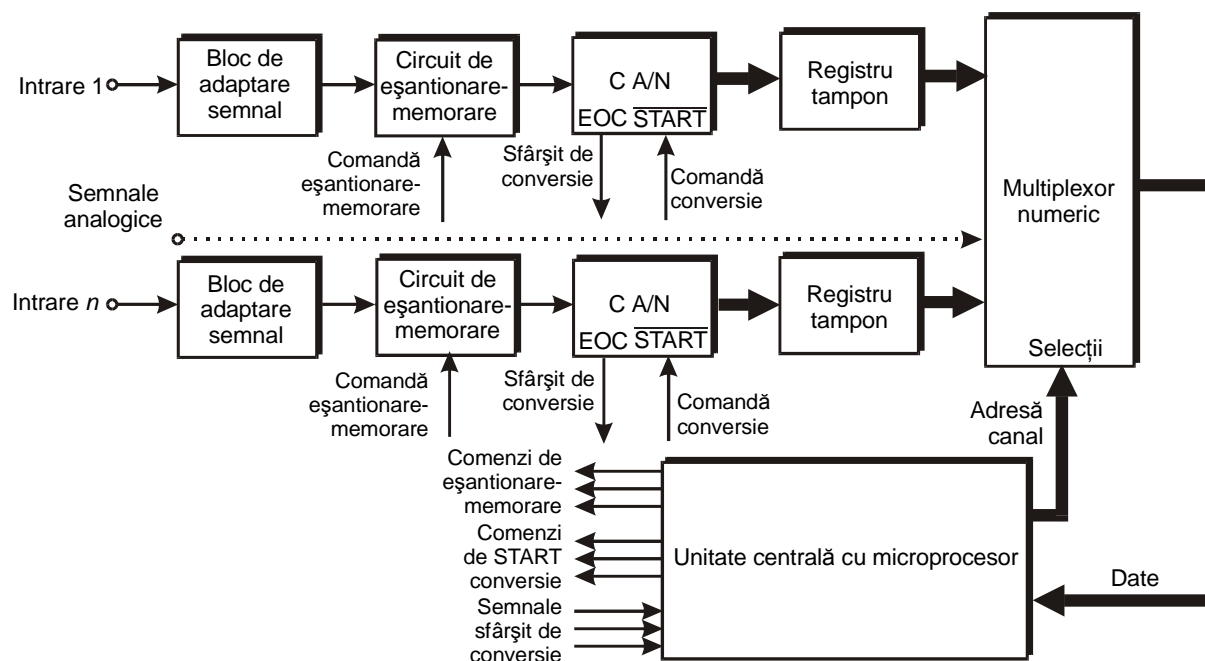


Fig. 3.5 - Sistem rapid de achiziții de date.

3.4 UNITATEA CENTRALĂ DE COMANDĂ

Unitatea centrală de prelucrare a unui sistem de achiziție de date trebuie să realizeze următoarele funcțiuni (fig. 2.1):

- selectarea canalului analogic, pe care se dorește să se facă achiziția;
- comanda eșantionării;
- comanda conversiei analog-digitale;
- sesizarea sfârșitului conversiei și citirea codului binar rezultat;
- încărcarea codului în memorie;
- corecția erorilor introduse de diferite blocuri componente;
- prelucrarea și afișarea datelor;
- testarea blocurilor componente în vederea identificării celor defecte.

Principial, sunt posibile două abordări:

- **logica cablată**, bazată pe circuite electronice cu grad redus de integrare, dar de viteză mare;
- **logica programată**, cu avantajul unei flexibilități și capacități de prelucrare a datelor foarte ridicate, dar cu o viteză mai redusă.

Logica cablată conduce la realizarea unei unități de comandă a achiziției, fără programe, deci fără parte *software*, folosind în exclusivitate resurse *hardware*, astfel conectate, încât să permită funcționarea achiziției în mod automat. Se folosește tehnologia **bipolară**, **MOS** (Metal Oxid Semiconductor - tehnologie metal-oxid-semiconductor) și **CMOS** (Complementary Metal Oxid Semiconductor - tehnologie metal-oxid-semiconductor, de tip complementar) de

realizare a circuitelor.

Logica programată se bazează pe simbioza dintre o parte electronică, fizică, așa-numitul **hardware**, care constituie suportul material al operațiilor de comandă și prelucrare și, pe de altă parte, programele, partea informațională, numită curent **software**; **software**-ul impune **hardware**-ului operațiile ce urmează să le desfășoare conform algoritmului proiectat de programator. Elementul central al **hardware**-ului este *microprocesorul*.

În practică, nu se utilizează niciodată, în exclusivitate, logica cablată, ci, fie o metodă combinată, fie doar logica programată.

3.5 SISTEME DE ACHIZIȚIE DE DATE CU MICROPROCESOR

Un sistem de achiziție de date, asociat cu un microsistem de calcul, se comportă ca un sistem **intelligent** (care poate lua decizii bazate pe informații anterioare, prelucrează informația, efectuează calcule, după care, pe baza rezultatelor obținute, adoptă o decizie, din mai multe soluții posibile).

3.5.1 UNITĂȚI CENTRALE DE PRELUCRARE TRADIȚIONALE

Microprocesorul reprezintă elementul funcțional esențial al microsistemului de calcul, fiind un circuit integrat pe scară largă, **LSI** (**L**arge **S**cale **I**ntegration - integrare pe scară largă), care poate realiza cinci funcții de bază:

- funcția de intrare (**INPUT**), care permite legătura dintre lumea exterioară și sistem;
- funcția de ieșire (**OUTPUT**), care permite legătura dintre sistem și lumea exterioară;
- funcția de memorare (**MEMORY**), care permite păstrarea informațiilor (date, rezultate) și, uneori, a instrucțiunilor programului;
- funcția de prelucrare (**COMPUTE**), implementată prin **ALU** (**A**rithmetic **L**ogic **U**nit - unitate aritmetico-logică), care permite efectuarea operațiilor aritmetice și logice din sistem;
- funcția de control (**CONTROL**), care înglobează totalitatea acțiunilor de secvențializare și control ale activității sistemului.

În interiorul microsistemului de calcul, informațiile sunt vehiculate prin intermediul *magistralei*. **Magistrala unui sistem de calcul** este compusă din trei secțiuni:

- *secțiunea de date* (magistrala de date), care asigură schimbul

bidirecțional de informație (date) între microprocesor, pe de o parte, și circuitele de memorie și de interfață, pe de altă parte;

- *secțiunea de adrese* (magistrala de adrese), unidirecțională, care asigură vehicularea biților de adresă, de la microprocesor sau de la un alt dispozitiv *master* către celelalte elemente ale sistemului;
- *secțiunea de control* (magistrala de control), unidirecțională, care permite vehicularea semnalelor de sincronizare și control, între microprocesor sau un alt dispozitiv *master* și celelalte resurse ale sistemului.

Marea diversitate de componente electronice, cu funcțiuni, performanțe și costuri foarte diferite, impun proiectantului o definire precisă a caracteristicilor și utilităților sistemului pe care îl concepe.

Alegerea microprocesorului și a circuitelor periferice asociate este dictată de analiza structurii *hardware* (constituite din componente de microinformatică, procesor, memorii, componente electronice clasice) și a structurii *software* (implementate în memorie), în vederea asigurării cerințelor de viteză de măsură și de versatilitate ale sistemului.

Gama actuală de microprocesoare cuprinde tipuri de **8, 16, 32 și 64** de biți; pe plan mondial ponderea o dețin, în continuare, microprocesoarele pe **8** biți, care oferă, la preț scăzut, performanțe satisfăcătoare.

Dintre microprocesoarele pe **8** biți, cele mai utilizate, cu performanțe bune, sunt tipurile: **INTEL 8080, INTEL 8085, ZILOG Z80, MOTOROLA 6800**. Dintre acestea, o largă utilizare o are, în aplicațiile curente, microprocesorul **ZILOG Z80**, care înglobează caracteristicile microprocesorului **INTEL 8080**, cu păstrarea compatibilității *software*, dar cu extinderea performanțelor acestuia.

În prezent, au început să fie utilizate pe scară largă *microcontroller*-ele pe **8 și 16** biți, care compensează o parte din dezavantajele microprocesoarelor tradiționale. Arhitectura unui *microcontroller* a fost astfel proiectată încât să ofere o *versatilitate* mult *superioară* celei a microprocesoarelor tradiționale, prin:

- *organizarea internă sub forma unui număr* de trei, patru sau cinci *porturi bidirecționale*, a căror funcționalitate poate fi stabilită de utilizator;
- *extinderea spațiului de adresare* prin separarea, din punct de vedere al accesului, a memoriei de date față de memoria de program;
- *înglobarea în structura microcontroller-ului a unei memorii de program* de tip **PROM** (Programmable Read-Only Memory - memorie programabilă, ce prezintă doar facilități de citire), cu capacitate relativ mică (în cele mai multe cazuri **4Kocteți**), ce poate fi parțial sau integral accesibilă operatorului pentru programul de aplicații. În cazul accesibilității parțiale a utilizatorului la această resursă, trebuie menționat faptul că într-o zonă a memoriei interne de program este rezident un interpretor **BASIC**, ceea ce permite programatorului să-și

- scrie aplicația în limbajul **BASIC** și nu în limbaj de asamblare;
- *înglobarea în structura internă a unor circuite de interfațare, cum ar fi interfețe seriale programabile de comunicație, interfețe paralele programabile bidirecționale, circuite de numărare programabile, convertoare analog-digitale, convertoare digital-analogice, etc;*
 - un astfel de **microcontroller** lucrează la o frecvență a ceasului de câteva ori mai mare decât un microprocesor tradițional, ceea ce permite *creșterea vitezei de prelucrare* aproximativ în aceeași măsură;
 - setul de instrucțiuni al unui **microcontroller** este substanțial îmbogățit, păstrând compatibilitatea cu cele ale microprocesoarelor tradiționale;
 - tehnologia de realizare **CMOS**, folosită pentru implementarea majorității **microcontroller**-elor actuale, permite *scăderea substanțială a puterii consumate* de la sursa de alimentare, *creșterea fiabilității și siguranței în exploatare*.

Ca exemplificare la cele menționate anterior, se constată utilizarea extensivă a două familii de microcontrollere: familia **8051**, concepută și realizată de firma **INTEL** și familiile **Z8⁷**, **Z87 Super** ale firmei **ZILOG**. Arhitectura generală a familiei **8051**, respectiv a familiei **Z8TM**, este prezentată în fig. 3.6, respectiv 3.7.

Ambele familii de microcontroller-e au elemente arhitecturale comune, cum ar fi:

- pentru minimizarea liniilor de interconexiune cu exteriorul, *secțiunea de date și secțiunea inferioară de adrese* ale magistralei sistemului *sunt multiplexate temporal*;
- atât **8051**, cât și **Z8** pot funcționa fie ca microsisteme specializate de intrare-ieșire, fie ca microsisteme specializate de lucru cu memoria;
- în ambele tipuri de aplicații, *spațiul de adresare este partajat în trei secțiuni*:
 - *memoria de program* (internă și externă), cu capacitate maximă de **64K**octeți;
 - *memoria de date* (externă), cu capacitate variind între **60** și **64** Kocteți. Este nevoie să se facă precizarea că dispozitivele de intrare-ieșire externe sunt adresate de **microcontroller** ca seturi de celule de memorie de date;
 - *setul de registre interne*, conținând registre de uz general, registrele unității centrale de prelucrare și registrele asociate celor patru porturi de intrare-ieșire ale **microcontroller**-ului;
- ambele familii de **microcontroller**-e permit operarea la nivel de bit, la nivel de octet sau octet codificat zecimal, sau la nivel de cuvânt cu lungime de **2** octeți;
- operarea în regim de intrare-ieșire poate fi coordonată prin întreruperi sau în modul **polling** (interogare periodică). Sunt disponibile cinci/șase

nivele de întreruperi, care pot fi mascate sau a căror prioritate poate fi modificată.

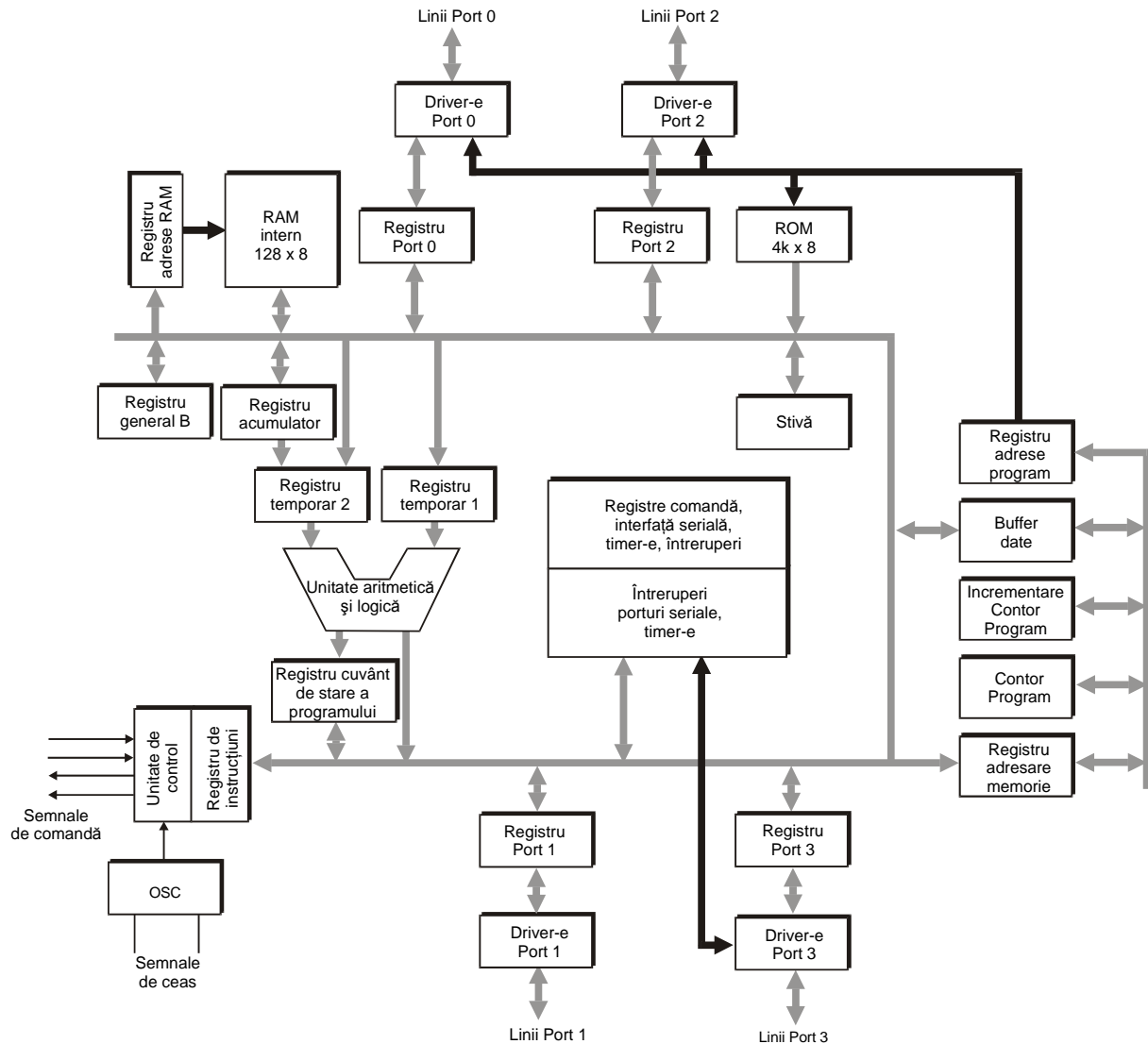


Fig. 3.6 - Arhitectura de bază a familiei de microcontroller-e INTEL 8051.

În afară de **microprocesor**, componenta cea mai complexă, ale cărei caracteristici determină, în esență, structura și performanțele unității centrale, se disting (fig. 3.14):

- *memoriile nevolatile;*
- *memoriile volatile;*
- *circuitele de interfațare;*
- *decodificatoarele.*

Memoriile nevolatile - cu conținut permanent: **ROM** (**R**ead-**O**nly **M**emory - memorie, cu conținut permanent, ce poate fi doar citită), **PROM**, **EPROM** (**E**rasable **P**rogrammable **R**ead-**O**nly **M**emory - memorie cu conținut permanent ce poate fi doar citită și care dispune de facilități de ștergere a conținutului cu

radiații ultraviolete), **EEPROM** (**E**lectrical **E**rasable **P**rogrammable **R**ead-Only **M**emory - memorie cu conținut permanent ce poate fi doar citită și care dispune de facilități de ștergere electrică a conținutului) - sunt destinate să stocheze programul pe care trebuie să-l efectueze *unitatea centrală de prelucrare*.

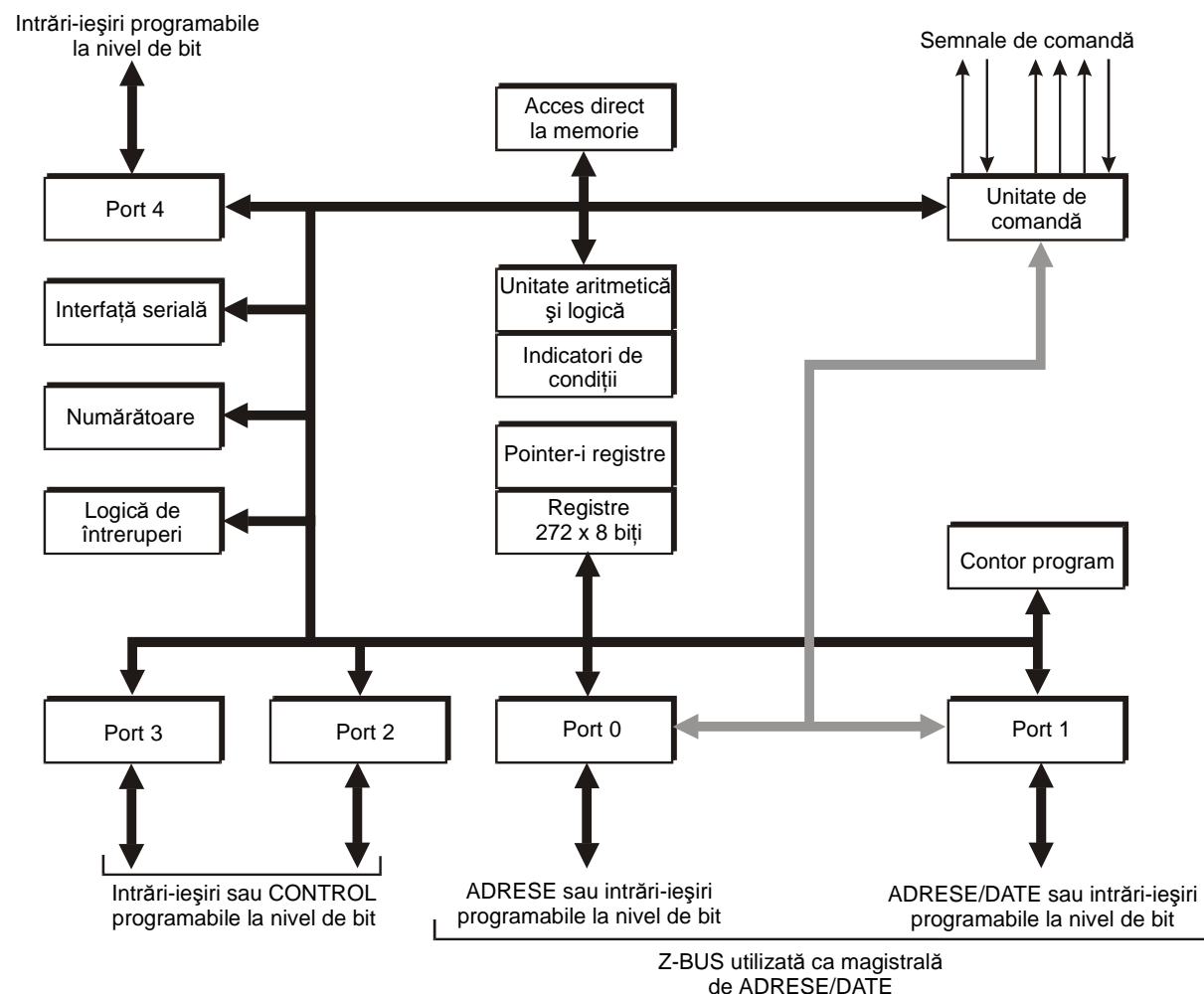


Fig. 3.7 Arhitectura de bază a familiei de microcontrollere Zilog Z8™ Super.

Memoriile volatile - cu conținut nepermanent: **RAM** (**R**andom **A**ccess **M**emory - memorie cu acces aleator), **SRAM** (**S**tatic **R**andom **A**ccess **M**emory - memorie cu acces aleator, având drept element de memorare un circuit basculant bistabil), **DRAM** (**D**ynamic **R**andom **A**ccess **M**emory - memorie cu acces aleator, având drept element de memorare un condensator) - sunt destinate să stocheze temporar eșantioanele, rezultatele parțiale și finale ale prelucrării, în timpul procesului secvențial de funcționare a sistemului de calcul.

Circuitele de interfațare sunt destinate să asigure comunicația microprocesorului cu echipamentele de intrare-ieșire. Prin intermediul porturilor de intrare, se citesc datele rezultate în urma conversiei analog-digiale, sau starea unui echipament periferic. Porturile de ieșire mijlocesc transferul de date de la unitatea de comandă la afișaj sau la alt calculator, respectiv sunt utilizate pentru

inițializarea și programarea echipamentelor periferice.

Decodificatoarele sunt utilizate pentru decodificarea adreselor și generarea unor semnale de selecție care se exclud reciproc (un singur semnal de ieșire al decodicatorului poate fi activ la un moment de timp dat). Ieșirile decodificatoarelor sunt active pe nivel coborât, pentru a se realiza cu mai multă ușurință interfațarea cu echipamentele externe. Decodificatoarele permit microprocesorului să selecteze resursele unității centrale, știind că microprocesorul nu poate executa decât o singură operație (citirea din memoria program, citirea sau scrierea datelor, citirea sau scrierea registrelor de comandă sau de stare ale interfețelor). În sistemele de calcul mai complexe, pentru generarea semnalelor de selecție, decodificatoarele sunt înlocuite cu circuite **PAL** (**P**rogrammable **L**ogic **A**rray - rețele logice programabile), circuite care implementează decodificatoare cu mai multe ieșiri (decodificatoarele integrate se realizează în următoarele configurații: **1:2**, **2:4**, **3:8** și **4:16**) și o logică complexă de condiționare a decodificării.

Configurația prezentată este absolut minimală, întrucât în majoritatea sistemelor există și alte circuite, ca de exemplu: *amplificatoare de magistrală unidirecționale și bidirecționale, circuite de tip registru cu trei stări, divizoare de frecvență*, etc.

Dacă *volumul de date* ce trebuie achiziționate și prelucrate este *considerabil*, se recomandă folosirea memoriei **RAM** dinamice, care permite o mare densitate de integrare, cu prețul unei viteze ceva mai reduse (timpul de acces este redus, de ordinul **50÷70 ns**, însă sunt necesare *cicluri de reîmprospătare* la nivel de pagini de memorie, la intervale de maximum **2 ms**, pentru ca informația memorată să nu fie afectată ca integritate).

Dacă este esențială viteza iar volumul de date este mic, se preferă utilizarea memoriilor **RAM** statice.

Datorită progreselor tehnologice înregistrate în ultimii ani, numeroase firme producătoare de componente, cum ar fi **INTEL**, **AMD**, **Micron**, **Hibrid Semiconductor**, realizează memorii **SRAM** de capacitate mare în tehnologie **CMOS** (de la **2Kx8** biți până la **128Kx8** biți, **256Kx16** biți), cu timp de acces cuprins între **20** și **35 ns** și performanțe deosebite de fiabilitate și cost. De asemenea, se realizează module de memorie **SRAM**, cu capacități cuprinse între **128Kx8** biți și **256Kx32** biți. Aceste circuite de memorie **SRAM**, respectiv modulele de memorie **SRAM**, sunt ideale pentru a echipa unitățile centrale de prelucrare, din punct de vedere al vitezei de acces, siguranței în exploatare, consumului deosebit de redus și gradului mare de integrare.

Funcționarea sistemului de achiziții de date cu microprocesor este următoarea:

- microprocesorul inițializează toate circuitele din sistem;
- prin intermediul unui port de ieșire, se încarcă în registrul de selecție al canalului analogic, numărul canalului a cărui achiziție se dorește;

- dacă achiziția poate începe în orice moment, microprocesorul dă semnalul de eșantionare și de inițiere a conversiei pentru primul eșantion. Dacă trebuie așteptată trecerea prin zero sau valoarea maximă a semnalului de intrare, circuitul sesizor de zero, respectiv de vârf, va da primul impuls de eșantionare și conversie, urmând ca, după aceea, microprocesorul să preia controlul;
- convertorul va emite semnalului **READY** la sfârșitul conversiei, care poate fi folosit ca semnal de întrerupere pentru microprocesor, ca să citească codul rezultat în urma conversiei. Dacă viteza nu e factor limitativ, microprocesorul poate citi periodic portul de intrare (modul *polling*), fiecare citire constituind totodată un nou semnal de eșantionare și de **START CONVERSIE**;
- eșantioanele sunt depuse, după fiecare citire, în memoria **RAM**; se urmărește numărul eșantioanelor efectuate, cu ajutorul unui numărător extern; procesul se oprește, atunci când s-a obținut numărul de eșantioane dorit;
- la sfârșit, eșantioanele sunt prelucrate și afișate rezultatele, conform programului elaborat și stocat în memoria **EPROM**.

Microprocesoarele și microcontroller-ele constituie resurse de comandă și prelucrare, care sunt deosebit de utile în procesul de gestiune a achiziției, de preluare și memorare a eșantioanelor rezultate în urma conversiei analog-digitale. Eficacitatea lor, în ceea ce privește procesul de prelucrare evoluată a eșantioanelor prelevate din proces, este mult mai redusă. Pentru o supraveghere precisă a unui proces, este, de regulă, necesară achiziționarea unor eșantioane, care convertite în formă numerică, au lungimi de **12** sau **16** biți. Folosirea unor algoritmi de prelucrare evoluată (filtrare numerică, analiză armonică, etc) comportă execuția unor instrucțiuni de înmulțire, adesea în virgulă mobilă, a două cuvinte de **2** octeți, rezultatul fiind disponibil pe un cuvânt de lungime dublă. Asemenea algoritmi pot fi cu greu implementați pe un *microcontroller* de **8** biți, cu consum nejustificat de timp și de resurse. De aceea, o soluție modernă de a realiza prelucrări complexe asupra datelor o constă utilizarea circuitelor **DSP** (**D**igital **S**ignal **P**rocessor - procesor digital de semnal), specializate în execuția unor astfel de instrucțiuni.

3.5.2 PROCESOARE DE SEMNAL: DSP

Procesoarele **DSP** sunt circuite specializate pentru prelucrarea semnalelor, permițând lărgirea considerabilă a câmpului de aplicabilitate a tehnicilor numerice în domeniul achizițiilor de date.

Pe lângă facilitățile extinse de operare numerică (înmulțire și împărțire rapidă în virgulă mobilă), de cele de prelucrare numerică a semnalelor (filtrare,

modulare, detecție, estimare de parametri, transformări neliniare, transformare Fourier rapidă - **FFT**), aceste circuite își găsesc deja aplicații în măsurări (analize de spectru, analize de regimuri tranzitorii), în telecomunicații (egalizatoare adaptive, modem-uri de înaltă viteză inteligente), în prelucrarea vorbirii (recunoașterea, transmisia și recepția vorbirii) și a imaginilor (reconstituirea, compresia, prelucrarea homomorfică a imaginilor). Trebuie remarcat că toate procesoarele **DSP** sunt *unități microprogramate*, cu facilități de prelucrare în timp real a semnalelor, dispunând de resurse logice destinate dezvoltării și punerii la punct a programelor de aplicații.

Pentru îmbunătățirea performanțelor, se depun eforturi pentru:

- mărirea spațiului de adresare (creșterea capacității memoriei interne **RAM** și **ROM**);
- utilizarea directă a memoriei externe;
- modificarea arhitecturii de bază în vederea creării unei memorii tampon;
- utilizarea optimală a magistralei interne, în vederea accelerării transferurilor de date;
- creșterea lungimii cuvintelor pentru executarea unor instrucțiuni în virgulă flotantă;
- creșterea frecvenței de operare prin utilizarea tehnologiei **CMOS**, ce permite creșterea gradului de integrare.

3.5.2.1 ARHITECTURA UNUI PROCESOR DE SEMNAL

Cele mai răspândite procesoare de semnal - **DSP** - sunt cele din seria **TMS 320XX**, produse de firma **Texas Instruments**, datorită performanțelor de calcul (5 milioane de instrucțiuni pe secundă) și versatilității lor. Viteza de calcul este obținută cu ajutorul unui unități aritmetico-logice ce încorporează un circuit de înmulțire paralel de **16 x 16** biți, cu rezultatul pe **32** biți, în **200ns**. Această înaltă performanță este destinată calculelor complexe de tipul convoluției, deconvoluției și transformării Fourier rapide.

Familia de procesoare **DSP TMS 320XX** dispun de o memorie **RAM**, având capacitatea de minimum **144** cuvinte de **16** biți, capacitate suficientă pentru a executa un algoritm **FFT** în **64** de puncte.

Procesoarele din familia **TMS 320XX** posedă **8** porturi de intrare-ieșire multiplexate, ce pot suporta o viteză de transfer de **40** milioane de biți pe secundă.

Primul component al acestei familii, procesorul **TMS 32010**, se caracterizează printr-o *arhitectură Harvard modificată* în vederea creșterii vitezei de funcționare și a versatilității. Modificările aduse permit transferurile între spațiile de memorie de program și de date: coeficienții înscriși în memoria de program pot fi citați în memoria de date, eliminând în acest fel necesitatea existenței unei memorii de program separate pentru coeficienți.

În arhitectura oricărui procesor **DSP** se disting *două secțiuni*:

- **secțiunea operativă**, organizată în jurul unei magistrale de date, cu o lungime de **16** biți; la magistrala de date sunt conectate:
 - un *sistem multiplicator* de **16 x 16** biți, rezultatul înmulțirii fiind disponibil pe **32** de biți. Sistemul de multiplicare dispune de două registre: registrul **T** (*registru temporar* în care se citește din memoria **RAM** de date unul dintre operanzi) și registrul **P** (*registru rezultat*) și de *circuitul de înmulțire* propriu-zis. O înmulțire se efectuează în două cicluri, unul necesar citirii în registrul **T** a unui operand și al doilea necesar citirii în registrul acumulator a celui alt operand;
 - o *unitate aritmetico-logică* operând pe **32** de biți și care are asociată un *registru acumulator (RA)*, de asemenea cu lungimea de **32** de biți. Registrul **RA** este divizat în două părți: partea mai semnificativă (biții **16÷31**) și partea mai puțin semnificativă (biții **0÷15**). Conținutul acumulatorului poate fi salvat în memorie prin două instrucțiuni succesive;
 - două *circuite de deplasare (SHIFTER)*, dintre care unul este destinat deplasării aritmetice programabile cu **0÷15** a datelor din memoria **RAM** de date, iar al doilea circuit efectuează o deplasare la stânga cu zero, cu una sau patru poziții, a părții superioare a acumulatorului;
 - o *memorie RAM* de date (**DATA RAM**), cu capacitatea de **144** de cuvinte de **16** biți. Această memorie este organizată sub forma a două pagini, dintre care prima pagină conține **16**, iar cea de-a doua **128** cuvinte. Indicatorii de adrese pot fi autoincrementați sau autodecrementați, incrementul fiind egal cu unitatea;
- **secțiunea de comandă**, organizată în jurul unei magistrale de program, conținând:
 - un registru denumit *contorul programului (PROGRAM COUNTER)* cu lungimea de **12** biți și o *stivă (STACK)* folosită pentru salvarea contextului programului. Registrul contor al programului conține adresa următoarei instrucțiuni din memoria de program;
 - *memoria de program (PROM)*, cu o capacitate de **1536** de cuvinte cu lungimea de **16** biți. Această memorie de program poate fi substituită cu o memorie externă de program, cu o capacitate de **4096** de cuvinte.

Această arhitectură a unui **DSP** permite funcționarea în modul *pipeline*, adică pe durata executării instrucțiunii curente, contorul programului se încarcă cu adresa următoarei instrucțiuni, instrucțiunea următoare fiind citită în paralel cu execuția instrucțiunii curente. Modul de lucru *pipeline* contribuie în mod decisiv la creșterea vitezei de lucru a procesorului.

În raport cu procesorul **TMS 32010**, cele ale următoarei generații - **TMS 32020** și **TMS 320C25** - sunt caracterizate prin următoarele modificări:

- eliminarea memoriei interne de program și înlocuirea ei cu o memorie

internă **RAM**, ce poate fi încărcată cu un program de aplicație de la o sursă externă;

- procesorul dispune de **544** de cuvinte de memorie, dintre care **256** cuvinte sunt utilizabile ca memorie de program;
- este disponibilă o memorie externă adresabilă de **64 K** cuvinte de program;
- realizarea operației de înmulțire cu înlănțuire într-un singur ciclu (**170 ns**);
- există cinci indicatori ai memoriei **RAM** interne; unitatea aritmetico-logică pentru calculul adresei asociată acestor indicatori permite o indexare cu pas variabil;
- circuit divizor de frecvență, integrat în structura procesorului;
- existența unor instrucțiuni specializate, destinate aritmeticii în virgulă flotantă;
- facilități de mascare a trei rezultate, în loc de unul singur.

Realizat în tehnologie **CMOS**, procesorul **TMS 320C25** este caracterizat de:

- durata unui ciclu instrucțiune de **100 ns**;
- consum energetic foarte redus;
- memorie de program (**ROM**) internă, cu capacitatea de **4K** cuvinte cu lungimea de **16** biți;
- un număr de **8** registre auxiliare, asociate unei unități aritmetice;
- stivă pentru salvarea contextului programului, organizată pe **8** nivele;
- două circuite de memorie **RAM** internă. Unul dintre aceste circuite poate fi configurat fie ca *memorie de program*, fie ca *memorie de date*.

Creșterea vitezei și eficienței de prelucrare numerică poate fi asigurată prin utilizarea unor procesoare de semnal cu arhitectură paralelă. Procesorul de semnal **SPROC-1400** face parte din seria **SPROC-1000**, conținând memoria de program, memoria de date, logica de secvențializare și logica de interfațare cu alte procesoare. Aceste resurse sunt necesare pentru proiectarea eficientă, implementarea și testarea sistemului. Procesoarele **SPROC-1000** pot fi configurate să lucreze atât în modul **MASTER**, cât și în modul **SLAVE**, pentru interconectarea cu alte procesoare **SPROC** sau cu un microprocesor tradițional.

Principalele caracteristici ale procesoarelor **SPROC-1400** sunt:

- integrarea tuturor subsistemelor de procesare de semnale pe un singur chip;
- arhitectură de tip multiprocesor optimizată;
- banda maximă de frecvență a semnalelor de intrare de maxim **250 kHz**;
- lungimea cuvintelor procesate de **24** de biți, cu posibilitatea de memorare pe **56** de biți;
- generator intern de ceas, cu frecvența de **50 MHz**;

- memorie **RAM** locală, reprogramabilă în mod dinamic;
- **4** porturi seriale, configurabile pentru manipularea a **8, 12, 16** sau **24** de biți de date;
- **1** port paralel, dispunând de **24** de linii, configurabil pentru manipularea unor cuvinte de date de **8, 12, 16** sau **24** de biți;
- tehnologia de realizare **CMOS** statică;
- inițializarea poate fi făcută atât intern, printr-un fișier de **16** Kocteți, cât și extern, prin folosirea unei memorii **ROM**;
- compatibilitate cu majoritatea microprocesoarelor aparținând familiilor **INTEL** și **Motorola**.

Seria de procesoare **SPROC-1000** utilizează o arhitectură cu memorie centrală optimizată (fig. 3.8) pentru procesarea concurentă a fluxurilor complexe, relaționale de date.

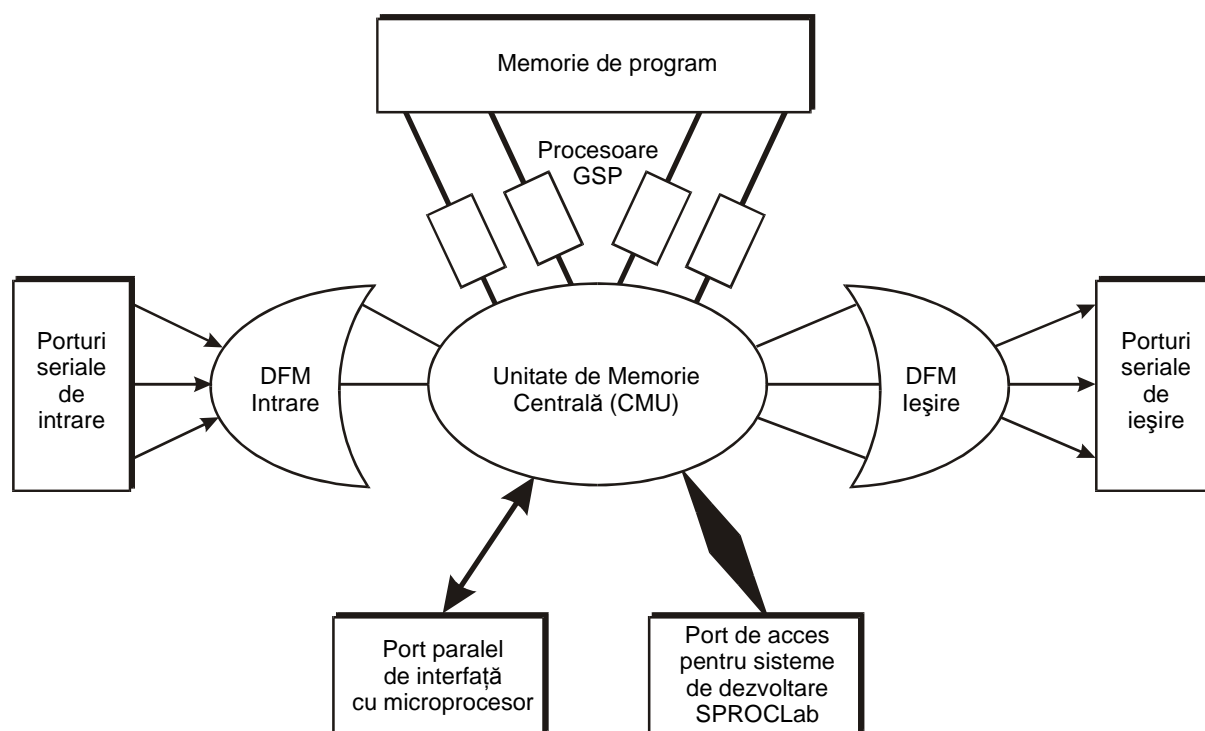


Fig. 3.8 Arhitectura procesorului de semnal SPROC 1000, cu unitate de memorie centrală.

Nucleul central al acestei arhitecturi este reprezentat de memoria de date partajată, de tip multiport, denumită **CMU** (Central Memory Unit - unitate de memorie centrală).

Un număr de patru procesoare generale de semnal, **GSP** (General Signal Processor - procesor general de semnal), implementate pe chip, realizează calculele și asigură procesarea paralelă.

Controloarele de fluxuri de date de intrare-ieșire, **DFM** (Data Flow

Manager - controlor al fluxului de date), coordonează fluxurile concurente de date, semnalele de interfață pentru canalele seriale și activează interfațarea cu un procesor extern.

Arhitectura **CMU** reprezintă o extensie de la abordarea monoprosesor la procesarea concurentă (*multiprosesor*). În locul multiplexării temporale, prin întreruperi, a unității de procesare, arhitectura **CMU** folosește mai multe procesoare și multiplexarea temporală a accesului acestora la memorie.

Accesarea multiplă a memoriei nu mai este gestionată prin întreruperi: **CMU** este un spațiu de memorie de date de tip multiport, ce utilizează o secvență compusă de perioade de acces la memorie, alocate fiecărui **GSP** sau ciclu de intrare-ieșire. Secvența de bază reprezintă un ciclu mașină al procesorului **SPROC**, respectiv cinci perioade de ceas. Perioadele 1÷4 sunt destinate procesoarelor generale de semnal, **GSP**, în vederea accesului la memorie, într-un mod secvențial bine determinat, iar cel de-al cincilea interval de timp este utilizat, prin subdivizare în 8 subintervale, pentru operații de intrare-ieșire paralele sau alte operații de intrare-ieșire.

Procesoarele **GSP** lucrează pe 24 de biți, în virgulă fixă și pot fi folosite fie individual, fie în grup, în funcție de cerințele aplicației.

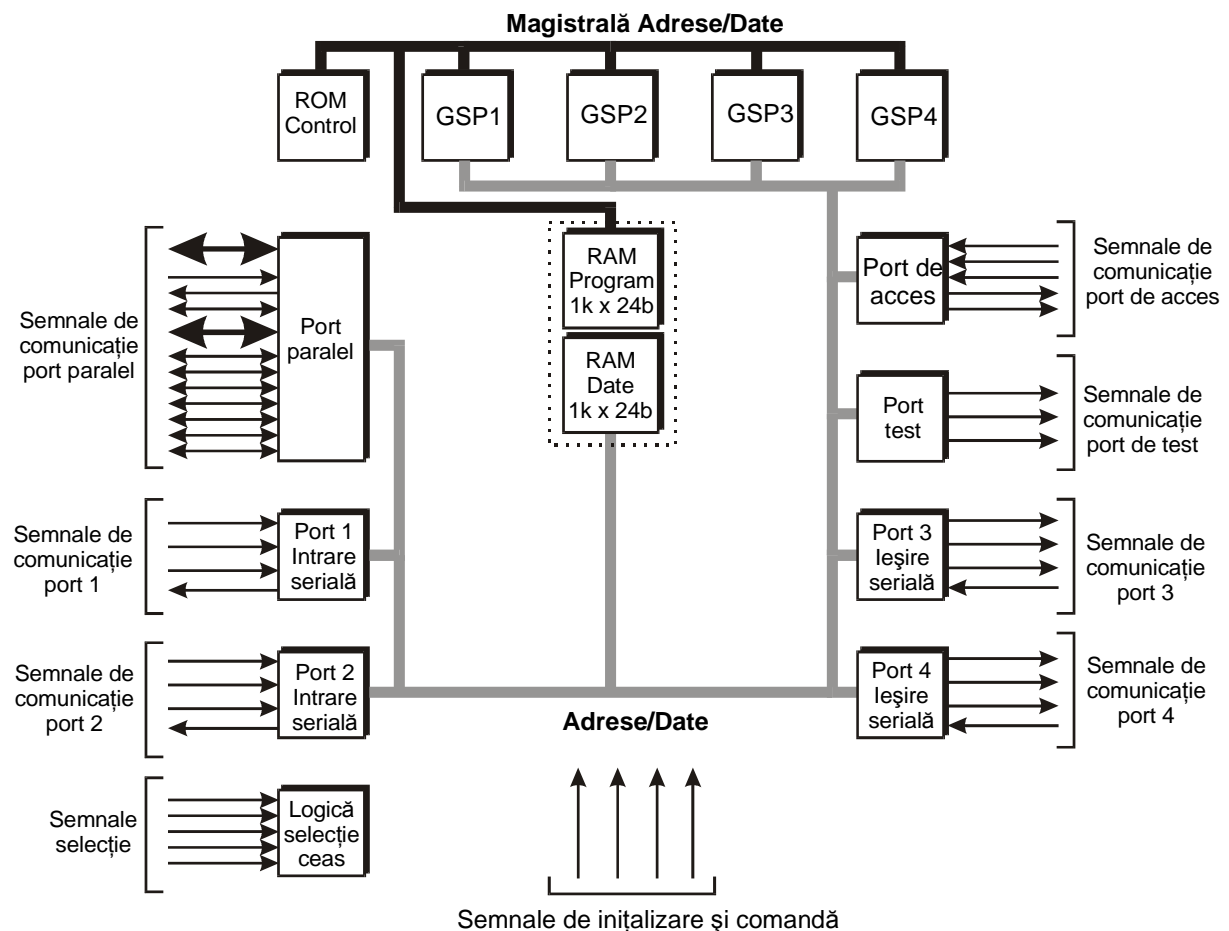


Fig. 3.9 Structura detaliată a procesorului de semnal **SPROC-1400**.

Controloarele de fluxuri de date gestionează introducerea/extragerea datelor în/din unitatea centrală de memorie, fără să afecteze performanțele procesoarelor **GSP**. Controloarele **DFM** comunică cu celelalte resurse interne pe o magistrală de 24 de biți, iar cu resursele externe prin intermediul porturilor seriale programabile.

Schema bloc a procesorului de semnal **SPROC-1400** este prezentată în fig. 3.9.

3.5.2.2 PORTUL SERIAL SINCRON AL FAMILIEI DSP TMS320C2XX

Familia de procesoare DSP **TMS320C2xx** dispune de un port serial sincron full-duplex, cu facilități de transmisie de cadre de date, suportând rate de transfer de până la 20 Mbps (pentru durata unui ciclu instrucțiune de 25 ns). Rata de transfer este jumătate din frecvența ceasului dispozitivului. Acest port serial sincron bidirecțional asigură comunicația directă cu dispozitive seriale, cum ar fi: CODEC-uri, convertoare analog-digitale seriale și alte echipamente seriale. Portul serial poate fi utilizat pentru comunicația între procesoare în cazul sistemelor multiprocesor. Atât secțiunea de transmisie, cât și cea de recepție, a portului serial dispun de un buffer sau memorie FIFO organizate pe 4 nivele de adâncime, permițând unității centrale de prelucrare să accepte întreruperi de la oricare din aceste nivele. Această facilitate înseamnă intervenția mai redusă a unității centrale de prelucrare, ca și creșterea flexibilității și eficienței transmisiilor de date.

Portul serial sincron al familiei **TMS320C2xx** are următoarele caracteristici:

- port serial sincron full-duplex, cu facilități de transmisie de cadre de date;
- buffer cu capacitatea de 4 cuvinte x 16 biți, pentru reducerea overhead-ului rutinelor de tratare a întreruperilor;
- port serial flexibil, eficient și de înaltă performanță;
- asigură rate de transfer de 20 Mbps, pentru durata unui ciclu instrucțiune de 25 ns;
- asigură rate de transfer de 14,28 Mbps, pentru durata unui ciclu instrucțiune de 35 ns;
- asigură rate de transfer de 10 Mbps, pentru durata unui ciclu instrucțiune de 50 ns;
- rata de transfer este jumătate din frecvența ceasului unității centrale de prelucrare.

Toate dispozitivele din familia **TMS320C2xx**, cu excepția circuitului **TMS329C209**, dispun de acest tip de port serial sincron. Fig. 3.10 ilustrează

organizarea portului serial sincron al procesoarelor DSP TMS320C2xx.

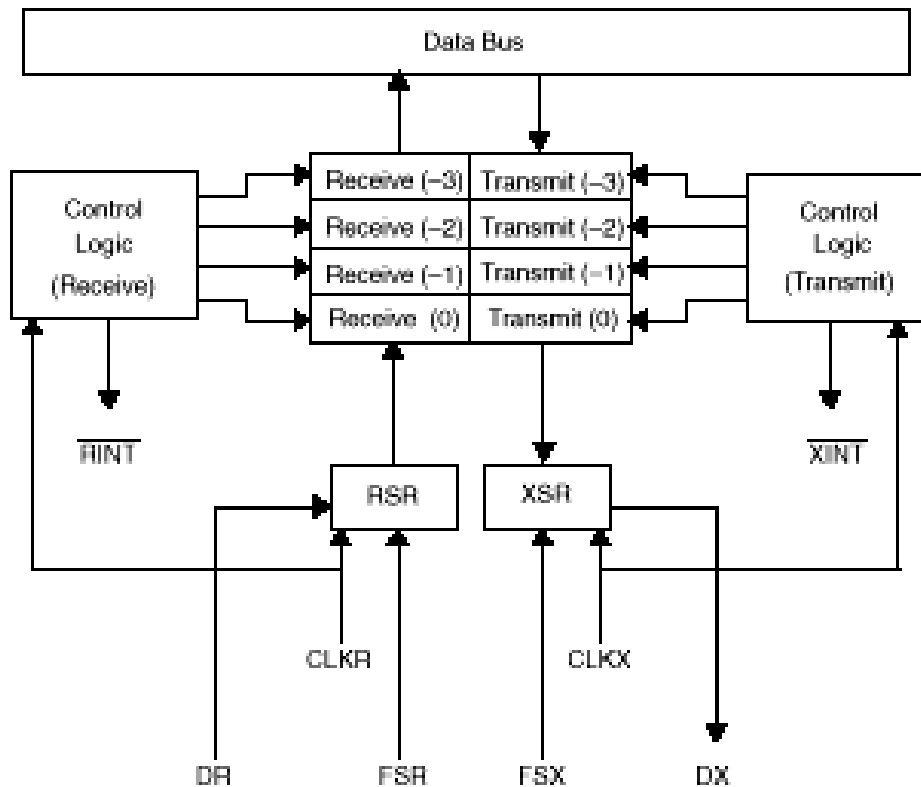


Fig. 3.10 Portul serial sincron al familiei de procesoare TMS320C2xx.

3.5.2.3 PORTUL SERIAL ASINCRON AL FAMILIEI DSP TMS320C2XX

Familia de procesoare de semnal TMS320C2xx dispune de un port serial asincron, full-duplex și dublu buffer-at. Acesta manipulează cuvinte de date cu lungimea de 8 biți și poate fi programat prin intermediul unui registru să accepte rate de comunicație de până la 2,5 Mbps. Portul serial asincron poate fi utilizat pentru comunicația cu alte dispozitive, cum ar fi microcontroller-e, prin conectare de tip RS-232 care suportă rate de transfer de date de până la 115,2 kbps.

Caracteristicile portului serial asincron al familiei de procesoare DSP TMS320C2xx sunt:

- port serial asincron full-duplex;
- portul serial dispune de o dublă buffer-are;
- portul serial permite transferuri de date de 8 biți;
- programarea ratei de transfer prin intermediul unui registru de 16 biți;
- portul serial asigură rate de transfer de 2,5 Mbps, pentru durata unui

ciclu instrucțiune de 25 ns.

Toate dispozitivele din familia **TMS320C2xx**, cu excepția circuitului **TMS329C209**, dispun de acest tip de port serial asincron.

Fig. 3.11 ilustrează organizarea portului serial asincron al procesoarelor DSP **TMS320C2xx**.

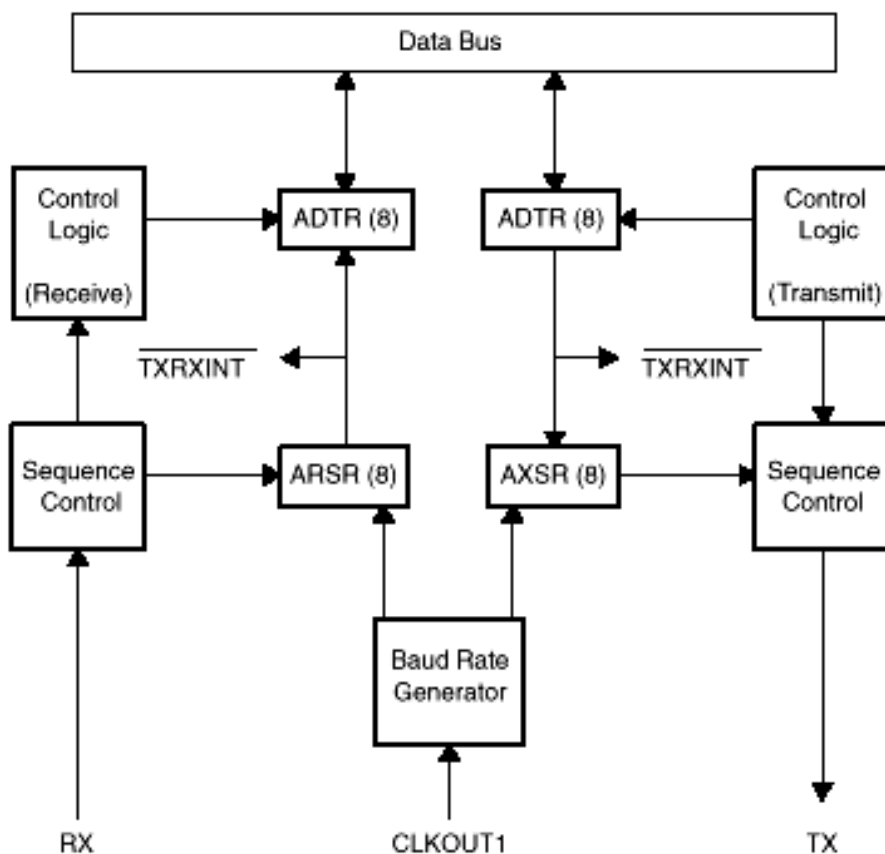


Fig. 3.11 Portul serial asincron al familiei de procesoare **TMS320C2xx**.

4. CONSIDERAȚII GENERALE ASUPRA INSTRUMENTAȚIEI VIRTUALE

4.1 INSTRUMENTE VIRTUALE

Se manifestă, în ultima perioadă, o tendință de creștere a flexibilității instrumentației, concretizată în obținerea unor arhitecturi deschise care permit dezvoltările necesare unui domeniu larg de aplicații. Acest lucru se manifestă atât la nivelul hardware-ului, cât și la nivelul software-ului, fiind susținute și de avantaje economice (cost, rapiditate, ușurință în utilizare).

Instrumentele de măsurare inteligente reprezintă entități independente, funcționând separat de un sistem de calcul, capabile să comunice un set redus de parametri și să execute o serie de comenzi. Toate echipamentele de măsurare dezvoltate în ultimii ani conțin interfețe prin intermediul cărora se transmit datele achiziționate, precum și comenzi unor relee cu care sunt echipate.

Indiferent de tipul de mărime măsurată, instrumentele inteligente sunt echipate cu relee care comandă direct echipamente externe, în funcție de valorile parametrului măsurat. După gradul de complexitate, pot fi însoțite de un pachet *software* aferent, executabil pe un sistem de calcul compatibil IBM-PC, pentru a putea executa citirile și comenzile la distanță. Foarte multe asemenea sisteme dispun de mijloace de memorare externă, care stochează variația în timp a parametrului măsurat sau valorile instantanee, la anumite intervale de timp.

Astfel, conceptul de aparat de măsură autonom este treptat înlocuit de un ansamblu modular, evolutiv, ușor adaptabil. Se asistă, în cadrul instrumentației, la o trecere de la funcționalitatea definită de constructor (constând într-o multitudine de aparate ce pot fi asamblate într-un sistem de măsură) la funcționalitatea definită de utilizator (constând din sisteme programabile, șasiuri conținând module ce permit dezvoltarea și implementarea a unui număr cu adevărat impresionant de aplicații).

Trebuie menționat în acest cadru rolul determinant al calculatoarelor, care prin puterea de calcul și mijloacele eficiente de afișare, au accelerat evoluția instrumentației. Dacă analizăm structura unui lanț de măsurare, se poate observa faptul că un sistem de calcul poate spori funcționalitatea instrumentelor prin interacțiunea sa asupra celor trei componente de bază:

- achiziția datelor;
- analiza datelor;
- prezentarea rezultatelor.



Fig. 4.1 Componentele procesului de măsurare.

În cadrul software-ului utilizat sunt înglobate biblioteci care servesc la programarea achiziției (module instrument), la prelucrarea datelor (module de analiză armonică de semnal, module de funcții statistice), la afișarea rezultatelor, conducând astfel la crearea de instrumente virtuale.

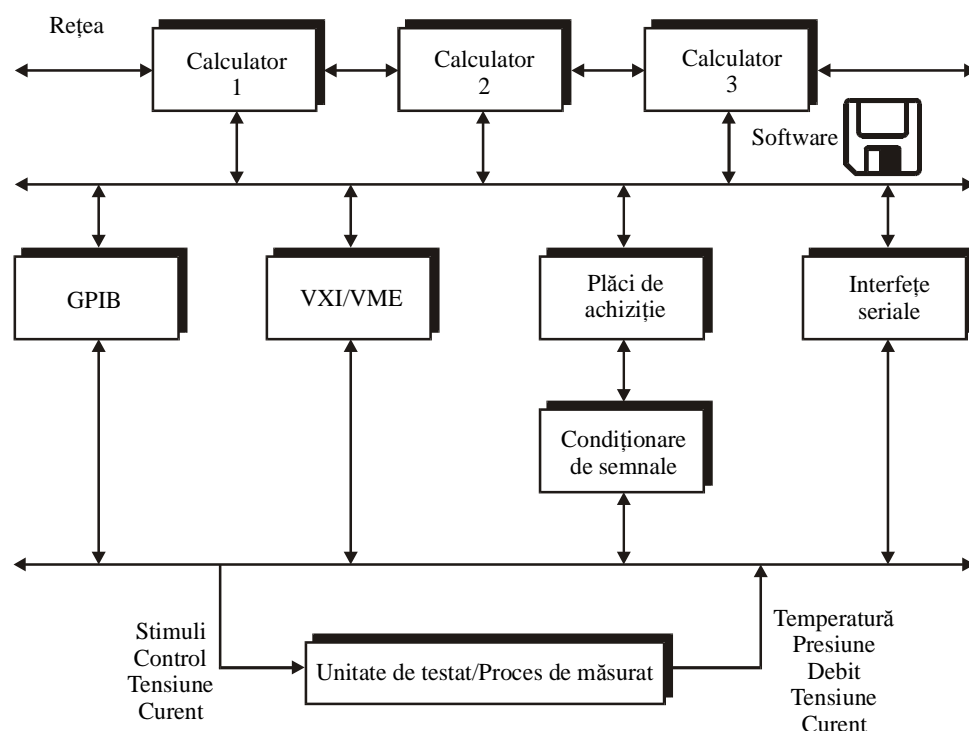


Fig. 4.2 Configurația instrumentației virtuale.

Un instrument virtual cuprinde, în principiu, următoarele elemente:

- unul sau mai multe sisteme de calcul;
- interfețe de rețea;
- software;
- dispozitive de intrare-ieșire (GPIB, VXI, plăci de achiziție);
- instrumente de măsură clasice, interfațabile;
- procesul controlat sau unitatea de testat,

fiind o combinație a instrumentelor programabile cu calculatoarele.

Calculatorul, prin versatilitatea pe care o prezintă, poate spori posibilitățile limitate (fixe) ale instrumentelor tradiționale.

Prin interconectarea în diverse combinații și programarea sistemelor de achiziție de date, a instrumentelor de măsură de care se dispune, folosind

aplicații software specifice, există posibilitatea de a crea propriile sisteme de instrumentație, denumite *sisteme de instrumentație virtuală*.

Datorită abilității de a construi sisteme de instrumentație chiar de către utilizator se câștigă timp, se economisesc bani și se poate interveni rapid în cadrul sistemului de măsurare sau de control.

Principalele caracteristici ale instrumentației virtuale - modularitatea și ierarhia - atât la nivel hardware, cât și software, îi conferă acesteia caracterul de flexibilitate și deschidere către noi domenii de aplicație.

Din punct de vedere al domeniului de utilizare, sistemele de instrumentație virtuală pot fi clasificate în:

- sisteme pentru testare și măsurări;
- sisteme de control automat al proceselor;
- sisteme pentru efectuarea de cercetări științifice.

Trebuie să remarcăm aici faptul că instrumentația virtuală are aplicabilitate în aproape toate compartimentele funcționale din cadrul unei organizații, și anume: cercetare-dezvoltare, proiectare, producție, calibrare, controlul și asigurarea calității, testare *“in situ”*, service.

Din punct de vedere al modului în care poate funcționa un instrument virtual putem menționa:

- modul de lucru *“live”*, în care se configurează fizic echipamentul de măsurare, cuplat la sistemul de calcul și care achiziționează date din procesul studiat, urmat de procesul de analiză, interpretare și afișare a rezultatelor;
- modul de lucru *“not live”* (simulare), în care se poate doar simula funcționarea echipamentului de măsurare asupra eșantioanelor unor semnale generate prin mijloace *software*.

În cele ce urmează vom prezenta pe scurt componentele unui instrument virtual, mai exact interfața calculator-proces de măsură sau control, precum și software-ul ce permite crearea instrumentelor virtuale.

În fig. 4.3 se prezintă un model schematic al unei prelucrări digitale generice de semnal. Se observă că semnalul de intrare este sub formă analogică, convertit apoi în valori numerice prin intermediul unui convertor analog-digital. Unitatea centrală (calculatorul) prelucrează aceste reprezentări numerice, urmând ca rezultatul să fie reconvertit în semnal analogic (prin intermediul unui convertor digital-analogic). Cât timp informația se găsește, încă, sub formă numerică, pot avea loc mai multe tipuri de operații asupra acesteia: analiză, memorare, afișare, transmisie etc.

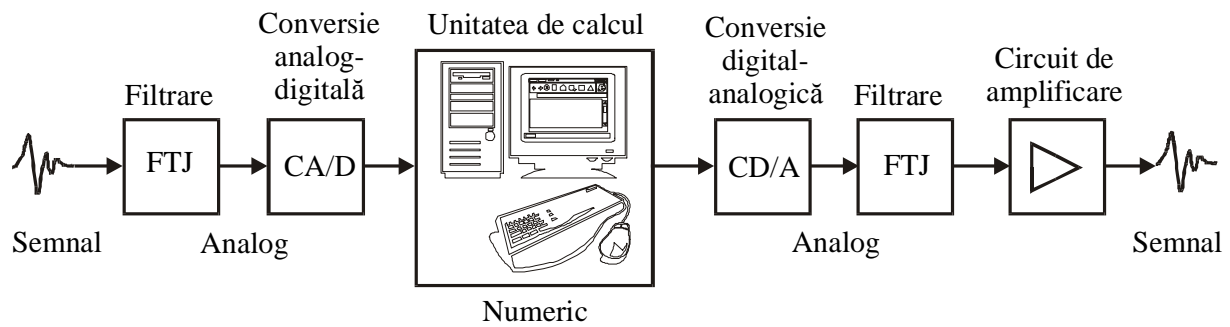


Fig. 4.3 Exemplu de sistem de prelucrare numerică a semnalelor.

Circuitele de intrare

Elementul de intrare al oricărei aplicații de procesare de semnal (analogică sau digitală) îl reprezintă traductorul (senzorul) specific mărimii de intrare, cu rolul de a converti mărimea de intrare într-un semnal de natură electrică. Pentru aplicațiile practice, senzorii de intrare pot fi de tipuri foarte diverse: microfon, în cazul semnalelor audio, traductoare de vibrații în seismologie, antene în aplicații radar, senzori cu infraroșu în teledetecție, transformatoare de măsurare în stațiile electrice, etc.

Circuitele de condiționare a semnalului

Scopul utilizării acestor module este acela de a converti valorile mărimii de ieșire a senzorului în valori compatibile cu domeniul de variație acceptat de convertorul analog-digital (de obicei 0-5V, sau 0-10V pentru convertoare setate pe mod unipolar). Circuitele de condiționare protejează, de asemenea, etajele superioare de prelucrare, uneori și prin separare galvanică, împotriva supratensiunilor accidentale. Modul de realizare al condiționerelor de semnal este bazat pe utilizarea circuitelor cu amplificatoare operaționale și/sau de instrumentație.

Filtrele anti-aliasing

Filtrele anti-alias (*aliasing*: replierea spectrelor) sunt filtre trece-jos care, în principiu, limitează superior viteza de variație în timp a semnalului analogic aplicat la intrarea sistemului. Funcția acestor filtre este critică, ea determinând direct modul în care restul sistemului este capabil să urmărească variațiile de interes ale semnalului.

Convertoarele analog-digitale

După cum sugerează și numele, convertoarele analog-digitale (*analog-to-digital converter*, ADC) furnizează o reprezentare numerică (binară) a

semnalului analogic aplicat la intrarea lor. Două dintre cele mai importante caracteristici ale unui ADC sunt rata de conversie și rezoluția. *Rata de conversie* (uneori definită și prin inversul său, timpul de conversie) exprimă cât de repede se efectuează operația de conversie. **Rezoluția** exprimă gradul de apropiere între valoarea numerică obținută la ieșirea convertorului și valoarea reală corespunzătoare semnalului analogic aplicat la intrare.

Procesorul

Procesorul efectuează operațiile matematice necesare procesării semnalului o dată convertit în formă numerică. De exemplu, în cazul în care aplicația este un amplificator, atunci procesarea digitală constă în înmulțirea valorii corespunzătoare intrării cu constanta de amplificare.

În primele aplicații de procesare numerică, procesorul numeric era constituit adesea dintr-un calculator de uz general dar, pe măsură ce domeniul s-a dezvoltat, au fost proiectate și utilizate o gamă largă de procesoare dedicate, de mare viteză, capabile să prelucreze un număr imens de date. Astăzi sunt utilizate, în principal, procesoarele de semnal care reprezintă, de fapt, o combinație de hardware de viteză mare, arhitectură specializată și un set de instrucțiuni dedicate procesării de semnale permițând implementarea eficientă a algoritmilor specifici.

Memorarea și transmiterea datelor

În memoria program se stochează instrucțiunile prin care se implementează algoritmi DSP. Dacă într-un calculator cu *arhitectură von Neumann*, datele și instrucțiunile sunt memorate într-un același bloc, în cele mai multe sisteme dedicate DSP, instrucțiunile sunt memorate separat de date, asigurându-se astfel o procesare mult mai rapidă a instrucțiunilor. Datele pot fi transferate pe magistrala proprie simultan cu executarea instrucțiunilor din program. De multe ori cele două magistrale - de date și de instrucțiuni - au dimensiuni (număr de linii de acces) diferite. Această arhitectură a fost dezvoltată pe baza cercetărilor efectuate inițial la Universitatea Harvard și, de aceea, este denumită ca fiind *arhitectură Harvard*.

Datele prelucrate de un sistem DSP sunt adesea destinate unor prelucrări ulterioare, în cadrul aceluiași sau al altui sistem. Ele pot fi memorate pe bandă magnetică, discuri optice (CD) sau alt suport.

Interfața cu utilizatorul

Nu toate sistemele DSP necesită dispozitive de afișare sau introducere a datelor de către utilizator, cu ajutorul unui keypad (tastatură) specializat, a unor

comutatoare discrete sau chiar a unei tastaturi clasice. Cu toate acestea, adeseori este foarte comodă reprezentarea grafică a semnalelor prelucrate.

Convertoarele digital-analogice

Multe dintre sistemele DSP impun o etapă de re-conversie a semnalului în formă analogică, această funcție fiind îndeplinită de convertoarele digital-analogice (*digital-to-analog converter*). Una dintre caracteristicile esențiale ale convertorului digital-analog este viteza (*slew rate*) cu care se obține tensiunea la ieșirea acestuia, după setarea valorii numerice dorite la intrarea convertorului. Acest parametru trebuie să corespundă ratei de conversie analog-digitală a întregului sistem DSP.

Filtre de netezire și amplificatoare de ieșire

Scopul filtrelor de netezire este acela de a micșora discontinuitățile în forma (de tipul scară) semnalului de la ieșirea convertorului digital-analogic. Uzual, un astfel de filtru are o caracteristică trece-jos implementată cu un circuit RC simplu.

Amplificatorul de ieșire este, în general, de tip clasic, prevăzut în scopul de a adapta impedanța mare a convertorului digital-analog la impedanța de intrare mică a traductorului de ieșire (difuzor, antenă, motor electric etc.) precum și pentru a regla puterea la nivelul cerut.

În funcție de specificul aplicației, **unele module din sistemul prezentat mai sus pot lipsi**. De exemplu, într-o aplicație de telegestiune a consumului energetic, convertorul D/A nu se mai regăsește. Important este, însă, faptul că modelul prezentat anterior este valabil pentru orice tip de aplicație din gama procesărilor numerice de semnal.

4.2 INTERFAȚA CALCULATOR - PROCES DE MĂSURARE SAU CONTROL

După cum se sugerează și în cadrul fig. 4.2, există următoarele tipuri de magistrale de proces pentru interconectarea instrumentelor programabile cu un sistem de calcul:

- **GPIB (General Purpose Interface Bus** - magistrală de interfațare de scop general) este specializată pentru comunicație cu instrumente programabile. Este disponibilă pentru platforme diverse (PC, IBM, MacIntosh, etc.) și oferă o rată de transfer de 1 Mbyte/secundă. Protocolul de comunicație este specificat de standardul industrial

IEEE-488 (protocol pe 8 biți, comenzi tip **ASCII**, maximum 14 instrumente aflate la cel mult 20m). O dezvoltare ulterioară, **HS-488**, a crescut rata de transfer la 8 Mbytes/secundă;

- **RS-232, RS-422, RS-485** sunt utilizate în special la sisteme personale de calcul. Sunt destinate comunicațiilor seriale sincrone / asincrone între instrumente, oferind o imunitate ridicată la zgomotul din mediul industrial. Dezavantajul major este rata de transfer redusă (sute de kbits/secundă). Protocolul de comunicare poate fi **ASCII** sau un protocol special al instrumentelor ce se interconectează. Ceea ce le diferențiază este distanța de interconexiune (de la 15m la **RS-232**, la 1200m la **RS-485**), care este mărită prin amplificări suplimentare pe intrări/ieșiri. Interfața **RS-485** permite legături multi-punct ("*daisy-chain*"), deci economii importante de cablu;
- **SCXI** (Signal Conditioning eXtensions for Instruments): aceste module, montate sub formă de "*rack*" (dulap cu sertare de conexiuni modulare), sunt utilizate la condiționarea semnalului. La intrare, se admit semnale provenite de la traductoare, cum ar fi: termocupluri, termistoare, mărci tensometrice etc, iar la ieșire se obțin semnale analogice (curenți sau tensiuni) sau semnale numerice. Se utilizează module de filtrare, liniarizare, izolare galvanică, ieșiri de putere, etc.;
- **VXI** (VME eXtension for Instrumentation, unde **VME** reprezintă VESA Module Eurocard) reprezintă un standard de instrumentație ale cărui baze au fost puse încă din 1987 și are ca obiectiv creșterea interoperabilității diverselor instrumente. Structura sa modulară permite o integrare atât din punct de vedere electronic, cât și mecanic, conferindu-i denumirea de "standard pentru instrument pe cartelă". Specificațiile **VXI** grupează norma **IEEE-488** și norma **VME**, prima remarcându-se prin compatibilitatea între constructori diferiți, iar cea de-a doua prin rapiditate.

Sistemele de instrumentație **VXI** s-au impus în ultima perioadă, deoarece:

- prezintă mare suplețe, mai multe nivele de inteligență și permit o sincronizare și o asociere optimală și performantă;
- au dimensiuni mici, sunt rapide, putând fi realizate și în variante portabile;
- sunt caracterizate de o arhitectură deschisă la nivel hardware (magistrală **VME** de 32 biți și *software* unificat **VXI "Plug & Play"** - recunoașterea tipului de cartelă se face automat de către mediul *software*, la fel inițializarea și parametrizarea circuitelor de pe cartelă), permițând utilizarea de module provenite de la diverși producători, fără a exista probleme la nivel electric sau mecanic. Compatibilitatea electromagnetică între module, compatibilitatea mecanică și termică permit interoperabilitatea între diferite produse hardware;

- permit atât transmisia serială a datelor (transmisie lentă, dar cu un nivel de prelucrare și de inteligență ridicat), cât și transmisia lor paralelă (viteză mare de transmisie);
- utilizează un controller extern sau intern. În primul caz, calculatorul comunică cu o interfață de tip **GPIB-VXI** sau cu o magistrală de mare viteză **VXI-MXI** (**M**ultisystem **E**Xtension **I**nterface **B**us). În cel de-al doilea caz, există un controller integrat în structura **VXI** care coordonează direct resursele acesteia, permițând aplicații multiple și rapide, cum ar fi controlul *în timp real* al unor procese;
- permite controlul interactiv, pe timpul programării, utilizând, de exemplu, programul **VIC** (**V**XI **I**nteractiv **C**ontrol).

Putem concluziona că sistemele **VXI** prin modularitatea, flexibilitatea, interoperabilitatea și interconectarea facilă la un sistem de calcul pe care le prezintă, acoperă un domeniu larg de aplicații, mergând de la teste și măsurări până la achiziția de date de înaltă performanță și automatizări industriale.

4.3 SOFTWARE PENTRU INSTRUMENTAȚIE VIRTUALĂ

Instrumentația virtuală este centrată pe software. Dintre aplicațiile software special proiectate în vederea muncii de programare a utilizatorului trebuie remarcat LabVIEW (**L**aboratory **V**irtual **I**nstrument **E**ngineering **W**orkbench). Abordarea se face printr-un concept revoluționar denumit *programare vizuală*. LabVIEW este un mediu de programare bazat pe limbajul grafic G și utilizat pentru achiziția și controlul datelor, analiza acestora și prezentarea rezultatelor. În cadrul abordărilor software tradiționale, după specificarea schemei unei aplicații urmează convertirea ei în codul propriu limbajului de programare. În mediul de programare LabVIEW “se construiesc” instrumente virtuale (se assemblează și se interconectează diferitele simboluri grafice, denumite “*icons*” pentru realizarea proiectului) în loc de a se scrie programe, deoarece schema bloc a instrumentului creat conține chiar codul sursă al aplicației.

Acest mediu de programare, dezvoltat începând cu 1980 și perfecționat în mod continuu, are ca principal obiectiv crearea posibilității oamenilor de știință și inginerilor de a-și dezvolta rapid propriile sisteme de instrumentație virtuală, corespunzător necesităților specifice, fără a fi necesară cunoașterea sintaxei limbajului grafic de programare.

Un instrument virtual creat în mediul de programare LabVIEW este compus din:

- panoul frontal, reprezentând interfața grafică cu utilizatorul;
- diagrama bloc, reprezentând codul sursă al instrumentului virtual sau al aplicației;

- simbolul grafic / conectorul, reprezentând interfața de apel.

Panoul frontal este o interfață interactivă pentru a controla intrările și a observa ieșirile. Fiind mijlocul prin care utilizatorul descrie secvența de măsurare de interes, el mai este denumit și interfața grafică cu utilizatorul. Este afișat pe ecranul calculatorului și are același rol cu panoul frontal al unui instrument fizic, conținând atât elemente de comandă (butoane, comutatoare, potențiometre), cât și elemente indicatoare și de vizualizare (afișaje, ecrane). Primele elemente simulează dispozitivele de intrare ale instrumentului, iar cele din urmă dispozitivele de ieșire, ce afișează datele achiziționate sau generate de către diagrama bloc a instrumentului virtual.

Crearea panoului frontal poate fi făcută înaintea realizării diagramei bloc. Elaborarea și editarea lui se face în cadrul modului de lucru “**Edit**” (celălalt mod de lucru, “**Run**”, fiind utilizat pentru funcționarea instrumentului virtual), prin selectarea și poziționarea obiectelor necesare alese dintre simbolurile grafice ale instrumentelor disponibile (tools palette, controls palette).

Când instrumentul virtual este complet implementat și devine funcțional, panoul frontal este utilizat pentru controlul sistemului de măsurare prin acționarea unor comutatoare, mișcarea unui cursor, selectarea unei opțiuni, etc.

Elementele indicatoare de pe panoul frontal al instrumentului virtual răspund imediat la aceste modificări, datorită răspunsului în timp real al sistemului.

Diagrama bloc reprezintă codul sursă al instrumentului virtual și conține:

- funcții de calcul aritmetice și logice;
- funcții de analiză complexă (achiziții complexe și instrumente de analiză);
- funcții de intrare-ieșire care permit comunicația cu sistemele de achiziții **GPIB**, **VXI**, instrumente reale externe.

Construirea diagramei bloc se face prin selectarea blocurilor funcționale din meniul (lista de comenzi) “**Functions**”, poziționarea lor și interconectarea terminalelor acestora prin linii ce dirijează fluxul informațional de la un bloc la altul.

Toate elementele de comandă și elementele indicatoare ce au fost poziționate pe panoul frontal al instrumentului apar și în diagrama bloc, de această dată cu propriile simboluri grafice și terminale pentru a fi interconectate.

Tipul de date care este vehiculat determină tipul conexiunii între două blocuri. De exemplu, grosimea liniilor indică dacă este vorba de o mărime de tip scalar (o singură dată) sau vectorial (rețea de date), iar culoarea poate indica dacă este vorba de valori numerice, booleene, de tip text, grafice, etc.

Odată realizate panoul frontal și diagrama bloc, instrumentul virtual devine operațional (modul de lucru “**Run**”).

Instrumentul virtual nou creat poate fi salvat și i se poate atribui un simbol grafic și o rețea de interconectare pentru a putea fi utilizat în cadrul unei aplicații

sau în cadrul diagramei bloc a unui instrument virtual mai complex.

Structura modulară constituie unul dintre marile avantaje ale instrumentației virtuale, deoarece dă posibilitatea ierarhizării pe grade de complexitate. Prin ierarhizare modulară se pot proiecta, modifica, interschimba și combina instrumente pentru a răspunde specificațiilor a cât mai multe aplicații.

De asemenea, pot fi efectuate atât simulări, cât și teste sau achiziții reale de date, având posibilitatea efectuării facile de modificări la orice nivel.

Construcția modulară, ierarhică și dinamică, atât la nivel hardware cât și la nivel software, conferă instrumentației virtuale flexibilitate, orientare către necesitățile utilizatorului, care o poate adapta conform propriilor aplicații și putând să o dezvolte într-un mod simplu, eficient și rapid.

4.3.1 ALEGEREA PLATFORMEI SOFTWARE: UNIX SAU WINDOWS?

Una din problemele cu care se confruntă utilizatorii de sisteme de achiziție de date și control al proceselor o constituie sistemul de operare sub care va funcționa ansamblul dispozitivelor automate. Utilizatorul este pus, de cele de multe ori, să aleagă între sistemele realizate sub Unix și cele realizate sub Windows NT. Unix este un sistem stabil și economic, care a câștigat încrederea utilizatorilor, astfel că în prezent este greu de răspuns la întrebarea dacă sau când Windows NT va fi preferat sistemului Unix. Mulți proiectanți (Industrietechnik AG, ABB) oferă produse similare capabile să funcționeze sub ambele sisteme de operare, chiar și atunci când platforma sistem este pe un loc secundar ca relevanță. Cu toate acestea, mulți utilizatori rămân foarte atașați de lumea stabilă (și verificată în ultimii ani) a sistemului Unix, cu toate că sunt convinși de funcționalitatea aplicațiilor sub Windows și de faptul că platformele NT sunt mai ieftine (instalate pe atât de răspânditul PC) decât cele Unix (care lucrează aproape în exclusivitate pe Workstations).

Domeniile în care aplicațiile dezvoltate sub Windows NT sunt neafectate de eventualele puncte critice ale sistemului de operare rămân cele aferente managementului informațiilor.

În ceea ce privește cel mai “critic” punct al aplicațiilor industriale, cel al operării în timp real, cartelele PC cu procesoare specializate garantează capacitatea de prelucrare în timp real, dar numai atât timp cât s-au introdus module speciale de comunicație între controller (sau PC) și sistemul supravegheat. Aceste module garantează până și așa-numita funcționare dificilă în timp real, soluția adoptată de ABB, spre exemplu, fiind aceea de a aplica fiecărei informații primite de controller o așa-numită “ștampilă” temporală (*Zeitstempel*) care va fi memorată și transmisă componentelor utilizator.

Prelucrarea propriu-zisă a datelor nu se va face în condițiile dificile de timp real ci ulterior, sincronizarea și evaluarea evenimentelor făcându-se prin apelarea momentelor de timp marcate.

La ora actuală (2001) se depun eforturi pentru realizarea de sisteme de conducere a proceselor industriale cu sisteme de operare mixte (NT și Unix). De fapt, se oferă soluțiile disponibile în prezent numai sub Unix și utilizatorilor de NT după o prealabilă asigurare împotriva riscurilor (în principal, bug-uri proprii sistemului Windows) de ordin tehnic. De asemenea, se pot integra soluții NT în lumea tehnică Unix. Scopul proiectării noilor oferte din acest domeniu (prin folosirea tehnicilor “object-oriented” de manipulare a datelor) este, însă, acela al independenței totale a aplicațiilor față de platforma sistem a utilizatorului.

4.4 PARTICULARITĂȚI ALE INSTRUMENTAȚIEI VIRTUALE

Implementarea unui instrument *software* de analiză a semnalelor electrice presupune utilizarea unui calculator - drept nucleu *hardware* de comandă - și a unui sistem de achiziții de date, putând fi realizată în mai multe moduri, funcție de sistemul de operare utilizat în cadrul sistemului de calcul.

Suportul *hardware* este format dintr-un lanț complex de măsurare, compus din: senzor, dispozitivul de condiționare a semnalului, placa de achiziții de date și sistemul de calcul aferent.

Ceea ce diferențiază aceste instrumente este prezența *software*-ului, un mediu de programare mobil, capabil să proiecteze pe monitorul unui calculator orice parametru măsurat, orice grafic preluat, orice reglaj, buton, etc., transformând practic calculatorul gazdă al aplicației, într-un instrument de măsurare.

Utilizarea sistemului de operare tradițional **MS DOS®** - **Microsoft Disk Operating System** - permite folosirea unor limbaje de programare de nivel înalt, ca de exemplu *Borland Pascal*, *Microsoft C* și *C++*. Un astfel de limbaj de programare permite:

- *gestionarea procesului de achiziții de date* prin intermediul unor rutine de comandă a resurselor *hardware* ale sistemului de achiziții. Aceste rutine, denumite în mod uzual *driver*-e, sunt specifice unei anumite configurații de sistem de achiziții de date și înglobează adesea linii sursă de cod scris în limbaj de asamblare. *Driver*-ele sistemului de achiziții de date sunt, de cele mai multe ori scrise de către producătorul sistemului hardware de achiziții de date, dar pot fi de asemenea scrise și optimizate de către utilizator, dacă acesta dispune de informații referitoare la adresele fizice de porturi de intrare-ieșire utilizate de către componentele programabile din sistem, la cuvintele de stare și de comandă asociate acestora, la secvențele de programare necesare pentru

manipularea resurselor;

- *calculul parametrilor caracteristici ai semnalelor electrice*. Această activitate presupune utilizarea unor funcții sau proceduri de analiză spectrală - transformată Fourier -, utilizarea unor rutine pentru aplicarea unor ferestre de eșantionare adecvate (dreptunghiulară, Hanning, Hamming, Bartlet, Blackman, etc.) asupra semnalelor originale pentru evidențierea particularităților acestora, utilizarea unor funcții și proceduri de filtrare și interpolare numerică, etc. Toate aceste rutine pot fi realizate în orice limbaj de programare de nivel înalt, cu eforturi de programare comparabile din punct de vedere al complexității;
- *reprezentarea, într-o formă cât mai intuitivă și cât mai reprezentativă (sub formă grafică și/sau tabelară), a parametrilor de stare ai semnalelor electrice analizate*. Realizarea acestor activități impune utilizarea primitivelor grafice ale limbajelor de programare de nivel înalt, astfel încât marea majoritate a programatorilor folosesc limbajele **C** și **C++**, datorită utilizării mai facile a primitivelor grafice ale acestora.

Utilizarea sistemului de operare **WINDOWS** (cu variantele **Windows 3.1, Windows 3.11 for Workgroups, Windows 95/98** sau **Windows NT**) deschide noi posibilități referitoare la implementarea unor astfel de instrumente software de analiză a semnalelor electroenergetice. Aceste facilități se referă la următoarele aspecte:

- *modul de lucru* în sistemul de operare **Windows** este de tip *multitasking* (se pot executa simultan mai multe aplicații, prin partajarea activității sistemului de calcul), spre deosebire de modul de lucru în sistemul de operare **DOS**;
- *facilități de lucru în background* (într-o fereastră “ascunsă” din fundal, operarea fiind transparentă față de utilizator), respectiv *foreground* (în fereastra principală);
- *interfața grafică* a sistemului de operare **Windows** este mult mai evoluată decât cea din sistemul de operare **DOS**, fiind, de asemenea, mult mai prietenoasă;
- folosirea unor *limbaje de programare de nivel înalt evolute, orientate pe obiecte*, care ușurează foarte mult activitățile de programare. Ca exemple de astfel de limbaje de programare putem menționa: **Microsoft Visual Basic**, **Visual C++**, **Borland Delphi Client-Server** sau **Delphi 32**, **Borland Visual dBase**, **Microsoft Visual FOX Pro**. Ultimele două aplicații sunt orientate spre baze de date, fiind necesare în cazul în care se dorește o *analiză statistică* a rezultatelor prelucrărilor;
- mediul de operare **Windows** asigură facilități de corelare și de transfer de informații între diverse aplicații de tip **Windows**, lucru mult mai dificil de realizat între aplicații **DOS**.

Numeroase firme producătoare de sisteme de achiziții de date, cum ar fi: **Axon Instruments**, **Burr Brown**, **Philips**, **National Instruments**, **Tektronix**, au realizat astfel de instrumente *software*, denumite *instrumente virtuale*, care modelează pe ecranul calculatorului panoul frontal al diferitelor echipamente de măsurare realizate și care, prin intermediul unor comenzi furnizate prin intermediul tastaturii sau a mouse-ului, configurează echipamentul fizic cuplat la sistemul de calcul prin intermediul canalelor seriale de tip **RS-232** sau **RS-485**, prin intermediul magistralei sistemului (în cazul sistemelor de achiziții de date de tip *plug-in*) sau prin intermediul unor magistrale dedicate, cum ar fi **IEC-625**, **GPiB**, **IEEE-488**, **VXIbus/ MXIbus**, etc. Sunt disponibile *instrumente virtuale* sub ambele sisteme de operare menționate anterior, **DOS** și **Windows**.

O problemă incomplet soluționată, din motive economice și de copyright (drepturi de distribuție), este aceea de *compatibilitate*, având un impact negativ major în special în situația în care suportul hardware dispune de o interfață nestandard cu sistemul de calcul. Un mediu de programare pentru instrumentația virtuală conține înglobate *driver*-ele destinate unui set bogat, dar nu exhaustiv, de echipamente de măsurare. Cu alte cuvinte, utilizatorul nu are acces nemijlocit la *driver*-ele sistemului de măsurare și, deci, la resursele sistemului de achiziții de date. Această problemă poate fi soluționată foarte simplu, în mediul de operare **Windows**, prin rularea în *background* a *driver*-elor echipamentului de măsurare, cu condiția ca acestea să fie direct disponibile. Compatibilizarea poate fi pe deplin realizată prin partajarea de către instrumentul virtual și de către sistemul de achiziții de date a unor fișiere de comunicație (de date), cu formate standardizate. Aceste fișiere pot fi modificate (actualizate) doar de către echipamentul de măsurare și pot fi citite doar de către instrumentul virtual.

4.5 NOI INSTRUMENTE DAQ SPECIALIZATE EXTIND NOȚIUNEA DE INSTRUMENT VIRTUAL

Utilizatorii de instrumentație de azi au nevoie de instrumente ce sunt performante și flexibile. Performanța reflectă funcționalitatea primară a instrumentului, adică osciloscoapele și generatoarele de semnal lucrează la viteze din ce în ce mai mari, multimetrele digitale sunt și mai precise, etc. Flexibilitatea instrumentului este dată de posibilitatea de a include în procesul de măsurare alte operații cum ar fi analiza, automatizarea operațiilor, accesul la bazele de date, transmisia datelor prin internet, etc. Instrumentul tradițional este foarte bun în performanță dar are probleme în a oferi flexibilitate deoarece nu se poate compara cu un computer PC. Instrumentele virtuale construite în interiorul computerelor PC oferă flexibilitate dar sunt mărginite în funcționalitate de limitările computerului în care sunt construite, adică limitările în tehnologia la zi a computerului, printre care se amintește viteza de transfer al

datelor din exterior în memoria RAM.

Ultimele îmbunătățiri din domeniul arhitecturii calculatoarelor PC sunt bus-ul PCI, procesorul Pentium, sistemul de operare Windows NT și interfața ASIC MITE, elaborată de National Instruments pentru implementarea de transferuri ultrarapide între placa de tip plug-in și memoria RAM. Aceasta a permis implementarea de instrumente virtuale a căror performanță egalează performanța instrumentelor tradiționale și care în plus oferă flexibilitatea pe care acestea nu o pot oferi.

În oferta sa, National Instruments anunță instrumentele specializate din clasa DAQ Instruments, *DACScope*, *DAQArb* și *DAQMeter*, ce sunt compuse din hardware plug-in (PCI, AT-bus sau PCMCIA) și aplicație software cu interfață grafică gata făcută. Aceste instrumente virtuale arată și funcționează exact ca și instrumentul tradițional, au performanța instrumentului tradițional și are flexibilitatea unui calculator PC.

În continuare sunt prezentate evoluțiile recente din domeniul arhitecturii computerului PC ce influențează instrumentația virtuală de azi și trei noi instrumente din clasa DAQ Instruments.

4.5.1 TRANSFERUL DE DATE ÎN BUS-UL PCI

Plăcile de tip plug-in realizate pentru bus de tip PCI sunt de două tipuri:

1. initiator - *bus master board*
2. target - *slave board*

Plăcile ce acționează ca bus-master transferă date între memoria FIFO internă și memoria RAM mai rapid și mai eficient în comparație cu plăcile de tip slave-boards.

Placa de tip slave efectuează transferuri de date secvențiale între FIFO și RAM, adică placa slave întrerupe unitatea CPU în mod repetat pentru a transfera date între memoria de pe placă și memoria din computer. Când unitatea CPU este liberă, aceasta va efectua transferul de date și va executa celelalte operații cerute, cum ar fi analiza numerică, display, stocare, etc. Placa de tip bus-master are capacitatea de a face transferuri de date între FIFO și RAM fără intervenția procesorului CPU, adică în timp ce datele sunt transferate, procesorul CPU efectuează alte operații, cum ar fi analiza numerică, display, control, etc. Folosirea plăcii plug-in PCI de tip bus-master îmbunătățește eficiența utilizării procesorului CPU cu un factor de 10:1 față de situația în care placa PCI este de tip slave.

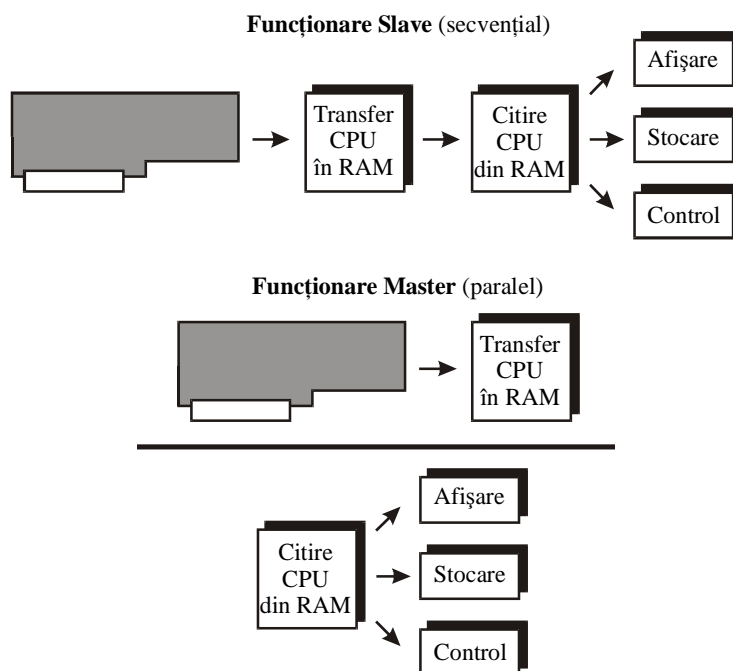


Fig. 4.4 Plăci de tip bus-master și de tip slave.

4.5.2 IMPLEMENTAREA DMA PE PLACA DE TIP PCI BUS MASTER. CHIP-UL ASIC MITE.

Spre deosebire de bus-ul de tip ISA sau EISA, a cărui arhitectură standard include procesorul DMA controller 8237A, arhitectura standard a bus-ului de tip PCI lasă implementarea de bus master DMA la latitudinea (capabilitatea) vendorului de plăci plug-in.

Majoritatea vendorilor de plăci plug-in pentru platforma PCI vor introduce plăci (ieftine) de tip slave-board ce execută transferuri pe bază de întreruperi repetate ale co-procesorului. Există la ora actuală foarte puține plăci plug-in pentru platforma PCI cu facilități de bus-master, adică ce implementează transferuri de tip DMA între memoria FIFO a plăcii și memoria RAM. National Instruments a devenit unul dintre producătorii de custom ASIC pentru implementarea DMA, prin introducerea chip-ului ASIC MITE ce implementează transferuri DMA de tip *scatter-gather*. Acest chip a fost inventat și realizat special pentru ca plăcile plug-in în platforma PCI să poată beneficia de viteza teoretică maximă a busului PCI ce este de 132Mbytes/sec. Chip-ul MITE este un motor ce transferă date între memoria FIFO a plăcii de achiziție și memoria RAM, fără intervenția co-procesorului. Chip-ul MITE implementează 3 canale de DMA și buffer-e FIFO pentru minimizarea timpului necesar scrierii și rescrierii de date ce are loc în timpul transferului. *Chipul MITE implementează transferuri de date de tip scatter-gather ce constituie o*

îmbunătățire față de tehnologia de transfer a chip-ului 8237A prin eliminarea necesității de reprogramare a procesorului DMA la fiecare sfârșit de pagină.

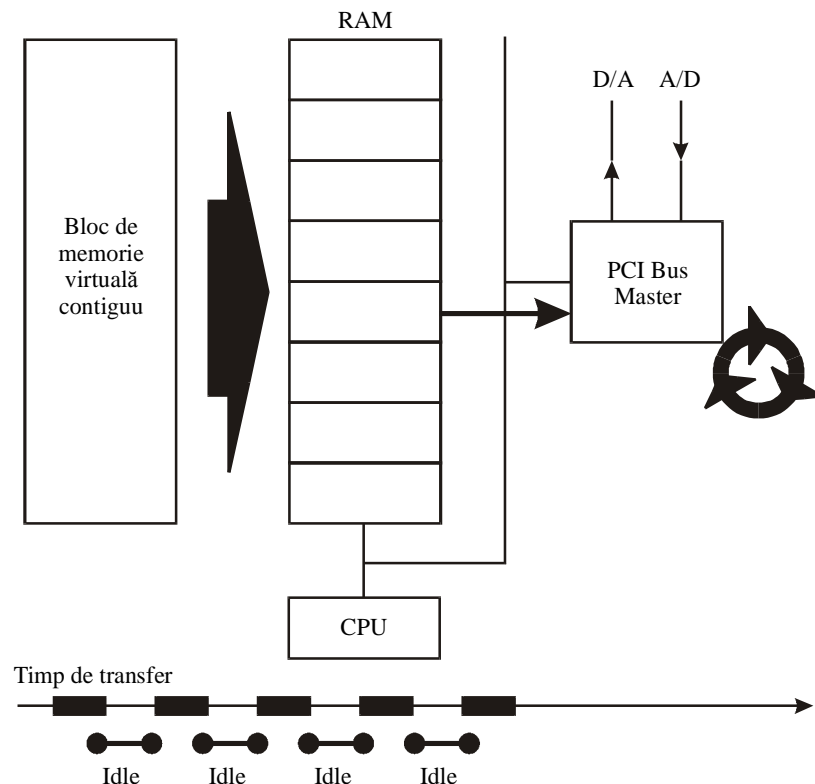


Fig. 4.5 Transfer *scatter-gather* (link-chaining).

Sistemele de operare cum este DOS-ul (Real-Time Operating Systems) au capacitatea de a alocă aplicației un bloc compact de memorie fizică. Aceasta mărește viteza de transfer DMA prin faptul că paginile de memorie în care datele sunt transferate sunt de lungime maximă (64k), deci numărul de reprogramări ale procesorului DMA 8237A, în timpul transferului este minimal. Dezavantajul pe care-l introduc aceste sisteme de operare este acela că memoria ce poate fi alocată este limitată (640Kbytes în DOS). Sistemele de operare evoluate, cum sunt Windows NT și Windows 95 folosesc memorie virtuală, care arată din punctul de vedere al aplicației ca memorie compactă, dar este în realitate o colecție de segmente mici de memorie fizică ce sunt răspândite peste tot, adică au adrese de start și lungimi diferite. În timpul transferului DMA, procesorul 8237A trebuie reprogramat la începutul fiecărui segment nou ce primește date. Acest fapt încetinește viteza de transfer. Chip-ul MITE implementează modul de transfer scatter-gather sau link-chaining, ce aduce următoarea îmbunătățire la tehnica de transfer: *înainte de transfer, chip-ul MITE este programat de către coprocesor cu adresele de început și mărimile mai multor segmente de memorie fizică rezervate aplicației și care vor primi date. Prin aceasta se elimină necesitatea de reprogramare repetată a procesorului DMA în timpul transferului, ceea ce conduce la creșterea vitezei de transfer.*

4.5.3 WINDOWS NT 4.0 ADUCE ÎMBUNĂTĂȚIRI IMPORTANTE PENTRU UTILIZATORII DE INSTRUMENTAȚIE VIRTUALĂ

Windows 95 și Windows NT 4.0 sunt similare prin faptul că introduc următoarele îmbunătățiri față de Windows 3.1:

- capacitatea de a rula aplicații de 16 biți și 32 biți;
- interfață grafică intuitivă;
- preemptive multitasking (capacitatea de a executa aplicații diferite în același timp și capacitatea de a aloca fiecărei aplicații ce rulează segmente de memorie proprii, căderea unei aplicații să nu influențeze pe celelalte ce rulează în același timp);
- arhitectură open network;
- Win32 API, OLE;
- ISA Plug and Play.

Windows NT aduce, în plus față de Windows 95, capacitate multiprocessor, protecție completă, stabilitate, robustețe.

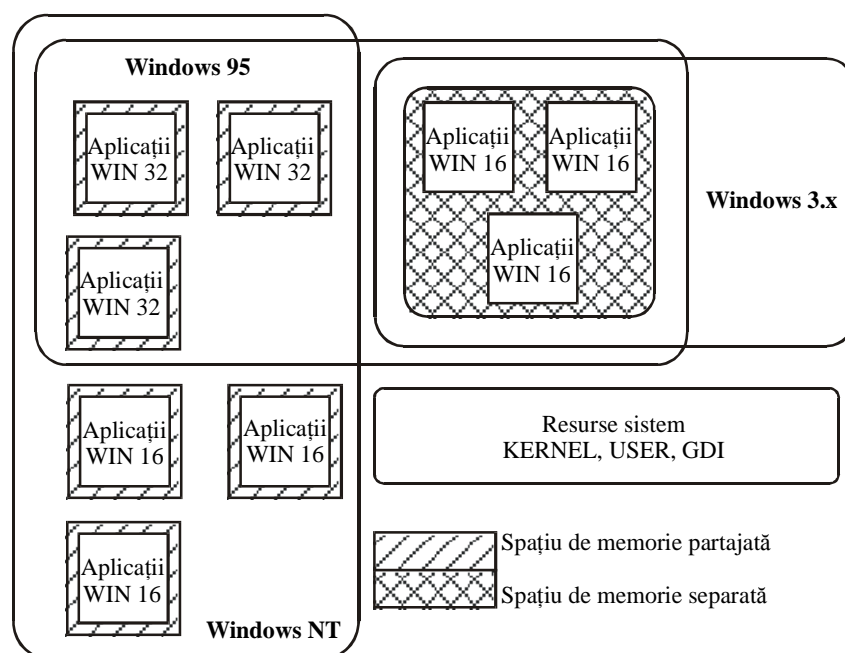


Fig. 4.6 Resursele sistem în cadrul Windows.

Stabilitatea unui sistem de operare depinde de structura de memorie pe care o folosește. Structurile de memorie din Windows sunt: User Memory și Kernel Memory. Aplicațiile sunt plasate în User Memory, pe când driver-ele și părțile critice ale sistemului de operare sunt plasate în Kernel Memory.

Windows 3.x are o singură zonă de adresare pentru toate aplicațiile de 16

biți care urmează a fi rulate. Dacă mai multe aplicații rulează în același timp și una dintre ele cade, atunci întregul sistem de operare cade (se blochează) și este necesară reboot-area sistemului.

Windows 95 îmbunătățește considerabil această structură prin faptul că fiecare aplicație de 32 biți se execută în segmente de memorie separate. Asta înseamnă că aplicațiile de 32 biți ce rulează în același timp nu se influențează reciproc, adică o aplicație ce cade nu poate corupe spațiul de memorie al alteia ce rulează în același timp. Windows 95 păstrează structura de memorie comună pentru aplicațiile de 16 biți.

Windows NT introduce cel mai robust sistem pentru că aplicațiile de 32 biți și cele de 16 biți operează în segmente separate de memorie și nu se pot influența reciproc.

Dacă se folosește Windows NT se face următorul compromis: se sacrifică viteza de execuție în operațiile de achiziție de tip single-point contra stabilității și securității sistemului de operare. Încă din perioada lui Windows 3.x s-a pus problema îmbunătățirii stabilității sistemului de operare. Mulți utilizatori de instrumentație virtuală implementează aplicații critice ce nu pot “cădea”, aplicații industriale de control continuu al unui proces unde nu viteza este importantă, ci siguranța și stabilitatea în execuție. Modelul de memorie implementat de Windows NT protejează împotriva accesului de memorie “interzisă” de către aplicația utilizator, care este sursa principală de cădere a sistemului de operare. Aplicația utilizatorului este plasată în zona de memorie User Mode (primii 0 - 2GB de memorie). Codul utilizatorului este validat de către sistemul de operare, după care este transmis zonei de memorie Kernel Mode (2GB - 4GB) ce conține driver-ele ce acționează direct asupra hardware-ului (cum este placa plug-in cu Ni-daq). Deci, aplicația utilizator în Windows NT, nu adresează direct hardware-ul și nu adresează direct kernel-ul. Aceasta întârzie execuția dar sporește securitatea.

Orice încercare a unei aplicații utilizator de a accesa direct Kernel-ul duce la terminarea aplicației de către sistemul de operare, deci aplicația utilizator în Windows NT nu poate cauza căderea sistemului (visul utilizatorului din mediul industrial).

Stabilitatea sistemului de operare era argumentul forte al utilizatorilor ce vor să folosească sisteme de operare cum este DOS sau sisteme de operare proprii în care nu există suport la nivel de driver pentru aplicație.

4.5.4 TERENUL ESTE PREGĂTIT PENTRU NOILE INSTRUMENTE DAQ

Disponând de un coprocesor rapid, bus-ul PCI cu transfer DMA rapid și eficient, sistem de operare Windows 95 sau Windows NT ce oferă stabilitate, drumul este deschis inovației de instrumente virtuale specializate. Care este impedimentul de a construi un instrument virtual în această platformă, instrument ce arată exact ca și cel tradițional, are aceeași funcționalitate (inclusiv viteză) și în plus beneficiază de toate facilitățile oferite de computerul ce-l găzduiește?

National Instruments a elaborat primele instrumente DAQ specializate și care vor face competiție directă instrumentelor tradiționale ale altor producători. De această dată este vorba de competiție directă la nivel de același instrument, funcție cu funcție, panou cu butoane, utilizare, etc.

Osciloscopul este unul dintre aparatele cele mai utilizate pentru vizualizarea formei de variație în timp a unor semnale sau a dependenței unui semnal în funcție de alt semnal.

Semnalele întâlnite în practică acoperă banda de frecvență de la 0 Hz la 50 GHz; existând, însă, atât semnale periodice cât și semnale tranzitorii, acestea din urmă putând să fie caracteristice unor fenomene cu viteze de variație în timp foarte diferite, se utilizează o mare varietate de osciloscopie generată de marea diversitate a aplicațiilor practice.

Osciloscopia se clasifică în două categorii mari, principial diferite: *analogice și numerice.*

Osciloscopia analogică sunt primele apărute. Ele pot fi clasificate, la rândul lor, în osciloscopia de *uz general*, osciloscopia *cu eșantionare* și osciloscopia *cu memorare pe tubul catodic*.

Osciloscopia analogică de uz general sunt destinate aplicațiilor cu caracter general, și anume vizualizării semnalelor periodice cu frecvența de repetiție de maxim 500MHz.

Osciloscopia analogică cu eșantionare sunt adecvate vizualizării semnalelor periodice cu frecvența de repetiție de până la 50 GHz.

Osciloscopia analogică cu memorare pe tubul catodic sunt destinate vizualizării semnalelor tranzitorii (de la foarte lente până la foarte rapide).

Osciloscopia numerică se deosebesc de cele analogice prin aceea că semnalele de vizualizat sunt mai întâi eșantionate, apoi convertite în formă numerică, memorate într-o memorie numerică, reconvertite în formă analogică și apoi redată pe ecranul tubului catodic.

Apariția și dezvoltarea acestui tip de osciloscopia au fost posibile datorită progresului înregistrat în domeniile convertoarelor analog-digitale (CA/D) și al tehnicii numerice.

Osciloscoapele numerice au banda de frecvență comparabilă cu cea a unui osciloscop analogic de uz general dar oferă o serie de avantaje: precizie superioară, posibilități extinse de măsurare și sincronizare, posibilitatea prelucrării numerice a eșantioanelor memorate. Un astfel de osciloscop poate îndeplini funcțiile mai multor aparate: voltmetru, numărător/frecvențmetru, analizor de spectru etc.

Baza de timp poate fi comandată de circuitul de sincronizare sau de dispozitivul de comandă. Acesta din urmă este, de regulă, realizat cu un microprocesor care furnizează toate semnalele de comandă necesare funcționării întregii scheme. În plus, acesta facilitează măsurarea diferiților parametri ai semnalului și permite prelucrarea numerică a eșantioanelor sale.

O variantă perfecționată utilizează ca bază de timp un convertor digital-analogic (CD/A) comandat de microprocesor. La ieșirea acestui CD/A se obține o tensiune variabilă în trepte egale.

4.5.4.1 TEHNICI DE EȘANTIONARE UTILIZATE ÎN OSCILOSCOAPELE NUMERICE

Osciloscoapele numerice sunt, toate, osciloscoape cu eșantionare. Ca urmare, la aceste osciloscoape nu mai există, ca la osciloscoapele analogice de uz general, o corespondență biunivocă între punctele imaginii de pe ecran și curba semnalului din care acestea provin. Se poate utiliza una dintre următoarele tehnici de eșantionare: eșantionarea secvențială, eșantionarea aleatoare și eșantionarea în timp real.

Eșantionarea secvențială poate fi utilizată numai în cazul semnalelor periodice și constă în prelevarea, în fiecare perioadă a semnalului de vizualizat, a unui singur eșantion, întârziat față de un moment de referință t_R de pe curba semnalului, cu o întârziere care crește de la o perioadă la alta cu Δt . Întârzierea Δt este numită *pas de eșantionare*. Deși perioada de eșantionare este $T + \Delta t$, perioada aparentă de eșantionare este Δt . Deoarece intervalul Δt poate fi făcut foarte mic, rezultă posibilitatea utilizării osciloscopului până la frecvențe mult superioare frecvenței de eșantionare. Spre exemplu, dacă se ia $\Delta t = 0,01 \cdot T$, frecvența de eșantionare este de 100 ori mai mare decât frecvența semnalului.

Eșantionarea secvențială reprezintă o eșantionare *în timp echivalent* deoarece redarea semnalului se face la o altă scară a timpului decât achiziția. Din aceeași categorie face parte și *eșantionarea aleatoare*. Deosebirea față de eșantionarea secvențială constă în aceea că întârzierea (față de momentul de referință t_R) cu care este prelevat eșantionul se modifică aleator de la o perioadă la alta, astfel încât punctele memorate apar într-o succesiune aleatoare pe curba vizualizată a semnalului.

O altă tehnică utilizată în osciloscoapele numerice este eșantionarea *în*

time real, adecvată în cazul fenomenelor singulare și al semnalelor de frecvență relativ joasă, situații în care eșantionarea în timp echivalent fie că nu este aplicabilă, fie că reclamă un timp relativ mare pentru achiziția numărului necesar de eșantioane.

Comparând cele trei tehnici de eșantionare prezentate, se poate concluziona faptul că eșantionarea secvențială permite obținerea unei rezoluții temporale foarte bune datorită faptului că pasul de eșantionare poate fi făcut extrem de mic. Ea nu este însă adecvată pentru vizualizarea fronturilor crescătoare ale impulsurilor cu factor de umplere mic (informația de pe front se pierde). Eșantionarea aleatoare este ideală pentru vizualizarea fronturilor crescătoare ale impulsurilor datorită faptului că eșantionarea apare aleator în raport cu tactul osciloscopului. Eșantionarea în timp real permite achiziția și reconstituirea și a fenomenelor singulare, cu condiția prelevării unui număr suficient de eșantioane.

4.5.5 DAQSCOPE

Instrumentul National Instruments DAQScope este un osciloscop pentru bus PCI, ISA și PCMCIA, cu viteza maximă de digitizare de 20 MHz, trigger analogic și transfer DMA al datelor în timp real.

Acest instrument intră în competiție directă cu osciloscoapele tradiționale realizate de alți vendori, după cum urmează:

	DAQScope 5102	CompuScope CS220	THM565 TekMeter	Fluke PM3335	HP 54603B
Canale de intrare	2	2	2	2	2
Frecvență maximă de eșantionare	20MS/s	20MS/s	25MS/s	20MS/s	20MS/s
Frecvență maximă de repetiție	1GS/s	-	-	-	10GS/s
Lungimea înregistrării	662kB	32kB	256kB	8kB	4kB
Rezoluția (biți)	8	8	8	8	8
Banda de frecvențe	15MHz	20MHz	5MHz	60MHz	60MHz
Trigger	analog și digital	analog	analog	analog	analog

Instrumentul DAQScope este realizat în trei variante, în funcție de bus-ul calculatorului PC folosit: PCI-5102, AT-5102 și DAQCard-5102.

Instrumentul DAQCard-5102 este singurul osciloscop cu 2 canale pe platformă PCMCIA ce poate fi cumpărat astăzi. Cea mai mare diferență dintre DAQScope și osciloscoapele tradiționale este memoria: 662Kbytes la DAQScope comparat cu 2Kbytes la majoritatea instrumentelor tradiționale. Osciloscoapele tradiționale pot pierde puncte în procese de achiziții rapide, de tip continuu, la fiecare grup de 2048 de puncte culese. Nu este excuș ca vendorii de osciloscoape să migreze spre oferta de instrumente plug-in din cauza limitării de memorie de care nu pot trece dacă vor să fie competitivi la preț cu instrumentul DAQScope.

Instrumentul DAQScope poate fi utilizat în mod standalone (de sine stătător) folosind panel-ul din executabilul ce-l însoțește, sau poate fi programat din LabVIEW, BridgeVIEW, LabWindows/CVI, VirtualBench sau Windows 3.x, 95 sau Windows NT.

Sunt prezentate în continuare specificațiile hardware complete ale instrumentului DAQScope PCI-5102/AT-5102/DAQCard-5102.

Input Characteristics

Number of input channels	2 single-ended, simultaneously sampled
ADC resolution	8 bits, 1 in 256
Maximum sample rate	20 MS/s each channel in realtime mode, 1 GS/s with Random Interleaved Sampling mode
Maximum input range	±50 V with a 10 x probe (gain of 1) ±5V with a 1 x probe (gain of 1)
Input signal ranges	±5 V at gain of 1 ±1 V at gain of 5 ±0.25 V at gain of 20 ±50 mV at gain of 100
Overvoltage protection	Input counting AC or DC, software selectable ±42 V powered on or off (without external attenuation) ACHO, ACH 1, ATRIG are protected
Onboard FIFO memory depth	663,552 samples
Data transfers	Programmed I/O, interrupts and DMA (supported on the AT-5102 and PCI-5102 devices only)

Transfer Characteristics

Relative accuracy	±1 LSB typ. ±1.8 LSB max
Differential nonlinearity	±0.3 LSB typ, ±0.5 LSB max
No missing codes	8 bits guaranteed
Offset error after calibration	±1.5 LSB max
Gain error after calibration	±1% max

Amplifier Characteristics

Input impedance	1 MΩ ±1%, 30 pF ±10 pF
-----------------	------------------------

Dynamic Characteristics

Bandwidth	
Small signal (-3 dB)	15 MHz typ.

Large signal (1% THD)	10 MHz typ.
AC coupling LF cutoff	11 Hz (1.1 Hz with 10x probe)
Settling for FS step to +1 %FSR	50 ns typ
System noise	0.5 LSB rms typ
Crosstalk	6.0 dB
S/H characteristics	
Interchannel skew	1 ns
Aperture jitter	1 μ s
Stability	
Recommended warm-up time	15 minutes
Triggers	
Analog Trigger	
Source	ACH(0, 1), ATRIG
Level	\pm FS (ACH(0,1)), \pm 5 V (ATRIG), software selectable
Slope	positive or negative, software selectable
Resolution	8 bits, 1 in 256
Hysteresis	software programmable, up to full scale
Bandwidth	15 MHz
Digital Triggers	
Compatibility	TTL/CMOS
Response	rising or falling edge, software selectable
Pulse width	10 ns min.
RTSI	
Trigger lines	7 I/O
Clock lines	1
Power Consumption	
5V DC (\pm 5%)	250 mA typ.

4.5.5.1 DE CE ESTE IMPORTANTĂ MĂRIMEA MEMORIEI ȘI VITEZA DE TRANSFER DMA LA UN OSCIOSCOP?

Există două arhitecturi pentru digitizarea (codificarea numerică) unui semnal: *Real Time Sampling* și *Equivalent Time Sampling*.

Metoda de digitizare Real Time constă în codificarea punctelor discrete de semnal la viteza ceasului ce controlează achiziția, adică fiecare puls de ceas declanșează o conversie A/D ce se termină înaintea venirii următorului puls. Folosind această metodă, instrumentul DAQScope poate merge până la 20 MHz pe fiecare canal.

Metoda Equivalent Time Sampling (ETS) se aplică semnalelor repetitive și este de două tipuri: *Sequential ETS* și *Random Interleaved Sampling (RID)*.

Convertorul de tip Sequential ETS culege un punct de semnal la fiecare trigger. Dacă semnalul are variație foarte rapidă (transient signal) această metodă “scapă” puncte importante de semnal (ca de altfel și metoda Real Time).

Un alt inconvenient al metodei este acela că sunt necesare 1000 de perioade ale semnalului. pentru a putea culege 1 000 de puncte din semnal.

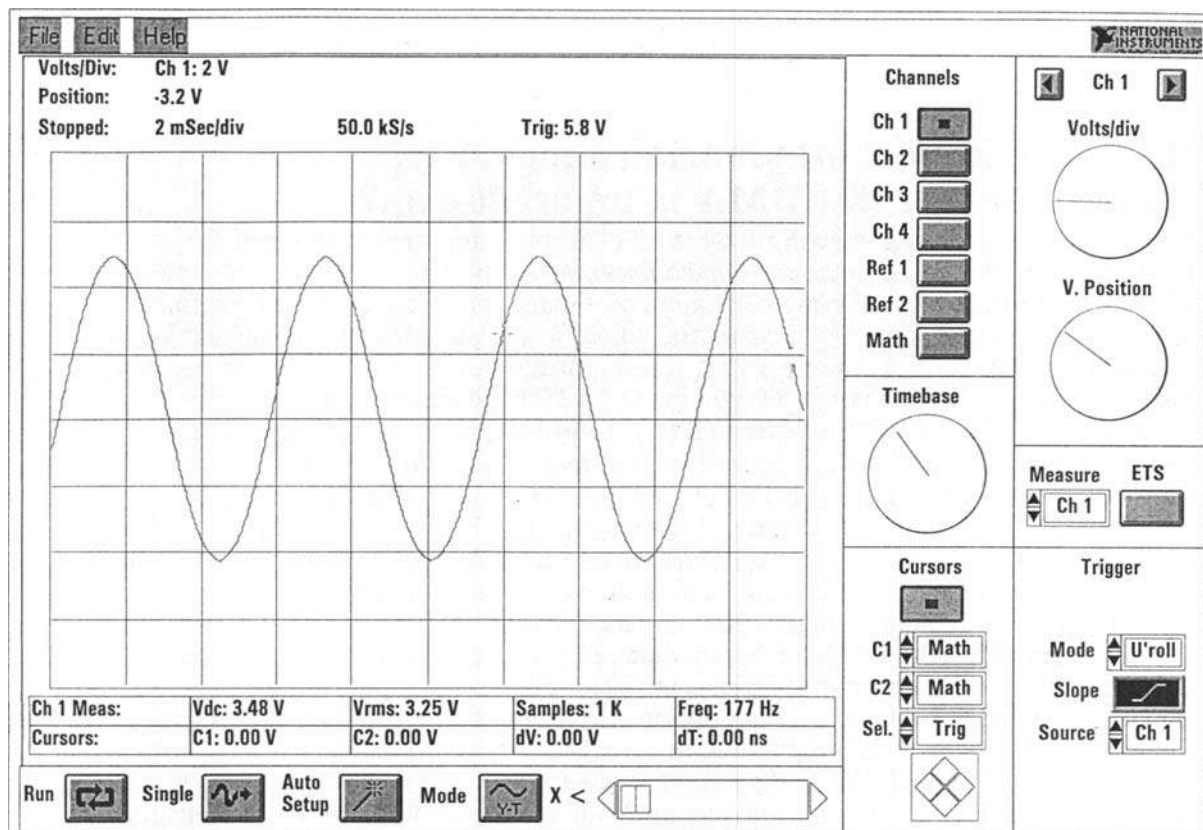


Fig. 4.7 Instrumentul virtual DAQScope.

Convertorul de tip RIS culege un *set de puncte* la fiecare trigger. Asta avantajează (singura metoda sigură) achiziționarea semnalelor cu variație rapidă de tip *transient signal*. Seturile de puncte de semnal, obținute în timpul scanării la viteza de 1 GB/s sunt combinate pentru a da o reprezentare corectă a semnalului achiziționat. Convertorul A/D nu știe când va veni semnalul de trigger, de aceea ceasul care controlează achiziția și punctul de trigger sunt nesincronizate, perioada de timp dintre momentul de trigger și culegerea următorului set de puncte variind de la set la set.

RIS măsoară timpul dintre momentul de trigger și conversia primului punct al setului, pentru fiecare set nou cules. Aceasta permite combinarea corectă a seturilor de puncte într-o imagine foarte curată a semnalului. Culegerea de semnale tranzitorii repetitiv la o viteză de 1GS/s, necesită memorie pentru a putea stoca seturile culese. Dacă nu există memorie suficientă fie că nu se poate implementa metoda RIS, fie implementarea folosește seturi mici de date. Memoria mare și transfer DMA rapid sunt deci absolut necesare pentru a putea implementa această metodă de digitizare.

4.5.6 DAQARB

Instrumentul virtual DAQArb este un generator de semnal cu viteza

maximă de generare de până la 40MHz, rezoluție de 12 biți (asigură o rezoluție de 10 mHz la o viteză de generare de 16 MHz), capabilitate **DDS** (**D**irect **D**igital **S**ynthesis) ce permite schimbarea vitezei de generare “on-the-fly”, memorie de 4 MB până la 16 MB, filtre, atenuare și implementare de secvențe. Acest instrument poate fi folosit ca generator de semnal definit sau arbitrar, fiind unul dintre cele mai avansate la ora actuală. Dintre posibilele aplicații, se pot menționa: controlul de mișcare, teste de modem, testare de materiale, testarea circuitelor integrate digitale, modulare **PSK** (**P**hase **S**hift **K**eying), simulări biomedicale, telemetrie, măsurare de răspuns de frecvențe, comunicare cu sateliți, radar, generare de impulsuri, testare de putere, testare video, testare aparatură electronică, aparat de măsură în industria automobilelor, telecomunicații, etc. DAQArb intră în competiție directă cu generatoare de semnal tradiționale oferite de alți vendori, astfel:

	DAQArb 5411	HP E1445A	HP E1340A	HP 33120A	Standford DS340
Tip	PCI/AT	VXI	VXI	tradițional	tradițional
Frecvență de generare	40 MHz	40 MHz	40 MHz	42 MHz	40 MHz
Memorie	4MB	256 kB	16 kB	16 kB	16 kB
Canale	1	1	1	1	1
Referință de clock extern	da	da	nu	Da	nu
Amplitudine	±10V	±10V	±20V	±10V	±10V
Rezoluție	12 biți	12 biți	12 biți	12 biți	12 biți
Trigger-e	multiple, RTSI	multiple, VXI	multiple	Multiple	unic
Format digi-tal de ieșire	16 biți	16 biți	-	-	-
Marker-e	da	da	nu	Nu	nu
Impedanță de ieșire	50 Ω, 75Ω	50 Ω, 75Ω	50 Ω	50 Ω	50 Ω

Instrumentul DAQArb este disponibil pentru două platforme: placa AT-5411 pentru bus-ul ISA (panel și aplicație VirtualBench-Arb pentru Windows NT/95/3.x, putând fi utilizată și ca placă de achiziție normală din Ni-DAQ, LabView, BridgeView, LabWindows/CVI, C/C++, Visual Basic) și placa PCI-5411 pentru bus-ul PCI.

4.5.7 DAQMETER

Instrumentul virtual DAQMeter este un multimetru digital de 5½ cifre, intrare AC/DC și precizie de 0,02%, putând fi utilizat la măsurarea precisă a tensiunilor, curenților și rezistențelor. Măsurarea tensiunilor continue se face în

intervalele de măsurare de: 20mV, 200mV, 2,0V, 25V și 250V. Măsurarea tensiunilor alternative se face în intervalele de măsurare (valori efective) de: 20mV, 200mV, 2,0V, 25V și 250V. Măsurarea rezistențelor se face în intervalele de măsurare de: 200Ω, 2kΩ, 20kΩ, 200kΩ, 2MΩ și 20MΩ. Viteza de măsurare este de maxim 60 de citiri pe secundă.

Instrumentul DAQMeter este disponibil sub forma plăcii PCMCIA DAQCard-4050, fiind utilizată în computere laptop. Poate fi utilizat cu aplicația VirtualBench-DMM sau din LabView, BridgeView, LabWindows/CVI, Ni-DAQ 5.x. De fapt, placa DAQCard-4050 are la bază un convertor analog-digital de 24 de biți, deci teoretic s-ar putea implementa un multimetru digital cu 7 digiți. Rezoluția reală este de 5½ cifre din cauza unor factori externi, cum este zgomotul.

Instrumentul DAQMeter dispune de trei frecvențe de măsurare, și anume 10Hz, 50Hz și 60Hz.

	DAQMeter 4050	THM 565 TekMeter	Fluke PM 2525
Precizie tensiuni continue	0,02%	0,5%	0,02%
Game de intrare tensiuni continue	20mV...250V	400mV...850V	200mV...2000V
Precizie tensiuni alternative	max. 2%	2%	0,2%
Game tensiuni alternative	20mV...250V	400mV...600V	200mV...2000V
Precizie ohmmetru	0,05%	0,5%	0,1%
Game rezistențe	200Ω...20MΩ	400Ω...40MΩ	200Ω...200MΩ
Precizie curenți continui	0,1%	-	0,1%
Precizie curenți alternativi	max. 2%	-	max. 3%
Game curenți	1μA...10A	-	1μA...10A

4.6 SOFTWARE SPECIALIZAT PENTRU ACHIZIȚIA DATELOR

Avantajul decisiv al sistemelor de măsurare conduse de calculator, față de cele clasice constă în multitudinea de configurări posibile ale componentelor sistemului. Dacă în cazul elementelor de măsurare clasice, soluțiile și rezultatele intermediare trebuiau reluate pentru fiecare domeniu de utilizare, sistemul de măsurare numeric condus de calculator poate realiza o aceeași funcție, schimbându-se numai interfața cu utilizatorul pentru particularizarea acesteia în cazul aplicației date (deci necesitând numai o intervenție la nivel de software).

În cazul în care sistemul de măsurare gravitează în jurul unui PC, prin

utilizarea unui software standardizat (*S-Soft*) asociat echipamentelor de măsurare corespunzătoare, se pot realiza sisteme complexe, mobile sau nu, de achiziție de date, supraveghere sau comandă, particularizate în funcție de necesitatea utilizatorului. În acest mod, posibilitățile de analiză a datelor brute achiziționate și comentariul pertinent al rezultatelor sunt mult extinse față de aria de utilizare a măsurărilor convenționale.

Soluționarea problemelor din gama “achiziției de date”, a “monitorizării proceselor”, a “comenzii” și “reglării” acestora, precum și a “analizei rezultatelor” prin software standardizat se poate face structurând, mai întâi, concepte construite flexibil și care permit schimbul informațional între diversele planuri ale unei prelucrări numerice de date. Se vor detalia câteva dintre problemele ce însoțesc realizarea sistemelor moderne de măsurare, orientate, în ultimii ani, pe configurații ce au ca element central calculatorul personal. După ce a fost stabilită configurația hardware a sistemului de măsurare, optarea pentru una din variantele de software (proiectarea unei aplicații dedicate, configurarea minimală a unuia dintre programele existente pe piața internațională în domeniul măsurărilor sau lăsarea la latitudinea utilizatorului final a acestei alegeri) ce trebuie să asigure prelucrarea, memorarea și evaluarea rezultatelor măsurărilor este un pas esențial, devenit din ce în ce mai dificil de făcut în ultimii ani.

4.6.1 SOFTWARE PENTRU ACHIZIȚIA DE DATE

Deși se consideră că, în general, nu este necesar ca sistemul de calcul pentru achiziția de date să fie extrem de puternic, în condițiile în care aplicația implică o combinație a achiziției propriu-zise cu analiza și reprezentarea grafică a datelor, exigențele transferate de la alte sisteme de software existente pe piață vor impune investiția într-o platformă hard puternică. Cele mai uzuale sisteme pentru controlul achiziției de date sunt PC-urile (generație 486 sau mai nouă) cu magistrala ISA (care permite transferuri pe 16 biți cu viteze de până la 500 kbyte/s; utilizând memorie FIFO, o logică specializată și interfețe IEEE-488 se pot realiza viteze de transfer de până la 1,4 Mbyte/s), EISA (transfer pe 32 biți și semnale suplimentare de control), PCI și platforme ce utilizează magistrala Apple Macintosh II NuBus. Problema portabilității sistemelor de achiziție și prelucrare numerică a datelor măsurate este, de cele mai multe ori, rezolvată de utilizarea Notebook-urilor a căror putere de prelucrare, stocare și posibilități de comunicație pot acoperi până la 90% din necesitățile unei achiziții de date “portabile”.

Paradigma “măsurări conduse de PC” a impus definiții pentru cerințele dar și pentru reprezentarea capacității de a greși a sistemelor de măsurare. În cele mai multe cazuri, astăzi decisivă nu este realizarea hardware, ci mediul software pus la dispoziție utilizatorului de sistemele de măsurare conduse de PC.

Una dintre cerințele esențiale ale acestor sisteme de măsurare este aceea că, pe lângă facilitarea comunicației cu aparate numerice inteligente, acestea să ofere o reprezentare în timp real a diverselor mărimi caracteristice procesului supravegheat. Din acest motiv, cerințele de mai sus pot fi clasificate în două categorii (fig. 4.8):

- cele relative la achiziția datelor;
- cele relative la prelucrarea numerică a informației achiziționate.

În ambele cazuri, însă, este necesară prezența facilității de evaluare interactivă a șirului de date.

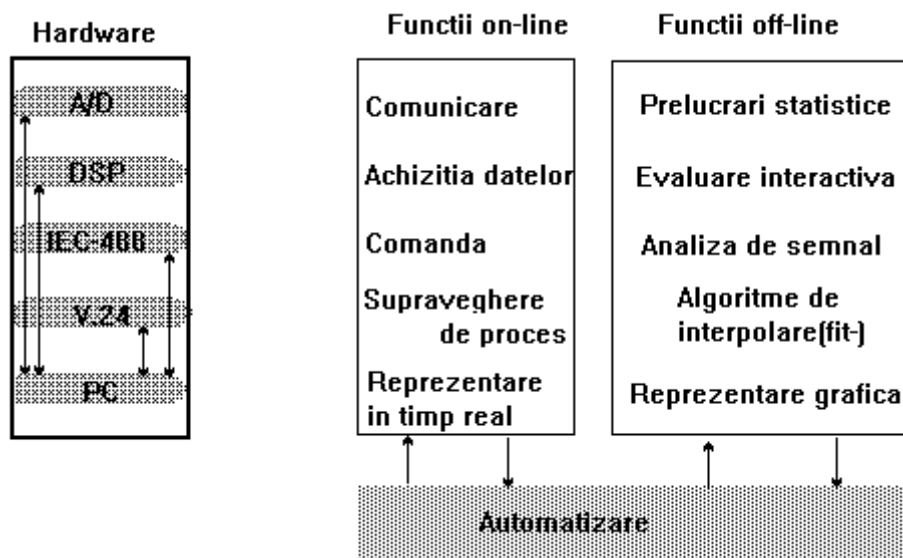


Fig. 4.8 Funcțiile unui pachet software dedicat sistemelor de măsurare.

Noțiunea de standard în domeniul software-ului nu este încă riguros fundamentată. De multe ori, prin acest termen se subînțelege una din variantele următoare:

- software care, datorită unei largi răspândiri, a devenit “standard”;
- software care permite rezolvarea unor anumite probleme “standard”, cum ar fi achiziția de date, trasarea curbelor de variație a unei anumite mărimi, interpolarea etc;
- software realizat pentru un anumit tip de calculator, devenit “standard” (de exemplu, pentru PC).

Cât timp nu au apărut norme privind achiziția, prelucrarea și reprezentarea datelor măsurate, nu se poate vorbi de un standard, ci doar de criterii care pot ușura alegerea pachetelor de programe destinate domeniului măsurărilor. Aceste criterii vor fi detaliate în cele ce urmează:

- capacitatea de dezvoltare ulterioară permisă;
- gradul de “user friendly” al programelor utilizate;
- posibilitatea de a primi asistență tehnică din partea firmei

producătoare;

- independența față de aparatele utilizate (portabilitate relativă la sistemul de măsurare folosit);
- adaptibilitatea.

O problemă aparte o constituie sistemul de operare sub care aceste programe pot rula, distingându-se programe care sunt create pentru:

- ISA (Industry Standard Architecture); în general, acestea nu sunt utilizabile pe OS/2-PC sau stații Unix;
- mașini Unix și calculatoare VMS (DEC); majoritatea au fost sau sunt în proces de "traducere" pentru PC; de exemplu, pachetul LabVIEW a fost conceput, inițial, pentru calculatoare Apple-Macintosh și numai ulterior a fost disponibil pe PC, inclusiv varianta pentru Windows, iar acum și varianta pentru stații SUN.

Deoarece în ultimii ani au apărut atât de multe sisteme de măsurare automate cu programele corespunzătoare, alegerea și, mai ales, valorificarea integrală a unui astfel de software a devenit extrem de dificilă. Criteriile care pot fi utilizate pentru o eventuală clasificare a pachetelor de programe disponibile sunt: adaptibilitatea, viteza de lucru, ușurința în implementare.

Creșterea continuă a complexității hardware-ului și a pretențiilor emise software-ului de prelucrare a datelor măsurate accentuează importanța și valoarea unor *driver*-e soft (nivelul la care se programează direct regiștrii hardware-ului de achiziție, îi administrează funcționarea și îi asigură integrarea cu resursele calculatorului) cât mai bune. Pentru aceasta, un driver trebuie să asigure achiziționarea datelor în *background* cu o rată impusă de utilizator și procesarea lor în *foreground*, controlul momentelor în care are loc transferul datelor către memorie și integrarea completă cu platforma hard folosită. El trebuie să utilizeze cât mai eficient facilitățile oferite de sistemul de operare folosit dar să fie compatibil și cu versiunile ulterioare ale acestuia. În aceste condiții se impune verificarea atentă a limitărilor în exploatarea resurselor sistemului hard (mai ales atunci când driver-ul este conceput de o firmă diferită decât cea care a proiectat suportul hard al aplicației). O astfel de verificare devine imperios necesară în cazul pachetelor integrate care acoperă toate aspectele unei achiziții de date și care, deși sunt utilizabile imediat, au o adaptabilitate limitată.

În elaborarea pachetelor software destinate aplicațiilor în tehnica măsurătorilor și prelucrării digitale a informației sunt implicate toate marile firme care realizează platformele hard specifice astfel că, în ultimul timp, supremația netă a firmei National Instruments, prin proiectul concretizat în LabView este pusă sub semnul incertitudinii cât timp au apărut pe piața dedicată programe proiectate de firmele ce realizează sisteme complexe de măsurare - Hewlett-Packard, Siemens, Hydra, Keithley - programe care exploatează, și ele, ideea programării vizuale și a "băncii" de instrumente virtuale configurabile

pentru aplicația concretă. În această adevărată inflație de programe dedicate măsurărilor cu ajutorul fie al PC-ului (LabView, LabWin, DaisyLab, PowerPoint, HP VEE etc.) fie al stațiilor de lucru (DaisyLab), alegerea devine extrem de grea cât timp platforma hard disponibilă va trebui să integreze sisteme de măsurare proiectate individual cu aparate provenite de la firme diferite, totul pentru a realiza un sistem integrat în jurul unui PC pentru care, la rândul lui, soft-ul disponibil este într-o schimbare continuă. De aceea, în multe cazuri, se pot obține avantaje mari prin lăsarea pe seama soft-ului de firmă realizarea fluxului algoritmilor și testarea acestuia iar programele dedicate aplicației să extindă, doar, funcțiile de prelucrare și analiză a datelor. În vorbirea curentă, prin *standard* se înțelege o direcție, o normă, un stil, care, în general, se obține urmând cerințele obișnuite de calitate și eficiență. Înțelesul cuvântului variază, de fapt, între limitele rigide ale etalonului (metrologiei, chiar) și cele caracteristice unei orientări în genul tematicii standard abordate de un anumit artist (pictor, de exemplu). În domeniul programelor de calcul, noțiunea de standard poate astăzi să releve mai multe aspecte: pe de o parte, faptul că într-un anumit domeniu de utilizare, cu toată diversitatea de produse existente pe piață, fabricanți diferiți înglobează în produsul lor un anumit modul software (de exemplu, cel de comunicație serială pentru diverse instrumente numerice-PC). Aici intervine compatibilitatea dintre diversele produse software care au ca obiectiv îndeplinirea unei aceleiași funcții și, ca avantaj, este de reținut cel al costului redus al consultanței necesare în fiecare caz.

Un alt aspect este cel al furnizorului de pachete software. Din acest punct de vedere se poate defini programul standard tocmai prin ceea ce el nu este, și anume, o colecție de module software standard.

Cea mai importantă definiție ar trebui, însă, să facă apel la sistemul de operare sub care programul în discuție rulează. Facilitățile, etaloanele și chiar “standardele” acestui mediu vor determina și programele adiacente. În fig. 4.9 este reprezentat un astfel de sistem de operare (sau mediu de programare, în cazul nostru) împreună cu elementele sale “standard”. O definiție neambiguă a software-ului standard reiese din observația că acesta trebuie să pună la dispoziție funcții standard unor probleme standard, în vederea construirii unor soluții (mai mult sau mai puțin) standard. În afară de necesitatea ca pachetul de programe să corespundă în limite predefinite unor cerințe caracteristice domeniului de utilizare, programele trebuie să îndeplinească și anumite condiții specifice domeniului informatic, precum și, în plus, să fie compatibil cu acele soluții hardware impuse de utilizator. De exemplu, pentru software-ul din domeniul măsurărilor electrice, este esențială centrarea acestuia pe soluții hardware anumite (PC, stații Unix, rețea intra- sau internet) și protocoale de comunicație adecvate interfețelor specializate (RS-232C, IEEE-488, etc).

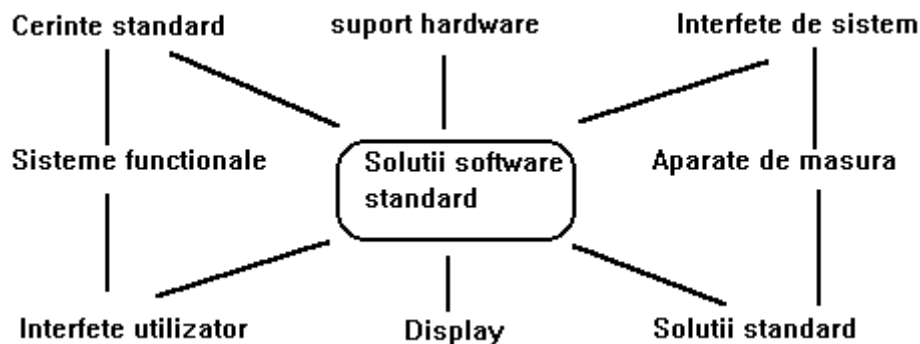


Fig. 4.9 Elementele standardizate într-un sistem de măsurare informatizat.

4.6.2 DETALII PRIVIND CERINȚELE IMPUSE UNUI PACHET SOFTWARE PENTRU MĂSURĂRI ELECTRICE

Pentru utilizatorii de software standardizat, dar și pentru aceia care vor să integreze produse soft ale unei alte firme în produsul propriu, decizia **pentru** sau **împotriva** unui anumit pachet este adesea greu de luat, chiar în condițiile în care prețul produselor subiect al deciziei este simțitor diferit.

Hotărâtor este întotdeauna răspunsul la întrebarea dacă o anumită soluție este economică sau nu. Criteriile pe baza cărora se stabilește dacă prelucrarea numerică a datelor cu ajutorul PC-ului este **economică**, sunt detaliate în cele ce urmează:

- **condițiile impuse de către utilizator:**
 - limba în care se dorește proiectarea și interfațarea cu utilizatorul a software-ului;
 - modul de introducere/preluare a datelor;
 - intervalul de timp de utilizare minim;
 - intervalul de timp de utilizare repetitivă;
 - gradul de explicitare al modulelor componente;
 - viteza de lucru;
 - gradul de dependență față de datele necunoscute;
 - capacitatea de ghidare/help oferită utilizatorului;
 - gradul de standardizare;
 - asistența tehnică în caz de anomalii.
- **flexibilitate:**
 - modularitate;
 - capacitatea de integrare;
 - independența față de sistem (portabilitatea);
 - posibilitatea de a pre-seta un anumit tip de măsurări.
- **performanța:**
 - fiabilitatea (protecția la greșelile utilizatorului, rată de eroare mică);

- resursele hard/soft cerute;
- gradul de generalitate;
- numărul și tipul funcțiilor implementate;
- posibilitatea implementării unor funcții speciale de către utilizator.

În final, se compară calitatea produsului cu costul software-ului. S-a observat că practica de până acum și anume aceea de a proiecta propriul pachet de programe în loc de a apela la un software standardizat, aproape că **a disparut**, din motive economice.

Înainte de alegerea pachetului software, este necesar însă să se verifice care dintre funcțiile sistemului trebuie implementate, pentru că nu întotdeauna se impune realizarea tuturor acestora.

A. Comanda aparatelor de măsurare și/sau a procesului:

- este software-ul compatibil cu aparatele de măsurare necesare (și care, în general, nu sunt cunoscute a priori) sau, cel puțin, are drivere pentru interfețele acestora (RS, HPIB etc.)?
- care este gradul de deschidere al sistemului: ce interfețe software sunt disponibile pentru a suplimenta lipsa unor elemente de comandă?
- cerințele de viteză sunt satisfăcute, de exemplu, pentru transferul datelor?
- care este dimensiunea complexității sistemului supravegheat?
- este o aplicație în timp real sau nu?

B. Achiziția datelor:

- datele trebuie achiziționate on-line și în timp real?
- ce se înțelege, pentru aplicația dată, prin timp real?
- unde este localizată gâtuirea (punctul slab) al canalului de achiziție/prelucrare a datelor?
- în cazul card-urilor de tip plug-in pentru PC:
 - există drivere și pentru alte module hard, în afară de cele furnizate de firma producătoare a software-ului utilizat?
 - se pot implementa drivere pentru hard propriu?
 - există o concepție unitară relativă la driverele soft?
 - este necesară o programare la nivel de regiștri?
 - viteza de rulare este compatibilă cu rata de transfer pe suportul fizic (hard) și este posibilă o scriere continuă pe acest suport, urmând ca prelucrarea să se facă ulterior?
- interfețe seriale:

- există un modul cu interfața suficient de detaliată pentru comunicația cu interfața serială?
- există drivere specifice noilor interfețe seriale (multiple)?
- interfețe CEI-625
 - sunt livrate deja drivere pentru aparate standard interfațate CEI?
 - se poate realiza un astfel de driver?
 - este deja disponibilă o interfață de nivel înalt pentru cele mai importante funcții CEI625?
- formate nestandard
 - este necesar să se schimbe date sau fișiere cu alte sisteme? Ce formate de scriere/citire sunt permise de rutinele software-ului? (ASCII, binar etc.) Există rutine de conversie (filtre)?

C. prelucrarea preliminară:

- ce funcții sunt necesare? (de ex.: calibrări, liniarizări de caracteristici, scalare, filtrare, compresie de date).

D. prelucrarea datelor:

- cât de puternice sunt bibliotecile de funcții?
- ce clase de funcții sunt cuprinse?
 - aritmetica semnalelor (conectarea semnalelor)
 - analiza în timp (determinarea valorilor caracteristice)
 - analiza în frecvență (FFT, spectre, convoluție, corelație)
 - regresie
 - statistică
 - interpolare
 - aproximări
- ce tipuri de date sunt admise? (întregi, reale în virgula fixă/mobilă)
- este necesară o prelucrare on-line? până la ce frecvență maximă a semnalelor?
- este necesară implementarea pe DSP-board?

E. Reprezentarea grafică:

- posibilități de reprezentare grafică generale;
- tehnica ferestrelor multiple
- mai multe curbe, axe, pe o aceeași fereastră;
 - zoom;
 - măsurări cu cursorul;
 - axe logaritmice;

- reprezentări 3D;
- posibilitatea reprezentărilor personalizate a altor tipuri de grafice;
- configurarea detaliilor (culori, for/background).
- este posibilă o reprezentare grafică on-line? până la ce frecvență?
- ce drivere de imprimantă sunt suportate sau există doar facilitatea hardcopy? care este rezoluția maximă de printare? (și analog pentru plotter)
- se pot salva datele aferente unui grafic într-un fișier cu format standard (PCX, TIFF, EPS)?

F. Stocarea datelor:

- este formatul în care se salvează datele specificat complet?
- este asigurată o protecție a datelor împotriva eventualelor greșeli de scriere (mai ales pentru formatele ASCII) ?
- există interfețe de conversie pentru suportul datelor în alte limbaje standard (dBASE, Lotus etc.)?
- este prevăzută comprimarea datelor?

G. Documentarea/prezentarea rezultatelor:

- există forme pre-definite pentru protocoale de măsurare?
 - înscrisuri;
 - antetul firmei;
 - inserție de desene;
 - legendă.
- sunt compatibile astfel de protocoale cu programele standard de procesare text?

Chiar dacă obiectul unei măsurări conduse de calculator poate îmbrăca diverse forme, există un mare număr de caracteristici comune acestor măsurări, astfel că se poate spune că, în general, orice sistem de automatizare (achiziție/prelucrare numerică/arhivarea rezultatelor) este guvernat de un software standard în sinergie cu un software specializat.

Este extrem de important pentru utilizatori să existe posibilitatea ca cerințele unei activități concrete să fie îndeplinite prin implementarea unor funcții specifice în cadrul software-ului. Această cerință a condus la formularea conceptului de interfață deschisă, prin care programatorul își inserează creativitatea și soluțiile proprii. Un alt avantaj este acela că utilizatorul rămâne independent față de casa de software producătoare a pachetului standard.

4.6.3 SCPI (STANDARD COMMANDS FOR PROGRAMMABLE INSTRUMENTATIONS)

Scopul standardului SCPI este acela de a facilita reducerea timpului de dezvoltare pentru programele implementate în sistemele de măsurare și control precum și de a permite schimbarea aparatelor (eventual produse de firme diferite) în cadrul unui aceluiași sistem de măsurare condus de calculator. Acest lucru se realizează prin introducerea unor comenzi standardizate către aparate și a unor tipuri posibile de răspuns al acestora. Spre exemplu, comanda:

“MEASURE:FREQ?”

are aceeași sintaxă pentru toate aparatele de un anumit tip (de exemplu, multimetre) fabricate de diverși producători - consistență verticală - dar și pentru aparate cu funcții diferite (de exemplu, osciloscoape, numărătoare) - consistență orizontală.

SCPI nu este un limbaj de programare (cum sunt, de exemplu, Pascal sau C), ci un limbaj de comenzi care definește comenzile către aparate, parametrii și formatul acestora. SCPI furnizează un șir de caractere ASCII care vor fi transmise prin rutine specifice aparatelor unde vor fi, apoi, prelucrate prin intermediul limbajului **TMSL** (*Hewlett-Packard's Test and Measurement System Language*).

SCPI s-a dezvoltat, în principal, păstrând o compatibilitate cu norma IEEE-488 și utilizând formatul de date al acestei norme. Cu toate acestea, utilizarea SCPI nu este limitată la interfața IEEE-488, comenzile acestuia putând fi transmise și prin interfețele VXI sau RS-232 și, în plus, fiind continuu supus completărilor necesitate de dezvoltarea hardware-ului specific, în urma recomandărilor făcute de un consorțiu de firme, alcătuit din reprezentanți ai leader-ilor pe piața instrumentelor de măsurare: Hewlett-Packard, Tektronix, Fluke (Philips), Keithley, Rohde&Schwarz etc.

Cu toate că SCPI definește comenzile specifice aparatelor de măsurare “inteligente”, nu sunt indicate nici un fel de date tehnice privind exactitatea, rezoluția, domeniul de măsurare ale acestora; astfel nu poate fi în totalitate garantată compatibilitatea aparatelor în cadrul unui sistem comandat prin intermediul SCPI.

Blocurile funcționale ale aparatelor

SCPI împarte fiecare aparat în blocuri funcționale (fig. 4.10), cărora le subordonează un așa-numit arbore de comenzi (*subtree*) exemplificat în fig. 4.11.

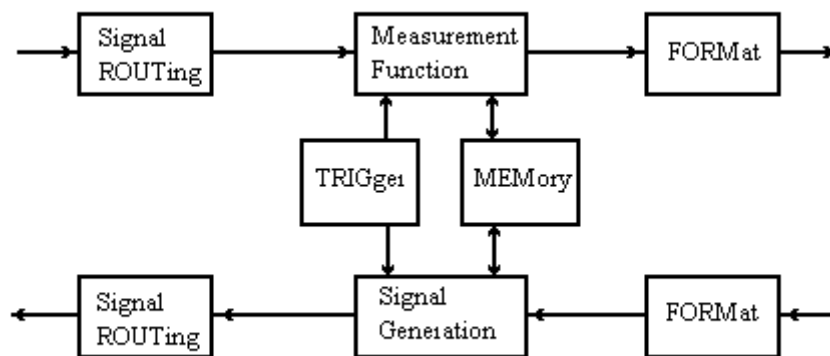


Fig. 4.10 Modelul SCPI al unui aparat programabil.

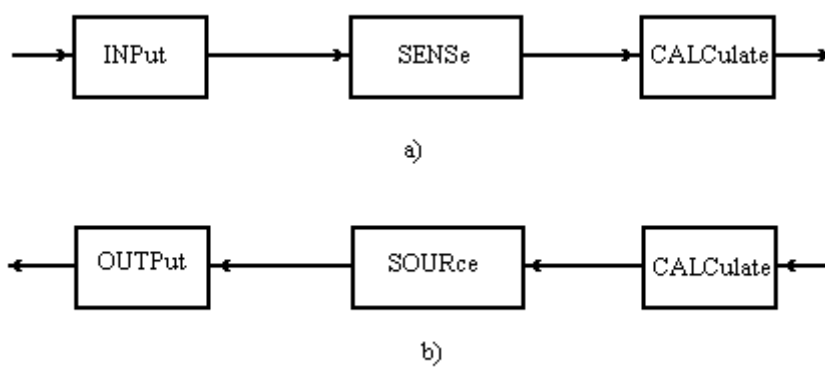


Fig. 4.11 Detalierea unei funcții SCPI:

a) funcția de măsurare; b) funcția de generare a semnalului.

În cele ce urmează sunt detaliate semnificațiile termenilor utilizați în fig. 4.10 și 4.11:

ROUTing	Comanda căii de semnal între bornele aparatului și funcția de măsurare internă;
MEASurement	Conversia semnalului fizic în valoare măsurată (format de date intern), inclusiv operația de condiționare a semnalului;
INPut	Condiționarea semnalelor de intrare (amplificare, atenuare, filtrare etc)
SENSE	Conversia semnalului fizic aplicat la bornele de intrare în valoare măsurată și stabilirea intervalului de măsurare;
CALCulate	Conversia valorii măsurate (de ex., în alte unități sau determinarea timpului de creștere al semnalului);

Signal Generation	Conversia valorilor numerice în semnal fizic, inclusiv condiționarea acestuia;
OUTPut	Condiționarea semnalelor de ieșire;
SOURce	Conversia valorilor numerice în semnale fizice;
TRIG	Sincronizarea, triggerarea operațiilor de măsurare sau generare a semnalelor;
MEMory	Funcția de memorare internă a datelor;
FORMat	Conversia formatelor de date pentru transfer extern (de ex. în format ASCII);

În fig. 4.12 este reprezentată structura arborescentă a blocului funcțional de măsurare de sensibilitate SENSE și a cărei sintaxă este:

:SENS:CURR:AC:RANG:AUTO:DIR:EITH

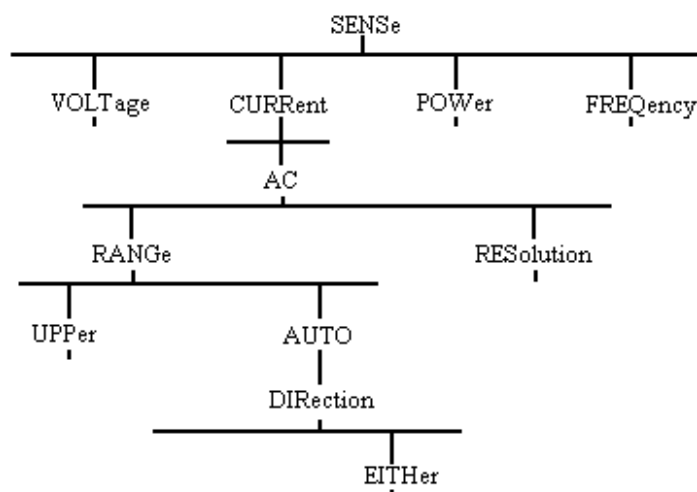


Fig. 4.12 Structura arborescentă a comenzii SENSE.

Nivelele structurale ale comenzilor

Comanda aparatelor de măsurare prin intermediul SCPI se face utilizând mai multe niveluri ale acestora:

- **Secvența de comenzi 1 (Low Level):**

*RST	Resetarea aparatului
:FUNC:VOLT:AC	Selecția funcției de măsurare a tensiunii alternative
:INP:IMP 50	Selectarea impedanței de intrare de 50 Ω
:INIT:IMM	Triggerarea operației de măsurare
FETCH:VOLT:AC?	Înscrierea valorii măsurate în bufferul de ieșire

• **Secvența de comenzi 2 (High Level):**

*RST	Resetarea aparatului
:MEAS:VOLT:AC?	Configurarea aparatului, măsurarea și înscrierea valorii măsurate în buffer-ul de ieșire

Observație: Comenzile de nivel 2 (High Level) necesită mai puține cunoștințe relative la partea hardware a echipamentului de măsurare dar necesită un grad de compatibilitate ridicat între aparatele ce eventual vor îndeplini aceeași funcție. De câte ori, însă, utilizăm configurări diferite de cele uzuale, se recomandă inserarea comenzilor de nivel 1 (low level).

Sintaxa

Program Header grupează cuvintele cheie cu ajutorul cărora se recunoaște o anumită comandă. Aparatele trebuie să accepte comenzi scrise atât cu litere mari cât și cu litere mici, grupate în :

- **Common Command Headers**
- **Instrument Control Headers**

Fiecare **Instrument Control Header** are și o formă de scriere prescurtată. Un aparat comandat prin SCPI trebuie să accepte instrucțiuni exprimate în forma completă (lungă) sau cea scurtă, abaterile de la una dintre acestea fiind semnalizate ca eroare de sintaxă. Forma lungă a comenzii este constituită dintr-un singur cuvânt (maxim 12 caractere, primul din ele fiind literă mare), iar forma scurtă este constituită din primele patru caractere ale celei anterioare, cu excepția cazului când ultimul este o vocală și nu mai este scris.

Comenzile SCPI se structurează **ierarhizat**, nivelele fiind separate în interiorul unei comenzi prin separator.

Dacă un aparat are mai multe canale de măsurare identice, atunci la sfârșitul fiecărui mnemonic se adaugă numărul canalului, implicit fiind considerat canalul numărul 1.

Majoritatea comenzilor există și în varianta în care șirului de comandă i se adaugă sufixul "?", caz în care aparatului i se cere răspunsul către calculator (valoarea măsurată), fără a mai fi nevoie de explicitarea unui *header* pentru aceasta.

Starea de bază după resetarea unui aparat

După primirea comenzii de resetare (*RST), toate aparatele de măsurare accesate prin SCPI se vor găsi într-o stare de bază, definită anterior printr-un alt set de comenzi SCPI. Ca regulă generală:

- toate aparatele trebuie să treacă în modul “*Trigger Idle State*”, ieșirile trebuind deconectate;
- configurarea semnalelor de intrare trebuie făcută fie pe domeniul implicit (*Autorange*) fie pe cel corespunzător rezoluției minime;
- aparatele trebuie să rămână în modul de funcționare (funcția) de bază.

Interpretarea stării (Status Report)

În SCPI este inclusă realizarea completă a modului de interpretare a stării corespunzător normei IEEE-488.2, inclusiv cel al regiștrilor de stare (*Device Status Reporting*). În plus, SCPI posedă comenzi corespunzătoare regiștrilor de operare (*OPERation Status -Register*) și de interogare (*QUEStionable Daten/Signal Status-Register*). În *OPERation Status -Register* se reflectă stările actuale ale aparatului:

Valoare	Stare
0	CALibrating
1	SETTling
2	RANGing
3	SWEping
4	MEASuring
5	waiting for TRIG
6	waiting for ARM
7	CORRecting
8	available to Designer
9	available to Designer
10	available to Designer
11	available to Designer
12	available to Designer
13	INSTRument Summary Bit
14	PROGram running
15	= 0

În *QUEStionable Data/Signal Status-Register* se mențin date referitoare la semnalul măsurat (de exemplu tensiunea, frecvența, timpul etc.) astfel că fiecare bit al acestui registru reprezintă suma a 16 biți din regiștrii prevăzuți în amonte.

5. PREZENTAREA MICROCONTROLLERULUI 80C552 (PHILIPS)

5.1 ARHITECTURA HARDWARE A MICROCONTROLLERULUI 80C552

5.1.1 MEMORIA INTERNĂ A MICROCONTROLLERULUI 80C552

5.1.1.1 MEMORIA DE PROGRAM (PROGRAM MEMORY)

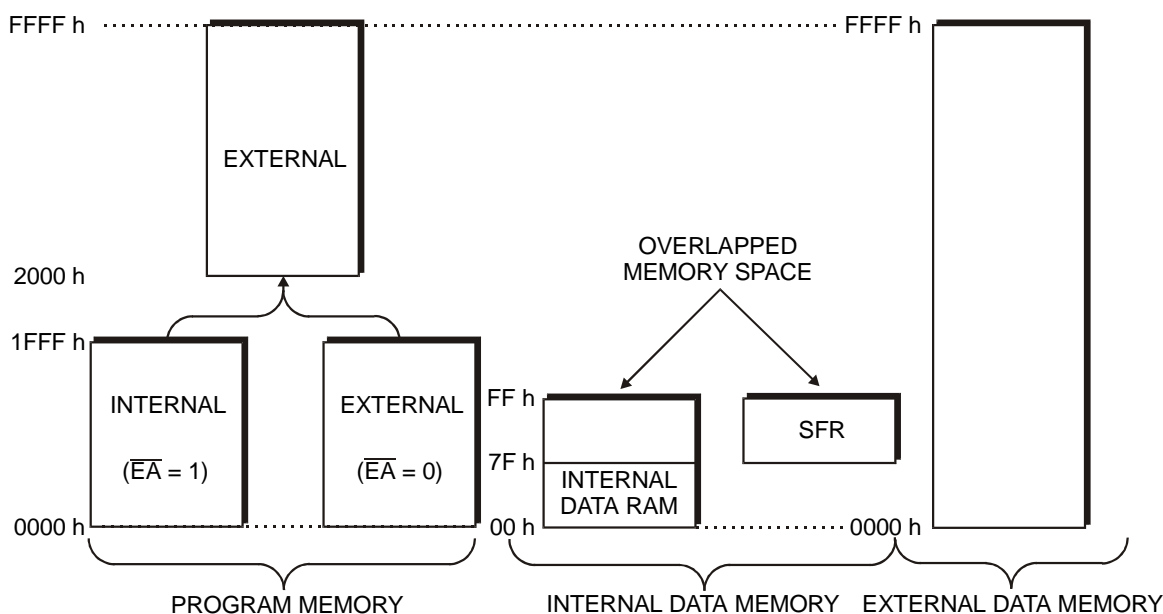


Fig. 5.1 Memoria microcontroller-ului 80C552.

Familia 8XC552 conține 8 kocteți de memorie de program implementată intern, memorie ce poate fi extinsă la 64 kocteți prin utilizarea unei memorii EPROM externă (fig. 5.1). Atunci când pinul EA este forțat la nivel ridicat, microcontroller-ul citește instrucțiunile (ciclu de *fetch*) din memoria internă de program dacă adresa acestora este inferioară valorii 1FFFFH. Locațiile cu adrese cuprinse între 2000H și FFFFH corespund memoriei externe de program. Atunci când pinul EA este forțat la nivel coborât, toate ciclurile de *fetch* (ciclurile de citire a instrucțiunii) sunt executate din memoria externă de program. Locațiile

de memorie de program cu adresele 0003H și 0007H sunt utilizate de rutinele de tratare a întreruperilor.

5.1.1.2 MEMORIA DE DATE (DATA MEMORY)

Microprocesorul conține o memorie internă cu citire/scriere cu adrese cuprinse între 00H și FFH. Aceasta este organizată pe trei secțiuni:

- zona inferioară a memoriei de date, cu capacitatea de 128 octeți, având adrese cuprinse între 00H și 7FH. Adresarea acestei zone de memorie se poate face direct sau indirect;
- zona superioară a memoriei de date, cu capacitatea de 128 octeți. Această zonă de memorie poate fi adresată numai indirect;
- zona registrelor destinate funcțiilor speciale, cu capacitatea de 128 octeți.

Cei 128 octeți ai zonei inferioare a memoriei de date sunt organizați astfel:

- primii 32 de octeți (cu adrese cuprinse între 00H și 1FH) sunt împărțiți în 4 bancuri de câte 8 octeți numite și registre generale și care pot fi apelate în instrucțiuni ca R₀-R₇. Selectarea unuia dintre bancuri la un moment dat se realizează cu ajutorul a doi biți din cuvântul de stare al programului (PSW). Aceste registre pot fi accesate și la nivel de bit;
- 16 octeți cu adrese cuprinse între 20H și 2FH, care pot fi adresați și ca un spațiu de 128 de biți cu adrese cuprinse între 00H și 7FH.

Zona superioară de memorie și zona registrelor destinate funcțiilor speciale împart același spațiu al adreselor de memorie cuprinse între 80H și FFH, deși ele sunt entități fizice distincte. Nu toți cei 128 octeți ai zonei registrelor cu funcții speciale sunt implementați fizic. Registrele destinate funcțiilor speciale ale căror adrese se termină în 0H sau 8H pot fi adresate și la nivel de bit.

Pe lângă cei 256 de octeți de date ai memoriei interne, microprocesorul poate adresa și până la 64 K octeți de memorie externă de date. Adresele memoriei de date externă pornesc tot de la 0000H. Adresa pentru memoria externă poate fi pe un octet sau pe doi octeți. Liniile portului de intrare-ieșire P₀ se folosesc multiplexat pentru a obține octetul inferior al adresei de memorie, respectiv octetul de date citit/scriș în memorie. În cazul în care adresa de memorie este de un octet (octet conținut într-unul din registrele generale ale bancului de registre activ), acesta se poate folosi în conjuncție cu un număr de linii ale portului de intrare-ieșire P₂ care paginează memoria. Dacă adresa este pe doi octeți, aceasta este conținută în registrul destinat funcțiilor speciale DPTR, iar adresarea memoriei se realizează folosind cele 16 linii ale porturilor de intrare-ieșire P₀ și P₂.

5.1.1.3 REGISTRELE CU FUNCȚII SPECIALE

Cele mai multe dintre cele 56 de registre speciale se folosesc pentru controlul *hardware*-ului de periferice aflat pe cip. Altele sunt registre aritmetice (ACC, B, PSW), indicator al stivei (SP), indicatoare de date (DPH, DPL). 16 dintre aceste registre pot fi adresate la nivel de bit. În continuare vor fi prezentate registrele cu funcții speciale cele mai des folosite.

Acumulatorul - este registrul implicit pentru multe instrucțiuni. În cadrul unei instrucțiuni este apelat cu numele **A**. Adresa sa directă este E0H. Poate fi adresat și la nivel de bit. Conținutul său devine 00H la resetarea microprocesorului.

Registrul B - este utilizat obligatoriu în instrucțiunile de înmulțire și împărțire. Poate fi folosit și în alte instrucțiuni. Adresa directă este F0H. Poate fi adresat și la nivel de bit. Conținutul său devine 00H la resetarea microprocesorului.

Cuvântul de stare al programului (PSW) - conține informația de stare. Denumirile și semnificațiile celor 8 biți ai PSW sunt date mai jos:

(MSB)							(LSB)
CY	AC	F0	RS ₁	RS ₀	OV	-	P

- **CY (PSW.7)** - indicatorul de transport din bitul cel mai semnificativ al acumulatorului, în cazul instrucțiunilor aritmetice;
- **AC (PSW.6)** - indicatorul de transport auxiliar. Se folosește pentru operații în BCD;
- **F₀ (PSW.5)** - indicatorul 0 folosit de utilizator;
- **RS₁ (PSW.4)** și **RS₂ (PSW.3)** - folosiți pentru selecția bancului activ al registrelor generale. Sunt setați sau resetati prin program pentru a selecta bancul dorit;
- **OV (PSW.2)** - indicatorul de depășire;
- **P (PSW.0)** - indicatorul de paritate. Este setat sau resetat prin *hardware* la fiecare ciclu de instrucțiune pentru a indica numărul de 1 (par sau impar) din acumulator.

Adresa directă a PSW este D0H.

Indicatorul de stivă (SP) - este un registru de 8 biți. Este incrementat înainte de execuția unei instrucțiuni PUSH sau CALL. Este decrementat după execuția unei instrucțiuni POP sau RET. La resetarea microprocesorului este inițializat la valoarea 07H, deci stiva începe de la adresa 08H. Poate fi încărcat prin program, deci stiva poate începe de la orice adresă. Adresa directă este 81H.

Indicatorul pentru date (DPTR) - este format din doi octeți, cel de

adresă mai mare fiind apelat ca DPH, iar cel de adresă mai mică fiind apelat ca DPL. El păstrează o adresă de 16 biți pentru apelarea memoriei RAM externe (dacă aceasta există). Poate fi manevrat ca un singur registru de 16 biți sau ca două registre de 8 biți. Adresa directă este 82H.

Registrele porturi $P_0 \div P_5$ - sunt registrele *latch* pentru porturile de intrare-ieșire $P_0 \div P_5$. Când se scrie o informație într-un bit al unui registru, aceasta apare la pinul de intrare-ieșire corespunzător. Când se citește informația de la un port de intrare-ieșire, aceasta este memorată în registrul port corespunzător. La resetarea microprocesorului registrele P_0 la P_4 sunt inițializate cu valoarea 00H. Adresele directe ale acestor registre sunt: P_0 - 80H, P_1 - 90H, P_2 - A0H, P_3 - B0H, P_4 - C0H, P_5 - C4H. Registrele $P_0 \div P_3$ sunt adresabile și la nivel de bit.

Buffer serial de date ($S_0\text{BUF}$) - este compus de fapt din două registre separate de 8 biți având aceeași adresă directă, unul pentru transmisie și altul pentru recepție, folosite pentru interfața serială SIO_0 . Când un octet de date este scris în $S_0\text{BUF}$, acesta este introdus în *buffer*-ul pentru transmisie și delanșează începutul transmisiei seriale a acestuia. Când se realizează o citire din $S_0\text{BUF}$, se citește conținutul *buffer*-ului de recepție. Adresa directă este 99H.

Registrele de control - există mai multe astfel de registre care conțin biți de control și de stare pentru întreruperi, temporizatoare/numărătoare și porturi seriale. Ele vor fi descrise la funcțiunile corespunzătoare.

Registrele de temporizare - există trei perechi de registre de câte 8 biți T_0 (TH_0 , TL_0), T_1 (TH_1 , TL_1), T_2 (TMH_2 , TML_2) numărătoare pe 16 biți, și un registru numărător pe 8 biți T_3 . T_3 este folosit ca *watchdog* pentru reinițializarea sistemului după un anumit interval de timp convenabil ales, în cazul când acesta s-a blocat și nu a putut fi reîncărcat temporizatorul T_3 .

Tabelul 4.1 Registrele microcontroller-ului 80C552.

Simbol	Descriere	Adresa directă	Adresa bit, simbol sau funcție alternativă port								Valoare RESET
ACC	Accumulator	E0 H	E7	E6	E5	E4	E3	E2	E1	E0	00 H
ADCH	A/D converter High	C6 H	A/D.9	A/D.8	A/D.7	A/D.6	A/D.5	A/D.4	A/D.3	A/D.2	xxxxxxxx B
ADCON	A/D control	C5 H	A/D.1	A/D.0	ADEX	ADCI	ADCS	ADR2	ADR1	ADR0	xx000000 B
B	B register	F0 H	F7	F6	F5	F4	F3	F2	F1	F0	00 H
CTCON	Capture control	EB H	CTN3	CTP3	CTN2	CTP2	CNT1	CNP1	CNT0	CNP0	00 H
CTH3	Capture 3 High	CF H									xxxxxxxx B
CTH2	Capture 2 High	CE H									xxxxxxxx B
CTH1	Capture 1 High	CD H									xxxxxxxx B
CTH0	Capture 0 High	CC H									xxxxxxxx B
CMH2	Compare 2 High	CB H									00 H
CMH1	Compare 1 High	CA H									00 H
CMH0	Compare 0 High	C9 H									00 H
CTL3	Capture 3 Low	AF H									xxxxxxxx B
CTL2	Capture 2 Low	AE H									xxxxxxxx B
CTL1	Capture 1 Low	AD H									xxxxxxxx B

ELECTRONICĂ APLICATĂ

CTL0	Capture 0 Low	AC H									xxxxxxx B
CML2	Compare 2 Low	AB H									00 H
CML1	Compare 1 Low	AA H									00 H
CML0	Compare 0 Low	A9 H									00 H
DPTR: DPH DPL	Data pointer Data pointer High Data pointer Low	83 H 82 H									00 H 00 H
IEN0	Interrupt enable 0	A8 H	AF EA	AE EAD	AD ES1	AC ES0	AB ET1	AA EX1	A9 ET0	A8 ET1	00 H
IEN1	Interrupt enable 1	E8 H	EF ET2	EE EMC2	ED EMC1	EC EMC0	EB ECT3	EA ECT2	E9 ECT1	E8 ECT0	00 H
IP0	Interrupt priority 0	B8 H	BF --	BE PAD	BD PS1	BC PS0	BB PT1	BA PX1	B9 PT0	B8 PX0	x0000000 B
IP1	Interrupt priority 1	F8 H	FF PT2	FE PCM2	FD PCM1	FC PCM0	FB PCT3	FA PCT2	F9 PCT1	F8 PCT0	00 H
P5	Port 5	C4 H	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	xxxxxxx B
P4	Port 4	C0 H	C7 CMT1	C6 CMT0	C5 MSR5	C4 MSR4	C3 MSR3	C2 MSR2	C1 MSR1	C0 MSR0	FF H
P3	Port 3	B0 H	B7 RD	B6 WR	B5 T1	B4 T0	B3 INT1	B2 INT0	B1 TXD	B0 RXD	FF H
P2	Port 2	A0 H	A7 A15	A6 A14	A5 A13	A4 A12	A3 A11	A2 A10	A1 A9	A0 A8	FF H
P1	Port 1	90 H	97 SDA	96 SCL	95 RT2	94 T2	93 CT3I	92 CT2I	91 CT1I	90 CT0I	FF H
P0	Port 0	80 H	87 AD7	86 AD6	85 AD5	84 AD4	83 AD3	82 AD2	81 AD1	80 AD0	FF H
PCON	Power control	87 H	SMOD	--	--	WLE	GF1	GF0	PD	IDL	00xx0000 B
PSW	Program status word	D0 H	D7 CY	D6 AC	D5 F0	D4 RS1	D3 RS0	D2 OV	D1 F1	D0 P	00 H
PWMP	PWM prescaler	FE H									00 H
PWM1	PWM 1 register	FD H									00 H
PWM0	PWM 0 register	FC H									00 H
RTE	Reset/toggle enable	EF H	TP47	TP46	RP45	RP44	RP43	RP42	RP41	RP40	00 H
SP	Stack pointer	81 H									07 H
S0BUF	Serial 0 data buffer	99 H									
S0CON	Serial 0 control	98 H	9F SM0	9E SM1	9D SM2	9C REN	9B TB8	9A RB8	99 TI	98 RI	00 H
S1ADR	Serial 1 address	DB H	SLAVE ADDRESS								GC
SIDAT	Serial 1 data	DA H									00 H
S1STA	Serial 1 status	D9 H	SC4	SC3	SC3	SC1	SC0	0	0	0	F8 H
SICON	Serial 1 control	D8 H	DF CR2	DE ENS1	DD STA	DC STO	DB SI	DA AA	D9 CR1	D8 CR0	00 H
STE	Set enable	EE H	TG47	TG46	SP45	SP44	SP43	SP42	SP41	SP40	C0 H
TH1	Timer 1 High	8D H									00 H
TH0	Timer 0 High	8C H									00 H
TL1	Timer 1 Low	8B H									00 H
TL0	Timer 0 Low	8A H									00 H
TMH2	Timer 2 High	ED H									00 H
TML2	Timer 2 Low	EC H									00 H
TMOD	Timer mode	89 H	GATE	C/T	M1	M0	GATE	C/T	M1	M0	00 H
TCON	Timer control	88 H	8F TF1	8E TR1	8D TF0	8C TR0	8B IE1	8A IT1	89 IE0	88 IT0	00 H
TM2CON	Timer 2 control	EA H	T2IS1	T2IS0	T2ER	T2B0	T2P1	T2P0	T2MS 1	T2MS 0	00 H
TM2IR	T2 interrupt flag register	C8 H	CF T2OV	CE CMI2	CD CMI0	CC CMI0	CB CTI3	CA CTI2	C9 CTI1	C8 CTI0	00 H
T3	Timer 3	FF H									00 H

5.1.2 STRUCTURA ȘI LUCRUL CU PORTURILE DE INTRARE-IEȘIRE

În fig. 5.2 este prezentată structura microcontroller-ului 80C552. Semnificațiile detaliate ale porturilor sunt prezentate în cele ce urmează.

Portul P_0 - este un port bidirecțional cu drena în gol de 8 biți. Dacă se scrie 1 în el tranzistorul de ieșire este blocat și ieșirea este în starea de impedanță ridicată. În cazul folosirii unei memorii externe (de program - ROM sau de date - RAM), pinii portului P_0 sunt multiplexați între octetul inferior de adresă ($A_0 \div A_7$) și octetul de date citit sau scris din sau în memorie. Poate fi accesat și la nivel de pin ca $P_{0.0} \div P_{0.7}$.

Portul P_1 - este un port bidirecțional de 8 biți. Pinii $P_{1.0} \div P_{1.5}$ sunt prevăzuți cu rezistență internă de pull-up, iar biții $P_{1.6}$ și $P_{1.7}$ sunt cu drena în gol. Pinii acestui port pot îndeplini și următoarele funcții alternative:

- $P_{1.0} \div P_{1.3}$ pot fi și CT0I \div CT3I, adică semnale de intrare pentru temporizatorul T_2 în modul captură (vezi funcționarea timer-ului T_2);
- $P_{1.4}$ poate fi și intrare externă de numărare a temporizatorului T_2 , numită chiar T_2 ;
- $P_{1.5}$ poate fi și intrare de reset a temporizatorului T_2 , adică RT_2 ;
- $P_{1.6}$ poate fi SCL, adică semnal de ceas pentru interfața serială SIO_1 ;
- $P_{1.7}$ poate fi SDA, adică linia de date a interfeței seriale SIO_1 .

Portul P_2 - este un port de 8 biți, bidirecțional, cu rezistențe interne de pull-up. În cazul adresării unei memorii externe, conține octetul superior al adresei $A_8 \div A_{15}$.

Portul P_3 - este un port bidirecțional de 8 biți, cu rezistențe interne de pull-up. Alternativ pinii săi pot îndeplini următoarele funcții:

- $P_{3.0}$ poate fi RxD , adică intrare de date pentru interfața serială, full duplex, SIO_0 ;
- $P_{3.1}$ poate fi TxD , adică ieșire de date pentru SIO_0 ;
- $P_{3.2}$ poate fi $\overline{INT0}$, adică prima întrerupere externă;
- $P_{3.3}$ poate fi $\overline{INT1}$, adică a doua întrerupere externă;
- $P_{3.4}$ poate fi T_0 , adică intrarea de numărare externă pentru temporizatorul T_0 ;
- $P_{3.5}$ poate fi T_1 , adică intrarea de numărare externă pentru temporizatorul T_1 ;
- $P_{3.6}$ poate fi \overline{WR} , adică semnalul de comandă a scrierii în memoria de date (RAM) externă;
- $P_{3.7}$ poate fi \overline{RD} , adică semnalul de comandă a citirii din memoria externă de date (RAM).

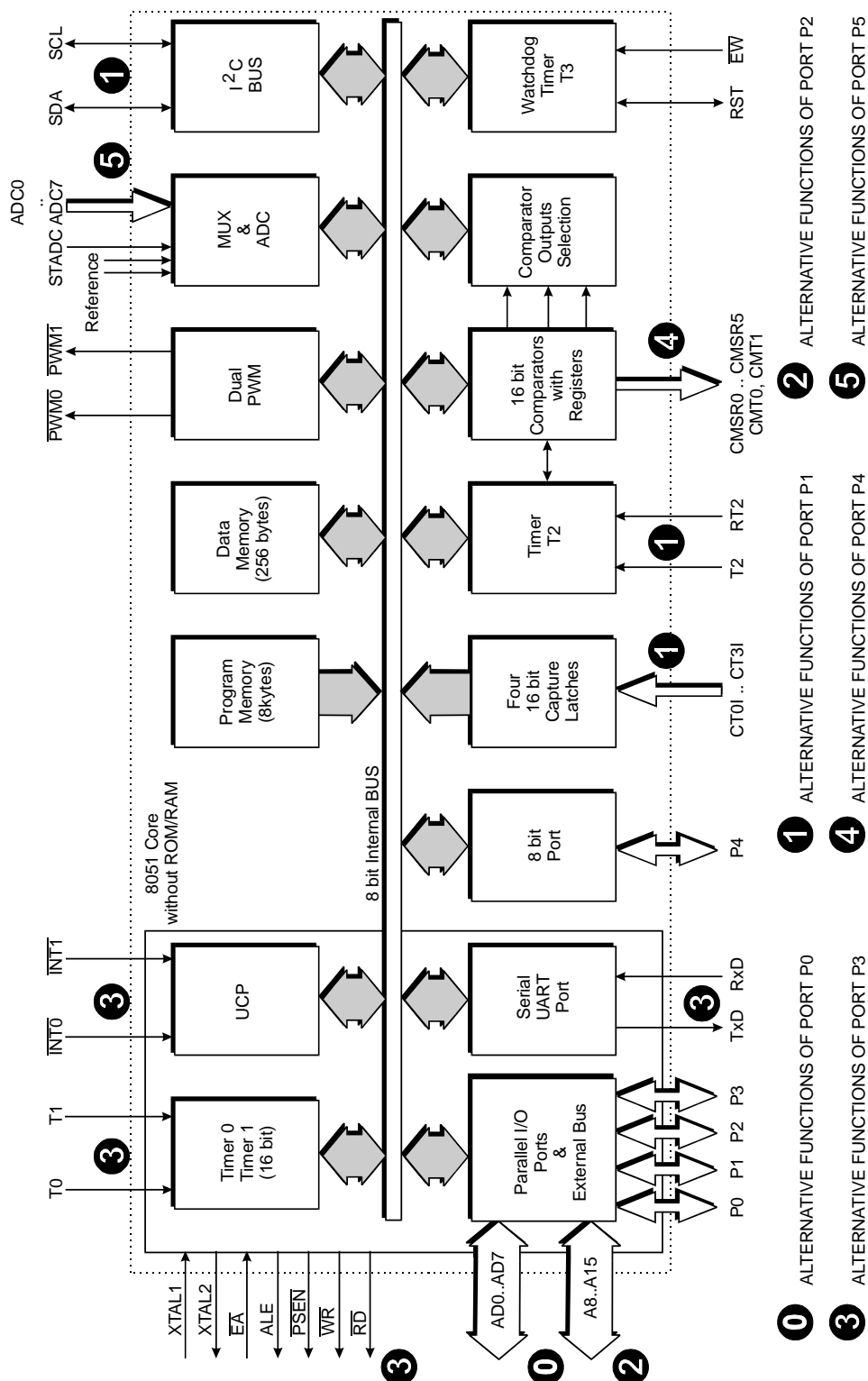


Fig. 5.2 Structura microcontroller-ului 80C552.

Portul P₄ - este un port bidirecțional de 8 biți, cu rezistențe interne de pull-up.

Portul P₅ - este un port de 8 biți numai de intrare. Pinii acestui port pot fi

și $ADC_0 \div ADC_7$, adică 8 canale analogice de intrare ale convertorului analogic-numeric incorporat.

Operațiunea de citire-modificare-scriere la un port. Unele instrucțiuni care citesc un port, citesc registrul pentru funcții speciale corespunzător, iar altele citesc direct starea la momentul respectiv a pinului. Instrucțiunile care citesc registrul sunt acelea care citesc o valoare pe care eventual o modifică și apoi o scriu din nou în registrul. Acestea se numesc instrucțiuni “citește-modifică-scrie”. Următoarele instrucțiuni acționează asupra registrului pentru funcții speciale corespunzător, atunci când în ele apare numele unui port de intrare-ieșire: ANL, ORL, XRL, JBC, CPL, INC, DEC, DJNZ, MOV PX.Y,C (scrie în bitul Y al portului X transportul (**CARRY**), CLR PX.Y (scrie 0 în bitul Y al portului X), SET PX.Y (scrie 1 în bitul Y al portului X). Nu este foarte evident că ultimele 3 instrucțiuni de mai sus sunt de tipul “citește-modifică-scrie”, dar ele sunt deoarece execuția lor constă în citirea tuturor celor 8 biți ai registrului, modificarea corespunzătoare a bitului dorit și rescrierea rezultatului în registrul. Acest lucru se realizează astfel, pentru că, dacă s-ar citi valoarea semnalului la pin (care de fapt reprezintă valoarea bitului corespunzător al registrului pentru funcții speciale asociat portului), din cauza circuitelor comandate nivelul de tensiune ar putea să nu fie cel corespunzător (de exemplu dacă pinul comandă în bază un tranzitor care are emitorul la masă, iar bitul corespunzător din registrul ar fi “1”, tensiunea pe pin ar fi egală cu tensiunea unei joncțiuni *pn* polarizată direct, adică 0,6-0,7 volți, care evident nu este un nivel corespunzător lui “1”).

5.1.2.1 PROGRAMAREA ȘI UTILIZAREA TEMPORIZATOARELOR

Așa cum s-a spus în paragraful relativ la registrele pentru funcții speciale, microprocesorul 80552 are 4 registre temporizatoare / numărătoare. Ele sunt descrise în continuare.

Temporizatorul 0 și temporizatorul 1 Ambele pot fi configurate să funcționeze ca temporizatoare sau ca numărătoare de evenimente. Configurarea este realizată cu ajutorul registrului de control a modului TMOD (el are adresa directă 89H), așa cum este indicat mai jos:

(MSB)				(LSB)			
gate	C/T	M ₁	M ₀	gate	C/T	M ₁	M ₀
Temporizator 0				Temporizator 1			

unde:

- **gate** - temporizatorul/numărătorul “x” este activat numai dacă bitul TR_x din registrul de control TCON are valoarea “1” și acest bit are

valoarea “0” sau semnalul de pe pinul $INT_x/$ are valoarea “1” (vezi portul de intrare-ieșire P_3);

- **C/T** - când acest bit are valoarea “0” registrul funcționează ca temporizator, iar semnalul de numărare este dat de oscilatorul microprocesorului divizat cu 12, care astfel incrementează registrul corespunzător la fiecare ciclu mașină. Dacă acest bit are valoarea “1” registrul funcționează ca numărător, fiind incrementat la fiecare tranziție din “0” în “1” a semnalului de pe pinul T_0 sau T_1 (vezi portul de intrare ieșire P_3). Semnalul de pe acest pin este testat o dată pentru fiecare ciclu mașină în starea S_5 faza P_2 a ciclului, dar numărătorul este efectiv incrementat abia în starea S_3 faza P_1 a ciclului următor celui care a detectat tranziția 1-0 a semnalului T_1 . De aceea frecvența maximă de numărare este $1/24$ din frecvența oscilatorului microprocesorului. Pentru a putea fi detectată tranziția, semnalul trebuie să stea în “0” o durată cel puțin egală cu a ciclului mașină (adică 12 perioade ale oscilatorului);
- **M_1, M_0** - selectează unul din cele patru moduri posibile de operare descrise în continuare.

Modul 0. Este atunci când $M_1=0$ și $M_0=0$. El este identic pentru ambele temporizatoare. În acest mod cele două numărătoare pe 8 biți TH_x și TL_x sunt conectate în cascadă, dar TL_x funcționează ca un numărător pe cinci biți (deci un divizor cu 32), folosindu-se numai primii 5 biți ai acestuia. Când numărătorul trece din starea în care toți biții sunt pe “1” în starea când toți biții sunt pe zero, se setează indicatorul de întrerupere TF_x din registrul de control TCON (TCON.7 pentru TF_1 și TCON.5 pentru TF_0);

Modul 1. Este atunci când $M_1=0$ și $M_0=1$. Este identic cu modul 0, doar că TL_x se folosește ca numărător pe 8 biți (se obține deci un numărător pe 16 biți). El este identic pentru temporizatoarele 0 și 1;

Modul 2. Este atunci când $M_1=1$ și $M_0=0$. Este identic pentru ambele temporizatoare. În acest mod se folosește numai registrul TL_x ca un numărător pe 8 biți cu reîncărcare automată. Conținutul acestuia trece din FFH în 00H, se setează TF_x și se încarcă în TL_x conținutul lui TH_x , fără a fi afectat conținutul lui TH_x . TH_x poate fi încărcat prin program;

Modul 3. Este atunci când $M_1=1$ și $M_0=1$. În acest mod temporizatorul 1 își oprește numărarea, conservându-și conținutul. Este ca și când TR_1 ar fi egal cu “0”. Temporizatorul 0 funcționează ca două numărătoare de 8 biți independente. TH_0 este incrementat de semnalul oscilatorului microprocesorului divizat cu 12, dacă $TR_1=1$. La trecerea conținutului lui de la valoare FFH la 00H este setat TF_1 . TL_0 poate fi incrementat fie de semnalul de la oscilator dacă $C/T=0$, fie de

semnalul aplicat la pinul T_1 dacă $C/T=1$. Incrementarea are loc dacă $TR_1=1$ și $gate=0$ sau $INT_0=1$. La schimbarea conținutului de la valoarea FFH la 00H este setat TF_0 .

Funcționarea celor două temporizatoare este controlată de registrul TCON. Configurația acestuia este dată în continuare:

(MSB)				(LSB)			
TF_1	TR_1	TF_0	TR_0	IE_1	IT_1	IE_0	IT_0

- TF_1 și TF_0 sunt setate așa cum s-a arătat mai sus, și sunt resetate prin *hardware* atunci procesorul intră în rutina de întrerupere;
- TR_1 și TR_0 inhibă (când sunt egale cu 0) sau permit numărarea (când sunt egale cu 1) a temporizatorului corespunzător. Sunt setate sau resetate prin program.

5.1.2.2 INTERFAȚA SERIALĂ SIO₀

Aceasta este o interfață full-duplex, adică poate transmite și recepționa date simultan. Are *buffer* la recepție, deci poate începe recepționarea unui nou octet de date înainte ca cel deja recepționat să fie preluat. Așa cum s-a menționat în paragraful referitor la registrele cu funcții speciale (vezi registrul SBUF), o scriere în SBUF încarcă registrul pentru transmisia datelor, iar citirea din SBUF se face din registrul de recepție a datelor, separat fizic de cel de transmisie. Interfața serială SIO₀ poate lucra în 4 moduri posibile. Acestea sunt următoarele:

Modul 0. Este atunci când $SM_0=0$ și $SM_1=0$ (biții 7 respectiv 6 din registrul de control al interfeței SCON - unul din registrele pentru funcții speciale). În acest mod interfața serială funcționează semiduplex. Se transmit câte 8 biți. Pe linia RxD ($P_{3,0}$) se emit și se recepționează date. Semnalul de ceas al transmisiei/recepției se transmite/recepționează pe linia TxD ($P_{3,1}$). Transmisia este inițiată prin scrierea unui octet de date în registrul SBUF. Ceasul pentru transmisie are o frecvență egală cu 1/12 din frecvența oscilatorului. La sfârșitul transmisiei celor 8 biți de date se setează bitul de întrerupere TI (bitul 1 din SCON), care apoi trebuie șters prin program. Recepția este validată atunci când bitul REN (bitul 4 din SCON) este 1 (el este setat și resetat prin program) și bitul de întrerupere pentru recepție RI (bitul 0 din SCON) este 0 (el este trecut în 1 la sfârșitul recepționării celor 8 biți de date și trebuie trecut în zero prin program). Datele sunt emise începând cu bitul cel mai puțin semnificativ.

Modul 1. Este atunci când $SM_0=0$ și $SM_1=1$. În acest mod transmisia este asincronă pe 10 biți: un bit de start având valoarea 0, 8 biți de date (primul este cel mai puțin semnificativ) și un bit de stop având

valoarea 1. Datele sunt emise pe linia TxD și recepționate pe linia RxD. Recepția este inițiată la detectarea unei tranziții din 1 în 0 a liniei RxD, dacă bitul REN=1. Octetul de date este încărcat în SBUF, iar bitul de stop este introdus în RB8 (adică bitul 2 din SCON) și poziționează în 1 bitul de întrerupere pentru recepție RI (discutat la modul zero) dacă acesta era 0, și dacă SM₂=0 (bitul 5 din SCON, setat sau resetat prin program) sau bitul de stop recepționat este 1. Rata transmisiei este dată de semnalul de depășire (TF1) al temporizatorului 1. Transmisia este inițiată prin scrierea unui octet de date în SBUF. În momentul transmisiei bitului de stop se poziționează în 1 bitul de întrerupere la transmisie SI. Recepția este inițiată în urma detecției unei tranziții 1-0 pe linia RxD.

Modul 2. Este atunci când SM₀=1 și SM₁=0. În acest mod transmisia este full duplex, asincronă, cu 11 biți transmiși (pe linia TxD) și recepționați pe linia RxD), astfel: un bit de start egal cu 0, 8 biți de date (primul este cel mai puțin semnificativ), un al nouălea bit de date programabil și un bit de stop egal cu 1. Cel de al nouălea bit de date în cazul transmisiei reprezintă conținutul lui TB8 (bitul 3 din SCON), iar în cazul recepției este introdus în RB₈. Transmisia este inițiată prin scrierea unui octet de date în SBUF. În timpul transmisiei bitului de stop este setat TI. Recepția este identică cu cea din modul 1, numai că în RB₈ se introduce al nouălea bit de date. Încărcarea datelor recepționate în SBUF și RB₈ și setarea lui RI este validată numai dacă RI=0 și, SM₂=0 sau al nouălea bit de date este 1. Altfel octetul de date recepționat se pierde. Rata de transmisie poate 1/32 din frecvența oscilatorului dacă SMOD=1 (bitul 7 din registrul pentru funcții speciale PCON) sau 1/64 din frecvența acestuia dacă SMOD=0.

Modul 3. Este atunci când SM₀=1 și SM₁=1. El este identic cu modul 2, cu singura deosebire că rata transmisiei de semnalul de depășire TF₁ al timerului T₁.

OBSERVAȚIE: În cazul modurilor 1 și 3 când este folosit temporizatorul 1 pentru fixarea ratei de transmisie, întreruperea acestuia trebuie dezactivată. El poate fi folosit ca “temporizator” sau “numărător” în oricare din cele 3 moduri posibile de funcționare. De obicei este folosit ca “temporizator” în modul 2 cu autoîncărcare. În acest caz rata de transmisie se calculează cu formula:

$$\frac{2^{\text{SMOD}}}{32} \times \frac{f_{\text{osc}}}{12 \times [256 - (\text{TH1})]} \quad (5.1)$$

5.1.2.3 IEȘIRILE MODULATE ÎN DURATĂ

Microcontroller-ul 80C552 posedă două canale de ieșire modulate în durată. Aceste ieșiri generează impulsuri ale căror durate și factori de umplere pot fi programate. Frecvența de repetiție a impulsurilor este controlabilă prin intermediul registrului PWMP, având lungimea de 8 biți. Expresia frecvenței impulsurilor de ieșire este dată de următoarea ecuație:

$$f_{\text{PWM}} = \frac{f_{\text{OSC}}}{2 \times (1 + \text{PWMP}) \times 255} \quad (5.2)$$

Registrul PWMP asigură semnalul de ceas pentru un numărător, atât registrul PWMP, cât și numărătorul fiind comune pentru ambele canale de ieșire. Numărătorul funcționează modulo 255. Valoarea de 8 biți a numărătorului este comparată cu două registre, PWM0 și PWM1, asociate fiecare câte unui canal de ieșire. Dacă conținutul acestor registre este mai mare decât valoarea curentă a numărătorului, atunci semnalele de ieșire sunt forțate la nivel coborât. În schimb, dacă conținutul acestor registre este mai mic, sau cel puțin egal, cu valoarea curentă a numărătorului, atunci semnalele de ieșire sunt forțate la nivel ridicat - HIGH. Ca urmare, duratele impulsurilor de ieșire sunt determinate de conținutul registrelor PWM0 și PWM1, denumite și registre de prescalare.

Prin integrarea semnalelor de la ieşirile modulate în durată se poate obţine un convertor digital-analog dual.

În fig. 5.3 este prezentată diagrama funcțională a celor două canale de ieșire modulate în durată.

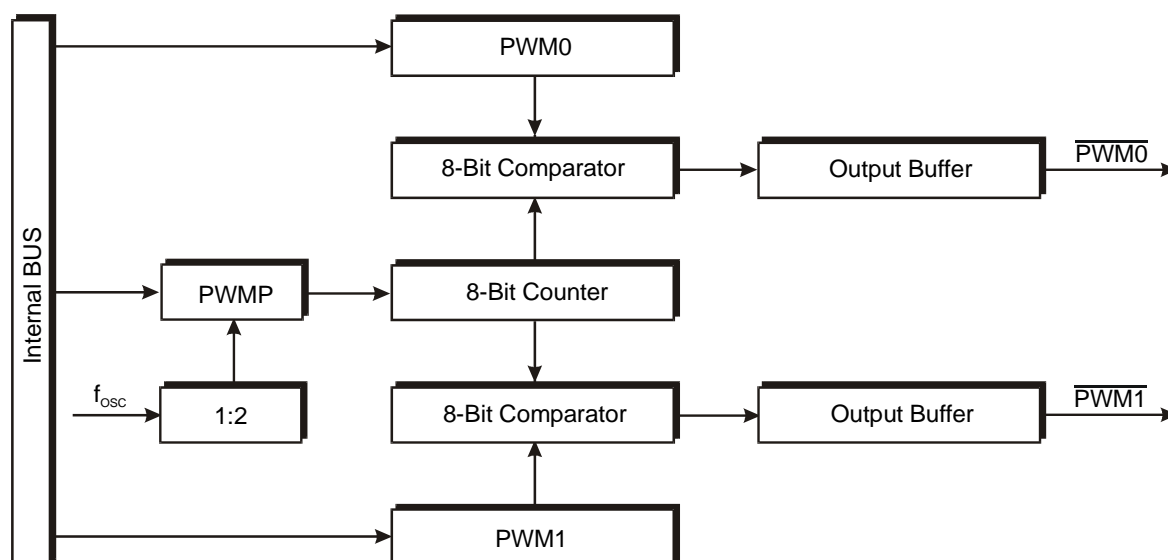


Fig. 5.3 Structura ieșirilor modulate în durată.

Încărcarea registrelor de prescalare cu valorile 00H sau FFH determină ca ieșirile canalelor să rămână constante la nivel ridicat sau coborât.

Reprogramarea registrelor de prescalare determină modificarea imediată a

ieșirilor modulate, nefiind nevoie să se aștepte până la terminarea perioadei curente.

Factorul de umplere al impulsurilor de la ieșire rezultă ca fiind:

$$\gamma = \frac{\text{PWM}}{255 - \text{PWM}} \quad (5.3)$$

5.1.2.4 SECȚIUNEA ANALOGICĂ A MICROCONTROLLERULUI

Secțiunea analogică este reprezentată de un multiplexor analogic și de un convertor analog-digital cu aproximații succesive, cu rezoluția de 10 biți, ce furnizează rezultatul în cod binar direct (fig. 5.4).

Tensiunea de referință a convertorului analog-digital se aplică structurii prin intermediul unui pin dedicat acestei manipulări.

Un ciclu de conversie durează 50 de cicli mașină, ceea ce înseamnă aproximativ 50 μs pentru frecvența ceasului de 11,0592 MHz. Codul binar de ieșire nu prezintă discontinuități în funcția de transfer a convertorului analog-digital.

Atât intrările multiplexorului analogic, cât și intrarea convertorului analog-digital acceptă tensiuni de intrare în gama (0 ÷ +5)V.

Conversia analog-digitală este efectuată prin metoda aproximațiilor succesive; în fig. 5.5 sunt reprezentate elementele componente ale convertorului analog-digital cu aproximații succesive. Structura conține un convertor digital-analogic (DAC) care convertește conținutul registrului de aproximații succesive într-o tensiune V_{DAC} . Această tensiune este comparată cu tensiunea de intrare, V_{in} ; ieșirea comparatorului este furnizată circuitului de comandă a registrului de aproximații succesive, care controlează funcționarea acestuia.

O conversie este inițiată prin setarea bitului ADCS (ADC Start) din registrul ADCON (A/D Control). Acest bit poate fi modificat atât prin *software*, cât și prin *hardware* sau combinat.

Declanșarea unei conversii analog-digitale exclusiv prin mijloace software se poate face doar atunci când bitul ADCON.5 (ADEX) este setat la “0”; începerea unui ciclu de conversie se efectuează prin setarea bitului de control ADCS. La inițierea prin software a unei conversii, aceasta este demarată la începutul ciclului mașină care urmează celui în care a fost setat bitul ADCS.

Declanșarea unei conversii analog-digitale prin mijloace combinate software-hardware se poate face doar atunci când bitul ADCON.5 = “1”; inițierea unui ciclu de conversie poate fi efectuată fie la fel ca în cazul precedent, prin setarea bitului de start (ADCS), fie prin aplicarea unui front crescător pinului extern STADC. În această ultimă situație, frontul aplicat trebuie precedat de un nivel logic coborât, care să dureze minimum un ciclu

mașină, respectiv urmat de un nivel logic ridicat, cu durata de minimum un ciclu mașină. Tranziția aplicată pinului STADC este recunoscută la sfârșitul ciclului mașină curent, iar conversia este inițiată la începutul următorului ciclu mașină.

Următoarele două cicluri mașină sunt folosite pentru inițializarea convertorului analog-digital. La sfârșitul primului ciclu, se citește indicatorul de stare ADCS; dacă indicatorul este citit pe durata efectuării unei conversii, rezultatul citirii acestui indicator constă în poziționarea acestuia în “1”. La sfârșitul celui de-al doilea ciclu mașină începe eșantionarea semnalelor de intrare.

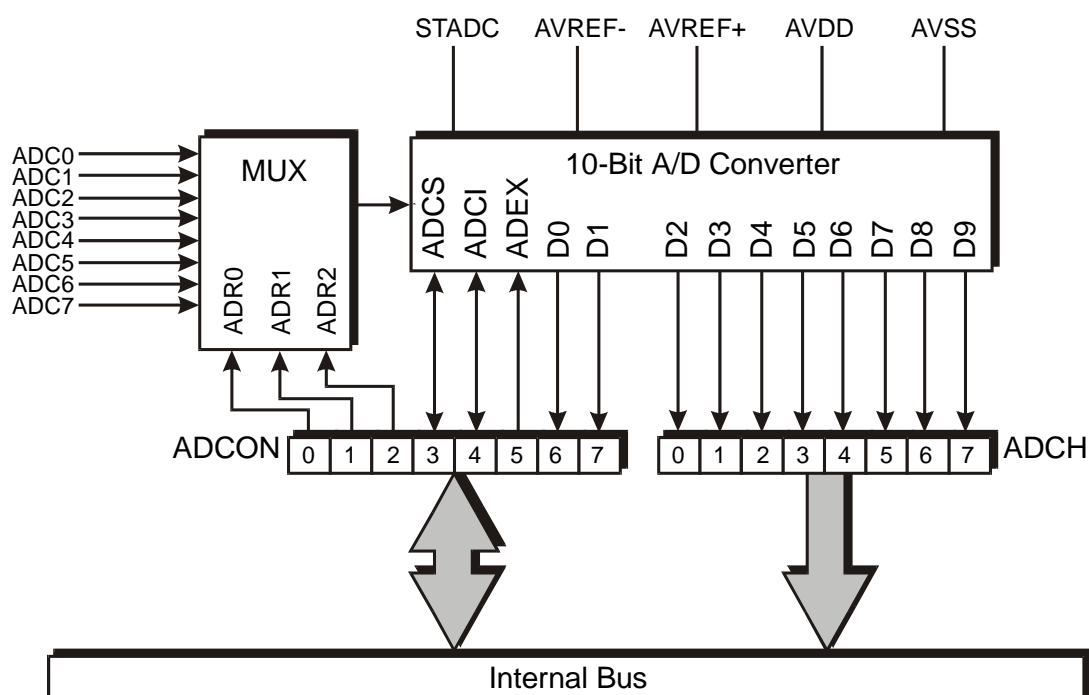


Fig. 5.4 Secțiunea analogică a microcontroller-ului 80C552.

Pe durata următoarelor 8 cicluri mașină este eșantionată tensiunea de intrare aplicată pinului, anterior selectat, al portului P₅. Această tensiune trebuie să rămână stabilă pe această durată pentru a se obține un rezultat corect. Se admite o viteză de variație a tensiunii eșantionate de cel mult 10V/ms.

În continuare, se efectuează conversia analog-digitală propriu-zisă. Aceasta se desfășoară prin setarea bitului cel mai semnificativ la “1”, de către circuitul de control al registrului de aproximații succesive; ieșirea registrului SAR este convertită într-o tensiune proporțională și comparată cu tensiunea de intrare. Dacă aceasta este mai mare, bitul MSB rămâne setat la “1”; în caz contrar, acest bit este resetat. Procesul continuă în ordine inversă a importanței biților, până când toți cei zece biți au fost testați. În acest moment, rezultatul conversiei este conținut în registru de aproximații succesive.

Sfârșitul conversiei analog-digitale pe 10 biți este semnalizată prin setarea

bitului ADCI (ADCON.4) în cadrul registrului de comenzi și stare. Cei mai semnificativi 8 biți ai rezultatului sunt memorați în registrul ADCH. Cei mai puțin semnificativi doi biți sunt memorați în biții ADCON.7 și ADCON.6 ai registrului de comenzi și stare. Utilizatorul poate ignora cei mai puțin semnificativi doi biți ai rezultatului și poate utiliza doar cei mai semnificativi 8 biți, memorați în registrul ADCH.

Fig. 5.6 ilustrează desfășurarea procesului de conversie analog-digitală cu aproximații succesive.

În orice situație, durata unei conversii este de 50 cicli mașină. Bitul de sfârșit de conversie, ADCI, este setat, iar bitul de start conversie, ADCS, este resetat după 50 de cicli mașină de la poziționarea pe nivel ridicat al acestuia din urmă.

Biții de control ADCON.0, ADCON.1 și ADCON.2 sunt utilizați pentru comanda intrărilor de selecție ale multiplexorului analogic, de tip 8:1.

O rutină de conversie aflată în desfășurare nu este afectată de o comandă de start, indiferent de natura sa (*hardware* sau *software*).

Rezultatul obținut la încheierea unui proces de conversie rămâne neafectat, cu bitul ADCI poziționat pe “1”, chiar dacă se comandă intrarea în modul “inactiv” (*idle*).

La intrarea în modul de lucru “*idle*” sau “*power-down*” al microcontroller-ului, o conversie analog-digitală aflată în plin proces de desfășurare este întreruptă. Rezultatele parțiale obținute până acum sunt iremediabil pierdute, nefiind transferate în registrele rezultat și/sau în registrul de comenzi și stare.

Convertorul analog-digital dispune de pini proprii de alimentare (AV_{DD} și AV_{SS}) și de doi pini pentru tensiunea de referință externă (V_{ref+} și V_{ref-}). Rezultatul conversiei poate fi calculat prin utilizarea următoarei caracteristici de transfer:

$$N = 2^{10} \times \frac{V_{IN} - V_{REF-}}{V_{REF+} - V_{REF-}} = 1024 \times \frac{V_{IN} - V_{REF-}}{V_{REF+} - V_{REF-}} \quad (5.4)$$

Prin utilizarea numai a celor mai semnificativi 8 biți ai rezultatului, conținută în registrul ADCH, caracteristica de transfer devine:

$$N = 2^8 \times \frac{V_{IN} - V_{REF-}}{V_{REF+} - V_{REF-}} = 256 \times \frac{V_{IN} - V_{REF-}}{V_{REF+} - V_{REF-}} \quad (5.5)$$

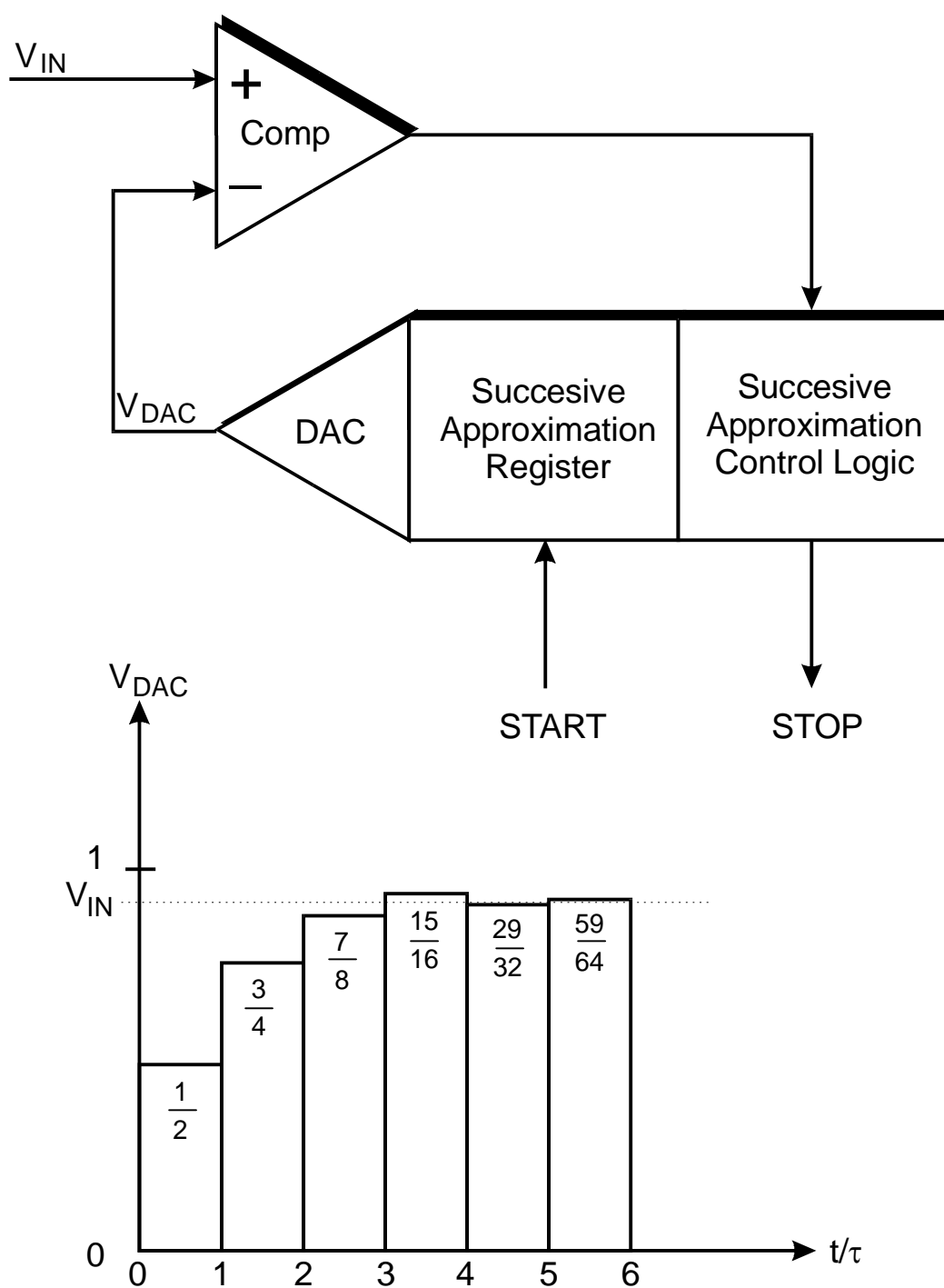


Fig. 5.5 Convertorul analog-digital.

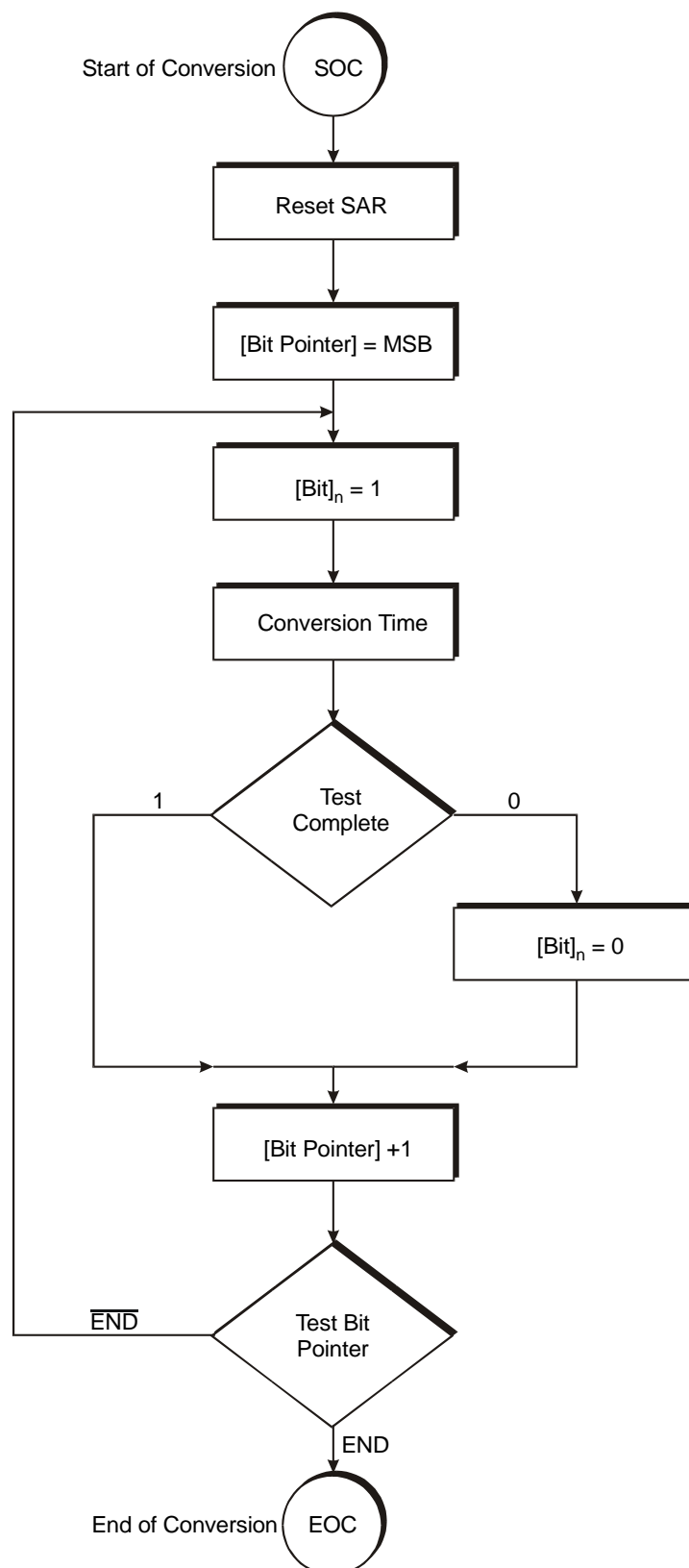


Fig. 5.6 Organigrama de desfășurare a unei conversii.

5.1.2.5 MĂSURAREA INTERVALELOR DE TIMP PRIN UTILIZAREA REGISTRELOR DE CAPTARE A EVENIMENTELOR

Atunci când un eveniment extern recursiv este reprezentat sub forma unui front crescător sau descrescător care este aplicat unuia dintre cei patru pini de captare a evenimentelor, intervalul de timp dintre două evenimente poate fi măsurat folosind temporizatorul T_2 și unul dintre cele patru registre de captare a evenimentelor, CT0, CT1, CT2 sau CT3. La apariția unui eveniment, conținutul timer-ului T_2 este transferat în registrul de captare a evenimentelor selectat și este generat un semnal de întrerupere specific acestui registru, CT0I, CT1I, CT2I sau CT3I. Indicatorii de stare anterior menționați reprezintă patru dintre cei 8 biți ai registrului de funcții speciale, denumit TM2IR.

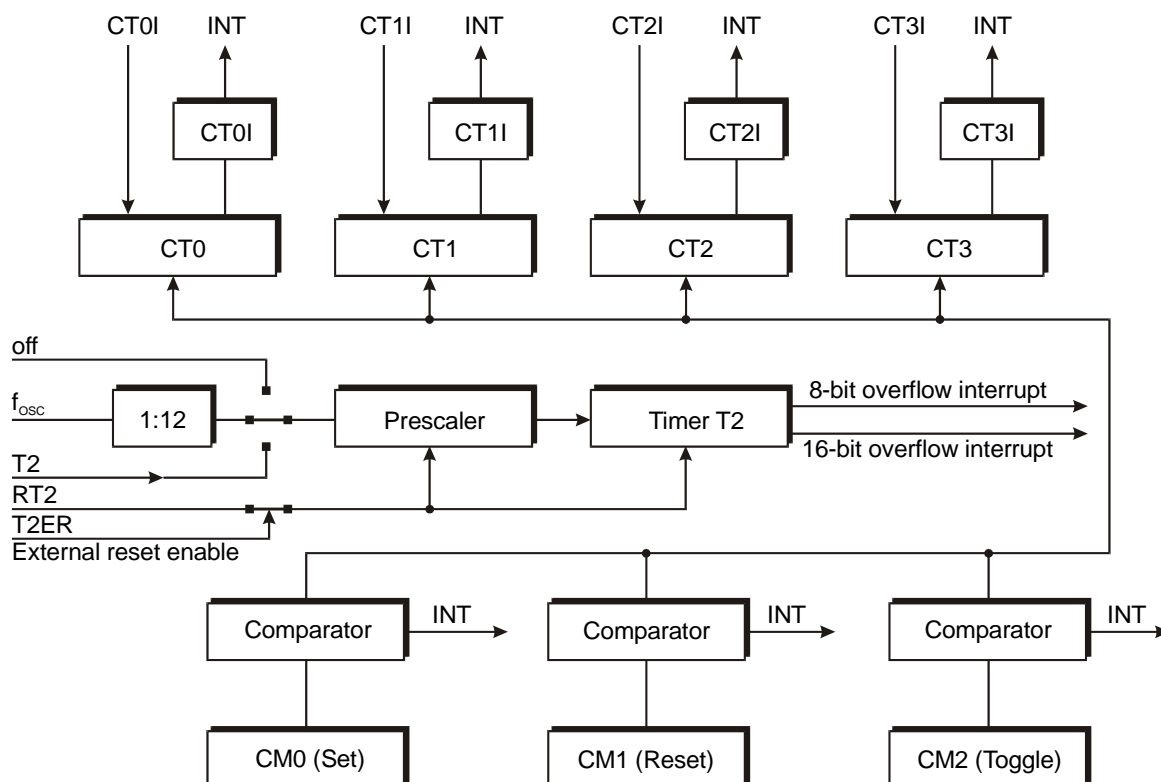


Fig. 5.7 Secțiunea de captare a evenimentelor și comparare a microcontrollerului 80C552.

Această rutină de întrerupere poate fi folosită pentru calcularea intervalului de timp corespunzător, dacă se cunoaște valoarea anterioară a timer-ului T_2 . Pentru o frecvență a ceasului de 12 MHz programarea timer-ului poate fi făcută astfel încât acesta să furnizeze indicații de depășire la fiecare 524 ms.

Dacă intervalul de timp dintre două evenimente succesive este mai scurt de 524 ms, atunci calculul intervalului de timp este foarte simplu, rutina de întrerupere fiind mult mai scurtă. Pentru intervale de timp mai mari de 524ms, trebuie utilizată o extensie a timer-ului T_2 .

Timer-ul T_2 este conectat la patru registre de 16 biți, menționate anterior și, de asemenea la trei registre de comparare, cu lungimea de 16 biți (fig. 2.7). Registrele de comparare pot fi folosite pentru a seta, reseta sau comuta pinii de ieșire ai portului P_4 la anumite intervale preprogramate de timp.

5.2 PREZENTAREA SETULUI DE INSTRUCȚIUNI AL MICROCONTROLLER-ULUI 80C51

1	ACALL	addr11				
Funcția:	Absolute Call					
Descriere:	Apelează necondiționat o subrutină localizată la adresa indicată. Instrucțiunea incrementează contorul programului de două ori, pentru a obține adresa următoarei instrucțiuni, apoi încarcă în stivă rezultatul pe 16 biți (primul octet este cel mai puțin semnificativ) și incrementează pointerul stivei de două ori. Adresa destinație e obținută prin concatenarea succesivă a celor 5 biți mai semnificativi ai contorului programului, după incrementare, biții 7÷5 ai codului operației și octetul al doilea al instrucțiunii. Subrutina apelată trebuie să înceapă în același bloc (de dimensiune 2 kocteți) al memoriei de program ca și primul octet al instrucțiunii care urmează instrucțiunii ACALL. Nu se afectează indicatorii de condiție.					
Exemplu:	Inițial registrul pointer de stivă, SP, conținea valoarea 07H. Eticheta SUBRTN e localizată în memorie la adresa 0345H. După execuția instrucțiunii: ACALL SUBRTN la locația 0123H, SP va conține valoarea 09H, locațiile interne RAM 08H și 09H vor conține 25H și, respectiv 01H, iar PC va conține 0345H.					
Octeți:	2					
Ciclii mașină:	2					
Codare:	<table><tr><td>a₁₀ a₉ a₈ 1</td><td>0 0 0 1</td></tr></table>	a ₁₀ a ₉ a ₈ 1	0 0 0 1	<table><tr><td>a₇ a₆ a₅ a₄</td><td>a₃ a₂ a₁ a₀</td></tr></table>	a ₇ a ₆ a ₅ a ₄	a ₃ a ₂ a ₁ a ₀
a ₁₀ a ₉ a ₈ 1	0 0 0 1					
a ₇ a ₆ a ₅ a ₄	a ₃ a ₂ a ₁ a ₀					
Operații:	ACALL (PC) ← (PC)+2 (SP) ← (SP)+1 (SP) ← (PC _{7÷0})					

$(SP) \leftarrow (SP)+1$
 $(SP) \leftarrow (PC_{15:8})$
 $(PC_{10:0}) \leftarrow \text{adresa blocului}$

2	ADD	A, <scr-byte>
Funcția:	Adunare	
Descriere:	<p>Instrucțiunea ADD adună acumulatorul cu octetul indicat, lăsând rezultatul în acumulator. Indicatorii de transport și de transport auxiliar sunt setați, respectiv dacă există un transport de la bitul 7 sau de la bitul 3, și sunt resetați dacă nu există transport. Când se adună întregi fără semn, indicatorul de transport indică depășire. Indicatorul de depășire, OV, este setat dacă este un transport la bitul 6, dar nu mai departe de bitul 7, sau un transport de la bitul 7, dar nu provenit de la bitul 6. În celelalte situații, OV este resetat. Când se adună întregi cu semn, indicatorul OV semnalizează un număr negativ ca sumă a doi operanzi pozitivi, sau o sumă pozitivă ca rezultat a adunării a doi operanzi negativi. Sunt permise 4 moduri de adresare a operandului sursă: adresare prin registru, adresare directă, adresare indirectă prin registru sau adresare imediată.</p>	
Exemplu:	<p>Acumulatorul conține valoarea 0C3H (11000011B) iar registrul 0 conține 0AAH (10101010B). Instrucțiunea:</p> <p style="text-align: center;">ADD A, R₀</p> <p>va determina ca valoarea conținută de acumulator să fie 6DH (01101101B), cu indicatorul AC resetat și atât indicatorul de transport, cât și OV, setate la 1.</p>	
3	ADDC	A, <scr-byte>
Funcția:	Adunare cu indicatorul de transport	
Descriere:	<p>Instrucțiunea ADDC adună acumulatorul cu octetul indicat și cu indicatorul de transport, lăsând rezultatul în acumulator. Indicatorii de transport și de transport auxiliar sunt setați, respectiv dacă există un transport de la bitul 7 sau de la bitul 3, și sunt resetați dacă nu există transport. Când se adună întregi fără semn, indicatorul de transport indică depășire. indicatorul de depășire, OV, este setat dacă este un transport la bitul 6, dar nu mai departe de bitul 7, sau un transport de la bitul 7, dar nu provenit de la bitul 6. În celelalte situații, OV este resetat. Când se adună întregi cu semn, indicatorul OV semnalizează un număr negativ ca sumă a doi operanzi pozitivi, sau o sumă pozitivă ca rezultat a adunării a doi</p>	

Exemplu: operanzi negativi. Sunt permise 4 moduri de adresare a operandului sursă: adresare prin registru, adresare directă, adresare indirectă prin registru sau adresare imediată. Acumulatorul conține 0C3H (11000011B) și registrul 0 conține 0AAH (10101010B) cu setarea indicatorului de transport. Instrucțiunea:
ADDC A, R0
 va determina ca acumulatorul să conțină valoarea 6EH (01101110B), cu indicatorul AC resetat, iar indicatorul de transport și OV setate la 1.

4 AJMP addr11
 Funcția: Absolute Jump
 Descrierea: AJMP transferă execuția programului la adresa indicată, care este formată la momentul rulării prin concatenarea celor 5 biți mai semnificativi ai contorului programului, PC, (după incrementarea de două ori a acestuia), biții 7÷5 de cod ai operației și al doilea octet al instrucțiunii. Destinația trebuie să fie în același bloc de 2 kocteți din memoria de program ca primul octet al instrucțiunii următoare instrucțiunii AJMP.
 Exemplu: Eticheta JMPADR e situată la locația de memorie 0123H. Instrucțiunea
AJMP JMPADR
 este situată la locația 0345H și va încarca contorul programului, PC, cu 0123H.

5 ANL <dest-byte>, <scr-byte>
 Funcția: ȘI LOGIC între variabile de tip octet
 Descriere: ANL efectuează operația de ȘI LOGIC între variabilele de tip octet indicate și încarcă rezultatul în variabila destinație. Nu afectează nici un indicator de condiție. Cei doi operanzi permit 6 combinații de moduri de adresare. Când destinația este acumulatorul, sursa poate fi adresată prin registru, direct, indirect prin registru sau imediat; când destinația este o adresă directă, sursa poate fi acumulatorul sau o dată imediată.
Notă: Când această instrucțiune este utilizată pentru a modifica un port de ieșire, valoarea utilizată ca dată inițială va fi citită de la ieșire, nu de la pinii de intrare.
 Exemplu: Dacă acumulatorul conține valoarea 0C3H (11000011B) și registrul 0 conține valoarea 55H (01010101B) atunci instrucțiunea

ANL A, R0

va determina ca acumulatorul să conțină valoarea 41H. Când destinația este un octet direct adresabil, această instrucțiune va reseta combinațiile de biți din orice locație RAM sau registru hardware. Octetul mascat care determină resetarea biților va fi o constantă conținută în instrucțiune sau o valoare creată în acumulator la rulare. Instrucțiunea

ANL P1, #0111001B

va reseta biții 7, 3 și 2 ai portului de ieșire 1.

6	ANL	C, <scr-bit>
Funcția:	ȘI LOGIC între variabile de tip bit	
Descriere:	Dacă valoarea booleană a bitului sursă este un 0 logic, atunci re setează indicatorul de transport; altfel acest indicator rămâne în starea curentă. Simbolul ('/') precedind operandul, în limbaj de asamblare, indică faptul că complementul logic al bitului adresat e utilizat ca valoare sursă, fără însă ca bitul sursă să fie afectat. Pentru operandul sursă este permisă numai adresarea directă.	
Exemplu:	Setează indicatorul de transport dacă și numai dacă P1.0=1, ACC.7=1 și OV=0: <div style="margin-left: 40px;"> MOV C, P1.0 ; incarcă carry cu bitul 0 al portului 1 ANL C, ACC.7 ; SI cu bitul 7 al acumulatorului ANL C, /OV ; SI cu negatul indicatorului depășire. </div>	
7	CJNE	<dest-byte>, <scr-byte>,rel
Funcția:	Comparație și salt dacă nu este egalitate	
Descriere:	Instrucțiunea CJNE compară cei doi operanzi și determină un salt în execuția programului dacă valorile lor nu sunt egale. Destinația saltului este compusă prin adunarea deplasării relative cu semn, în ultimul octet al instrucțiunii, cu conținutul contorului programului, PC, după incrementarea acestuia la adresa de start a următoarei instrucțiuni. Indicatorul de transport e setat dacă valoarea întregului fără semn a octetului destinație e mai mică decât valoarea întregului fără semn a <scr-byte>; altfel, indicatorul de transport e resetat. instrucțiunea nu afectează nici un operand. Cei doi operanzi permit 4 combinații de moduri de adresare: acumulatorul poate fi comparat cu orice octet direct sau imediat adresabil, cu orice locație indirectă de RAM sau	

registrii de lucru pot fi comparați cu o constantă imediată.

Exemplu: Acumulatorul conține valoarea 34H. Registrul 7 conține valoarea 56H. Prima instrucțiune din secvența:

```
CJNE R7, #60H, NOT_EQ
R7=60H
NOT_EQ: JC REQ_LOW
```

setează indicatorul de transport și sare la instrucțiunea cu eticheta NOT_EQ. Prin testarea indicatorului de transport, această instrucțiune determină dacă conținutul registrului R7 e mai mare sau mai mic decât 60H.

Dacă data care a fost prezentă la portul 1 este 34 H, atunci instrucțiunea

```
WAIT: CJNE A, P1, WAIT
```

resetează indicatorul de transport și continuă cu următoarea instrucțiune din secvență, până când acumulatorul se egalează cu data citită din portul P1. Dacă în portul P1 au fost introduse alte valori, programul va bucla în acest punct până când data din portul P1 va deveni 34H.

8	CLR	A
Funcția:	Resetează (inițializează) acumulatorul	
Descriere:	Acumulatorul este resetat (toti biții sunt 0). Nu este afectat nici un indicator de condiție.	
Exemplu:	Acumulatorul conține valoarea 5CH (01011100B). Instrucțiunea: CLR A va determina conținutul acumulatorului să fie 00H (00000000B).	
9	CLR	bit
Funcția:	Resetează bitul	
Descriere:	Bitul specificat este resetat (poziționat la 0). Nici un alt indicator de condiție nu este afectat. Instrucțiunea CLR bit poate opera asupra indicatorului de transport sau asupra oricărui bit direct adresabil.	
Exemplu:	Portul 1 a fost înscris cu valoarea 5DH (01011101B). Instrucțiunea CLR P1,2 va lăsa portul P1 setat la valoarea 59H (01011001B).	
10	CPL	A
Funcția:	Complementarea acumulatorului	

Descriere: Fiecare bit al acumulatorului este complementat logic (complement față de 1). Biții care anterior erau 1 devin 0, și invers. Nici un indicator de condiții nu este afectat.

Exemplu: Acumulatorul conține valoarea 5CH (01011100B).

Instrucțiunea:

CPL A

va determina ca acumulatorul să conțină valoarea 0A3H (10100011B).

11 CPL bit

Funcția: Complementează o variabilă de tip bit

Descriere: Instrucțiunea efectuează complementul variabilei de tip bit specificate. Un bit care era 1 devine 0, și invers. Nu sunt afectați indicatorii de condiții. CLR poate opera asupra bitului de transport sau asupra oricărui alt bit direct adresabil.
Notă: Când instrucțiunea este utilizată pentru a modifica un bit de ieșire, valoarea utilizată ca dată originală va fi citită ca dată de ieșire și nu de la intrare.

Exemplu: Portul 1 a fost înscris anterior cu valoarea 5DH (01011101B). Secvența de instrucțiuni:

CPL P1,1

CPL P1,2

va determina ca portul să fie setat la valoarea 5BH (01011011B).

12 DA A

Funcția: Ajustarea zecimală a acumulatorului

Descriere: Instrucțiunea DA A ajustează valoarea pe 8 biți din acumulator, ca rezultat al ultimei instrucțiuni de adunare a două variabile (reprezentate în format BCD-împachetat), producând două cifre de câte 4 biți. Pentru a se efectua instrucțiunea de adunare au fost folosite instrucțiunile ADD sau ADDC. Dacă biții 3÷0 ai acumulatorului reprezintă un număr mai mare decât 9 (xxx1010÷xxx1111), sau dacă indicatorul AC este setat, se adună valoarea 6 la acumulator, producând astfel un digit optimal în format BCD. Această adunare internă va seta indicatorul de transport dacă s-a propagat o depășire de la cei mai puțin semnificativi 4 biți spre biții mai semnificativi, dar altfel nu va reseta indicatorul de transport. Dacă indicatorul de transport e setat în urma acestei operații, sau dacă cei 4 biți mai semnificativi reprezintă un număr mai mare decât 9 (1010xxx÷1111xxxx),

acești biți sunt incrementați cu 6, producând digitul optimal în format BCD. Indicatorul de transport va indica dacă suma celor două variabile BCD originale este mai mare decât 100, permițând sumarea zecimală în precizie multiplă. Indicatorul OV este afectat. Execuția acestei instrucțiuni durează un singur ciclu. Elementul de noutate este acela că această instrucțiune permite conversia zecimală adăugând 00H, 06H, 60H sau 66H la conținutul acumulatorului.

Notă: Instrucțiunea DA A nu poate converti, pur și simplu, un număr hexazecimal, conținut în acumulator, în format BCD.

Exemplu:

Acumulatorul are valoarea 56H (01010110B) reprezentând digiții în format BCD împachetat ai numărului zecimal 56. Registrul R3 conține valoarea 67H (01100111B), reprezentând digiții în format BCD împachetat ai numărului 67. Indicatorul de transport este setat. Secvența de instrucțiuni:

```
ADDC    A,R3
```

```
DA      A
```

va face o adunare în complement binar față de doi, având ca rezultat valoarea 24H (00100100B) indicând în format BCD împachetat numărul zecimal 24, cele mai mici două cifre ale sumei zecimale între 56, 67 și transportul intern. Indicatorul de transport va fi setat, indicând că a apărut o depășire zecimală. Suma reală între 56, 67 și 1 este 124. Variabilele BCD pot fi incrementate sau decrementate prin adăugarea lui 01H sau 99H. Dacă inițial acumulatorul conține valoarea 30H, secvența:

```
ADD A, #99H
```

```
DA      A
```

va determina ca acumulatorul să conțină valoarea 99H (30+99=129). Octetul mai puțin semnificativ al sumei poate fi interpretat ca fiind: 30-1=29.

13 DEC	byte
Funcția:	Decrementare variabilă de tip octet
Descriere:	Variabila de tip octet indicată este decrementată cu 1. O valoare inițială de 00H va determina ca rezultatul să fie 0FFH. Nu afectează nici un indicator. Sunt permise 4 moduri de adresare: acumulator, adresare prin registru, adresare directă, adresare indirectă prin registru.
	Notă: Când instrucțiunea este utilizată ca să modifice un port de ieșire, valoarea utilizată ca dată originală va fi citită de la

ieșire, și nu de la intrare.

Exemplu: Registrul R0 conține valoarea 7FH (01111111B). Locațiile interne RAM 7EH și 7FH conțin valorile 00H și 40H. Instrucțiunile:

```
DEC @R0
DEC R0
DEC @R0
```

vor lăsa registrul R0 setat la valoarea 7EH și locațiile interne RAM 7EH și 7FH conținând 0FFH și 3FH.

14	DIV	AB
----	------------	-----------

Funcția: Împărțire

Descrierea: Instrucțiunea DIV AB împarte întregul fără semn, pe 8 biți, din acumulator la întregul, pe 8 biți, fără semn din registrul B. După execuția instrucțiunii, acumulatorul va conține câtul împărțirii, iar registrul B restul. Indicatorul de transport și OV vor fi resetate.

Excepție: Dacă registrul B conținea inițial valoarea 00H, valoarea returnată în acumulator și registrul B vor fi subdefinite și va fi setat indicatorul de depășire. Indicatorul de transport va fi resetat în orice caz.

Exemplu: Acumulatorul conține valoarea zecimală 251 (0FBH sau 11111011B) și registrul B conține valoarea 18 (12H sau 00010010B). Instrucțiunea:

```
DIV AB
```

va determina ca acumulatorul să conțină valoarea 13 (0DH sau 00001101B) și valoarea 17 (11H sau 00010001B) în B. Atât indicatorul de transport, cât și indicatorul OV vor fi ambii resetati.

15	DJNZ	<byte>, <rel-addr>
----	-------------	---------------------------------------

Funcția: Decrementare și salt dacă rezultatul nu este zero

Descriere: Instrucțiunea DJNZ decrementează octetul indicat și determină un salt la adresa indicată de al doilea operand dacă valoarea rezultată nu e zero. O valoare inițială 00H va deveni 0FFH. Nu afectează indicatorii de condiții. Destinația saltului va fi obținută prin adăugarea valorii deplasării relative cu semn, în ultimul octet al instrucțiunii, la conținutul PC, după incrementarea acestuia la adresa primul octet al instrucțiunii următoare. Operandul decrementat poate fi un registru sau direct octetul adresat.

Notă: Atunci când această instrucțiune este utilizată ca să

Exemplu: modifice un port de ieșire, valoarea utilizată ca dată inițială va fi citită de la ieșire, nu de la pinii de intrare.

Locațiile interne RAM 40H, 50H, 60H conțin respectiv valorile 01H, 70H, 15H. Secvența de instrucțiuni:

```
DJNZ40H, LABEL_1
DJNZ50H, LABEL_2
DJNZ60H, LABEL_3
```

va cauza un salt la instrucțiunea cu eticheta LABEL_2 cu valorile 00H, 6FH și 15H în cele 3 locații RAM. Primul salt n-a fost făcut, pentru că rezultatul era 0.

Această instrucțiune asigură o metodă simplă de a executa o buclă de program de un număr de ori, sau adăugarea printr-o singură instrucțiune a unei întârzieri (între 2 și 512 cicli mașină). Instrucțiunea:

```
MOV R2, #8
TOGGLE: CPL P1.7
DJNZR2, TOGGLE
```

va schimba bitul P1.7 de 8 ori, determinând apariția a 4 impulsuri de ieșire la bitul 7 al portului de ieșire P1. Fiecare puls înseamnă trei cicluri mașină, doi pentru execuția instrucțiunii DJNZ și unul pentru modificarea bitului.

16 INC	<byte>
Funcția:	Incrementare octet
Descriere:	Instrucțiunea INC incrementează cu 1 variabila octet indicată. O valoare inițială de 0FFH va deveni în urma incrementării 00H. Nu se afectează nici un indicator de condiții. Sunt permise trei moduri de adresare: adresare prin registru, adresare directă, adresare indirectă prin registru. Notă: Atunci când această instrucțiune este utilizată ca să modifice un port de ieșire, valoarea utilizată ca dată inițială va fi citită de la ieșire și nu de la intrare.
Exemplu:	Registru R0 conține valoarea 7EH (01111110B). Locațiile interne RAM cu adresele 7EH și 7FH conțin valorile 0FFH și respectiv 40H. Secvența de instrucțiuni:
	INC @R0
	INC R0
	INC @R0
	va lăsa registrul R0 setat la 7FH și locațiile interne RAM cu adresele 7EH și 7FH conținând valorile 00H și 41H.

17	INC	DPTR
Funcția:	Incrementarea pointer-ului de date	
Descriere:	<p>Instrucțiunea incrementează cu 1 pointerul de date (16 biți). Este utilizată o incrementare pe 16 biți. O depășire la octetul mai puțin semnificativ al pointerului (DPL), de la valoarea 0FFH la 00H, va incrementa octetul mai semnificativ (DPH). Nu sunt afectați indicatorii de condiții. Singurul registru care poate fi manipulat de această instrucțiune este pointerul de date, DPTR.</p>	
Exemplu:	<p>Registrele DPH și DPL conțin valorile 12H și respectiv 0FEH. Setul de instrucțiuni:</p> <pre> INC DPTR INC DPTR INC DPTR </pre> <p>va schimba conținutul registrelor DPH și DPL la valorile 13H și respectiv 01H.</p>	
18	JB	bit, rel
Funcția:	Salt dacă bitul e setat la 1	
Descriere:	<p>Dacă bitul indicat e un 1 se efectuează un salt în program la adresa indicată, altfel se trece la executarea instrucțiunii următoare. Destinația saltului este obținută prin adunarea deplasării relative cu semn, în al treilea octet al instrucțiunii, la conținutul contorului de program, PC, după incrementarea PC la valoarea primului octet al instrucțiunii următoare. Bitul de test nu este modificat și nu se afectează nici un indicator.</p>	
Exemplu:	<p>Data de la portul de intrare P1 e 11001010B. Acumulatorul conține valoarea 56. Secvența de instrucțiuni:</p> <pre> JB P1.2, LABEL1 JB ACC.2, LABEL2 </pre> <p>va cauza un salt al execuției programului la eticheta LABEL2.</p>	
19	JBC	bit, rel
Funcția:	Salt dacă bitul e setat la 1 și șterge bitul	
Descriere:	<p>Dacă bitul indicat este 1, se efectuează un salt la adresa indicată. Altfel, se trece la executarea instrucțiunii următoare. Bitul nu va fi resetat dacă este deja 0. Destinația saltului este obținută prin adunarea deplasării relative cu semn, în al treilea octet al instrucțiunii, la conținutul contorului de program, PC, după incrementarea PC la valoarea primului octet al instrucțiunii următoare. Nici un indicator de condiții</p>	

nu este afectat.

Notă: Când instrucțiunea este utilizată ca sa testeze un pin de ieșire, valoarea utilizată ca dată inițială va fi citită de la ieșire, și nu de la intrare.

Exemplu: Acumulatorul conține valoarea 56H (01010110B). Setul de instrucțiuni:

JBC ACC.3, LABEL 1

JBC ACC.2, LABEL 2

va face ca execuția programului să continue de la instrucțiunea identificată prin eticheta LABEL2, cu acumulatorul modificat la valoarea 52H (01010010B).

20 JC rel

Funcția: Salt dacă este setat indicatorul de transport

Descriere: Dacă indicatorul de transport e setat, se efectuează un salt în program, la adresa indicată. Altfel, se trece la executarea instrucțiunii următoare. Destinația saltului este obținută prin adunarea deplasării relative cu semn, în al doilea octet al instrucțiunii, la conținutul contorului de program, PC, după incrementarea PC de două ori. Nu sunt afectați indicatorii de condiție.

Exemplu: Indicatorul de transport este resetat .Secvența de instrucțiuni:

JC LABEL1

CPL C

JC LABEL2

va seta indicatorul de transport și va face ca execuția programului să continue cu instrucțiunea de la eticheta LABEL2.

21 JMP @A+DPTR

Funcția: Salt indirect

Descriere: Instrucțiunea adună conținutul pe 8 biți, fără semn, al acumulatorului cu pointerul de date pe 16 biți și încarcă suma rezultată în contorul programului, PC. Conținutul acestuia va reprezenta adresa instrucțiunii următoare. Adunarea pe 16 biți se face astfel: transportul de la cei 8 biți mai puțin semnificativi se propagă spre biții mai semnificativi. Nu se afectează indicatorii și nu se schimbă conținutul acumulatorului și nici al pointerului de date.

Exemplu: În acumulator se află un număr oarecare, de la 0 la 6. Următoarea secvență de instrucțiuni va efectua un salt la una din cele 4 instrucțiuni AJMP începând de la JMP_TBL.

```

MOV DPTR, #JMP_TBL
JMP @A+DPTR
AJMP LABEL0
JMP_TBL: AJMP LABEL1
AJMP LABEL2
AJMP LABEL3

```

Dacă acumulatorul conține 04H când începe această secvență, execuția va sări la eticheta LABEL2. Amintim că AJMP e o instrucțiune pe doi octeți, deci instrucțiunea de salt va începe la orice altă adresă.

22 JNB bit, rel

Funcția: Salt dacă bitul nu e setat la 1

Descriere: Dacă bitul indicat este 0, se efectuează un salt la adresa indicată. Altfel, se trece la executarea instrucțiunii următoare. Destinația saltului este obținută prin adunarea deplasării relative cu semn, în al treilea octet al instrucțiunii, la conținutul contorului de program, PC, după incrementarea PC la valoarea primului octet al instrucțiunii următoare. Bitul testat nu este modificat. Nu este afectat nici un indicator de condiții.

Exemplu: Data prezentă la portul P1 de intrare este 11001010B. Acumulatorul conține valoarea 56H (01010110B). Secvența de instrucțiuni:

```

JNB P1.3, LABEL1
JNB ACC.3, LABEL2

```

va face ca execuția programului să continue de la instrucțiunea cu eticheta LABEL2.

23 JNC rel

Funcția: Salt dacă indicatorul de transport nu este setat

Descriere: Dacă indicatorul de transport este 0, se sare instrucțiunea de la adresa indicată. Altfel, se trece la executarea instrucțiunii următoare. Destinația saltului este obținută prin adunarea deplasării relative cu semn, în al doilea octet al instrucțiunii, la conținutul contorului de program, PC, după incrementarea PC de două ori, pentru a se ajunge la adresa primului octet al instrucțiunii următoare. Indicatorul de transport nu e modificat.

Exemplu: Indicatorul de transport este setat. Secvența de instrucțiuni:

```

JNC LABEL1
CPL C

```

JNC LABEL2

va reseta indicatorul de transport și va face ca execuția programului să continue de la instrucțiunea specificată de eticheta LABEL2.

24	JNZ	rel
Funcția:	Salt dacă conținutul acumulatorului nu este 0	
Descriere:	Dacă unul dintre biții acumulatorului este 1, se efectuează un salt la adresa indicată. Altfel, se continuă cu executarea instrucțiunii următoare. Destinația saltului este obținută prin adunarea deplasării relative cu semn, în al doilea octet al instrucțiunii, la conținutul contorului de program, PC, după incrementarea PC de două ori. Acumulatorul nu se modifică și nici un indicator de condiții nu este afectat.	
Exemplu:	<p>Acumulatorul are valoarea 00H. Instrucțiunile:</p> <pre> JNZ LABEL1 INC A JNZ LABEL2 </pre> <p>vor seta acumulatorul la valoarea 01H și vor determina continuarea programului de la instrucțiunea cu eticheta LABEL2.</p>	
25	JZ	rel
Funcția:	Salt dacă conținutul acumulatorului este zero	
Descriere:	Dacă toți biții din acumulator sunt zero, se execută un salt la adresa indicată. Altfel, se continuă cu execuția instrucțiunii următoare. Destinația saltului este obținută prin adunarea deplasării relative cu semn, în al doilea octet al instrucțiunii, la conținutul contorului de program, PC, după incrementarea PC de două ori. Acumulatorul nu se modifică și nici un indicator de condiții nu este afectat.	
Exemplu:	<p>Acumulatorul conține inițial valoarea 01H. Secvența de instrucțiuni:</p> <pre> JZ LABEL1 DEC A JZ LABEL2 </pre> <p>va determina ca acumulatorul să conțină valoarea 00H și va face ca execuția programului să continue cu instrucțiunea de la LABEL2.</p>	
26	LCALL	addr16
Funcția:	Long Call	

Descriere:	Instrucțiunea LCALL va apela o subrutină aflată la adresa indicată. Instrucțiunea incrementează cu trei contorul programului pentru a genera adresa următoarei instrucțiuni, și salvează rezultatul (pe 16 biti) în stivă (mai întâi octetul mai puțin semnificativ). Se incrementează pointerul stivei cu 2. Octeții, mai puțin semnificativ și mai semnificativ, ai contorului programului, PC, sunt încărcăți cu al doilea și al treilea octet al instrucțiunii LCALL. Execuția programului va continua cu instrucțiunea de la această adresă. Subrutina poate începe oriunde în cei 64 kocteți ai spațiului de memorie de program. Nu sunt afectați indicatorii de condiție.
Exemplu:	Inițial, pointerul stivei are valoarea 07H. Eticheta SUBRTN este asociată locației de memorie de program cu adresa 1234H. După execuția instrucțiunii: <div style="text-align: center;">LCALL SUBRTN</div> la locația de memorie cu adresa 0123H, pointerul stivei va conține 09H, locațiile interne RAM cu adresele 08H și 09H vor conține 26H și 01H, iar PC va conține 1235H.

27 LJMP	addr16
Funcția:	Long Jump
Descriere:	Instrucțiunea LJMP produce un salt necondiționat la adresa indicată, prin încărcarea octeților mai puțin semnificativ și mai semnificativ ai contorului programului, PC, cu al doilea și al treilea octet al instrucțiunii. Destinația poate fi oriunde în spațiul de adresare al memoriei program de 64 kocteți. Nu se afectează indicatorii de condiții.
Exemplu:	Eticheta JMPADR e asociată instrucțiunii localizate la adresa 1234H în memoria de program. Instrucțiunea: <div style="text-align: center;">LJMP JMPADR</div> va încărca contorul programului cu valoarea 1234H.

28 MOV	<dest-byte>, <scr-byte>
Funcția:	Mută variabila sursă, de tip octet, în variabila destinație, de tip octet
Descriere:	Variabila octet indicată prin al doilea operand este copiată în locația specificată de primul operand. Octetul sursă nu este afectat. Nu se afectează nici un registru sau indicator de condiții. Este de departe cea mai flexibilă operațiune. Permite 14 combinații de moduri de adresare ale sursei și destinației.
Exemplu:	Locația internă RAM cu adresa 30H conține valoarea 40H. Locația internă RAM cu adresa 40H conține valoarea 10H.

Data de la portul de intrare P1 este 11001010B (0CAH).
Instrucțiunile:

```
MOV R0, #30H
MOV A, @R0
MOV R1, A
MOV B, @R1
MOV @R1, P1
MOV P2, P1
```

lasă valoarea 30H în registrul R0, 40H în acumulator și registrul P1, 10H în registrul B și 0CAH (11001010B) în locația RAM cu adresa 40H și în portul de ieșire P2.

29	MOV	<dest-bit>, <scr-bit>
Funcția:	Mută data de tip bit de la sursă la destinație	
Descriere:	Variabila booleană indicată prin al doilea operand este copiată la locația specificată de primul operand. Unul dintre operanzi trebuie să fie indicatorul de transport, celălalt poate fi orice bit adresabil direct. Nici un alt registru sau indicator nu este afectat.	
Exemplu:	Indicatorul de transport este inițial setat. Data prezentă la portul de intrare P3 este 11000101B. Data înscrisă anterior în portul de ieșire P1 este 35H (00110101B). Instrucțiunile: <pre>MOV P1.3, C MOV C, P3.3 MOV P1.2, C</pre> vor lăsa indicatorul de transport resetat și portul P1 la valoarea 39H (00111001B).	
30	MOV	DPTR, #data16
Funcția:	Încarcă pointerul de date cu o constantă pe 16 biți	
Descriere:	Pointerul de date este încărcat cu constanta pe 16 biți indicată. Aceasta se încarcă în al doilea și al treilea octet al instrucțiunii. Al doilea octet (DPH) este octetul mai semnificativ, iar al treilea octet (DPL) conține octetul mai puțin semnificativ al constantei specificate. Nu se afectează indicatorii. Este singura instrucțiune de tranfer pe 16 biți.	
Exemplu:	Instrucțiunea: <pre>MOV DPTR, #1234H</pre> va încărca valoarea 1234H în pointerul de date. DPH va conține 12H și DPL va conține 34H.	

31	MOVC	A,@A+<base-reg>
Funcția:	Mută octetul de cod	
Descriere:	<p>Instrucțiunea MOVC încarcă acumulatorul cu un octet de cod sau o constantă din memoria-program. Adresa octetului reprezintă suma conținutului, pe 8 biți, fără semn, al acumulatorului și conținutul, pe 16 biți al registrului de bază, care poate fi pointerul de date sau contorul programului. În ultimul caz, PC este incrementat la adresa următoarei instrucțiuni dinaintea sumării cu acumulatorul. Altfel, registrul bază nu e modificat. Adunarea pe 16 biți se face astfel încât un transport de la cei 8 biți mai puțin semnificativi să poată fi propagat la ceilalți. Nu sunt afectați indicatorii de condiții.</p>	
Exemplu:	<p>În acumulator se găsește o valoare cuprinsă între 0 și 3. Următoarele instrucțiuni vor transla valoarea din acumulator înspre una din cele 4 valori definite la directiva DB (define byte):</p> <pre> REL_PC: INC A MOVC A, @A+PC RET DB 66H DB 77H DB 88H DB 99H </pre> <p>Dacă subrutina este apelată cu acumulatorul având valoarea 01H, va returna 77 în acumulator. Instrucțiunea INC A plasată înaintea instrucțiunii MOVC a permis “ocolirea” instrucțiunii RET din secvență. Dacă câțiva octeți de cod separă începutul de instrucțiunea MOVC din secvență, numărul corespunzător va fi adăugat la acumulator.</p>	
32	MOVX	<dest-byte>,<scr-byte>
Funcția:	Mutare externă	
Descriere:	<p>Instrucțiunea MOVX transferă date între acumulator și un octet al memoriei externe. Sunt două tipuri de instrucțiuni diferite, după cum se furnizează o adresare indirectă la RAM-ul extern, pe 8 sau pe 16 biți. În primul caz, conținutul registrelor R0 și R1 furnizează o adresă pe 8 biți multiplexată cu data din portul P0. 8 biți sunt suficienți pentru decodarea extensiei I/O externe pe o arie de RAM mică. Pentru o arie mai mare, pinii porturilor pot fi utilizați pentru ieșirea biților mai semnificativi de adresă. Acești biți vor fi controlați de o</p>	

instrucțiune de ieșire care va urma instrucțiunii MOVX. În al doilea caz, pointerul de date generează o adresă pe 16 biți. Portul P2 va marca ieșirea celor 8 biți superiori de adresă (conținutul DPH) iar portul P0 va multiplexa cei 8 biți inferiori (DPL) cu cei de date. Registrul funcțiilor speciale P2 va reține conținutul anterior, iar bufferul de ieșire P2 va emite conținutul lui DPH. Această formulă e mai rapidă și mai eficientă când se accesează zone de date foarte mari (mai mari de 64 octeți), deoarece nu este nevoie de instrucțiuni suplimentare pentru a seta porturile de ieșire. Este posibilă și combinația celor două tipuri de instrucțiuni MOVX. O zonă extinsă de memorie RAM și liniile sale de adresă de ordin superior, administrate de portul P2, pot fi adresate prin pointerul de date, iar codul de ieșire al biților superiori de adresă al portului P2 va fi urmat de o instrucțiune MOVX utilizând registrele R0 sau R1.

Exemplu:

O zonă de memorie RAM externă, cu lungimea de 256 octeți, utilizând multiplexarea liniilor de adrese și de date este conectată la portul P0 al microcontrollerului 8051. Portul P3 asigură liniile de control pentru memoria RAM externă. Porturile P1 și P2 sunt utilizate pentru intrările și ieșirile normale. Registrele R0 și R1 conțin valorile 12H și respectiv 34H. Locația cu adresa 34H a RAM extern conține valoarea 56H. Instrucțiunile:

```
MOVX    A,@R1
MOVX    @R0,A
```

copiază valoarea 56H atât în acumulator, cât și în locația RAM externă cu adresa 12H.

33 MUL

AB

Funcția: Înmulțire

Descriere: Instrucțiunea MUL AB înmulțește întregii pe 8 biți din acumulator și din registrul B. Octetul mai puțin semnificativ al produsului pe 16 biți este lăsat în acumulator, iar octetul mai semnificativ în registrul B. Dacă produsul e mai mare decât 255 (0FFH) indicatorul de depășire este setat; altfel, este zero. Indicatorul de transport este întotdeauna resetat.

Exemplu:

Inițial, acumulatorul conține valoarea 80 (50H), iar registrul B, valoarea 160 (0A0H). Instrucțiunea

```
MUL AB
```

va furniza produsul, 12.800 (3200H), registrul B devine 32H (00110010B) iar acumulatorul 00H. Indicatorul de depășire

este setat, iar cel de transport este zero.

34 NOP

Funcția: Nici o operație

Descriere: Execuția continuă cu instrucțiunea următoare. În afara contorului de program, PC, care este incrementat, nici un registru sau indicator nu este afectat.

Exemplu: Este nevoie de producerea unui puls de depășire scurt pe bitul 7 al portului P2, durând exact 5 cicluri. O secvență SETB/CLR va genera un puls cu durata de un ciclu, la care trebuie adunați alți patru, ca în secvența:

```
CLR P2.7
NOP
NOP
NOP
NOP
SETBP2.7
```

35 ORL <dest-byte>, <scr-byte>

Funcția: SAU LOGIC pentru variabile de tip octet

Descriere: Instrucțiunea ORL realizează funcția SAU LOGIC între variabilele de tip octet indicate, încărcând rezultatul în octetul destinație. Nu se afectează nici un indicator de condiții. Cei doi operanzi permit 6 combinații ale modurilor de adresare. Când destinația este acumulatorul, sursa poate fi adresată ca registru, direct, registru-indirect sau imediat; când destinația este o adresă directă, sursa poate fi acumulatorul sau o data imediată.

Notă: Atunci când această instrucțiune este utilizată ca să modifice un port de ieșire, valoarea utilizată ca dată inițială va fi citită de la ieșire și nu de la intrare.

Exemplu: Dacă acumulatorul conține valoarea 0C3H (11000011B) și registrul R0 valoarea 55H (01010101B), instrucțiunea

```
ORL A,R0
```

va lăsa în acumulator valoarea 0D7H(11010111B).

Când destinația este un octet adresat direct, instrucțiunea poate seta combinații de biți în orice locație RAM sau registru hardware. Șablonul biților care trebuie setați este determinat de un octet-mască, care poate fi o constantă în instrucțiune sau o variabilă creată în acumulator la rulare.

Instrucțiunea:

```
ORL P1, #00110010B
```


va seta biții 5, 4 și 1 ai portului de ieșire P1.

36	ORL	C, <scr-bit>
Funcția:	SAU LOGIC pentru variabile de tip bit	
Descriere:	Instrucțiunea ORL C, bit setează indicatorul de transport dacă valoarea booleană e un 1 logic. Instrucțiunea nu modifică starea indicatorului de transport. Simbolul (“/”) precedând operandul, în limbaj de asamblare, indică faptul că se utilizează complementul logic al bitului adresat ca valoare sursă, fără ca bitul sursă să fie afectat. Nici un alt indicator nu e afectat.	
Exemplu:	<p>Secvența prezentată setează indicatorul de transport dacă și numai dacă P1.0=1, ACC.7=1, OV=0:</p> <pre> ORL C, P1.0 ORL C, ACC.7 ORL C, /OV </pre>	
37	POP	direct
Funcția:	Extragere din stivă	
Descriere:	Instrucțiunea citește conținutul locației interne RAM adresată prin pointerul stivei, iar pointerul stivei este decrementat cu 1. Valoarea citită este apoi transferată în octetul direct adresabil indicat. Nu se afectează nici un indicator de condiții.	
Exemplu:	<p>Pointerul stivei conține inițial valoarea 32H și locațiile interne RAM cu adresele 30H până la 32H conțin valorile 20H, 23H, 01H. Instrucțiunile:</p> <pre> POP DPH POP DPL </pre> <p>lasă pointerul stivei setat la valoarea 30H și pointerul de data la 0123H. În acest punct, instrucțiunea:</p> <pre> POP SP </pre> <p>va seta pointerul stivei la valoarea 20H. În acest caz special, pointerul stivei a fost decrementat la valoarea 2FH înainte de încărcarea cu valoarea extrasă (20H).</p>	
38	PUSH	direct
Funcția:	Salvare în stivă	
Descriere:	Pointerul stivei, SP, este incrementat cu 1. Conținutul variabilei indicate e copiat în locația internă RAM adresată de pointerul stivei. Nici un alt indicator nu este afectat.	
Exemplu:	Întrând într-o rutină de întrerupere, pointerul stivei, SP,	

conține valoarea 09H. Pointerul de date conține valoarea 0123H. Instrucțiunile:

PUSH DPL
PUSH DPH

vor seta pointerul stivei la valoarea 0BH și vor încărca 23H și 01H în locațiile interne RAM cu adresele 0AH și 0BH.

39 RET

Funcția:	Revenire din subrutină
Descriere:	Instrucțiunea RET extrage succesiv octeții mai semnificativ și mai puțin semnificativ ai adresei din stivă, decrementind pointerul stivei cu 2. Execuția programului continuă de la adresa rezultată, în general instrucțiunea ce scdece instrucțiunile ACALL sau LCALL. Nici un indicator nu este afectat.
Exemplu:	Pointerul stivei conține inițial valoarea 0BH. Locațiile interne RAM cu adresele 0AH și 0BH conțin valorile 23H și, respectiv 01H. Instrucțiunea RET va seta pointerul stivei la valoarea 09H, iar execuția programului va continua de la locația cu adresa 0123H.

40 RETI

Funcția:	Revenire din rutina de tratare a unei întreruperi
Descriere:	Instrucțiunea RETI extrage succesiv octeții mai semnificativ și mai puțin semnificativ ai contorului programului, PC, din stivă și reactivează logica de întreruperi să accepte întreruperi suplimentare, cu același nivel de prioritate ca cea tocmai procesată. Pointerul stivei e decrementat prin 2. Nici un alt registru nu este afectat. Cuvântul de stare a programului, PSW, nu este reîncărcat automat cu starea sa anterioară tratării întreruperii. Execuția programului continuă de la adresa rezultată, care e în general, instrucțiunea imediat următoare celei după care a fost detectată apelarea întreruperii. Dacă o întrerupere cu nivel de prioritate mai mic sau egal a apărut în timp ce se executa RETI, acea instrucțiune va fi executată înaintea revenirii din întrerupere.
Exemplu:	Pointerul stivei conține valoarea inițială 0BH. O întrerupere a fost detectată în timpul instrucțiunii care se termină la locația cu adresa 0122H. Locațiile interne RAM cu adresele 0AH și 0BH conțin valorile 23H și respectiv 01H. Instrucțiunea: RETI va lăsa pointerul stivei setat la valoarea 09H și va continua

execuția programului de la adresa 0123H.

41	RL	A
Funcția:	Rotește acumulatorul la stânga	
Descriere:	Cei 8 biți ai acumulatorului sunt rotiți cu un bit la stânga. Bitul 7 ajunge în poziția bitului 0. Nici un indicator nu e afectat.	
Exemplu:	Acumulatorul conține valoarea inițială 0C5H (11000101B). Instrucțiunea: RL A va determina ca acumulatorul să conțină valoarea 8BH (10001011B), fără a afecta indicatorul de transport.	
42	RLC	A
Funcția:	Rotirea acumulatorului la stânga prin indicatorul de transport	
Descriere:	Cei 8 biți ai acumulatorului și indicatorul de transport sunt amândoi rotiți cu un bit la stânga. Bitul 7 ajunge în indicatorul de transport, iar acesta pe poziția bitului 0. Nu afectează nici un alt indicator.	
Exemplu:	Acumulatorul conține valoarea inițială 0C5H (11000101B), și indicatorul de transport este 0. Instrucțiunea: RLC A va lăsa în acumulator valoarea 8BH (10001010B), cu setarea indicatorului de transport.	
43	RR	A
Funcție:	Rotirea acumulatorului la dreapta	
Descriere:	Cei 8 biți ai acumulatorului sunt rotiți cu un bit la dreapta. Bitul 0 ajunge în poziția bitului 7. Nici un indicator nu e afectat.	
Exemplu:	Acumulatorul conține valoarea 0C5H (11000101B). Instrucțiunea: RR A va seta acumulatorul la valoarea 0E2H (11100010B), cu indicatorul de transport neafectat.	
44	RRC	A
Funcția:	Rotirea acumulatorului la dreapta prin indicatorul de transport.	
Descriere:	Cei 8 biți ai acumulatorului și indicatorul de transport sunt rotiți împreună cu un bit la dreapta. Bitul 0 ajunge în indicatorul de transport, iar acesta în poziția bitului 7. Nici un	

Exemplu: alt indicator nu este afectat.
 Acumulatorul conține valoarea 0C5H, iar transportul este 0.
 Instrucțiunea:
 RRC A
 va determina ca acumulatorul să conțină valoarea 62H (01100010B), cu indicatorul de transport setat la 1.

45 SETB <bit>
 Funcția: Setează bitul specificat
 Descriere: Instrucțiunea SETB setează bitul indicat la 1. SETB poate opera asupra indicatorului de transport sau oricărui alt bit direct adresabil. Nu afectează alți indicatori.
 Exemplu: Indicatorul de transport este resetat. Portul de ieșire P1 a fost înscris cu valoarea 34H (00110100B). Instrucțiunile:
 SETBC
 SETBP1.0
 vor seta indicatorul de transport la 1 și data de ieșire la portul P1 va fi 35H (00110101B).

46 SJMP rel
 Funcția: Short Jump
 Descriere: Instrucțiunea SJMP determină un salt necondiționat în program, la adresa indicată. Destinația saltului este compusă prin sumarea deplasării cu semn, în al doilea octet al instrucțiunii, cu conținutul contorului programului, PC, după incrementarea PC de două ori. Saltul permis are o valoare de la -128 de octeți (precedenți instrucțiunii SJMP) la +127 octeți (următori instrucțiunii SJMP).
 Exemplu: Eticheta RELADR e asociată instrucțiunii de la locația cu adresa 0123H. Instrucțiunea:
 SJMPRELADR
 va duce la locația cu adresa 0100H. După ce instrucțiunea este executată, PC va conține valoarea 0123H.

47 SUBB A, <src-byte>
 Funcția: Scădere cu împrumut.
 Descriere: Instrucțiunea SUBB va scădea atât variabila octet indicată, cât și indicatorul de transport din acumulator, lăsând rezultatul în acumulator. SUBB setează indicatorul de transport (de împrumut) dacă este nevoie de un împrumut pentru bitul 7 și sterge indicatorul C altfel (dacă C a fost setat înaintea execuției instrucțiunii SUBB, aceasta indică că a fost

necesar un împrumut la pasul anterior, într-o scădere cu precizie multiplă, astfel încât transportul este scăzut din acumulator odată cu operandul sursă). Indicatorul AC e setat dacă a fost necesar un împrumut pentru bitul 3 și resetat în caz contrar. Indicatorul OV e setat dacă e necesar un împrumut la bitul 6, dar nu la bitul 7, sau la bitul 7, dar nu la bitul 6. La scăderea întregilor cu semn, OV indică un număr negativ când o valoare negativă e scăzută dintr-o valoare pozitivă, sau un rezultat pozitiv când un număr pozitiv e scăzut dintr-un număr negativ. Operandul sursă permite 4 moduri de adresare: prin registru, directă, indirectă prin registru, imediată.

Exemplu: Acumulatorul conține valoarea 0C9H (11001001B), registrul R2 conține valoarea 54H (01010100B) și indicatorul de transport e setat. Instrucțiunea:

SUBB A, R2

va determina valoarea 74H (01110100B) în Acumulator, cu indicatorul de transport și AC reșetați, dar indicatorul OV setat. Se observă că:

$0C9H - 54H = 75H$

Diferența între acest rezultat și cel de mai sus este datorat faptului că indicatorul de transport (împrumut) a fost setat înaintea operației. Dacă starea indicatorului de transport nu e cunoscută înainte de începerea unei scăderi în precizie simplă sau multiplă, va fi în mod explicit resetat de o instrucțiune:

CLR C.

48	SWAP	A
Funcție:	Interschimb intern în acumulator	
Descriere:	Instrucțiunea SWAP A interșchimbă câmpurile de câte 4 biți, mai semnificativ și mai puțin semnificativ, ale acumulatorului (biții 7÷4 și biții 3÷0). Operația poate, de asemenea, să fie gândită ca o instrucțiune de rotație pe 4 biți. Nici un indicator nu este afectat.	
Exemplu:	Acumulatorul conține valoarea 0C5H (11000101B). Instrucțiunea: SWAP A determină în acumulator valoarea 5CH (01011100B).	
49	XCH	A, <byte>
Funcție:	Schimbă conținutul acumulatorului cu o variabilă de tip octet	
Descriere:	Instrucțiunea XCH încarcă acumulatorul cu conținutul	

variabilei indicate, scriind în același timp conținutul original al acumulatorului în variabila de tip octet indicată. Operatorii sursă și destinație pot folosi adresarea prin registru, adresarea directă sau indirect prin registru.

Exemplu: Registrul R0 conține adresa 20H. Acumulatorul conține valoarea 3FH (00111111B). Locația internă RAM cu adresa 20H conține valoarea 75H (01110101B). Instrucțiunea:

XCH A,@R0

va determina ca locația RAM cu adresa 20H să conțină valoarea 3FH (00111111B) și 75H (01110101B) în acumulator.

50 XCHD A,@Ri

Funcția: Schimbă digit

Descriere: Instrucțiunea XCHD schimbă câmpul de 4 biți mai puțin semnificativ al acumulatorului (biții 3÷0), reprezentând în general un digit hexazecimal sau BCD, cu acela al locației interne RAM adresată indirect prin registrul specificat. Câmpul de 4 biți mai semnificativ (biții 7÷4) ai registrelor nu sunt afectați. Nici un indicator de condiții nu este afectat.

Exemplu: Registrul R0 conține adresa 20H. Acumulatorul conține valoarea 36H (00110110B). Locația internă RAM cu adresa 20H conține valoarea 75H (01110101B). Instrucțiunea:

XCHD A, @R0

va determina ca locația RAM cu adresa 20H să conțină valoarea 76H (01110110B) și acumulatorul 35H (00110101B).

51 XRL <dest-byte>, <src-byte>

Funcție: SAU EXCLUSIV între variabile de tip octet

Descriere: Instrucțiunea XRL realizează funcția SAU EXCLUSIV la nivel de bit între variabilele octet indicate, încărcând rezultatul în octetul destinație. Nu afectează indicatorii de condiții. Cei doi operanzi permit 6 combinații de moduri de adresare. Când destinația este acumulatorul, sursa poate fi adresată ca registru, direct, registru-indirect sau imediat; când destinația este o adresă directă, sursa poate fi acumulatorul sau o dată imediată.

Notă: Atunci când această instrucțiune e utilizată ca să modifice un port de ieșire, valoarea utilizată ca dată inițială va fi citită de la ieșire, nu de la intrare.

Exemplu: Acumulatorul conține valoarea 0C3H (11000011B) și

registru R0 conține valoarea 0AAH (10101010B).
Instrucțiunea:

XRL A, R0

va determina ca acumulatorul să conțină valoarea 69H (01101001B).

Când destinația e un octet direct adresabil, această instrucțiune poate încărca complementele logice ale combinațiilor de biți în orice locație RAM sau registru hardware. Șablonul biților care vor fi complementați e determinat de un octet mască, ce poate fi atât o constantă conținută în instrucțiune, cât și o variabilă obținută în acumulator în timpul rulării programului. Instrucțiunea:

XRL P1, #00110001B

va complementa biții 5, 4 și 0 ai portului de ieșire P1.

5.3 SISTEM DE DEZVOLTARE CU MICROCONTROLLER 80C552

Sistemul de dezvoltare inițial, **IMC500**, a fost astfel conceput încât permite dezvoltarea rapidă a aplicațiilor în domenii diverse - automatizări industriale, aparate de măsură, industrie ușoară, medicină, industria automobilelor, domeniul casnic, etc.

Utilizarea tehnologiei **CMOS** îl recomandă pentru aplicațiile care necesită un consum redus de energie și care necesită imunitate ridicată la perturbații.

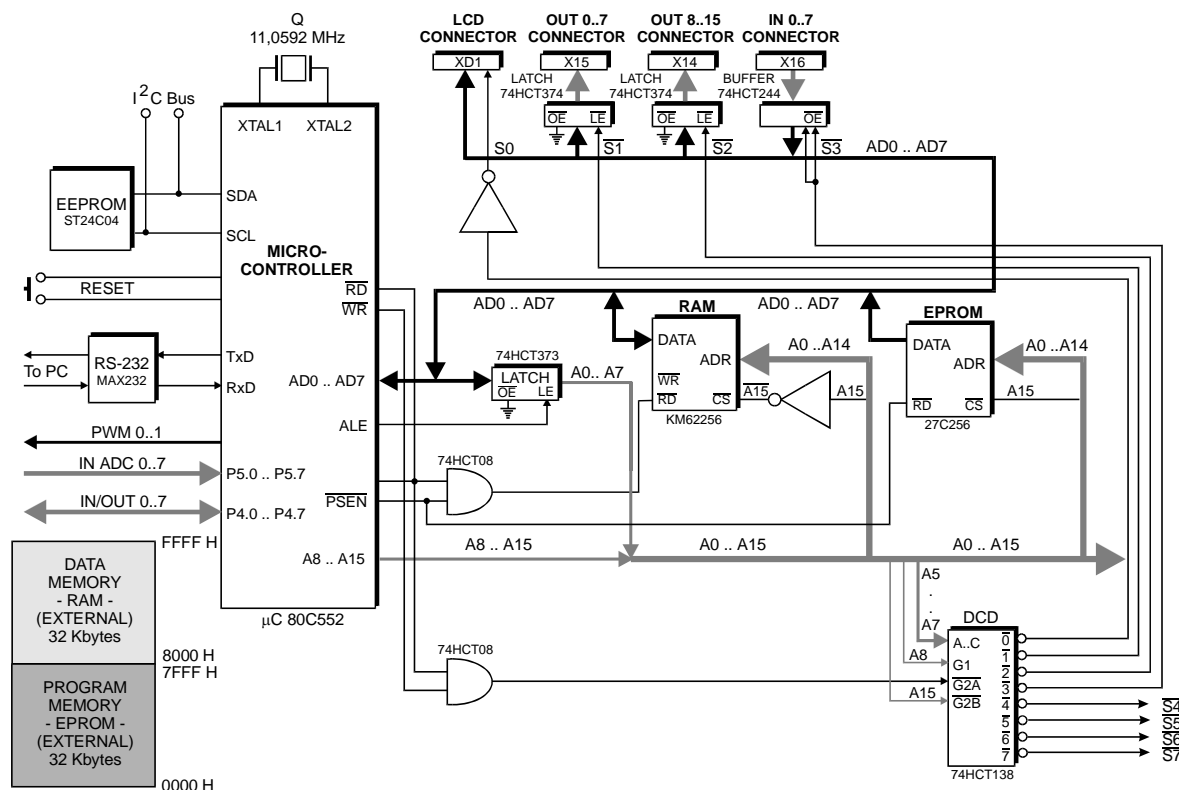
Prețul scăzut de cost îl recomandă atât pentru produsele de serie, cât și pentru prototipuri și unicate.

În fig. 5.8 este prezentată structura generală a sistemului de dezvoltare **IMC500**.

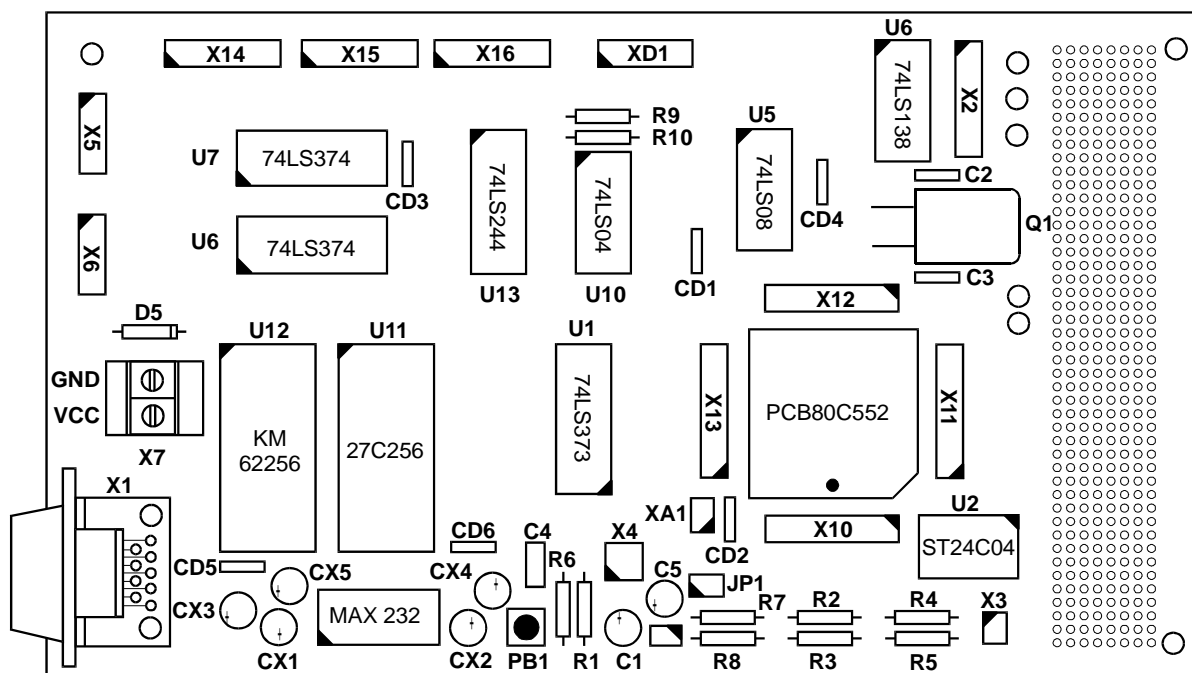
În fig. 5.9 este prezentată detaliat structura de interconexiune cu exteriorul a sistemului de dezvoltare cu microcontroller **80C552**.

Sistemul de dezvoltare este destinat în principal dezvoltării de programe. **Hardware**-ul suplimentar utilizat - de exemplu tastatura, afișaj cu cristale lichide, etc. - permite unificarea din punct de vedere constructiv a diferitelor produse. Acest proces de unificare **hardware** direcționează efortul de proiectare spre programe de aplicație.

Acest sistem de dezvoltare, bazat pe microcontroller-ul **80C552**, acoperă din punct de vedere **hardware** și **software** aplicațiile dezvoltate cu microprocesoarele **80C31**, **80C32** și alte procesoare din familia **8051**, putând fi folosit la dezvoltarea de aplicații cu aceste procesoare.



a) Structura de bază a sistemului de dezvoltare **IMC500**.



b) Amplasarea componentelor sistemului de dezvoltare **IMC500**.

Fig. 5.8 Sistemul de dezvoltare **IMC500**.

CONECTOR X1 (SERIAL_LINK)			
1	NC	2	TXD
3	RXD	4	NC
5	GND	6	NC
7	NC	8	NC
9	NC		
CONECTOR X2 (DECODE)			
1	VCC	2	S0/
3	VCC	4	S1/
5	VCC	6	S2/
7	NC	8	S3/
9	NC	10	S4/
11	GND	12	S5/
13	GND	14	S6/
15	GND	16	S7/
CONECTOR X3 (I2C_LINK)			
1	SDA	2	SCL
CONECTOR X4 (INTERNAL_MEM)			
1	VCC	2	EA/
3	GND	4	EA/
CONECTOR X5 (ROM_SEL)			
1	PSEN/	2	CE_P
3	RD/	4	CE_P
CONECTOR X6 (ROM_EN)			
1	VCC_P	2	VCC
3	GND	4	NC
CONECTOR X7 (MAIN_SUPPLY)			
1	GND	2	VCC
CONECTOR X8 (JP1)			
1	EW/	2	GND
CONECTOR X9 (SW1_RESET)			
1	R6_VCC	2	R1_GND
CONECTOR X10 (μC)			
1	CMSR2	2	CMSR4
3	CMSR0	4	CMSR1
5	PWM1	6	EW/
7	STAD	8	PWM1
9	ADC0	10	VCC
11	ADC2	12	ADC1
13	ADC4	14	ADC3
15	ADC6	16	ADC5
17	AVDD	18	ADC7
CONECTOR X11 (μC)			
1	CMSR4	2	CMSR3
3	CMT2	4	CMSR5
5	RST	6	CMT1
7	CT1I	8	CT0I
9	CT3I	10	CT2I
11	RT2	12	T2
13	PSDA	14	PSCL
15	PTXD	16	PRXD
17	T0	18	INT0
CONECTOR X12 (μC)			
1	T0	2	INT1
3	WR/	4	T1
5	NC	6	RD/
7	XT2	8	NC
9	GND	10	XT1
11	NC	12	GND
13	A9	14	A8
15	A11	16	A10
17	A14	18	A12
CONECTOR X13 (μC)			
1	AVSS	2	ADC7
3	AVREF-	4	AVREF+
5	AD1	6	AD0
7	AD3	8	AD2
9	AD5	10	AD4
11	AD7	12	AD6
13	ALE	14	EA/
15	A15	16	PSEN/
17	A13	18	A14
CONECTOR X14 (OUT_HIGH)			
1	AX8	2	VCC
3	AX9	4	VCC
5	AX10	6	NC
7	AX11	8	NC
9	AX12	10	NC
11	AX13	12	NC
13	AX14	14	GND
15	AX15	16	GND
CONECTOR X15 (OUT_LOW)			
1	AX0	2	VCC
3	AX1	4	VCC
5	AX2	6	NC
7	AX3	8	NC
9	AX4	10	NC
11	AX5	12	NC
13	AX6	14	GND
15	AX7	16	GND
CONECTOR X16 (INPUT)			
1	IX0	2	VCC
3	IX1	4	VCC
5	IX2	6	NC
7	IX3	8	NC
9	IX4	10	NC
11	IX5	12	NC
13	IX6	14	GND
15	IX7	16	GND
CONECTOR XA1 (SUPPLY)			
1	VCC	2	GND
CONECTOR XD1 (LCD)			
1	VSS	2	VDD
3	V0	4	RS
5	R/W	6	EN
7	D0	8	D1
9	D2	10	D3
11	D4	12	D5
13	D6	14	D7

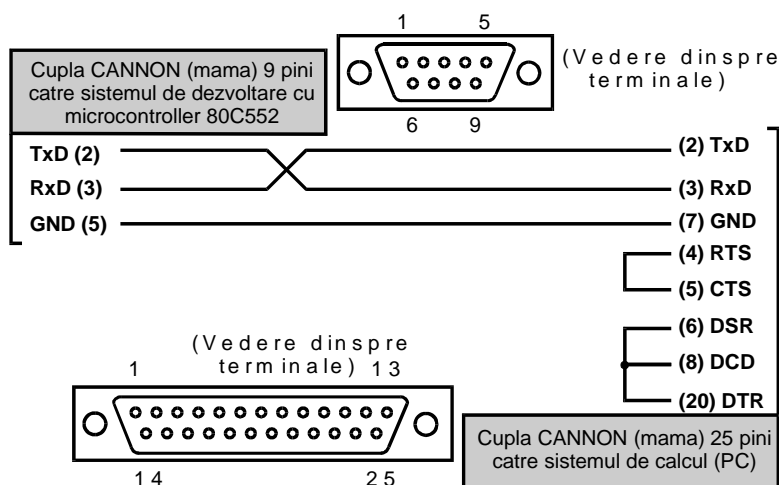


Fig. 5.9 Structura interconexiunilor sistemului de dezvoltare IMC500.

Sistemul de dezvoltare realizat dispune de următoarele resurse *hardware* și caracteristici tehnice:

- microcontroller **PCB80C552** (fără memorie internă de program), lucrând la o frecvență maximă a ceasului de 16 MHz;
- frecvența ceasului sistemului de dezvoltare 11,059200 MHz;
- memoria de date externă (**DATA MEMORY**), statică, implementată cu un circuit de tip **KM62256AL**, realizat în tehnologie **CMOS**, cu capacitatea de 32 kocteți și caracterizat de un timp de acces de 35ns. Spațiul de adresare ocupat de memoria de date, în cadrul sistemului de dezvoltare, este cuprins între adresele 8000H și FFFFH. Selectarea memoriei RAM, activă pe nivel coborât, se efectuează cu semnalul \overline{A}_{15} , iar semnalele de control sunt \overline{WR} pentru scriere și $\overline{PSEN} \cdot \overline{RD}$ pentru citire (pentru a se putea rula din memoria RAM aplicațiile transferate pe interfața serială de la PC);
- memoria de program externă (**PROGRAM MEMORY**), implementată cu un circuit **EPROM** de tip **27C256**, realizat în tehnologie **CMOS**, cu capacitatea de 32 kocteți și caracterizat de un timp de acces de 70ns. Spațiul de adrese ocupat de memoria de program externă în cadrul sistemului de dezvoltare, este cuprins între 0000H și 7FFFH. Selectarea memoriei RAM, activă pe nivel coborât, se efectuează cu semnalul A_{15} , iar semnalul de control este \overline{PSEN} pentru citire. Memoria externă de program conține programul de aplicație sau în faza de dezvoltare a acestuia conține un program monitor;
- interfață serială compatibilă **RS-232** de mare viteză, full duplex, fără semnale de dialog, funcționând doar cu protocol *software*;
- bus serial **I²C** (bus *multimaster* cu arbitrare de priorități și viteză mare de transmisie - 100 kbytes pe secundă în modul standard și 400 kbytes pe secundă în modul rapid -, frecvența maximă a ceasului serial este 100 kHz. Destinația principală este comunicația cu circuite integrate sau controller-e, prevăzute cu interfața **I²C**, aflate în aceeași carcasă;
- memorie **EEPROM** serială, implemetată cu un circuit de tip **ST24C04**, realizat în tehnologie **CMOS**, cu capacitatea de 512 octeți și conectată la interfața **I²C**;
- 2 porturi paralele de ieșire de 8 biți;
- 1 port paralel de intrare de 8 biți;
- 8 intrări multiplexate la un convertor analog-digital cu rezoluția de 10 biți, implementat în structura microcontroller-ului 80C552 și caracterizat de un timp de conversie de 50 cicluri mașină (aproximativ 50 μs);
- 8 ieșiri decodificate de selecție porturi, specificate în cadrul tabelului

5.1;

Dintre cele 8 semnale de decodificare porturi ocupă un spațiu de adrese cu dimensiunea de FFH, așa după cum reiese din tabelul prezentat anterior. Dintre cele 8 semnale de selecție sintetizate, în structura sistemului de dezvoltare sunt utilizate doar 4, și anume:

- \overline{S}_0 - semnal de selecție pentru afișajul cu cristale lichide LCD;
- \overline{S}_1 - semnal de selecție pentru portul de ieșire mai puțin semnificativ;
- \overline{S}_2 - semnal de selecție pentru portul de ieșire mai semnificativ;
- \overline{S}_4 - semnal de selecție pentru portul de intrare;
- $\overline{S}_4 \div \overline{S}_7$ - neutilizate (disponibile pentru extensii *hardware*);

Tabelul 5.1 Spațiul de adrese pentru selecțiile de porturi.

Liniiile de adrese A ₀ ÷ A ₁₅															Ieșire DCD	Adresa (H)	
A ₁₅	A ₁₄	A ₁₃	A ₁₂	A ₁₁	A ₁₀	A ₉	A ₈	A ₇	A ₆	A ₅	A ₄	A ₃	A ₂	A ₁			A ₀
0	X	X	X	X	X	X	1	0	0	0	X	X	X	X	X	\overline{S}_0	100H..11FH
0	X	X	X	X	X	X	1	0	0	1	X	X	X	X	X	\overline{S}_1	120H..13FH
0	X	X	X	X	X	X	1	0	1	0	X	X	X	X	X	\overline{S}_2	140H..15FH
0	X	X	X	X	X	X	1	0	1	1	X	X	X	X	X	\overline{S}_3	160H..17FH
0	X	X	X	X	X	X	1	1	0	0	X	X	X	X	X	\overline{S}_4	180H..19FH
0	X	X	X	X	X	X	1	1	0	1	X	X	X	X	X	\overline{S}_5	1A0H..1BFH
0	X	X	X	X	X	X	1	1	1	0	X	X	X	X	X	\overline{S}_6	1C0H..1DFH
0	X	X	X	X	X	X	1	1	1	1	X	X	X	X	X	\overline{S}_7	1E0H..1FFH
MOV P ₂ ,#1								MOV R ₀ ,#(A ₇ A ₆ A ₅ A ₄ A ₃ A ₂ A ₁ A ₀)B MOV @R ₀ ,A									

Extinderea numărului de porturi de intrare-ieșire poate fi făcută fie prin utilizarea semnalelor de selecție disponibile, ceea ce conduce la încărcarea magistralei interne a sistemului de dezvoltare, fie prin subdecodificarea liniilor inferioare de adrese neutilizate $A_4 \div A_0$ și multiplexarea, respectiv demultiplexarea, intrărilor, respectiv a ieșirilor, portului de intrare, respectiv a portului de ieșire mai puțin semnificativ. Procesul de multiplexare se realizează bazat pe circuite cu ieșiri de tip *three-state*, minimizând deci numărul de resurse *hardware* suplimentare necesare;

- 2 ieșiri analogice de 8 biți modulate în durată. Prin integrarea lor se pot obține două convertoare digital-analogice de 8 biți;
- 3 numărătoare de tip *timer* / *counter*;

- **1 *watchdog*** programabil (mijloc de auto-deblocare în cazul execuției eronate a programelor, datorită perturbațiilor sau interferențelor;
- **15** linii de întreruperi, dintre care **6** linii externe;
- reset la punerea sub tensiune;
- conectarea directă a unui afișaj cu cristale lichide.

5.3.1 DOMENIUL DE APLICABILITATE

Echipamentul prezentat, poate fi ușor extins la un sistem de măsură și control construit în jurul unui microcontroller tip 80C552, destinat unei largi clase de aplicații în:

- **mediul industrial:** măsurarea / reglarea unor parametri: temperatura, nivelul, turația etc.;
- **procesul de învățământ:** pentru dotarea laboratoarelor de automatizări industriale;
- **dezvoltarea unor programe de aplicații.**

Echipamentul poate funcționa independent sau conectat la un sistem de calcul ierarhic superior.

5.3.2 DETALIEREA RESURSELOR SISTEMULUI

5.3.2.1 UNITATEA CENTRALĂ DE PRELUCRARE

- Microcontroller de tip 80C552, cu frecvența ceasului de 11,0592 MHz;
- Memorie RAM externă, 32 Kocteți;
- Memorie EPROM externă, 32 Kocteți;
- Memorie EEPROM externă, 512 octeți, conectată pe magistrala I²C.

5.3.2.2 INTERFAȚA CU PROCESUL CONTROLAT

- **2 intrări analogice** pentru semnal unificat în curent. Sistemul dispune de două intrări analogice: IN_ANA_1 și IN_ANA_2, pentru măsurarea semnalelor de intrare în curent în domeniul (4...20)mA, furnizate de un traductor.

Semnalele de intrare în curent sunt convertite intern în tensiune în domeniul (0,4...2)V.

Acestea corespund canalelor de intrare ADC0 și ADC1 ale portului

P5 al microcontroller-ului 80C552.

Manipularea conversiei A/D se face prin registrele ADCON și ADCH.

Caracteristici:

1. Semnal de intrare: (4...20)mA
2. Caracteristica de transfer a convertorului analog-digital:

$$N = 2^{10} \times \frac{V_{IN} - V_{REF-}}{V_{REF+} - V_{REF-}}, \text{ cu rezultatul reprezentat pe 10 biti} \quad (5.6)$$

$$N = 2^8 \times \frac{V_{IN} - V_{REF-}}{V_{REF+} - V_{REF-}}, \text{ cu rezultatul reprezentat pe 8 biti}$$

în care $V_{REF-} = 0,4V$; $V_{REF+} = 2V$ și $V_{IN} = (0,4...2)V$;

3. Furnizează tensiunea de alimentare a traductorului: 24V / max. 100mA

Demararea conversiei A/D implică programarea registrului ADCON.

1. Selectarea canalului de intrare (biții ADR2...ADR0);
2. Declanșarea software a conversiei: ADCS=1;
3. Testarea sfârșitului conversiei: ADCI=1.

Rezultatul conversiei este depus în registrele ADCH (AD9...AD2) și ADCON (AD1, AD0).

Registrul ADCON. Adresa: **C5H**. Valoare la reset: xx000000H

Bit 7 (MSB)					Bit 0 (LSB)		
A/D1	A/D0	ADEX	ADCI	ADCS	ADR2	ADR1	ADR0
					0 0 0 - canal de intrare ADC0;		
					0 0 1 - canal de intrare ADC1;		
					1 1 1 - canal de intrare ADC7.		
					Setarea la 1 - declanșarea conversiei A/D;		
					0 - conversie în curs de desfășurare;		
					1 - sfârșit de conversie.		
					0 -conversie declanșată software;		
					1 - conversie declanșată fie hardware prin intermediul semnalului STADC, fie software ca în cazul precedent		
Bitul 0 (LSB) al rezultatului							
Bitul 1 al rezultatului							

Registrul ADCH. Adresa: **C6H**. Valoare la reset: xxxxxxxxH

Bit 7 (MSB)				Bit 0 (LSB)			
A/D9	A/D8	A/D7	A/D6	A/D5	A/D4	A/D3	A/D2
Biții mai semnificativi ai rezultatului conversiei							

Secvență de cod, în limbaj de asamblare, pentru conversia analog-digitală:

```
MOV    A, #0          ; CONVERSIE A/D CANAL 0
MOV    ADCON, A
```

```

        ORL    A, #08H
        MOV    ADCON, A
WAIT1:  MOV    A, ADCON
        JNB    ACC.4, WAIT1
        MOV    A, ADCH

sau
        MOV    A, #1          ; CONVERSIE A/D CANAL 1
        MOV    ADCON, A
        ORL    A, #08H
        MOV    ADCON, A
WAIT1:  MOV    A, ADCON
        JNB    ACC.4, WAIT1
        MOV    A, ADCH
    
```

- **2 intrări logice**, compatibile TTL, cu rezistențe interne de pull-up, active pe nivel, pentru captarea unor condiții externe. Sistemul dispune de două intrări logice: IN_LOG_1 și IN_LOG_2, pentru captarea unor condiții externe.

Caracteristici:

1. semnale de intrare compatibile TTL;
2. intrările sunt prevăzute cu rezistențe interne de pull-up;
3. intrările sunt active pe nivel.

Determinarea stării intrărilor implică citirea Port Intrări Logice.

Port intrări logice. Adresa: **160H**

Bit 7 (MSB)						Bit 0 (LSB)	
X	X	X	X	X	X	In_Log_2	In_Log_1
Neutilizați						Intrări logice	

Secvență de cod în limbaj de asamblare pentru citirea intrărilor logice:

```

        MOV    P2, #1
        MOV    R0, #60H
        MOVX   A, @R0
    
```

- **1 intrare digitală**, activă pe front, pentru măsurarea duratei între două evenimente externe succesive. Această intrare este dedicată conectării unui senzor cu ultrasunete. Semnalul de intrare este de tip impuls pozitiv cu amplitudinea cuprinsă între (2...12)V;
- **1 intrare digitală**, activă pe front, pentru măsurarea frecvenței unui semnal periodic. Semnalul de intrare este de tip impuls pozitiv cu amplitudinea cuprinsă între (2...12)V sau de tip alternativ, cu amplitudinea cuprinsă în intervalul (4...24)V_{vv}. **Alternativ**, această intrare poate fi reconfigurată hardware cu caracteristicile de la punctul anterior;

Sistemul dispune de două canale de intrare pentru determinarea duratei între două evenimente externe succesive și implicit a frecvenței unui semnal de intrare.

Primul canal este asociat cu registrele de captare CT0 și CT1.

Al doilea canal este asociat cu registrele de captare CT2 și CT3.

Registrele de captare CT0, CT1, CT2 și CT3 sunt cuplate cu timer-ul T2 și captează starea acestuia la apariția semnalului de captare asociat CT0I, CT1I, CT2I și CT3I.

Semnalele de captare poziționează indicatorii de stare asociați (CTI0, CTI1, CTI2 și CTI3) din registrul TM2IR.

Durata între evenimente rezultă din diferența conținutului registrelor de captare.

Fiecare canal are asociate două semnale:

- câte un semnal de ieșire pentru comanda captării, SEND_1, respectiv SEND_2. Aceste semnale inițializează atât circuitele interne de captare (semnalele de captare CT0I respectiv CT2I) cât și dacă este cazul, circuitele proprii senzorului conectat.
- câte o intrare pentru semnalul captat: ECHO_1 respectiv ECHO_2. Primul front crescător al semnalului de intrare activează semnalele de captare CT0I respectiv CT2I (are ca efect captarea în registrul CT0 respectiv CT2 a conținutului timer-ului T2 la acest moment de timp) și inițializează semnalele de captare CT1I respectiv CT3I. Al doilea front crescător al semnalului de intrare activează semnalele de captare CT1I respectiv CT3I (are ca efect captarea în registrul CT1 respectiv CT3 a conținutului timer-ului T2 la acest moment de timp).

Prima intrare (SEND_1, ECHO_1) este activă pe front crescător și este rezervată pentru măsurarea duratei între două evenimente externe succesive.

Această intrare permite conectarea unui senzor de nivel cu ultrasunete din seria -wms- al firmei Microsonic.

A doua intrare (SEND_2, ECHO_2) este de asemenea activă pe front crescător și este rezervată măsurării frecvenței unui semnal periodic. Această intrare permite conectarea unei sonde pentru măsurarea frecvenței de tip optic sau inductiv sau a unui generator de semnal.

Caracteristici:

1. SEND_1 și SEND_2: ieșiri open-colector;
2. ECHO_1: intrare de semnal, impuls pozitiv, cu amplitudinea (2...12)V;
3. ECHO_2: intrare de semnal, impuls pozitiv, cu amplitudinea (2...12)V sau semnal periodic alternativ cu amplitudinea (4...24)V_{vv}.

Inițializarea secțiunii de captare implică:

- dezactivarea întreruperilor provenite de la:
 - depășirea superioară pe 8/16 biți a conținutului timer-ului T2.
 - registrele de captare a evenimentelor.
 - circuitele de comparare a conținutului timer T2 cu registrele de comparare.
- inițializarea (încărcare cu 0) a timer-ului TML2/TMH2.
- programarea registrului CTCON astfel încât registrele de captare CT0, CT1, CT2 și CT3 să fie activate pe fronturile crescătoare ale semnalului de intrare.
- programarea timer-ului T2 prin activarea acestuia, cu dezactivarea depășirii pe 8/16 biți, ceasul de numărare fiind constituit de un oscilator intern cu frecvența de 1/12 din frecvența oscilatorului microcontroller-ului urmat de un registru de divizare cu 2.

Notă: Registrele de comparare CM0...CM2 și logica aferentă, controlate prin intermediul registrelor RTE și STE, pot fi utilizate pentru supravegherea hardware a încadrării unor parametri între limite programabile.

Registrul TM2CON. Adresa **EAH**. Valoare la reset **00H**

Bit 7 (MSB)				Bit 0 (LSB)			
T2IS1	T2IS0	T2ER	T2B0	T2P1	T2P0	T2MS1	T2MS0
							0 0 - Timer T2 oprit; 0 1 - ceas = $f_{osc}/12$; 1 0 - mod test; 1 1 - ceas extern (T2).
							0 0 - ceas : 1 (intern sau extern); 0 1 - ceas : 2 (intern sau extern); 1 0 - ceas : 4 (intern sau extern); 1 1 - ceas : 8 (intern sau extern).
							Indicator întrerupere depășire pe 8 biți Timer T2
							Activare reset extern Timer T2; dacă este 1, T2 poate fi resetat cu un front crescător pe pinul RT2 (P1.5).
							Selectare întrerupere de depășire pe 8 biți Timer T2
							Selectare întrerupere de depășire pe 16 biți Timer T2

Registrul IEN1. Adresa **E8H**. Valoare la reset **00H**

Bit 7 (MSB)				Bit 0 (LSB)			
ET2	ECM2	ECM1	ECM0	ECT3	ECT2	ECT1	ECT0
							Activare întrerupere registru captare 0
							Activare întrerupere registru captare 1
							Activare întrerupere registru captare 2
							Activare întrerupere registru captare 3
							Activare întrerupere comparator 0 timer T2
							Activare întrerupere comparator 1 timer T2
							Activare întrerupere comparator 2 timer T2
							Activare întrerupere depășire timer T2

Registrul CTCON. Adresa **EBH**. Valoare la reset **00H**

Bit 7 (MSB)				Bit 0 (LSB)			
CTN3	CTP3	CTN2	CTP2	CTN1	CTP1	CTN0	CTP0
							Captare pe front ↑ al CT0I
							Captare pe front ↓ al CT0I
							Captare pe front ↑ al CT1I
							Captare pe front ↓ al CT1I
							Captare pe front ↑ al CT2I
							Captare pe front ↓ al CT2I
							Captare pe front ↑ al CT3I
							Captare pe front ↓ al CT3I

Registrul RTE. Adresa **EFH**. Valoare la reset **00H**

Bit 7 (MSB)				Bit 0 (LSB)			
TP4.7	TP4.6	RP4.5	RP4.4	RP4.3	RP4.2	RP4.1	RP4.0
							Dacă e 1, P4.0 e resetat la egalitatea CM1 și T2
							Dacă e 1, P4.1 e resetat la egalitatea CM1 și T2
							Dacă e 1, P4.2 e resetat la egalitatea CM1 și T2
							Dacă e 1, P4.3 e resetat la egalitatea CM1 și T2
							Dacă e 1, P4.4 e resetat la egalitatea CM1 și T2
							Dacă e 1, P4.5 e resetat la egalitatea CM1 și T2
							Dacă e 1, P4.6 basculează la egalitatea CM2 și T2
							Dacă e 1, P4.7 basculează la egalitatea CM2 și T2

Registrul STE. Adresa **EEH**. Valoare la reset **00H**

Bit 7 (MSB)				Bit 0 (LSB)			
TG4.7	TG4.6	SP4.5	SP4.4	SP4.3	SP4.2	SP4.1	SP4.0
							Dacă e 1, P4.0 e setat la egalitatea CM0 și T2
							Dacă e 1, P4.1 e setat la egalitatea CM0 și T2
							Dacă e 1, P4.2 e setat la egalitatea CM0 și T2
							Dacă e 1, P4.3 e setat la egalitatea CM0 și T2
							Dacă e 1, P4.4 e setat la egalitatea CM0 și T2
							Dacă e 1, P4.5 e setat la egalitatea CM0 și T2
							Basculare bistabil
							Basculare bistabil

Registrul TM2IR. Adresa **C8H**. Valoare la reset **00H**

Bit 7 (MSB)							Bit 0 (LSB)
T2OV	CM2	CM1	CM0	CTI3	CTI2	CTI1	CTI0
							Indicator întrerupere CTI0
						Indicator CTI1	Indicator întrerupere CTI1
					Indicator întrerupere CTI2		
			Indicator întrerupere CTI3				
		Indicator întrerupere CM0					
		Indicator întrerupere CM1					
	Indicator întrerupere CM2						
Indicator depășire pe 16 biți a timer-ului T2							

Demararea captării implică activarea semnalelor de comandă Send_1 respectiv Send_2 prin bascularea 0-1-0 a bitului 0 respectiv 1 al portului de ieșiri logice.

Port ieșiri logice. Adresa: **140H**

Bit 7 (MSB)						Bit 0 (LSB)	
Linie 4	Linie 3	Linie 2	Linie 1	Out_Log_2	Out_Log_1	Send_2	Send_1
						Comandă Captare	

Secvență de program în limbaj de asamblare pentru comanda captării:

```

MOV      A, #xxxxxx00B
MOV      P2, #1
MOV      R0, #40H
MOVX     @R0, A
    
```

- **1 ieșire analogică** în semnal unificat (4...20)mA;
 - **1 ieșire analogică** tip **PWM** în impulsuri de curent (0...16)mA.
- Alternativ**, fiecare ieșire analogică se poate reconfigura hardware.

Sistemul dispune de două ieșiri analogice: OUT_ANA_1 și OUT_ANA_2 pentru generarea semnalelor de semnal unificat în curent.

Aceste semnale se obțin prin integrarea celor două ieșiri modulate în durată din cadrul structurii microcontroller-ului 80C552 astfel: OUT_ANA_1 corespunde cu PWM0, iar OUT_ANA_2 cu PWM1.

Ieșirea în curent este proporțională cu valoarea factorului de umplere al semnalului generat pe calea PWM0 respectiv PWM1.

Caracteristici:

1. Frecvența semnalelor de la cele două ieșiri (se recomandă alegerea unei frecvențe de lucru de aproximativ 1KHz) este aceeași și este programată prin intermediul registrului PWMP (adresa FEH), conform ecuației:

$$f_{\text{PWM}} = \frac{f_{\text{OSC}}}{2 \times (1 + \text{PWMP}) \times 255} \quad (5.7)$$

2. Durata impulsurilor este determinată de registrele PWM0 și PWM1 (adresele FCH, respectiv FDH), fiind direct proporțională cu valoarea negată a acestora;
3. Factorul de umplere al semnalelor de ieșire este determinat de ecuația:

$$\gamma_i = \frac{\text{PWM}_i}{255 - \text{PWM}_i} \quad (5.8)$$

4. semnalul de ieșire este izolat optic;
5. sarcina maximă admisă: 500Ω;
6. semnal analogic de ieșire: (4...20)mA;
7. semnal de ieșire în impulsuri: (0...16)mA.

Nota: 1. În varianta de echipare standard la ieșirea OUT_ANA_1 este generat un semnal unificat în curent, 4mA...20mA, iar la ieșirea OUT_ANA_2 este generat un semnal de tip PWM în impulsuri de curent 0mA...16mA.

2. Alternativ, fiecare ieșire poate fi reconfigurată hardware cu caracteristicile de la punctele 6 respectiv 7.

- **2 ieșiri logice** pentru comenzi externe, izolate galvanic (releu, 2x N.I./N.D., 6A / 380V).

Sistemul dispune de două ieșiri logice pentru comenzi externe: OUT_LOG_1 și OUT_LOG_2.

Caracteristici:

1. Ieșirile sunt izolate galvanic.
2. Comanda externă se face prin intermediul unei perechi de contacte N.I / N.D ale unui releu și admite o sarcină de max. 6A / 380V.

Generarea comenzilor externe implică activarea semnalelor de comandă Out_Log_1 și respectiv Out_Log_2 prin poziționarea în "1" a bitului 2 respectiv 3 al portului de ieșiri logice.

Port ieșiri logice. Adresa: 140H

Bit 7 (MSB)				Bit 0 (LSB)			
Linie 4	Linie 3	Linie 2	Linie 1	Out_Log_2	Out_Log_1	Send_2	Send_1
Comandă ieșiri logice							

Manipularea portului de ieșiri logice (scriere) se face cu următoarea secvență de program în limbaj de asamblare:

```
MOV      A, #xxxx00xxB
MOV      P2, #1
MOV      R0, #40H
MOVX     @R0, A
```

5.3.2.3 INTERFAȚA CU OPERATORUL

- tastatură matriceală locală cu 16 taste, cu următoarea configurație:

	Col. 1	Col. 2	Col.3	Col. 4
Lin. 1	F1	7	8	9
Lin. 2	F2	4	5	6
Lin. 3	F3	1	2	3
Lin. 4	F4	C	0	E

Citirea tastaturii implică:

1. poziționarea succesivă în 0 a biților corespunzători liniilor prin programarea corespunzătoare a portului de ieșiri logice (coloanele sunt ținute în 1 prin rezistențe de pull-up);
2. citirea portului de intrare tastatură;
3. interpretarea rezultatului.

Port ieșiri logice. Adresa: 140H

Bit 7 (MSB)				Bit 0 (LSB)			
Linie 4	Linie 3	Linie 2	Linie 1	Out_Log_1	Out_Log_0	Send_1	Send_0
Linii tastatură matriceală							

Manipularea portului de ieșiri digitale (scriere) se face cu următoarea secvență de program în limbaj de asamblare:

```
MOV  A,#1110xxxxB
MOV  P2,#1
MOV  R0,#40H
MOVX @R0,A
```

Port intrare tastatură. Adresa: 120H

Bit 7 (MSB)				Bit 0 (LSB)			
Linie 4	Linie 3	Linie 2	Linie 1	Col 4	Col 3	Col 2	Col 1
Linii tastatură matriceală				Coloane tastatură matriceală			

Manipularea portului de intrare tastatură (citire) se face cu următoarea secvență de program în limbaj de asamblare:

```
MOV  P2,#1
```

```
MOV    R0, #20H
MOVX   A, @R0
```

- display alfanumeric tip LCD, cu 2 linii x 16 caractere. Sistemul dispune de un display alfanumeric cu cristale lichide cu următoarele caracteristici:
 1. memorie internă de date pentru 80 de caractere, (Display Data RAM);
 2. generator de caractere, 160 caractere, 5x7 puncte (Character Generator ROM);
 3. generator de caractere programabil, 8 caractere, 5x7 puncte (Character Generator ROM);
 4. Display Data RAM și Character generator ROM adresabile de microcontroller;
 5. comenzi: Clear Display, Cursor Home, Display ON/OFF, Cursor ON/OFF, Blink Character, Cursor Shift, Display Shift;
 6. circuitul este resetat la punerea sub tensiune;
 7. oscilator incorporat.

Unitatea de afișare recepționează codul caracterului transmis de microcontroller, memorează codul în memoria internă de date (Display data RAM), convertește codul caracterului într-un pattern asociat (o matrice 5x7 puncte) și afișează caracterul.

Unitatea de afișare dispune de asemenea de un generator de caractere programabil (Character Generator ROM) care permite definirea de către utilizator a 8 caractere speciale.

Pentru afișarea unui caracter microcontroller-ul transmite pe magistrala de date comenzile de poziționare care sunt înscrise în Registrul de Instrucțiuni. Codul caracterului (ASCII) este transmis apoi tot pe magistrala de date și înscris în Registrul de Date. Starea unității de afișare este citită din Registrul de Stare. Ca urmare, este afișat caracterul corespondent codului transmis în poziția specificată.

Unitatea de afișare incrementează / decrementează automat poziția caracterului afișat după fiecare intrare astfel încât este necesară numai transmiterea succesivă a codului pentru afișarea unui șir de caractere.

Instrucțiunile de control a deplasării cursorului permit introducerea caracterelor de la stânga la dreapta sau de la dreapta la stânga.

Unitatea de afișare este coordonată de microcontroller prin intermediul Registrului de Instrucțiuni, Registrului de Stare și a Registrului de Date.

Registrul de Instrucțiuni. Adresa: 100H. Display Clear

DB7 (MSB)	DB6	DB5	DB4	DB3	DB2	DB1	DB0(LSB)
0	0	0	0	0	0	0	1

Clear enter display area, Restore display from shift, Load address counter with DD RAM address 00H. Execution Time: max. 1,64 ms.

Registrul de Instrucțiuni. Adresa: 100H. Display /Cursor Home

DB7 (MSB)	DB6	DB5	DB4	DB3	DB2	DB1	DB0(LSB)
0	0	0	0	0	0	1	X

Restore display from shift, Load address counter with DD RAM address 00H. Execution Time: max. 1,64ms.

Registrul de Instrucțiuni. Adresa: 100H. Entry Mode Set

DB7 (MSB)	DB6	DB5	DB4	DB3	DB2	DB1	DB0(LSB)
0	0	0	0	0	1	I/D	S

Specify cursor advance direction and display shift mode. This operation take place after each data entry. I/D=1, Increment / I/D=0, Decrement / S=1, Display Shift On. Execution Time: max. 40μs.

Registrul de Instrucțiuni. Adresa: 100H. Display ON / OFF

DB7 (MSB)	DB6	DB5	DB4	DB3	DB2	DB1	DB0(LSB)
0	0	0	0	1	D	C	B

Specify and activation of display (D), cursor (C), and blinking of character at cursor position. Execution Time: max. 40μs.

Registrul de Instrucțiuni. Adresa: 100H. Display / Cursor Shift

DB7 (MSB)	DB6	DB5	DB4	DB3	DB2	DB1	DB0(LSB)
0	0	0	1	S/C	R/L	X	X

Shift display or move cursor. S/C=1, Shift Display /S/C=0, Move Cursor. Execution Time: max. 40μs.

Registrul de Instrucțiuni. Adresa: 100H. Function Set

DB7 (MSB)	DB6	DB5	DB4	DB3	DB2	DB1	DB0(LSB)
0	0	1	DL	N	0	X	X

Set interface data length (DL) and numbers of display lines (N). DL=1, 8 bits / N=1, Dual Lines. Execution Time: max. 40μs.

Registrul de Instrucțiuni. Adresa: 100H. CG RAM Address Set

DB7 (MSB)	DB6	DB5	DB4	DB3	DB2	DB1	DB0(LSB)
0	1	Acg: Caracter Generator RAM Address					

Load the Address Counter with CG RAM Address. Subsequent data is CG RAM data. Execution Time: max. 40μs.

Registrul de Instrucțiuni. Adresa: 100H. DD RAM Address Set

DB7 (MSB)	DB6	DB5	DB4	DB3	DB2	DB1	DB0(LSB)
1	Add: Display Data RAM Address						

Load the Address Counter with DD RAM Address. Subsequent data is DD RAM data. Execution Time: max. 40μs.

Registrul de Stare. Adresa: 102H. Busy Flag / Address Counter Read

DB7 (MSB)	DB6	DB5	DB4	DB3	DB2	DB1	DB0(LSB)
BF	AC: Address Counter						

Read busy flag (BF) and contents of address counter. Execution Time: max. 0μs.

Registrul de Date (Write). Adresa: 101H. CG RAM / DD RAM Data Write

DB7 (MSB)	DB6	DB5	DB4	DB3	DB2	DB1	DB0(LSB)
Write Data							

Write data to CG RAM or DD RAM. Execution Time: max. 40μs.

Registrul de Date (Read). Adresa: **103H. CG RAM / DD RAM Data Read**

DB7 (MSB)	DB6	DB5	DB4	DB3	DB2	DB1	DB0(LSB)
Read Data							

Read data from CG RAM or DD RAM. Execution Time: max. 40μs.

5.3.2.4 INTERFAȚA CU UN SISTEM DE CALCUL

- Comunicație serială, standard RS232, full-duplex cu rata de transfer programabilă.

5.3.3 RESURSE SOFTWARE. UTILIZARE

Resursele software disponibile:

- sub sistemul de operare MS-DOS:
 - *asamblor*, A51.EXE și variante (documentație);
 - *compiler C*, C51.EXE;
 - *link-er*, L51.EXE;
 - *program conversie*, OHS51.EXE;
 - *program monitor*, MT.EXE;
- sub sistemul de operare WINDOWS:
 - *Franklin Compiler 1997* (include toate facilitățile anterioare, exclusiv *program monitor*).

Faza de dezvoltare a unui program de aplicație pentru o platformă hardware coordonată de o unitate centrală de prelucrare cu microcontroller 80C552 constă din:

- **Scrierea programului** cu ajutorul unui editor ASCII de texte, ca de exemplu **EDIT.COM** din sistemul de operare **MS-DOS**, **NCEDIT.EXE** din programul **NORTON COMMANDER** sau **NOTEPAD.EXE** din sistemul de operare **WINDOWS** (se impune utilizarea unui editor ASCII pentru a nu se introduce caractere speciale de control în sursa programului).

În sursa programului de aplicație sunt declarate explicit resursele suplimentare ale microcontroller-ului 80C552 în raport cu 8051, într-o secțiune declarativă la început.

Programul sursă de aplicație poate fi scris în limbajul de asamblare al familiei de microcontroller-e 8051, caz în care numele său va fi **PROGRAM.ASM** sau într-o variantă specializată a limbajului C pentru familia de microcontroller-e 8051, caz în care numele său va fi

PROGRAM.C.

- **Asamblarea sursei programului** de aplicație, scris în limbaj de asamblare, cu ajutorul asamblorului **A51.EXE**, specializat pentru familia de microcontroller-e 8051, folosind sintaxa: **A51.EXE PROGRAM.ASM**

În urma executării asamblării programului sursă de aplicații, asamblorul **A51.EXE** generează două fișiere:

- **PROGRAM.LST**, un fișier listă ce este destinat localizării eventualelor erori rezultate în urma procesului de asamblare și
- **PROGRAM.OBJ**, un fișier de tip obiect, care va fi folosit în continuare pentru obținerea formatului executabil;
- *sau*, **Compilarea sursei** programului de aplicație, scris în limbaj C, cu ajutorul asamblorului **C51.EXE**, specializat pentru familia de microcontroller-e 8051, folosind sintaxa: **C51.EXE PROGRAM.C**
- **Obținerea formatului executabil**, de tip **INTEL HEX**, pe baza fișierului de tip **OBJ**, folosind programul **OHS51.EXE** cu sintaxa:

OHS51.EXE PROGRAM.OBJ

sau

OHS51.EXE PROGRAM

Este generat fișierul executabil **PROGRAM.HEX** ce constituie programul propriu-zis de aplicație.

Ca orice fișier în format **HEX**, acesta începe cu un *header* ce conține numărul de octeți ai programului, adresa la care este organizat programul (specificată prin directiva **ORG** în prima linie), urmat de corpul programului și se încheie cu o sumă de control.

- **Rularea programului PROGRAM.HEX** presupune rularea pe un calculator gazdă, de tip PC, a programului monitor al unității centrale cu microcontroller 80C552, denumit **MT.EXE**, care are în primul rând rolul de a stabili comunicația serială între sistemul cu microcontroller și calculatorul de tip PC.

Sintaxa este: **MT.EXE X**

în care **X**=1, 2, 3, 4 specifică indicativul portului serial utilizat pentru comunicație (1 specifică portul serial COM1, 2 specifică portul serial COM2, 3 specifică portul serial COM3, 4 specifică portul serial COM4). Dacă parametrul **X** lipsește, este utilizat în mod implicit portul serial COM1.

Stabilirea comunicației seriale între cele două sisteme (pe viteza de 9600 bauds, folosind cuvinte de date cu lungimea de 8 biți, fără paritate, cu un bit de STOP și protocol XON-XOFF) este indicată prin apariția pe *display*-ul sistemului de calcul a unui mesaj, urmat de prompter-ul de monitor (caracterul #).

- **Transferarea programului de aplicație.** În acest stadiu poate fi transferat către sistemul cu microcontroller programul de aplicație, **PROGRAM.HEX**, folosind comanda de *up-load* **F2** și indicând numele programului de aplicație.

Urmează procesul de *up-load*, indicat prin apariția pe *display*-ul calculatorului a unei succesiuni de linii, de lungime constantă cu excepția ultimei, ce încep cu ”#:_”, și sunt alcătuite din câte 16 octeți în format *hexa* (date) plus un octet sumă de control.

La terminarea transferului de date pe legătura serială, pe *display*-ul calculatorului apare prompter-ul de monitor (caracterul #).

- **Lansarea în execuție a programului** se execută cu comanda de monitor **GO ADR**, în care **ADR** reprezintă adresa la care este organizat programul în memoria de date a microcontroller-ului, specificată în prima linie a acestuia prin directiva **ORG**.

Sintaxa este următoarea: **G 8000**

5.3.4 RUTINE DE BAZĂ PENTRU MANIPULAREA RESURSELOR SISTEMULUI DE DEZVOLTARE

```

;*****
;*  PROGRAM CONVERSIE A / D 80C552 CU MONITOR 80X51 PE UN CANAL
;*  DE INTRARE SELECTABIL SI AFISARE IN HEXA, PE TREI CIFRE,
;*  A VALORII TENSIUNII DE INTRARE CONVERTITA PE PRIMII 8 BITI
;*****
; Programul citește de la tastatura locala una dintre tastele
; apasate. Corespunzator codului tastei se selectează unul dintre
; cele 8 canale analogice de intrare (daca codul tastei este 0..7),
; se convertește ; tensiunea aplicata intrarii selectate
; (ADC0 ... ADC8) si se afiseaza rezultatele conversiilor, sub forma
; hexa, pe trei cifre.
; Registrul ADCON contine fanioanele de control a conversiei, precum
; si cei mai putin semnificativi doi biti ai conversiei, astfel:
;   ADCON.7 = bitul 1 al conversiei
;   ADCON.6 = bitul 0 al conversiei
;   ADCON.5 = 0 (start conversie numai soft)
;   ADCON.4 = 0
;   ADCON.3 = 1 (start conversie)
;   ADCON.2, ADCON.1, ADCON.0 - specifica canalul analogic de
; intrare
;   ADCH - registru cu primii 8 biti semnificativi ai conversiei,
; dupa ce bitul ADCON.4=1.
; Conform documentatiei microcontrollerului 80C552, pentru
; registrele ADCON si ADCH s-a atribuit spatiul din memoria RAM
; interna de la adresele 0C5H, respectiv 0C6H.
; Spatiul utilizator incepe de la adresa 8000H, deci acest program
; va fi stocat incepand de la aceasta adresa.
      ORG      8000H

```

PREZENTAREA MICROCONTROLLERULUI 80C552 (PHILIPS)

```
; declararea adreselor RAM ale resurselor suplimentare
        ADCON          equ    0C5H
        ADCH           equ    0C6H
;
; program principal
        LJMP    INIT          ; long jump la rutina de initializare
                                ; afisaj cu cristale lichide LCD
INIT:    ACALL    INILCD      ; apel rutina de initializare afisaj
        ACALL    CLRLCD      ; apel rutina stergere LCD
        ACALL    KEYBRD      ; citire tastatura
ACH:     CLR      A           ; initializeaza acumulatorul cu 0
        MOV      ADCON,A      ; initializeaza registrul de comanda A/D
        MOV      A,R1         ; se pune in acumulator codul canalului
                                ; de intrare selectat de la tastatura
        ORL      A,#08H       ; bitii de control
        MOV      ADCON,A      ; start conversie prin software
WAIT:    MOV      A,ADCON      ; A:=registrul de stare al A/D
        JNB      ACC.4,WAIT    ; asteapta terminarea conversiei
        MOV      A,ADCH       ; citeste rezultatul conversiei
        MOV      A,ADCH       ; se restaureaza rezultatul conversiei
        MOV      B,#33H       ; impartire cu 51=255/5
        DIV      AB           ; obtine cifra unitatilor
        PUSH     ACC           ; salveaza cifra unitatilor in stiva
        MOV      A,B          ; reface restul 1
        MOV      B,#10        ; B:=10
        MUL      AB           ; inmultire cu 10
        MOV      R7,A         ; stocheaza octet inferior rezultat
        MOV      A,B          ; restaureaza octet superior rezultat
        ANL      A,#01H       ; test depasire
        JNB      ACC.0,LABEL1 ; salt continuare daca nu e depasire
        ADD      A,#4          ; calculul partii semnificative a catului
LABEL1:   CLR      C           ; anuleaza CARRY
        MOV      R4,A         ; stocheaza partea semnificativa a catului
        MOV      B,#33H       ; impartire cu 51=255/5
        MOV      A,R7         ; restaureaza octet inferior rezultat
        DIV      AB           ; calculeaza cat 1 partial
        ADD      A,R4         ; calculeaza prima zecimala
        PUSH     ACC           ; salveaza prima zecimala in stiva
        MOV      A,B          ; reface restul 2
        MOV      B,#10        ; B:=10
        MUL      AB           ; inmultire cu 10
        MOV      R7,A         ; stocheaza octet inferior rezultat
        MOV      A,B          ; restaureaza octet superior rezultat
        ANL      A,#01H       ; test depasire
        JNB      ACC.0,LABEL2 ; salt continuare daca nu e depasire
        ADD      A,#4          ; calculul partii semnificative a catului
LABEL2:   CLR      C           ; anuleaza CARRY
        MOV      R4,A         ; stocheaza partea semnificativa a catului
        MOV      B,#33H       ; impartire cu 51=255/5
        MOV      A,R7         ; restaureaza octet inferior rezultat
        DIV      AB           ; calculul cat 2 partial
        ADD      A,R4         ; calcul a doua zecimala
        PUSH     ACC           ; salveaza a doua zecimala in stiva
        MOV      A,R1         ; numarul canalului selectat
        ACALL    HEXASC       ; converteste in ASCII
```

```

MOV     R4,A           ; stocheaza cod numar canal
MOV     R2,#0C7H       ; adresa pentru numar canal
ACALL   WRCMD          ; pozitionare cursor
MOV     A,R4           ; restaureaza cod numar canal
MOV     R2,A           ; R2 contine codul de afisat
ACALL   TRX            ; scrie la LCD numar canal
POP     ACC            ; a doua zecimala
ACALL   HEXASC         ; convertire cod ASCII
MOV     R4,A           ; stocheaza cod ASCII a doua zecimala
MOV     R2,#0CDH       ; adresa pentru a doua zecimala
ACALL   WRCMD          ; pozitionare cursor
MOV     A,R4           ; restaureaza cod a doua zecimala
MOV     R2,A           ; R2 contine codul de afisat
ACALL   TRX            ; scrie la LCD a doua zecimala
POP     ACC            ; reface prima zecimala
ACALL   HEXASC         ; converteste in cod ASCII
MOV     R4,A           ; stocheaza codul primei zecimale
MOV     R2,#0CCH       ; adresa primei zecimale
ACALL   WRCMD          ; pozitionare cursor
MOV     A,R4           ; restaureaza codul primei zecimale
MOV     R2,A           ; R2 contine codul primei zecimale
ACALL   TRX            ; scrie la LCD prima zecimala
POP     ACC            ; reface cifra unitati
ACALL   HEXASC         ; converteste in cod ASCII
MOV     R4,A           ; stocheaza codul unitatilor
MOV     R2,#0CAH       ; adresa scriere unitati
ACALL   WRCMD          ; pozitionare cursor
MOV     A,R4           ; restaureaza codul unitatilor
MOV     R2,A           ; R2 contine codul unitatilor
ACALL   TRX            ; scriere la LCD unitati
AJMP    ACH            ; proces ciclic de conversie
HEXASC:                ; rutina de conversie hexa-ascii
    ANL     A,#0FH      ; se mascheaza cei patru biti mai
                        ; semnificativi ai acumulatorului
    JNB     ACC.3,NOADJ  ; daca bitul 3 = 0 salt la NOADJ
    JB      ACC.2,ADJ    ; daca bitul 2 = 1 salt la ADJ
    JNB     ACC.1,NOADJ  ; daca bitul 1 = 0 salt la NOADJ
ADJ:      ADD     A,#07H  ; aduna acumulatorul cu #07H
NOADJ:    ADD     A,#30H  ; aduna acumulatorul cu #30H
    RET                     ; revenire din rutina HEXASC
    END                  ; sfarsitul programului

;*****
;*      PROGRAM DE TESTARE A IESIRILOR MODULATE IN DURATA
;*****
; Se genereaza la canalul 0 (pinul 4 al 80C552), respectiv la
; canalul 1 (pinul 5 al 80C552)semnale cu ; frecventa specificata de
; continutul registrului PWMP si cu factorul de umplere specificat
; de registrul ADCH, ceea ce inseamna ca, prin integrarea iesirii
; canalelor, se poate implementa un convertor D/A dual.
; Definim registrele ce memoreaza parametri semnalelor generate ca
; fiind locatii din memoria RAM interna a 80C552, avand adresele
; specificate in catalog.
; Factorul de umplere nu este acelasi pentru ambele calale de
; iesire; factorul de umplere poate varia intre 0 si 100%, prin

```

```
; modificarea continutului registrelor PWM0, PWM1.
; Frecventa semnalului de iesire, aceiasi pentru ambele canale, este
; precizata prin intermediul continutului registrului PWMP, fiind
; data de formula:  $f = f_{osc} / (2 * (1 + PWMP) * 255)$ , in care fosc reprezinta
; frecventa ceasului microcontrollerului ( $f_{osc} = 11,059\text{MHz}$ ).
; Semnalul este generat continuu la iesirile 4 si 5, deoarece
; parametrii au fost memorati in registrele PWMP si PWM0, PWM1.
; Spatiul utilizator incepe de la adresa 8000H, deci acest program
; va fi stocat incepand de la aceasta adresa.
```

```
    ORG    8000H
; declararea adreselor RAM ale resurselor suplimentare
    PWMP   equ    0FEH
    PWM0   equ    0FCH
    PWM1   equ    0FDH
    MOV    A, #20 ; se incarca acumulatorul cu #20 pentru
                  ; frecventa semnalului de iesire sa
                  ; fie  $1\text{kHz} = 11,059\text{MHz} / (2 * (1 + 20) * 255)$ 
    MOV    PWMP, A ; se programeaza registrul PWMP
    MOV    A, #55 ; incarca in acumulator valoarea imediata 55H
    CPL    A      ; se complementeaza acumulatorul
                  ; deoarece iesirea PWM0 este negata
    MOV    PWM0, A ; reconstituire semnal analogic
    MOV    A, #0AA ; incarca in acumulator valoarea imediata AAH
    CPL    A      ; se complementeaza acumulatorul
                  ; deoarece iesirea PWM1 este negata
    MOV    PWM1, A ; reconstituire semnal analogic
    END      ; sfarsitul programului
```

```
; *****
; *      PROGRAM DEMONSTRATIV DE TESTARE SI PROGRAMARE
; *      A UNUI AFISAJ CU CRISTALE LICHIDE CU DOUA LINII DE CATE
; *      16 CARACTERE FIECARE, CUPLAT LA UN SISTEM DE DEZVOL-
; *      TARE CU MICROCONTROLLER 80C552, CU MONITOR 80C51.
; *****
```

```
; Dialogul cu afisajul LCD se efectueaza prin intermediul unor
; subrutine:
;   WRCMD - scrierea comenzii din registrul R2 la LCD
;   WRDAT - scrierea datelor din registrul R2 la LCD
;   WLCD - asteapta terminarea operatiunilor interne ale LCD
;   INLCD - initializarea LCD pentru transfer pe 8 biti.
; Comenzile folosite sunt:
;   CLEAR - 1
;   CURSOR HOME - 2, 3
;   MODE - 1/INC/SHIFT
;   CONTROL - 1/DisplayON/CursorON/FlashON
;   SHIFT - 1/SHIFT/RIGHT/*/*
; Spatiul utilizator incepe de la adresa 8000H, deci acest program
; va fi stocat incepand de la aceasta adresa.
```

```
    ORG    8000H
; program principal
;
;   LJMP    INIT      ; long jump la rutina de initializare
;                   ; afisaj cu cristale lichide LCD
INIT: ACALL  INILCD    ; initializare afisaj
      ACALL  CLRLCD    ; apel rutina stergere LCD
```

```

ACALL SEC1          ; rutina de intarziere cu o secunda
MOV DPTR,#TEXT1     ; se memoreaza in DPTR adresa de in-
                    ; ceput a sirului de caractere TEXT1
ACALL MESAJ         ; apel rutina de afisare mesaj
ACALL CLEAR         ; apel rutina stergere temporizata
SJMP INIT          ; program care se executa ciclic
INILCD:             ; rutina de initializare LCD.
; Aceasta rutina trimite la LCD comanda #38H = 0011 1000B. Conform
; datelor de catalog, acesta comanda seteaza urmatoorii parametri ai
; afisajului:
;   - bitul 5 - defineste aceasta comanda (function set);
;   - bitul 4(DL-Data Length)=1 seteaza dialogul pe 8 biti
;     intre procesor si LCD;
;   - bitul 3(N-Number of display lines)=1 seteaza numarul
;     liniilor afisajului ca fiind 2^N=2;
;   - bitul 2(F-Character Font)=0 seteaza forma caracterului
;     ca fiind 5*7 puncte
MOV R2,#38H         ; incarca R2 cu cuvantul de comanda
                    ; pregatind dialogul pe 8 biti cu
                    ; driverele de afisaj
ACALL WRCMD         ; transmite comanda anterioara la
                    ; port, efectuand programarea LCD
MOV R4,#50          ; o intarziere
DEL4MS:             ; intarziere cu 4 ms, necesara functio-
                    ; narii display-ului LCD
ACALL DELAY         ; absolute call rutina DELAY
DJNZ R4,DEL4MS      ; se decrementeaza registrul R4 si
                    ; salt la eticheta DEL4MS daca conti-
                    ; nutul acestuia este nenul
MOV R4,#4           ; se incarca R4 cu valoarea imediata 4
LINI:               ; LINI seteaza modul de lucru al
                    ; afisajului
                    ; cu cristale lichide LCD
ACALL WRCMD         ; se scrie comanda la driverele LCD
ACALL DELAY         ; apel rutina DELAY
DJNZ R4,LINI        ; se decrementeaza registrul R4 si salt
                    ; la eticheta LINI daca este nenul con-
                    ; tinutul acestui registru
ACALL WLCD          ; testarea starii driverelor LCD
MOV R2,#6           ; seteaza modul de lucru "entry"
ACALL WRCMD         ; transmite comanda la LCD
ACALL WLCD          ; testarea starii driverelor LCD
MOV R2,#0EH         ; seteaza modul "display on"
ACALL WRCMD         ; transmite comanda la LCD
ACALL WLCD          ; testarea starii driverelor LCD
MOV R2,#1           ; stergere LCD
ACALL WRCMD         ; transmite comanda la LCD
ACALL WLCD          ; testarea starii driverelor LCD
RET                 ; revenire din subrutina INILCD
WRCMD:              ; rutina WRCMD scrie o comanda la
; driverele de afisaj, in urma verificarii starii acestora. Se
; selecteaza afisajul LCD si comanda continuta de registrul R2 este
; transferata driverelor din LCD
ACALL WLCD          ; se verifica starea driverelor (BF=0)
MOV A,R2            ; comanda continuta in registrul R2 e

```

```

; transferata in acumulator
MOV    P2,#1      ; se selecteaza portul LCD (S0=1)
MOV    R0,#0      ; se selecteaza pe magistrala de adrese
; cu A0 si A1 combinatia 0000 0000 B,
; corespunzatoare scrierii unei comenzi
; la driverule afisajului LCD
MOVX   @R0,A      ; se scrie in driverule afisajului LCD
; (la adresa stabilita anterior) comanda
RET     ; revenire din rutina WRCMD
WLCD:   ; rutina ce testeaza daca este sau
; nu posibila transmiterea unui nou caracter catre LCD, pe baza in
; formatiilor furnizate de rutina RDCMD. Se testeaza bitul 7 (BF -
; Busy Flag) al registrului de stare al LCD.
; Daca BF=1, inseamna ca LCD efectueaza o operatie interna si nu
; poate accepta un nou caracter.
    ACALL RDCMD   ; apel rutina RDCMD
    JB     ACC.7,WLCD ; se verifica daca BF=1 si asteapta
; trecerea lui in 0 (terminarea ope-
; ratiunilor interne)
    RET          ; revenire din subrutina WRCL
RDCMD:   ; subrutina RDCMD selecteaza afisajul
; si citeste din registru de stare starea curenta a driverelor
; acestuia, informatie care este depusa in acumulator
    MOV    P2,#1      ; selectie port (S0=1)
    MOV    R0,#2      ; se realizeaza pe magistrala de adrese
; selectia cu A0 si A1 (0000 0010 B),
; pentru a se citi starea driverelor de
; afisaj LCD
    MOVX   A,@R0      ; se transfera in acumulator valoarea
; gasita la adresa specificata anterior
    RET          ; revenire din rutina RDCMD
DELAY:   ; subrutina DELAY realizeaza o intarziere
; cu 80 microsecunde
    MOV    R3,#17     ; registrul R3 este incarcat cu valoarea
; imediata 17
; 80E-6secunde = 17 * 2instructiuni NOP *
; * 12 perioade de ceas/instructiune NOP*
; 1/11.059MHz (perioada ceas procesor)
LL1:    NOP           ; no operation
    NOP           ; no operation
    DJNZ   R3,LL1     ; se decrementeaza registrul R3, se compa-
; ra continutul sau cu 0. Daca rezultatul
; compararii este 1, salt la eticheta LL1
    RET          ; revenire din subrutina DELAY
CLRLCD:  ; subrutina de stergere a afisajului
; cu cristale lichide si setare a modului de afisare
    MOV    R2,#1      ; stergere afisaj LCD
    ACALL  WRCMD      ; transmite comanda la LCD
    MOV    R2,#1100B  ; 1, display on, cursor off, flash off
    ACALL  WRCMD      ; transmite comanda la LCD
    RET          ; revenire din rutina de CLRLCD
TRX1:    ; subrutina de scriere sir de caractere
; la afisajul cu cristale lichide LCD
    CLR    A          ; initializeaza acumulatorul cu 0
    MOVC   A,@A+DPTR  ; muta in acumulator adresa gasita

```

```

; la adresa @A+DPTR
CJNE  A,#24H,TRCAR1; compara continutul acumulatorului
; cu #24H (codul hexa al caracterului
; $ - sfarsit de mesaj) si daca este
; diferit se efectueaza salt la eticheta
; TRCAR1
RET   ; iesire din subrutina TRX1
TRCAR1: MOV  R2,A ; muta continutul acumulatorului in
; registrul R2
ACALL WRDAT ; apel subrutina de scriere date in
; registrele afisajului LCD
INC  DPTR ; adresa urmatorului caracter de scris
SJMP TRX1 ; small jump la eticheta TRX1, pentru
; a scrie urmatorul caracter
TRX: ; subrutina de afisare caracter la LCD
CLR  A ; initializeaza acumulatorul cu 0
MOV  A,R2 ; muta in acumulator continutul regis-
; trului R2
LJMP TRCAR ; long jump la eticheta TRCAR
LL7: RET ; revenire din subrutina TRX
TRCAR: MOV  R2,A ; muta in registrul R2 continutul Acc.
ACALL WRDAT ; scrie in driverele de date ale LCD
; codul caracterului de afisat
SJMP LL7 ; small jump la LL7: RET
WRDAT: ; subrutina WRDAT este identica cu
; WRCMD dar este utilizata pentru scrierea la driverele LCD codul
; caracterelor ce urmeaza a fi afisate
ACALL WLCD ; testarea starii driverelor LCD
MOV  P2,#1 ; se selecteaza portul LCD (S0=1)
MOV  R0,#1 ; se selecteaza pe magistrala de adrese
; cu A0 si A1 combinatia 0000 0001 B,
; corespunzatoare scrierii unui caracter
; la driverele afisajului LCD
MOV  A,R2 ; se transfera caracterul in acumulator
MOVX @R0,A ; se transmite caracterul driverelor LCD
RET ; revenire din rutina WRDAT
MESAJ: ; subrutina de afisare mesaj, continand
; doua linii de cate 16 caractere la afisaj cu cristale lichide
ACALL TRX1 ; apel rutina de transmisie mesaj
MOV  R2,#0C0H ; pozitionare pe a doua linie
ACALL WRCMD ; scriere comanda pozitionare la LCD
INC  DPTR ; incrementare DPTR (linia a doua)
ACALL TRX1 ; apel rutina de transmisie mesaj
RET ; revenire din subrutina MESAJ
CLEAR: ; subrutina de stergere si setare a
; modului de lucru a afisajului cu cristale lichide LCD
ACALL SEC1 ; mentine starea afisajului 1 secunda
MOV  R2,#1 ; comanda stergere display
ACALL WRCMD ; scriere comanda la LCD
MOV  R2,#2 ; cursor home pe prima linie
ACALL WRCMD ; scrie comanda la LCD
RET ; revenire din subrutina CLEAR
SEC1: ; rutina de intarziere cu 0,5 secunde
; 0,5sec = (8instr.NOP x 12perioade ceas/NOP x 256 x 256)/11,059MHz
MOV  R6,#255 ; se incarca registrul R6 cu #255

```

PREZENTAREA MICROCONTROLLERULUI 80C552 (PHILIPS)

```
LOOP1: MOV     R7,#255          ; se incarca registrul R7 cu #255
LOOP:  NOP                      ; 16 instructiuni NOP
      NOP
      NOP
      NOP
      NOP
      NOP
      NOP
      DJNZ     R7,LOOP          ; se decrementeaza registrul R7 si
                                ; daca este nenul, salt la LOOP
      DJNZ     R6,LOOP1         ; se decrementeaza registrul R6 si
                                ; daca este nenul, salt la LOOP1
      RET                      ; revenire din subrutina SEC3
TEXT1: DB 'TESTARE DISPLAY$'    ; text prezentare 1, linia 1
      DB 'LCD 2lin.*16car$'     ; text prezentare 1, linia 2
      END                      ; sfarsitul programului

;*****
;*      PROGRAM DE CITIRE A UNEI TASTATURI, CU TREI LINII SI
;*      PATRU COLOANE ( 12 TASTE FUNCTIONALE ), REALIZATA
;*      PRIN INTERMEDIUL UNUI PORT DE IESIRE SI A PORTULUI DE
;*      INTRARE AL MICROCONTROLLERULUI 80C552, CU MONITOR
;*      PENTRU FAMILIA INTEL 80C51
;*****
; program citire in bucla a unei tastaturi de 3 linii * 4 coloane,
; pana la apasarea unei taste.
; Rezultatul citirii este convertit in hexa si memorat in registrul
; R1, urmand a fi afisat pe LCD.
; Spatiul utilizator incepe de la adresa 8000H, deci acest program
; va fi stocat incepand de la aceasta adresa.
      ORG      8000H
; progam principal de citire in bucla a tastaturii
; citirea tastaturii, prin baleierea celor trei linii, citirea celor
; patru coloane si verificare daca s-a tastat ceva.
; prima linie
NEXT:  MOV     A,#0E0H          ; initializeaza acumulatorul cu #0E0H
                                ; (se selecteaza prima linie de taste)
      MOV     P2,#1            ; selecteaza decodificatorul (A8=1)
      MOV     R0,#20H          ; incarca R0 cu 0010 0000 B:
                                ; selectie port de iesire 1 (S1=0)
      MOVX    @R0,A            ; muta continutul acumulatorului
                                ; la portul de iesire 1 (AX0=0)
      MOV     P2,#1            ; selecteaza decodificatorul (A8=1)
      MOV     R0,#60H          ; incarca R0 cu 0110 0000B:
                                ; selecteaza portul de intrare
      MOVX    A,@R0            ; citeste in acumulator valoarea gasita
                                ; la adresa specificata anterior
      CPL     A                ; complementeaza acumulatorul
      MOV     R3,A             ; salveaza valoarea in registrul R3
      ANL     A,#0FH           ; mascheaza bitii mai semnificativi
                                ; ai acumulatorului (codul liniei)
      JZ      NEXT1            ; daca A=0 (valoarea citita este 0FH-
                                ; nu s-a tastat nimic),salt la eticheta
                                ; NEXT1 (linia 1 de taste)
      JNZ     COL              ; daca continutul acumulatorului nu
```



```

; este zero, salt la determinarea
; codului HEXA al tastei, in functie
; de linia activa si coloana pe care
; este situata tasta apasata

; a doua linie
NEXT1: MOV    A,#0D0H    ; se initializeaza acumulatorul cu #0D0H
                        ; se selecteaza a doua linie de taste)
      MOV     P2,#1      ; selecteaza decodificatorul (A8=1)
      MOV     R0,#20H    ; incarca R0 cu 0010 0000 B:
                        ; selectie port de iesire 1 (S1=0)
      MOVX    @R0,A      ; muta continutul acumulatorului
                        ; la portul de iesire 1 (AX0=0)
      MOV     P2,#1      ; selecteaza decodificatorul (A8=1)
      MOV     R0,#60H    ; incarca R0 cu 0110 0000B:
                        ; selecteaza portul de intrare
      MOVX    A,@R0      ; citeste in acumulator valoarea gasita
                        ; la adresa specificata anterior
      CPL     A          ; complementeaza acumulatorul
      MOV     R3,A       ; salveaza valoarea in registrul R3
      ANL     A,#0FH     ; mascheaza bitii mai semnificativi
                        ; ai acumulatorului (codul liniei)
      JZ      NEXT2      ; daca A=0 (valoarea citita este 0FH)
                        ; salt la eticheta NEXT
      JNZ     COL        ; daca continutul acumulatorului nu
                        ; este zero, salt la determinarea
                        ; codului HEXA al tastei, in functie
                        ; de linia activa si coloana pe care
                        ; este situata tasta apasata

; a treia linie
NEXT2: MOV     A,#0B0H    ; se initializeaza acumulatorul cu #0D0H
                        ; se selecteaza a treia linie de taste)
      MOV     P2,#1      ; selecteaza decodificatorul (A8=1)
      MOV     R0,#20H    ; incarca R0 cu 0010 0000 B:
                        ; selectie port de iesire 1 (S1=0)
      MOVX    @R0,A      ; muta continutul acumulatorului
                        ; la portul de iesire 1 (AX0=0)
      MOV     P2,#1      ; selecteaza decodificatorul (A8=1)
      MOV     R0,#60H    ; incarca R0 cu 0110 0000B:
                        ; selecteaza portul de intrare
      MOVX    A,@R0      ; citeste in acumulator valoarea gasita
                        ; la adresa specificata anterior
      CPL     A          ; complementeaza acumulatorul
      MOV     R3,A       ; salveaza valoarea in registrul R3
      ANL     A,#0FH     ; mascheaza bitii mai semnificativi
                        ; ai acumulatorului (codul liniei)
      JZ      NEXT      ; daca A=0 (valoarea citita este 0FH)
                        ; salt la eticheta NEXT (prima linie)
      JNZ     COL        ; daca continutul acumulatorului nu
                        ; este zero, salt la determinarea
                        ; codului HEXA al tastei, in functie
                        ; de linia activa si coloana pe care
                        ; este situata tasta apasata
COL:   JB      ACC.0,COL0 ; salt la COL0, daca bitul 0 al acu-
                        ; mulatorului este 1 (s-a apasat o
                        ; tasta de pe coloana 0)

```

PREZENTAREA MICROCONTROLLERULUI 80C552 (PHILIPS)

```

JB      ACC.1,COL1      ; salt la COL1, daca bitul 1 al acu-
                        ; mulatorului este 1 (s-a apasat o
                        ; tasta de pe coloana 1)
JB      ACC.2,COL2      ; salt la COL2, daca bitul 2 al acu-
                        ; mulatorului este 1 (s-a apasat o
                        ; tasta de pe coloana 2)
JB      ACC.3,COL3      ; salt la COL3, daca bitul 3 al acu-
                        ; mulatorului este 1 (s-a apasat o
                        ; tasta de pe coloana 3)
COL0:   MOV      R1,#00H ; s-a apasat tasta de pe coloana 0
        MOV      A,R3    ; se reconstituie acumulatorul
        SWAP     A        ; se interschimba bitii mai semnificativi
                        ; cu cei mai putini semnificativi ai
                        ; acumulatorului
        ANL      A,#00001111B ; se mascheaza bitii mai semnificativi
                        ; pentru identificarea codului liniei
        JB      ACC.1,LINIE1 ; salt daca tasta apasata era pe linia 1
        JB      ACC.2,LINIE2 ; salt daca tasta apasata era pe linia 3
        SJMP     PLAY      ; salt la rutina de afisare, tasta 0
COL1:   MOV      R1,#01H ; s-a apasat tasta de pe coloana 0
        MOV      A,R3    ; se reconstituie acumulatorul
        SWAP     A        ; se interschimba bitii mai semnificativi
                        ; cu cei mai putini semnificativi ai
                        ; acumulatorului
        ANL      A,#00001111B ; se mascheaza bitii mai semnificativi
                        ; pentru identificarea codului liniei
        JB      ACC.1,LINIE1 ; salt daca tasta apasata era pe linia 1
        JB      ACC.2,LINIE2 ; salt daca tasta apasata era pe linia 2
        SJMP     PLAY      ; salt la rutina de afisare, tasta 1
COL2:   MOV      R1,#02H ; s-a apasat o tasta de pe coloana 2
        MOV      A,R3    ; se reconstituie acumulatorul
        SWAP     A        ; se interschimba bitii mai semnificativi
                        ; cu cei mai putini semnificativi ai
                        ; acumulatorului
        ANL      A,#00001111B ; se mascheaza bitii mai semnificativi
                        ; pentru identificarea codului liniei
        JB      ACC.1,LINIE1 ; salt daca tasta apasata era pe linia 1
        JB      ACC.2,LINIE2 ; salt daca tasta apasata era pe linia 2
        SJMP     PLAY      ; salt la rutina de afisare, tasta 2
COL3:   MOV      R1,#03H ; s-a apasat o tasta de pe coloana 3
        MOV      A,R3    ; se reconstituie acumulatorul
        SWAP     A        ; se interschimba bitii mai semnificativi
                        ; cu cei mai putini semnificativi ai
                        ; acumulatorului
        ANL      A,#00001111B ; se mascheaza bitii mai semnificativi
                        ; pentru identificarea codului liniei
        JB      ACC.1,LINIE1 ; salt daca tasta apasata era pe linia 1
        JB      ACC.2,LINIE2 ; salt daca tasta apasata era pe linia 2
        SJMP     PLAY      ; salt la rutina de afisare, tasta 3
LINIE1: MOV      A,#04H ; tasta apasata era pe linia 1
        ADD      A,R1    ; se aduna codul coloanei cu codul liniei
        MOV      R1,A    ; codul hexa al tastei apasate este in R1
        SJMP     PLAY      ; salt la rutina de afisare, tastele 4..7
LINIE2: MOV      A,#08H ; tasta apasata era pe linia 2
        ADD      A,R1    ; se aduna codul coloanei cu codul liniei
        MOV      R1,A    ; codul hexa al tastei apasate este in R1

```

```
        SJMP  PLAY      ; salt la rutina de afisare, tastele 8..B
PLAY:    AJMP  NEXT      ; proces ciclic de citire a tastaturii
END.
```


6. SISTEM UNIVERSAL, MODULAR, DE ACHIZIȚII DE DATE

6.1 MĂRIMI DE INTRARE ÎN SISTEMUL DE ACHIZIȚII DE DATE

Pentru proiectarea oricărui sistem de achiziții de date dedicat este esențială studierea mărimilor de intrare prelevate din procesul studiat. Pentru un sistem de achiziții de date universal, ce poate fi utilizat în multe domenii de aplicații, cum este cel proiectat și care va fi descris, din punct de vedere funcțional, în continuare, trebuie avute în vedere o gamă largă de considerații de proiectare, pentru a se putea îndeplini cerințele de măsurare pentru diversitatea mare de domenii de utilizare. Se va arăta că înregistrarea variației în timp a trei tensiuni și a trei curenți permite determinarea tuturor indicatorilor și parametrilor sistemelor electrice.

6.2 SPECIFICAȚIILE DE PROIECTARE ALE SISTEMULUI DE ACHIZIȚII DE DATE

Analiza semnalelor electrice va fi realizată utilizând un sistem specializat de achiziții de date și de analiză, de concepție originală, dispunând de următoarele caracteristici:

- echipament de măsurare lucrând în domeniu temporal, dispunând de șase canale analogice de intrare, dispunând de următoarele variante de configurare:
- șase canale analogice de tensiune, sau
- trei de canale analogice de tensiune și trei canale analogice de curent, sau
- șase canale analogice de curent;
- atât intrările de tensiune, cât și cele de curent, sunt complet diferențiale, pentru a se putea conecta cu ușurință atât în scheme de măsurare de tip stea (Y), cât și triunghi (Δ);
- intrările analogice de tensiune și de curent asigură izolarea galvanică a semnalelor de intrare în raport cu sistemul de achiziții de date;
- intrările analogice de tensiune și de curent permit achiziția unor semnale de intrare cu dublă polaritate, iar sensibilitatea intrărilor (gama mărimilor analogice de intrare) poate fi selectată prin *software*;

- circuitele de intrare de tensiune și de curent își conservă caracteristicile de exactitate pentru o depășire de 1,2 ori a mărimilor nominale de intrare. De asemenea, consumul circuitelor de intrare, atât de tensiune, cât și de curent, nu depășește 3 VA;
- echipamentul de măsurare permite atât achiziția sincronă a celor șase semnale analogice de intrare, cât și facilități de supraeșantionare pentru semnalele de același tip (tensiune sau curent);
- echipamentul de măsurare asigură o conversie analog-numerică pe 8/12 biți și cel puțin 32 de eșantioane pe o perioadă a semnalelor de intrare de frecvență fundamentală (maximum 10 kHz). Această ultimă afirmație este echivalentă cu aceea că frecvența minimă de eșantionare, pentru șase canale analogice de intrare, să fie de cel puțin 32 kHz, ceea ce satisface teorema eșantionării a lui Shannon;
- sistemul posedă un grad înalt de inteligență, asigurat sub o formă distribuită de către o unitate centrală de prelucrare locală a sistemului și de către un calculator gazdă performant, având rolul:
- de a degreva un sistem de calcul gazdă, de gestionarea procesului de achiziționare a semnalelor de intrare;
- de a realiza identificarea perturbațiilor care se manifestă în sistemul (procesul) studiat și de a adapta analiza în conformitate cu acestea, pentru a conferi un maximum de exactitate;
- de a realiza comunicația, în forme compatibile, cu alte sisteme de calcul, în vederea transmiterii informațiilor de stare și, eventual, de alarmare, către un nivel ierarhic superior;
- întreg sistemul de achiziție memorează eșantioanele prelevate din proces, iar după încheierea procesului de achiziție transferă rezultatele, pe un canal serial de mare viteză, unui sistem de calcul de tip **IBM - PC** pentru prelucrare numerică ulterioară și afișare.

Sistemul de achiziții de date universal, multifuncțional, proiectat integral în acest scop și care va fi prezentat în capitolele următoare, este un sistem portabil, 100% compatibil cu orice calculator de tip **IBM - PC** sau compatibil.

Sistemul posedă două moduri de lucru:

- modul *supraveghere*, caracterizat de o eficiență mai redusă - modul de identificare a perturbațiilor, care poate fi selectat atât local, cât și prin intermediul calculatorului gazdă și care se execută doar local;
- modul *analiză*, caracterizat de o eficiență crescută - identificare și analiză a perturbațiilor, care se execută distribuit în cadrul sistemului și a calculatorului gazdă.

Sistemul de achiziții de date este alcătuit din trei blocuri funcționale importante, și anume:

- *interfața de achiziții de date* propriu-zisă;
- *unitatea centrală de prelucrare locală*;

- *instrumente software distribuite*, incluzând:
 - *programe de gestionare flexibilă și adaptivă a procesului de achiziție de date*, dezvoltate pe calculatorul gazdă și transferate prin intermediul interfeței seriale sistemului de achiziții de date. Acest subnucleu *software* este scris în limbajul de asamblare al familiei de *microcontroller*-e **8051**, pentru accesarea facilă a resurselor sistemului și, de asemenea, pentru a se minimiza și optimiza lungimea codului executabil. O a doua posibilitate este aceea ca acest subnucleu *software* să fie înscris în memoria de program a unității centrale de prelucrare;
 - *programe de comunicație* între sistemul de achiziții de date și analiză și calculatorul gazdă de tip **IBM - PC (host computer)**. Comunicația pe interfața serială full duplex, este implementată în *software*-ul rezident în memoria de program a unității centrale de prelucrare locală, însă este completat cu un program, scris în limbajul de nivel înalt **C**, pentru salvarea eșantioanelor prelevate din proces sub forma unui fișier de date cu structură compatibilă cu pachetul de programe de analiză. Există posibilitatea de a utiliza facilitățile de comunicație ale platformei **HP VEE** pentru memorarea eșantioanelor prelevate din proces sub forma unui fișier de date;
 - *pachet de programe de analiză a semnalelor electrice*, denumit sugestiv **ESA - Electrical Signal Analyser** - și implementat sub forma unui *instrument virtual*, folosind platforma **HP VEE** sub sistemul de operare **Windows**.

Schema bloc a sistemului de achiziții de date și analiză este prezentată în fig. 6.1 (variantea simplificată), iar în fig. 6.2 este prezentată *structura hardware detaliată* (variantea simplificată) a sistemului de achiziții de date.

În următoarele capitole vor fi prezentate detaliat elementele funcționale ale subsistemelor componente, elementele de proiectare de bază și soluțiile originale implementate în cadrul sistemului de achiziții de date și analiză.

Întreaga arhitectură a sistemului de achiziții de date a fost elaborată ca să îndeplinească criterii de versabilitate și adaptabilitate la cerințele în continuă modificare ale standardelor privind analiza semnalelor electrice.

După cum va reieși din analiza detaliată a arhitecturii sistemului de achiziții de date, prezentată în subcapitolele următoare, acest instrument combinat, *hardware* și *software*, dispune de diferite opțiuni de echipare *hardware*, astfel încât să asigure o exactitate cât mai ridicată de măsurare și analiză.

Concepția acestui echipament este originală, având la bază criteriile de proiectare specificate în standardele internaționale aplicabile la această dată. Soluțiile de creștere a exactității de măsurare și a vitezei de achiziție sunt rodul

unui efort iterativ de proiectare, în vederea utilizării unui număr cât mai redus de componente electronice de înaltă performanță și fiabilitate, care să permită o compatibilitate deplină a interconectării.

Rezultatul acestui efort de proiectare constă într-un echipament simplu, versatil, cu facilități crescute de adaptabilitate, fiabil, cu consum redus de energie electrică, ce respectă normele de compatibilitate electromagnetică prevăzute în standardele internaționale.

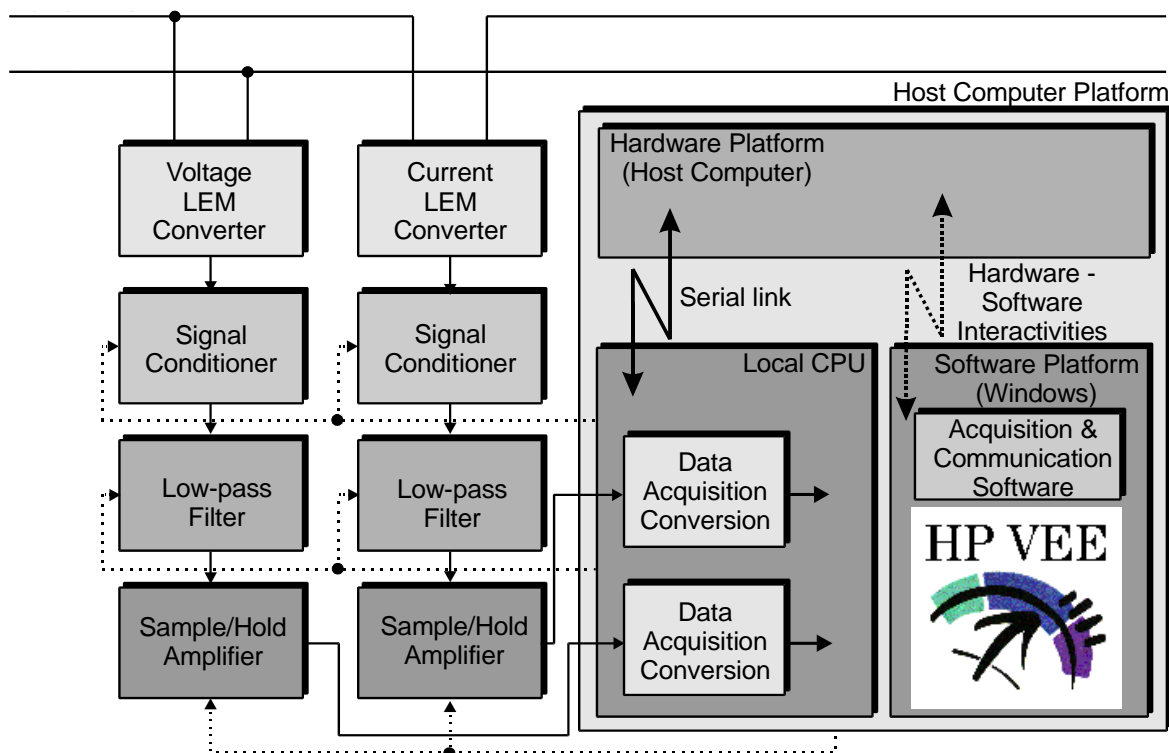


Fig. 6.1 Schema bloc a sistemului de achiziții de date și de analiză (variantea simplificată).

6.3 INTERFAȚA SPECIALIZATĂ DE ACHIZIȚII DE DATE A SISTEMULUI DE ACHIZIȚII DE DATE

Interfața de achiziții de date a sistemului proiectat este o interfață sincronă/asincronă, destinată achiziționării mărimilor rapid variabile, de uz general, folosită pentru prelevarea semnalelor de curent și de tensiune.

Criteriile principale de proiectare, utilizate pentru implementarea acestei interfețe specializate de achiziții de date, au la bază obținerea unei viteze crescute de achiziție și asigurarea unei exactități de măsurare cât mai ridicate.

6.3.1 INTERFAȚA DE ACHIZIȚII DE DATE PROPRIU-ZISĂ

Interfața de achiziții de date propriu-zisă permite achiziția simultană pe 6 canale a mărimilor electrice prelevate din proces. Interfața de achiziții de date propriu-zisă se compune din următoarele blocuri funcționale:

- *blocul de adaptare a semnalelor analogice*, cu rolul de compatibilizare a nivelelor de tensiune și de curent cu nivelele de tensiune acceptate de intrările circuitelor filtrelor *antialiasing*, circuitelor de eșantionare-memorare, respectiv de convertoarele analog-digitale;
- *blocul filtrelor antialiasing* (*antirepliere*);
- *blocul circuitelor de eșantionare-memorare*;
- *blocul circuitelor de conversie analog-digitală*;
- *blocul de interfață cu unitatea centrală de prelucrare locală*.

6.3.1.1 BLOCUL DE ADAPTARE A SEMNALELOR ANALOGICE

Blocul de adaptare a semnalelor analogice se compune din:

- trei, sau șase, convertoare curent-tensiune cu izolare galvanică cu intrări diferențiale;
- trei, sau șase, amplificatoare de instrumentație cu izolare optică;

Măsurarea curenților, folosind convertoarele curent-tensiune cu izolare galvanică, presupune luarea de măsuri speciale pentru a se asigura o exactitate ridicată și, de asemenea, o protecție eficientă a sistemului de achiziții de date (fig. 6.3).

Măsurarea curenților se efectuează foarte simplu, cu ajutorul unui șunt înseriat cu circuitul analizat, intrările de măsurare fiind complet diferențiale. Conectarea șuntului la intrarea blocului de măsură se efectuează cu cablu torsadat, pentru a asigura o imunitate cât mai ridicată la zgomot. Alimentarea circuitului de intrare este izolată de alimentarea restului schemei, fiind asigurată de o baterie, sau de o sursă stabilizată de tensiune continuă independentă, cu valoarea de 9V.

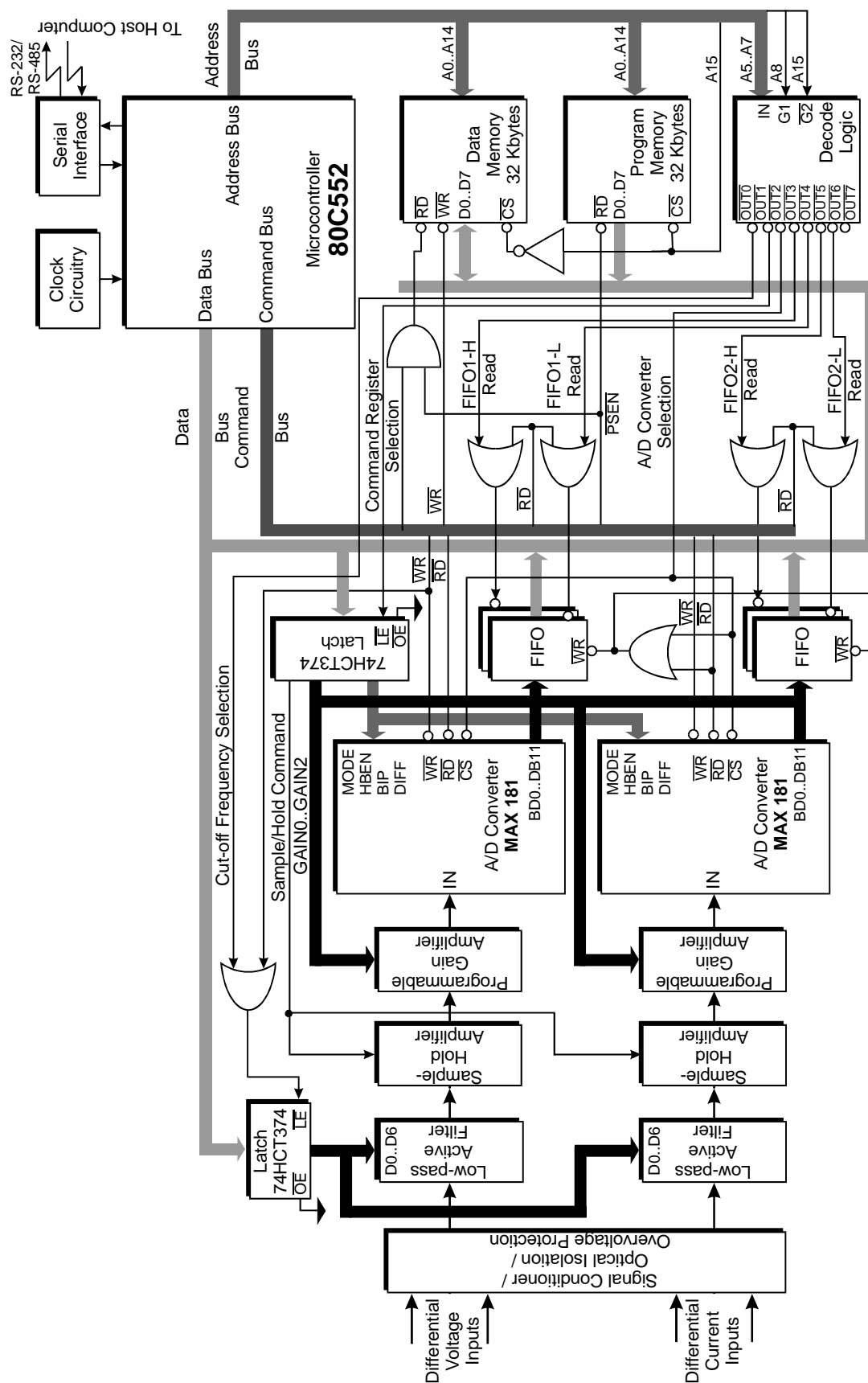


Fig. 6.2 Arhitectura detaliată a sistemului de achiziții de date și analiză (variantea simplificată).

La baza schemei de măsurare stă un amplificator cu izolație galvanică, cu factor de rejecție a modului comun de valoare mare ($>140\text{dB}$) de tip **HCPL7800**. Acest circuit a fost special conceput pentru a asigura performanțe ridicate de precizie, stabilitate și liniaritate, în condiții de medii zgomotoase, pentru aplicații ce necesită izolarea galvanică a semnalelor analogice.

Circuitul **HCPL7800** utilizează tehnologia *sigma-delta* ($\Sigma\text{-}\Delta$) a convertoarelor analog-digitale, amplificatoare cu chopper și o topologie complet diferențială.

Convertorul *sigma-delta* transformă semnalul analogic de intrare într-un cuvânt serial de biți, a cărui durată este proporțională cu semnalul de intrare.

Acest cuvânt serial este codat și transmis optic către circuitul detector. Semnalul detectat este decodat și este convertit în nivele analogice, precise, de tensiune. Prin filtrarea acestor nivele de tensiune se obține tensiunea de ieșire, proporțională cu semnalul analogic de intrare.

Mentținerea preciziei în timp și cu temperatura este asigurată de stabilitatea amplificatoarelor interne cu *chopper*-e.

Intrarea circuitului este conectată la șuntul utilizat pentru monitorizarea curentului.

Ieșirea diferențială a amplificatorului cu izolație galvanică este convertită într-un semnal raportat la masă cu ajutorul unui amplificator diferențial simplu.

Astfel relația dintre tensiunea de ieșire și curentul de intrare este:

$$V_{\text{OUT}} = \frac{R_2}{R_1} \cdot R_S \cdot I_{\text{IN}} = 40 \cdot R_S \cdot I_{\text{IN}} \quad (6.1)$$

Deși simplă, această structură necesită câteva precizări:

- căderea de tensiune maximă acceptată la bornele șuntului este de $\pm 200\text{ mV}$. Dacă forma de undă la ieșire este limitată sau distorsionată, atunci căderea de tensiune la bornele șuntului este prea mare;

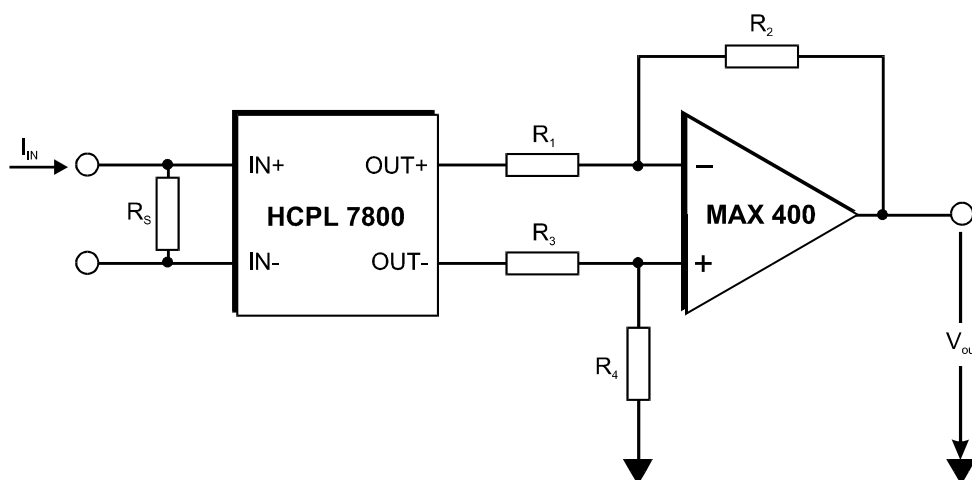


Fig. 6.3 Schema de principiu a convertorului I-U cu izolare galvanică.

- amplificatorul operațional cu care este implementat amplificatorul diferențial, cuplat la ieșirea circuitului **HCPL7800**, trebuie să fie suficient de precis pentru a nu afecta performanțele de offset și de derivă reduse ale circuitului. În plus, acest amplificator operațional trebuie să fie caracterizat de o bandă largă de frecvență și de slew-rate mare, pentru a nu fi afectate performanțele globale de viteză;
- șuntul utilizat trebuie să aibă o valoare mică, pentru a se minimiza puterea disipată. De asemenea, inductanța proprie a șuntului trebuie să fie cât mai redusă, pentru a se asigura o precizie corespunzătoare pentru semnale de curent rapid variabile sau de frecvență mare. Banda de frecvență a semnalelor de intrare determinată de circuitul **HCPL7800** și de amplificatorul operațional ce îl succede este de 65 kHz, permițând monitorizarea semnalelor de curent rapid variabile.

Monitorizarea tensiunilor este efectuată prin intermediul a trei amplificatoare de instrumentație cu izolare galvanică cu optocuploare. Această metodă asigură protecția eficientă a sistemului de achiziții de date împotriva supratensiunilor accidentale, adaptarea precisă a nivelelor de tensiune de ieșire și de intrare, impedanță de intrare foarte mare, factor de rejecție a tensiunilor și a zgomotelor de mod comun foarte ridicat. Din păcate, dezavantajul prezentat de aceste amplificatoare de instrumentație cu izolare optică este reprezentat de liniaritatea relativ necorespunzătoare a răspunsului (circa 10% procent de neliniaritate).

6.3.1.2 BLOCUL FILTRELOR ANTIREPLIERE

Semnalele reale de tensiune sau de curent pot conține componente spectrale de frecvență superioară frecvenței Nyquist (corespunzătoare jumătății frecvenței de eșantionare). În aceste situații, se manifestă un efect de repliere al spectrului semnalelor, cunoscut în literatura de specialitate sub denumirea de “*alias*”, care falsifică măsurările. Cea mai simplă și eficientă soluție pentru evitarea acestui fenomen constă în filtrarea semnalelor originale, utilizând un filtru de tip trece-jos, caracterizat de o frecvență de tăiere egală cu frecvența Nyquist, adică egală cu jumătate din frecvența de eșantionare. Acest filtru se numește filtru *antialiasing* (antirepliere).

Se impune ca atenuarea filtrului *antialiasing* în afara benzii de trecere să fie mai mare de 50 dB.

Problema care se ridică în implementarea unui asemenea tip de filtru este aceea că interfața specializată de achiziții de date a sistemului permite selectarea frecvenței de eșantionare prin *software*, într-un interval larg de valori. Cu alte cuvinte, frecvența de eșantionare nefiind constantă, frecvența Nyquist este variabilă și deci frecvența de tăiere a filtrului antirepliere trebuie să se modifice

pentru realizarea corectă a funcției de eliminare a frecvențelor superioare.

Având în vedere aceste considerente, implementarea filtrelor *antialiasing* a fost realizată utilizând un filtru activ de tip trece-jos, de ordinul al doilea, având frecvența de tăiere programabilă digital (prin *software*). S-au folosit circuite de tip **MAX 271**, care conțin două astfel de filtre, programabile fie în mod independent, fie simultan, în aceeași capsulă.

Structura filtrelor antialiasing este prezentată în fig. 6.4.

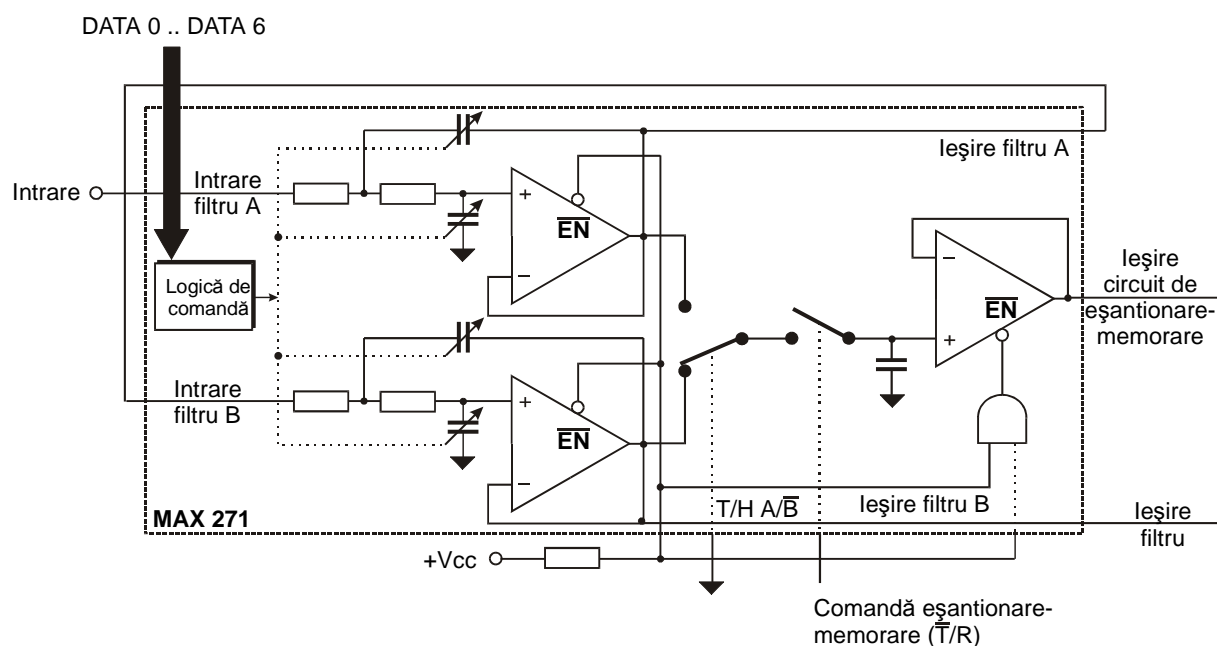


Fig. 6.4 Schema de conectare a filtrelor active, de tip trece-jos, programabile (*antialiasing*).

Aceste circuite au implementată o interfață cu un microprocesor pe 8 biți, permițând programarea frecvenței de tăiere a filtrului trece-jos în 128 de pași, prin intermediul a 7 linii de comandă. Frecvența de tăiere a filtrelor poate fi programată între limitele 1Hz și 25 kHz.

Ecuatiile care permit calcularea frecvenței de tăiere a filtrului sunt:

$$f_T = \frac{87,5}{87,5 - \text{CODE}} \cdot 1\text{kHz}; \quad \text{CODE} = (0 \div 63); f_T = (1 \div 3,75)\text{kHz} \quad (6.2)$$

$$f_T = \frac{262,5}{137,5 - \text{CODE}} \cdot 1\text{kHz}; \quad \text{CODE} = (64 \div 127); f_T = (3,75 \div 25)\text{kHz}$$

în care CODE reprezintă combinația de date aplicată terminalelor de comandă, DATA₀ ÷ DATA₆ (DATA₆ este bitul cel mai semnificativ). Intrările de date ale circuitului **MAX 271** sunt compatibile atât TTL, cât și CMOS.

Frecvența de tăiere a filtrului înregistrează erori dependente de datele de programare folosite. Astfel, eroarea frecvenței de tăiere este minimă pentru CODE=0, situație pentru care filtrele sunt ajustate special din fabricație, iar eroarea maximă (circa ±9,5%) se înregistrează pentru CODE=127. În această

situație, frecvența de tăiere a filtrului se încadrează în intervalul $22,63 \div 27,38$ kHz.

Atenuarea în afara benzii de trecere pentru filtrele active implementate în acest circuit este de 40 dB/decadă. Pentru respectarea cerințelor de atenuare recomandate de normativele în vigoare, s-a folosit legarea în cascadă a celor două filtre din aceeași capsulă, obținându-se o atenuare de 80 dB/decadă.

În vederea reducerii efortului de programare a filtrelor active antialiasing, circuitele **MAX 271** sunt configurate în modul "*pin-programming*". Acest mod de programare este realizat prin conectarea pinului MODE la tensiunea pozitivă de alimentare. În această configurație, ambele filtre de tip trece jos sunt caracterizate de aceeași frecvență de tăiere, specificată de combinația de date aplicată terminalelor de comandă, $DATA_0 \div DATA_6$. Astfel programarea ambelor filtre este realizată într-o singură etapă (ciclu de programare).

În acest mod de funcționare, semnalele de interfațare cu magistrala sistemului nu au nici o semnificație, însă utilizatorul are acces la comenzile de activare a circuitului de eșantionare-memorare intern, de selecție a semnalului (ieșirea filtrului A, respectiv ieșirea filtrului B) ce urmează a constitui sursa de intrare a circuitului de eșantionare-memorare și la comanda de eșantionare-memorare propriu-zisă. Legarea în cascadă a celor două filtre permite simplificarea comenzilor disponibile utilizatorului, astfel (conform fig. 6.4):

- este activată în permanență funcționarea circuitului de eșantionare-memorare ($T/H_EN=1$);
- intrarea circuitului de eșantionare-memorare este întotdeauna constituită de ieșirea filtrului B ($T/H_A/B=0$);
- comanda de eșantionare-memorare este disponibilă în exterior și este asigurată de către unitatea centrală de prelucrare locală.

Circuitul de eșantionare-memorare, implementat în circuitul **MAX 271**, este caracterizat de următorii parametri:

- timp de stabilire (pentru o eroare de 0,1%): 500ns;
- timp de achiziție (pentru o eroare de 0,1%): $1,5\mu s$;
- rata de cădere a tensiunii de ieșire (în starea de memorare): $30\mu V/\mu s$;
- tensiune de decalaj (incluzând și tensiunea de offset a filtrelor): $\pm 4mV$.

Impedanța de intrare a filtrelor nu este constantă, ci este dependentă de frecvență, fiind cuprinsă între $5M\Omega$ în curent continuu și $100k\Omega$ la 25kHz, fiind însă suficient de mare pentru a nu introduce erori de măsurare.

Bateria de filtre, obținută prin legarea în cascadă a celor două filtre implementate în circuitul **MAX 271**) este caracterizată prin distorsiuni armonice totale mai mici de -70dB și de atenuare în bandă de practic 0dB.

6.3.1.3 BLOCUL CIRCUITELOR DE EȘANTIONARE-MEMORARE SUPLIMENTARE (EXTERNE)

Deoarece semnalele de intrare, în sisteme electrice, pot fi rapid variabile și se caracterizează prin prezența armonicilor superioare, având frecvențe multipli întregi ai frecvenței de 50 Hz, cât și frecvențe care nu sunt multipli întregi ai frecvenței fundamentale, semnale care poartă denumirea de interarmonici, (în tehnică s-a constatat că numai armonicile până la ordinul 40 dau efecte nedorite în instalațiile electrice) sistemul de achiziții de date utilizează 6 circuite de eșantionare-memorare. Acestea pot fi constituite fie de circuitele de eșantionare-memorare implementate în cadrul circuitelor **MAX 271**, fie de circuite suplimentare externe, de tip **LF198**, pentru a menține la intrarea convertoarelor analog-digitale semnalul de măsurat constant ($\pm 0,25\text{LSB}$) pe toată durata conversiei. De asemenea, caracterizarea unor anumite procese impune eșantionarea sincronă a semnalelor de intrare. Astfel, comanda de eșantionare-memorare poate fi comună tuturor celor 6 circuite, realizându-se astfel o achiziție sincronă, sau poate fi individualizată la nivel de câte un circuit, realizându-se un proces de achiziție asincronă sau poate fi implementată o schemă cu facilități de supraeșantionare la nivel de grupuri de câte trei circuite.

Circuitele externe folosite, de tip **LF198**, sunt circuite de eșantionare-memorare, realizate monolitic, ce utilizează tehnologia **BI-FET** pentru a obține o precizie superioară atât în curent continuu, cât și pentru achiziția semnalelor rapid variabile.

Funcționând ca circuite repetoare, acestea sunt caracterizate printr-o precizie de 0,002% a amplificării și de un timp de achiziție de 5 μs , pentru o precizie de 0,01%. Folosirea tehnologiei bipolare în realizarea etajului de intrare asigură tensiuni de offset mici și o bandă largă de frecvență (1MHz), fără probleme de stabilitate. Impedanța de intrare, de $10^{10}\Omega$, permite achiziționarea semnalelor de excitație ce provin de la surse de semnal cu impedanță internă ridicată, fără a fi afectată exactitatea.

Amplificatorul de ieșire combină dispozitive bipolare și tranzistoare JFET cu canal P, pentru a putea asigura rate de cădere de 5mV/min, utilizând o capacitate de memorare de 1 μF .

Se constată că aceste circuite de eșantionare-memorare externe sunt mult mai performante decât cele implementate în structura filtrelor antialiasing, sistemul de achiziții de date putând fi echipat opțional cu acestea din urmă. Intrările în circuitele de eșantionare-memorare externe sunt constituite de ieșirile filtrelor B din cadrul fiecărui bloc de tip filtru antialiasing.

Trebuie însă remarcat faptul că, la cele două tipuri de circuite, comenzile de eșantionare-memorare sunt inversate (comanda pentru starea de eșantionare

în cazul circuitelor implementate intern în structura filtrelor antirepliere este activă pe nivel coborât, în cazul circuitelor suplimentare externe fiind activă pe nivel ridicat; în mod similar pentru trecerea în starea de memorare). Acest impediment este rezolvat simplu, prin *software*, existând variante de programe, ușor diferite, pentru diferitele variante de implementare *hardware*.

Circuitul este caracterizat de un factor de rejecție a surselor de alimentare ridicat (110dB), atât în modul eșantionare, cât și în modul memorare.

Intrările logice ale circuitului **LF198** sunt diferențiale și au curenți de intrare mici, fiind interfațabile direct cu familiile logice TTL, CMOS.

Compensarea offset-ului static (în curent continuu) se realizează prin conectarea pinului de compensare la cursorul unui semireglabil de $1\text{k}\Omega$, având unul din terminale legat la tensiunea pozitivă de alimentare, iar celălalt terminal înseriat cu o rezistență (ce trebuie să asigure un curent de aproximativ $0,6\text{ mA}$ prin potențiometru) la masă.

Compensarea offset-ului dinamic (anularea pasului de memorare) se realizează prin folosirea suplimentară a unui inversor cu potențiometrul de ajustare conectat între intrare și ieșire. Un condensator de 10pF , conectat între cursor și capacitatea de memorare, va asigura ajustarea cu $\pm 4\text{mV}$ a pasului de memorare, utilizând o capacitate de memorare de $0,01\mu\text{F}$ și tensiunea de alimentare logică de $+5\text{V}$.

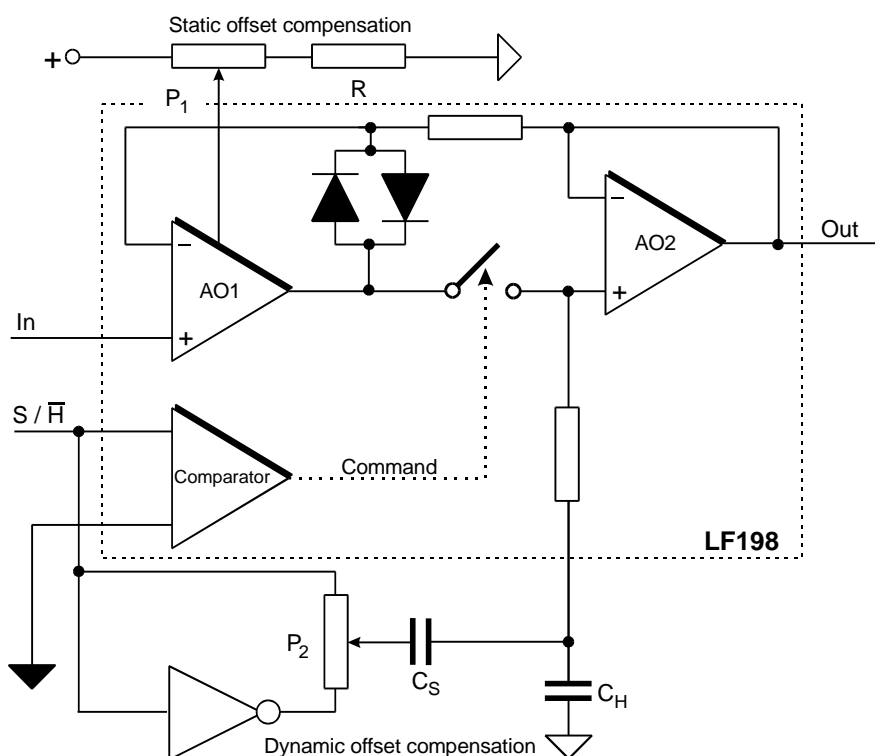


Fig. 6.5 Schema de utilizare a circuitelor de eșantionare-memorare LF 198.

În fig. 6.5 este ilustrată structura circuitelor de eșantionare-memorare,

conform schemei de utilizare în sistemul de achiziții de date (incluzând opțiunile de compensare a tensiunii de decalaj - offset-ul static - și de compensare a pasului de memorare - offset-ul dinamic -).

Pentru o funcționare corectă, semnalele logice aplicate circuitului de eșantionare-memorare **LF198** trebuie să fie caracterizate de o viteză de variație mai mare decât $1\text{V}/\mu\text{s}$. Semnalele de comandă mai lente pot determina valori sensibil crescute ale pasului de memorare. În situația concretă a sistemului proiectat, fronturile semnalelor de comandă fiind asigurate de circuite din familia TTL nu determină erori în funcționarea circuitului de eșantionare-memorare.

6.3.1.4 BLOCUL CONVERTOARELOR ANALOG-DIGITALE

În sistemul de achiziții de date, mai precis în cadrul interfeței specializate de achiziții, achiziția semnalelor de intrare se efectuează pe 12 biți, ceea ce permite o precizie de măsurare de 0,04875%. Gama tensiunilor de intrare în sistemul de achiziții de date este selectabilă *software* și/sau *hardware* în intervalele:

- $(-0,3125 \div +0,3125)\text{V}$;
- $(-0,625 \div +0,625)\text{V}$;
- $(-1,25 \div +1,25)\text{V}$;
- $(-2,5 \div +2,5)\text{V}$;
- $(-5 \div +5)\text{V}$;
- $(-10 \div +10)\text{V}$.

Conversia analog-digitală se efectuează cu ajutorul a șase convertoare analog-digitale, cu rezoluție de 12 biți, cu aproximații succesive, de tipul **MAX 181**, realizate în tehnologie CMOS.

6.3.1.4.1 DESCRIEREA FUNCȚIONALĂ A BLOCULUI DE CONVERSIE ANALOG-DIGITALĂ DIN CADRUL INTERFEȚEI SPECIALIZATE DE ACHIZIȚII DE DATE

Interfața specializată de achiziții de date utilizează șase circuite de tip **MAX181** pentru conversia analog-digitală a eșantioanelor de tensiune și de curent prelevate din proces.

Circuitul **MAX 181** este un sistem de achiziții complet pe 12 biți, care include un multiplexor analogic de intrare cu șase canale, un circuit de eșantionare-memorare de bandă largă, o sursă de tensiune de referință, realizată cu o diodă Zener cu derivă mică ($25\text{ ppm}/^{\circ}\text{C}$) și o interfață flexibilă cu

microprocesoare de 8/16 biți, caracterizată prin viteză ridicată de transfer.

Circuitul este caracterizat printr-o rată ridicată de conversie (maxim 100 kHz pentru un canal), pentru care raportul semnal-zgomot al convertorului are o valoare de minim 74dB iar distorsiunile armonice totale au o valoare de minim -89dB, o excelentă liniaritate ($\pm 0,05\text{LSB}$) și un consum redus de energie (110 mW). **MAX 181** poate fi configurat *software* pentru conversia semnalelor unipolare ($0\div 5$)V sau bipolare ($-2,5\div +2,5$)V, diferențiale sau nediferențiale, în mod independent pe fiecare canal.

Arhitectura unui astfel de circuit este prezentată în fig. 6.6.

Capacitatea de intrare se comportă ca o capacitate de memorare, fiind încărcată de semnalul de intrare, la fiecare conversie analog-digitală. Această capacitate este încărcată prin intermediul unei rezistențe interne, cu valoare de $1\text{k}\Omega$, conectată în serie cu semnalul de la intrare.

În modul de lucru nediferențial, între două cicluri de conversie (atunci când semnalul $\overline{\text{BUSY}}$ este pe nivel logic ridicat), intrarea analogică selectată este conectată la bornele capacității de memorare. Când se inițiază o conversie, se comandă deconectarea capacității de memorare de la intrarea sistemului, în acest mod realizându-se memorarea tensiunii de intrare. Atunci când comutatorul se închide, la sfârșitul conversiei, capacitatea de memorare este reconectată la semnalul de intrare, urmând un nou ciclu de încărcare. Efectul de sarcină al intrărilor multiplexorului analogic asupra semnalelor de intrare este foarte redus, astfel încât, de obicei, nu este necesară folosirea unui repetor de viteză mare, deoarece convertorul analog-digital este deconectat de la terminalul de intrare, pe durata conversiei curente.

Durata necesară circuitului de eșantionare-memorare pentru achiziționarea semnalului de intrare depinde de constanta de timp a procesului de încărcare a capacității de memorare. Dacă impedanța sursei de semnal este mare, timpul de achiziție se lungește și determină scăderea ratei de conversie. Timpul de achiziție al semnalului poate fi calculat astfel (fig. 6.7):

$$t_s = 10 \cdot (R_s + R_{\text{IN}}) \cdot 20\text{pf} > 1,875\mu\text{s} \quad (6.3)$$

în care:

- t_s reprezintă timpul de eșantionare;
- R_s este impedanța sursei de semnal;
- $R_{\text{IN}} = 1\text{k}\Omega$

Circuitul de eșantionare prevăzut la intrarea convertorului analog-digital este realizat pentru a optimiza urmărirea semnalelor cu amplitudini mari și bandă largă de frecvență. Banda de frecvență tipică a circuitului de eșantionare-memorare este de 6MHz, permițând măsurarea semnalelor periodice cu frecvențe superioare vitezei maxime de eșantionare de 100kHz pentru care este garantată rezoluția de 12 biți, folosind tehnica supraeșantionării.

Din structura fiecărui convertor este utilizat doar un singur canal de intrare al multiplexorului analogic implementat intern, acest canal monitorizând

câte o mărime de intrare asociată fiecărei faze a rețelei trifazate. Fiecare convertor are prevăzute reglaje de offset (ajustare a zeroului) și un reglaj comun de ajustare a câștigului (prin ajustarea tensiunii de referință). Toate cele trei convertoare sunt selectate prin intermediul unui semnal de selecție unic, astfel încât, la un moment de timp, poate fi declanșat un proces de trei conversii analog-digitale, în paralel. Motivația acestei soluții de implementare rezidă în alegerea drept criteriu principal de proiectare creșterea vitezei de achiziție a interfeței specializate de achiziții de date. Convertoarele **MAX181** funcționează în modul de lucru *port de intrare-ieșire*, pe 16 biți, lucru realizat prin conectarea semnalelor de configurare HBEN="0" și MODE="1". Convertoarele folosesc sursa de tensiune de referință internă a unui singur convertor. Această facilitate asigură o precizie sporită a procesului de conversie analog-digitală.

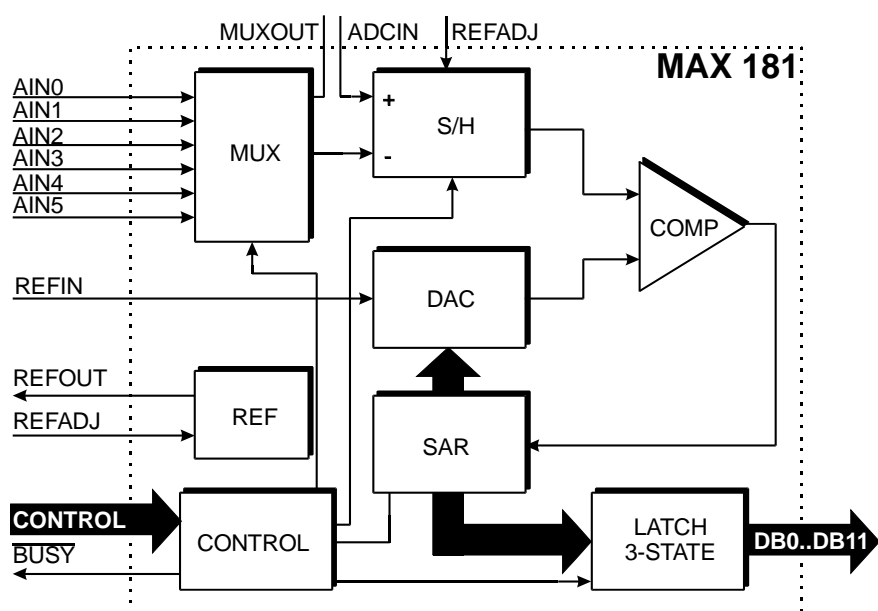


Fig. 6.6 Structura sistemului de achiziții de date MAX 181.

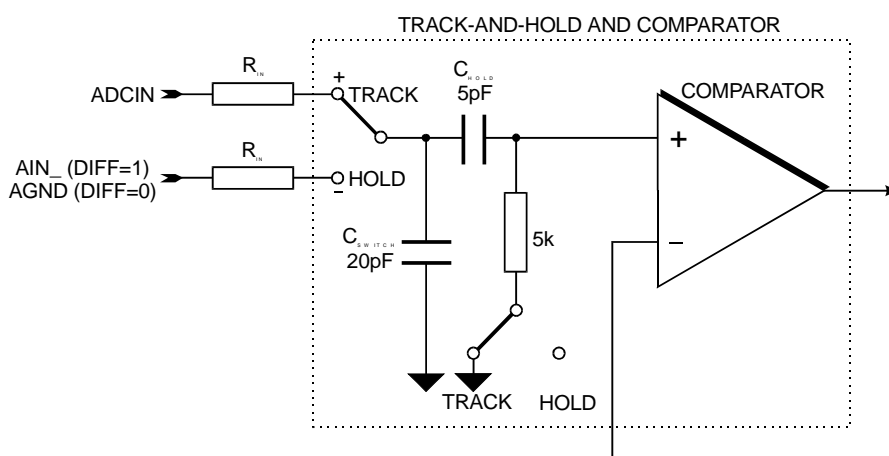


Fig. 6.7 Circuitul de eșantionare-memorare al convertorului MAX 181.

sistemului de achiziții de date trebuie să îndeplinească criterii foarte stricte de exactitate și stabilitate.

Astfel, amplificatorul operațional utilizat a fost ales de tip **MAX 400**, fiind caracterizat de amplificarea în buclă deschisă foarte mare (tipic 110 dB), de o tensiune de offset extrem de mică ($< 1\mu\text{V}$) și de o stabilitate în timp și cu temperatura foarte bune.

Prin utilizarea unor rezistențe de precizie (cu toleranță de 0,01%), caracterizate și de o bună stabilitate în timp și cu temperatura, și care au o valoare sensibil mai mare decât rezistența în stare de conducție a canalului selectat al multiplexorului analogic implementat intern în structura convertorului analog-digital - pentru a se putea neglija efectul acestuia în calculul amplificării circuitului -, de exemplu minim $1\text{k}\Omega$, s-a realizat un circuit cu câștig reglabil (programabil *software* prin selectarea canalului de intrare al convertorului analog-digital **MAX181**) de înaltă precizie și performanță.

În cazul prezentat în fig. 6.8, nivelurile de amplificare rezultă 1, 2, 4, 8, 16, 32. Recalcularea valorilor rezistențelor, conform relației (caracteristica de transfer a unui amplificator neinvertor cu amplificator operațional):

$$A_i = 1 + \frac{\sum_{k=1}^i R_k}{\sum_{k=i+1}^6 R_k} \quad (6.4)$$

conduce la obținerea unor niveluri de amplificare diferite, conform dorințelor.

O variantă opțională de a realiza *software* selectarea gamelor semnalelor analogice de intrare, constă în intercalarea între ieșirea circuitului de eșantionare-memorare, implementat intern în structura convertorului **MAX 181**, și intrarea în circuitul propriu-zis de conversie analog-numerică, din structura aceluiași circuit, a unui amplificator cu câștig reglabil (fig. 6.9).

Multiplexorul care comandă rezistența de pe bucla de reacție a amplificatorului operațional, folosit în configurație de amplificator neinvertor, este un multiplexor adresabil, realizat în tehnologie **CMOS**, fiind caracterizat de o rezistență a canalului în starea **ON** (în conducție) foarte mică (circa 5Ω). De asemenea, împerecherea canalelor este foarte precisă, abaterile rezistențelor canalelor în starea **ON** fiind sub valoarea de 0,25%. Multiplexorul folosit, de tip **MAX 368**, este caracterizat și de protecția intrărilor contra supratensiunilor accidentale: un canal, indiferent dacă este în stare de conducție sau de blocare, suportă o tensiune de intrare de maximum 45 V, chiar dacă circuitul nu este alimentat. Un canal selectat (în stare de conducție) va trece în stare OFF la aplicarea pe intrarea corespunzătoare a unei supratensiuni.

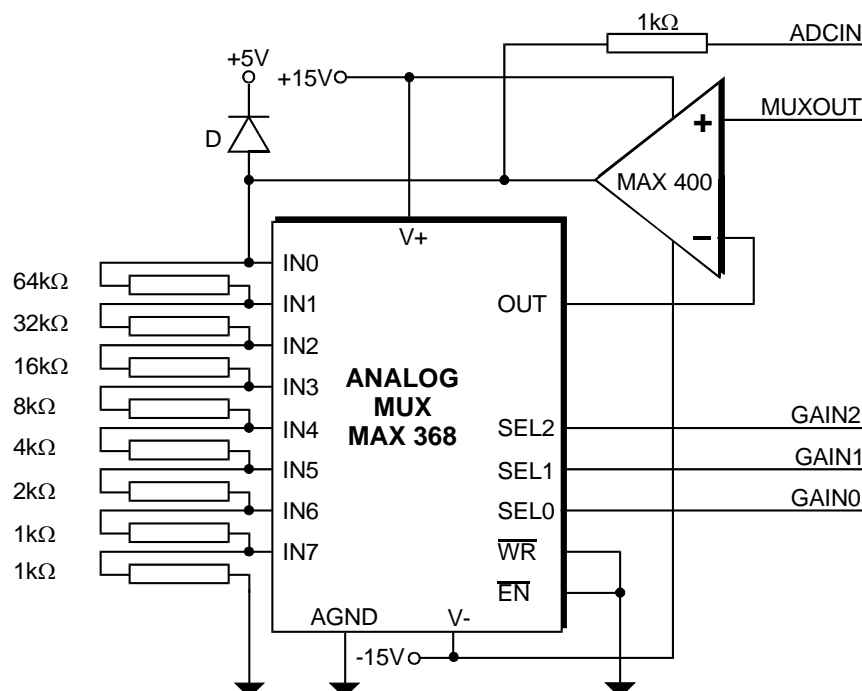


Fig. 6.9 Schema electrică a amplificatorului cu câștig reglabil (variantă opțională).

În cazul prezentat în fig. 6.9, nivelurile de amplificare rezultă de tipul 2^i , în care $i = 0 \div 7$. Recalcularea valorilor rezistențelor permite obținerea unor niveluri de amplificare diferite, conform necesităților.

Deoarece este realizat în tehnologie **CMOS** și funcționează în modul *dispozitiv de intrare-ieșire*, ale cărui diagrame temporale sunt prezentate în fig. 6.10, ieșirile de date ale convertorului de tip **MAX 181** sunt sensibile la activitatea pe magistrala de date a sistemului pe durata efectuării unei conversii.

Tranzițiile de pe magistrala de date a sistemului pe durata efectuării unei conversii cauzează erori în funcționarea comparatorului ce comandă registrul de aproximații succesive și, deci, produce erori în rezultatul conversiei. Pentru rezolvarea acestei probleme s-a adoptat soluția ca ieșirile de date ale convertoarelor să fie prevăzute cu câte o memorie **FIFO (First In First Out)** cu lungimea cuvântului de 16 biți, de tip **Am7204A**, în defavoarea celeilalte soluții posibile: introducerea procesorului din unitatea centrală de prelucrare locală în stare de așteptare pe durata efectuării conversiei, deoarece prima soluție duce la creșterea vitezei de achiziție (fig. 6.10). De asemenea, utilizarea acestor memorii permite achiziționarea datelor cu o frecvență ridicată și citirea acestora, de către unitatea centrală de prelucrare, cu o rată mai redusă.

Adâncimea memoriilor **FIFO**, adică numărul de locații ale acestora, este ales în funcție de numărul maxim de eșantioane care vor fi prelevate din rețeaua electrică trifazată (în fig. 6.11 este utilizată o memorie **FIFO** cu 4096 locații). Extinderea capacităților de memorare poate fi realizată foarte ușor, prin

utilizarea unor memorii de tip **Am7205A** (cu 8192 locații), deoarece circuitele sunt compatibile pin la pin.

Blocul memoriilor **FIFO** dispune de o intrare de inițializare care este comandată de către circuitul de inițializare al unității centrale de prelucrare locală.

Citirea rezultatelor obținute în urma efectuării conversiei se face printr-o singură instrucțiune de citire, ceea ce determină înscrierea datelor de ieșire ale fiecărui convertor (pe doi octeți) în câte o pereche de memorii **FIFO**. O funcție logică de tip SAU între semnalul de citire furnizat simultan celor șase convertoare și semnalul comun de selecție al acestora determină înscrierea simultană a datelor de ieșire în memoriile **FIFO**. În urma acestei operații, se poate comanda o nouă conversie.

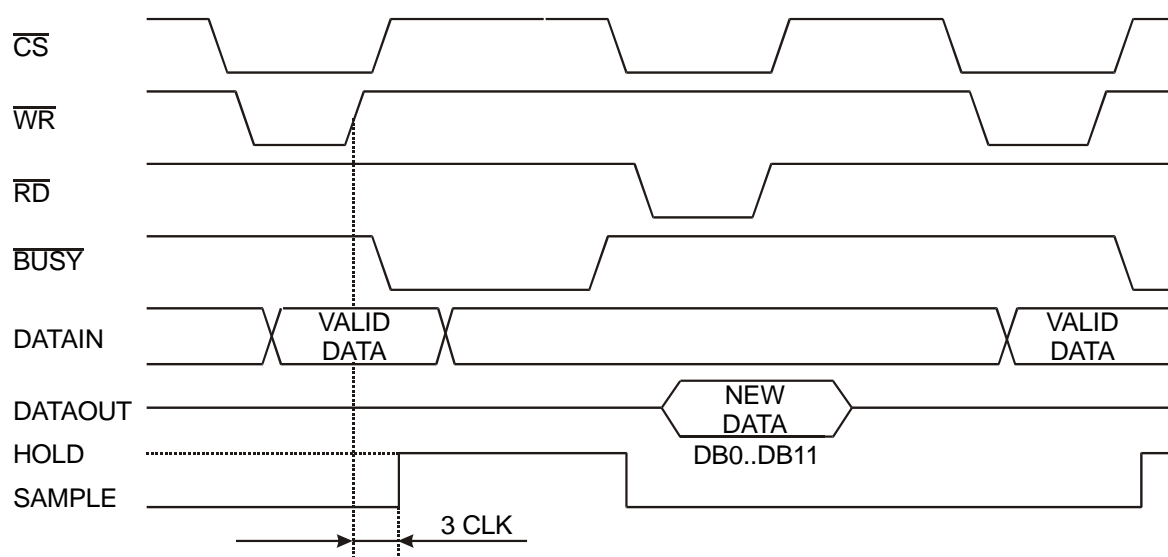


Fig. 6.10 Diagramele de funcționare ale convertorului analog-digital.

Scrierea în memoria **FIFO** determină incrementarea atât a *pointer*-ului de scriere, cât și a *pointer*-lui de citire. Procesul de memorare continuă până la memorarea temporară a numărului prestabilit inițial de eșantioane sau până la umplerea memoriei. În oricare dintre aceste două situații, la încheierea operațiilor de memorare, se activează facilitatea de **RETRANSMIT**, prin activarea semnalului **RT**, fie de către semnalul **FF** (**FIFO Full**), fie de către un semnal dedicat inițializării ciclului de citire.

Memoriile **FIFO** au ieșiri de tip three-state, preluarea conținutului făcându-se cu ajutorul unor semnale de selecție distincte. Aceste semnale sunt obținute prin condiționarea ieșirilor unui decodificator de porturi (adrese) cu semnalul de citire al procesorului. Prin manipularea severnțială a seturilor de ieșiri three-state ale memoriilor **FIFO** se realizează, de fapt, multiplexarea numerică a datelor de ieșire, obținute în urma procesului de conversie analog-digitală.

Pentru a facilita preluarea și interpretarea rezultatelor, semnalul de selecție **MSBYTE**, corespunzător fiecărui convertor, permite preluarea, pe cei patru biți mai puțin semnificativi, a informațiilor de pe liniile de ieșire $DB_{11} \div DB_8$ ale circuitelor, iar semnalul de selecție **LSBYTE** determină preluarea informațiile de pe liniile de ieșire $DB_7 \div DB_0$.

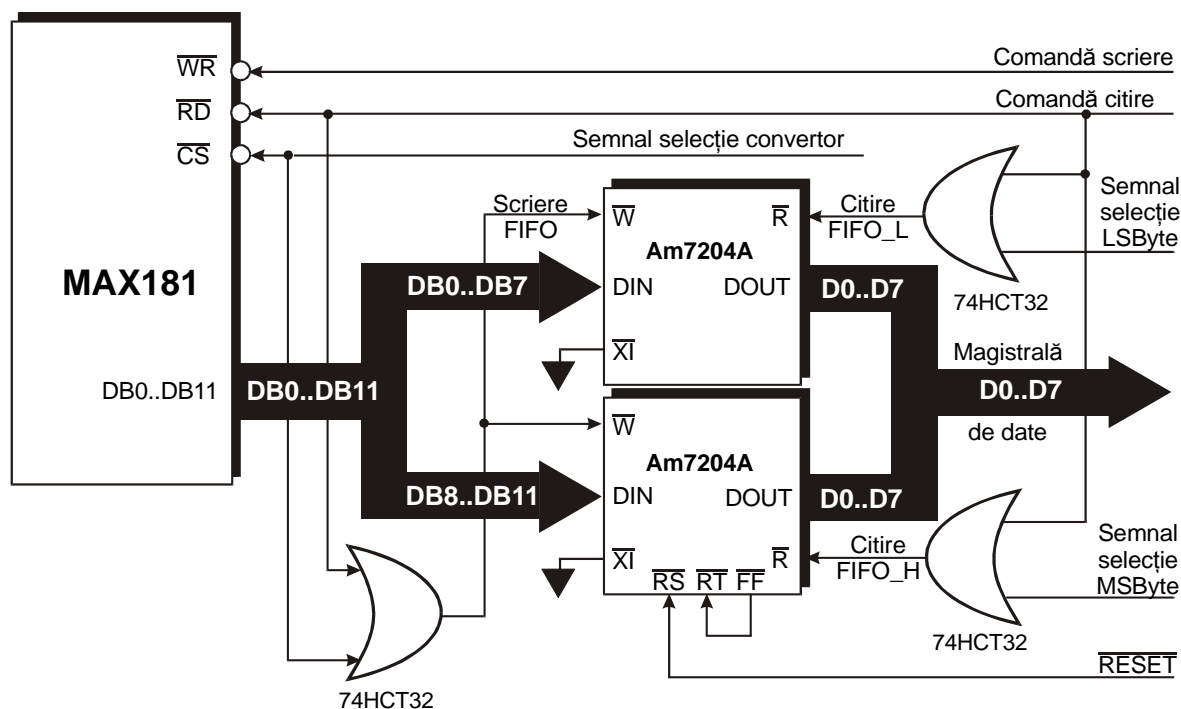


Fig. 6.11 Buffer-aria ieșirilor de date ale convertorului A/D cu memorii FIFO.

Biții mai semnificativi, neutilizați, ai memoriei **FIFO** selectată cu semnalul **MSBYTE** vor conține informațiile referitoare la nivelul de amplificare programat, astfel:

- cei trei biți mai semnificativi preiau informația de selecție a amplificării din registrul de comandă;
- cel de-al patrulea bit preia informația de depășire de la un comparator cu fereastră; tensiunile de referință ale acestuia sunt obținute de la sursa de referință internă a sistemului de achiziții **MAX181**.

Prin alocarea diferită a biților de date de la convertorul analog-digital **MAX 181** la liniile de intare ale memoriilor **FIFO**, se poate realiza foarte ușor comutarea *software* între rezoluția de interpretare a rezultatelor conversiilor pe 12/8 biți.

Trebuie remarcată simplitatea modului de interfațare a blocului de conversie analog-digitală cu unitatea centrală de prelucrare: prin intermediul unor porturi de intrare-ieșire.

Această structură simplă, dar, în același timp, eficientă, a fost adoptată pentru a păstra o compatibilitate deplină cu modul de organizare al

microcontroller-ului **80C552** (sub formă de porturi cu funcții dedicate), ce stă la baza unității centrale.

Pentru evitarea problemelor de sincronizare ce pot apare în funcționare (semnalele de selecție, citire și scriere asociate convertorului trebuie să fie corelate cu ceasul acestuia), ceasul, folosit de convertor pentru efectuarea conversiilor, este obținut prin divizare din ceasul unității centrale de prelucrare.

Timpul minim de achiziție, compus din:

- timpul de eșantionare (maxim 5μs),
- timpul de stabilire a ieșirii circuitelor eșantionare-memorare (maxim 0,5μs),
- timpul de conversie analog-digitală (maxim 9,5μs),
- la care se adaugă durata instrucțiunilor de selecție a parametrilor conversiei și de comandă a unei conversii analog-digitale, de comandă a circuitului de eșantionare-memorare (4μs)

rezultă de maximum 19μs, ceea ce permite o frecvență de achiziție de 51,2 kHz pe un canal, extrem de utilă în aplicațiile de acest tip necesitând analiză spectrală a semnalului eșantionat.

6.3.1.5 BLOCUL DE CONVERSIE DIGITAL-ANALOGICĂ

Secțiunea ieșirilor analogice este implementată utilizând un circuit de tip **MAX526**, ce conține patru convertoare digital-analogice pe 12 biți cu ieșire în tensiune. Circuitul include amplificatoare operaționale de precizie de tip buffer pentru a asigura ieșirile în tensiune. Pentru funcționare circuitul MAX526 necesită o tensiune de alimentare pozitivă, în gama (+12 ÷ +15)V, precum și o tensiune de alimentare negativă, cu valoarea de -5V, în raport cu tensiunea de referință a schemei (masa electrică 0V). Tensiunile de offset, amplificările și liniaritatea sunt ajustate tehnologic, astfel încât eroarea totală să nu depășească

$$1\text{LSB} \left(1\text{LSB} = \frac{V_{\text{REF}}}{2^{12}} = \frac{V_{\text{REF}}}{4096} \right).$$

Acest circuit dispune de o interfață digitală dublu buffer-ată, prin intermediul unui registru de intrare de 12 biți și a unui registru al convertoarelor D/A de 12 biți. Cuvântul înscris în acest din urmă registru este folosit pentru obținerea tensiunii de ieșire a convertorului. **MAX526** poate fi interfațat cu un microcontroller sau cu un microprocesor prin intermediul unei magistrale externe de date, cu lungimea de un octet. Cuvântul de programare este înscris în registrul de intrare prin intermediul a două instrucțiuni de scriere (activarea semnalului de scriere pentru cei 8 biți mai puțin semnificativi, respectiv pentru cei mai semnificativi 4 biți). Este disponibil și un semnal asincron de încărcare a registrului convertorului A/D, denumit $\overline{\text{LDAC}}$ și care este activ pe nivel logic

coborât. Toate intrările logice sunt compatibile atât TTL, cât și CMOS.

MAX526 conține patru convertoare A/D cu ieșire în tensiune. Convertoarele A/D sunt de tip rețea rezistivă R-2R “inversată”, care convertește cuvântul digital de intrare, pe 12 biți, într-o tensiune de ieșire proporțională cu acesta și dependentă de tensiunea de referință externă aplicată convertorului. Circuitul dispune de două intrări de tensiune de referință: o primă intrare este partajată de convertoarele D/A A și B ($V_{REFA/B}$); cea de-a doua tensiune de referință este utilizată de către celelalte două convertoare, respectiv C și D ($V_{REFC/D}$). Aceste intrări pentru tensiunile de referință, aplicate din exterior, permit obținerea unor game de tensiuni de ieșire diferite pentru cele două perechi de convertoare digital-analogice, A și B pe de o parte, respectiv C și D pe de altă parte.

Circuitul **MAX526** poate fi utilizat în aplicații ce necesită multiplicarea analogică a semnalelor de intrare. Tensiunile de referință pot fi atât tensiuni continue cât și tensiuni alternative. Tensiunea externă aplicată fiecărei intrări V_{REF} determină capătul de scală al tensiunilor de ieșire pentru fiecare pereche de convertoare D/A. Impedanța de intrare prezentată de aceste intrări este dependentă de codul binar aplicat intrărilor digitale ale circuitului. Valoarea minimă a impedanței de intrare, cu valoarea tipică de $6k\Omega$, apare pentru codul binar de intrare cu valoarea 0101 0101 0101. Valoarea maximă a impedanței de intrare, cu valoarea tipică de $60k\Omega$, se manifestă pentru codul binar de intrare cu valoarea 0000 0000 0000. Valoarea impedanței intrărilor pentru tensiunile de referință, dependentă de valoarea codului de intrare binar aplicat circuitului, necesită stabilizarea tensiunilor de referință externe. Este, de asemenea, necesar ca sursele de tensiune externă de referință să fie caracterizate de o impedanță de ieșire cât mai mică, în special la frecvențe mari.

Impedanța minimă de intrare garantată este de $5k\Omega$. Atunci când ambele intrări sunt comandate de aceeași sursă externă de tensiune de referință, impedanța minimă garantată este de $2,5k\Omega$. O primă modalitate de asigurare a unei exactități cât mai ridicate constă în utilizarea unor surse externe de referință cât mai stabile (de exemplu, tensiunea de ieșire a sursei de tensiune de referință **MAX674** ($V_{REF} = 10V$) variază cu maximum 0,33LSB dacă comandă două intrări în loc de una singură. Performanțe mai bune pot fi obținute prin utilizarea unor circuite cu caracteristici de stabilitate superioare, cum ar fi de pildă **MAX670/MAX671**.

O a doua metodă de obținere a unei exactități cât mai ridicate constă în buffer-area cu un amplificator operațional de precizie a sursei de tensiune de referință. Impedanța de ieșire a amplificatorului funcționând în buclă de reacție negativă trebuie să fie mai mică decât $0,05\Omega$. Această valoare a impedanței de ieșire asigură o eroare maximă de 0,08 LSB în cazul în care sunt comandate ambele intrări. Se recomandă utilizarea unui amplificator operațional de precizie

ridicată, de tip **MAX400** sau **OP07**.

Capacitatea intrărilor de tensiune de referință este, de asemenea dependentă de codul binar aplicat intrărilor digitale ale circuitului și variază între 125pF ÷ 300pF.

Ieșirile A/D pot fi reprezentate ca niște surse de tensiune controlate numeric: $V_{OUT_i} = \frac{N_B}{4096} \times V_{REF}$, $i = A \div D$, în care N_B reprezintă valoarea zecimală a codului binar aplicat la intrările numerice ale circuitului $N_B = 0 \div 4095$.

Amplificatoarele interne de tip buffer cu care sunt prevăzute ieșirile celor patru convertoare D/A din cadrul circuitului MAX526 sunt în configurație de repetoare de tensiune și sunt caracterizate printr-un slew-rate tipic de 5V/μs. Aceasta determină ca timpul de stabilire a ieșirii, în limite de ±0,5LSB, pentru o excursie de 10V la ieșire, să fie tipic 3μs. Condițiile în care a fost determinat acest timp sunt caracterizate printr-o sarcină de 5kΩ || 100pF

Intrările digitale ale circuitului MAX526 sunt compatibile TTL și CMOS. Circuitul dispune de o interfață digitală cu un microprocesor, pe 8 biți. Structura de intrare este dublu buffer-ată și este constituită dintr-un registru de intrare de 12 biți (8+4) și un registru corespunzător fiecărui convertor A/D. Tensiunea de ieșire a fiecărui convertor reflectă cuvântul memorat în registrul DAC corespunzător. Liniile de adres A_0 și A_1 sunt utilizate pentru a selecta care dintre convertoare primește date de la magistrală, așa după cum reiese din următorul tabel:

A_1	A_0	REGISTRUL DE INTRARE SELECTAT
L	L	registru de intrare DAC A
L	H	registru de intrare DAC B
H	L	registru de intrare DAC C
H	H	registru de intrare DAC D

Intrările \overline{CSLSB} , \overline{CSMSB} , \overline{WR} permit încărcarea datelor de pe magistrală în registrele de intrare selectate prin intermediul liniilor de adrese A_0 și A_1 . Prin activarea semnalelor \overline{CSLSB} și \overline{WR} se încarcă cei mai puțin semnificativi 8 biți în registrul de intrare. Prin activarea semnalelor \overline{CSMSB} și \overline{WR} se încarcă cei mai semnificativi 4 biți în registrul de intrare. Ordinea de încărcare a datelor (cei mai puțin semnificativi 8 biți sau cei mai semnificativi 4 biți) nu este importantă. Este de asemenea posibilă încărcarea concurentă a tuturor celor 12 biți prin activarea simultană a semnalelor \overline{CSLSB} , \overline{CSMSB} și \overline{WR} . Trebuie, însă, menționat faptul că biții 11÷8 vor fi identici cu biții 3÷0.

$\overline{\text{CSLSB}}$	$\overline{\text{CSMSB}}$	$\overline{\text{WR}}$	$\overline{\text{LDAC}}$	FUNCȚIA
L	H	L	H	încarcă octetul mai puțin semnificativ în registrul de intrare selectat
L	H	↑	H	memorează octetul mai puțin semnificativ în registrul de intrare selectat
↑	H	L	H	memorează octetul mai puțin semnificativ în registrul de intrare selectat
H	L	L	H	Încarcă octetul mai semnificativ în registrul de intrare selectat
H	L	↑	H	Memorează octetul mai semnificativ în registrul de intrare selectat
H	↑	L	H	Memorează octetul mai semnificativ în registrul de intrare selectat
X	X	H	L	Transferă datele din registrele de intrare în registrele DAC
H	H	H	↑	Memorează registrele DAC
H	L	L	L	Încarcă octetul mai semnificativ în registrul de intrare selectat și încarcă registrele de intrare în registrele DAC
↑	X	H	H	nu se efectuează nici o operație; dispozitivul nu este selectat
L	L	L	L	încărcare concurentă 12 biți în registrele de intrare, transferă datele din registrele de intrare în registrele DAC
L	L	L	H	încărcare concurentă 12 biți în registrele de intrare
L	H	L	L	încarcă octetul mai puțin semnificativ în registrul de intrare selectat și încarcă registrele de intrare în registrele DAC
H	H	L	L	Transferă datele din registrele de intrare în registrele DAC
H	H	L	H	nu se efectuează nici o operație.

Datele sunt memorate în registrul de intrare selectat pe frontul crescător al semnalului $\overline{\text{WR}}$.

Datele sunt transferate din registrele de intrare în registrele DAC prin forțarea semnalului $\overline{\text{LDAC}}$ la nivel logic coborât. Se reactualizează simultan toate cele patru convertoare D/A.

Circuitul **MAX526** poate funcționa atât în mod unipolar, cât și bipolar, pentru etajele de ieșire.

Întreaga arhitectură a interfeței de achiziții de date a fost concepută astfel încât să ofere sistemului o flexibilitate cât mai mare, soluții cât mai facile de testare și calibrare și o exactitate cât mai ridicată de măsurare.

6.4 UNITATEA CENTRALĂ DE PRELUCRARE LOCALĂ CU MICROCONTROLLER 80C552

Unitatea centrală de prelucrare locală, organizată în jurul unui microcontroller **80C552**, de fapt a unui sistem de dezvoltare reproiectat, cu microcontroller **80C552**, conferă sistemului avantajul de a putea fi amplasat în imediata vecinătate a procesului controlat, calculatorul central, compatibil **IBM-PC**, putând fi localizat la distanță.

6.4.1 DESCRIEREA FUNCȚIONALĂ A UCPL

Unitatea Centrală de Prelucrare Locală (UCPL) a sistemului de achiziții de date are la bază un sistem de dezvoltare reproiectat, organizat în jurul unui microcontroller **80C552**, realizat în tehnologie **CMOS** de firma **PHILIPS**, compatibil *software* cu familia de procesoare **80C51**.

Bazat pe această structură versatilă și extensibilă, s-a proiectat unitatea centrală de prelucrare a sistemului de achiziții de date. S-a modificat și restructurat sistemul de dezvoltare în sensul creșterii numărului de resurse *hardware*: filtre *antialiasing* programabile, amplificatoare cu câștig reglabil programabile, convertoare analog-digitale interfașabile și programabile, *buffer*-e de date constituite de memorii **FIFO**, interfață serială de tip **RS-485**, registre de comandă și configurare, detectoare ale trecerii prin zero ale semnalelor analizate în scopul determinării frecvenței acestora; s-a extins, de asemenea, numărul de semnale de selecție în conformitate cu resursele *hardware* suplimentare; s-a utilizat un microcontroller de tip **80C552** capabil să funcționeze la o frecvență a ceasului de 30 MHz. S-au păstrat integral, din punct de vedere *hardware* și al configurației în spațiul de adresare, toate resursele sistemului de dezvoltare inițial, ele găsindu-și utilitatea în cadrul sistemului de achiziții de date proiectat.

În fig. 6.12 este prezentată schema bloc detaliată a unității centrale de prelucrare locale, **UCPL**, a sistemului de achiziții de date și de analiză a semnalelor electrice.

6.4.2 RESURSELE UNITĂȚII CENTRALE DE PRELUCRARE LOCALE A SISTEMULUI DE ACHIZIȚII DE DATE

Unitatea centrală de prelucrare locală a sistemului de achiziții de date și de analiză a semnalelor electrice, dispune de următoarele resurse *hardware* și caracteristici tehnice:

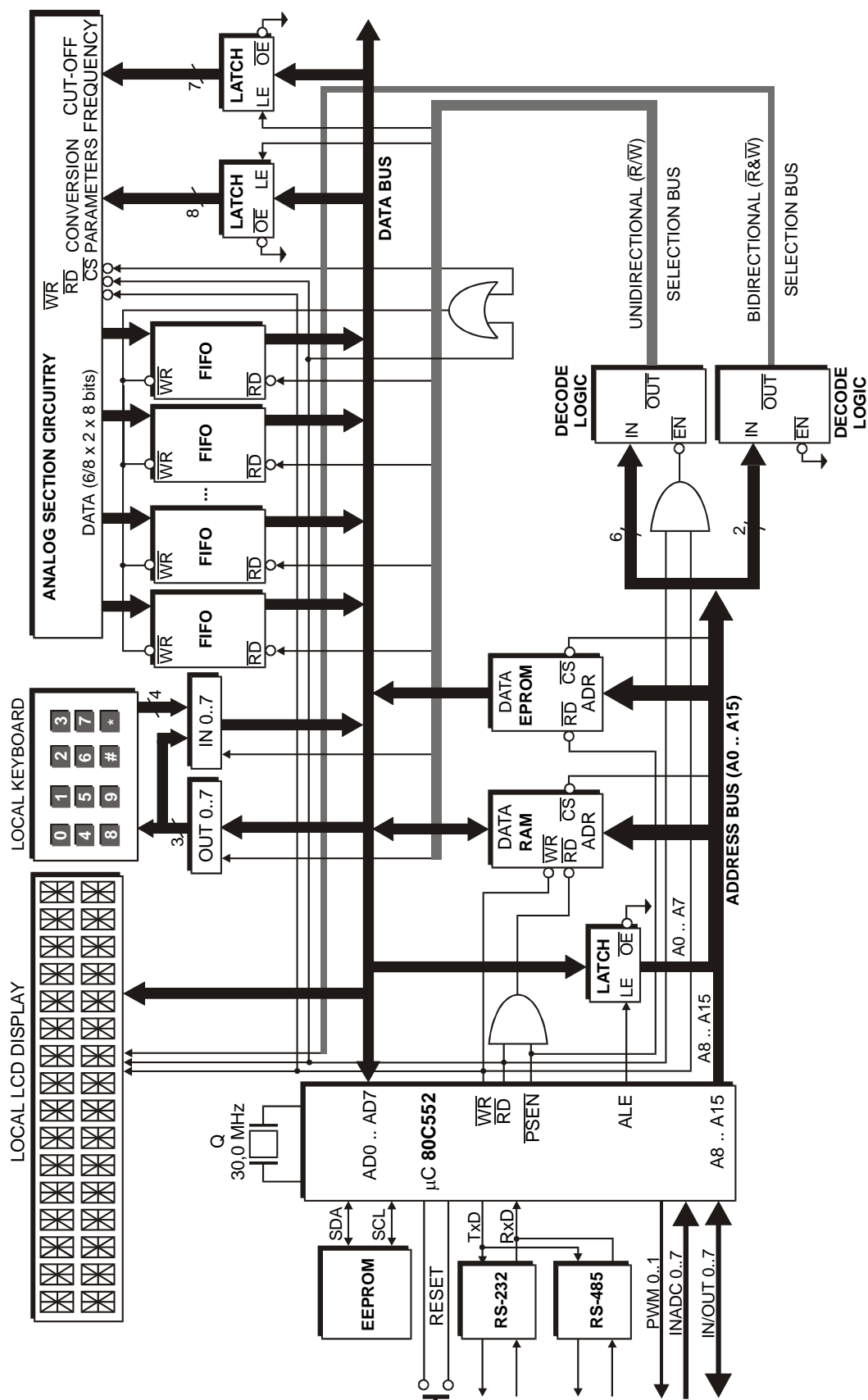


Fig. 6.12 Structura unității centrale de prelucrare locale UCPL.

- microcontroller PCB80C552 (fără memorie internă de program), lucrând la o frecvență maximă a ceasului de 30 MHz;
- frecvența ceasului unității centrale de prelucrare este de 30 MHz;
- memoria de date externă (**DATA MEMORY**), de tip static, implementată cu un circuit **KM62256AL**, realizat în tehnologie **CMOS**, cu capacitatea de 32 kocteți și caracterizat de un timp de acces de 35ns. Spațiul de adresare ocupat de memoria de date este cuprins între 8000H și FFFFH (fig. 6.13);
- memoria de program externă (**PROGRAM MEMORY**), implementată cu un circuit **EPROM** de tip **27C256**, realizat în tehnologie **CMOS**, cu capacitatea de 32 kocteți și caracterizat de un timp de acces de 70ns. Spațiul de adrese ocupat de memoria de program externă este cuprins între 0000H și 7FFFH (fig. 6.13);

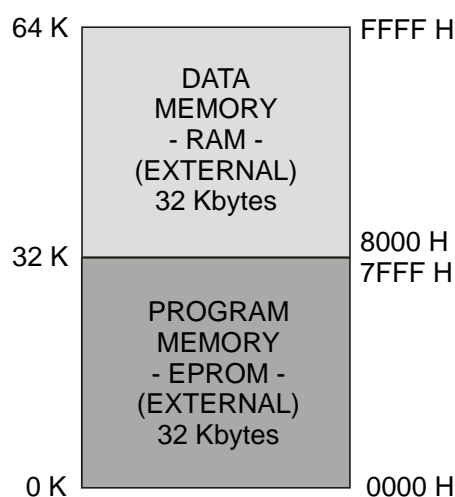


Fig. 6.13 Harta memoriei unității centrale de prelucrare.

- interfață serială compatibilă **RS-232** / **RS-485** de mare viteză, full duplex. Interfața serială de tip **RS-232** folosește protocol *software* de comunicație, fără semnale de dialog, permițând viteze de comunicație cuprinse între 110 bauds și 19200 bauds, iar interfața **RS-485** posedă și semnale de dialog, permițând viteze de comunicație cuprinse între 1200 bauds și 115000 bauds;
- bus serial **I²C** (bus *multimaster* cu arbitrare de priorități și viteză mare de transmisie - 100 kbytes pe secundă în modul *standard* și 400 kbytes pe secundă în modul *rapid* -, frecvența maximă a ceasului serial este 100kHz. Destinația principală este comunicația cu circuite integrate sau controller-e, prevăzute cu interfața **I²C**, aflate în aceeași incintă;
- memorie **EEPROM** serială, implementată cu un circuit de tip **ST24C04**, realizat în tehnologie **CMOS**, cu capacitatea de 512 octeți

și conectată la interfața I^2C . Această memorie nevolatilă conține valorile limită ale parametrilor supravegheați;

- 4 porturi paralele de ieșire de 8 biți (constituite de: registrul de comandă și configurare a parametrilor conversiei, registrul de programare a filtrelor *antialiasing*, registrul de ieșire **OUT 0..7**, registrul de ieșire **OUT 8..15**);
- 17 porturi paralele de intrare de 8 biți (constituite de portul de intrare **IN 0..7** și cele 6/8 perechi de *buffer*-e de date, constituite de memorii **FIFO**, de tip **Am7204A**);
- 2 porturi paralele, bidirecționale, de 8 biți (constituite de ansamblul celor 6/8 convertoare analog-digitale, de tip **MAX181** și de afișajul cu cristale lichide cu două linii de câte 16 caractere, de tip **PVC160205AYL**);
- 8 intrări multiplexate la un convertor analog-digital cu rezoluția de 10 biți, implementat în structura microcontroller-ului 80C552 și caracterizat de un timp de conversie de 50 cicluri mașină;
- 32 ieșiri decodificate de selecție porturi (tabelul 6.1).

Porturile **OUT 0..7** și **IN 0..7** sunt utilizate pentru implementarea tastaturii locale, de tip matriceal, cu trei linii și patru coloane. Liniile sunt baleiate secvențial (sunt active pe "0") prin programarea liniilor 0..2 ale portului **OUT 0..7**, prin intermediul portului de intrare **IN 0..7** se citesc pe liniile mai semnificative codul liniei, iar pe liniile mai puțin semnificative codul coloanei, de pe care s-a acționat o tastă.

Registrul de comandă și configurare a achiziției analog-digitale reprezintă un port de ieșire pentru comanda sistemului de achiziții de date specializat, extern, caracterizat de o rezoluție mai mare și o viteză de achiziție mai ridicată decât a celui implementat în cadrul microcontroller-ului. Selectarea acestui port de ieșire se face cu semnalul **CMDREG (CoMmand REGISTER)**, ocupând un spațiu de adrese între 108H și 10FH. În cazul în care portul suplimentar este utilizat pentru comanda unui sistem de achiziții de date specializat, biții portului au următoarea semnificație:

- bitul 0 (cel mai puțin semnificativ) - comandă de eșantionare-memorare, configurabilă prin program pentru circuitele de eșantionare-memorare implementate în cadrul filtrelor *antialiasing* **MAX 271** sau pentru circuitele de eșantionare-memorare externe, de tip **LF198**;
- bitul 1 - comanda modului de funcționare al convertoarelor analog-digitale **MAX181 (MODE="1")**;
- bitul 2 - comanda modului de preluare a rezultatului unei conversii analog-digitale (**HBEN="0"** - preluare rezultat pe 12 biți);
- bitul 3 - comanda modului de configurare a intrărilor convertoarelor

- analog-digitale (**DIFF**="0" - intrări nediferențiale);
- bitul 4 - comanda modului de configurare a intrărilor convertoarelor analog-digitale (**BIP**="1" - intrările acceptă semnale bipolare);
 - bitul 5 - comanda amplificatorului cu câștig reglabil, **GAIN0** - bitul cel mai puțin semnificativ;
 - bitul 6 - comanda amplificatorului cu câștig reglabil, **GAIN1** - bitul intermediar;
 - bitul 7 (cel mai semnificativ) - comanda amplificatorului cu câștig reglabil, **GAIN2** - bitul cel mai semnificativ.

Tabelul 6.1 Spațiul de adrese pentru selecțiile de porturi.

Linii de adrese $A_0 \div A_{15}$																Ieșire DCD	Adresa (H)
A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0		
0	X	X	X	X	X	X	1	0	0	0	0	0	X	X	X	LCD	100H..07H
0	X	X	X	X	X	X	1	0	0	0	0	1	X	X	X	ADCSEL	108H..10FH
0	X	X	X	X	X	X	1	0	0	0	1	0	X	X	X	CMDREG	110H..117H
0	X	X	X	X	X	X	1	0	0	0	1	1	X	X	X	CUTOFF	118H..11FH
0	X	X	X	X	X	X	1	0	0	1	0	0	X	X	X	OUT 0..7	120H..127H
0	X	X	X	X	X	X	1	0	0	1	0	1	X	X	X	FIFO1L	128H..12FH
0	X	X	X	X	X	X	1	0	0	1	1	0	X	X	X	FIFO1H	130H..137H
0	X	X	X	X	X	X	1	0	0	1	1	1	X	X	X	FIFO2L	138H..13FH
0	X	X	X	X	X	X	1	0	1	0	0	0	X	X	X	OUT 8..15	140H..147H
0	X	X	X	X	X	X	1	0	1	0	0	1	X	X	X	FIFO2H	148H..14FH
0	X	X	X	X	X	X	1	0	1	0	1	0	X	X	X	FIFO3L	150H..157H
0	X	X	X	X	X	X	1	0	1	0	1	1	X	X	X	FIFO3H	158H..15FH
0	X	X	X	X	X	X	1	0	1	1	0	0	X	X	X	IN 0..7	160H..167H
0	X	X	X	X	X	X	1	0	1	1	0	1	X	X	X	FIFO4L	168H..16FH
0	X	X	X	X	X	X	1	0	1	1	1	0	X	X	X	FIFO4H	170H..177H
0	X	X	X	X	X	X	1	0	1	1	1	1	X	X	X	FIFO5L	178H..17FH
0	X	X	X	X	X	X	1	1	0	0	0	0	X	X	X	FIFO5H	180H..187H
0	X	X	X	X	X	X	1	1	0	0	0	1	X	X	X	FIFO6L	188H..18FH
0	X	X	X	X	X	X	1	1	0	0	1	0	X	X	X	FIFO6H	190H..197H
0	X	X	X	X	X	X	1	1	0	0	1	1	X	X	X	FIFO7L	198H..19FH
0	X	X	X	X	X	X	1	1	0	1	0	0	X	X	X	FIFO7H	1A0H..1A7H
0	X	X	X	X	X	X	1	1	0	1	0	1	X	X	X	FIFO8L	1A8H..1AFH
0	X	X	X	X	X	X	1	1	0	1	1	0	X	X	X	FIFO8H	1B0H..1B7H
0	X	X	X	X	X	X	1	1	0	1	1	1	X	X	X	NC	1B8H..1BFH

- 2 ieșiri analogice de 8 biți modulate în durată. Prin integrarea lor se pot obține două convertoare digital-analogice de 8 biți;
- 3 numărătoare de tip *timer* / *counter*;
- 1 *watchdog* programabil (mijloc de autodeblocare în cazul execuției eronate a programelor datorită perturbațiilor sau interferențelor);
- 15 linii de întreruperi dintre care 6 linii externe;
- reset / autoreset (la punerea sub tensiune);
- conectarea directă a unui afișaj cu cristale lichide, cu două linii de câte 16 caractere.

Secțiunea de captură a evenimentelor, din cadrul microcontroller-ului 80C552 este utilizată pentru măsurarea frecvenței semnalelor preluate din cadrul rețelei trifazate analizate.

Utilizând un comparator de tip neinversor, cu tensiune de referință egală cu zero, pot fi detectate trecerile prin zero ale semnalelor de intrare. La trecerea prin zero și începutul semialternanței pozitive a semnalului de intrare (tensiune sau curent), ieșirea comparatorului basculează în “1” logic. Acest front este utilizat generarea unui semnal (cu perioadă dublă raportată la cea a semnalului de intrare), destinat inițializării *timer*-ului T_2 și reprezintă, de asemenea, frontul ce determină captarea conținutului său în registrul CT_0 . Astfel la cea de-a doua captură, registrul CT_0 va conține un număr proporțional cu perioada semnalului de intrare. Rezoluția de 16 biți permite o precizie foarte ridicată de determinare a perioadei semnalelor.

6.5 INTERFAȚAREA UNITĂȚII CENTRALE DE PRELUCRARE, CU MICROCONTROLLER 80C552, CU UN SISTEM HARDWARE EXTERN (INTERFAȚA DE ACHIZIȚII DE DATE)

6.5.1 MODALITĂȚI DE CUPLARE A UNITĂȚII CENTRALE DE PRELUCRARE CU UN DISPOZITIV HARDWARE EXTERN

Unitatea centrală de prelucrare locală prezentată, organizată în jurul unui microcontroller de tip PCB 80C552, poate fi interfațată cu un *hardware* extern, dedicat extinderii facilităților unității centrale de prelucrare.

În ceea ce urmează, se va prezenta modalitatea de cuplare a unui sistem de achiziții de date performant la unitatea centrală de prelucrare locală cu microcontroller 80C552. Cuplarea oricărui sistem *hardware* extern, indiferent de funcția realizată de acesta, poate fi făcută în două moduri:

- direct, prin intermediul secțiunilor de date, adrese și comenzi ale magistralei sistemului de dezvoltare, de fapt ale magistralei (porturilor dedicate acestor scopuri) ale microcontroller-ului 80C552;
- prin intermediul porturilor de intrare/ieșire ale sistemului de dezvoltare, care sunt disponibile în exterior prin elemente dedicate de interconectare.

Prima variantă prezentată, deși conferă avantajul unei versatilități crescute - soluțiile de implementare fiind practic limitate doar de spațiul de adresare al microcontroller-ului -, prezintă dezavantajul major al încărcării magistralei sistemului de dezvoltare. Trebuie precizat faptul că, așa după cum reiese dintr-o analiză atentă a structurii sistemului, liniile magistralei de date, care este multiplexată temporal cu secțiunea inferioară a magistralei de adrese prin intermediul pinilor portului P_0 al microcontroller-ului 80C552, nu permite o soluție simplă de *buffer*-are a acesteia. Această situație se datorează faptului că nu se dispune de un semnal care să indice sensul de manipulare (transfer) al datelor pe această secțiune a magistralei. De aceea, este recomandat ca secțiunea de date a magistralei să nu fie încărcată suplimentar. De asemenea, este foarte important faptul că, datorită amplasării pe plăci diferite a sistemului de dezvoltare și a echipamentului extern de achiziții de date, interconectarea celor două sisteme necesită practic un număr mare de interconexiuni, cu lungimi mari, realizate cu cabluri panglică, care reprezintă liniile magistralei sistemului: secțiunea de date pentru transferul informației în ambele sensuri, secțiunea de adrese pentru decodificarea de porturi și secțiunea de control pentru specificarea tipului de acțiune realizată.

Cea de-a doua variantă enunțată prezintă avantajul minimizării interconexiunilor între cele două sisteme, cel al folosirii unor semnale care sunt *buffer*-ate față de semnalele de pe magistrala internă a sistemului de dezvoltare și cel al utilizării unor conectori externi de interconexiune, care asigură o grupare optimală a semnalelor pe grupuri funcționale.

6.5.2 DESCRIERE FUNCȚIONALĂ A ANSAMBLULUI UNITATE CENTRALĂ DE PRELUCRARE LOCALĂ - INTERFAȚA SPECIALIZATĂ DE ACHIZIȚII DE DATE

În urma unei analize atente a sistemelor de achiziții de date existente pe plan mondial, concluzia care s-a tras a fost aceea că o placă de achiziții introdusă în calculatorul central (de tip *plug-in*) nu reprezintă o soluție optimă, deoarece impune prezența sistemului de calcul în imediata vecinătate a punctului de măsurare și un sistem de calcul dedicat acestei activități.

Amplasarea sistemului gazdă în vecinătatea procesului studiat ridică probleme legate de compatibilitatea electromagnetică. Câmpurile electro-

magnetice intense din punctul de măsurare determină erori tranzitorii în funcționarea sistemului de calcul. Este cunoscut efectul acestor perturbații de foarte scurtă durată asupra funcționării procesoarelor evaluate, înglobând coprocesoare matematice, cum ar fi de pildă generația INTEL 486 DX/DX2/DX4 sau Pentium, sau asupra procesoarelor de semnal DSP: erori în efectuarea calculelor, care conduc de cele mai multe ori la imposibilitatea recuperării datelor procesate.

Soluția propusă elimină aceste dezavantaje, permițând și o cuplare în rețea a mai multor astfel de sisteme de achiziții de date la același calculator central (maximum 4 sisteme de achiziții de date specializate, de acest tip, pot fi cuplate la un calculator gazdă).

Comunicația sistem de achiziții-calculator se face pe o legătură serială full-duplex, compatibilă **RS-232**, viteza de comunicație fiind programabilă *software* și cuprinsă între 110 bauds și 19200 bauds.

În cazul în care câmpurile electromagnetice produc perturbații ce afectează calitatea transmisiei pe canalul serial, se poate efectua transmisia în curent între sistemul de achiziții de date și calculatorul compatibil **IBM-PC**. Transmisia în curent, deși se știe că este mai lentă, prezintă o imunitate foarte ridicată la câmpuri perturbatoare.

Pe linia de comunicație serială calculatorul central transmite sistemului de achiziții rata de conversie, numărul de eșantioane care trebuie prelevate din proces și rata serială de transfer a rezultatelor. La inițializarea procesului de achiziție, atât sistemul de achiziții de date, cât și calculatorul gazdă, demarează comunicația pe viteza de 9600 bauds. Comunicația este inițiată de calculatorul central, sub controlul direct al operatorului uman. Acesta precizează, selectând dintr-o fereastră de meniuri, parametrii achiziției precizați anterior. Rata de transfer a rezultatelor achiziției poate fi crescută până la 19200 bauds, pentru creșterea operativității sistemului, sub acțiunea operatorului uman.

Am optat pentru acest tip de unitate centrală de prelucrare locală deoarece:

- este caracterizată printr-o fiabilitate foarte ridicată și prin consum energetic foarte redus, fiind realizată în tehnologie **CMOS**;
- spațiul de adresare este dublat, deoarece cele două tipuri de memorie externă (memoria de program - 64 kocteți, memoria de date și porturi de intrare-ieșire - 64 kocteți) sunt selectate cu semnale de control diferite;
- setul de instrucțiuni este mai bogat (instrucțiuni de înmulțire și împărțire) și asemănător cu cel al microprocesoarelor tradiționale pe 8 biți;
- viteza de lucru este ridicată, durata tipică a unui ciclu mașină fiind de aproximativ 0,4μs;
- dispune de o interfață **I²C**, de tip multimaster, permițând dezvoltări

ulterioare de sisteme multiprocesor. De asemenea, pe această magistrală serială, caracterizată de o rată ridicată de transfer, poate fi conectată o memorie suplimentară de tip **EEPROM** serial, în care pot fi memorați parametri prestabiliți ai sistemului sau care poate fi folosită pentru identificarea tipurilor de perturbații din rețeaua electrică analizată, înainte de efectuarea prelucrărilor numerice asupra eșantioanelor prelevate din proces;

- unul dintre cele patru porturi ale microcontroller-ului cuprinde semnalele de control specifice funcționării ca microprocesor și semnale de comunicație serială (**RxD**, **TxD**). În acest mod, pentru comunicația cu un alt sistem de calcul, este necesar doar un minim de *hardware*, respectiv circuitele de interfață pentru compatibilizare cu standardul **RS-232**. Aceasta este realizată cu ajutorul unui circuit specializat de tip **MAX232**, circuit care conține o pereche de emițătoare-receptoare, translatoare de nivel **TTL÷RS-232** și **RS-232÷TTL** și două circuite sursă în comutație, asigurând tensiunile de $\pm 10V$ necesare funcționării translatoarelor de nivel. Pentru funcționare, acest circuit necesită doar patru condensatoare externe. Rata serială de transfer poate fi programată între 110 bauds și 19200 bauds, prin utilizarea unuia dintre cele două circuite *timer* / *counter* de 16 biți, implementate intern.

6.6 ESTIMAREA ERORILOR CE SE MANIFESTĂ ÎN CADRUL SISTEMULUI DE ACHIZIȚII DE DATE

6.6.1 ESTIMAREA ERORILOR SOFTWARE

Aplicarea transformatei FFT presupune eșantionarea și digitizarea semnalului analogic de intrare. În funcție de raportul existent între frecvența de eșantionare și frecvența semnalului analizat, se pot distinge:

- *eșantionare coerentă*, când frecvența de eșantionare este un multiplu al frecvenței semnalului (fig. 6.14a);
- *eșantionare necoerentă*, când frecvența de eșantionare nu este un multiplu întreg al frecvenței semnalului (fig. 6.14b).

În cazul *eșantionării coerente*, singurele erori care afectează amplitudinea și faza fundamentalei (A , φ) provin din zgomotul aleator suprapus peste semnalul util, a cărui valoare efectivă a fost considerată $\Delta V_{ef}=25$ LSB pentru $A=5$ V.

Se deduc expresiile dispersiei erorilor ε_A și ε_φ în condițiile unui zgomot de cuantizare care afectează eșantioanele $\{s_k\}$. Generalizând aceste expresii pentru

un zgomot alb oarecare, cu distribuție normală și medie nulă, așa cum poate fi considerat zgomotul de bandă largă care afectează lanțul de achiziție, rezultă:

$$\sigma_A^2 = \frac{\sigma_N^2}{N_s} \quad (6.5)$$

$$\sigma_\varphi^2 = \frac{\sigma_N^2}{A^2}$$

unde σ_N^2 reprezintă dispersia zgomotului.

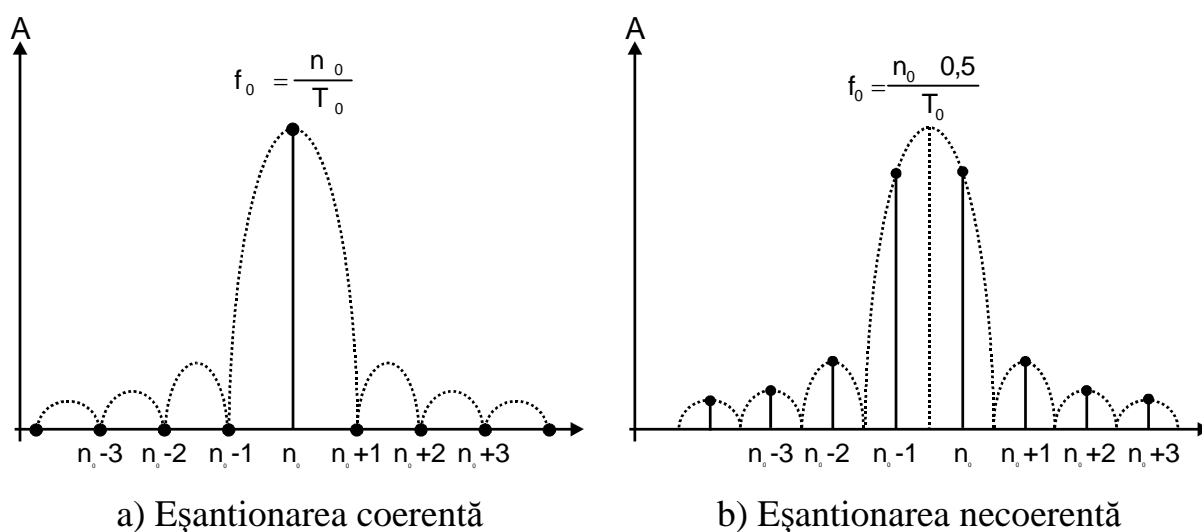


Fig. 6.14 Eșantionarea coerentă și necoerentă.

Înlocuind $\sigma_N = 25\text{LSB}$, $N_s = 160$, $A = 5\text{V}$ rezultă:

$$\sigma_A = 2,79\text{LSB} \quad (6.6)$$

$$\sigma_\varphi = 0,006\text{rad}$$

Erorile relative medii (raportate la capul de scală) rezultă a fi:

$$\sigma_A = 0,006\% \quad (6.7)$$

$$\sigma_\varphi = 0,1\%$$

Aceste valori sunt neglijabile, dar dificultățile aplicării eșantionării coerente în condițiile fluctuației frecvenței rețelei, fac metoda inaplicabilă pentru aplicația propusă.

Eșantionarea necoerentă plasează linia spectrală a fundamentalei în locația cea mai apropiată de poziția adevărată, și prin aceasta conduce la o eroare de estimare a frecvenței de maximum:

$$\Delta f = \frac{1}{2} T_e \quad (6.8)$$

Considerând $N_s = 160$, $f_e = 1,97\text{kHz}$ (ceea ce corespunde la 40

puncte/perioadă) rezultă $\Delta f = 0,7\text{Hz}$, respectiv o eroare relativă de:

$$\varepsilon_f = 1,4\% \quad (6.9)$$

În plus, înregistrarea unui număr neîntreg de perioade ale semnalului de intrare, conduce la “scurgerea” liniei spectrale a fundamentalei în toate locațiile DFT, astfel încât amplitudinea aparentă a fundamentalei va scădea. În cazul cel mai nefavorabil, când frecvența semnalului este la jumătate între două locații succesive, $k \frac{f_e}{N_s}$ și $(k+1) \frac{f_e}{N_s}$, eroarea de amplitudine este:

$$\varepsilon_A = 20 \log \frac{\hat{A}}{A} \approx 4\text{dB} \quad (6.10)$$

în care am notat cu \hat{A} amplitudinea aparentă a fundamentalei.

În general, eroarea comisă la determinarea amplitudinii este aproximativ egală procentual cu abaterea frecvenței de eșantionare de la cel mai apropiat multiplu al frecvenței semnalului. Fixând $f_e = 5000\text{Hz}$ (pentru a putea analiza armonicile până la ordinul 50), dacă frecvența semnalului înregistrează o abatere de $\pm 1\%$ față de valoarea de 50Hz , vom avea o abatere asupra amplitudinii componentelor spectrale de 1% .

Această eroare poate fi redusă până la o limită acceptabilă folosind o tehnică clasică de fereastră, ceea ce conduce însă la pierderea informației despre fază.

Această metodă de estimare este recomandată atunci când coeficientul de distorsiune ce afectează semnalele este mare ($>5\%$), iar fenomenele de nesimetrie sunt neglijabile sau foarte reduse.

6.6.2 ESTIMAREA ERORILOR HARDWARE

În cele ce urmează, se va estima eroarea introdusă de către partea **hardware** a sistemului specializat de achiziții de date, pentru a ne asigura că exactitatea de măsurare a acestui sistem este suficient de ridicată ca să permită determinarea unor coeficienți de distorsiune cu valori de cel puțin un ordin de mărime mai mici decât 1% .

Pentru sistematizarea problemei, se consideră că modulele componente care afectează exactitatea de măsurare a sistemului de achiziții de date sunt:

- *circuitele de adaptare a nivelelor analogice de intrare;*
- *circuitele de tip filtru antialiasing;*
- *circuitele de eșantionare-memorare opționale;*
- *circuitele de conversie analog-digitală.*

Circuitele de adaptare a nivelelor analogice de intrare sunt constituite de elemente pasive (rețele de divizare rezistive). Aceste circuite înglobează, din

punct de vedere al erorilor, și amplificatoarele cu câștig reglabil programabile. Asigurarea unor toleranțe cât mai ridicate (0,001%) și a unor coeficienți de variație cu temperatura de valori cât mai reduse (maxim 10ppm/°C) pentru componentele rezistive, conduce la o eroare maximă:

$$\varepsilon_{\text{CANA}} = 0,0025\% \quad (6.11)$$

pentru o variație a intervalului de temperaturi de funcționare de 50°C.

Pentru *circuitele de tip filtru antialiasing*, din documentația circuitelor **MAX 271**, se poate preciza că eroarea de amplificare maximă (pentru codul de programare cu valoarea 127) este:

$$\varepsilon_{\text{FTJ}} = 0,015\% \quad (6.12)$$

Pentru *circuitele de eșantionare-memorare interne*, implementate în cadrul circuitelor **MAX 271**, pot fi determinate următoarele erori:

- eroarea de câștig, datorată amplificării finite în buclă deschisă a amplificatorului operațional conținut în structura circuitului:

$$\varepsilon_{\text{AO(1)}} = 0,0075\% \quad (6.13)$$

- eroarea de descărcare a condensatorului de memorare pe durata efectuării conversiei analog-digitale:

$$\varepsilon_{\text{DESC(1)}} = \Delta V_{\text{CH}} \frac{T_{\text{C}}}{V_{\text{FS}}} \cdot 100[\%] = 30\mu\text{V} \frac{10\mu\text{s}}{5\text{V}} \cdot 100[\%] = 0,006\% \quad (6.14)$$

în care ΔV_{CH} reprezintă valoarea scăderii tensiunii la bornele condensatorului de memorare, exprimată în $\frac{\mu\text{V}}{\mu\text{s}}$, T_{C} reprezintă durata conversiei analog-digitale, iar V_{FS} reprezintă tensiunea de intrare de capăt de scală;

- eroarea prin *offset*-ul de sarcină asupra tensiunii la care este încărcat condensatorul de memorare, C_{H} :

$$\varepsilon_{\text{QOH(1)}} = 0,02\% \quad (6.15)$$

- eroarea de încărcare a condensatorului de memorare:

$$\varepsilon_{\text{CH(1)}} = 0,02\% \quad (6.16)$$

Eroarea totală maximă a circuitului de eșantionare-memorare va fi:

$$\begin{aligned} \varepsilon_{\text{E/M(1)}} &= \varepsilon_{\text{AO(1)}} + \varepsilon_{\text{DESC(1)}} + \varepsilon_{\text{QOS(1)}} + \varepsilon_{\text{CH(1)}} = \\ &= (0,0075 + 0,006 + 0,02 + 0,02)\% = 0,0535\% \end{aligned} \quad (6.17)$$

Pentru *circuitele de eșantionare-memorare opționale externe*, de tip **LF 198**, din datele de catalog ale producătorului, pot fi extrase următoarele informații:

- eroarea de câștig, datorată amplificării finite în buclă deschisă a amplificatoarelor operaționale conținute în structura circuitului:

$$\varepsilon_{AO(2)} = 0,005\% \quad (6.18)$$

- eroarea de descărcare a condensatorului de memorare pe durata efectuării conversiei analog-digitale:

$$\varepsilon_{DESC(2)} = \frac{I_P}{C_H} \frac{T_C}{V_{FS}} \cdot 100[\%] = \frac{100\text{pA}}{1\text{nF}} \frac{10\mu\text{s}}{10\text{V}} \cdot 100[\%] = 10^{-5}\% \quad (6.19)$$

în care I_P reprezintă valoarea curentului de pierderi prin condensatorul de memorare, C_H reprezintă valoarea capacității de memorare, T_C reprezintă durata conversiei analog-digitale iar V_{FS} reprezintă tensiunea de intrare de capăt de scală;

- eroarea prin *offset*-ul de sarcină asupra tensiunii la care este încărcat condensatorul de memorare, C_H :

$$\varepsilon_{QOH(2)} = \frac{\Delta V_H}{V_{FS}} \cdot 100[\%] = \frac{\frac{C_{DG}}{C_H} \cdot \Delta V_{CDA}}{V_{FS}} \cdot 100[\%] = 0,01\% \quad (6.20)$$

în care C_{DG} reprezintă valoarea capacității drenă-poartă de intrare a circuitului *buffer* din aval de condensatorul de memorare, C_H reprezintă valoarea capacității de memorare, ΔV_{CDA} reprezintă excursia maximă a tensiunii logice de comandă a circuitului de eşantionare-memorare iar V_{FS} reprezintă tensiunea de intrare de capăt de scală;

- eroarea de încărcare a condensatorului de memorare:

$$\varepsilon_{CH(2)} = 0,01\% \quad (6.21)$$

Eroarea totală maximă a circuitului de eşantionare-memorare va fi:

$$\begin{aligned} \varepsilon_{E/M(2)} &= \varepsilon_{AO(2)} + \varepsilon_{DESC(2)} + \varepsilon_{QOS(2)} + \varepsilon_{CH(2)} = \\ &= (0,005 + 10^{-5} + 0,01 + 0,01)\% = 0,025\% \end{aligned} \quad (6.22)$$

În ceea ce priveşte **circuitele de conversie analog-digitală**, în tabelul 6.2 sunt prezentate performanţele sintetice ale circuitului **MAX 181**.

Tabelul 6.2 Performanţele sintetice ale convertorului analog-digital **MAX 181**.

Parametrul Considerat	Valoare		Unitate de măsură
	Tipică	Maximă	
Rezoluţie	12	12	Bit
Tensiune de intrare	-	-2,5 .. +2,5	V
Curent de intrare		1	μA
Curent de referinţă		-2	MA
Eroare de offset	± 1	± 4	LSB
Eroare de amplificare	± 2	± 10	LSB
Eroare de conversie	$0,012\% \pm \frac{1}{2} \text{ LSB}$	-	-

În cadrul aplicației, tensiunea de intrare variază între $-2,5V..+2,5V$, rezultând excursia la capăt de scală a tensiunii $V_{FS} = 5V$, iar $1LSB = 1,22mV$. În aceste condiții, **circuitul de conversie analog-digitală prezintă o eroare maximă de conversie** (care înglobează eroarea de amplificare, de neliniaritate și de cuantificare), având valoarea:

$$\varepsilon_{CAD} = 0,012 + \frac{1LSB}{2 \cdot V_{FS}} \cdot 100[\%] = 0,012 + \frac{1,22mV}{10V} \cdot 100[\%] = 0,0244\% \quad (6.23)$$

Eroarea totală maximă anticipată pentru partea **hardware** a sistemului de achiziții de date, pentru o variație a intervalului de temperaturi de funcționare de $50^{\circ}C$, presupunând că erorile de offset și de amplificare ale lanțului analogic au fost compensate, va fi:

$$\begin{aligned} \varepsilon_{SAD(1)} &= \varepsilon_{CANA} + \varepsilon_{FTJ} + \varepsilon_{E/M(1)} + \varepsilon_{CAD} = \\ &= 0,0025\% + 0,015\% + 0,0535\% + 0,0244\% \approx 0,095\% \end{aligned} \quad (6.24)$$

$$\begin{aligned} \varepsilon_{SAD(2)} &= \varepsilon_{CANA} + \varepsilon_{FTJ} + \varepsilon_{E/M(2)} + \varepsilon_{CAD} = \\ &= 0,0025\% + 0,015\% + 0,025\% + 0,0244\% \approx 0,067\% \end{aligned}$$

Un alt mod de exprimare a erorii maxime totale este determinarea sa în LSB. Se vede imediat că valoarea obținută în ecuația 4.40 corespunde unei erori absolute:

$$\Delta U = \begin{cases} 3LSB; \text{ pentru varianta (1)} \\ 4LSB; \text{ pentru varianta (2)} \end{cases} \quad (6.25)$$

Având în vedere faptul că sursele de erori sunt numeroase și independente, un mod mai realist de a exprima eroarea relativă este ca medie pătratică a erorilor individuale:

$$\varepsilon_{SAD} = \sqrt{\varepsilon_{CANA}^2 + (\varepsilon_{AO}^2 + \varepsilon_{QOS}^2 + \varepsilon_{CH}^2 + \varepsilon_{DESC}^2) + \varepsilon_{CAD}^2} \quad (6.26)$$

Înlocuind cu valorile calculate până acum, eroarea relativă rezultă:

$$\begin{aligned} \varepsilon_{SAD(1)} &= 0,0414591\% \\ \varepsilon_{SAD(2)} &= 0,0324285\% \end{aligned} \quad (6.27)$$

7. SOFTWARE DE ANALIZĂ A SEMNALELOR ELECTRICE

7.1 CONSIDERAȚII GENERALE ASUPRA INSTRUMENTELOR SOFTWARE DE ANALIZĂ A SEMNALELOR ELECTRICE

Instrumentele de măsurare inteligente reprezintă entități independente, separate de un sistem de calcul, capabile să comunice un set redus de parametri și să execute o serie de comenzi. Toate echipamentele de măsurare dezvoltate în ultimii ani conțin interfețe prin care transmit datele și comenzi unor relee cu care sunt echipate.

Indiferent de tipul de mărime măsurată: energie, putere, curent, tensiune, factor de putere, etc., instrumentele inteligente sunt echipate cu relee care comandă direct echipamente externe, în funcție de valorile parametrului măsurat. După gradul de complexitate, pot fi însoțite de un pachet *software* aferent, executabil pe un sistem de calcul compatibil IBM-PC, pentru a putea executa citirile și comenzile la distanță. Foarte multe asemenea sisteme dispun de memorare externă, care stochează variația în timp a parametrului măsurat sau valorile instantanee, la anumite intervale de timp.

Implementarea unui instrument *software* de analiză a semnalelor electrice, provenite din sistemul electroenergetic, presupune utilizarea unui calculator, utilizat ca nucleu *hardware* de comandă, și a unui sistem de achiziții de date, putând fi realizată în mai multe moduri, funcție de sistemul de operare utilizat în cadrul sistemului de calcul.

Suportul *hardware* este format dintr-un lanț de măsurare compus din: senzor, dispozitivul de condiționare a semnalului, placa de achiziții de date și sistemul de calcul aferent. Ceea ce diferențiază aceste instrumente este prezența *software*-ului, un mediu de programare mobil, capabil să proiecteze pe monitorul unui calculator orice parametru măsurat, orice grafic preluat, orice reglaj, buton, etc., transformând practic calculatorul gazdă al aplicației, într-un instrument de măsurare.

Utilizatorul vede această interfață *software* ca o imagine grafică, cu indicatoare, butoane, ideograme (imagini reprezentând acțiuni sau prelucrări specifice), iar cu un ajutorul mouse-ului obține funcția simbolizată de ele: vizualizarea datelor, analize matematice complexe, generare de semnale, citirea datelor de intrare. *Software*-ul de instrumentație dispune de biblioteci bogate, care scutesc utilizatorul de rutină, dar și de dificultăți de programare, cum ar fi comunicarea cu *hardware*-ul, deloc simplă. Ele sunt axate pe următoarele

domenii:

- achiziția datelor și transferul acestora (prin *driver-e software* - programe specializate pentru comanda unor dispozitive sau automate programabile, reglatoare numerice, dispozitive pentru înregistrare);
- controlul dispozitivelor, prin interfațări de tip **GPIO, VXI, RS-232**, etc;
- analiza și reprezentarea datelor.

7.2 PLATFORMA HP VEE PENTRU WINDOWS. PREZENTAREA GENERALĂ A ANALIZORULUI ESA

Software-ul de analiză a semnalelor electrice prelevate dintr-un proces este realizat sub forma unui *instrument virtual*, implementat cu ajutorul programului **HP VEE - for Windows** - versiunea 3.12 (July 07 1995) @ Copyright Hewlett-Packard Corporation 1991-1995.

Acest program lucrează cu obiecte predefinite sau create de către utilizator, care sunt plasate în spațiul de lucru și care sunt interconectate pentru a realiza o diagramă-bloc executabilă. Fiecare obiect permite vizualizarea sa în două moduri:

- modul de vizualizare restrânsă (ca *icon* în **Windows**);
- modul de vizualizare detaliată (*detail view*).

Observație: Orice obiect al programului **HP VEE**, atunci când este apelat, este reprezentat în modul de vizualizare detaliată, câmpurile de parametri (attribute) ale obiectului fiind direct accesibile.

De asemenea, fiecare obiect este caracterizat de un meniu propriu, care permite modificarea dimensiunii, poziției, titlului, precum și altor attribute ale acestuia.

A fost creat *un instrument virtual de analiză semnalelor electrice*, intitulat “**ELECTRICAL SIGNAL ANALYSER**”, prescurtat **ESA**. Acest instrument este, de fapt, un obiect creat de utilizator, ce permite:

- demararea procesului de analiză, prin acționarea butonul “**Run**”, prezent pe panoul global al oricărei aplicații **HP VEE**. La demararea procesului de analiză, este deschisă o fereastră din care poate fi selectat fișierul de date folosit ca punct de plecare în cadrul reprezentării grafice a mărimilor electrice. Fișierele de date sunt de tip standard de date, cu extensia *.dat*, conținând pe câte doi octeți eşantioanele de pe un număr

- de maxim șase canale;
- selectarea mărimilor de intrare prin intermediul unor liste circulare ce dispun de următoarele opțiuni:
 - vizualizarea formei unui semnal de tensiune din cele maxim trei posibile, respectiv de curent;
 - selectarea valorii inițiale a eșantioanelor ce urmează a fi afișate, prin intermediul unui comutator rotativ, denumit **knob**, în gama 0÷20000 și cu o rezoluție de 256 puncte. Acest **knob** poartă denumirea sugestivă de “*From sample...*”;
 - selectarea valorii finale a eșantioanelor ce urmează a fi afișate, de asemenea prin intermediul unui **knob**, gradat între 0 și 20000, cu o rezoluție de 256 puncte. Acest **knob** poartă de numirea sugestivă de “*...to sample*”;
 - vizualizarea simultană a doi parametri: formă de semnal de tensiune, formă de semnal de curent, corespunzând mărimilor de intrare, cu ajutorul a două instrumente de tip osciloscop, care indică:
 - prin intermediul *titlului*, imaginile grafice ce se vizualizează la un moment dat;
 - indicații ale mărimilor corespunzătoare fiecărei axe, cum ar fi: *numele* mărimii (de exemplu: *time* pentru axa Ox, respectiv *amplitudine*, pentru axa Oy), *unitatea de măsură* a acesteia, *intervalul de vizualizare* și *gradarea axelor* (unități pe diviziune);
 - prin intermediul unui **marker**, se pot obține informații, legate de valoarea instantanee a semnalelor, în funcție de timp;
 - instrumentul de vizualizare prezintă facilități de auto-scalare, fie independent pe fiecare axă prin intermediul unor butoane dedicate fiecărui instrument, fie simultan pe ambele axe prin intermediul unui buton cu care este echipat instrumentul de vizualizare de tip osciloscop;
 - afișarea, sub formă numerică, a *valorilor minime și maxime a semnalelor*, atât pentru tensiune, cât și pentru curent, prin intermediul a patru indicatoare alfanumerice.

Trebuie menționat că toate **butoanele** și **knob**-urile, prin intermediul cărora este controlat procesul de achiziție și analiză, prezintă facilități de “*autoexecute*”, ceea ce înseamnă că acționarea oricăruia dintre ele determină declanșarea unui nou proces de analiză.

În fig. 7.1, 7.2 și 7.3 este reprezentat panoul frontal al analizorului în trei situații diferite: vizualizarea formei de semnal de tensiune și de curent pentru semnalele 1 și 2 (fig. 7.1), vizualizarea formei de undă de semnal de tensiune și de curent pentru semnalele 3 și 2 (fig. 7.2), respectiv vizualizarea formei de undă de semnal de tensiune și de curent pentru semnalele 3 și 4 (fig. 7.3).

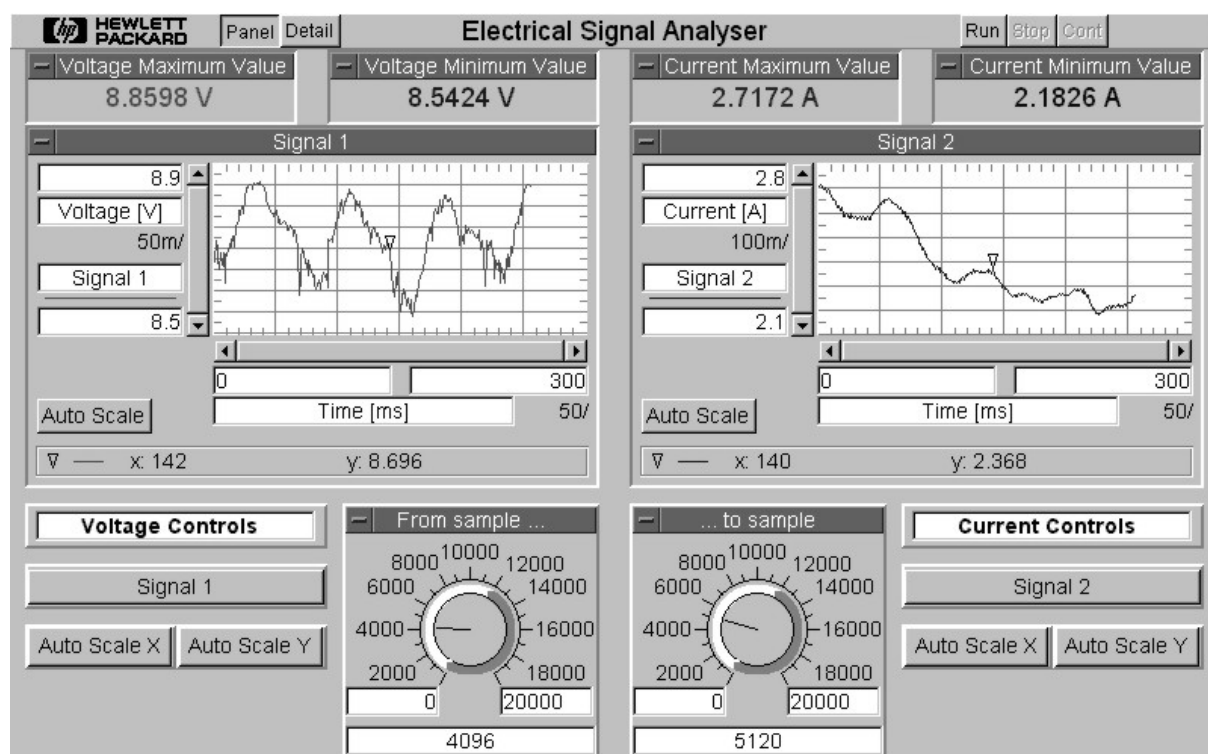


Fig. 7.1 Vizualizarea forme de semnal de tensiune și de curent pentru semnalele 1 și 2.

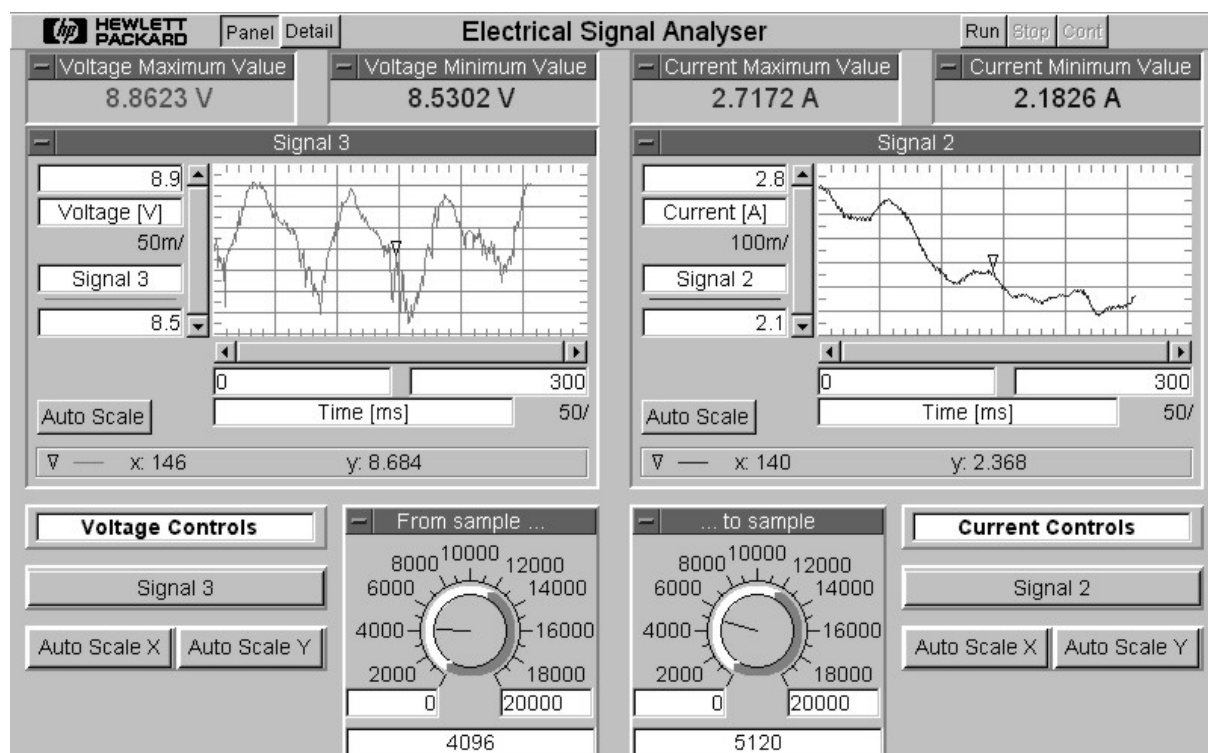


Fig. 7.2 Vizualizarea forme de semnal de tensiune și de curent pentru semnalele 3 și 2.

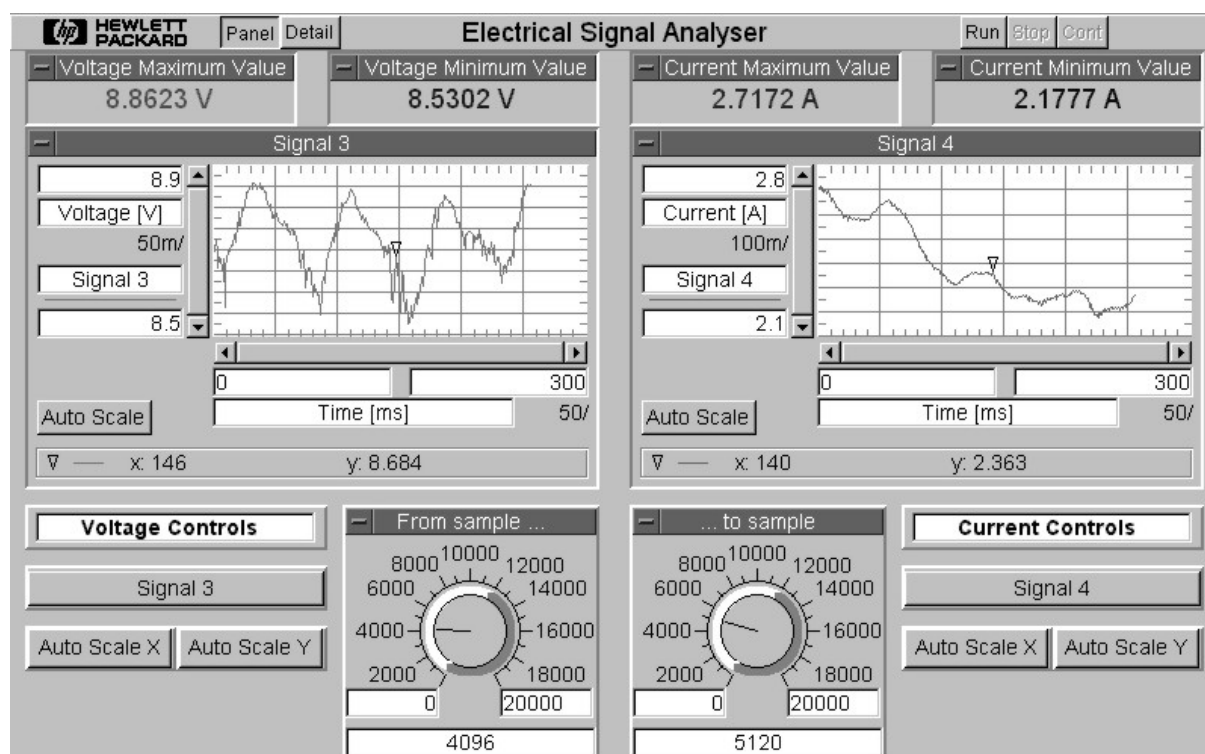


Fig. 7.3 Vizualizarea formei de semnal de tensiune și de curent pentru semnalele 3 și 4.

7.3 IMPLEMENTAREA ANALIZORULUI ESA

Fig. 7.4 conține reprezentarea detaliată a blocurilor care compun analizorul intitulat **ESA** și bazat pe platforma Windows HP VEE 3.12.

Aceste blocuri funcționale sunt:

- *modulul de prelucrare primară a fișierului de date de intrare*, primind ca intrare un fișier cu extensia .dat. Acest bloc furnizează ca ieșiri un număr de maxim șase vectori conținând eșantioanele sub formă numerică a mărimilor de intrare;
- *circuitele de control ale dispozitivelor de afișare*;
- *dispozitivele de afișare*, de tip oscilograf;
- *blocul de afișare sub formă digitală a valorilor minime/maxime ale semnalelor de intrare*.

Toate aceste blocuri funcționale sunt înglobate într-un obiect utilizator nou creat, având două posibilități de reprezentare:

- reprezentarea detaliată - **detail** - (fig. 7.4), în care sunt puse în evidență blocurile și interconexiunile funcționale;
- reprezentarea de tip panou - **panel** -, care ilustrează panoul propriu-zis al analizorului, adică dispozitivul de afișare, reprezentat în fig. 7.5.

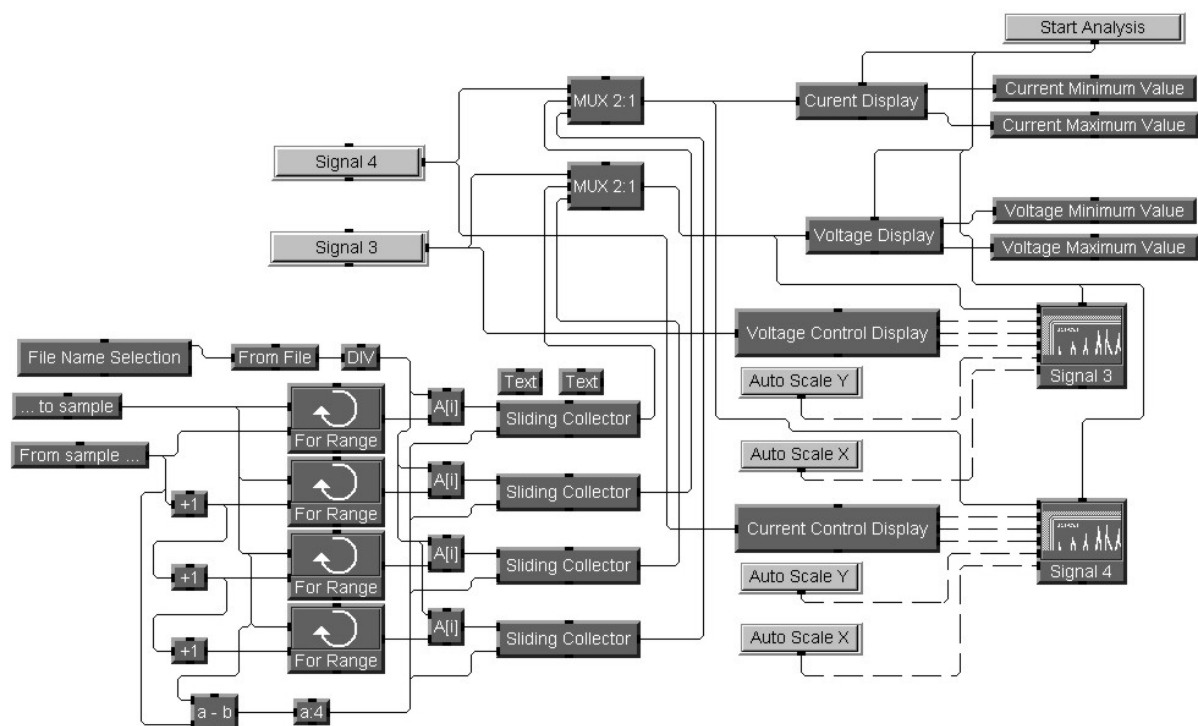


Fig. 7.4 Reprezentarea detaliată a analizorului ESA.

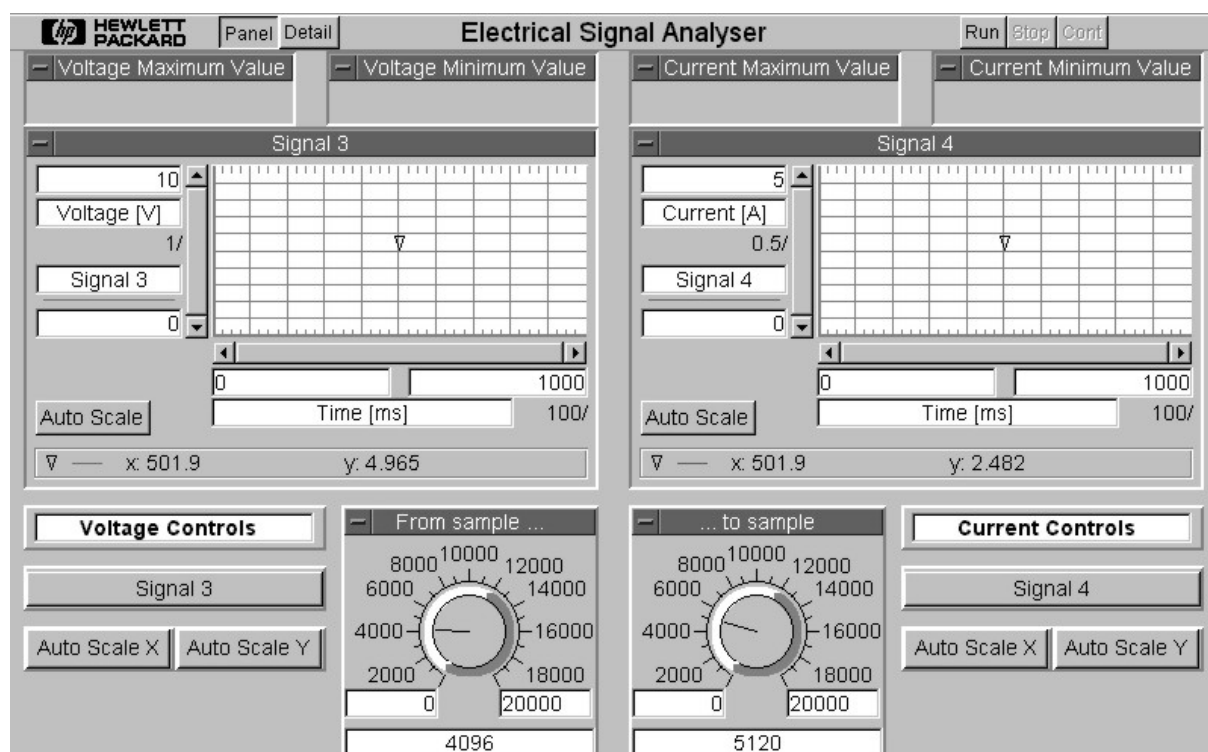


Fig. 7.5 Reprezentarea panoului propriu-zis al analizorului ESA.

7.3.1 BLOCUL DE PRELUCRARE A FIȘIERULUI DE DATE DE INTRARE

Reprezentarea detaliată a blocului de prelucrare a fișierului de date de intrare este ilustrată în fig. 7.6.

În urma declanșării unui proces de analizare a semnalelor achiziționate din proces, prin acționarea butonului **“Run”** de pe panoul analizorului **ESA**, este deschisă o fereastră care permite selectarea fișierului de date de intrare. În această fereastră sunt listate doar fișierele de date (cele care au extensia **dat**). Prin selectarea cu ajutorul mouse-ului a fișierului de date dorit, se demarează procesul de analiză. Fișierul de date conține alternativ eșantioanele pe doi octeți a mai multor semnale de intrare. În vederea reprezentării grafice a unei mărimi, acest fișier trebuie prelucrat.

Eșantioanele, pe câte doi octeți, reprezintă rezultatul conversiei analog-digitale pe 12 biți a unor semnale analogice de tensiune și/sau curent, convertite în gama de tensiune (0÷10) V. În prima fază, trebuie reconstituite valorile analogice ale eșantioanelor. Aceasta este realizată prin împărțirea celor doi octeți, ce reprezintă un eșantion, la valoarea $4096=2^{12}$, și înmulțirea cu 10, reprezentând capătul de scală. În programul implementat se realizează împărțirea la $409,6 = \frac{4096}{10} = \frac{2^{12}}{10}$. Pentru alte capete de scală, poate fi găsit în mod similar un coeficient de scalare corespunzător.

În cea de-a doua fază, trebuie identificați indicii cuvintelor de date din cadrul seriei de valori, obținute în cadrul pasului anterior, corespunzători fiecărei mărimi din cadrul fișierului de date. Dacă în cadrul fișierului de date sunt memorate eșantioanele a patru mărimi analogice de intrare, atunci eșantioanele cu indici de tip $4k$ corespund primei mărimi, eșantioanele cu indici de tip $4k+1$ corespund celei de-a doua mărimi, eșantioanele cu indici de tip $4k+2$ corespund celei de-a treia mărimi, iar eșantioanele cu indici de tip $4k+3$ corespund ultimei (cele de-a patra) mărimi. În mod similar poate fi găsită o strategie pentru identificarea eșantioanelor corespunzătoare în cazul în care fișierul de date conține mai mult de patru mărimi analogice. Extragerea cuvintelor de date din cadrul fișierului prelucrat, corespunzătoare fiecărei mărimi este realizată utilizând un obiect de tip **“For Range”**, care are ca parametri valoarea inițială precizată prin intermediul **knob**-ului **“From sample...”** adunată cu 0, 1, 2, respectiv 3, valoarea finală precizată prin intermediul **knob**-ului **“...to sample”** și incrementul (**“step”**) 4, care calculează indicii eșantioanelor mărimii 0, 1, 2, 3 din cadrul fișierului de date. Se determină apoi elementele sau valorile corespunzătoare indicilor determinați anterior, folosind o funcție utilizator de tip **A[i]**.

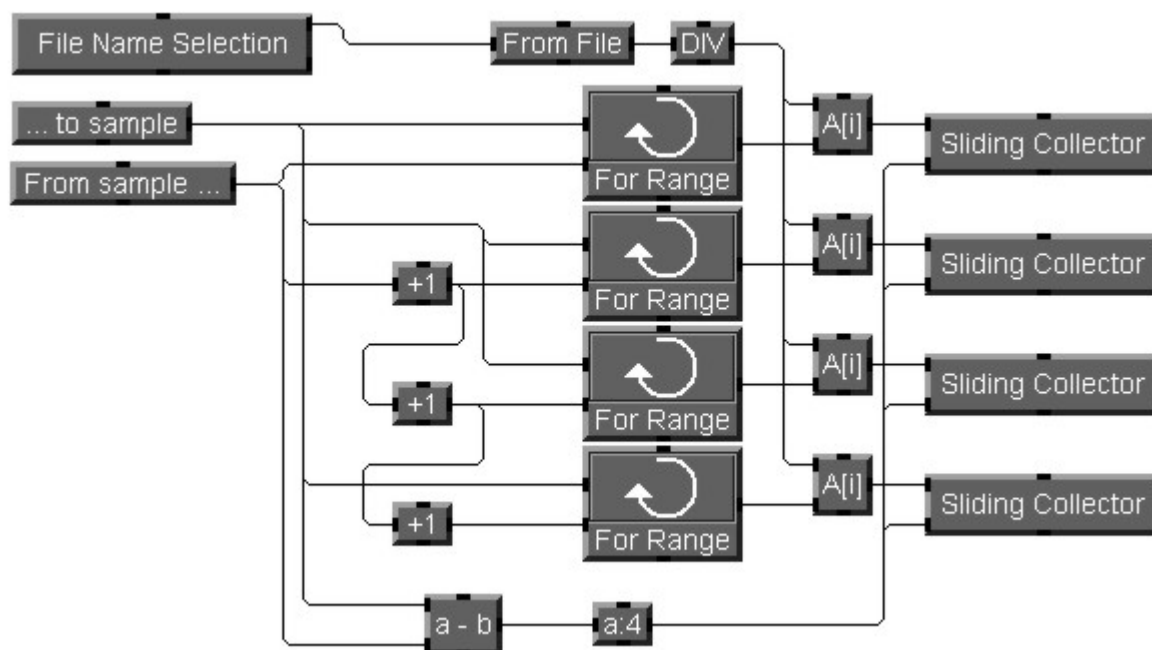


Fig. 7.6 Reprezentarea detaliată a blocului de prelucrare a fișierului de date de intrare.

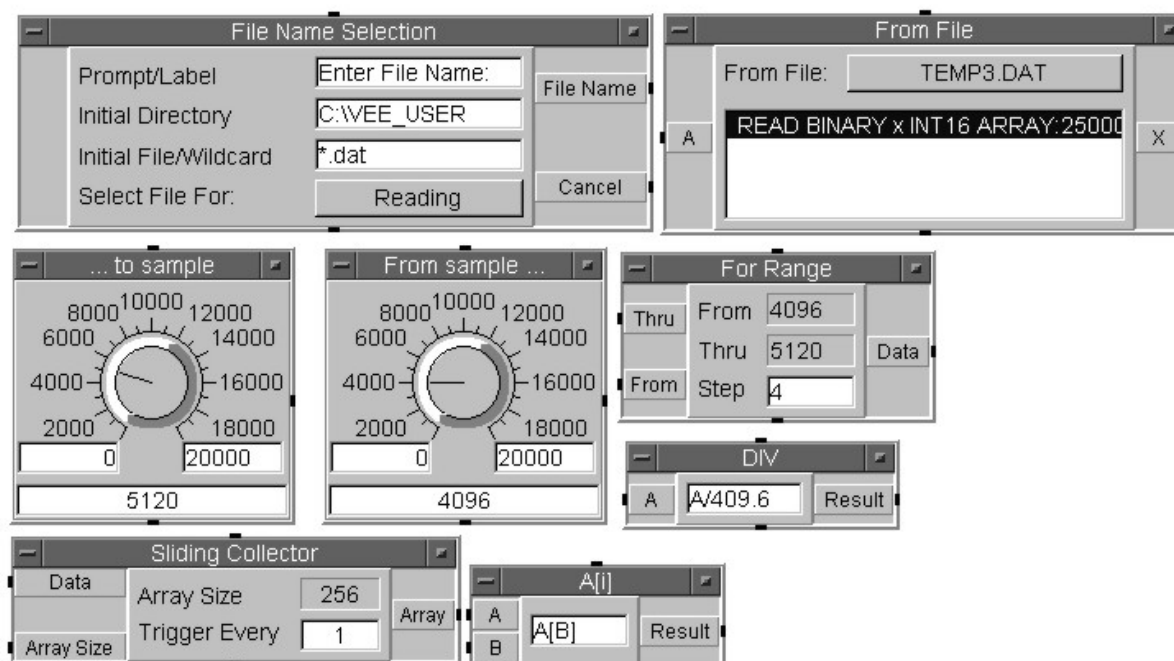


Fig. 7.7 Reprezentarea detaliată a obiectelor utilizate în cadrul modulului de prelucrare a fișierului de date de intrare.

În final se obține câte un vector, cu un număr de elemente egal cu diferența “...to sample” - “From sample...” și divizată cu patru, conținând eșantioanele fiecărei mărimi, folosind un obiect predefinit, denumit “*Sliding*”

Collector".

În fig. 7.7 este ilustrată reprezentarea detaliată a principalelor obiecte utilizate în implementarea blocului de prelucrare primară a fișierului de date de intrare.

7.3.2 BLOCURILE PENTRU CONTROLUL DISPOZITIVELOR DE AFIȘARE

Blocurile pentru controlul dispozitivelor de afișare de tip osciloscop sunt constituite de două obiecte de tip utilizator nou create. Sunt în număr de două, fiind foarte asemănătoare din punct de vedere al implementării. În continuare va fi prezentat în detaliu blocul de control al dispozitivului de afișare pentru tensiuni (fig. 7.8), urmând a se preciza ulterior diferențele comparative pentru blocul de control al dispozitivului de afișare pentru curenți.

Acest modul utilizator este constituit din trei multiplexoare de tip 2:1. Toate cele trei multiplexoare au intrările de selecție comandate prin intermediul unei unice liste circulare dispunând de două opțiuni: "*Signal 1*" și, respectiv "*Signal 3*" (în cazul blocului de control al dispozitivului de afișare pentru curenți există o altă listă circulară, cu opțiunile: : "*Signal 2*" și, respectiv "*Signal 4*").

Primul multiplexor primește ca intrări două constante de tip text, denumite "*Title 1a*" pe intrarea A, respectiv "*Title 1b*" pe intrarea B. Ieșirea acestui multiplexor comandă intrarea de comandă, denumită "*Titles*" a dispozitivului de afișare de tip osciloscop (în cadrul blocului de control al dispozitivului de afișare pentru curenți, primul multiplexor primește ca intrări două constante de tip text, denumite "*Title 2a*" pe intrarea A, respectiv "*Title 2b*" pe intrarea B). În fig. 7.9 este ilustrată reprezentarea detaliată a obiectului de tip multiplexor 2:1 și constantele de tip text.

Cel de-al doilea multiplexor primește ca intrări două constante de tip înregistrare, denumite "*Trace 1a*" pe intrarea A, respectiv "*Trace 1b*" pe intrarea B. Ieșirea acestui multiplexor comandă intrarea de comandă, denumită "*Traces*" a dispozitivului de afișare de tip osciloscop (în cadrul blocului de control al dispozitivului de afișare pentru curenți, cel de-al doilea multiplexor primește ca intrări două constante de tip înregistrare, denumite "*Trace 2a*" pe intrarea A, respectiv "*Trace 2b*" pe intrarea B).

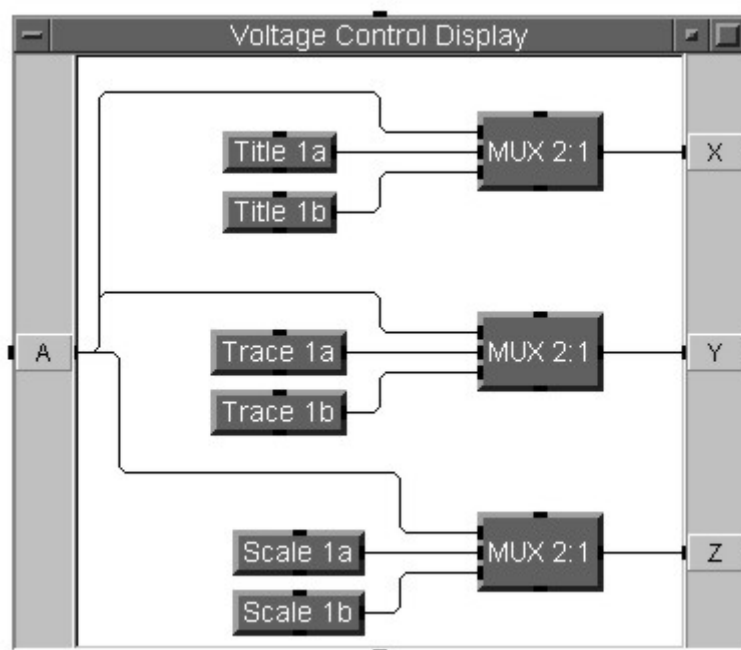


Fig. 7.8 Blocul de control al dispozitivului de afișare pentru tensiuni.

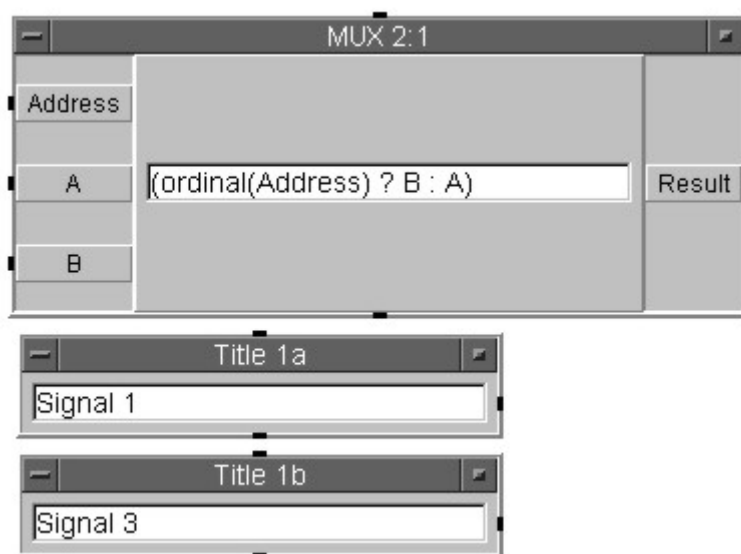


Fig. 7.9 Reprezentarea detaliată a obiectului multiplexor și a constantelor de tip text.

În fig. 7.10 sunt reprezentate constantele de tip înregistrare care constituie întările celui de-al doilea multiplexor.

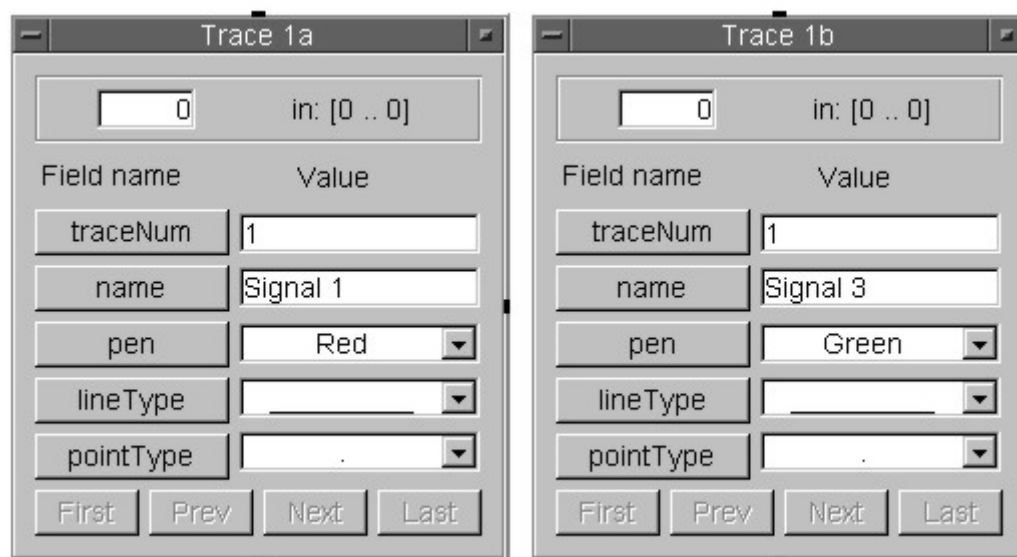


Fig. 7.10 Reprezentarea detaliată a constantelor de tip înregistrare pentru selecția liniei de reprezentare.

Aceste înregistrări de tip înregistrare comandă tipul și culoarea liniei de reprezentare asociată mărimii selectate, precum tipul punctului de reprezentare, dispunând de cinci câmpuri (numărul spotului, implicit 1 pentru că nu se face reprezentarea simultană a două semnale pe ecran; numele semnalului reprezentat, semnal 1 sau semnal 3, pentru tensiune și semnal 2 sau semnal 4, pentru curent; culoarea spotului ce poate fi aleasă dintr-o listă de *tip “pop-down”*; tipul liniei de reprezentare, ce poate fi aleasă dintr-o listă de *tip “pop-down”*; tipul punctului de reprezentare ce poate fi ales dintr-o listă de *tip “pop-down”*). Din analiza figurii 5.10, se constată că pentru fiecare mărime, semnal 1 respectiv semnal 3, sunt asociate culori ale liniilor de reprezentare diferite pentru o identificare rapidă și o diferențiere facilă.

Cel de-al treilea multiplexor primește pe intrările de date două constante de tip înregistrare dublă cu câte cinci câmpuri, denumite “*Scale 1a*” pe intrarea A, respectiv “*Scale 1b*” pe intrarea B (fig. 7.11). Ieșirea acestui multiplexor comandă intrarea de comandă, denumită “*Scales*” a dispozitivului de afișare de tip osciloscop (în cadrul blocului de control al dispozitivului de afișare pentru curenți, cel de-al doilea multiplexor primește ca intrări două constante de tip înregistrare duble, denumite “*Scale 2a*” pe intrarea A, respectiv “*Scale 2b*” pe intrarea B).

Cele cinci câmpuri ale fiecărei înregistrări sunt:

- numele axei de coordonate (X sau Y);
- numele asociat fiecărei axe de coordonate (Time [ms] sau Voltage [V]);
- valoarea minimă de la care începe reprezentarea pe fiecare axă (implicit valoare inițială nulă pentru timp; implicit valoare inițială nulă pentru semnale monopolare de tensiune sau curent);

- valoarea maximă a scalei pe care se face reprezentarea (valoare finală 1000 ms pentru timp; respectiv valoare maximă de 10 V sau 5A pentru tensiune sau curent);
- tipul de scală de reprezentare utilizat (liniară sau logaritmică).

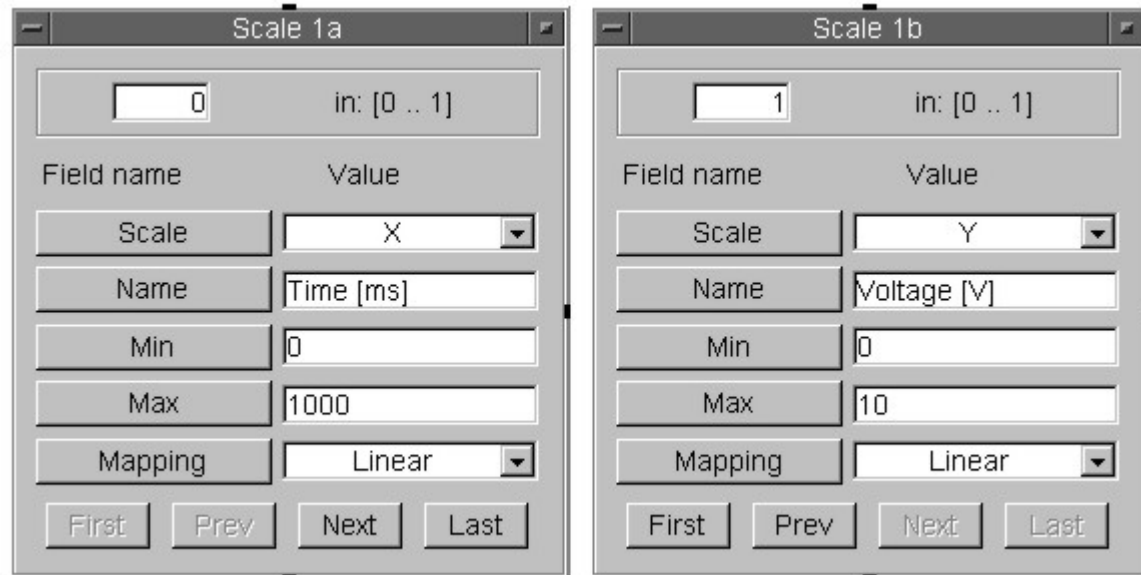


Fig. 7.11 Reprezentarea detaliată a constantelor de tip înregistrare dublă pentru selecția scalei de reprezentare.

Cele două înregistrări duble corespunzătoare fiecărui set de câte două mărimi de același tip sunt de fapt identice, dar sunt folosite ca atare din două motive: în primul rând este necesar ca toate intrările obiectelor folosite să fie conectate pentru ca programul implementat să funcționeze corect, iar în al doilea rând, pentru ca cele trei module bazate pe multiplexare să aibă structuri similare.

7.3.3 DISPOZITIVELE DE AFIȘARE DE TIP OSCIOGRAF

Dispozitivele de afișare de tip osciloscop, reprezentate în fig. 7.12, provin din obiectele, predefinite în cadrul platformei **HP VEE**, cu denumirea “**XY Trace**”, la care au fost adăugate cinci intrări suplimentare:

- intrarea “**Scale**”, destinată predefinirii unor capete de scală pentru cele două axe de coordonate, necesită pentru comandă o constantă de tip înregistrare dublă cu cinci câmpuri, detaliată în capitolul anterior (fig. 7.11);
- intrarea “**Trace**”, destinată predefinirii numelui, culorii, tipului punctului și liniei de reprezentare, necesită pentru comandă o constantă de tip înregistrare cu cinci câmpuri (fig. 7.10);
- intrarea “**Title**”, destinată predefinirii numelui mărimii de intrare care

este reprezentată, necesită pentru comandă o constantă de tip text;

- intrarea “***Autoscale X***”, destinată autoscalării reprezentării grafice pe axa X, necesită pentru comandă un buton, denumit “***Autoscale X***”, cu facilități de *autoexecute*;
- intrarea “***Autoscale Y***”, destinată autoscalării reprezentării grafice pe axa Y, necesită pentru comandă un buton, denumit “***Autoscale Y***”, cu facilități de *autoexecute*.

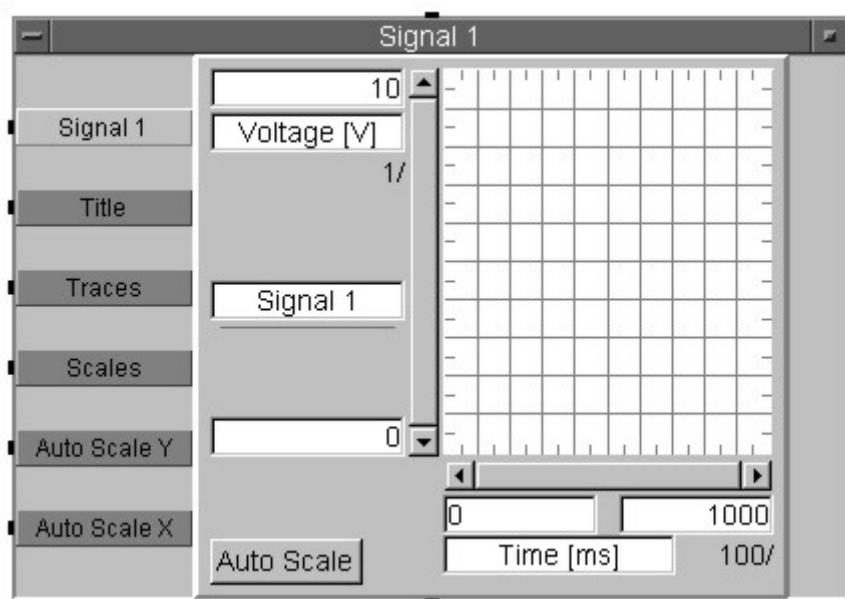


Fig. 7.12 Reprezentarea instrumentului de reprezentare, de tip osciloscop.

7.3.4 BLOCUL DE AFIȘARE SUB FORMĂ DIGITALĂ A VALORILOR MINIME/MAXIME ALE SEMNALELOR DE INTRARE

În cadrul instrumentului virtual creat, în partea superioară a panoului analizorului, sunt afișate valorile minime, respectiv maxime, ale celor două mărimi de intrare reprezentate la un moment dat pe ecran.

Blocul de afișare sub formă digitală a valorilor minime/maxime ale semnalelor de intrare, reprezentat în fig. 7.13, constă din:

- obiectului utilizator ce determină valorile minime/maxime ale mărimilor de intrare, reprezentat în fig. 7.13, compus din:
- modulul de calcul statistic ce determină valoarea minimă din vectorul de eșantioane al semnalului de intrare;
- modulul de calcul statistic ce determină valoarea maximă din vectorul de eșantioane al semnalului de intrare;
- modulul utilizator “***U***”, respectiv “***I***”, ce realizează sumarea dintre

valoarea numerică minimă/maximă determinată din vectorul de eşantioane al semnalului de intrare şi o constantă de tip text, care reprezintă unitatea de măsură a mărimii reprezentate (V sau A). Acest lucru este realizat pentru ca citirea indicaţiilor să fie facilă şi intuitivă. În urma procesului de sumare se obţine o constantă de tip text, în care reprezentarea numărului este efectuată pe 99 de zecimale. Pentru o reprezentare intuitivă, este limitat numărul de caractere de reprezentat la valoarea 6, care include şi punctul zecimal.

Valorile astfel obţinute sunt afişate pe panoul frontal utilizând obiectele predefinite, denumite “*AlphaNumeric Display*”.

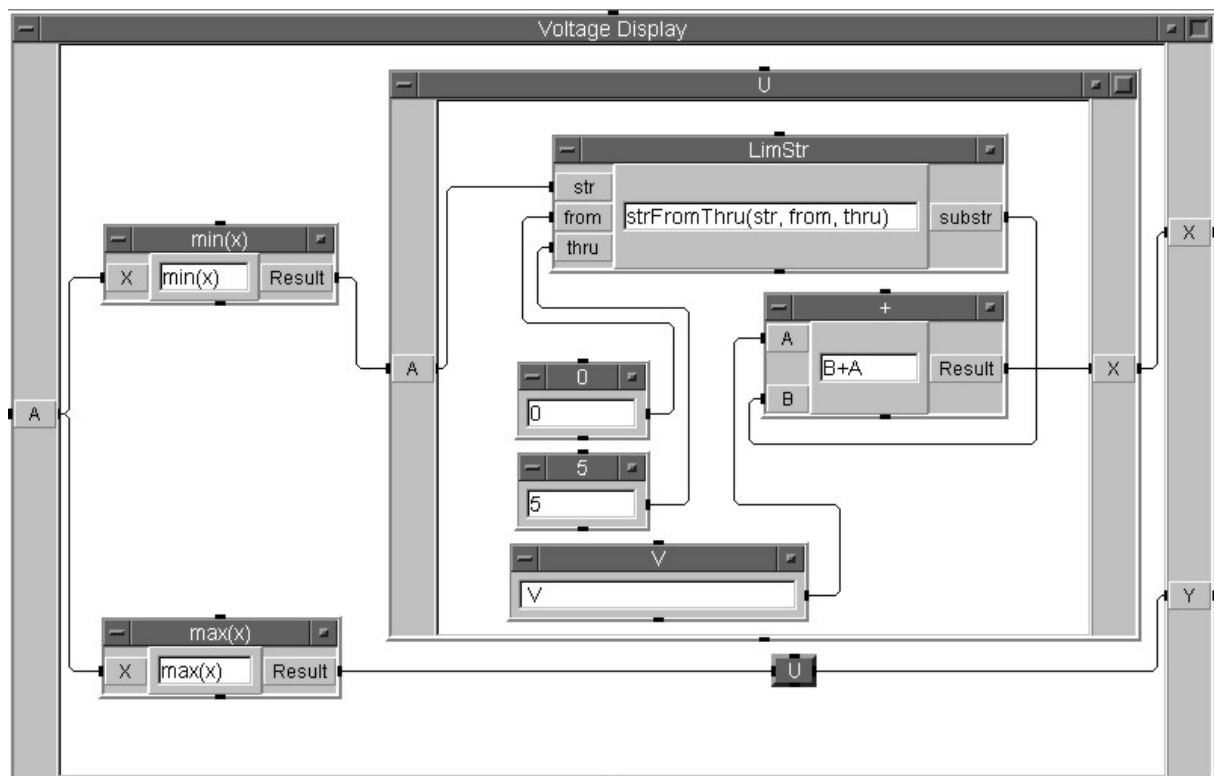


Fig. 7.13 Reprezentarea detaliată a obiectului utilizator ce determină valorile minime/maxime ale mărimilor de intrare.

7.4 INTERFAŢAREA INSTRUMENTULUI VIRTUAL DE ANALIZĂ A SEMNALELOR ELECTRICE, ESA, CU INTERFAŢA SPECIALIZATĂ DE ACHIZIȚII DE DATE

În implementarea propusă, comunicaţia între instrumentul virtual de analiză a semnalelor electrice, **ESA** şi interfaţa specializată de achiziţii de date se realizează prin intermediul a două elemente:

- demararea procesului de achiziţie a tensiunii de fază şi a curentului de

linie se realizează software, prin lansarea în execuție a ferestrei de comandă “**Run PC Program (Dos)**” de sub platforma **HP VEE for Windows**. În câmpurile predefinite ale acestei ferestre, se introduc numele și calea (drive, director, subdirector, etc.) programului de achiziție scris în limbaj de asamblare. La sfârșitul acestei operații, se acționează prin intermediul mouse-ului butonul “**OK**”. Pe durata transferării programului, prin intermediul interfeței seriale, către sistemul de dezvoltare controlul este preluat de programul monitor. După ce programul de achiziție a fost lansat în execuție, controlul este din nou preluat de către instrumentul virtual **ESA**. La terminarea ciclului de achiziție și conversie, sistemul de dezvoltare transmite către calculatorul gazdă, pe care este rezident instrumentul **ESA**, eșantioanele semnalelor achiziționate. Acestea vor fi salvate, la sfârșitul transmisiei, sub forma unui fișier de date, compatibil **HP VEE**. Fișierul de date este accesat de analizorul **ESA** de câte ori se comandă procesul de analiză a semnalelor electrice;

- analiza se efectuează folosind datele din fișierul de comunicație. Există mai multe variante posibile de lucru cu acest fișier:
 - fișierul de date poate fi reactualizat la fiecare proces de achiziție și conversie;
 - fișierul de date poate fi completat cu noi serii de rezultate obținute în urma unui proces ciclic de achiziție și conversie, folosind opțiunea de “**append to file**”;
 - poate fi creată o serie de fișiere, având același nume de bază, dar indicative numerice diferite - de exemplu **filexxx**; fiecare fișier de date din cadrul seriei conține informațiile obținute în urma procesului de achiziție și conversie cu indicativul generic **xxx** (**xxx=0÷999**).

De asemenea, există posibilitatea de a utiliza mai multe formate pentru fișierul de date. Această structură, sau o formă echivalentă, în care datele sunt reprezentate sub formă **hexa**, sunt însă cele mai recomandabile.

Structura fișierului conține un antet care indică: semnificația datelor (“**waveform**” - formă de semnal), numărul de dimensiuni pentru reprezentarea formei de semnal sub formă matriceală (“**numDims=1**” - o singură dimensiune, deci reprezentare vectorială), lungimea structurii de date (“**size=2048**”) și sunt listate, în ordine, elementele fiecărei coloane din matricea de reprezentare, precedate de numărul coloanei (**dim 1**) și de extensia fiecărui element. Elementele sunt separate între ele prin spații (**space**).

7.5 DETALII SUPLIMENTARE PRIVIND IMPLEMENTAREA INSTRUMENTULUI VIRTUAL ESA

Fișierele de date pe care este făcută reprezentarea grafică a mărimilor de intrare (temp.dat, temp1.dat, temp2.dat, temp3.dat) au fost achiziționate cu ajutorul unui sistem de achiziții de date **PLM-16**, produs de firma **National Instruments**, folosind programul de aplicații **NIDAQ** al aceleiași firme. Achiziția de date este cu multiplexare temporală pe patru canale și este caracterizată de o frecvență de eșantionare de 4000 Hz pe toate cele patru canale. Acest lucru ne conduce la concluzia că într-o secundă s-au efectuat 1000 de eșantioane pentru fiecare mărime de intrare, adică informația pentru axa X este exact timpul, exprimat în milisecunde. Dacă frecvența de eșantionare este diferită, trebuie efectuată o scalare a valorilor de pe axa X, pentru exprimarea timpului în milisecunde. Mărimile achiziționate reprezintă forțele tangențiale, respectiv normale, măsurate cu ajutorul unor senzori inductivi de proximitate, într-un sistem pentru determinarea coeficienților de frecare. Din păcate, aceste mărimi nu sunt periodice și nu se poate efectua o analiză armonică a acestor semnale (analiza Fourier, ca mijloc de analiză a semnalelor, se aplică pentru semnale periodice).

Codul sursă al aplicației este generat automat prin amplasarea în spațiul de lucru a obiectelor predefinite și a celor create de utilizator și interconectarea acestora. Singurele modificări manuale în acest cod sunt legate de modificarea unor proprietăți ale modulelor utilizate. Crearea panoului frontal al analizorului, care conține doar instrumentele de afișare și butoanele de control, a fost efectuată prin utilizarea facilității “*Add to Panel*” pentru aceste obiecte, reprezentate în spațiul de lucru în modul “*detail*”.

Deoarece este vorba de un analizor care nu este destinat unei aplicații concrete, specifice, au fost create o serie suplimentară de obiecte utilizator, ce vor fi descrise în capitolul următor, care pot fi apelate în spațiul de lucru și interconectate cu obiectele preexistente, ceea ce conduce la modificarea analizorului pentru focalizarea pe o situație concretă.

7.6 RESURSE SUPLIMENTARE ALE ANALIZORULUI DE SEMNALE ELECTRICE, ESA

Analizorul **ESA** pentru semnale electrice a fost completat cu o serie de obiecte utilizator nou create, pentru extinderea facilităților acestuia.

În primul rând, a fost creat un nou obiect utilizator, denumit “*Auto_XY*”, care reprezintă un instrument de afișare de tip osciloscop care efectuează automat autoscalarea pe ambele axe de coordonate. Pornind de la acest obiect, a

fost creat un obiect utilizator, denumit “**2Auto_XY**”, capabil să afișeze grafic, prin multiplexare, două mărimi de intrare de același tip.

În fig. 7.14 este ilustrată schemă bloc a analizorului **ESA** ce utilizează acest instrument de afișare. Schema bloc funcțională a analizorului pentru semnale electrice este, în acest caz, mult mai simplă, având în vedere înglobarea tuturor obiectelor necesare în vederea reprezentării grafice în modulul obiect nou creat.

În fig. 7.15 este reprezentat panoul frontal al analizorului **ESA** ce utilizează acest nou instrument de afișare. Se constată că panoul frontal, în situația folosirii unor instrumente de afișare cu facilități de autoscalare încorporate, este asemănător cu cel prezentat anterior.

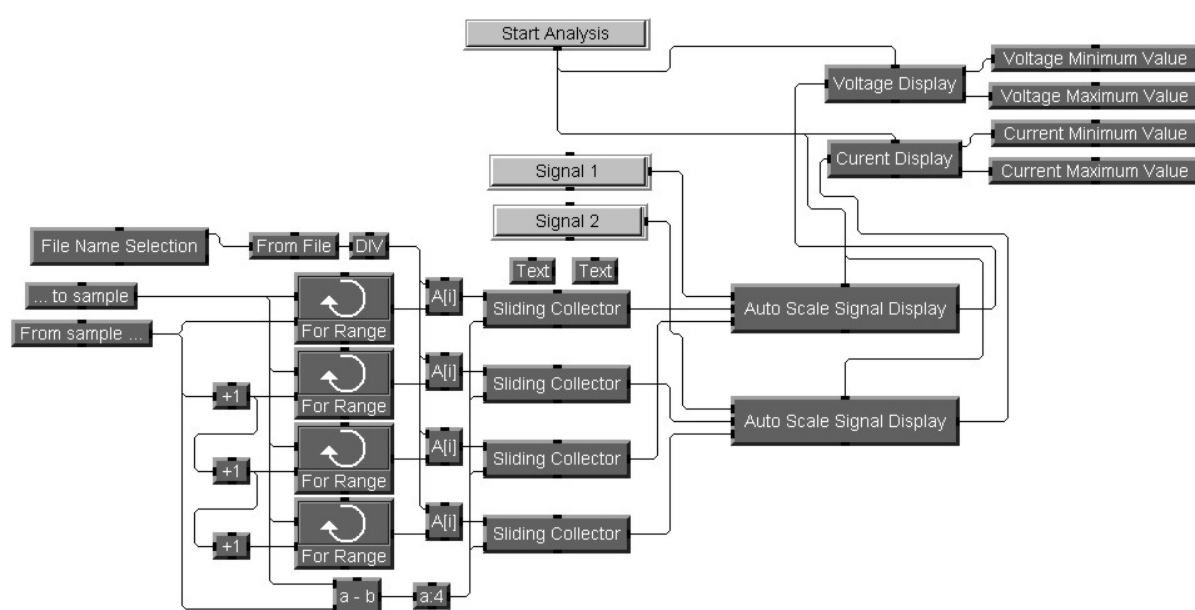


Fig. 7.14 Reprezentarea detaliată a analizorului ESA.

Modulul obiect denumit “**2Auto_XY**”, a cărui reprezentare de tip “*panel*” este prezentată în fig. 7.16, permite vizualizarea prin multiplexare a două mărimi de intrare de același tip (tensiuni sau curenți), selecția efectuându-se prin intermediul unei liste circulare cu două opțiuni, de tipul celor prezentate anterior.

În fig. 7.17 este figurată reprezentată reprezentarea detaliată a modulului obiect denumit “**2Auto_XY**”, acesta compunându-se din două blocuri:

- obiectul utilizator denumit “**Display Control**”, care înglobează toate elementele legate de autoscalare, titlu, tipul, culoarea punctului și liniei de reprezentare;
- un modul obiect predefinit, descris anterior, de tip “**XY Display**”, la care au fost adăugate intrările de comandă “**Scales**”, “**Traces**” și “**Title**”.

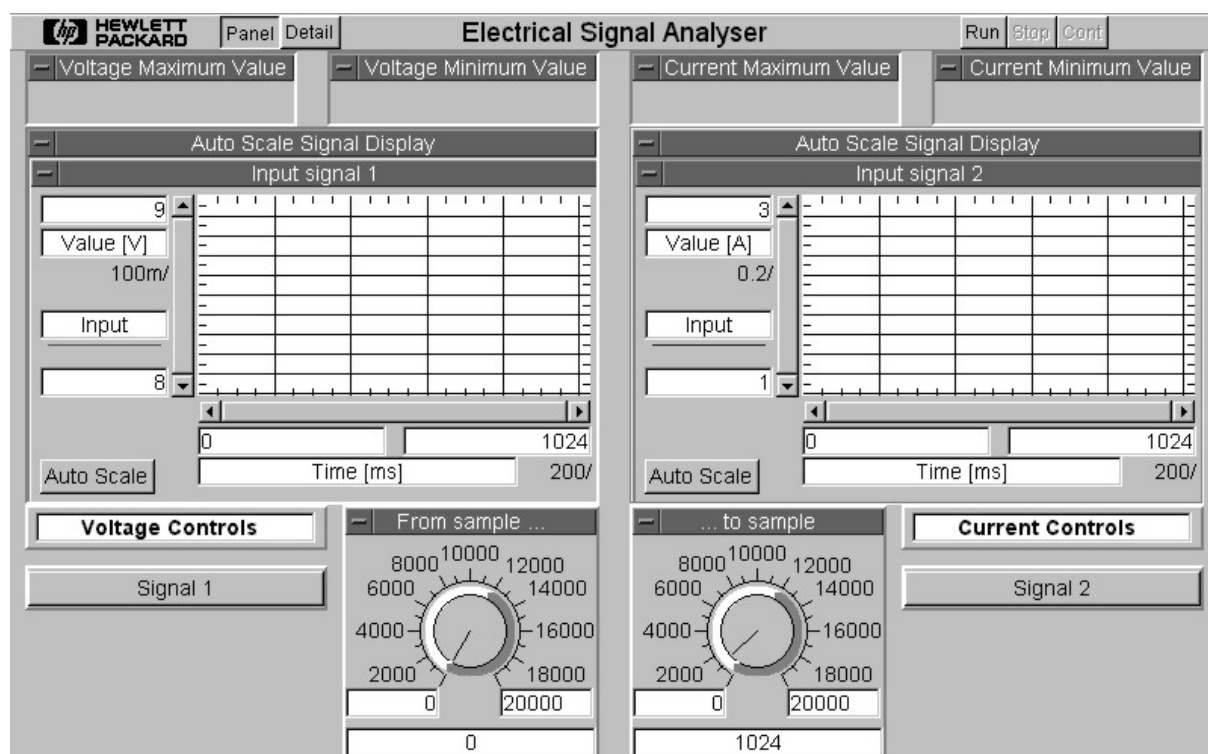


Fig. 7.15 Reprezentarea panoului analizorului ESA.

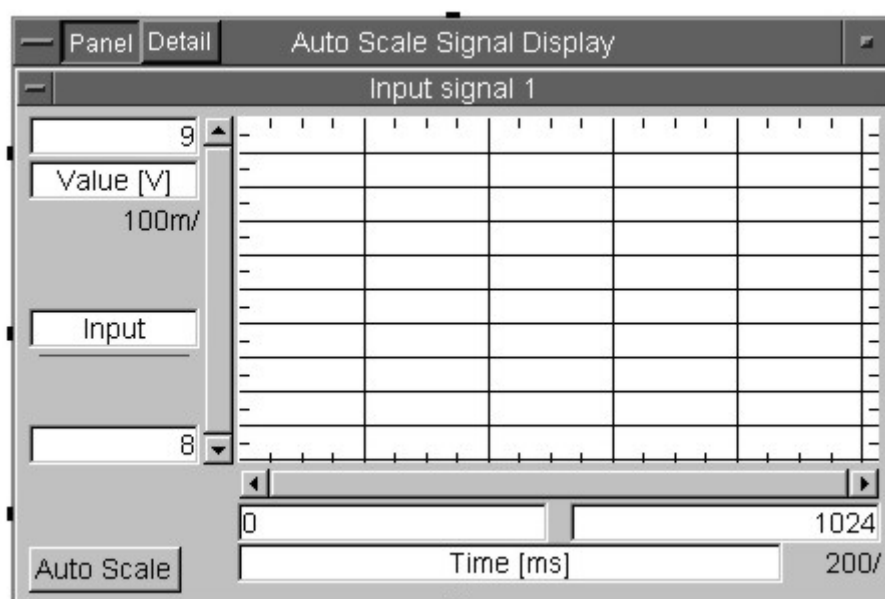


Fig. 7.16 Reprezentarea noului instrument de afișare cu autoscalare.

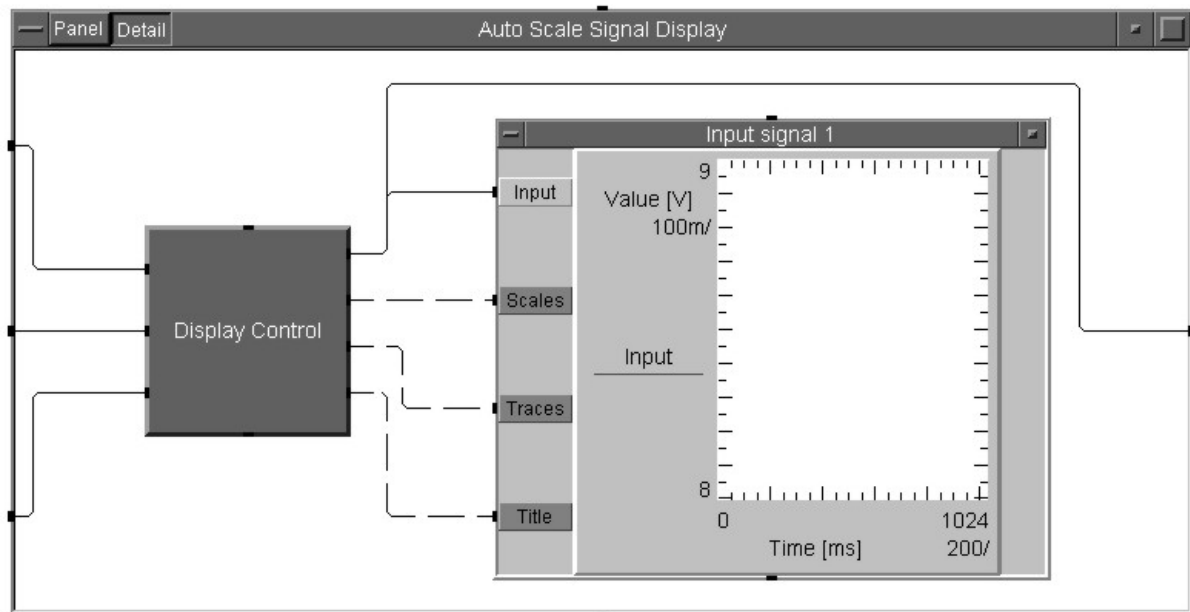


Fig. 7.17 Reprezentarea detaliată a noului instrument de afișare cu autoscalare.

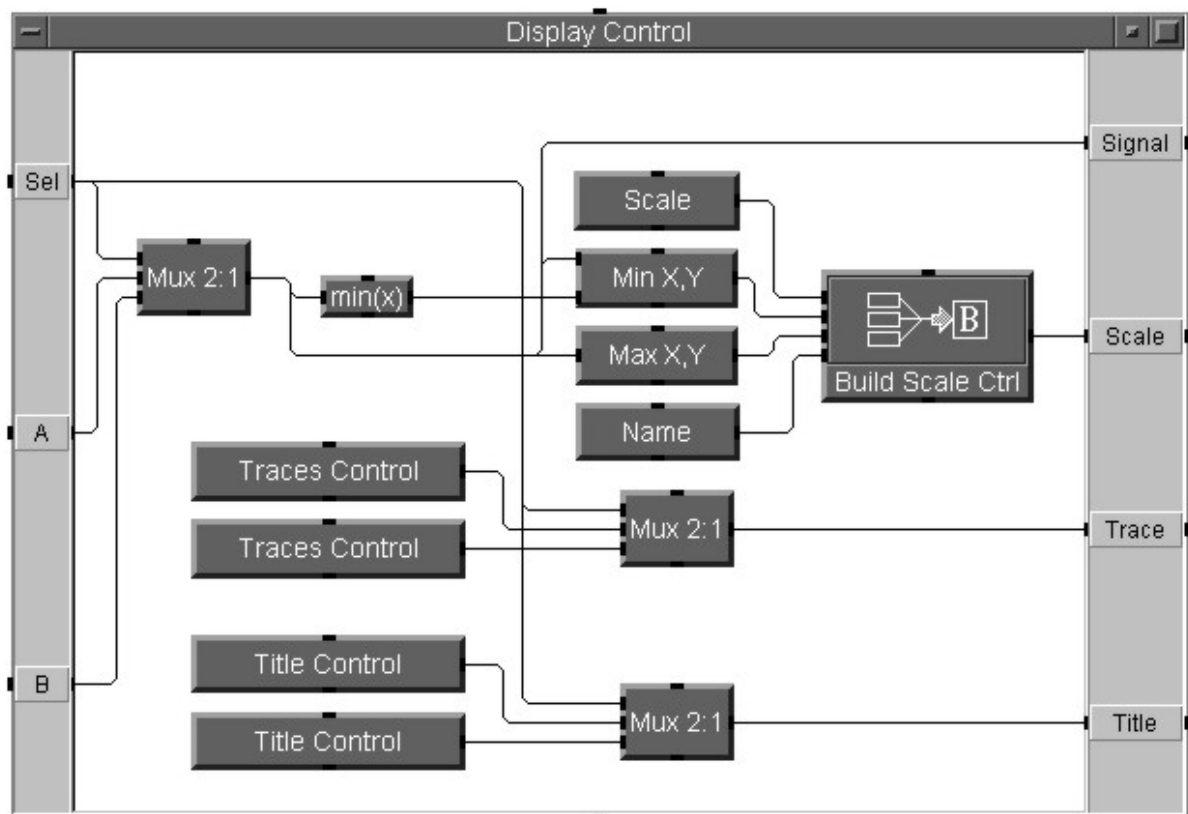


Fig. 7.18 Reprezentarea detaliată a modulului “*Display Control*”.

În fig. 7.18 este figurată reprezentarea detaliată a obiectului utilizator nou creat, denumit “*Display Control*”, care aduce elementele de noutate (autoscalare pe ambele axe de coordonate prin mijloace proprii, fără a folosi butoane de

autoscalare sau facilitățile mijlocului de reprezentare grafică.

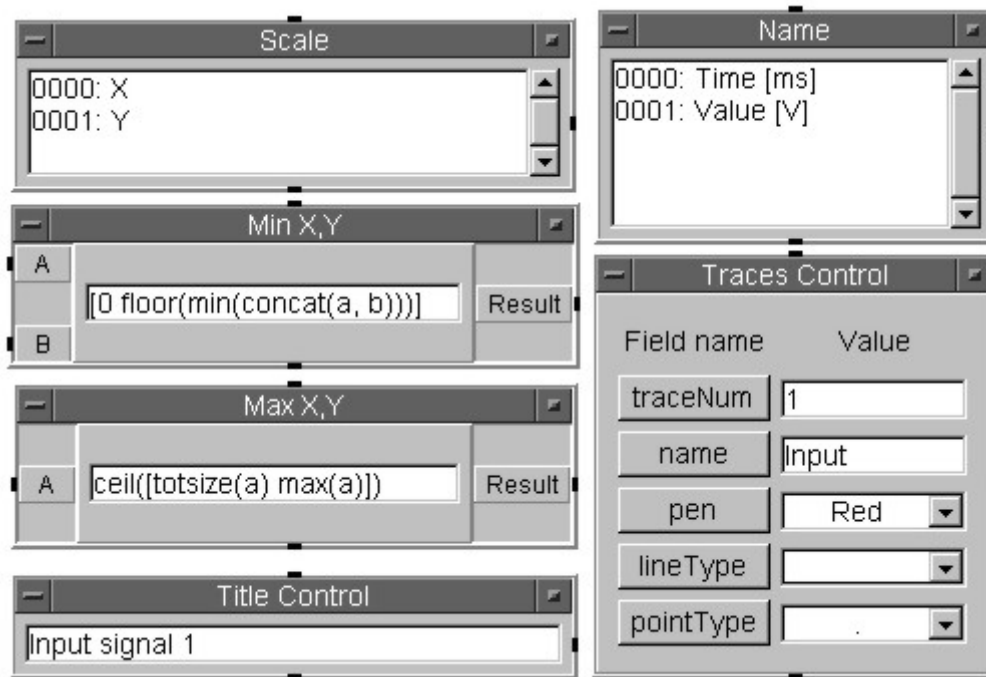


Fig. 7.19 Reprezentarea blocurilor componente ale modului “*Display Control*”.

În fig. 7.19 sunt prezentate blocurile componente ale obiectului “*Display Control*”, și anume:

- circuitul de calcul al valorii minime pentru axele de coordonate, denumit “*Min X,Y*”;
- circuitul de calcul al valorii maxime pentru axele de coordonate, denumit “*Max X,Y*”;
- constantele de tip text pentru scală și nume (“*Scales*”, “*Name*”);
- constanta de tip înregistrare cu cinci câmpuri pentru reprezentarea grafică pe ecran.

7.6.1 ANALIZORUL FOURIER

Reprezentarea detaliată a analizorului Fourier dublu este ilustrată în fig. 7.20.

Semnalelor de intrare, de tensiune și de curent, li se aplică ferestre de eșantionare de tip dreptunghiular, Hanning, Blackman sau Bartlet. Aceste ferestre de eșantionare, sub forma lor temporală, sunt descrise de ecuațiile (7.1), (7.2), (7.3) și (7.4) și sunt caracterizate printr-o durată de 8 perioade ale semnalelor de intrare de frecvență fundamentală:

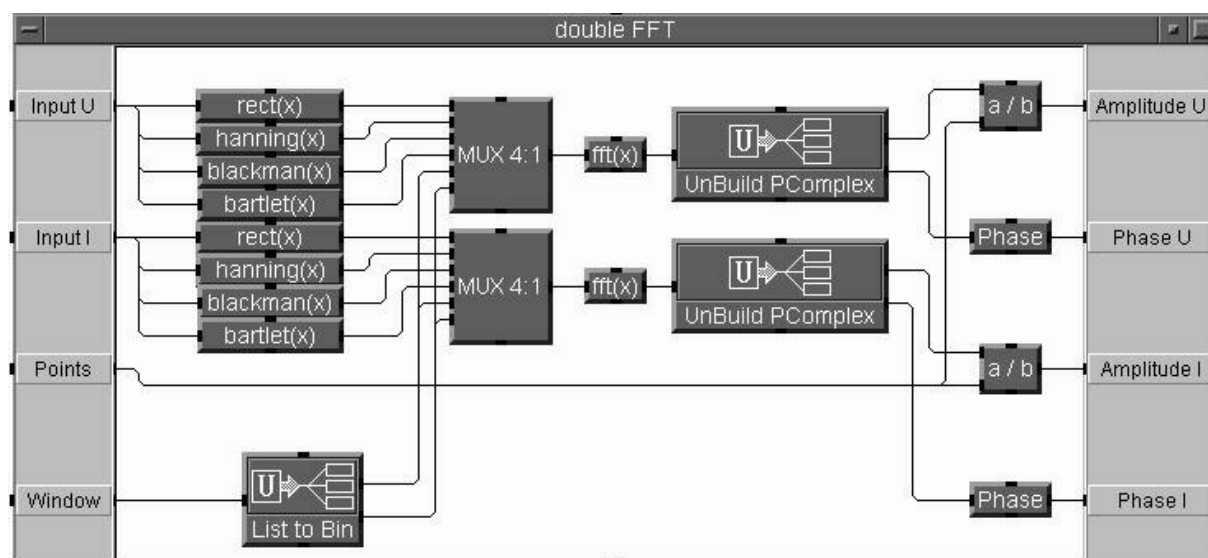


Fig. 7.20 Structura analizorului Fourier dublu (reprezentare detaliată).

- fereastră de eşantionare **Dirichlet**, descrisă de următoarea ecuație:

$$w(n) = \begin{cases} 1, n = 0, 1, \dots, N-1 \\ 0, n \geq N \end{cases} \quad (7.1)$$

- fereastră de eşantionare **Hanning**, descrisă de următoarea ecuație:

$$w(n) = 0,5 \left[1 + \cos\left(\frac{2\pi n}{N}\right) \right]; n = 0, 1, \dots, N-1 \quad (7.2)$$

- fereastră de eşantionare **Blackman**, descrisă de următoarea ecuație;

$$w(n) = 0,42 - 0,5 \cos\left(\frac{2\pi}{N} n\right) + 0,08 \cos\left(\frac{2\pi}{N} 2n\right); n = 0, 1, \dots, N-1 \quad (7.3)$$

- fereastră de eşantionare **Bartlett**, descrisă de următoarea ecuație:

$$w(n) = 2 \cdot \frac{(-1)^k \cdot \frac{n}{N-1}}{N-1}; n = 0, 1, \dots, N-1; k = \begin{cases} 0; \text{daca } n < \frac{N}{2} \\ 1; \text{daca } n > \frac{N}{2} \end{cases} \quad (7.4)$$

Toate aceste ferestre de eşantionare au fost implementate ca funcții utilizator (**user function**), conform ecuațiilor de definiție prezentate anterior.

Obiectele “**ferestre de eşantionare**” au fost implemmentate ca funcții utilizator (**user function**), la fel ca și obiectul “**algorithm FFT**” (fig. 7.21).

Rezultatele înmulțirii, în domeniul timp, ale semnalelor de intrare cu ferestrele de eşantionare sunt multiplexate utilizând un multiplexor de tip 4:1. Acest multiplexor este un obiect utilizator nou creat, folosind multiplexorul de tip 2:1 implementat în cadrul platformei **HP VEE**. Intrările de selecție ale multiplexorului sunt comandate de către un obiect denumit “**List to Bin**”, nou creat. Acest obiect convertește conținutul unei liste de intrare, de tip text, într-un

cuvânt binar de doi biți, reprezentând poziția în listă a ferestrei de eșantionare selectată.

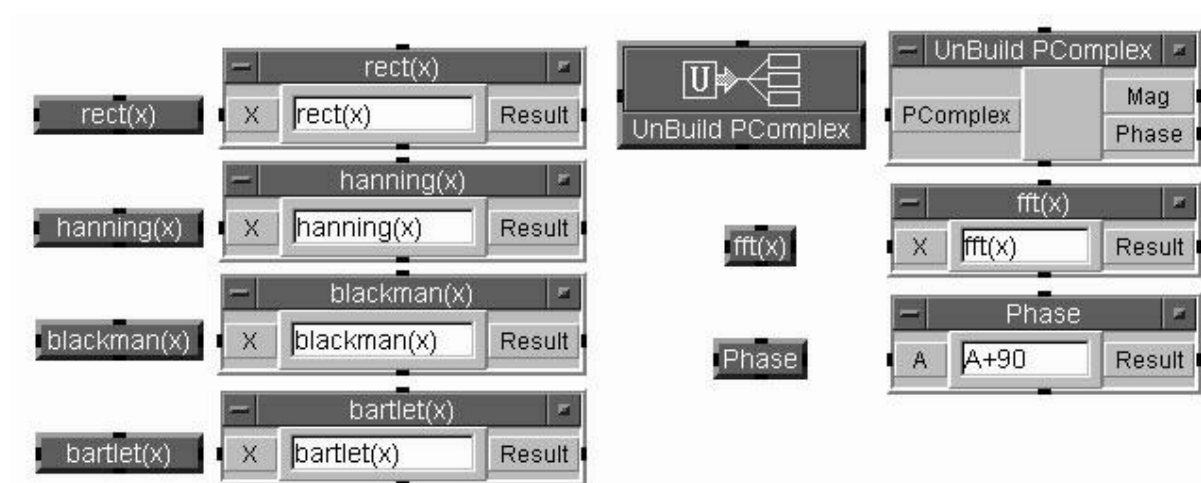


Fig. 7.21 Funcțiile utilizator utilizate în implementarea analizorului Fourier dublu.

Acest proces de conversie text-cuvânt binar este realizat, conform figurii 7.22, prin compararea titlului reprezentând numele ferestrei de eșantionare din listă cu patru înregistrări de tip text, predefinite și realizarea unei funcții cablate (**JCT**) între ieșirile celor patru comparatoare. Rezultatul funcției cablate **JCT** indică, în format zecimal, poziția textului selectat în lista de intrare. În continuare, se realizează convertirea numărului zecimal obținut în formă binară, prin utilizarea funcției **bit(x,n)** ce furnizează ca rezultat bitul de pe poziția n a variabilei de intrare x .

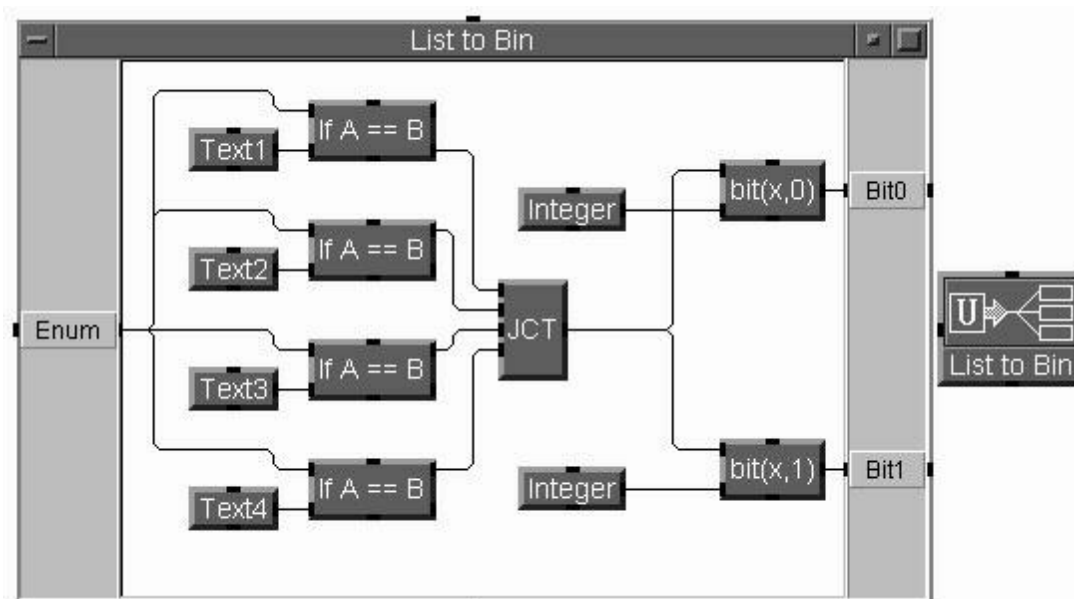


Fig. 7.22 Reprezentarea detaliată a obiectului utilizator “List to Bin”.

Prin selectarea unui anumit tip de fereastră de eşantionare, semnalelor de intrare multiplicare cu fereastra aleasă li se aplică un algoritm de transformare Fourier redus, cu decimare în frecvență.

Conform figurii 7.21, obiectul “**FFT**” este o funcție utilizator. Rezultatul transformării **FFT** este un vector de numere complexe, fiecare element al vectorului conținând informații referitoare la amplitudinea și faza inițială a armonicii corespunzătoare. Utilizând funcția “**UnBuild PComplex**”, se determină mărimile anterior menționate. Trebuie menționat că trebuie efectuată o corecție a fazei armoniilor pentru a fi reprezentată pe intervalul $[0, +\pi]$, deoarece în urma calculelor, faza se încadrează în intervalul $[-\pi, +\pi]$. De asemenea, calculul amplitudinilor fiecărei armonicii necesită împărțirea modulului numărului complex corespunzător din vectorul transformării **FFT** la lungimea acestuia.

În concluzie, soluția de implementare utilizând fișiere de comunicații pentru date, respectiv pentru comenzi, crește productivitatea utilizării instrumentului virtual de analiză a regimului deformant în rețele electrice poluate armonic prin creșterea independenței de platforma *hardware* de achiziții de date. Compromisul care trebuie acceptat constă în scăderea, într-o oarecare măsură, a timpului de răspuns a instrumentului virtual, datorită execuției de tip secvențial a aplicațiilor.

Instrumentele virtuale reprezintă o modalitate facilă, intuitivă și productivă de integrare a unor echipamente de măsurare microprogramate într-o aplicație industrială.

7.7 TESTAREA INSTRUMENTULUI VIRTUAL ESA ȘI REZULTATE EXPERIMENTALE

Testarea instrumentului virtual de măsurare și analiză a semnalelor electrice, **ESA**, a fost făcută, în primă fază, prin aplicarea la intrările analizorului a unor fișiere de date reprezentând eşantioanele unor semnale de intrare (tensiune și curent), de formă dreptunghiulară, simulate *software* cu ajutorul obiectului predefinit “*function generator*”.

În a doua fază, testarea a fost efectuată prin aplicarea la intrările analizorului a unor fișiere de date reprezentând eşantioanele unor semnale de intrare (tensiune și curent), de formă dreptunghiulară, achiziționate de la un generator de semnal dreptunghiular.

Se constată, analizând rezultatele obținute, că prin aplicarea unor semnale de tensiune și de curent de formă dreptunghiulară (simulate *software* sau reale), cu frecvență de 50 Hz și cu amplitudini de 100 V, respectiv 5 A, prin analiză armonică se obțin doar armonici impare, a căror valoare este invers proporțională cu ordinul acesteia în raport cu fundamentală. Erorile rezultate

sunt inferioare valorii de 0,377% și pot fi justificate prin următoarele considerente:

- semnalele de intrare, reale sau simulate, nu sunt perfect dreptunghiulare (duratele fronturilor sunt foarte scurte, dar nenule), acest aspect contribuind în mod decisiv la eroarea analizei. Acest lucru poate fi justificat, analizând rezultatele experimentale, prin faptul că erorile relative nu depind de valoarea amplitudinii semnalului de intrare (semnalul de tensiune are amplitudine de 100 V, iar cel de curent are amplitudine de 5 A). În cadrul rezultatelor experimentale, a fost limitat ordinul maxim al armonicilor la 50, care reprezintă ordinul maxim al armonicilor de tensiune, respectiv de curent, care dau efecte nedorite în instalațiile electroenergetice, deși instrumentul **ESA** poate pune în evidență armonici cu ranguri mai mari (în analizele efectuate în vederea testării acestui instrument virtual s-au pus în evidență armonici cu ranguri de până la 100);
- precizia de calcul, cu pondere net inferioară, instrumentul virtual lucrând în precizie dublă (maxim $10^{-6}\%$);
- erorile de trunchiere, rezultatele fiind limitate din punct de vedere al reprezentării la maxim 4 zecimale, având o pondere mică (maximum $10^{-2}\%$).

Se constată că erorile de determinare ale valorilor efective ale semnalelor de tensiune și de curent, nu depășesc 0,01%, ceea ce conferă o exactitate ridicată de analiză pentru acest instrument virtual.

7.8 LISTINGUL PROGRAMULUI DE ACHIZIȚIE DE DATE

```
; Programul citește de la tastatura locală una dintre
; tastele apasate. Aceasta este de tip matriceal, cu 3
; linii și 4 coloane. Corespunzător codului tastei, se
; selectează unul dintre cele 6 canale analogice de
; intrare (dacă codul tastei este 0..5), sau o pereche
; de canale de intrare tensiune-curent (dacă codul tastei
; este 6..8), sau toate canalele de intrare (dacă codul
; tastei apasate este 9) - achiziție sincronă -, se
; convertește tensiunea aplicată intrării selectate și se
; memorează rezultatele în memoria RAM de date, urmând a
; fi transmise pe interfata serială la PC.
;
; Dialogul cu afișajul LCD se efectuează prin intermediul unor
; subrutine:
;   WRCMD - scrierea comenzii din registrul R2 la LCD
;   WRDAT - scrierea datelor din registrul R2 la LCD
;   WLCD - așteaptă terminarea operațiunilor interne ale LCD
;   INLCD - inițializarea LCD pentru transfer pe 8 biți.
; Comenzile folosite sunt:
;   CLEAR - 1
```

```

;      CURSOR HOME - 2, 3
;      MODE - 1/INC/SHIFT
;      CONTROL - 1/DisplayON/CursorON/FlashON
;      SHIFT - 1/SHIFT/RIGHT/*/*
;
; Spatiul utilizator incepe de la adresa 8000H, deci acest
; program va fi stocat incepand de la aceasta adresa.
;
;      ORG      8000H
;
; program principal
;
;      LJMP     INIT          ; long jump la rutina de initializare
;                               ; afisaj cu cristale lichide LCD
INIT: ACALL INILCD          ; apel rutina de initializare afisaj
      ACALL CLRLCD          ; apel rutina stergere LCD
; prezentarea programului de achizitie pe LCD - dispaly 1
      MOV      DPTR,#TEXT1   ; se memoreaza in DPTR adresa de in-
;                               ; ceput a sirului de caractere TEXT1
      ACALL MESAJ           ; apel rutina de afisare mesaj la LCD
      ACALL CLEAR           ; apel rutina de stergere temporizata
; prezentarea programului de achizitie pe LCD - dispaly 2
      MOV      DPTR,#TEXT2   ; se memoreaza in DPTR adresa de in-
;                               ; ceput a sirului de caractere TEXT2
      ACALL MESAJ           ; apel rutina de afisare mesaj la LCD
      ACALL CLEAR           ; apel rutina de stergere temporizata
; prezentarea programului de achizitie pe LCD - dispaly 3
      MOV      DPTR,#TEXT3   ; se memoreaza in DPTR adresa de in-
;                               ; ceput a sirului de caractere TEXT3
      ACALL MESAJ           ; apel rutina de afisare mesaj la LCD
      ACALL CLEAR           ; apel rutina de stergere temporizata
; prezentarea programului de achizitie pe LCD - dispaly 4
      MOV      DPTR,#TEXT4   ; se memoreaza in DPTR adresa de in-
;                               ; ceput a sirului de caractere TEXT4
      ACALL MESAJ           ; apel rutina de afisare mesaj la LCD
      ACALL CLEAR           ; apel rutina de stergere temporizata
; prezentarea programului de achizitie pe LCD - dispaly 5
      MOV      DPTR,#TEXT5   ; se memoreaza in DPTR adresa de in-
;                               ; ceput a sirului de caractere TEXT5
      ACALL MESAJ           ; apel rutina de afisare mesaj la LCD
      ACALL CLEAR           ; apel rutina de stergere temporizata
; prezentarea programului de achizitie pe LCD - dispaly 6
      MOV      DPTR,#TEXT6   ; se memoreaza in DPTR adresa de in-
;                               ; ceput a sirului de caractere TEXT6
      ACALL MESAJ           ; apel rutina de afisare mesaj la LCD
      ACALL CLEAR           ; apel rutina de stergere temporizata
; prezentarea programului de achizitie pe LCD - dispaly 7
      MOV      DPTR,#TEXT7   ; se memoreaza in DPTR adresa de in-
;                               ; ceput a sirului de caractere TEXT7
      ACALL MESAJ           ; apel rutina de afisare mesaj la LCD
      MOV      R2,#0CBH      ; adresa scriere cod tasta
      ACALL WRCMD           ; pozitionare cursor
      MOV      R2,#1101B     ; display on, cursor off, flash on
      ACALL WRCMD           ; scrie comanda la LCD
; citirea tastaturii, prin baleierea celor trei linii, citirea celor

```

```

; patru coloane si verificare daca s-a tastat ceva.
; prima linie
NEXT:  MOV     A,#0E0H      ; initializeaza acumulatorul cu #0E0H
                                ; (se selecteaza prima linie de taste)
        ACALL  WRPRDP      ; se apeleaza rutina de scriere/citire
                                ; port
        MOV    R3,A         ; salveaza valoarea in registrul R3
        ANL    A,#0FH       ; mascheaza bitii mai semnificativi
                                ; ai acumulatorului (codul liniei)
        JZ     NEXT1        ; daca A=0 (valoarea citita este 0FH-
                                ; nu s-a tastat nimic),salt la eticheta
                                ; NEXT1 (linia 1 de taste)
        JNZ    COL          ; daca continutul acumulatorului nu
                                ; este zero, salt la determinarea
                                ; codului HEXA al tastei, in functie
                                ; de linia activa si coloana pe care
                                ; este situata tasta apasata

; a doua linie
NEXT1: MOV     A,#0D0H      ; se initializeaza acumulatorul cu #0D0H
                                ; se selecteaza a doua linie de taste)
        ACALL  WRPRDP      ; se apeleaza rutina de scriere/citire
                                ; port
        MOV    R3,A         ; salveaza valoarea in registrul R3
        ANL    A,#0FH       ; mascheaza bitii mai semnificativi
                                ; ai acumulatorului (codul liniei)
        JZ     NEXT2        ; daca A=0 (valoarea citita este 0FH)
                                ; salt la eticheta NEXT
        JNZ    COL          ; daca continutul acumulatorului nu
                                ; este zero, salt la determinarea
                                ; codului HEXA al tastei, in functie
                                ; de linia activa si coloana pe care
                                ; este situata tasta apasata

; a treia linie
NEXT2: MOV     A,#0B0H      ; se initializeaza acumulatorul cu #0D0H
                                ; se selecteaza a treia linie de taste)
        ACALL  WRPRDP      ; se apeleaza rutina de scriere/citire
                                ; port
        MOV    R3,A         ; salveaza valoarea in registrul R3
        ANL    A,#0FH       ; mascheaza bitii mai semnificativi
                                ; ai acumulatorului (codul liniei)
        JZ     NEXT        ; daca A=0 (valoarea citita este 0FH)
                                ; salt la eticheta NEXT (prima linie)
        JNZ    COL          ; daca continutul acumulatorului nu
                                ; este zero, salt la determinarea
                                ; codului HEXA al tastei, in functie
                                ; de linia activa si coloana pe care
                                ; este situata tasta apasata

COL:    JB     ACC.0,COL0    ; salt la COL0, daca bitul 0 al acu-
                                ; mulatorului este 1 (s-a apasat o
                                ; tasta de pe coloana 0)
        JB     ACC.1,COL1    ; salt la COL1, daca bitul 1 al acu-
                                ; mulatorului este 1 (s-a apasat o
                                ; tasta de pe coloana 1)
        JB     ACC.2,COL2    ; salt la COL2, daca bitul 2 al acu-
                                ; mulatorului este 1 (s-a apasat o

```

```

                                ; tasta de pe coloana 2)
                                ; salt la COL3, daca bitul 3 al acu-
                                ; mulatorului este 1 (s-a apasat o
                                ; tasta de pe coloana 3)
COL0:  JB      ACC.3,COL3      ; s-a apasat tasta de pe coloana 0
        MOV     R1,#00H      ; se reconstituie acumulatorul
        MOV     A,R3         ; se interschimba bitii mai semnificativi
        SWAP    A            ; cu cei mai putini semnificativi ai
                                ; acumulatorului
        ANL     A,#0FH       ; se mascheaza bitii mai semnificativi
                                ; pentru identificarea codului liniei
        JB      ACC.1,LINIE1  ; salt daca tasta apasata era pe linia 1
        JB      ACC.2,LINIE2  ; salt daca tasta apasata era pe linia 2
        SJMP    PLAY         ; salt la rutina de afisare, tasta 0
COL1:  MOV     R1,#01H      ; s-a apasat tasta de pe coloana 0
        MOV     A,R3         ; se reconstituie acumulatorul
        SWAP    A            ; se interschimba bitii mai semnificativi
                                ; cu cei mai putini semnificativi ai
                                ; acumulatorului
        ANL     A,#0FH       ; se mascheaza bitii mai semnificativi
                                ; pentru identificarea codului liniei
        JB      ACC.1,LINIE1  ; salt daca tasta apasata era pe linia 1
        JB      ACC.2,LINIE2  ; salt daca tasta apasata era pe linia 2
        SJMP    PLAY         ; salt la rutina de afisare, tasta 1
COL2:  MOV     R1,#02H      ; s-a apasat o tasta de pe coloana 2
        MOV     A,R3         ; se reconstituie acumulatorul
        SWAP    A            ; se interschimba bitii mai semnificativi
                                ; cu cei mai putini semnificativi ai
                                ; acumulatorului
        ANL     A,#0FH       ; se mascheaza bitii mai semnificativi
                                ; pentru identificarea codului liniei
        JB      ACC.1,LINIE1  ; salt daca tasta apasata era pe linia 1
        JB      ACC.2,LINIE2  ; salt daca tasta apasata era pe linia 2
        SJMP    PLAY         ; salt la rutina de afisare, tasta 2
COL3:  MOV     R1,#03H      ; s-a apasat o tasta de pe coloana 3
        MOV     A,R3         ; se reconstituie acumulatorul
        SWAP    A            ; se interschimba bitii mai semnificativi
                                ; cu cei mai putini semnificativi ai
                                ; acumulatorului
        ANL     A,#0FH       ; se mascheaza bitii mai semnificativi
                                ; pentru identificarea codului liniei
        JB      ACC.1,LINIE1  ; salt daca tasta apasata era pe linia 1
        JB      ACC.2,LINIE2  ; salt daca tasta apasata era pe linia 2
        SJMP    PLAY         ; salt la rutina de afisare, tasta 3
LINIE1: MOV     A,#04H      ; tasta apasata era pe linia 1
        ADD     A,R1         ; se aduna codul coloanei cu codul liniei
        MOV     R1,A        ; codul hexa al tastei apasate este in R1
        SJMP    PLAY         ; salt la rutina de afisare, tastele 4..7
LINIE2: MOV     A,#08H      ; tasta apasata era pe linia 2
        ADD     A,R1         ; se aduna codul coloanei cu codul liniei
        MOV     R1,A        ; codul hexa al tastei apasate este in R1
        SJMP    PLAY         ; salt la rutina de afisare, tastele 8..B
PLAY:  MOV     A,R1         ; codul hexa al tastei in acumulator
        CALL    HEXASC       ; conversie in ASCII a codului tastei,
                                ; pentru afisare pe LCD
        MOV     R2,A        ; codul HEXA al tastei apasate in R2

```

```

ACALL TRX          ; scrie la LCD codul tastei
MOV R2,#0CBH       ; adresa scriere cod tasta
ACALL WRCMD         ; pozitionare cursor
MOV R2,#1100B       ; display on, cursor off, flash off
ACALL WRCMD         ; scrie comanda la LCD
ACALL CLEAR         ; apel rutina stergere temporizata a LCD
; prezentarea programului de achizitie pe LCD - dispaly 8
MOV DPTR,#TEXT8     ; se memoreaza in DPTR adresa de in-
                    ; ceput a sirului de caractere TEXT8
ACALL MESAJ         ; apel rutina afisare mesaj la LCD
MOV A,R1             ; codul tastei apasate in acumulator
JZ SCH              ; daca este 0, canalul selectat este
                    ; canalul 0 si salt la eticheta SCH
DEC A               ; decrementeaza acumulatorul
JZ SCH              ; daca este 0, canalul selectat este
                    ; canalul 1 si salt la eticheta SCH
DEC A               ; decrementeaza acumulatorul
JZ SCH              ; daca este 0, canalul selectat este
                    ; canalul 2 si salt la eticheta SCH
DEC A               ; decrementeaza acumulatorul
JZ SCH              ; daca este 0, canalul selectat este
                    ; canalul 3 si salt la eticheta SCH
DEC A               ; decrementeaza acumulatorul
JZ SCH              ; daca este 0, canalul selectat este
                    ; canalul 4 si salt la eticheta SCH
DEC A               ; decrementeaza acumulatorul
JZ SCH              ; daca este 0, canalul selectat este
                    ; canalul 5 si salt la eticheta SCH
DEC A               ; decrementeaza acumulatorul
JZ DCH              ; tasta apasata era 6 si se selecteaza
                    ; perechea de canale 0 si 3 (esantionare
                    ; sincrona)
DEC A               ; decrementeaza acumulatorul
JZ DCH              ; tasta apasata era 7 si se esantioneaza
                    ; sincron perechea de canale 1 si 3
DEC A               ; decrementeaza acumulatorul
JZ DCH              ; tasta apasata era 8 si se esantioneaza
                    ; sincron perechea de canale 2 si 5
DEC A               ; decrementeaza acumulatorul
JZ ALL              ; tasta apasata era 9 si se esantioneaza
                    ; sincron canalele 0 .. 5
SCH: MOV A,R1
ACALL SH
ACALL SINGLE
LJMP NEXT
DCH: MOV A,R1
CLR C
SUBB A,#6
MOV R7,A
ACALL SH
ACALL SINGLE
MOV A,R7
INC A
INC A
INC A

```

```

        ACALL  SINGLE
        LJMP  NEXT
ALL:    MOV    A,R1
        CLR   C
        SUBB  A,#9
        MOV   R7,A
        ACALL SH
        ACALL SINGLE
        MOV   A,R7
        INC   A
        MOV   R7,A
        ACALL SINGLE
        MOV   A,R7
        INC   A
        MOV   R7,A
        ACALL SINGLE
        MOV   A,R7
        INC   A
        MOV   R7,A
        ACALL SINGLE
        MOV   A,R7
        INC   A
        MOV   R7,A
        ACALL SINGLE
        LJMP  NEXT
;
SH:     ORL    A,#11000000B
        PUSH  ACC
        ACALL WRCDA           ; comanda de esantionare, selectie
                                ; canal si amplificare
        ACALL WAIT5          ; asteapta 5us pentru achizitia
                                ; semnalului de intrare

        POP   ACC
        ANL   A,#00111111B
        PUSH  ACC
        ACALL WRCDA           ; comanda de memorare pentru canalul
                                ; de intrare selectat

        POP   ACC
        RET
;
SINGLE:
        ACALL WRCONV          ; start conversie
        ACALL WAIT10          ; se asteapta sfarsitul conversiei
        ACALL RDCONV          ; se transfera rezultatul conversiei
                                ; pe 12 biti, in registrele LSBYTE si
                                ; MSBYTE
        ACALL RDLSBYTE        ; se citeste octetul mai putin semni-
                                ; ficativ al conversiei, din registrul
                                ; LSBYTE

        ACALL WRMEM           ; se salveaza octetul in memorie
        ACALL RDMSBYTE        ; se citeste octetul mai semnificativ

```

```

                                ; al conversiei, din registrul MSBYTE
ACALL  WRMEM                    ; se salveaza octetul in memorie
RET

;
WRCDA:                          ; rutina de selectie a parametrilor
; conversiei: numarul canalului selectat, comanda de esantionare,
; amplificarea amplificatorului cu cqstig reglabil
MOV    P2,#1                    ; se selecteaza porturile de iesire
MOV    R0,#28H                  ; se selecteaza portul COMAND
MOVX   @R0,A                    ; muta continutul acumulatorului
                                ; la portul de iesire COMAND
RET

;
WRCONV:    MOV    P2,#1
MOV    R0,#20H
MOVX   @R0,A
RET

;
RDCONV:    MOV    P2,#1
MOV    R0,#20H
MOV    A,@R0
RET

;
RDLMSBYTE:
MOV    P2,#1
MOV    R0,#68H
MOV    A,@R0
RET

;
RDMSBYTE:
MOV    P2,#1
MOV    R0,#70H
MOV    A,@R0
RET

;
WAIT5: MOV    R3,#5
LOOP2: NOP
      DJNZ   R3,LOOP2
      RET

;
WAIT10: MOV    R3,#10
LOOP3: NOP
      DJNZ   R3,LOOP3
      RET

;
WRMEM: NOP
      RET

;
INILCD:                          ; rutina de initializare LCD.
; Aceasta rutina trimite la LCD comanda #38H = 0011 1000B.
; Conform datelor de catalog, acesta comanda seteaza urmatorii
; parametri ai afisajului:
;   - bitul 5 - defineste aceasta comanda (function set);
;   - bitul 4(DL-Data Length)=1 seteaza dialogul pe 8 biti
;   intre procesor si LCD;

```



```

;      - bitul 3(N-Number of display lines)=1 seteaza numarul
;      liniilor afisajului ca fiind 2^N=2;
;      - bitul 2(F-Character Font)=0 seteaza forma caracterului
;      ca fiind 5*7 puncte
MOV     R2,#38H      ; incarca R2 cu cuvantul de comanda
                    ; pregatind dialogul pe 8 biti cu
                    ; driverele de afisaj
ACALL   WRCMD        ; transmite comanda anterioara la
                    ; port, efectuand programarea LCD
MOV     R4,#50
DEL4MS:              ; rutina de intarziere cu 4 ms,
; necesara functionarii display-ului LCD
ACALL   DELAY        ; absolute call rutina DELAY
DJNZ    R4,DEL4MS    ; se decrementeaza registrul R4 si
                    ; salt la eticheta DEL4MS daca conti-
                    ; nutul acestuia este nenul
MOV     R4,#4        ; se incarca R4 cu valoarea imediata 4
LINI:    ; LINI seteaza modul de lucru al LCD
ACALL   WRCMD        ; se scrie comanda la driverele LCD
ACALL   DELAY        ; apel rutina DELAY
DJNZ    R4,LINI      ; se decrementeaza registrul R4 si salt
                    ; la eticheta LINI daca este nenul con-
                    ; tinutul acestui registru
ACALL   WLCD        ; testarea starii driverelor LCD
MOV     R2,#6        ; seteaza modul de lucru "entry"
ACALL   WRCMD        ; transmite comanda la LCD
ACALL   WLCD        ; testarea starii driverelor LCD
MOV     R2,#0EH      ; seteaza modul "display on"
ACALL   WRCMD        ; transmite comanda la LCD
ACALL   WLCD        ; testarea starii driverelor LCD
MOV     R2,#1        ; stergere LCD
ACALL   WRCMD        ; transmite comanda la LCD
ACALL   WLCD        ; testarea starii driverelor LCD
RET     ; revenire din subrutina INILCD
;
WRCMD:    ; rutina WRCMD scrie o comanda la
; driverele de afisaj, in urma verificarii starii acestora. Se se-
; lecteaza afisajul LCD si comanda continuta de registrul R2 este
; transferata driverelor din LCD.
ACALL   WLCD        ; se verifica starea driverelor (BF=0)
MOV     A,R2        ; comanda continuta in registrul R2 e
                    ; transferata in acumulator
MOV     P2,#1        ; se selecteaza portul LCD (S0=1)
MOV     R0,#0        ; se selecteaza pe magistrala de adrese
                    ; cu A0 si A1 combinatia 0000 0000 B,
                    ; corespunzatoare scrierii unei comenzi
                    ; la driverele afisajului LCD
MOVX    @R0,A        ; se scrie in driverele afisajului LCD
                    ; (la adresa stabilita anterior) comanda
RET     ; revenire din rutina WRCMD
;
WLCD:    ; rutina ce testeaza daca este sau
; nu posibila transmiterea unui nou caracter catre LCD, pe baza in-
; formatiilor furnizate de rutina RDCMD. Se testeaza bitul 7 (BF -
; Busy Flag) al registrului de stare al LCD. Daca BF=1, inseamna ca

```

```

; LCD efectueaza o operatie interna si, deci, nu poate accepta un
nou
; caracter.
    ACALL  RDCMD          ; apel rutina RDCMD
    JB     ACC.7,WLCD     ; se verifica daca BF=1 si asteapta
                          ; trecerea lui in 0 (terminarea ope-
                          ; ratiunilor interne)
    RET                      ; revenire din subrutina WRCL
;
RDCMD:                      ; subrutina RDCMD selecteaza afisajul
; si citeste din registru de stare starea curenta a driverelor
acestui,
; informatie care este depusa in acumulator
    MOV    P2,#1          ; selectie port (S0=1)
    MOV    R0,#2          ; se realizeaza pe magistrala de adrese
                          ; selectia cu A0 si A1 (0000 0010 B),
                          ; pentru a se citi starea driverelor de
                          ; afisaj LCD
    MOVX   A,@R0          ; se transfera in acumulator valoarea
                          ; gasita la adresa specificata anterior
    RET                      ; revenire din rutina RDCMD
;
DELAY:                      ; subrutina DELAY realizeaza o intarziere
; cu 80 microsecunde
    MOV    R3,#17         ; registrul R3 este incarcat cu valoarea
                          ; imediata 17
                          ; 80E-6secunde = 17 * 2instructiuni NOP *
                          ; * 12 perioade de ceas/instructiune NOP*
                          ; 1/11.059MHz (perioada ceas procesor)
LL1:    NOP                ; no operation
        NOP                ; no operation
        DJNZ  R3,LL1       ; se decrementeaza registrul R3, se compa-
                          ; ra continutul sau cu 0. Daca rezultatul
                          ; compararii este 1, salt la eticheta LL1
    RET                      ; revenire din subrutina DELAY
;
CLR_LCD:                    ; subrutina de stergere a afisajului
; cu cristale lichide si setare a modului de afisare
    MOV    R2,#1          ; stergere afisaj LCD
    ACALL  WRCMD          ; transmite comanda la LCD
    MOV    R2,#1100B      ; 1, display on, cursor off, flash off
    ACALL  WRCMD          ; transmite comanda la LCD
    RET                      ; revenire din rutina de CLR_LCD
;
TRX1:                      ; subrutina de scriere sir de caractere
; la afisajul cu cristale lichide LCD
    CLR    A              ; initializeaza acumulatorul cu 0
    MOVC   A,@A+DPTR      ; muta in acumulator adresa gasita
                          ; la adresa @A+DPTR
    CJNE   A,#24H,TRCAR1  ; compara continutul acumulatorului
                          ; cu #24H (codul hexa al caracterului
                          ; $ - sfarsit de mesaj) si daca este
                          ; diferit se efectueaza salt la eticheta
                          ; TRCAR1
    RET                      ; iesire din subrutina TRX1

```

```

TRCAR1:      MOV      R2,A      ; muta continutul acumulatorului in
                                ; registrul R2
              ACALL    WRDAT      ; apel subrutina de scriere date in
                                ; registrele afisajului LCD
              INC      DPTR      ; adresa urmatorului caracter de scris
              SJMP     TRX1      ; small jump la eticheta TRX1, pentru
                                ; a scrie urmatorul caracter
;
TRX:          ; subrutina de afisare la LCD a unui
; caracter ASCII
              CLR      A          ; se initializeaza acumulatorul cu 0
              MOV      A,R2      ; se transfera continutul registrului
                                ; R2 in acumulator
              LJMP     TRCAR      ; long jump la eticheta TRCAR
LL7:          RET              ; iesire din rutina TRX
TRCAR:        MOV      R2,A      ; se restaureaza in R2 continutul Acc.
              ACALL    WRDAT      ; se scriu datele in driverele LCD
              SJMP     LL7      ; short jump la eticheta LL7
;
WRDAT:        ; subrutina WRDAT este identica cu WRCMD
; dar este utilizata pentru scrierea la driverele LCD codul
; caracterelor ce urmeaza a fi afisate
              ACALL    WLCD      ; testarea starii driverelor LCD
              MOV      P2,#1      ; se selecteaza portul LCD (S0=1)
              MOV      R0,#1      ; se selecteaza pe magistrala de adrese
                                ; cu A0 si A1 combinatia 0000 0001 B,
                                ; corespunzatoare scrierii unui caracter
                                ; la driverele afisajului LCD
              MOV      A,R2      ; se transfera caracterul in acumulator
              MOVX     @R0,A      ; se transmite caracterul driverelor LCD
              RET              ; revenire din rutina WRDAT
;
MESAJ:        ; subrutina de afisare la LCD a doua linii
; de cate 16 caractere
              ACALL    TRX1      ; apel rutina de transmisie mesaj
              MOV      R2,#0C0H   ; pozitionare pe a doua linie
              ACALL    WRCMD      ; scriere comanda pozitionare la LCD
              INC      DPTR      ; incrementare DPTR (linia a doua)
              ACALL    TRX1      ; apel rutina de transmisie mesaj
              RET              ; revenire din rutina MESAJ
;
CLEAR:        ; subrutina mentine starea afisajului
; timp de 3 secunde, sterge afisajul si pozitioneaza cursorul pe
; prima pozitie, de pe prima linie
              ACALL    SEC3      ; mentine starea afisajului 3 sec.
              MOV      R2,#1      ; comanda stergere display
              ACALL    WRCMD      ; scriere comanda la LCD
              MOV      R2,#2      ; cursor home, prima linie
              ACALL    WRCMD      ; scriere comanda la LCD
              RET              ; revenire din subrutina MESAJ
;
WRPRDP:       ; rutina de scriere/citire port
              MOV      P2,#1      ; selecteaza decodificatorul (A8=1)
              MOV      R0,#20H    ; incarca R0 cu 0010 0000 B:
                                ; selectie port de iesire 1 (S1=0)

```

[illegible]

```
    NOP
    NOP
    DJNZ    R7,LOOP      ; se decrementeaza registrul R7 si
                        ; daca este nenul, salt la LOOP
    DJNZ    R6,LOOP1     ; se decrementeaza registrul R6 si
                        ; daca este nenul, salt la LOOP1
    RET                ; revenire din subrutina SEC3
;
TEXT1: DB ' ACHIZITIE A/D: $'
      DB 'PE 6 CANALE: 12b$'
TEXT2: DB 'AFISARE: 2 LINII$'
      DB 'DE 16 CARACTERE $'
TEXT3: DB 'STOCARE: 2 BYTES$'
      DB ' IN MEMORIA RAM $'
TEXT4: DB 'TRANSFER SERIAL$'
      DB 'LA PC:9600 BAUDS$'
TEXT5: DB 'CONVERSIE D/A:8b$'
      DB 'REFACERE SEMNAL$'
TEXT6: DB 'APASATI O TASTA:$'
      DB ' (NUMAR: 0..11) $'
TEXT7: DB 'TASTA ACTIONATA:$'
      DB ' TASTA " " $'
TEXT8: DB ' SEMNAL INTRARE $'
      DB ' CANAL : . V$'
;
HEXASC:                ; rutina de conversie hexa-ascii
    ANL     A,#0FH      ; se mascheaza cei patru biti mai
                        ; semnificativi ai acumulatorului
    JNB     ACC.3,NOADJ  ; daca bitul 3 = 0 salt la NOADJ
    JB      ACC.2,ADJ    ; daca bitul 2 = 1 salt la ADJ
    JNB     ACC.1,NOADJ  ; daca bitul 1 = 0 salt la NOADJ
ADJ:  ADD    A,#07H      ; aduna acumulatorul cu #07H
NOADJ: ADD    A,#30H     ; aduna acumulatorul cu #30H
    RET                ; revenire din rutina HEXASC
;
    END                ; sfarsitul programului
```


8. SISTEM CU MICROCONTROLLER PENTRU MĂSURAREA ȘI CONTROLUL TEMPERATURII

8.1 SPECIFICAȚIILE DE PROIECTARE ALE SISTEMULUI PENTRU MĂSURAREA ȘI CONTROLUL TEMPERATURII

În condițiile actuale, tot mai multe echipamente de măsurare și reglare, în domeniile industrial, industriei auto și casnic, au la bază sisteme organizate în jurul unui microprocesor sau a unui microcontroller. Acest tip de abordare a proiectării prezintă o serie de avantaje substanțiale, dintre care putem remarca: simplitatea și caracterul de compatibilitate a proiectării, grad ridicat de integrare, gabarit redus, consum redus de energie electrică, facilități de adaptabilitate la determinarea și controlul unor noi parametri (sau în situația modificării relațiilor de calcul ale acestora), imunitate ridicată la perturbații și zgomote electrice și, bineînțeles, nu în ultimul rând, fiabilitate ridicată.

În cele ce urmează, va fi prezentată structura și elementele de proiectare ale unui sistem destinat măsurării și controlului temperaturii în mediu industrial. Acest sistem va fi organizat în jurul unui microcontroller de tip 80C552, fiind caracterizat de următorii parametri:

- măsurarea temperaturii în 8 puncte al procesului, folosind senzori de temperatură de tip LM135, cu rezoluție de $0,5^{\circ}\text{C}$, sau senzori de temperatură cu ieșire în curent (semnal unificat 4 .. 20 mA);
- intervalul de temperatură controlat este cuprins între 0°C și $+100^{\circ}\text{C}$;
- conversie analog-digitală pe 8 biți, în format binar direct, a informațiilor de temperatură preluate din proces;
- memorarea temporară a codurilor binare de temperatură într-o memorie de date, de tip RAM static;
- programarea, prin intermediul unui dispozitiv local de introducere a datelor (tastatură locală) a temperaturii de echilibru în cadrul sistemului controlat;
- afișarea locală, pe un display cu cristale lichide, atât a temperaturii de lucru preprogramate, cât și a temperaturii medii determinate;
- controlul continuu al elementului de execuție (elementul de încălzire) pentru menținerea temperaturii preprogramate;
- extinderea posibilităților de control la două elemente de execuție;
- transmiterea informațiilor de temperatură prelevate din proces, pe o legătură serială full-duplex, de mare viteză, compatibilă RS-232, către un calculator gazdă, în vederea analizei statistice și interpretării la un nivel înalt a rezultatelor.

Va fi elaborat și un program de aplicație, scris în limbaj de asamblare, care permite măsurarea și controlul temperaturii în cadrul procesului. Limbajul de asamblare, deși mai greu de utilizat și de interpretat, prezintă avantajul minimizării codului aplicației.

În cele ce urmează, se va concentra efortul de proiectare pentru minimizarea hardware-ului utilizat, însă cu asigurarea performanțelor de exactitate specificate.

În subcapitolele următoare vor fi descrise, într-o ordine ce ține seama de complexitatea structurilor utilizate, elementele hardware și software care compun acest sistem microprocesat destinat măsurării și controlului temperaturii.

8.2 DESCRIEREA FUNCȚIONALĂ A SISTEMULUI DE MĂSURARE ȘI CONTROLUL TEMPERATURII

Sistemul de măsurare și controlul temperaturii se compune din mai multe unități funcționale, după cum urmează:

- *blocul de măsurare a temperaturii*, care conține:
 - senzorul de temperatură, de tip β A135;
 - sursele de tensiune de referință și de prag;
- *sistemul de dezvoltare cu microcontroller 80C552*, folosit pentru:
 - prescrierea temperaturii programate;
 - conversia analog-digitală a informației de temperatură;
 - memorarea rezultatelor obținute în urma supravegherii procesului;
 - controlul continuu al temperaturii în cadrul procesului supravegheat;
 - transmisia serială a rezultatelor prelevate în cadrul procesului supravegheat către un sistem de calcul mai puternic, de tip IBM-PC sau compatibil, în vederea analizei detaliate, reprezentării grafice a regimului tranzitoriu de control al temperaturii în cadrul procesului controlat și a analizei statistice a rezultatelor obținute;
- *blocul de comandă al elementului de execuție*.

8.2.1 BLOCUL DE MĂSURARE A TEMPERATURII

Blocul de măsurare a temperaturii se compune din următoarele module funcționale:

- senzorul de temperatură, de tip β A135;
- sursele de tensiune de referință și de prag.

8.2.1.1 TRADUCTORUL DE TEMPERATURĂ

8.2.1.1.1 CONECTAREA TRADUCTORULUI DE TEMPERATURĂ

Senzorul de temperatură este conectat, prin intermediul unor etaje de adaptare, la portul P_5 al microcontroller-ului, prin care sunt vehiculate alternativ și semnalele de control ale secțiunii analogice. Informația de temperatură furnizată de senzorul specializat este vehiculată de semnalul ADC0 ... ADC7.

8.2.1.1.2 SENZORUL DE TEMPERATURĂ

Senzorul de temperatură, de tip $\beta A135$, permite măsurarea temperaturilor în domeniul $-55^{\circ}\text{C} \div +150^{\circ}\text{C}$, gamă ce acoperă intervalul de temperatură $0^{\circ}\text{C} \div +100^{\circ}\text{C}$, impus prin tema de proiectare. Acest senzor de temperatură este constituit de fapt de o diodă Zener a cărei tensiune, în polarizare inversă, variază liniar cu $+10\text{mV/grad}$, funcție de temperatură pe întreg domeniul de temperaturi de lucru specificat.

Senzorul de temperatură $\beta A135$ este caracterizat de o rezistență dinamică foarte redusă (tipic 1Ω), ceea ce ușurează mult preluarea informației de temperatură de la acest circuit. Circuitul funcționează normal pentru o gamă largă de curenți de polarizare, respectiv între $400\text{ }\mu\text{A}$ și 5 mA .

Senzorul de temperatură folosit este caracterizat de o eroare de maximum 1°C . Această eroare inițială poate fi compensată prin conectarea unui potențiomtru cu valoarea de $10\text{k}\Omega$ între anodul și catodul diodei, cursorul potențiometrului fiind conectat la terminalul de compensare al senzorului de temperatură.

Senzorul de temperatură $\beta A135$ furnizează la ieșire o tensiune proporțională cu temperatura absolută, în Kelvin.

Schema electrică de utilizare a acestui senzor de temperatură este prezentată în fig. 8.1, în care P_1 este potențiometrul de compensare ($P_1=10\text{k}\Omega$).

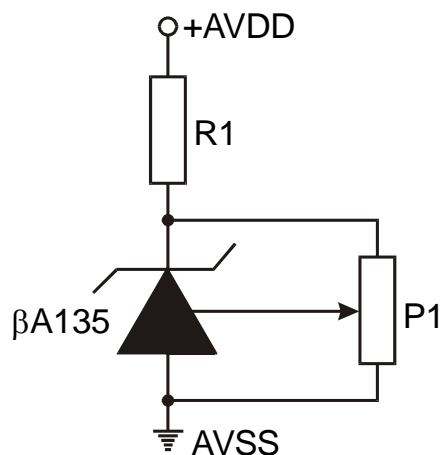


Fig. 8.1 Senzorul de temperatură.

Pentru acest circuit se poate scrie:

$$I_{Z_1} = \frac{V_{CC} - U_{Z_1}(T)}{R_1} - \frac{U_{Z_1}(T)}{P_1} \quad (8.1)$$

Curentul prin dioda D_1 ($\beta A135$) trebuie să se încadreze în intervalul $400\mu A \div 5mA$, în vederea unei funcționări corecte, pe întreg intervalul de temperatură specificat ($0^\circ C \div +100^\circ C$).

Se poate deduce:

$$\begin{aligned} U_{Z_1}(273K) &= U_{Z_1}(0^\circ C) = 2,73V \\ U_{Z_1}(373K) &= U_{Z_1}(+100^\circ C) = 3,73V \end{aligned} \quad (8.2)$$

În concluzie:

$$\begin{aligned} I_{Z_{1,min}} &= \frac{V_{CC} - U_{Z_1}(100^\circ C)}{R_1} - \frac{U_{Z_1}(100^\circ C)}{P_1} \\ I_{Z_{1,max}} &= \frac{V_{CC} - U_{Z_1}(0^\circ C)}{R_1} - \frac{U_{Z_1}(0^\circ C)}{P_1} \end{aligned} \quad (8.3)$$

Rezultă prin utilizarea ecuațiilor anterioare limitele intervalului de apartenență pentru rezistența de polarizare R_1 :

$$\begin{aligned} R_{1,max} &= \frac{V_{CC} - U_{Z_1}(100^\circ C)}{I_{Z_{1,min}} + \frac{U_{Z_1}(100^\circ C)}{P_1}} = 1,2359k\Omega \\ R_{1,min} &= \frac{V_{CC} - U_{Z_1}(0^\circ C)}{I_{Z_{1,max}} + \frac{U_{Z_1}(0^\circ C)}{P_1}} = 0,435k\Omega \end{aligned} \quad (8.4)$$

Trebuie satisfăcută relația:

$$R_{1,\min} \leq R_1 \leq R_{1,\max} \quad (8.5)$$

Se alege $R_1=0,68k\Omega$, valoare standard care satisface condițiile de funcționare specificate anterior.

8.2.1.1.3 MODULUL SURSELOR DE TENSIUNE DE REFERINȚĂ

Rolul acestui modul funcțional este acela de a obține folosind sursa de tensiune analogică (AVDD în raport cu AVSS) care alimentează convertorul analog-digital implementat intern în cadrul structurii microcontroller-ului 80C552 a două tensiuni de referință, V_{REF-} și V_{REF+} .

Alegerea acestor două tensiuni de referință se bazează pe faptul că sistemul de achiziții de date din cadrul microcontroller-ului 80C552 convertește pe 10 biți tensiunea de intrare delimitată de intervalul $[V_{REF-}, V_{REF+}]$, conform ecuației:

$$N = 2^{10} \cdot \frac{V_{IN} - V_{REF-}}{V_{REF+} - V_{REF-}} = 1024 \cdot \frac{V_{IN} - V_{REF-}}{V_{REF+} - V_{REF-}} \quad (8.6)$$

Prin utilizarea numai a celor mai semnificativi 8 biți ai rezultatului, conținută în registrul ADCH, caracteristica de transfer devine:

$$N = 2^8 \cdot \frac{V_{IN} - V_{REF-}}{V_{REF+} - V_{REF-}} = 256 \cdot \frac{V_{IN} - V_{REF-}}{V_{REF+} - V_{REF-}} \quad (8.7)$$

Astfel, se asociază codul binar $N=0$ pentru $V_{IN}=V_{REF-}$ și $N=1024$ (sau 256, dacă sunt utilizați doar cei mai semnificativi opt biți ai rezultatului) pentru $V_{IN}=V_{REF+}$.

Pentru a se realiza o asociere ușor de interpretat a rezultatului conversiei, obținut sub formă binară, cât și o minimizare a erorii la interpretarea acestuia (pentru simplitate se vor prelua doar cei mai semnificativi opt biți ai rezultatului conversiei analog-digitale, conținută în registrul ADCH), tensiunile de referință care se aplică convertorului analog-digital vor corespunde unui interval extins de temperatură, și anume $0^\circ\text{C} \div +128^\circ\text{C}$:

$$\begin{aligned} V_{REF-} &= U_{Z_1}(0^\circ\text{C}) = 2,73\text{V} \\ V_{REF+} &= U_{Z_1}(128^\circ\text{C}) = 4,01\text{V} \end{aligned} \quad (8.8)$$

În acest mod, codul binar corespunzător lui $N=0$ se obține pentru tensiunea de intrare corespunzătoare temperaturii $t_0=0^\circ\text{C}$ iar codul binar corespunzător lui $N=256$ se obține pentru o tensiune de intrare corespunzătoare temperaturii $t_1=128^\circ\text{C}$. Rezoluția conversiei sub formă numerică a temperaturii este de 1bit/ $0,5^\circ\text{C}$, putându-se astfel interpreta temperaturi variind în pași de jumătate de grad. Pentru temperaturi măsurate și controlate cu variații de un

grad (rezoluție impusă prin tema de proiectare) se obține întotdeauna un cod binar de ieșire direct reprezentând un număr întreg. Astfel este realizată minimizarea eroarea de măsurare și de interpretare a rezultatului.

Prin alegerea unor valori corespunzătoare ale tensiunilor de referință aplicate convertorului analog-digital, se obține un rezultat reprezentat în cod binar direct proporțional cu temperatura exprimată în grade Celsius, fără a fi necesară utilizarea unui amplificator diferențial pentru realizarea conversiei Kelvin / grad Celsius.

Schema electrică a acestui modul funcțional este prezentată în fig. 8.2.

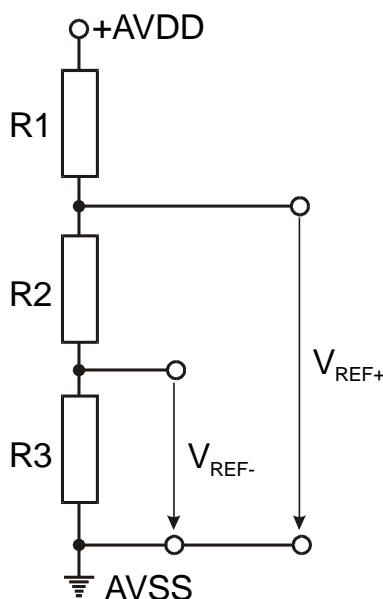


Fig. 8.2 Modulul surselor de referință.

Conform circuitului reprezentat în fig. 8.2, se poate scrie:

$$\begin{aligned} V_{REF-} = U_{Z_1}(0^\circ\text{C}) &= 2,73\text{V} = \frac{R_3}{\sum_{i=1}^3 R_i} \cdot AVDD \\ V_{REF+} = U_{Z_1}(128^\circ\text{C}) &= 4,01\text{V} = \frac{R_2 + R_3}{\sum_{i=1}^3 R_i} \cdot AVDD \end{aligned} \quad (8.9)$$

în care $AVDD = +5\text{V}$.

Se alege un curent prin divizorul R_1, R_2, R_3 , notat I , egal cu 1mA . În aceste condiții, rezultă:

$$\left. \begin{aligned} I &= \frac{AVDD}{\sum_{i=1}^3 R_i} = 1\text{mA} \\ AVDD &= 5\text{V} \end{aligned} \right\} \Rightarrow \sum_{i=1}^3 R_i = 5\text{k}\Omega \quad (8.10)$$

Trebuie menționat că valoarea acestui curent a fost aleasă suficient de mare astfel încât să poată fi neglijate curenții absorbiți de intrările tensiunilor de referință ale convertorului analog-digital.

Ca urmare, folosind ecuația rezultă:

$$V_{REF-} = \frac{R_3}{\sum_{i=1}^3 R_i} \cdot AVDD \Rightarrow R_3 = \sum_{i=1}^3 R_i \cdot \frac{V_{REF-}}{AVDD} = 2,73k\Omega$$

$$V_{REF+} = \frac{R_2 + R_3}{\sum_{i=1}^3 R_i} \cdot AVDD \Rightarrow R_2 + R_3 = \sum_{i=1}^3 R_i \cdot \frac{V_{REF+}}{AVDD} = 4,01k\Omega \quad (8.11)$$

Se alege $R_3=2,73k\Omega$ valoare standard, rezultând valoarea lui R_2 :

$$R_2 = \sum_{i=2}^3 R_i - R_3 = 4,01k\Omega - 2,73k\Omega = 1,28k\Omega \quad (8.12)$$

Valoarea rezistenței R_1 rezultă conform ecuației:

$$R_1 = \sum_{i=1}^3 R_i - \sum_{j=2}^3 R_j = 5k\Omega - 4,01k\Omega = 0,99k\Omega \quad (8.13)$$

Pentru a se obține valoarea calculată a rezistenței R_1 , aceasta va fi compusă din două rezistențe standard cu valorile $1k\Omega$ și respectiv $100k\Omega$, legate în paralel.

$$R_1 = \frac{1k\Omega \cdot 100k\Omega}{1k\Omega + 100k\Omega} = 0,990099k\Omega \quad (8.14)$$

Pentru a se obține valoarea calculată a rezistenței R_2 , aceasta va fi compusă din trei rezistențe standard cu valorile $1,5k\Omega$, $10k\Omega$ și respectiv $68k\Omega$, legate în paralel.

$$R_2 = \frac{\frac{1,5k\Omega \cdot 10k\Omega}{1,5k\Omega + 10k\Omega} \cdot 68k\Omega}{\frac{1,5k\Omega \cdot 10k\Omega}{1,5k\Omega + 10k\Omega} + 68k\Omega} = 1,279999k\Omega \quad (8.15)$$

Toate rezistențele care implementează blocul surselor de referință sunt alese cu precizie cât mai ridicată ($\pm 0,25\%$ în cazul de față) și cu un coeficient de variație cu temperatura de valoare cât mai scăzută, pentru a se putea garanta exactitatea de măsurare impusă. De asemenea, valorile acestor rezistențe sunt relativ mici (ordin $k\Omega$), pentru a se putea neglijă curenții de intrare ai convertorului analog-digital din punctele V_{REF-} și V_{REF+} .

8.2.1.1.4 SENZOR DE TEMPERATURĂ CU IEȘIRE UNIFICATĂ ÎN CURENT

Pentru realizarea compatibilității cu echipamentele standard de automatizări industriale a fost realizată și o sondă de temperatură cu ieșire unificată în curent 4÷20 mA pentru un interval de temperatură 0÷100 °C. Acest traductor de temperatură este prezentat în fig. 8.3.

Curentul de ieșire $I_s(t)$ este dat de următoarele componente:

- curentul prin rezistența R_2 , dependent de temperatură și asigurat integral de colectorul tranzistorului Q_1 :

$$I_{R_2} = I(t) = \frac{U(t) + xP_1 \cdot (I_- + I_{REF})}{R_2} \quad (8.16)$$

- curentul de alimentare al senzorului de temperatură, I_- , forțat de un generator de curent realizat cu tranzistorul Q_1 :

$$I_- = I_+ = \frac{U_{EB}}{R_1} = \frac{0,6V}{4,7k\Omega} = 0,1276595744681mA \quad (8.17)$$

- curentul prin potențiometrul P_1 , determinat de sursa de tensiune de referință:

$$I_{REF} = \frac{V_{REF} - xP_1 \cdot I_-}{R_3 + P_1} \quad (8.18)$$

- curentul prin terminalul de ajustare al sursei de referință:

$$I_{ADJ} = (50 \div 100)\mu A = \text{neglijabil} \quad (8.19)$$

Sumând aceste componente, se constată că valoarea minimă a curentului $I_s(t)$ este de 4mA, obținut pentru $t=0^\circ C$, adică $U(t)=0V$. Această valoare poate fi ajustată prin intermediul potențiometrului P_1 . Valoarea maximă a curentului $I_s(t)$ este de 20mA, obținut pentru $t=100^\circ C$, adică $U(t)=1,00V$ (de fapt la valoarea minimă ajustată la valoarea de 4mA se adaugă valoarea $\frac{U(100^\circ C)}{R_2} = \frac{1V}{62,5\Omega} = 16mA$). Se constată, de asemenea, că această schemă nu

necesită reglajul ambelor capete de scală, în condițiile în care rezistența R_2 este aleasă într-o clasă de precizie bună ($\pm 0,25\%$, de exemplu).

În această schemă este utilizat un senzor de temperatură centigrad, de tip LM35, a cărui ieșire variază cu 10mV pe grad Celsius. Stabilizatorul de tensiune ajustabil, de tip LM317, este utilizat în această schemă doar pentru referința sa internă, cu valoare de 1,25V, pentru a crea “zero-ul viu” al senzorului de temperatură cu ieșire unificată în curent.

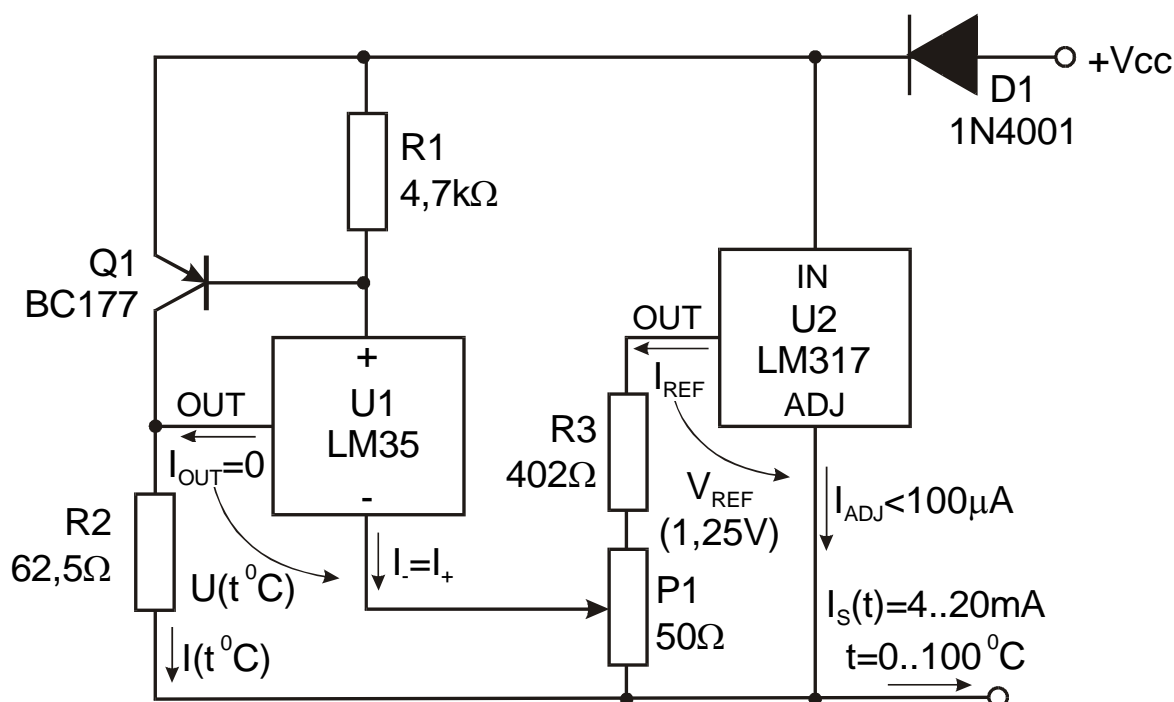


Fig. 8.3 Senzor de temperatură cu ieșire unificată în curent.

8.2.2 UTILIZAREA RESURSELOR UNITĂȚII CENTRALE DE PRELUCRARE

Din cadrul sistemului de dezvoltare, organizat în jurul unui microcontroller 80C552, și care constituie unitatea centrală de prelucrare locală a sistemului de măsurare și controlul temperaturii, descris detaliat în cadrul capitolului al treilea al acestei lucrări, sunt utilizate, în cadrul acestei aplicații, următoarele resurse:

- **microcontroller**-ul 80C552, utilizat în configurație de microprocesor pentru gestionarea achiziției analog-digitale pe 8 canale a temperaturii controlate din proces, pentru memorarea temporară a datelor prelevate în memoria de date a sistemului și pentru manipularea acestora prin intermediul interfeței seriale bidirecționale către calculatorul gazdă, pentru gestionarea a două ieșiri modulate în durată, utilizate pentru controlul continuu al temperaturii în cadrul procesului, pentru gestionarea porturilor de intrare-ieșire locale ale sistemului de dezvoltare;
- **memoria de program**, conținând programul monitor al sistemului de dezvoltare, utilizată pentru: încărcarea programului de aplicație (prin folosirea facilităților de down-load), transferul serial al datelor către calculatorul gazdă (prin folosirea facilităților de up-load) și bineînțeles,

fapt transparent pentru utilizator, inițializarea și programarea resurselor sistemului de dezvoltare (rutine de transmisie/recepție pe legătura serială, rutine de întreruperi interne și externe, etc.);

- **memoria de date**, utilizată pentru memorarea programului de aplicație și a eșantioanelor de temperatură prelevate din procesul controlat;
- **tastatura locală** de tip matriceal, cu 12 taste, este utilizată pentru programarea temperaturii de referință a sistemului controlat. Tastatura locală utilizează două porturi locale: un port de intrare, pe care este citit octetul care va furniza prin prelucrare software codul tastei acționate (cei mai semnificativi patru biți ai acestui octet conțin codul liniei baleiate, iar cei mai puțin semnificativi patru biți conțin codul coloanei pe care este situată tasta acționată) și un port de ieșire, de la care sunt utilizați doar cei patru biți mai puțin semnificativi, prin intermediul cărora sunt baleiate în buclă liniile tastaturii;
- **afișajul cu cristale lichide**, cu două linii de câte 16 caractere, este utilizat pentru afișarea continuă și comparativă a temperaturii de referință preprogramate și a temperaturii medii măsurate. Acest dispozitiv este utilizat pentru afișarea la începutul execuției programului de aplicație a câtorva mesaje de prezentare succintă a acesteia.

Programul de aplicație, scris în limbaj de asamblare 8051, asamblat și convertit în format INTEL HEX standard (acest format este executabil, conține un header format din adresa de început a programului din memoria RAM (specificată în cadrul programului sursă prin directiva “ORG adresă”), numărul de octeți ai programului (lungimea acestuia exprimată în număr de octeți) și informații de control (sumă de control) și este structurat sub forma unor linii de câte 16 octeți, reprezentați sub forma a două cifre hexa, reprezentând codurile instrucțiunilor utilizate în cadrul programului și o sumă de control, de asemenea sub forma unui octet, calculată ca reprezentând valoarea, exprimată sub forma a două cifre hexa, care adunată cu toți cei 16 octeți de informație conduce la valoarea FFH), conține rutine pentru:

- **inițializarea resurselor** utilizate în cadrul sistemului pentru măsurarea și controlul temperaturii: inițializarea afișajului cu cristale lichide (prin intermediul rutinei INITLCD care inițializează dialogul pe 8 biți cu driver-ele de afișaj, testează starea driver-elor de afișaj, realizează ștergerea afișajului și setează modul de lucru (display on, entry-mode);
- **citirea tastaturii** locale de tip matriceal, într-o secvență de program care utilizează o apelare de două ori a rutinei KEY pentru programarea temperaturii de referință.

Rutina de citire a tastaturii, denumită KEY, forțează ciclic pe fiecare linie a tastaturii valoarea binară "0", în timp ce toate celelalte linii sunt poziționate pe "1" logic. Se citește, prin intermediul portului de intrare,

configurația de linii și coloane, verificându-se dacă unul din biții reprezentând coloanele tastaturii este "0" logic. Dacă da, atunci a fost acționată tasta situată la intersecția liniei forțate la "0" (codul ei este preluat pe cei mai semnificativi 4 biți ai portului de intrare) și coloana pe care a fost detectat nivelul logic "0". Dacă nu, înseamnă că nu a fost acționată nici o tastă de pe linia baleiată și se trece la linia următoare. Rutina se execută în buclă până se acționează o tastă. În urma acționării unei taste (numerice), se trece la obținerea codului acesteia, prin intermediul unei secvențe de program care furnizează drept rezultat un octet ce conține pe cei patru biți mai puțin semnificativi codul BCD al tastei acționate. Prin convertirea acestui octet în format ASCII, utilizând rutina BINASC, codul obținut poate fi transmis către driver-ele de afișare ale display-ului LCD.

În cadrul secvenței de program pentru citirea tastaturii din cadrul acestei aplicații, se apelează de două ori rutina de citire a tastaturii și de afișare a codului ASCII al tastei acționate (pe poziții diferite și adiacente ale uneia dintre liniile de afișare ale display-ului LCD) și, de asemenea, se obține un octet unic, reprezentând valoarea temperaturii de echilibru programate, de tip împachetat BCD. Acest octet va fi memorat într-unul dintre registrele disponibile ale microcontroller-ului și va fi utilizat pentru controlul temperaturii;

- *gestionarea afișării* pe display-ul cu cristale lichide constă în utilizarea, în mod organizat, a rutinelor RDCMD (citire a informațiilor de stare de la driver-ele de afișare), WRCMD (scriere a comenzilor de setare a modului de lucru, de poziționare a cursorului, de setare a tipului de interfațare sau a mărimii caracterelor sau a tipului de cursor, la driver-ele de afișare), WLCD (testarea driver-elor de afișare în vederea transmiterii unui nou caracter), WRDAT (scrierea unui caracter ASCII la driver-ele de afișare), TRX1 (scrierea unui șir de caractere al display-ului cu cristale lichide, pentru afișarea unor mesaje sau a unor caractere la display-ul local
- *gestionarea achiziției* analog-digitale: specificarea canalului de intrare pe care se efectuează conversia, inițializarea prin software a conversiei analog-digitale, preluarea rezultatului și prelucrarea acestuia în vederea interpretării ulterioare.

Deoarece în cadrul acestei aplicații se folosesc 8 traductoare de temperatură, de tip LM135, acestea vor fi conectate pe intrările 0÷7 ale multiplexorului din microcontroller. Astfel, adresa canalului inițial selectat va fi 000, respectiv biții 0, 1, 2 din cadrul registrului ADCON vor fi poziționați la "0" prin cuvântul de programare. Este declanșat procesul de achiziție analog-digitală, prin software (este necesar ca bitul 6 al registrului de control ADCON, denumit ADEX, să fie

poziționat la "0") de pe canalul de intrare selectat prin poziționarea bitului 3 (ADCS) al registrului de control ADCON. Sfârșitul conversiei este detectat tot software prin citirea periodică (polling) a bitului 5 (ADCI) din registrul de control ADCON. când acest bit devine "1", procesul de conversiei s-a încheiat. Rezultatul, pe 10 biți, al conversiei analog-digitale este disponibil în registrele ADCH (cei 8 biți mai semnificativi) și ADCON (cei doi biți cei mai puțin semnificativi, fiind biții 7 și 8 din cadrul acestui registru). Rezultatul, considerat pe 8 biți, se află în registrul ADCH și va fi transferat în acumulator pentru prelucrări. În continuare, este incrementată adresa canalului de intrare selectat și se declanșează un nou proces de conversie, până când sunt baleiate toate cele 8 traductoare de temperatură. Toate rezultatele intermediare sunt memorate în memoria externă de date, iar când s-au efectuat toate cele 8 conversii analog-digitale rezultatele intermediare sunt mediate aritmetic, pentru a determina temperatura medie din incinta supravegheata.

În urma analizei efectuate anterior, s-a constatat că rezultatul binar al conversiei are o rezoluție de 2biți/grad. Astfel, prin împărțire la 2 a acestuia, se obține o valoare binară reprezentând exact temperatura măsurată, exprimată în grade Celsius. Urmează obținerea celor două cifre ale temperaturii măsurate, ce urmează a fi afișate: aceasta se realizează prin împărțirea la 10 a codului binar reprezentând temperatura (se obțin doar două cifre, deoarece s-a impus încadrarea temperaturii măsurate între 0 și 100°C). Cei doi octeți, reprezentând formatul BCD ale celor două cifre, sunt convertiți în format ASCII în vederea afișării. De asemenea, prin interschimbarea celor patru biți mai semnificativi cu cei patru biți mai puțin semnificativi ai octetului reprezentând cifra zecilor și prin efectuarea unei instrucțiuni SAU-LOGIC la nivel de octet se formează un octet împachetat BCD care conține informația referitoare la temperatura medie în cadrul procesului supravegheat. Acest octet, împreună cu acela care conține informația referitoare la temperatura preprogramată, vor fi utilizați în continuare pentru controlul temperaturii;

- *gestionarea ieșirilor modulate în durată* pentru controlul continuu al temperaturii în sistem este realizată în cadrul programului de aplicație prin programarea registrului PWMP pentru ca frecvența impulsurilor de ieșire să fie de 1kHz, conform relației:

$$f_{\text{PWM}} = \frac{f_{\text{OSC}}}{2 \cdot (1 + \text{PWMP}) \cdot 255} \quad (8.20)$$

Factorul de umplere al impulsurilor de ieșire este controlat, prin intermediul registrului de comandă PWM₁, de către diferența registrelor ce conțin octeții reprezentând temperatura programată și

temperatura curentă din cadrul sistemului controlat, conform relației:

$$\gamma = \frac{PWM_1}{255 - PWM_1}$$

$$PWM_1 = R_4 - R_5$$

$$PWM_2 = R_5 - R_4$$
(8.21)

în care registrul R4 conține octetul ce reprezintă temperatura programată la începutul procesului de control, iar registrul R5 conține octetul ce reprezintă temperatura medie măsurată. Componenta continuă a tensiunii de ieșire variază între 0V (la echilibru, când temperaturile programată și măsurată sunt egale) și 3V (atunci când diferența dintre temperatura programată și cea măsurată, exprimate sub forma a doi octeți împachetați BCD, este maximă, adică atunci când temperatura programată coincide cu capătul superior al intervalului de temperatură (+99°C) iar temperatura măsurată coincide cu capătul inferior al intervalului (0°C), conform ecuației:

$$(PWM_1)_{\min} = 0;$$

$$(PWM_1)_{\min} = 99_{BCD} = 153_Z;$$

$$(V_{out})_{\max} = \frac{153}{255} \cdot 5V = 3V$$
(8.22)

8.2.3 ETAJELE DE IEȘIRE PENTRU COMANDA ELEMENTELOR DE EXECUȚIE

Fiecare etaj de ieșire de comandă folosește ca informație primară impulsurile de la ieșirea corespunzătoare modulată în durată a microcontroller-ului 80C552 pe care o prelucrează (integrare cu un circuit simplu de tip RC și eșantionare-memorare în vederea obținerii componentei continue a acestora, inversarea polarității tensiunii și amplificare cu doi, sumare cu o tensiune de referință cu valoare ajustabilă în limite de $\pm 10\%$ în jurul tensiunii de 8V), pentru a o utiliza drept tensiune de comandă a unui circuit de comandă pe fază a tiristoarelor. Tiristorul comandat are drept sarcină elementul de încălzire sau elementul de ventilație, pentru controlul temperaturii.

Circuitul de integrare, de tip RC, este astfel proiectat încât să se încheie procesele tranzitorii de la ieșirea sa în situația în care factorul de umplere al impulsurilor de la ieșire este maxim (valoarea sa maximă este de 60%). Conform ecuației:

$$t_1 = RC \ln 10 = 2,3\tau = 60\% \cdot \frac{1}{1\text{kHz}} = 0,6\text{ms}$$
(8.23)

se alege $C=62\text{nF}$ și $R=10\text{k}\Omega$.

Circuitul de eșantionare-memorare folosit, de tip LF198, este un circuit realizat monolitic, ce utilizează tehnologia BI-FET pentru a obține o exactitate superioară atât în curent continuu, cât și pentru achiziția semnalelor rapid variabile. Funcționând ca circuit repetor, acesta este caracterizat printr-o exactitate de 0,002% a amplificării și de un timp de achiziție de 5 μs , pentru o exactitate de 0,01%. Folosirea tehnologiei bipolare în realizarea etajului de intrare asigură tensiuni de offset mici și o bandă largă de frecvență (1 MHz), fără probleme de stabilitate. Impedanta de intrare, de $10^{10} \Omega$, permite achiziționarea semnalelor de excitație ce provin de la surse de semnal cu impedanță internă ridicată, fără a fi afectată exactitatea. Amplificatorul de ieșire combină dispozitive bipolare și tranzistoare JFET cu canal P, pentru a putea asigura rate de cădere de 5mV/min, utilizând o capacitate de memorare de 1 μF .

Frontul căzător al fiecărui impuls de la ieșirea modulată în durată va determina circuitul de eșantionare-memorare să intre în starea de memorare, reținând de fapt valoarea maximă de tensiune de la ieșirea circuitului de integrare.

Ieșirea circuitului de eșantionare-memorare este aplicată pe una dintre cele două intrări ale unui sumator inversor cu amplificare -2. Pe cea de-a doua intrare se aplică o tensiune de referință cu valoarea de -4V. La ieșirea sumatorului se obține o tensiune pozitivă variind între 8V (la echilibru termic) și respectiv 2V (când diferența dintre temperaturile programată și măsurată este maximă). Această tensiune constituie semnalul de control al circuitului de comandă a tiristorului ce acționează elementul de încălzire/ventilație. Diagrama de funcționare a acestui circuit este reprezentată în fig. 8.4.

Descrierea funcționării circuitului βAA145 poate fi rezumată astfel:

- funcționarea circuitului este sincronizată cu frecvența rețelei electrice de alimentare (prin intermediul unui divizor rezistiv de tensiune căruia i se aplică tensiunea rețelei). Acest divizor este constituit din două rezistențe $R_1=22\text{k}\Omega$ și $R_2=8,2\text{k}\Omega$. Puterea disipată de către rezistența R_1 este de circa 2,2W și, de aceea, se alege $R_1=22\text{k}\Omega/3\text{W}$;
- generatorul de rampă din structura circuitului generează o tensiune liniar variabilă sincronizată cu frecvența rețelei, așa după cum reiese din fig. 8.4. Această tensiune, denumită v_R , are o amplitudine de 8V, amplitudine determinată de blocul intern de alimentare a circuitului;
- pe intrarea de comandă a fazei se aplică o tensiune de control, denumită v_S , provenită de la ieșirea circuitului de eșantionare-memorare;
- ori de câte ori se manifestă coincidența între tensiunea de comandă și rampa crescătoare a tensiunii liniar variabile, se generează un impuls de comandă pe poarta tiristorului (triacului), a cărui durată este determinată de elementele de temporizare ale unui circuit basculant

monostabil din structura circuitului β AA145. Elementele de temporizare, $P_2=250k\Omega$, $R_6=5,6k\Omega$, $C_3=47nF$, permit reglarea duratei impulsurilor de comandă între 0,1ms și 4ms. Comanda triacului se face în cadranele I și III prin implementarea unei funcții de tip ȘI-cablat între ieșirile de comandă ale circuitului. Aceste ieșiri sunt de tip colector în gol și au prevăzute rezistențe de pull-up, cu valoarea de 820Ω . Deoarece curentul de poartă al triacului este de maxim 300mA, este necesară buffer-area în curent a acestora cu ajutorul unui tranzistor de medie putere, de tip BD135/137/139.

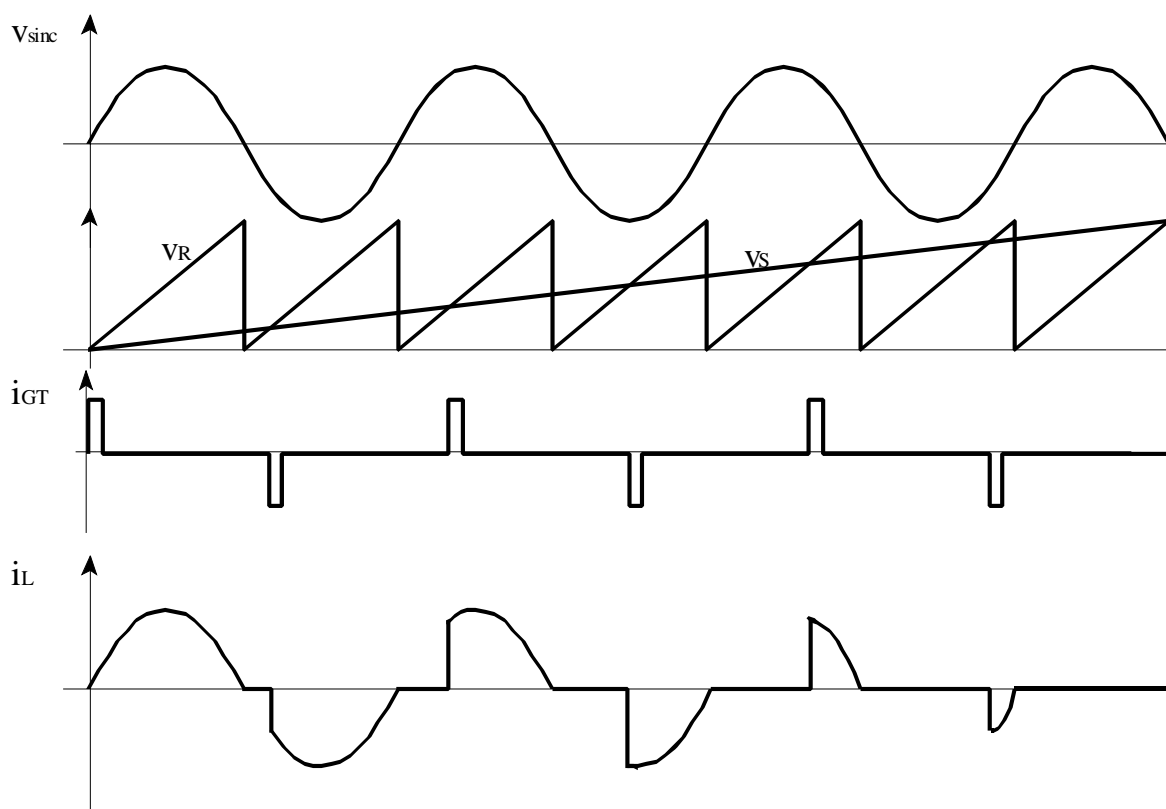


Fig. 8.4 Diagrama temporală de funcționare a elementelor de comandă.

Schema de principiu a etajului de comandă pe fază a triacului este prezentată în fig. 8.5.

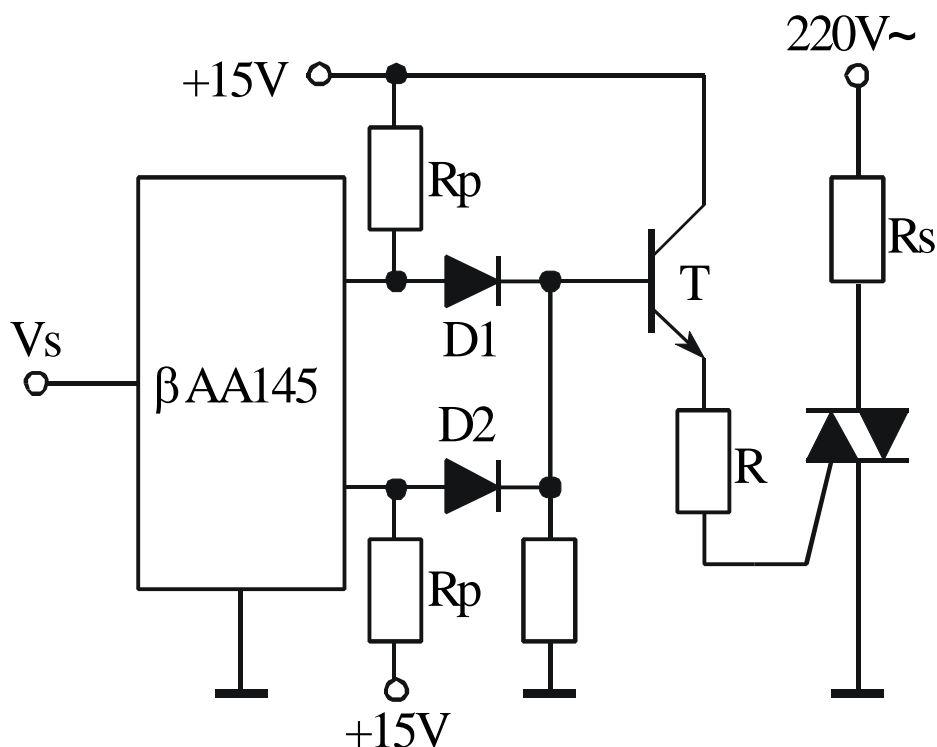


Fig. 8.5 Schema de principiu a etajului de comandă pe fază.

8.3 SOFTWARE DE ANALIZĂ A REZULTATELOR

Software-ul de analiză și monitorizare a temperaturilor prelevate dintr-un proces este realizat sub forma unui *instrument virtual*, implementat cu ajutorul programului **HP VEE - for Windows** -versiunea 3.12 (July 07 1995) @ Copyright Hewlett-Packard Corporation 1991-1995.

Acest program lucrează cu obiecte predefinite sau create de către utilizator, care sunt plasate în spațiul de lucru și care sunt interconectate pentru a realiza o diagramă-bloc executabilă. Fiecare obiect permite vizualizarea sa în două moduri:

- modul de vizualizare restrânsă (ca *icon* în **Windows**);
- modul de vizualizare detaliată (*detail view*).

De asemenea, fiecare obiect este caracterizat de un meniu propriu, care permite modificarea dimensiunii, poziției, titlului, precum și altor attribute ale acestuia.

Am fost creat *un instrument virtual de monitorizare a temperaturii*, intitulat **PROTERM**. Acest instrument este, de fapt, un obiect creat de utilizator, ce permite:

- demararea procesului de analiză, prin acționarea butonul “**Run**”, prezent pe panoul global al oricărei aplicații **HP VEE**. La demararea procesului de analiză, este deschisă o fereastră din care poate fi selectat

fișierul de date folosit ca punct de plecare în cadrul reprezentării grafice a temperaturilor. Fișierele de date sunt de tip standard de date, cu extensia **.dat**, conținând pe câte doi octeți eșantioanele de pe un număr de maxim opt canale;

- selectarea mărimilor de intrare prin intermediul unor liste circulare ce dispun de următoarele opțiuni:
 - vizualizarea formei de variație a două temperaturi din cele maxim opt posibile;
 - selectarea valorii inițiale a eșantioanelor ce urmează a fi afișate, prin intermediul unui comutator rotativ, denumit **knob**, în gama 0÷20000 și cu o rezoluție de 256 puncte. Acest **knob** poartă denumirea sugestivă de “**From sample...**”;
 - selectarea valorii finale a eșantioanelor ce urmează a fi afișate, de asemenea prin intermediul unui **knob**, gradat între 0 și 20000, cu o rezoluție de 256 puncte. Acest **knob** poartă de numirea sugestivă de “**...To sample**”;
 - vizualizarea simultană a doi parametri: formă de semnal de comandă, formă de semnal de răspuns al sistemului, cu ajutorul a două instrumente de tip osciloscop, care indică:

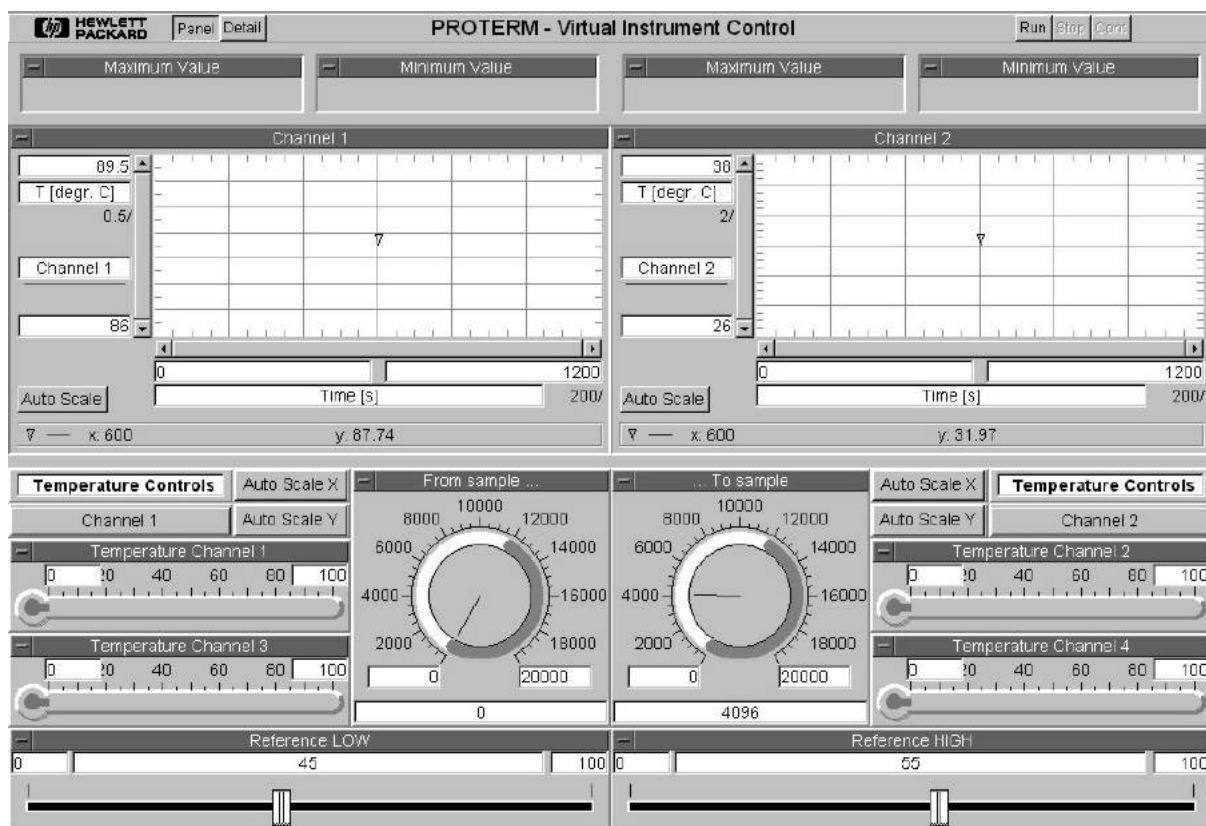


Fig. 8.6 Panoul frontal al instrumentului virtual **PROTERM**.

- prin intermediul *titlului*, imaginile grafice ce se vizualizează la un

moment dat;

- indicații ale mărimilor corespunzătoare fiecărei axe, cum ar fi: *numele* mărimii (de exemplu: *timp* pentru axa Ox, respectiv *temperatură*, pentru axa Oy), *unitatea de măsură* a acesteia, *intervalul de vizualizare* și *gradarea axelor* (unități pe diviziune);
- prin intermediul unui **marker**, se pot obține informații, legate de valoarea instantanee a semnalelor, în funcție de timp;
- instrumentul de vizualizare prezintă facilități de autoscalare, fie independent pe fiecare axă prin intermediul unor butoane dedicate fiecărui instrument, fie simultan pe ambele axe prin intermediul unui buton cu care este echipat instrumentul de vizualizare de tip osciloscop;
- afișarea, sub formă numerică, a *valorilor minime și maxime a parametrilor*, atât pentru comandă, cât și pentru răspuns, prin intermediul a patru indicatoare alfanumerice.

Trebuie menționat că toate **butoanele** și **knob**-urile, prin intermediul cărora este controlat procesul de achiziție și analiză, prezintă facilități de “**autoexecute**”, ceea ce înseamnă că acționarea oricăruia dintre ele determină declanșarea unui nou proces de analiză.

În fig. 8.6 este reprezentat panoul frontal al instrumentului virtual **PROTERM**.

8.3.1 IMPLEMENTAREA ANALIZORULUI PROTERM

Fig. 8.7 conține reprezentarea detaliată a blocurilor care compun instrumentul virtual intitulat **PROTERM** și bazat pe platforma Windows HP VEE 3.12.

Aceste blocuri funcționale sunt:

- *modulul de prelucrare primară a fișierului de date de intrare*, primind ca intrare un fișier cu extensia **.dat**. Acest bloc furnizează ca ieșiri un număr de maxim opt vectori conținând eșantioanele sub formă numerică a mărimilor de intrare;
- *circuitele de control ale dispozitivelor de afișare*;
- *dispozitivele de afișare*, de tip oscilograf;
- *blocul de afișare sub formă digitală a valorilor minime/maxime ale semnalelor de intrare*.

Toate aceste blocuri funcționale sunt înglobate într-un obiect utilizator nou creat, având două posibilități de reprezentare:

- reprezentarea detaliată - **detail** - (fig. 8.7), în care sunt puse în evidență blocurile și interconexiunile funcționale;
- reprezentarea de tip panou - **panel** -, care ilustrează panoul propriu-zis

al analizorului, adică dispozitivul de afișare, reprezentat în fig. 8.6.

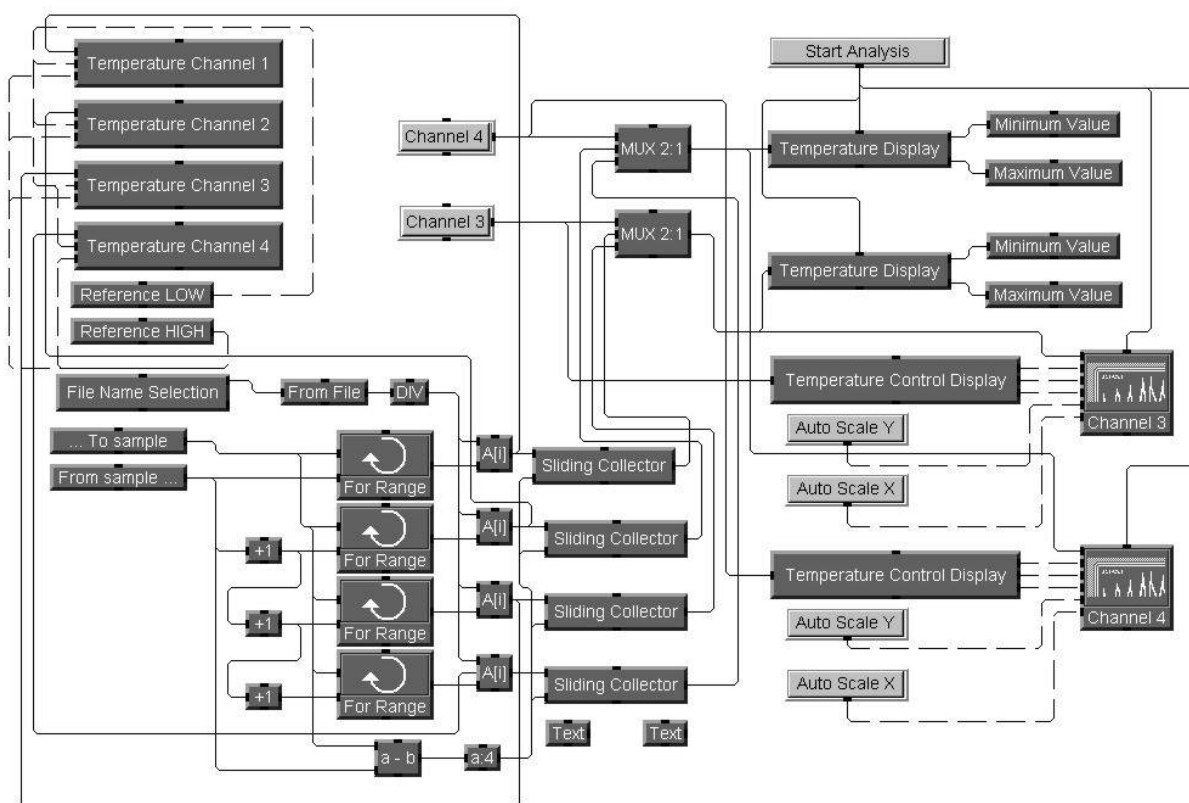


Fig. 8.7 Reprezentarea detaliată a analizorului **PROTERM**.

8.4 LISTINGUL APLICAȚIEI DE MĂSURARE ȘI CONTROL A TEMPERATURII

```
; *****
; PROGRAM PENTRU ECHIPAMENTE DE MASURARE SI CONTROL A TEMPERATURII
; CU UNITATE CENTRALA DE PRELUCRARE LOCALA CU MICROCONTROLLER 80C552
; IN LIMBAJ DE ASAMBLARE AL FAMILIEI 80X51
; *****
      ORG      8000H          ; adresa de inceput a programului
; secventa de declarare a resurselor suplimentare ale
; microcontroller 80C552 utilizate in cadrul aplicatiei
      ADCON   equ    0C5H    ; adresa registrului ADCON
      ADCH    equ    0C6H    ; adresa registrului ADCH
      PWMP    equ    0FEH    ; adresa registrului PWMP
      PWM0    equ    0FCH    ; adresa registrului PWM0
      PWM1    equ    0FDH    ; adresa registrului PWM1
; secventa de initializare echipamentului si de programare a
; parametrilor de functionare (introducerea valorii temperaturii
; programate).
      LJMP    INIT           ; salt la eticheta INIT
INIT:  ACALL  INILCD          ; apel rutina initializare LCD
      ACALL  CLRLCD          ; apel rutina stergere LCD
```

```
MOV    DPTR,#TEXT    ; se memoreaza in DPTR adresa de in-
                        ; ceput a sirului de caractere TEXT
ACALL  TRX1           ; apel rutina de transmisie mesaj
MOV    R2,#0C0H       ; pozitionare pe a doua linie
ACALL  WRCMD          ; scriere comanda pozitionare la LCD
INC    DPTR           ; incrementare DPTR (linia a doua)
ACALL  TRX1           ; apel rutina de transmisie mesaj
ACALL  SEC            ; mentine starea afisajului 3 sec.
ACALL  CLRLCD         ; apel rutina stergere LCD
MOV    DPTR,#TEXT1    ; se memoreaza in DPTR adresa de in-
                        ; ceput a sirului de caractere TEXT1
ACALL  TRX1           ; apel rutina de transmisie mesaj
MOV    R2,#1100B      ; display on, cursor off, flash off
ACALL  WRCMD          ; scrie comanda la LCD
MOV    R2,#8DH        ; scrie in R2 pozitia cursorului
ACALL  WRCMD          ; pozitionare cursor
MOV    R2,#0DFH       ; codul caracterului grade
ACALL  TRX            ; scrie codul caracterului la LCD
MOV    R2,#0C0H       ; pozitionare pe a doua linie
ACALL  WRCMD          ; scriere comanda pozitionare la LCD
INC    DPTR           ; incrementare DPTR (linia a doua)
ACALL  TRX1           ; apel rutina de transmisie mesaj
MOV    R2,#1100B      ; 1,display on,cursor off, flash off
ACALL  WRCMD          ; scrie comanda la LCD
MOV    R2,#0CDH       ; scrie in R2 pozitia cursorului
ACALL  WRCMD          ; pozitionare cursor
MOV    R2,#0DFH       ; codul caracterului grade
ACALL  TRX            ; scrie codul caracterului la LCD
MOV    A,#20          ; incarca acumulatorul cu #20 pentru
                        ; frecventa semnalului de iesire sa
                        ; fie 1kHz=11,059MHz/(2*(1+20)*255)
MOV    PWMP,A         ; se programeaza registrul PWMP
ACALL  KEY            ; citire tastatura
MOV    R2,#1110B      ; display on, cursor off, flash off
ACALL  WRCMD          ; scrie comanda la LCD
MOV    R2,#08BH       ; adresa scriere cod tasta
ACALL  WRCMD          ; pozitionare cursor
MOV    A,R1           ; codul hexa al tastei in acumulator
MOV    R4,A           ; transfera acumulatorul in R4
CALL   HEXASC         ; conversie ASCII a codului tastei,
                        ; pentru afisare pe LCD
MOV    R2,A           ; codul hexa al tastei in acumulator
ACALL  TRX            ; scrie la LCD codul tastei
ACALL  SEC            ; rutina intarziere
ACALL  KEY            ; citire tastatura
MOV    A,R1           ; codul hexa al tastei in acumulator
MOV    R5,A           ; transfera acumulatorul in R5
CALL   HEXASC         ; conversie ASCII a codului tastei,
                        ; pentru afisare pe LCD
MOV    R2,A           ; codul HEXA al tastei apasate in R2
ACALL  TRX            ; scrie la LCD codul tastei
MOV    A,R4           ; transfera in acc. registrul R4
SWAP   A              ; interschimb biti acumulator
ORL    A,R5           ; sau logic intre acumulator si R5
MOV    R4,A           ; transfer acc. in R4 - va contine
```

```

; forma BCD a temperaturii prescrise
; secventa de masurare in bucla a celor 8 valori ale temperaturii,
; de la cele 8 traductoare de temperatura, calculul temperaturii din
; valorile binare citite de la convertorul A/D si stocarea in
; memorie a fiecarui rezultat incepand de la adresa B000H
ACH:  MOV    DPTR,#0B000H; adresa de inceput a bufferului
      CLR    A           ; initializeaza acumulatorul cu 0
      MOV    R1,#00H     ; R1 initializat va memora canalul
                          ; de intrare selectat
      MOV    R5,#08H     ; numarul maxim de canale de intrare
ACH11: MOV    A,R1        ; selectarea canalului de intrare
      ORL    A,#08H      ; bitii de control
      MOV    ADCON,A      ; start conversie prin software
WAIT:  MOV    A,ADCON     ; A:=registrul de stare al A/D
      JNB    ACC.4,WAIT   ; asteapta terminarea conversiei
      MOV    A,ADCH       ; citeste rezultatul conversiei
      MOV    B,#2         ; B pentru impartire la 2
      DIV    AB          ; obtine cod binar temperatura
      MOVX   @DPTR,A      ; salveaza in memorie temperatura
                          ; citita de la traductorul "i"
      INC    DPTR         ; incrementeaza pointerul adresei
                          ; din bufferul de memorie
      INC    R1           ; adresa canalului urmator
      DJNZ   R5,ACH11     ; se decrementeaza registrul R5 si
                          ; salt la conversia A/D pe canalul
                          ; urmator daca R5 nu este nul

; secventa de calcul a mediei aritmetice a celor 8 valori ale tem-
; peraturii, in raport cu care se efectueaza reglarea temperaturii.
; calculul sumei celor 8 valori ale temperaturilor masurate
      MOV    R5,#08H      ; numarul maxim de valori de sumat
      MOV    DPTR,#0B000H ; adresa de inceput a bufferului de
                          ; memorie
      MOV    R1,#00H      ; initializarea octet inferior suma
      MOV    R2,#00H      ; initializarea octet superior suma
SUM:   MOVX   A,@DPTR     ; citire in acc. valoare din buffer
      ADD    A,R1         ; adunare cu valoare anterioara
      MOV    R1,A         ; salvare suma partiala (octet inf.)
      MOV    A,R2         ; citire octet sup. suma partiala
      ADDC   A,#0         ; eventual propagarea transportului
      MOV    R2,A         ; salvare octet superior rezultat
      INC    DPTR         ; incrementare adresa buffer memorie
      DJNZ   R5,SUM       ; decrementare contor R5 si salt la
                          ; calculul unei alte sume partiale
                          ; daca contor nenul

; calculul valorii medii a temperaturii, prin impartirea sumei celor
; 8 valori ale temperaturilor la valoarea 8, ceea ce este echivalent
; cu shiftarea la dreapta cu 3 pozitii a rezultatului sumei obtinute
; anterior pe 2 octeti.
; rezultatul (valoarea medie a temperaturii) este continut in R1
      MOV    R3,#03H      ; R3 initializat cu nr de shifturi
SHIFTR:
      MOV    A,R2         ; citeste octet superior in A
      CLR    C           ; stergere CARRY
      RRC    A           ; rotire la dreapta prin CARRY
      MOV    R2,A         ; salvare octet superior

```

```
MOV    A,R1          ; citire octet inferior
RRC    A              ; rotire la dreapta prin CARRY
MOV    A,R1          ; salvare octet inferior
DJNZ   R3,SHIFTR     ; decrementare contor si efectuarea
                      ; unei noi rotatii pe 2 octeti daca
                      ; continutul contorului este nenul
; secventa de transmisie seriala a valorii temperaturii medii
; masurate folosind interfata seriala de comunicatie a sistemului de
; masurare si control a temperaturii (1 bit de START, 8 biti de
; date, 1 bit STOP, fara semnale hardware de protocol, cu protocol
; software XON-XOFF)
MOV    A,R1          ; citesc in acumulator valoarea
                      ; temperaturii medii
MOV    SBUF,A        ; pun aceasta valoare in reg. de
                      ; transmisie pe seriala
CLR    TI            ; reset bit TI (Transmit Interrupt)
WAIT1:JNB TI,WAIT1   ; se asteapta in bucla pozitionarea
                      ; pe 1 a acestui bit (octet preluat)
CLR    TI            ; initializare bit TI
; secventa de afisare locala pe LCD a valorii medii a temperaturii
; masurate (calcul cifrelor acesteia printr-un algoritm iterativ de
; impartire la 10, calcul valoare ASCII a cifrelor obtinute si
; afisare propriu-zisa) si codificarea sub forma a unui octet
; impachetat BCD a acestei valori medii, memorat in registru R5.
MOV    R2,#1100B     ; 1,display on, cursor on, flash off
ACALL  WRCMD         ; scriere comanda
MOV    R2,#0CBH      ; adresa pentru prima cifra
ACALL  WRCMD         ; pozitionare cursor
MOV    A,R1          ; valoare binara temperatura medie
MOV    B,#10         ; B pentru impartire la 10
DIV    AB            ; in A cifra semnificativa a temp.
ANL    A,#00FH       ; mascheaza bitii mai semnificativi
MOV    R6,A          ; salveaza acumulatorul in R6
ACALL  HEXASC        ; convertire in cod ASCII
MOV    R2,A          ; in R2 codul caracterului
ACALL  TRX           ; transmite caracter la LCD
MOV    A,B           ; in A cifra mai putin semnificativa
                      ; a temperaturii
ANL    A,#00FH       ; mascheaza bitii mai semnificativi
MOV    R7,A          ; salveaza acumulatorul in R7
ACALL  HEXASC        ; conversie in cod ASCII
MOV    R2,A          ; R2 contine codul de afisat
ACALL  TRX           ; scrie la LCD a doua cifra
MOV    A,R6          ; transfera acumulatorul cu R6
SWAP   A             ; interschimb biti acumulator
ORL    A,R7          ; sau logic intre acumulator si R7
MOV    R5,A          ; salveaza acc in R5-va contine
                      ; forma BCD a temp. medii masurate
; secventa de reglare a temperaturii prin programarea celor doua
; iesiri modulate in durata (programarea factorilor de umplere cu o
; valoare proportionala cu diferenta intre valoarea programata si
; valoarea medie).
; doar una dintre iesiri va fi activa la un moment dat pentru
; controlul elementului de incalzire sau de racire.
MOV    A,R4          ; citire in A a temp. programate
```

```

SUBB  A,R5      ; scadere cu temperatura medie
JC    LABEL1    ; daca rezultatul este negativ
                ; salt la eticheta LABEL1
CPL   A         ; daca rezultatul este pozitiv, este
                ; complementat, pt. ca iesirea modu-
                ; lata in durata este complementata
MOV   PWM1,A    ; programare iesire modulata
CLR   A         ; initializare acumulator
CPL   A         ; complementare acumulator
MOV   PWM0,A    ; dezactivare iesire modulata
                ; pentru comanda element de racire
; pentru comanda elementului de incalzire
SJMP  LABEL2    ; salt neconditionat la LABEL2
LABEL1:
CLR   A         ; initializare acumulator
CPL   A         ; complementare acumulator
MOV   PWM1,A    ; dezactivare iesire modulata
                ; pentru comanda element incalzire
MOV   A,R5      ; citire in A a temperaturii medii
SUBB  A,R5      ; scadere cu temperatura programata
CPL   A         ; rezultatul este complementat,
                ; pentru ca iesirea modulata
                ; este complementata
MOV   PWM0,A    ; programare iesire modulata
                ; pentru comanda element de racire
LABEL2:
ACALL SEC       ; rutina de intarziere
AJMP  ACH       ; reluare proces de achizitie
;
; secventa de subrutine scrise si utilizate in cadrul programului
; principal
;
; Rutina de citire a unei cifre zecimale de la o tastatura
; matriciala cu trei linii si patru coloane. Rezultatul citirii sub
; forma HEXA se afla in registrul R1.
;
KEY:  MOV  A,#0E0H ; adresa primei linii de taste
      ACALL PORT  ; subrutina de scriere/citire port
      JZ   NEXT   ; daca acumulatorul este 0 (nu s-a
                  ; tastat nimic) salt la eticheta NEXT
      JNZ  COL    ; daca acumulatorul este nenul, salt la
                  ; eticheta COL
NEXT: MOV  A,#0D0H ; adresa liniei a doua de taste
      ACALL PORT  ; apel subrutina de scriere/citire port
      JZ   NEXT1  ; daca acumulatorul este zero, salt la
                  ; eticheta NEXT1
      JNZ  COL    ; daca acumulatorul este nenul, salt la
                  ; eticheta COL
NEXT1:
      MOV  A,#0B0H ; adresa liniei a treia de taste
      ACALL PORT  ; apel subrutina de scriere/citire port
      JZ   KEY    ; daca acumulatorul este nul, proces
                  ; ciclic de citire taste
      JNZ  COL    ; daca acumulatorul este nenul, salt la
                  ; eticheta COL, pentru determinarea co-

```

```
COL:  JB      ACC.0,COL0      ; loanei pe care se afla tasta apasata
                                           ; daca bitul 0 al acumulatorului este 1,
                                           ; tasta se afla pe coloana 0 si salt la
                                           ; eticheta COL0
        JB      ACC.1,COL1      ; daca bitul 1 al acumulatorului este 1,
                                           ; tasta se afla pe coloana 1 si salt la
                                           ; eticheta COL1
        JB      ACC.2,COL2      ; daca bitul 2 al acumulatorului este 1,
                                           ; tasta se afla pe coloana 2 si salt la
                                           ; eticheta COL2
        JB      ACC.3,COL3      ; daca bitul 3 al acumulatorului este 1,
                                           ; tasta se afla pe coloana 3 si salt la
                                           ; eticheta COL3
COL0:  MOV      R1,#00H        ; registrul R1 se incarca cu 0
        MOV      A,R3          ; se transfera in acumulator registrul R3
        SWAP     A              ; interschimb biti acumulator
        ANL      A,#00001111B  ; mascare biti mai semnificativi
        JB      ACC.1,LINIE1    ; daca bitul 1 al acumulatorului este 1,
                                           ; salt la eticheta LINIE1
        JB      ACC.2,LINIE2    ; daca bitul 2 al acumulatorului este 1,
                                           ; salt la eticheta LINIE2
        SJMP     PLAY          ; salt la eticheta PLAY
COL1:  MOV      R1,#01H        ; registrul R1 se incarca cu 1
        MOV      A,R3          ; se transfera in acumulator registrul R3
        SWAP     A              ; interschimb biti acumulator
        ANL      A,#00001111B  ; mascare biti mai semnificativi
        JB      ACC.1,LINIE1    ; daca bitul 1 al acumulatorului este 1,
                                           ; salt la eticheta LINIE1
        JB      ACC.2,LINIE2    ; daca bitul 2 al acumulatorului este 1,
                                           ; salt la eticheta LINIE2
        SJMP     PLAY          ; salt la eticheta PLAY
COL2:  MOV      R1,#02H        ; registrul R1 se incarca cu 2
        MOV      A,R3          ; se transfera in acumulator registrul R3
        SWAP     A              ; interschim biti acumulator
        ANL      A,#00001111B  ; mascare biti mai semnificativi
        JB      ACC.1,LINIE1    ; daca bitul 1 al acumulatorului este 1,
                                           ; salt la eticheta LINIE1
        JB      ACC.2,LINIE2    ; daca bitul 2 al acumulatorului este 1,
                                           ; salt la eticheta LINIE2
        SJMP     PLAY          ; salt la eticheta PLAY
COL3:  MOV      R1,#03H        ; registrul R1 se incarca cu 3
        MOV      A,R3          ; se transfera in acumulator registrul R3
        SWAP     A              ; interschimb biti acumulator
        ANL      A,#00001111B  ; mascare biti mai semnificativi
        JB      ACC.1,LINIE1    ; daca bitul 1 al acumulatorului este 1,
                                           ; salt la eticheta LINIE1
        JB      ACC.2,LINIE2    ; daca bitul 2 al acumulatorului este 1,
                                           ; salt la eticheta LINIE2
        SJMP     PLAY          ; salt la eticheta PLAY
LINIE1: MOV      A,#04H        ; acumulatorul se incarca cu 4
        ADD      A,R1          ; aduna acumulatorul cu R1
        MOV      R1,A          ; se transfera acumulatorul in R1
        SJMP     PLAY          ; salt la eticheta PLAY
LINIE2:
```

```

        MOV     A,#08H           ; acumulatorul se incarca cu 8
        ADD     A,R1             ; aduna acumulatorul cu R1
        MOV     R1,A             ; se transfera acumulatorul in R1
        SJMP    PLAY             ; salt la eticheta PLAY
PLAY:    RET                     ; revenire din subrutina
;
; Rutina de scriere la un port de iesire si de citire de la un port
; de intrare. Informatiile sunt vehiculate prin intermediul
; acumulatorului.
;
PORT:    MOV     P2,#1           ; selecteaza decodificatorul (A8=1)
        MOV     R0,#20H         ; incarca R0 cu 0010 0000 B:
                                   ; selectie port de iesire 1 (S1=0)
        MOVX    @R0,A           ; muta continutul acumulatorului
                                   ; la portul de iesire 1 (AX0=0)
        MOV     P2,#1           ; selecteaza decodificatorul (A8=1)
        MOV     R0,#60H         ; incarca R0 cu 0110 0000B:
                                   ; selecteaza portul de intrare
        MOVX    A,@R0           ; citeste in acumulator valoarea gasita
                                   ; la adresa specificata anterior
        CPL     A               ; complementeaza acumulatorul
        MOV     R3,A            ; salveaza valoarea in registrul R3
        ANL     A,#0FH          ; mascheaza bitii mai semnificativi
                                   ; ai acumulatorului (codul liniei)
        RET                     ; revenire din subrutina
;
; Rutina de scriere sir de caractere la afisajul cu cristale lichide
; LCD.
;
TRX1:    CLR     A               ; initializeaza acumulatorul cu 0
        MOVC    A,@A+DPTR       ; muta in acumulator adresa gasita
                                   ; la adresa @A+DPTR
        CJNE    A,#24H,TRCAR1   ; compara continutul acumulatorului
                                   ; cu #24H (codul hexa al caracterului
                                   ; $ - sfarsit de mesaj) si daca este
                                   ; diferit se efectueaza salt la eticheta
                                   ; TRCAR1
        RET                     ; revenire din subrutina
TRCAR1:  MOV     R2,A            ; muta continutul acumulatorului in
                                   ; registrul R2
        ACALL   WRDAT           ; apel subrutina de scriere date in
                                   ; registrele afisajului LCD
        INC     DPTR            ; adresa urmatorului caracter de scris
        SJMP    TRX1           ; small jump la eticheta TRX1, pentru
                                   ; a scrie urmatorul caracter
;
; Rutina de intarziere cu 0.5 secunde:
; 0.5sec = (7instr.NOP x 12perioade ceas/NOP x 256 x 256)/11,059MHz.
;
SEC:     MOV     R6,#255         ; se incarca registrul R6 cu #255
LOOP1:   MOV     R7,#255         ; se incarca registrul R7 cu #255
LOOP:    NOP                     ; 7 instructiuni NOP
        NOP
        NOP

```

```
NOP
NOP
NOP
NOP
DJNZ    R7,LOOP    ; se decrementeaza registrul R7 si
                  ; daca este nenul, salt la LOOP
DJNZ    R6,LOOP1    ; se decrementeaza registrul R6 si
                  ; daca este nenul, salt la LOOP1
RET      ; revenire din subrutina
;
TEXT: DB 'PROGRAMARE TEMP:$'    ; text prezentare 1, linia 1
      DB '(TEMP.: 0 .. 99)$'    ; text prezentare 1, linia 2
TEXT1: DB 'TEMP. REF: 00 C $'   ; text prezentare 2, linia 1
      DB 'TEMP. MAS: 00 C $'   ; text prezentare 2, linia 2
;
; Rutina de conversie hexa-ascii. Sursa si destinatia sunt
; constituite de acumulator.
;
HEXASC:
      ANL    A,#0FH            ; se mascheaza cei patru biti mai
                  ; semnificativi ai acumulatorului
      JNB    ACC.3,NOADJ       ; daca bitul 3 = 0 salt la NOADJ
      JB     ACC.2,ADJ         ; daca bitul 2 = 1 salt la ADJ
      JNB    ACC.1,NOADJ       ; daca bitul 1 = 0 salt la NOADJ
ADJ:   ADD    A,#07H           ; aduna acumulatorul cu #07H
NOADJ: ADD    A,#30H           ; aduna acumulatorul cu #30H
      RET      ; revenire din subrutina
;
; Rutina de transmisie la LCD a unui caracter ASCII continut in
; registrul R2.
;
TRX:   CLR    A                ; se initializeaza acumulatorul cu 0
      MOV    A,R2              ; se transfera continutul registrului
                  ; R2 in acumulator
      LJMP   TRCAR             ; long jump la eticheta TRCAR
LL7:   RET                    ; revenire din subrutina
TRCAR: MOV    R2,A              ; se restaureaza in R2 continutul Acc.
      ACALL  WRDAT             ; se scriu datele in driverele LCD
      SJMP   LL7              ; short jump la eticheta LL7
;
; Rutina de initializare LCD. Aceasta rutina trimite la LCD comanda
; #38H = 0011 1000B. Conform datelor de catalog, acesta comanda
; seteaza urmatoorii parametri ai afisajului:
;   - bitul 5 - defineste aceasta comanda (function set);
;   - bitul 4(DL-Data Length)=1 seteaza dialogul pe 8 biti
;     intre procesor si LCD;
;   - bitul 3(N-Number of display lines)=1 seteaza numarul
;     liniilor afisajului ca fiind 2^N=2;
;   - bitul 2(F-Character Font)=0 seteaza forma caracterului
;     ca fiind 5*7 puncte.
;
INILCD:
      MOV    R2,#38H           ; incarca R2 cu cuvantul de comanda
                  ; pregatind dialogul pe 8 biti cu
                  ; driverele de afisaj
```



```

        ACALL  WRCMD          ; transmite comanda anterioara la
                                ; port, efectuand programarea LCD
        MOV    R4,#50         ; se incarca registrul R4 cu valoarea
                                ; imediata 50
DEL4MS:
        ACALL  DELAY          ; apel subrutina DELAY
        DJNZ   R4,DEL4MS      ; se decrementeaza registrul R4 si
                                ; salt la eticheta DEL4MS daca conti-
                                ; nutul acestuia este nenul
        MOV    R4,#4          ; se incarca R4 cu valoarea imediata 4
LINI:   ACALL  WRCMD          ; se scrie comanda la driverele LCD
        ACALL  DELAY          ; apel subrutina DELAY
        DJNZ   R4,LINI        ; se decrementeaza registrul R4 si salt
                                ; la eticheta LINI daca este nenul con-
                                ; tinutul acestui registru
        ACALL  WLCD           ; testarea starii driverelor LCD
        MOV    R2,#6          ; seteaza modul de lucru "entry"
        ACALL  WRCMD          ; transmite comanda la LCD
        ACALL  WLCD           ; testarea starii driverelor LCD
        MOV    R2,#0EH        ; seteaza modul "display on"
        ACALL  WRCMD          ; transmite comanda la LCD
        ACALL  WLCD           ; testarea starii driverelor LCD
        MOV    R2,#1          ; seteaza modul "clear LCD"
        ACALL  WRCMD          ; transmite comanda la LCD
        ACALL  WLCD           ; testarea starii driverelor LCD
        RET                    ; revenire din subrutina
;
; Subrutina de stergere a afisajului cu cristale lichide si setare a
; modului de lucru.
;
CLRLCD:
        MOV    R2,#1          ; stergere afisaj LCD
        ACALL  WRCMD          ; transmite comanda la LCD
        MOV    R2,#1100B      ; display on, cursor off, flash off
        ACALL  WRCMD          ; transmite comanda la LCD
        RET                    ; revenire din rutina de CLRLCD
;
; Rutina ce testeaza daca este sau nu posibila transmiterea unui nou
; caracter catre LCD, pe baza informatiilor furnizate de rutina
; RDCMD. Se testeaza bitul 7 (BF - Busy Flag) al registrului de
; stare al LCD. Daca BF=1, inseamna ca LCD efectueaza o operatie
; interna si, deci, nu poate accepta un nou caracter.
;
WLCD:   ACALL  RDCMD          ; apel rutina RDCMD
        JB     ACC.7,WLCD      ; se verifica daca BF=1 si asteapta
                                ; trecerea lui in 0 (terminarea ope-
                                ; ratiunilor interne)
        RET                    ; revenire din subrutina
;
; Subrutina RDCMD selecteaza afisajul si citeste din registru de
; stare starea curenta a driverelor acestuia, informatie care este
; depusa in acumulator.
;
RDCMD:
        MOV    P2,#1          ; selectie port (S0=1)

```

```
        MOV     R0,#2           ; se realizeaza pe magistrala de adrese
                                ; selectia cu A0 si A1 (0000 0010 B),
                                ; pentru a se citi starea driverelor de
                                ; afisaj
        MOVX    A,@R0           ; se transfera in acumulator valoarea
                                ; gasita la adresa specificata anterior
        RET                     ; revenire din subrutina
;
; Rutina WRCMD scrie o comanda la driverele de afisaj, in urma
; verificarii starii acestora. Se selecteaza afisajul LCD si comanda
; continuta de registrul R2 este transferata driverelor din LCD.
;
WRCMD:
        ACALL   WLCD           ; se verifica starea driverelor (BF=0)
        MOV     A,R2           ; comanda continuta in registrul R2 e
                                ; transferata in acumulator
        MOV     P2,#1          ; se selecteaza portul LCD (S0=1)
        MOV     R0,#0          ; se selecteaza pe magistrala de adrese
                                ; cu A0 si A1 combinatia 0000 0000 B,
                                ; corespunzatoare scrierii unei comenzi
                                ; la driverele afisajului LCD
        MOVX    @R0,A          ; se scrie in driverele afisajului LCD
                                ; (la adresa stabilita anterior) comanda
        RET                     ; revenire din subrutina
;
; Subrutina WRDAT este identica cu WRCMD dar este utilizata pentru
; scrierea la driverele LCD codul caracterelor ce urmeaza a fi
; afisate.
;
WRDAT:
        ACALL   WLCD           ; testarea starii driverelor LCD
        MOV     P2,#1          ; se selecteaza portul LCD (S0=1)
        MOV     R0,#1          ; se selecteaza pe magistrala de adrese
                                ; cu A0 si A1 combinatia 0000 0001 B,
                                ; corespunzatoare scrierii unui caracter
                                ; la driverele afisajului LCD
        MOV     A,R2           ; se transfera caracterul in acumulator
        MOVX    @R0,A          ; se transmite caracterul driverelor LCD
        RET                     ; revenire din subrutina
;
; Subrutina DELAY realizeaza o intarziere cu 80 microsecunde.
;
DELAY:
        MOV     R3,#17         ; registrul R3 este incarcat cu valoarea
                                ; imediata 17
                                ; 80E-6secunde = 17 * 2instructiuni NOP *
                                ; * 12 perioade de ceas/instructiune NOP*
                                ; 1/11.059MHz (perioada ceas procesor)
LL1:    NOP                    ; no operation
        NOP                    ; no operation
        DJNZ    R3,LL1         ; decrementare registrul R3, se compa-
                                ; ra continutul sau cu 0. Daca rezultatul
                                ; compararii este 1, salt la eticheta LL1
        RET                     ; revenire din subrutina
END                                ; sfarsitul programului
```

9. IMPLEMENTAREA HARDWARE-SOFTWARE A UNUI INSTRUMENT DE VIZUALIZARE A SEMNALELOR (EASY SCOPE)

În cadrul acestui capitol se va prezenta structura hardware și software a unui osciloscop digital cu două canale de intrare, a cărei arhitectură este organizată în jurul unui microcontroller PIC16C71 funcționând cu o frecvență a ceasului de 20MHz. În esență, este utilizată secțiunea de conversie analog-digitală implementată în cadrul familiei de circuite PIC16Cxx, completată cu o schemă originală, simplă dar eficientă de prelucrare analogică a semnalelor, pentru a se realiza compatibilizarea cu circuitele de intrare uzuale ale osciloscopelor analogice și digitale (impedanță de intrare cu valoarea de $1M\Omega$, trepte de atenuare de 1:1, 1:2, 1:5, 1:10).

9.1 DESCRIEREA FUNCȚIONALĂ A OSCILOSCOPULUI DIGITAL EASY SCOPE

Orice pasionat al electronicii, tehnician sau inginer, și-ar dori să aibă propriul osciloscop. Însă în cele mai multe situații este dificilă justificarea costului ridicat al unui asemenea echipament de măsurare. Este mult mai probabil ca fiecare dintre acești specialiști să posede un calculator compatibil PC. Osciloscopul digital Easy Scope transformă calculatorul într-un osciloscop digital cu memorie, cu două canale. Ca și un osciloscop analogic, un osciloscop digital cu memorie permite vizualizarea semnalelor electrice pe ecran. Însă, spre deosebire de un osciloscop analogic, acesta convertește semnalele de intrare în formă digitală prin eșantionarea intrărilor utilizând un convertor analog-digital. Acest procedeu conferă osciloscopului digital cu memorie avantaje comparative cu unul analogic, ca de exemplu:

- capacitatea de a vizualiza un singur eșantion al semnalului analogic de intrare;
- capacitatea de a afișa forma de semnal atât dinaintea cât și după un semnal de trigger;
- capacitatea de a salva datele pentru vizualizarea lor ulterioară sau pentru prelucrarea acestora;

Bineînțeles că cei care preferă utilizarea osciloscopelor analogice vor sublinia faptul că variantele digitale nu permit vizualizarea variațiilor semnalelor de intrare între două operații succesive de eșantionare ale acestora. Acest aspect poate constitui un dezavantaj în anumite situații deosebite.

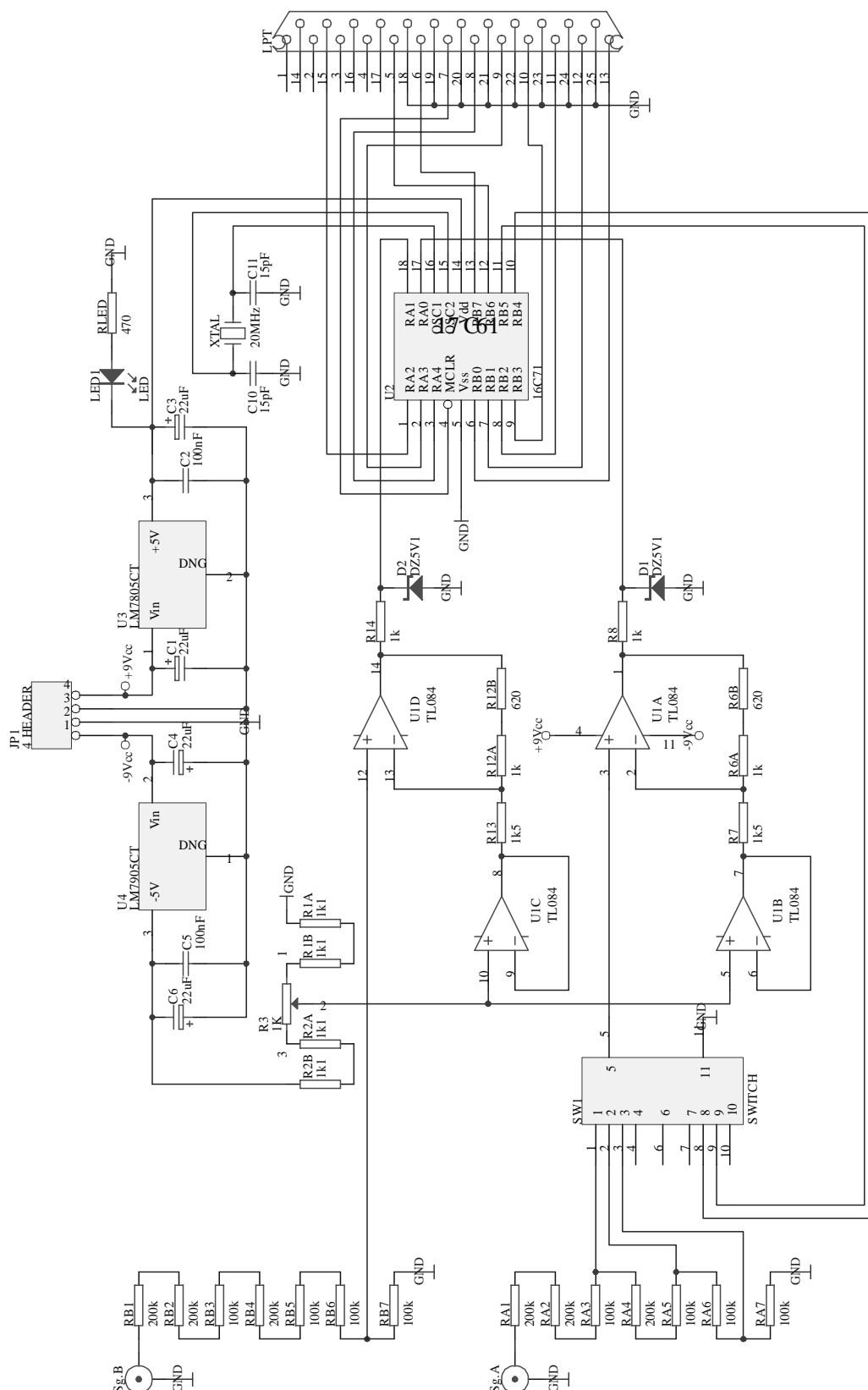


Fig. 9.1 Schema electrică a osciloscopului Easy Scope.

Osciloscopul Easy Scope este un osciloscop digital cu memorie (**D**igital)

Storage Oscilloscope - **DSO**), interfațabil cu calculatoare personale compatibile IBM PC, de preț redus și caracterizat de următoarele caracteristici principale:

- cuplarea (interfațarea) pe portul paralel al calculatorului are avantajul de a nu solicita un conector (slot) de magistrală;
- este portabil, de dimensiuni reduse, funcționează alimentat de la baterii și are un consum redus. În combinație cu un laptop, acest instrument devine un adevărat echipament de test;
- rata de eșantionare maximă este de aproximativ 50KHz, depinzând de frecvența de lucru a calculatorului;
- dispune de caracteristici **DVM** (**D**igital **V**olt**M**eter) pentru efectuarea de măsurători statice de tensiuni;
- dispune de 3 game de sensibilitate pentru intrări: $\pm 2,4\text{V}$, $\pm 6,0\text{V}$ și $\pm 12\text{V}$, utilizând o sondă de osciloscop $1\times$;
- poate fi utilizată o sondă de osciloscop $10\times$ pentru a se obține trei game suplimentare de intrare ($\pm 24\text{V}$, $\pm 60\text{V}$ și $\pm 120\text{V}$);
- software pentru PC ușor de utilizat;
- facilități de adaptare a circuitului pentru aplicații atipice.

Cum se poate să se înglobeze atât de multe facilități într-o construcție atât de compactă?

Circuitul conține două secțiuni analogice și o secțiune digitală. Întreaga schemă se bazează pe un microcontroller PIC16C71. Acesta prezintă avantaje substanțiale în comparație cu controller-ele din seria PIC16C5x, utilizate frecvent în multe aplicații. În plus față de circuitele PIC uzuale, 16C71 înglobează un convertor analog-digital (A/D).

Convertorul A/D implementat în cadrul microcontroller-ului PIC16C71 funcționează în gama $0 \dots +5\text{V}$. Pentru a utiliza convertorul analog-digital pentru mai multe game de tensiuni de intrare, inclusiv pentru tensiuni de intrare negative, este necesară utilizarea unei secțiuni de condiționare a semnalelor înainte de aplicarea acestora la intrările convertorului analog-digital. Osciloscopul digital Easy Scope dispune de două canale ce necesită condiționarea semnalelor de intrare înainte de aplicarea la intrările circuitului de conversie. Circuitul de condiționare a semnalului pentru canalul A este implementat cu un circuit TL084 (amplificator operațional cuadruplu), un comutator rotativ cu două secțiuni de câte 4 contacte și o serie de rezistențe. În primul rând, semnalul este aplicat unui divizor rezistiv compus dintr-o grupare serie de rezistențe, cu rezistența totală de $1\text{M}\Omega$. Semnalul este preluat dintr-unul dintre cele trei puncte ale divizorului rezistiv (corespunzând atenuărilor de 1:2, 1:5 sau 1:10 față de semnalul de intrare) prin intermediul comutatorului SW1. Semnalul atenuat este aplicat pe intrarea neînversoare a unui amplificator sumator-subtractor realizat cu U1. Amplificarea acestuia este egală cu $(R6/R7)+1$. O tensiune de offset de $2,5\text{V}$ este suprapusă peste semnalul util. Această tensiune de offset este obținută prin intermediul unui divizor rezistiv

ajustabil (R1, R2 și semireglabilul multitură R3), căruia i se aplică tensiunea stabilizată de $-5V$ furnizată de un stabilizator integrat de tensiune de tip LM7905. Tensiunea de referință obținută pe cursorul semireglabilului trebuie să fie egală cu $-2,3V$ și poate fi ajustată fin prin reglarea lui R3. Această tensiune de referință este buffer-ată, utilizând un al doilea amplificator operațional în configurație de repetor de tensiune, și este aplicată intrării inversoare a amplificatorului sumator-subtractor. Aplicând o tensiune de $0V$ la intrarea canalului A, ieșirea amplificatorului sumator-subtractor trebuie să fie de $2,5V$. Dacă la intrarea canalului A se aplică o tensiune negativă, ieșirea circuitului este mai mică decât $2,5V$ iar aplicarea unui semnal de polaritate pozitivă la intrarea canalului A determină ca tensiunea de la ieșirea circuitului să fie superioară valorii de $2,5V$. Ieșirea amplificatorului sumator-subtractor este aplicată intrării analogice RA0 a microcontroller-ului PIC16C71 prin intermediul unei rezistențe, R8. Dioda Zener D1 asigură protecția la supratensiune a intrării convertorului A/D, limitând excursia de semnal la intervalul $-0,6V...+5,1V$.

Cea de-a doua secțiune a comutatorului SW1 acționează asupra biților portului B (RB4 și RB5) astfel:

RB4	RB5	Atenuare
1	1	1:2 (gama $\pm 2,4V$)
0	1	1:5 (gama $\pm 6,0V$)
1	0	1:10 (gama $\pm 12V$)

Așa cum se poate observa din analiza schemei electrice a osciloscopului Easy Scope, circuitul de condiționare a semnalului pentru canalul B este identic cu cel implementat pentru canalul A, cu excepția faptului că pentru acest canal s-a ales gama fixă de tensiuni de intrare $\pm 12V$. În variantele ulterioare se pot prevedea facilități de selecția a gamelor de tensiuni de intrare la fel ca pentru canalul A, utilizând un comutator SW1 cu 3 secțiuni.

Convertorul A/D din cadrul microcontroller-ului PIC16C71 este setat să efectueze conversii ale semnalelor analogice aplicate intrărilor RA0 și RA1, furnizând rezultatele sub formă binară (0...255). Intreruperea de la timer-ul intern din structura circuitului PIC16C71 este utilizată pentru generarea unei baze de timp stabile în vederea eșantionării. La apariția întreruperii generate de timer se citește conținutul registrului ADRES, valoarea citită este memorată și se declanșează o nouă conversie A/D. Dacă rata de eșantionare este setată la 10KHz, atunci trebuie efectuată o conversie A/D la fiecare 100 μs . Intreruperea de timer trebuie setată în această situație la fiecare 100 μs . Între două întreruperi succesive datele sunt transferate la PC prin intermediul portului paralel. Transferul se efectuează paralel pe 4 biți și utilizează protocolul handshake REQ*/ACK*.

Interfața cu portul paralel al calculatorului (PC) include în total 10 semnale; 5 dintre ele sunt furnizate de către PC (intrări în microcontroller-ul

16C71) iar 5 sunt furnizate de PIC16C71 (intrările în PC). Semnalele de interfață sunt prezentate, de asemenea, în cadrul schemei electrice a osciloscopului digital Easy Scope. Unul dintre semnalele furnizate de către PC controlează intrarea de reset a controller-ului PIC (MCLR*); aceasta permite calculatorului să inițializeze funcționarea microcontroller-ului la orice moment de timp. Trei dintre semnalele de la PC sunt dedicate pentru controlul modului (MODE control). La aplicarea unui semnal de reset, controller-ul PIC citește semnalele MODE pentru a determina modul de funcționare. Modurile de lucru include: funcționarea ca osciloscop, funcționarea ca voltmetru pe canalul 0, funcționarea ca voltmetru pe canalul 1 și stare/încărcare constante de timp și bascularea ieșirilor (modul debug). Patru dintre pinii controller-ului PIC sunt configurați ca pini de date. Celelalte două semnale rămase reprezintă semnalele de protocol de tip handshake (request și acknowledge) utilizate pentru transferurile de date. Transferul celor două grupe de câte 4 biți demarează prin activarea semnalului REQ* de către PC. Simbolul * indică faptul că semnalul este activ pe “0” logic. Aceasta înseamnă că semnalul REQ* devine “0” logic. Atunci când controller-ul PIC transferă datele pe pinii corespunzători, el activează semnalul ACK*. Activarea semnalului ACK* indică calculatorului că datele sunt disponibile. După ce PC-ul citește datele, dezactivează semnalul REQ* (nivel logic “1”), pentru a indica faptul că datele au fost citite. În urma identificării acestei situații, controller-ul PIC dezactivează semnalul ACK*. Următorul cadru de 4 biți este transferat în mod similar, cu excepția faptului că semnalul MODE A este setat la “0” pe durata transferului.

9.2 PREZENTAREA CIRCUITULUI PIC16C71 (MICROCHIP)

Caracteristicile principale ale familiei de microcontroller-e PIC16C71X sunt:

- arhitectură CPU de tip RISC de înaltă performanță;
- set simplu și eficient de instrucțiuni (35 instrucțiuni pe un cuvânt);
- toate instrucțiunile durează un singur ciclu, cu excepția celor de salt în program care durează două cicluri;
- frecvența maximă a cesului de sistem 20 MHz, durata unui ciclu instrucțiune minim 200 ns;
- memorie de program - Program memory - de maxim 2k x 14 cuvinte, maxim 128 x 8 octeți de memorie de date - Data Memory (RAM);
- facilități de întreruperi;
- stivă hardware organizată pe maximum 8 nivele;
- capabilități de adresare directă, indirectă și relativă;
- Power-on Reset (POR);
- Power-up Timer (PWRT) și Oscillator Start-up Timer (OST);

- Watchdog Timer (WDT) cu oscilator RC propriu implementat pe chip pentru funcționare sigură;
- protecție software a codului;
- implementarea modului de economie de energie (power saving) - SLEEP mode;
- opțiuni de selectare a oscilatorului;
- memorie EPROM în tehnologie HighSpeed-CMOS cu putere consumată redusă;
- implementare complet statică;
- gamă largă a tensiunilor de alimentare: de la 2,5V la 6,0V;
- capacitate mare a curenților debitați de pini: 25 mA
- gamă de temperaturi de funcționare: comercială, industrială și extinsă;
- circuit de verificare a parității memoriei de program cu Parity Error Reset (PER) (doar la PIC16C715);
- putere totală consumată redusă:
 - < 2 mA @ 5V, 4 MHz;
 - 15 mA tipic @ 3V, 32 kHz;
 - < 1 mA tipic curent de alimentare în stand-by;
- organizare sub forma a 13 pini de intrare/ieșire controlați individual.

9.2.1 DESCRIERE GENERALĂ A PIC16C71

Familia de microcontroller-e PIC16C71X este caracterizată de costuri scăzute, înaltă performanță, implementare în tehnologie CMOS statică, reprezentând o familie de microcontroller-e de 8 biți ce integrează în cadrul structurii un convertor analog-digital (A/D) și situată în cadrul familiei PIC16CXX. Toate microcontroller-ele PIC16/17 utilizează o arhitectură avansată de tip RISC. Familia de microcontroller-e PIC16CXX prezintă caracteristici performante ale nucleului de bază, stivă cu adâncime de 8 nivele și mai multe surse de întreruperi interne și externe.

Separarea secțiunilor de instrucțiuni și de date ale arhitecturii Harvard permite lungimea de 14 biți a formatului instrucțiunilor și separat cuvinte de date cu lungimea de 8 biți. Execuția de tip pipeline (pe două nivele) a instrucțiunilor asigură executarea acestora într-un singur ciclu, cu excepția acelor de ramificare a programului, acestea din urmă necesitând pentru execuție două cicluri. Utilizatorului îi este disponibil un set redus, dar eficient, conținând 35 de instrucțiuni. În plus, setul complex de registre interne permite obținerea unor performanțe deosebite prin utilizarea inovațiilor arhitecturale.

Microcontroller-ele PIC16CXX asigură în medie o compresie a codului de 2 ori și o creștere a vitezei de 4 ori în comparație cu alte microcontroller-e de

8 biți din aceeași clasă.

Dispozitivele **PIC16C710/71** dispun de 36 octeți de memorie RAM (Data Memory), **PIC16C711** dispune de 68 octeți de memorie RAM iar **PIC16C715** are 128 octeți de memorie RAM. Fiecare reprezentant al familiei dispune de 13 pini de intrare/ieșire (I/O). În plus este disponibil un circuit timer/counter. De asemenea, în cadrul structurii este implementat un convertor analog-digital rapid, cu rezoluție de 8 biți și dispunând de 4 canale analogice de intrare multiplexate. Rezoluția de 8 biți este ideală pentru aplicații necesitând interfațare analogică de cost redus, cum ar fi comanda termostatelor, măsurarea presiunii, etc.

Familia de microcontroller-e PIC16C71X prezintă caracteristici speciale menite să reducă la maxim numărul de componente externe, reducând astfel costurile, determinând creșterea fiabilității și reducerea puterii consumate.

Sunt disponibile patru opțiuni de oscilator, dintre care oscilatorul RC asigură a soluție foarte economică din punct de vedere cost, oscilatorul LP minimizează puterea consumată, oscilatorul XT reprezintă un cristal standard de cuarț, iar oscilatorul HS este destinat pentru cristale High Speed. Caracteristica SLEEP (power-down) asigură modul de economisire a puterii consumate. Utilizatorul poate scoate circuitul din modul SLEEP în mai multe moduri, utilizând întreruperi externe și interne, precum și semnale de reset.

Un circuit fiabil și cu funcționare sigură de tip Watchdog Timer pilotat de propriul său oscilator RC implementat pe chip asigură protecția eficientă împotriva blocărilor software.

Capsulele Cerdip cu fereastră de cuarț pentru ștergere cu ultraviolete (UV) a memoriei de program (UVEPROM) sunt recomandate pentru dezvoltarea de aplicații, pe când variantele OTP (One-Time-Programmable), sensibil mai ieftine sunt eficiente pentru producția la orice nivel.

Familia PIC16C71X este eficientă din punct de vedere hardware-software pentru aplicații diverse, de la aplicații gen sisteme de securitate și senzori inteligenți comandați de la distanță, până la sisteme de control în locuințe și în industria auto. Utilizarea tehnologiei EPROM pentru memoria de program permite adaptarea facilă și rapidă a programelor de aplicații (coduri de transmisie, controlul turației motoarelor, receptoare de frecvență, etc.). Varietatea amprentelor componentelor permite utilizarea acestor microcontroller-e în cadrul aplicațiilor ce impun limitări din punct de vedere a spațiului ocupat. Prețul redus, puterea consumată redusă, performanțele ridicate, ușurința utilizării și flexibilitatea mare a liniilor de intrare/ieșire (I/O) determină ca familia PIC16C71X să fie deosebit de atractivă și versatilă chiar în domenii în care nu s-a avut în vedere până acum utilizarea microcontroller-elor (cum ar fi implementarea unor funcții de timer, comunicația serială, captarea și compararea, funcții PWM (Power Width Modulation) și aplicații de tip coprocesor matematic).

9.2.2 PREZENTARE ARCHITECTURALĂ

Performanțele deosebite din punct de vedere a eficienței calculelor și a versatilității ale familiei de microcontroller-e PIC16CXX se datorează în principal unui număr de caracteristici arhitecturale tipice pentru microprocesoarele de tip RISC. În primul rând, familia de circuite PIC16CXX utilizează o arhitectură Harvard, în cadrul căreia instrucțiunile și datele sunt accesate din memorii dedicate distincte utilizând magistrale distincte. Aceasta îmbunătățește lărgimea de bandă comparativ cu arhitectura tradițională von Neumann, în cadrul căreia instrucțiunile și datele sunt manipulate pe aceeași magistrală din cadrul unei memorii comune. Separarea magistrelor de instrucțiuni (program) și de date permit dimensiuni (lungimi) diferite ale instrucțiunilor și datelor. Codul operației are o lungime de 14 biți permițând ca toate instrucțiunile să fie codificate într-un singur cuvânt. Ciclul de fetch (citire a instrucțiunii) aduce codul instrucțiunii (cu lungimea de 14 biți) pe magistrala dedicată din memoria de program într-un singur ciclu. O structură pipeline pe două nivele suprapune fazele de citire și execuție a instrucțiunilor. Ca urmare, toate instrucțiunile (35) sunt executate într-un singur ciclu (200 ns @ 20 MHz), cu excepția instrucțiunilor de ramificare în program.

În tabelul următor se prezintă memoria de program (EPROM) și memoria de date (RAM) pentru fiecare reprezentant al familiei de microcontroller-e PIC16C71X.

Reprezentant	Capacitate memorie de program	Capacitate memorie de date
PIC16C710	512 x 14	36x8
PIC16C71	1K x 14	36x8
PIC16C711	1K x 14	68x8
PIC16C715	2K x 14	128x8

Circuitele din familia PIC16CXX pot adresa direct sau indirect registrele interne sau memoria de date. Toate registrele de funcții speciale, inclusiv contorul programului (Program Counter) sunt mapate în memoria de date. Familia PIC16CXX dispune de un set de instrucțiuni ortogonal (simetric), ceea ce face posibilă execuția oricărei instrucțiuni cu oricare registru, utilizând oricare dintre modurile de adresare. Această caracteristică de simetrie și lipsa “situațiilor speciale de optimizare” fac ca programarea pentru PIC16CXX să fie simplă, dar cu toate acestea eficientă. În plus, curba de învățare se reduce semnificativ.

Circuitele din familia PIC16CXX conțin o unitate aritmetico-logică (ALU) și un registru de lucru de 8 biți. ALU este o unitate aritmetică de uz general. Execută funcții aritmetice și logice între datele conținute în registrul de

lucru și oricare alt registru. ALU lucrează cu operanzi pe 8 biți și permite adunarea, scăderea, shift-area și operații logice. Cu excepția unor mențiuni speciale, operațiile aritmetice se execută în complement față de 2. În cadrul instrucțiunilor cu 2 operanzi, tipic unul dintre operanzi este registrul de lucru (registrul W). Celălalt operand este un alt registru de date sau o valoare imediată. În cadrul instrucțiunilor cu un singur operand, acesta este constituit fie de registrul de lucru (W) sau de un registru de date.

Registrul W este un registru de lucru de 8 biți utilizat pentru operațiile în cadrul ALU. Nu este un registru adresabil.

În funcție de instrucțiunea executată, unitatea aritmetico-logică poate afecta valorile indicatorilor Carry (C), Digit Carry (DC) și Zero (Z) din cadrul registrului STATUS. Indicatorii C și DC au semnificația unor biți de **borrow** și **digit borrow**, respectiv în cadrul operațiilor de scădere.

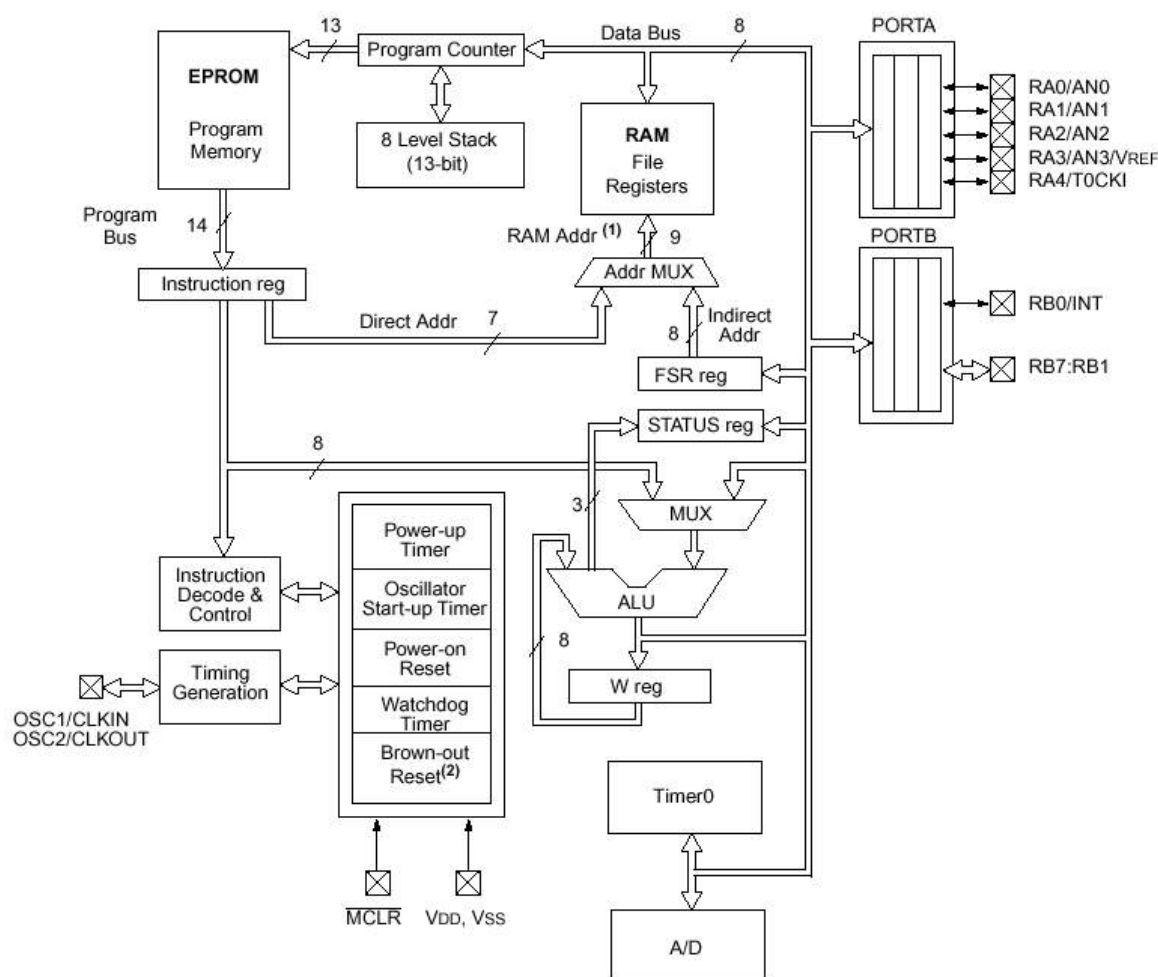


Fig. 9.2 Arhitectura microcontroller-ului PIC16C71.

9.2.2.1 CEASUL DE SISTEM / CICLUL INSTRUCȚIUNE

Intrarea de ceas (OSC1) este intern divizată cu patru pentru a genera patru semnale de ceas în cuadratură și fără suprapunere, Q1, Q2, Q3 și Q4. Intern, contorul programului (PC) este incrementat la fiecare perioadă Q1, codul instrucțiunii este citit din memoria de program și este memorat în registrul de instrucțiuni în Q4. Instrucțiunea este decodificată și executată în următoarele perioade Q1 până la Q4. Diagramele temporale ale semnalelor de ceas și fluxul de execuție al instrucțiunii sunt ilustrate în fig. 9.3.

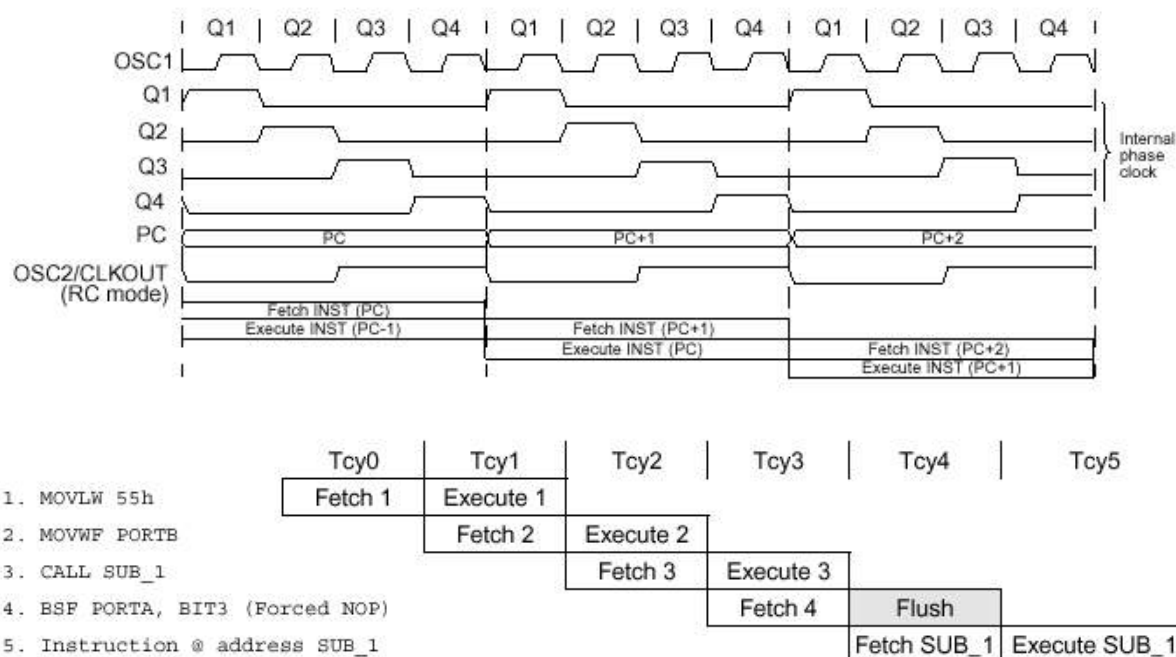


Fig. 9.3 Ceasul de sistem, ciclul instrucțiune

9.2.2.2 FLUXUL DE EXECUȚIE AL INSTRUCȚIUNII / PIPELINE-ING

Un ciclu instrucțiune conține patru cicli de ceas Q (Q1, Q2, Q3 și Q4). Citirea din cadrul memoriei de program și execuția codului instrucțiunii folosesc tehnica pipeline, astfel încât ciclul de fetch durează un ciclu instrucțiune iar decodificarea și executarea durează un alt ciclu instrucțiune. Cu toate acestea, datorită tehnicii pipeline, fiecare instrucțiune este executată efectiv în cadrul unui singur ciclu. Dacă instrucțiunea curentă determină modificarea conținutului contorului programului (de exemplu GOTO) atunci sunt necesare două cicluri pentru executarea sa.

Un ciclu de fetch începe prin incrementarea contorului de program (PC) pe durata Q1.

În cadrul ciclului de execuție, instrucțiunea citită din memoria de program este memorată în cadrul registrului de instrucțiuni “Instruction Register” (IR) pe durata Q1. Această instrucțiune este apoi decodificată și executată în ciclurile de ceas Q2, Q3 și Q4. Memoria de date (Data memory) este citită pe durata Q2 (citirea operandului) și scrisă în ciclul Q4 (scrierea rezultatului).

9.2.3 ORGANIZAREA MEMORIEI DE PROGRAM (PROGRAM MEMORY)

Familia de microcontroller-e PIC16C71X dispune de un contor de program (PC) cu lungimea de 13 biți, capabil să adreseze un spațiu de memorie de program de 8k x 14 cuvinte.

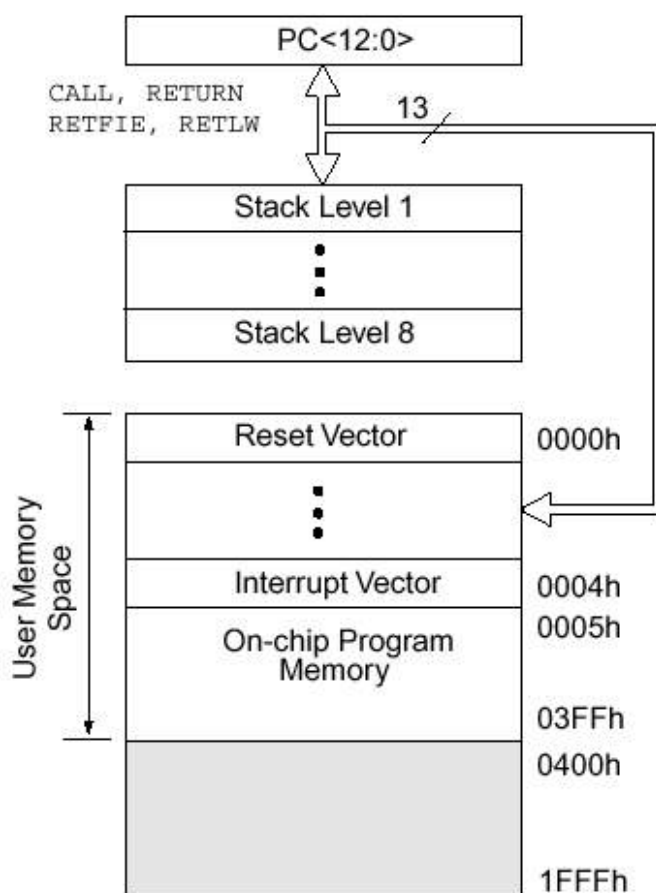


Fig. 9.4 Organizarea memoriei de program la familia de microcontroller-e PIC16C71X.

Capacitatea memoriei de program disponibilă pentru fiecare reprezentant al familiei este prezentată în tabelul următor:

Reprezentant	Capacitatea memoriei de program	Spațiul de adrese
PIC16C710	512 x 14	0000h-01FFh
PIC16C71	1k x 14	0000h-03FFh
PIC16C711	1k x 14	0000h-03FFh
PIC16C715	2K x 14	0000h-07FFh

Pentru acei reprezentanți dispunând de o capacitate a memoriei de program mai mică de 8K, accesarea unei locații de memorie în afara spațiului fizic implementat va determina o blocare a funcționării.

Vectorul de reset se află la adresa 0000h, iar vectorul de întrerupere la adresa 0004h.

9.2.4 ORGANIZAREA MEMORIEI DE DATE (DATA MEMORY)

Memoria de date este partiționată în două bancuri care conțin registrele de uz general (General Purpose Registers) și registrele de funcții speciale (Special Function Registers). Bitul RP0 determină selectarea bancului de registre (bank select bit).

- RP0 (STATUS<5>) = 1 → Bank 1
- RP0 (STATUS<5>) = 0 → Bank 0

Fiecare banc se extinde până la 7Fh (dispune de 128 octeți). Locațiile inferioare din cadrul fiecărui banc sunt rezervate pentru registrele de funcții speciale. La adresele imediat superioare se găsesc registrele de uz general, implementate ca memorie RAM statică. Atât Bank 0 cât și Bank 1 conțin registre de funcții speciale. Unele registre de funcții speciale frecvent utilizate din Bank 0 sunt oglindite în Bank 1 din motive de minimizare a codului și facilitare a accesului.

9.2.4.1 REGISTRUL STATUS

Registrul de stare (STATUS register) conține starea unității aritmetico-logice (ALU), starea de RESET și biții de selectare a bancului de registre pentru memoria de date. Registrul STATUS poate constitui destinație a oricărei instrucțiuni, ca și oricare alt registru. Dacă registrul STATUS reprezintă destinația unei instrucțiuni care afectează indicatorii de condiție (Z, DC sau C), atunci scrierea acestor biți este dezactivată. Acești biți sunt setați sau resetati corespunzător logicii dispozitivului. În plus, biții TO și PD nu pot fi scriși. De aceea rezultatul unei instrucțiuni în care se utilizează registrul STATUS ca

destinație poate fi diferit față de rezultatul preconizat.

De exemplu, în urma execuției instrucțiunii CLRF STATUS se vor reseta cei mai semnificativi 3 biți și va fi setat indicatorul Z. Conținutul registrului STATUS în urma acestei operații va fi **000N|N1NN** (N = nemodificat). Ca urmare, se recomandă ca doar instrucțiunile BCF, BSF, SWAPF și MOVWF să fie utilizate în corelație cu modificarea registrului STATUS, deoarece aceste instrucțiuni nu afectează indicatorii de condiții (Z, C și DC) din cadrul registrului STATUS.

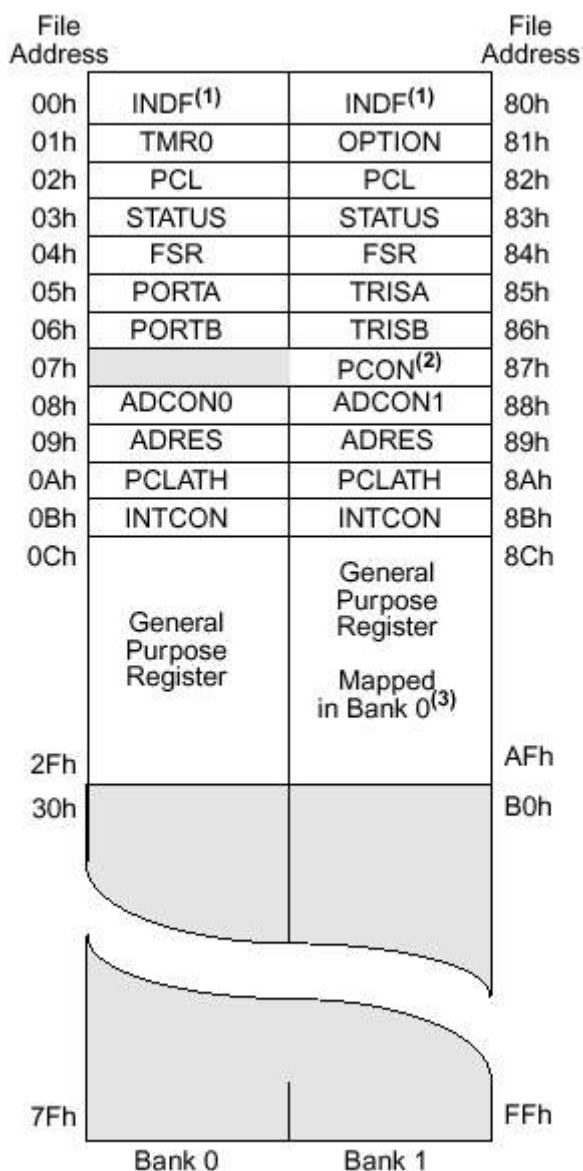


Fig. 9.5 Organizarea bancurilor de registre.

Nota 1: Pentru dispozitivele care nu utilizează biții IRP și RP1 (STATUS<7:6>), se recomandă menținerea acestor biți reșetați pentru a se asigura compatibilitatea cu produse din generațiile următoare.

Nota 2: Biții C și DC funcționează ca semnale **borrow** și **digit borrow** în cadrul operațiilor de scădere (a se vedea instrucțiunile SUBLW și SUBWF).

9.2.5 PORTURI I/O

Unii dintre pinii acestor porturi de intrare/ieșire (I/O) sunt multiplexați cu funcții alternative pentru implementarea caracteristicilor periferice speciale ale dispozitivelor. În general, atunci când este activată o astfel de funcție, pinul corespunzător nu poate fi utilizat ca un pin general de intrare/ieșire.

9.2.5.1 REGISTRELE PORTA ȘI TRISA

PORTA este implementat ca un latch de 5 biți. Pinul RA4/T0CKI funcționează ca o intrare de tip Trigger Schmitt input și ca ieșire de tip drenă în gol. Toți ceilalți pini ai portului RA sunt caracterizați de nivele logice de intrare de tip TTL și de driver-e de ieșire de tip CMOS. Toți pinii sunt controlați de biți de direcție (registrul TRIS) care pot configura acești pini fie ca intrări, fie ca ieșiri. Setarea unui bit din cadrul registrului TRISA determină comandarea driver-ului de ieșire al pinului corespunzător în starea de mare impedanță (hi-Z mode). Resetarea unui bit din cadrul registrului TRISA determină transferarea conținutului latch-ului de ieșire la pinul corespunzător.

Citirea registrului PORTA determină citirea stării pinilor asociați, iar scrierea registrului PORTA înseamnă scrierea la registrul latch al portului. Toate operațiile de scriere sunt de tip citire-modificare-scriere. Deci, o scriere la un port implică citirea stării pinilor portului, modificarea valorii citite și apoi scrierea datelor în registrul latch al portului.

Pinul RA4 este multiplexat cu semnalul de intrare de ceas al modulului Timer0, devenind astfel pinul RA4/T0CKI.

Ceilalți pini ai portului PORTA sunt multiplexați cu intrările analogice și intrarea de tensiune de referință V_{REF} . Funcționarea corespunzătoare a fiecărui pin este comandată prin resetarea/setarea biților de control din cadrul registrului ADCON1 (A/D Control Register1).

Registrul TRISA controlează (comandă) sensul transferului pe pinii RA, chiar în situația în care sunt utilizați ca intrări analogice. Utilizatorul trebuie să se asigure că biții registrului TRISA sunt menținuți setați atunci când utilizează pinii portului PORTA ca intrări analogice.

Notă: La Power-on Reset, acești pini sunt configurați ca intrări analogice și citiți ca '0'.

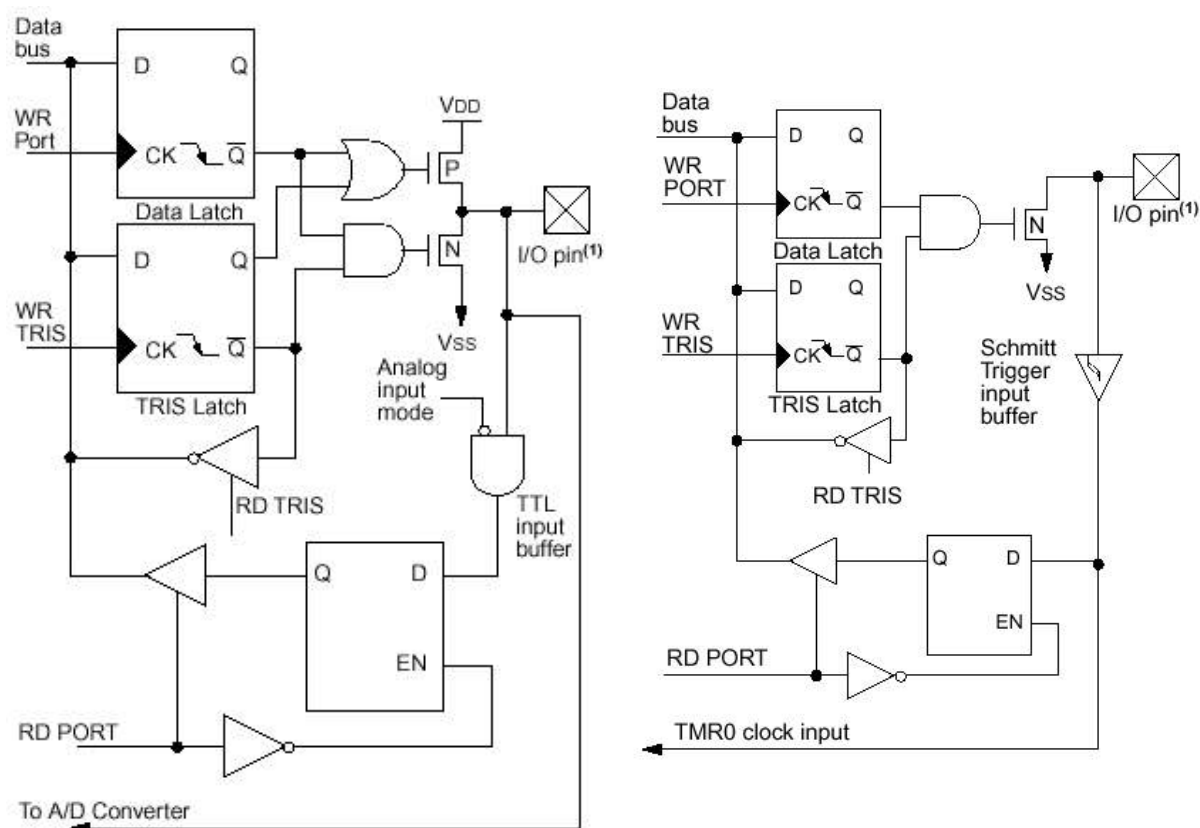


Fig. 9.6 Configurația internă a biților portului PORTA .

9.2.5.2 REGISTRELE PORTB ȘI TRISB

PORTB este un port bidirecțional dispunând de 8 biți. Registrul corespunzător sensului transferului individual la nivel de pin este TRISB. Setarea unui bit din cadrul registrului TRISB determină comandarea driver-ului de ieșire al pinului corespunzător în starea de mare impedanță (hi-Z mode). Resetarea unui bit din cadrul registrului TRISA determină transferarea conținutului latch-ului de ieșire la pinul corespunzător.

Fiecare dintre pinii portului PORTB dispune de un pull-up intern de valoare mare. Un singur bit de control poate activa toate aceste dispozitive de pull-up. Aceasta poate fi realizată prin resetarea bitului RBPU (OPTION<7>). Dispozitivele de pull-up sunt automat dezactivate atunci când pinii portului sunt configurați ca ieșiri. Dispozitivele de pull-up sunt dezactivate la apariția Power-on Reset.

Patru dintre pinii portului PORTB, RB7:RB4, dispun de facilități de întrerupere la modificarea caracteristicilor acestora. Doar pinii configurați ca intrări pot produce apariția unei întreruperi (adică oricare dintre pinii RB7:RB4 configurați ca ieșire sunt excluși din facilitatea de generare a unei întreruperi în procesul de comparare a modificării stării). Pinii configurați ca intrări (dintre

RB7:RB4) sunt comparați cu valorile anterioare memorate la ultima citire în cadrul registrului portului PORTB. Ieșirile necoincidențelor pinilor RB7:RB4 sunt aplicate unei funcții SAU pentru a genera întreruperea corespunzătoare modificării portului RB (RB Port Change Interrupt), caracterizată de flag-ul RBIF (INTCON<0>). Această întrerupere poate determina ca dispozitivul să părăsească modul SLEEP. Utilizatorul, în cadrul rutinei de tratare a întreruperii, poate șterge întreruperea în unul dintre următoarele moduri:

- prin orice citire sau scriere la portul PORTB, ceea ce va încheia condiția de necoincidență;
- resetarea indicatorului RBIF. O condiție de necoincidență va continua să seteze indicatorul RBIF. Citirea portului PORTB va încheia condiția de necoincidență și va permite resetarea flag-ului RBIF. Generarea unei întreruperi în condiția de necoincidență, împreună cu circuitele de pull-up configurabile software pentru acești patru pini, permit o interfațare simplă cu o tastatură și fac posibilă activarea sistemului la apăsarea unei taste.

Se recomandă ca întreruperea generată în caz de necoincidență (schimbarea valorii logice aplicată pe pinii de intrare ai portului PORTB) să fie utilizată pentru activarea sistemului la acționarea unei taste și în cazul în care PORTB este utilizat doar pentru generarea unei astfel de întreruperi. Efectuarea polling-ului pe PORTB nu este recomandată dacă se utilizează întreruperile determinate de modificarea stării intrărilor portului PORTB.

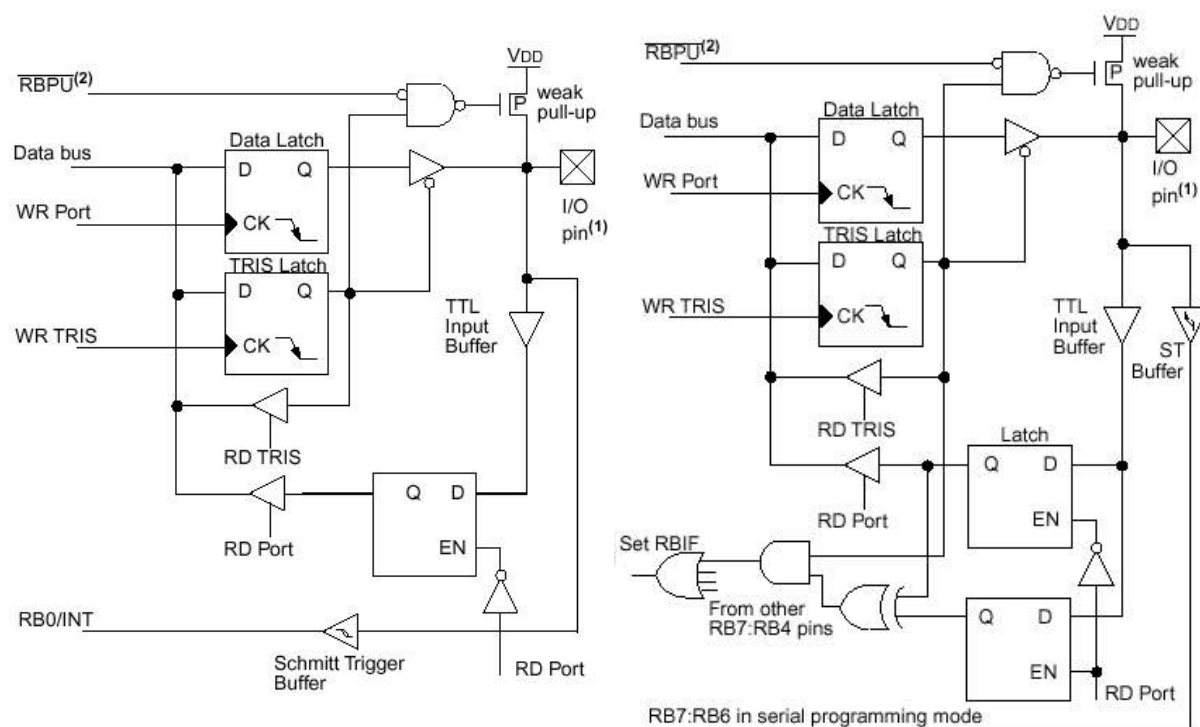


Fig. 9.7 Configurația internă a pinilor portului PORTB.

Exemplul 9.1: Inițializarea PORTB

```
BCF      STATUS, RP0      ;  
CLRF     PORTB            ; inițializează PORTB prin  
                        ; resetarea ieșirilor  
                        ; de date ale latch-ului  
BSF      STATUS, RP0      ; selectează Bank 1  
MOVLW 0xCF                ; valoare utilizată pentru  
                        ; inițializarea direcției  
MOVWF TRISB              ; setează RB<3:0> ca intrări  
                        ; RB<5:4> ca ieșiri  
                        ; RB<7:6> ca intrări
```

Notă: Pentru PIC16C71, dacă apare o modificare a stării pinilor I/O în cursul executării unui ciclu de citire (începutul perioadei Q2) există posibilitatea ca indicatorul RBIF să nu fie setat.

9.2.6 CONSIDERAȚII DE PROGRAMARE I/O

9.2.6.1 PORTURI I/O BIDIRECȚIONALE

Orice instrucțiune de scriere funcționează intern ca o operație de citire urmată de o operație de scriere. Instrucțiunile BCF și BSF, de exemplu, citesc conținutul registrului în CPU, execută operația la nivel de bit și scriu rezultatul obținut în registru. În situația în care aceste instrucțiuni sunt aplicate pentru un port în care unii dintre biți sunt configurați ca intrări iar alții ca ieșiri trebuie luate măsuri de precauție speciale. De exemplu, o instrucțiune BSF asupra bitului 5 al portului PORTB va determina citirea tuturor celor opt biți ai portului. Apoi are loc executarea instrucțiunii BSF asupra bitului 5 și PORTB este scris în latch-ul de ieșire. Dacă un alt bit al portului PORTB este utilizat ca pin bidirecțional (de exemplu bitul 0) și este definit ca intrare la acest moment de timp, semnalul de intrare prezent pe acest pin va fi citit de către CPU și rescris în latch-ul de date asociat acestui pin, suprascriind conținutul anterior. Atât timp cât pinul funcționează în mod intrare nu apare nici o problemă în funcționare. Cu toate acestea, dacă bitul 0 este comandat în modul ieșire, conținutul registrului de date poate deveni neprecizat în această situație.

Citirea registrului portului reprezintă citirea valorilor logice aplicate pinilor portului. Scrierea în registrul portului înseamnă scrierea valorii corespunzătoare în latch-ul de date al portului. Când se utilizează instrucțiuni de citire-modificare-scriere (de exemplu BCF, BSF, etc.) asupra unui port, se citesc valorile logice ale pinilor portului, se efectuează operația dorită cu aceste valori iar rezultatul este scris în cadrul latch-ului portului.

În exemplul 9.2 se ilustrează efectul a două instrucțiuni consecutive de citire-modificare-scriere asupra unui port de intrare/ieșire (I/O).

Exemplul 9.2: Instrucțiuni consecutive de citire-modificare-scriere asupra unui port I/O

```
; Setări inițiale: PORTB<7:4> intrări
; PORTB<3:0> ieșiri
; PORTB<7:6> pull-up activat, fără a fi conectate în exterior
;
;                PORT latch                PORT pins
;                -----                -----
BCF  PORTB, 7      ; 01pp pppp                11pp pppp
BCF  PORTB, 6      ; 10pp pppp                11pp pppp
BSF  STATUS, RP0   ;
BCF  TRISB, 7      ; 10pp pppp                11pp pppp
BCF  TRISB, 6      ; 10pp pppp                10pp pppp
; Utilizatorul se putea aștepta ca valorile pinilpr să fie
; 00pp pppp. Cea de-a doua instrucțiune BCF a determinat
; ca RB7 să fie memorat ca valoarea pinului (high).
```

Un pin configurat ca ieșire activă “Low” sau “High” nu trebuie comandat în același timp de dispozitive externe pentru a schimba nivelul logic asociat ieșirii (“SAU-cablat”, “ȘI-cablat”). Curenții de ieșire de valori mari pot determina defectări iremediabile ale circuitului.

9.2.6.2 OPERAȚII SUCCESIVE ASUPRA PORTURILOR I/O

Scrierea curentă la un port de intrare/ieșire (I/O) are loc la sfârșitul ciclului instrucțiune, în timp ce pentru citire datele trebuie să fie valide de la începutul ciclului instrucțiune. De aceea este necesară atenție dacă se execută o operație de scriere urmată de o operație de citire la/de la același port de intrare/ieșire. Secvența de instrucțiuni trebuie să permită stabilizarea tensiunii la nivel de pini (în cazul dependenței de sarcină) înainte de execuția următoarei instrucțiuni, ce determină ca datele să fie citite de către CPU. În caz contrar, este posibilă și mai probabilă citirea stării anterioare decât a celei curente. Atunci când există îndoieli, este utilă separarea acestor instrucțiuni printr-o instrucțiune NOP sau oricare altă instrucțiune care nu accesează portul I/O respectiv.

9.2.7 MODULUL DE CONVERSIE ANALOG-DIGITALĂ

Modulul de conversie analog-digitală conține patru intrări analogice. Convertorul A/D asigură o rezoluție de 8 biți a conversiei semnalelor analogice de intrare. Ieșirea circuitului de eșantionare-memorare este conectată la intrarea

convertorului A/D, ce funcționează după metoda aproximațiilor succesive. Tensiunea de referință poate fi selectată prin software fie ca fiind tensiunea de alimentare (V_{DD}), fie ca fiind tensiunea aplicată pinului RA3/AN3/VREF.

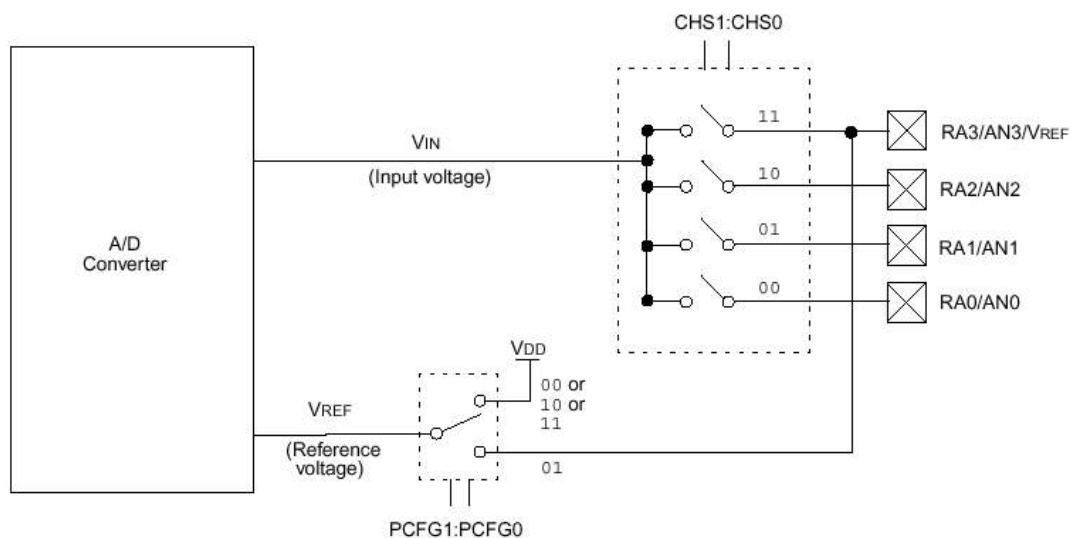


Fig. 9.8 Configurația modului de conversie A/D.

Convertorul A/D se distinge prin aceea că este capabil să funcționeze chiar dacă microcontroller-ul se găsește în modul de operare SLEEP. Pentru aceasta, ceasul de conversie A/D trebuie obținut din oscilatorul RC intern. Modulul A/D dispune de trei registre. Aceste registre sunt:

- A/D Result Register (ADRES);
- A/D Control Register 0 (ADCON0);
- A/D Control Register 1 (ADCON1).

Registrul de control ADCON0 comandă și coordonează funcționarea modului A/D.

Registrul de control ADCON1 permite configurarea funcțiilor pinilor portului. Pinii pot fi configurați ca intrări analogice (RA3 poate, de asemenea, fi configurat a sursă de tensiune de referință) sau ca intrări/ieșiri digitale.

Registrul ADRES conține rezultatul conversiei A/D. Atunci când s-a încheiat procesul de conversie A/D, rezultatul este transferat în registrul ADRES, bitul GO/DONE (ADCON0<2>) este resetat iar indicatorul de întrerupere A/D, bitul ADIF, este setat.

În urma configurării după necesități a modului A/D, trebuie selectat canalul de intrare înainte de declanșarea unui proces de conversie. Canalele de intrare analogice trebuie să aibă biții corespunzători TRIS selectați ca intrări.

După expirarea timpului de eșantionare poate fi demarat un proces de conversie A/D. Pentru efectuarea unei conversii analog-digitale trebuie îndepliniți următorii pași:

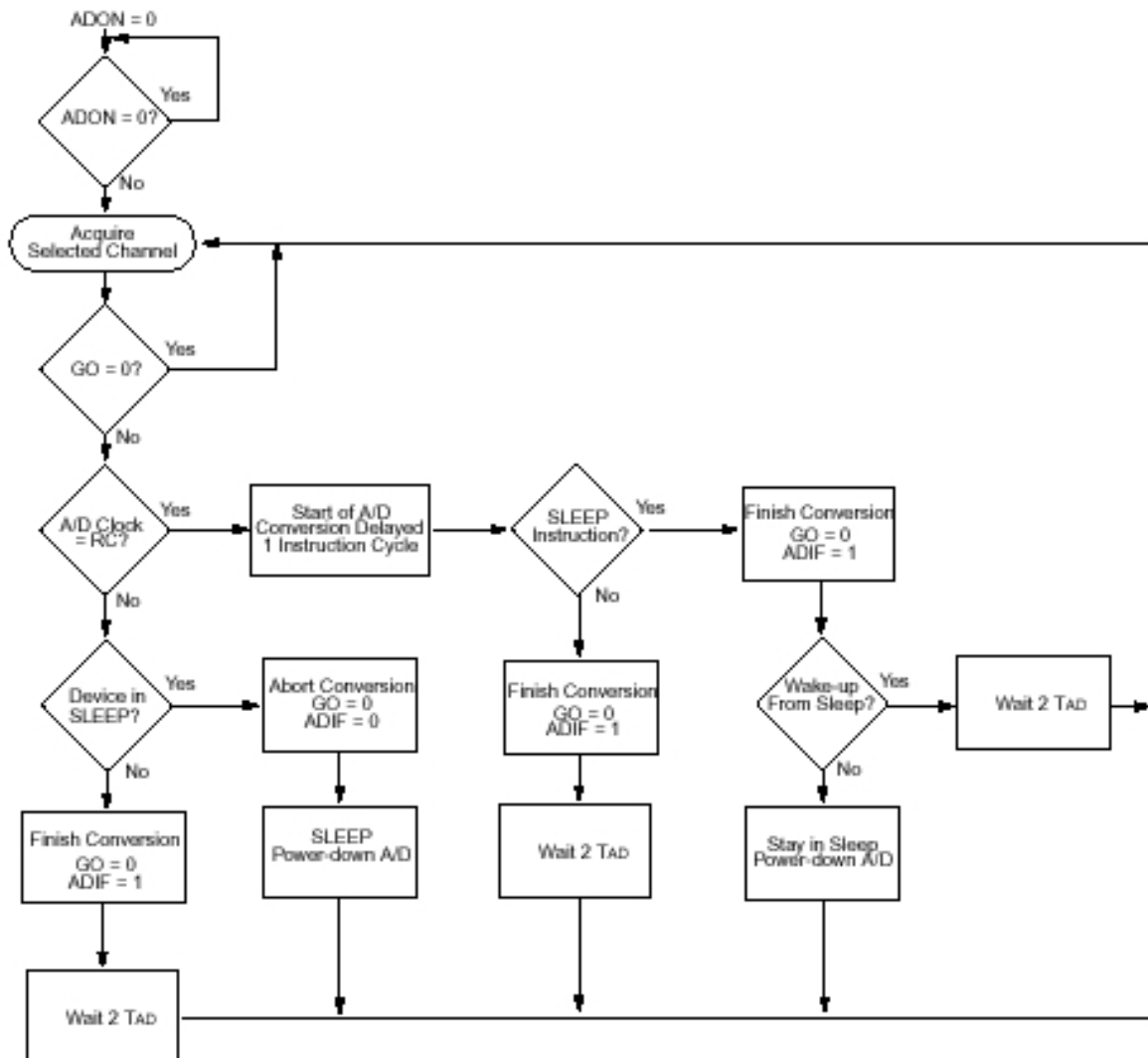


Fig. 9.9 Organigrama de efectuare a conversiei A/D.

1. configurarea modului A/D:
 - configurarea pinilor intrări analogice / referința de tensiune / intrări-ieșiri digitale (ADCON1);
 - selectarea canalului analogic de intrare (ADCON0);
 - selectarea ceasului de conversie A/D (ADCON0);
 - activarea modului A/D (ADCON0);
2. configurarea întreruperilor A/D (opțional):
 - resetarea bitului ADIF;
 - setarea bitului ADIE;
 - setarea bitului GIE;
3. respectarea timpului de eșantionare;
4. Start conversie:
 - setarea bitului GO/DONE (ADCON0);
5. așteptarea sfârșitului conversiei A/D, fie:
 - polling pentru ca bitul GO/DONE să fie resetat, sau

- așteptarea unei întreruperi;
6. citirea rezultatului conversiei în registru rezultat (ADRES), resetarea bitului ADIF (dacă este necesar);
 7. pentru următoarea conversie, înapoi la pasul 1 sau 2, după cum este necesar. Timpul de conversie pe bit este definit ca T_{AD} . Pentru începerea unui nou proces de conversie este necesar un interval de timp de întârziere de minim $2T_{AD}$.

9.2.7.1 SPECIFICAȚII PENTRU ACHIZIȚIA A/D

Pentru ca o conversie analog-digitală să fie caracterizată de precizia specificată, este necesar ca să i se permită condensatorului de memorare (C_{HOLD}) încărcarea completă până la valoarea tensiunii aplicate canalului analogic de intrare. Modelul analogic al intrării este prezentat în fig. 9.10. Impedanța sursei de semnal (R_S) și rezistența internă a comutatorului din circuitul de eșantionare-memorare (R_{SS}) afectează în mod nemijlocit timpul de încărcare al condensatorului de memorare, C_{HOLD} . Rezistența internă a comutatorului, R_{SS} , variază în funcție de tensiunea de alimentare V_{DD} . Impedanța sursei de semnal afectează decalajul de tensiune a intrării analogice (datorită curentului de pierderi).

Impedanța maximă recomandată a sursei analogice de semnal este de 10 k Ω . După selectarea canalului analogic de intrare, trebuie să expire timpul de eșantionare calculat pe baza elementelor mai sus menționate și, abia după aceasta, poate fi declanșat un proces de conversie A/D.

Nota 1: Valoarea tensiunii de referință (V_{REF}) nu are efect asupra timpului de eșantionare.

Nota 2: Condensatorul de memorare (C_{HOLD}) nu se descarcă complet după efectuarea unei conversii.

Nota 3: Impedanța maximă recomandată a sursei analogice de semnal este de 10 k Ω , fiind impusă de respectarea recomandărilor privind curentul de pierderi.

Nota 4: După încheierea unei conversii A/D, este necesară o întârziere de minimum $2,0 T_{AD}$ înainte de reînceperea unui ciclu de achiziție. Pe durata acestui timp, condensatorul de memorare nu este conectat la canalul de intrare selectat.

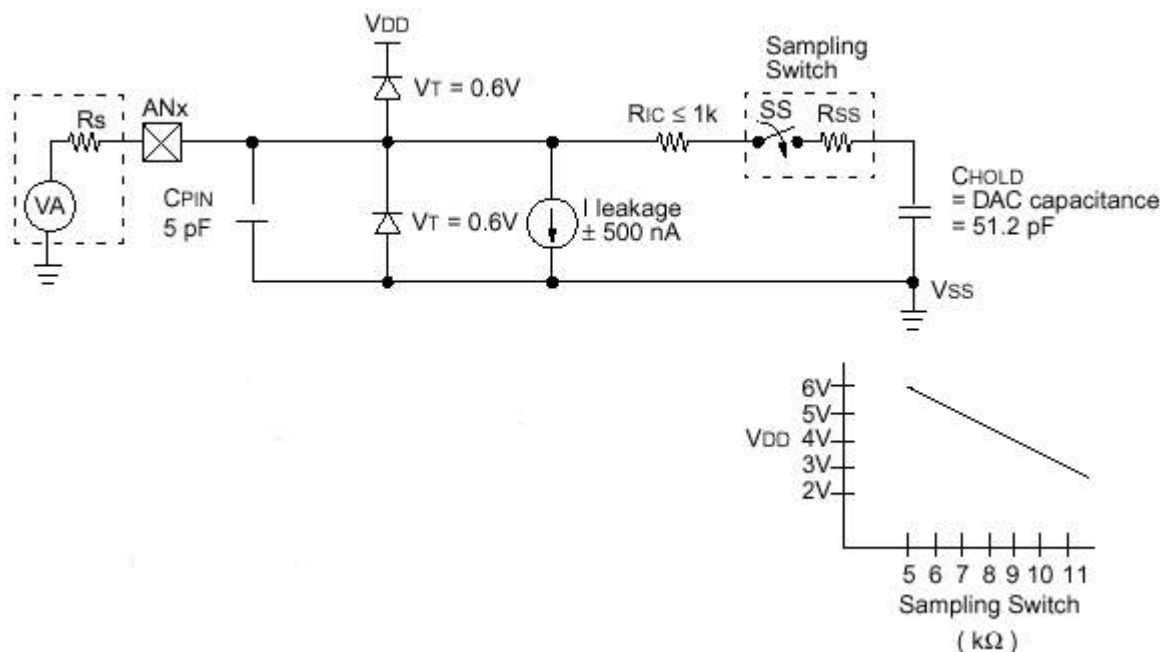


Fig. 9.10 Configurația canalelor analogice de intrare.

9.2.7.2 PARTICULARITĂȚI DE UTILIZARE A CIRCUITULUI DE CONVERSIE ANALOG-DIGITALĂ A MICROCONTROLLER-ULUI PIC16C71

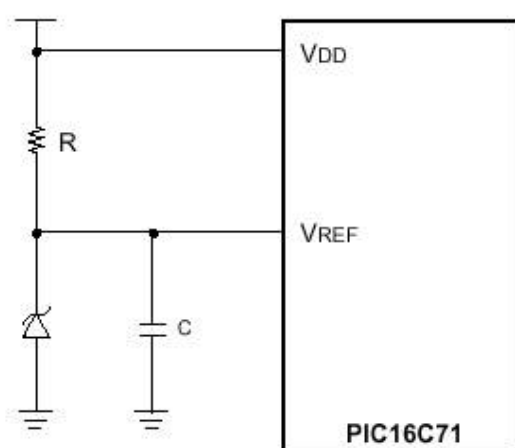
Secțiunea de conversie analog digitală a PIC16C71 este ușor de setat și de utilizat. În continuare se prezintă câteva considerații și recomandări generale asupra acestui subiect.

1. Se alege drept referință de tensiune fie tensiunea de alimentare V_{DD} fie o sursă de referință externă, V_{REF} .

Atunci când se utilizează o sursă de tensiune de referință externă, trebuie avut în vedere ca valorile analogice ale tensiunilor de intrare nu trebuie să depășească V_{REF} . O cale economică și puțin costisitoare de a genera o tensiune de referință V_{REF} constă în utilizarea unei diode Zener (fig. 9.11a). Majoritatea diodelor Zener uzuale asigură o precizie de 5%. Curenții de polarizare inversă pot fi de valori foarte reduse ($\geq 10\mu A$). Cu toate acestea, se recomandă utilizarea unor curenți de polarizare inversă de valori mai mari (1 mA ... 20 mA) din considerente de stabilitate, precum și pentru a se asigura o impedanță de ieșire mai mică a sursei de tensiune de referință V_{REF} .

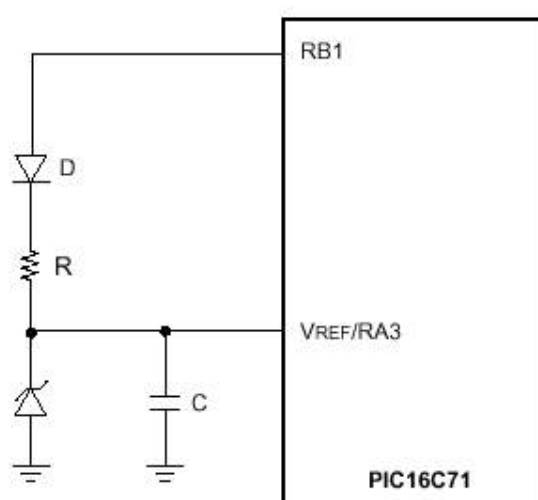
În cazul aplicațiilor sensibile la variația tensiunii de alimentare, utilizatorul poate comanda sursa de tensiune de referință V_{REF} utilizând un alt pin de intrare/ieșire (fig. 9.11b): comandând la nivel logic '1' pinul RB1, de exemplu, atunci când se utilizează secțiunea de

conversie A/D sau comandând la nivel logic '0' pinul RB1 atunci când nu este utilizat convertorul A/D. Trebuie menționat faptul că în această situație pinul RB1 nu funcționează în regim three-state. Chiar dacă V_{REF} va înregistra valori intermediare, aceasta nu va determina ca buffer-ul de intrare al pinului RB1 să absoarbă curent. Ca variantă, pot fi utilizați unul dintre pinii RA0, RA1 sau RA2 pentru a asigura curent în loc de pinul RB1. Pinul RA utilizat trebuie configurat ca pin analogic (aceasta va dezactiva buffer-ul său digital de intrare).

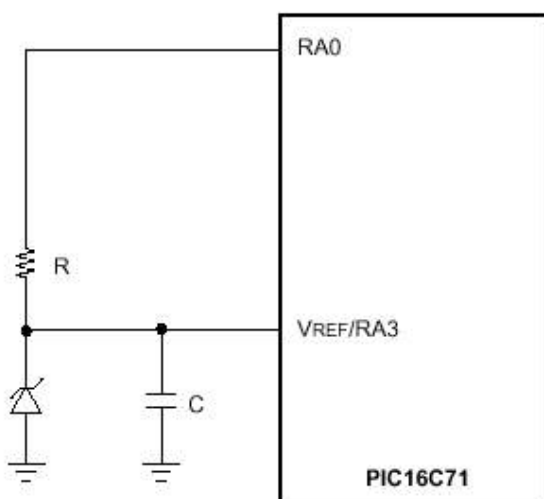


$C = 0.01 \text{ to } 0.1 \mu\text{F}$

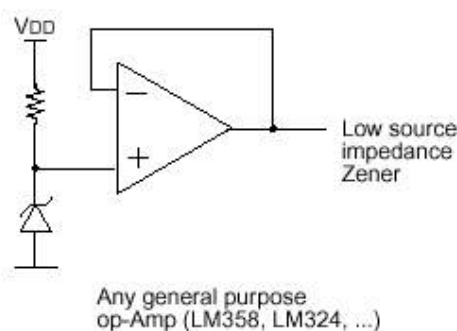
a) Tensiune de referință obținută cu diodă Zener.



b) Tensiune de referință comandată prin intermediul pinului RB1.



c) Tensiune de referință comandată prin intermediul unui pin RA.



d) Scăderea impedanței de ieșire a sursei de tensiune de referință.

Fig. 9.11 Sursa de referință externă.

În momentul actual există numeroase surse de tensiune de referință integrate, caracterizate de o precizie sensibil mai ridicată decât cea a diodelor Zener (tabelul 9.1).

Tabelul 9.1 Diode Zener și surse de tensiune de referință

Tipul diodei Zener	Tensiunea V_Z	Toleranța
1N746	3,3V	$\pm 5\%$
1N747	3,6V	$\pm 5\%$
1N748	3,9V	$\pm 5\%$
1N749	4,3V	$\pm 5\%$
1N750	4,7V	$\pm 5\%$
1N751	5,1V	$\pm 5\%$
1N752	5,6V	$\pm 5\%$
Referințe de tensiune	Tensiunea V_{REF}	Toleranța
AD580 (Maxim)	2,5V	$\pm 3\% \dots \pm 0,4\%$
LM385	2,5V	$\pm 1,5\%$
LM1004	2,5V	$\pm 1,2\%$
LT1009 (LIN. Tech.)	2,5V	$\pm 0,2\%$
LT1019 (LIN. Tech.)	5,0V	$\pm 0,2\%$
LT1021 (LIN. Tech.)	5,0V	$\pm 0,05\% \dots \pm 1\%$
LT1029 (LIN. Tech.)	5,0V	$\pm 0,2\% \dots \pm 1\%$

Ideal, sursa de tensiune de referință V_{REF} trebuie să fie caracterizată de o impedanță de ieșire cât mai redusă posibil. Conform fig. 9.11a, impedanța sursei de tensiune de referință V_{REF} este aproximativ egală cu R . Dacă se micșorează valoarea lui R (pentru micșorarea impedanței sursei de tensiune de referință) se constată o creșterea consumului. Deoarece sursa de tensiune de referință V_{REF} este utilizată pentru încărcarea rețelei de condensatoare care stă la baza funcționării convertorului A/D și a condensatorului de memorare ($C_{HOLD} \approx 51$ pF), trebuie să se asigure următoarele condiții:

$$T_{AD} = 6 \cdot (1k + R) \cdot 51,2 \text{ pF} + 1,677 \text{ } \mu\text{s}$$

în care T_{AD} reprezintă perioada ceasului de conversie A/D. Pentru $T_{AD} = 2 \mu\text{s}$; $C_{HOLD} = 50$ pF, rezultă $V_{REF} \approx 50 \Omega$.

Dacă impedanța sursei de tensiune de referință V_{REF} este mai mare decât valoarea calculată anterior, atunci perioada ceasului de conversie A/D (T_{AD}) trebuie mărită în mod corespunzător.

În tabelul 9.2 sunt prezentate câteva exemple referitor la rata maximă de conversie pe bit corelată cu impedanța sursei de tensiune de referință.

Tabelul 9.2 Rata maximă de conversie pe bit

Impedanța sursei de referință	T_{AD} (max)
1k	2,29 μ s
5k	3,52 μ s
10k	5,056 μ s
50k	16,66 μ s
100k	32,70 μ s

Pentru a obține o impedanță mai scăzută a sursei de tensiune de referință decât prin utilizarea unei diode Zener, se recomandă utilizarea unui repetor de tensiune (fig. 9.11d).

Doi dintre biții registrului ADCON1, anume PCFG1 și PCFG0, controlează modul de configurare a pinilor RA3:RA0.

Ori de câte ori unul dintre acești pini este configurat ca analogic:

- buffer-ul digital de intrare al pinului este dezactivat pentru reducerea curentului de intrare. Operația de citire a portului va interpreta ca '0' valoarea pinului corespunzător;
- bitul TRIS va continua să comande buffer-ul de ieșire al acestui pin. Astfel, în mod normal, bitul TRIS va fi setat (intrare);
- dacă bitul TRIS va fi resetat, atunci pinul corespunzător va furniza la ieșire conținutul latch-ului de date.

Ori de câte ori unul dintre acești pini vor fi comandați ca fiind digitali:

- intrarea analogică este conectată direct la convertorul A/D și, în concluzie, pinul poate fi utilizat ca intrare analogică;
- buffer-ul digital de intrare nu este dezactivat.

Astfel, utilizatorul dispune de flexibilitate în configurarea acestor pini.

Se selectează frecvența ceasului de conversie A/D (T_{AD}): $2T_{OSC}$, $8T_{OSC}$, T_{OSC} or T_{RC} (ceas RC intern). Pentru primele trei opțiuni este recomandat ca durata conversiei $T_{AD} \geq 2,0 \mu$ s. Dacă este necesar un timp de conversie riguros determinat, se recomandă selectarea T_{OSC} ca bază de timp. Dacă este necesară efectuarea unor conversii A/D pe durata SLEEP, se recomandă selectarea T_{RC} ca bază de timp.

2. **Selectarea canalului:** dacă se dorește efectuarea unor achiziții A/D monocanal, registrul ADCON1 trebuie programat cu valoarea 03h. Această operație configurează pinii A/D ca pini de I/O digitală. Dacă se dorește efectuarea unor achiziții A/D multicanal, înainte de declanșarea fiecărei conversii este necesară selectarea canalului corespunzător.
3. **Eșantionarea și conversia:** după selectarea unui nou canal analogic, este necesar un timp minim de eşantionare înainte de poziționarea

bitului GO/DONE din ADCON0 pentru declanșarea unei conversii A/D. Odată demarat un proces de conversie, se poate selecta următorul canal, însă *eșantionarea nu se efectuează decât după încheierea conversiei curente*. De aceea, este întotdeauna necesară asigurarea timpului minim de eșantionare:

- i) după sfârșitul unei conversii;
 - ii) după selectarea unui nou canal analogic;
 - iii) după activarea facilităților de conversie A/D (bitul ADON=1);
4. **Citirea rezultatului:** încheierea procesului de conversie A/D poate fi identificată fie prin polling pe bitul GO/DONE (resetat), fie prin polling pe indicatorul ADIF (setat), fie așteptând o cerere de întrerupere ADIF.

În plus:

- a) Nu se recomandă setarea biților GO/DONE și ADON în cadrul aceleiași instrucțiuni. În primul rând, se setează (activează) funcționarea convertorului A/D prin setarea bitului ADON. Ulterior trebuie să se asigure un timp de eșantionare de minim 5 μ s înainte de declanșarea procesului de conversie (setarea bitului GO/DONE).
- b) Întreruperea unui ciclu de conversie aflat în desfășurare: O conversie A/D poate fi întreruptă prin resetarea bitului GO/DONE. Convertorul A/D își va înceta funcționarea și va reveni în starea de eșantionare.

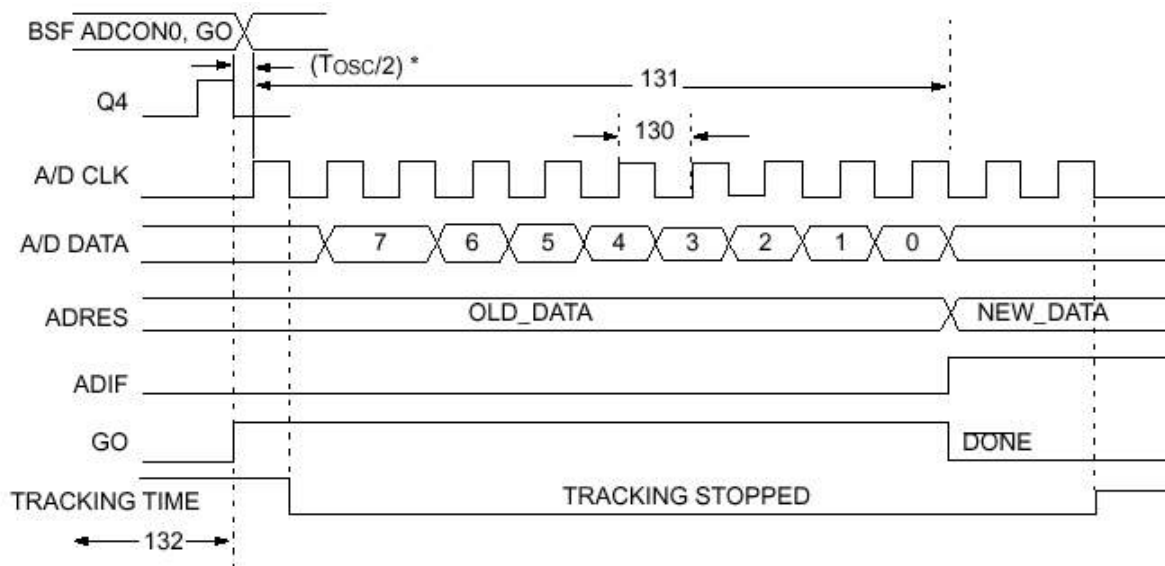


Fig. 9.12 Diagrama temporală pentru configurarea și achiziția A/D.

- c) Se recomandă utilizarea registrului ADRES ca registru dedicat: convertorul A/D scrie în registrul ADRES doar la sfârșitul procesului de conversie. Cu toate acestea este posibilă utilizarea registrului ADRES ca un registru de transfer între două conversii succesive sau atunci când convertorul A/D este dezactivat.

În următoarele patru exemple se prezintă secvențe de cod pentru manipularea modului de conversie A/D din cadrul PIC16C71.

Achiziție monocanal

```
; InitializeAD, initializeaza si seteaza nucleul hardware A/D.
; Se utilizeaza doar canalul ch2, cu oscilator RC.
InitializeAD
    bsf     STATUS, 5           ; selecteaza Bank1
    movlw   b'00000000'        ; selecteaza RA3-RA0
    movwf   ADCON1              ; ca intrari analogice
    bcf     STATUS, 5           ; selecteaza Bank0
    movlw   b'11010001'        ; selecteaza RC osc, ch2...
    movwf   ADCON0              ; activeaza A/D
Convert    call    sample-delay  ; timp de esantionare
;
    bsf     ADCON0, 2           ; start conversie A/D
loop
    btfsc   ADCON0, 2           ; conversie A/D gata?
    goto    loop               ; nu, atunci polling
    movf    adres, w           ; da, citeste valoarea A/D
```

Achiziție multicanal, în mod ciclic

```
; InitializeAD, initializeaza si seteaza nucleu hardware A/D.
; Selecteaza ch0 pana la ch3 circular, oscilator RC intern.
; Incarca rezultate la 4 adrese consecutive incepand cu
; ADTABLE (10h)
;
InitializeAD
    bsf     STATUS, RP0         ; selecteaza Bank1
    movlw   b'00000000'        ; selecteaza RA3-RA0
    movwf   ADCON1              ; ca intrari analogice
    bcf     STATUS, RP0         ; selecteaza Bank0
    movlw   b'11000001'        ; selecteaza: RC osc, ch0...
    movwf   ADCON0              ; activeaza A/D
    movlw   ADTABLE             ; fsr pozitionat la ...
    movwf   FSR                 ; inceputul tabelii
;
new_ad     call    sample_delay  ; timp esantionare
    bsf     ADCON0, GO          ; start conversie A/D
loop       btfsc   ADCON0, GO    ; conversie A/D gata?
    goto    loop               ; nu, polling
;
    movf    adres, w           ; da, citeste valoare A/D
    movwf   0                  ; incarcare indirecta
    movlw   4                  ; selecteaza canal urmator
    addwf   ADCON0              ; /
    bcf     ADCON0, ADIF        ; reset indicator intrerupere
; incrementeaza pointer corectie offset tabela.
    clrf    temp               ; sterge registru temporar
    btfsc   ADCON0, CH50        ; test lsb canal selectat
    bsf     temp, 0             ; set daca ch1 selectat
```

```

btfsc    ADCON0, CH51    ; test msb canal selectat
bsf      temp, 1         ; /
movlw    ADTABLE         ; adresa tabelii
addwf    temp, w         ; aduna cu temp
movwf    FSR             ; muta indirect la ...
goto     new_ad

```

Manipularea întreruperilor A/D

```

        org      0x00
        goto     start
        org      0x04
        goto     service_ad    ; vector intrerupere
;
;      org 0x10
start
        movlw    b'00000000'    ; initializare porturi I/O
        movwf    PORT_B
        tris     PORT_B
;
        call     InitializeAD
update
        bcf      flag, adover    ; reset software flag A/D
        call     SetupDelay      ; delay >= 10uS.
        bcf      ADCON0, adif    ; reset A/D flag intrerupere
        bsf      ADCON0, go      ; start conversie A/D
        bsf      INTCON, gie     ; activare globala intreruperi
loop
        btfsc    flag, adover    ; conversie A/D gata?
        goto     update          ; da, start noua conversie
        goto     loop            ; nu, verifica
; InitializeAD, initializeaza si seteaza nucleul hardware A/D.
; Selecteaza ch0 - ch3, oscilator RC, intreruperi A/D.
InitializeAD
        bsf      STATUS, RP0     ; selecteaza Bank1
        movlw    b'00000000'    ; selecteaza RA0-RA3...
        movwf    ADCON1         ; ca intrari analogice
        bcf      STATUS, RP0     ; selecteaza Bank0
        clrf     INTCON          ; sterge toate intreruperile
        bsf      INTCON, ADIE    ; activeaza intrerupere A/D
        movlw    b'11010001'    ; selecteaza: RC, ch2...
        movwf    ADCON0         ; activeaza convertor A/D
        return
;
service_ad
        btfss    ADCON0, ADIF    ; intrerupere A/D?
        retfie    ; nu, atunci ignora
        movf     ADRES, W        ; citeste valoare A/D
        return    ; nu activez intreruperi

```

Efectuarea conversiei A/D în modul “sleep”

```

; InitializeAD, initializeaza si seteaza nucleul hardware A/D.
; Selecteaza ch0 - ch3, oscilator RC intern.

```

```
; In cursul unei conversii initiaza modul sleep.
; Aceasta va minimiza interferentele si zomotele digitale.
; Oscilatorul RC al A/D trebuie folosit obligatoriu
; in aceasta situatie.
;
InitializeAD
    bsf        STATUS, RP0        ; selecteaza Bank1
    movlw      b'00000000'        ; selecteaza RA0-RA3...
    movwf      ADCON1             ; ca intrari analogice
    bcf        STATUS, RP0        ; selecteaza Bank0
    movlw      b'11000001'        ; selecteaza osc RC, ch0...
    movwf      ADCON0             ; activeaza A/D si ADIE
    movlw      ADTABLE            ; fsr pointeaza la inceputul...
    movwf      FSR                ; tabeli
;
new_ad
    bsf ADCON0, GO                ; start conversie A/D
    sleep                        ; mod sleep
; la terminarea conversiei programul continua din acest punct
;
    movf       ADRES, w           ; citeste valoarea A/D
```

9.3 CARACTERISTICI CONSTRUCTIVE. TESTARE ȘI CALIBRARE EASY SCOPE

Osciloscopul digital Easy Scope este construit pe circuit imprimat dublu placat. Toate componentele prezentate în cadrul schemei electrice sunt amplasate pe cablaj, pe o singură față (fața cu componente sau plantată). Rezistențele utilizate pentru implementarea divizoarelor de intrare, precum și cele care intră în componența amplificatoarelor sumator-subtractor sunt cu toleranțe reduse (0,1%), asigurând precizia impusă acestui echipament. Sunt prevăzute condensatoare de filtrare a surselor de tensiune de alimentare. Intrările sunt prevăzute cu mufe BNC cu plantare pe circuitul imprimat pentru a se realiza compatibilizarea deplină cu sondele de osciloscop existente.

În ceea ce privește punerea în funcțiune a osciloscopului, s-au urmărit următoarele etape:

- circuitul imprimat a fost echipat cu toate componentele pasive, mufele BNC de intrare, mufele de alimentare, mufa pentru interconectare cu portul paralel al calculatorului;
- au fost montate cele două stabilizatoare de tensiune integrate LM7805 (+5V) și LM7905 (-5V) și a fost alimentat montajul. Au fost măsurate cu un voltmetru digital tensiunile de la ieșirile stabilizatoarelor de tensiune integrate ($\pm 5V$) și tensiunile de alimentare ale capsulei TL084 ($\pm 9V$);
- s-a măsurat cu un voltmetru digital tensiunea de pe cursorul potențiometrului multitură R3 și această tensiune a fost ajustată la

- valoarea de $-2,3\text{V}$;
- în urma verificării corectitudinii tensiunilor din cadrul schemei, s-a întrerupt alimentarea și s-a montat pe soclul corespunzător circuitul TL084 (amplificator operațional cuadru). S-a alimentat montajul. Intrările canalelor A și B au fost conectate la masă și s-au verificat tensiunile de la ieșirile circuitelor de condiționare a semnalelor ($+2,5\text{V}$). A fost reajustată tensiunea de referință (tipic $-2,3\text{V}$) pentru a se obține la intrarea convertorului A/D din microcontroller tensiunea de $2,5\text{V}$ în cazul aplicării unor tensiuni de intrare nule (cod de ieșire A/D 7FH);
 - s-a decuplat alimentarea și a fost montat pe soclu și circuitul PIC16C71 programat în mod corespunzător (codul sursă în limbaj de asamblare este prezentat în capitolul 6);
 - se cuplează osciloscopul Easy Scope la portul paralel al calculatorului utilizând un cablu paralel, cu lungime maximă de $1,8\text{m}$, cu mufe DB25M la ambele capete;
 - se alimentează osciloscopul Easy Scope și se rulează programul EasyScope.exe de pe PC;
 - în urma apelării programului EasyScope.exe pentru prima dată există posibilitatea apariției unui mesaj indicând absența unui fișier de configurare. Este un mesaj normal la rulare pentru prima dată a aplicației. Pentru continuare se apasă orice tastă și apare un meniu de opțiuni. Selectarea opțiunilor se face prin utilizarea tastelor funcționale **F1...F10**. Se selectează opțiunea **F8: CONFIGURE**. Se selectează portul paralel la care este conectat osciloscopul (LPT1 sau LPT2), prin utilizarea tastelor 1 sau 2 urmate de ENTER. Urmează selectarea culorii liniilor de reprezentare pentru canalele A și B, precum și culoarea grid-ului. Se introduc culorile dorite sub formă numerică. Aceste atribute ale reprezentării grafice pot fi modificate ulterior oricând. Selectarea acestor opțiuni va determina generarea unui fișier de configurare. Ulterior, s-a selectat opțiunea F1 - DVM mode. Acest program va afișa un heading, citirea A/D pentru canalul B și apoi gama tensiunilor ($\pm 2,4$, ± 6 or ± 12), citirea A/D în hex și tensiunea măsurată pentru canalul A. Se scurtcircuitază intrarea canalului A la masă pentru a obține tensiunea de $0,0\text{V}$ la intrare. Se selectează gama de $\pm 2,4\text{V}$ prin intermediul comutatorului SW1. Se ajustează potențiometrul R3 (dacă mai este necesar) până când valoarea citită pe canalul A este stabilă și are valoarea 7FH și tensiunea citită este $0,0\text{V}$. Dacă se scurtcircuitază intrarea canalului B la masă, citirea convertorului A/D pentru acest canal va fi tot 7FH. Se testează funcționarea convertorului A/D pentru fiecare canal prin conectarea intrărilor la o baterie. Valoarea citită trebuie să se modifice conform

tensiunii de la bornele bateriei. În acest moment testarea și calibrarea osciloscopului Easy Scope a fost efectuată cu succes.

9.4 DESCRIEREA MENIULUI APLICAȚIEI EASYSCOPE

Pe durata calibrării osciloscopului digital Easy Scope a fost utilizat modul de lucru DVM (opțiunea **F1**), așa ca în cele ce urmează nu mai sunt necesare explicații suplimentare referitoare la acest mod de lucru.

EASYSCOPE

F1 - DVM mode
F2 - SCOPE mode : SINGLE Channels 1
F3 - TRIGGER level : 0.00
F4 - TRIGGER slope : +
F5 - TRIGGER position : 4/8
F6 - ACQUISITION period : 100 uS Frequency 10.00 KHz
F7 - ACQUIRE
F8 - CONFIGURE
F9 - DISPLAY DATA BUFFER
F10 - Exit

Fig. 9.13 Panoul aplicației EASYSCOPE.

Opțiunea **F2** selectează modul de funcționare al Easy Scope: CONTINUOUS, SINGLE sau NORMAL. Se selectează, de asemenea, numărul canalelor de intrare active (1= A, 2 = A&B). În modul CONTINUOUS, osciloscopul va eșantiona în mod continuu semnalele de intrare și va actualiza afișarea, fără a utiliza opțiunea trigger. În modul SINGLE, osciloscopul va achiziționa și afișa un singur eșantion al semnalelor de intrare utilizând triggerul. În modul NORMAL, osciloscopul va achiziționa și va afișa datele utilizând trigger-ul, ca un osciloscop analogic.

Opțiunea **F3** permite setarea nivelului tensiunii de trigger.

Opțiunea **F4** permite setarea polarității trigger-ului ca “+” sau “-”.

Opțiunea **F5** permite selectarea poziției trigger-ului în cadrul datelor achiziționate în incremente de 1/8. Valoarea 0/8 semnifică faptul că toate datele afișate sunt ulterioare semnalului de trigger, ca în cazul unui osciloscop analogic. Valoarea 3/8 semnifică faptul că 3/8 dintre datele afișate sunt achiziționate înainte de semnalul de trigger iar restul de 5/8 sunt achiziționate după acest semnal.

Opțiunea **F6** permite setarea perioadei de achiziție în μ s.

Opțiunea **F7** startează modul osciloscop, datele fiind afișate grafic în cadrul ecranului. Pentru ieșirea din modul de lucru DVM sau ACQUIRE se utilizează tasta ESC.

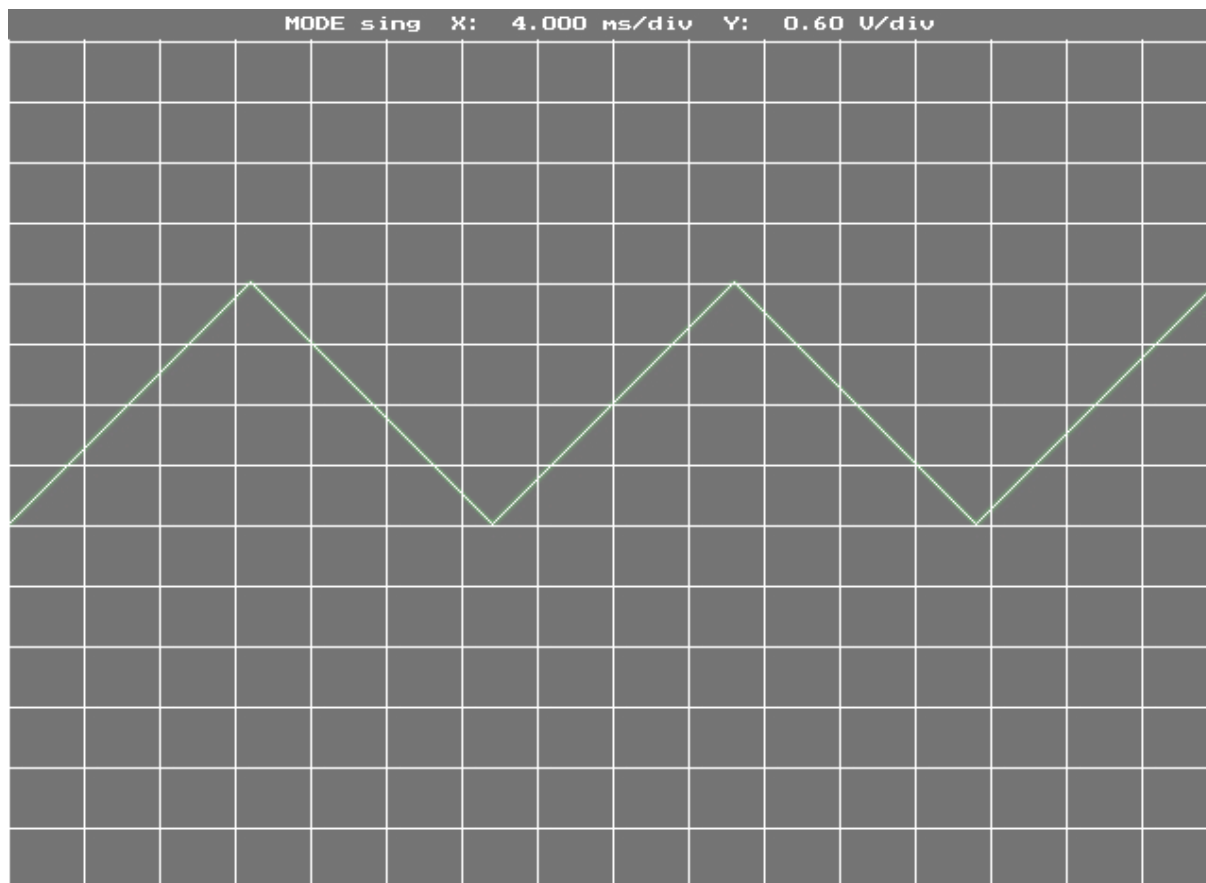


Fig. 9.14 Vizualizarea buffer-ului de date.

Easy Scope reprezintă un instrument ieftin și flexibil, oferind utilizatorului șansa să lucreze cu un Digital Storage Oscilloscope pentru o fracțiune nesemnificativă din prețul unui osciloscop uzual.

9.5 CODUL SURSĂ ÎN LIMBAJ DE ASAMBLARE AL MICROCONTROLLER-ULUI PIC16C71 PENTRU OSCILOSCOPUL DIGITAL EASY SCOPE

```
; EASY SCOPE rev 1.0   RGB 19/04/2000
; Fisier sursa PIC16C71 pentru EASY SCOPE
;
    list P=16C71
#include "P16CXX.INC"
    __FUSES      _XT_OSC&_WDT_OFF&_CP_OFF
    timecnt      equ 10h
    temp_w       equ 11h
    temp_status  equ 12h
    head         equ 13h
    poweron      equ 14h
    temp_fsr     equ 15h
    tmp          equ 16h
    mode         equ 17h

    org          0
    goto         start
    org          4

int:
    movwf        temp_w
    swapf        STATUS,W
    movwf        temp_status
    movf         FSR,w
    movwf        temp_fsr
    btfsc        mode,0
    goto         int2
    bsf          ADCON0,GO
    movf         timecnt,w
    movwf        TMR0
    incf         head,w
    andlw        2fh
    movwf        head
    xorwf        FSR,w
    btfsc        STATUS,Z
    goto         overflow
    movf         head,w
    movwf        FSR
    movf         ADRES,w
    movwf        INDF
    bcf          INTCON,INTF
    bcf          INTCON,T0IF
    movf         temp_fsr,w
    movwf        FSR
    swapf        temp_status,w
    movwf        STATUS
    swapf        temp_w,f
    swapf        temp_w,w
    retfie
```

```
int2:
    btfsc    mode,1
    goto     int3
    bcf      ADCON0,CHS0
    bsf      mode,1
    goto     int4
int3:
    bsf      ADCON0,CHS0
    bcf      mode,1
    nop
    nop
int4:
    nop
    nop
    nop
    nop
    nop
    nop
    movf     timecnt,w
    movwf    TMR0
    incf     head,w
    andlw    2fh
    movwf    head
    xorwf    FSR,w
    btfsc    STATUS,Z
    goto     overflow
    movf     head,w
    movwf    FSR
    movf     ADRES,w
    movwf    INDF
    bsf      ADCON0,GO
    bcf      INTCON,INTF
    bcf      INTCON,T0IF
    movf     temp_fsr,w
    movwf    FSR
    swapf    temp_status,w
    movwf    STATUS
    swapf    temp_w,f
    swapf    temp_w,w
    retfie
overflow:
    movf     temp_fsr,w
    movwf    FSR
    swapf    temp_status,w
    movwf    STATUS
    swapf    temp_w,f
    swapf    temp_w,w
    return
start:
    bcf      STATUS,RP0
    movlw    1fh
    movwf    PORTA
    movlw    00fh
    movwf    PORTB
    bsf      STATUS,RP0
```

```
    movlw    02h
    movwf    ADCON1
    movlw    01h
    movwf    OPTION_REG
    movlw    1bh
    movwf    TRISA
    movlw    0f0h
    movwf    TRISB
    bcf      STATUS,RP0
    movlw    5ah
    xorwf    poweron,w
    btfsc    STATUS,Z
    goto     start1
    movlw    0a7h
    movwf    timecnt
    movlw    5ah
    movwf    poweron
start1:
    btfsc    PORTB,7
    goto     scope0
    btfsc    PORTA,4
    goto     adcona
diag:
    bsf      PORTA,2
    bsf      PORTB,0
    bsf      PORTB,1
    bsf      PORTB,2
    bsf      PORTB,3
    nop
    nop
    bcf      PORTA,2
    bcf      PORTB,0
    bcf      PORTB,1
    bcf      PORTB,2
    bcf      PORTB,3
    goto     diag
adcona:
    btfsc    PORTB,6
    goto     adconb
    movlw    81h
    movwf    ADCON0
    goto     adconc
adconb:
    movlw    89h
    movwf    ADCON0
adconc:
    btfsc    PORTA,3
    goto     adconc
    bsf      ADCON0,GO
adconcl:
    btfsc    ADCON0,GO
    goto     adconcl
    movf     ADRES,w
    movwf    PORTB
    movwf    tmp
```

```

        bcf          PORTA, 2
        nop
        nop
adconc2:
        btfss       PORTA, 3
        goto        adconc2
        bsf         PORTA, 2
        swapf       tmp, w
        movwf       PORTB
adcon2a:
        btfsc       PORTA, 3
        goto        adcon2a
        bcf         PORTA, 2
        nop
        nop
adcon3:
        btfss       PORTA, 3
        goto        adcon3
        bsf         PORTA, 2
        goto        adconc
scope0:
        btfsc       PORTB, 6
        goto        scope1
tbit:
        bcf         PORTB, 3
        bsf         PORTB, 2
        btfsc       PORTB, 4
        goto        tbitab
        bcf         PORTB, 0
        goto        tbitac
tbitab:
        bsf         PORTB, 0
tbitac:
        btfsc       PORTB, 5
        goto        tbitae
        bcf         PORTB, 1
        goto        tbitaf
tbitae:
        bsf         PORTB, 1
tbitaf:
        clrf        tmp
tbit0:
        btfsc       PORTA, 3
        goto        tbit0
        btfsc       PORTB, 7
        bsf         tmp, 0
        bcf         PORTA, 2
        nop
        nop
tbit0a:
        btfss       PORTA, 3
        goto        tbit0a
        bsf         PORTA, 2
tbit1:
        btfsc       PORTA, 3

```

```
        goto      tbit1
        btfsc     PORTB, 7
        bsf       tmp, 1
        bcf       PORTA, 2
        nop
        nop
tbit1a:
        btfss     PORTA, 3
        goto      tbit1a
        bsf       PORTA, 2
tbit2:
        btfsc     PORTA, 3
        goto      tbit2
        btfsc     PORTB, 7
        bsf       tmp, 2
        bcf       PORTA, 2
        nop
        nop
tbit2a:
        btfss     PORTA, 3
        goto      tbit2a
        bsf       PORTA, 2
tbit3:
        btfsc     PORTA, 3
        goto      tbit3
        btfsc     PORTB, 7
        bsf       tmp, 3
        bcf       PORTA, 2
        nop
        nop
tbit3a:
        btfss     PORTA, 3
        goto      tbit3a
        bsf       PORTA, 2
tbit4:
        btfsc     PORTA, 3
        goto      tbit4
        btfsc     PORTB, 7
        bsf       tmp, 4
        bcf       PORTA, 2
        nop
        nop
tbit4a:
        btfss     PORTA, 3
        goto      tbit4a
        bsf       PORTA, 2
tbit5:
        btfsc     PORTA, 3
        goto      tbit5
        btfsc     PORTB, 7
        bsf       tmp, 5
        bcf       PORTA, 2
        nop
        nop
tbit5a:
```

```

        btfss      PORTA, 3
        goto      tbit5a
        bsf       PORTA, 2
tbit6:
        btfsc     PORTA, 3
        goto      tbit6
        btfsc     PORTB, 7
        bsf       tmp, 6
        bcf       PORTA, 2
        nop
        nop
tbit6a:
        btfss     PORTA, 3
        goto      tbit6a
        bsf       PORTA, 2
tbit7:
        btfsc     PORTA, 3
        goto      tbit7
        btfsc     PORTB, 7
        bsf       tmp, 7
        bcf       PORTA, 2
        nop
        nop
tbit7a:
        btfss     PORTA, 3
        goto      tbit7a
        bsf       PORTA, 2
        movf      tmp, w
        movwf     timecnt
        bcf       mode, 0
tbitm:
        btfsc     PORTA, 3
        goto      tbitm
        btfsc     PORTB, 7
        bsf       mode, 0
        bcf       PORTA, 2
        nop
        nop
tbitma:
        btfss     PORTA, 3
        goto      tbitma
        bsf       PORTA, 2
        goto      tbit
scopel:
        movlw     81h
        movwf     ADCON0
        movlw     20h
        movwf     head
        movwf     FSR
        bcf       mode, 1
        clrf      TMR0
        bsf       INTCON, T0IE
        bsf       INTCON, GIE
        bsf       ADCON0, GO
        movf      timecnt, w

```



```
        movwf    TMR0
loop:
        movf     FSR,w
loopa:
        xorwf    head,w
        btfsc    STATUS,Z
        goto     loop
        movf     INDF,w
        movwf    PORTB
loop1:
        btfsc    PORTA,3
        goto     loop1
        bcf      PORTA,2
        swapf    INDF,w
loop3:
        btfss    PORTA,3
        goto     loop3
        bsf      PORTA,2
        movwf    PORTB
loop4:
        btfsc    PORTA,3
        goto     loop4
        bcf      PORTA,2
        incf     FSR,w
        andlw    2fh
loop5:
        btfss    PORTA,3
        goto     loop5
        bsf      PORTA,2
        movwf    FSR
        goto     loopa
        end
```

9.6 CODUL SURSĂ ÎN LIMBAJ C PENTRU OSCIOSCOPUL DIGITAL EASY SCOPE

```
#include <stdio.h>
#include <dos.h>
#include <conio.h>
#include <graphics.h>

#define F1 0x3b00
#define F2 0x3c00
#define F3 0x3d00
#define F4 0x3e00
#define F5 0x3f00
#define F6 0x4000
#define F7 0x4100
#define F8 0x4200
#define F9 0x4300
#define F10 0x4400

#define MINPERIOD 20
#define MAXPERIOD 267

#define RESETPIN 0x20
#define MODEAPIN 0x08
#define MODEBPIN 0x10
#define MODECPIN 0x40
#define REQPIN 0x80

#define ACKPIN 0x08

unsigned char data[1024],buf[1024];
unsigned char lookup[256];
int dataport=0x278,statusport=0x279,portval;
int first=0,last=1023,ch1color=14,ch2color=13,gridcolor=7;
int smode=0,channels=0,trigslope=1,trigpos=4,period=100,acqrage=0,easyrev=0;
float volrange=2.4,triglevel=0.0,acqrate=10000.0,
main()
{
int c,i;

for(i=0;i<1024;i++)buf[i]=(((i%256)<128)?i:256-i);
makelookup(lookup);
rdconfig();
for(;;){
clrscr();
printf("\n\t\tEASYSCOPE Rev 1.0\n\n");
printf("\t\t(C) COPYRIGHT 2000 A&C\n");
printf("\t\tAll Rights Reserved\n");
easyrev=(getconfig()>>6)&0x03;
printf("\t\tEASYSCOPE REV %x\n",easyrev);
printf("\t\tF1 - DVM mode\n");
printf("\t\tF2 - SCOPE mode : %s Channels %d\n",
smode?(smode==1?"SINGLE":"NORM"):"CONTINUOUS",channels+1);
printf("\t\tF3 - TRIGGER level : %5.2f\n",triglevel);
printf("\t\tF4 - TRIGGER slope : %s\n",trigslope?"+":"-");
printf("\t\tF5 - TRIGGER position : %d/8\n",trigpos);
```

```

printf("\t\tF6 - ACQUISITION period : %3d uS Frequency %5.2f KHz\n"
      ,period,acqrate*0.001);
printf("\t\tF7 - ACQUIRE\n");
printf("\t\tF8 - CONFIGURE\n");
printf("\t\tF9 - DISPLAY DATA BUFFER\n");
printf("\t\tF10 - exit\n");
c=bioskey(0);
switch(c){
    case F1:      dvm();
                  break;
    case F2:      channels++;
                  channels &=0x01;
                  if(!channels)smode++;
                  if(smode>2)smode=0;
                  break;
    case F3:      printf("Enter trigger level in volts :");
                  scanf("%f",&triglevel);
                  break;
    case F4:      trigslope++;
                  trigslope &=0x01;
                  break;
    case F5:      trigpos++;
                  trigpos &=0x07;
                  break;
    case F6:      if(easyrev==2){
                  do{
                      printf("select acquisition range Standard/Low (S/L)\n");
                      c=getch()&0xdf;
                  }while((c!='S')&&(c!='L'));
                  }
                  if(c=='L'){
                      printf("enter acquisition period in uS (1600 - 17088) :");
                      scanf("%d",&period);
                      i=period/64;
                      if((i<MINPERIOD)||i>MAXPERIOD)
                          i=100;
                      period=i*64;
                      acqrange=1;
                  }
                  else {
                      printf("enter acquisition period in uS (25 - 267) :");
                      scanf("%d",&period);
                      if((period<MINPERIOD)||period>MAXPERIOD)
                          period=100;
                      acqrange=0;
                      i=period;
                  }
                  acqrate=1000000.0/(double)period;
                  c=configeasy(267-i);
                  voltrange=(c==0x30?12.0:(c==0x10?6.0:2.4));
                  break;
    case F7:      scope();
                  break;
    case F8:      configure();
                  break;
    case F9:      display();
                  break;
    case F10:     exit(0);
}
}

```

```

}

makelookup()
{
int i,data,tmp;

for(i=0;i<256;i++){
    switch(i&0xc0){
        case 0x00:    tmp=0x40;
                      break;
        case 0x40:    tmp=0xc0;
                      break;
        case 0x80:    tmp=0x00;
                      break;
        case 0xc0:    tmp=0x80;
                      break;
    }
    data=i&0x3f;
    data |=tmp;
    switch(i&0x0c){
        case 0x00:    tmp=0x04;
                      break;
        case 0x04:    tmp=0x0c;
                      break;
        case 0x08:    tmp=0x00;
                      break;
        case 0x0c:    tmp=0x08;
                      break;
    }
    data &=0xf3;
    data |=tmp;
    lookup[i]=data;
}
}

datain(unsigned char buf[],int pre,int post,int trig,int slope)
{
int data,data1,i=0,timeout=0,lasttrig=0,triggered=0;
int t;

if(trig== -1)triggered=1;
while(post){
    outportb(dataport,0x7f);
    for(timeout=0;timeout<32766;timeout++)
        if(0==((data=inportb(statusport))&ACKPIN))break;
    data >>=4;
    if(timeout==32766)return(-1);
    outportb(dataport,0xff);
    for(timeout=0;timeout<32766;timeout++)
        if((inportb(statusport)&ACKPIN))break;
    while(!(inportb(statusport)&ACKPIN));
    outportb(dataport,0x77);
    if(timeout==32766)return(-1);
    for(timeout=0;timeout<32766;timeout++)
        if(!((data1=inportb(statusport))&ACKPIN))break;
    while((data1=inportb(statusport))&ACKPIN);
    outportb(dataport,0xff);
    if(timeout==32766)return(-1);
    data |=(data1&0xf0);
    data=lookup[data];
}
}

```

```

    buf[i++]=data;
    i &=0x3ff;
    if(pre)pre--;
    else{
        if(triggered)post--;
        else {
            if(slope){
                if(data>trig){
                    if(lasttrig)triggered=1;
                }
                else lasttrig=1;
            }
            else {
                if(data<trig){
                    if(lasttrig)triggered=1;
                }
                else lasttrig=1;
            }
        }
    }
    while(!(inportb(statusport)&ACKPIN));
    for(timeout=0;timeout<32766;timeout++)
        if((inportb(statusport)&ACKPIN))break;
    if(timeout==32766)return(-1);
}
return(i);
}

datain2(unsigned char buf[],int pre,int post,int trig,int slope)
{
    int data,data1,i=0,timeout=0,lasttrig=0,triggered=0;
    int t;

    if(trig== -1)triggered=1;
    while(post){
        outportb(dataport,0x7f);
        for(timeout=0;timeout<32766;timeout++)
            if(0==((data=inportb(statusport))&ACKPIN))break;
        data >>=4;
        if(timeout==32766)return(-1);
        outportb(dataport,0xff);
        for(timeout=0;timeout<32766;timeout++)
            if((inportb(statusport)&ACKPIN))break;
        while(!(inportb(statusport)&ACKPIN));
        outportb(dataport,0x77);
        if(timeout==32766)return(-1);
        for(timeout=0;timeout<32766;timeout++)
            if(!((data1=inportb(statusport))&ACKPIN))break;
        while((data1=inportb(statusport))&ACKPIN);
        outportb(dataport,0xff);
        if(timeout==32766)return(-1);
        data |= (data1&0xf0);
        data=lookup[data];
        buf[i++]=data;
        i &=0x3ff;
        if(pre)pre--;
        else{
            if(triggered)post--;
            else if(i&0x01){
                if(slope){

```

```
        if(data>trig){
            if(lasttrig)triggered=1;
        }
        else lasttrig=1;
    }
    else {
        if(data<trig){
            if(lasttrig)triggered=1;
        }
        else lasttrig=1;
    }
}
while(!(inportb(statusport)&ACKPIN));
for(timeout=0;timeout<32766;timeout++)
    if((inportb(statusport)&ACKPIN))break;
if(timeout==32766)return(-1);
}
return(i);
}
```

```
int configeasy(int count)
{
    int mask,ret,endmask=0x200;

    outportb(dataport,0xd0);
    delay(100);
    outportb(dataport,0xf0);
    delay(5);
    ret=lookup[inportb(statusport)];
    printf("configeasy ret %02x\n",ret);
    getch();
    count &=0xff;
    if((channels)&&(ret&0xc0))count |=0x100;
    else channels=0;
    if(ret&0x80){
        if(acqrage)count |=0x200;
        endmask=0x400;
    }
    for(mask=0x01;mask<endmask;mask <=<=1){
        if(count&mask){
            outportb(dataport,0xf0);
            outportb(dataport,0x70);
        }
        else {
            outportb(dataport,0xe0);
            outportb(dataport,0x60);
        }
        while(inportb(statusport)&ACKPIN);
        outportb(dataport,0xf0);
        while(!(inportb(statusport)&ACKPIN));
    }
    outportb(dataport,0xf0);
    return(ret);
}
```

```
int getconfig()
{
    outportb(dataport,0xd0);
    delay(100);
```

```
outportb(dataport,0xf0);
delay(5);
return(lookup[inportb(statusport)]);
}

dvm()
{
int ad,con,ad1;
float val;

clrscr();
printf("EASY SCOPE\n");
printf("Digital Voltmeter Mode\n\n");

for(;;){
    con=getconfig();
    ad=getad0();
    ad1=getad1();
    con &=0x30;
    if(con==0x30)val=(ad-127)*0.09375;
    if(con==0x10)val=(ad-127)*0.046875;
    if(con==0x20)val=(ad-127)*0.01875;
    printf("Ch 2 %02x Ch 1 Range %5.2f A/D reading %02x Voltage %6.2f\r",
        ad1,con==0x30?12.0:(con==0x10?6.0:2.4),ad,val);
    if(kbhit()){if(getch()==27)break;}
}
}

int getad0()
{
int val;

outportb(dataport,portval=MODECPIN|REQPIN);
delay(100);
outportb(dataport,portval |=RESETPIN);
delay(1);
outportb(dataport,portval & ~REQPIN);
while(inportb(statusport)&ACKPIN);
val=inportb(statusport)>>4;
outportb(dataport,portval);
while(!(inportb(statusport)&ACKPIN));
outportb(dataport,portval & ~REQPIN);
while(inportb(statusport)&ACKPIN);
val |=inportb(statusport)&0xf0;
outportb(dataport,portval);
val=lookup[val];
return(val);
}

int getad1()
{
int val;

outportb(dataport,portval=MODECPIN|MODEAPIN|REQPIN);
delay(100);
outportb(dataport,portval |=RESETPIN);
delay(1);
outportb(dataport,portval & ~REQPIN);
while(inportb(statusport)&ACKPIN);
val=inportb(statusport)>>4;
```

```

outportb(dataport,portval);
while(!(inportb(statusport)&ACKPIN));
outportb(dataport,portval & ~REQPIN);
while(inportb(statusport)&ACKPIN);
val |=inportb(statusport)&0xf0;
outportb(dataport,portval);
val=lookup[val];
return(val);
}
scope()
{
int gdriver=VGA,gmode=VGAHI,errorcode;
int j,w,i,position,pre,post,c,cold,trigad,pass=0;
char linebuf[128];
configeasy(267-period);
c=cold=getconfig()&0x30;
voltrange=(c==0x30?12.0:(c==0x10?6.0:2.4));
if(smode)trigad=triglevel/voltrange*127.0+127;
else trigad= -1;
if((trigad>253)||((trigad<2))trigad= -1;
pre=(640*trigpos)/8+1;
post=640-pre+1;
errorcode=registerbgidriver(EGAVGA_driver);
if(errorcode<0){
    printf("graphics error: %s\n",grapherrormsg(errorcode));
    printf("press any key\n");
    getch();
    return(0);
}
initgraph(&gdriver,&gmode,"");
errorcode=graphresult();
if(errorcode != grOk){
    printf("graphics error: %s\n",grapherrormsg(errorcode));
    printf("press any key\n");
    getch();
    return(0);
}
setcolor(15);
setwritemode(COPY_PUT);
cleardevice();
setfillstyle(EMPTY_FILL,0);
sprintf(linebuf," EASY SCOPE");
w=textwidth(linebuf);
bar(319-w/2,5,319+w/2,txtheight(linebuf)+5);
outtextxy(319-w/2,5,linebuf);
sprintf(linebuf,"MODE %s X: %6.3f ms/div Y: %5.2f V/div",
    smode?((smode==1)?"sing":"norm"):"cont", (float)period*40.0/1000.0,
    voltrange/4.0);
w=textwidth(linebuf);
outtextxy(319-w/2,18,linebuf);
setcolor(gridcolor);
for(position=0;position<640;position +=40)
    line(position,479,position,30);
for(i=479;i>30;i -=32)line(0,i,639,i);
line(639,479,639,30);
do {
    outportb(dataport,~RESETPIN);
    delay(100);
    outportb(dataport,0xff);
    while((inportb(statusport)&0x10)==0);
}

```



```

if(channels)j=datain2(data,pre,post,trigad,trigslope);
else j=datain(data,pre,post,trigad,trigslope);
if(j== -1){
    closegraph();
    printf("Data Underrun Error reduce acquisition rate - press any key\n");
    getch();
    return(-1);
}
for(i=639;i>=0;i--){
    buf[i]=(data[j]);
    j--;
    if(j<0)j=1023;
}
if(pass){
    c=getconfig()&0x30;
    if(c!=cold){
        cold=c;
        voltrange=(c==0x30?12.0:(c==0x10?6.0:2.4));
        sprintf(linebuf,"MODE %s X: %6.3f ms/div Y: %5.2f V/div",

smode?((smode==1)?"SING":"NORM"):"CONT",(float)period*40.0/1000.0,
        voltrange/4.0);
        w=textwidth(linebuf);
        setcolor(15);
        bar(319-w/2,18,319+w/2,textheight(linebuf)+18);
        outtextxy(319-w/2,18,linebuf);
    }
    bar(0,29,639,479);
    setcolor(gridcolor);
    for(position=0;position<640;position +=40)
        line(position,479,position,30);
    for(i=479;i>30;i -=32)line(0,i,639,i);
    line(639,479,639,30);
    for(position=0;position<640;position +=40){
        for(i=195;i>66;i -=16)putpixel(position,i,1);
    }
}
for(i=0;i<640;i++){
    if(channels&&(i&0x01))putpixel(i,478-buf[i],ch2color);
    else putpixel(i,286-buf[i],ch1color);
}
pass=1;
if(kbhit())break;
}while(smode!=1);
getch();
closegraph();
return(0);
}
display()
{
int gdriver=VGA,gmode=VGAHI,errorcode;
int j,w,i,position,pre,post,c,cold,trigad,pass=0;
char linebuf[128];
smode=1;
errorcode=registerbgidriver(EGAVGA_driver);
if(errorcode<0){
    printf("graphics error: %s\n",grapherrormsg(errorcode));
    printf("press any key\n");
    getch();
    return(0);
}

```

```

}
initgraph(&gdriver,&gmode,"");
errorcode=graphresult();
if(errorcode != grOk){
    printf("graphics error: %s\n",grapherrormsg(errorcode));
    printf("press any key\n");
    getch();
    return(0);
}
setcolor(15);
setwritemode(COPY_PUT);
cleardevice();
setfillstyle(EMPTY_FILL,0);
sprintf(linebuf,"EASY SCOPE");
w=textwidth(linebuf);
bar(319-w/2,5,319+w/2,textheight(linebuf)+5);
outtextxy(319-w/2,5,linebuf);
sprintf(linebuf,"MODE %s X: %6.3f ms/div Y: %5.2f V/div",
    smode?((smode==1)?"sing":"norm"):"cont", (float)period*40.0/1000.0,
    voltrange/4.0);
w=textwidth(linebuf);
outtextxy(319-w/2,18,linebuf);
setcolor(gridcolor);
for(position=0;position<640;position +=40)
    line(position,479,position,30);
for(i=479;i>30;i -=32)line(0,i,639,i);
line(639,479,639,30);
for(i=0;i<640;i++){
    if(channels&&(i&0x01))putpixel(i,478-buf[i],ch2color);
    else putpixel(i,286-buf[i],ch1color);
}
getch();
closegraph();
return(0);
}
rdconfig()
{
    FILE *fp;
    char line[128];
    int i;
    if((fp=fopen("EASYSCII.CFG","rt"))==NULL){
        clrscr();
        printf("Can NOT open config file - Using DEFAULTS - Press any key\n");
        dataport=0x278;
        statusport=dataport+1;
        getch();
        return(0);
    }
    fgets(line,128,fp);
    sscanf(line,"%x",&dataport);
    statusport=dataport+1;
    fgets(line,128,fp);
    sscanf(line,"%d",&ch1color);
    fgets(line,128,fp);
    sscanf(line,"%d",&ch2color);
    fgets(line,128,fp);
    sscanf(line,"%d",&gridcolor);
    fclose(fp);
}
info()

```

```

{
clrscr();
printf("\n\n\n\t\tEASY SCOPE\n");
printf("\n\t\tA&C 2000\n");
getch();
}
int configure()
{
int cal,c;
FILE *fp;
printf("Select port LPT1 (378) or LPT2 (278) or LPT3 (3BC) (1/2/3) :");
do{
    c=getch();
}while((c!='1') && (c!='2') && (c!=3));
if(c=='1')dataport=0x378;
else if(c=='2')dataport=0x278;
else dataport=0x3bc;
printf("\nSelect channel 1 color (1-15) :");
scanf("%d",&ch1color);
printf("Select channel 2 color (1-15) :");
scanf("%d",&ch2color);
printf("Select grid color (1-15) :");
scanf("%d",&gridcolor);
if((fp=fopen("EASYSCII.CFG","wt"))==NULL){
    clrscr();
    printf("CAN NOT OPEN CONFIG FILE - PRESS ANY KEY\n");
    getch();
    return(0);
}
fprintf(fp,"%x\n",dataport);
fprintf(fp,"%d\n",ch1color);
fprintf(fp,"%d\n",ch2color);
fprintf(fp,"%d\n",gridcolor);
fclose(fp);
statusport=dataport+1;
easyrev=(getconfig())>>6)&0x03;
printf("complete - press any key\n");
getch();
}
file()
{
int c,i;
FILE *fp;
char file[128],dummy[496];
clrscr();
for(i=0;i<496;i++)dummy[i]=0;
printf("\n\t\tRGB LOGIC ANALYZER FILE MENU\n");
printf("\n\t\tF1 - SAVE DATA FILE\n");
printf("\n\t\tF2 - LOAD DATA FILE\n\n");
printf("\n\t\tF10 - CANCEL\n\n");
c=bioskey(0);
switch(c){
    case F1: printf("\t\tenter file name : ");
        scanf("%s",file);
        if((fp=fopen(file,"wb"))==NULL){
            printf("CAN NOT OPEN FILE - PRESS ANY KEY");
            getch();
            return(0);
        }
        fwrite(&masterclock,sizeof(masterclock),1,fp);

```

```
        fwrite(&clockrate,sizeof(clockrate),1,fp);
        fwrite(&trigger,sizeof(trigger),1,fp);
        fwrite(&dontcare,sizeof(dontcare),1,fp);
        fwrite(&triglenth,sizeof(triglenth),1,fp);
        fwrite(&first,sizeof(first),1,fp);
        fwrite(&last,sizeof(last),1,fp);
        fwrite(dummy,sizeof(dummy[0]),496,fp);
        fwrite(dbuf,sizeof(dbuf[0]),8192,fp);
        fclose(fp);
        break;
    case F2: printf("\t\t\tenter file name : ");
        scanf("%s",file);
        if((fp=fopen(file,"rb"))==NULL){
            printf("CAN NOT OPEN FILE - PRESS ANY KEY");
            getch();
            return(0);
        }
        fread(&masterclock,sizeof(masterclock),1,fp);
        fread(&clockrate,sizeof(clockrate),1,fp);
        fread(&trigger,sizeof(trigger),1,fp);
        fread(&dontcare,sizeof(dontcare),1,fp);
        fread(&triglenth,sizeof(triglenth),1,fp);
        fread(&first,sizeof(first),1,fp);
        fread(&last,sizeof(last),1,fp);
        fread(dummy,sizeof(dummy[0]),496,fp);
        fread(dbuf,sizeof(dbuf[0]),8192,fp);
        fclose(fp);
        break;
    case F10: return(0);
}
}
```

10. TRADUCTOR INTELIGENT PENTRU MĂSURAREA NIVELULUI

Tematica propusă în cadrul acestui capitol constă în cercetarea și proiectarea unui echipament pentru determinarea nivelului lichidelor depozitate în vase staționare la presiune joasă, denumit **NIVELMETRU CU ULTRASUNETE**.

Determinarea nivelului lichidului în vas se bazează pe măsurarea intervalului de timp între momentul emisiei unei unde ultrasonice dirijate perpendicular pe suprafața lichidului de către un senzor specializat amplasat la partea superioară a vasului, într-o poziție determinată, și momentul recepției de către acesta a undei reflectate. Dat fiind că temperatura mediului în care se propagă unda sonoră afectează timpul de propagare, se impune corecția acestuia. În acest scop se amplasează în vecinătatea senzorului ultrasonic un senzor de temperatură.

Scopul echipamentului constă în determinarea nivelului lichidului în vas și a temperaturii în punctul de amplasare a senzorului ultrasonic, putând fi utilizat în aplicații industriale care necesită determinarea nivelului lichidului, în limitele condițiilor de lucru impuse de traductorul ultrasonic utilizat.

Caracteristicile echipamentului de măsurare a nivelului includ:

- măsurarea și afișarea nivelului lichidului din vas și a temperaturii în punctul de amplasare a senzorului;
- semnalizarea (alarmare) la atingerea unor mărimi programate: nivel maxim și nivel minim;
- posibilitatea de conectare a echipamentului la un calculator ierarhic superior, printr-o legătură serială **RS-232**, în vederea unor prelucrări ulterioare a informațiilor prelevate din proces;
- furnizarea unui semnal unificat de curent proporțional cu nivelul lichidului în vas.

Parametrii tehnici ai echipamentului de măsurare sunt:

- domeniul de măsurare a nivelului: (800...6000)mm;
- precizia de măsurare a nivelului: ± 4 mm;
- rezoluția de măsurare a nivelului: ± 1 mm;
- precizia de măsurare a temperaturii: $\pm 1^{\circ}\text{C}$;
- temperatura mediului de lucru: $(-20...+100)^{\circ}\text{C}$;
- temperatura de operare: $(-20...+70)^{\circ}\text{C}$.

Noile aplicații în domeniul automatizărilor industriale sunt definite de un grad înalt de inteligență, ceea ce permite adoptarea de noi algoritmi de funcționare, atât în prelucrarea și transmiterea informațiilor, cât și în conducerea proceselor.

Tema propusă se încadrează în eforturile de introducere în producția de serie de noi tipuri de traductoare inteligente, cu referire la măsurarea nivelului.

Performanțele impuse prin temă, cât și soluțiile tehnice adoptate justifică încadrarea nivelmetrului cu ultrasunete în clasa traductoarelor inteligente. Gradul de inteligență este determinat de funcțiile traductorului și anume:

1. Programarea regimurilor de funcționare:
 - domeniul de măsurare a nivelului;
 - limitele de alarmare, superioară și inferioară, a valorii nivelului;
 - valoarea zonei de insensibilitate la semnalizarea ieșirii din limite;
2. Protecție la accesul neautorizat pentru programarea regimurilor de funcționare prin utilizarea unei parole de acces;
3. Utilizarea unei tastaturi funcționale pentru programarea regimurilor de funcționare și pentru introducerea parolei de acces;
4. Afișarea pe un display alfanumeric cu cristale lichide a valorilor programate și a parametrilor măsurați;
5. Asigurarea funcționării în clasa de exactitate impusă, în condițiile modificării valorii temperaturii mediului ambiant;
6. Transmiterea a două comenzi de alarmare, la atingerea unor valori critice ale nivelului lichidului în vas;
7. Transmiterea unui semnal unificat de curent proporțional cu nivelul lichidului;
8. Transmiterea pe o linie serială a datelor prelevate din proces către un calculator ierarhic superior, ceea ce este necesar în sistemele de conducere sau monitorizare a proceselor tehnologice, în monitorizarea unor sisteme în ecologie și în cercetări pe stații pilot.

10.1 PREZENTAREA HARDWARE A TRADUCTORULUI INTELIGENT DE NIVEL

Schema funcțională a traductorului inteligent de nivel cuprinde următoarele module.

- Unitatea Centrală de Prelucrare. Unitatea Centrală de Prelucrare permite programarea prin intermediul unei tastaturi, memorarea într-o memorie nevolatilă și vizualizarea pe un afișaj cu cristale lichide a regimurilor de funcționare alese de către operatorul uman după confirmarea unei parole, efectuarea calculelor necesare îndeplinirii funcțiilor principale ale aparatului și pentru asigurarea comunicației cu un calculator ierarhic superior;
- Celula de măsurare a nivelului. Celula de măsurare a nivelului, referită în continuare ca senzor de ultrasunete, are rolul de a emite un fascicol de ultrasunete și de a recepționa unda reflectată. Comanda emisiei se

face de către unitatea centrală de prelucrare, care primește de la celula de măsurare și semnalul determinat de către unda reflectată;

- Senzorul de temperatură. Senzorul de temperatură servește pentru obținerea unui semnal privind valoarea temperaturii mediului ambiant, valoare care este utilizată în calcule pentru corecția cu temperatura a informației de nivel;
- Interfața serială de comunicații. Interfața serială de comunicație permite transmiterea informațiilor rezultate din procesul de măsurare a nivelului și temperaturii, la un calculator superior ierarhic, utilizat pentru conducerea sau monitorizarea unui proces tehnologic;
- Modulul de alarmare / semnalizare. Modulul de alarmare / semnalizare transmite în exterior două comenzi asociate cu atingerea unor valori critice, programabile, a nivelului lichidului în vas: Limita Superioară și Limita Inferioară;
- Modulul pentru semnal unificat. Modulul pentru semnal unificat transmite în exterior un semnal unificat de curent, proporțional cu nivelul lichidului în vas;
- Modul pentru tensiuni de alimentare.

Traductorul de nivel este compatibil cu aparatura din sistemele unificate de automatizare, prin furnizarea unui semnal unificat standardizat și prin utilizarea standardului de comunicație pe linii seriale **RS-232**.

Modulul surselor de alimentare conține:

SURSA DE ALIMENTARE “+5V”:

Utilizare:

- alimentarea unității centrale de prelucrare;
- alimentarea unității de afișare alfanumerică;
- alimentarea tastaturii;
- alimentarea circuitului de intrare a optocuplorului;

SURSA DE ALIMENTARE “+24V”:

Utilizare:

- alimentarea senzorului de nivel cu ultrasunete;
- alimentarea traductorului de nivel (daca se utilizează);
- alimentarea traductorului de temperatură;
- Alimentarea circuitelor de alarmare.

SURSELE DE ALIMENTARE “+15V” și “-15V”

Utilizare:

- alimentarea generatorului de curent (4...20)mA;

este și comanda de start pentru contorul de evenimente CT1 prin semnalul CT0I.

Circuitul de recepție a semnalului reflectat este organizat în jurul unui tranzistor în montaj EC. Semnalul CT1I, cules în colectorul tranzistorului, este semnalul de stop pentru contorul de evenimente.

Generatorul pentru semnal unificat în curent

În vederea prelucrării exterioare a informației de nivel, este prevăzută și o ieșire pentru semnal unificat în curent (4...20)mA.

În acest scop se utilizează un generator de curent comandat de unul din canalele cu ieșire modulată în durată, prin semnalul PWM0.

Din considerente de protecție această secțiune este separată galvanic de restul circuitelor, prin utilizarea unei surse de alimentare dedicate și izolată optic pe calea circuitului de comandă.

Generatoarele pentru semnal unificat în curent este un convertor tensiune (0...5)V / curent (4...20)mA.

Tensiunea de comandă se obține, după filtrare, din semnalul PWM0 (un tren de impulsuri modulate în durată, cu frecvența de 10KHz), aplicate la intrarea convertorului prin intermediul unui optocuplor comandat la intrare prin intermediul unui tranzistor.

Circuitele de conversie curent / tensiune

Există două astfel de circuite. Acestea convertesc un semnal de curent unificat în domeniul (4...20)mA într-un semnal de tensiune în domeniul (1...5)V, în vederea prelucrării ulterioare în secțiunea analogică a unității de control la care se conectează prin semnalele ADC0 și respectiv ADC1.

Constructiv, aceste circuite de conversie simulează o rezistență echivalentă de 250Ω cu o precizie de 0,1%.

Primul circuit este rezervat conectării unui traductor de nivel cu ieșire în curent, ca soluție alternativă la utilizarea senzorului cu ieșire logică, iar cel de al doilea conectării unui traductor de temperatură. Ambele traductoare sunt alimentate de la sursa +24V.

Alimentarea convertorului analog / digital

Terminalele de alimentare sunt AVDD conectat la sursa +5V și AVSS conectat la masa sistemului.

Pentru obținerea tensiunilor de referință s-a utilizat un circuit specializat LM 336-5V, având o bună precizie și stabilitate termică.

Tensiunea de referință REF+ (+5V) se obține direct la bornele referinței, iar REF- (+1V) se obține pe cursorul unui semireglabil.

Circuite de alarmare

Sunt prevăzute două circuite de alarmare, activate la atingerea unor nivele

critice programabile.

Circuitele sunt controlate la intrare de semnalele AX9 și respectiv AX10 (portul de ieșire “output high”), care prin intermediul unor tranzistoare comandă două relele.

Ieșirea circuitului “Alarmare Superioară” este constituită de perechea de contacte N.I./N.D. a releului K1.

Ieșirea circuitului “Alarmare Inferioară” este constituită de perechea de contacte N.I./N.D. a releului K2.

Simulator pentru senzorul de ultrasunete

În lipsa senzorului ultrasonic, a fost utilizat un circuit de simulare a acestuia. Circuitul introduce o întârziere reglabilă în domeniul (0...45)ms, pe calea de la circuitul de comandă a senzorului (semnal AX8) la circuitul de recepție (semnal CT1I), simulând timpul parcurs de fascicolul ultrasonic de la emițător la suprafața țintă și înapoi.

Circuitul de întârziere se conectează în locul senzorului ultrasonic.

Primul etaj al circuitului, introduce o întârziere reglabilă cu ajutorul unui potențiomtru, în domeniul (0...45)ms. Acesta este un monostabil comandat de tranziția “1↓0” a impulsului de comandă (semnalul AX8 inversat).

Al doilea etaj este, de asemenea, un monostabil, comandat de ieșirea “Q” a primului monostabil la tranziția “1↓0” a acesteia. Rezistența variabilă R6 permite ajustarea duratei impulsului cules la ieșirea negată a monostabilului, care comandă tranzistorul (în colectorul căruia se culege semnalul CT1I) corespunzător unde reflectate.

10.2 PROGRAMUL DE APLICAȚII NIV.ASM

Faza de dezvoltare a programului de aplicație destinat determinării înălțimii coloanei de lichid dintr-un vas, folosind un senzor cu ultrasunete conectat la o unitate centrală de prelucrare cu microcontroller 80C552, a constat din:

- scrierea programului, în limbajul de asamblare al familiei de microcontroller-e 8051, cu ajutorul unui editor ASCII de texte, ca de exemplu **EDIT.COM** din sistemul de operare **MS-DOS**, **NCEDIT.EXE** din **NORTON COMMANDER** sau **NOTEPAD.EXE** din sistemul de operare **WINDOWS** (se impune utilizarea unui editor ASCII pentru a nu se introduce caractere speciale de control în sursa programului). În sursa programului de aplicație sunt declarate în mod explicit resursele suplimentare ale microcontroller-ului 80C552 în raport cu 8051, într-o secțiune declarativă la început (aceasta secțiune conține linii de cod în limbaj de asamblare de tip

ADCON equ 0C5H). Programul sursă de aplicație are numele **NIV.ASM**;

- asamblarea sursei programului de aplicație, cu ajutorul asamblorului **A51.EXE**, specializat pentru familia de microcontroller-e 8051, folosind sintaxa:

A51.EXE NIV.ASM

În urma executării asamblării programului sursă de aplicații, asamblorul **A51.EXE** generează două fișiere: **NIV.LST**, un fișier listă ce este destinat localizării eventualelor erori rezultate în urma procesului de asamblare și **NIV.OBJ**, un fișier de tip obiect, care va fi folosit în continuare pentru obținerea formatului executabil;

- obținerea formatului executabil, de tip **INTEL HEX**, pe baza fișierului de tip **OBJ**, folosind programul **OHS51.EXE** cu sintaxa:

OHS51.EXE NIV.OBJ

Este generat fișierul executabil **NIV.HEX** ce constituie programul propriu-zis de aplicație. Ca orice fișier format **HEX**, acesta începe cu un *header* ce conține numărul de octeți ai programului, adresa la care este organizat programul (specificată prin directiva **ORG** în prima linie), urmat de corpul programului și se încheie cu o sumă de control.

- rularea programului **NIV.HEX** presupune rularea pe un calculator gazdă, de tip PC, a programului monitor al unității centrale cu microcontroller 80C52, denumit **MT.EXE**, care are în primul rând rolul de a stabili comunicația serială între sistemul cu microcontroller și calculatorul de tip PC. Sintaxa utilizată este următoarea:

MT.EXE X

în care **X=1, 2, 3, 4** specifică indicativul portului serial utilizat pentru comunicație (1 specifică portul serial COM1, 2 specifică portul serial COM2, 3 specifică portul serial COM3, 4 specifică portul serial COM4). Dacă parametrul **X** lipsește, este utilizat în mod implicit portul serial COM1.

Stabilirea comunicației seriale între cele două sisteme (pe viteza de 9600 bauds, folosind cuvinte de date cu lungimea de 8 biți, fără paritate, cu un bit de STOP și protocol XON-XOFF) este indicată prin apariția pe display-ul sistemului de calcul a unui mesaj, urmat de prompter-ul de monitor (caracterul #). În acest stadiu poate fi transferat către sistemul cu microcontroller programul de aplicație, **NIV.HEX**, folosind comanda de up-load **F2** și indicând numele programului de aplicație. Urmează procesul de up-load, indicat prin apariția pe display-ul calculatorului a unei succesiuni de linii, de lungime constantă cu excepția ultimei, ce încep cu ”#:_”, și sunt

alcătuite din câte 16 octeți în format *hexa* (date) plus un octet sumă de control. La terminarea transferului de date pe legatura serială, pe display-ul calculatorului apare prompter-ul de monitor (caracterul #). Lansarea în execuție a programului se execută cu comanda de monitor **GO ADR**, în care **ADR** reprezintă adresa la care este organizat programul în memoria de date a microcontroller-ului, specificată în prima linie a acestuia prin directiva **ORG**. În cazul de față, sintaxa este următoarea:

G 8000

În faza ulterioară punerii la punct a programului de aplicație, acesta va fi înscris în memoria de program a unității centrale de prelucrare cu microcontroller.

Interconectarea sistemului de calcul cu unitatea centrală de prelucrare cu microcontroller este realizată prin intermediul unui cablu serial cu trei fire.

10.2.1 DESCRIEREA PROGRAMULUI DE APLICATIE NIV.ASM

Programul sursă de aplicație, cu denumirea **NIV.ASM**, este organizat modular, sub forma unui corp principal al aplicației și o colecție de subrutine. Acestea reprezintă secvențe de linii de cod care au o frecvență de apariție mare în cadrul aplicației. O astfel de subrutină este apelată cu instrucțiunea **ACALL subrutina** și se încheie cu instrucțiunea **RET** pentru a se reveni la poziția imediat ulterioară apelării acesteia în cadrul programului principal.

În fig. 10.2 este prezentată organigrama generală a programului **NIV.ASM**.

10.2.1.1 SECȚIUNEA DE ÎNȚIALIZARE ȘI DECLARATIVĂ

Secvența de linii de cod în limbaj de asamblare ce reprezintă programul principal începe cu instrucțiunea **ORG adresa**, specificând adresa din memoria de date a microcontroller-ului la care începe programul.

În continuare urmează declararea tuturor resurselor specifice ale microcontroller-ului 80C552 (care nu există în cadrul microcontroller-ului 8051) utilizate în cadrul aplicației, cum ar fi:

- **ADCON** - adresa registrului de control al convertorului analog-digital;

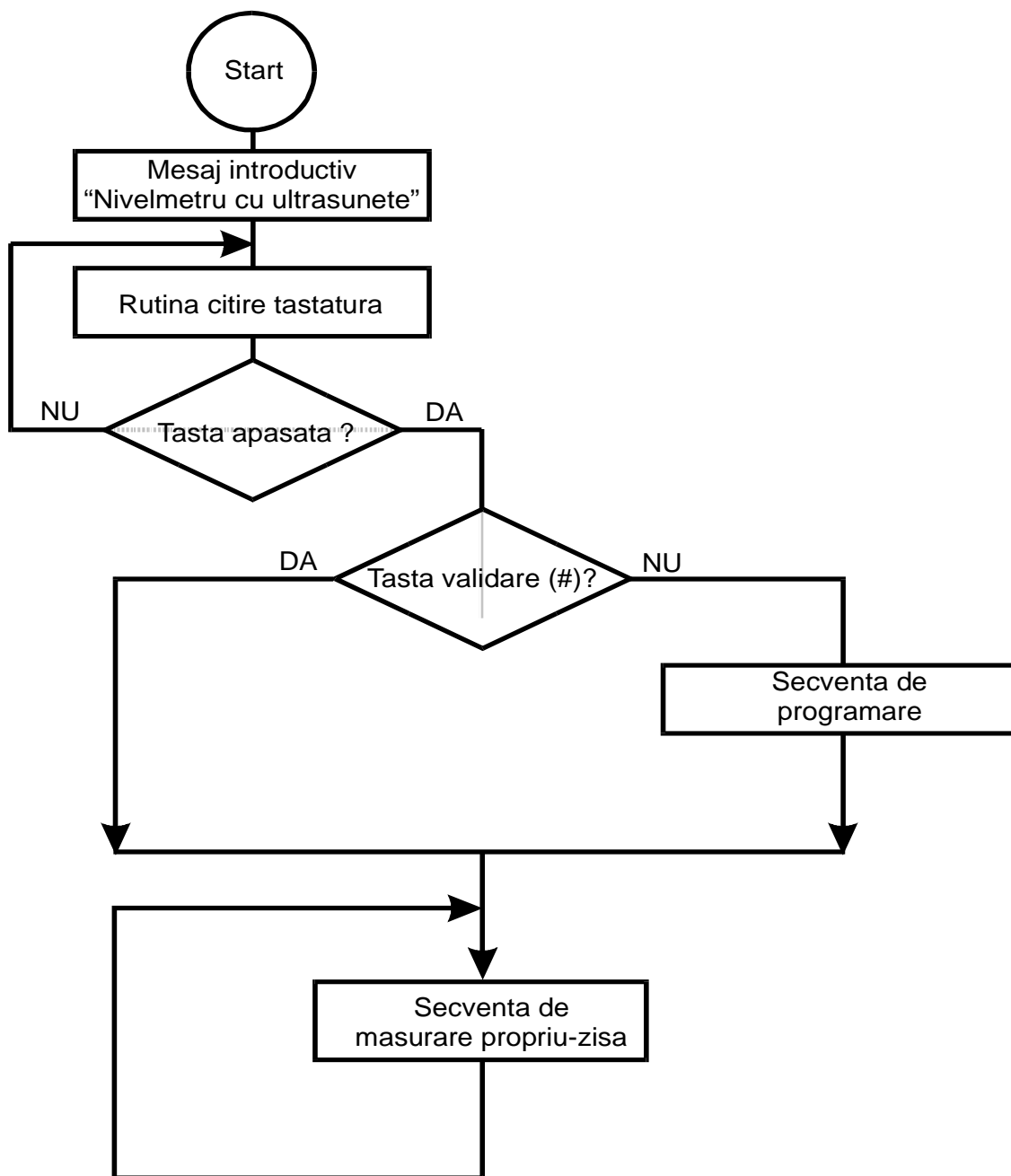


Fig. 10.2 Organigrama generală a programului NIV.ASM.

- **ADCH** - adresa registrului care conține octetul cel mai semnificativ al rezultatului conversiei analog-digitale;
- **PWMP** - adresa registrului de setare a frecvenței de ieșire comun ambelor canale modulate în durată;
- **PWM0, PWM1** - adresele registrelor de setare a factorului de umplere pentru cele două canale cu ieșiri modulate în durată;
- **CTH0, CTL0** - adresele registrelor superior, respectiv inferior de captare a evenimentelor CT0;

- **CTH1, CTL1** - adresele registrelor superior, respectiv inferior de captare a evenimentelor CT1;
- **TMH2, TML2** - adresa octeților superior, respectiv inferior ale timer-ului T2;
- **TM2CON** - adresa registrului de control al timer-ului T2;
- **CTCON** - adresa registrului de control al registrelor de captare a evenimentelor;
- **IEN1** - adresa registrului de activare a întreruperilor asociate registrelor de captare a evenimentelor și de comparare.

Urmează o secțiune introductivă de linii de cod, care inițializează resursele unității centrale cu microcontroller (afișajul cu cristale lichide, canalele de ieșire modulate în durată, portul de ieșire de comenzi), afișează un mesaj introductiv, afișează un al doilea mesaj destinat validării operării echipamentului de măsurare sau de parcurgere a secțiunii de programare a funcționării. Ramificarea în cadrul programului este realizată prin citirea tastaturii: dacă este acționată tasta de validare (#), atunci se execută un salt în program până la secțiunea de măsurare (funcționare normală a echipamentului); dacă este acționată tasta de programare (*) sau oricare altă tastă, se intră în secțiunea de programare a funcționării echipamentului de măsurare.

10.2.1.2 SECȚIUNEA DE PROGRAMARE A PARAMETRILOR DE FUNCȚIONARE

În cadrul secțiunii de programare a funcționării echipamentului de măsurare, accesul este limitat prin validarea unei parole prestabilite: în urma afișării unui mesaj de validare a parolei, se așteaptă acționarea temporizată a două taste, având drept rezultat un octet împachetat BCD. Acționarea unei taste se soldează cu afișarea a câte unui caracter “*” pe cea de-a doua linie a afișajului LCD. Octetul împachetat BCD este comparat cu parola prestabilită, proprie echipamentului de măsurare. În cazul în care cei doi octeți nu coincid, se afișează un mesaj de incorectitudine a parolei, iar programul revine în buclă la secțiunea de validare a parolei. În cazul coincidenței celor doi octeți, se afișează un mesaj de corectitudine a parolei, iar utilizatorul are acces la secțiunea efectivă de programare a parametrilor de măsurare.

În prima fază, este programată înălțimea vasului în care se face determinarea înălțimii coloanei de lichid (H), prin intermediul tastaturii. Se tastează patru cifre de la tastatură, ceea ce conduce la obținerea a doi octeți împachetați BCD. În paralel se execută și afișarea valorii programate. Cei doi octeți împachetați BCD, reprezentând înălțimea vasului, sunt depuși în memoria de date a unității centrale la adresele FF00H și FF01H.

În faza a doua, este programat nivelul superior de alarmare (NSA), prin

intermediul tastaturii. Se tastează patru cifre de la tastatură, ceea ce conduce la obținerea a doi octeți împachetați BCD. În paralel se execută și afișarea valorii programate. Cei doi octeți împachetați BCD, reprezentând nivelul superior de alarmare, sunt comparați cu înălțimea vasului, programată în pasul anterior și depuși în memoria de date a unității centrale la adresele FF02H și FF03H doar dacă rezultatul scăderii pe 16 biți (H - NSA) este pozitiv. Dacă nivelul superior de alarmare programat este mai mare decât înălțimea vasului, programul revine în buclă la secțiunea de programare a nivelului superior de alarmare.

În faza a treia, este programat nivelul inferior de alarmare (NIA), prin intermediul tastaturii. Se tastează patru cifre de la tastatură, ceea ce conduce la obținerea a doi octeți împachetați BCD. În paralel se execută și afișarea valorii programate. Cei doi octeți împachetați BCD, reprezentând nivelul inferior de alarmare, sunt comparați cu nivelul superior de alarmare, programat în pasul anterior și depuși în memoria de date a unității centrale la adresele FF04H și FF05H doar dacă rezultatul scăderii pe 16 biți (NSA - NIA) este pozitiv. Dacă nivelul inferior de alarmare programat este mai mare decât nivelul superior de alarmare, programul revine în buclă la secțiunea de programare a nivelului inferior de alarmare.

În cea de-a patra fază a secțiunii de programare a parametrilor de funcționare ai echipamentului este realizată calibrarea generatorului de curent comandat cu care este echipat sistemul. Această calibrare este realizată în două etape:

- în prima etapă, este programat factorul de umplere al ieșirii modulate în durată PWM0 la valoarea FFH, astfel încât la ieșire (care este negată) se obține un nivel zero (0V). Acest nivel este utilizat pentru calibrarea capătului inferior de scală a generatorului de curent la valoarea de 4 mA. Programul așteaptă în buclă acționarea unei taste; la acționare se trece la cea de-a doua etapă;
- în cea de-a doua etapă, este programat factorul de umplere al ieșirii modulate în durată PWM0 la valoarea 00H, astfel încât la ieșire (care este negată) se obține un nivel ridicat (+5V). Acest nivel este utilizat pentru calibrarea capătului superior de scală a generatorului de curent la valoarea de 20 mA. Programul așteaptă în buclă acționarea unei taste; acționarea oricărei taste încheie secțiunea de programare a parametrilor de funcționare ai echipamentului;
- trebuie menționat faptul că în cadrul celor două etape descrise anterior pentru calibrarea independentă a capetelor de scală ale generatorului de curent, poate fi efectuată și calibrarea circuitelor de intrare cu care sunt echipate două dintre cele opt intrări ale convertorului analog-digital, implementat în structura microcontroller-ului.

În fig. 10.3 este prezentată organigrama secțiunii de programare a parametrilor de funcționare a echipamentului nivelmetru cu ultrasunete.

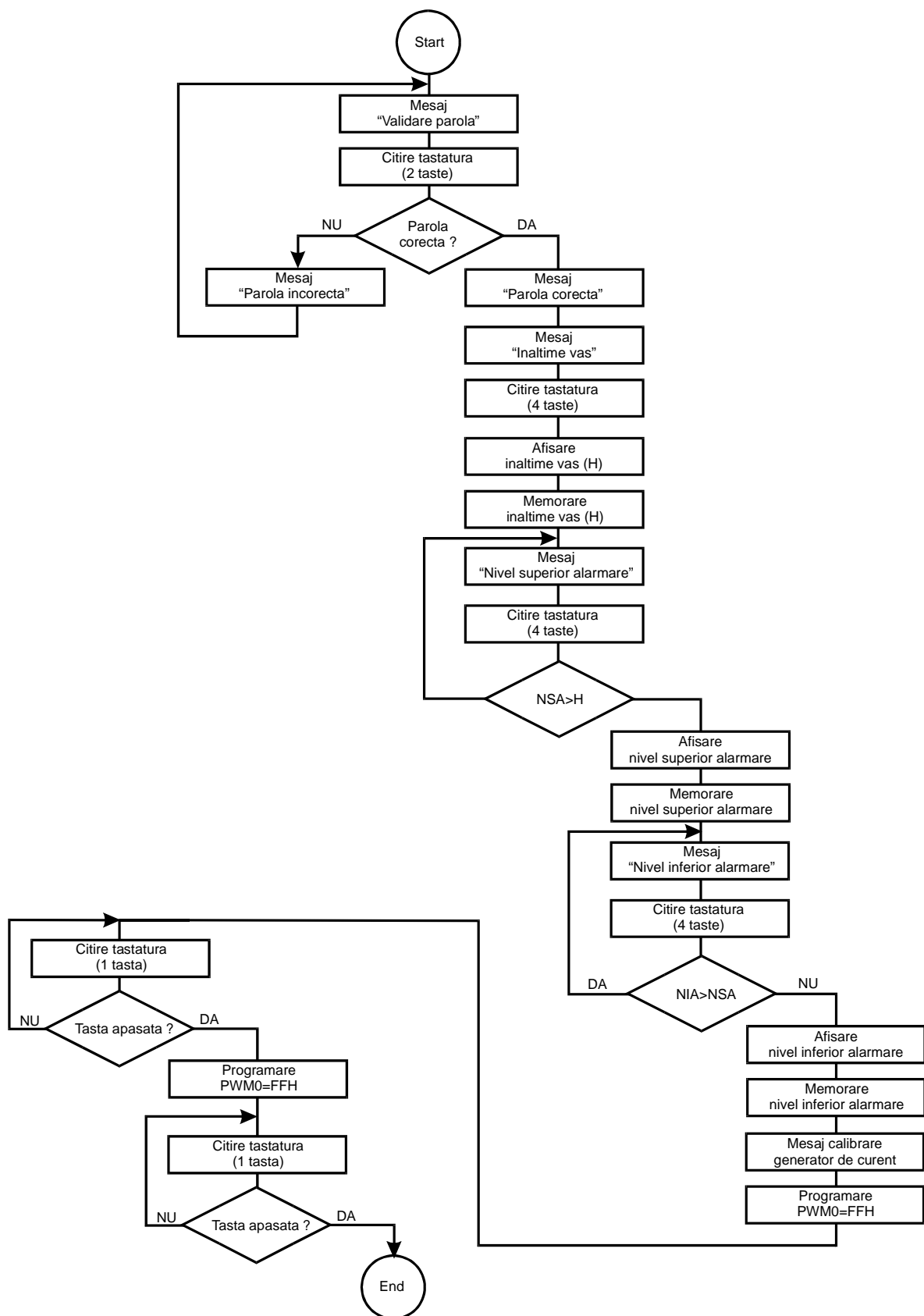


Fig. 10.3 Organigrama secțiunii de programare a parametrilor de funcționare.

10.2.1.3 SECȚIUNEA DE MĂSURARE PROPRIU-ZISĂ

În acest punct, cele două ramuri - de validare a operării normale a echipamentului, respectiv de programare a parametrilor de funcționare, converg și se intră în secțiunea de măsurare propriu-zisă.

Secțiunea de măsurare propriu-zisă începe cu afișarea unui mesaj de avertizare a faptului că s-a ajuns în secțiunea de măsurare, care este menținut circa 0,5 secunde. Se continuă cu inițializarea secțiunii de captare a evenimentelor:

- sunt dezactivate întreruperile provenite de la depășirea superioară pe 8/16 biți a conținutului timer-ului T2, de la registrele de captare a evenimentelor, de la circuitele de comparare a conținutului timer-ului T2 cu registrele special prevăzute în acest sens;
- sunt inițializate (încărcate cu 0) registrele CTL0, CTH0, CTL1, CTH1, TML2, TMH2;
- este programat registrul CTCON, astfel încât registrele de captare a evenimentelor CTI0 și CTI1 să fie activate de fronturile descrescătoare ale impulsurilor externe;
- este programat timer-ul T2, prin activarea acestuia, cu dezactivarea depășirii pe 8/16 biți, ceasul de numărare fiind constituit de un oscilator intern cu frecvența 1/12 din frecvența oscilatorului microcontroller-ului, urmat de un registru de divizare cu 8 a acestui ceas.

Este, în continuare, activat emițătorul senzorului cu ultrasunete prin bascularea nivel coborât / nivel ridicat a bitului AX8. Frontul crescător al semnalului AX8 determină captarea în registrul CTI0 a conținutului timer-ului T2 la acest moment de timp. După reflexia ultrasunetelor de suprafața lichidului, acestea sunt captate de senzor, care emite un semnal de același tip cu cel de activare. Frontul crescător al acestui semnal determină captarea conținutului timer-ului T2 în registrul CTI1. Se determină numărul de impulsuri captate pe durata emisie-recepție prin efectuarea diferenței pe 2 octeți între registrele CTI1 și CTI0. Această diferență este codificată, urmând să fie utilizată pentru determinarea înălțimii coloanei de lichid din vas, conform următoarei relații:

$$CTI1 - CTI0 \xrightarrow{\text{CODIFICARE}} N$$

$$\frac{N_{\text{MAX}} - N}{N_{\text{MAX}}} \cdot 6 = h \quad (10.1)$$

În această fază de elaborare a programului, codificarea este realizată prin extragerea biților 11÷8, respectiv 7÷4, ai diferenței dintre CTI1 și CTI0 și alcătuirea unui octet (N). Obținerea înălțimii coloanei de lichid din vas se obține conform ecuației de mai sus, în care N_{MAX} reprezintă valoarea octetului de cod

pentru înălțimea maximă a vasului, respectiv depinde de tipul de traductor ($N_{MAX} = 127$ pentru 6 m). Înălțimea coloanei de lichid este afișată după determinarea cifrelor BCD ale acesteia, utilizând un algoritm iterativ de împărțire la 10. Cele patru cifre BCD obținute prin acest algoritm sunt împachetate în doi octeți codificați BCD și memorate în memoria de date a microcontroller-ului la adresele FF006H și FF07H

Cuvântul de date care corespunde înălțimii coloanei de lichid, complementat, adică $2 \cdot (N_{MAX} - N)$, este folosit pentru programarea factorului de umplere al impulsurilor de la ieșirea modulată în durată PWM0, care comandă generatorul de curent.

Se verifică depășirea nivelurilor prestabilite de alarmare (NSA și NIA), comparând înălțimea coloanei de lichid din vas cu pragurile programate anterior. Se efectuează diferența dintre nivelul superior de alarmare (NSA) și înălțimea coloanei de lichid din vas, iar dacă diferența este negativă (setarea indicatorului CARRY) se poziționează pe "1" logic bitul AX9. Se efectuează diferența dintre înălțimea coloanei de lichid din vas și nivelul inferior de alarmare (NIA), iar dacă diferența este negativă (setarea indicatorului CARRY) se poziționează pe "1" logic bitul AX10. Acești biți ai portului de ieșire acționează releele corespunzătoare. Atât valorile nivelului superior, cât și inferior de alarmare, NSA și NIA, sunt citite din memoria de date a microcontroller-ului, de la adresele specificate anterior.

Urmează măsurarea temperaturii mediului în care este efectuată determinarea înălțimii coloanei de lichid. Ieșirea în curent, de tip unificat (4...20) mA, a sondei de temperatură este aplicată unui convertor curent-tensiune de tip sunt. Această tensiune este aplicată intrării ADCIN0 a convertorului analog-digital. Conversia este demarată prin software, programând în registrul ADCON bitul ADCCON.3 la "1" logic, iar biții ADCON.2, ADCON.1, ADCON.0 conform adresei canalului de intrare utilizat (în această situație, cei trei biți sunt programați la "0" logic). Se așteaptă sfârșitul conversiei analog-digitale, lucru sesizat prin poziționarea în "1" logic a bitului ADCON.4 (se execută de fapt o verificare în buclă a acestui bit, prin intermediul acumulatorului). Urmează interpretarea rezultatului și afișarea acestuia. Rezultatul conversiei este transferat în acumulator, este împărțit la 2 pentru a se realiza conversia în domeniul specificat de temperatură, apoi se obțin cifrele temperaturii printr-un algoritm de același tip ca în cazul determinării înălțimii coloanei de lichid.

Programul execută un salt necondiționat la secvența de inițializare a timer-ului T2, pentru reluarea în buclă infinită a secțiunii de măsurare propriu-zisă. Ieșirea din program se poate face doar prin reset-area unității centrale cu microcontroller. Poate fi prevăzută, însă, facilitatea de citire a tastaturii după fiecare secvență de măsurare și în cazul acționării tastei de validare (#) să se încheie forțat execuția acestuia.

Rutinele scrise și utilizate în cadrul acestei aplicații sunt:

- subrutina de citire a unei cifre zecimale de la o tastatură matricială cu trei linii și patru coloane. Rezultatul citirii sub formă BCD se află în registrul R1;
- subrutina de scriere la un port de ieșire și de citire de la un port de intrare, utilizată în cadrul subrutinei de gestionare a tastaturii. Informațiile sunt vehiculate prin intermediul acumulatorului;
- subrutina de scriere șir de caractere la afișajul cu cristale lichide LCD;
- subrutina de întârziere cu 0,5 secunde ($0,5\text{sec} = (7 \text{ instr. NOP} \times 12 \text{ perioade ceas/NOP} \times 256 \times 256)/11,059\text{MHz}$);
- subrutina de conversie hexa-ASCII, utilizată pentru afișarea la LCD. Sursa și destinația sunt constituite de acumulator;
- subrutina de transmisie la LCD a unui caracter ASCII conținut în registrul R2;
subrutina de inițializare LCD. Această rutină trimite la LCD comanda #38H = 0011 1000B. Conform datelor de catalog, această comandă setează următorii parametri ai afișajului: bitul 5 - definește această comandă (function set); bitul 4 (DL-Data Length) = 1 setează dialogul pe 8 biți între procesor și LCD; bitul 3 (N-Number of display lines) = 1 setează numărul liniilor afișajului ca fiind $2^N=2$; bitul 2 (F-Character Font) = 0 setează forma caracterului ca fiind 5*7 puncte;
- subrutina de ștergere a afișajului cu cristale lichide și setare a modului de lucru;
- subrutina ce testează dacă este sau nu posibilă transmiterea unui nou caracter către LCD, pe baza informațiilor furnizate de rutina RDCMD. Se testează bitul 7 (BF-Busy Flag) al registrului de stare al LCD. Dacă BF=1, înseamnă că LCD efectuează o operație internă și, deci, nu poate accepta un nou caracter;
- subrutina RDCMD selectează afișajul și citește din registrul de stare al LCD starea curentă a driver-elor acestuia, informație care este depusă în acumulator;
- rutina WRCMD scrie o comandă la driver-ele de afișaj, în urma verificării stării acestora. Se selectează afișajul LCD și comanda conținută de registrul R2 este transferată driver-elor din LCD;
- subrutina WRDAT este identică cu WRCMD, dar este utilizată pentru scrierea la driver-ele LCD a codului caracterelor ce urmează a fi afișate;
- subrutina DELAY, ce realizează o întârziere cu 80 microsecunde, pentru funcționarea internă a LCD.

În fig. 10.4 este prezentată organigrama secțiunii de măsurare propriu-zisă a înălțimii coloanei de lichid din vas și a temperaturii. În fig. 10.5 este prezentată organigrama rutinei de gestionare a tastaturii matriciale cu trei linii și patru coloane.

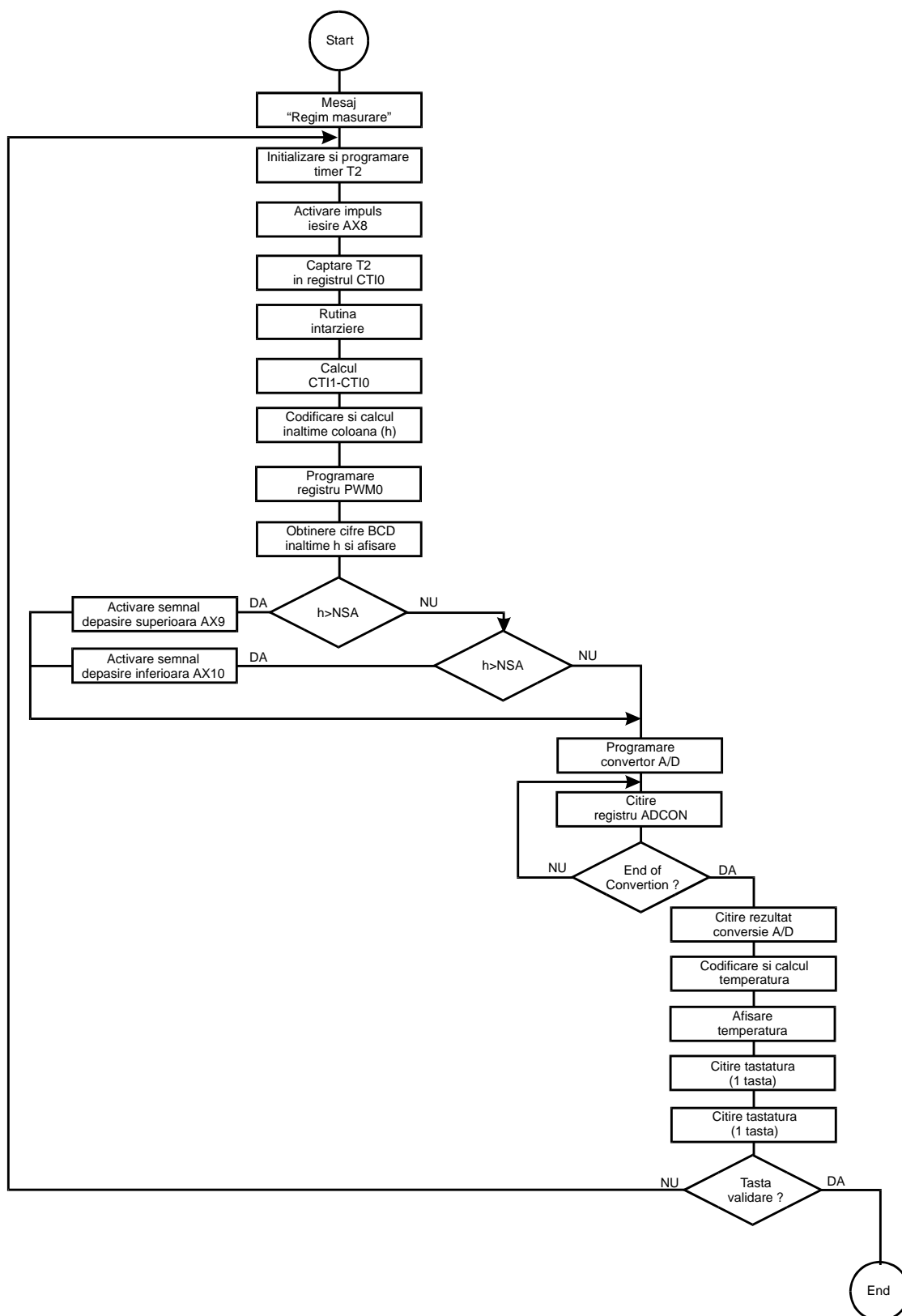


Fig. 10.4 Organigrama secțiunii de măsurare propriu-zisă.

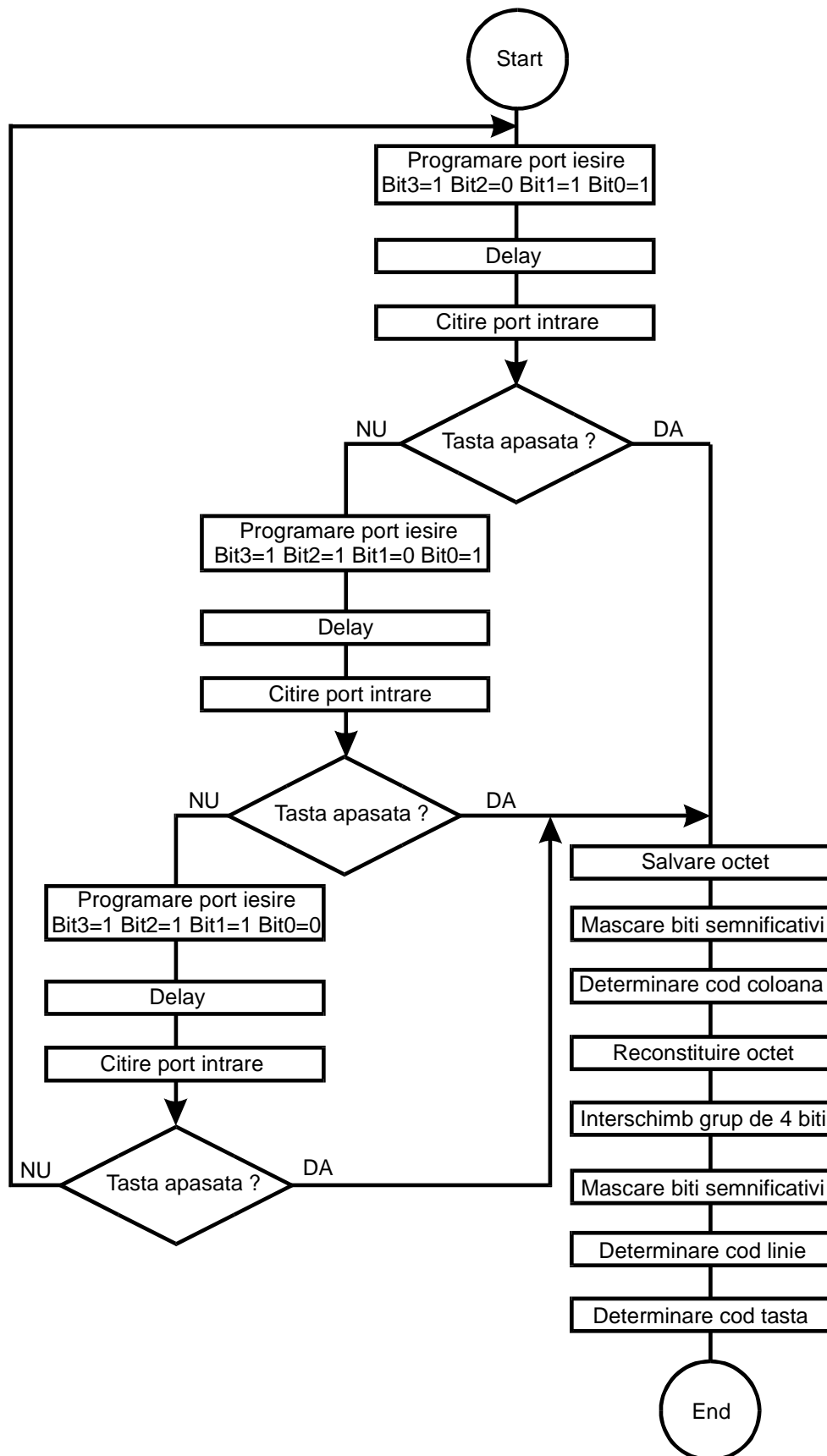


Fig. 10.5 Organigrama rutinei de gestionare a tastaturii.

10.3 PROGRAMUL DE APLICAȚII NIVOK.ASM

A doua variantă a programului sursă de aplicație, cu denumirea **NIVOK.ASM**, este organizat modular, sub forma unui corp principal al aplicației și o colecție de subrutine.

10.3.1 SECȚIUNEA DE ÎNȚIALIZARE ȘI DECLARATIVĂ

Secvența de linii de cod în limbaj de asamblare ce reprezintă programul principal începe cu instrucțiunea **ORG adresa**, specificând adresa din memoria de date a microcontroller-ului la care începe programul.

În continuare urmează declararea tuturor resurselor specifice ale microcontroller-ului 80C552 (care nu există în cadrul microcontroller-ului 8051) utilizate în cadrul aplicației.

Urmează o secțiune introductivă de linii de cod, care inițializează resursele unității centrale cu microcontroller (afișajul cu cristale lichide, canalele de ieșire modulate în durată, portul de ieșire de comenzi), afișează un mesaj introductiv, afișează un al doilea mesaj destinat validării operării echipamentului de măsurare sau de parcurgere a secțiunii de programare a funcționării. Ramificarea în cadrul programului este realizată prin citirea tastaturii: dacă este acționată tasta de validare (ENTER), atunci se execută un salt în program până la secțiunea de măsurare (funcționare normală a echipamentului); dacă este acționată tasta de programare (MODE), se intră în secțiunea de programare a funcționării echipamentului de măsurare.

10.3.2 SECȚIUNEA DE CALIBRARE A GENERATORULUI DE CURENT

Se intră într-un meniu (opțional) destinat calibrării generatorului de curent, care afișează un ecran indicând operația curentă. Calibrarea generatorului de curent este validată prin acționarea tastei de programare (MODE) sau poate fi “ocolită” prin acționarea tastei de validare (ENTER). Calibrarea generatorului de curent al sistemului necesită reglarea manuală a capetelor de scală (reglajele sunt independente), efectuându-se, de obicei, doar în faza de realizare a echipamentului de măsurare și comportând două etape:

- în prima etapă, este programat factorul de umplere al ieșirii modulate în durată PWM0 la valoarea 00H, astfel încât la ieșire (care este directă) se obține un nivel zero (0V) și se afișează un ecran care indică

operația curentă. Acest nivel este utilizat pentru calibrarea capătului inferior de scală a generatorului de curent la valoarea de 4 mA. Programul așteaptă în buclă acționarea tastei de validare (ENTER); la acționarea ei se trece la cea de-a doua etapă;

- în cea de-a doua etapă, este programat factorul de umplere al ieșirii modulate în durată PWM0 la valoarea FFH, astfel încât la ieșire (care este directă) se obține un nivel ridicat (+5V) și se afișează un ecran care indică operația curentă. Acest nivel este utilizat pentru calibrarea capătului superior de scală a generatorului de curent la valoarea de 20 mA. Programul așteaptă în buclă acționarea tastei de validare (ENTER); acționarea ei încheie secțiunea de calibrare a generatorului de curent;
- trebuie menționat faptul că în cadrul celor două etape descrise anterior pentru calibrarea independentă a capetelor de scală ale generatorului de curent, poate fi efectuată și calibrarea circuitelor de intrare (convertoare curent-tensiune) cu care sunt echipate două dintre cele opt intrări ale convertorului analog-digital, implementat în structura microcontroller-ului.

Se afișează un mesaj destinat selecției fie a operării echipamentului de măsurare, fie de parcurgere a secțiunii de programare a funcționării. Ramificarea în cadrul programului este realizată prin citirea tastaturii: dacă este acționată tasta de validare (ENTER), atunci se execută un salt în program până la secțiunea de măsurare (funcționare normală a echipamentului); dacă este acționată tasta de programare (MODE), se intră în secțiunea de programare a funcționării echipamentului de măsurare.

10.3.3 SECȚIUNEA DE PROGRAMARE A PARAMETRILOR DE FUNCȚIONARE

În cadrul secțiunii de programare a funcționării echipamentului de măsurare, accesul este limitat prin validarea unei parole prestabilite: în urma afișării unui mesaj ce indică acțiunea de introducere a parolei, se acționează tastele UP / DOWN pentru configurarea primei cifre a parolei, se validează prima cifră prin acționarea tastei de validare, se poziționează a doua cifră prin acționarea tastelor UP / DOWN, urmată de validarea acesteia și, implicit a parolei. Configurarea celor două cifre ale parolei au drept rezultat un octet împachetat BCD. Octetul împachetat BCD este comparat cu parola prestabilită, proprie echipamentului de măsurare. În cazul în care cei doi octeți nu coincid, se afișează un mesaj de incorectitudine a parolei, iar programul revine în buclă la secțiunea de validare a parolei. În cazul coincidenței celor doi octeți, se afișează un mesaj de corectitudine a parolei, iar utilizatorul are acces la secțiunea

efectivă de programare a parametrilor de măsurare.

Aceasta variantă de aplicație utilizează valori prestabilite ale parametrilor globali utilizați de către echipamentul de măsurare: înălțimea vasului, nivelul superior de alarmare, respectiv nivelul inferior de alarmare, pentru a se reduce timpul de programare. Valoarea implicită a înălțimii vasului este 6.000 m (conform specificației de proiectare), iar valorile implicite ale nivelurilor de alarmare reprezintă două treimi, respectiv o treime din înălțimea vasului (6.000 m), adică $NSA=4.000m$ iar $NIA=2.000m$.

În prima fază, este programată înălțimea vasului în care se face determinarea înălțimii coloanei de lichid (H), prin intermediul tastaturii. Se afișează meniul de programare a înălțimii vasului și valoarea implicită a acesteia. Cursorul este poziționat pe prima cifră a înălțimii, urmând modificarea acesteia, folosind tastele UP / DOWN sau validarea ei, folosind tasta de validare. Validarea cifrei determină deplasarea cursorului pe cea de-a doua cifră, urmată de modificarea sau validarea ei. Operațiile se repetă pentru ultimele două cifre. Validarea ultimei cifre înseamnă și validarea înălțimii vasului. Se afișează un mesaj de confirmare și se așteaptă acționarea tastei de programare pentru reprogramarea înălțimii sau tasta de validare pentru a se continua. Validarea celor patru cifre conduce la obținerea a doi octeți împachetați BCD, reprezentând înălțimea vasului, ce sunt depuși în memoria de date a unității centrale la adresele FF00H și FF01H. În continuare, cei doi octeți împachetați BCD sub forma MSZU (mii-sute-zeci-unități de milimetru), reprezentând înălțimea vasului, sunt convertiți în binar conform algoritmului:

$$(MSZU)_H \rightarrow (M)_H \times 1000 + (S)_H \times 100 + (Z)_H \times 10 + (U)_H \quad (10.2)$$

și depuși în memorie pentru calculele ulterioare.

În faza a doua, este programat nivelul superior de alarmare (NSA), prin intermediul tastaturii, folosind un algoritm identic. Cei doi octeți împachetați BCD reprezentând nivelul superior de alarmare sunt comparați cu înălțimea vasului, programată în pasul anterior și depuși în memoria de date a unității centrale la adresele FF02H și FF03H doar dacă rezultatul scăderii pe 16 biți (H - NSA) este pozitiv. Dacă nivelul superior de alarmare programat este mai mare decât înălțimea vasului, programul revine în buclă la secțiunea de programare a nivelului superior de alarmare. În continuare, cei doi octeți împachetați BCD sub forma MSZU (mii-sute-zeci-unități de milimetru), reprezentând valoarea nivelului superior de alarmare, sunt convertiți în binar conform algoritmului:

$$(MSZU)_{NSA} \rightarrow (M)_{NSA} \times 1000 + (S)_{NSA} \times 100 + (Z)_{NSA} \times 10 + (U)_{NSA} \quad (10.3)$$

și depuși în memorie pentru calculele ulterioare.

În faza a treia, este programat nivelul inferior de alarmare (NIA), prin intermediul tastaturii, folosind un algoritm identic. Cei doi octeți împachetați BCD reprezentând nivelul inferior de alarmare sunt comparați cu nivelul superior de alarmare, programat în pasul anterior și depuși în memoria de date a unității centrale la adresele FF04H și FF05H doar dacă rezultatul scăderii pe 16

biți (NSA - NIA) este pozitiv. Dacă nivelul inferior de alarmare programat este mai mare decât nivelul superior de alarmare, programul revine în buclă la secțiunea de programare a nivelului inferior de alarmare. În continuare, cei doi octeți împachetați BCD sub forma MSZU (mii-sute-zeci-unități de milimetru), reprezentând valoarea nivelului inferior de alarmare, sunt convertiți în binar conform algoritmului:

$$(\text{MSZU})_{\text{NIA}} \rightarrow (\text{M})_{\text{NIA}} \times 1000 + (\text{S})_{\text{NIA}} \times 100 + (\text{Z})_{\text{NIA}} \times 10 + (\text{U})_{\text{NIA}} \quad (10.4)$$

și depuși în memorie pentru calculele ulterioare.

10.3.4 SECȚIUNEA DE MĂSURARE PROPRIU-ZISĂ

În acest punct, cele două ramuri - de validare a operării normale a echipamentului, respectiv de programare a parametrilor de funcționare, converg și se intră în secțiunea de măsurare propriu-zisă.

Secțiunea de măsurare propriu-zisă începe cu afișarea unui mesaj de avertizare a faptului că s-a ajuns în secțiunea de măsurare, care este menținut circa 0,5 secunde. Se continuă cu inițializarea secțiunii de captare a evenimentelor:

- sunt dezactivate întreruperile provenite de la depășirea superioară pe 8/16 biți a conținutului timer-ului T2, de la registrele de captare a evenimentelor, de la circuitele de comparare a conținutului timer-ului T2 cu registrele special prevăzute în acest sens;
- sunt inițializate (încărcate cu 0) registrele CTL0, CTH0, CTL1, CTH1, TML2, TMH2;
- este programat registrul CTCON, astfel încât registrele de captare a evenimentelor CTI0 și CTI1 să fie activate de fronturile descrescătoare ale impulsurilor externe;
- este programat timer-ul T2, prin activarea acestuia, cu dezactivarea depășirii pe 8/16 biți, ceasul de numărare fiind constituit de un oscilator intern cu frecvența 1/12 din frecvența oscilatorului microcontroller-ului, urmat de un registru de divizare cu 2 a acestui ceas.

Este, în continuare, activat emițătorul senzorului cu ultrasunete prin bascularea nivel coborât / nivel ridicat a bitului AX8. Primele două fronturi crescătoare ale semnalului recepționat de la ieșirea senzorului cu ultrasunete sunt prelucrate în vederea obținerii a două semnale diferite: frontul crescător al primului semnal determină captarea în registrul CTI0 a conținutului timer-ului T2 la acest moment de timp; frontul crescător al celui de-al doilea semnal (obținut prin recepționarea ultrasunetelor captate în urma reflexiei pe suprafața lichidului) determină captarea conținutului timer-ului T2 în registrul CTI1. Se determină numărul de impulsuri captate pe durata emisie-recepție prin

efectuarea diferenței pe 2 octeți între registrele CTI1 și CTI0.

Urmează secvența de măsurare a temperaturii mediului în care este efectuată determinarea înălțimii coloanei de lichid. Ieșirea în curent, de tip unificat $4\div 20$ mA, a sondei de temperatură este aplicată unui convertor curent-tensiune de tip șunt. Această tensiune este aplicată intrării ADCIN0 a convertorului analog-digital. Conversia este demarată prin software, programând în registrul ADCON bitul ADCCON.3 la "1" logic, iar biții ADCON.2, ADCON.1, ADCON.0 conform adresei canalului de intrare utilizat (în această situație, cei trei biți sunt programați la "0" logic). Se așteaptă sfârșitul conversiei analog-digitale, lucru sesizat prin poziționarea în "1" logic a bitului ADCON.4 (se execută de fapt o verificare în buclă a acestui bit, prin intermediul acumulatorului). Urmează interpretarea rezultatului și afișarea acestuia. Rezultatul conversiei transferat în acumulator, este împărțit la 2 pentru a se realiza conversia în domeniul specificat de temperatură (00H pentru temperatura de -20°C , respectiv 80H pentru temperatura de 108°C). Apoi se obțin cifrele temperaturii printr-un algoritm de comparare cu valoarea 14H și efectuarea scăderii (14H - rezultat) pentru afișarea temperaturilor negative, respectiv (rezultat - 14H) pentru afișarea temperaturilor pozitive și împărțire 10 pentru obținerea cifrelor pentru grade și zeci de grade.

Urmează secvența de determinare a nivelului lichidului din vas, conform algoritmului:

$$h[\text{mm}] = H[\text{mm}] - \frac{1}{2} \times v_0 \left[\frac{\text{mm}}{\text{ms}} \right] \times \frac{N \times T[\mu\text{s}] \times \left(1 + 0,17\% / ^{\circ}\text{C} \times \Delta t[^{\circ}\text{C}] \right)}{1000 \left[\frac{\mu\text{s}}{\text{ms}} \right]} \quad (10.5)$$

în care: h reprezintă nivelul lichidului din vas, H reprezintă valoarea programată a înălțimii vasului, v_0 reprezintă viteza ultrasunetelor la temperatura de 0°C ($v_0 = 321,5 \frac{\text{m}}{\text{s}}$), N reprezintă numărul de impulsuri captate pe durata emisie-recepție, T reprezintă durata unui ciclu mașină al microcontroller-ului ($T = 12 \times \frac{1}{11,059200\text{MHz}} = 1,085\mu\text{s}$), $0,17\% / ^{\circ}\text{C}$ reprezintă coeficientul de variație a vitezei de propagare a ultrasunetelor cu temperatura, iar Δt reprezintă temperatura mediului ambiant. Toate calculele se efectuează în binar, cu rezultatul pe 2 octeți, ceea ce asigură o rezoluție de măsurare a nivelului mai bună de 1 mm.

Înălțimea coloanei de lichid este afișată după determinarea cifrelor BCD ale acesteia, utilizând un algoritm iterativ de împărțire la 10. Cele patru cifre BCD obținute prin acest algoritm sunt împachetate în doi octeți codificați BCD și memorate în memoria de date a microcontroller-ului la adresele FF006H și FF07H.

Cuvântul de date care corespunde înălțimii coloanei de lichid, redus la o valoare pe un octet folosind un algoritm de codificare, este folosit pentru programarea factorului de umplere al impulsurilor de la ieșirea modulată în durată PWM0, care comandă generatorul de curent.

Se verifică depășirea nivelurilor prestabilite de alarmare (NSA și NIA), comparând înălțimea coloanei de lichid din vas cu pragurile programate anterior, compararea având un histerezis de ± 25 mm. Se efectuează diferența dintre nivelul superior de alarmare (NSA) și înălțimea coloanei de lichid din vas, iar dacă diferența este negativă (setarea indicatorului CARRY) se poziționează pe "1" logic bitul AX9. Se efectuează diferența dintre înălțimea coloanei de lichid din vas și nivelul inferior de alarmare (NIA), iar dacă diferența este negativă (setarea indicatorului CARRY) se poziționează pe "1" logic bitul AX10. Acești biți ai portului de ieșire acționează relele corespunzătoare. Atât valorile nivelului superior, cât și inferior, de alarmare, NSA și NIA, sunt citite din memoria de date a microcontroller-ului, de la adresele specificate anterior.

Se citește tastatura locală. Dacă nu s-a acționat nici o tastă, programul execută un salt necondiționat la secvența de inițializare a timer-ului T2, pentru reluarea în buclă a secțiunii de măsurare propriu-zisă. Dacă, însă, s-au acționat simultan (pentru evitarea unei acționări accidentale a tastaturii) tastele de programare (MODE) și de validare (ENTER), programul execută un salt la începutul secțiunii de programare a parametrilor globali de funcționare.

Rutinele specifice scrise și utilizate în cadrul acestei aplicații sunt:

- KEY - subrutina de citire a unei taste de la o tastatură cu o linie și patru coloane. Rezultatul citirii se află în acumulator;
- INPORT - subrutina de citire de la un port de intrare, utilizată în cadrul subrutinei de gestionare a tastaturii. Informațiile sunt vehiculate prin intermediul acumulatorului;
- UP - subrutina de incrementare a unei cifre BCD conținută în registrul R5, folosind tastatura locală;
- DOWN - subrutina de decrementare a unei cifre BCD conținută în registrul R5, folosind tastatura locală;
- DKEY1 - subrutina de programare a două cifre utilizând tastatura locală, afișarea celor două cifre la afișajul cu cristale lichide pe pozițiile 7 și 9 ale celei de-a doua linii și împachetare într-un octet codificat BCD;
- DKEY2 - subrutina de programare a două cifre utilizând tastatura locală, afișarea celor două cifre la afișajul cu cristale lichide pe pozițiile 10 și 11 ale celei de-a doua linii și împachetare într-un octet codificat BCD;
- SCR - afișarea a patru cifre la afișajul cu cristale lichide pe pozițiile 7, 9, 10 și 11 ale celei de-a doua linii. Cele patru cifre se obțin din doi octeți codificați BCD, manipulați prin intermediul registrelor R4 și R5;

- SCR1 - afișarea a patru cifre la afișajul cu cristale lichide pe pozițiile 7, 9, 10 și 11 ale primei linii. Cele patru cifre se obțin din doi octeți codificați BCD, manipulați prin intermediul registrelor R4 și R5.

10.4 LISTINGUL PROGRAMULUI DE APLICAȚIE PENTRU MĂSURAREA NIVELULUI NIVOK.ASM

```

; -----
; "Nivelmetru cu ultrasunete"
; -----
        ORG      8000H
; declararea resurselor suplimentare
        ADCON    equ      0C5H
        ADCH     equ      0C6H
        PWMP     equ      0FEH
        PWM0     equ      0FCH
        PWM1     equ      0FDH
        CTH0     equ      0CCH
        CTL0     equ      0ACH
        CTH1     equ      0CDH
        CTL1     equ      0ADH
        TMH2     equ      0EDH
        TML2     equ      0ECH
        TM2CON   equ      0EAH
        CTCON    equ      0EBH
        IEN1     equ      0E8H
        RAM      equ      70H
; secventa de initializare
        LJMP     INIT
INIT:    MOV      A, #00H
        MOV      P2, #1
        MOV      R0, #40H
        MOVX     @R0, A
        MOV      A, #1
        MOV      PWMP, A
        MOV      SCON, #40H
        MOV      TMOD, #20H
        MOV      PCON, #00H
        MOV      TH1, #0FDH
        MOV      TCON, #40H
        CLR      TI
        LCALL    INILCD
        LCALL    CLRLCD
        MOV      DPTR, #TEXT
        LCALL    TRX1
        MOV      R2, #0C0H
        LCALL    WRCMD
        INC      DPTR
        LCALL    TRX1
; definirea valorilor initiale
        MOV      DPTR, #0E000H

```

```

MOV     A, #60H
MOVX    @DPTR, A
INC     DPTR
MOV     A, #00H
MOVX    @DPTR, A
MOV     DPTR, #0E004H
MOV     A, #40H
MOVX    @DPTR, A
INC     DPTR
MOV     A, #00H
MOVX    @DPTR, A
MOV     DPTR, #0E008H
MOV     A, #20H
MOVX    @DPTR, A
INC     DPTR
MOV     A, #00H
MOVX    @DPTR, A
MOV     DPTR, #0E012H
MOV     A, #00H
MOVX    @DPTR, A
INC     DPTR
MOV     A, #00H
MOVX    @DPTR, A
; calibrare generator de curent
LCALL   SEC
LCALL   CLRLCD
MOV     DPTR, #TEXT8
LCALL   TRX1
MOV     R2, #0C0H
LCALL   WRCMD
INC     DPTR
LCALL   TRX1
OK4:    LCALL   KEY
        LCALL   SEC1
        JB      ACC.0, NON4
        JB      ACC.1, OK4
        JB      ACC.2, OK4
        JB      ACC.3, GC4
GC4:    LCALL   SEC
        LCALL   CLRLCD
        MOV     DPTR, #TEXT14
        LCALL   TRX1
        MOV     R2, #0C0H
        LCALL   WRCMD
        INC     DPTR
        LCALL   TRX1
        MOV     A, #0FFH
        MOV     PWM0, A
OK5:    LCALL   KEY
        LCALL   SEC1
        JB      ACC.0, GC20
        JB      ACC.1, OK5
        JB      ACC.2, OK5
        JB      ACC.3, OK5
GC20:   LCALL   SEC

```

```

        LCALL  CLRLCD
        MOV    DPTR,#TEXT15
        LCALL  TRX1
        MOV    R2,#0C0H
        LCALL  WRCMD
        INC    DPTR
        LCALL  TRX1
        MOV    A,#00H
        MOV    PWM0,A
OK6:    LCALL  KEY
        LCALL  SEC1
        JB     ACC.0,NON4
        JB     ACC.1,OK6
        JB     ACC.2,OK6
        JB     ACC.3,OK6
; selectare mod de functionare
NON4:   LCALL  SEC
        LCALL  CLRLCD
        MOV    DPTR,#TEXT1
        LCALL  TRX1
        MOV    R2,#0C0H
        LCALL  WRCMD
        INC    DPTR
        LCALL  TRX1
        LCALL  SEC
TAST:   LCALL  KEY
        LCALL  SEC1
        JB     ACC.0,OPER
        JB     ACC.1,TAST
        JB     ACC.2,TAST
        JB     ACC.3,PROGR
OPER:   LJMP   ACH
; introducere parola
PROGR:  LCALL  CLRLCD
        MOV    DPTR,#TEXT2
        LCALL  TRX1
        MOV    R2,#0C0H
        LCALL  WRCMD
        INC    DPTR
        LCALL  TRX1
        MOV    R5,#0
        MOV    R2,#1110B
        LCALL  WRCMD
KEY1:   MOV    R2,#0C7H
        LCALL  WRCMD
        LCALL  KEY
        LCALL  SEC1
        JB     ACC.0,VAL1
        JB     ACC.1,SUS1
        JB     ACC.2,JOS1
        JB     ACC.3,PROGR
SUS1:   LCALL  UP
        SJMP   PLAY1
JOS1:   LCALL  DOWN
PLAY1:  MOV    A,R5

```

```
        LCALL  HEXASC
        MOV   R2,A
        LCALL  TRX
        AJMP  KEY1
VAL1:   MOV   A,R5
        ANL   A,#0FH
        SWAP  A
        PUSH  ACC
        MOV   R5,#0
KEY2:   MOV   R2,#0C8H
        LCALL  WRCMD
        LCALL  KEY
        LCALL  SEC1
        JB    ACC.0,VAL2
        JB    ACC.1,SUS2
        JB    ACC.2,JOS2
        JB    ACC.3,PROGR
SUS2:   LCALL  UP
        SJMP  PLAY2
JOS2:   LCALL  DOWN
PLAY2:  MOV   A,R5
        LCALL  HEXASC
        MOV   R2,A
        LCALL  TRX
        AJMP  KEY2
VAL2:   POP   ACC
        ORL   A,R5
        SUBB  A,#59H
        JNZ   RAU
        JZ    BINE
RAU:    LCALL  CLRLCD
        MOV   DPTR,#TEXT3
        LCALL  TRX1
        MOV   R2,#0C0H
        LCALL  WRCMD
        INC   DPTR
        LCALL  TRX1
        LCALL  SEC
        AJMP  ACH
BINE:   LCALL  CLRLCD
        MOV   DPTR,#TEXT4
        LCALL  TRX1
        MOV   R2,#0C0H
        LCALL  WRCMD
        INC   DPTR
        LCALL  TRX1
; programare inaltime vas
INALT:  LCALL  SEC
        LCALL  CLRLCD
        MOV   DPTR,#TEXT5
        LCALL  TRX1
        MOV   R2,#0C0H
        LCALL  WRCMD
        INC   DPTR
        LCALL  TRX1
```

```

MOV     DPTR,#0E000H
MOVX    A,@DPTR
MOV     R5,A
INC     DPTR
MOVX    A,@DPTR
MOV     R4,A
LCALL   SCR
LCALL   DKEY1
MOV     DPTR,#0E000H
MOV     A,R5
MOVX    @DPTR,A
LCALL   DKEY2
MOV     DPTR,#0E001H
MOV     A,R5
MOVX    @DPTR,A
; validare a inaltimei vasului
LCALL   SEC
LCALL   CLRLCD
MOV     DPTR,#TEXT11
LCALL   TRX1
MOV     R2,#0C0H
LCALL   WRCMD
INC     DPTR
LCALL   TRX1
MOV     DPTR,#0E000H
MOVX    A,@DPTR
MOV     R5,A
INC     DPTR
MOVX    A,@DPTR
MOV     R4,A
LCALL   SCR1
OK1:    LCALL   KEY
        LCALL   SEC1
        JB      ACC.0,NSA
        JB      ACC.1,OK1
        JB      ACC.2,OK1
        JB      ACC.3,NON1
NON1:   AJMP    INALT
; programare nivel superior alarmare
NSA:    LCALL   SEC
        LCALL   CLRLCD
MOV     DPTR,#TEXT6
LCALL   TRX1
MOV     R2,#0C0H
LCALL   WRCMD
INC     DPTR
LCALL   TRX1
MOV     DPTR,#0E004H
MOVX    A,@DPTR
MOV     R5,A
INC     DPTR
MOVX    A,@DPTR
MOV     R4,A
LCALL   SCR
LCALL   DKEY1

```



```
MOV     DPTR,#0E000H
MOVX    A,@DPTR
CLR     C
SUBB    A,R5
JC      NSA
PUSH    ACC
MOV     DPTR,#0E004H
MOV     A,R5
MOVX    @DPTR,A
LCALL   DKEY2
POP     ACC
CLR     C
SUBB    A,#0
JNZ     GATA
MOV     DPTR,#0E001H
MOVX    A,@DPTR
CLR     C
SUBB    A,R5
JC      NSA
GATA:   MOV     DPTR,#0E005H
        MOV     A,R5
        MOVX    @DPTR,A
; validare nivel superior alarmare
        LCALL   SEC
        LCALL   CLRLCD
        MOV     DPTR,#TEXT12
        LCALL   TRX1
        MOV     R2,#0C0H
        LCALL   WRCMD
        INC     DPTR
        LCALL   TRX1
        MOV     DPTR,#0E004H
        MOVX    A,@DPTR
        MOV     R5,A
        INC     DPTR
        MOVX    A,@DPTR
        MOV     R4,A
        LCALL   SCR1
OK2:    LCALL   KEY
        LCALL   SEC1
        JB      ACC.0,NIA
        JB      ACC.1,OK2
        JB      ACC.2,OK2
        JB      ACC.3,NON2
NON2:   AJMP    NSA
; programare nivel inferior alarmare
NIA:    LCALL   SEC
        LCALL   CLRLCD
        MOV     DPTR,#TEXT7
        LCALL   TRX1
        MOV     R2,#0C0H
        LCALL   WRCMD
        INC     DPTR
        LCALL   TRX1
        MOV     DPTR,#0E008H
```

```

MOVX    A,@DPTR
MOV     R5,A
INC     DPTR
MOVX    A,@DPTR
MOV     R4,A
LCALL   SCR
LCALL   DKEY1
MOV     DPTR,#0E004H
MOVX    A,@DPTR
CLR     C
SUBB    A,R5
JC      NIA
PUSH    ACC
MOV     DPTR,#0E008H
MOV     A,R5
MOVX    @DPTR,A
LCALL   DKEY2
POP     ACC
CLR     C
SUBB    A,#0
JNZ     GATA1
MOV     DPTR,#0E005H
MOVX    A,@DPTR
CLR     C
SUBB    A,R5
JC      NIA
GATA1:  MOV     DPTR,#0E009H
        MOV     A,R5
        MOVX    @DPTR,A
; validare nivel inferior alarmare
        LCALL   SEC
        LCALL   CLRLCD
        MOV     DPTR,#TEXT13
        LCALL   TRX1
        MOV     R2,#0C0H
        LCALL   WRCMD
        INC     DPTR
        LCALL   TRX1
        MOV     DPTR,#0E008H
        MOVX    A,@DPTR
        MOV     R5,A
        INC     DPTR
        MOVX    A,@DPTR
        MOV     R4,A
        LCALL   SCR1
OK3:    LCALL   KEY
        JB      ACC.0,ACH
        JB      ACC.1,OK3
        JB      ACC.2,OK3
        JB      ACC.3,NON3
NON3:   AJMP    NIA
; masurare h si temperatura
ACH:    LCALL   CLRLCD
        MOV     DPTR,#TEXT9
        LCALL   TRX1

```

```
MOV     R2,#0C0H
LCALL   WRCMD
INC     DPTR
LCALL   TRX1
LCALL   SEC
; calcul valoare binara H
MOV     DPTR,#0E001H
MOVX    A,@DPTR
MOV     R7,A
ANL     A,#00001111B
MOV     R3,A
MOV     A,R7
SWAP    A
ANL     A,#00001111B
MOV     B,#10
MUL     AB
ADD     A,R3
PUSH    ACC
MOV     DPTR,#0E000H
MOVX    A,@DPTR
MOV     R7,A
ANL     A,#00001111B
MOV     R1,#0
MOV     R2,A
MOV     R3,#0
MOV     R4,#64H
MOV     A,R2
MOV     B,R4
MUL     AB
PUSH    ACC
MOV     A,R4
MOV     R4,B
MOV     B,R1
MUL     AB
ADD     A,R4
MOV     R5,A
MOV     A,B
ADDC    A,#0
MOV     R6,A
MOV     A,R2
MOV     B,R3
MUL     AB
ADD     A,R5
PUSH    ACC
MOV     A,B
ADDC    A,R6
MOV     R6,A
MOV     A,R3
MOV     B,R1
MUL     AB
ADD     A,R6
MOV     A,B
ADDC    A,#0
MOV     A,R7
SWAP    A
```

```
ANL      A,#00001111B
MOV      R1,#0
MOV      R2,A
MOV      R3,#03H
MOV      R4,#0E8H
MOV      A,R2
MOV      B,R4
MUL      AB
PUSH     ACC
MOV      A,R4
MOV      R4,B
MOV      B,R1
MUL      AB
ADD      A,R4
MOV      R5,A
MOV      A,B
ADDC     A,#0
MOV      R6,A
MOV      A,R2
MOV      B,R3
MUL      AB
ADD      A,R5
PUSH     ACC
MOV      A,B
ADDC     A,R6
MOV      R6,A
MOV      A,R3
MOV      B,R1
MUL      AB
ADD      A,R6
MOV      A,B
ADDC     A,#0
POP      ACC
MOV      R3,A
POP      ACC
MOV      R4,A
POP      ACC
MOV      R5,A
POP      ACC
MOV      R6,A
POP      ACC
MOV      R7,A
MOV      A,R7
CLR      C
ADD      A,R6
MOV      R6,A
MOV      A,R5
ADDC     A,#0
MOV      R5,A
MOV      A,R6
CLR      C
ADD      A,R4
MOV      R4,A
MOV      A,R5
ADDC     A,#0
```

```
ADD    A,R3
MOV    R3,A
MOV    DPTR,#0E002H
MOV    A,R3
MOVBX  @DPTR,A
INC    DPTR
MOV    A,R4
MOVBX  @DPTR,A
; calcul valoare binara NSA
MOV    DPTR,#0E005H
MOVBX  A,@DPTR
MOV    R7,A
ANL    A,#00001111B
MOV    R3,A
MOV    A,R7
SWAP   A
ANL    A,#00001111B
MOV    B,#10
MUL    AB
ADD    A,R3
PUSH   ACC
MOV    DPTR,#0E004H
MOVBX  A,@DPTR
MOV    R7,A
ANL    A,#00001111B
MOV    R1,#0
MOV    R2,A
MOV    R3,#0
MOV    R4,#64H
MOV    A,R2
MOV    B,R4
MUL    AB
PUSH   ACC
MOV    A,R4
MOV    R4,B
MOV    B,R1
MUL    AB
ADD    A,R4
MOV    R5,A
MOV    A,B
ADDC   A,#0
MOV    R6,A
MOV    A,R2
MOV    B,R3
MUL    AB
ADD    A,R5
PUSH   ACC
MOV    A,B
ADDC   A,R6
MOV    R6,A
MOV    A,R3
MOV    B,R1
MUL    AB
ADD    A,R6
MOV    A,B
```

```
ADDC    A,#0
MOV     A,R7
SWAP    A
ANL     A,#00001111B
MOV     R1,#0
MOV     R2,A
MOV     R3,#03H
MOV     R4,#0E8H
MOV     A,R2
MOV     B,R4
MUL     AB
PUSH    ACC
MOV     A,R4
MOV     R4,B
MOV     B,R1
MUL     AB
ADD     A,R4
MOV     R5,A
MOV     A,B
ADDC    A,#0
MOV     R6,A
MOV     A,R2
MOV     B,R3
MUL     AB
ADD     A,R5
PUSH    ACC
MOV     A,B
ADDC    A,R6
MOV     R6,A
MOV     A,R3
MOV     B,R1
MUL     AB
ADD     A,R6
MOV     A,B
ADDC    A,#0
POP     ACC
MOV     R3,A
POP     ACC
MOV     R4,A
POP     ACC
MOV     R5,A
POP     ACC
MOV     R6,A
POP     ACC
MOV     R7,A
MOV     A,R7
CLR     C
ADD     A,R6
MOV     R6,A
MOV     A,R5
ADDC    A,#0
MOV     R5,A
MOV     A,R6
CLR     C
ADD     A,R4
```

```
MOV     R4,A
MOV     A,R5
ADDC    A,#0
ADD     A,R3
MOV     R3,A
MOV     DPTR,#0E006H
MOV     A,R3
MOVX    @DPTR,A
INC     DPTR
MOV     A,R4
MOVX    @DPTR,A
; calcul NSAI
MOV     A,R4
CLR     C
SUBB    A,#3CH
MOV     R4,A
MOV     A,R3
SUBB    A,#00H
MOV     R3,A
MOV     DPTR,#0E014H
MOV     A,R3
MOVX    @DPTR,A
INC     DPTR
MOV     A,R4
MOVX    @DPTR,A
; calcul valoare binara NIA
MOV     DPTR,#0E009H
MOVX    A,@DPTR
MOV     R7,A
ANL     A,#00001111B
MOV     R3,A
MOV     A,R7
SWAP    A
ANL     A,#00001111B
MOV     B,#10
MUL     AB
ADD     A,R3
PUSH    ACC
MOV     DPTR,#0E008H
MOVX    A,@DPTR
MOV     R7,A
ANL     A,#00001111B
MOV     R1,#0
MOV     R2,A
MOV     R3,#0
MOV     R4,#64H
MOV     A,R2
MOV     B,R4
MUL     AB
PUSH    ACC
MOV     A,R4
MOV     R4,B
MOV     B,R1
MUL     AB
ADD     A,R4
```

```
MOV    R5,A
MOV    A,B
ADDC   A,#0
MOV    R6,A
MOV    A,R2
MOV    B,R3
MUL    AB
ADD    A,R5
PUSH   ACC
MOV    A,B
ADDC   A,R6
MOV    R6,A
MOV    A,R3
MOV    B,R1
MUL    AB
ADD    A,R6
MOV    A,B
ADDC   A,#0
MOV    A,R7
SWAP   A
ANL    A,#00001111B
MOV    R1,#0
MOV    R2,A
MOV    R3,#03H
MOV    R4,#0E8H
MOV    A,R2
MOV    B,R4
MUL    AB
PUSH   ACC
MOV    A,R4
MOV    R4,B
MOV    B,R1
MUL    AB
ADD    A,R4
MOV    R5,A
MOV    A,B
ADDC   A,#0
MOV    R6,A
MOV    A,R2
MOV    B,R3
MUL    AB
ADD    A,R5
PUSH   ACC
MOV    A,B
ADDC   A,R6
MOV    R6,A
MOV    A,R3
MOV    B,R1
MUL    AB
ADD    A,R6
MOV    A,B
ADDC   A,#0
POP    ACC
MOV    R3,A
POP    ACC
```



```
MOV    R4,A
POP     ACC
MOV     R5,A
POP     ACC
MOV     R6,A
POP     ACC
MOV     R7,A
MOV     A,R7
CLR     C
ADD     A,R6
MOV     R6,A
MOV     A,R5
ADDC    A,#0
MOV     R5,A
MOV     A,R6
CLR     C
ADD     A,R4
MOV     R4,A
MOV     A,R5
ADDC    A,#0
ADD     A,R3
MOV     R3,A
MOV     DPTR,#0E00AH
MOV     A,R3
MOVBX   @DPTR,A
INC     DPTR
MOV     A,R4
MOVBX   @DPTR,A
; calcul NIAS
MOV     A,R4
CLR     C
ADD     A,#3CH
MOV     R4,A
MOV     A,R3
ADDC    A,#00H
MOV     R3,A
MOV     DPTR,#0E016H
MOV     A,R3
MOVBX   @DPTR,A
INC     DPTR
MOV     A,R4
MOVBX   @DPTR,A
; capture evenimente START si STOP
ACH1:  CLR     A
MOV     IEN1,A
MOV     CTL0,A
MOV     CTH0,A
MOV     CTL1,A
MOV     CTH0,A
MOV     TML2,A
MOV     TMH2,A
MOV     A,#00000101B
MOV     CTCON,A
MOV     A,#00100101B
MOV     TM2CON,A
```

```

MOV     A,#01H
MOV     R3,A
MOV     DPTR,#0E012H
MOVX    A,@DPTR
ORL     A,R3
MOV     R3,A
INC     DPTR
MOVX    A,@DPTR
ORL     A,R3
MOV     P2,#1
MOV     R0,#40H
MOVX    @R0,A
MOV     A,#00H
MOV     R3,A
MOV     DPTR,#0E012H
MOVX    A,@DPTR
ORL     A,R3
MOV     R3,A
INC     DPTR
MOVX    A,@DPTR
ORL     A,R3
MOV     P2,#1
MOV     R0,#40H
MOVX    @R0,A
LCALL   SEC1
MOV     A,#00100100B
MOV     TM2CON,A
MOV     A,CTL1
MOV     R4,CTL0
CLR     C
SUBB    A,R4
MOV     R4,A
MOV     A,CTH1
MOV     R3,CTH0
SUBB    A,R3
MOV     R3,A
; salvare N
MOV     DPTR,#0E00CH
MOV     A,R3
MOVX    @DPTR,A
INC     DPTR
MOV     A,R4
MOVX    @DPTR,A
; masurare temperatura
LCALL   CLRLCD
MOV     DPTR,#TEXT10
LCALL   TRX1
MOV     R2,#0C0H
LCALL   WRCMD
INC     DPTR
LCALL   TRX1
MOV     R2,#1100B
LCALL   WRCMD
MOV     R2,#0C8H
LCALL   WRCMD

```

```

MOV     R2,#0DFH
LCALL   TRX
CLR     A
MOV     ADCON,A
ORL     A,#08H
MOV     ADCON,A
WAIT:   MOV     A,ADCON
JNB     ACC.4,WAIT
        MOV     A,ADCH
MOV     B,#2
DIV     AB
MOV     DPTR,#0E00FH
MOVX    @DPTR,A
; REZ1 = 17 x DELTA_T + 10000
        MOV     DPTR,#0E00FH
MOVX    A,@DPTR
MOV     B,A
MOV     A,#17
MUL     AB
MOV     R1,#27H
MOV     R2,#10H
CLR     C
ADD     A,R2
MOV     R4,A
MOV     A,B
ADDC    A,R1
MOV     R3,A
; REZ2 = N x (10000 + 17 x DELTA_T)
MOV     DPTR,#0E00CH
MOVX    A,@DPTR
MOV     R1,A
INC     DPTR
MOVX    A,@DPTR
MOV     R2,A
        MOV     A,R2
MOV     B,R4
MUL     AB
PUSH    ACC
MOV     A,R4
MOV     R4,B
MOV     B,R1
MUL     AB
ADD     A,R4
MOV     R5,A
MOV     A,B
ADDC    A,#0
MOV     R6,A
MOV     A,R2
MOV     B,R3
MUL     AB
ADD     A,R5
PUSH    ACC
MOV     A,B
ADDC    A,R6
MOV     R6,A

```

```

MOV     A,R3
MOV     B,R1
MUL     AB
ADD     A,R6
PUSH    ACC
MOV     A,B
ADDC    A,#0
PUSH    ACC
; REZ3 = REZ2 / 16384
POP     ACC
MOV     R4,A
POP     ACC
MOV     R5,A
POP     ACC
MOV     R6,A
POP     ACC
MOV     R3,#6
SHR1:   MOV     A,R4
        CLR     C
        RRC     A
        MOV     R4,A
        MOV     A,R5
        RRC     A
        MOV     R5,A
        MOV     A,R6
        RRC     A
        MOV     R6,A
        DJNZ    R3,SHR1
; REZ4 = REZ3 x 9353
MOV     A,R5
MOV     R3,A
MOV     A,R6
MOV     R4,A
MOV     R1,#24H
MOV     R2,#89H
MOV     A,R2
MOV     B,R4
MUL     AB
PUSH    ACC
MOV     A,R4
MOV     R4,B
MOV     B,R1
MUL     AB
ADD     A,R4
MOV     R5,A
MOV     A,B
ADDC    A,#0
MOV     R6,A
MOV     A,R2
MOV     B,R3
MUL     AB
ADD     A,R5
PUSH    ACC
MOV     A,B
ADDC    A,R6

```

```
MOV    R6,A
MOV    A,R3
MOV    B,R1
MUL    AB
ADD    A,R6
PUSH   ACC
MOV    A,B
ADDC   A,#0
PUSH   ACC
; s = REZ4 / 16384
POP    ACC
MOV    R4,A
POP    ACC
MOV    R5,A
POP    ACC
MOV    R6,A
POP    ACC
MOV    R3,#6
SHR2:  MOV    A,R4
        CLR    C
        RRC    A
        MOV    R4,A
        MOV    A,R5
        RRC    A
        MOV    R5,A
        MOV    A,R6
        RRC    A
        MOV    R6,A
        DJNZ   R3,SHR2
; calcul h = H - s
MOV    DPTR,#0E002H
MOVBX  A,@DPTR
MOV    R3,A
INC     DPTR
MOVBX  A,@DPTR
MOV    R4,A
MOV    A,R4
CLR     C
SUBB   A,R6
MOV    R6,A
MOV    A,R3
SUBB   A,R5
MOV    R5,A
; corectie h + 800
CLR     C
MOV    A,R6
ADD     A,#20H
MOV    R6,A
MOV    A,R5
ADDC   A,#03H
MOV    R5,A
; salvare h
MOV     DPTR,#0E010H
MOV     A,R5
MOVBX  @DPTR,A
```

```

        INC     DPTR
        MOV     A,R6
        MOVX    @DPTR,A
; cifra milimetri
        MOV     DPTR,#0E010H
        MOVX    A,@DPTR
        MOV     R6,A
        INC     DPTR
        MOVX    A,@DPTR
        MOV     R5,A
        MOV     R3,A
        CLR     A
        MOV     R2,A
        MOV     R4,#8
        MOV     R7,A
        MOV     A,R6
        MOV     B,#10
        DIV     AB
        MOV     R1,A
        MOV     R6,B
T_TWO1:  MOV     A,R5
        RLC     A
        MOV     R5,A
        MOV     A,R6
        RLC     A
        MOV     R6,A
        MOV     A,R7
        RLC     A
        MOV     R7,A
COMP1:  CJNE    A,#0,DONE1
        MOV     A,R6
        CJNE    A,#10,DONE1
        CJNE    R5,#0,DONE1
DONE1:  CPL     C
B_QUO1:  MOV     A,R2
        RLC     A
        MOV     R2,A
        JNB     ACC.0,LOP1
SUBS1:  MOV     A,R6
        SUBB    A,#10
        MOV     R6,A
        MOV     A,R7
        SUBB    A,#0
        MOV     R7,A
LOP1:   DJNZ    R4,T_TWO1
        MOV     A,#10
        MOV     B,R2
        MUL     AB
        MOV     B,A
        MOV     A,R3
        SUBB    A,B
        PUSH    ACC
        CLR     OV
; cifra centimetri
        MOV     A,R1

```

```

MOV     R6,A
MOV     A,R2
MOV     R5,A
MOV     R3,A
MOV     R4,#8
CLR     A
MOV     R2,A
MOV     R7,A
MOV     A,R6
MOV     B,#10
DIV     AB
MOV     R1,A
MOV     R6,B
T_TWO2: MOV     A,R5
        RLC     A
        MOV     R5,A
        MOV     A,R6
        RLC     A
        MOV     R6,A
        MOV     A,R7
        RLC     A
        MOV     R7,A
COMP2:  CJNE    A,#0,DONE2
        MOV     A,R6
        CJNE    A,#10,DONE2
        CJNE    R5,#0,DONE2
DONE2:  CPL     C
B_QUO2: MOV     A,R2
        RLC     A
        MOV     R2,A
        JNB     ACC.0,LOP2
SUBS2:  MOV     A,R6
        SUBB    A,#10
        MOV     R6,A
        MOV     A,R7
        SUBB    A,#0
        MOV     R7,A
LOP2:   DJNZ    R4,T_TWO2
        MOV     A,#10
        MOV     B,R2
        MUL     AB
        MOV     B,A
        MOV     A,R3
        SUBB    A,B
        PUSH    ACC
        CLR     OV
; cifra decimetri si metri
        MOV     A,R2
        MOV     B,#10
        DIV     AB
        MOV     R2,A
        MOV     A,B
        PUSH    ACC
        MOV     A,R2
        PUSH    ACC

```

```

; afisare h
    MOV     R2,#84H
    LCALL   WRCMD
    POP     ACC
    LCALL   HEXASC
    MOV     R2,A
    LCALL   TRX
    MOV     R2,#86H
    LCALL   WRCMD
    POP     ACC
    LCALL   HEXASC
    MOV     R2,A
    LCALL   TRX
    MOV     R2,#87H
    LCALL   WRCMD
    POP     ACC
    LCALL   HEXASC
    MOV     R2,A
    LCALL   TRX
    MOV     R2,#88H
    LCALL   WRCMD
    POP     ACC
    LCALL   HEXASC
    MOV     R2,A
    LCALL   TRX
; afisare temperatura
    MOV     DPTR,#0E00FH
    MOVX    A,@DPTR
    MOV     R5,A
    CLR     C
    SUBB    A,#20
    MOV     R4,A
    JC      TNEG
TPOZ:  MOV     R2,#0C4H
    LCALL   WRCMD
    MOV     R2,#00101011B
    LCALL   TRX
    MOV     R2,#0C5H
    LCALL   WRCMD
    MOV     A,R4
    MOV     B,#10
    DIV     AB
    LCALL   HEXASC
    MOV     R2,A
    LCALL   TRX
    MOV     R2,#0C6H
    LCALL   WRCMD
    MOV     A,B
    LCALL   HEXASC
    MOV     R2,A
    LCALL   TRX
    LJMP    SALT
TNEG:  MOV     R2,#0C4H
    LCALL   WRCMD
    MOV     R2,#00101101B

```



```
LCALL TRX
MOV R2,#0C5H
LCALL WRCMD
MOV A,#20
CLR C
SUBB A,R5
MOV B,#10
DIV AB
LCALL HEXASC
MOV R2,A
LCALL TRX
MOV R2,#0C6H
LCALL WRCMD
MOV A,B
LCALL HEXASC
MOV R2,A
LCALL TRX
SALT: LCALL SEC
; programare generator curent
; codificare h pe un octet
MOV DPTR,#0E010H
MOVX A,@DPTR
MOV R6,A
INC DPTR
MOVX A,@DPTR
MOV R7,A
MOV R3,#3
SHL1: CLR C
MOV A,R7
RLC A
MOV R7,A
MOV A,R6
RLC A
MOV R6,A
DJNZ R3,SHL1
; codificare H pe un octet
MOV DPTR,#0E002H
MOVX A,@DPTR
MOV R4,A
INC DPTR
MOVX A,@DPTR
MOV R5,A
MOV R3,#3
SHL2: CLR C
MOV A,R5
RLC A
MOV R5,A
MOV A,R4
RLC A
MOV R4,A
DJNZ R3,SHL2
; calcul valoare hx255
MOV A,R6
MOV B,#0FFH
MUL AB
```

```

; calcul (hx255)/H
    MOV     R6,B
    MOV     R5,A
    MOV     R3,A
    MOV     A,R4
    MOV     R0,A
    MOV     RAM,A
    MOV     R4,#8
    CLR     A
    MOV     R2,A
    MOV     R7,A
    MOV     A,R6
    MOV     B,R0
    DIV     AB
    MOV     R1,A
    MOV     R6,B
T_TWO3:  MOV     A,R5
    RLC     A
    MOV     R5,A
    MOV     A,R6
    RLC     A
    MOV     R6,A
    MOV     A,R7
    RLC     A
    MOV     R7,A
COMP3:  CJNE   A,#0,DONE3
    MOV     A,R6
    CJNE   A,RAM,DONE3
    CJNE   R5,#0,DONE3
DONE3:  CPL     C
B_QUO3:  MOV     A,R2
    RLC     A
    MOV     R2,A
    JNB     ACC.0,LOP3
SUBS3:  MOV     A,R6
    SUBB    A,R0
    MOV     R6,A
    MOV     A,R7
    SUBB    A,#0
    MOV     R7,A
LOP3:   DJNZ   R4,T_TWO3
    MOV     A,R0
    MOV     B,R2
    MUL     AB
    MOV     B,A
    MOV     A,R3
    SUBB    A,B
    CLR     OV
    MOV     A,R2
; programare PWM0
    CPL     A
    MOV     PWM0,A
; citire h in binar
    MOV     DPTR,#0E010H
    MOVBX   A,@DPTR

```

```

        MOV     R1,A
        INC     DPTR
        MOVX    A,@DPTR
        MOV     R2,A
; depasire inferioara
; citire NIA in binar
DEI:    MOV     DPTR,#0E00AH
        MOVX    A,@DPTR
        MOV     R4,A
        INC     DPTR
        MOVX    A,@DPTR
        MOV     R5,A
; citire NIAS in binar
        MOV     DPTR,#0E016H
        MOVX    A,@DPTR
        MOV     R6,A
        INC     DPTR
        MOVX    A,@DPTR
        MOV     R7,A
; calcul h-NIA si semnalizare
        MOV     A,R2
        CLR     C
        SUBB    A,R5
        MOV     A,R1
        SUBB    A,R4
        JC      DEPINF
        JNC     NDI
DEPINF: MOV     A,#04H
        MOV     DPTR,#0E012H
        MOVX    @DPTR,A
        MOV     P2,#1
        MOV     R0,#40H
        MOVX    @R0,A
        AJMP    CONTS
NDI:    MOV     DPTR,#0E012H
        MOVX    A,@DPTR
        ANL     A,#04H
        JB      ACC.2,CAZ1
        JNB     ACC.2,CAZ2
; scadere NIAS-h si avertizare
CAZ1:   MOV     A,R7
        CLR     C
        SUBB    A,R2
        MOV     A,R6
        SUBB    A,R1
        JNC     DEPINF
        JC      NDPI
NDPI:   MOV     A,#00H
        MOV     DPTR,#0E012H
        MOVX    @DPTR,A
        MOV     P2,#1
        MOV     R0,#40H
        MOVX    @R0,A
        AJMP    DES
CAZ2:   AJMP    NDPI

```

```

; depasire superioara
; citire NSAI in binar
DES:  MOV    DPTR,#0E014H
      MOVX   A,@DPTR
      MOV    R4,A
      INC    DPTR
      MOVX   A,@DPTR
      MOV    R5,A
; citire NSA in binar
      MOV    DPTR,#0E006H
      MOVX   A,@DPTR
      MOV    R6,A
      INC    DPTR
      MOVX   A,@DPTR
      MOV    R7,A
      MOV    DPTR,#0E013H
      MOVX   A,@DPTR
      ANL    A,#02H
      JB     ACC.1,CAZ3
      JNB    ACC.1,CAZ4
CAZ3:  MOV    A,R2
      CLR    C
      SUBB   A,R5
      MOV    A,R1
      SUBB   A,R4
      JC     NDPS
      JNC    DEPSUP
CAZ4:  MOV    A,R2
      CLR    C
      SUBB   A,R7
      MOV    A,R1
      SUBB   A,R6
      JC     NDPS
      JNC    DEPSUP
DEPSUP: MOV    A,#02H
      MOV    DPTR,#0E013H
      MOVX   @DPTR,A
      MOV    P2,#1
      MOV    R0,#40H
      MOVX   @R0,A
      AJMP   CONTS
NDPS:  MOV    A,#00H
      MOV    DPTR,#0E013H
      MOVX   @DPTR,A
      MOV    P2,#1
      MOV    R0,#40H
      MOVX   @R0,A
      AJMP   CONTS
CONTS: LCALL  SEC
; transmisie seriala
      MOV    R7,#3
      MOV    DPTR,#0E00FH
TRANS: MOVX   A,@DPTR
      MOV    SBUF,A
      CLR    TI

```

```

WT:      JNB      TI,WT
          CLR      TI
          INC      DPTR
          DJNZ     R7,TRANS
          LCALL    INPORT
          JZ       GO
          JB       ACC.0,PRO
PRO:      LJMP     PROGR
GO:       LJMP     ACH1
;
UP:       MOV      A,R5
          INC      A
          DA       A
          ANL      A,#0FH
          MOV      R5,A
          RET

;
DOWN:     MOV      A,R5
          ANL      A,#0FH
          CLR      C
          SUBB     A,#1
          JC       ADJST
          JNC      CONT
ADJST:    MOV      A,#9H
CONT:     ANL      A,#0FH
          MOV      R5,A
          RET

;
KEY:      LCALL    INPORT
          JZ       KEY
          RET

;
INPORT:   MOV      P2,#1
          MOV      R0,#60H
          MOVX     A,@R0
          CPL      A
          ANL      A,#0FH
          RET

;
DKEY1:    MOV      A,R5
          MOV      R3,A
          SWAP     A
          ANL      A,#0FH
          MOV      R5,A
          MOV      R2,#1110B
          LCALL    WRCMD
KEY3:     MOV      R2,#0C6H
          LCALL    WRCMD
          LCALL    KEY
          LCALL    SEC1
          JB       ACC.0,VAL3
          JB       ACC.1,SUS3
          JB       ACC.2,JOS3
SUS3:     LCALL    UP
          SJMP     PLAY3

```

```
JOS3:  LCALL  DOWN
PLAY3:  MOV    A,R5
        LCALL  HEXASC
        MOV    R2,A
        LCALL  TRX
        AJMP   KEY3
VAL3:   MOV    A,R5
        ANL    A,#0FH
        SWAP   A
        PUSH   ACC
        MOV    A,R3
        ANL    A,#0FH
        MOV    R5,A
KEY4:   MOV    R2,#0C8H
        LCALL  WRCMD
        LCALL  KEY
        LCALL  SEC1
        JB     ACC.0,VAL4
        JB     ACC.1,SUS4
        JB     ACC.2,JOS4
SUS4:   LCALL  UP
        SJMP   PLAY4
JOS4:   LCALL  DOWN
PLAY4:  MOV    A,R5
        LCALL  HEXASC
        MOV    R2,A
        LCALL  TRX
        AJMP   KEY4
VAL4:   POP    ACC
        ORL    A,R5
        MOV    R5,A
        RET

;
DKEY2:  MOV    A,R4
        MOV    R3,A
        SWAP   A
        ANL    A,#0FH
        MOV    R5,A
        MOV    R2,#1110B
        LCALL  WRCMD
KEY5:   MOV    R2,#0C9H
        LCALL  WRCMD
        LCALL  KEY
        LCALL  SEC1
        JB     ACC.0,VAL5
        JB     ACC.1,SUS5
        JB     ACC.2,JOS5
SUS5:   LCALL  UP
        SJMP   PLAY5
JOS5:   LCALL  DOWN
PLAY5:  MOV    A,R5
        LCALL  HEXASC
        MOV    R2,A
        LCALL  TRX
        AJMP   KEY5
```

```
VAL5:  MOV    A,R5
        ANL    A,#0FH
        SWAP   A
        PUSH  ACC
        MOV    A,R3
        ANL    A,#0FH
        MOV    R5,A
KEY6:  MOV    R2,#0CAH
        LCALL  WRCMD
        LCALL  KEY
        LCALL  SEC1
        JB     ACC.0,VAL6
        JB     ACC.1,SUS6
        JB     ACC.2,JOS6
SUS6:  LCALL  UP
        SJMP   PLAY6
JOS6:  LCALL  DOWN
PLAY6: MOV    A,R5
        LCALL  HEXASC
        MOV    R2,A
        LCALL  TRX
        LJMP   KEY6
VAL6:  POP     ACC
        ORL    A,R5
        MOV    R5,A
        RET

;
SCR:   MOV    R2,#1100B
        LCALL  WRCMD
        MOV    R2,#0C6H
        LCALL  WRCMD
        MOV    A,R5
        SWAP   A
        ANL    A,#0FH
        CALL   HEXASC
        MOV    R2,A
        LCALL  TRX
        MOV    R2,#0C8H
        LCALL  WRCMD
        MOV    A,R5
        ANL    A,#0FH
        LCALL  HEXASC
        MOV    R2,A
        LCALL  TRX
        MOV    R2,#0C9H
        LCALL  WRCMD
        MOV    A,R4
        SWAP   A
        ANL    A,#0FH
        CALL   HEXASC
        MOV    R2,A
        LCALL  TRX
        MOV    R2,#0CAH
        LCALL  WRCMD
        MOV    A,R4
```

```

        ANL      A,#0FH
        LCALL    HEXASC
        MOV      R2,A
        LCALL    TRX
        RET

;
SCR1:   MOV      R2,#1100B
        LCALL    WRCMD
        MOV      R2,#086H
        LCALL    WRCMD
        MOV      A,R5
        SWAP     A
        ANL      A,#0FH
        CALL     HEXASC
        MOV      R2,A
        LCALL    TRX
        MOV      R2,#088H
        LCALL    WRCMD
        MOV      A,R5
        ANL      A,#0FH
        LCALL    HEXASC
        MOV      R2,A
        LCALL    TRX
        MOV      R2,#089H
        LCALL    WRCMD
        MOV      A,R4
        SWAP     A
        ANL      A,#0FH
        CALL     HEXASC
        MOV      R2,A
        LCALL    TRX
        MOV      R2,#08AH
        LCALL    WRCMD
        MOV      A,R4
        ANL      A,#0FH
        LCALL    HEXASC
        MOV      R2,A
        LCALL    TRX
        RET

;
TRX1:   CLR      A
        MOVC     A,@A+DPTR
        CJNE     A,#24H,TRCAR1
        RET

TRCAR1: MOV      R2,A
        LCALL    WRDAT
        INC      DPTR
        SJMP     TRX1

;
SEC:    MOV      R6,#255
LOOP1:  MOV      R7,#255
LOOP:   NOP
        NOP
        NOP
        NOP

```



```

NOP
NOP
NOP
NOP
DJNZ R7,LOOP
DJNZ R6,LOOP1
RET
;
SEC1: MOV R6,#255
LOOP3: MOV R7,#255
LOOP2: NOP
NOP
NOP
NOP
DJNZ R7,LOOP2
DJNZ R6,LOOP3
RET
;
TEXT: DB ' NIVELMETRU $'
DB ' CU ULTRASUNETE $'
TEXT1: DB 'OPERARE (ENTER)$'
DB 'PROGRAMARE(MODE)$'
TEXT2: DB 'VALIDARE PAROLA:$'
DB ' 00 $'
TEXT3: DB ' PAROLA $'
DB ' INCORECTA! $'
TEXT4: DB ' PAROLA $'
DB ' CORECTA! $'
TEXT5: DB ' INALTIME VAS $'
DB ' H = . m $'
TEXT6: DB 'NIV.SUP.ALARMARE$'
DB 'NSA = . m $'
TEXT7: DB 'NIV.INF.ALARMARE$'
DB 'NIA = . m $'
TEXT8: DB 'CALIBRARE G.C.?$'
DB '(D-MODE N-ENTER)$'
TEXT9: DB ' REGIM NORMAL $'
DB ' DE FUNCTIONARE $'
TEXT10: DB 'h = . m $'
DB 't = C $'
TEXT11: DB ' H = . m ?$'
DB '(D-ENTER N-MODE)$'
TEXT12: DB 'NSA = . m ?$'
DB '(D-ENTER N-MODE)$'
TEXT13: DB 'NIA = . m ?$'
DB '(D-ENTER N-MODE)$'
TEXT14: DB ' CALIBRARE 4 mA $'
DB ' (GATA - ENTER) $'
TEXT15: DB ' CALIBRARE 20mA $'
DB ' (GATA - ENTER) $'
;
HEXASC: ANL A,#0FH
JNB ACC.3,NOADJ
JB ACC.2,ADJ
JNB ACC.1,NOADJ

```

```
ADJ:   ADD     A, #07H
NOADJ: ADD     A, #30H
      RET
;
TRX:   CLR     A
      MOV     A, R2
      LJMP    TRCAR
LL7:   RET
TRCAR: MOV     R2, A
      LCALL   WRDAT
      SJMP    LL7
;
INILCD:      MOV     R2, #38H
      LCALL   WRCMD
      MOV     R4, #50
DEL4MS:      LCALL   DELAY
      DJNZ    R4, DEL4MS
      MOV     R4, #4
LINI:   LCALL   WRCMD
      LCALL   DELAY
      DJNZ    R4, LINI
      LCALL   WLCD
      MOV     R2, #6
      LCALL   WRCMD
      LCALL   WLCD
      MOV     R2, #0EH
      LCALL   WRCMD
      LCALL   WLCD
      MOV     R2, #1
      LCALL   WRCMD
      LCALL   WLCD
      RET
;
CLRLCD:      MOV     R2, #1
      LCALL   WRCMD
      MOV     R2, #1100B
      LCALL   WRCMD
      RET
;
WLCD:   LCALL   RDCMD
      JB      ACC.7, WLCD
      RET
;
RDCMD:  MOV     P2, #1
      MOV     R0, #2
      MOVX    A, @R0
      RET
;
WRCMD:  LCALL   WLCD
      MOV     A, R2
      MOV     P2, #1
      MOV     R0, #0
      MOVX    @R0, A
      RET
;
```

```
WRDAT: LCALL  WLCD
        MOV    P2, #1
        MOV    R0, #1
        MOV    A, R2
        MOVX   @R0, A
        RET

;
DELAY:  MOV    R3, #17
LL1:    NOP
        NOP
        DJNZ   R3, LL1
        RET

;
        END
```


BIBLIOGRAFIE

1. Albu, M., Vlaicu, C., ș.a.: *Probleme legate de proiectarea, alegerea și utilizarea programelor specializate pentru măsurări electrice*, Lucrările Sesiunii Jubiliare de Comunicări Științifice, Facultatea de Electrotehnică, decembrie 1996, pg.111-115;
2. Antoniu, M., Poli, Șt., Antoniu, E.: *Măsurări electronice - Aparate și sisteme de măsură numerice*, Editura SATYA, 1997;
3. Damms, K. J., Heinze, R. : *Leittechnik: Unix und Windows NT*, ETZ Archiv, Heft 1-2/1998, pg. 8-10;
4. Dale, N., Weems, C.: *Introduction to PASCAL and Structured Design*, D. C. Heath and Company, 1987;
5. Dale, N., Lilly, C. S.: *PASCAL Plus Data Structures. Algorithms and Advanced Programming*, D. C. Heath and Company, 1989;
6. Fitzgerald, J.: *Business Data Communications - Basic Concepts, Security and Design*, J. Wiley & Sons Inc., 1990;
7. Ghinea, D., Popescu, C., Popescu, M., *Analiza spectrală a semnalelor energetice*, Ars docendi, București, 1999;
8. Golovanov, C., Ștefănescu, C.: *Real Time Microcontroller - Based System For Assessing Voltage Quality With Relation To Unbalance*, Rev. Roum. Sci. Techn. - Electrotechn. et Energ., 41, 3, Bucharest, 1996;
9. Golovanov, C., Ștefănescu, C.: *Intelligent Data Acquisition System for Assessing Quality to Unbalance*, AMSE Press - Modelling, Measurement & Control, A, Vol. 59, No. 3, pag. 1-10, 1995;
10. Gray, P. R., Meyer, R. G.: *Circuite integrate analogice. Analiză și proiectare*, Editura Tehnică, București, 1983;
11. Hilburn, J.: *Manual of Active Filter Design*, McGraw Hill, 1973;
12. Hilton, H. E.: *A 10 Megasample-per-Second Analog-to-Digital Converter with Filter and Memory*, Hewlett - Packard Journal, Volume 44, Number 5, October 1993;
13. Holland, R. C.: *Microcomputers and Their Interfacing*, Pergamon Press, 1987;
14. Kientzle, T.: *The Working Programmer's Guide to Serial Protocols*, Coriolis Group Books, 1995;
15. Lacanette, K.: *A Basic Introduction to Filters, Active, Passive and Switched-Capacitor*, AN 779, National Semiconductor, 1991;

16. Lipovski, G., J.: *Object - Oriented Interfacing to 16-Bit Microcontrollers*, Prentice-Hall, Texas, 1993;
17. Melkonian, L. G.: *Dynamic Specifications for Sampling A/D converters*, AN 769, National Semiconductor, 1991;
18. Millman, J., Grabel, A.: *Microelectronics (second edition)*, Mc Graw Hill, 1987;
19. Posdărăscu, E., Cupcea, N., Ștefănescu, C. - *Sistem automat de măsură și control al transferului termic pentru incinte izoterme în regim staționar*, A Doua Conferință Internațională de Sisteme Electromecanice SIELMEC'99, vol. III, pag. 63-68, 8-9 octombrie 1999, Chișinău;
20. Posdărăscu, E., Cupcea, N., Ștefănescu, C. - *Stand pentru studierea experimentală a amortizoarelor hidraulice de translație*, A Doua Conferință Internațională de Sisteme Electromecanice SIELMEC'99, vol. III, pag. 168-173, 8-9 octombrie 1999, Chișinău;
21. Potolea, E., Gheorghe, Șt.: *Calitatea tensiunii în rețelele electrice poluate cu armonici. Sistem de analiză și datele obținute*, Simpozionul Național de Rețele Electrice, Suceava 1994, vol. I, pag. 121-131;
22. Potolea, E., Pătrășcoiu, S., Gheorghe, Șt.: *Supravegherea rețelelor și analiza regimului deformant în sistemele electroenergetice zonale*, Revista Energetica nr. 1, seria B, Editura Tehnică, 1992;
23. Schumny, H.: *Meßtechnik mit dem Personal Computer*, Springer Verlag, Berlin, 1993;
24. Ștefănescu, C.: *Sistem inteligent de analiză a regimului deformant în rețele electrice*, Teză de doctorat, U.P.B., București, 1998;
25. Ștefănescu, C., ș.a.: *Sistem inteligent de analiză a calității tensiunii în rețele electrice trifazate, funcționând în regim nesinusoidal*, Conferința Națională de Energetică Industrială CNEI '98, Bacău 30 oct - 1 nov 1998;
26. Ștefănescu, C., Cupcea, N., ș.a.: *Sistem cu microcontroller pentru măsurarea nivelului lichidelor depozitate în vase staționare la presiune joasă*, Conferința Națională de Energetică Industrială CNEI '98, Secția I - Electroenergetică, Bacău 30 oct - 1 nov 1998;
27. Ștefănescu, C., Cupcea, N., Popescu, D. - *Intelligent Power Quality Analyzer*, Proceedings of 5th International Conference on Information Systems Analysis and Synthesis ISAS '99, vol. 7 - Industrial Systems, pag. 192-198, Orlando, Florida, July 31 - August 4, 1999;
28. Ștefănescu, C., Cupcea, N. - *Echipamente numerice versatile destinate proceselor de reglare în domeniul industrial*, A Doua Conferință Internațională de Sisteme Electromecanice SIELMEC'99, vol. II, pag. 15-18, 8-9 octombrie 1999, Chișinău;

29. Ștefănescu, C., Cupcea, N. - *Sisteme inteligente de măsurare a perioadei semnalelor electrice*, A Doua Conferință Internațională de Sisteme Electromecanice SIELMEC'99, vol. III, pag. 15-18, 8-9 octombrie 1999, Chișinău;
30. Ștefănescu, C., Cupcea, N., Simion, V.: *Microcontroller-based development system for system control applications*, Proceedings of the Symposium on Advanced Topics in Electrical Engineering ATEE - 2000, pag. 139-146, Bucharest, 2000;
31. Ștefănescu, C., Sărăcin, M.: *Sistem inteligent, multifuncțional, de achiziții de date*, Grupul ICPE: Electronica Prezent și Perspective, pag. M32-M36, București, 30-31 oct. 1995;
32. Ștefănescu, C., Simion, V.: *General-purpose digital scope based on PIC microcontrollers*, Proceedings of the Symposium on Advanced Topics in Electrical Engineering ATEE - 2000, pag. 147-154, Bucharest, 2000;
33. Tedeschi, F.: *Active Filter Cookbook*, Tab Book No. 1133, 1979;
34. Tertișco, M., Stoica, P., Popescu, Th.: *Identificarea asistată de calculator a sistemelor*, Editura Tehnică, București, 1987;
35. Tran Tien Lang: *Acquisition et traitement des signaux de mesure a l'aide de microprocesseur*, Techniques de l'ingenieur, Mesures et Controle, R 525-1 SUPELEC, Paris 1989;
36. Tran Tien Lang: *Microprocesseurs de traitement du signal de mesure*, Techniques de l'ingenieur, Mesures et Controle, R 526-1 SUPELEC, Paris 1989;
37. Tran Tien Lang: *Systemes de Mesures Informatises - mise en oeuvre des microprocesseurs et microcontrolleurs en instrumentation*, Masson Paris, 1992;
38. Turturea, D.: *Totul despre WINDOWS 3.1*, Editura Tehnică, București, 1993;
39. Vlaicu, C.: *Sisteme digitale de măsurare*, Editura ICPE, 2000;
40. Vlaicu, C., Albu, M.: *Sisteme numerice de măsurare*, Îndrumar de laborator, Litografia UPB, 1999;
41. Wong, T. : *Understanding USB*, in Electronics World, November 1999, pg. 890-900;
42. Young, D.A.: *Motif® Debugging and Performance Tuning*, Prentice Hall, 1995;
43. ***: *Advanced Micro Devices - CMOS Memory Products Data Book/Handbook*, 1991;

44. ***: *Advanced Micro Devices - PAL Device Data Book and Design Guide*, 1993;
45. ***: *Analog Devices - Data Acquisition Components and Subsystems Data Book*, 1996;
46. ***: *Burr Brown - Intelligent Instrumentation: Personal Computer Instrumentation - Data Acquisition, Test, Measurement and Control -*, 1994;
47. ***: *Cyberresearch - PC Systems HandBook for Scientists and Engineers*, No. 1, 1997;
48. ***: *Getting Started with HP VEE for Windows*, Hewlett-Packard Company, 1995;
49. ***: *Hybrid Memory Products Ltd. - Advanced MOS Memory*, 1990;
50. ***: *Keithley Metrabyte Data Acquisition - Catalog & Reference Guide*, 1993;
51. ***: *National Instruments - Data Acquisition Systems Data Book*, 1997;
52. ***: *Micron Semiconductor Inc, - SRAM Data Book*, 1993;
53. ***: *Micron Semiconductor Inc, - DRAM Data Book*, 1993;
54. ***: *Micron Semiconductor Inc, - Specialty DRAM Data Book*, 1993;
55. ***: *National Semiconductor - Data Acquisition Systems Data Book*, 1996;
56. ***: *National Semiconductor - Interface Products Data Book*, 1996;
57. ***: *National Semiconductor - Linear Applications' Data Book (second edition)*, 1995;
58. ***: *Philips Semiconductor - 80C51-Based 8-Bit Microcontrollers*, 1997.