

STIINTA SI
TEHNICA

SUPLIMENT

'90

DE INFORMATICA SI CALCULATOARE

1





INTERNATIONAL DATA GROUP
The World's Leader in Information Services on Information Technology

În sfârșit...

...Sîntem în măsură să vă oferim
mult așteptatul supliment
de informatică și calculatoare intitulat —
INFOCLUB!

De ce „INFOCLUB“?

Foarte simplu! Deoarece acest supliment nu apare pe un teren arid, el continuă, în mod firesc, rubrica cu același nume, din revista „Știință și tehnică” care a promovat în mod constant informatica, sub diverse rubrici, oferind cititorilor și pasionaților informaticii o gamă cit mai variată de date și informații dintr-un domeniu de foarte mare interes.

Datorită nenumăratelor scrisori primite de la dumneavoastră, dragi cititori, rubricile de informatică au căutat — în limitele spațiului de care a dispus și a unor condiții „obiective” asupra cărora nu mai revenim — să acopere din cit mai multe unghiuri problemele calculatoarelor. Așa au apărut rubricile: „Un limbaj de programare în serial”, „Școala la ora informaticii”, „Infoclub”, „Programe comentate”, „Consultații BASIC”, etc., care s-au bucurat de un deosebit succes.

Acesta este și motivul pentru care vom publica în acest prim număr al suplimentului nostru, ceea ce noi am intitulat „Bibliografia ST de informatică” pe ultimii 4 ani (fără ca aceasta să fie răspunsul complet la întrebarea: „ce am făcut în ultimii 5 ani?”) pentru a veni în întâmpinarea tuturor iubitorilor informaticii din țară, contribuind, cu mijloace proprii, la instruirea lor.

Ce își propune „INFOCLUB”? sau Cui se adresează INFOCLUB?

Situația obiectivă a industriei (hard și soft) de profil din țara noastră, cât și cerințele reale ale începătorilor dar și ale celor care doresc să se perfecționeze, au condus la structurarea acestui supliment în trei părți distincte: o bună parte (aproape jumătate) este destinată compatibilelor Spectrum; dată fiind răspîndirea acestui tip de „home computer” în țara noastră; un spațiu însemnat este acordat problemelor legate de calculatoarele personale profesionale și, în sfârșit, am rezervat un număr de pagini și unor aplicații pentru microcalculatoarele compatibile CP/M.

Desigur că nu lipsesc noutățile de ultimă oră din domeniu, tendințe actuale de piață, informații la zi despre cluburi și societăți de informatică care au fost înființate în țara noastră, toate acestea întregind imaginea unui domeniu foarte dinamic. Scopul principal a întregului supliment este, după cum veți vedea, prezentarea aplicațiilor concrete și nu a teoriei, astfel încît, după parcurgerea acestuia, cititorul să posede un bagaj sporit de cunoștințe în capitolul care îl interesează. Sperăm că am reușit să răspundem preferințelor dv., în orice caz, vă așteptăm cu sugestiile — așa cum ați făcut-o în toți acești ani pentru rubrica de informatică din revistă — astfel încît conținutul suplimentului să vină în întâmpinarea preferințelor și dorințelor tuturor iubitorilor informaticii.

Dorim să încheiem, dragi prieteni, cu mulțumirile noastre pentru atenția cu care ați susținut permanent rubricile de informatică din revista „Știință și tehnică”, cu credința că, în viitor, vom colabora la fel de strîns pentru a recupera în cel mai scurt timp inerțiile obiective ce ne-au grevat activitatea în anii ce au trecut!

Michael Godev

● PUBLICISTICA'

1989

1986

Animație pe calculator la ora întrebărilor/9
 Discul între muzică și... calculator/10
 Calculatorul a învățat... să navigheze/11
 Ce înseamnă proiectarea asistată de calculator/12

1987

Calculatorul optic (I): Electronul sau fotonul?/1
 Proiectarea asistată de calculator: periferice grafice/1; 2
 Dificila genză a tranzistorului optic/2
 Interlocutor — calculatorul/3
 Proiectarea asistată de calculator în automatizările industriale/3
 Aventura unui microprocesor — Z8000/3; 4; 5; 6
 Super și hipercalculatoarele(grupaj)/4
 Universul copilăriei în „Era informaticii“/6
 Instrumente grafice utilizate în CAD/7
 Televiziunea interactivă — o nouă formă de dialog om-mașină/8
 Școala de ora informaticii (grupaj-anchetă)/8
 Inteligența artificială — mit sau realitate?/1
 Cinematograful asistat de calculator/12
 Calculatoarele personale între 16 și 32 de biți/12

1988

O nouă generație de calculatoare personale: PS/2/1; 2
 Marile performanțe ale micilor dispozitive/1; 2; 4
 A transmite bit cu bit/2; 3
 Informatica la genul feminin/2
 O perspectivă asupra programării/3
 Încotro se îndreaptă cercetările în domeniul inteligenței artificiale?/4
 De la circuite la calculatoare memoriale/4; 5
 Aventura matematicii și unealta ei indispensabilă: calculatorul/5; 7; 9; 12
 Jucăriile intră în laboratoare/6
 Supercalculatoarele, mașinile ultimului deceniu?/7
 Componente pentru calculatoarele viitorului (grupaj)/8
 Informatica medicală (grupaj)/10
 Simularea prin animație interactivă/10
 Imperiul contraatacă: IBM și Personal System 2/12.

Imperiul contraatacă IBM și Personal System 2/1; 2
 Supercalculatoarele bătrînului continent/1; 2
 Calculatoarele RISC: un risc al informaticii?/3
 Componente și interconexiuni (grupaj)/4
 NeXT — mașina anilor '90?/5
 Dincolo de ecran și tastatură: codurile informaticii/6
 Muzică în cod-mașină sau portativul pe 8 biți/7
 Periferice cu valențe grafice/7; 8; 9
 Între om și calculator: noi dispozitive de introducere a datelor/8; 9

● LIMBAJE DE PROGRAMARE

1986

Consultați BASIC (precedată în anul 1985 de rubrica intitulată „Inițiere în BASIC“)/1; 3; 5
 Curs de inițiere în LOGO/1; 2; 3; 4; 5; 6
 Pilot și limbajele de autor/10.
 Lisp/11; 12

1987

Lisp/1; 2; 3; 4; 5; 6; ; 8; 9; 10; 11; 12
 Consultați BASIC/3; 7; 10; 11
 Spre limbajul natural?/5

1988

Consultați BASIC/1; 2; 3; 4; 9; 10
 Lisp/1;2;3
 Instrumente grafice utilizate în CAD/1
 Să învățăm dBASE/4; 5; 6; 7; 8; 9; 10; 11; 12
 Infoclub/4; 5; 6; 7; 8; 9; 10; 11; 12
 Wordstar/6; 7; 8; 9

1989

Să învățăm dBASE/1; 2; 3; 4; 5; 6
 Infoclub/1; 2; 3; 4; 5; 6; 7; 8; 9; 10; 11; 12
 Școala la ora informaticii/3; 4; 5
 afl Programe comentate/6; 7; 8; 9; 10; 11; 12
 Introducere în Pascal (continuă și în prezent)/7; 8; 9; 10; 11; 12

CUPRINS CONTENTS

3	Editorial — Editorial în sfârșit... — At last...
4	Biblioteca ST de informatică ST informatics library
6	Curriculum vitae
8 ●	Flash — Flash Stații de lucru ingineresti — Engineering workstations
15 ▲	Actualitatea PC — PC news Cuploare grafice — Graphic controllers
24 ●	Ghidul utilizatorului — User's Guide Utilizarea sistemului TURBO Pascal — TURBO Pascal
25 ■	Spot — Spot Placa de bază IBM PC-XT — Mother board IBM PC-XT
38 ◆	Laborator Spectrum — Spectrum laboratory Utilizarea rutinei de citire din memoria ROM
42 ▲	Laborator Commodore — Commodore laboratory Informatica — un joc serios!
43 ▲	Atelier Spectrum — Spectrum Practice Interfață serială pentru calculatoarele compatibile HC-85
45 ■	Aplicații pentru toți — Applications for all Admitere — software educațional — Software for education
47 —	Contact — Contact
49	Umor



**Valeriu
IORGA**

Născut în Lugoj, în anul 1943. Absolvent al Facultății de Energetică, secția Automatică.

Hobby: muzica, literatura SF, cuvinte încruciate.

Repartizat, după absolvirea facultății, în anul 1966, ca asistent preparator la catedra de calculatoare a facultății de Automatică din cadrul Institutului Politehnic București. Ocupă prin concurs posturile de asistent titular (1971), șef de lucrări (1975) și conferențiar (1990).

În anul 1982 a obținut titlul științific de doctor inginer, cu o lucrare din domeniul conducerii numerice directe a proceselor tehnologice. În toată această perioadă desfășoară activitate didactică de predare de cursuri în domeniul calculatoarelor cum ar fi: „Programarea calculatoarelor”, „Calcul numeric” și „Tehnici de programare”.

În activitatea de cercetare are preocupări legate de sisteme de operare și aplicații în timp-real, teoria algoritmilor.

A desfășurat o bogată activitate publicistică, avînd tipărite 11 cărți și manuale precum și numeroase articole în domeniul programării calculatoarelor, inclusiv în revista „Știință și tehnică”. Pentru viitor așteptăm cu încredere noi apariții editoriale în domeniul calculatoarelor și a aplicațiilor acestora.

Născută în București, în anul 1954. Absolventă a Facultății de Automatică, secția calculatoare.

Hobby: baletul și muzica clasică.

Nu după mulți ani de la absolvirea facultății, alături de foarte mulți tineri dar și specialiști cu experiență, s-a integrat în rîndul cercetătorilor din cadrul Institutului de Tehnică de Calcul din București. În cei 11 ani de cînd își desfășoară activitatea în colectivul acestui institut și-a orientat activitatea de cercetare științifică, în principal, în domeniul de specialitate cum ar fi: informatica medicală, baze de date, grafică și proiectare asistată de calculator ș.a.

Nefiind la prima „încercare” publicistică sperăm într-o colaborare număr la număr la această nouă apariție editorială intitulată atît de sugestiv „Infoclub”.



**Doina
ISTRĂTESCU**

Născut în București, în anul 1951. Absolvent al Facultății de Fizică.

Hobby: pescuitul, fotografia, electronica, filatelia și, bineînțeles, calculatorul.

Profesor de fizică la Școala nr.25 din București și-a început activitatea legată de această mare pasiune — informatica — printr-o colaborare la primul număr al „Buletinului de informatică” editat de Inspectoratul Școlar al municipiului București, la revista „Învățămîntul liceal și tehnic profesional”, precum și la sesiunile de comunicări științifice ale „Societății de științe matematice” — secțiunea de informatică. Ulterior, a trecut de la simpla colaborare la redactarea (împreună și cu alți profesori) a buletinului de informatică al inspectoratului școlar din capitală.

Pentru viitor își propune editarea unei reviste de informatică orientată spre problematica școlară.

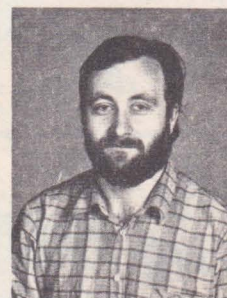


**Virgil
IONESCU**

Născut în București, în anul 1958. Absolvent al Facultății de Geodezie.

Hobby: Karate — antrenor centura neagră, literatura și filmele SF.

În tot acest timp, are o pasiune constantă, calculatorul, căruia îi dedică cea mai mare parte a timpului. Cu mult entuziasm, alături de un grup de „fani”, în anul 1987, înființează în cadrul Casei de Cultură a studenților, Clubul de calculatoare din București, deschis tuturor posesorilor de „personale” din București și din țară. În ianuarie 1990, Clubul de calculatoare devine Clubul Român de Calculatoare (după cum reiese și din materialul pe care îl prezentăm în acest supliment) condus și animat de același Călin Obretin. În intențiile viitoare ale clubului: contactarea cluburilor similare din străinătate (au fost deja realizate contacte foarte promițătoare), editarea unei reviste a clubului (noi, în mod sincer, sportiv, le dorim succes!), punerea la punct a unei rețele și multe altele.



**Călin
OBRETIN**

Curriculum vitae



**Liviu
DUMITRAȘCU**

Născut în România cu un număr de ani în urmă!! Absolvent al Facultății de Mecanică Fină.

Hobby: muzica, atât ca ascultător, cât și ca interpret.

După absolvirea facultății, timp de 10 ani activează în cadrul CEPECA unde predă cursuri privind limbaje de programare a calculatoarelor.

După absolvirea în Franța a unor cursuri de specializare în informatică și tehnică de calcul îl găsim ca șef de lucrări în cadrul Institutului de Petrol și Gaze din Ploiești unde își desfășoară activitatea și în prezent. Între timp, în anul 1986 își susține și teza de doctorat în domeniul automatizării instalațiilor petroliere.

Activitatea publicistică, în calitate de autor sau coautor s-a materializat în cadrul Editurii Academiei și Editurii Tehnice prin numeroase articole și prin șase volume care s-au bucurat de o mare popularitate, unul dintre acestea fiind editat și în limba engleză. Iată câteva titluri: „Învățăm microelectronica interactivă”, „Tehnici de construire a programelor cu structuri alternative”, „Elemente de inteligență artificială pentru conducerea operativă a producției”.

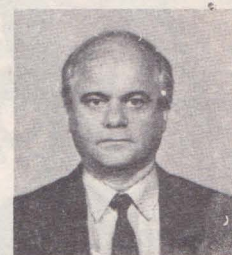
Are numeroase proiecte de viitor pe care însă nu le vom dezvălui deocamdată. Poate într-un număr viitor al suplimentului nostru de informatică și calculatoare!

Născut în comuna Stoenеști, jud. Olt, în anul 1937. Absolvent al Facultății de Matematică.

Hobby: umorul și... șahul.

Doctor matematician, binecunoscut de peste 25 de ani informaticienilor din țară, autor a numeroase lucrări, circa 14 volume apărute în Editura Didactică și Pedagogică, Editura Științifică și Enciclopedică, precum și în Editura Tehnică, duce de peste două decenii o susținută activitate didactică și de cercetare atât în învățământul preuniversitar cât și în cel universitar.

Activitatea de cercetare științifică și publicistică s-a materializat în peste 20 de lucrări de cercetare publicate în țară și în străinătate referitoare la limbaje de programare, instruire asistată de calculator, cercetări operaționale ș.a. Sperăm într-o activitate publicistică viitoare pe măsura numelui și a experienței sale didactice atât în paginile revistei „Știință și tehnică” dar mai ales în paginile acestui supliment de informatică și calculatoare care se așteaptă pe viitor într-o apariție trimestrială la îndemina tuturor iubitorilor acestui domeniu mirific al științei.



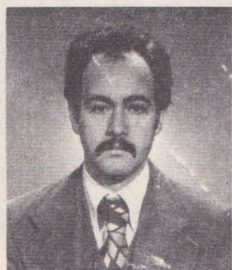
**Stelian
NICULESCU**

Născut în București, în anul 1957. Absolvent al Facultății de Electronică și Telecomunicații.

Hobby: fotografia, muzica progresivă și... speologia.

Imediat după terminarea facultății a lucrat un număr de ani în domeniul exploatării emițătoarelor de Radio-TV, apoi în proiectarea acestora. Pasiunea nedeclarată însă pentru informatică și calculatoare a fost mai puternică, fapt pentru care, în ultimii cinci ani, îl găsim în rândul cercetătorilor Institutului de Tehnică de Calcul din București.

Specializarea în domeniul atât de importante ale informaticii cum ar fi bazele de date, grafică și proiectare asistată de calculator, rețele de calculatoare și sisteme de operare, dublată de un real talent publicistic îi deschid un câmp larg de acțiune pe această linie. Ca urmare, în numerele viitoare ale suplimentului de informatică și calculatoare al revistei „Știință și tehnică”, sperăm ca împreună cu colega sa de institut și nu numai... să vadă lumina tiparului o serie de alte materiale publicistice legate de limbaje neprocedurale în baze de date, algoritmi de grafică precum și un minicurs de „C”.



**Eugen
GEORGESCU**

Născut în București, în anul 1951. Absolvent al Facultății de Cibernetică.

Hobby: tenis și bridge.

După absolvirea facultății, în anul 1975, a activat în cadrul Institutului de Tehnică de Calcul din București, conducând un colectiv cu preocupări în domeniul calculatoarelor personale și o temă de cercetare privind utilizarea calculatoarelor în rocesul de educație.

Are o intensă activitate publicistică în domeniu, fiind prezent prin o multitudine de articole în paginile revistelor de popularizare a științei pentru copii și tineret, deci și a revistei „Știință și tehnică”, fiind în același timp o prezență activă în numeroasele cercuri de calculatoare pentru copii. Este autor sau coautor al unor lucrări din care amintim câteva titluri mai semnificative: „Dialog cu viitorul”, „Partenerul meu de joc — calculatorul”, „Elemente de informatică pentru cercurile tehnico-aplicative”, „Buletin periodic de informatică” ș.a.



**Ion
DIAMANDI**



digital

Flash



Stații de lucru ingineresti

Stațiile de lucru ingineresti reprezintă un domeniu relativ nou, apărut după cel al calculatoarelor personale din care s-a desprins. În categoria stațiilor de lucru ingineresti sînt cuprinse stațiile grafice, echipamentele pentru: proiectarea asistată de calculator (CAD), fabricația asistată de calculator (CAM), ingineria asistată de calculator (CAE) etc. O dată cu apariția și producerea calculatoarelor personale de mare performanță, stațiile de lucru ingineresti cunosc și ele o mare dezvoltare.

De remarcat faptul că în anul 1986 vânzările de stații de lucru ingineresti au depășit pe cele ale calculatoarelor personale, piața SUA în acest an înregistrînd o creștere de 40%. De altfel, precedentul fusese creat: în anul 1985 fuseseră deja vîndute în SUA 21 000 de stații de lucru ingineresti care reprezentau o valoare de 735 milioane dolari. De fapt drumul stațiilor de lucru ingineresti fusese schimbat de firma Apollo în anul 1981 care, ulterior, a și dominat această piață cu un procent de 30—40% din vînzări. În anul 1984, însă, și-a făcut o apariție meteorică firma SUN care, în perioada 1985-1986, a preluat conducerea acestei piețe.

Deoarece tendințele arată că bătălia viitoarelor calculatoare se va da între calculatoarele de birou și stațiile de lucru ingineresti se cuvine să se facă o analiză mai amănunțită a acestora, prin aflarea răspunsurilor la o serie de întrebări:

- ce sînt sau ce se înțelege, de fapt, prin stații de lucru ingineresti?
- care sînt producătorii de stații de lucru ingineresti?
- care este situația pieței?
- care sînt caracteristicile care fac ca stațiile de lucru să se constituie într-o formă unică de prelucrare a informațiilor?
- care sînt aplicațiile ce se pretează abordărilor cu stații de lucru ingineresti?

Iată o posibilă definiție a unei stații de lucru ingineresti bazată în primul rînd pe tehnologie și componență: unitate centrală pe 32 de biți (sau mai mult); memorie internă de 2-4 Mo și, de obicei, suport de memorie virtuală; display integral cu o rezoluție de minim 640 x 480 pixeli; interfețe pentru rețea (de obicei Ethernet); sistem de operare multitasking (nefiind neapărat necesar un sistem de operare multiutilizator, deși unele stații prezintă acest tip de sistem).

Există firește și alte definiții care nu exclud elementele din lista expusă, permițînd stației să prelucreze unul sau mai multe taskuri tipice. O dimensiune mare a registrelor memoriei interne precum și un spațiu de adresare suplimentar (eventual virtual) sînt necesare pentru rularea unor aplicații tipice ingineresti. Rezoluția înaltă depinde direct de magistrala microprocesorului și ea permite o interfață cu utilizatorul prin ferestre, ceea ce are ca urmare creșterea productivității muncii la utilizarea stației. Ușurința integrării într-o rețea este prima armă împotriva formării unor „insule de prelucrare informațională” și, în sfîrșit, sistemul de operare multitasking leagă toate aceste elemente împreună.

Exemple tipice de mașini care se po-

trivesc definiției date sînt stațiile care aparțin liniei VAX a firmei Digital Equipment (DEC) precum și produsele cele mai utilizate ale producătorilor tradiționali ca SUN și Apollo. De remarcat că există mai multe firme (InterAct, InterPro) unite sub numele de Intergraph care produc stații pe bază de VAX sau MicroVax și care dețineau împreună în anul 1988 circa 29% din piața stațiilor de lucru.

DEC a intrat pe piața stațiilor de lucru grafice în anul 1984 cu prima stație VAX pentru satisfacerea necesităților impuse de aplicațiile CAD/CAM/CAE. Deși cu performanțe ceva mai scăzute și costuri mai ridicate, linia de stații VAX a cîștigat teren în mod continuu, devenind a treia putere după SUN și Apollo. În anii 1985-1986 modelele VAX station II și VAX GPX reprezentau unele din cele mai căutate produse pe piața mondială oferind, în afara facilităților grafice specifice domeniului și celelalte programe aplicative funcționînd sub sistemul de operare VMS (după unele estimări, DEC a livrat în 1985 echipamente CAD/CAM pentru circa 1/3 din piața mondială). De fapt, în acea perioadă, avîndu-se în vedere faptul că producătorii declarau mărirea capacității și performanțelor grafice, se prognoza pentru stațiile de lu-

cru cu grafică în 3D o creștere de 60% pentru 1988. Dar, Charles D. Wise, software manager pentru VAX/VAM, a prevăzut un conflict între stațiile VAX și calculatoarele VAX multiutilizator: „DEC are un interes vast la mini și supermini și, din această cauză, nu are de gând să facă o afacere din stațiile de lucru care ar reduce vânzările pentru VAX 8 000”. Urmarea a fost o scădere simțitoare a procentului pe piața stațiilor de lucru deținută de DEC. Deci, la sfârșitul anului 1987 piața stațiilor de lucru ingineresti se prezenta astfel: Sun Microsystems —33%; Apollo — 18%; DEC — 3%; HP— 2%.

Iată și prognoza Dataquest la acea vreme în ceea ce privește performanțele stațiilor (viteză de lucru): 1987: 7-10 MIPS; 1988: 16-20 MIPS; 1989: 30-40 MIPS. Un exemplu de stație DEC la nivelul anilor 1987-1988: stația VAX 3 200 cu performanțe de 10 MIPS.

Pe piața stațiilor de lucru au apărut și producători mai puțin tradiționali. Astfel, Hewlett Packard care a venit din domeniul minicalculetoarelor, precum și Tektronix, din domeniul terminalelor, sînt două exemple. Hewlett Packard de mai mulți ani are un rol, deloc neglijabil pe piața stațiilor de lucru. Deși a vîndut un număr mare de unități, unele din acestea nu intră în definiția dată anterior stațiilor. Astfel, deși a vîndut bine seria 9 000 (care se potrivește standardului, performanțelor și prețului), Hewlett Packard rămîne mai mult un furnizor de nișă pe piața stațiilor de lucru, majoritatea acestora intrînd în categoria sistemelor la cheie. Tektronix a început să fabrice a doua sa serie de stații de lucru. Prima încercare (seriile 61XX și 62XXs) a fost o „semireușită” prezentînd cîteva probleme atît de producție — cu circuitul integrat National 32 000 — cît și cu software-ul. Concepția firmei este că piața tradițională de terminale grafice se va transforma rapid în piața de stații de lucru. Date fiind mărimea și forța sa financiară Tektronix rămîne o firmă a cărei evoluție pe piața stațiilor de lucru trebuie urmărită.

Un alt producător important este, desigur, IBM. Însă singurul său produs care se potrivește definiției date este stația 6150 PC/RT (microcalculator de tip RISC cu set redus de instrucțiuni), cu toate că și PS/2 Model 80 cu sistemul de operare OS/2 și Model 55 cu microprocesor 80 385 SX pot fi considerate adevărate stații de lucru. Deși a făcut îmbunătățiri stației 6150 PC/RT, aceasta este departe de a reprezenta un succes răsunător. Stațiile PC/RT sînt declarate de firmă echipamente profesionale pe 32 de biți, cu

sistem IBM/AIX, prețul lor fiind destul de mare mai ales prin faptul că sînt realizate cu microprocesoare IBM originale. Iată și membrii familiei 6150 PC/RT împreună cu prețurile corespunzătoare anilor 1988-1989: IBM PC RT Model 1 (desktop) — 13 000 dolari; IBM PC RT Model 2 (floor-standing) — 40 000 dolari; IBM PC RT 130 (desktop) — 23 000 dolari; IBM PC RT 135 (floor standing) — 30 000 dolari; IBM PC B 35 (+ IBM 5080 Graphics) — 32 000 dolari.

Cu toată această gamă variată PS/2 Model 80 și mult discutatul Model 90 se încadrează mai bine în definiția dată stațiilor de lucru. Dată fiind baza instalată de PC-uri, aceste mașini pot muta IBM-ul într-o categorie foarte serioasă de producători de stații. De asemenea, calculatorul personal MAC II cu sistemul de operare A/UX poate aduce și firma Apple în lumea stațiilor de lucru.

Liderul stațiilor de lucru ingineresti, SUN, și-a întinerit toată gama de produse introducînd în 1989 zece noi modele pe linia mașinilor cu microprocesor 68 000 precum și pe cele cu arhitectură cu set redus de instrucțiuni (RISC). Iată două din produsele lansate împreună cu caracteristicile lor:

●SUN D3/80 cu microprocesor 68030 la 20 MHz, memorie internă de 4 Mo și sistem de operare UNIX sau MS/DOS. Performanțe: 3 MIPS;

●SUN D3/400 cu microprocesor 68030 la 33 MHz, memorie internă de 8-12 Mo. Performanțe: 7 MIPS. De remarcat că prețul este comparabil cu cel al unui calculator personal Macintosh II sau PS/2 Model 70.

De la aplicații ingineresti la birotică

Deși în mod tradițional stațiile de lucru au fost utilizate mai mult în aplicații ingineresti, un mare număr de stații sînt utilizate în mod curent în alte domenii: industria prelucrătoare folosește stațiile pentru controlul proceselor, analiza calității, gestiunea stocurilor etc.

În aceste cazuri posibilitățile de multitasking și interfețele grafice sînt cerute de controlul în timp real. Domeniul publicațiilor, în particular influențat puternic de revoluția electronică, este un consumator tot mai mare de stații de lucru. Cîm prețul stațiilor tinde să scadă mult, ele vor înlocui calculatoarele personale ca resurse desk-top pentru cei care au posibilități financiare mai mari. De exemplu, tra-

diționala piață a automatizării birourilor (biroticii) dominată acum de PC-uri va fi caracterizată în curând de echipamente orientate pe interfețe iconice, cu meniuri controlate și ferestre etc., oferind astfel o productivitate de grup mai bună și mai puține posibilități de erori. Pentru întreprinderile medii, puterea de rețea a stațiilor poate rezolva problemele de achiziționare a informațiilor. Pentru manageri stațiile pot reprezenta o adevărată revoluție, în sensul utilizării unor aplicații ca: **sisteme de raportare „on line“** care nu necesită probleme deosebite de aptitudini și experiență în utilizarea tastaturii.

O altă aplicație însemnată a stațiilor o reprezintă așa-numita „**vizualizare a software-ului**“. Există o relativ nouă direcție de dezvoltare, provenită din prezentări grafice, deosebit de utilă în cercetarea științifică fundamentală pentru realizarea de modele în domenii ca: dinamica fluidelor, prognoze meteorologice, fizica nucleară etc. Vizualizarea software este utilizată pentru a memora și prelucra volume mari de informații produse de sisteme de prelucrare a imaginii și de supercalculatoare. Stațiile de lucru prezintă un prilej unic, datorat posibilităților grafice și de cuplare în rețele, în vederea controlului în timp real a acestor resurse.

Stații de lucru sau... calculatoare tradiționale?

Răspunsul la această întrebare este aproape imediat (sau chiar evident), deoarece diferența principală se referă la tipul interfeței grafice utilizator. Pentru calculatoarele personale compatibile IBM precum și tradiționalele sisteme de prelucrare pe loturi utilizează interfețe utilizator pentru linii de comandă similare. Pe aceste sisteme grafica este dedicată unor aplicații specifice ca, de exemplu, cele de prezentări grafice. Stațiile de lucru oferă un avantaj major față de prelucrarea convențională (de tip personal sau pe loturi) existând totuși două obstacole care împiedică (încă) răspunderea lor pe scară largă și anume: costul și lipsa uneltelor de dezvoltare de aplicații. Deși costul stațiilor a scăzut simțitor ele sînt încă scumpe față de calculatoarele personale și în special față de terminale.

Cea mai importantă posibilitate de scădere a prețurilor poate veni însă din ideea de a privi stațiile ca „termi-



COMPOQ DESKPRO 386/20TM

nalele super inteligente“ și nu drept calculatoare din partea de jos a gamei. Dacă producătorii vor începe să ofere terminale cu procesoare adecvate, memorie suficientă, rezoluție grafică înaltă precum și posibilități de cuplare la rețele, atunci acestea vor putea realiza aplicațiile specifice stațiilor de lucru.

O astfel de stație poate fi realizată pe o singură placă așa cum sînt deseori proiectate terminalele. În acest caz, costul va fi (virtual) similar cu al unui terminal, diferența fiind legată de mărimea memoriei (care în prezent este ieftină) și de microprocesorul utilizat.

„Unelte“ grafice folosite în prezent sînt subrutine utilizate de programatori în urmă cu cel puțin 15 ani pentru dezvoltare de aplicații pe sisteme mari de calcul (mainframes) și pe minicalculatoare. Mai recent, standardele CORE și GKS au permis aplicații pe loturi cu care s-a utilizat puterea locală în grafică a terminalelor. După apariția și dezvoltarea calculatoarelor personale performante, acestea nu au integrat display-uri cu grafică interactivă și cu facilități grafice deosebite (de exemplu, operații raster). Acest fapt precum și puterea de prelucrare și capacitatea de memorie prea mici pentru a realiza transformări de vectori au făcut ca PC-urile să nu poată fi utilizate în aplicații grafice de anvergură. Stațiile de lucru, însă, pot oferi aceste facilități, iar producătorii caută un model pentru standard.



SPARC STATION I

Din perspectiva stațiilor de lucru ingineresti: calculatoare RISC

Calculatoarele cu set redus de instrucțiuni (RISC) au apărut pe piața stațiilor de 32 de biți, reușind în mai puțin de 3 ani să realizeze o adevărată invazie pe această piață (vezi „Știință și tehnică” nr. 3/1989). Toate calculatoarele RISC de astăzi exprimă ideile încorporate în trei sisteme care au reprezentat un pionierat la vremea loc. Este vorba de sistemele realizate de echipele de cercetători de la IBM, Stanford și Berkeley și care au văzut lumina zilei în perioada anilor 1980-1983. Cercetările efectuate la acea vreme au vizat obținerea unei viteze de prelucrare superioare la un preț redus. Primele sisteme realizate cu aceste concepte imitau, de fapt, primele calculatoare ale anilor '50. Acestea puneau la dispoziție puține instrucțiuni simple care funcționau cu date din memorie sau aduse în memorie. Nașterea tranzistorului și apoi a circuitelor integrate a permis realizarea în anii '60 și '70 a calculatoarelor cu circuite foarte rapide și unități centrale capabile să stocheze sute de instrucțiuni. Dar unitățile centrale realizate cu tehnologie pe siliciu erau foarte scumpe iar din acest motiv proiecții vizau alte materiale pentru memoria de lucru. Astfel au apărut feritele, memorii mai ieftine

(dar și mai lente) folosite pentru memorarea datelor.

Cercetările au vizat, apoi, creșterea vitezei prin reducerea drastică a numărului de transferuri de date extrase din memoria centrală, cu alte cuvinte prin scrierea unor instrucțiuni mai complexe. Au apărut astfel **calculatoarele cu set complex de instrucțiuni (CISC)**. Pentru a nu încărca excesiv unitatea centrală s-a procedat la realizarea unui microcod (memorie de control) situat lângă memoria centrală care să înglobeze instrucțiunile complexe. A rezultat o memorie centrală mai mică, însă operațiile erau mai lente deoarece fiecare instrucțiune complexă era decodificată atât de unitatea centrală cât și de memoria de control.

La sfârșitul anilor '70 situația începuse să se modifice. Memoriile se ieftiniseră iar utilizarea memoriilor cache de viteză foarte mare a avut ca rezultat faptul că vitezele memoriilor au început să depășească vitezele memoriilor centrale.

În acest timp, prima cercetare realizată asupra modului în care programele erau executate de calculatoarele CISC (cercetare pornită în scopul sprijinirii dezvoltării de compilatoare) a dezvăluit un rezultat șocant: **numai 20% din setul de instrucțiuni îndeplineau 80% din operații**. Aceasta însemna că multe din instrucțiunile complexe erau, de fapt, redundante. Soluția prevăzută de IBM, Berkeley și Stanford a fost reîntoarcerea la arhitecturile construite cu micile inele de ferită.

Astăzi arhitecturile RISC au instrucțiunile implementate în partea de hard iar orice operație complexă este îndeplinită prin intermediul unui șir de instrucțiuni simple (mai lente). Se utilizează, de obicei, pînă la 32 de registre de uz general precum și memorii rapide (de tip cache) de dimensiuni mari. Viteza lor este îmbunătățită prin instrucțiuni de tip conductă (cu format fix, lungime de 32 de biți și puține moduri de adresare). Deși acest mic procent de instrucțiuni complexe este prelucrat mai lent, rezultatul final este reprezentat de o importantă creștere în viteza de prelucrare. Specialiștii în arhitecturile cu set redus de instrucțiuni prevăd că **viteza acestora va continua să se dubleze la fiecare 12-18 luni pentru următorii ani**, mai ales că se presupune și folosirea unor tehnologii performante ca: CMOS, ECL sau arseniură de galiu. Se observă astfel un avantaj net față de cel promis de microprocesoare ca Intel 80846 sau Motorola 68040. De aceea se prevede că arhitecturile RISC vor pătrunde în

circuitele P (Intel 80x86 și Motorola 680x0). Deja IBM a introdus această tehnologie în stația 6150 și PC RT precum și la unele din minicalculatoarele pe care le produce. Digital Equipment lucrează cu tehnologiile RISC „în spatele ușilor închise”. Interesul arătat de aceste uriașe firme este datorat faptului că în prezent calculatoarele cu arhitectură RISC oferă o viteză de 100 de ori mai mare față de VAX 11/780.

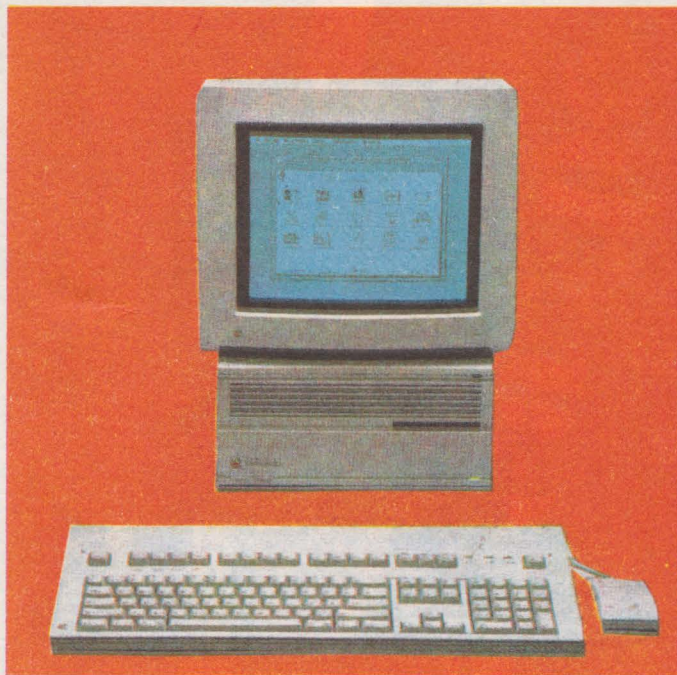
Sigur, există și probleme. Cu cât sînt mai multe firme care comercializează echipamente cu arhitectură RISC încercînd să facă din ea un standard, cu atît mai grea va fi selectarea unei opțiuni.

Cîntărind factorii RISC

Din punct de vedere al modului în care sînt realizate, arhitecturile RISC sînt, în același timp, atît diverse cît și complexe. De exemplu, echipele de la IBM și Stanford împing o serie de procese spre software în scopul optimizării compilatoarelor. În alte locuri (Berkeley) se preferă implementarea mai multor operații complexe în hardware prin maximizarea registrelor unității centrale. Astăzi proiectanții de arhitecturi RISC au de ales între cîteva alternative și anume: IBM 801 (care prezintă 120 de instrucțiuni), Stanford RISC (139 de instrucțiuni) și Berkeley MIPS (55 de instrucțiuni). Se observă că nu există o mărime standard pentru setul de instrucțiuni. Mai mult, unele mașini realizează operații în virgulă mobilă prin software, în timp ce altele le inserează în partea de hard sau folosesc coprocesoare de virgulă mobilă.

Toate aceste diferențe trebuie să fie, însă, invizibile utilizatorilor finali ai stațiilor de lucru, aplicațiile UNIX standard putînd fi modificate pentru a rula pe orice mașină prin compilatoare standard. În schimb, un circuit „standard” este mult mai dificil de impus. Se pare, însă că Sun este pe cale să realizeze acest lucru cu produsul său SPARC (Scalable Processor Architecture). SPARC este „fiul” lui RISC (Berkeley), rivalii săi fiind Precision Architecture (HP), Pyramid precum și unele produse IBM.

Sun și-a conceput arhitectura astfel încît, dacă la început setul redus de instrucțiuni era identificabil în special pe stațiile de lucru Sun, în prezent apare din ce în ce mai frecvent la calculatoare personale, eclipsînd tehnologia CISC prezentă, în special, la microprocesoarele Intel. Setul redus de instrucțiuni prezintă mai puține instrucțiuni încorporate în hardware avînd o



APPLE MACHINTOSH CX

schemă mult mai eficientă. Din aceste motive este mai rapid față de CISC însă mai costisitor.

SPARC prezintă o unitate centrală care constă într-o unitate de calcul cu numere întregi (Integer Unit) și o unitate de calcul în virgulă mobilă concurentă (Floating Point Unit) care poate fi implementată în același circuit. Are la bază tehnologia RISC (numai 50 de instrucțiuni) și optimizează calculele în virgulă flotantă în comparație cu rivalii săi IBM și MIPS. A fost proiectat în mod special pentru rula programelor în limbaj C și sub sistem de operare UNIX.

De remarcat faptul că deja au început să apară copii (imitații) după arhitecturile SPARC („clones”-uri) care prezintă prețuri de 10% mai mici față de cele originale. De asemenea, producătorii de circuite RISC au început să se alinieze la standardul SPARC, existînd deja, 5 firme importante (Texas Instruments, Fujitsu, LSI Logic, Bipolar Integrated Technology și Cypress) care oferă seturi de circuite compatibile SPARC. Ultimul set de circuite compatibile SPARC oferit de firma Cypress prezintă ca performanțe 24 MIPS față de 6 MIPS comparativ cu microprocesorul Intel 80386 la 25 MHz.

Copiile după arhitecturile SPARC rulează sub sistem de operare UNIX fiind puse la dispoziție și circa 400 de programe de aplicații. Se prognozează că în anul 1990 prețul acestor stații de lucru va fi cuprins între 5 000 și 10 000 de dolari.



ARCHIMEDES 410

lată câteva tipuri de cele mai cunoscute stații de lucru SPARC împreună cu cele mai importante caracteristici tehnice ale lor:

- **SPARC Station 1** prezintă o viteză de 12,5 MIPS (1,4 MFlops) iar pentru îmbunătățirea acesteia se poate atașa un coprocesor în virgulă mobilă. Capacitatea de memorie internă este de 8-16 Mo, iar a celei de masă de 160, existând și unitate de bandă backup. Are în componență un procesor (accelerator) pentru prelucrarea imaginilor și sînt puse la dispoziție circa 500 de pachete de programe (Sparcware).

- **SPARC Station 300** prezintă o viteză de 16 MIPS (2-6 MFlops), memorie internă de 8-40 Mo iar memoria de masă 1,3 GO.

- **SPARC Station 370** are o capacitate de memorie internă și de masă sporite (56 Mo și respectiv 5,6 GO).

lată și caracteristicile tehnice ale unei stații de lucru cu circuit RISC realizată de firma Tektronix: D88 realizată cu microprocesor Motorola 88 000; viteză: 14-17 MIPS (7-12 MFlops); memorie internă: 8 Mo.

Cu toate că arhitecturile SPARC par să devină un standard și alte firme dezvoltă arhitecturi noi pornind de la cele cu set redus de instrucțiuni. Motorola, HP și DEC realizează arhitecturi RISC optimizate pentru rularea sub UNIX. Motorola 88 000, de exemplu, prezintă 51 de instrucțiuni și 32 de registre, iar unitățile de calcul în numere întregi și

în virgulă mobilă sînt conținute într-un singur circuit. Una din cele două unități de memorie rapidă (cache) standard este folosită pentru date iar celelalte pentru instrucțiuni. Pot fi incluse pînă la 8 unități de memorie cache astfel încît arhitectura este proiectată pentru diverse tipuri de aplicații.

Cum lista implementărilor RISC crește este de presupus că se va mări și cea a diferențelor între caracteristici. Iată și alte realizări: **2 900 (AMD)**, **Precision Arhitectura (HP)**, **ABAQ (Atari)**, **Acorn, RISC Machine (Acorn, Marea Britanie)**. Cel din urmă exemplu pare a fi optim pentru piața calculatoarelor personale. Prin utilizarea unei mici suprafețe de siliciu și a câtorva tranzistoare se preconizează o reducere importantă a prețului.

Cu aceste implementări este improbabil faptul că un singur ofertant RISC va putea să se impună singur ca un standard de factor în viitorul apropiat, mai ales în condițiile în care programele pentru utilizatori vor rămîne portabile la un nivel ridicat. Astfel ca și în alte cazuri, testul pentru arhitecturile RISC va fi reprezentat de capacitatea lor de a realiza aplicații specifice.

MAJORITATEA sistemelor RISC depind de compilatoarele lor, iar performanțele, de gradul de optimizare față de un anumit limbaj sau sistem de operare. Multe din aceste stații de lucru sînt realizate cu materiale și tehnologii avansate (de exemplu, versiunea cu arseniur de galiu) care le adaugă performanțe suplimentare. Dar se așteaptă ca sistemele cu set redus de instrucțiuni să producă cele mai mari performanțe atunci cînd setul redus de instrucțiuni va fi combinat cu prelucrarea paralelă care va pune probabil în frunte hibridul RISC **Transputerul firmei Inmos** și arhitectura **PRISM (Apollo)**. Arhitectura PRISM (Parallel Reduced Instruction Set Multiprocessing) prezintă circuite de 64 de biți și ajunge la o viteză de peste 100 de ori superioară lui VAX 11/780.

Piața de stații de lucru prezintă o rată de creștere de două ori mai mare de cea a întregii industrii de calculatoare. Primii doi mari în domeniul tehnicii de calcul și informaticii mondiale, IBM și DEC, recunosc importanța strategică a acestor calculatoare! În acest context este posibilă orice răsturnare de situație în care să cîștige piața o firmă ce este astăzi poate necunoscută, deoarece în lumea informaticii ultimului deceniu succesul poate veni uneori foarte repede! Dar și reversul lui...



actualitatea pe

Doina
ISTRĂTESCU

Eugen
GEORGESCU

Cuploarele grafice ale calculatoarelor compatibile IBM PC

Unul dintre elementele ce au contribuit în mare măsură la explozia pe piață a calculatoarelor personale a fost grafica, deoarece reprezintă un mod de comunicare sintetic, mai adaptat simțurilor umane și mai flexibil. O serie de noutăți pe plan tehnologic și conceptual au făcut ca unele modele de vîrf din clasa calculatoarelor personale să fie tolerate în clubul aristocratic al echipamentelor grafice profesionale, deși prețul lor de cost este mult mai scăzut. În literatura de specialitate această stare de fapt este caracterizată prin sloganul „high-end PC²low-end Workstation”, lansat evident de producătorii de PC-uri.

Pînă la apariția PC-urilor, a face grafică însemna în general o prelucrare de date pe o unitate centrală și o transmisie a datelor către un terminal folosind un protocol serial asincron sau sincron, la viteze cuprinse între 2 400 bps și 2 Mbps. Acest terminal a evoluat foarte mult ca posibilități, preluînd la nivel local aproape tot ce înseamnă grafică: trasări de elemente geometrice, umplerea (filling) conturilor cu diverse modele (pattern), prezentarea în spațiu tridimensional. Cu această ocazie s-a schimbat și denumirea din terminal grafic în stație grafică terminală, unul din cele mai populare modele fiind IBM 5080. Evoluția nu s-a oprit însă aici, stațiile grafice moderne fiind actualmente unități de calcul autonome care înglobează și unul sau mai multe procesoare pentru deservirea activităților de grafică.

Calculatoarele personale nu au mers însă pe această linie costisitoare, ele speculînd în mod inteligent o caracteristică proprie și anume aceea că memoria video este conținută în spațiul propriu de adresare al procesorului. Imaginea este formată în memoria video direct fără a mai fi nevoie de transmisie pe linie serială, obținîndu-se astfel o viteză de lucru foarte mare. Primele care au beneficiat de aceasta au fost jocurile și în general programele care necesitau animație. Scrierea direct în memoria ecran reprezintă numai o fază a procesorului. Pentru afișarea propriu-zisă această memorie video este citită secvențial de un bloc lo-

gic, independent ca funcționare de procesor, care realizează semnalele pentru monitorul TV conform standardului acestuia. Deși această funcție pentru cuplorul grafic nu este nici singura și nici cea mai complicată, ea este cea mai importantă și cu prioritate maximă la accesul resurselor.

Calculatoarele pentru care sînt valabile principiile mai sus enunțate sînt foarte numeroase, dar ne vom rezuma în cele ce urmează la prezentarea cuploarelor grafice (graphics adapter) cele mai utilizate în calculatoarele compatibile **IBM PC/XT/AT** și în noile modele **PS/2**.

Din punct de vedere istoric primele cuploare grafice ce au echipat calculatoarele din gama IBM PC/XT au fost **Hercules** și **CGA**. Cuplorul Hercules este de tip monocrom dar de bună rezoluție. Prețul de cost relativ redus al ansamblului cuplor-monitor i-a adus o oarecare popularitate la începutul anilor '80. Particularitățile sale constructive nu l-au făcut apt pentru dezvoltări ulterioare și practic a fost eliminat de cuplorul **CGA (Color Graphics Adapter)** care, deși oferea o rezoluție mai redusă, (320x200) aducea o nouă dimensiune, și anume culoarea. Acest cuplor a devenit practic un standard pentru calculatoarele XT deși poate fi întîlnit și în unele configurații ieftine de AT. Răspîndirea a fost impulsionată într-o măsură decisivă de avalanșa de software grafic pentru domenii ca **proiectarea asistată, statistici economice** și bineînțeles **jocuri**. Totuși

acesta a fost numai un început fiindcă CGA prezenta o serie de inconveniente majore cum ar fi: rezoluția redusă; număr de culori simultan afișabile foarte redus și gama de culori — din care se putea face selecția — aproape nesemnificativă.

La fel ca și la sistemele de televiziune, următoarele cuploare grafice au urmat calea firească a compatibilității: deși au apărut moduri noi de lucru, cuploarele răspundeau și la vechile pachete software. Se remarcă astfel în 1984 apariția cuplorului de tip **EGA (Enhanced Graphics Adapter)** care prin concepție se apropie de echipamentele grafice profesionale, deși performanțele continuă să rămână deficitare la unele capitole cum ar fi numărul de culori afișabile simultan, paleta din care pot fi acestea selectate și într-o oarecare măsură chiar rezoluția. Totuși această apariție este un eveniment deoarece introduce o serie de elemente funcționale ca planele de culoare, tabela de culori (Color Table) și ALU (Arithmetic Logic Unit) pentru scrierea pixelilor, elemente ce sînt specifice echipamentelor grafice performante.

În anul 1987, firma IBM introduce o versiune perfecționată denumită **VGA (Video Graphics Array)** care devine echipare standard pentru noua serie PS/2 la modelele 50, 60 și 80 unde este implementată direct (built-in) pe placa de bază (mother-board). Noul cuplor păstrează compatibilitatea cu modelele anterioare dar aduce un spor de rezoluție, crește fantastic paleta de culori și într-unul din modurile de lucru, poate afișa 256 de culori simultan dar din păcate la rezoluție redusă (320x200). Acest mod era impus de un nou domeniu de activitate și anume prelucrarea de imagini.

Deși nu se încadrează strict în tematica acestui articol vom încerca să ară-

tăm foarte pe scurt de ce la ora actuală numărul de culori simultane și paleta din care se pot selecta tind să devină mai importante decît rezoluția. Pentru a afișa cît mai veridic un obiect tridimensional pe suprafața bidimensională a unui ecran e necesară să dăm impresia de relief prin umbriri, deci prin gradații de culoare și saturație. O imagine TV de bună calitate acceptă aproximativ 5 000 de gradații. La un cuplor de tip VGA cele 256 de culori simultane, pot fi de fapt 256 gradații de luminanță ale aceleiași culori. Această problemă mai capătă o dimensiune dacă vrem să redăm și suprafața obiectului (lucioasă, mată, raster etc.) care înseamnă un sistem de microreflexii, deci de microgradații de culoare. Pentru a încheia prezentarea cuploarelor grafice de largă răspîndire ar mai trebui remarcat că pentru PS/2 Model 30 firma IBM a realizat cuplorul MCGA (Multi-Color Graphics-Array) care este compatibil cu CGA.

În continuare vom prezenta în sinteză principalele facilități ale cuploarelor menționate.

În cadrul tabelului 1 apariția de mai multe ori a unui cuplor înseamnă de fapt moduri diferite de lucru ale acestuia.

În această expunere au fost complet neglijate modurile de lucru alfanumerice. Și aici posibilitățile au evoluat foarte mult de la CGA la VGA. Astfel, numărul de linii de text afișabile pe un ecran a crescut de la 25 (CGA), 43 (EGA) pînă la 50 (VGA). Matricea de definire a unui caracter de 8x14 pixeli (CGA) a căpătat noi versiuni: 8x8 (EGA) și 8x16 (VGA). Se pot utiliza concomitent pînă la 4 tipuri de seturi de caractere (EGA/VGA) care pot fi definite și utilizate on-line. Această ultimă facilitate este importantă cînd se dorește utilizarea simultană a mai multor seturi de caractere (latin, chirilic, gre-

TABELUL 1

CUPLOR	REZOLUȚIE	NR. DE CULORI SIMULTANE	PALETA	MEMORIA VIDEO (kB)	BIOS
Hercules	720x348	2	2*	4	System
CGA	320x200	4	2**	11	System
CGA	640x200	2	2*	16	System
EGA	640x350	16	64	256	Extins
VGA	640x480	16	256 K	256	Extins
VGA	320x200	256	256 K	256	Extins
VGA***	800x600	16	256 K	256	Extins

* - Mod monocrom

** - Există doar două tipuri de combinații de coloristandard, denumite paletetele CMW (Cyan, Magenta, White) și GRY (Green, Red, Ywllow). La unele jocuri s-a dovedit că există și o a treia paletă nestandard.

*** - Acest mod nu este standard și poate fi implementat prin atacul direct la posturi al cuplorului cu condiția ca monitorul să aibă o frecvență de scanare orizontală de cel puțin 40 kHz (ex.: NEC Multisync II).

cesc, arab) sau a unor seturi de caractere naționale (ex.: ä, î, ș, ț în română). Descrierea acestor facilități va face obiectul unui alt articol.

Exploatarea posibilităților unui cuplor se poate face în general prin intermediul BIOS-ului și anume cu ajutorul întreruperii 10H. Pentru cuploarele EGA și VGA, BIOS-ul capătă o extensie care nu se mai găsește pe placa de bază ci în niște PROM-uri situate fizic chiar pe cuplor. Pentru aplicații performante, în care se dorește exploatarea la maxim a tuturor facilităților unui cuplor, iar factorul timp este prohibitiv, se recomandă folosirea directă a porturilor.

Pentru înțelegerea unor principii funcționale moderne și ținând cont de tendințele actuale, accentul prezentării se va pune pe EGA și VGA, referirile la CGA fiind sumare.

MEMORIA VIDEO

Este unul din elementele funcționale de bază în cadrul oricărui sistem grafic. Performanțele ei (timpul de acces și dimensiunea) influențează decisiv rezoluția iar forma de organizare are efect asupra numărului de culori afișabile simultan. Există două forme principale de organizare.

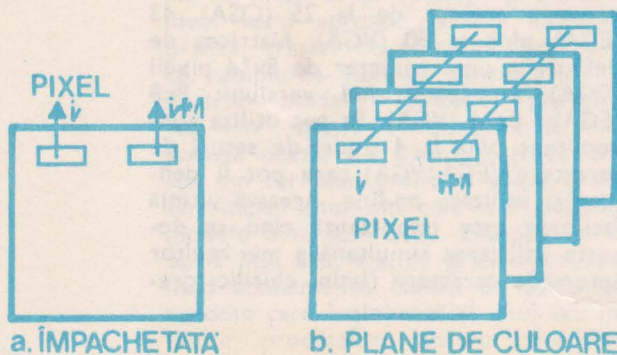


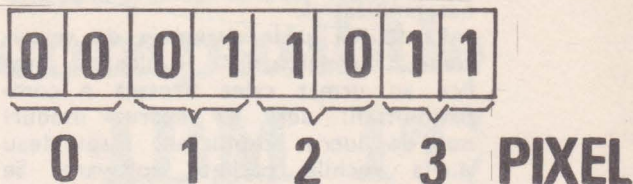
FIGURA 1

Prima versiune folosește o organizare bidimensională a memoriei și din punct de vedere istoric are prioritatea apariției. Este folosită la CGA. Să vedem cazurile concrete cum se realizează aceasta.

CGA 320x200/4 culori

Folosindu-se numai 4 culori simultane, pentru codificarea culorii unui pixel sînt necesari numai 2 biți, deci în cadrul unui octet putem avea 4 pixeli,

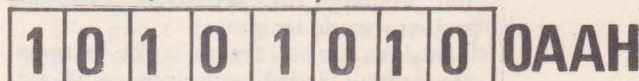
deci pentru o linie avem nevoie de 80 de octeți. Pentru acest mod adresa de început a memoriei video este B800:0000H (segment:offset). Astfel dacă vom scrie la această adresă octetul:



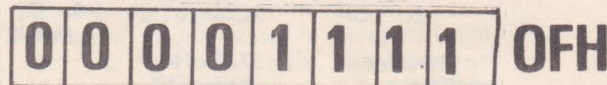
vom observa în colțul din stînga sus un pixel de culoarea fondului (background) și alți 3 pixeli de culori diferite (foreground). De remarcant că la CGA între culoarea care se observă fizic pe ecran și alegerea codificării nu există decît două variante așa cum s-a mai arătat.

CGA 640x200/2 culori

Acesta este un mod de lucru monocrom care duce însă la dublarea rezoluției pe orizontală. În acest mod pentru fiecare pixel este necesar un singur bit. Adresa de început a memoriei video în acest caz este tot B800:0000H. Avînd în vedere rezoluția se observă că pentru o linie sînt necesari tot 80 de octeți. Apare totuși o ciudățenie în organizarea memoriei video și anume: liniile impare sînt grupate în spațiul de adrese începînd de la B800:0000H iar liniile pare sînt grupate începînd de la BA00:0000H. La un calcul simplu se poate deduce că între cele două zone rămîne un spațiu de 192 de octeți nefolosiți. Pentru a înțelege acest mod se poate face o experiență simplă și anume, dacă vom scrie cîte 80x100 octeți de la adresele mai sus menționate cu conținutul:



vom observa că ecranul s-a umplut cu niște linii verticale foarte fine. Dacă în acest loc vom folosi alt octet:



vom observa niște bare verticale groase.

Aceste experiențe se pot face direct din DEBUG-er cu condiția ca înainte de a accesa memoria video să trecem în modul 4 (320x200) sau 6 (640x200) folosind întreruperea 10H din BIOS.

A doua versiune de organizare a memoriei video, cea din plane de culoare, este incomparabil mai complexă dar este singura capabilă să asigure un nivel înalt al performanțelor. Dacă în

versiunea anterioară vom încerca să mărim numărul de culori sau rezoluția vom avea dificultăți cu timpul de acces, fiindcă viteza de baleiaj a monitorului TV trebuie să rămână între anumite limite. Deci dacă la citirea octet cu octet nu putem depăși o anumită viteză dictată de factori tehnologici, atunci o soluție ar fi să citim în paralel mai multe memorii dar la o viteză mai redusă.

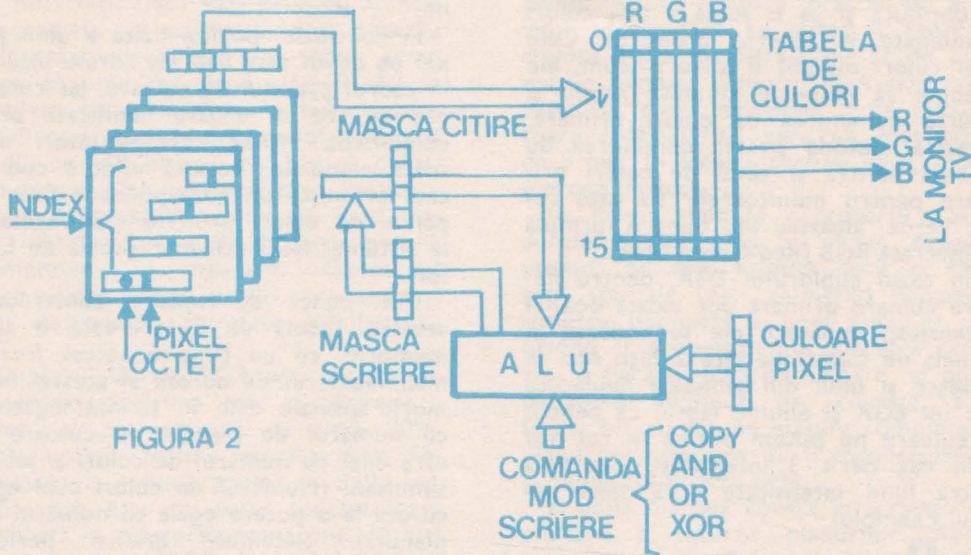


FIGURA 2

Vom începe descrierea structurii pornind de la planurile de culoare. Dacă luăm cazul unui singur plan de culoare vom observa că în interiorul fiecărui octet, fiecărui bit îi corespunde un pixel, la fel ca la funcționarea monocromă studiată anterior. Deci un plan de culoare implementează practic aspectul bidimensional x-y al monitorului TV. Pentru display (afișaj) culoarea apare ca o a treia dimensiune iar aceasta se transpune în memoria video ca axa z, deci adâncimea. Vom explica acest lucru mai bine printr-un exemplu. Adresa de început a memoriei video în acest caz este A000:0000H, iar toate cele 4 planuri răspund fizic la aceeași adresă. Modul în care se rezolvă acest conflict în operații de scriere/citire va fi explicat în cursul prezentării cuplorului. Dacă la adresa de început avem următorii octeți:

0	1	0	1	0	0	1	1	PLAN 0
0	0	1	1	0	0	0	1	PLAN 1
0	0	0	0	1	0	1	1	PLAN 2
0	0	0	0	0	1	0	1	PLAN 3

Această structură are însă o complexitate sporită prin atașarea obligatorie a altor componente funcționale. Pentru exemplificare prezentăm în tabelul 2 o versiune simplificată a cuplorului EGA. Diferențele față de VGA sînt relativ mici și în general o astfel de structură seamănă principal cu echipamentele grafice profesionale.

atunci în colțul din stînga sus al ecranului vom avea o succesiune de pixeli de următoarele culori:

PIXEL	CULOAREA
1	0
2	1
3	2
4	3
5	4
6	8
7	5
8	15

După cum se observă codurile de culoare ale pixelilor se obțin prin componerea pe adâncime (axa z) a pixelilor corespunzători.

Acest cod nu are un corespondent fix în spațiul culorilor afișabile care, așa cum am văzut, poartă denumirea de paletă. Codul este numai un fel de „culoare logică” adică un index într-o tabelă de culori așa cum se observă în tabelul 2. La locația selectată în acest tabel cu ajutorul indexului se găsesc acele informații care dau efectiv culoarea pe ecran. Deci Tabela de Culori face translația de la „culoare logică” la „culoare fizică”. Această modali-

tate de transcodare provine din tehnicile de codare unde mai poartă denumirea de Look-Up Table, denumire care se găsește ca alternativă în documentațiile unor firme din S.U.A.

Pentru a înțelege mai bine rolul acestei tabele este necesar să facem o scurtă divagație pe teme de colorimetrie. Orice culoare existentă în natură, indiferent de nuanță și saturație poate fi obținută și ca o sumă a trei culori combinate în diverse proporții. Cele trei culori nu pot fi alese oricum, ele trebuie să respecte anumite reguli și poartă denumirea de culori primare. Această metodă poartă denumirea de sinteză aditivă și setul de culori primare pentru monitoarele TV este roșu, verde, albastru sau după o formulă consacrată RGB (Red-Green-Blue).

În cazul cuplorului EGA, pentru fiecare culoare primară pot exista doar 4 intensități deoarece ele se codează în Tabela de Culori pe câte 2 biți. Aici se găsește și unul din punctele neplăcute ale lui EGA și anume faptul că pentru o culoare nu putem obține în cel mai bun caz decât 3 intensități, cea de-a patra fiind intensitate nulă, deci negru. Exemplu:

R	G	B	
0	0	0	NEGRU
0	1	0	GALBEN 25%
1	0	0	GALBEN 50%
1	1	0	GALBEN 100%

Una din tehnicile de a mai obține diverse luminozități este de a mai adăuga sau scădea, dacă este posibil, culoarea albă. Aceasta înseamnă a umbla la saturația culorii dar un ochi neexperimentat va accepta aceasta ca o variație de luminozitate. Acest artificiu ne va ajuta ca pentru o culoare primară R, G sau B să mai putem obține 2 nuanțe, dar se pot găsi și combinații la care să nu mai putem obține altă luminozitate, ca de exemplu:

R	G	B
1	1	1
1	0	0
0	0	1

Se observă că nu putem face operații ± fără a altera culoarea care, în acest caz, poate fi definită ca un fel de galben ce bate spre violet.

Privind din nou la tabela de culori se observă că toate combinațiile ce se pot realiza cu cele trei culori funda-

mentale sînt în număr de 64(2⁶). Caracterizînd cuplorul EGA prin rezoluție, numărul de culori simultane și paletă, se poate afirma că acești parametri se degradează față de minimul cerut unei aplicații profesionale tot în această ordine, ultimul parametru, paleta, fiind de-a dreptul dezamăgitor pentru că numărul de biți pentru o culoare primară e mai mic decît numărul de planuri.

În concluzie, poziția fizică a unui pixel pe ecran este dată de adresa bitului în cadrul planului de culoare, iar culoarea sa are o valoare codificată prin combinația biților corespunzători din toate planurile. Această valoare codificată primește un corespondent fizic în paleta de culori printr-o transformare la sistemul RGB utilizînd Tabela de Culori.

Din punct de vedere constructiv această Tabelă de Culori este o altă memorie, cu un timp de acces foarte mic. Numărul de adrese al acestei memorii speciale este în strînsă legătură cu numărul de planuri de culoare și este egal cu numărul de culori afișabile simultan. (Numărul de culori este egal cu doi la o putere egală cu numărul de planuri). Sistemele grafice performante lucrează de obicei cu 8 sau 12 planuri, adică pot afișa 256 și respectiv 4 096 de culori.

Dimensiunea cuvîntului în Tabela de Culori este, de asemenea, deosebit de importantă pentru cazurile în care vrem să facem umbriri sau să modelăm textura suprafeței. Deși luate independent, ochiul nu poate distinge foarte multe nuanțe, totuși în imagini de ansamblu sînt absolut necesare. Încercați să ghiciți cîte nuanțe sînt în reflexul unei raze de soare pe suprafața unui pahar cu apă. Dacă soarele este puternic sînt peste 10 000. Pentru astfel de modelări la unele sisteme, cum ar fi Hewlett-Packard 9 000/seria 300, Tabela de Culori are cîte 8 biți pentru fiecare componentă RGB.

După cum se observă însă din tabelul 2, indexul de culoare al unui pixel nu atacă direct Tabela de Culori, ci trece printr-o Mască de Afișare (Display Enable Mask) care poate face ca unele planuri să nu participe la formarea indexului de culoare, ele fiind considerate 0. În aceste condiții în acel plan se poate face orice operație de desenare fără ca în acest timp să se altereze imaginea de pe display.

Prezentarea unor astfel de detalii interne din structura unor cuploare grafice pare lipsită de sens în condițiile în care compilatoarele actuale au biblioteci grafice atît de ușor de utilizat. Inconvenientul major este că acestea

doresc să acopere o gamă cât mai largă de cuploare și, în acest caz, ele se bazează numai pe posibilitățile comune. De exemplu, bibliotecile compilatoarelor produse atât de Microsoft cât și de Borland nu oferă nici o facilitare de a exploata Masca de Afișare care, împreună cu Tabela de Culori, stau la baza animației pe calculator, atât de gustată dar greu de exeplicat pentru cei neavizați. Dacă veți privi cu atenție programele de animație, veți observa că numărul de culori este redus, semn sigur că unul sau două planuri sînt în umplere cu noua imagine iar celelalte se afișează. Cu cât programul este mai „violet” cu atât el are mai puține culori simultane. Altă versiune este ca decolorul să fie desenat în unul sau două planuri iar restul să fie folosite pentru elementele în mișcare. Cunoșcînd acum aceste detalii priviți cu atenție și veți descoperi „secretele de fabricație”.

Cuplorul de tip VGA are o serie de îmbunătățiri. Prima este o creștere a rezoluției pe verticală cu circa 30%, prețul plătit pentru aceasta fiind renunțarea la a doua pagină. A doua îmbunătățire este cu totul spectaculoasă și constă în creșterea numărului de biți pe componentă RGB de la 2 la 6 biți, deci paleta a crescut de la 64 la 256 K culori! Tabela de Culori a fost menținută din considerente de compatibilitate, dar ea nu mai conține informația RGB, ea devine doar un nou codor care, prin combinații cu alte registre, formează un sistem de adresare a 255 de Registre de Culoare conținînd informația RGB. Sistemul de adresare este destul de sofisticat și diferit pentru modurile de lucru, așa că nu vom intra în detalii. Tot ca o îmbunătățire remarcabilă este și modul de lucru cu 256 de culori simultane dar din păcate la o rezoluție redusă (320x200). Organizarea memoriei are în acest caz un aspect hibrid și anume ea este organizată pe 4 planuri dar în fiecare plan la codare participă cîte 2 biți ca la CGA în modul cu aceeași rezoluție. Acest lucru este însă transparent pentru utilizator pentru care există corespondent direct între pixel și octet începînd de la adresa A000:0000H.

O ultimă îmbunătățire la VGA față de EGA este și faptul că registrele cuplorului pot fi citite, deci starea cuplorului poate fi determinată cu exactitate în orice moment.

Toate detaliile prezentate anterior s-au referit la citirea memoriei video în scopul formării imaginii pe monitorul TV. Această citire se face ciclic sub controlul unui bloc funcțional care poartă denumirea de Controller CRT.

El asigură formarea semnalelor de culoare RGB și sincro pentru monitorul TV.

Vom studia în continuare interacțiunea dintre procesor și memoria video. În cazul în care memoria este de tip împachetat, ea este adresată normal de către CPU, dar în varianta în care se lucrează cu planuri de culoare, lucrurile sînt mult mai complicate deoarece planurile sînt mapate practic pe aceleași adrese. Ele ar putea fi teoretic tratate cu o magistrală de 32 de biți, dar în general cuplorul EGA este conceput să poată funcționa și pe calculatoare IBM PC/XT care au magistrală de 8 biți. În această situație există două metode de acces.

În prima metodă procesorul poate adresa fiecare plan în parte. Această metodă nu este folosită decît în aplicații speciale (ex.: animație) deoarece scrierea completă a unui pixel ar cuprinde și 4 operații de selectare separată a fiecărui plan de culoare. În plus, deoarece indexul de culoare s-ar forma bit cu bit pe măsura completării planurilor, efectele ar fi incontrollable și neplăcute pentru ochi.

A doua metodă constă în scrierea simultană a tuturor planurilor. Acest lucru se realizează cu ajutorul unui Registru de Culoare Pixel. Reprogramarea acestui registru este necesară numai cînd se schimbă culoarea de trasare, deci mult mai rar. Din punct de vedere al procesorului, scrierea ar decurge ca într-un singur plan deși se completează toate conform registrului.

Aceasta a fost totuși o viziune simplificată. Dacă revenim la figura 2 vom observa, pe lîngă o serie de registre de mascare la scriere și citire, un important element funcțional și anume ALU (Arithmetic Logic Unit) care poate face ca noua culoare a unui pixel să fie dependentă de vechea valoare. În cuploarele EGA și VGA circuitul ALU folosit poate executa patru funcții: COPY; AND; OR; XOR.

Față de alte echipamente grafice performante, numărul de funcții implementat e mic, dar sînt cele utilizate în 80% din aplicații. Cu titlu de curiozitate menționăm că la calculatorul Felix M118 circuitul ALU are 16 funcții!!! Funcția COPY atunci cînd se face desenarea prin supraimprimare. Funcțiile AND și OR sînt folosite pentru desenări cu efecte de transparență. Foarte importantă este funcția XOR care face inversare și reprezintă mecanismul de bază pentru cursoare grafice și unele versiuni de animație.

Așa cum s-a precizat anterior, una din metodele de a folosi resursele cuploarelor EGA/VGA este prin inter-

mediul BIOS-ului extins prezent pe placă. Această metodă este folosită de multe cuploare inteligente cum ar fi de exemplu cel de hard disk. Întreruperea 10H care deservește ecranul, practic se dublează ca număr de funcții pentru operațiuni de grafică, culori, scriere texte, seturi de caractere. Adresa la care începe BIOS-ul extins este C000:0000H și conține în primii trei octeți o semnătură. Dacă în cursul operației de bootstrap se depistează această semnătură se lansează programul de la C000:0003H care face toate operațiunile necesare integrării cu BIOS-ul de bază. Detalii concrete asupra funcțiilor de la întreruperea 10H se pot găsi în bibliografia de specialitate sau folosind programul HELP.

Pentru operații care nu sînt prevăzute în BIOS se impune utilizarea directă a registrelor cuplorului care sînt grupate în 5 seturi de bază. Adresele lor sînt prezentate în tabelul 2.

TABELUL 2

GRUP	ADRESE (HEX)
General	3DA, 3CA, 3C2, 3CC
Sequenser	3C4, 3C5
CRT Controller	3D4, 3D5
Graphics	3CE, 3DF
Atribute	3C0, 3C1

În continuare vom descrie succint fiecare grupă și modul de acces.

Registreele generale. Nu sînt interesante pentru creatorul de soft. Unele dintre ele sînt imaginea unor pini din conectorul monitorului iar altele sînt legate de configurarea cuplorului.

Sequenser. Conține o serie de 5 registre ce controlează funcții referitoare la modurile de lucru alfanumerice și seturile de caractere. Toate registrele sînt cuplate pe o magistrală comună și răspund la aceeași adresă. Selecția lor se face prin intermediul unui demultiplexor ca în figura 3.

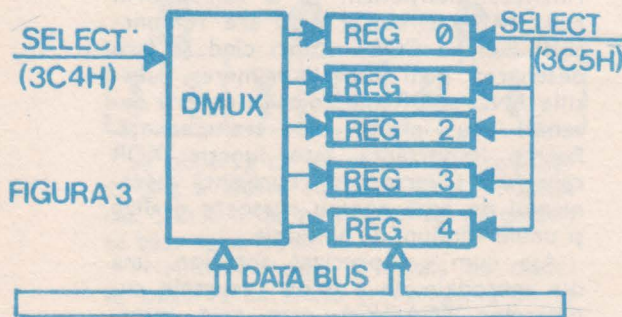


FIGURA 3

Pentru a putea accesa un registru la un moment dat, trebuie să efectuăm două secvențe: selecția în multiplexor și scrierea datelor. Dacă se transmite succesiv multe date către același registru, selecția multiplexorului e suficient să se facă o singură dată la începutul secvenței.

Un exemplu în limbaj C ar fi:
 outp (0x3C4, nr.registru);
 outp (0x3C5, valoare).

Pentru creatorii de grafică, în acest grup este interesant numai registrul 2 care conține o mască pentru selecția planurilor de culoare în care este permisă scrierea.

CRT Controller. Este format dintr-o serie de 26 de registre care conțin în general toți parametrii necesari pentru formarea semnalelor de comandă ale monitorului TV. Modul de adresare este similar cu cel de la sequenser, adresele demultiplexorului și registrelor fiind 3D4H și respectiv 3D5H. Schimbarea parametrilor funcționali ai acestui bloc poate avea efecte spectaculoase dar și unele catastrofice.

Astfel, prin schimbarea numărului de octeți citiți pe o linie, a numărului de linii și a timpului de blankare se pot obține două moduri de lucru nestandard. Primul constă în forțarea rezoluției pînă la 800x600 dar numai unele monitoare mai sînt capabile să sincronizeze. Un monitor care nu se poate sincroniza este supus unor regimuri tranzitorii periculoase în blocul de baleiaj așa că nu recomandăm astfel de experimente. Alt mod de lucru nestandard constă în afișarea unei ferestre 640x350 care poate fi deplasată pe un desen realizat pe un spațiu 800x600. Acest ultim mod este exemplificat în binecunoscutul program EGADEMO la imaginea tomografică. Mult mai simplu de realizat este o fereastră 640x350 (640x480-VGA) care glisează pe un desen 640x720. Acest lucru se obține prin modificarea valorii registrelor care specifică linia de unde începe afișarea (registrele 0CH pentru high/0DH pentru low).

Graphics Controller. Are în componență 9 registre care răspund la cele mai importante funcții de grafică:

- masca de culoare pentru scrierea pixelilor, adică culoarea de trasare;
- masca de protecție a unor planuri;
- selecția unei culori de comparație;
- selecția funcțiilor ALU și rotația octetului de date;
- selecția unui plan pentru citire;
- selecția modurilor de citire și

scriere;

- funcții de lucru pe 16 biți simultan în două planuri;
- mască de planuri invalidate la comparație;
- mască de scriere a octetului.

Attribute Controller.

Conține 19 registre cu rol direct în controlul culorilor. Primele 16 registre sînt de fapt Tabela de Culori, iar următoarele se ocupă cu: controlul modului de lucru (grafic/alfanumeric); controlul culorii chenarului; masca de afișare a planurilor.

Și aceste registre se adresează prin intermediul unui demultiplexor dar din păcate adresa demultiplexorului și a registrelor este aceeași și anume 3C0H. Acest conflict este reglementat de un bistabil care dirijează datele so-site la portul controller-ului de atribute de culoare o dată spre multiplexor și o dată spre bancul de registre. Acest bistabil poate fi adus într-o stare inițială printr-o operație fără conținut de date la portul 3DAH. În limbajul C secvența ar fi următoarea:

```
inp (0x3da);
outp (0x3C0, nr.registru);
outp (0x3C0, valoare).
```

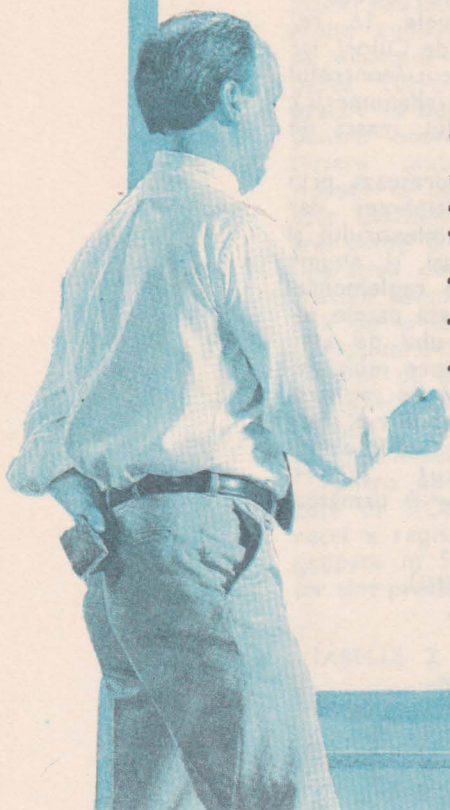
Info Club



Cele prezentate nu au avut rolul să dea o informație completă, organizată în genul unui breviar, ci să vă familiarizeze cu niște principii generale care, în acest caz, sînt mai greu de desprins din literatură. Documentația de firmă este extrem de condensată și fără o pregătire prealabilă e greu de parcurs. O alternativă ar fi cartea scrisă de Richard F. Ferraro — Programmer's Guide to the EGA — VGA Oard (Addison-Wesley/July 1988), cu toate că este exagerat de detaliată și cele exact 600 de pagini fac ca esențialul să fie greu de sintetizat la prima parcurgere.

Dacă am reușit să stîrnim interesul dumneavoastră, într-un episod următor putem prezenta o bibliotecă grafică pentru limbajul C cu exploatarea tuturor facilităților cuplorului EGA (VGA)!

▲ Actualitatea PC



Abstractizarea și rolul ei în evoluția limbajelor de programare

Stelian
NICULESCU

Familia limbajelor de programare (FLP) se situează, ca evoluție, între limbajul binar (LB, propriu calculatoarelor electronice) și limbajul natural (LN). În timp ce LB reprezintă limita stângă a FLP, avînd în vedere evoluția în timp, fiind primul membru al FLP, LN este limita dreaptă, continuînd să constituie încă un ideal greu de atins. Nu putem să nu observăm că un salt calitativ remarcabil în evoluția FLP este datorat apariției generației a V-a de calculatoare, limbajele de generațiile I - IV fiind de tip imperativ și cu elemente descriptive, iar cele de generația V fiind de tip imperativ și cu elemente imperative. Iată saltul de la **CUM** la **CE** în activitatea de programare. Ar mai fi de menționat faptul că limbajele noi care apar se constituie în restricții ale limbajului natural, din ce în ce mai apropiate de acesta.

În cele ce urmează vom face unele considerații privind rolul abstractizării în evoluția în timp a FLP, rol pe care îl considerăm deosebit de însemnat.

O abstractizare a unui obiect reprezintă o caracterizare a acestuia prin niște atribute, ceea ce duce la o clasă de echivalență, în care elementul respectiv este membru. Obiectele clasei sînt numite realizări, implementări, rafinamente sau instanțieri ale abstrac-

tizării. În fapt, o abstractizare este concepută ca o specificare de tip **CE**, în timp ce implementările ei sînt asociate cu specificații de tip **CUM**, procesul de abstractizare prezentînd interes practic numai în condițiile în care specificarea de tip **CE**, caracterizînd atributele esențiale ale unui obiect, este substanțial mai simplă decît specificația de tip **CUM**.

Abstractizarea are un rol deosebit în studiul și asigurarea modularității programelor, fiecare modul avînd o specificare de tip **CE** (abstractizare), prin care se precizează ce se face, precum și una de tip **CUM** (rafinament, realizare sau implementare), care materializează modul de realizare a specificației **CE**.

Alături de abstractizare există două principii care duc la sporirea rolului specificațiilor de tip **CE**, în detrimentul celor de tip **CUM**, ceea ce concordă cu noile viziuni în ingineria programării: principiul localizării informațiilor implicate în realizarea abstractizării; principiul lui Parnas, cunoscut sub denumirea „ascunderea informației”, avînd drept scop să facă „vizibile” numai acele proprietăți ale modulelor care sînt strict necesare în interfațarea lor.

În procesul de programare sînt cu-

D.P. TIBERIU

PLACA DE BAZĂ

IBM PC-XT

Placa de bază (**mother board**) a calculatorului PC-XT poate fi considerată ca un calculator de 16 biți de utilizare generală. După conectarea echipamentelor periferice și a comenzilor acestora rezultă un **calculator personal profesional**. Firma IBM a proiectat familia de calculatoare PC ca niște sisteme deschise, care se pot dezvolta prin conectarea la magistrala universală a modulelor suplimentare. Deoarece IBM a publicat datele tehnice ale magistralei, alte firme au putut produce atît plăcile de bază cit și modulele funcționale (**add-on cards**).

Placa prezentată de noi în acest poster este denumită **MEGABOARD**. Aceasta conține circa 100 circuite integrate, opt conectoare pentru add-on cards și se poate împărți (diviza) în următoarele blocuri funcționale:

- blocul procesorului (8088, 8087) și magistrala locală;

- blocul accesului direct la memorie (circuit DMA-8237);

- magistrala de extensie (Expansion Bus);

- blocul de comandă al magistralei externe (External Bus Control);

- blocul de memorie cu acces aleator (Random Access Memory);

- blocul memoriei de tip ROM (Read Only Memory);

- blocul echipamentelor de intrare-ieșire (Peripherals).

Cele trei magistrale sînt: de sistem, externă și de extensie. Acestea pot fi comandate de procesor sau de comanda DMA.

În cele ce urmează ne vom opri în detaliu asupra unor blocuri funcționale, mai importante.

Blocul procesorului. Elementele de bază ale blocului sînt procesorul

8088 varianta de 8 biți, sau 8086 pentru 16 biți — și co-procesorul numeric 8087 (bloc 1). Circuitul 8288 (U19 bloc 4) decodifică semnalele de stare ale procesorului, S0/, S1/, S2/ și generează semnalele de comandă: IOW/, IOR/, MEMW/, MEMR/, MWTC/, INTA/ și, suplimentar, semnale de comandă ale buffer-ului magistralei de date U1: ALE, DEN, DT-R.

Semnalele de tact propriu din circuitul 8284 (U3 — bloc 2) comandat de un cuarț de 14,31818 MHz. Condensatorul C8 permite fixarea riguroasă a frecvenței. Circuitul 8284 are trei ieșiri: OSC (frecvența f) — accesibil doar pe magistrala de extensie; CLK 88 (frecvență $f/3$) — semnalul de tact de bază pentru procesor și majoritatea circuitelor de pe placa de bază; PCLK (frecvență $f/6$) — folosit pentru citirea tastaturii și, după împărțirea la doi în circuitul U35 — ca semnal de tact pentru controlul 8253.

După conectarea alimentării sau acționarea comenzii S1-RESET, procesorul și alte circuite sînt șterse cu semnalele RES 88 și RESET/Z din circuitul U3. Blocul format din R1-R2-C12 și CR1 asigură o durată suficient de mare a semnalului de ștergere. Pentru conversia liniilor locale ADO-AD7 și AA12-AA19 sînt folosite buffer-ele LS373 (U2 și U7 — bloc 5), iar liniile AAB și AA11 folosesc jumătate din U4-LS244. Liniile locale de date sînt legate de U1-LS245. Aceste circuite trec în stare de impedanță ridicată în momentul preluării comenzii de către DMA (8237).

Circuitele 8088, 8087, 8237 pot trece în stare de așteptare (wait state) dacă un echipament oarecare input-output sau memoria necesită un timp îndelungat de acces. Circuitul special de generare a semnalului de confirmare READY construit din bistabilii U36, U37, U44 și porți (bloc 3), comandînd intrarea AEN1/ a circuitului U3-8284, impune starea de așteptare a procesorului.

Încetinirea lucrului procesorului depinde de tipul transmisiei, care poate fi:

1. scriere sau citire date de către procesor în și din memorie;

2. scriere sau citire date de către procesor în și din echipamentele in-out (periferice);

3. acces direct la memorie (DMA).

În primul caz, dacă memoria se află pe placa de bază, nu trebuie introdusă starea de așteptare. Accesul la memoria rezidentă pe modulele suplimentare poate fi încetinit prin impunerea nivelului jos de tensiune pe linia I-OCHRDY a magistralei de extensie. În al doilea caz, transmisia este întotdeauna prelungită cu un ciclu de așteptare. Dacă echipamentul in-out se găsește pe modulul extern (exemplu pe cel de comandă al floppy) transmisia poate fi suplimentar întîrziată prin linia I-OCHRDY.

Procesorul trimite sau recepționează date, realizînd operațiile: scriere-citire în sau din memorie; scriere-citire în sau din echipamentele in-out; confirmarea acceptării întreruperii.

Operațiunea de citire a memoriei de către procesor (detaliată în figură) se poate împărți în următoarele etape:

- trimiterea către comanda magistralei 8288 a informației privind începearea operației de citire (liniile S0, S1);

- plasarea pe liniile locale a adresei celei respective de memorie;

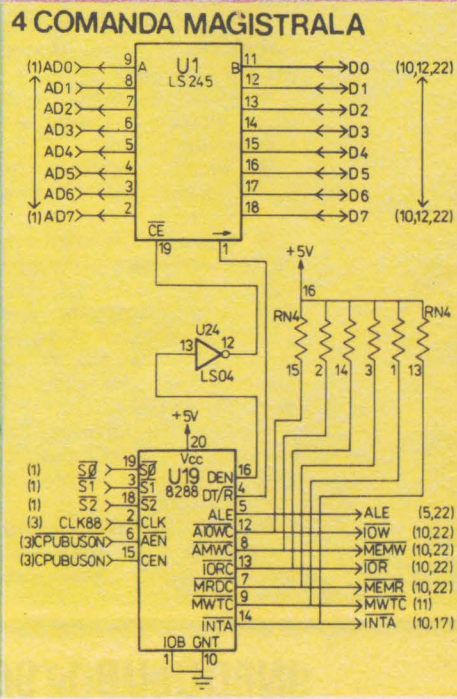
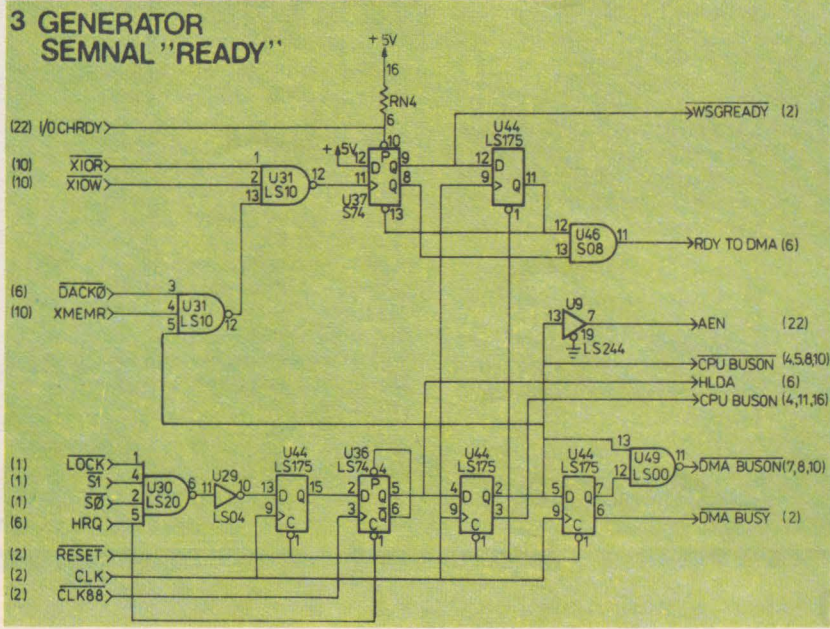
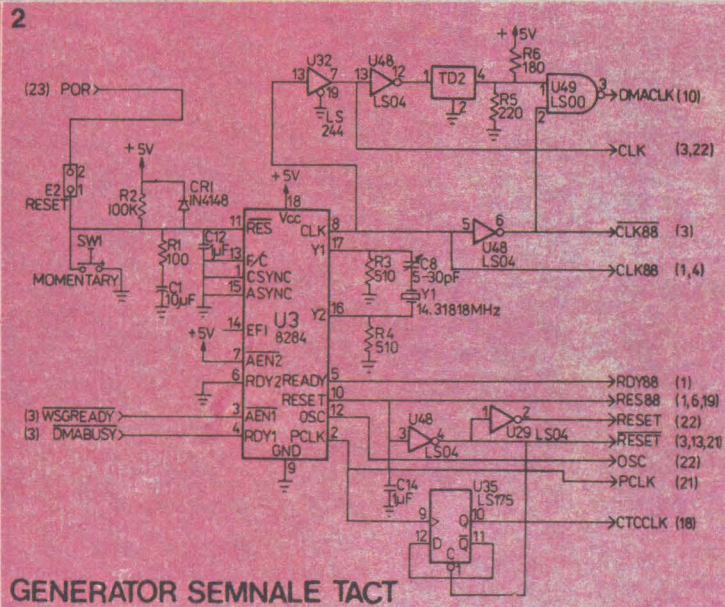
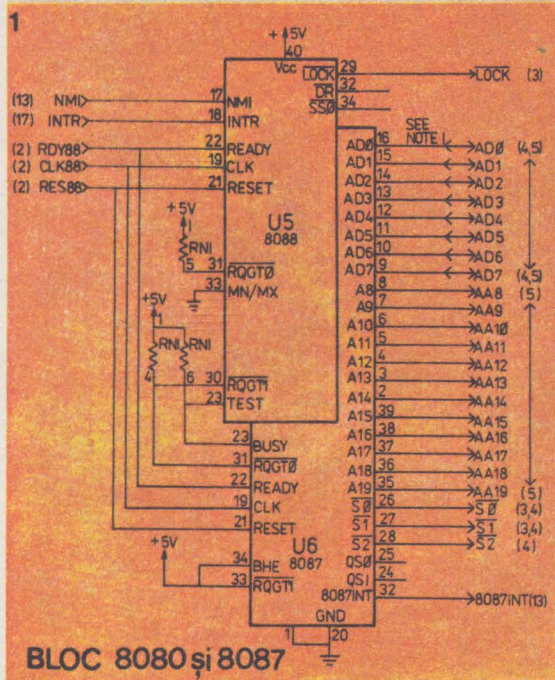
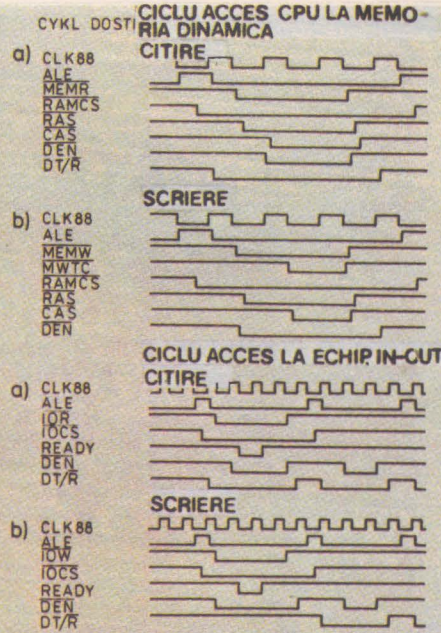
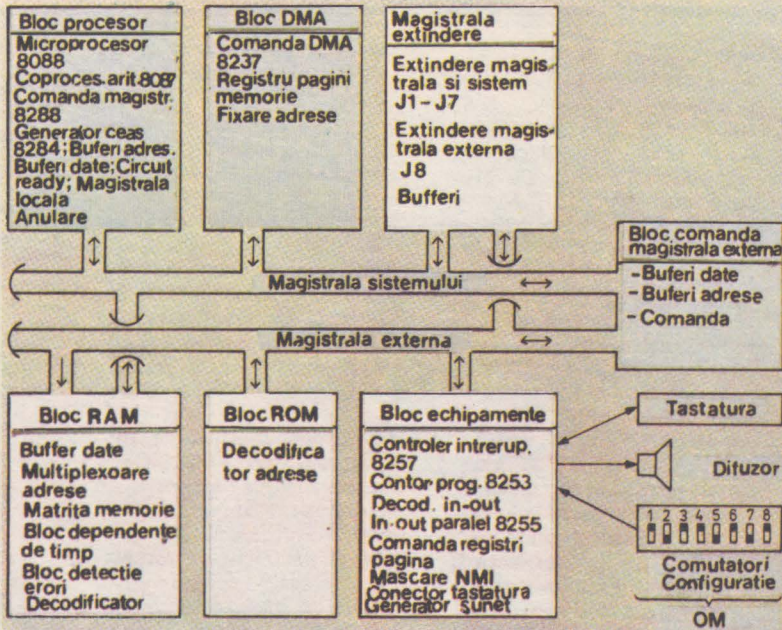
- decodificarea în comanda 8288 a stării, transmiterea semnalului ALE și fixarea adresei pe magistrala de sistem (circuitele U2, U4, U7 — bloc 5), bufferarea adresei pe magistrala externă (U8, U9, U10 — bloc 9);

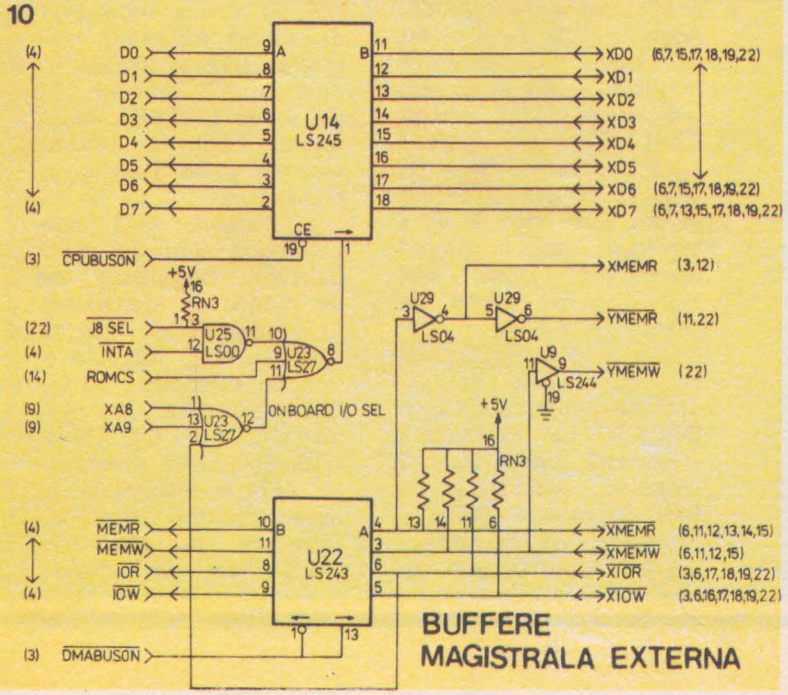
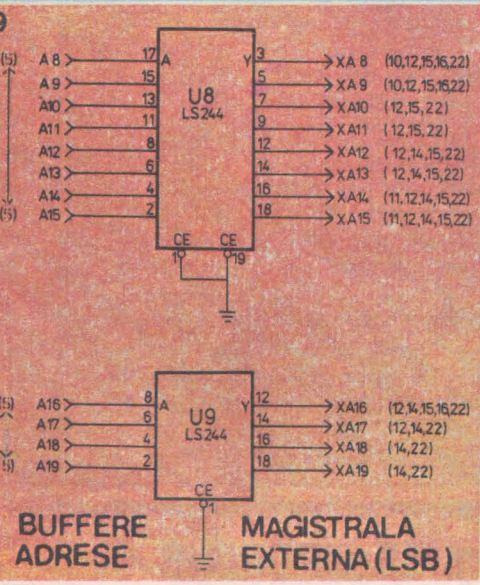
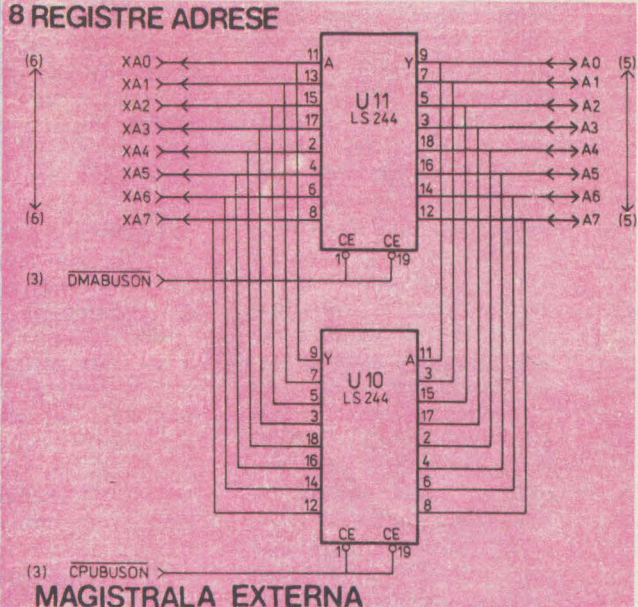
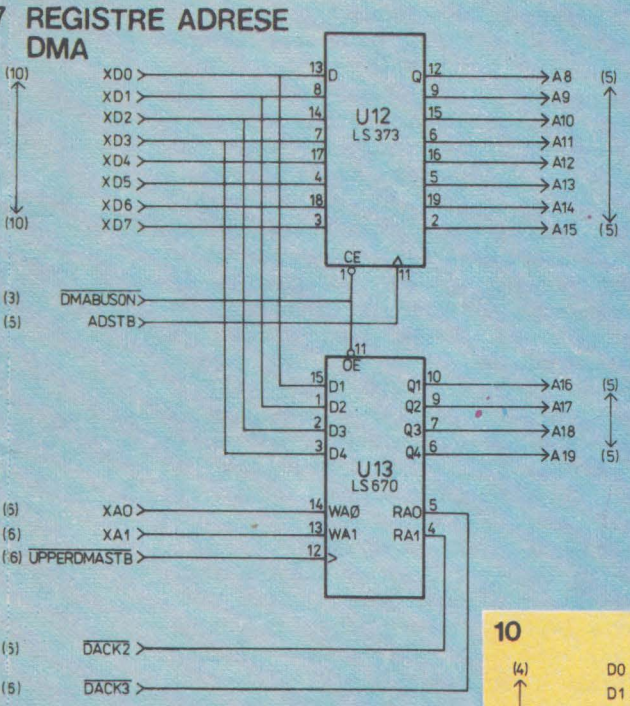
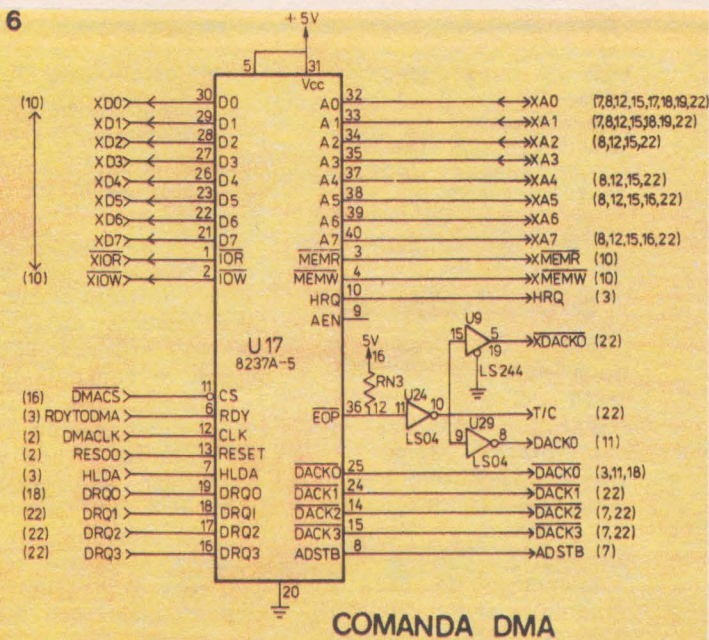
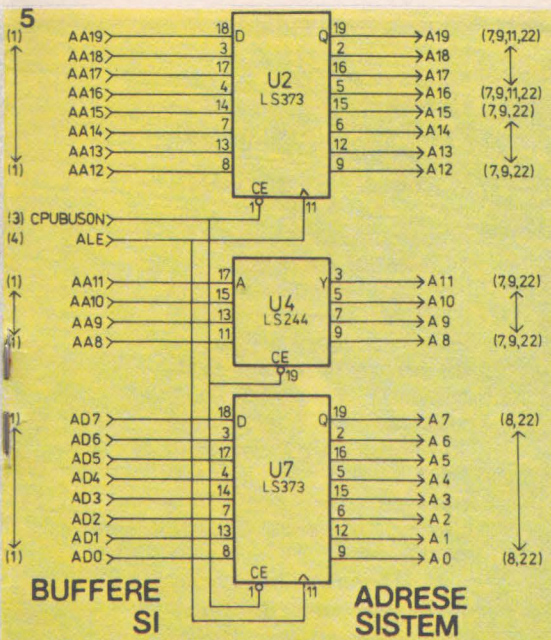
- deblocarea bufferului de date U1 și fixarea direcției de transmitere de pe magistrala de sistem pe magistrala locală;

- generarea semnalului MEMR/ și transformarea lui U22 (bloc 10) în semnal de citire XMEMR/;

- decodificarea adresei de către toate circuitele de memorie;

- plasarea datei pe magistrala corespunzătoare (în cazul memoriei interne ROM conectată la magis-





trala externă acționează buffer-ul U14 — bloc 10);

● eventuala prelungire a ciclului prin inițierea ciclului de așteptare (operația de citire constă din minimum 4 cicluri de ceas ale procesorului — respectiv 840 ns).

Blocul DMA

Blocul DMA realizează două funcții: împrospătarea ciclică a memoriei (refresh memory) și transmiterea rapidă între echipamentele in-out conectate la magistrala de extensie și memoria sistemului.

Elementul de bază al modului este circuitul 8237 A-5 (U7 — bloc 6). Acest circuit folosește două registre auxiliare U12-74LS373 și U13-74LS670 (bloc 7).

Registrul U12 trimite cei 8 biți de adresă A8-A15 indicați de comandă pe magistrala externă de date. Împreună cu biții A0-A7 generați din nou la transmiterea fiecărui octet, se formează cei 16 biți ai adresei. Registrul este încărcat la începutul fiecărei transmisii și reactualizat după generarea transferului de către contorul care creează semnalele A0-A7. Circuitul U13 extinde adresa DMA cu 4 biți necesari pe magistrala de 20 de biți a microprocesorului 8088. Acest circuit este plasat în zona de adresare in-out și ocupă în aceasta trei adrese (81H, 82H și 83H) corespunzătoare celor trei canale DMA (2, 3 și 1). În aceste adrese se pot scrie valorile de 4 biți care apoi, în timpul operațiilor DMA sînt transmise pe liniile de adrese A16-A19. Alegearea celor 4 biți în adresa respectivă este comandată de liniile DACK2/ și DACK3/. La transmitere în canalul 2 este înscrisă valoarea memorată în 81H, în canalul 3-82H și în 1-83H. La transmiterea în canalul 0, circuitul

U13 nu este folosit.

În timpul transmisiei DMA se efectuează trimiterea de date între echipamentele in-out și memorie, conform următoarei scheme:

1. Programul de execuție inițiază canalul DMA conectat cu echipamentul respectiv in-out, scriind în comanda DMA 16 biți mai puțin semnificativi ai adresei de început a bufferului de memorie iar în registrul U13 cei 4 biți semnificativi. De asemenea, în comanda DMA este scrisă cifra octeților trimiși și instrucțiunea de citire sau scriere;

2. Comanda așteaptă pînă cînd echipamentul transmite pe magistrala de extensie semnalul DRQn, semnalizînd pregătirea pentru transmisie;

3. Comanda răspunde cu semnalul HRQ care pentru procesor constituie cererea de a permite comanda magistralei;

4. Dacă magistrala nu este blocată (semnalul LOCK/ nu este activ) procesorul trimite semnalul HLDA și permite astfel comenzi DMA preluarea comenzii magistralei. Semnalul HLDA este generat de U30, U44 și U36;

5. Comanda scrie biții de adresă A8-A15 în registrul U12 și fixează liniile A0-A7; trecînd „0” pe liniile DACKn/ informează echipamentele externe că va avea loc o transmisie;

6. Bistabilul U44 (care generează semnalul CPU BUSON) decuplează de pe magistrala de sistem generatoarele de linii de adresare, linii de date și linii de comandă, cu influență asupra circuitelor U1, U19, U2, U4 și U10. Prin U9-7 și semnalul AEN la magistrala de extensie, bistabilul informează echipamentele in-out că pe durata

transmisiei DMA nu mai trebuie să decodifice adresele magistralei. Semnalul CPU BUSON blochează și decoderele plăcii de bază și toate echipamentele in-out rezidente pe aceasta;

7. Cu semnalele DMA BUSON (circuitul U44) sînt activate registrele de adrese și semnalele de comandă (circuitul U12, U13 — bloc 7, U11, U22 — bloc 8). Semnalul DMA BUSY/ (U44-7) oprește procesorul de a prelua comanda magistralei;

8. Următorul pas depinde de felul de transmisie. La citirea datelor din memorie comanda DMA generează semnalul XMEMR/, iar memoria adresată trimite data pe magistrala de sistem. În următorul ciclu comanda DMA generează semnalul XIOW/ scriind data în echipamentul activat cu semnalul DACKn. La scrierea datelor în memorie comanda generează semnalul XIOR/, iar echipamentul activat de DACKn/ trimite data pe magistrala de sistem. În următorul ciclu comanda generează semnalul XMEMW/ scriind data în celula adresată.

În orice caz transmisia este prelungită cu minimum un ciclu de așteptare. Semnalele XIOR/, XIOW/ prin U31, U37, U44 formează semnalul de „gata” al canalului 1, 2 sau 3 — RDYODMA.

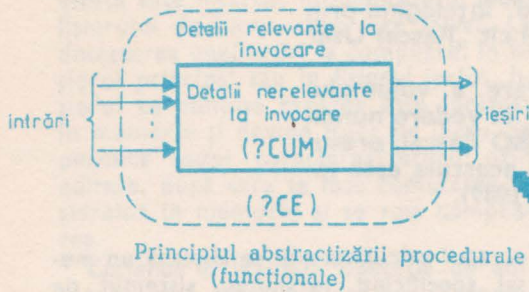
9. Semnalele de comandă HRQ și DACKn/ sînt retrase;

10. Circuitul U44 anulează semnalul CPU BUSON/ și blochează generatoarele canalului DMA și activează bufferii modului procesor;

11. Este anulat semnalul DMABUSY/ și dacă procesorul se găsea în stare de așteptare la terminarea transmisiei DMA, trece acum în funcționare normală.

noscute trei tipuri de abstractizări, corespunzătoare celor trei componente conceptuale fundamentale ale unui limbaj: abstractizarea procedurală, abstractizarea datelor, abstractizarea controlului.

Abstractizarea procedurală sau funcțională a fost cel mai bine suportată de limbajele convenționale, ea corespunzând conceptului clasic de subrutină (procedură). La momentul invocării subrutinei se poate trata ca o „cutie neagră”, fiind separate cele două tipuri de detalii, așa cum se ilustrează în figura alăturată.

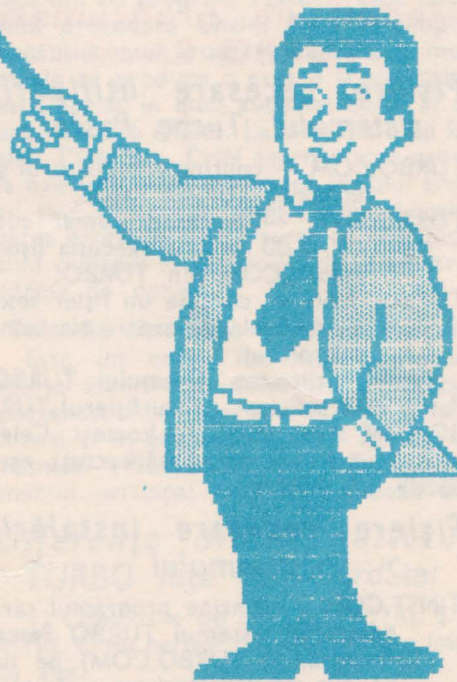


Abstractizarea datelor permite programatorului să se preocupe doar de comportamentul obiectelor implicate în program, adică de tipul informației care poate fi stocată sau extrasă (detalii de tip **CE**). El nu trebuie să se preocupe de modul de reprezentare a obiectelor și nici de algoritmi de memorare sau acces (detalii de tip **CUM**), abstractizarea datelor avînd rolul de a se amîna astfel de decizii pînă la o etapă ulterioară proiectării programului.

O **abstractizare a controlului** presupune o metodă de secvențiere a acțiunilor dintr-un program, majoritatea limbajelor de nivel înalt asigurînd abstractizări predefinite de tipul **if/then/else** și **while/do**. Limbajele mai noi permit în plus abstractizări ale controlului concepute de programator, acestea reprezentînd de fapt generalizări ale metodelor de secvențiere repetitivă disponibile în majoritatea limbajelor de programare.

Abstractizările se regăsesc în limbajele de programare fie sub formă implicită, fie sub formă explicită. Cele implicite sînt irevocabile (structuri de date predefinite, structuri de control), iar cele explicite (sau extinse) sînt introduse (concepute) de programator, limbajul trebuînd să furnizeze mecanisme speciale pentru definirea de noi abstractizări, ceea ce conferă limbajului facilități de extensibilitate.

Se apreciază că majoritatea limbajelor convenționale conțin prea multe abstractizări implicite și prea puține (sau insuficient de generale) mecanisme pentru definirea explicită de abstractizări. Abstractizările implicite constituie pentru multe dintre limbajele de programare o barieră artificială pentru programator, atît din punct de vedere al utilizării, cît și din punct de vedere conceptual. Introducerea de abstractizări explicite presupune ca limbajul să conțină mecanisme lingvistice corespunzătoare, atît pentru definirea și extinderea lor, cît și pentru utilizare, aceasta fiind o cale de urmat.



Încheiem scurta noastră intervenție prin a observa că viitorul este al lui **CE** în detrimentul lui **CUM**, ceea ce duce practic la prosperitatea limbajelor PASCAL, C, LISP, PROLOG ș.a.

Utilizarea sistemului TURBO Pascal

TURBO Pascal este un produs deosebit de reușit al firmei Borland, realizat în anul 1983 de către Philippe Kahn. TURBO Pascal nu este un simplu compilator, ci un mediu de programare interactiv, extrem de productiv, asigurând suport pentru compilarea, execuția, depanarea și testarea programelor Pascal.

Versiunea 3.0 este implementată atât pe microcalculatoare de 8 biți realizate cu microprocesoare Z80 sub sistemul de operare CP/M-80, cât și pe microprocesoare de 16 biți sub o diversitate de sisteme de operare: MS-DOS, PC-DOS, CP/M-86. Versiunile ulterioare: 4.0, 5.0, 5.5 au fost dezvoltate numai pe microcalculatoare cu 16 biți.

TURBO Pascal urmărește fidel limbajul Pascal „standard” înțelegând prin aceasta nu atât standardul ANSI Pascal/IEEE770.X3 97-1983 cât „Pascal User Manual and Report” a lui K. Jensen și N. Wirth.

Prezentarea pe care v-o propunem este o prelucrare a volumului „TURBO Pascal — version 3.0 — Reference Manual” și are în vedere numai particularitățile specifice ale limbajului și operarea în TURBO Pascal, presupunând cunoscut limbajul Pascal standard (o prezentare a acestuia este făcută în revista „Știință și Tehnică”, începând din nr. 7/1989).

Mediul de programare TURBO Pascal

Fișiere necesare utilizării sistemului Turbo Pascal.

- TURBO.COM — conține compilatorul și editorul TURBO Pascal;
- TURBO.OVR — necesar numai sub CP/M-80 pentru execuția fișierelor .COM din TURBO;
- TURBO.MSG — conține un fișier text cu mesajele de eroare ale compilatorului;

Pentru utilizarea sistemului TURBO Pascal este suficient numai fișierul TURBO.COM care ocupă 16 kocteți. Celelalte două fișiere ocupă 33 kocteți, respectiv 1.5 kocteți.

Fișiere necesare instalării sistemului

- TINST.COM — conține programul care adaptează sistemul TURBO Pascal (programul TURBO.COM) pe un anumit tip de calculator și de terminal prin introducerea unor parametri de instalare;
- TINST.DTA — conține parametrii de instalare pentru diferite tipuri de terminale;
- TINST.MSG — conține un fișier text cu mesajele programului de instalare.

Lansarea sistemului TURBO Pascal

După introducerea discului conținând fișierele TURBO Pascal în unitatea curentă se face lansarea sistemului în execuție prin comanda: **TURBO<CR>**. Pe

ecranul calculatorului se afișează un mesaj specificând versiunea, sistemul de operare și se interoghează utilizatorul (prin Y/N) dacă să încarce sau nu mesajele de eroare (fișierul TURBO.MSG).

```
TURBO-Pascal System      Version N.NNX  
                          [System]  
  
Copyright (C 1983, 1984, by BORLAND inc.  
No terminal selected  
Include error message (Y/N)? [ ]
```

În continuare pe ecran apare meniul principal.

Meniul principal

Precizarea comenzilor disponibile:

Logged drive: A

Work file:

Main file:

Edit	Compile	Run	Save
Dir	Quit	compiler	Options

text: 0 bytes
free: 62903 bytes

Comenzile se introduc prin inițiala lor (exceptând comanda execute). Comanda selectată este executată imediat, meniul dispărând de pe ecran; pentru a-l face să reapară se introduce <CR>.

Pentru schimbarea unității de discuri curente se dă comanda L și la răspunsul sistemului **New drive:** se introduce noua unitate selectată (de exemplu B:<CR>).

Fișierul de lucru (folosit la editare, compilare, execuție) este specificat prin comanda **W** și la răspunsul sistemului **Work file name**: se introduce numele fișierului (de exemplu TEST.PAS). Numele fișierului poate avea sau nu extensie, în acest din urmă caz fișierul primind în mod automat extensia .PAS. Nu se recomandă folosirea extensiilor .COM, .CHN, .CMD, .BAK întrucât ele au destinații speciale. Fișierul specificat, dacă există pe disc, este citit în memorie, iar în caz contrar apare mesajul **New File**.

Comanda **M** permite numirea unui fișier principal care să conțină directive de includere (*\$! nume*) de fișiere; acesta este încărcat în memorie în locul fișierului de lucru, care este salvat. La detectarea unei erori la compilare, în fișierul principal sau în fișierul inclus, fișierul ce conține eroarea este încărcat în memorie și devine fișier de lucru. Se permite astfel corectarea erorii prin editare, după care se face comutarea fișierelor în memorie și se reia compilarea.

Comanda **E** lansează operația de editare asupra fișierului de lucru. Meniul principal dispare și este activat editorul. Editorul TURBO este o variantă simplificată a editorului WordStar.

Comanda **C** lansează compilarea fișierului principal. Dacă nu este specificat un fișier principal este compilat fișierul de lucru. Compilarea poate fi întreruptă în orice moment prin apăsarea oricărei taste. Din compilare poate rezulta un program rezident în memorie, sau un program rezident pe disc într-un fișier .COM sau într-un fișier .CHN, în funcție de opțiunile de compilare specificate.

Comanda **R** lansează în execuție un program rezident în memorie sau un fișier .COM sau un fișier .CMD.

Comanda **X** (execuție) prezintă în meniul principal numai în sistemul de operare CP/M-80 permite lansarea în execuție a unui program (fișier .COM) din meniul principal TURBO.

Comanda **S** salvează pe disc fișierul de lucru. După compilare se recomandă salvarea fișierului sursă rezultat din editare.

Comanda **D** listează catalogul unității curente și precizează spațiul neutilizat de pe discul curent. La introducerea comenzii **D** se afișează mesajul **Dir mask: []**; pentru listarea catalogului se tastează <CR>; listarea unui singur nume TURBO sub sistemul de operare, după ce în prealabil utilizatorul e întrebat dacă dorește salvarea fișierului de lucru. de fișier sau a unui grup de fișiere se face indicând un nume de fișier sau o mască conținând caracterele * și ?.

Comanda **Q** face ieșirea din sistemul

Comanda **O** selectează un meniu în care se pot modifica opțiunile implicite ale compilatorului și asigură o funcție de localizare a erorilor la execuție din programele compilate. La introducerea acestei comenzi apare un nou meniu privind modul de lucru al compilatorului având forma:

```
compile — Memory
           Com-file
           cHn-file
```

command line Parameter:

Find run-time error Quit

Comenzile **M**, **C**, **H** selectează destinația unde se depune codul rezultat din compilare (implicit în memorie). În cazul în care codul este rezident în memorie, el poate fi lansat în execuție prin comanda **Run**.

Comanda **C** depune codul generat (și biblioteca Pascal) într-un fișier .COM pe disc. Lansarea în execuție se face în acest caz prin numele fișierului.

Comanda **H** depune codul generat (fără biblioteca Pascal) într-un fișier .CHN pe disc. Lansarea în execuție se face din alt program TURBO Pascal folosind procedura Chain. Dacă în timpul execuției unui program compilat în memorie se produce o eroare la execuție, este apelat în mod automat editorul și eroarea este afișată. La execuția unui fișier .COM sau .CHN apariția unei erori la execuție duce la afișarea codului erorii și a contorului programului. Localizarea erorii în textul sursa se face folosind comanda **F** specificând ca parametru adresa de oprire.

Editorul de texte TURBO Pascal

Este un editor orientat pe ecran foarte asemănător editorului WordStar prezentat sub forma unui breviar de comenzi și în paginile revistei „Știință și tehnică” (1988). El poate fi lansat din meniul principal folosind comanda **E**.

Diferențe dintre editorul TURBO față de WordStar

1) Comenzile de deplasare CTRL S și CTRL D nu permit trecerea de la o linie la alta.

2) Comanda CTRL KT de marcare a unui singur cuvânt ca un bloc este o comandă specifică editorului TURBO.

3) Comanda CTRL KD termină editarea revenind în mediul principal (în WordStar această comandă face și salvarea fișierului pe disc).

4) Comanda CTRL QL reface o linie cât timp cursorul nu a părăsit-o.

5) Comanda TAB pune tab-stopuri în linia curentă pe începuturile de cuvinte din linia precedentă.

6) Comanda CTRL QI de activare/inhibare autoindentare este specifică editorului TURBO.



Directive ale compilatorului TURBO Pascal

Acestea se introduc sub forma unor comentarii cu sintaxă specială, ca de exemplu:

(*\$B—,R+,V—*)

Toate directivele au valori implicite, alese astfel încât să optimizeze viteza de execuție și să minimizeze codul generat. Valorile implicite ale opțiunilor sînt cele subliniate.

a) Directive comune tuturor sistemelor de operare

- 1) **B+** dispozitivul consolă (CON:) este asignat fișierelor standard Input și Output
B— dispozitivul terminal (TRM:) este asignat fișierelor standard Input și Output
- 2) **C+** citire CTRL C întrerupe execuția programului
C— CTRL S activează/inhibă afișarea pe ecran
C— CTRL C nu este interpretat
- 3) **I+** tratează erorile în operațiile de intrare/ieșire
I— tratarea erorilor este făcută de programator
- 4) **I** nume-fișier — include în programul compilat fișierul cu numele specificat
- 5) **R+** verificarea la execuție a încadrării indicilor tablourilor în dimensiuni
R— nu se face nici o verificare asupra indicilor
- 6) **V+** verificarea congruenței listelor de parametri actuali și formali
V— nu se face nici o verificare
- 7) **U+** permite suspendarea execuției programului prin CTRL C
U— execuția programului nu poate fi oprită

b) Directive ale compilatorului TURBO Pascal sub sistemele PC-DOS și MS-DOS.

- 1) **Gn** definește bufferul fișierului standard de intrare Input
GO — fișierul Input se referă la dispozitivul CON: sau TRM:
 Se plasează înaintea părții declarative.
- 2) **Pn** definește bufferul fișierului standard de ieșire Output
PO — fișierul Output se referă la dispozitivul CON: sau TRM:
- 3) **D+** interoghează asupra stării unui fișier text deschis prin Reset, Rewrite sau Append. Dacă fișierul este un dispozitiv, atunci operațiile de intrare/ieșire se vor face caracter cu caracter (fără buffering)
D— nu se face nici o verificare și operațiile se fac bufferat
- 4) **Fn** controlează numărul de fișiere care pot fi deschise simultan (implicit F16).
 Se plasează înaintea părții declarative.

c) Directive ale compilatorului comune sub sistemele PC-DOS, MS-DOS, CP/M-86

- 1) **K+** generează cod de verificare pentru alocarea stivei (se verifică dacă este spațiu suficient în stivă pentru variabilele locale, înaintea fiecărui apel de subprogram).

d) Directive ale compilatorului sub sistemul CP/M-80

- 1) **A+** controlează generarea de cod absolut nerecursiv (nu sînt permise apeluri recursive)
A— sînt permise apeluri recursive
- 2) **Wn** controlează numărul nivelurilor de îmbricare a instrucțiunilor with (cite înregistrări pot fi deschise în interiorul unui bloc $1 \leq n \leq 9$; implicit W2)
- 3) **X+** controlează generarea de cod care să asigure accesul la tablouri cu viteza optimă

Diferențe între TURBO Pascal și standard

Extensii TURBO față de standard

- 1) Simboluri suplimentare:
 - litere: (subliniere)
 - simboluri speciale : \$
 - simboluri cuvinte :

absolute shl	external shr	inline string	overlay xor	
— identificatori standard:				
Addr	Assign	Aux	AuxInPtr	AuxOutPtr
BlockRead	BlockWrite	BufLen	Byte	Chain
Close	ClrEOL	ClrScr	Con	ConInPtr
ConOutPtr	Concat	ConstPtr	Copy	CrtExit
CrtInit	DelLine	Delay	Delete	Erase
Execute	Exit	FilePos	FileSize	FillChar
Flush	Frac	GetMem	GotoXY	Halt
HeapPtr	Hi	IOresult	InsLine	Insert
Int	Kbd	KeyPressed	Length	Lo
LowVideo	Lst	LstOutPtr	Mark	Mem
MemAvail	Move	NormVideo	Pi	Port
Pos	Ptr	Random	Randomize	Release
Rename	Seek	SizeOf	SeekEof	SekEoln
Str	Swap	Trm	UpCase	Usr
UsrInPtr	UsrOutPtr	Val		

- 2) tipuri standard suplimentare:
 byte=0...255;
 string[n]=array[0...n] of char;
 3) Caractere de control:

ValASCII {caracterul având valoarea ASCII specificată}
 ^Caracter {Control-Caracter}

Secvențele de caractere de control se concatenează fără separatori între ele.

- 4) Secțiunile din partea declarativă pot apare de mai multe ori în orice ordine;
 5) Etichetele pot fi numere și/sau identificatori;

6) Constanta reală predefinită pi=3.1415926536E+00;

- 7) Operatori suplimentari:
 shl, shr (multiplicativi)
 xor (aditivi)

8) Operatorul **not** poate fi aplicat unui operand întreg;

9) Instrucțiunea **case** cu clauza **else**;

10) Funcții și proceduri predefinite de lucru cu șirurile de caractere:

a) *delete* (var s: string; p,n:byte); șterge n caractere din șirul s începând din poziția p.

b) *insert* (ss: string; var s:string; p:byte); inserează subșirul ss în șirul s începând din poziția p a acestuia.

c) *str* (v: integer; var s: string); convertește valoarea întregă v într-un șir de caractere pe care-l depune în s;

d) *val* (s:string; var v:tip; var cod:byte); convertește expresia șir de caractere s într-o valoare întregă sau reală pe care o depune în v. Parametrul cod e pus pe 0 dacă nu apar erori sau indică poziția primului caracter eronat;

e) *copy* (s:string; p,n:byte): string; crează un subșir format din n caractere ale șirului s, începând din poziția p a acestuia.

f) *concat* (s1,s2,...,sn:string): string; crează un șir prin concatenarea șirurilor s1,s2,...,sn. Același efect se obține folosind operatorul +. Exemplu:
 s:=s1+s2;

g) *length* (s:string): byte; dă lungimea șirului de caractere s.

h) *pos* (ss,s:string):byte; determină poziția primei apariții a subșirului ss în șirul s (0 dacă nu apare).

11) Acces la memorie și la porturile de date folosind tablourile predefinite Mem și Port.

12) Constante cu tipuri

a) nestructurate — sînt asemănătoare cu variabilele inițializate. Exemplu:

```
const
  e:real=2.7182818;
  nume:string[7]=„POPESCU“;
  b) structurate — asigură tabluri și mulțimi inițializate (pentru conversii, teste, etc.).
```

0 **constantă tablou** inițializează un tablou cu valori constante incluse între paranteze și separate prin virgule.

Exemplu:

```
type
  materii=(analiza, fizica, mecanica);
  sesiune=array [materii] of
  string[8];
```

```
const
  examen:sesiune=(„analiza“, „fizica“, „mecanica“);
```

La definirea unei **constante înregistrare** se inițializează câmpurile înregistrării cu valori specificate. Exemplu:

```
type
  data=record
    zi : 1...31;
    luna: 1...12;
    an : 1900...2000
  end;
```

```
const
  azi:data=(zi:4; luna:9; an:1987);
```

La definirea unei **constante mulțime** se inițializează elementele mulțimii cu valori dintr-o mulțime construită.

Exemplu:

```
type
  litmari=set of 'A'..'Z';
const
  vocale: litmari=['A', 'E', 'I', 'O', 'U'];
```

13) proceduri predefinite de lucru cu fișiere

a) *assign* (VF, nume);
 — asociază șirul de caractere „nume“ cu variabila fișier VF.

b) *rewrite* (VF);
 — crează un nou fișier identificat prin variabila fișier VF (modul generare)

c) *reset* (VF);
 — pregătește în vederea prelucrării (în mod inspectare) fișierul identificat prin variabila fișier VF.

d) *read* (VF, Val);
 — citește din fișierul disc identificat prin variabila fișier VF o componentă în variabila Val și avansează poziționarea la

următoarea componentă (modul inspec-tare).

e) write (VF, Val);

— scrie în fișierul disc identificat prin variabila fișier VF o componentă transfe-rată din variabila Val și avansează pozi-ționarea la următoarea componentă (modul generare).

f) seek (VF, n);

— avansează poziționarea la cea de-a n-a componentă din fișierul identificat prin variabila fișier VF.

Pentru extinderea unui fișier după ul-tima sa componentă se utilizează: seek (VF, File Size (VF));

g) flush (VF);

— golește tamponul fișierului identi-ficat prin VF realizând astfel scrierea pe disc.

h) close (VF);

— închide fișierul disc asociat cu va-riabila fișier VF.

i) erase (VF);

— șterge fișierul disc asociat cu varia-bila fișier VF. Fișierul care se șterge tre-buie în prealabil închis.

j) rename (VF, nume);

— fișierul disc asociat cu variabila fi-șier VF primește ca nume valoarea șiru-lui de caractere nume. Fișierul trebuie în prealabil să fie închis.

k) EOF (VF);

— funcție booleană ce furnizează va-loarea „adevărat” dacă poziționarea este la sfârșitul fișierului (după ultima com-ponentă).

l) FilePos (VF);

— funcție întregă ce furnizează pozi-ția curentă în fișier (prima componentă are poziția 0).

m) Filesize (VF);

— funcție întregă ce dă numărul de componente ale fișierului (dimensiunea fișierului). Un fișier vid are 0 compo-nente.

14) Controlul alocării memoriei la nivel de octet:

a) GetMem (VRef, N);

— alocă N octeți în heap adresați prin variabila referința VRef.

b) FreeMem (VRef, N);

— eliberează N octeți din heap care au fost alocați prin GetMem.

c) MaxAvail;

— furnizează dimensiunea celui mai mare bloc liber din heap.

15) Includerea de fișiere sursă în textul programului folosind directivă (*\$! nu-me-fișier*).

16) Inserarea de instrucțiuni în cod ma-șină în textul programului prin instruc-țiunea **inline**.

17) Alocarea de variabile la adrese abso-lute de memorie. Exemple:

var

x: integer **absolute** \$0000:\$00F4;

În care primul parametru indică

adresa segmentului bază iar al doilea deplasamentul; această formă este utili-zabilă sub sistemele de operare PC-DOS, MS-DOS, CP/M-86. Sub CP/M-80 se indică numai adresa de me-morie adică:

var

x: integer **absolute** \$40FA;

Două variabile pot fi **echivalate**, adică pot fi alocate începînd de la ace-eași adresă de memorie. Exemplu:

var

a: string[10];

b: integer **absolute** a;

18) Obținerea adresei din memorie a unui obiect (variabilă, procedură, func-ție) folosind funcția Addr (nume).

19) Conversia tipurilor scalare folosind facilitatea Retype. Identificatorii de ti-puri ordinale pot fi solosiți ca desemna-tori de funcție cu un parametru. Exem-ple:

type

zi=(lu, ma, mi, jo, vi, si, du);

litere='a'..'z';

apelurile de mai jos întorc valorile:

integer (jo)=3

zi (5)=si

char (65)='A'

litere (3)='d'

20) Proceduri standard de intrare/ieșire pentru terminal

ClrEol — șterge toate caracterele din poziția cursorului pînă la sfîrși-tul liniei fără deplasarea curso-rului.

ClrScr — șterge ecranul și plasează cur-sorul în colțul stînga sus.

CrtInit — trimite pe ecran secvența de inițializare a terminalului

CrtExit — trimite pe ecran secvența de resetare a terminalului, definită la instalare

Delay (Timp) — crează o buclă de întîr-ziere de <Timp> milisecunde

DelLine — șterge linia conținînd curso-rul, mutînd liniile de sub ea cu o poziție în sus

InsLine — inserează o linie vidă în pozi-ția cursorului

GotoXY (X, Y) — mută cursorul în po-ziția de pe ecran specificată prin valorile întregi ale celor 2 parametri (colțul stînga sus al ecranului are coordonatele 1,1).

Exit — produce ieșirea din blocul cu-rent (întoarcere din subpro-gram sau terminarea progr-amlui)

Halt — oprește execuția programului cu întoarcere în sistemul de operare

LowVideo — setare ecran cu atributul 'Start of Low Video'

NormVideo — setare ecran cu atributul 'Start of Normal Video'

Randomize — inițializează generatorul de numere aleatoare

Move(S, D, N) — copiază un bloc de N octeți din S la destinația D

FillChar (D, N, V) — umple o zonă de N octeți începînd de la adresa D cu valoarea V.

Hi(I) — valoarea octetului superior a întregului I.

Lo(I) — valoarea octetului inferior a întregului I

KeyPressed — valoarea True dacă a fost acționată o tastă de la consolă

Random — generează un număr aleator cuprins între 0 și 1

Random (N) — generează un număr aleator cuprins între 0 și N.

ParamCount — furnizează numărul de parametri transmiși programului în zona tampon a liniei comandă

ParamStr(N) — furnizează cel de-al N-lea parametru din zona tampon a liniei de comandă

SizeOf (Nume) — număr de octeți ocupați în memorie de variabila sau tipul Nume

Swap(I) — interschimb între cei doi octeți ai argumentului

UpCase (Car) — conversia în majusculă a argumentului

21) Facilități de segmentare a programelor.

Restricții TURBO față de standard

- 1) Procedurile New și Dispose nu acceptă înregistrări cu variante;
- 2) Procedurile standard Get și Put nu sînt implementate;
- 3) Procedurile standard Pack și Unpack nu sînt implementate. Simbolul cuvînt packd nu are nici un efect (nu realizează o memorie mai economică);
- 4) Nu sînt permisi parametri procedurali;
- 5) Procedura Page nu este implementată;
- 6) Recursivitatea nu este o opțiune de compilare implicită;
- 7) Nu este implementat mecanismul tablourilor conforme.

Mesaje de eroare ale compilatorului

La apariția unei erori în cursul compilării se afișează numărul erorii. Explicații suplimentare asupra erorii se dau numai dacă la înasarea sistemului TURBO Pascal a fost încărcat fișierul cu mesaje de eroare TURBO.MSG. Lista mesajelor de eroare cuprinde:

01 ':' expected
 02 ':' expected
 03 ':' expected
 03 '(' expected
 05 ')' expected
 06 '=' expected

07 ':=' expected
 08 '[' expected
 09 ']' expected
 10 ',' expected
 11 '..' expected
 12 BEGIN expected
 13 DO expected
 14 END expected
 15 OF expected
 16 PROCEDURE or FUNCTION expected
 17 THEN expected
 18 TO or DOWNT0 expected
 20 Boolean expression expected
 21 File variable expected
 22 Integer constant expected
 23 Integer expression expected
 24 Integer variable expected
 25 Integer or real constant expected
 26 Integer or real expression expected
 27 Integer or real variable expected
 28 Pointer variable expected
 29 Record variable expected
 30 Simple type expected
 31 Simple expression expected
 32 String constant expected
 33 String expression expected
 34 String variable expected
 35 Textfile expected
 36 Type identifier expected
 37 Untyped file expected
 40 Undefined label
 O instrucțiune se referă la o etichetă nedefinită

41 Unknown identifier or syntax error
 În instrucțiune apare o entitate nedeclarată (o etichetă, un tip, o variabilă, un identificator de cîmp) sau o eroare de sintaxă

42 Undefined pointer type in preceding type definitions
 Definierea unui tip pointer conține o referință la un identificator de tip necunoscut

43 Duplicate identifier or label
 Identificatorul sau eticheta a fost deja folosit în blocul curent

44 Type mismatch
 1) tipul variabilei și expresiei dintr-o atribuire sînt incompatibile
 2) tipul parametrilor și argumentelor dintr-o declarație și un apel de procedură sau funcție sînt incompatibili
 3) tipurile operanzilor dintr-o expresie sînt incompatibile

45 Constant out of range (constanta depășită)

46 Constant and CASE selector type does not match
 Tipul selectorului și constantelor etichete dintr-o instrucțiune CASE diferă

47 Operand type does not match operator
 Tipul operanzilor nu corespunde operației precizate (de exemplu: 'a' * 2)

48 Invalid result type
 Tipul rezultatului (calculat de funcție) este incorect.
 Tipuri corecte sînt: tipurile scalare, tipurile șiruri de caractere (string), tipurile referință (pointer)

49 Invalid string length
 Lungimea șirului de caractere nu este în intervalul 1...255

- 50 String constant length does not match type**
- 51 Invalid subrange base type**
Tipuri de bază pot fi numai tipurile scalare exceptând tipul real.
- 52 Lower bound > Upper bound**
Valoarea ordinală a limitei superioare trebuie să fie mai mare sau egală cu a limitei inferioare.
- 53 Reserved word**
Cuvintele rezervate nu pot fi folosite ca identificatori.
- 54 Illegal assignment**
- 55 String constant exceeds line**
O constantă șir de caractere nu poate fi continuată de pe o linie pe alta.
- 56 Error in integer constant**
O constantă întreagă nu se conformează regulilor de sintaxă sau depășește domeniul —32768...32767.
- 57 Error in real constant**
Constanta reală nu se conformează regulilor de sintaxă.
- 58 Illegal character in identifier**
- 60 Constants are not allowed here.**
- 61 Files and pointers are not allowed here.**
- 62 Structured variables are not allowed here.**
- 63 Textfiles are not allowed here.**
- 64 Textfiles and untyped files are not allowed here.**
- 65 Untyped files are not allowed here.**
- 66 I/O not allowed here.**
- 67 Files must be VAR parameters.**
- 68 File components may not be files**
Nu sînt permise construcții de tipul file of file
- 69 Invalid ordering of fields**
- 70 Set base type out of range**
Tipul de bază al mulțimii trebuie să fie un scalar cu cel mult 256 valori posibile sau un subdomeniu cu limitele cuprinse între 0 și 255.
- 71 Invalid GO TO**
O instrucțiune GO TO nu poate referi o etichetă în interiorul unui ciclu FOR
- 72 Label not within current block**
O instrucțiune GO TO nu poate referi o etichetă din afara blocului curent
- 73 Undefined forward procedure(s)**
Un subprogram a fost declarat cu directiva forward, dar corpul lui nu apare nicideunde declarat.
- 74 INLINE error**
- 75 Illegal use of ABSOLUTE**
1) Într-o declarație de variabilă cu absolute, înaintea caracterului ':' poate apare numai un identificator.
2) Clauza absolute nu poate fi folosită într-o înregistrare
- 76 Overlays can not be forwarded**
Directiva forward nu poate fi folosită în cazul segmentării
- 77 Overlays not allowed in direct mode**
Segmentele pot fi folosite numai din programe compilate într-un fișier
- 90 File not found**
Fișierul inclus specificat nu există
- 91 Unexpected end of source**
Programul are probabil mai mulți delimitatori begin decît end.
- 92 Unable to create overlay file**
- 93 Invalid compiler directive**

97 Too many nested WITHs

Folosiți directivă de compilare W pentru a mări numărul instrucțiunilor WITH imbricate. În mod implicit (sub CP/M 80) sînt permise numai 2 instrucțiuni WITH imbricate.

98 Memory overflow

S-a încercat alocarea de mai multă memorie decît este disponibilă.

99 Compiler overflow

Memoria disponibilă este insuficientă pentru compilarea programului. Programul se împarte în mai multe segmente și se folosesc fișiere incluse.

Diferențe TURBO Pascal față de standard

Limbajul Turbo Pascal respectă limbajul standard Pascal definit de Wirth și Jensen în „Pascal-User Manual and Report”, cu cîteva diferențe minore apărute din motive de eficiență a programelor executabile.

Variabile dinamice

Procedura New nu se poate folosi pentru înregistrări cu variante. Această restricție poate fi ocolită prin utilizarea procedurii standard GetMem.

Recursivitate (în CP/M-80)

Din cauza modului în care sînt tratate variabilele locale într-un apel recursiv, o variabilă locală unui subprogram nu trebuie transmisă ca parametru 'var' într-un apel recursiv.

Get și Put

Procedurile standard de I/E Get și Put nu sînt implementate, dar procedurile Read și Write au fost extinse pentru a satisface toate cerințele operațiilor de I/E. Motivele acestei decizii sînt trei: Read și Write sînt mai rapide, memoria pentru variabile este mai mică deoarece nu se mai folosesc variabile buffer de fișier, iar procedurile Read și Write sînt mai flexibile și mai ușor de înțeles ca Get și Put.

Instrucțiuni Goto

O instrucțiune goto nu poate face salt în afara blocului curent.

Procedura Page

Procedura standard Page nu este implementată, deoarece sistemul de operare CP/M nu definește un caracter de avans la pagina nouă.

Variabile împachetate

Cuvîntul rezervat 'packed' nu are nici un efect în Turbo-Pascal, dar este admis. Împachetarea se face automat oricînd este posibilă. Din același motiv nu sînt implementate procedurile standard Pack și Unpack.

Parametri proceduri

Nu se pot transmite proceduri sau funcții ca parametri la apelarea unor proceduri sau funcții.

Mesaje de eroare la execuție

Erorile apărute în cursul execuției produc oprirea programului și afișarea unui mesaj de forma următoare:

Run-time error NN, PC=adr

Program aborted

unde NN este codul numeric al erorii, iar

'adr' este adresa de memorie unde s-a produs eroarea. Ambele numere sînt în baza 16.

01 Depășire la calcule cu reali (în virgula mobilă)

02 Încercare de împărțire la zero

03 Argument negativ la funcția sqrt de extragere radical

04 Argument negativ sau zero la funcția de logaritmare Ln

10 Eroare de lungime șir de caractere:

1) Rezultatul concatenării a două șiruri are mai mult de 255 de caractere;

2) Conversie șir de lungime diferită de 1 în caracter

11 Indice incorect într-un șir

Expresie indice în afara domeniului 1...255 la apelarea procedurilor Copy, Delete, Insert.

90 Indice în afara limitelor

Expresie indice la un element de tablou în afara domeniului declarat pentru acel tablou

92 Întreg prea mare

La conversie din real în întreg cu Trunc sau Round rezultă o valoare în afara domeniului —32768...32767

F0 Fișier cu segment de program (overlay) negăsit

FF Memorie insuficientă la alocare dinamică.

Apel la procedura New sau procedura recursivă și nu mai există memorie liberă între memoria 'heap' (adresa HeapPtr) și vârful stivei de recursivitate (adresa RecurPtr)

Mesaje de eroare la intrări—ieșiri

Dacă se produce o eroare la operații de intrare—ieșire și este activă directiva I de verificare automată a rezultatului I/E atunci programul se oprește și se afișează următorul mesaj de eroare:

I/O error NN, PC=adr
Program aborted

unde NN este codul numeric al erorii de I/E, iar 'adr' este adresa de memorie unde s-a produs eroarea (ambele în hexazecimal). Dacă este dezactivată verificarea operațiilor de I/E, {\$I-}, atunci programul nu se oprește, dar nu mai trebuie executate alte operații de I/E pînă cînd nu se apelează funcția I/Oresult pentru examinarea rezultatului operației de I/E; în caz contrar programul se poate 'pierde' într-o altă eroare de I/E.

01 Fișier inexistent

Nu există un fișier cu numele dat în procedurile Reset, Erase, Rename, Execute, Chain.

02 Citire din fișier ne-deschis

1) Încercare de citire (cu Read sau Readln) dintr-un fișier care nu a fost deschis anterior prin Reset sau Rewrite

2) Încercare de citire dintr-un fișier text deschis prin Rewrite și deci gol

3) Încercare de citire de la dispozitivul LST:, care se poate folosi numai pentru scriere.

03 Scriere în fișier ne-deschis

1) Încercare de scriere (cu Write sau Writeln) într-un fișier care nu a fost deschis prin Reset sau Rewrite,

2) Încercare de scriere într-un fișier text

deschis cu Reset.

3) Încercare de scriere la dispozitivul logic KBD:, care se poate folosi doar pentru citire.

04 Fișier ne-deschis

Încercare de acces (cu BlockRead sau BlockWrite) la un fișier, fără un Reset sau Rewrite anterior pe acel fișier.

10 Eroare de format la citire numere

Șirul citit dintr-un fișier text într-o variabilă numerică nu respectă formatul pentru contante numeerice

20 Operație interzisă la un dispozitiv logic

Încercare de utilizare proceduri Erase, Rename, Execute sau Chain pentru un fișier asociat unui dispozitiv logic.

21 Operație interzisă în modul direct

Încercare de apelare Execute sau Chain dintr-un program executat în mod direct, adică prin comanda Run din Turbo cu opțiune de compilare în memorie.

22 Atribuire la fișiere standard prin Assign

90 Lungime de înregistrare incorectă

Lungimea înregistrării la o variabilă fișier diferă de lungimea efectivă a înregistrărilor din fișierul extern cu care se asociază variabila.

91 Poziționare prin Seek dincolo de sfîrșitul fișierului

99 Sfîrșit de fișier neașteptat

1) La citirea dintr-un fișier text nu s-a găsit caracterul Ctrl-Z și s-a ajuns la sfîrșitul fizic al fișierului

2) Încercare de citire după sfîrșitul unui fișier

3) Nu se poate citi sectorul următor din fișier cu Read sau cu BlockRead.

F0 Eroare la scriere pe disc

Disc plin la încercarea de extindere a unui fișier. Această eroare poate apare la scriere (Write, BlockWrite, Flush) dar și la închidere de fișier sau la o citire precedată de golirea zonei tampon pe disc.

F1 Tabela directoare plină

Nu mai este loc în tabela directoare a discului la crearea unui nou fișier cu Rewrite (prea multe fișiere).

F2 Fișier prea mare

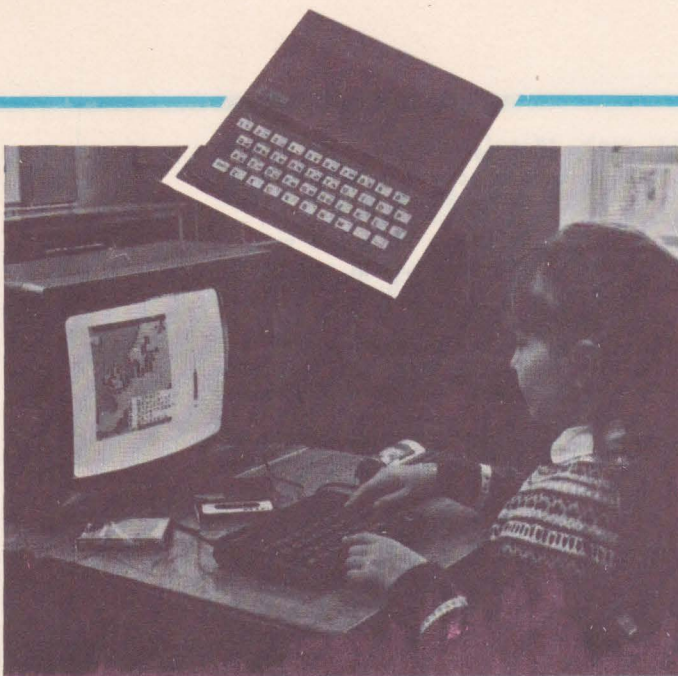
Încercare de scriere a unei înregistrări cu număr mai mare de 65535.

F3 Prea multe fișiere deschise

FF Fișier dispărut

Încercare de închidere, prin Close, a unui fișier negăsit în tabela directoare, de exemplu datorită schimbării discului.

Info
Club

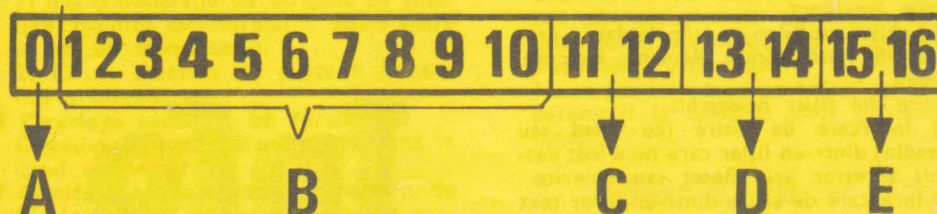


Ion DIAMANDI
Florin VASILIȚĂ

Utilizarea rutinei de citire din memoria - ROM

Laborator Spectrum Sinclair

Informațiile referitoare la un fișier salvat pe caseta magnetică (tipul fișierului, lungimea sa etc.) se regăsesc în partea de antet („header”) care însoțește fișierul. Într-adevăr, fișierele sînt separate în două părți, fiecare avînd un format diferit. Prima parte sau antetul fișierului conține diverse informații relative la fișier. Ea ocupă 17 octeți organizați în felul următor:



A. Tipul fișierului: 0=program BASIC („program”); 1=fișier de date numerice („number array”); 2=fișier de date alfanumerice („character array”); 3=program cod mașină („bytes”); 4=program cod mașină creat prin Editor/Asamblor. **B.** Numele fișierului. **C.** Lungimea fișierului salvat: - lungimea programului în octeți împreună cu variabilele (pentru tipul 0-program BASIC); -lungimea programului sau fișierului cod mașină (pentru tipul 1, 2 sau 3). **D.** Linia de autostartare (pentru tipul 0-program BASIC). Adresa de implantare a programului (pentru tipul 3-program în cod mașină). (Dacă tipul este 1 sau 2-fișier de date-nu se pune nimic.) **E.** Lungimea programului BASIC fără variabile. (Dacă tipul este 1, 2 sau 3 nu se pune nimic.)

Partea a doua a fișierului conține datele propriu-zise ale fișierului. Pentru un fișier care memorează un program în cod mașină, această parte va conține toți octeții programului.

Interfața cu casetofonul. Bitul 3 al portului 254 este utilizat pentru scriere (OUT) iar bitul 6 este utilizat pentru citire (IN). Acești doi biți sînt înlocuiți de rutinele de scriere și citire casetă care sînt situate în memoria ROM și care permit scrierea sau citirea de pe casetă a unui număr de octeți. Numărul de octeți, precum și poziția sînt fixate de utilizator.

Rutina de scriere este situată la adresa 4C2H (1218). La început, IX va trebui să conțină adresa de început a zonei de transferat, iar DE, numărul de octeți care se transferă. Registrul A va avea valoarea 0 (dacă se dorește scrierea unui antet de fișier) sau FFH (dacă se dorește scrierea corpului unui fișier). Următoarea suită de instrucțiuni va permite scrierea uneia din părțile fișierului pe caseta magnetică:

R1

Rutina de citire este situată la adresa 556H (1366) și necesită aceleași valori de intrare. În plus, va fi necesar ca indicatorul Carry să fie poziționat pe 1. Următoarea suită de instrucțiuni va permite citirea de pe casetă a unei părți a fișierului:

R2

lată și suita de instrucțiuni (dezasamblate cu MONS) care permit citirea informațiilor din antet referitoare la fișier (pe prima coloană, adresele instrucțiunilor în zecimal; a 2-a coloană, codurile instrucțiunilor în hexazecimal; a 3-a coloană, mnemonicele de instrucțiuni cu operanzi în zecimal; a 4-a coloană, comentarii):

R3

R1

```
LD IX , adresa ; adresa de inceput a zonei
; care contine octetii de transferat
LD DE , numar ; numarul de octeti de transferat
LD A , cod ; 00 sau FFH in functie de partea de transferat
; (antet sau fisier propriu-zis)
CALL 4C2H ; scriere
```

R2

```
LD IX , adresa ; adresa zonei in care vor fi transferati
; octetii cititi
LD DE , numar ; numarul de octeti de citit
LD A , cod ; 00 sau FFH in functie de partea de citit
; (antet sau fisier propriu-zis)
SCF ;
CALL 556H ; citire
```

R3

```
27986 37 SCF ; set carry flag
27987 3E00 LD A , 00 ; se incarca acumulatorul cu 0
27989 DD21606D LD IX , 27000 ; se incarca IX cu adresa zonei
; in care vor fi transferati octetii
; (28000)
27993 111100 LD DE , 00017 ; se incarca DE cu 17 (numarul de
; octeti ai antetului)
27996 CD5605 CALL 01366 ; se apeleaza rutina de citire din ROM
27999 C9 RET
```

Următorul program reprezintă o aplicație care va utiliza informațiile extrase din antetele fișierelor existente pe caseta magnetică prin intermediul rutinei de citire casetă prezentate anterior. Programul va crea un fișier de date pentru ținerea evidențelor referitoare la programele conținute într-o bibliotecă personală.

Rutina de citire a informațiilor din antet se va localiza la adresa **27986** (vezi linia 1) și va fi apelată din linia 150.

Rutina de citire se introduce din **BASIC** prin intermediul unei linii **DATA** (linia 30) care conține codificarea instrucțiunilor în cod mașină și a operanzilor în zecimal. Se poate observa că: 55 = 37 H; 62 = 3EH etc., regăsindu-se astfel codurile instrucțiunilor în hexazecimal din programul dezasamblat (coloana a 2-a). Datele din linia **DATA** au fost introduse ca șiruri de caractere, urmînd a fi transformate, apoi, în program în valori numerice prin intermediul funcției **VAL**. În acest mod se realizează o economie de memorie.

În linia 50 se realizează o rezervare de memorie începînd de la adresa **28 000** (adică de unde se termină programul în cod mașină) și de la această adresă se va memora fișierul de date cu informațiile conținute în antetele citite.

Utilizare. Programul prezentat este un program utilitar cu ajutorul căruia se poate ține evidența înregistrărilor pe casete magnetice. Memorarea informațiilor (evidențelor) se va face tot pe caseta magnetică.

Programul prezintă un tablou pe care vor fi afișate informațiile. După lansarea în execuție a programului se pornește casetofonul pentru parcurgerea casetei înregistrate cu programe. Programul va afișa (eventual pe mai multe pagini — ecrane — care conțin cîte 9 titluri) denumirea fișierului, tipul (program **BASIC** — **Prog** —, program cod mașină — **Bytes** — fișier de date numerice — **Num A** — sau fișier de date alfanumerice — **Chr A**), lungimea fișierului și linia de autostartare (pentru programe **BASIC**) sau adresa de implantare a programului (pentru programe în cod mașină).

La terminarea parcurgerii unei casete sau cînd utilizatorul dorește se va acționa orice tastă (dar numai pînă la 4-5 secunde de la afișarea informațiilor referitoare la fișierul citit) și se va intra în al doilea mod de lucru, ale cărei opțiuni sînt: **S** (Sus) pentru afișarea paginii inferioare, **V** (Virf), pentru afișarea primei pagini, **J** (Jos) pentru afișarea paginii superioare, **R** (Reîntoarce) — pentru intrare în primul mod de lucru, **T** (Tipărire) — pentru imprimare

rea fişierului la imprimantă. În cazul tastării lui R (Reîntoarcere) apar alte 3 opţiuni: S (Salvare) pentru salvarea programului împreună cu fişierul creat, N (Nou) — pentru ştergerea fişierului creat şi trecerea la o nouă casetă, C (Continuare) — pentru explorarea în continuare a casetel.

În cazul testării lui T (Tipărire) apare două opţiuni şi anume: I (Întreg) — pentru tipărirea la imprimantă a tuturor paginilor fişierului (întreg fişier) şi P (Pagină) — pentru tipărirea numai a paginii curente.

Se recomandă ca fişierul creat să se salveze la începutul casetei respective.

Menţionăm faptul că programul de-

codifică numai informaţiile din antete. Dacă în cazul unor fişiere antetul lipseşte (aceste fişiere sînt denumite de unele programe de copiere „headerless”) atunci aceste fişiere vor fi ignorate de programul utilitar. Introducerea valorii 5 la adresa 28 000 (linia 1), care reprezintă începutul fişierului creat, şi, apoi (eventual) la adresa 28 000 + multiplu de (17+1) se realiază în program tocmai în scopul ignorării fişierelor care nu sînt de tipul 0, 1, 2, 3 sau 4.

La o leşire accidentală programul se poate relua cu CONTINUE sau cu RUN, caz care este echivalent cu opţiunea N pentru casetă nouă.

```

1 CLEAR 27985: POKE 28000,5
2 LET c8=9: LET p=1: LET c=1
8 INPUT "Cod Casetă (2 cifre)
":y
10 BORDER 0: PAPER 0: INK 7: C
LS
20 DATA "55","62","0","221","3
3","96","109","17","17","0","205
","86","5","201"
30 GO SUB 40: GO TO 80
40 RESTORE 20: FOR a=27986 TO
27999: READ a$: POKE a,VAL a$: N
EXT a
50 DEF FN p(a)=PEEK a+256*PEEK
(a+1)
60 DEF FN a(p,c)=28000+17*(c-1
)+17*c8*(p-1): RETURN
70 INPUT :: PRINT #1: PAPER 1:
FLASH 1:" Opriti casetofonul
": FLASH 0: TAB 25:"PAG ":p:
RETURN
80 GO SUB 500
150 RANDOMIZE USR 27986: GO SUB
780: IF PEEK FN a(p,c)=5 THEN G
O TO 150
160 GO SUB 600
170 GO SUB 800
180 IF c<c8 THEN LET c=c+1: LET
et=150: GO TO 200
190 IF c=c8 THEN LET p=p+1: LET
c=1: LET et=80: GO TO 200
200 POKE FN a(p,c),5: POKE 2799
2,INT (FN a(p,c)/256): POKE 2799
1,FN a(p,c)-256*INT (FN a(p,c)/2
56)
210 GO TO et
500 REM Desen cap de tabel
520 CLS : PRINT PAPER 1:"DIR*Ca
seta ";y:" Adr.Lungime.Auto"
530 PLOT 0.165: DRAW INK 2:255.

```

```

0
540 PLOT 0.166: DRAW INK 2:255.
0
550 INPUT :: PRINT #1: PAPER 1:
FLASH 1:"Lasati casetofonul por
nit!": FLASH 0: TAB 25:"PAG ":p:
RETURN
570 LPRINT "DIR*Caseta ";y:" Ad
r.Lungime.Auto"
580 FOR i=0 TO 31: LPRINT "--":
NEXT i: LPRINT : RETURN
600 REM Decodificare heder
602 GO SUB 610: GO SUB 680: RET
URN
610 LET ad=FN a(p,c)
620 LET t=PEEK ad
630 DIM n$(10)
640 FOR i=1 TO 10: LET n$(i)=CH
R$ PEEK (ad+i): NEXT i
650 LET l=FN p(ad+11)
660 LET s=FN p(ad+13)
670 LET x=FN p(ad+15)
672 RETURN
680 GO SUB 700+t: GO TO 710
700 PRINT "Prog. ": RETURN
701 PRINT "Num.A ": RETURN
702 PRINT "Chr.A ": RETURN
703 PRINT "Bytes ": RETURN
705 RETURN
710 PRINT PAPER 1:n$: PAPER 0:
":
720 IF t=3 THEN PRINT s:".":l:
GO TO 770
730 PRINT l:
740 IF t>0 THEN GO TO 770
750 PRINT "/" :x:
760 IF s>=0 AND s<10000 THEN PR
INT "/" :s:
770 PRINT "": RETURN
780 BEEP .1.-16: BEEP .1.32: BE

```



```

EP .1.31: BEEP .1.48: RETURN
805 INPUT :: PRINT #1: FLASH 1:
PAPER 2:"Pentru OPRIRE actionat
i o TASTA URGENT ! ": FLASH
0: PAPER 1:TAB 25:"PAG ":p
810 FOR w=1 TO 180
820 IF INKEY$="" THEN GO TO 100
0
830 LET a$=INKEY$: GO SUB 780
840 LET cc=c: LET pp=p
850 INPUT :: PRINT #1: FLASH 1:
" ": FLASH 0: PAPER 1:"SusMinfDo
s ReitoarcereTiparine":TAB 25:"P
AG ":p
860 IF INKEY$="" THEN GO TO 860
870 LET a$=INKEY$: GO SUB 780
880 IF a$="r" THEN LET p=pp: GO
SUB 500: GO SUB 70: FOR c=1 TO
cc: GO SUB 600: NEXT c: LET c=cc
: INPUT "salvare sau noua caseta
sau":TAB 2:"continua?":a$: I
F a$="s" THEN GO SUB 780: GO SUB
9999
881 IF a$="n" THEN GO SUB 780:
RUN
884 IF a$="t" THEN : GO TO 2500
890 IF a$="s" THEN LET p=p-1: G
O TO 920
900 IF a$="v" THEN LET p=1: GO
TO 920
904 IF a$="c" THEN GO TO 914
910 IF a$="j" THEN LET p=((p AN
D p=pp)+((p+1) AND p(pp))
912 GO TO 920
914 LET p=pp: GO SUB 950: LET c
=cc: GO TO 1030
920 IF p<1 THEN LET p=1
930 IF p>pp THEN LET p=pp
940 GO SUB 500: GO SUB 70: GO S
UB 960: GO TO 850
950 GO SUB 500
960 FOR c=1 TO ((c8 AND p(pp)+
cc AND p=pp))
970 GO SUB 600
980 NEXT c
990 RETURN
1000 NEXT w
1010 GO SUB 550
1020 RETURN
1030 LET w=w+1: IF w<179 THEN GO
TO 1030
1040 GO TO 1000
1100 REM Tiparine la imprimanta
1110 GO SUB 610
1120 GO SUB 1200+t
1130 GO TO 1210

```

```

1200 LPRINT "Prog. ": RETURN
1201 LPRINT "Num.A ": RETURN
1202 LPRINT "Chr.A ": RETURN
1203 LPRINT "Bytes ": RETURN
1205 RETURN
1210 LPRINT n$: " ":
1220 IF t=3 THEN LPRINT s:"," :1:
: GO TO 1270
1230 LPRINT l:
1240 IF t>0 THEN GO TO 1270
1250 LPRINT "/" :x:
1260 IF s)=0 AND s<10000 THEN LP
RINT "/" :s:
1270 LPRINT : RETURN
2500 INPUT #1:"Tiparine integ f
isierul sau numai pagina cure
nta?":a$
2510 IF a$="p" THEN GO SUB 780:
GO SUB 2600
2520 IF a$="i" THEN GO SUB 780:
GO SUB 2700
2530 GO TO 850
2600 GO SUB 570: FOR c=1 TO ((c8
AND p(pp)+(cc AND p=pp)): GO SU
B 1100: NEXT c
2610 LPRINT ':TAB 25:"PAG.":p: G
O SUB 580: LPRINT "'
2620 LET c=((c8 AND p(pp)+(cc AN
D p=pp)): RETURN
2700 FOR p=1 TO pp: GO SUB 2600:
NEXT p: LET p=pp: RETURN
9000 BORDER 0: PAPER 0: INK 7: C
LS : PRINT AT 10,2: PAPER 2: FLA
SH 1:"Fisierul Director se incar
ca": GO SUB 550: LOAD ""CODE
9010 GO SUB 950: GO SUB 40: GO T
O 180
9999 LET y$="DIR*Cas "+STR$ y: S
AVE y$ LINE 9000: SAVE "director
"CODE 28000,17*c+17*c8*(p-1): RE
TURN

```

DIR*Caseta 93 Adr.Lungime.Auto

Prog. ManicMiner	94/94/1
Bytes mml	22528.768
Bytes MM2	32768.32768
Prog. Android 1	977/977/200
Bytes M/C	24960.7508
Bytes Screen	16384.6912
Prog. CASETA PR1	63/63/1
Bytes CASETA SR1	16384.6912
Prog. CASETA PR1	546/467/10

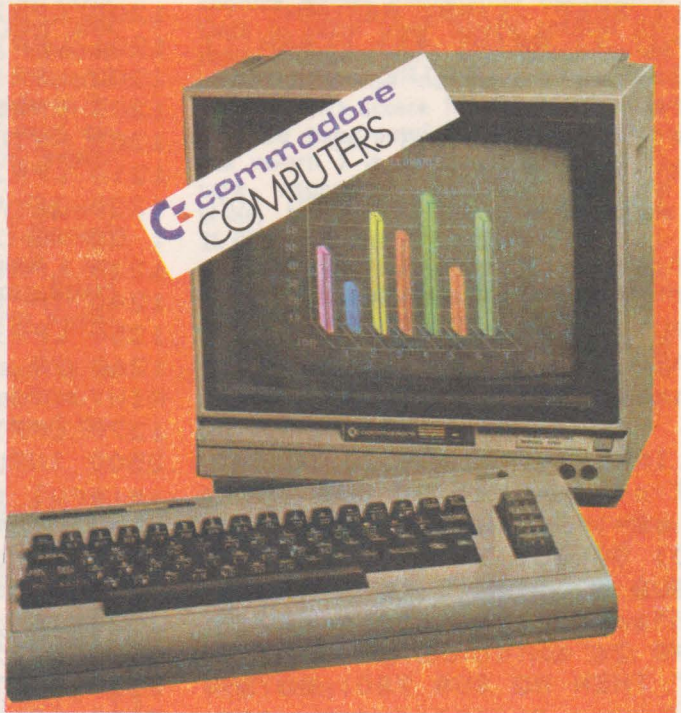


Călin OBRETIN

INFORMATICA -un joc serios !

Toți cei ce folosesc calculatorul în timpul liber cunosc cât de dificil este să ajungi la sfârșitul unui program, mai ales când acesta este de tipul „trage înaintea adversarului”.

Cred că este util să știm câteva POKE-uri care să ne ajute să urmărim un joc pînă la capăt. Aceste instrucțiuni se introduc după ce s-a încărcat programul de pe bandă sau disc; după această operație se tastează comanda RUN.



JOCUL

Quest of tires
Donkey Kong
Sea fox
Shamus I
Shamus II
Zaxxon
Lady TUT

POKE

7341,199
12118,234
7337,173
18486,162
23558,169
15475,238
11353,X
2392,5

EFFECT

vieți infinite
idem
idem
idem
idem
X-nr. de vieți
vieți infinite

Tot pentru posesorii de Commodore 64 indicăm și alte instrucțiuni utile ce se pot folosi la crearea de programe:

SYS 65499 — resetează ceasul (intern) la 00:00;

SYS 42115 — cursorul funcționează fără READY;

SYS 44808 — apare mesajul ?SYNTAX ERROR;

WAIT 653,1..2..4 — calculatorul așteaptă apăsarea tastelor SHIFT;C =, respectiv CTRL;

WAIT 203,63 — se așteaptă apăsarea tastei cu codul 63.

Pușini cunosc că un Commodore 64

poate funcționa în două moduri:

SLOW

POKE 53265,PEEK-(53265) AND 239 :
POKE 53296,PEEK-(53296) OR 1

FAST

POKE 53296,PEEK-(53296) AND 254 :
POKE 53265,PEEK-(53265) OR 16

Tot în mod FAST pot lucra și cei ce au un Commodore 128

prin simpla comandă:

POKE 53424, PEEK(53424) OR 3

Și pentru a încheia această scurtă listă, câteva facilități ale microprocesorului 6510 la

scrierea programelor:

- programe BASIC alocate în zona C000

POKE 44,192 :

POKE 56,208 :

POKE 49152,0 : NEW

- dispariția cursorului SYS 64328 (comanda se restabilește cu tastele RUN/STOP-RESTORE)

- cursor rapid

POKE 6325,5...255

- dezactivare (activare) tasta RUN/STOP

POKE 788,52...49

- dezactivare (activare) tasta RESTORE

POKE 792,193...71

- cursor static 4 sec.

POKE 651,255.

Și o ultimă instrucțiune: când vrem să citim un fișier de pe bandă, iar această citire se face în mijlocul unui alt program ce apelează acest fișier (deci citirea se face în timpul rulării unei secvențe BASIC) se folosește:

POKE 3600,266 :SYS7082 : LOAD

Această comandă are ca efect rularea în continuare a secvenței BASIC fără întreruperea ei, fișierul dorit fiind în memoria calculatorului.

Virgil IONESCU
 Radu CONSTANTINESCU

Interfața serială pentru calculatoare compatibile HC-85

Utilizarea protocolului DTR permite cuplarea interfeței cu imprimanta numai prin trei linii (masă, linia de transmisie a datelor, linia de recepție a semnalului DTR). Transmisia datelor (TxD) se face prin intermediul bitului D1 al portului de adresă 1 iar recepția semnalului DTR folosește bitul D5 al aceluiași port.

Partea hardware (fig. 1), utilizând circuite nespecializate, accesibile, a fost concepută pentru o cât mai mare simplitate, folosind un minimum de componente. Decodificarea liniilor de comandă (IORQ, M1, WR, RD) și a liniei AO se face cu porți NAND (CDB 400). Pentru ieșirea datelor este utilizat un bistabil de tip D (7474), iar pentru intrare o poartă de mare impedanță (74125). Adaptarea la nivelurile cerute de standardul RS232C se realizează pe fiecare linie, cu câte un translaor de nivel (1488, 1489). Cuplarea cu calculatorul este asigurată de conectorul interfață (fig. 2), iar cu imprimanta, de un conector cu cel puțin 3 contacte. Corespondența dintre ieșirile interfeței și intrările imprimante este următoarea:

Interfață...Imprimantă
 TxR1...RxD
 DTR1...DTR
 GND...GND

Rezultatele unor programe de calcul, graficele obținute în urma prelucrării unor date, fișierele create cu editoare de text, listingurile unor programe lungi sînt greu de urmărit numai pe display. O imprimantă cuplată la calculator poate ușura foarte mult munca. Deoarece HC-85 nu are prevăzută constructiv o interfață care să permită transmisia datelor către o imprimantă, prezentăm în continuare un astfel de dispozitiv. Principalele sale caracteristici sînt:

Tip: serială, nestandard

Compatibilitate: RS232C

Protocol: DTR (Data Terminal Ready)

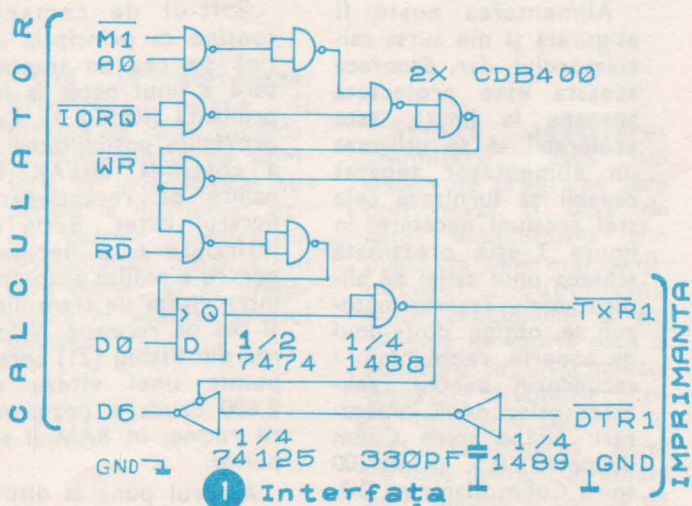
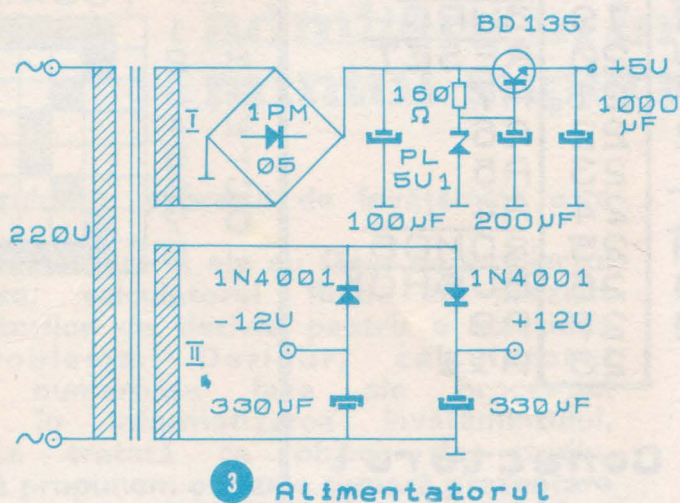
Viteza de transfer: 9600 bauds

Formatul transmisiei:

1 bit de start;

8 biți de date, fără paritate;

1 bit de stop.



Atelier Spectrum Sinclair

A15	1	A14
A13	2	A12
D7	3	+5U
	4	
////	5	////
D0	6	GND
D1	7	GND
D2	8	CLK
D6	9	A0
D5	10	A1
D3	11	A2
D4	12	A3
INT	13	IORGE
NMI	14	
HALT	15	
MREQ	16	
IORQ	17	
RD	18	
WR	19	BURQ
-5U	20	RESET
WAIT	21	A7
+12	22	A6
	23	A5
M1	24	A4
RFSH	25	ROMCS
A8	26	BUSHCK
A10	27	A9
	28	A11

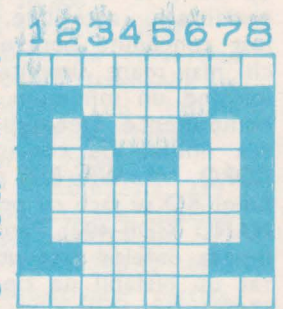
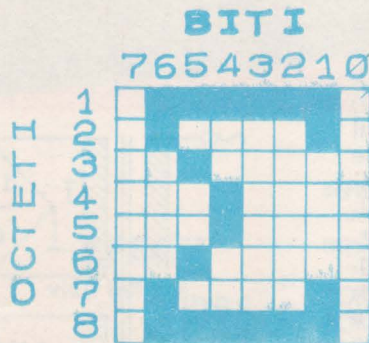
2 Conectorul interfață

Alimentarea poate fi asigurată și din sursa calculatorului, dar, deoarece aceasta este proiectată aproape la limită, este preferabil să se utilizeze un alimentator separat capabil să furnizeze cele trei tensiuni necesare. În figura 3 este prezentată schema unui astfel de alimentator. Transformatorul se obține dintr-unul de sonerie, rebobinându-i secundarul pentru realizarea celor două înfășurări: I=156 spire CuEm diametru 0,5 și II=200 spire CuEm diametru 0,2.

```

PRINT CALL #1F54; BREAK
JR C, PR0
RST #08
DEFB 12
PR0 IN A, (1)
BIT 5, A
JR NZ, PRINT
DI
XOR A
CALL PR_OUT
LD A, B
LD B, 8
PR1 CALL PR_OUT
RRA
DJNZ PR1
LD A, 1
CALL PR_OUT
EI
RET
PR_OUT OUT (1), A
PUSH BC
LD B, 21
DJNZ TIMER
POP BC
RET
TIMER

```

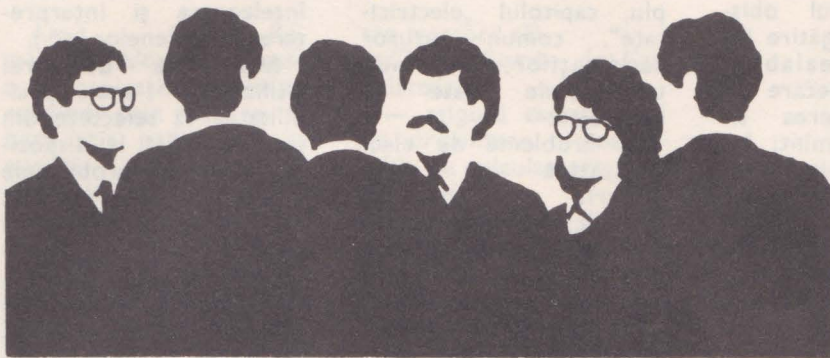


Soft-ul de comandă conține ca principală rutină pe cea de transmitere a unui octet la imprimantă (listing 1). Este prevăzută posibilitatea de a comanda BREAK înainte de recepționarea fiecărui octet. Bucula de întârziere este necesară pentru a realiza adaptarea între viteza de transmisie și cea de recepție. Valoarea din listing (21) corespunde unei viteze de 9 600 bauds și poziționării rutinei în RAM-ul superior.

Autorul pune la dispo-

ziția doritorilor programe care exploatează facilitățile imprimantei matriciale ROBOTRON K6313-seriale, folosibile și pentru alte imprimante respectând particularitățile acestora.

Promitem să revenim cu aplicații interesante: modificarea programului Tasword II pentru LPRINT și COPYCOLOR — un program de transpunere în tonuri de gri a culorilor ecranului, sau cu alte sugestii pe care le așteptăm din partea dv.



Liviu
DUMITRAȘCU
Stelian
GUȚU

ADMITEREA - produs software educațional pentru pregătirea candida- ților la concursul de admitere în învăța- mîntul superior

Importanța calculatorului în procesul de învățămînt este un fapt de netăgăduit.

Argumentele pro sînt numeroase și ele au făcut deja obiectul a numeroase anchete, calculatorul fiind în prezent în centrul atenției factorilor de decizie pentru a soluționa optim această problemă. Desigur, calculatorul poate interveni în numeroase faze ale procesului de învățămînt, atît în informatizarea învățămîntului, cît și în informatica tratată ca obiect de studiu.

În cele ce urmează, vă propunem o foarte sumară prezentare a unui pachet de programe (în dBASE) dedicate unui subiect foarte „fierbinte”: admiterea în învățămîntul superior.

Scop: Pregătirea asistată de calculator a examenului de admitere în învățămîntul superior la disciplina „fizică”. Se utilizează microcalculatoare de tip M118, TPD, JUNIOR, CUB-Z sub sistemul de operare CP/M.

Cui se adresează: Produsul poate fi utilizat de către candidații la concursul de admitere în învățămîntul superior la disciplina „fizică”, cadrele didactice din învățămîntul liceal, elevii care doresc să se pregătească în vederea acestui examen.

Scurtă descriere: „Admitere” este un sistem de programe care gestionează o bază de date semnificativă din punct de vedere al conținutului, cuprinzînd toate problemele și subiectele de teorie date la admitere în învățămîntul superior, disciplina „fizică” în ultimii zece ani în țară. Interogarea bazei de date poate fi efectuată ușor, simplu și comod de

■ Aplicații pentru toți

către utilizatorul obișnuit, fără o pregătire informatică prealabilă. Punctul de plecare îl constituie alegerea tipului de învățământ: învățământul tehnic, învățământul universitar (profilul fizică), învățământul medico-farmaceutic (medicină și stomatologie). Învățământul tehnic este structurat pe următoarele profiluri:

T1: mecanic, electric, energetic, mine, petrol, metalurgie, construcții, geodezie, tehnologie și chimia texturilor, tehnologia și chimia produselor alimentare, forestier (specializare industria lemnului), geologie.

T2: chimie și mecano-chimic.

T3: învățământul de subingineri (trei ani) — toate profilurile, specializările din învățământul pedagogic de chimie-fizică (trei ani) și matematică-fizică (trei ani).

Învățământul medico-farmaceutic conține profilurile medicină și stomatologie. Mai departe, problemele aparținând profilurilor menționate sînt structurate pe capitole: mecanică, fizică moleculară și căldură, electricitate, optică (numai pentru profilurile medicină și stomatologie, precum și învățământul universitar, profilul fizică), fizică atomică și nucleară (pentru învățământul universitar, profilul fizică). Fiecare capitol este, la rîndul său, structurat pe clase de probleme. De exem-

plu, capitolul „electricitate”, comun tuturor candidaților, cuprinde următoarele clase de probleme:

1. Probleme de electrostatică cu: cîmp electric și potențial electric; condensatoare și energia cîmpului electric dintre armături; mișcarea particulelor în cîmp electric.

2. Probleme de electrocinetică cu: legea lui Ohm și teoremele lui Kirchoff; bilanțul energetic; legile electrolizei.

3. Probleme de magnetism: de cîmp magnetic al curentului electric; de mișcare a particulelor în cîmp electric și magnetic; de inducție electromagnetică.

4. Probleme cu circuite în curent alternativ.

5. Probleme cu tuburi electronice și semiconductoare.

6. Probleme combinate.

Fiecărei probleme îi este atașat un grad de dificultate de la 1-3 corespunzător următoarelor clasificări:

Gradul 1: probleme relativ simple care se rezolvă aplicînd, de regulă, formula (majoritatea problemelor date la examenele de subingineri);

Gradul 2: probleme mai complexe care se rezolvă aplicînd o anumită succesiune de formule;

Gradul 3: probleme la care înainte de a aplica formulele cunoscute, trebuie efectuate anumite artificii (de calcul,

înțelegerea și interpretarea fenomenelor etc.).

Mod de utilizare: Utilizatorul are posibilitatea să selecteze din tezaurul aflat la dispoziția sa toate problemele aparținînd unui interval sau pe ultimii n ani (n cuprins între 1-10) specificînd capitolul (profilul, tipul de învățământ) și gradul de dificultate. După ce va încerca rezolvarea, utilizatorul va putea confrunta rezultatul cu răspunsul oferit de sistem. Menționăm că, începînd cu anul 1988, problemele conțin și baremurile cu punctajul fixat de minister.

Avantaje: Tezaur complet pe ultimii zece ani (teorie plus probleme); acoperă toate profilurile; grad de detaliere dorit de utilizator; grad de dificultate adaptat nivelului de pregătire al utilizatorului; răspunsuri exacte garantate de sistem.

Mod de livrare. Programele și baza de date se află pe suport magnetic structurate pe trei domenii: învățământul tehnic, învățământul universitar (profilul fizică) și învățământul medico-farmaceutic. Baza de date (pe ultimii zece ani) conține atît subiectele de teorie (fără rezolvări) cît și problemele propriu-zise cu soluțiile lor.

Dischetele sînt însoțite de un manual care cuprinde: descrierea produsului, modul de utilizare și programele-sursă.

Redacția noastră intenționează să publice un supliment dedicat acestei baze de date, în ipoteza în care cerința va fi suficient de mare!

Așadar, după ce veți lua cunoștință despre această inedită bază de date, extrem de necesară tuturor candidaților (noi o oferim „în lucru” pentru anul școlar 1990-1991), așteptăm răspunsul dv. făcînd mențiunea pe plic: „ADMITERE”.

Vă mulțumim!

Art.1 — Clubul Român de Calculatoare este o organizație apolitică, reunind prin afiliere liberă atât persoanele ce posedă un calculator personal, cât și cei ce doresc inițierea în acest domeniu.

Art.2 — Condiția unică de apartenență la Clubul Român de Calculatoare o constituie aprobarea statului organizației. (...)

Art.7 — Principalul obiectiv al CRC este liberele schimb de idei, opiniile, informații și soft în-

tre membrii săi.

Clubul Român de Calculatoare:

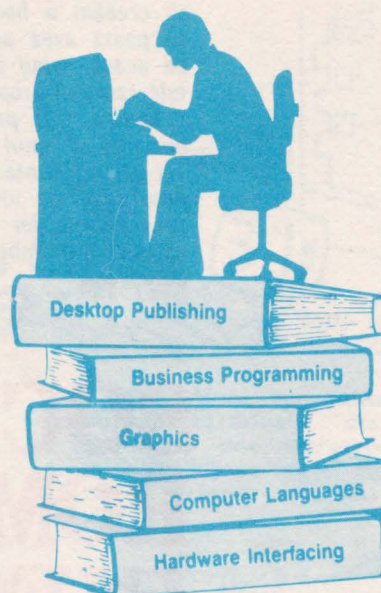
— asigură cadrul legal de organizare a posesorilor de calculatoare personale. (...)

— creează bancă de programe.

— creează și actualizează un sistem propriu de documentație. (...)

— apără interesele membrilor săi în relațiile cu alte cluburi.

(...)
Art.17 — Prezentul statut intră în vigoare astăzi 28.01.1990.



UN CLUB DE CARE ERA NEVOIE !

CLUBUL ROMÂN DE CALCULATOARE

Nu am făcut decît să prezint cîteva articole din statutul clubului nostru. Cred că o mică prezentare a sa este binevenită. În 1987 cu ajutorul conducerii Casei de Cultură a Studenților a luat ființă un cerc care se dorea de microinformatică. Timid la început, compus doar din 7 membri, cercul își desfășura activitatea duminica dimineața. Ne axam mai mult pe calculatoare tip Spectrum. Încet, încet, cercul și-a mărit numărul de membri. Au apărut tot mai mulți posesori de Commodore în detrimentul celor cu Spectrum. Au apărut și „rarități”: Atari 800, Atari 1024, Apple, Toshiba. Numărul de programe, atât distractive, cât și utilitare, a crescut aproape logaritmice.

Cum era și normal, după 22 decembrie, cercul s-a legalizat, devenind astăzi CLUBUL ROMÂN DE CALCULATOARE. Membri fondatori ai acestui club sînt: ing. Obretin Călin, ing. Gorodcov Mihaela, ing. Pîrlog Mircea, stud. Mașoiu Dan și stud. Florean Cătălin.

În prezent sînt înscrise în club 124 persoane și numărul lor crește cu fiecare ședință. Sînt persoane inoculate cu acest „microb” al calculatorului folosit acasă, ingineri, studenți, muncitori, copii, vîrstnici, adică majoritatea posesorilor de calculatoare personale din București. Avem și corespondenți din țară, membri ai clubului cu activitate permanentă.

Am reușit în acest interval de 3 ani să strîngem o bogată documentație atât

hard, cât și soft. Specialiștii clubului, fără falsă modestie, pot face față oricărei probleme; de fapt acest lucru este cunoscut de un număr tot mai mare de posesori de hard ce au avut probleme de rezolvat.

Ce se poate face la o ședință a clubului? În primul rînd schimb de programe; avem peste 2 500 de programe atât de Spectrum cât și Commodore, utilitare (turbo tape, turbo disc, procesoare de texte, soft pentru grafică și muzică) dar și programe distractive pentru toate genurile și toate vîrstele. Din păcate la noi în țară, majoritatea programelor ajung (sau ajungeau, pînă acum) cu o întîrziere de aproape un an față de data apariției lor în vest. Sperăm ca prin contactele stabilite de club cu cîteva firme și cluburi din exterior termenul acesta să se micșoreze.

O altă preocupare a noastră este actualizarea sistemului de documentație, sistem accesibil oricărui membru al clubului. Pentru cei interesați, avem ultimele cataloage de hard ale firmelor Apple, Macintosh, IBM precum și soft-ul acestora.

Dar, pe lîngă toate aceste preocupări de rutină, acum încercăm să inițiem prima rețea de calculatoare personale (la domiciliu), rețea ce va utiliza liniile telefonice existente. Avem deja acordul forului competent, MODEM-ul este realizat, chiar legătura între două Commodore a fost experimentată, mai rămîn mici amănunte de protocol de pus la punct. Dorim

Contact

să creăm o bancă de programe la care să poată avea acces orice membru CRC, de acasă, stînd comod în fotoliul din fața televizorului propriu.

Pentru toți prietenii, posesori sau neposessori ai unui calculator personal, care vor să ne contacteze pentru schimbul de programe, de idei, care vor să comunice preocupările lor din domeniu, care doresc să devină membrii CRC, îi rog să ne scrie pe adresa:

CLUBUL ROMÂN DE CALCULATOARE
P.O. Box 37-131
BUCUREȘTI.

precizînd în scrisoare tipul calculatorului cît și lista de programe ce le oferă și cea dorită în cazul schimbului de soft.

Pe cei din București îi așteptăm în fiecare sîmbătă la ora 11 la Casa de Cultură a Studenților, Calea Plevnei nr.61.

SOCIETATEA ROMÂNĂ DE INFORMATICĂ

În două adunări care au avut loc, în martie 1990, într-unul din marile amfiteatre ale Institutului Politehnic din București, s-a constituit **Societatea Română de Informatică**. Au participat specialiști de la Politehnică, de la ASE și de la Universitate, dar intenția este de a fi atrași în cadrul societății toți cei care, indiferent de domeniul în care lucrează, pot beneficia de aportul gîndirii algoritmice și utilizării calculatoarelor. S-a discutat un proiect de statut, s-a ales prin vot secret un consiliu de conducere și s-a decis ca în noiembrie 1990 să aibă loc o nouă adunare generală. Sediul Societății Române de Informatică este în sala EC106 de la Facultatea de Automatică a Institutului Politehnic București (Splaiul Independenței 313, sectorul 6), iar informațiile privind modalitatea de a deveni membru al SRI se pot obține la int. 320, la oricare dintre numerele de telefon 31.04.69, 31.40.10, 31.41.20.

Scopul este promovarea științei calculatoarelor, a metodelor de producere, prelucrare, stocare și transmitere a informației și stimularea educației în domeniul informaticii, la toate nivelurile, de la copii pînă la adulți. Societatea urmărește să contribuie la clarificarea rolului informaticii în societatea românească actuală, la studiul dezvoltării și proiectării, realizării și utilizării calculatoarelor și a tehnicilor de prelucrare a informației, la elaborarea metodelor și programelor corespunzătoare. De asemenea, societatea își propune să faciliteze schimbul liber

de idei, cunoștințe și informații din domeniul calculatoarelor, între diferite grupuri de interes din țară și din străinătate.

Dintre activitățile societății de informatică, menționăm: ● elaborarea și însușirea unor norme de conduită profesională ● reprezentarea opiniilor, a punctelor de vedere și a experienței membrilor săi în probleme legate de evoluția informaticii, în cadrul comisiilor de lucru pe probleme ● promovarea și stimularea cercetărilor originale din domeniu prin organizarea de conferințe, simpozioane, seminarii, expoziții, concursuri profesionale și premii acordate de societate ● implicarea acestora în elaborarea strategiei de informatizare a societății românești ● instituirea unor relații de acces specifice membrilor săi la bibliotecile de specialitate și alte centre de informare din țară și din străinătate ● stabilirea de contacte și colaborări cu alte societăți și instituții de specialitate ● organizarea de cursuri de specialitate și tabere de instruire în domeniul informaticii ● stimularea introducerii metodelor, ideilor și rezultatelor informaticii în științele naturii și în cele umaniste, în artă și literatură.

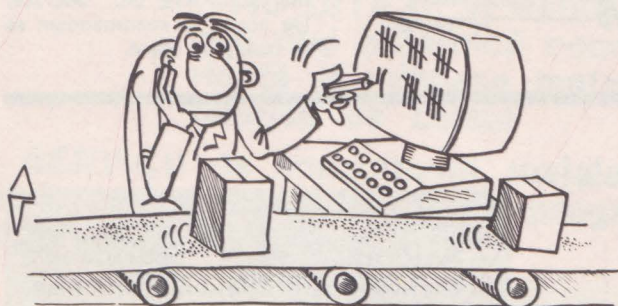
Societatea de Informatică va organiza, în cadrul ei, comisii de lucru pe probleme și grupuri de interes. Un punct important îl constituie faptul că poate deveni membru al său orice persoană, indiferent de cetățenie. Sperăm astfel să aducem în cadrul societății pe numeroșii informaticieni români care lucrează în străinătate.

INFO

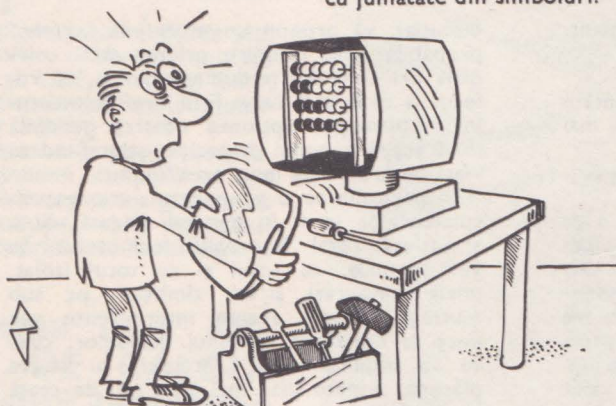
- sfaturi!
- previziuni!

1. Calculatoarele tind să devină din ce în ce mai mici, așa încât, dacă procesul continuă mereu, vor dispărea cu... desăvîrșire.

2. Perfecționarea continuă a programării va duce la scrierea de programe care să-și corecteze singure erorILELE...



3. Evoluția calculatoarelor, pe linie hardware, poate să ducă la apariția ordinatorilor unare, care oferă avantajul imediat: viteză dublă, lucrînd numai cu jumătate din simboluri.



4. Se întrezărește, am zice de mai multă vreme, dispariția FORTRAN-ului, dar credem că nu se realizează acest lucru pentru motivul că s-a propus a fi înlocuit de... MORTRAN.

5. Gurile rele (sau invidioșii) spun că o soartă similară o are și fratele COBOL, căruia să-i ia locul MORBOL-ul, și asta pentru a avea încă o aplicație a zicalii „să moară și capra con-fratelui”.

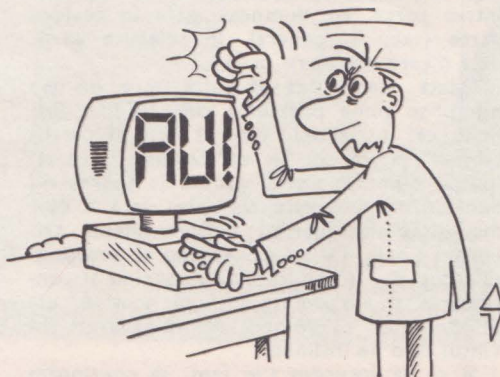


6. În Africa vor exista numai terminale display-color, cele alb/negru (a se citi alb pe negru) devenind interzise cu... desăvîrșire.



7. Atunci cînd păstrezi ceva în memoria calculatului, fii prevăzător și ține minte unde... ai memorat.

8. Fii precaut în activitatea de programare deoarece un program face ce-i spui și nu... ce-ai vrea să facă.



Umor

9. Multă, multă precauție în ce privește fenomenul GIGO (gunoi pui, gunoi obții) aplicat în programare.

10. Atenție la detectarea și corectarea erorilor unui program deoarece, în



multe cazuri, după ce se efectuează o corecție se deduce că, de fapt... era bine înainte.

11. Atenție, fără lălaială în activitatea de programare, deoarece un program poate fi considerat depășit înainte de a fi implementat.

12. Nu ștergeți un fișier decât după ce v-ați convins că... nu este al dumneavoastră.

13. Decît să vă întristați că nu știți să apelați un program dintr-o bibliotecă, mai bine... mirați-vă cum de a ajuns acolo.

14. Limbajul ASSIRIS nu a fost creat de asirieni și nici SOCRATE de... Socrate. De aceea vă recomandăm să le tratați ca atare.

Stelian NICULESCU

O posibilă discuție...

Axiomă: toate persoanele prezente sînt excluse/incluse, după cum, respectiv, este vorba despre aspecte negative/pozitive sau invers, cum vă convine.

Vă rog să-mi permiteți să nu mă prezint, cel puțin din trei motive:

- 1) n-ați putea face același lucru;
- 2) nu se întîmplă nimic rău dacă acceptăm ideea că ne cunoaștem mai mult ori mai puțin ... mult;
- 3) un al treilea motiv poate fi găsit de oricare dintre dumneavoastră.

Și ca să începem discuția, vă propun ca aceasta să fie de tip n-log. Dacă $n=1$ cădem peste monolog, ceea ce nu preferăm, căci are o singură parte, ceea ce implică monotonie. În cazul $n=2$ apare dialogul, care are două părți. Așadar, nu sîntem toți într-o parte (ca în cazul monologului). Și ca să fim mai expliciti, să considerăm că eu sînt într-o parte, iar dumneavoastră în cealaltă parte (sau, în general, în celelalte părți, dacă n este mai mare ca 2).

Odată stabilit că discuția este de tip n-log, se pune problema cine începe. Evident, cel mai simplu este să înceapă partea „într-o”, adică eu. De ce? Simplu. Dacă ați începe dumneavoastră, atunci ar apărea n-l lema. Cînd $n=2$ este clar, dar dacă n este mai mare sau egal cu 3 apare dilema, trilema ș.a.m.d. De aceea, vă rog să acceptați să încep eu, în felul acesta asigurînd și continuarea și sfîrșitul discuțiunii noastre, urmînd, totuși, ca dumneavoastră să aveți cuvîntul cînd ne întîlnim.

Și ca să începem (de fapt, să continuăm

discuția), vă propun un model de „vreme” probabilă la o întîlnire prietenească, colegială ori cum doriți dumneavoastră, cu referiri la ce fu și la ce va fi în jurul momentului în discuție, opțiunea noastră generală fiind aceea a bunei dispoziții, știut fiind că viața fiecăruia este mult prea finită.

În perioada ce o prognozăm, vremea calculatoarelor va fi în general plăcută, cu o atmosferă total favorabilă frumosului. Se vor semna, cu totul și cu totul izolat, unele innourări și/sau zîmbete „pe sub mustață”. Unele averse intermitente vor duce la creșterea nivelului lichidelor, ceea ce va implica unele „încălziri” cu efecte plăcute, urmate (dar mai bine nu) de ceață locală, asociate cu fenomene izolate de vedere bidirecțională (în sensul de dublă).

Spre sfîrșitul intervalului ce-l prognozăm (din păcate mult prea scurt spre a fi pe măsura binelui și bucuriilor ce vi le dorim cu toată sinceritatea momentului) va predomina buna dispoziție locală particulară, eventual tulburată, cu totul izolat, de posibile averse locale provocate, de exemplu, de despărțiri, pe diverse fronturi... atmosferice locale.

În concluzie, timp predominant favorabil unui mediu ambiant plăcut, cu toate că în final pot să apară nuanțe nostalgice, dar care estimăm a fi grabnic trecătoare, fiind izolate.



Întreprinderea de Calculatoare Electronice

NU UITAȚI: În România ICE reprezintă cea mai puternică concentrare de experiență și mijloace materiale în domeniul tehnicii de calcul

SERVICIILE de consultanță, instalare, asistență tehnică, școlarizări, elaborări de programe aplicative — contractate direct cu ICE.

ÎNTEPRINDEREA DE CALCULATOARE ELECTRONICE pune la dispoziția celor interesați o gamă completă de echipamente de tehnică de calcul pentru configurarea de aplicații personalizate:

- minicalculatoare 16/32 biți, compatibile 100% cu modele PDP 11 și VAX ale firmei DEC;
- microcalculatoare compatibile IBM PC/XT și AT;
- echipamente periferice: discuri magnetice de până la 750 MBy, monitoare color grafice de înaltă rezoluție, rețele de calculatoare, imprimante matriceale și cu laser.

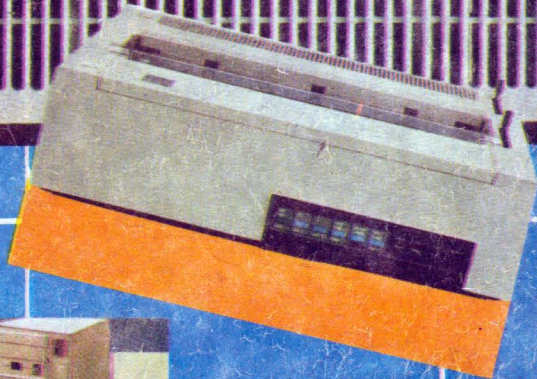
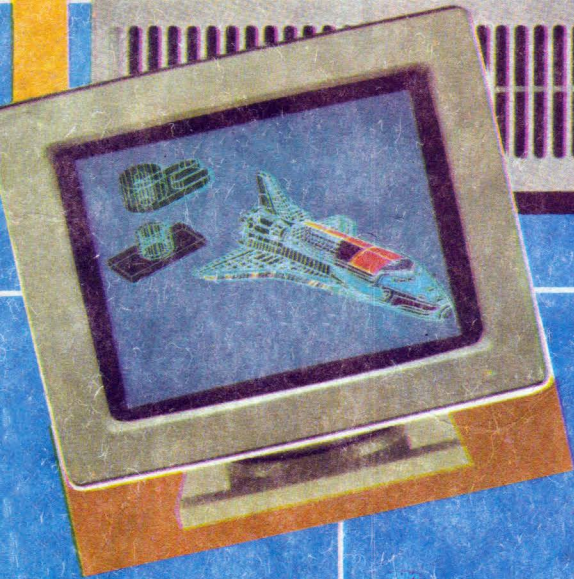
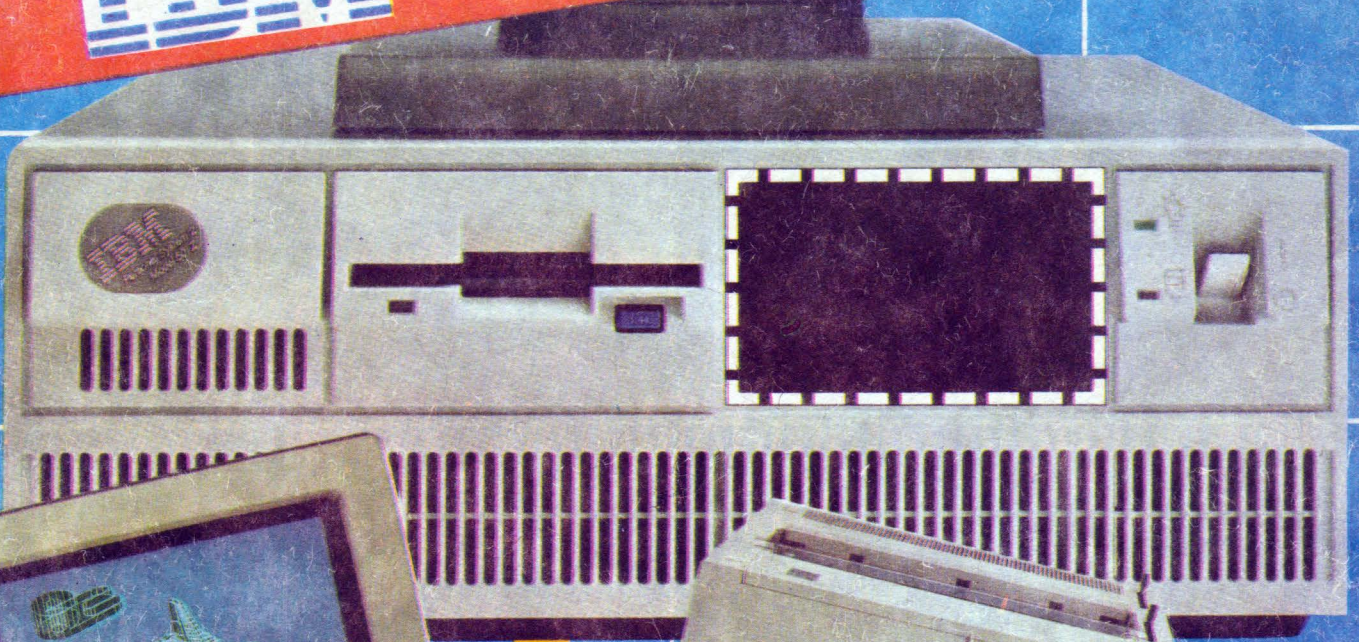
Puterea unei **REȚELE DE CALCULATOARE** reprezintă siguranța unei utilizări eficiente. În plus rețelele eterogene oferite de noi garantează dezvoltarea viitoare fără modificări în dotarea existentă.



ÎNTEPRINDEREA DE CALCULATOARE ELECTRONICE BUCUREȘTI
Str. Ing. G. Constantinescu nr. 2, sectorul 2, telefon: 88.22.95, telex
11 626 felix r, telefax: 88.78.20 felix r

IBM comercializează produse în România prin intermediul reprezentanței sale — IBM, telefon: 14.39.29 —, a agenților comerciali, a marketerilor autorizați de RBM și de reprezentanța sa din țara respectivă

Oferă soluții pentru orice aplicație!
Oferă soluții pentru orice necesitate!



AS/400

AS/400
ES 9370

PS/2

IBM



IBM oferă o gamă largă de produse și servicii în domeniul calculatoarelor electronice din familiile PS/1, PS/2, AS/400, ES 9 370, 9 000, RISC System 6 000 etc:

- consulting și engineering
- service și instalări de echipamente
- service software
- școlarizări pentru produsele achiziționate
- dezvoltări de programe

PS/1, PS/2, AS/400, ES 9 370, Risc System 6 000, Seria 9 000 a.s.o. are registered trademark of IBM Corporation

