



MIKROSZÁMITÓGÉP  
MAGAZIN

A NEUMANN JÁNOS  
SZÁMITÓGÉP-  
TUDOMÁNYI  
TÁRSASÁG  
LAPJA

Ár: 30 Ft

1986  
augusztus



**Újdonság!**

## KODEX 2000 szövegszerkesztő rendszer

A KODEX 2000 szövegszerkesztő rendszerrel igen könnyen, ugyanakkor hatékonyan végrehajtható szövegek összeállítását és nyomtatását. A program lehetővé teszi, hogy a felhasználó a szöveget videoterminalon, ROBOTRON S 6011 típusú írógépen gépelje le, menüközpontban változtassa és javítsa, kiadja a végső formát, majd tetszőleges példányszámban kinyomtassa.

A KODEX 2000 kurzor-orientált rendszer, ami azt jelenti, hogy a felhasználó a szöveg tetszőleges részén dolgozhat, miután a felülel és lefelé, valamint a balra és jobbra mozgatható kurzort a kívánt pozícióra állította. A KODEX 2000 szövegszerkesztő dinamikus képernyőformattalás jellemzi, vagyis a képernyő mindig a legutóbbi aktuális állapotot mutatja, a szöveg a nyomtatandó példánnyal azonos alakban jelenik meg. A szövegszerkesztés kezdetekor néhány egyszerű paranccsal megadható a sorhosszúság, a bal margó stb., valamint a fejléc és lap aljának formája. Ezt követően a lap formattalása automatikusan történik. Az aktuális formátumértékek bármikor megjelölhetők.

A sorhosszon túlyuló szavak automatikusan a következő sorba kerülnek. A szavakba beiktathatók láthatatlan kötőjelek is, melyek csak akkor kerülnek kinyomtatásra, ha a szó elválasztott formában sor végére kerül.

A KODEX 2000 képernyőjén megjelenő szöveg mozgatható felülel és lefelé, illetve balra és jobbra. A szöveg vagy egyes különálló sorok középre helyezhetők, vagy balra, ill. jobbra átköztesztetők, a sorok tetszőlegesen rihkthatók. A szöveg bármely része átmozgatható, mozgatható vagy törölhető. A berendezés kezeléséhez számítástechnikai ismeretek nem szükségesek. A KODEX 2000 hatékonyan alkalmazható mindenütt, ahol a fenti lehetőségek előnyű biztosítanak a hagyományos gépirás megoldással szemben: pl. szerződés, címlisták, ajánlatok stb. készítése, tárolása, időnkénti karbantartása, automatikus kiírása, formalevelek készítése, egyszerűbb ügyviteli feladatok elvégzése.

Színvonalas hardver és szoftver ellátás, garancia, folyamatos kéllékellátás.

**Gyártó:**

**Budapest, XX. ker. H—1201**  
Helsinki út 53.  
Telefon: 279-200  
Telex: 22-4399

**KONTAKTA**

### Egy írógép kítúnó memóriával



**Újdonság!**

**Műszaki adatok:**

**Írószerkezet:** A Robotron 6011 típusú beütáras elektronikus írógép szabvány magyar írógép billentyűzettel carbon- és textilszalag használatának lehetőségével. Esztétikus, kifogástalan írásépel.

**Megjelenítő egység (képernyő):**

- 25 sor 85 karakter soronként
- teljes magyar karakterkészlet
- folyamatos ernyőirtalom mozgatás (Soft scroll)
- reflexiómentes képső
- képtől: 12"

**Háttér-memória:**

- 1 db 5 1/4" 2 oldalas, dupla sűrűségű floppy disc egység (a közös házban max. 2 egység helyezhető el)
- 1 Mbyte kapacitás lemezenként

**Vezérlő elektronika:**

- M 6809 típusú CPU
- 64 kbyte RAM, 32 kbyte EPROM memória (bővíthető max. 128 k RAM-ra)
- 2 soros, 1 paruzamos interárce (opcionális gyorsnyomtató, idegényelvű billentyűzet csatlakoztatása és lokális hálózat kialakításra)
- floppy csatló max. 4 lemezegység k szolgálatára
- alfanumerikus display vezérlés

**Méret:** 480 x 440 x 450mm

**Súly:** 20 kg

**Hálózat iészültség:** 220 V, 50 Hz

**Teljesítményigény:** 120 W

**Védettsé:** kéttős szigetelésű

**Fejlesztette:** BME Folyamat szabályozási Tanszék

Budapest IX. ker. H—1093

Dimitrov tér 14.

Tel.: 175-081

Telex: 22-4736

**Forgalmazza:**

**MIGÉRT**

író- és számológép osztály

**Szerviz:**

**ITV**

Budapest V. ker. H—1052

Bécsi u. 8—10.

Tel.: 184-899

Telex: 22-4381



## A NEUMANN JÁNOS SZÁMÍTÓGÉP-TUDOMÁNYI TÁRSASÁG LAPJA

A kiadvány a Tudományos- és Informatikai Intézettel együttműködve készül

A szerkesztőbizottság vezetője:  
Kovács Győző

E számunkat szerkesztették:  
Bakos Tamás (programozástechnika)

Broczkó Péter (hírek)

Kovács Győző (levelezés)

Lindner László (sakkprogramozás)

Petróczy Judit (könyvek)

Simonyi Endre (klub)

Vadkerti János (µprogramok)

Varga András (iskola — számítógép)

A szerkesztőség munkatársa:  
Kardos Zsuzsa  
PR menedzser  
Pálhalmi Vali

Felelős szerkesztő:  
Könyves Tóth Pál  
Szerkesztőség:  
1027 Budapest II., Fő u. 68.  
Telefon: 154-250

Kiadja a Delta Szaklapkiadó és Műszaki Szolgáltató Leányvállalat

Felelős kiadó:  
dr. Varga György igazgató  
1442 Budapest VII., Garay u. 5.  
Telefon: 415-583, 215-440

Terjeszti a Magyar Posta Előfizethető a hírlapkézbesítő hivataloknál és a Posta Hírlapelőfizetési és Lapellátási Irodáján (1900 Budapest V., József nádor tér 1.) vagy átutalással a 215-96162 pénzforgalmi jelzőszámra.

Megjelenik havonta  
Egy szám ára 30,— Ft  
Előfizetési díj:  
egy évre 360,— Ft  
fél évre 180,— Ft  
Külföldön terjeszti a Kultúra,  
1389 Budapest, pf. 149.  
és a Magyar Média  
1932 Budapest, pf. 279.  
86—253



Szikra Lapnyomda  
Budapest (86-3475)

Felelős vezető:  
Csóndes Zoltán vezérigazgató

INDEX: 25629  
ISSN 0236-6088

Címképünk:

A Híradástechnika Szövetkezet  
PTA-4000-es számítógépe



### Tartalom

A nyár és az informatika	2
Assemblerek, cross-assemblerek	11
Hogyan működnek a lapos kijelzők?	14
Adok — veszek — cserélek	15
A minőség közügy	16
Commodore 64 segédtablázat	23
Olvastunk ...	36

### ISKOLA — SZÁMÍTÓGÉP

Grafikus hardcopy C16-ra	3
Lencsék képkalkotása	4
Gépi kódú grafikát segítő program	5
Az ismeretlen C16	6
Mi van a bolgár iskolákban?	7

### DIÁKROVAT

A C64 grafikus lehetőségei	8
Kódkonverter	10

### PROGRAMOZÁSTECHNIKA

BASIC és gépi kód	17
Z80 programozási gyakorlatok	18

### µPROGRAMOK

Disassembler program Sinclair számítógépekre	19
RESTORE n	26
Képernyőn a Primo agya!	27
Megszakításvezérelt BREAK-figyelés	28

### µKLUB

PTA—4000	32
PTA—4000. Fedélzeti számítógép az autóban	33

### JÁTÉKPROGRAMOK

	40
--	----

### KÖNYVEK

	41
--	----

### AZ OLVASÓ ÍRJA

	43
--	----

### FÓRUM

	44
--	----

### SAKKPROGRAMOZÁS

Bitek és figurák	46
------------------	----

HÍREK, ÉRDEKESÉGEK	48
--------------------	----

# A nyár és az informatika

„(Apám!) A mesterem nem [osztja meg] velem tudását, erős karja [elzár] engem az írni-írástól, a tudás háttéréről, hogy his írnek maradjak: nem [engedi], hogy a tábla házában felsős testvér legyek.”

„(A tábla házában fia) — Sumér eredetű fordította Komoróczy Géza 1970, 1983.)

Talán felesleges is mondanom, hogy a lapkészítés kiszámíthatatlan időbeosztása miatt most május elején éppen júliusban kell gondolkodnom, pedig még csak tegnap kezdődött el a május, még javában virágznak a fák, a legbátrabbak talán már belekóstoltak a balatoni fürdés örömeibe is. Azért inkább tavasz van még, mint nyár, de a nyár sincs már messze, hiszen egymás után kapom a leveleket, hogy Salgótarjánban, Baján, Kecskeméten, Békéscsabán és ki tudja még merre az országban, szervezik a nyári informatikai táborokat. Megszállott amatőrök, tanárok és ifjúsági vezetők foglalkoznak a tanulni vágyó fiatalokkal, tartanak alapfokú és továbbképző tanfolyamokat, megteremtve az informatikát szerető gyerekeknek azt az ideális környezetet, amelyben reggeltől estig mással sem foglalkozhatnak, mint a számítógéppel.

Emlékszem az első táborokra vagy 5-6 évvel ezelőtt, amikor különböző manipulációkkal sikerült a táborokba 1-2 gépet szerezni. Ezeket legtöbbször egy gondosan őrzött szobában helyezték el, ahol sokszor zöld felügyelők éber tekintete előtt lehetett játszani vagy programot írni, persze csak azoknak, akik a gépekhez értek.

A fejlődés szerencsére nem állt meg, egyre több ifjúsági táborban jelent meg a számítógép — úgy hallottam, hogy az elmúlt évben már az volt a feltűnő, ha valahol nem lehetett számítógépen dolgozni. Egy- két tanárral beszélgettem, akik elmondták, hogy a legtöbb problémát az jelentette, hogy az ifjúság ne csak a számítógépet mellett üljön, de focizzon is, ússzon is, kiránduljon is, egyszerűen legyen mértékletes a számítógépidő fogyasztásában.

A táborokról szóló hírek az emberben azt a megnyugtató érzést keltik, hogy a diákok nyári informatikai lehetőségei elegendiek az igényeket, ami persze nem azt jelenti, hogy ezután már egyáltalán nem lesznek panaszok, nem lesz egyetlen rosszul szervezett tábor sem, ennyire azért nem lehetünk optimisták.

Ha valamiért szót emelek, annak az az oka, hogy a felnőttek részére — úgy látszik — hasonló lehetőségekről nem gondoskodunk. Pedig az informatika iránti érdeklődés, azt hiszem, egyáltalán nem csökkent, sőt talán még fokozódik is. Ha jól látom a helyzetet, akkor egyre többen látják be, hogy nem elvesztegetett idő, amit a számítástechnika tanulására fordítanak, hiszen egyre több munkahelyen ér többet az, aki a számítógépeket is használni tudja.

Az elmúlt hetekben több, egyetemet és főiskolát, de középiskolát is most végző ifjú ember keresett meg azzal a meglepő hírrel, hogy tanulmányait befejezte el sze-

retnének helyezkedni, de egyre több helyen kérdezik meg tőlük, hogy ismerik-e a számítógépeket, tudnak-e programozni, van-e számítógépes gyakorlatuk. Őszinte örömmre szolgált, hogy — a néhány tíz megkeresésből következtetve — már vannak olyan munkahelyek, ahol az informatikai ismeret a munkát most kezdő szakemberjelölteknél — alapkövetelmény. Horribile dictu! még azt is hallottam, hogy az egyik középállalatnál a felvétel alkalmából a TV BASIC bizonyítvány határozott előnynek számított, ennek a birtokában ugyanis lelkes telefonálón többeket megelőzve kapta meg a nem is számítástechnikai állást.

Hozzáem forduló barátaim leginkább segítségét kérik, tanulni szeretnének — olcsón és gyorsan —, pótolni az egyetemeken elmulasztottakat.

A SZÁMALK tanfolyamát szoktam ajánlani, ami az egyéni tanulóknak meglehetősen drága. Úgy hallottam, ezek a tanfolyamok főleg a munkahelyek küldik azokat a szakembereket, akiknek az informatikai képzését szükségesnek tartják. A másik probléma, hogy a SZÁMALK-tanfolyamok elsősorban azoknak a szakembereknek szólnak, akik a tanfolyam(ok) elvégzése után főfoglalkozású számítástechnikusként szeretnének a számítástechnika valamelyik részterületével foglalkozni. Ugyanakkor az orvos, a közgazdász, a mérnök vagy bárki más, aki saját szakterületén kíván maradni, de az informatikát is meg akarja ismerni, hogy alkalmazni tudja, nehéz helyzetben van. Ha szerencsésük van, akkor be tudnak iratkozni egy szakmai továbbképző tanfolyamra, ha ott van informatika, vagy maradnak számukra a TIT-tanfolyamok, az NJSZT előadások, esetleg a GM-ek, a PJT-k, vagy ha gépet is vesznek, akkor a gyártó intézmények tanfolyamai.

A következő öt éves tervben a kormány külön forrásokat határozott meg az informatika oktatására, ezen belül az ún. lakossági oktatásra is; azt sajnos még nem tudjuk, hogy mennyit, és azt sem, hogy a forrást mikor lehet igénybevenni. A lakossági oktatásért — jobbnak tűnik, ha informatikai tömegoktatásnak nevezzük — az NJSZT a felelős, de az már biztos, hogy a feladatot egyedül megoldani egyetlen szervezet, így az NJSZT sem tudja. Ezért hoztunk létre széles körű együttműködést mindazokkal a szervezetekkel, amelyeknek nemcsak érdeke, hogy minél több, az informatikához értő ember legyen az országban, de nem keveset tettek is az informatika terjesztése és oktatása érdekében.

Az NJSZT legfontosabb szövetségesei a KISZ, az MTESZ tudományos egyesületek, a TIT, a Hazafias Népfőnt, a Közművelődési Tanács, hogy csak a legismertebbeket említsem. Hiba lenne az informatikai tömegoktatást önállóan, a közoktatástól és a tanfolyami oktatástól elválasztva kezelni, ezért az oktatási rendszer kialakítása során szorosan együtt kell működnünk a Művelő-

dési Minisztérium intézményeivel, a Tudományos Szervezési és Informatikai Intézet, az Országos és a Fővárosi Pedagógiai Intézet, az Országos Oktatástechnikai Központtal, illetve a tanfolyami oktatásban a Központi Statisztikai Hivatallal és a SZÁMALK-kal, valamint az Ipari Minisztériummal és oktatási szervezeteivel, de sorolhatnánk más főhatóságokat és a hozzájuk tartozó oktatási intézményeket is. Nagyon lényegesnek tartom az OMFB-vel, mint az elektronizációs programért felelős főhatósággal a kapcsolatot, hiszen az informatikai tömegoktatás csak akkor láthatja el jól a feladatát, ha része lesz az elektronizációs programnak.

Látszólag messze kerültem a nyártól és a nyári informatikai táboroktól, valójában nem. En azt hiszem, hogy a felnőttek részére szervezett informatikai tömegoktatás hatékonyságát is növelni lehetne oly módon, ahogyan ma az ifjúsági informatikai táborokat szervezzük. Sőt — meg sem kellene várni, amíg a rendszeres tömegoktatás beindul, a nyaralás már most is fel lehetne használni alapfokú informatikai ismeretek megszerzésére. Egészen biztos vagyok abban, hogy sokan szívesen aldoznák szabadidejüket, nyaralásuk egy részét arra, hogy a nyaralásuk alatt az informatika alkalmazásával megismerkedjenek. Arra gondolok, hogy az üdülési szezonban — általában a szervezett üdülés 2-2 hétig tart — az üdülők egy részében kétéhes informatikai tanfolyamot lehetne szervezni, amelyhez az előadást a televízióban és a rádióban, a számítógépeket pedig a vállalati, illetve a SZOT-üdülőkben lehetne biztosítani. Ma, amikor egy számítógép kb. tízezer forintba kerül, hozzá a legegyszerűbb tv legfeljebb négyezer, azt hiszem, hogy a vállalatoknak megérné, hogy üdülő munkatársaiknak a fele vagy a negyede úgy jöjjön vissza a nyári szabadságról, hogy tud már valamit a számítógépről és a számítástechnikáról. Nagyon sokszor ér az a vád, hogy megvalósíthatatlan ötleteim vannak, már hallom is, naivitás azt hinnem, hogy majd a számítógép lép az ultipartí vagy a rómizés helyébe. Én ezt nem hiszem, de azt igen, hogy a sokaság szórakozás mellett üdítő változatossá lehet az informatika, ez az intellektuális játék, ami nemcsak hasznos ismereteket ad, de unaloműzőnek sem akármilyen szórakozás!

Amikor a  $\mu\text{M}$ -nak ezt a számát olvassák, már bőven tart az üdülési szezon, egy ilyen televíziós-rádiós nyári továbbképző tanfolyam megszerzésére most már nagyon kevés az idő. De arra talán lenne lehetőség, hogy a nem túlságosan kihaznárt személyi számítógépeiket a vállalatok nyárára az üdülőbe telepítsék, és néhány szűnődiós diáknak azért fizessenek, hogy a tanulni vágyó dolgozóknak elmagyarázzák, mit is lehet a számítógépekkel csinálni.

Nyár van. Ne felejtjük el, az informatika korában a nyár a pihenve tanulás ideje!

KOVÁCS GYŐZŐ

## Grafikus hardcopy C16-ra

A C16 népszerűségét többek között jó grafikaijának is köszönheti, ami BASIC-ből is könnyen elérhető.

Programom egy viszonylag rövid (1 page) gépi rutin, amellyel a grafikus képernyő tartalma menthető ki nyomtatóra. Keletkezésének története szorosan összefügg az Úry-féle C16-könyv 236. oldalával. Itt található ugyanis az a BASIC program, amelynek ugyanez lenne a célja, de sajnos ez nem működik. Lehet, hogy nyomdai hiba, bár nem valószínű. Így hát készítettem egy másik programot, de már teljes gépi kódban, szubrutinként hívhatóan.

A &HCOPY monitorból olvasható be, egyszerűen az L paranccsal. Belépési címe &1700, BASIC-ből, a SYS DEC ("1700") utasítással hívható.

Ha jobban szemügyre vesszük a programot, látható, hogy nem is igazán &1700-nál, hanem &179F-nél kezdődik. Itt kezdődik a Kernal-rutinok felhasználásával a nyomtató megnyitása (kb. OPEN 4.4). Ezután következik a program lelke, illetve annak meghívása a JSR &1710-zel.

A program megírásában az első nehézséget az jelentette, hogy a nyomtatót a grafikus adatokat csak függőleges formájában tudja feldolgozni. Ezzel szemben a képernyőpontok a memóriában 8-as „csomagokban” tárolódnak. Egy 8×8-as karakternyi bitrácstot forgat át a &171F-től kezdődő pár sor. Ez a &1708-tól kezdve 8 bájton elhelyezi a karakter felső hét sorát, de úgy, hogy ez már nyomtatásra alkalmas legyen, vagyis egy bájtra helyezve az azonos bite-

ket, és mindenütt b7=1. Az így elkészített adatsort a &160F-en levő rutin küldi el a nyomtatónak.

Ez előbb pár sorról beszéltem, mégpedig azért, mert a &171F rutini a nyomtató egy másik furcsaságát is leküzdö, hogy csak hét pontsört tudok egyszerre kiírni. Itt lép be &DE eltolásjelző, ami megmutatja, hogy mennyi maradt meg az előző sorban vett nyolcas karaktercsomagból. Ugyancsak itt használok a &DC-&DD, valamint a &171D-&171E-n levő mutatókat.

Ennek a „hét sorosságának” egy másik következménye, hogy a képernyő hasznos területéből az utolsó kiírandó adatsorba csak 4 bitsor maradt. Ezért kellett a &160F rutinba egy AND-et elhelyezni.

A program a kiírás után lezárja a nyomtatót (&17F1) és RET-tel tér vissza.

TAASSONYI KADOCSEA

16F0	A2 08	LDX #008	173E	8D 1D 17	STA \$171D	1796	20 D2 FF	JSR \$FFD2
16F2	BD 07 17	LDA \$1707,X	1741	AD 1E 17	LDA \$171E	1799	CE 07 17	DEC \$1707
16F5	2D 06 17	AND \$1705	1744	69 01	ADC #001	179C	A9 00	LDA #000
16F8	20 D2 FF	JSR \$FFD2	1746	8D 1E 17	STA \$171E	179E	60	RTS
16FB	CA	DEX	1749	18	CLC	179F	A2 04	LDX #004
16FC	D0 F4	BNE \$16F2	174A	90 D0	BCC \$171C	17A1	A0 00	LDY #000
16FE	60	RTS	174C	A2 08	LDX #008	17A3	A9 04	LDA #004
16FF	00	BRK	174E	A9 FF	LDA #0FF	17A5	20 BA FF	JSR \$FFBA
1700	4C 9F 17	JMP \$179F	1750	2A	ROL	17A8	20 C0 FF	JSR \$FFC0
1703	4C 00 00	JMP \$0000	1751	7E 07 17	ROR \$1707,X	17AB	A2 04	LDX #004
1706	8F	???	1754	CA	DEX	17AD	20 C9 FF	JSR \$FFC9
1707	FF	???	1755	D0 F9	BNE \$1750	17B0	A9 0D	LDA #00D
1708	BF	???	1757	20 F0 16	JSR \$16F0	17B2	20 D2 FF	JSR \$FFD2
1709	A8	TAY	175A	EA	NOP	17B5	A9 08	LDA #008
170A	C8	INY	175B	C6 DF	DEC \$DF	17B7	20 D2 FF	JSR \$FFD2
170B	98	TYA	175D	A5 DF	LDA \$DF	17BA	A9 00	LDA #000
170C	88	DEY	175F	D0 16	BNE \$1777	17BC	85 DC	STA \$DC
170D	F8	SED	1761	A9 0D	LDA #00D	17BE	85 DE	STA \$DE
170E	98	TYA	1763	20 D2 FF	JSR \$FFD2	17C0	A9 20	LDA #020
170F	88	DEY	1766	A5 DE	LDA \$DE	17C2	85 DD	STA \$DD
1710	A4 DE	LDY \$DE	1768	D0 F5	BNE \$175F	17C4	A9 28	LDA #028
1712	A5 DC	LDA \$DC	176A	A5 DC	LDA \$DC	17C6	85 DF	STA \$DF
1714	8D 1D 17	STA \$171D	176C	38	SEC	17C8	A9 1C	LDA #01C
1717	A5 DD	LDA \$DD	176D	E9 40	SBC #040	17CA	8D 07 17	STA \$1707
1719	8D 1E 17	STA \$171E	176F	85 DC	STA \$DC	17CD	A9 FF	LDA #0FF
171C	B9 78 40	LDA \$078,Y	1771	A5 DD	LDA \$DD	17CF	8D 06 17	STA \$1706
171F	A2 08	LDX #008	1773	E9 01	SBC #001	17D2	20 10 17	JSR \$1710
1721	2A	ROL	1775	85 DD	STA \$DD	17D5	AD 07 17	LDA \$1707
1722	7E 07 17	ROR \$1707,X	1777	A5 DC	LDA \$DC	17D8	D0 F3	BNE \$17D3
1725	CA	DEX	1779	18	CLC	17DA	A9 8F	LDA #08F
1726	D0 F9	BNE \$1721	177A	69 08	ADC #008	17DC	8D 06 17	STA \$1706
1728	C8	INY	177C	85 DC	STA \$DC	17DF	20 10 17	JSR \$1710
1729	C8	INY	177E	A5 DD	LDA \$DD	17E2	AD 07 17	LDA \$1707
172A	98	TYA	1780	69 00	ADC #000	17E5	10 F8	BPL \$17DF
172B	88	DEY	1782	85 DC	STA \$DC	17E7	A9 0F	LDA #00F
172C	29 07	AND #007	1784	A5 DF	LDA \$DF	17E9	20 D2 FF	JSR \$FFD2
172E	C5 DE	CMP \$DE	1786	D0 16	BNE \$179E	17EC	A9 0D	LDA #00D
1730	F0 1A	BEQ \$174C	1788	A9 28	LDA #028	17EE	20 D2 FF	JSR \$FFD2
1732	C0 08	CPY #008	178A	85 DF	STA \$DF	17F1	A9 04	LDA #004
1734	D0 E6	BNE \$171C	178C	C6 DE	DEC \$DE	17F3	20 C3 FF	JSR \$FFC3
1736	A0 00	LDY #000	178E	A9 07	LDA #007	17F6	20 CC FF	JSR \$FFCC
1738	AD 1D 17	LDA \$171D	1790	25 DE	AND \$DE	17F9	60	RTS
173B	18	CLC	1792	85 DE	STA \$DE	17FA	31 37	AND (&37),Y
173D	69 40	ADC #040	1794	A9 08	LDA #008	17FC	30 30	BMI \$182E
						17FE	04	???

A program C16-os gépre készült; felhasználható az általános iskolák 8. osztályos fizikaóráin a fénytán tanításánál. Domború és homorú lencsék képalkotását mutatja be. A fókusz távolság és a képtávolság megadása után a képernyőn megjelenik a kép szerkesztéséhez szükséges sugármenet, a tárgy távolság és a nagyítás számértéke.

A program a TIH által forgalmazott, ékezetes betűket tartalmazó gépre készült, ezért a listában a kiírandó szövegek kissé „furcsák”; a nyomtató ugyanis az eredetileg definiált karaktereket írja ki. A program begépelésekor ezeket a sorokat a következőképpen írjuk:

```
2 PRINTTAB (11) "LENCSEK KÉPAL-
KOTÁSA"
4 SCNCLR:PRINT "Figyelj a fókusz távol-
ság megadásánál!"
5 PRINT:PRINT "A homorú lencse fók-
usz távolsága [ ] negatív [ ]!"
7 SCNCLR:INPUT "A lencse fókusz távol-
sága (cm)"; t
20 PRINT " [ ] A képtávolság" ...
21 PRINT " [ ] A nagyítás" ...
22 PRINT "Új adatok → [ ] U [ ]
Befejezés → [ ] B [ ]";
35 ... PRINT " [ ] Nincs képalkotás" ...
```

A program begépelésénél ügyeljünk arra, hogy szöközt ne nagyon írjunk, mert akkor esetleg a program nem fér be a finom grafika miatt szűkös memóriaterületre.

A program ötletet adhat általános iskolai tanároknak tantárgyi programok írására.

DR. PORONYI GÁBOR

## Lencsék képalkotása

```
1 GRAPHIC2:PRINTCHR$(14);COLOR$(2);COLOR$(1)
2 PRINTTAB(11) "LENCSEK KÉPALKOTÁSA"
3 FOR I=1 TO 3000 NEXT
4 SCNCLR:PRINT "LEVELT A FOKUSZ TAVOLSG MEGADÁSÁNA!"
5 PRINT "PRINT" HOMORU LENCSE FOKUSZ TAVOLSG (NEGATIV)"
6 FOR I=1 TO 3000 NEXT
7 SCNCLR:INPUT "LENCSE FOKUSZ TAVOLSG (CM)";G:INPUT "TARGY TAVOLSG (CM)";T
8 IF T<0 OR G<0 THEN "ELSE" #35 GOTO 35
9 IF T<0 AND G<0 THEN "ELSE" #35 GOTO 35
10 IF T<0 THEN "ELSE" #35 GOTO 35
11 IF T<0 THEN "ELSE" #35 GOTO 35
12 IF T<0 THEN "ELSE" #35 GOTO 35
13 IF T<0 THEN "ELSE" #35 GOTO 35
14 IF T<0 THEN "ELSE" #35 GOTO 35
15 IF T<0 THEN "ELSE" #35 GOTO 35
16 IF T<0 THEN "ELSE" #35 GOTO 35
17 IF T<0 THEN "ELSE" #35 GOTO 35
18 IF T<0 THEN "ELSE" #35 GOTO 35
19 K=INT(G/T)
20 PRINT "K=INT TAVOLSG (CM)";PRINTUSING"#####.###";K;PRINT "CM"
21 PRINT "MAGNIFIKACIO";PRINTUSING"#####.###";ABS(K/T)
22 PRINT "TARGY TAVOLSG (CM)";PRINTUSING"#####.###";T
23 GETKEYVAR:IF #1 THEN "ELSE" #35 GOTO 35
24 CIRCLE(1,95,30,62),50,130 CIRCLE(1,225,30,62),275,310
25 DRAW(1,142,40 TO 178,40) DRAW(1,142,128 TO 178,128) RETURN
27 DRAW(1,158,-P,50 TO 158,-P,50 TO 162,-P,50 TO 162,-P,50) PRINT "1,61;-P,77"
28 DRAW(1,158,-P,50 TO 160,50) RETURN
29 DRAW(1,160,50 TO 245,101) DRAW(1,10,50 TO 245,97)
30 DRAW(1,234,80 TO 234,95 TO 236,95 TO 236,80) PRINT "1,235,91" RETURN
31 DRAW(1,160,50 TO 279,116) DRAW(1,60,50 TO 279,113)
32 DRAW(1,230,80 TO 230,110) DRAW(1,110,50 TO 230,110) PRINT "1,259,81" RETURN
33 DRAW(1,150,50 TO 320,148) DRAW(1,85,50 TO 320,145)
34 DRAW(1,289,80 TO 389,142 TO 316,142 TO 316,80) PRINT "1,310,81" RETURN
35 DRAW(1,160,50 TO 319,140) DRAW(1,110,50 TO 319,170) PRINT "2,110,81" RETURN
36 DRAW(1,160,50 TO 220,86) DRAW(1,130,50 TO 220,140)
37 L=50 FOR K=1 TO 100 STEP 2 DRAW(1,K,L,L-1,5) NEXT
38 L=50 FOR K=160 TO 800 STEP 3 DRAW(1,K,L,L-1,5) NEXT DRAW(1,97,50 TO 97,217 TO 104,217 TO 104,217)
39 RETURN
40 GOSUB 48 DRAW(1,18,50 TO 210,90)
41 DRAW(1,122,80 TO 122,72) DRAW(1,125,80 TO 125,72) RETURN
42 GOSUB 48 DRAW(1,60,50 TO 210,90)
43 DRAW(1,128,80 TO 128,70) DRAW(1,125,80 TO 125,70) RETURN
44 GOSUB 48 DRAW(1,85,50 TO 219,109) DRAW(1,129,80 TO 129,67) DRAW(1,131,80 TO 131,67) RETURN
45 RETURN
46 GOSUB 48 DRAW(1,110,50 TO 219,110) DRAW(1,134,80 TO 134,65 TO 137,65 TO 137,80) RETURN
47 GOSUB 48 DRAW(1,130,50 TO 210,130) DRAW(1,140,80 TO 140,62 TO 143,62 TO 143,80) RETURN
48 DRAW(1,160,50 TO 210,20) L=50 FOR K=160 TO 110 STEP 3 DRAW(1,K,L,L-1,5) NEXT RETURN
```

### Kiegészítés

a „Pályázat közoktatási számítógépes programok elkészítésére” c. felhíváshoz  
(Mikroszámítógép Magazin, 1986. április)

A Tudományos- és Informatikai Intézet értesíti az érdeklődőket, hogy a Közoktatási Programpályázaton az új iskolaszámítógépre írt programokkal is részt lehet venni. Jelenleg a következő gépeken futó programokkal lehet pályázni.

1. Általános iskolai kategóriában: **COMMODORE—16, PRIMO, PRO/PRIMO, TV-COMPUTER**
2. Középszintű iskolai kategóriában: **COMMODORE—64, HT (valamennyi típus), PRIMO, PRO/PRIMO, TV-COMPUTER**

A pályázat részletes feltételeinek leírását — kérésre — postán megküldjük vagy személyesen átadjuk.

Címünk: Tudományos- és Informatikai Intézet,

Budapest XI., Egy József utca 1—9. — BME „E” épület — XI. em. 5.

PÁRIS GYÖRGY sk.  
A Tudományos- és Informatikai  
Intézet igazgatója

# HT-1080Z Gépi kódú grafikat segítő program

A BASIC-ben készíthető grafika minden számítógépnél, így a HT-nál is lassú. Ez a lassúság elsősorban olyan esetekben, amikor ugyanabban a programban egy-egy képernyőképet változtatlan vagy csak kevésbé módosított formában többször meg kell jeleníteni (játékprogramok, demonstrációs programok stb.), igen zavaró lehet, és egy magára „valamit adó” programozó számára nem is kielégítő megoldás.

A képek gépi kódban történő elkészítése a segítségi problémákat megoldja ugyan, de körülmenyes, lassú és idegőrlő munka.

A program, melyet itt bemutatok, a képernyőképeket BASIC-ből állítja elő, majd a kész képet a BASIC számára el

nem érhető, a RAMTOV feletti tárterületre helyezi át (1. program). Az áthelyezett képek a forgalomban levő számos segédprogram egyikével, például a MONIT-tal betölthető gépi kódú ("SYSTEM") programmá másolhatók ki a tárból. Célszerű a képeket tároló tármezőket és az ismertetésre kerülő segédprogramot egymás mellé illeszteni; így egyetlen, összefüggő gépi kódú programhoz jutunk.

Az így előállított programot a RAMTOV feletti mezőbe töltjük, majd BASIC betöltéssel bevisszük a BASIC programot is. Ezután a tárban levő képek a BASIC-ből hívott blokkmozgató gépi kódú rutinál igen gyorsan átvihetők a képernyőre.

## 1. program

```

10 CLS:DEFSTRE
20 B1=CHR$(191)+CHR$(128):B2=CHR$(128)+CHR$(191)
30 B4=CHR$(128)+CHR$(191)
40 GOSUB120:GOSUB260
50 GET3000:PUT7000:FIELD0:400:SAVE
60 CLS:B3=B1:B1=B2:B2=B3:B6=B4:B4=B5:B5=B6
70 GOSUB120:GOSUB260
80 PUT7400:SAVE
90 PUT7400:LOAD:FORI=0TO70:NEXT
100 PUT7000:LOAD:FORI=0TO70:NEXT
110 GOTO90
120 REM KERET
130 REM KERET FELSO
140 A=0:B=B1:BB=B5:GOSUB400
150 A=64:B=B2:BB=B4:GOSUB400
160 REM KERET FUGGOLEGES
170 FORI=128TO704STEP128
180 PRINT0I;.B1:PRINT0(I+61).B2:PRINT0(I+64).B2:
190 PRINT0(I+64).B2:PRINT0(I+125).B1:
200 NEXT
210 PRINT0I;.B1:PRINT0(I+61).B2:
220 REM KERET ALSO
230 A=B32:B=B2:BB=B4:GOSUB400
240 A=B96:B=B1:BB=B5:GOSUB400
250 RETURN
260 REM "GRAPH"
270 DATA150.169.128.151.169.128.150.169
280 DATA128.151.169.128.149.170
290 DATA149.176.128.181.154.128.181.186
300 DATA128.181.154.128.181.186
310 DATA165.186.128.151.164.128.149.170
320 DATA128.149.128.128.149.170
330 FORJ=344TO477STEP64
340 FORI=0TO13:READA:PRINT0(J+I).CHR$(A)::NEXTI
350 NEXTJ
360 RESTORE
370 PRINT0592;"GEPI KODU GRAFIKAT SEGITO PROGRAM":
380 PRINT0662;"KESZITETTE: NAGY INRE":
390 RETURN
400 REM KERET VIZSZINTES RESZE
410 FORI=0TO60STEP2:PRINT0(A+I).B:;NEXT
420 PRINTBB:
430 RETURN

```



## 2. program

```

1 :GEPI KODU GRAFIKA SEGITESE
2 :
3 :
4 :
5 :
6 417C C3807F ORG 417CH
7 417F C38B7F JP FIELD
8 4182 C3907F JP PUT
9 ORG 418BH
10 4188 C3A87F JP ALDAD
11 ORG 41A0H
12 41A0 C3987F JP SAVE
13 ORG 41E2H
14 41E2 ORG (HL) :AUTOM. INDULAS.
15 :
16 :
17 :
18 7FB0 CDBE7F FIELD: CALL ADDR :A BYTE-OK SZAMA
19 7FB3 ED53B87F LD (BYTS).DE
20 7FB7 C9 RET
21 7FB8 CDBE7F GET: CALL ADDR :A VIDEO KEZDOCIM
22 7FB8 ED53BA7F LD (SCRN).DE
23 7FBF C9 RET
24 7F90 CDBE7F PUT: CALL ADDR :A TAR KEZDOCIM
25 7F93 ED53BC7F LD (MEMADR).DE
26 7F97 C9 RET
27 7F98 E5 SAVE: PUSH HL :VIDEO ---) TAR
28 7F99 2ABA7F LD HL.(SCRN)
29 7F9C DE5B8C7F LD DE.(MEMADR)
30 7FA0 ED4BB87F LD BC.(BYTS)
31 7FA4 EDB0 LDIR
32 7FA6 E1 POP HL
33 7FA7 C9 RET
34 7FAB E5 ALOAD: PUSH HL :TAR ---) VIDEO
35 7FA9 2ABC7F LD HL.(MEMADR)
36 7FAC ED5BBA7F LD DE.(SCRN)
37 7FB0 ED4BB87F LD BC.(BYTS)
38 7FB4 EDB0 LDIR
39 7FB6 E1 POP HL
40 7FB7 C9 RET
41 :
42 :
43 7FB8 0000 :A BYTE-OK SZAMA ES A CIMEK
44 7FB8 0000 BYTS: DN OH
45 7FBC 0000 SCRNI: DN OH
46 MEMADR: DN OH
47 :
48 7FBE CDE57F :A CIMEK TOMORITESE 2 BYTE-BA
49 7FC1 CB27 ADDR: CALL CHARIN
50 7FC3 CB27 SLA A :BITEK A 4 MSB HELYE
51 7FC5 CB27 SLA A
52 7FC7 CB27 SLA A
53 7FC9 57 LD D.A
54 7FCA 23 INC HL :A SZOVEGHUT.NOVELESE
55 7FCB CDE57F CALL CHARIN
56 7FCE B2 OR D
57 7FCE 57 LD D.A :H" BYTE KESZ
58 7FD0 23 INC HL :A SZOVEGHUT.NOVELESE
59 7FD1 CDE57F CALL CHARIN
60 7FDA CB27 SLA A :BITEK A 4 MSB HELYE
61 7FDB CB27 SLA A
62 7FDD CB27 SLA A
63 7FDD CB27 SLA A
64 7FDD 5F LD E.A
65 7FDD 23 INC HL :A SZOVEGHUT.NOVELESE
66 7FDE CDE57F CALL CHARIN
67 7FE1 B3 OR E
68 7FE2 5F LD E.A :L" BYTE KESZ
69 7FE3 23 INC HL :A SZOVEGHUT.NOVELESE
70 7FE4 C9 RET
71 7FE5 AF CHARIN: XOR A (HL)
72 7FE6 84 ADD A (HL)
73 7FE7 FE40 CP #0H :BETU?
74 7FE9 DAE77F JP C.NUM :NEM
75 7FEC D437 SUB #3H :IGEN
76 7FEE EA0F AND #FM :MASZKOLAS 4 LSB-RE
77 7FF0 C9 RET
78 END

```

A program a HT—BASIC lemezes rendszerének utasításait használja, így elkerülhető az USR alkalmazásával járó kényelmetlenségek.

A felhasznált utasítások és formátumuk:

PUTnnnn a tármező kezdő-címének megadása

GETnnnn a VIDEO-RAM mező kezdőcímének megadása

FIELDnnnn a mozgatni kívánt bájtok száma

SAVE átvitel a képernyőről a tárbá

LOAD átvitel a tárból a képernyőre

A GET, a PUT és a FIELD utasításokban a címek (nnnn) hexadecimálisak. A bemutatott formátum kötelező, az utasításokban felesleges karakterek (például betűk) nem lehetnek, és a címeknek mindig 4 karakterből kell állniuk.

A program 16 kb-ot gépre készült. 64 kb-ot gép alkalmazásakor a FOPROGRAM megjegyzést követő ORG 7F80H helyett ORG 0FF80H-t kell írni, így a program ebben az esetben is a tár végére kerül.

A programot az EDI assembler-editorral kell elkészíteni. A hibátlan forrásprogramot az A parancsok és OPTION C-vel mindjárt kezettára fordítjuk, az EXEC.ADDR kérdésre 06CCH-t adva meg. A programnév tetszőleges lehet. A gépi kódú program betöltés után automatikusan elindul, a vezérlést a BASIC-nek visszaadva. Mivel a 41E2H című rekeszben a JP (HL) utasítás kódja van, a SYSTEM parancs használatakor előre ki nem számított problémák adódhatnak. Célszerű ezért ennek a rekesznek a tartalmát a program kezdetén eredeti értékére visszaállítani egy POKE 16866,195 utasítással.

Ha a program alkalmazása során (például csak egy képet kívánunk mozgatni) a GET, PUT és FIELD utasításokkal nem változtatjuk meg a tárműveletet és a bájtámszámot, ezek az utasítások a LOAD alkalmazása előtt feleslegesek.

A program alkalmazásának szemléltetésére bemutatok egy példát. A 2. program BASIC-ben előállítja az ábrán látható képet, ezt tárolja, előállítja a „keret” inverzét, ezt is tárolja, majd a két képet váltakozva jeleníti meg a képernyőn.

NAGY IMRE

1985 decemberében az ÁPISZ jóvoltából hozzájutottam egy C16-os mikroszámítógéphez. Mivel a hozzáférhető irodalomból viszonylag kevés információt kaptam, magam láttam a gép lelkivilágának feltárásához. Bár ez a munka még nem fejeződött be, szeretném tapasztalataimat megosztani az ez iránt érdeklődőkkel. E cikksorozat keretében először a gép rendszerváltozóiról és beépített rutinjairól írok.

Az egyes változók, eljárások leírásánál igyekszem egységesen eljárni. A megnevezéssel kezdem, amely általában egy angol nyelvű azonosító és megegyezik a C64-leírásokban található azonosítókkal. Ezután az azonosítóhoz tartozó cím következik, előbb hexadecimálisan, majd zárójelben decimálisan. Ez a cím változók esetén a változó kezdő- és végcímé, eljárások esetén pedig az eljárás belépési pontja.

Ezután rövid leírás következik a használatról, működésről. Végül, kapcsolódva a „BASIC és gépi kód” c. cikksorozathoz, a változó vagy eljárás címe a C64 és VC20-as gépeken, csak hexadecimálisan, esetleg annak jelzése, hogy a szóban forgó eljárás vagy változó ezeken a gépeken nem található meg.

A hibakezelésről annyit, hogy a C64-en és a VC20-on hiba esetén a programfutás hibáuzenettel félbeszakad. A C16-on ez rendszerint csak akkor történik meg, ha saját hibakezelő rutinunk nem aktív (lásd a TRAP BASIC utasítást).

### Változók

LINNUM \$14-\$15 (20—21)  
16 bites, előjel nélküli egész szám. A BASIC általában utasítássorszámok, illetve gépi címek átmeneti tárolására használja. (C64, VC20: ugyanitt)  
TXTTAB \$2B—\$2C (43—44)

# Az ismeretlen C16

## Rendszerváltozók, beépített rutinok

Mutató a BASIC programterület és a programszöveg kezdetére. (C64, VC20: ugyanitt)  
VARTAB \$2D—\$2E (45—46)

Mutató a BASIC változóterület kezdetére. (C64, VC20: ugyanitt)  
ARYTAB \$2F—\$30 (47—48)

Mutató a BASIC tömbök kezdetére. (C64, VC20: ugyanitt)  
STREND \$31—\$32 (49—50)

Mutató a BASIC tömbök utáni első bájtira. Ez a cím a szabad terület alsó határa. (C64, VC20: ugyanitt)  
FRETOP \$33—\$34 (51—52)

Mutató a karakterláncok által elfoglalt terület alsó határa. Ez egyúttal a szabad terület utáni első, nem szabad bájt címe. (C64, VC20: ugyanitt)  
FRESPEC \$35—\$36 (53—54)

Segédmutató a karakterláncműveletek elvégzéséhez. A létrehozandó karakterlánc kijelölt kezdőcímét tartalmazza. A programozási gyakorlatban nem találkozunk vele, mert csak a BASIC rendszer belső rutinjai használják. Talán ez az oka annak, hogy az irodalomban sokszor „megfeleldeknek” ennek a változónak a létezéséről. (C64, VC20: ugyanitt)  
NEMSIZ \$37—\$38 (55—56)

A BASIC által használható RAM-terület felső határának mutatója. (C64, VC20: ugyanitt)  
TXTPTR \$3B—\$3C (59—60)  
BASIC szövegmutató: a feloldozás alatti BASIC programszöveg aktuális karakterére mutat. Elsősorban a CHRGET eljárás használja. (C64, VC20: \$7A—\$7B)  
SAREG \$07F2 (2034)

A SYS utasítás használatakor ezen a változón keresztül lehet értéket adni a processzor A regiszterének, a gépi kódú rutin lefutása után innen lehet kiolvasni az A regiszter visszatérési előtti utolsó tartalmát. Szerepe a BASIC és a SYS utasítással hívott gépi rutin közötti

átadásra korlátozódik. BASIC-ből a hozzáférés a POKE utasítással, illetve a PEEK függvénnyel lehetséges. (C64, VC20: \$030C)  
SXREG \$07F3 (2035)

Szerepe azonos az SAREG változóval, de az A regiszter helyett az X regisztert kezeli. Ezt a változót a beépített monitor saját céljaira is használja. (C64, VC20: \$030D)  
SYREG \$07F4 (2036)

Szerepe azonos az SXREG-ével, de az X regiszter helyett az Y regiszterhez tartozik. (C64, VC20: \$030E)  
SPREG \$07F5 (2037)

Szerepe azonos az SAREG-ével, de az állapotregiszter tartalmát lehet vele kezelni. A POKE használatánál óvatossá kell lenni, mert például az I vagy D bit 1-re állításának kellemetlen következményei lehetnek (C64, VC20: \$030F)

### Eljárások

CHRGET \$0473 (1139)  
A BASIC szövegmutató (TXTPTR) eggyel megnöveli, majd az általa mutatott karaktert az A regiszterbe tölti. Ha szökőzt talált (\$20), akkor a fentieket megismétli. Végül beállítja a feltételeket az alábbiak szerint:

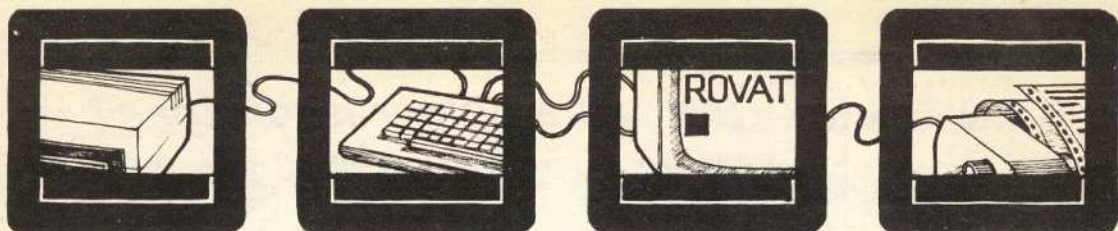
Z=0, ha BASIC utasítás-vég-karaktert (kettőspontot vagy bináris 0-t) talált, különben Z=1.  
C=0, ha ASCII számjegyet talált, különben C=1. (C64, VC20: \$0073)  
CHRGT \$0479 (1145)

A CHRGET belső rutinja. Működésében annyiban különbözik, hogy a szövegmutató értékét nem változtatja. Szököz esetén a CHRGET címére ugrik. (C64, VC20: \$0079)  
INDTXT \$04A5 (1189)

A TXTPTR által mutatott címről annak tartalmát az A regiszterbe tölti. A feltételeket az LDA gépi utasításnak megfelelően állítja be. (C64,







# A C64 GRAFIKUS LEHETŐSÉGEI

A C64 grafikus lehetőségeinek megismeréséhez érdemes tisztoznunk néhány, a normál üzemmóddal összefüggő dolgot.

Ahhoz, hogy a tévé képernyőjén, annak bármely karakterhelyén megjelenjen egy alfanumerikus vagy grafikus jel képe, programozási szempontból a gépnek tudnia kell, hogy hol kezdődik a képernyő-memória, hol kezdődik a színmemória, hol kezdődik a karakter ROM vagy RAM és hogy hol áll a kurzor. Csak ezek ismeretében foglalkozhat a különböző megjelenítési módokból adódó eltérésekkel. A felsoroltak kezdő értékei üzembe helyezéskor, az indítási folyamat alatt automatikusan beállítódnak. Értékeik rendre: 1024, 55296, 53248 (ROM), PEEK (214) a kurzor aktuális sorának, PEEK (221) pedig azon belüli oszlopának a helye.

A keret színét a későbbiekben is figyelmen kívül hagyva, a normál üzemmód grafikai jellemzői a következők: a háttér-papír (kikapcsolt pontok) színe az egész képernyőre választható, de egységes. Értékét POKE 53281, szinkód állítja be. A megjelenített karaktertintá (bekapcsolt pontok) színe a karakterhelyenként változhat. Értékét a PRINT utasításban leütött színvezérlő karakter vagy a színmemória aktuális helyére bevitt szinkód határozza meg. A kép tarkítható az inverz be- és kikapcsolásával, valamint saját karakterkészlet használatával. Az utóbbi ismertetését lásd alább!

Normál üzemmódban a képernyő adott sorának adott oszlopában történő tetszőleges karakter megjelenítéséhez a szokásos PRINT utasításon kívül még két lehetőség:

POKE 1024+40\*sor+oszlop, karakterkód: POKE 55296+40\*sor+oszlop, szinkód, illetve

POKE 214,sor:POKE211, oszlop:SYSS58732:PRINT "ka-



rakter" ahol sor a 0—24, oszlop a 0—39, szinkód a 0—15, karakterkód pedig a 0—255 intervallum egész számainak valamelyike.

A címben szereplő üzemmódok, illetve a saját karakterkészlet használatához különböző átkapcsolások, beállítások szükségesek. A B/K feladatok ellátásáért a VIC chip a felelős. Információt mindig a memória valamely 16 k-s szeletéből veszi a színmemória kivételével, mely nem mozgatható, mindig az 53296 címen kezdődik, és 1000 bajt hosszú. Azt, hogy a VIC melyik 16 k-s szeletből veszi az információt, a CIA—2 A kapujának 0. és 1. bitjei határozzák meg. Az A kapu címe 56576. Ahhoz, hogy a szeletet megváltoztathassuk, előbb az A kapu adatirány-regiszternek alsó két bitjét kell magának állítanunk. Az A kapu adatirány-regiszterének címe 56578. A szelet állítása a

POKE 56578,PEEK(56578)  
OR3 és a  
POKE 56576,(PEEK(56576)  
AND252)ORSZ

parancsokkal megoldható, ahol SZ a 0,1,2,3 értékeket veheti fel. Ekkor a VIC által használt szelet rendre a 49152, 32768, 16384, 0 címeken kezdődik; az alaphelyzet az utolsó!

A képernyő-memória adott szeleten belüli kezdetét az 53272 cím felső négy bite határozza meg. Amennyi ennek — mint négy bites egész számnak — az értéke, annyi k-t ad-

```

0 GOTO10
1 *****
2 *          TOBBSZIN UZEMMOD!          *
3 *****
10 POKE56,55:POKE55,255:CLR
15 CI=14336:V=53248
20 POKE646,9:PRINT"Q"CHR$(8)
25 FORC=256TO263:POKECI+C,0:NEXT
30 FORC=CITOCI+7:POKEC,27:NEXT
35 FORC=8TO15:READAZ:POKECI+C,AZ:NEXT
40 DATA 24, 60,102,126,102,102, 0
45 POKEV+32,5:POKEV+33,15
50 POKEV+34,12:POKEV+35,11
55 POKEV+32,216:POKEV+24,31
60 FORC=8TO1000:POKE1024+C,0:NEXT
65 POKE646,0
70 PRINTTAB(9) "
75 PRINTTAB(9) "AAAAAAAAAAAAAAAAAAAA "
80 PRINTTAB(9) "
85 WAIT203,62:POKE1024,0:SYS1024
  
```

### 1. program

```

0 goto10
1 *****
2 *          bovitett hatterszin uzemmod!          *
3 *****
10 v=53248:Print"Q"chr$(8);
20 Pokev+32,12:Pokev+33,12:Pokev+24,23
30 Pokev+34,2:Pokev+35,6:Pokev+36,0
40 Pokev+17,Peek(v+17)/or64
50 Print"az elet szep!"
60 Print"az ELET SZEPE"chr$(161)
70 Print"az elet szep!"
80 Print"az ELET SZEPE"chr$(161)
90 gosub140
100 ift<t+6then100
110 ifaz=7then90sub140
120 reada:Pokev+36,a:aZ=aZ+1
130 t=t1-goto100
140 aZ=0:restore
150 forl=0to200:nextl:return
160 data11,12,15,15,12,11,0
  
```

### 2. program

junk a szelet kezdetéhez ahhoz, hogy a képernyő-memória kezdő címét megkapjuk. Alaphelyzetben ez az érték 1, így a képernyő-memória a 1024 címen kezdődik.

Az állításnál vigyázzunk, mert az 53272-es cím alsó négy bite az aktuális karaktermemória 2 k-s blokkjának szeleten belüli kezdetét határozza meg, pontosabban a 0. bitet figyelmen kívül hagyjuk. Így tehát a képernyő-memória kezdetét szeleten belüli k-s lépésekben állíthatjuk a

```

20 GOSUB330:COLOUR0,0
25 REM *****
26 REM * NYOMTATO KIRAJZOLASA! *
27 REM *****
30 FILL17,2,35,8,160,7:FCOL17,3,30,7,9
50 FCHR17,3,5,7,0:FILL18,37,3,4,0,9
70 FILL20,37,1,5,160,7
80 FCHR17,8,29,1,228
90 PRINTAT(2,17)"CHR$(164)
100 PRINTAT(8,23)"COMMODORE";
105 PRINT" MPS001"
110 PRINTAT(9,18)"L L":GOSUB430
130 PRINTAT(8,16)"DUPLICATE";
131 :
132 REM *****
133 REM * NYOMTATAS! *
134 REM *****
140 FORC=0TO15:CY=15-C
141 FORL=0TO20:NEXTL
150 MOVE16,8,25,1,CY,8:NEXTC
151 :
160 READX$:X=LEN(X$):IFX$="*"THEN240
170 X$=LEFT$(X$+DUPLICATE",21),21)
180 FORC=0TOX
185 IFMID$(X$,C+1,1)<>" THENPLAY2
186 FORL=0TO20:NEXTL
190 RIGHTW18,9,23,1:NEXTC
200 FORC=0TOX:LEFTW18,9,23,1:NEXTC
210 UPW0,10,21,17:PRINTAT(10,16)"X$
230 GOTO160
231 :
240 PRINT"END
245 :
270 DATA " SIMON'S BASIC"
280 DATA " KARAKTER GRAFIKAI"
290 DATA " DEMO PROGRAM"
300 DATA ""," (C) BY FOLDI ENDRE"
320 DATA "","*
321 :
322 REM *****
323 REM * BEALLITASOK. *
324 REM *****
329 REM *** "e" KARAKTER ATIRASA. ***
330 MEM:DESIGN2,$E000
340 @BBBBBBB
350 @.....
360 @BBBBBBB
370 @.....
380 @BBBBBBB
390 @.....
400 @BBBBBBB
410 @.....
420 RETURN
425 REM *** HANG GENERATOR BEALL. ***
430 VOL15:WAVE1,10000000
460 ENVELOPE1,1,1,1,0:MUSIC1,"JICSD3"
470 RETURN

```

### 3. program

POKE 53272,(PEEK(53272)  
AND15) OR KE

parancsral, ahol KE a 0, 1, 2, ... ,15 egész számok valamelyike.

A karaktermemória 2 k-t foglal el. Ennek szeleten belüli kezdetét a

POKE 53272,(PEEK(53272)  
AND240) OR KA

parancsral állíthatjuk 2 k-s lépésekben (a 0. bit értéke ezért közömbös), ahol KA a 0, 2, 4, ... ,14 egész számok valamelyike lehet.

Alapállapotban a karakter ROM a B/K regiszterek mögött, alatt helyezkedik el az 53248—56832 címeiken. Ezt a területet csak a VIC érheti el. Saját karakterek használatához a karakter ROM-ot a RAM valamely, az előzőekben ismertetett 2 k-s részére másoljuk át, majd a szükséges karakterképek megváltoztatása után hajtsuk végre az átkapcsolást. A másolás alatt megszakításkérés nem adható ki, a B/K regisztereket ki kell kapcsolni, ezért a billentyűzet is tiltva van! A másolás előtt tehát adjuk ki a  
POKE 56334,PEEK(56334)  
AND254  
— megszakítás tiltva  
POKE 1,PEEK(1)AND251 — B/K ki parancsokat, majd a másolás végén a  
POKE 1,PEEK(1)OR4  
— B/K be,  
POKE 56334,PEEK(56334)  
ORI — megszakítás engedélyezve  
parancsokat. Ezek után következhet a karakterkészlet új alakjának betöltése az adott helyre, majd az átkapcsolás az új karakterkészletre. Az új karakterkészlet helye általában a BASIC terület elején van, ezért először a már ismert módon ennek beállításáról gondoskodjunk.

## Többszínű karakter-üzemmód

Az 53281-es cím tartalma a papír vagy a háttér-0 színt határozza meg, az egész képernyőre érvényesen. Az 53282 és 53283 címek a szakirodalom szerint a háttér—1 és háttér—2 színeket adják, mi azonban inkább tinta—1 és tinta—2 színeknek neveznénk őket, hiszen miattuk lesz többszínű a karakter! A tényleges tinta színe karakterhelyenként változhat, értékét a színmemória aktuális helyének alsó három bite határozza meg a 0—7 szinkódokkal!

Mit jelent ez a sok szín? A karakter nyolc bájtnakban biteit csoportosítsuk kettesével balról jobbra! A kapott bitpárok értéke 00, 01, 10, 11 lehet. A helyükön a képernyőn lévő két képpont színe rendre háttér—0, tinta—1, tinta—2 és tintaszínű lesz, ha a többszínű üzemmódot bekapcsoltuk, és a színmemória aktuális helyének 4. bite is magas. A kép olyan, mintha a karakter „középet” jelentő 11 bitpár tintaszínt karakterenként változtatnánk, a karakterkép „széleit” jelentő 01, illetve 10 bitpárnál pedig a 00 által jelölt háttérhez viszo-

nyítva két különböző színnel kihúznánk a karakter bal és jobb oldalát.

A többszínű üzemmód használatához az 53270-es cím 4-es bitejét kell magasra állítani, azaz a  
POKE 53270,PEEK(53270)  
OR16  
parancsral be-, a  
POKE 53270,PEEK(53270)  
AND239 parancsral kikapcsolhatjuk.

Ha a bekapcsolás után az aktuális színmemória szinkódja a 0—7 tartományba esik, akkor a karakter normál üzemmódban jelenik meg a választott színben, ellenkező esetben többszínű üzemmódban, de a tinta színet csak az alsó három bit (0—7) határozza meg. Így a többszínű és normál megjelenítés a színmemória aktuális helye 4. bitejének magasra-alacsonyra állításával karakterenként vezérelhető, azaz a két megjelenítési mód egy képernyőn belül is keverhető. Természetesen ebben az üzemmódban is használhatunk saját tervezésű karakterkészletet.

## Bővített háttérszín-üzemmód

A többszínű üzemmódban az egész képernyőre érvényes háttér—0 szín, illetve tinta—1 és tinta—2 szín mellett csak a tinta színet változtathatjuk karakterenként, most azonban négy háttérszint definiálhatunk. Azt, hogy az aktuális helyen a karakter milyen színű háttér előtt jelenjen meg, a négy szín közül karakterhelyenként megválaszthatjuk, a tinta színével együtt. A tinta színet a PRINT utasításba vitt színvezérlő karakterekkel vagy a színmemória aktuális pozíciójára beírt szinkódokkal határozhatjuk meg.

A karakterenként választható négy háttérszint kódját (háttér—0 — háttér—3) az 53281, 53282, 53283, 53284 címekre írhatjuk be. Az üzemmód beállításához az 53265 cím 6. bitejét kell magasra állítani, azaz a be-, illetve kikapcsolást a  
POKE 53265,PEEK(53265)  
OR64 illetve a  
POKE 53265,PEEK(53265)  
AND191 parancsok végzik el.

A sok színek ára most is van, ugyanis megjeleníteni csak a 0—63 képernyőkódú eredeti vagy saját karaktereket tudjuk, mert a karakterköd 7. és 6. bitejének értéke együttesen határozza meg az aktuális karakterhelyen érvényes háttér színt. Ha két bit együttes értéke 00, 01, 10, 11, akkor a ka-

rakter az aktuális tintaszínnel rendre a háttér—0, ..., háttér—3 színű papír előtt jelenik meg.

## A programok

Az 1. program azt mutatja be, hogy miként lehet többszínű üzemmódba kapcsolt képernyőn normál karaktert vagy szöveget megjeleníteni. Magyarazata:

- 10 a BASIC terület végének beállítása
- 15 szelet és VIC báziscímének beállítása
- 20 tintaszín-beállítás, karakterkészlet-váltás
- 25 szövekkarakter nullázása
- 30 @ átirása
- 35 nagy A helyén nagy A készítés (több karakter nincs!)
- 45 képernyőszínek beállítása
- 55 többszínű üzemmód és szeletátkapcsolás
- 60 képernyőtöltés
- 65 fekete tinta
- 70 kiírás
- 80 billentyűre vár, alaphelyzetbe vissza

A 2. program a bővített háttérszín-üzemmódot illusztrálja. Magyarazata:

- 10 VIC báziscím, karakterkészlet-váltás letiltása
- 20 képernyőszín, kisbetű
- 30 háttérszín
- 40 bővített háttérszín-üzemmód bekapcsolása
- 50 normál kiírás
- 60 shiftelt kiírás
- 70 inverz kiírás
- 80 shiftelt inverz kiírás
- 90 alaphelyzetbe állít, vár
- 100 villogat
- 160 szinkódok

A 3. program karaktergrafikai összefoglaló, mely egyben a legerjedtebb BASIC-bővítés, a SIMON'S BASIC lehetőségeinek egy részét is felvillantja. Mozgó grafika, egy MPS801-es nyomtatott szimulál. Begépelése előtt töltjük be és futtassuk a SIMON'S BASIC programot. Magyarazata:

- 20 képernyőszín-beállítás
- 30—90 nyomtatótest kirajzolása
- 100—105 feliratozás
- 110 nyomtatófej-rajzolás, hanggenerátor hívása
- 130 papírcsikrajzolás
- 140—150 üres papír megjelenítése
- 160—230 szöveg nyomtatása
- 270—320 a nyomtatandó szöveg
- 330—420 a nyomtatótest csikok részét előállító karakter tervezése a hanggenerátor beállítása, a 460-as sor inverz bal alsó negyedkörre F2 lenyomására jelenik meg. FÖLDI ENDRE — ÉNEKES FERENC
- 430—



## Kódkonverter

Az alábbi kis assembly program segítségével bármely, 0—65535-ig terjedő decimális számot átválthatunk hexadecimális számrendszerbe. A programot töltjük be a gépi kódú területre, vagy a címetek átirva és a BASIC elejére. A BASIC program az A változó értékét átadja a gépi kódú résznek. A-t itt binárisan ábrázoljuk, ezért kell &7904-re &B2-t tölteni. A gépi

```

10:PRINT "DIÁK MAGAZIN ROVAT"
20:CALL G2001:PRINT "END"
30:CALL G2001:PRINT "END"
40:CALL G2001:PRINT "END"
50:CALL G2001:PRINT "END"
60:CALL G2001:PRINT "END"
70:CALL G2001:PRINT "END"
80:CALL G2001:PRINT "END"
90:CALL G2001:PRINT "END"
100:CALL G2001:PRINT "END"
110:CALL G2001:PRINT "END"
120:CALL G2001:PRINT "END"
130:CALL G2001:PRINT "END"
140:CALL G2001:PRINT "END"
150:CALL G2001:PRINT "END"
160:CALL G2001:PRINT "END"
170:CALL G2001:PRINT "END"
180:CALL G2001:PRINT "END"
190:CALL G2001:PRINT "END"
200:CALL G2001:PRINT "END"
210:CALL G2001:PRINT "END"
220:CALL G2001:PRINT "END"
230:CALL G2001:PRINT "END"
240:CALL G2001:PRINT "END"
250:CALL G2001:PRINT "END"
260:CALL G2001:PRINT "END"
270:CALL G2001:PRINT "END"
280:CALL G2001:PRINT "END"
290:CALL G2001:PRINT "END"
300:CALL G2001:PRINT "END"
310:CALL G2001:PRINT "END"
320:CALL G2001:PRINT "END"
330:CALL G2001:PRINT "END"
340:CALL G2001:PRINT "END"
350:CALL G2001:PRINT "END"
360:CALL G2001:PRINT "END"
370:CALL G2001:PRINT "END"
380:CALL G2001:PRINT "END"
390:CALL G2001:PRINT "END"
400:CALL G2001:PRINT "END"
410:CALL G2001:PRINT "END"
420:CALL G2001:PRINT "END"
430:CALL G2001:PRINT "END"
440:CALL G2001:PRINT "END"
450:CALL G2001:PRINT "END"
460:CALL G2001:PRINT "END"
470:CALL G2001:PRINT "END"
480:CALL G2001:PRINT "END"
490:CALL G2001:PRINT "END"
500:CALL G2001:PRINT "END"
510:CALL G2001:PRINT "END"
520:CALL G2001:PRINT "END"
530:CALL G2001:PRINT "END"
540:CALL G2001:PRINT "END"
550:CALL G2001:PRINT "END"
560:CALL G2001:PRINT "END"
570:CALL G2001:PRINT "END"
580:CALL G2001:PRINT "END"
590:CALL G2001:PRINT "END"
600:CALL G2001:PRINT "END"
610:CALL G2001:PRINT "END"
620:CALL G2001:PRINT "END"
630:CALL G2001:PRINT "END"
640:CALL G2001:PRINT "END"
650:CALL G2001:PRINT "END"
660:CALL G2001:PRINT "END"
670:CALL G2001:PRINT "END"
680:CALL G2001:PRINT "END"
690:CALL G2001:PRINT "END"
700:CALL G2001:PRINT "END"
710:CALL G2001:PRINT "END"
720:CALL G2001:PRINT "END"
730:CALL G2001:PRINT "END"
740:CALL G2001:PRINT "END"
750:CALL G2001:PRINT "END"
760:CALL G2001:PRINT "END"
770:CALL G2001:PRINT "END"
780:CALL G2001:PRINT "END"
790:CALL G2001:PRINT "END"
800:CALL G2001:PRINT "END"
810:CALL G2001:PRINT "END"
820:CALL G2001:PRINT "END"
830:CALL G2001:PRINT "END"
840:CALL G2001:PRINT "END"
850:CALL G2001:PRINT "END"
860:CALL G2001:PRINT "END"
870:CALL G2001:PRINT "END"
880:CALL G2001:PRINT "END"
890:CALL G2001:PRINT "END"
900:CALL G2001:PRINT "END"
910:CALL G2001:PRINT "END"
920:CALL G2001:PRINT "END"
930:CALL G2001:PRINT "END"
940:CALL G2001:PRINT "END"
950:CALL G2001:PRINT "END"
960:CALL G2001:PRINT "END"
970:CALL G2001:PRINT "END"
980:CALL G2001:PRINT "END"
990:CALL G2001:PRINT "END"

```

rész A-t átváltja hexadecimálissá, majd AS-be tölti.

A programot a saját készítésű assembly-disassembly programmal csináltam, és kiírtam KA 150-es nyomtatón. A programot DEF H-val futtathatjuk, majd írjuk be az átváltandó számot.

BALÁZS GYÖRGY

# Monológ

Végeztes is megértük! Hogy mit? Hát te nem vettél részt a számítástechnikai OKTV iskolai fordulóján? De igen? Akkor biztosan egyetértés velem, hogy ez volt az utóbbi évek legszínvonalasabb versenye. Már úgy érte, hogy a feladatlap... Azt mondd, neked is tetszett? Hogy nem ragaszkodtál kizárólag a HT ismeretéhez? Hát persze, talán ez volt a legnagyobb előrelépés a régi-ekhez képest. Mi? Hogy már nemcsak azt kellett kitalálni, minek bonyolították feleslegesen a feladatot holmi változótipusokkal!

De nem volt rossz az általános ismeretek tesztje sem. Sőt! Belekóstoltattak még egy másik nyelvbe is, ez volt az egyik legérdekesebb feladat. Szinte felüldülés volt a sok BASIC után. Hogy nem is volt benne olyan sok? Ja, igen. Mit szólna az algoritmusos feladathoz? Kissé bizarr volt, nem? De azért jó ötlet volt ez is. Annak örültél, hogy jól áttekinthető programokat kellett elemezni? Na igen, ez régebben nem volt túlságosan jellemző.

Reménykedjünk, hogy ezennél legalább ilyen színvonalúak lesznek a számítástechnikai versenyek. Most meg hívó rohanás? Mi? Bejutottál a második fordulóra? Gratulálok! — mondta X, és új partner után nézett, akinek tovább esztelheti a feladatlap nagyszerűségét.

PALLAGI LÁSZLÓ  
Vörösmarty Gimnázium  
Érd

## SEGÉDPROGRAM KEZDŐK RÉSZÉRE

A program segítségével meghatározhatjuk a HT-gépeken a grafikus karakterek kódját. A grafikus karakter hat részből áll:

1	2
4	8
16	32

Kódját megkaphatjuk, ha 128-hoz hozzáadjuk azokat a számokat, amelyek a világitó mezőkben találhatóak. Ennek kiszámítását könnyíti meg a program. Futtatásakor a kiírt számok egy-egy mezőt jelölnek:

1	2
3	4
5	6

A kivilágítani kívánt pontoknak megfelelő számokat üssük le! Ha befejeztük, akkor a NEW LINE billentyű lenyomása után megkapjuk az elkészült karakter kódját és a karaktert is, eredeti nagyságban. A programot akkor érdemes használni, ha több grafikus karakter felhasználásával aka-



```

<<000 REM GRAFIKUS
<<KARAKTEREK HT=1
<<010 DATA 1,4,2,6,4,
<<128,8,129,14,192,32,193
<<020 FOR A=1 TO 31
<<READ A(A),B(A),HEX(A)
<<030 CLS:PRINT:PRINT"12"
<<PRINT"34":PRINT"56"
<<040 PRINT"VEGE:NEW LINE"
<<050 B=128
<<060 A$=INKEY$
<<IF A$="*",GOTO 40060
<<070 IF A$=CHR$(13),
<<GOTO 40120
<<0800 X=VAL(A$)
<<0900 IF C(X)=0,
<<B=B+(A(X)-1)C(X)=1
<<0100 IF B(X)=0,
<<PRINT$(X),CHR$(191)
<<0110 GOTO 40060
<<0120 PRINT$(50,B,CHR$(B)
<<0130 IF INKEY$="*",
<<GOTO 40130
<<0140 RUN 40000

```

runk képernyőt tervezni. Ilyenkor célszerű a készülő program után másolni (MERGE — lásd a µM 1985/1. szám 8. oldalán); használata után pe-

dig DELETE 60000—60130 utasítással törölhetjük.  
JOÓ ZOLTÁN  
Piarista Gimnázium  
Budapest

# Assemblerek, cross-assemblerek

## 6. Az operátorkészlet I.

A következőkben megadjuk a rendszer leírásához használt operátorok definícióit. Az operátorokkal kapcsolatban előljáróban két dolgot kell kiemelni, illetve ismételni.

Egyrészt a bemutatott megoldás csak egy a sokféle lehetséges közül. Több olyan operátor fog szerepelni (például a 13., 16., 19. operátorok), amelyek a feladata, működése annyira összetett, hogy talán célszerűnek látszana további, elemibb operátorok együttesére felbontani őket. Mégsem tartottuk indokoltnak az operátorok ilyen mértékű felaprózását; célunk inkább a rendszer működési folyamatában a nagyobb léptékű összefüggések megvilágítása. Így alakult ki a bemutatásra kerülő heterogén operátorhalmaz.

Másrészt, az előző fejezetek tárgyalásmódjának az általánosíthatóságot hangsúlyozó jellegével szemben, egyes operátorok működésének leírásakor már kevésbé tudunk eltekinteni a kiválasztott nyelvről, az Intel 8080 mikroprocesszor assembly nyelvén sajátosságaitól. Ezeknek a korláta a korábban tett feltételezések és korlátozások mellett az operátorok felvételére helyenként döntő jelentőségű. Természetesen más mikroprocesszor más nyelvéhez írt fordítóprogram hasonló feladatokat ellátó operátorokat igényel, ha ezek minden részletben nem is egyeznek meg az itt bemutatottakkal.

Ebben és a következő részben összesen 24 operátort definiálunk, amelyek segítségével építjük fel a rendszer kapcsolatábráját.

**1. operátor: paraméterlekérdező és kezdeti érték-beállító (1. ábra)**

**Nyolcrészes sorozatunk a mikroszámítógépek assemblereiről szól. Célja, hogy a cross-assemblerek példáján keresztül megismertesse az olvasót az assembler programok működésével. A bemutatáshoz a rendszermodellezési eszközöket használjuk fel, s így készítjük el az assemblerek működésének egy szabványosított algoritmusát, modelljét. E cikkünk a sorozat hatodik részze.**

**1. (Cross-)assemblerek és a rendszermodellezés**

**2. Az Intel 8080 assembly nyelv**

**3. A (cross-)assembler program mint rendszer**

**4. A rendszer működése I. – A fordítás két menete**

**5. A rendszer működése II. – Táblák és adatterületek**

**6. Az operátorkészlet I.**

**7. Az operátorkészlet II. – A rendszer kapcsolatábrája**

**8. Példák a rendszer működésére**

Feladata:

— a programindítás után lekérdezi a rendszer bemeneti paramétereit:

○ a bemeneti periféria típusát, illetve mágneslemez esetén a forrásprogramfájl nevét,

○ a kimeneti periféria típusát, illetve mágneslemez esetén a tárgyprogramfájl nevét,

— beállítja a rendszerváltozók és egyéb változók kezdeti értékét, azaz:

○ a PASS, DEF, MIND, IF1, IF2 az A1 állapotnak megfelelő értékeket vesz ki fel,

○ a pufferterületek és táblák mutatói a megfelelő kezdeti értékre állítodnak be,

○ a sorfeldolgozási ciklussal kapcsolatos indikátorok is felveszik a kezdeti értéket,

— kijelöl a mágneslemezen előre meghatározott nagyságú munkaterületeket a forrásprogram, a modul- és a makrócímketáblák számára,

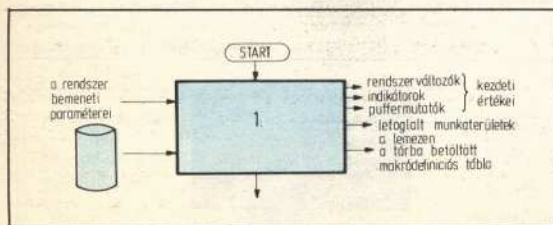
— betölti a makródefiníciós táblát (vagy annak az utolsó rekordját) a központi tárbá.

**2. operátor: forrásprogram-olvasó nem lemez bemeneti perifériáról (2. ábra)**

Feladata: egy utasítássort olvas be a bemeneti perifériáról az első bemeneti pufferterrültre (SPB1).

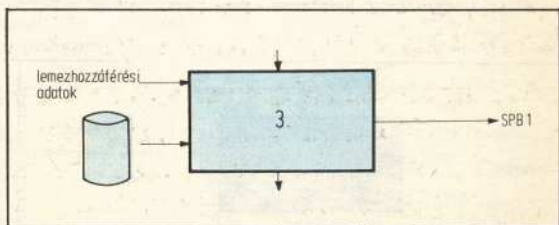
**3. operátor: forrásprogram-olvasó mágneslemezzel (3. ábra)**

Feladata: egy utasítássort olvas be SPB1-re mágneslemezzel, akár mint az első menetben a le-



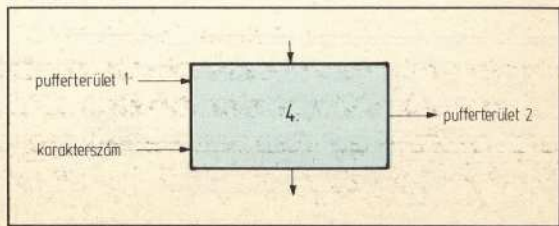
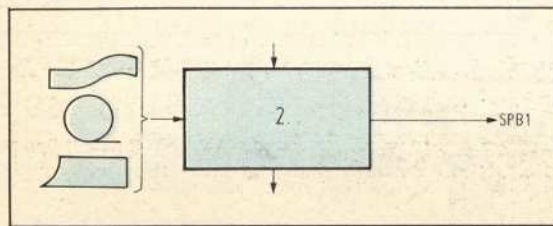
1. ábra

2. ábra



3. ábra

4. ábra



mezre felvitt kifejtett forrásprogram átmeneti tárolójáról, akár mint a makródefiníciók tárolóhelyéről.

**4. operátor: karakterlánc-mozgató (4. ábra)**  
Feladata: egy adott pufferterületről adott számú karaktert visz át egy másik pufferterületre.

**5. operátor: egy forrásprogramot mezőkre bontó (5. ábra)**

Feladata: az SPB2 pufferterületen tárolt forrásprogramot logikai alkotóelemeire bontja fel, s kitölti az egyes mezőknek megfelelő önálló pufferterületeket.

A mezők: megjegyzésmező, címkemező, műveletikód-mező, operandusmező. Az operandusmezőbe csak a valódi operan-

dustartalmat viszi át, és esetleg a forrásprogram (ok) után vagy helyett álló megjegyzés helyét szóközzel tölti ki. A mezőhatárokat előre rögzítettek.

**6. operátor: több-bites párhuzamos komparátor (6. ábra)**

Feladata: a bemenetére adott két bitkombináció egyenlőségét megállapítani. Ez jelölheti két 1 bájttal hosszú karaktert vagy egy 2 bájtos egész típusú változó értékének és egy konstansnak az összehasonlítását.

**7. operátor: formális-aktuális paramétercserét végző (7. ábra)**

Feladata: makrókifejtés esetén — ha a sor nem a kezdő, a MACRO direktívát tartalmazza — elvégzi a definícióban szereplő

formális paramétereknek a hívással megadott aktuális paraméterekre való cseréjét. Elvileg mind a címké-, mind a műveletikód-, mind az operandusmezőben megengedett a formális paraméterek használata, fordítóprogramunkban azonban csak az operandusmezőben tesszük lehetővé a paramétercserét. Az operátor egyenként megvizsgálja, hogy az operandusmezőben — ha ez nem üres — szereplő, s egymástól vesszővel elválasztott egyszerű kifejezések nem egyeznek-e meg a megszakítási hely és makróparaméterek veremtarában az aktuális bejegyzésben tárolt valamelyik formális paraméterrel. Ha valahol egyezést talál, akkor a kifejezés helyére a megtalált formális paraméter sorszámával azonos sorszámú aktuális paramétert teszi be, amelyet ugyanebből a verembejegyzésből olvas ki. Látható, hogy rendszerünkben nem engedjük meg formális paramétereknek összetett kifejezésekben való megjelenését, bár az operátor működése kiterjeszthető lehetne erre az esetre is.

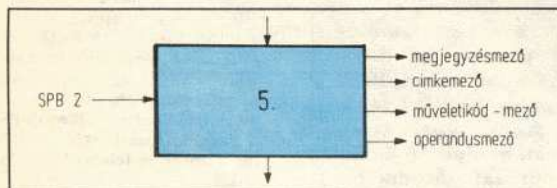
Az aktualizált operandusmezőt elhelyezi az SPB2 pufferterület megfelelő helyére is, hogy a lemezre az aktualizált forrás sor kerüljön fel.

Ha a MIND rendszerváltozó értéke nulla, akkor az operátor hívása után azonnal be is fejezi a működését.

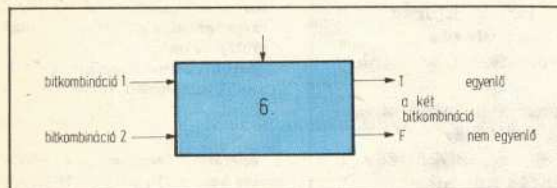
**8. operátor: címketáblában keresés és címkeinformáció szolgáltató (8. ábra)**

Feladata: a modul- vagy az aktuális makró-címketáblában egy megadott szimbolikus név meglétét dönti el, s ha a keresés sikeres, akkor a névhez tartozó címkebejegyzést hozzáférhetővé teszi. Modulcímketáblában való keresés esetén a keresést nemcsak az MLB pufferterületen, hanem a már a lemezre kivitt címke-rekordoknak a DFB3 pufferterületre való egyenkénti visszahívásával is végzi. Makrócímkék esetén a kutatási terület a DFB4 puffer (K=1). A megtalált címkebejegyzést a pufferterületen belüli kezdőcím szolgáltatásával teszi hozzáférhetővé.

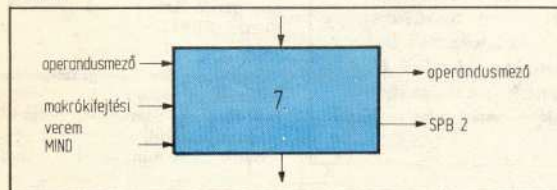
A második menetben, ha a keresés sikeres, akkor beállít egy hibaiüzenet-indiká-



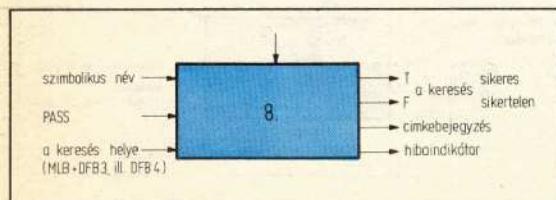
5. ábra



6. ábra

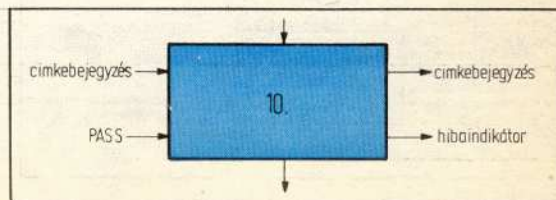
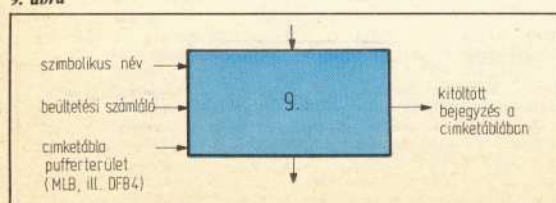


7. ábra



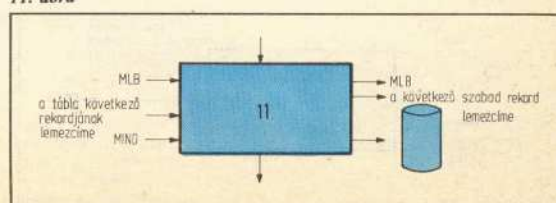
8. ábra

9. ábra



10. ábra

11. ábra



12. jelezvén, hogy a szimbolikus név az első menüben azért nem került be a táblába, mert a tábla számára fenntartott munkaterület betelt.

**9. operátor: címkeinformációt a címkeablákba elhelyező (9. ábra)**

Feladata: a megadott címke vagy azonosító név alapján bejegyzést készít a modul vagy az aktuális makrócímkeablákban. A bejegyzés az MLB, illetve a DFB4 pufferterület szabad területének elejére kerül, az operátor a szimbolikus nevet helyezi el, s relatív címék feltételezve kitölti a flag- és az értékadatokat. A bejegyzés elkészítése után a puffer szabadterület-mutatóját elő-állítja.

Ha akár a makrócímkek, akár a modulcímék esetén a lemezen lefoglalt munkaterület már betelt, akkor a szimbolikus név nem kerül be a táblába (a megengedettnél több modul-, illetve makrócímke szerepel a programban).

**10. operátor: többszörös címkedefiniálást kezelő (10. ábra)**

Feladata: nem megengedett többszörös címkedefiniálás esetén érvényteleníti a címkeablákban szereplő bejegyzést. A korábban elmondottak értelmében minden olyan esetben, amikor a fordítóprogram az első menüben a címkemező tartalmával megegyező nevű és nem SET direktívával generált bejegyzést talál a megfelelő címkeablákban, hiba van a forrásprogramban. Az operátor a hibát jegyzi be a címkeablákba azzal, hogy a flaget a többszörös definiálást jelző értékre (esetünkben 5-re) állítja be, s a címke értékét nullára változtatja meg.

Ha a második menüben hívjuk az operátort, akkor azt vizsgálja meg, hogy a bejegyzésben a címkeflag egyenlő-e a többszörös definiálást jelző értékkel (5-tel). Ha igen, akkor a többszörös definiáláshoz tartozó hibaindítót a hiba jelzésére állítja be.

**11. operátor: a címkeablát lemeze író (11. ábra)**

Feladata: a modulcímkeablák pufferterületének (MLB) betelttségétől függően egy rekordnyi címkeadatot ír ki a lemeze, a címkeablák számára fenntartott munkaterület sora következő rekordjaként.

A rekord kiírás után módosítja a munkaterület következő szabad rekordjának kezdőcímét, s az MLB puffermutatóját a pufferterület elejére állítja be, jelezve, hogy a puffer üres, illetve tartalma érvénytelen. Gyorsíthatjuk a rendszer működését, ha a fenti vizsgálatot az operátor csak  $MIND = \emptyset$  esetén hajtja végre, mert makrókifejtés esetén MLB-re írás nem történik, így az nem is telhet be.

Ha  $K > 1$ , akkor az operátornak ugyanezeket a feladatokat makrókifejtés esetén a makrócímkeablákra is végre kell hajtania.

**12. operátor: makróhívást felismerő (12. ábra)**

Feladata: a műveletikód-mezőben levő karaktersorozatot összehasonlítja a makródefiniációs táblában szereplő makrónevekkel. Ha valahol egyezést talál, akkor a makrónévhez tartozó bejegyzést hozzáférhetővé teszi a definiációs táblában.

**13. operátor: makrókifejtést indító (13. ábra)**

Feladata: a makrókifejtés indításával kapcsolatos, a fordítási menet által előírt műveletek elvégzése. Mivel a makrókifejtés indításának folyamatával a 2. és az 5. részben részletesen foglalkoztunk, itt csak vázlatosan soroljuk fel a szükséges műveleteket.

Az operátor által végzett műveletek az első menüben:

— Ha a megszakított szinten is makrókifejtéses állapotban van a rendszer ( $MIND \neq \emptyset$ ), akkor a hozzá tartozó makrócímkeablát átmenetileg lezárja és kiírja a LEMEZCÍM12 ( $K > 1$ ), ill. LEMEZCÍM11 ( $K = 1$ ) által meghatározott kezdőcímű lemezerületre, DFB4 mutatóját a terület elejére állítja.

— Mélyíti a megszakítási hely és makróparaméterek, valamint a makrócímkeablaveremtarakat: nevezetesen az előzőbe a megszakítási hely és az aktuális paraméterek beírásával egy nem lezárt bejegyzést készít, az utóbbiba pedig az LMCT megnövelt értékét írja be. Érvényes „megszakítási hely” adat csak abban az esetben van, ha a megszakított szinten a bemeneti periféria mágneslemez. (A megszakítási hely meghatározása külön figyelmet kíván, mert a sorfeldolgozási ciklussal párhuzamosan indi-

tott olvasást ez a sort már behozta SPB1-re, így a pillanatnyi lemezadat- és rekordelhelyezési mutatók már nem az elmentendő állapotot tükrözik.)

— Ha a megszakított szint a nulladik és a bemeneti periféria nem mágneslemez, akkor a sorfeldolgozási ciklussal párhuzamos (a következő sort szolgáltató) olvasás befejeződése után elmenti a beolvasott sort SPB1-ről SPB3 területre.

— A makródefiniációs táblából nyert, s a makródefiniációnak a lemezen való elhelyezkedését megadó kezdőcím alapján a 3. operátor belső hívásával beolvassa a definíció első — a MACRO direktívát tartalmazó — sorát az SPB1 területre.

— MIND értékét eggyel növeli.

Az operátor által végzett műveletek a második menüben:

— növeli LMCT értékét és mélyíti a makrócímkeablá-vermet,

— az LMCT új értéke által meghatározott lemezcintről beolvassa az új makrókifejtési szinthez tartozó makrócímkeablát, illetve ennek első rekordját,

— MIND értékét eggyel növeli.

**14. operátor: műveletikód-kereső és műveletikód-információkat szolgáltató (14. ábra)**

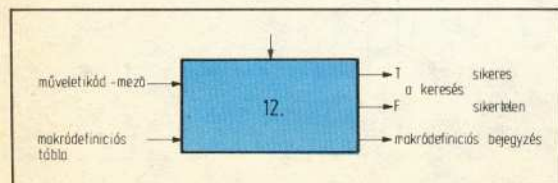
Feladata: a műveletikód-mező tartalmával azonos nevű bejegyzést keres a műveletikód-táblában vagy annak csak egy kijelölt részén. Ha a keresés sikeres, akkor a bejegyzésben szereplő műveletikód-információkat hozzáférhetővé teszi. Ha a keresés sikertelen, akkor beállít egy ennek jelzésére szolgáló hibaindítót.

A keresés tartománya — tehát, hogy az egész táblában vagy annak csak egy részében kutasson — például a tartomány kezdőcímének és a megvizsgálandó műveletikód-bejegyzések számának megadásával jelölhető ki. (A kapcsolatábrában a keresendő karakterláncokat tüntetjük fel ezen adatok helyett.)

**15. operátor: direktívfelismerő (15. ábra)**

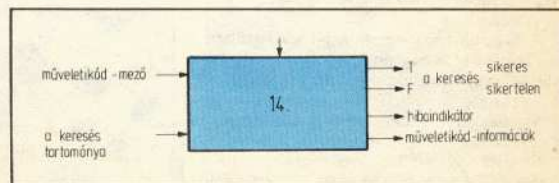
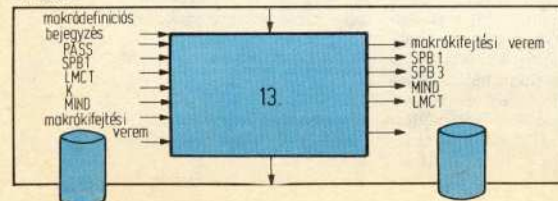
Feladata: a műveletikód-információk alapján eldönti, hogy a műveletikód-mezőben gépi utasítás vagy direktíva mnemonikus kódja szerepel-e.

VÁRGEDŐ TAMÁS



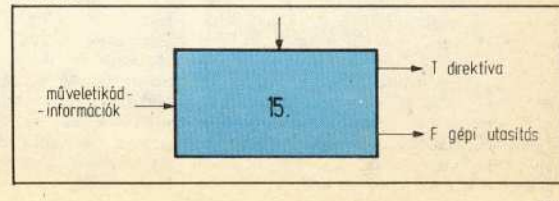
12. ábra

13. ábra



14. ábra

15. ábra



# Hogyan működnek a lapos kijelzők?

A képernyős megjelenítők rendkívül fontos szerepet töltenek be a számítógép-használat során, hiszen azok teremtik meg az ember számára a géppel való vizuális kapcsolatot. A napjainkban használatos adatmegjelenítőknak szinte kivétel nélkül katódsugárcsőes képernyőjük van. Ezek hátránya viszonylag nagy energiaigényükön túl a terjedelmes kivétel is.

A kutatók megpróbálták olyan új utakat, elveket keresni, amelyek lehetővé teszik a képernyős megjelenítők minél laposabb kivitelű előállítását. Az eddig elért eredmények tükrében tekintsük át — a teljesség igénye nélkül — azokat a részben vagy teljesen új működési elveket, amelyeket már sikerrel alkalmaztak a tömeggyártásban is.

## Kristályos folyadék (KRIFO) kijelzők

Má ez a leggyakrabban használt kijelzési technika a lapos kijelzőkben. Nem számít énegyesen újnak, hiszen a zsebszámológépek és a kvarcórák kijelzőjeként már találkozhattunk velük. Újdonságot csupán a kijelzőfelület megnövekedése jelent. E technikában a japán ipar áll az élen: színes, „hordozható” tévét is gyártottak ilyen típusú képernyővel.

A KRIFO-képernyők a passzív kijelzők családjába tartoznak, mivel — ellentétben a más elveken működő kijelzőkkel — nincs saját fényük, hanem a környezeti fényt hasznosítják. A működési elv azon alapul, hogy a KRIFO megváltoztatja optikai tulajdonságait a ráadott feszültség hatására. Ilyen típusú kijelző felépítését láthatjuk az 1. ábrán.

Kövessük végig a fény útját a kijelzőben. A fény először egy polarizációs szűrőn halad át, majd az elektródamátrixszal ellátott KRIFO-rétegbe jut. További útja során egy második polarizációs szűrőn áthatolva, a végállomást jelentő tükrörről visszaverődve, a fentebb leírt utakat újból végigjárva jut a szemünkbe. Az így módon visszaverődő fény tehát akadálytalanul éri el szemünket. Ha azonban az elektródamátrix egyik pontjára feszültséget adunk, akkor megváltozik a fény polarizációs iránya, így annak áthaladása lehetetlenné válik, és egy sötét pont keletkezik a mátrixpontban. Az elektródamátrixot sűrű drothálólóként képzeljük el. Belátható, hogy az így nyert kép annál finomabb felbontású, minél sűrűbb a háló.

Az ilyen elvű kijelzők rendkívül laposak, viszonylag kis súlyúak. Alacsony az energiaigényük. E szempontok miatt különösen

alkalmasak „hordozható”, táskakivitelű számítógépekhez. Jóllehet kevés környezeti fényt igényelnek, nagy megvilágítású környezetben is használhatók. Áruk viszonylag alacsony. Hátrányuk, hogy a többi kijelzőkhöz képest kontrasztszegényebb képet adnak, és viszonylag szűk szögterületben láthatók. Az ilyen elvű színes képernyők előállítása költséges.

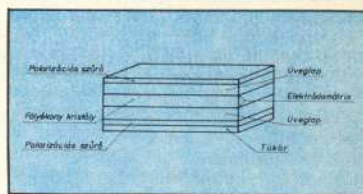
## Ionizált gázos (plazma)képernyők

A plazmaképernyő fénycsővel alapján működik, és mostoha körülmények között is kifogástalanul használható. Ezért főleg katonai területeken terjedt el, de a számítógépgyártó cégek már irodai alkalmazásra is gyártanak ilyen elven működő kijelzőket.

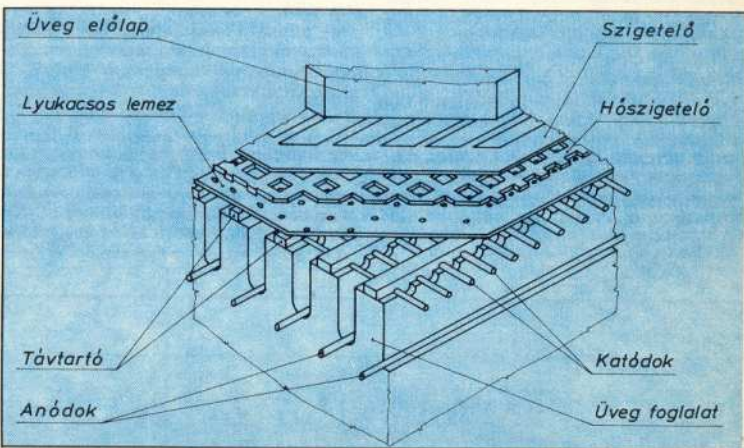
Működésük gázkisülési jelenségen alapul, amelyet a háló alakban elhelyezett anód és katód elektródák hoznak létre. Fel-

get juttat a katódra és pozitívot az anódra, akkor azok keresztezési pontjain a kisülés következtében a gáz világító plazmává alakul. Ez a fényeffektus a homlokfelületen láthatóvá válik a lyukacsos lemez segítségével. A kép felbontás finomságát itt is az elektródaháló sűrűsége határozza meg.

A plazmaképernyők fényereje nagy, kontrasztgazdag képet adnak. Ezáltal a képernyőtartalom jól látható. A kép nyugodt, vibrációmentes. A technológiai fejlettség következtében ma már nagy felü-



1. ábra



2. ábra

építésüket a 2. ábrán láthatjuk. Az üveglapba hosszirányú hornyokat marnak, ebbe fektetik az anódot alkotó fémhuzalokat. Ezekre merőleges irányban helyezkednek el a katódszálak. A vezérlést az elektronika biztosítja. Az így elkészített hálót olyan lyukacsos lemez borítja, melynek furatai pontosan az anód-katód kereszteződési pontjait fölé esnek. Az elektródák közti teret nemesgázzal, például neonnal töltik ki. Ha a vezérlő elektronika negatív feszültsé-

tű, nagy felbontóképességű ernyőket készítenek. Ez a típusú képernyő viszonylag érzéketlen az ütessel, rázkódással szemben.

Az aktív elvű megjelenítők között ez a technika a leggyakoribb. Ma már olyan képernyőket készítenek, amelyek 700 000 képpont megjelenítésére képesek. A megjelenítés színe fekete-fehér, helyesebben fekete-vörös. Színes megjelenítőt még nem sikerült ezzel a technikával előállítani. E megjelenítők hátránya még az is, hogy vi-



# ADOK – VESZEK – CSERÉLEK

Ebben a rovatban rövid, szöveges, a mikroszámítógépekkel kapcsolatos hírdetéseket közlünk. A díjszabás: közületeknek gépelt soronként (60 karakter) 100,— Ft, magánszemélyeknek az első sor 50,— Ft, minden további sor 20,— Ft. Az NJSZT tagjainak az első három sor ingyenes. Hírdetéseiket a szerkesztőség címére várjuk.

■ Commodore 16-os gépre írt játéktérprogramok cseréjéhez partnert keresek. László István, Békéscsaba, Máriássy u. 24.

■ Commodore VC20-ra készült programok eladása: assembler, disassembler, karaktermódosító, finomgrafikai, zene stb. Ár: 100–300 Ft/db. A programok mindegyike alapgépen is használható. Kérjen tájékoztatást! Juhász György, 3100 Salgótarján, Pf. 157.

■ Timex Sinclair 1000 (ZX81) nagyon olcsón eladó. RAM: 2 kb-át. Telefon: 561-274 este.

■ Sinclair ZX81 16 k-s bővítővel, gépkönyvvel eladó. Lutár Zoltán, Budapest, Erzsébet u. 16. 1043. Telefon: 896-006.

■ VZ 200-as számítógéphez 16 kb-átos memóriabővítés eladó. Csáky Szabolcs, Dombóvár, Gyöngyvirág krt. 52.

■ Commodore 64-re készült játéktérprogramokat cserélek. Varga Vince, Szeged, Palánkai u. 18. Telefon: 19-637.

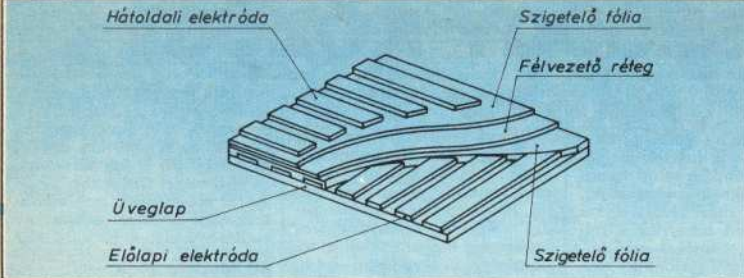
■ Commodore 64-re készült játéktérprogramokat cserélek. Wandler Gergely, Budapest, Krammer S. u. 24. IV. 25. Telefon: 582-462.

■ ZX—Spectrumhoz felhasználói és játéktérprogramokat cserélek. Válaszokat listával erre a címre kérek: Kiss László, Mezőtúr, Szabadság tér 29. 3/3/22. 5400.

■ 48 k-s Spectrumhoz felhasználói és játéktérprogramokat cserélek. Pillány Gábor, Pécs, Bajcsy-Zsilinszky út 3/9. 7622.

■ Eladó Commodore VIC—20, magyar nyelvű gépkönyvvel. Ár: 11 000 forint. Cím: Karcag, Kálvin út 19. 5300

■ VC20 datasettel, 20 k-s bővítővel és szakirodalommal eladó. Tóth Zoltán, Jászberény, Uzsoda u. 1. 5100.



3. ábra

szonylag nagy energiaigényük miatt elemes üzemmódban nem használhatók.

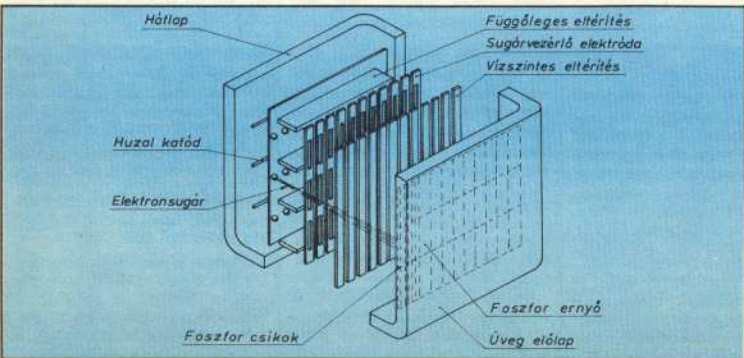
## Világító félvezetős (elektrolumineszcens) kijelzők

Az elektrolumineszcens kijelzők technikai szempontból közeli rokonai a plazmás kijelzőknek, azzal a különbséggel, hogy ez esetben nem ionizált gáz, hanem félvezető anyagból készült réteg világít.

Működésük meglehetősen egyszerű, felépítésüket a 3. ábra mutatja. Itt nem gázkiülés világít, hanem az elektródák által keltezt elektromos mezőben elhelyezkedő félvezető állítja elő a fényt. A plazmás kijelzőkhöz hasonlóan itt is mátrix alakban elhelyezkedő elektródákat találunk a képernyőfelület mögött kifeszítve úgy, hogy a függőleges és vízszintes elektródák között található az a „szendvics”, amely két vékony szigetelőfólia között elhelyezkedő félvezető rétegből áll. A félvezető anyagtól függ az előállított fény színe.

Az elektrolumineszcens kijelzők fénye megfelel a hagyományos képcsővekének. Mivel a világító félvezető réteg szilárd, jobban viselik a rázkódást, ütközést. Rendkívül lapos kivitelben készíthetők, és kis súlyúak. Hátrányt jelent a viszonylag magas elektromosenergia-igény, így elemes üzemmód nem jöhet számításba. Meglehetősen drágák, egyelőre csak monokromatikus kép előállítására alkalmasak. Amerikai és japán kutatólaboratóriumok törekednek színes képernyő előállítására is.

4. ábra



## Lapos képcsővek

Ezek a kijelzők működési elvüket tekintve nem különböznek a hagyományos monitorokban alkalmazott képcsővektől. Fizikai megjelenésükben jóval laposabbak azoknál. A laposságot az biztosítja, hogy több katódot alkalmaznak. Felépítésüket a 4. ábra szemlélteti.

Egy lapos, légüres térrel rendelkező „üvegdát” hátfoldalán helyezkednek el a katódként szolgáló fémzálak. Ezek elé egy furattal ellátott maszkot helyezve, pontszerű elektronforrásokat nyerhetünk. Az elektronsugár intenzitását a maszk elé helyezett vezérlő elektródafelülettel szabályozhatjuk. A képcsőnek vízszintes és függőleges vezérlő elektródái is vannak, melyek meghatározzák, hogy a foszforbevonatú képernyő melyik képpontját kell kivilágítani. Így az egyes elektronsugarak csak a számukra kijelölt képernyőfelületet érhetik el. A fény így továbbra is a hagyományos foszforbevonatú ernyőn jelenik meg, az elektronok futási útja viszont jelentősen megrövidült, azaz a képernyő laposabbá vált.

Nagy előnye ennek a technikának, hogy a már ismert, hagyományos módon lehet színes képet előállítani. Hátránya a többi típussal szemben, hogy még mindig nem sikerült eléggé lapossá tenni a képcsövet. Például a PANASONIC által így módon előállított képcső 9 cm mély. További hátrány, hogy a képcsővek viszonylag súlyosak, nagy az energiaigényük, és ütésre ugyanúgy érzékenyek, mint régebbi társaik.

SZEBENSZKI SÁNDOR

# A MINŐSÉGÜGY KÖZÜGY

## A végtérmeke minősége mint más jellemzők függvénye

Egyelőre tekintünk el attól, hogy a minőséghez egzaktusra törekvően meghatározzuk! (Ezt természetesen csak időlegesen tehetjük meg, hiszen például minden konkrét gyakorlati feladat tényleges megoldásában elkerülhetetlen az ezgakt definiálás. Elgondolkodtat, hogy a gyakorlat a minőséget sokkal hamarabb és sokkal többször igyekszik pontosan mérhetően definiálni, mint ahogyan azt az elmélet teszi.)

Feltételezzük továbbá még azt is, hogy a minőséget mérni is tudjuk, és e mérés olyan, mintha eredménye egyetlen egyszerű egyenletes, monoton skálán való leolvással lenne megállapítható; ugyanúgy, ahogy például tömeget mérünk.

Szedjük össze azokat a jellemzőket, amelyek a minőséget befolyásolják! E jellemzőket rögzített sorrendben egy vektorba fogva össze, a lehetséges vektorok halmazán egy minőség értékű (valós) függvényt értelmezhetünk. Természetes igény e függvény alakulásának, különösen szélső értékeinek megismerése. E feltérképező munka kielégítő elvégzése azonban általában nagyon nehéz, mert e függvények bonyolultak szoktak lenni. Mindig vannak azonban szerencsések. Sajnos általában kevesen. Keveseknek jut ugyanis az a szerencse, hogy a minőségfüggvény szélsőérték-meghatározási kísérletük kudarcba fullad. A többség ugyanis minden eszközzel kikényszeríti a sorstól a saját és vállalata balszerencsését a szélső értékek kiszámítására — és „gyakorlati hasznosítására” — révén.

Nem kell túl sok gyakorlati tapasztalat ahhoz, hogy ezt beállítsuk. A minőséget ugyanis, mint más jellemzők függvényét vizsgálni szinte mindig egyetlen-értékű egy katasztrófális félre-  
csúszással, azzal, hogy igyekszünk a legjobb minőséget megközelíteni. Pedig a minőség csak egy a jellemzők között, és közvetve vagy közvetlenül, szerepeltetni illik ugyan minden helyesen megállapított célfüggvényben (lásd később), de a minőséget a célfüggvény-nyelvi azonosítani szinte mindig durva hiba.

És mégis csinálják. Az ilyen tevékenység káros volta csak akkor enyhül egy kicsit, ha a célfüggvénybe nem egy, hanem több lényeges minőségjellemzőt is bevesznek.

A helyes megoldás azonban csak úgy közelíthető meg, hogy minden lényeges jellem-

ző szerephez jut a célfüggvényben. Hogy e jellemzők megfelelő szempontból mind minőségjellemzőknek foghatók fel, az egészen más kérdés. Nem volna semmi hiba, ha a célfüggvény a terméket más termékekkel együtt előállító gazdasági egység gazdálkodási minőségét mérné. Az ilyen felgöngyölítés — ez volna az igazi — azonban ma még erőltetettnek hatna, ezért kezeltük a minőségjellemzőket a „többi” jellemzőtől elkülönítetten.

Térjünk vissza a fogalmak tisztázására! A *célfüggvény* lényegében — röviden — olyan függvény, amely a választási lehetőségek közötti értékesség, fontosság (minőség!) szerinti megkülönböztetés elvégzését teszi lehetővé. A lehetőségek halmazán értelmezett, a lehetőségek értékét, fontosságát (minőségét!) mutató függvény.

A *minőség* pedig valamilyen meghatározott igényrendszer (kiválasztásrendszer, elvárásrendszer) valami általi kielégítésének mértéke. A minőség jellemzője arról informál, hogy valami valahogy milyen mértékben illeszkedik. Minden minőségjellemző (minden minőség mérőszám), tehát szükségképp valamilyen approximáció jószágmérője — más nem lehet. Hogy sokféle követelményrendszer, sokféle ezt approximáló megvalósítás és sokféle jószágmérés lehetséges, az természetes. Ebből következőleg az is természetes, hogy e sokaságban megbízható elméleti eszközök nélkül könnyű eltévedni.

Hamis képet festenek, ha azt állítanánk, hogy birtokunkban van legalábbis kielégítően jó módszer, vagy legalábbis nagyjából jól működő iránytű — hacsak a józan kockázatvállalást és az okos óvatosságot nem tartjuk annak. Anvak tartjuk, de hát ezeket nehéz lenne specifikus, tudományos eszközökkel elfogadni. Talán nem is azok — többek annál.

Most, hogy a klasszikus értelemben vett minőséget — a minőség érdekében — eltávolítottuk a középpontból és egyrészt mint egy (vagy több) minőségjellemzőt bevontuk a célfüggvényben szereplő többi jellemző közé, másrészt pedig általánosítva a minőség fogalmát, minden jellemzőt minőségjellemzőként fogtuk fel, azzal a kérdéssel találjuk magunkat szemben, hogy hogyan határozzuk meg rendszerek viselkedésének minőségjellemzőit.

Erre a kérdésre — minden gyakorlati esetben alkalmazható — ezgakt választ adni túlságosan merész vállalkozás volna. Konkrét válaszunkhoz pedig alapos helyzetfelmérés szükséges. Megállíthatjuk a tudományt? Többre nem futja, mint hibás előítéletek lerombolására? Részben ez a helyzet. Nem titkolhatjuk, hogy a minőséggel kapcsolatos bajok okai nemcsak az emberi *tréhanyságban és felelőtlenségben* vannak. A minőség-problematika megoldása nemcsak szigor és nemcsak morál kérdése. A minőséggel kapcsolatos mai válság más okai a *minőségfogalom és a minőségelmélet válságára* mutatnak. Baj van a minőségelmélet és alkalmazása minőségével is. A minőségelmélet jelenleg rendelkezésre álló módszerei gyártmány- és szolgáltatásminőséggel, ezek irányításával kapcsolatos nagyon egyszerű módszerek. Ezek pedig nem alkalmasak a jelenlegi nagy bonyolultságú és jellegében is más feladatok kezelésére és megoldására. Az, hogy ugyanazt a „minőség” szót használjuk ott is és itt is, csak megtévesztő — sokan be is dölnék.

Néhány hasznos, konkrét munkára felkészítő gyakorlati tanácsal fejezzük be mostani elmélkedésünket.

Hogyan fogjunk hozzá a munkához a gyakorlati ember a „terepen”, ha például kinevezik vállalati minőségőrnek vagy minőségfelelősnek, netán egy minőségügyi osztály együttetésébe vagy élére kerülve. Játssza a lehetőségekkel, legyen szó például szoftverminőségről!

Mérje fel először a „terepet”! Állapítsa meg, hogy a jelenlegi termelési, szolgáltatási rendszerben milyen erők, milyen ösztönzések működnek, milyen irányban, és ezek milyen folyamatokat hoznak létre. Ha elér arra a szintre, hogy már nem csodálkozik azon, hogy a minőség miért olyan amilyen, elmondhatja, hogy az első lépést sikerrel tette meg. Ha még egy gráfit is fel tud rajzolni, amelyben a hatáskapcsolatokat legalábbis hozzávetőlegesen szemléltetni képes, bizhat benne, hogy közben tudja majd tartani a további lépések irányítását is.

Ezek után következhet egy nehezebb feladat. Szerkeszzen célfüggvényeket, amelyek a jelenlegi viszonyokra vonatkoz-

nak! Ha most ez folyik, mire is törekszik most a rendszer? Mit tart jónak? Mit mivel szemben minősít saját számára értékebbnek? Mi a fontos számára? Mi minél fontosabb a számára? Ha e feladat megoldásában nem is jut el kvantitatív eredményekig, a jó kvalitatív felmérések is rengeteg hasznú lehet.

Kellő jártasság után érdemes hozzáfogni olyan célfüggvények megszerkesztéséhez, amelyeket ideális esetben használna. Különböző gazdasági, termelői, szolgáltatói stb. egységeknek, funkcióknak természetesen külön-külön kell célfüggvényt szerkeszteni, hiszen az egész szervezet irányításában több „kisebb” irányítási tevékenység is szerepet játszik.

Az ilyen gyakorlatokat végző azonban nagy veszélyben van. Észre sem veszi, hogy valamiféle ideális eset, ideális állapot felé igyekszik terelni a befolyási körébe esőket, attól a veszedelmes és legtöbbször nagy kárt okozó előítéletől vezetve, hogy az a legjobb, ha a *legjobb*at igyekszik megközelíteni. Ez a viselkedéspolitikai azonban végtelen hibás (is) lehet. Ha például már elkövetünk egy hibát, akkor nem mindig a hiba kijavítása a helyes politika. Ha valaki például rossz irányba indult el, és bizonyos idő múlva észrevette tévedését, nem biztos, hogy az lesz a legjobb, ha visszamegy oda, ahonnan elindult, és most másodszorra a helyes irányba indulva, megkezdí célja elérésére irányuló műveleteit. Ha közben a cél is változhat, az újrakezdés, hibakijavító politika helytelensége kiütöközik.

Nem helyes tehát az a felgöngyölítés, amely általános ideális állapot elérésére törekszik, konkrét ideális viselkedés megvalósítása helyett. A minőségügy, különösen a mai gyorsan változó világban meg kell hogy szabaduljon a fél évszázaddal ezelőtti *állapotközpontúságtól*, amikor beállították és minél hosszabb ideig igyekeztek fenntartani egy jó állapotot. Ez akkor elég volt. Ma gyorsan változó körülmények között ez nemcsak lehetetlen, hanem *hibás* is volna. *Ma nem a létrehozás és fenntartás műveletei dominálnak, hanem a navigáció.* Ma navigálni kell.

És nem a termék és szolgáltatás minőségén múlik a siker, hanem a navigáció minőségén, amelybe természetesen a termék, szolgáltatás minősége is beleszól.

## BASIC és gépi kód

Legutóbb a saját gépi kódú rutinok elhelyezésének lehetőségeiről és a gépi kódú eljárások aktiválásának módjairól volt szó. Szerepelt egy kis feladat is: megfejtetni, hogy mit csinál a példaprogramban szereplő gépi kódú rutin. A megoldás: megszámlálja az a regiszter tartalmának 1-es értékű biteit, és a darabszámot kiírja a képernyőre.

Érdekes azon is gondolkodni, hogy hogyan oldható meg ez a feladat BASIC-ben?

Most a mikroprocesszor regisztereivel és néhány vezérlés-átadó gépi utasítással ismerkedünk meg. Szükség lesz az áprilisi szám 23–26. oldalain található „A 65XX mikroprocesszor gépi és assembler szintű utasításkészlete” című táblázatára, melyre a továbbiakban „segédlet” néven fogok hivatkozni.

### A regiszterek

A segédletben a Jelölések cím alatt találjuk a 65XX regisztereinek rövid leírását. Cél szerű a táblázatban a regiszterek leírása és az SR biteinek leírása közé egy függőleges elválasztó vonalat húzni, hogy a két egymás mellett táblázat el különüljön egymástól.

Mikroprocesszorunknak 6 regisztere van. A memória egy bájtna nem regiszter, csak a jelölés egységessége miatt került a segédletben a regiszterek közé. A regiszterek általában 8 bitesek, kivétel a 16 bites utasításszámláló és a 9 bites veremutató. Ez utóbbi nyolcbitesnek is tekinthető, hiszen a 9., a legmagasabb helyiértékű bit mindig 1-es.

Lássuk egyenként a regisztereket!

### Az utasításszámláló

A program futása közben általában a következő végrehajtandó utasítás címét tartalmaz-

za. Ez persze nem pontosan így igaz, működése sokkal bonyolultabb, de ebbe most korai lenne belemerülni. Azért 16 bites, hogy a 64 kilobájtos memória bármely címén levő utasítás elérhető legyen. Tartalma minden gépi utasítás végrehajtása közben változik, a vezérlés-átadó utasítások tovább módosítják.

Angol nevét (program counter) PC-nek rövidítik. A továbbiakban mi is ezt a jelölést használjuk.

### Az akkumulátor

A regiszternek is nevezik, általános célú regiszter. Az aritmetikai és logikai műveletek végrehajtása ebben a regiszterben történik. Bár a memóriamezők közötti adatátvitel az indexregisztereken keresztül is történhet, lehetőségei a különböző címzési módok révén, az A regiszteren keresztül a legnagyobbak. A processzor veremmemóriájához is csak ezen a regiszteren keresztül lehet közvetlenül hozzáférni.

### Az indexregiszterek

Két indexregiszter van, az X és az Y. Az egyszerű indexelt címzési módoknál és a memóriabájtok közötti adatátvitelnél a szerepük azonos, de van néhány speciális feladat, melyekre csak az egyik indexregiszter alkalmas.

A veremmutató tartalmához való hozzáférésre és annak megváltoztatására csak az X regiszteren keresztül van lehetőség. Az úgynevezett indexelt indirekt címzési mód is csak ezzel a regiszterrel használható. (Ezzel a címzési móddal egyébként bemutatott példánkban kívül eddig még nem találkoztam.)

A másik hasonló nevű az indirekt indexelt címzési mód. Ez az előbbivel ellentétben nagyon gyakran fordul elő, és csak az Y regiszterrel együtt használható.

### A veremmutató

Az általunk használt mikroprocesszorok a memória \$0100 és \$01FF közötti területét veremmemóriaként használják. Erről később majd részletesebben írok. Most csak annyit, hogy a veremkezelő utasítások segítségével itt lehet bizonyos információkat átmenetileg tárolni. A veremmutató mindig a következő beírandó információ címét tartalmazza. Mivel az említett terület csak 9 bittel címezhető, a veremmutató regiszter 9 bites, de a legmagasabb helyiértékű bit — megváltoztathatatlanul — mindig 1-es értékű.

A gép bekapcsolása utáni első műveletek egyike a veremmutató inicializálása. Ez a verem legmagasabb címére való beállítását jelenti.

Az angol elnevezés (stack pointer) rövidítésével a veremmutatót SP-vel szokás jelölni.

### Az állapotregiszter

Angol nevének rövidítése (processor status register) nyomán kétféle jelölés is használatos: P és SR. Bitei, az úgynevezett jelzőbitek a segédlet SR BITJEI táblázatában találhatók. Az egyes bitek szerepéről majd az utasítások megismerésekor lesz szó, most csak kettőt említek.

Az 5. (nem használt) bitet a processzor minden gépi utasításnál 1-re állítja. Ennek nincs jelentősége, de az állapotregiszter tartalmának kiolvasásakor nem árt tudni róla.

A 3. (D) bit a decimális üzemmódot jelzi. A Commodore gépeken nem használják gépi szinten a decimális aritmetikát, ezért a bekapcsolás utáni első utasítások egyike a decimális mód letiltása. (Remélem, hogy e sorozat keretében sikerül a decimális aritmetika használatával megismernünk.)

A BASIC SYS utasításának használatakor a P regiszterhez

hozzá tudunk férni. Erre a következő alkalommal egy gyakorlati példát is láthatunk.

### Vezérlés-átadó utasítások

Most néhány feltétel nélküli vezérlés-átadó utasítást ismerünk meg, a következő alkalommal pedig már használni is fogjuk őket.

Közös tulajdonságaik: — módosítják az utasításszámláló tartalmát, — az állapotregiszter biteit nem változtatják, — működésük összehasonlítható egy-egy BASIC utasításával.

### A feltétel nélküli ugrás

A JMP utasítás két változata közül most csak az abszolút címzési móddal foglalkozunk. Gépi kódja: \$4C.

A BASIC GOTO utasításhoz hasonlítható, de itt utasításszám helyett egy gépi cím az operandus. Az operandus kétbájtos, az ugrási cím bájttjai a már leírt módon alacsony helyiérték — magas helyiérték sorrendben követik egymást.

### Ugrás szubrutinra

A JSR utasítás a GOSUB BASIC utasításhoz hasonlítható. Csak abszolút címzésű változata van. Gépi kódja: \$20.

A JMP utasításhoz hasonlóan működik, de a szubrutinra ugrás előtt a verembe teszi a szubrutinból való visszatéréshez szükséges címet. A JSR utasítás kétbájtos operandusa a szubrutin címe.

### Visszatérés szubrutinból

Az RTS utasításnak nincs operandusa. A RETURN BASIC utasításnak feleltethető meg. A veremből kivезi a JSR által ott elhelyezett visszatérési címet, és az utasításszámláló regiszterbe teszi. A program futása a JSR utáni utasításon folytatódik. Gépi kódja: \$60.

BARNA LÁSZLÓ

## Z80 programozási gyakorlatok

A sorozat mindazoknak szól, akik valamilyen könyvből már megtanulták, vagy éppen most tanulják a Z80 CPU programozását. Különböző gépfüggetlen példákon keresztül mutat be programozási fogásokat és ad az olvasó kezébe olyan kész szubrutinokat, amelyekkel bátran használhat saját programjai részeként. Azt szeretnék, hogy ha valaki a sorozatot végigolvassa, akkor bele tudjon vágni komolyabb problémákba, de tudja azt is, hogy nem kell mindent gépi kódban vagy assembly nyelven megírni.

Nézünk először, hogy mi is az assembly és a gépi kód között a különbség! A gépi kód a gép számára közvetlenül érthető, az assemblyt le kell fordítani gépi kódra. Általában gépi kód programozásnak hívják azt, amikor az assembly programot kódtáblázat segítségével, kézzel fordítják. Fáradsztó, sok hibalehetőséget magában rejtő munka. Egyesek képesek meglepően nagy programot így megírni, de erre az ember többnyire csak kényszerhelyzetben vállalkozik. Erre a munkára van ugyanis (ha van) egy segítőtárs: az assembler fordítóprogram, vagy röviden: assembler. Ez olyan program, amely a gépbe assembly nyelven beírt programot szintaktikusan ellenőrizi, és ha hibátlanok találja, akkor lefordítja és betölti a tárbá.

Az assembler számára meg kell azt is mondani, hogy milyen módon fordítsa le a programot: milyen címtől kezdje a fordítást, hol hagyjon a program közben szabad helyet a tárban és mennyit stb. Ezekre pszeudo utasítások vagy más néven direktívák szolgálnak, melyek assemblerként különbözőek, nagyon sokféle lehetnek, ezért mindig utána kell nézni az általunk használt assembler leírásában. A sorokat megjelölhetjük címekkel, amelyekhez a fordítás során a fordító azokat az értékeket rendeli, ahova az adott sorból generált kódok az első bájtra kerül. Ez a használataban azt jelenti, hogy egy sor elé címkét írva, egy, a sorra hivatkozó CALL, JP vagy JR vagy akár egy LD utasítás után is elég leírni a címkét a szám helyett.

Lehetőség van arra is, hogy egy címkehez átlank megadott értéket rendeljünk. Erre legutóbb az EQU direktíva szolgál. Így például a CR EQU 13 utasítással a CR címkehez (carriage return = kocsi vissza) 13-at rendelve, a program további részeiben mindenhol CR-t írhatunk, ahova 13-at írunk. (Természetesen 136 helyett nem írhatunk CR(06-t) Célserű azonban csak olyan helyre írni ezt a címkét, ahol a 13 mint sorvégiel szerepel, egyébként, bár a program helyesen fordítódik le, nehezen olvasható, sőt félrevezető. Erdemes mindig olyan címkét használni, ami emlékeztet a jelentésre.

A program későbbi olvashatóságát segítik a program szövegeiben elhelyezett megjegyzések is, melyek a legtöbb assemblernél a program sor pontosvesszővel lezárultak végén vannak. Ezeket a fordító figyelmen kívül hagyja. Rendkívül fontosak, mert az assembly programok sokkal nehezebben olvashatók, mint a BASIC vagy PASCAL nyelven írtak.

Ugyancsak ezért nagyon fontos a programok dokumentálása. Assembly programozásnál a regiszterek nevei töröltek, és így nem utálnak arra, hogy mit tárolunk bennük. A regiszterek felhasználhatósága azonban bizonyos mértékig megszabja, hogy mire melyik regisztert használjuk. Így például a HL regiszterpárt arra szoktak használni, hogy memóriahelyekre mutasson, a B regisztert ciklusszámlálóknak, az A regisztert, azaz akkumulátort aritmetikai és logikai műveletekre és így tovább. Elég nehéz szabályt kimondani. Általában: egy-egy feladatra mindig azt a regisztert kell használni, amellyel a legrövidebben, legegyszerűbben megoldható a probléma. Eleinte elég nehéz a választás, később azonban, bizonyos gyakorlatot szerzve, természetessé válik. Persze ekkor is megtörténhet, hogy egy rutin számára kényelmesebb, ha egy előzőleg megírt rutint átirunk úgy, hogy más regisztereket használunk, de ezt mindig jelezni kell a dokumentációban, illetve új dokumentációt kell készíteni. Egy-egy rutin dokumentációjának négy dolgot mindenképpen tartalmaznia kell:

1. Mit csinál a rutin.
2. A rutin mely regisztereket és memóriahelyeket használ fel bemenetként.
3. Mely regiszterekben, illetve memóriahelyeken adja az eredményt.
4. Mely egyéb regisztereket változtat meg.

Minden rutint meg lehet írni úgy, hogy azon regiszterek kivételével, amelyekben az eredményt adja, ne írjon át egy más regisztert sem, azaz hogy visszatérskor a regiszterek értékei megegyezzenek a híváskor tartalmozottakkal. Erre többnyire verem- (stack-) kezelő utasításokat szoktak használni. Ha tehát valamely szubrutinban használjuk a B és C regisztert, de meg akarjuk őrizni az értékeit a hívó program számára, akkor mielőtt felhasználnánk őket, egy PUSH BC utasítást kell végrehajtatni a géppel, majd a visszatérés előtt egy POP BC utasítást. Akkor is ki kell venni a veremből az értéket, ha rutin közben dől el, hogy még sincs szükség rá. Gyakori hibaforrás, hogy bent maradjon valami a veremben, például a PUSH BC-nem követi sehol a program folyamán POP BC, és a RET utasítás így a BC verembe került értéket tölti a programszámlálókba, azaz ott folytatódik a futás, ez pedig többnyire elkerülendő. (Persze van kivétel is, a Spectrum gép BASIC értelmező programja ilyen módon hajtja végre az USR utasítást.)

Fontos tudni, hogy mely utasítások milyen jelzőbitet (flaget) állítanak. Sokszor használják az akkumulátor törlésére az XOR A utasítást, ez azonban amellett, hogy nulla kerül az akkumulátorba, törli a carry (átvitel) bitet és egyre állítja a zéró jelzőbitet. Mivel a carry jelzőbit törlésére nincs közvetlen utasítás, csak az egymást követő SCF és CCF utasításokkal érhető el a carry törlése, ezért gyakran használják az AND A vagy OR A utasításokat a carry törlésére. Ez azonban nem mindig tehető meg, mert ezek az utasítások állítják a zéró jelzőbitet is A értékének megfelelően, ezért ugyanezen utasításokat szokták használni a CP, ót bájtot elfoglaló utasítás helyett is. A zéróbitet egyre lehet állítani a CP A utasítással.

Ezek az apró programozói fogások rövidebbé és gyorsabbá teszik a programot. Igaz ugyan, hogy csak egy-két bájtról és milliszekundumokról van szó, de sok kicsi sokra megy. Konkrétan: A HT-1080Z számítógép ROM program eredetileg 8080 processzorra készült, így sok Z80 utasítást nem használhat fel a programozók. Nincs a ROM-ban mégsem olyan rutin, amelyet lényegesen rövidebben és gyorsabban meg lehetne írni, mégis egy, a teljes Z80 utasításkészletet használó BASIC értelmező mérhetően gyorsabban fut. Sokat lehet tanulni a profi programozók munkáiból. Kapható leírás, igaz nem kis pénzért, a Spectrum és a HT-1080Z számítógépek fixtáraról.

Nézünk meg bevezetőnek három rövid példát.

1. Írjunk programot, amely HL címtől keres egy nem szököz karaktert, azt betölti az akkumulátorba, HL-t pedig ráállítja a következő karakterre!

Először betöltjük A-ba a karaktert, mert egyszerű, ha szököz, csak így tudjuk eldönteni, hogy valóban az-e, ha pedig nem, akkor egyébként is oda kell tenni. Növeljük HL-t, mert ha az akkumulátorban a szököz kódja van, akkor úgyis a következő karaktert kell venni, ha pedig nem, akkor így HL a nem szököz karakter utánra mutat. Ezután eldöntjük, hogy az akkumulátorban szököz van-e, vagy sem. Ha az van, akkor a hasonló utasítás hatására zéró bit=1, nem tér vissza a program, hanem veszi a következő karaktert. Ha szököz van az akkumulátorban, akkor visszatér az őt hívó helyre (1. program).

Gyakorlásképpen próbáljuk meg kézzel, kódtábla segítségével lefordítani a programot. Vajon mindig leáll-e a ciklus?

2. Írjunk szubrutint, amely kiszámítja az akkumulátorban ASCII kóddal adott hexadecimá-

lis számjegy értékét! Ha nem hexadecimális számjegy kódja van A-ban, akkor visszaérkezőskor carry=1, különben carry=0 legyen!

Ha „0” kódja és „9” kódja közé esik a számjegy, akkor „0” kódtól kell levonni, ha „A” kódja és „F” kódja közé esik, akkor „A” kódja minusz 10-et kell levonni ahhoz, hogy megkapjuk a számjegy értékét. Ha a kód nem tartozik egyik tartományba sem, akkor hibát kell jelezni (2. program).

A kódból először levonjuk a „0” kódtól. Ha ezután carry=1, akkor az érték kisebb volt, mint „0” kódja, visszatérés, és carry jelzi a hibát. Ha a már kisebbített érték kisebb mint 10, azaz az eredeti érték nem volt nagyobb mint „9”, akkor a 10-zel való hasonlítás után carry=1 lesz, és ebben az esetben nem is kell tovább alakitunkunk a számot, de mivel a hiba jelzése éppen a carry jelzőbittel történik, ezért az ellentettjére állítjuk. Ezután ha a szám nagyobb volt, mint „9” kódja, akkor levonunk belőle 7=A-10-„9”-t. Így már megkaptuk a végső értéket, csupán azt kell eldönteni és a carry bitet aszerint beállítani, hogy az érték helyes-e (lásd az 1. ábrát). Nem lehet az érték kisebb 10-nél és nagyobb 15-nél, hiszen a kód, ha helyes, „A” és „F” közé esik. Ha „0” és „9” közé esett, akkor a RET NC utasítással megtörtént a visszatérés. Ha tehát a szám kisebb mint 10, a hasonlító utasítás után carry=1, ha pedig nagyobb 15-nél, akkor a 16-tal való hasonlítás után carry=0, amit az ellentettjére váltva, megkapjuk a hibajelést.

Nézünk meg most ugyanezt a szubrutint egy másik assembly listán. Nem használunk megjegyzéseket, és nem használjuk az ASCII kódkarakterrel megadható alakját (3. program).

Az így leírt program sokkal nehezebben olvasható. Az assembly nyelven megírt programnak soha nem szabad úgy kinéznie, mintha egy gépi kódról visszafordított lista lenne.

3. Írjunk szubrutint, amely összehasonlítja HL-t és DE-t! Az eredményt carry és zéróbitet jelezzék!

	CARRY	ZÉRO
HL < DE	1	0
HL = DE	0	1
HL > DE	0	0

1. ábra
2. ábra

Karakter	Kód	„0”-t levonva	„A”-10-„0”-t levonva
0	48	0	
1	49	1	
2	50	2	
3	51	3	
4	52	4	
5	53	5	
6	54	6	
7	55	7	
8	56	8	
9	57	9	
A	65	17	10
B	66	18	11
C	67	19	12
D	68	20	13
E	69	21	14
F	70	22	15

# Disassembler program Sinclair számítógépekre



A példa második mondatának értelme csupán az, hogy a carry és zéróbiték állapotából meghatározható legyen HL és DE viszonya. Lehet tehát, hogy zéró=1 azt jelentse: HL<DE. Ez azonban célszerűtlen, a szubrutint használó programot nehezen olvashatóvá teszi, sőt félrevezető is lehet, és a szubrutint is nehéz lenne így megírni, hosszú is, lassú is lenne. A legtermészetesebb az, ha az információt hasonlóképpen hordozzák a jelzőbiték, mint az egy bajtot használt utasítások után (lásd a 2. ábrát és a 4. programot).

A szubrutin először csak a számok felső bajtját hasonlítja össze. Ha ezek nem egyenlők, akkor az összehasonlítás eredményeképpen a carry bit E és L regiszterek értékétől függetlenül hordozza a megfelelő információt. Az alsó bajtokat csak akkor kell összehasonlítani, ha a felső bajtok egyenlők.

Próbáljuk végig a program futását konkrét értékekkel! Mely regisztereket használja fel a rutin? Meg lehet-e őrizni A regiszter értékét PUSH AF és POP AF utasításokkal? Hogyan lehet megőrizni A regiszter értékét? Nézzük meg ennek a problémának egy másik megoldását! (5. program)

Az előző program hossza 6 bajt, ez csupán öt. Az felülírja az akkumulátort, ez nem. Az előzőnek azonban van egy nagy előnye, hogy lényegesen gyorsabb, még abban az esetben is, ha a felső bajtók egyenlők. Hogy melyiket válasszuk, csak a konkrét helyzetben dönthetjük el, ha tudjuk, hogy mi a fontosabb: a sebesség vagy kis méret.

VERHÁS PÉTER

## 1. program

```
GTCHR LD A, (HL)
INCL HL
CP " " ;szóköz kódja
RET NZ
JR GTCHR
```

## 2. program

```
HEXA SUB "0" ;"0" kódja
RET C ;kisebb volt, mint "0"
CP "9" - 48 + 1 ;felső határ "9"
CCF
RET NC ;nem volt nagyobb, mint "9"
SUB "A" - 10 - "10"
CP 10
RET C ;kisebb volt, mint "A"
CP 16 ;felső határ "F" ellenőrzése
CCF
RET
```

## 3. program

```
SUB 48
RET C
CP 10
CCF
RET NC
SUB 7
CP 10
RET C
CP 10H
CCF
RET
```

## 4. program

```
COMP LD A,H ;felső helyiérték
CP D ;hasonlítás
RET NZ ;nem egyenlő/készen is van
LD A,L ;alsó helyiérték
CPE ;hasonlítás
RET
```

## 5. program

```
COMP PUSH HL ;elmentés a kivonás miatt
OR A ;carry törlése
SBC HL,DE ;beállítjuk a jelzőbitéket
POP HL ;vissza az eredeti érték
RET
```

1982 nyarán kezdtem használni egy ZX81 számítógépet, ami akkor egészen új eszköz volt, és kezdetben teljesen elbűvölt. Néhány hónapig boldogan programoztam BASIC-ben, de egy idő múlva fúrni kezdte az oldalamat a kíváncsiság: hogyan is működik a számítógépem igazából, mi van a rejtélyes ROM-ban, amiről a Sinclair-kézikönyv nem túl sokat árul el.

Beszereztem tehát a megfelelő irodalmat a Z80 mikroprocesszor gépi kódú programozásáról, és megpróbáltam egyszerű programrészeket írni gépi kódban. Természetesen ez rendkívül fáradságos munka volt, mivel semmi szoftver nem segítette próbálkozásaimat. A legkisebb hiba esetén összeomlott a programom, és kezdetem előlrol az egészet. Rengeteg időm ráment, látható eredmény nélkül.

Végül megértettem, hogy nem jutok előbbre valamilyen szoftvertömogatás nélkül. Mivel abban az időben nem lehetett beszerezni ilyet, elhatároztam, hogy magam készítek egy olyan BASIC-ben írt programot, amely gépi kódú rutinokat fordít vissza assembly nyelvre, vagyis egy disassemblert.

Hogy miért éppen disassemblert és nem assemblert? Hát ehhez kitalálhatnék valamilyen körmönfönt magyarázatot, de egyszerűbb, ha bevallom, hogy azért döntöttem így, mert ennek elkészítése sokkal egyszerűbb.

Nos, a disassembler elkészült, és sikeresen használtam hosszú ideig. A ZX81-en kicsit lassan működött, de sebjai! Amikor szereztem egy Spectrumot, azonnal átírtam arra is. Meglepetésemre a program a Spectrumon lényegesen gyorsabban működött!

Idővel a Sinclair gépek szoftverellátottsága gyökeresen megjavult. Ma tucatnyi jobbnál jobb assemblert és monitorprogramot lehet kapni, ezek nálunk is közismertek. E programok többségét gyors gépi kódban írták meg: az én egyszerű készítményem nem versenyezhet velük. Ma már én sem ezt használom programfejlesztéshez. Kitaláltam azonban egy olyan alkalmazást, amely ma is időszerűvé teszi. Nevezetesen megtanítottam a disassemblert arra, hogy felismerje a Spectrum ROM specialitásait: a táblázatokat, karaktergenerátort, üzeneteket és végül az RST 08 és RST 28H utasítások egyedi szintaktikáját.

Ian Logan magyarul „A Sinclair ZX-Spectrum ROM programja” címmel megjelent kitűnő könyve részletesen leírja a lebegőpontos aritmetika érdekes megvalósítását a Spectrumban. A kalkulátor rutinjai az RST 28H utasítást követő címeké hívják. Ezek úgy foghatók fel, mint egy programozható zsebszámológép egyszerű utasításai: tulajdonképpen egy egyszerűbb nyelv a bonyolultabb nyelven (BASIC) belül. Nos, a program most az RST 28H-t a Spectrumban használt címekkel fordítja vissza.

A közreadott két program közül az egyik a program eredeti, ZX81-re készült változata (1. lista). Ez mindkét gépen működik. Betöltése után mindkét gépen szükség van az adatok bevitelére is; ezek a mnemonikok, címek és egyebek. A program gyors működése érdekében ezeket az adatokat nem DATA-READ típusú programrészletekben, hanem tömbváltozóknak tároljuk. Felöltésük a 6000-tól kezdődő sorokkal történik.

Először minden sorból töröljük ki a védelem céljából beírt REM kulcsszókat, majd GOTO 6000-rel indítsuk el a programot. Ekkor a program kérni fogja a megfelelő adatokat, amelyeket a 2., 3., 4. és 5. lista szerint gondosan gépeljünk be. Ha valamilyik adatlistával végeztünk, célszerű megszakítani a program futását, visszairni az 1. lista szerint a lefutott programrészebe a REM-eket, majd GOTO-val újraindítva visszairni az éppen bevitt listát. Ha hibát találunk, azt egyszerűen parancs üzemmódban javítsuk.

Fontos, hogy a programlistákon ne változtassunk, mivel a program kiszámított GOSUB utasításokkal működik. Ha tehát a sorszámok nem az 1. lista szerintiék, a program hibásan fog működni.

Ha az összes adatot hibátlanul bevittük, a 6000. sorszám feletti sorokat ki is törölhetjük. Vigyáznunk kell azonban, hogy CLEAR vagy RUN parancsot soha ne adjunk ki ezután, mivel ezek a változatokat és ezzel a bevitt adatokat is törlik: programunk nem fog működni.

Most programunk egyszerűbb változata működőképes. A ZX81-tulajdonosoknak ennnyivel be kell érniük. A Spectrumosok, miután kazettára vagy microdrive-cartidge-ra mentették a programot, begépelhetik a 6. listán levő másik programot, majd

1. lista

```

100 PRINT AT 4,10,"DISASSEMBLER
210 PRINT TAB 10; "*****"
220 PRINT AT 0,0;"MINDEN SZAM O
CEGINALE"
230 PRINT AT 11,1;"[ ] HASZNAL
JON BURUJON:"
240 PRINT AT 13,10;"PARANCSTI"
250 PRINT AT 15,12;"BARAK-KEL LE
HE HELLYEN"
260 PRINT AT 21,0;"HELVIK CIMTO
370 INPUT J
380 C=0
390 LET I=0
340 LET K=1/PEEK J
350 GO SUB (1000+2*VAL BS(K))
360 IF I=255 THEN GO TO 300
410 GO TO 250
420 IF BS(K)=0
430 IF BS(K)=2 OR BS(K)=Z THEN GO
TO 450
440 IF BS(K)=Z THEN GO TO 440
470 RETURN
600 LET J=J+1
610 LET I=PEEK J
620 RETURN
700 GO SUB 600
710 IF I=255 THEN LET A=R
720 IF I=Z THEN LET A=R-256
840 RETURN
900 GO SUB 600
910 LET I=I+1
920 LET C=R+I*2
930 LET C=R+I*2+I*(C+1)
940 RETURN
1000 GO SUB 1300
1010 RETURN
1020 GO SUB 1800
1030 RETURN
1040 GO SUB 1700
1050 RETURN
1060 GO SUB 1700
1070 RETURN
1080 GO SUB 1800
1090 RETURN
1100 GO SUB 1900
1110 RETURN
1120 GO SUB 2100
1130 RETURN
1140 GO SUB 2500
1150 RETURN
1160 GO SUB 2300
1170 RETURN
1180 RETURN
1190 J=J+1
1200 PRINT J,TAB 7;A(K)
1210 RETURN
1220 LET X=VAL X(K)
1230 IF X=1 THEN LET X=10
1240 PRINT J,TAB 7;A(K),TO X:
1250 GO SUB 800
1260 PRINT A;A(K),X+1 TO 1
1270 RETURN
1280 LET X=VAL X(K)
1290 IF X=1 THEN LET X=10
1290 PRINT J,TAB 7;A(K),TO X:
1300 GO SUB 800
1310 LET A=
1320 GO SUB 900
1330 LET A=
1340 PRINT G;A(K),X+1 TO 1
1350 IF X=144 GO K+15 THEN PRIN
1760 PRINT
1770 RETURN
1780 LET X=VAL X(K)
1790 IF X=1 THEN LET X=10
1800 PRINT J,TAB 7;A(K),TO X:
1810 GO SUB 700
1820 LET A=R+Z
1830 PRINT A,TAB 25;Z+A
1840 RETURN
1850 PRINT J,TAB 7;
1860 LET A=R+Z
1870 LET A=RA+R*(A TO VAL P(AA)
1880 PRINT E;A(K) TO VAL P(AA)
1890 IF BS(K)=0
1900 RETURN
1910 LET A=I+X
1910 PRINT J,TAB 7;
1920 GO SUB 800
1930 LET H=CHS A
1940 GO TO 1170
1950 IF H=BS(K) THEN GO TO 21
1960 NEXT N
2160 NEXT N
2170 PRINT DEFB "A
2180 RETURN
2190 LET H=CODE R(N,2)
2200 LET Q=CODE R(N,3)
2210 GO SUB (3000+100*CODE R(N,
2220
2210 RETURN
2230 LET S=I+Y
2240 GO TO 2110
2250 PRINT I,TAB 7;
2260 GO SUB 800
2270 LET H=CHS A
2280 IF H=Z THEN LET S=
2290 IF H=BS(K) THEN GO TO 25
2300
2310 NEXT N
2320 PRINT DEFB "A
2330 RETURN
2340 LET H=CODE R(N,2)
2350 LET Q=CODE R(N,3)
2360 GO SUB (4000+100*CODE R(N,
2370
2380 NEXT N
2390 PRINT DEFB "A
2400 RETURN
2410 LET H=CODE R(N,2)
2420 LET Q=CODE R(N,3)
2430 GO SUB (4000+100*CODE R(N,
2440
2450 NEXT N
2460 PRINT DEFB "A
2470 RETURN
2480 LET H=CODE R(N,2)
2490 LET Q=CODE R(N,3)
2500 GO SUB (4000+100*CODE R(N,
2510
2520 NEXT N
2530 PRINT DEFB "A
2540 RETURN
2550 LET H=CODE R(N,2)
2560 LET Q=CODE R(N,3)
2570 GO SUB (4000+100*CODE R(N,
2580
2590 NEXT N
2600 PRINT DEFB "A
2610 RETURN
2620 LET H=CODE R(N,2)
2630 LET Q=CODE R(N,3)
2640 GO SUB (4000+100*CODE R(N,
2650
2660 NEXT N
2670 PRINT DEFB "A
2680 RETURN
2690 LET H=CODE R(N,2)
2700 LET Q=CODE R(N,3)
2710 GO SUB (4000+100*CODE R(N,
2720
2730 NEXT N
2740 PRINT DEFB "A
2750 RETURN
2760 LET H=CODE R(N,2)
2770 LET Q=CODE R(N,3)
2780 GO SUB (4000+100*CODE R(N,
2790
2800 NEXT N
2810 PRINT DEFB "A
2820 RETURN
2830 LET H=CODE R(N,2)
2840 LET Q=CODE R(N,3)
2850 GO SUB (4000+100*CODE R(N,
2860
2870 NEXT N
2880 PRINT DEFB "A
2890 RETURN
2900 LET H=CODE R(N,2)
2910 LET Q=CODE R(N,3)
2920 GO SUB (4000+100*CODE R(N,
2930
2940 NEXT N
2950 PRINT DEFB "A
2960 RETURN
2970 LET H=CODE R(N,2)
2980 LET Q=CODE R(N,3)
2990 GO SUB (4000+100*CODE R(N,
3000

```

2. lista

```

1 AS(I) X(I) B(I)
1 NOP 3 0
2 LD BC, 6 2
3 LD (BC),A 9 0
4 INC BC 6 0
5 INC B 5 0
6 DEC B 5 0
7 LD B, 5 1
8 RLCA 4 0
9 EX AF,AF' 9 0
10 ADD HL,BC 9 0
11 LD A,(BC) 9 0
12 DEC BC 6 0
13 INC C 5 0
14 DEC C 5 0
15 LD C, 5 1
16 RRC A 4 0
17 DJNZ 5 4
18 LD DE, 6 2
19 LD (DE),A 9 0
20 INC DE 6 0
21 INC D 5 0
22 DEC D 5 0
23 LD D, 5 1
24 RLA 3 0
25 JR 3 4
26 ADD HL,DE 9 0
27 LD A,(DE) 9 0
28 DEC DE 6 0
29 INC E 5 0
30 DEC E 5 0
31 LD E, 5 1
32 RRA 3 0
33 JR NZ, 6 4
34 LD HL, 6 2
35 LD ( ),HL 4 3
36 INC HL 6 0
37 INC H 5 0
38 DEC H 5 0
39 LD H, 5 1
40 DAA 3 0
41 JR Z, 5 4
42 ADD HL,HL 9 0
43 LD HL,( ) 7 3
44 DEC HL 6 0
45 INC L 5 0
46 DEC L 5 0
47 LD L, 5 1
48 CPL 3 0
49 JR NC, 6 4
50 LD SP, 6 2
51 LD ( ),A 4 2
52 INC SP 6 0
53 INC (HL) 8 0
54 DEC (HL) 8 0
55 LD (HL), 8 1
56 SCF 3 0
57 JR C, 5 4

```

```

58 ADD HL,SP 9 0
59 LD A,( ) 6 3
60 DEC SP 6 0
61 INC A 5 0
62 DEC A 5 0
63 LD A, 5 1
64 CCF 3 0
65 LD B,B 6 0
66 LD B,C 6 0
67 LD B,D 6 0
68 LD B,E 6 0
69 LD B,H 6 0
70 LD B,L 6 0
71 LD B,(HL) 9 0
72 LD B,A 6 0
73 LD C,B 6 0
74 LD C,C 6 0
75 LD C,D 6 0
76 LD C,E 6 0
77 LD C,H 6 0
78 LD C,L 6 0
79 LD C,(HL) 9 0
80 LD C,A 6 0
81 LD D,B 6 0
82 LD D,C 6 0
83 LD D,D 6 0
84 LD D,E 6 0
85 LD D,H 6 0
86 LD D,L 6 0
87 LD D,(HL) 9 0
88 LD D,A 6 0
89 LD E,B 6 0
90 LD E,C 6 0
91 LD E,D 6 0
92 LD E,E 6 0
93 LD E,H 6 0
94 LD E,L 6 0
95 LD E,(HL) 9 0
96 LD E,A 6 0
97 LD H,B 6 0
98 LD H,C 6 0
99 LD H,D 6 0
100 LD H,E 6 0
101 LD H,H 6 0
102 LD H,L 6 0
103 LD H,(HL) 9 0
104 LD H,A 6 0
105 LD L,B 6 0
106 LD L,C 6 0
107 LD L,D 6 0
108 LD L,E 6 0
109 LD L,H 6 0
110 LD L,L 6 0
111 LD L,(HL) 9 0
112 LD L,A 6 0
113 LD (HL),B 9 0
114 LD (HL),C 9 0
115 LD (HL),D 9 0
116 LD (HL),E 9 0
117 LD (HL),H 9 0

```

feltölthetik adatokkal az előbb ismertetett módon, a 7. lista szerint. Ezután a két programot MERGE paranccsal össze kell másolni, és ezzel a program végleges változata is használatra kész. Figyelem, még egyszer: csak GOTO 1-gyel indítható! A program kivetélt kazettára úgy, hogy betöltéskor automatikusan elinduljon, az olvasóra bízuk.

A program beindulás után megkérdezi, melyik címtől kérjük a disassemblálást, majd elkezd a fordítást. Az assembly lista a szabványos Z80 mnemonikokat írja ki. Relatív ugrásoknál a kiszámított abszolút címet is kiírja (8. lista).

Olyan memóriaterületeken, ahol a Spectrum ROM-ban táblázatok vannak, a program DEFB „szám”-ot ír ki, ha a táblázat számokat tartalmaz, illetve karaktert, ha karaktert tartalmaz, például a kulcsszó-táblázatban. A program minden számot decimális alakban jelez ki (10. és 11. lista).

A karaktergenerátor memóriaterületen a program a címet és a pontszter nyolcszorosára kinagyított képet írja ki. Így a képernyőre kirajzolódnak az egyes karakterek nagyított képei (12. lista).

Az RST 08 utasítás a hibázüzeneteket kezeli. Minden RST 08 utasítást egybajtós szám követ, mely a megfelelő hibázüzenet

kódolja. Ezt a bájtot a program DEFB „szám”-ként disassemblálja.

Az RST 40 (vagy hexadecimális jelöléssel RST 28H) utasítást a Spectrum ROM a lebegőpontos kalkulátor belépési rutinjának használja. Az RST 40 utasítás után álló néhány bájtt segítségével hívja a ROM a lebegőpontos kalkulátor egyes szubrutinjait. Így a CALL „cim” vagy a JP „cim” hárombájtos utasítások helyett csak egybajtós utasítás használható. Logan említett könyve ezeknek a bájtoknak „beszédese” neveket: címkeket adott, mi is ezeket használtuk (lásd a 11. listát).

118 LD (HL),L	9	0
119 HALT	4	0
120 LD (HL),A	9	0
121 LD A,B	6	0
122 LD A,C	6	0
123 LD A,D	6	0
124 LD A,E	6	0
125 LD A,H	6	0
126 LD A,L	6	0
127 LD A,(HL)	9	0
128 LD A,A	6	0
129 ADD A,	6	0
130 ADD A,	6	0
131 ADD A,	6	0
132 ADD A,	6	0
133 ADD A,	6	0
134 ADD A,	6	0
135 ADD A,(HL)	9	0
136 ADD A,	6	0
137 ADC A,	6	0
138 ADC A,	6	0
139 ADC A,	6	0
140 ADC A,	6	0
141 ADC A,	6	0
142 ADC A,	6	0
143 ADC A,(HL)	9	0
144 ADC A,	6	0
145 SUB B	5	0
146 SUB C	5	0
147 SUB D	5	0
148 SUB E	5	0
149 SUB H	5	0
150 SUB L	5	0
151 SUB (HL)	8	0
152 SUB A	5	0
153 SBC A,B	7	0
154 SBC A,C	7	0
155 SBC A,D	7	0
156 SBC A,E	7	0
157 SBC A,H	7	0
158 SBC A,L	7	0
159 SBC A,(HL)	1	0
160 SBC A,A	7	0
161 AND B	5	0
162 AND C	5	0
163 AND D	5	0
164 AND E	5	0
165 AND H	5	0
166 AND L	5	0
167 AND (HL)	8	0
168 AND A	5	0
169 XOR B	5	0
170 XOR C	5	0
171 XOR D	5	0
172 XOR E	5	0
173 XOR H	5	0
174 XOR L	5	0
175 XOR (HL)	8	0
176 XOR A	5	0
177 OR B	4	0
178 OR C	4	0
179 OR D	4	0
180 OR E	4	0
181 OR H	4	0
182 OR L	4	0
183 OR (HL)	7	0
184 OR A	4	0
185 CP B	4	0
186 CP C	4	0
187 CP D	4	0
188 CP E	4	0
189 CP H	4	0
190 CP L	4	0
191 CP (HL)	7	0
192 CP A	4	0
193 RET NZ	4	0
194 POP BC	6	0
195 JP NZ,	6	3
196 JP	3	3
197 CALL NZ,	8	3
198 PUSH BC	7	0

### 3. lista

199 ADD A,	6	1
200 RST 00	6	0
201 RET Z	5	0
202 RET	3	0
203 JP Z,	5	3
204 204	3	5
205 CALL Z,	7	3
206 CALL	5	3
207 ADC A,	6	1
208 RST 08	6	0
209 RET NC	6	0
210 POP DE	6	0
211 JP NC,	6	3
212 OUT ( ),A	5	1
213 CALL NC,	8	3
214 PUSH DE	7	0
215 SUB	4	1
216 RST 16	6	0
217 RET C	5	0
218 EXX	3	0
219 JP C,	5	3
220 IN A,( )	6	1
221 CALL C,	7	3
222 222	3	6
223 SBC A,	6	1
224 RST 24	6	0
225 RET PD	6	0
226 POP HL	6	0
227 JP PD,	6	3
228 EX(5P),HL	9	0
229 CALL PD,	8	3
230 PUSH HL	7	0
231 AND	4	1
232 RST 32	6	0
233 RET PE	6	0
234 JP (HL)	7	0
235 JP PE,	6	3
236 EX DE,HL	8	0
237 CALL PE,	8	3
238 238	3	7
239 XOR	4	1
240 RST 40	6	0
241 RET P	5	0
242 POP AF	6	0
243 JP P,	5	3
244 DI	3	0
245 CALL P,	7	3
246 PUSH AF	7	0
247 OR	3	1
248 RST 48	6	0
249 RET H	5	0
250 LD SP,HL	8	0
251 JP M,	5	3
252 EI	2	0
253 CALL M,	7	3
254 254	3	8
255 CP	3	1
256 RST 56	6	0

1 CODE R#(1,1,2,3,4) Y#(1)		
1 9 4 7 1	ADD,BC	
2 25 4 7 1	ADD,DE	
3 41 4 7 1	ADD,HL	
4 57 4 7 1	ADD,SP	
5 33 3 4 2	LD B,( )	
6 34 4 7 4	LD B,(+)	
7 35 4 4 1	INC	
8 42 3 5 2	LD C,( )	
9 43 4 4 1	DEC	
10 52 5 6 4	INC(+)	
11 53 5 6 4	DEC(+)	
12 54 4 5 5	LD(+)	
13 70 4 7 4	LD B,(+)	
14 76 6 7 4	LD B,(+)	
15 86 6 7 4	LD D,(+)	
16 94 6 7 4	LD E,(+)	
17 102 6 7 4	LD H,(+)	
18 110 6 7 4	LD L,(+)	
19 122 6 7 4	LD(+),B	
20 112 4 5 4	LD(+),C	
21 113 4 5 4	LD(+),D	
22 114 4 5 4	LD(+),E	
23 115 4 5 4	LD(+),H	
24 116 4 5 4	LD(+),L	
25 117 4 5 4	LD(+),A	
26 119 4 5 4	LD(+),A	
27 134 7 8 4	ADD A,(+)	
28 147 7 8 4	ADD C,(+)	
29 158 5 6 4	SUB(+)	
30 158 7 8 4	SBC A,(+)	
31 166 5 6 4	AND(+)	
32 174 5 6 4	XOR(+)	
33 182 6 6 1	OR(+)	
34 198 4 5 4	CF(+)	
35 203 1 1 6		
36 225 4 4 1	POP	
37 227 8 8 1	EX(SP),	
38 229 5 5 1	PUSH	
39 233 4 4 1	JP(+)	
40 248 6 6 1	LD SP,	
41 4 5 6 4	RLO(+)	
42 14 5 6 4	RRC(+)	
43 22 4 5 4	RL(+)	
44 38 4 5 4	RR(+)	
45 38 5 6 4	SLA(+)	
46 46 5 6 4	SRA(+)	
47 62 5 6 4	SWAP(+)	
48 78 7 8 4	BIT 0,(+)	
49 79 7 8 4	BIT 1,(+)	
50 86 7 8 4	BIT 2,(+)	
51 94 7 8 4	BIT 3,(+)	
52 102 7 8 4	BIT 4,(+)	
53 110 7 8 4	BIT 5,(+)	
54 118 7 8 4	BIT 6,(+)	
55 126 7 8 4	BIT 7,(+)	
56 134 7 8 4	RES 0,(+)	
57 142 7 8 4	RES 1,(+)	
58 150 7 8 4	RES 2,(+)	
59 158 7 8 4	RES 3,(+)	
60 166 7 8 4	RES 4,(+)	
61 174 7 8 4	RES 5,(+)	
62 182 7 8 4	RES 6,(+)	
63 190 7 8 4	RES 7,(+)	
64 198 7 8 4	SET 0,(+)	
65 206 7 8 4	SET 1,(+)	
66 214 7 8 4	SET 2,(+)	
67 222 7 8 4	SET 3,(+)	
68 230 7 8 4	SET 4,(+)	
69 238 7 8 4	SET 5,(+)	
70 246 7 8 4	SET 6,(+)	
71 254 7 8 4	SET 7,(+)	
72 64 8 8 1	IN B,(C)	
73 72 8 8 1	IN C,(D)	
74 80 8 8 1	IN D,(E)	

75 88 8 8 1	IN E,(C)
76 96 8 8 1	IN H,(C)
77 104 8 8 1	IN L,(C)
78 120 8 8 1	IN A,(C)
79 65 8 8 1	OUT C,(B)
80 73 8 8 1	OUT C,(D)
81 81 8 8 1	OUT C,(E)
82 89 8 8 1	OUT C,(H)
83 97 8 8 1	OUT C,(L)
84 105 8 8 1	OUT C,(A)
85 121 8 8 1	OUT C,(A)
86 64 8 8 1	SBC HL,BC
87 82 8 8 1	SBC HL,DE
88 98 8 8 1	SBC HL,HL
89 114 8 8 1	SBC HL,SP
90 67 4 8 2	LD C,(BC)
91 83 4 8 2	LD C,(DE)
92 99 4 8 2	LD C,(HL)
93 115 4 8 2	LD C,(SP)
94 6 8 8 1	NEG
95 69 8 8 1	RETN
96 78 8 8 1	IM 0
97 86 8 8 1	IM 1
98 94 8 8 1	IM 2
99 71 8 8 1	LD I,A
100 74 8 8 1	ADC HL,BC
101 90 8 8 1	ADC HL,DE
102 106 8 8 1	ADC HL,HL
103 122 8 8 1	ADC HL,SP
104 75 7 8 2	LD BC,( )
105 91 7 8 2	LD DE,( )
106 107 7 8 2	LD HL,( )
107 123 7 8 2	LD SP,( )
108 77 8 8 1	RETI
109 79 8 8 1	LD R,A
110 87 8 8 1	LD R,L
111 95 8 8 1	LD R,H
112 103 8 8 1	RRD
113 111 8 8 1	RLD
114 160 8 8 1	LDI
115 161 8 8 1	CPI
116 162 8 8 1	INRI
117 163 8 8 1	OUTI

### 4. lista

118 160 8 8 1	LDI
119 169 8 8 1	CFD
120 170 8 8 1	IND
121 171 8 8 1	OUTD
122 176 8 8 1	LDIR
123 177 8 8 1	CFJR
124 178 8 8 1	INIR
125 179 8 8 1	STIR
126 184 8 8 1	LDRI
127 185 8 8 1	CFOR
128 186 8 8 1	INDR
129 187 8 8 1	OTDR

A lebegőpontos kalkulátor működését egyszerűsítve úgy foghatjuk fel, hogy a ROM tartalmát általában Z80 gépi kódban írták meg, azonban az RST 40 utasítás után egy ennél magasabb szintű nyelv utasításainak értelmezésére tér át az interpreter. Ezeket az utasításokat egy „end calc” címkejű utasítás zárja le. Ettől kezdve az interpreter visszatér a normál gépi kódú utasítások használatára.

A lebegőpontos kalkulátor néhány utasítása hasonlóan működik, mint a BASIC függvényei: független változókat, paramétereket vár. Ilyen például a „series-06” vagy a „stk-data” utasítás. Ezek a paramé-

terek a Sinclair BASIC-nek megfelelő formátumú öt bájtos lebegőpontos számok vagy sztringek, melyek közvetlenül követik az illető címke (utasítást). Ezeket a paramétereket a program DEFB „szám” alakban, bájtként írja ki.

A disassembler program alapötlete rendkívül egyszerű. A program végtelen ciklusban minden bájtot sorban megvizsgál. Ha a K-adik bájtot 203 (CBH) vagy 237 (EBH), akkor több-bájtos utasításról van szó, és a program elágazik. Ha nem, kinyomtatja az egybájtos mnemonikot, amely nem más, mint az AS adattombb megfelelő eleme: AS (PEEK K+1).





ORA absz.

13 RETURN

3328 CT M

# Commodore 64 segéd táblázat

Ismert dolog, hogy a Commodore 64 számítógépek BASIC-je nem nyújt elég támogatást a gép hardverlehetőségeinek teljes kihasználásához. Ezért sokan alkalmazták azt a módszert, hogy BASIC nyelven megírt főprogramjukból gépi kódú szubrutinokat hívnak, amelyek a tár különböző részein helyezkednek el. Leggyakrabban a kazettapuffert (helye a tárban: 828–1019 vagy hexadecimálisan \$033C–\$03FB), illetve a 49152 (\$C000)-nál kezdődő 4 k-s RAM-területet használják erre a célra. De hogyan kerülnek a gépi kódú rutinok a megfelelő helyre? Erre több lehetőség is van; például monitorprogram segítségével, amely megengedi az utasítások mnemonikus formában történő bevitelét. Az így megírt program lemezre menthető, majd a főprogram indítása előtt a megfelelő tárcímre tölthető.

Nem mindig célszerű azonban ezt a megoldást választani, hiszen ha a program csak néhány utasítást tartalmaz, akkor fölösleges háttértároló segítség igénybe venni. Ebben az esetben a legegyszerűbb az utasításkódoknak megfelelő számokat DATA-sorokban tárolni, majd READ utasításokkal a kívánt tárcímre olvastatni.

Többek között ehhez a megoldáshoz nyújthat hasznos segítséget a táblázat, amely első pillantásra kissé zűzöltonak tűnhet, használata azonban rövid idő alatt elsajátítható.

Válasszunk ki egy tetszőleges mezőt és nézzük meg, milyen információkat tartalmaz! Legyen ez a 0. sor és a D. oszlop találkozásában található, azaz a \$00 hexadecimális számnak megfelelő mező (lásd az ábrát).

A felső sorban található „ORA absz.” a gépi kódú programozásban jártasak számára magától értetődő, ez a \$0D utasításkódnak megfelelő gépi kódú utasítás mnemonikus formája a megfelelő címzés móddal együtt. Az adott esetben abszolút címzés módú utasításról van szó.

A középső sorban két információ is található: a 13 a \$0D hexadecimális számnak megfelelő decimális érték; a RETURN a PRINT CHR\$(13) utasítás hatását mutatja. Általánosítva: a bal oldali szám a jobboldalt látható vezérlőutasítás, karakter stb. ASCII kódja. Ha a vizsgált mező jobb alsó sarkában is található valamilyen információ (az adott esetben CT M), akkor ez azt jelenti, hogy az adott funkció még egy másodan megvalósítható, példánk esetén a

CTRL és az M billentyűk együttes lenyomásával.

Végül a bal alsó sarkokban látható számrol: ez mindig a fölötte levő szám 256-szorosra, azaz  $13 \times 256 = 3328$ .

Nézzünk néhány példát a táblázat gyakorlati alkalmazására! Legyen az első feladat a \$C01B hexadecimális szám átváltása decimálisra! Első lépésként válasszuk szét az adott számot a felső (HIGH) és alsó (LOW) bájtokra! Keressük ki a táblázatból a felső bájtnak (\$C0) megfelelő decimális értéket, amely természetesen az adott mező bal alsó sarkában levő 256-szoros számot jelenti. Ez a mi esetünkben 49 152. Végezzük el ugyanezt az alsó bájttal (\$1B) is, itt azonban a középső sorban levő számról lesz szükségünk, azaz 27-re. Adjuk össze a két számot. Ezzel a feladatot meg is oldottuk: a \$C01B hexadecimális szám decimális megfelelője tehát  $49\ 152 + 27 = 49\ 179$ .

Visszafelé majdnem ugyanilyen egyszerű a számolás. Váltunk át hexadecimálisra a 34 519-et! Keressük ki a táblázatból az adott számnál nem nagyobb, hozzá legközelebb eső számot! Ez a mi esetünkben 34 304, ezzel megkaptuk a felső bájtot, azaz \$86-ot. Vonjuk ki ezt a számot az eredeti számból:  $34\ 519 - 34\ 304 = 215$ , az ennek megfelelő hexadecimális szám (\$D7) adja az alsó bájtot. Eredményünk tehát: 34 519 = \$86D7.

Harmadik példánk azt mutatja meg, hogy hogyan lehet a legkényelmesebben leltitani, illetve engedélyezni a gép kétély karakterkészlete közötti átkapcsolást. Bizonyára sokak előtt ismert, hogy a PRINT CHR\$(8) utasítás leltítja a C = és a SHIFT billentyűk egyidejű lenyomásával végrehajtható átváltást, a PRINT CHR\$(9) utasítás pedig újra engedélyezi az eredeti állapotot. Ha azonban megnézzük táblázatunkban a \$08 és a \$09 mezőket, akkor észrevehetjük, hogy ugyanezt a hatást sokkal egyszerűbben is elérhetjük a CTRL és a H, illetve a CTRL és az I billentyűk egyidejű megnyomásával.

Végül nézzük egy rövid gépi kódú program DATA sorokba írását a táblázat segítségével. Legyen a feladat egy fekete A betű kiírása a képernyő bal felső sarkába! Először írjuk meg a forrásnyelvi programot. Ehhez azonban ismernünk kell néhány címet és kódot. Az A betű képernyőkódja 1, a fekete színkódja 0. A képernyőtároló első bájtyának címe, ahová a betű kerül: 1024, a színmémória megfelelő címe, ahová majd a

felete színkódja kerül: 55 296. Keressük ki a már leírt eljárás segítségével ezeknek a számoknak a hexadecimális megfelelőjét, majd írjuk meg a programot!

```
LDA # $01
STA $0400
LDA # $00
STA $D800
RTS
```





















































Vegyük sorra az utasításokat és a hozzájuk tartozó operandusokat! Az LDA # utasítás kódja a táblázat alapján \$A9 (= 169), az operandus \$01 (= 1). A következő utasítás abszolút címzés módú STA kódja \$8D (= 141). Miután itt 2 bájtos az operandus, a bájtok felcserélve kerülnek a tárgyprogramba, tehát \$00 (= 0); \$01 (= 1) sorrendben! A továbbiakban ennek megfelelően folytatva, végezetül a kész DATA-sor:

```
DATA 169,1,141,0,4,169,0,141,0,218,96
```

Azoknak, akik már jártasak a gépi kódú programozásban, bizonyára feltűnt, hogy olyan mnemonikok is találhatóak a táblázatban (ezek eltérő színháttérrel vannak jelölve), amelyek ismeretlenek előttük. A Mikroszámítógép Magazin egyik régebbi száma közölt egy cikket „Ismeretlen kódot” címmel, amelyben ezek egy részéről már volt szó, csak az elnevezések tértek el az itt leírtaktól. Az itt közölt táblázat az SMON nevű monitorprogram elnevezéseit használja, és csekély kivételtől eltekintve szinte az összes lehetséges kódot értelmezi: az elméletileg lehetséges 256-ból 242-t! Miután ezeknek az utasításoknak a többsége valószínűleg ismeretlen az olvasó előtt, röviden nézzük át a kódot jelentését!

A legelső „új” utasítás tulajdonképpen két másik, már ismert utasítást egyesít, tehát a státuszregiszter megfelelő jelzőbitjeit is ennek függvényében állítja.

LAX	Load Akku and X Végrehajt egy LDA és egy LDX utasítást a címzés módnak megfelelően
DCP	Decrement and ComPare DEC és CMP utasítások egyesítése
ISC	Increment and SubtraCt INC és SBC
RLA	Rotate Left AND Akku ROL és AND
RRA	Rotate Right and Add with carry ROR és ADC
SLO	Shift Left OR Akku

	0	1	2	3	4	5	6	7
0	BRK 0 0	ORA (zp,x) 1 CTRL A 256	CRA 2 CTRL B 512	SLO (zp,x) 3 CTRL C 768	NOP-2 4 CTRL D 1024	ORA zp 5 FEHÉR CTRL E 1280	ASL zp 6 CTRL F 1536	SLO zp 7 CTR 1792
1	BPL 16 CTRL P 4096	ORA (zp,y) 17 CRSR CTRL Q 4352	CRA 18 RVS ON CTRL R 4608	SLO (zp,y) 19 HOME CTRL S 4864	NOP-2 20 DEL CTRL T 5120	ORA zp,x 21 CTRL U 5376	ASL zp,x 22 CTRL V 5632	SLO zp 23 CTR 5888
2	JSR abs. SPACE 32 8192	AND (zp,x) 33 ! 8448	CRA 34 " 8704	RLA (zp,x) 35 ≠ 8960	BIT zp 36 \$ 9216	AND zp 37 % 9472	ROL zp 38 & 9728	RLA zp 39 9984
3	BMI 48 0 12288	AND (zp,y) 49 1 12544	CRA 50 2 12800	RLA (zp,y) 51 3 13056	NOP-2 52 4 13312	AND zp,x 53 5 13568	ROL zp,x 54 6 13824	RLA zp 55 14080
4	RTI 64 @ 16384	EOR (zp,x) 65 A 16640	CRA 66 B 16896	SRE (zp,x) 67 C 17152	NOP-2 68 D 17408	EOR zp 69 E 17664	LSR zp 70 F 17920	SRE zp 71 18176
5	BVC 80 P 20480	EOR (zp,y) 81 Q 20736	CRA 82 R 20992	SRE (zp,y) 83 S 21248	NOP-2 84 T 21504	EOR zp,x 85 U 21760	LSR zp,x 86 V 22016	SRE zp 87 22272
6	RTS 96  24576	ADC (zp,x) 97  24832	CRA 98  25088	RRA (zp,x) 99  25344	NOP-2 100  25600	ADC zp 101  25856	ROR zp 102  26112	RRA zp 103  26368
7	BVS 112  28672	ADC (zp,y) 113  28928	CRA 114  29184	RRA (zp,y) 115  29440	NOP-2 116  29696	ADC zp,x 117  29952	ROR zp,x 118  30208	RRA zp 119  30464
8	NOP-2 128 32768	STA (zp,x) 129 NARANCS 33024	NOP-2 130 33280	SAX (zp,x) 131 33536	STY zp 132 33792	STA zp 133 f1 34048	STX zp 134 f3 34304	SAX zp 135 34560
9	BCC 144 FEKETE 36864	STA (zp,x) 145 CRSR 37120	CRA 146 RVS OFF 37376	— 147 CLR 37632	STY zp,x 148 INST 37888	STA zp,x 149 BARNA 38144	STX zp,y 150 ROZSASZ 38400	SAX zp 151 S.S. 38656
A	LDY # 160 SH-SPACE 40960	LDA (zp,x) 161  41216	LDX # 162  41472	LAX (zp,x) 163  41728	LDY zp 164  41984	LDA zp 165  42240	LDX zp 166  42496	LAX zp 167  42752
B	BCS 176  45056	LDA (zp,y) 177  45312	CRA 178  45568	LAX (zp,y) 179  45824	LDY zp,x 180  46080	LDA zp,x 181  46336	LDX zp,y 182  46592	LAX zp 183  46848
C	CPY # 192  49152	CMP (zp,x) 193  49408	NOP-2 194  49664	DCP (zp,x) 195  49920	CPY zp 196  50176	CMP zp 197  50432	DEC zp 198  50688	DCP zp 199  50944
D	BNE 208  53248	CMP (zp,y) 209  53504	CRA 210  53760	DCP (zp,y) 211  54016	NOP-2 212  54272	CMP zp,x 213  54528	DEC zp,x 214  54784	DCP zp 215  55040
E	CPX ≠ 224 SH-SPACE 57344	SBC (zp,x) 225  57600	NOP-2 226  57856	ISC (zp,x) 227  58112	CPX zp 228  58368	SBC zp 229  58624	INC zp 230  58880	ISC zp 231  59136
F	BEQ 240  61440	SBC (zp,y) 241  61696	CRA 242  61952	ISC (zp,y) 243  62208	NOP-2 244  62464	SBC zp,x 245  62720	INC zp,x 246  62976	ISC zp 247  63232

	8	9	A	B	C	D	E	F
L G	PHP 8 SHIFT-C-TILT 2648 CTRL H	ORA # 9 SHIFT-C-ENGED 2364 CTRL I	ASL A 10 CTRL J 2560	— 11 CTRL K 2816	NOP-3 12 CTRL L 3072	ORA absz. 13 RETURN 3328 CTRL M	ASL absz. 14 KISBETŰ 3584 CTRL N	SLO absz. 15 CTRL O 3840
X L W	CLC 24 CTRL X 6144	ORA absz,y 25 CTRL Y 6400	NOP 26 CTRL Z 6656	SLO absz,y 27 CTRL [ 6912	NOP-3 28 PIROS 7168	ORA absz,x 29 CRSR→ 7424	ASL absz,x 30 ZÖLD 7680	SLO absz,x 31 KÉK 7936
	PLP 40 ( ) 10240	AND # 41 ) 10496	ROL A 42 * 10752	— 43 + 11008	BIT absz. 44 , 11264	AND absz. 45 - 11520	ROL absz. 46 . 11776	RLA absz. 47 / 12032
X 7	SEC 56 8 14336	AND absz,y 57 9 14592	NOP 58 : 14848	RLA absz,y 59 ; 15104	NOP-3 60 < 15360	AND absz,x 61 = 15616	ROL absz,x 62 > 15872	RLA absz,x 63 ? 16128
G	PHA 72 H 18432	EOR # 73 I 18688	LSR A 74 J 18944	— 75 K 19200	JMP absz 76 L 19456	EOR absz 77 M 19712	LSR absz 78 N 19968	SRE absz. 79 O 20224
X W	CLI 88 X 22528	EOR absz,y 89 Y 22784	NOP 90 Z 23040	SRE absz,y 91 [ 23296	NOP-3 92 £ 23552	EOR absz,x 93 ] 23808	LSR absz,x 94 ↑ 24064	SRE absz,x 95 ← 24320
	PLA 104 <input type="checkbox"/> 26624	ADC # 105 <input type="checkbox"/> 26880	ROR A 106 <input type="checkbox"/> 27136	— 107 <input type="checkbox"/> 27392	JMP(absz) 108 <input type="checkbox"/> 27648	ADC absz 109 <input type="checkbox"/> 27904	ROR absz 110 <input checked="" type="checkbox"/> 28160	RRA absz 111 <input type="checkbox"/> 28416
X O	SEI 120 <input checked="" type="checkbox"/> 30720	ADC absz,y 121 <input type="checkbox"/> 30976	NOP 122 <input checked="" type="checkbox"/> 31232	RRA absz,y 123 <input checked="" type="checkbox"/> 31488	NOP-3 124 <input checked="" type="checkbox"/> 31744	ADC absz,x 125 <input type="checkbox"/> 32000	ROR absz,x 126 <input type="checkbox"/> 32256	RRA absz,x 127 <input checked="" type="checkbox"/> 32512
fs	DEY 136 f7 34816	NOP-2 137 f2 35072	TXA 138 f4 35328	— 139 f6 35584	STY absz 140 f8 35840	STA absz 141 SH-RET 36096	STX absz 142 NAGYBETŰ 36352	SAX absz 143 36608
y ZÜRKE	TYA 152 SZÜRKE 38912	STA absz,y 153 V. ZÖLD 39168	TXS 154 V. KÉK 39424	— 155 V.SZÜRKE 39680	— 156 BIBOR 39936	STA absz,x 157 CRSR— 40192	— 158 SÁRGA 40448	— 159 CIÁN 40704
	TAY 168 <input checked="" type="checkbox"/> 43008	LDA # 169 <input checked="" type="checkbox"/> 43264	TAX 170 <input type="checkbox"/> 43520	— 171 <input checked="" type="checkbox"/> 43776	LDY absz 172 <input type="checkbox"/> 44032	LDA absz 173 <input type="checkbox"/> 44288	LDX absz 174 <input type="checkbox"/> 44544	LAX absz 175 <input type="checkbox"/> 44800
y O	CLV 184 <input type="checkbox"/> 47184	LDA absz,y 185 <input type="checkbox"/> 47360	TSX 186 <input type="checkbox"/> 47616	— 187 <input type="checkbox"/> 47872	LDY absz,x 188 <input type="checkbox"/> 48128	LDA absz,x 189 <input type="checkbox"/> 48384	LDX absz,y 190 <input type="checkbox"/> 48640	LAX absz,y 191 <input checked="" type="checkbox"/> 48896
	INY 200 <input type="checkbox"/> 51200	CMP # 201 <input type="checkbox"/> 51456	DEX 202 <input type="checkbox"/> 51712	— 203 <input type="checkbox"/> 51968	CPY absz 204 <input type="checkbox"/> 52224	CMP absz 205 <input type="checkbox"/> 52480	DEC absz 206 <input checked="" type="checkbox"/> 52736	DCP absz 207 <input type="checkbox"/> 52992
	CLD 216 <input checked="" type="checkbox"/> 55296	CMP absz,y 217 <input type="checkbox"/> 55552	NOP 218 <input checked="" type="checkbox"/> 55808	DCP absz,y 219 <input checked="" type="checkbox"/> 56064	NOP-3 220 <input checked="" type="checkbox"/> 56320	CMP absz,x 221 <input type="checkbox"/> 56576	DEC absz,x 222 <input type="checkbox"/> 56832	DCP absz,x 223 <input checked="" type="checkbox"/> 57088
	INX 232 <input checked="" type="checkbox"/> 59392	SBC # 233 <input checked="" type="checkbox"/> 59648	NOP 234 <input type="checkbox"/> 59904	— 235 <input checked="" type="checkbox"/> 60160	CPX absz 236 <input type="checkbox"/> 60416	SBC absz 237 <input type="checkbox"/> 60672	INC absz 238 <input type="checkbox"/> 60928	ISC absz, 239 <input type="checkbox"/> 61184
X O	SED 248 <input type="checkbox"/> 63488	SBC absz,y 249 <input type="checkbox"/> 63744	NOP 250 <input type="checkbox"/> 64000	ISC absz,y 251 <input type="checkbox"/> 64256	NOP-3 252 <input type="checkbox"/> 64512	SBC absz,x 253 <input type="checkbox"/> 64768	INC absz,x 254 <input type="checkbox"/> 65024	ISC absz,x 255 <input type="checkbox"/> 65280

ORA absz.

13 RETURN  
3328 CT M



1800	1800	1	DATA	NUMIO	MLLITRAN						
1800	1100	1	A	NUMIO	1	SOR	ELERKE				
1800	1300	1	HEVIG								
1800	1500	1	BYS	MS	GOSUB	MOOD					
1800	1400										
8027	20	72	90	PHASZ							
8028	23	20	190	CRP	HEJ						
8029	26	56	400	BRG	BYT	SCINTRIKTA	VIZDOLKTA				
8030	28	73	800	JOB	MOV						
8031	32	80	620	CRP	HEV						
8032	36	57	420	BRG	BYT						
8033	39	72	800	JOB	MOV						
8034	20	85	AD	250	JOB	MOV	10000	UTANI	2000	ATKALAKTA	10000
8035	20	87	BT	400	JOB	MOV	10000	UTANI	2000	ATKALAKTA	10000
8036				470	JOB	MOV	10000	UTANI	2000	ATKALAKTA	10000
8037	20	20	200	REML	LDA	K28					
8038	20	20	200	REML	LDA	K28					
8039	20	20	200	REML	LDA	K28					
8040	20	20	200	REML	LDA	K28					
8041	20	20	200	REML	LDA	K28					
8042	20	20	200	REML	LDA	K28					
8043	20	20	200	REML	LDA	K28					
8044	20	20	200	REML	LDA	K28					
8045	20	20	200	REML	LDA	K28					
8046	20	20	200	REML	LDA	K28					
8047	20	20	200	REML	LDA	K28					
8048	20	20	200	REML	LDA	K28					
8049	20	20	200	REML	LDA	K28					
8050	20	20	200	REML	LDA	K28					
8051	20	20	200	REML	LDA	K28					
8052	20	20	200	REML	LDA	K28					
8053	20	20	200	REML	LDA	K28					
8054	20	20	200	REML	LDA	K28					
8055	20	20	200	REML	LDA	K28					
8056	20	20	200	REML	LDA	K28					
8057	20	20	200	REML	LDA	K28					
8058	20	20	200	REML	LDA	K28					
8059	20	20	200	REML	LDA	K28					
8060	20	20	200	REML	LDA	K28					
8061	20	20	200	REML	LDA	K28					
8062	20	20	200	REML	LDA	K28					
8063	20	20	200	REML	LDA	K28					
8064	20	20	200	REML	LDA	K28					
8065	20	20	200	REML	LDA	K28					
8066	20	20	200	REML	LDA	K28					
8067	20	20	200	REML	LDA	K28					
8068	20	20	200	REML	LDA	K28					
8069	20	20	200	REML	LDA	K28					
8070	20	20	200	REML	LDA	K28					
8071	20	20	200	REML	LDA	K28					
8072	20	20	200	REML	LDA	K28					
8073	20	20	200	REML	LDA	K28					
8074	20	20	200	REML	LDA	K28					
8075	20	20	200	REML	LDA	K28					
8076	20	20	200	REML	LDA	K28					
8077	20	20	200	REML	LDA	K28					
8078	20	20	200	REML	LDA	K28					
8079	20	20	200	REML	LDA	K28					
8080	20	20	200	REML	LDA	K28					
8081	20	20	200	REML	LDA	K28					
8082	20	20	200	REML	LDA	K28					
8083	20	20	200	REML	LDA	K28					
8084	20	20	200	REML	LDA	K28					
8085	20	20	200	REML	LDA	K28					
8086	20	20	200	REML	LDA	K28					
8087	20	20	200	REML	LDA	K28					
8088	20	20	200	REML	LDA	K28					
8089	20	20	200	REML	LDA	K28					
8090	20	20	200	REML	LDA	K28					
8091	20	20	200	REML	LDA	K28					
8092	20	20	200	REML	LDA	K28					
8093	20	20	200	REML	LDA	K28					
8094	20	20	200	REML	LDA	K28					
8095	20	20	200	REML	LDA	K28					
8096	20	20	200	REML	LDA	K28					
8097	20	20	200	REML	LDA	K28					
8098	20	20	200	REML	LDA	K28					
8099	20	20	200	REML	LDA	K28					
8100	20	20	200	REML	LDA	K28					

## COMMODORE 64 RESTORE n

Többször előfordult, hogy sok DATA közül egy adott csoportra volt szükségem. Ha cikluson belül többször kellett hozzáférfi ugyanazokhoz az adatokhoz, akkor jó lett volna használni a RESTORE n utasítást, ezt azonban a C64 nem engedi meg. Ezért készítettem egy gépi kódú programot, amely helyettesíti a RESTORE n-et.

Három követelményt támasztottam a programmal szemben: könnyen kezelhető legyen; a HELP-pel végzett újrásorszámozás esetén az n értéke módosuljon; hibás használat esetén hibáüzeneteket adjon.

A program hívása:  
SYS855:GOSUB n (n a kívánt sorszám)  
Megszorítás, hogy a kívánt sornak DATA-val kell kezdődnie.

Az utasítás helytelen begépelésére a gép "SYNTAX ERROR"-ral válaszol. Nem létező sorra hivatkozáskor "?UNDEF'D STATEMENT ERROR" jelenik meg. Amennyiben a hívott sor első utasítása nem DATA, akkor "?NINCS DATA A SORBAN" kerül a képernyőre.

Az 1. listán az Assembly program, a 2. listán pedig a bemutató program látható.  
FEKETE IMRE

## Mit ismer a HELP + ?

Az 1986. márciusi számukban „Floppy-kezelés Kernal-rutinok segítségével” címmel közreadott cikkben téves információkat jelentettek meg. A HELP + nevű program assembler értelmező része igen ismeri a cikkben egyébként hiányzott <> jeleket. Ezeknek felhasználásához mellékelem a következő assembler-listát, ami a nevezett program assemblerével készült.

ÁDÁMKÓ LAJOS  
Ózd

### 1. lista

### 2. lista

```

10 OPEN#4:PRINT#4," H";SPC(5);"B";SPC(5);"C"
20 PRINT#4," "
30 DATA 1,2,3,4
40 DATA 5,6,7,8
50 FOR I=1 TO 5
60 SYS 855:GOSUB 40:REND H:B:C
70 PRINT#4,H;SPC(3);B;SPC(3);C
80 NEXT I
90 CLOSE 4
    
```

H	B	C
5	6	7
5	6	7
5	6	7
5	6	7
5	6	7

### PROBA3

```

0000          100      #=&C000
0035          110      ALAP      #=&C333
0000 A9 C3          LDA #ALAP>
0002 BD 00 C1      STA #C100
0005 A9 C3          LDA #ALAP>
0007 BD 01 C1      STA #C101
000A A2 C3          LDX #ALAP>
000C A9 C3          LDY #ALAP>
000E BE 02 C1      STX #C102
0011 BC 03 C1      STY #C103
0014 60            RTS
0015              &5535      .END
    
```

ZEILEN:12 SYMBOLE:1 FEHLER:0  
ALAP =#C333

### ASL és ORA

Shift Right and EOR Akku  
LSR és EOR

### SAX Store Akku AND X

Logikai ÉS műveletet hajt végre az akku és az X regiszter között, az eredményt a címzés mód szerinti memóriacímre teszi

### CRA CRAs

„Lefagyasztja” a processzor működését

### NOP

NOP—2 Üres utasítás, a processzor erre az NOP—3 utasításra érve nem hajt végre semmilyen műveletet, sőt az utasítástól függően még 1, illetve 2 bajtot nem vesz figyelembe

Az „új” utasítások eredményes alkalmazását egyelőre korlátozza az a tény, hogy pillanatnyilag nincs olyan assembler fordítóprogram, amely ismerné ezeket a kódokat. (Magyar szakemberek, figyeltem! Itt a lehetőség egy ilyen program megírására! Egészen biztos, hogy lenne rá kereslet!) Az általam használt SMON monitorprogram is csak visszafordítani képes az új utasításokat, beírni nem lehet vele. Létezik ezenkívül egy FMON nevű monitorprogram, amely már ezt is tudja, azonban más elnevezéseket alkalmaz. Azoknak, akiket érdekel és esetleg a kezükbe kerül az FMON, felsorolom az általa használt mnemonikákat:

SMON	FMON
LAX	LTX
DPC	DEM
ISC	INB
RLA	RAN
RRA	RAD
SLO	ASO
SRE	ESE
SAX	AXX
CRA	ABS
NOP	NO1
NOP—2	NO2
NOP—3	NO3

A táblázatban használt rövidítések:  
absz. = abszolút  
zp. = zero page (0. lap)  
CT v. CTRL = CONTROL  
SH = SHIFT  
v. = világos  
s. = sötét

SZATMÁRI LÁSZLÓ

# Képernyőn a Primo agya!

A számítógép belső világának, tárolóterületeinek, adatmozgatásának áttekintése nem könnyű a kezdő számára. Ebben szeretnék segíteni programommal, amely a képernyőre vetíti a gép RAM-ját. Az pedig még a haladónak is élmény lehet, hogy a kép él, mozog; megfigyelhető a rendszerváltozók és a verem állandó változása, az adatok, tömbök elhelyezkedése, mozgása. Gyakorlati célt nem szolgál a program, csak ismerkedés, barátkozás a géppel, bepillantás lelkivilágába!

A képernyőre vetítés érdekében a RAM-ot 6 kb-igra korlátoztam, így egyszerre jelenik meg a képernyő tetőjén a munkaterület (16384—17385), alatta beírt programunk szövege (17386-tól), a változó- és tömbtábla, a folyton izgó-mozgó verem, a sztringterület, a legalsó sorban pedig a rövid gépi kódú szubrutin, amely a képernyő-RAM-ba másolást végzi.

A gépi kódú szubrutin tömbmásolást végez: LD HL, 4000H a RAM kezdete

LD DE, 4 képernyőjár kezdőcíme, a Primo típusától függően!

LD BC, 1800H az átírandó terület mérete

LDIR

RET

A továbbiakat maga a program magyarázza meg. A kezdők tájékoztatására meg kell jegyezni, hogy a képernyőn a tárolók tartalma bitenként jelenik meg: a világító pont 1-et, a sötét 0-at jelent.

FEKETE GYÖRGY

**A szerkesztő megjegyzése:** Az érdekes és látványos, de a szerző szerint is gyakorlati célt nem szolgáló programnak van egy sokkal egyszerűbb, mégis többet tudó és gyakorlati hasznú DRAGON (TRS80 Color) változata. A program egyetlen utasítás párból áll, mellyel a memória (a teljes 64 kb-ot!) tetszés szerinti 0,5 kb-igját jelölhetjük ki képernyőterületre, illetve ezt a kijelölést törölhetjük. A bekapcsolás alapcíme 65479, a kikapcsolásé 65478. Ezekhez kell hozzáadni

2-hog(4-M)

a-h M a kijelendő memória kezdőcíme kb-igat egysegekben. Az így kiadódó címre kell beírni egy 0—255 közötti értéket. Ennek a programcskának programok működés közbeni követésénél van szerepe.

```

10 REM *****
20 REM * KÉPERNYŐN A PRIMO AGYA! *
30 REM *****
40 POKE 1651,127:POKE 1652,156
50 DEFINT A-H:DEFINT B-D:DEFINT I
60 G=256-477:DEFINT J:DEFINT K
70 POKE 6550,65479:POKE 6551,65478:DEFINT L
80 POKE 6552,255:CLS:PRINT (CHR$(65))
90 FOR D=0 TO 159:HCALL G: NEXT
100 PRINT "1.4.1"
110 PRINT "3.5.2"
120 PRINT "4.5.3"
130 PRINT "11.8.7"
140 PRINT "13.8.4"
150 PRINT "15.8.4"
160 FOR D=0 TO 20000: NEXT
170 FOR D=0 TO 159:HCALL G: NEXT
180 PRINT "8.8.8"
190 FOR D=0 TO 6000: NEXT
200 FOR A=0 TO 400:FOR B=0 TO 400:
210 HCALL G: NEXT
220 PRINT "7.8.7"
230 FOR D=0 TO 20000: NEXT
240 FOR G=0 TO 100
250 "11.11.11"
260 HCALL G: NEXT
270 PRINT "7.8.7"
280 FOR D=0 TO 20000: NEXT
290 PRINT "11.11.11"
300 POKE 6552,255
310 FOR A=0 TO 159:FOR B=0 TO 159:
320 G=256-477+4*A+B:HCALL G:
330 NEXT B: NEXT A
340 NEXT D
350 CLS:PRINT "7.15.7"
360 POKE 1651,127:POKE 1652,156

```

## COMMODORE 64

### Üzemmódok keverése

```

0000 78 SEI
0001 A9 24 LDA #24
0003 8D 14 03 STA $0314
0006 A9 00 LDA #00
0008 8D 15 03 STA $0315
000B A9 00 LDA #00
000D 8D 12 D0 STA $D012
0010 AD 11 D0 LDA $D011
0013 29 7F AND #7F
0015 8D 11 D0 STA $D011
0018 A9 81 LDA #81
001A 8D 1A D0 STA $D01A
001D A9 7F LDA #7F
001F 8D 00 D0 STA $D00D
0022 58 CLI
0023 60 RTS
0024 AD 19 D0 LDA $D019
0027 8D 19 D0 STA $D019
002A AD 12 D0 LDA $D012
002D F0 12 BEQ $C041
002F A9 1B LDA #1B
0031 8D 11 D0 STA $D011
0034 A9 15 LDA #15
0036 8D 18 D0 STA $D018
0039 A9 00 LDA #00
003B 8D 12 D0 STA $D012
003E 4C BC FF JMP $FBC
0041 A9 3B LDA #3B
0043 8D 11 D0 STA $D011
0046 A9 28 LDA #28
0048 8D 18 D0 STA $D018
004B A9 CA LDA #CA
004D 8D 12 D0 STA $D012
0050 4C 31 EA JMP $FA31
0053 FF ???

```

## COMMODORE 64

### ROM-rutint bevezető program

A program a 0810H című 08CBH-ig tart. Így lehetőség van arra, hogy elé egy BASIC indítóprogramot írjunk. A gépi kódú főprogram 081AH-tól kezdődik, tehát SYS 2074-re. A program egy pár BASIC- és KERNAL-rutint és használatukat mutatja be (képernyőtörölés, véletlenszám-generálás, egészé alakítás, lebegőpontos szorzás stb.).

A kezdeti értékek beállítása után végtelen ciklusban dolgozik. Ebből esetleges elágaztatásra az F1 gomb hatására lép ki. Jelen esetben egy BASIC megkezdést hajt végre, ami természetesen átirható akármilyen másra. Ez az utasítás a memória 083BH címen helyezkedik el.

Minden ciklusban két véletlenszám generálódik. Az egyik 0—1 közé eső számot 20-szal, a másikat 12-vel szorozza meg, és egész részüket elhelyezi a memória egy-egy bájtián. Ezután a képernyő közepétől jobbra le, jobbra fel, balra fel és balra le szimmetrikusan, a két véletlenszámnak megfelelően a képernyőre kirja az A6H értéknek megfelelő jelet, persze mindig más színnel. Ez érdekes hatást vált ki a képernyőn.

Azt, hogy milyen nagy legyen a kép, befolyásolhatjuk azzal, hogy a memóriában

elhelyezett két lebegőpontos állandót megváltoztatjuk, úgyelve arra, hogy a vízszintes nagyság ne haladja meg a 20-at, a függőleges a 12-t.

A vízszintes konstans 08A5H-tól kezdődően 5 bájti, a függőleges 08AB-től helyezkedik el. Ha kisebb a kép, akkor gyorsabb a változás. A 0857H címen helyezkedik el a kiírandó karakter kódja. Ha ezt megváltoztatjuk A6H-ról például 71H-ra, akkor pöttyöket fog kiírni négyzetálló helyett.

A programot mindenki kiegészítheti saját ízlése szerint. Érdekes hatásokhoz jutunk a különböző változtatásokkal.

AKLI PÉTER

```

:0810 20 A2 B8 20 2B DA 20 CC
:081B BC 6A 20 2E 03 20 B1 98
:0820 06 15 20 0E 09 B4 14 20
:0828 97 0E A5 62 20 98 20 20
:0830 50 08 A5 C5 C9 04 F0 83
:0838 4C 1D 00 6C 02 A0 A9 00
:0840 8D 20 D0 8D 21 D0 20 44
:0848 E5 A9 0C 05 16 A9 14 95
:0850 17 60 19 20 86 E5 A9 A6
:0858 20 D2 FF 10 60 19 A5 17
:0860 65 15 A8 A5 16 65 14 A4
:0868 20 52 08 18 A5 17 65 15
:0870 A8 18 A5 16 E5 14 A4 20
:0878 52 08 A5 17 E5 15 18 A8
:0880 A5 16 E5 14 A4 20 52 08
:0888 18 A5 17 E5 15 A8 18 A5
:0890 16 65 14 A4 20 52 08 60
:0898 29 0F C9 00 F0 83 8D 86
:08A0 02 60 4C 1D 00 65 A0 80
:08A8 00 00 00 00 00 00 00 00
:08B0 00 29 97 E0 20 0C BC A9
:08B8 A5 A0 00 4C 1D 08 20 97
:08C0 E0 20 0C BC A9 A8 A0 08
:08C8 4C 1D 00 00 00 00 00 00
..

```

Az 1986/1. számban megjelent, Megszakítás II. című cikkhez kapcsolódó program a raszter szerinti megszakítás használatára, a grafikus és a karakteres üzemmód keverését teszi lehetővé. Az alsó hat karakteres normál üzemmódban dolgozhatunk, fölülte pedig a nagy felbontású grafikus üzemmódot használhatjuk. A grafikus kép

a memóriában 8 k—16 k, a hozzá tartozó színmemória pedig 2 k—3 k között található. A program hívása: SYS 49152 utasítással történik.

GYURICZA PÉTER

II. gimn. tanuló

Szolnok

# Megszakításvezérelt BREAK-figyelés gé

```

10 *****
20 * GEPI KODU BREAK *
30 * INTERRUPT *
40 * START: 64973 *
50 * RUTIN CIME: )FDFD *
60 * RUTIN VEGE: 65249 *
70 * VECTOR CIME: )FC00 *
80 * RAMTOP: 64511 *
90 *****
100 ADR EQU )FDCD
110 RAMT EQU 64511
120 ORG ADR
130 ENT ADR
140 LD HL,CONT
150 LD IX, )FC00
160 LD B,129
170 PUT LD (IX),H
180 LD (IX+1),L
190 INC IX
200 INC IX
210 DJNZ PUT
220 * CIMTABLAZAT KESZ
230 LD HL,RAMT
240 LD (23730),HL
250 LD (HL), )3E
260 DEC HL
270 POP DE
280 LD SP,HL
290 LD BC, )1303
300 PUSH BC
310 LD (23613),SP
320 LD A, )FC
330 LD I,A
340 IM 2
350 EX DE,HL
360 JP (HL)
370 CONT PUSH IX
380 POP IX
390 POP IX
400 PUSH IX
410 DEC SP
420 DEC SP
430 PUSH AF
440 PUSH HL
450 PUSH DE
460 PUSH BC
470 LD HL,(23672)
480 INC HL
490 LD (23672),HL
500 LD A,H
510 OR L
520 JR NZ,KEY
530 INC (IY+64)
540 KEY PUSH IX
550 CALL )02BF
560 EI
570 POP BC
580 LD A,B
590 AND )C0
600 JP Z,END
610 LD A, )FE
620 IN A, )FE)
630 RRA
640 JP C,END
650 LD A, )7F
660 IN A, )FE)
670 RRA
680 JP C,END
690 LD A,(COUNTA)
700 CP 50
710 JP NC,OFLOW
720 INC A
730 LD (COUNTA),A
740 JP ENDY
750 OFLOW DI
760 LD A,2
    
```

```

770 CALL )1601
780 LD B,6
790 LD IX,TNAME
800 LD A,0
810 LD (FUG),A
820 CYKL LD A, )16
830 RST )10
840 LD A,(FUG)
850 RST )10
860 LD A,19
870 RST )10
880 LD (STBC),BC
890 LD A,(IX)
900 RST )10
910 LD A,(IX+1)
920 RST )10
930 LD A,5B
940 RST )10
950 INC IX
960 INC IX
970 POP BC
980 PUSH BC
990 CALL CONVER
1000 LD B,C
1010 CALL CONVER
1020 LD A,59
1030 RST )10
1040 LD DE,SPACE
1050 LD BC,B
1060 CALL )203C
1070 POP BC
1080 CALL )2D2B
1090 CALL )2DE3
1100 CALL )16BF
1110 LD A,(FUG)
1120 INC A
1130 LD (FUG),A
1140 LD BC,(STBC)
1150 DJNZ CYKL
1160 EI
1170 JR ERROR
1180 CONVER LD A,B
1190 AND )FO
1200 RRA
1210 RRA
1220 RRA
1230 RRA
1240 CALL LOOK
1250 LD A,B
1260 AND )OF
1270 CALL LOOK
1280 RET
C 1290 LOOK ADD A, )30
1300 CP )3A
1310 JR C,SMALL
1320 ADD A,7
1330 SMALL RST )10
1340 RET
1350 STBC DEFB 0
1360 SPACE DEFB " "
1370 DEFB )16
1380 FUG DEFB 0,27
1390 TNAME DEFB "BCDEHLAIXPC"
1400 * REG. KIIRATAS VEGE
1410 ERROR RST )0B
1420 DEFB )14
1430 END XOR A
1440 LD (COUNTA),A
1450 ENDY POP BC
1460 POP DE
1470 POP HL
1480 POP AF
1490 POP IX
1500 RET
1510 COUNTA DEFB 0
1520 DEFB 0
    
```

A program tesztelési segítséget jelent a Spectrum-felhasználók számára: a felhasználói (RAM-ban futó) programok futási ideje alatt figyelni a CAPS—BREAK gombkombinációt, és ezeknek 1 másodpercnél hosszabb ideig történő lenyomása esetén megszakítja az éppen futó programot, kiírja a regiszterek tartalmát, és a rendszer hibakezelő rutinján keresztül visszaadja a vezérlést a felhasználónak.

## Megkötések a felhasználói programra

1. A maszkolható megszakításoknak engedélyeztetnek kell lenniük (nem adható ki a DI utasítás).

2. A 2-es megszakítási üzemmód nem használható, mert ez foglalt a felügyelő rutin számára.

A megszakítás elfogadásakor a képernyő jobb felső sarkába írónak ki a főregiszterek IY kivételével, és PC jelzéssel az utasítászámláló. A kiíratás decimális és hexadecimális alakban történik. A képernyő aló részében megjelenik a

'BREAK INTO PROGRAM' hibáüzenet, hasonlóan a BASIC-BREAK-hez.

## Implementáció

A program csak 48 k változaton fut, nem kompatibilis a 16 k-s gépre. A szűken vett hossza 276 bájt, de a 2-es megszakítási üzemmódhoz szükséges ugrási címtáblázat további fél k-t foglal. Ezt a program az első futtatáskor magának felépíti. A táblázat címe: \*FC00. Szintén az első futtatáskor beállítódik az új RAMTOP érték: 64511. Betöltési és indítási cím: 64973.

## Futtatás

A BASIC-betöltő beírása és futtatása után mentjük ki a programot, majd RANDOMIZE USR 64973. Ezután már él a BREAK-figyelés. Üres gépen, saját programok tesztelése előtt, célszerű egy LOAD ...CODE: RANDOMIZE USR 64973-mal indulni, így már nem kell félni a ciklusba eséstől. A rutin felügyelete a következő NEW utasításig tart, nem szükséges tehát minden használat után újraindítani.

## Futtatási megkötések

1. A RAMTOP fölötti kotorászás rendszerösszeomláshoz vezet(het).

2. Minden NEW utasításnál újra kell indítani a rutint.

NAGY PÉTER

## 5dű programokhoz

A BASIC-betöltő

```
1000 DATA 33,253,253,221,33,0,25
2,6,129,221,116,0,221,117,1,221,
35,221,35,16,244,33,255,251,34,1
78,92,54,62,43,209,249,1,3,19,19
7,237,115,61,92
1005 DATA 4813
1010 DATA 62,252,237,71,237,94,2
35,233,221,229,221,225,221,225,2
21,229,59,59,245,229,213,197,42,
120,92,35,34,120,92,124,181,32,3
,253,52,64,221,229,205,191
1015 DATA 6305
1020 DATA 2,251,193,120,230,192,
202,213,254,62,254,219,254,31,21
8,213,254,62,127,219,254,31,218,
213,254,58,224,254,254,50,210,69
,254,60,50,224,254,195,217,254
1025 DATA 7167
1030 DATA 243,62,2,205,1,22,6,6,
221,33,199,254,62,0,50,197,254,6
2,22,215,58,197,254,215,62,19,21
5,237,67,189,254,221,126,0,215,2
21,126,1,215,62
1035 DATA 5070
1040 DATA 58,215,221,35,221,35,1
93,197,205,162,254,65,205,162,25
4,62,59,215,17,191,254,1,8,0,205
,60,32,193,205,43,45,205,227,45,
205,191,22,58,197,254
1045 DATA 5476
1050 DATA 60,50,197,254,237,75,1
89,254,16,183,251,24,49,120,230,
240,31,31,31,31,205,179,254,120,
230,15,205,179,254,201,198,48,25
4,58,56,2,198,7,215,201
1055 DATA 5632
1060 DATA 0,0,32,32,32,32,32,22,
0,27,66,67,68,69,72,76,65,70,73,
88,80,67,207,20,175,50,224,254,1
93,209,225,241,221,225,201,0,0,0
,0,0
1065 DATA 3515
1070 LET S=64973: FOR I=1 TO 7
1080 LET C=0: FOR K=1 TO 40: REA
D A: POKE S,A: LET C=C+A: LET S=
S+1: NEXT K
1090 READ A: IF C<>A THEN PRINT
"HIBAS ADATSOR:"; (I-1)*10+1000:
STOP
1100 NEXT I: PRINT "MINDEN O.K."
1110 REM SAVE "BREAK?"CODE 6497
3,276
1120 REM *Nagy Peter*
```

# A NOVOTRADE RT

az alábbi szoftverek

bemutatóját tartja

a Hotel Rege

„KÁRTYA” termében  
(Bp. II. ker., Pálos u. 2.)  
de. 10 órától,  
melyre  
minden érdeklődőt  
szeretettel várunk!

### IX. 1.

Főkönyv- és folyószámla-  
könyvelési rendszer  
bemutatása

### IX. 2.

Készletgazdálkodási  
rendszer

### IX. 3.

Bér- és munkaügyi rendszer  
Személyzeti rendszer

### IX. 4.

Alap-, illetve fejlesztői  
rendszerek  
és lokális hálózatok  
IBM kompatibilis  
gépekből kialakítva

### IX. 5.

Állóeszköz-gazdálkodás

**NOVOTRADE**

C64 tulajdonosok  
figyelmébe!

# EXPANDER—64



Elsősorban a könnyű gépkezelést, a hatékony programírást támogatja 21 új BASIC utasításával. Alkalmazásakor a funkciógomboknak előre definiált jelentése van, mely tetszőlegesen módosítható a felhasználó által. Így egyetlen gombnyomással aktivizálhatunk egy utasítást.

## Az EXPANDER—64 bővítménnyel kiegészített C64 összes hibaüzenete magyar nyelvű!

- **CATAL parancs:** megjeleníti a diszk katagólusát a képernyőn, anélkül, hogy a tárban levő program törlődne.
- **FINC parancs:** a programban megkeresi egy adott karaktersorozat előfordulási helyét.
- **MAX, MIN függvény:** vektorváltozók maximumát, illetve minimumát adja vissza.
- **WPOKE utasítás:** kétbyte-os szám memóriába írására szolgál.
- **WPEEK függvény:** kétbyte-os szám olvasása.
- **AT utasítás:** kiíratás kurzorpozicionálással.
- **MSAVE parancs:** kezdő és végcímével adott memóriataromány (vagy program) diszkre, kazettára másolására szolgál. Monitorprogram helyett használható.
- **TRACE parancs:** a futó program végrehajtódásának követésére szolgál. A TROFF paranccsal megszüntethető.
- **APPEND parancs:** a programhozzátöltés tárban levő programhoz. Olyan felhasználók számára előnyös elsősorban, akik előre megírt szubrutinokat, programrészeket szeretnének a programjukhoz szerkeszteni.

segítségével egy karaktersorozat bármely betűjelét módosíthatjuk, felesleges értékdadás nélkül.

- **POS utasítás:** visszaadja egy tetszőleges karaktersorozat előfordulási pozícióját egy másik karakterláncban.
- **DUMP parancs:** begépeléskor kiírja az összes addig használt változó pillanatnyi értékét.
- **HCOPY utasítás:** a teljes képernyő tartalma a printerre íródik.
- **GCOPY utasítás:** EPSON printerre kinyomtatja a nagyfelbontású grafikus képernyő tartalmát 320 x 200 pont méretben. Nagyfelbontású grafikát támogató utasítások:
  - **GIN, GOFF:** grafikus képernyő be- és kikapcsolás
  - **GCLEAR:** grafikus képernyő törlése
  - **GCOL:** grafikus képernyő színbeállítása
  - **GLOT:** koordinátájával meghatározott grafikus képpont kigyújtása

Forgalmazza az  
Alkotó Ijtáság Egyesülés  
Számítástechnikai Iroda  
Budapest V., Garibaldi u. 2.  
Levél cím: 1519 Budapest,  
Pf. 330.  
Telefon: 112-666  
Telex: 22-7272.



**TPA  
felhasználóknak  
ajánljuk!**

# MULTIFMS, TÖBBTERMINÁLÓS FORMÁTUMKEZELŐ RENDSZER

A MULTIFMS rendszer célja olyan software-környezet biztosítása a TPA felhasználó számára, amelyben könnyen készíthet Basic—Plus 2, Fortran—IV, illetve Fortran—IV—Plus többfelhasználós adatgyűjtő, rögzítő, lekérdező programrendszer(ek)eit.

## A MULTIFMS HASZNÁLATÁNAK ELŐNYEI:

### 1. Egyetlen program több terminált szolgál ki.

A DOS—RV/OS—RV/E/ operációs rendszer alatt egy program (task) annyi példányban fut (foglalja a helyet a memóriában), ahány terminálról használják. Mivel a rögzítés, lekérdezés általában egy nagy számú terminállal egyidőben történik, software-segítség szükséges ahhoz, hogy az ilyen típusú feladatok megoldhatóak legyenek. A MULTIFMS rendszer max. 16 terminál kiszolgálását vállalja egyetlen felhasználói programpéldánnyal.

### 2. Formátumvezérelt képernyőkezelés.

A MULTIFMS formátumokkal dolgozik. A formátum egy képernyőterv jellemzőit tartalmazza tömör formában, elkészítése interaktív formátumszerkesztővel történik. A bevitt elemi adatokon formai és tartalmi ellenőrzéseket végez el, és az ellenőrzés eredményétől függően vezérli a további adatbevitelt. A MULTIFMS rendszer a képernyőtervezési fázistól kezdve a kitéltés vezérlésén át az elemi ellenőrzésekig hatékony segítséget nyújt a felhasználóknak, nagymértékben csökkentve az ilyen típusú rendszerek készítéséhez szükséges időt. Mivel a képernyőkezelést a rendszer végzi, a felhasználói rendszerek képernyőkezelése egységes lesz, így a különböző rögzítéseket végző operátorok betanítása is leegyszerűsödik.

### 3. Könnyen kezelhető rendszer.

A terminál kezelőjének nem kell ismernie az operációs rendszer bejelentkezési eljárását. A rendszer indítása központilag (pl.: operátori konzolról) történik. A MULTIFMS kapcsolja (attach-el) a számára kijelölt felhasználói terminálokat. Azokat a terminálokat, amelyek csak a MULTIFMS-sel fognak kommunikálni, slave (szolga) állapotba lehet tenni, ezek csak a MULTIFMS futása idején fognak működni. Különösen távoli termináloknál működő lekérdező rendszereknél hasznos ez a tulajdonság. A nem slave terminálok dinamikusan megszüntethetők (újrateremthetők) a MULTIFMS-sel való kapcsolatot.

### 4. Dinamikus memóriahely-foglalás.

A MULTIFMS rendszer vezérlő részének (MLTFMS) memóriahelye 10-12 kszó között van. Ez az állandó helyigény. A vezérlő rész a felhasználói terminálról érkező input alapján indítja, illetve kapcsolja az adott inputtal foglalkozó felhasználói programot, és a terminál számára memóriaterületet (ún. dynamic regiont) foglal le, amelybe majd a terminálhoz kapcsolható adatok kerülnek. Ennek a területnek a mérete a felhasználói program szükségletének függvénye. Ha egyidőben több terminálról érkezik azonos típusú (ugyanannak a felhasználóknak szóló) input, azok feldolgozása változatlanul egyetlen programpéldánnyal történik, csak valamennyi terminál számára létrejön egy-egy region.

### 5. Memória- és CPU-időtakarékos rendszerkialakítás.

Az elindított, de éppen használaton kívül levő felhasználói programok nem terhelik a CPU-t, szükség esetén checkpointolhatóak, a memóriába csak akkor kerülnek újra vissza, ha ténylegesen szükség van rájuk. Maga a vezérlő program is többnyire checkpointolható, azaz tologatható a memóriában.

### 6. Rugalmas felhasználói rendszer kialakítási (bővítési) lehetőség.

Egy felhasználói rendszer a feladat természetéhez illeszkedő struktúrában egy vagy több programmal valósítható meg. A rendszernek lehetnek egymástól független (párhuzamos) programjai, valamint hierarchikus részei is.

### 7. Több többterminálós rendszer párhuzamos futtatásának lehetősége.

A MULTIFMS-t több különböző program egyidejűleg használhatja, azaz több különböző rögzítési, lekérdező feladat végezhető el a gépen egyidejűleg.

### 8. Funkciók megvalósítása szubrutinhívásokkal.

A formátumkönyvtárban tárolt formátumok és egy szubrutinyűtemény segítségével a felhasználói programok a képernyőkezeléssel kapcsolatos tevékenységeket egyszerűen szubrutinhívásokkal valósítják meg.

### 9. Könnyű és gyors formátumtervezés.

A formátum elkészítéséhez nincs szükség előzetes képernyőtervre, sem pedig speciális formátumleíró nyelvre. A formátum elkészítése interaktív formátumszerkesztővel történik (FED).

### 10. Formátumfüggetlen programkészítés.

Az elkészített formátumhoz rendelt értékek (sorszám, oszlopszám, kilárandó szöveg, kezdőérték, igazítás, feltöltés stb.) nem függenek a felhasználói programtól, így azok módosításakor nem kell a programot megváltoztatni (újraépitni).

A MULTIFMS csak lehetőséget teremt hatékony többterminálós rendszerek felépítéséhez. A tényleges hatékonyság nagymértékben függ a felhasználói programok megírásai módjától, a funkciók megfelelő módon való széttagolásától, a felhasználói rendszer file-jainak szerkezetétől és a hozzáférési módtól.



**Forgalmazza az  
Alkotó Ifjúság Egyesülés**

**Számítástechnikai Iroda  
Budapest V., Garibaldi u. 2.**

**Lévélcím: 1519 Budapest, Pf. 330.**

**Telefon: 112-666**

**Telex: 22-7272**

# PTA-4000

## Csatlakozók

A rendszer ismertetését a gép belsejében található két csatlakozósáv vonalainak tárgyalásával fejezzük be.

A 40 vonalas csatlakozó vezetékeit az 1. táblázat tartalmazza. A jelölések:

- 1 — tápfeszültség
- 2-3 — kiválasztó
- 4-5 — címkijelölő (0000H-3FFFH ill. 6000H-6700H)
- 6 — kiválasztó a WAIT-jelhez
- 7-14 — adatbusz
- 15 — letiltás
- 16-19 — címkijelölő (4800H-4FFFH, 5000H-57FFH, 5800H-5FFFH, 8000H-BFFFH)
- 20-21 — tápfeszültség
- 22-37 — címbusz
- 38 — kimenetletiltás
- 39 — olvasás/írás
- 40 — tápfeszültség

A 60 vonalas csatlakozót a 2. táblázat összegzi. A jelölések:

- 1-8 — címbusz
- 9-10 — üres
- 11-12 — tápfeszültség
- 13-14 — üres
- 15-16 — kiválasztó
- 17-24 — adatbusz
- 25 — letiltás
- 26 — külső WAIT-jel
- 27 — kazettabemenet
- 28 — WAIT-állapot bemenet
- 29 — kazettakimenet
- 30 — megszakításkérés
- 31-38 — címbusz
- 39-40 — üres
- 41-48 — tápfeszültség
- 49 — üres
- 50 — tápfeszültség-kimenet
- 51 — órajel
- 52-55 — tápfeszültség
- 56 — mint 6
- 57 — olvasás/írás
- 58 — mint 6, csak az ME1-lapra
- 59 — az ME1-lap kiválasztó
- 60 — mint 38

## Szoftver

A szoftver ismertetését a rendszerveremben tárolt állandók bemutatásával kezdjük. Ezek a 786BH-7B0CH címterományban találhatóak. A címeket és a tárolt állandók jelölését a 3. táblázat tartalmazza. Az egyes jelek magyarázata:

- külső „beep”, ki/be kapcsoló,
- minden kétbájtos tárolóban az első a magasabb helyiértékű.

A következő részben a BASIC sajátosságaival foglalkozunk. A gépi kódú programozáshoz szükséges segédlet, a mikroprocesszor kódtáblázata a HCC-nél önköltségi áron kapható.

### 1. táblázat

1	VCC
2	PV
3	PU
4	Y0
5	S4
6	DME0
7-14	D7-D0
15	INHIBIT
16-18	S1-S3
19	Y2
20	VGG
21	GND
22-37	AD15-AD0
38	OD
39	R/W
40	GND

### 2. táblázat

1-8	AD7-AD0
9	PB0
10	PC7
11-12	VCC
13-14	NC
15	PV
16	PU
17-24	D7-D0
25	INHIBIT
26	WEX
27	CMTIN
28	W1
29	CMTOUT
30	INT
31-38	AD8-AD15
39	PB1
40	NC
41-42	VCC
43	F.GND
44-48	VBAT
49	NC
50	BFO
51	ΦOS
52-55	GND
56	DME0
57	R/W
58	DME1
59	ME1
60	OD

### 3. táblázat

786BH-	RMT/BEEP	788EH-	TRACE CONDI- TION
7871H-	WAIT Y/N (WAIT-0, WAIT0-3, WAIT1-2)	788FH-	OUTPUT BUF- FER POINTER
7872H-	7873H- WAIT COUN- TER	7890H-	FOR POINTER
7874H-	7874H- CURSOR ENAB- LE (1)	7891H-	GOSUB POIN- TER
7875H-	7875H- CURSOR POIN- TER (0-155)	7894H-	STRING BUF- FER POINTER
787DH-	787DH- BLINK CHA- RACTER	7895H-	USING F/F
787EH-	787EH- BLINK CUR- SOR	7896H-	USING M
788DH-	788DH- TRACE	7897H-	USING §
		7898H-	USING m
		7899H-	789AH- VARIABLE PO- INTER
		789BH-	ERL
		789CH-	789DH- CURRENT LI- NE
		789EH-	789FH- CURRENT TOP
		78A0H-	78A1H- PREVIOUS ADDRESS
		78A2H-	78A3H- PREVIOUS LI- NE
		78A4H-	78A5H- PREVIOUS TOP
		78A6H-	78A7H- SEARCH ADD- RESS
		78A8H-	78A9H- SEARCH LINE
		78AAH-	78ABH- SEARCH TOP
		78ACH-	78ADH- BREAK ADD- RESS
		78AEH-	78AFH- BREAK LINE
		78B0H-	78B1H- BREAK TOP
		78B2H-	78B3H- ERROR ADD- RESS
		78B4H-	78B5H- ERROR LINE
		78B6H-	78B7H- ERROR TOP
		78B8H-	78B9H- ON ERROR ADDRESS
		78BAH-	78BBH- ON ERROR LI- NE
		78BCH-	78BDH- ON ERROR TOP
		78BEH-	78BFH- DATA POINTER
			79D1H- OPN DV
		79E0H-	79E1H- USER COUN- TER X
		79E2H-	79E3H- USER COUN- TER Y
		79E4H-	79E5H- SCISSORING COUNTER Y
		79E6H-	79E6H- ABSOLUTE PO- SITION X
		79E7H-	79E8H- SCISSORING COUNTER X
		79EAH-	79EAH- LINE TYPE
		79EBH-	79EBH- DOT LINE CO- UNTER
		79ECH-	79ECH- UP/DOWN
		79EDH-	79EDH- X MOTOR HOLD CO- UNTER
		79EEH-	79EEH- PORT C
		79EFH-	79EFH- Y MOTOR HOLD CO- UNTER
		79F0H-	79F0H- GRAPH/TEXT
		79F2H-	79F2H- ROTATE
		79F3H-	79F3H- COLOR
		79F4H-	79F4H- CSIZE
		79FFH-	79FFH- LOCK
		7B00H-	7B07H- RND NUMBER
		7B0AH-	7B0CH- AUTO P-OFF COUNTER

# PTA—4000 Fedélzeti számítógép az autóban

A jövő század számára elképzelt szériakocsikban, de már az autószaalonok legújabb csodáiban is jelentős szerepet kapnak a szilíciumlapkák, a chipek felhasználásával kialakított különböző célú elektronikus készülékek, amelyek közül néhány már valóban megérdemli a fedélzeti számítógép elnevezést. Ha a fantáziát nem korlátozzák gyakorlati és gazdasági megfontolások, akkor a gépkocsiba minden beépíthető, és az autó alkalmas arra is, hogy a kívánalmaknak megfelelő guru elektronikus laboratóriumot alakítsunk ki benne. Az azonban nyilvánvaló, hogy a gépkocsi nem mindenben tekinthető optimális helynek a vezetéskötést és üzemellenőrzést segítő vagy egyéb elektronikus készülékek széles körű alkalmazására. Erre előző példaként felhozni, hogy a ma már több évtizedes alkalmazási tapasztalatok ellenére sem általánosan a magyar utakon futó szériakocsikban az elektronikus gyújtás és az érintkezésmentes löttyeszabályozás készülékek, csupán ezeket említtve a már megoldott eljárások közül.

A gépkocsiban alkalmazható elektronikus berendezéseket két csoportra oszthatjuk. Az egyik csoport készülékei közvetlenül a motorhoz vagy a fékrendszerhez kapcsolódva irányítástechnikai feladatokat oldanak meg. Ide tartoznak a blokkolásmentes fékrendszerek elektronikai, a benzinfecskendezés szabályozói, a gyújtásszabályozó és -megszakító rendszerek. A másik csoport készülékei az ellenőrző funkciót látják el, és kiegészítő információkat szolgáltatnak a vezető vagy üzemeltető számára. Most ennek az utóbbi csoportnak a készülékeivel foglalkozunk, hiszen jóval többen lehetnek olvasóink között olyanok, akik meglévő gépkocsijukat kívánják jobb hatással üzemeltetni, és elérhetik, hogy személyiszámítógép-beszerzésüket ne a család színes videójáték-igénye, hanem a hálózati feszültségtől független és gépkocsiban történő alkalmazhatóság befolyásolja.

Gépkocsikba a műszerezés kiegészítésére különböző funkciójú, inkább díszítésre alkalmas készülékek vásárolhatók, amelyekre általában jellemző „az erdőben elvesztett tüt a tábornőtől kereső kiscserkész esete”: nem a szükséges, hanem az egyszerűen



mérhető paraméterek jelzését biztosítják. Van, aki a fordulatszámérőre esküszik, mások a szívótorokban mérhető nyomást favorizálják. A témában elmélyedők azonban igen hamar tapasztalják, hogy a pillanatnyi üzemanyag-fogyasztást mérő érzékelő nélkül nem sokra lehet menni, még számítógépes feldolgozással sem.

Cikkünknek az ad aktualitást, hogy hazai gyártásból megfelelő számítógép és az NDK-ban gyártott, az újabb Wartburg gépkocsikba már be is szerelt üzemanyagszenzor rendelkezésre áll.

## Miért előnyös?

A PTA—4000 számítógépből kialakított, a Wartburg szenzort tartalmazó rendszer leírása előtt tekintsük át azokat a feladatokat, amelyek többlétszolgáltatásként elvárhatók egy viszonylag kicső szériakocsiba utólag beépíthető, fedélzeti számítógéptől: — mutassa az indulástól megtett utat km-ben, legalább 10 méteres felbontásban, — mérje és jelezze az út során elfogyasztott üzemanyag mennyiségét,

— tegye meg ugyanezt a tankolási időponttól, illetve egy kiválasztott kezdőponttól számítva,

— adjon információt egy adott indulási ponttól az átlagsebességre vonatkozóan, — jelezze a pillanatnyi futásra jellemző fogyasztást 100 km megtett útra vonatkoztatva,

— jelezze a tankban még rendelkezésre álló üzemanyag mennyiségét,

— jelezze, hogy ezzel az üzemanyagmennyiséggel, alapul véve az utolsó 100 km

megtett út fogyasztását, még milyen távolságot futhatunk be,

— rendelkezzen az átlagos vezető látási viszonyaihoz alkalmazkodó digitális és analóg kijelzési lehetőséggel,

— szolgáltassa a pontos időt,

— adjon lehetőséget arra, hogy a vezető kívánsága szerint, értékelhető formában jelezze a kívánt funkciók végeredményét.

A sort még folytathatnánk különböző igényekkel, de mutatóba ennyi is elég. Teljessé tehetjük a felsorolást azzal, hogy megkérjük a számítógéptől, hogy a fel nem sorolt újabb funkciókat szoftverúton tegye lehetővé. Ilyen lehet az analóg kijelzés, ahol egy megengedett érték túllépése módosítva jelenik meg, az előre programozott üzemanyag-fogyasztás vagy út-idő diagrafomtól való eltérés jelzése stb.

## Mi kell hozzá?

Az igényeink teljesítéséhez minimálisan három bemenőjel szükséges az ideálap, az üzemanyagszenzor és a pillanatnyi üzemanyag-fogyasztás mérésére, továbbá egy út-távadó. Számítógépként olyan eszköz jöhet szóba, amelynek fogyasztása a gépkocsi álló állapotában az akkumulátor önkisülésének nagyságrendjébe esik, és az igényeknek megfelelő kijelzést produkálja, valamint elhelyezhető a gépkocsiban, kellő rálátást adva a vezető és esetleg az utas számára is.

A jámbor óhajon, külföldi napidíjak számlálgatásán és a szakirodalom tanulmányozásán felül néhány éve véletlen szerencsés folytán a Dobi István úti autópiacon találok meg egy eladásra kínált originál cso-

magolású autó-számítógéppel, amelyhez üzemanyagmérő is tartozott. Tulajdonosa, aki valami egyszerű eszközt várt ajándékként a kellően felcsomózott kocsijába, igyekezett túladni rajta, miután túlságosan bonyolultnak találta a készüléket és a beszerelési és üzemeltetési utasítást is, ami természetesen csupán angol nyelven állt rendelkezésre. Így a készüléket igen jutányos feltételekkel (1 Ft=1,2 osztrák schilling) sikerült megvásárolnom. Részletes szétszedéssel együtt járó hardvervizsgálat után, amit szerencsésen túlélt, üzembe is állítottam a számológépet. Ez a Lada 1200 típusú gépkocsimba épített készülék a felsorolt igények zömét részben teljesíti. Már több mint két éve használom, és eltekintve a szerviz utáni újrabehatározások kényelmetlenségeitől, ami az akkuszerelvénél történő teljes lekapcsolása miatt szükséges, a készülék hibamentesen működik, és a szolgáltatott adatok alapján bármikor elődönthetem hosszabb vidéki utakon, hogy hány forintomba kerül, ha 10 perccel hamarabb érkezem meg. Ennél lényegesebb azonban az, hogy kiválasztott útszakaszok fogyasztásait figyelve elég pontos információt kapok a motort és a karburátort megfelelő beállításáról, az utánállítás szükségességéről.

A készülék három egységet tartalmaz. A bolygófogó üzemanyagszenzor az AC-pumpa fölé beépített, és az átfolyt üzemanyag-mennyiséggel arányos számú impulzust ad. Az útávadó az alvázhoz erősített vasmagos tekercs, melynek mágneses terét a kardanra ragasztott mágnesek befolyásolják. A harmadik egység egy 5 dekádós plazmakijelzőt és 4 bites NS-processzort tartalmazó mikrogép.

## Az üzemanyagszenzor

Érzékelője egy körpályára kényszerített golyó, amelyet az érintőlegesen bevezetett üzemanyag egy része magával ragad és keringtet. Az üzemanyag kivétele a bevezetésnél szinte azonos ponton, a pályára merőlegesen történik. A fekete műanyag golyó egy, a szenzorba épített infrakaput zár, aminek érzékelőjéről impulzusokat kapunk. A készülék a leírás szerint gyárilag nincs hitelesítve, és az előírás az, hogy a tankolásnál betöltött mennyiségekkel kell a kalibrációját korrigálni, amennyiben arra szükség van. A 30 literes tankolásoknál általában 1 literes eltéréseket tapasztaltam a készülék által mutatott és a benzinkútnál betöltött értékek között. Ez a töltőállomásonál megkívánt pontossági osztályhoz közel álló eredmény. A szerkezet kialakítása olyan, hogy a benzin útját semmilyen formában sem tudja elzárni, hiszen a golyó átmérője kisebb, mint a körpályáé.

A megtett utat mérő szenzor a kardan-tenyegre erősített 4 mágnesből és az alvázhhoz csavarozott tekercsből áll. Itt is szoft-

verúton történik a kalibráció, és azt 2 vagy több kilométer lefutott távolság alapján lehet elvégezni.

Az akkora állandóan rákötött tápvezeték és a szenzorok bemenővezetékein felül még a gyújtás- és világításkapcsolóról van csatlakozás, amely arra szolgál, hogy a készülék plazmakijelzőjének csak üzem közben adjon tápellátást, és fényerejét csökkentse az esti vagy éjszakai vezetésnél.

## A kijelző

Öt decimális helyértékből áll, tizedes-ponttal ellátva, ami csak értékeket közöl, így emlékezni kell arra, mi volt a készülékhez intézett utolsó kérdés. A kérdések az előlapon rendelkezésre álló nyomógombokkal aktiválhatók. A beállításához takart nyomógombok állnak rendelkezésre. A kezelési utasítás többször figyelemztet arra, hogy beállítani csak álló gépjárműben szabad, hiszen vezetni és programozni egy időben veszélyes művelet, bár itt a programozás csak a távolságrészékelő és benzinmérő szenzorának állandóit, valamint az idő beállítását jelenti.

A készülék az eredetileg programozott funkciókon felül továbbiak elvégzésére nem alkalmas. Ezek az 1-6 és 9 pontnak felelnek meg. A kijelző napűtésben nagyon nehezen olvasható, és a pillanatnyi fogyasztásnál a készülék nem veszi figyelembe a gyorsulásra fordított energiát. Erre akkor van szükség, ha a motor hatásfokára kívánunk következtetni.

A készülék szenzorai, amelyeket a Ladába beépítettem, már lehetőséget adtak arra, hogy egy kis fogyasztású számítógéppel a kívánóságaimnak megfelelő rendszert alakítsunk ki. A rendelkezésünkre álló több lehetőség alternatíva közül (HP-41cx IL csatlóval, CASIO PB700, CASIO FX-802P, HD-75, HP-71B+IL, SHARP-PC1500A) többet részletesen vizsgáltunk. A leggyorsabb eredményt a HP-gépek alkalmazása ígerte, mivel rendelkezésre álltak a HP-IL csatlóhurokhoz illesztett számlálók. A szenzorokról kapott impulzusokat ICM7227 típusú számlálókkal megszámlálva, a számlálókat HP-82166 moduldal az IL hurokhoz csatlóval kialakítottunk egy olyan elrendezést, amely a feladatnak megfelelően működött, és igen egyszerű, BASIC-ben elvégezhető programozási lehetőséget adott.

A kísérlethez a forgószárnyas vízbeszégmérőkhöz hitelesítésére kialakított berendezést alkalmaztuk néhány napig. A HP-gépekhez sajnos nem sikerült megszerezni a gépi nyelvű programozáshoz szükséges eszközöket, így a BASIC-ben megírt programok az eredeti készüléknél valamivel lassabban működtek, de lehetőséget adtak a HP-82161 típusú kizárás tároló alkalmazására, amire kb. 100 kb-ot rögzíthető. Mi-

után egy ilyen elrendezés ára magasabb, mint a gépkocsié, gyakorlati felhasználása, eltekintve néhány tesztelési feladattól, illuzorikus.

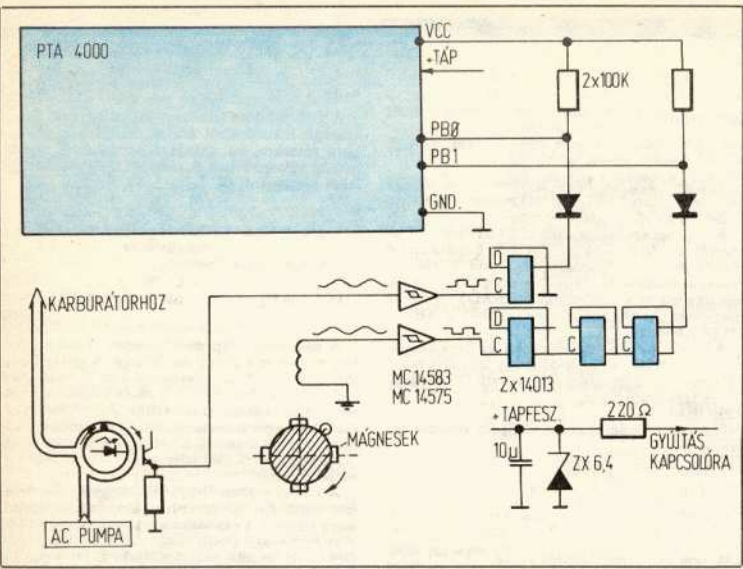
## Hazai géppel

Ezután a SHARP-PC1500A géppel kezdtünk hozzá az ismertetett berendezés szolgáltatásainak bővítéséhez és a felsorolt specifikációk teljesén megfelelő rendszer létrehozásához. A munka kezdetén még nem tudtuk, hogy a PC1500A-nak megfelelő, de annál nagyobb memóriakapacitású gép, a PTA-4000+16k hazai gyártásból is rendelkezésre fog állni, és így ez a munka az egyéni autósámítógép-hobbin túlmenően jelentőséget is kaphat. A SHARP-gép vásárlásában igen jelentős volt az a tényező, hogy sikerült hozzájutni a gép belső dokumentációjához, ami lehetőséget adott arra, hogy ne az IL csatlókon keresztül, hanem közvetlenül kapcsoljunk két hardverszámlálót egy HP-géphez. Az első néhány összeállításnál itt is az IL hurkon keresztül vetjük igénybe a gépet.

Csak ezután következett a szoftverszámláló kialakítása. Az üzemanyagszenzor jele a PB0, az útadó jele a PBI csatlakozópontra került. A védelem és az egyszerűbb programozhatóság érdekében a jeleket egy 4013 típusú CMOS FF áramkörrel leosztottuk. Így már lehetőség van arra is, hogy felhasználjuk a gép kernelijének rutinjait, ami igen előnyös egy kísérleti összeállításban.

A PTA-4000 gépkocsiban történő próbája közben szükségessé vált, hogy a gépet ne csak fedélzeti számítógépként, hanem eredeti funkciójának megfelelően, zseb-számítógépként is felhasználhassuk. Ehhez azonban további dekórt, ROM és esetleg RAM elemet tartalmazó hardver kell. A gép rendszerének ismeretében a feladat bizonyos korlátokkal szoftverúton is megoldható, ha a műszerállandókat a gép által nem használt, pontosabban a második karakterkészlet számára fenntartott memóriaterületen helyezjük el. Ez a megoldás azonban korlátozottan tűnt, így inkább 2 k EP-ROM és RAM alkalmazása mellett döntöttünk, amely a H8000 címre helyezhető el. Ha a gép fő funkciójának a gépkocsiban történő üzemeltetést tekintjük, akkor erre a hardverre természetesen nincs szükség.

Az alkalmazott gépi kódú program folyamatosan visszatér a ME1 HF00F címen jelentkező bemenőpontok olvasására, és impulzust jelez, ha elérte a beolvasott jelek az előző ciklusban beolvasottól. Ezután arra van szükség, hogy a programot a valós idejű operációnak megfelelően építsük fel úgy, hogy a mintavételek közé beférjenek a kívánt funkciók. A szoftvermunkát igen egyszerűsítette volna a lapban már ismertetett MC 14534 típusú számláló alkalmazása az IFSP illesztő felhasználásával, de ér-



rutinokat alakíthatunk ki, amelyekkel megakadályozhatjuk, hogy nagyobb programjainkat megszakításokkal befolyásolják, erről azonban helyhiány miatt most nem tudunk részletes információt adni.

**Felhasználói programok**

A PTA—4000-hez készített felhasználói program-variációink mind számban, mind effektív bájthosszban egyre inkább duzzadtak, és kezelésük egyre lassúbbá vált a magnetofonkazetta adathordozóként történő alkalmazása miatt. A gyorsabb munka érdekében az IFSP használatával egy RS232 interfészen keresztül a PTA—4000-et C64 rendszerrel működtettük együtt, ami némileg megkönnyítette a már elkészült gépi rutinok visszakeresését az 1541-es floppyról. Ekkor adódott az az ötlet, hogy ki kellene hagyni a C64-et, az IFSP illesztőt és természetesen a C64 RS232 csatlóját is, megtápnitva a PTA—4000-et a C64 soros buszára és így a C64-hez kapcsolt perifériáknak a kezelésére. A munka sikerrel járt, és a KA 160 illesztőt úgy módosítottuk, hogy a kazettás magnetofon REMOTE csatlakozót felhasználva, a C64-hez a soros IEC buszon csatlakozó perifériákat kezelni tudjuk.

Miután sikerült, megírtuk és teszteltük a legkritikusabb, a működést bizonyító programrészeket. A Híradástechnika Szövetkezet rendelkezésünkre bocsátott egy PTA—4000+16 gépet és egy Wartburg taxiba építettünk be. A Wartburgban üzemanyag-szenzor rendelkezésre áll, és a leplombált taxióra hitelen bizonyíthatja a berendezés üzembiztonságát. A taxiban elhelyezett gép nyomtatóval együtt üzemel, és a szenzorok csatlakoztatására a floppyillesztéshez kialakított soros busz DATA és CLOCK vonalait alkalmazzuk. Ennek megfelelően természetesen módosítanunk kell a már elkészített programjainkat, de addig is értékes információkkal gazdagodunk arra nézve, hogy a KA 160 nyomtató és a PTA—4000 hogyan viselkedik egy vibrációnak erősen kitett környezetben.

A PTA—4000+16 géppel megoldhatók a menetdiagram-regisztrálások, és az adatok floppyra vagy akár kazettára olvasása lehetőséget ad központi ellenőrző rendszerek kialakítására is, az eredetileg általunk a HP-gépekkel megvalósított műszaki paraméterek mellett. Ehhez azonban még igen jelentős szoftvermunkát kell elvégeznünk.

A hobbibil megkezdett feladatunk elvégzésében jelentősen túllentünk azon az eredeti célkitűzésen, hogy napfényben is le tudjuk olvasni a ZONIC plazmakijelzőjét, és egy gyorsalással kompenzált fogyasztási rutint alkalmazzunk a már beépített szenzorokhoz.

PÁLÓS LÁSZLÓ—ARDAY ZSOLT

**TANKOLÁS**  
 FOGYASZTÁS 27.31lt  
 MEGTETT UT 243.7km  
**ÁTLAG 11.21l**  
 1km min 6.85lt max 17.85lt/100km-utrá  
 ELŐZŐ TANKOLÁS IDEJE 32515.42  
 DATUM 52612.38  
 UJRAKALIBRALJAK? I  
 HANY lt.t TANKOLT? 29  
 HIBAHA TÁRON BELÜL  
 \*\*\*\*\*  
 FUTASI PARAMÉTEREK  
 INDÍTÁS 52608.0210  
 INDULÁS 52608.0240  
 MEGÁLLT 52608.2952  
 LEFUTOTT km 17.43  
 KEVERÉK 1.86lt  
**ÁTLAG 10.67l**  
 +++++ \*\*\*\*\* +++++

kell hogy legyen, mint 5 ms, ami csak igen körültekintő rendszerszervezéssel teljesíthető. Segít a helyzeten egy további 14013 osztóként történő alkalmazása (lásd az ábrát), ami a rendelkezésre álló időt 20 ms-ra emeli. Felmerült annak a lehetősége is, hogy a maszkolható megszakítást használjuk fel a számlálásra, amit az üzemanyag és távolságszenzor jelével aktiválunk. A megoldást azért vetettük el, mert ez a pont nincs kivezetve a csatlakozóra, így nem használható a géphe történő belenyúlás nélkül. A BRK/ON gombbal aktivált megszakítást egyébként felhasználtuk a programban. Használatáról annyit kell tudni, hogy a megszakítás akkor megy a felhasználó által a H79DB,DC memóriahelyen megadott címre, ha előzőleg H55-öt írunk a H79DA-ra. Ezt meg kell ismételni a bekapcsolás után, mert a tápfeszültség megjelenése után végrehajtott RESET rutin törli a H79DA memóriahely tartalmát.

Itt kell megjegyezni, hogy a PTA—4000 POKE utasítása megfelelő háttér-dokumentációk nélkül kellemetlen meglepetések forrása lehet, amit még az ALL RESET sem szüntet meg minden esetben. Annak érdekében, hogy a fejlesztés során fellépő elkerülhetetlen végtelen hurkokból újra feltámasztott gépünkkel azonos helyzetből induljunk újra, egy ÜTHENGER-nek elnevezett gépi programot is készítettünk, amely néhány bajt kivételével nullára írja a teljes tárterületet és kikapcsolja a gépet. Az új bekapcsolás a program lefutása után az ALL RESET-nek megfelelően jelentkezik be. A maszkolt megszakítás felhasználói rutinra történő elágaztatásával különböző

A PTA—4000-es print out listája. A dátum kiírása a TIME funkció szerinti: 5. hó, 26. nap, 12 óra, 3 perc, 8 mperc.

demben vonzónak tűnt szoftverszámlálók kialakítása a feladat megoldásához. A számláló megkívánt mintavételi sebessége 6000/perc = 100/s maximális motorfordulatnál a kárdánra helyezett 4 mágneses és 2-es frekvenciaosztással számolva kisebb

## ...egy könyvet

mely több, mint négyszáz oldalon, nyolc nagyobb, jól megírt, jól strukturált és jól olvasható, kipróbált programot is tartalmaz.

Erről ismét eszünkbe jutott, hogy nem lehet nyelveket tanulni jó olvasókönyvek nélkül. Ez igaz a számítástechnikára, a programozási nyelvekre is. Miller könyve jó olvasókönyv. Nem követei például az e lap hasábjain is meghonosodott rossz hagyományt, hogy a kurzort mozgató karaktereket rosszul olvasható grafikus jelekkel listázza. Ehelyett minden programját valahogy így kezdi:

```
100 rem *** — a program neve — **
110:
120:
```

```
130 rem ** — inicializálás — **
140 home$ = chr$(147) : rem clr/home
      képernyőtörés
```

```
150: :cl$ = chr$(17) : rem kurzor le
160: :cf$ = chr$(145) : rem kurzor fel
170: :cb$ = chr$(157) : rem kurzor balra
180: :cj$ = chr$(29) : rem kurzor
      jobbra
```

```
190: :fe$ = chr$(144) : rem fekete
```

```
240 poke 53280, 14 : rem a képernyő
      széle világoskék
```

```
250 poke 53272, 23 : rem kis- és
      nagybetűs írásmód
```

stb.  
Ennek megfelelően a képernyőt például így kezeli:

```
17000 rem ** — az alprogram neve — **
```

```
17010 print home$ : rem a képernyőt
      töröltem, kurzor a
      bal felső sarokban
```

```
17020 c1 = 2 : rem 2 sorral lefele,
      paraméter a szubrutinnak
```

```
17030 gosub 9000 : rem ez mozgatja c1
      számú sorral lefele
      a kurzort
```

```
17040 print "Ide jön a megjeleníteni
      kívánt szöveg"
```

```
9000 rem ** — a kurzort lefelé mozgató
      szubrutin — **
```

```
9010 for i = 1 to c1
```

```
9020 print c1$ : rem ezzel mozgatom
      a kurzort
```

```
9030 next i
```

```
9040 return
```

A könyv ajánlja, hogy mindenki használja a gép mellé adott TEST/DEMO lemezen található "C64 WEDGE" segédprogramot.

A gép bekapcsolása után a TEST/DEMO lemezről LOAD "C64 WEDGE", 8 parancssal töltjük be a segédprogramot, majd RUN parancssal indítsuk el. Miután a segédprogram bejelentkezett és a képernyőn a READY felirat is megjelent, kivethetjük a meghajtóból a TEST/DEMO lemezt.

A C64 WEDGE segítségével a lemezek kezelése egyszerűbbé válik. Például egy lemezen tárolt PROBA1 nevű program így hívható fel a lemezről és így indítható el:

```
↑PROBA1
Nagyobb programrendszerek pedig például így építhetők fel:
```

```
100 rem *** — a nagyobb programrendszer
      neve —
      ***
```

```
130 rem ** — inicializálás — **
140:
150:
160:
170:
180:
190:
200:
210:
220:
230:
240:
250:
260:
270:
280:
290:
300:
310:
320:
330:
340:
350:
360:
370:
380:
390:
400:
410:
420:
430:
440:
450:
460:
470:
480:
490:
500:
510:
520:
530:
540:
550:
560:
570:
580:
590:
600:
610:
620:
630:
640:
650:
660:
670:
680:
690:
700:
710:
720:
730:
740:
750:
760:
770:
780:
790:
800:
810:
820:
830:
840:
850:
860:
870:
880:
890:
900:
910:
920:
930:
940:
950:
960:
970:
980:
990:
1000:
```

```
500 rem ** — a fő menü
      (a választási lehetőségek) kiírása — **
```

```
770 input "Melyik lehetőséget
      választja?": v
```

```
775 :: rem a v változóba beolvasom
      a kezelő választását
```

```
800 if v = 1 then 1000
```

```
810 if v = 2 then 2000
```

```
1000 rem ** az első választásnak
      megfelelő alprogram hívása — **
```

```
1030 ↑"alprog1": rem a C64 Wedge segítségével
      felhívom és elindítom az első
      alprogramot
```

```
2000 rem ** — a második választásnak megfelelő
      alprogram hívása — **
```

```
2030 ↑"alprog2": rem felhívom és elindítom a
      második alprogramot
```

A Commodore 64-est felhasználók nagy része nem szeret ún. random (vagy C64 "tájszólásban": relative) fájlokkal dolgozni, ahol pedig a lemezen tetszés szerinti sorrendben hozzáférhetne a fájl bármelyik rekordjának bármelyik mezőjéhez. A tartózkodás egyik oka minden bizonyosan az, hogy a C64-en a random (relative) fájlok kezelése első látásra kissé nehézkes, a gép mellé adott leírás pedig olyan, mint egy rejtély.

A C64 random (relative) fájliai egyenlő hosszúságú rekordokból állnak. Minden egyes rekord részére, ún. mezőkre osztható. A mezők hossza változó lehet. A random fájl például így lehet megnyitni:

```
140 open 2,8,15 : rem a parancscsatorna
      megnyitása
150 open 3,8,4 "pelda1,"
      + chr$(50)
152 rem figyelem! A file neve utáni karakter
      kis „l” betű
```

A megnyitott fájl neve "pelda1". A rekordok hossza ebben a példában 50 bájttal. A parancscsatornához a 140. utasításban a 2-es fájlszámot, a "pelda1" random típusú fájlhoz a 150. utasításban a 3-as fájlszámot rendeljük. A további utasításokban a parancscsatornára és a "próba1" fájlra ezzel a két számmal hivatkozunk. A 150. utasítás végén a chr\$(50) adja meg a rekord hosszát, az 50 bájtot.

A 150-es sorban érdekesség, hogy a szekvenciális fájlokkal ellentétben nem kell megadni, hogy írásra vagy olvasásra nyitottuk meg a fájlt.

A fájlba úgy írunk, hogy előbb a parancscsatornán keresztül pozicionáljuk a lemezezségység (1541) író/olvasó fejét a kívánt rekord kívánt mezője fölé. Így:

```
180 print #2, "p": chr$(4): rec$: mez$:
```

A 2-es a parancscsatornára utal. A "p" jelzi, hogy pozicionálunk. A chr\$(4)-ben visszautalunk arra, hogy a 150. utasításban a "próba1" fájlhoz a 4-es csatornát jelöltük ki. A rec\$ egy két karakterből álló füzér (string), mely az alább kifejtett, kissé bonyolult módon, 256-os számszerben ábrázolva tartalmazza a kiválasztott rekord sorszámát.

A mez\$ tartalmazza a rekordon belül azt a bájtot, ahol a kiválasztott mező kezdődik.

Nézzük előbb a rekord megadását:

```
rec$ = chr$(ra) + chr$(rf)
```

ahol ra és rf egy-egy 0-tól 255-ig terjedő szám, és a rekord sorszáma, illetve ra és rf között az alábbi összefüggés áll fenn:

a kiválasztott rekord sorszáma = 256 \* rf + ra

Megjegyzés: rf — felső bájttal; ra — alsó bájttal.

A táblázat néhány példát mutat a rekord sorszámának megadására a pozicionáló utasításban.

Ha a kiválasztandó rekord sorszáma	akkor	
	ra értéke	rf értéke
1	1	0
2	2	0
255	255	0
256	0	1
257	1	1
511	255	1
512	0	2
513	1	2
.	.	.
.	.	.
.	.	.
.	.	.
stb.	.	.

Az átszámítás a sorszám és ra, illetve rf között természetesen gépesíthető. Ha a kiválasztandó rekord sorszáma i, akkor ra és rf értéket így számíthatjuk ki:

```
160 i = ... : rem a rekord
      sorszámanak
      megadása
170 gosub 1000: rem ugrás az ra és rf értékét
      kiszámító szubrutinra
```

```
180 print #2, "p"; chr$(ra);
      chr$(rf); mez$
```

```
1000 rem ** — rekordsorszám
      ki(át)-számítás — **
```

```
1010 ra = i
1020 rf = 0
1030 if ra > 255 then 1050
1040 return
1050 rf = int(ra/256)
1060 ra = ra - 256 * rf
1070 return
```

A mező kezdő bájttját egyszerűen így jelöljük ki: mez\$ = chr\$( <a bájtszám> )  
Tehát például ha a mező a rekord 25. bájttjánál kezdődik, akkor mez\$ = chr\$( 25).

Ha például a 72. rekord első bájttjától kívánánk írni vagy olvasni, akkor a pozicionáló utasítás így nézne ki:

```
180 print #2, "p"; chr$(4); chr$(72);
      chr$(0); chr$(1)
```

Ugyanez a 257. rekord 27. mezőjétől:

```
180 print #2, "p"; chr$(4); chr$(1);
      chr$(1); chr$(27)
```

A pozicionálás után az írás már egyszerű. Például:

```
190 print #3, "ez minden"
```

A 180. utasítással pozicionált helyre ezzel fel is írjuk az „ez minden” szöveget. Ha olvasni akarunk volna például az a\$ változóba, akkor ezt az alábbi utasítással tehetjük volna meg:

```
190 input #3, a$
```

Az alprogram szerkezete olyan lehet, mint a főprogramé. Itt is van inicializálás, itt is lehet menü (almenü). Legyen a főprogram egy leltári nyilvántartó program, melynek egyik alprogramja új tételeket vesz fel a leltárba úgy, hogy előbb kéri az új leltári tárgy nevét, leírását, árát stb., majd a leltárt tartalmazó random fájlhoz fordul, például így:

```
2000 rem ** — alprogram: felírás
      a fájlba — **
2010 :: rem előbb meg kell tudni, hogy eddig
      hány darab rekord van a fájlban. Ezt a számot a program
      célszerűen egy külön szekvenciális fájlban tárolja
      ugyanazon a lemezen. Ezt a fájlt pointer-fájlnak
      hívjuk
2020 : rem * — a pointer-fájl olvasása — *
2030 open 2,8,2, "pointer, s, r"
2040 input #2, ptr: rem a ptr változó most azt
      tartalmazza, hogy eddig hány rekord van a fájlban
```

```
2050 close 2
2060 if status = 16 then 3000
2065 :: rem az előző utasítás azért kell, mert ha
      valaki most kezd először dolgozni a programmal,
      akkor még nem talál „pointer” nevű szekvenciális
      fájlt. A gép st nevű változója tartalmazza
      automatikusan az erre vonatkozó információit.
      A 3000. utasításnál kezdődik az a rutin, mely
      ebben az első esetben létrehozza a „pointer”
      fájlt
```

```
2070 ptr = ptr + 1
2080 :
2090 : rem * — felírás az adatfájlba — *
2100 open 15, 8, 15 : rem a parancscsatorna
      megnyitása
2110 open 3, 8, 3 "leltar,1",
      + chr$(100) : rem
      a rekordok hossza
      100 bájtt lesz
2140 ra = ptr : rem itt használjuk fel a
      szekvenciális fájlból kiolvasott számot
```

```
2150 rf = 0
2160 if ra > 255 then gosub 15000
2165 rem ilyenkor át kell számítani
      256-os számszrendszerbe
2170 rec$ = chr$(ra) + chr$(rt)
2180 :
2190 mez$ = chr$(1) : rem az első
      bájtból irok
2200 print #15, "p"; chr$(3); rec$; mez$
2210 print #3, szöveg$: rem a szöveg $
      változó tartalmazza azt a szöveget,
      melyet a lemezre fel akarok írni
```

```
2380 close 3 : close 15
2390 :
2400 :
2410 rem * — a pointer-fájl tartalmának
      módosítása — *
2420 open 2, 8, 2 "D0: pointer, s, w"
2430 print #2, ptr
2440 close 2
```

```
2470 goto 4000 : rem "akarja folytatni"
      rutin
```

```
4080 rem ha nem, akkor ugrás 6000-re, az
      alprogramot befejező rutinra
```

```
6080 ↑ fő menü program : rem a főprogram
      visszahívása
```

Befejezésül nézzük meg, hogy milyen technikát sugall a könyv akkor, ha a kezelőtől jelzést vár arra, hogy tovább lehet-e menni a feldolgozásban:

```
2510 print "Ha folytatni kívánja, nyomja meg a
      RETURN gombot"
2520 gosub 19000 : rem ugrás a RETURN
      megnyomását figyelő szubrutinra
```

```
19000 rem ** — a RETURN
      figyelése — **
19010 poke 198, 0 : rem a klaviatúra-
      puffert törlése
19020 for i = 631 to 640
19030 poke i, 0 : rem kiürítés
19040 next i
19050 x = peek (197)
19060 if x = 1 then 19080 : rem ha megnyomták
      a RETURN gombot
19070 goto 19050 : rem még nem
      nyomták meg
19080 poke 198, 1 : rem kurzor
19090 poke 631, 0 : rem a klaviatúra
      törlése
```

```
19120 return
```

Szóval olvastunk egy jó könyvet. Mindenki-nek ajánljuk, aki a C64 fájlkezelésének rejtelmét meg akarja ismerni. De azoknak is, akik csupán jó BASIC-OLVASÓKÖNYVET keresnek. Ha valamilyik szemfűlő kiadónak nem kezdett még hozzá a magyar kiadás előkészítéséhez, akkor már talán késő. Mire a magyar fordítás megjelenne, a könyv valószínűleg aktualitását vesztené. Az is lehetséges, hogy a könyvet szerzője át-dolgozza más, újabb gépekre is. Ha ez sem történne meg, annyi haszna lehet ennek a cikknek, hogy közzétette: még BASIC-ben is lehet jó olvasókönyvet írni.

Adjunk ezekből minél többet az angolul nem tudó, vagy idegen nyelvű könyvekhez nehezebben hozzáfutó, kispénzű, magyar olvasók (diákok) kezébe is. Nagy szükség van rá. Mert a fiatalok egész nemzedéke fogott neki a nagy számítástechnikai „Egri csillagok”, „Csongor és Tünde” megírásához csupán „számítástechnikai nyelvtankönyvi ismeretek alapján. Hogyan fog ezekben a művekben például a „tudós monológja” hangzani?

(David Miller: Commodore 64 Data Files, 1984. Prentice-Hall International, Inc., London)



**Commodore 64-től . . . . . IBM PC AT-ig  
számítógépek garanciával  
Hi-Fi-, video-, fotó- és egyéb műszaki  
cikkek VÉTELE ÉS ELADÁSA  
budapesti és vidéki szaküzleteinkben:**

I., Fő u. 37/c	159-869
V., Magyar u. 1.	178-854
V., Múzeum krt. 19.	173-043
V., Váci u. 23.	183-240
VI., Szív u. 38.	
VII., Dohány u. 5.	422-507
VIII., Akácfa u. 59.	222-278
VIII., Baross tér 6.	134-116
VIII., József krt. 40.	131-478
IX., Ráday u. 9.	176-093
XI., Móricz Zsigmond körtér 7.	868-787
XIII., Rajk László u. 46/b	299-604
XIV., Sugár (Örs vezér tere)	836-567
Debrecen, Szabó I. altb. tér 6.	52-29-636
Eger, Széchenyi u. 5.	36-11-649
Győr, Bem tér 1.	96-12-802
Kaposvár, Füredi u. 24.	82-16-307
Miskolc, Korvin Ottó u. 5.	46-17-025
Pápa, Főtér 14.	89-24-402
Pécs, Jókai u. 5.	72-14-302
Székesfehérvár, Széchenyi u. 15/a	22-18-228
Szombathely, Tolbuhin u. 33.	94-18-277



# PRONET LOKÁLIS HÁLÓZAT

## HARDWARE

- A PROPER—16/A, PROPER—16/W, PROPER—16/m IBM PC kompatibilis számítógépcsaládból összeállítható hálózat kétirányú busz szervezésű
- A hálózat állomásai közös átviteli közegre kapcsolódnak
- Könnyű a hálózat rekonfigurálása, újabb állomások beiktatása
- A hálózat kiépítéséhez a PROPER—16 számítógépbe bedugaszolható opcionális hálózati kártya szükséges
- Főbb műszaki jellemzők:
  - adatátviteli sebesség : 1 Mbit/s
  - adatátviteli közeg : csavart érpárú vagy telefonkábel
  - a főkábel hossza : maximum 500 m (ajánlott hossz max. 300 m)
- Az állomások elméleti maximális száma 255  
(a hálózatba kapcsolt erőforrásoktól és a felhasználás jellegétől függően ajánlott szám kb. 2—10)

## SOFTWARE

- PROPOS V. 3. 02 vagy ezzel kompatibilis operációs rendszer
  - Többfeladatos, többfelhasználós környezetben PROMOS operációsrendszer-kiégészítés
- A hálózat szolgáltatásainak elérése külső parancsok segítségével, operátori konzolról vagy programból (felhasználói interface) biztosított

AZ **Szki**  **STABIL PARTNER!**

Számítástechnikai Kutató Intézet és Innovációs Központ  
Bővebb felvilágosítás: SCI—L Rendszerértékesítő Iroda  
1011 Budapest, Iskola u. 10. Telefon: 153-204

## A zuhanó emberke

A 18. listában állítjuk be mind a játék kezdetén, mind a játék ismétlődő részében a zuhanó emberke pozícióját. Kezdsénnél: MF=0 (280-as sor)

Amennyiben MF=1, akkor a program nem állít elő új zuhanó emberkét.

## A helikopter elővétele

A 19. listában a megfelelő irányú, szembenéző helikoptert kezdő helyzetbe teszi a program (23. ábra). Emlékeztetőül:

CL(23,15)  
CR(23,16)

CM(23,17) mérettű. A rajz előállításához a következő módosítást tettem:

625 GOTO625

A rajz elkészítések a programot „normális” módon futtatjuk.

## A helikopter földet érése

A 20. lista tartalmazza a helikopter földet érése esetén teendőket. Ez egyszerűen azt jelenti, hogy a „földet” újra kell rajzolni. Ugyanez a tennivaló minden olyan esetben, amikor egy rajzelem egy, a továbbiakban ismét szükséges rajzelem helyére kerül, mert ekkor a korábbi letörlő.

## Tennivalók

A 21. listában megvizsgáljuk, hogy van-e új zuhanó ember. Amennyiben igen, úgy a program a folytatásra ugrik. Kezdsénnél az 580-as sorban levő értéktől függően lehetséges, hogy lesz.

## Új zuhanó emberke

A 22. listában elővesszük a megfelelő pozíciójú emberkét (24. ábra). Emlékeztetőül: M(5,13)  
M1(5,13)  
M2(6,13)

A szükséges programmódosítások:

625 nincs  
580 MF+1:PM=1  
655 GOTO655

A programot itt is „normális” módon futtatjuk.

## MELLÉKÁGAK

### Tennivalók

A 23. listában szerepel annak vizsgálata, hogy a zuhanó emberke lezuhant-e vagy sem? Ha igen, akkor a játék „normális” menete megszakad, és a megszakítást vizsgáljuk.

# A játékprogramozás technikája

## AZ IGAZI JÁTÉK KEZDETE

```
570 IFMF=1THEN600 ELSE 580
580 IFRND(NF)<2THEN MF=1
590 MX=RND(220)+12:MY=35
```

### 18. lista

```
680 IF0=1THEN PUT(X-3,Y-2)-(X+20,Y+14),CR,PSET
610 IF0=2THEN PUT(X,Y-2)-(X+22,Y+14),CL,PSET
620 IF0=3THEN PUT(X,Y-2)-(X+22,Y+14),CH,PSET
```

### 19. lista

```
630 IFY>125THEN LINE(90,160)-(130,160),PSET
```

### 20. lista

```
640 IFMF<>1THEN710
```

### 21. lista

```
650 ON PM GOSUB1410,1450,1430,1450
1410 PUT(MX,MY)-(MX+5,MY+13),M,PSET
1420 RETURN
1430 PUT(MX,MY)-(MX+5,MY+13),M1,PSET
1440 RETURN
1450 PUT(MX,MY)-(MX+6,MY+13),M,PSET
1460 RETURN
```

### 22. lista

```
660 MY=MY+MS
670 IFMY>147THEN MF=0:GOTO1470
```

### 23. lista

```
1470 PUT(MX,MY)-(MX+8,MY+15),M,PSET
1480 FORT=31TO18STEP-2:PLAY"U"+STR$(T)+
"IT255I03:DEAD".INEXTT:GOTO1370
```

### 24. lista

```
1370 PLAY"D1V31T4L20P100P80L4GP100L20L
4B-A8GG0F+0"
```

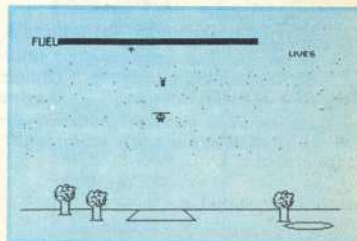
### 25. lista

```
1490 FORT=1TO18:INEXTT
1500 CLS:INB="GAME OVER"
1510 FORT=1TO9:PRINT(185-T,HID$(AB,T,1))
4SOUND180-(T+10):1:IFDRL=1TO200:INEXTD,T
1520 PRINT#200,"YOU SCORED"1BC1
1530 IFSCHS THEN H$=SC:GOTO1540 ELSE GOTO1560
1540 PRINT#200,"WELL DONE, YOU'VE BEATEN
TODAY'S HIGH SCORE!"
1550 GOTO1570
1560 PRINT#220,"TODAY'S HIGH SCORE IS"(HS
```

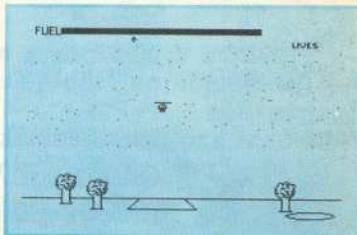
### 26. lista

```
1570 PRINT#300,"DO YOU WANT ANOTHER GO? (Y/N)"
1580 IS=INKEY$:IFIS="Y" THENZ40 ELSE IFI
="N" THEN1600
1590 GOTO1590
1600 CLS:PRINT"BYE, THANKS FOR PLAYING!"
1610 FORT=1TO25:INEXTT:GOTO220
```

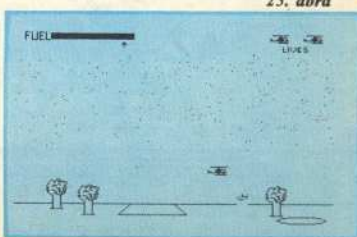
### 27. lista



23. ábra



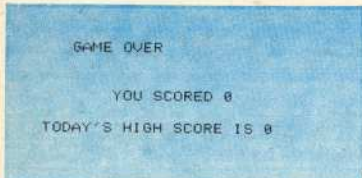
24. ábra



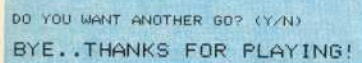
25. ábra

A 24. listában a földet ért emberkét fölrajzoljuk. Az 1480-as sorban levő furcsa jel, ami már az első cikkből is előfordult, az, amit a „S” helyett a nyomtató kiír. A listában szereplő DEAD szójáték. Angolul ha-

## A kritika kritikája



26. ábra



27. ábra

lottát jelent, de egyúttal egy kis dallam is. A kép a 25. ábrán látható. A szükséges programmodulok: 655 nincs  
1385 GOTO1385  
A futtatás itt is a szokásos.

### „Életszám”-vizsgálat

A 25. lista gyászzenét produkál, majd csökkenti az „életet” számát (LI). Ha nincs már több „élet”, akkor végeredmény-kijelzés, ha még van, a már ismert eredménykijelzéssel visszatér a játék menetébe.

### Végeredmény

A 26. lista letörli a képernyőt, majd hangaláfestéssel kiírja a végeredményt. (A listában levő „É”-t az „at” karakter helyett írja ki a nyomtató.) Ha az elért pontszám (SC) nagyobb, mint az eddigi rekord (HS), akkor értékelés és ennek kiírása következik (26. ábra).

A programváltoztatás:  
1385 nincs  
1575 GOTO1575  
A futtatás:  
RUN 1490

### Vége?

A 27. lista megkérdezi, hogy akar-e még játszani? Ha nem, a program búcsúzik (27. ábra), ha igen, akkor visszatér a játék elejére (a játékszabály ismertetését kivéve).

A módosítások:  
1575 nincs  
1605 GOTO1605  
A futtatás:  
RUN 1600

A cikkben szereplő egyetlen új utasítás (26. lista) a SOUND. Ezt két paraméter követi, vesszővel elválasztva. Az első a zörej hangmagasságát adja, és értéke a zongora középső C-jének (1–255)/89 része. A második a hangzás hossza, melynek értéke (1–255)/16 másodperc.

GALINA FERENC

A Mikroszámítógép Magazin 1985/2. számában Várgédd Tamás tollából bírálatot közölt „A számítógépről mindenkinek” című, Szluka Emil kollégámmal közösen írott könyvünkről (Hirlapkiadó, 1984). Ezt a bírálatot sajnos nem hagyhatom szó nélkül.

A könyvet, amely voltaképpen egy, a Népszabadságban 1984 tavaszán megjelent cikksorozat kissé átdolgozott változata, ketten írtuk. A két rész élesen elválik egymástól: az első részt Szluka Emil írta, a másodikát én. Ilyenformán az első részre vonatkozó bírálattal kapcsolatban illetéktelenek érzem magam, ehhez nem is szólok hozzá, mert nem szeretnék a fogadatlan prókator nevéseges pózában tetszelegni: ha Szluka Emilnek van mondanivalója, majd elmondja ő maga.

Az általam írott második részről a bíráló csak röviden emlékezik meg. Kissé meglepő számomra, hogy éppen egy számítástechnikai szakember kezelje ironikusban a számítógépet, és írja azt, hogy szerinte hibásan tüntetem fel a számítógépet „a bölcsék kövé”-nek; ezt majd a jövő eldönti.

Ami azonban arra kényszerít, hogy hozzászóljak a bírálathoz, az a V. T.-nak az a megjegyzése, hogy „a szerző csak akkor árulja el, hogy teljes tájékozatlanság (1–P.G.P.) a témában, amikor ilyen mondatokat ír le” — majd idéz egy mondatot, egyetlen egyet a mintegy negyven nyomtatott oldalnyi szövegből. Erre a mondatra mindjárt viszszaírtam, különös tekintettel arra, hogy vajon ez a mondat valóban „teljes tájékozatlanság”-ról árulkodik-e. Előbb azonban hadd tegyek egy megjegyzést.

Nem vagyok számítástechnikai szakember, soha nem is próbáltam ilyen színben feltüntetni magamat, jöllehet az elmúlt években sok a számítástechnikával kapcsolatos cikket szerkesztettem, gondoztam (már az Élet és Tudománytól 1961 és 1968 között is), és nem kevés cikket írtam. E munkám során sok kiváló szakemberrel sikerült kapcsolatba kerülőm, akik türelmesen oktattak a számítástechnikával kapcsolatos ismeretekre, és én igyekeztem figyelmes, jó tanulólenni. Az első „leckéket” Tarján Részótló kaptam, de sok más „tanáromat” is megnevezhetném, anélkül, hogy bármelyiküknek oka volna szégyenkeznie miattam. Kézirataim elkészítése során mindig igyekeztem a legjobb szakember segítségét igénybe venni, nemcsak az informálásban, hanem lektoráltattam is kézirataimat. Bizonyára ennek is köszönhetőem, hogy például néhány évvel ezelőtt a Neumann János Számítógéptudományi Társaság sajtópályázatán első díjat nyertem.

Túlzás és hiúság nélkül állíthatom tehát, hogy „teljes tájékozatlanságot” azért a számítástechni-

ka terén aligha árulhatok el. Ezt már csak azért sem tehettem volna meg, mert könyvem — és előtte cikksorozatom — szigorú szakmai ellenőré nem engedte volna meg. Nevét főlélegesen ideírom: megtalálja az olvasó a folyóirat impresszumában mint a szerkesztőbizottság vezetőjét.

Nem azért említettem ezt, mert rá akarok hárítani a hibát, hanem azért, hogy figyelmeztsem a bírálót, hogy mintha — számomra ismeretlen okokból — kissé elragadtató volna magát és elrugaskodott volna a tényektől.

És ha tényekről van szó, térjünk vissza arra az egyetlen mondatra, amellyel V. T. az én „teljes tájékozatlanságomat” bizonyítani óhajtott. Ez a mondat a bírálatban így hangzik: „A gép memóriája egy megabyte, ami virtuális tárrakkal 4 Mbr-n bővíthető; az táruk mágneslemezek és mágnesszalagok.” És ehhez zárójelben meg gúnyosan és latin tudását fitogtatóan hozzáfűzi: Sic! Vagyis: így! Nos: nem egészen így.

A könyv 102. lapján, a második hasábján egy, az IBM 370-et követő sorozatból való gépről ez áll: „A gép központi memóriája egy megabyte (vagy ahogyan egyikük s „magyarosan” írják: megabájt, rövidítve Mb= egymillió bájt, ami virtuális tárrakkal 4 Mbr-n bővíthető; a táruk mágneslemezek (diszketek) és mágnesszalagok.” A két mondat nem egészen azonos: a könyvből kiirt (utóbbi) szövegben feltűnik, hogy a vagy szó előtt megnyitott zárójel nem zárul le, ami arra vall, hogy ebbe a mondatba beletenyérelt „a szedőszekrény őrdőbe”, ahogyan az újságírók és a nyomdászok a sajtóhibát nevezik. De ettől függetlenül: miért bizonyítja ez az — akár a valódi, akár a bíráló által „idézett” — mondat a szerző „teljes tájékozatlanságát” a számítástechnikában? Ezt kellett volna odairnani a „Sic!” helyett a bírálónak: akkor az olvasó és a megbírált szerző egyaránt okult volna irásából. Ha ugyanis a bíráló csak arra gondol, hogy a virtuális szót meg kellett volna ismételni a táruk szó előtt, akkor nagyon lebecsülte az olvasók intelligenciáját: ők ugyanis bizonyára megértették, hogy azok diszketek és mágnesszalagok. Ha pedig ez tévedés, a bíráló feladata lett volna tájékoztatni a szegény félrevezetett olvasót és a tudatlan szerzőt. (A lektorálór nem is szólva.)

Várgédd Tamás láthatólag élvezi saját ironiáját, mert bírálatát azzal zárja, hogy könyvünk tanulsága az: a rossz információ, ha érteknél nem is érték, árunk lehet őr. Csakhogy ez nem tartozik a bírálóra — aminthogy a szerzőre sem: ez a kiadó dolga. Hogy neki ez az őr hasznos hozott-e vagy sem, a könyv tartalmának megítélés szempontjából közömbös. Sőt, meglehetősen rosszízű megállapítás. A bírálatnak — legalábbis egy ilyen igényes folyóiratban — meg kellene maradnia a tények szírad talaján.

PETŐ GÁBOR PÁL

## Tisztelt Szerkesztőség!

Furcsa írást olvastam a µMagazin 1986. 3. (márciusi) számában: Simonovits professzor „Számítástechnika” c. tankönyvéről egy eléggé elmarasztaló kritika jelent meg. Magával a recenziónál nem kívánok foglalkozni, mert úgy érzem, P. Cs. füstölését igazából senki sem veszi komolyan — még talán ő maga sem. A vélemény neveltségétől jól, sokkal általánosabb dolgok érdekelnének, nevezetesen az, hogy milyen mértékben tudott a szerkesztőség az írásról, illetve mennyire értettek/értének egyet vele. Úgy érzem ugyanis, hogy bárki bármit, bárhol elmondhat, leírhat — ez szive joga. Ugyanakkor egy olyan kérdésben, mint a számítástechnika közoktatása, az Önökhöz hasonló laptól jobb kritikai érzék

várható el, mint amiről az írás tanúskodik. Például, vajon miért nem olvashatunk egyetlen vitát sort sem a TV-BASIC c. műsorral/könyvvel vagy az iskola-számítógép rovatokkal kapcsolatban a µMagazin hasábjain? Ezek a szerkesztőség minden tagjának szabadéltalanul tetszettek?

A µ '86 alkalmával rendezett szerkesztő-olvasó találkozósn többen kifogásoltuk az „Alapozás” címen folytatott tanfolyamot (szerző ugyancsak P. Cs.). Ez az a pont, ahol ismét az Önök véleményére lennék kíváncsi, immáron tehát részletezve a magam kifogásait:

1. Szerintem különös, sőt talán kissé erkölcstelen is, hogy olyasvalaki írjon ennyire szubjektív, ledorongoló kritikát egy könyvről, aki maga is

próbálkozik hasonló dologgal. Persze, ezt ön-magával kapcsolatban nem mindenki veszi észre, de a szerkesztőség egyik feladata éppen ez a kontroll lenne.

2. Pedagógiai szempontból nem lehet mellékes, hogy valami érthető-e vagy sem. A megbírált könyvben — jóllehet — sok apróság vitatható, korrigálható, az azonban TÉNY — tanárok és diákok tömege bizonyítja —, hogy ÉRTHETŐ, sőt még érdekesnek is mondható. Ugyanakkor hiába kérdeztem végig kb. 100 embert — köztük pedagógust, középiskolást, egyetemistát, szakembert stb. —, egyetlenegy sem akadt, aki véleményét tudott volna mondani az „Alapozás”-ról. Egyikük sem érdekelte, és egyikük sem értette!

3. Átgondolták-e Önök a szóban forgó recenzió várható hatását tanár—szülő—diák körben (pardon, nem szabályos háromszögben)? Oktatói rendszerünk joggal áll az elmarasztalásra kész kritika reflektörjében, de erről nem tehet sem a számítástechnika, sem egyetlen pedagógus. Ismét kérdelem, helyes-e pont ékezt bántani, ráadásul — szerintem — indokolatlanul?

4. Kiváncsi vagyok továbbá, hogy Önök szerint kinek (milyen rétegnek) az érdeklődés tarthat számot P. Cs. sorozata, melyhez szinte tökéletesen illeszkedik a 86/3. számban megjelent (induló!) „OKTATÁS-MÓDSZERTAN”?

**Móttó:** Fehérek közt egy európai

J. A. Th. Mann üdvözlése

## Tisztelt főszerkesztő, kedves Kovács Győző!

Azért fordulok levélben Magához, mert a 3. szám elszáll. A „büntett az ifjúság ellen” — Pogány Csaba könyvkritikája — a  $\mu$ Magazin ellen is elkövetett büntény.

Több oknál fogva.

1. A könyv jó.
2. Jobb, mint a szokásos tankönyvek.
3. Ha rosszabb volna, akkor is körültekintően kellett volna könyvkritikát közzélni.
4. Hát még egy tankönyvről!
5. A kritika hamis és ostoba: (A legsúlyosabbnak tartható Weierstrass „hibásan” kimondott tétele számítógépes környezetben helyes így, hiszen egy korlátos intervallumot a gép csakis zártként tud tekinteni. Csakis diszkrét értékei lehetnek.) Mindezt azért tartom fontosnak közölni, mert még jó másfél évvel ezelőtt, amikor még könyv alakban nem volt olvasható, én Pogány Csaba jelenlétében lelkesedtem — ezzel talán irigységet vagy rosszindulatot keltemtet? Ugyanakkor ígértem a könyvsorozat vezetőjének — 85 novemberében — egy kritikát, épp erről, de sajnos nem készültem idejében el vele. A kritikának az lett volna a címe (tudom, szereti J. A.-t): **Fehérek közt egy európai.** Ez nem túlzás. Ez egy kivételes műfajú, szemléletű könyv. Én azt remélem, hogy le fogják fordítani néhány nyelvre.

Üdvözléssel egy volt rovatvezetője  
VOTISKY ZSUZSA

Mennyiben azonosulnak a „Zöldesgkert” pókhendi stílusával? Remélem, ezeket csak apró figyelmen kívül hagyások éltetik, avagy Önök is vállalják ezt a tudományoskodó, misztifikáló, szórásalagató stílust, nyelvi csúrés-csavarást, amely eleve lemondhat a megértésről, és teljesen idegen a megkedveltetéstől? Ez jelentené az „informatika társadalmasítását”-t? Azt még megértem, hogy P. Cs. szerint rajta kívül senki másnak nem lenne szabad tollat fognia, de úgy tűnik, a lap is hajlik felé: a két utolsó számban P. Cs. összesen HAT írással szerepel!

Fenti észrevételekre, kérdésekre nagyon kíváncsian várom a választ. Elismerve a  $\mu$ Magazin sok értékét (a legutóbbi számból pl. a 65XX összefoglaló, több programötlet stb.), nagyon keveslem a tanárképzéssel, középiskola-alapozással, ismeretterjesztéssel kapcsolatos hasznos anyagokat. A teljesség kedvéért: hiánypótló a „Számítógép anatómiája”, általában tetszik a „Favágás”, hasznos a „Strukturált programozás”. Több száma visszamenőleg azonban korántsem ilyen részös a kép. Ugyanakkor így érzem, hogy az ország immáron legterjedelmesebb számítástechnikai lapjának nem lehet létegetlen terület sem a tanárok, sem a haladóbb amatőrök képzése.

Tisztelettel: TÖRÖK TURUL

matematikus

MTA KFKI

## Margó

A  $\mu$ M soha le nem irt alapokmányában egy olyan passzus biztosan szerepelne, hogy — a lap demokratikus fórumot biztosít mindenkinek, aki szakmai véleményét ki akarja fejteni, és talán egy kiegészítés is volna, hogy — a lap biztosítja a bírálathoz és az ellenbírálat-hoz is a jogot. Miután mindkét levél burkoltan arra is céloz, hogy mondjuk el a szerkesztőség véleményét is, megteszem. Elmondom a magamét.

1. A Simonovits-könyvet még nem olvastam, így véleményem csak objektív lehet. El fogom viszont olvasni, mint azt valószínűleg sokan megteszik, akik egyébként ma még nem olvasták el, de P. Cs. kritikája és a két levél olvasata után mohón keresni fogják a könyvesboltokban. A kritika így üzleti szempontból föltétlenül hasznos volt. (Valaki — egy hires ember — egyszer azt mondta: „Mindent, hogy mit, csak írjanak rólam.”) Ennyit a recenzió hatásáról.
2. Lapunk borzasztóan unalmas volna, ha csak azok a cikkek jelennének meg benne, melyek a felelős szerkesztőnek is és a szerkesztői-zottság vezetőjének is tetszenek. Ilyen alapú szelekczió még nem tudok.
3. A lapban megjelenő valamennyi cikkkel nem értek egyet, mégis megjelenik. Példaként említeném Pető Gábor Pál a számítógépről mindenkinek c. könyvnek kritikáját, amivel azért sem értem egyet, mert a könyvet én lektoráltam. Tehát tetszett a könyv, mert közérthetően szól a számítástechnika alkalmazásáról az informatikában képzetlen közönségnek. Úgy látszik nekik is tetszett, mert a könyv elfogyott. Kevésre becsültem magamat, ha a kritika megjelenését megakadályoztam volna csak azért, mert rólam is szól — egyébként okultam belőle.
4. Az „Alapozás” sorozat befejeződött, az olvasók közül az ifjúságnak nem tetszett, az érettebb korosztálytól kapott levelek egy része dicsegett. (Kinek a Papné...) A Zöldesgkert stílusát is megváltoztattuk, szimpatikusabb lesz, ha hangvételében a Ludas Matyi „Tücsök és bogár” rovathoz fog hasonlítani.
5. A stílusa valóban ügyelni fogunk, már csak azért is, mert bizonyos stílus hasonló stílusú válaszokat indít, akkor pedig hová jut ez a szegény  $\mu$ M!

A leveleket megköszönve szívélyes üdvözléssel küldi

KOVÁCS GYŐZŐ

PS. A meg sem indult vitát ezzel az írással rögtön le is zárjuk.

György Zoltán, Fehérgyarmat,

Tolnai út 13. 4900

Két éve foglalkozom programozással, BA SIC-ben és gépi kódban. Már mind a kétféle jól megy, most a számítógépek hardverfelépítését foglalkoztat. Főleg a Z80  $\mu$ P-os gépek érdekelnek, ezek közül is a HT, Primo és a Videoton TV-számítógép.

A kéresem az lenne, hogy küldjék el nekem ezek közül azokat a rajzokat, amik önöknek megvannak.

Aziránt is szeretnék érdeklődni, hogy tervezik-e Z80  $\mu$ P-os számítógép KIT forgalmazását, s ha igen, akkor mikortól várható ez, milyen áron, felépítésben?

Azt hiszem, hogy a kért rajzokat gépkönyvekben találja meg, ezeket a gyártók az eladott géphez mellékelik, érdeklődésre általában nem küldik el. Mi sem tudjuk odaadni. KIT-igényben a helyzet változatlan. Nem üzlet, ezért nem gyártják.

Andrási Zoltán, Budapest,

Tito u. 6. IV. 22. 1045

Előjáróban szeretném leszögezni, hogy a hobbi-számítógépekkel és programozással foglalkozó lapok között (Mikro Magazin, BIT-LET, Mikrovilág, Rádiótechnika) én az önök magazinját és a BIT-LET-et kedvelem a legjobban. Erkölcstelennek tartom azt a megoldást, amikor a lap második száma közel a duplájába kerül az elsőnek, ráadásul hosszasan taglalja, hogy miért ne öntsünk forró kávékat a lemezegységbe, meg egyéb „hasznos tanácsok” (Mikrovilág).

Szintén nem nyerte meg tetszésemet az a megoldás, amikor beharangoztak érdekes cikkeket (EPROM-programozó, külső programtár stb.), majd a beharangozó utáni számokban már csak azt lehet megtudni, hogy ezek a készülékek megvásárolhatók a DIGITAL üzletben (Rádiótechnika).

Ezekhez hasonló durva hibákat a magazinban még nem találtam, de ennek ellenére az a véleményem, hogy lehetne jobb is.

Fantasztikus, hogy egy számítógéppel foglalkozó lap (amely ily módon a csústechnológiát képviseli, mely ha közvetve is) képtelen elérni azt, hogy a mérete egyforma legyen. A birtokomban lévő 14 szám 4-féle magassággal jelent meg. Ez már csak azért is bosszantó, mert a papírméret és a kúalak egyébként arra inspirálna, hogy bekötösem.

Szívesen látnék több programot. Az a mennyiség, amit C64-re közölnek, megfelelő, de miért csak COMMODORE? Meggyőződésem, hogy a SINCLAIR, PRIMO stb. gépek tulajdonosai is szívesen látnának több, lehetőleg gépi kódot tartalmazó programot.

Nagyon hasznosnak tartottam a repülésszimulációs programokat ismertető cikket. Közismert, hogy a programok itthon „gépről gépre” (kalózmásolat) terjednek, aminek etikusságáról lehet vitatkozni, de ez a valóság. Ennek az a következménye, hogy rengeteg játék, felhasználói program szinte használhatatlan, mert nincs hozzá az alapvető tudnivalókat, információkat tartalmazó leírás. És itt nemcsak a játékok programokra gondolok! Sikerült megszereznem például a

**Amikor megkapom a leveleket, nagyon sokszor úgy érzem, hogy a rovatot át kellene adni már valaki másnak, hiszen nagyon sok szombat-vasárnapon kizárólagos programja a levélolvasás és a válaszfírás. Aztán elkezdem olvasni a leveleket és a bosszúság helyét átveszi az öröm. Öröm látni, hogy — elnézést kérek — olvasóink is fejlődnek. Nem kevés kritikát kapunk, amelyekből süt a lap iránti aggodás, a szeretet és a felelősségérzet. Nemrégem valaki bírát néhány írást, talán észre sem vette, hogy úgy fogalmazott: „... a mi lapunk nem engedheti meg magának, hogy ...” Nagyon jó lenne, ha a  $\mu$ M minél többeknek lenne „a mi lapunk”, a szerkesztőség mindent megtesz ennek érdekében.**

**Nagyon sok olvasónktól kaptunk levelet a 85/6. számban megjelent UNIN rutinnal kapcsolatosan. Sajnos a nyomtatott lista utolsó oszlopa lemaradt, a javítást a májusi számban közzöltük.**

Spectrumon futtatható C64-szimulátort, de a mai napig nem tudom felhasználni, mert van valami „trükkje”. Egyszóval szívesen látnám a gyári programokat ismertető cikkeket. Gondolom, más is.

Levelére már válaszoltam, itt a levél egy részét közlöm. *Ami a lap méretét illeti, az a nyomdai adottságoktól is függ. Ebben az évben az új nyomda a lapot ebben a méretben vállalta. Hogy milyen programokat közlünk, az attól függ, hogy milyen programokat kapunk. Azokat a programokat közöljük, amelyek tartalmilag, programtechnikailag is a legjobbak. Távól áll tőlünk, hogy a Commodore 64-et bármilyen módon is előnyben részesítjük a többi géppel szemben.*

*A  $\mu$ M-ban gyári programokat csak akkor talál, ha ennek a jogát hivatalosan is megkapjuk, kalózmásolatokkal mi sem foglalkozunk. Nem tagadom, a jó, itthon készült programokat előnyben részesítjük.*

Szupper György, Tamási,

Szabadság u. 26/A 7090

Nagy szeretettel olvasom az önké lapját, s mondhatom, igen tetszik. A nevem Szupper

György, és nyolcadikos tanuló vagyok. Ne haragudjanak, hogy levelemmel zavarom önöket. Hiányolom a lapból a Commodore 64-re a programokat. Ha lehetne kérni, szeretném, hogy a C64-nek szorítsanak egy kicsivel több helyet. Továbbá még meg szeretném tudni, hogy hol kaphatnék postán C64-re játékelemeket? Az új-ságot rendszeresen olvasom, és onnan jött az ötlet, hogy önkéltől tudakozódjam. Már két helyre írtam, de ott elutasították (udvariasan).

*Nem tudom, kinek higgyek, András Zoltánnak vagy önnek, tehát sok-e a C64-program, vagy kevés. Maradunk annál, amit A. Z.-nek írtam. Ha megtudom, hogy hol lehet postán C64-játékelemeket kapni, megírom. Addig is forduljon a Novotrade-hez, 1136 Budapest, Fürst Sándor utca 24—26.*

Dudás Ernő, Mezőtúr,

Petőfi tér 1. 5400

Szeretném segítségüket, ill. tanácsukat kérni. A gondom az, hogy vásároltam egy CASIO Pb—200 mikroszámítógépet, de nem tudom kihasználni összes lehetőségeit. Ezt értem szoftver és hardver terén is.

Ha megoldható lenne, hogy más, hasonló gép tulajdonosával kommunikálni tudnék, számomra ez nagy segítség lenne a továbbiakban.

*Remélem, hogy olvasóink közül valaki segíteni tud, hogy a CASIO-t minél jobban megismerje.*

István Takáts, 3690. Bathurst str. # 102.

Toronto, Ont. M6A 2E7 — Canada —

Egy 8 bites ATARI 800XL-re épített rendszer tulajdonosa vagyok és a „Toronto Atari Federation” nevű Computer Klub tagja; aki szeretne minél többet megtudni a számítástechnikáról szívesen megbeszél.

Bár a tudásom e téren enyhén szőve alapfokú, kedvenc időöltötésem, hogy felállítok valamilyen problémát és megpróbálok írni rá egy BASIC programot.

Tavaly nyáron Budapesten járva megláttam a lapjuk 2. számát, amit elhoztam magammal, általában böngészek. Így született az az ötletem, hogy jó lenne valakivel vagy valamilyen klubbal kapcsolatot tartani.

Elég nagy számú közfoglalomban levő szoftverrel rendelkezem (főleg a graphics, sound, alkalmazás és az oktatás területén), melyeket örömmel megosztanék az érdeklődőkkel.

Meggyőződésem, hogy véleményét, ötletét, információit és programokat cserélve hasznos ismereteket tudnánk gyűjteni.

*Mindig tudtam, hogy kicsi a világ, lám, már másodszerre hallom, hogy a  $\mu$ M eljutott Kanadába is. Remélem, hogy az ATARI-tulajdonosok válaszolnak a szíves hívásra. Ha lesz a HCC-ben ATARI-szekció, egészen biztosan szívesen hozunk létre kapcsolatot távolra szakadt hazánkfiával.*

Imre Miklós, Budapest,

Tito u. 8. IV. 20. 1045

Önszorgalommal tanultam meg a BASIC-nyelvet. Előbb egy VC 20-assal, majd egy

C64-gyel. Tavaly pedig vettem egy SPECTRUM-ot. Nagyon megszerettem, a hanggenerátorát leszámítom. Ettől a csekélyesgőlt eltekintve, véleményem szerint, bármely mikroszámítógéppel felveheti a versenyt.

Most kezdem el tanulgatni a Z80-as mikroprocesszor gépi kódú programozását. A BASIC után most ezzel is szeretnék megismerkedni. Vetem is könyveket, csak hogy ezekben csak töredékek vannak. Lenne egy kéresem a tisztelt szerkesztőséghez! Szeretném felvenni valakivel a kapcsolatot, akinek ilyen anyagok a birtokában vannak (könyvek, fénymásolatok, vagy kéziratok, egészen az alapoktól). Persze megfelelő anyagi szolgáltatás ellenében kellene ezek a számomra igen fontos anyagok.

Nagyon hálás lennék érte, ha a szerkesztőség segítené problémáim megoldásában.

*1. Levele olvasóink közül valaki biztosan válaszolni fog és segít.*

*2. Azt javaslom, hogy keresse fel a HCC Sinclair-szekció délutáni klubfoglalkozásait, meglátja, minden kérdésre választ fog kapni. A klubfoglalkozások időpontja: HÉTFŐ 18<sup>h</sup>—20<sup>h</sup>-ig. SZEL-LŐZŐ MŰVEK Bp. XI., ÉPÍTÉSZ U.*

Kanyó József, Budapest,

Peterdy u. 39. fsz. 4. 1071

Szüleimtől kaptam egy Commodore 116-os személyi számítógépet. Már nagyon régóta megveszem az önké újságját, de még sohasem találtam benne Commodore 116-os, illetve 16-os játékelemeket. Sőt egyáltalán semmilyen programot C16-os géphez.

Szeretném, ha ezentúl közölnének néhány programot újságjukban C16-os számítógéphez is.

Nagyon örülnék bármilyen programnak. Előre is köszönöm.

*En nem nagyon szimpatizálok a C16-os géppel, de biztosíthatom, hogy nem ez az oka annak, hogy nem talál ilyen programokat a lapban. Nem küldenek. Ha kapunk jó és érdekes programokat, feltétlenül közöljük.*

Szabó Róbert, Makó,

Beloianisz u. 128/A 6900

Számítógépes szakkörünknek C16-osai vannak, és alig van hozzá „olyan igazai” grafikai játékok. Ezért szeretném, hogyha küldene nekem vagy lapjukban közölnének ilyen programot.

*Kérem, olvassa el a fenti válaszomat.*

Ziegler Gábor, Baja

Mártonszállási u. 66. 6503

17 éves vagyok, lapjuk régi olvasója. A bajai III. Béla Gimnázium 3. osztályába járok. A mostani számban olvastam a FORTH rendszerekről szóló rovatban, hogy a HT—1080Z gépen futó tinyFORTH programhoz nem találtak magyar nyelvű leírást. Nemrégem történt, hogy kölcsönkaptam róla két angol nyelvű tájékoztatót. (EASY SCRIPT adatállományként ma is megvan.) Címük: tinyFORTH (The tinyFORTH—EDITOR). A gimnáziumban angolul tanulok, te-

hát magamnak nincs szükségem a fordításra. Ha érdeklő önök, szívesen lefordítom és elküldöm közlésre. Tartozott még hozzá egy oldal sűrűn és áttekinthetetlenül pégelt magyar tájékoztató és példák. Ha kell, ezt is elküldhetem, de ez sajtóképes formában több lesz, mint két oldal. (Az angol nyelvű szöveg eredeti állapotban 12 oldal volt, kicsit szellősebben és EASY SCRIPT-tel nyomtatva jó kétszeresére rügött.) Várom válaszukat!

Más. Többször olvastam, hogy programok illetéktelen sokszorosítása, árusítása törvénytelen. Engem az érdekel, ha egy programot egyszer, nem üzleti célból, hanem saját használatra lemásolom, megbüntethetnek-e? Szerintem nem, mert ez épp olyan, mintha barátom frissen vásárolt lemezét átmsólam a saját kazettámra. Pedig a lemezre is érvényes a szerzői jogdíj, mégsem hallottam arról, hogy ezért valakit is megbüntettek volna. Ha pénzért teszik másnak, az természetesen más téma, de épp ezért hangsúlyoztam, hogy ingyen és saját használatra teszem. Kérem, válaszoljanak erre is.

*Ha a fordítást elküldi, megköszönjük, valószínűleg közölni nem fogjuk, ti. nagyon sok Forth-könyv kapható, ezek számát nem szeretnénk szaporítani. Sorozatot kezdünk viszont Forth-mesteriskola címmel.*

Nem vagyok igazán jógazs, ezért említett cikkem írása előtt megkérdeztem az okosokat, hogy szabad-e védelem nélküli kazettát, mágneselemez átírásai csak azért, hogy egyvalaki otthon használja, azt mondhat, "lehet", ti. a copyright csak a terjesztésre vonatkozik, terjeszteni programot engedély nélkül tilos.

**Patkós Tamás, 6. oszt. tanuló,**

**Budapest, Kárpát u. 40. 1133**

Olvastam a cikket az újságjokban a „2C” számítástechnikai áruházzal. Ebben az állt, hogy az alagsorban lévő gépek nem lehet játszani. El is mentem egy barátommal, megkérdeztük, hogy van-e ilyen lehetőség. A válasz: nincs. Az a kérdés, hogy a cikk írója honnan vette, hogy ebben az áruházban lehet játszani a számítógépeket?

Nekünk is ezt mondták, a levelet elküldtem a Novotrade igazgatójának, Rényi Gábor elvtársnak.

**Somodi Zsolt, Szeged,**

**Vértói u. 5. 212. 6724**

Evel a levelemmel együtt már a negyedik leveletem írom önökhez, s közülük csak egyre kaptam választ.

Nem tudom, amit írt nekem, igaz-e: mármint az, hogy megjelent egy könyv a  $\mu$ M jövőtől, melyben csak programok találhatóak (és majdnem mind játékprogram)?

Szeretném megtudni, hogy tényleg megjelent-

e, mert én itt Szegeden egyetlenegy üzletben sem találok!

*Sajnos, minden levélre nem tudunk válaszolni. Ebben a rovatban azokat az információkat közlöm, amelyekre a válasz talán sokakat érdekel. A könyv megjelent a Műszaki Könyvkiadó gondozásában, a könyvesboltok árulták. A címe: Csupa játék ZX Spectrumra.*

**Méresei Kálmán, Salgótarján,**

**Kemerőv krt. 23. I. 2. 3100**

Magamról csak röviden annyit, hogy 32 éves villanyszerelő vagyok. A számítástechnikával kb. fél éve amatőr alapon foglalkozom.

Problémáimat egyedül igyekszem megoldani, segítséget csak végső esetben kérek. Most ezt teszem, mert úgy vélem, nem tudok tovább lépni.

Egy 48 „kilós” PRIMO gépem van. A problémám nem egyedi, azt hiszem ez minden PRIMO-tulajdonosnál jelentkezik, akit ez a gép komolyan érdekel. A BASIC-et különböző módon feldolgozó, ismertető szakkönyvek a rendelkezésemre állnak, de ezek vagy más gépre készültek, vagy általánosságban foglalkoznak a programozással. A gépre szabott könyvem a PRIMO-felhasználói könyvvel kimerül. Az önképpel februári számában megjelent rovat szerkesztői megjegyzés 2. pontjában említés történt egy gyártó által megjelentetett hardver és szoftver segédkönyvről. Eddig bárhol érdeklődtem, sehol sem tudnak róla. Kérném, hogy az említett két kiadvány (könyv) megvásárlásához segítsenek hozzá. Ha egy külföldi géptípusról lenne szó, talán magamat okolnám, miért azt a típust választottam, de egy hazai típus esetében...?

Levelem második témája az önképpel. A laptop nem akarom itt méltatni, ezt helyettem mások jogosan megteszik. Csak ennyit: SIKERLAP.

Az igények érthetően szerzeteázóak. Én, mint PRIMO-tulajdonos, több PRIMO-val foglalkozó cikket, leírást olvasnék szívesen.

*A PRIMO-val kapcsolatban azt javasolom, forduljon levelem a gyártóhoz, a leírásról szóló híronnan származik. A dícséretet szeretjük és köszönjük. Minden jó PRIMO programot szívesen közlünk, ha kapunk.*

**Tóth Gergely, 7. oszt. tanuló,**

**Budapest, Abavár u. 45. 1141**

Az alábbi, ZX81-re írt program érdekes, kelek grafikus ábrát eredményez. Ha a 29-as sorban levő 44-es számot elkezdjük csökkenteni 5-ig, akkor a „csipkétől” a „napsugár” különféle érdekes ábrákat kapunk.

10 FOR N=0 TO 90  
15 FOR F=0 TO 18  
20 LET A=N/44 PI  
25 LET K=32+F SIN A  
30 LET J=22+F COS A  
35 PLOT K,J  
40 NEXT F  
45 NEXT N

Köszönjük!

**Szieberth András, Székesfehérvár,**

**Sebes u. 17/B 8000**

A „Játék a betűkkel, vagy ÁMITÁS” című cikkhez lenne néhány észrevétel.

Az igaz, hogy tökéletesen magyar betűkészletet igen nehéz készíteni, de több olyan program létezik, ami megközelíti azt. Ezek képesek a magas szintű olvashatóságot biztosítani, használatauk igen kényelmes, gyors.

A leg több szövegszerkesztő program a svéd ábécé betűit használja. Szerintem ez nem akkora fogyatékoság, mint több cikkírójuk szerint. Bizonyítja — bizonyíthatja — ezt ez a levél is. Nem hiszem, hogy a tartalom rovására megy a külalak, esetleg félreprogramozható lenne a szöveg.

A ZX SPECTRUM-ra írt szövegszerkesztők közül az egyik legjobb a TASMURD H, amelyet Vidákovics Attila 1984-ben magyarrá lefordított.

Egy korábbi MIKRO-ban volt már róla szó. Használata igen könnyen megtanulható, felhasználását könnyíti, hogy bármikor leihívható (természetesen magyarul) a „Manko” — anélkül, hogy a beírt szöveg elveszne. A program az összes létező SPECTRUM-nyomatossal kompatibilis, mivel ha a program már nem tud mit kezdeni a nyomatossal, mi is beírhatjuk a nyomtatás-vezérlő karaktereket (persze, ha ismerjük őket).

*Én nem rajongok a svéd betűkészletért, zavar, hogy nincs benne í, ó, ő, ü, kellemetlen olvasni pl. a levelet is, amelynek utóirata: „A helyesírási hibákért elnézés kérek!” Tudjon az a gép magyarul, ha már ide került!*

Ennyi volt a válogatás az utolsó két hónapi postából. A leveleket szíves örömmel várja:

**KOVÁCS GYÖZÖ**

## Jelentkezem!

Alulírott Endrei Simony rövidesen beküldöm a pályázatra (Mikroszámítógép Magazin 1986. márc. 36. old. — A szerk.) az utóbbi hónapokban végzett munkám eredményét. Csatolom mellé a 200 Ft-ot is, aminek fejében a legalzatosságban kérem olvassa el a mellékelt tájékoztatómat, majd a programot tegye a polcra a többi mellé lebonolyítás végett!

További 300 Ft-ot küldök azért, hogy szíveskedjen lehúzni az ismertetőjét a jó odabigyeztet nevetem, amivel hivatkozva saját munkámat merészeltem megkérdegni elismerni. Kérem továbbá, kegyeskedjék a saját nevét, mint jogos tulajdonosát aláírni.

Felhívását kiegyesítendő, beküldök még további 400 Ft-ot az újsághirdetés feladás költségeire, amivel majd saját használatra bevételt keres a programra.

Kérése szerint küldök továbbá 50 Ft-ot, valamint egy megcimzett, felülbejegyzett és lezárított borítékot feladás végett, amelyben saját magamat értesítem, hogy öt kedves rokona a Jólátvunköket családból megkapta megérdemelt munkájára jutalmát.

Drága nagylelkű jötevőm, maradjon örök hűve: Endrei Simony

**K**islapon a decemberben történt gépvásárlás (2 db C16-os) után jó ideig felebotlatlanul, elzárva árvalódtak a masinák. Az indok: nincs hozzájuk színes tévé. Miután azonban sikerült ennek beszerzése (s sajnos csak mégdrágára, 26 000 forintot készült velük kapható), azt gondoltam, hogy elhárultak az akadályok. Felkerestem az iskola igazgatóját, felajánlva eddig „összeszedett” BASIC-tudományt, könyveimet, például dr. Úry könyvét, a már meglévő és a folyamatosan megjelenő Őlet és Mikromagazin példányaimat, jelezve, hogy a számítógépek használatához szerintem mindegyik feltétlenül szükség van. Kértem, hogy rendeljék meg a folyóiratokat és könyveket, vegyenek kasszettekát, azzal ugyanis, hogy megvannak a gépek, nem ért véget a munka, hanem éppen hogy el kellene kezdeni.

Mindezt abban a reményben tettem (hiú remény volt), hogy használni akarják a gépeket, és szívesen fogadják tapasztalataimat. Felajánlottam, hogy a levelezést és a jól bevált kazetás programcsereit is vállalom, lévén benne a Primo kapszán egyéves gyakorlatom. Ez a módszer egyben költségkímélő is, azon túl, hogy olyan programokhoz is hozzájuthat így az ember, amelyek nincsenek is kereskedelmi forgalomban, vagy ha kaphatók, meglehetősen drágák. A folyóiratokban mindig találni olyan címeket, ahonnan információk és programok szereshetők be, ráadásul *el-lenzolgatás nélkül*, ha nem számítjuk annak a csereprogramokat.

Felajánlottam továbbá segítségemet és szabadidőmet az iskola igazgatójának, és kértem, tegye lehetővé a délutáni órákban, hogy hozzájussak a számítógéphez. Természetesen bárkit szívesen látok, aki érdeklődik, és szívesen dolgozom együtt a szaktanárokkal. Nem játékaikamat kértem, hanem lehetőséget arra, hogy lefuttassam az otthon papíron megírt egyszerű BASIC-programjaimat, és a könyvből, amely az iskolának nincs meg, beírhasam és kasszétára vehessem a példaprogramokat. Ugyanezt terveztem a folyóiratokban már megjelent programokkal is. Úgy gondoltam, ilik ezzel mintegy honorálni a géphez jutás lehetőségét. Azzal is bíztattam, hogy valószínűleg előbb-utóbb kereskedelmi forgalomba kerülnek majd szaktárgyi oktatóprogramok is. Ez azóta meg is történ.

Az igazgatót hidegen hagyta a lelkesedésem. Segítség sem tudta megkérdeni például, hogy mire is kell az üres kazetta. Az első problémát a számítógéphez való hozzáférés jelentette, ugyanis én csak munkaidő után érek rá, akkor viszont az iskola és a gépek is gondosan be vannak zárva. A géppel való „foglalkozásra” két matematika szakos tanár lett „kijelölve”, ők mindketten éppen alakpók BASIC-tanfolyamra jártak — hol jártak, hol nem — Nagyapolra. Igaz, hogy ott nem találkozhattak

**Alábbi írásunk szerzője beleszeretett a számítástechnikába, és úgy gondolta, hogy szerzett tudásával megpróbálja a helyi iskolát segíteni. Egyszer már levelet váltottunk, akkor biztattam a szerzőt, hiszen törekvése nagyon is egyezik a művelődési kormányzat célkitűzéseivel. Szeretnénk ugyanis, ha az iskolákban nemcsak tanításra használnák a gépeket, de délutánonként pl. klub formában ezek a gépek a felnőttoktatást, sőt a közművelődést is szolgálják. Különbözn is az iskolákban ma még nincs elég pedagógus, aki a számítógépekért lelkesedne, és így képes volna a gyerekekkel is megszerettetni a számítástechnikát.**

**A cikkben a város nevét megváltoztattam, a szerzőt pedig elhallgatom, hiszen a jelenség az érdekes, nem pedig az, hogy hol is történt az eset. Azon sem csodálkoznék persze, ha többen magukra ismernének és végül is belátnák, hogy nagy céljainkat kicsinyességgel, bürokratikus úton nem lehet elérni, együtt, társadalmi összefogással viszont biztosan. Ebben bízik: Kovács Győző.**

## Számítógépek Kislapon

C16-ossal, de a programozás alapjaival megismerkedhetek.

Ennek megörültem, legalább lesz mivel együttműködnöm. De örömom korai volt, nem kiderült, hogy egyikük nemcsak a szülői megy, és a másik sem ér igazán rá, mivel kémia-továbbképzésre jár, ami minden energiáját lekötí, és különben is kétgyermekes családanya, munkaidő után siet hazára. Otthon talán szakíthatna némi időt a géppel való ismerkedésre, de jaj!, a gépet hazavinni, azt nem lehet! Nem baj, javaslom, addig én magam foglalkozom a géppel, és ha majd a feladattal megbízottak is ráérnek, legfeljebb átadom a tapasztalataimat.

Így sikerült is 2-3 alkalommal délutánonként néhány órát töltönnöm a géppel négyezsemközt. Éppen a kapott játéprogramokat futtattam, de az utolsó kétfőre már nem jutott időm, mert a takarítónő is befejezte már a munkáját.

Másnap újra megjelentem, de legnagyobb meglepetésemre a szertár kulcsa már nem volt a szokásos helyén, így a takarítónő nem tudta ideadni a gépet. Egyébként bizonyosan közölte velem, hogy töltessem kifogásolták, miért bárálhat egy idegen a géppel, amikor ők még nem is látták, különben is a két tanár nevén vannak a gépek, és mi lesz, ha elrontom őket.

Nem fészegtettem a „kulcskérdest”, inkább szóba hoztam, hogy szívesen vállalnám egy szakkör beindítását, mivel a megbízott szaktanárral decemberig nem ért rá. — Hát arról szó sem lehet — válaszolták —, mert szakkört csak

szepetembertől lehet indítani! Velem lehet alkudni — mondtam —, hívhatjuk másképpen is a foglalkozást. Nem, nem, különben sincs rá keret. — Ejnye — mondtam —, elvállalnám én díjazás nélkül is, de ha ez zavarná, a TIT-en keresztül megszervezhetjük, így volt elintéztve a Primós tanfolyammal is. — De így sem jó, mert éppen ma hallottam a GAMESZ-től (ez a szervintézéshez szükséges összes tudnivalót tartalmazza a gépkönyv is. — Na nem baj — mondta az igazgató (ez éppen pénteken történt) —, hétfőn majd rákérdezek, azt után eldöntjük, mi is a teendő. Én persze nem vártam hétfőig, még aznap megtudtam, hogy a számítógéphez vett színes tévéhez a Gelka tetőantennát szerelt fel, mert a GAMESZ szerint a számítógéphez ez szükséges, különben még elromlik!

Többet nem engedtek a számítógéphez nyúlni, pedig hajlandó lettem volna anyagi felelősségemet írásba is adni. Egyszerűen nem vagyok jogosult a kezelésére, mivel nem tartozom a tanári karhoz. Erdéklődtem, hol van valamilyen tiltó jogszabály. A válasz: ilyenről ugyan nem tudnak, de olyanról sem, ami ezt lehetővé tenné! (Elismerem, remérlíposzt.) De egyébként is már puszognak a kártszak, hogy hogy jön egy idegen ahhoz, hogy az iskola számítógépet

bárálja! Egyébként meg — tolták elém a tematikát — mi nagyon jól tudjuk, mire való egy számítógép, maga is jobban teszi, ha elolvassa. Hát elolvastam, hangosan: matematika, valószínűség-számítás, algoritmusok stb. Hoppá — mondom —, itt a lényeg, amiről eddig hiába beszéltem. De pechem volt, mert addig már nem jutott el az olvasásban.

Ekkor értem meg, miért ragaszkodnak foggal-kéggel ahhoz, hogy csak a matematika szakos tanár jogosult a használatra, a többinek meg minek a számítógép. Újra makacsul magyarázni kezdtem, hogy a matematikánál jóval szélesebb körben használható, ha van hozzá megfelelő program. Erre azt vágták a fejemhez, hogy féltő, hogy a tanulmányi munka rovására menne a dolog, mert 45 perc amúgy sem elég az oktatásra, hát még ha közben a számítógéppel kell játszani!

Szívesen tartanék foglalkozást nyáron is — mondtam —, akkor legalább biztosan nem menne a tanulmányi munka rovására. Az is nagy szó lenne, ha csak játszanának gyerekek a géppel. Sőt jöjjön el a szülő is, hadd lássa, hogy végre az ő gyereke is hozzájuthat Kislapon a számítógéphez. Ez a javaslatom sem tetszett.

Próbáltam fejtegetni, hogy az iskolának nem szabadna ilyen megreven elzárkózni attól, ami egyébként szerencsére máshol már jól bevált gyakorlat, hogy az intézmények a szabad gépidőt klubformában elérhetővé teszik az érdeklődő fiatalok és felnőttek számára. Azt mertem mondani, hogy szerintem tágabb értelemben a gép tulajdonképpen nem is az iskolák, hanem a falu közössége, lévén a vásárlást eszközölő tanács állítgató a falu gazdája. Na ennél eretenebb dolgot nem is mondhattam volna, ezzel betelt a pohár. Tudtomra adták, hogy a számítógép igenis az iskola tulajdona, és mint ilyen, kizárólag az iskola rendelkezik vele.

Nem láttam értelmét a további költelűzésnek. Annnyit még utánam vetettek, hogy majd szólnak, ha szükségük lesz a segítségemre, amit egyébként köszönök. En is!

Eddig tart a történet. Azt hiszem, jókora tanulsággal szolgálhat. Ezért is mertem megírni. Egyetlen remény élt meg: hátha egyedi eset az itteni, hátha máshol elő sem fordul hasonló. Tenni kelene annak érdekében, hogy ne is fordulhasson elő ilyesmi, ne érthessék félre szándékosan az írott szót, ne a hatalom diadalmaskodjon a jó szándék felett, mert egyébként jaj a haladásnak!

Volt egy olyan érzésem, mintha az egész nem is Magyarországon történe, hanem egy információkötő elzár szigetén, ahol közkorszaki gondolkodási sémák rögzültek. Elragad a szenvedély: szerintem az ilyen magatartás bűn, és ellentétes az azalatt az ügyre többlet hallott jelszóval, hogy „A számítástechnika mindenkié!”.

## BITEK ÉS FIGURÁK

### Állásértékelés III.

Az eddigiekben megismertük az állás értékelésének az anyag, a mozgékonyaság és a centrumellenőrzés komponensével. Ez azonban az állás pontos értékeléséhez még nem elegendő, ezért további szempontokat is figyelembe kell venni. Ez alkalommal a fejlődés és a sáncolás jelentőségével, ezek kvantitatív meghatározásával foglalkozunk.

### Az értékelőfüggvény fejlődéskomponense

A fejlődés szerepe tulajdonképpen a megnyitás kezdeti stádiumában a legjelentősebb. A sakkjáték megnyitásában fejlődnek ki a figurák. Az a cél, hogy a középjátékban minél aktívabb szerephez jusssanak. A játszmában az van kedvezőbb helyzetben, aki tempelőnyvel rendelkezik, mert tulajdonképpen ez diktálja a játszma további menetét. Ezért nagyon fontos, hogy figuráinkkal minél előbb kifejlesztjük, és ennek érdekében a játszma elején ne lépünk többször azonos bábuvál, hanem próbáljuk meg a teljes haderőt mozgósítani. Sakkoktatásukon már a legáltalánosabb szinten is tanítják az „anyagot áldozok, időt nyerek” elvet, melyben az egyik játékos anyagot áldoz a fejlődés érdekében, hogy tempelőnyt szerezzen. Ez sok esetben már meghatározza a parti végső kimenetelét.

A fejlődés és a korábbi számokban említett mozgékonyasági érték szorosan összefügg. Amelyik félnek fejlettebb a hadereje, annak a mozgékonyasági tényezője is nagyobb. A mozgékonyasági konstans tárgyalásánál említettük a különböző cseljatkók előnyét, amelyek azon alapszanak, hogy az egyik fél anyagi áldozatával a figurát kifejlesztzi, és ezáltal a bábuk mozgékonyasági értékét is növeli. Alapfokon általában azt tanítják, hogy a fejlettségben három lépés előny körülbelül egy gyalogot ér, ami eléggé durva közelítés, de sokszor jól tükrözi az állás aktivitását. Nézzük a következő lépéssorozatot, amelyben egyértelműen látható a sötét haderő fejlődési előnye:

1. Hg1—g3 d7—d5
2. Hf3—g1 Hb8—c6
3. Hg1—f3 Hg8—f6
4. Hf3—g1

Ha két egyforma erős játékos innen folytatná a játszmát, a parti sorsa egyértelműen sötét javára billenne. Ez a példa mesterkelt, de kifejezi a lépésméltós hátrányát, amely rossz játékvezetéssel a megnyitás későbbi szakaszában gyakran előfordul. Először a könnyű tisztekkel (futóval, huszárral) kell kifejleszteni, mert ellenkező esetben — ha a bástyát vagy a vezért centralizáljuk — az ellenfél figurái fejlődés közben tempót nyerhetnek, ha bástyánk és vezértünk megtámadásával újabb bástya- vagy vezérlelépésre kényszerítik.

A fejlettség értékelésében sokféle tényezőzt hasznos megvizsgálni. Először is meg kell vizsgálni, hogy mely figurák hagyták el eredeti helyüket. Ennek figyelembevételével az előbb említettekkel egybehangzóan elsőként a huszárok és a futók kifejlesztése, majd a sáncolás és végül a nehéz tisztek aktivizálása a helyes sorrend.

A fejlettség másik, kevésbé ismert szempontja, hogy ameddig vezér van a táblán, addig a sáncolás majdnem létfenntartás. Ennek több oka van. A legtöbb megnyitásban a középvonalak jóval hamarabb kinyílnak, mint a szélsők. A centrumban bekövetkező cserék során ezek a vonalak sok esetben viszonylag korán szabadná válnak a bástyák vagy a vezér számára. Így érthető, hogy a középben maradt királyra ezeken a vonalakon több veszély leselkedik, mint a sáncolás után, amikor az eredeti helyükön levő f, g és h gyalogok (vagy hosszú sánc esetén az a, b és c gyalogok) szinte a saját testükkel oltalmazzák a király életét. Az egyszerűség megbillentését pedig azáltal okozza a középben rekedt király, hogy megszakítja a két szárny — a vezér — és a királysárgny — közötti összeköttetést. Így szinte közlekedési akadályt képez, és kettéosztja saját haderejét, meggátolva annak egyszerű elhelyezkedését és átcsoportosítását. Ha viszont a vezérek lecserélődtek — legalábbis az ellenfél vezére nincs a táblán —, akkor már nem olyan alapvető a sáncolás.

Hogyan lehet ezeket a szempontokat beépíteni az értékelőfüggvénybe, hogyan lehet ezeket kvantitatív módon kifejezni? — erre David Levy egyszerű, de érthető példát adott. Ebben a mennyiség ugyan nem pontos, de eléggé érzékeny ahhoz, hogy a gondolatmenetet világosan tükrözze, és ezáltal egy sakkprogram írásához tökéletes adjon, amit tovább lehet finomítani. Levy példája:

**Fejlettség =**  
 $D/3 - U/4 - (k * C)$

- ahol:**
- D** = azoknak a könnyűtiszteknek a száma, amelyek nincsenek eredeti helyükön;
  - U** = 0, ha a vezér még nem mozdult, vagy leütötték, illetve = a fejletlen könnyű- és nehéztisztek száma, ha a vezér már elmozdult, de még nem ütötték le;
  - C** = 2, ha az ellenfél vezére még a táblán van, illetve =  $(1 - P/4)$ , ha az ellenfél vezére már nincs a táblán, ahol P a kölcsönösen leütött bástyák és könnyűtisztek száma;
  - k** = 0, ha a játékos elsáncolt; = 1/3, ha a játékos elvesztette a vezérszárnny sáncolási jogát;
  - = 2/3, ha a játékos elvesztette a királysárnny sáncolási jogát;
  - = 1, ha a játékos mindkét szárnnyon való sáncolási jogát elvesztette.

Az így megadott függvény első

két tagjában külön ki kell emelni, hogy a "D" változoban nem az eredeti helyükön levő tisztek számát kell figyelembe venni, hogy az eredeti helyen leütött tisztert ne járjon fejlődési büntetés, mivel az a saját hibáján kívül nem léphetett még. Ha a játékos vezére még a táblán van, de már mozdult, akkor a játékos minden tisztejét büntetés jár a függvény második tagja szerint. A fejlettségi függvénynek ez a két tagja biztosítja a már említett stratégiát, hogy elsőként a könnyűtisztek fejlődjenek ki, majd csak ezután következzen a vezér és a bástyák aktivizálása. Tulajdonképpen ez a két tag áttelelesen a sáncolást is elősegíti, mivel ez — mint kettős lépés — a királyt is és egyben a bástyát is kifejleszt. Azonban a tapasztalat szerint a sáncolás ösztönzésére ez kevés, ezért kell külön figyelembe venni a fejlettségi függvény harmadik tagjában a játékos sáncolási jogának elvesztését. A "k" szorzó értéke akkor a legnagyobb, ha a játékos mindkét oldali sáncolási jogot elvesztette. Így fokkal csekélyebb a szorzó értéke, ha csak a királysárnny sáncolás joga veszett el, és még csekélyebb, ha csak a vezérszárnny. A királysárnny sáncolás a gyalogok elhelyezkedése miatt biztonságosabb a vezérszárnnyinál.

A sáncolási jog elvesztésének megállapításánál a már említett okokból figyelembe kell venni, hogy az ellenfél vezére fent van-e még a táblán. Ennek megfelelően a "C" szorzó értéke kettő, ha a vezér játszik. Viszont ha az ellenfél vezére elvesztett, vagy lecserélődött, akkor a sáncolás fontossága csökken, és így "C" értéke a következőképpen számítható ki:

$(1 - (P/4))$ , ahol P a kölcsönösen leütött tisztek száma, beleértve mind a könnyű- mind a nehéztiszteket.

Ha jobban megvizsgáljuk ezt a kifejezést, akkor kiderül, hogy "C" értéke negatív is lehet. Ez akkor következik be, ha például az ellenfélnek már csak egy tiszte van, ha a vezér játéktárra kerül sor. Ez arra ösztönzi a programot, hogy ebben az esetben mellezse a sáncolást, mert ez a végjátékban kimondottan hátrányos. Ott ugyanis a királynak aktív szerephez kell jutnia.

Nézzünk példaként egy játszmat, amelyben a mozgékonyaság és a fejlettség játzsza a vezető szerepet. Oszváth András és Eperjesi L. 1967-ben játszották.

1. e4, c6 2. Hc3, d5 3. Hf3, Fg4 4. h3, Ff3 5. Vf3, Hf6 6. e5! 7. Hf7 7. Vg3, e6 8. d4! 9. Vb6 9. Fd3, Vd4 10. 0—0 11. Fe3, Vb4 12. a3! 13. Va5 13. b4, Vc7 14. f4, a6 15. Ba1, e6 16. Ff8! 17. Ff5 18. Hc5 18. Fg5, He7 19. Fg4, h5 20. Vh4, Hg6 (lásd az ábrát).

Láthatjuk, hogy világos figurák mennyivel mozgékonyabbak, fejlettebbek, mint a sötétek. Világosnak minden tiszte mozgott, kifejlesztődött, és így aktívan részt vesz a küzdelemben. Királyát sáncolással biztonságba helyezte. Sötét viszont csak passzív szerephez jutott, királya a középben rekedt, ennek következtében királyállása is nagyon bizonytalan. Futója és mindkét bástyája az eredeti helyen maradt, és így ezeknek a figuráknak az aktivitásáról nem is beszélhetünk. Világos ki is használja mozgékony állását, és egy leleményes kombinációval — melyet a számítógépek is hamar megtalálnak — gyorsan megnyeri a partit.

21. Hd5!! Vc6 (Vc6-ra 22.e6! gyorsan dönt)

Ha most tüzetesebben megvizsgáljuk az állást, és az előbb megadott fejlődési függvénybe behelyettesítjük az értékeket, akkor bizonyosságot szerzünk arról, hogy a közelítés eléggé pontos. Nézzük konkrétan a függvény egyes komponenseinek értékét!

**D:** nem az eredeti helyükön levő könnyűtisztek száma: 3 világos, 2 sötét.

**U:** világos esetében 0, mivel a vezér már elmozdult eredeti helyéről, viszont minden figurája ki is fejlődött. Sötét számára viszont 3, mert vezért tiszteinek kifejlesztése előtt aktivizálta; az bűntetőpont levonásával jár.

**C:** mindkét fél esetében 2, mert a vezérek a táblán vannak.

**k:** 0 a világos és sötét számára is annak ellenére, hogy sötét nem sáncolhat, mert a világos futó útvesztőnölát nem lépheti át. (Ennek a kivételre esetnek a pontozására, értékelésre későbbi cikkünkben visszatérünk.) Így a végső eredmény a következőképpen alakul:

$FV = 3/3 - 0/4 - 2 \cdot 0$   
 $FS = 2/3 - 3/4 - 2 \cdot 0$

Itt FV a világos, FS a sötét fejlődési függvényt jelöli. Látható, hogy világos pontértéke +1, míg sötété -1/2, tehát a különbséget jól szemlélteti. Ha a mozgékonyasági értéket is figyelembe vesszük, akkor még nagyobb különbség látható: világos 40, míg sötété csak 32. Így világosan kitűnik, hogy ezeknek a szempontoknak a figyelembevételével az állás milyen pontosan értékelhető. Ez a számítási mód finomított, még hatékonyabbá tehető.

Erdekességként nézzük meg a játszma csattanós befejezést.

22. Fh5!! Bh5 23. Vh5, Vd5 24. Bf7!! Hde5 25. Be5! Vd4+ (25...Ve5 26. Vg6 27. Vg5 28. He5 29. Bb7 30. Természetesen szintén reménytelen sötét számára)
26. Bf2 26. Bf7 27. Vg6+ Kd7 28. Vf7+ Kc6 29. Ff4! Va1+ 30. Kh2 és sötét feladta.

### KOVÁCS P. ATTILA





Kész szoftvertermékeket  
ajánl IBM PC/XT és ezzel  
kompatibilis  
számítógépekre a



**PcFOK**  
főkönyvi és folyószámla-könyvelési  
rendszer

Általános jellegű főkönyvi könyvelésre alkalmas programcsomag, amelynél — egy újszerű szervezési megoldás révén — az adatbeviteli munka mintegy 60%-kal csökken, és a havonként elkülönülő zárás lehetővé teszi a folyamatos könyvelést.

**PcGAZD**  
készletgazdálkodási rendszer

A programcsomag naprakész információt szolgáltat a készletek alakulásáról, ezzel egyidejűleg

- főkönyvi feladással támogatja a főkönyvi könyvelést,
- utókalkulációval az anyagellátást,
- anyagfelhasználási mérleg készítésével a statisztikai adat-szolgáltatást,
- likvidációs feldolgozásokkal a pénzügyi elszámolást,
- leltárfeldolgozásokkal a leltározási munkát.

**PcALLO**  
állóeszköz-nyilvántartási és -gazdálko-  
dási rendszer

A programcsomag gondoskodik az állóeszközök értékeiben bekövetkezett változások átvezetéséről, főkönyvi költség-helyi feladásáról, a mérlegtáblák automatikus kiegészítéséről amellet, hogy tartalmazza a statisztikai, műszaki és leltározást támogató információk előállítását.

**A mielőbbi közvetlen kapcsolat reményében  
várjuk megrendelésüket.**

**Cím: 1024 Budapest, Petrezselyem u. 6.  
Telefon: 351-950 és 351-972**



**SZOFTVERHÁZ** 1027 Budapest,  
Csalogány u. 9—11.

Tel.: 355-521, 352-151, telex: 225138

**1986. július 1-jével a számítástechnikai fejlesztések komplexebb ellátása, a piaci igények hatékonyabb kiszolgálása érdekében az ÉGSZI vezetése megalakította a SZOFTVERHÁZ-át.  
FŐ MŰKÖDÉSI TERÜLETEK:**

- kis- és mikroszámítógépes hálózatokra szoftverfejlesztés (RC 3600, PDP-alapú kisgépek, IBM PC/XT/AT, PROPER—16)
- osztott rendszerek fejlesztése. Referenciahelyek: SKÁLA, FÜSZÉRT, HÍDÉPÍTŐ
- magasraktár-szervezés
- vonalkódtechnika alkalmazása

Részletesebb felvilágosítást a fenti telefonszámon  
SZULA Szabolcs igazgató, TURAI Tamás igazgatóhelyettes,  
dr. RÉVFALVY Miklós, marketingvezető ad.

**AZ ORSZÁG MINDEN RÉGIÓJÁBAN ÉGSZI**

## C16 klub

Az elmúlt év végén jelentek meg a hazai boltokban a Commodore 16-os mikrogépek, nálunk akkor még szokatlanul olcsó, nyolcezer forint alatti áron. Tavasszal az általános iskolák számára iskolagép-ként ajánlott típusok listájára is felvették. Szinte egyik napról a másikra, gyorsan terjedtek el, és ezért kevés hozzjuk a szoftver, a dokumentáció és általában a tapasztalat.

A felhasználók intézményes segítségnyújtására a SZÁMALK KISZ-bizottsága vállalkozott. Létréhoztak egy Commodore 16 klubot. Vettek néhány gépet, ezek képezik a klub alapját. A klubössze-jöveleteket hétfőnként tartják a SZÁMALK Szakassz Árpád út 68. szám alatti épületének aulájában.

Segítségét nyújtanak a klubtagoknak a programírásban, sőt a kész programtermékek menedzselésére is vállalkoznak. Megbízást adnak különböző segédprogramok megírására; például készültek már a klubmunka keretében különböző turbo-programok, aszemler-segédprogramok, a képernyőtartalmat kinyomtató programok és egyéb ötletek megvalósításai. Ezek közül a piacpekeket az ÁSZ-bolthálózat forgalmazza. A klub-foglalkozások nyilvánosak, se hélepődij, se tagsági igazolvány nincsen.

## Beszédsszintézis

### házi gépre is

Új magyar szabadalmon alapuló beszélő perifériát készített a BME Híradástechnika Elektronikai Intézet. A készülék a széles körben elterjedt soros adatátviteli vonalon (RS232) keresztül vezérelhető, így minden olyan számítógép, személyi és mikroszámítógép, képernyős megjelenítő, amelynek RS232 kimenete van, beszélhet a segítségével; a ZX-Spectrum például az In-

terface 1-en, a Commodore 64 pedig egy RS232 illesztőadapteren keresztül. Egy-egy szó vagy kifejezés kimondása egy vezérlő ködszó kiadását igényli, így a központi egységet csak igen rövid időre foglalja le.

A készülék saját memóriájából mintegy 300 magyar szót képes kimondani. Az alapszótáron kívül speciális szókészlet is igényelhető, amelynek kifejlesztésével a BME-n foglalkoznak.

Az új periféria főbb felhasználási területei a parancsok, utasítások, eredmények kimondása, az oktató- és játékprogramokban való alkalmazás, akusztikus jelzések és riasztás a folyamatirányítás és a termelésirányítás terén, valamint a beszélő ipari robotok előállításánál.

## A látássérültekért

Tavasszal 23 db hazai gyártmányú Brailab típusú gépet kaptak a látássérültek a Tudományszervezési és Informatikai Intézetből. A Brailab gépet a szocialista országok első gyártásba vett házi számítógépéből, a Homelaból alakították ki a Lukács testvérek. A gép minden lenyomott billentyűt számítógéppel szintetizált hanggal visszajelez, amit hangszórón vagy fülhallgatón keresztül hallhat a gép kezelője. A számítógépes beszédsszintézis feladatot a KFKI szakemberei oldották meg.

A Vakok Országos Szövetsége tavasszal az új számítógépek alkalmazásba vételére tizhetes tanfolyamot indított. Ennek feladata egyrészt az általános számítástechnikai kultúra emelése, másrészt a tehetségkutató, a továbbtanulás előkészítése volt. Ez utóbbi vonatkozásában mind a SZÁMALK, mind az ELTE kész a segítségre. Például a SZÁMALK már az elmúlt ősszel beiskolázott egy vak hallgatót.

A számítógép-alkalmazási program célja, hogy a vakok számára újabb munkahelyeket teremtsen. Szóba jöhetnek például a gépirói, a telexkezelői, a diszpécser és az adatbázis-kezelői munkakörök.

Személyiszámítógép-javítás, karbantartás  
közületeknek, magánsemeleyeknek.  
Egyedi megrendelés alapján  
kiegészítő berendezések gyártása.  
Pl. Sinclair fényceruza, Joystick Interface,  
oktatási intézménynek kabinet kialakítása.

Pásztor Ferenc személyiszámítógép-  
javító és -karbantartó kisiparos.  
Szolnok, Mátyás király u. 2. V/3.

PRODUKTORG  
SZERVEZÉSI VÁLLALAT

## C64, M08X, IBM PC XT kompatibilis rendszerekre ajánljuk programcsomagjainkat:

- főkönyvi könyvelés,
- költségfelosztás,
- állóeszköz-nyilvántartás,
- rendelés-nyilvántartás,
- bérelszámolás,
- adóelszámolás.

**Bővebb felvilágosítással  
készséggel állunk rendelkezésükre!**

*Bővebb  
felvilágosítással  
készséggel állunk  
rendelkezésükre!*

## SZÁMÍTÓKÖZPONT- VEZETŐK!

Ne dobják ki sérült  
mágneslemezeiket!

Gyorsan,  
olcsón,

6 havi garanciával megjavítjuk!

Commodore 64 számítógépükhöz  
— az adatögzítést megkönnyítő —  
kiegészítő numerikus klaviatúrát  
raktárról szállítunk,  
1 éves garanciával.

Megvásárolható  
az Econorg

1. sz. számítástechnikai  
szaküzletében:

Bp. VI., Szinyei Merse Pál u. 1.  
Tel.: 127-628

Égészségügyi Elektronikai Gm  
1045 Bp., Erzsébet u. 14. X. 59.

Telefonügyelet: Grósz Andor, 632-720 (9—17 óráig)



KI A LEGFONTOSABB EGY  
IRODÁBAN?

a VT 16  
a titkárnő  
titkárnője

Természetesen a titkárnő.  
Leveleket megír és átír, címekeket tart  
fejben, nyilvántartja a hivatalos iratokat.  
A tipográfiához is értenie kell, hiszen  
szerkeszteni, tördelni kell a levelek, irat-  
ok szövegét.  
Elvégzi az ismétlődő szövegrészek és  
az első példányok újragépelésének  
munkaigényes feladatát.  
Olykor azonban a legtökéletesebb tit-  
kárnő is elfárad.  
És máris ott a hiba lehetősége.  
Hacsak!  
Hacsak nincs a keze ügyében a VT 16  
számítógép, „akire” nyugodtan rábízhat-  
ja magát. Mert

Magyar karakterkészletes billentyűzet  
és nyomtató, 256 kb-át memória.  
Nagy grafikus felbontású, szemkímélő  
képernyő.  
Ara: alapgép floppy diszkkal  
260 000,— Ft  
Winchester diszkkal 440 000,— Ft  
Országos vevőszolgálati hálózat!  
Felvilágosítás: 804-133.

**A VIDEOTON PROGRAMJA – A JÖVŐ PROGRAMJA**

