MINI PLUS-Comp BASIC fordító

Talán mire megjelennek ezek a sorok, már nem is számít újdonságnak a hír: elkészült a PLUS-Comp, a Plus/4-es BASIC fordítóprogram kazettás verziója. (A lemezes változatot lapunk ez évi 4. számában ismertettük.)

Tudomásom szerint ez az első olyan Commodore BASIC fordítóprogram, amely nem mágneslemezzel, hanem kazettával működik. Bízom benne, hogy nem az utolsó, de erről majd később . . .

Miben különbözik a MINI a PLUS—Comp-tól? Elsősorban a lefordítható BASIC program nagyságában:

	Lemezes	Kazettás
A BASIC sorok max. száma	1600	310
A vezérlésátadó utasítások száma	2000	400
A programméret	kb. 25 kbájt kb. 12 kbájt	15 kbájt (nem graf.) 7 kbájt (grafika)

A másik különbség a munkaterület helye. A PLUS—Comp ugyanis mágneslemezen tárolja a munkafájlokat, a MINI esetében pedig a memóriában készül a futtatható P-kódú program. Kazettára csak a hibafájl íródik, ha ezt a fordítás előtt kérte.

A fordítóprogram, hasonlóan a „nagyobb testvérhez", elsősorban a CBM 2.0-ás BASIC verziót támogatja, de direktívák segítségével a 3.5-ös verzió is beépíthető a P-kódú programba.

Mivel a fordító felülről lefelé (top-down), balról jobbra (left-right), visszalépés nélküli algoritmussal dolgozik, ha az elemzett BASIC kifejezés legelső tagja nem BASIC 3.5-ös utasítás, direktívával meg kell jelölni, hogy a fordító helyesen építse be a P-kódú programba. Az így megjelölt sor nem tartalmazhat vezérlésátadást és FOR . . . NEXT ciklust (FOR, NEXT, STEP).

A direktívában álló kifejezés vagy sor szintaktikus elemzése csak futás közben folyik. Így fordításkor ezekben a részekben még maradhat hiba. Általánosan igaz, hogy csak alaposan tesztelt programokat érdemes lefordítani.

A lefordítandó BASIC programban a fordító nem használhatja az alábbi utasításokat:

CONT, LIST = a P-kódból származó korlátozás
ELSE
DO UNTIL/WHILE EXIT LOOP
RESTORE n
TRAP/TRON/TROFF/RESUME

A fejlett programozási technikáról való lemondás az ára a mintegy 4—12-szeres sebességnövekedésnek. A legnagyobb sebességet egész típusú FOR ciklusba ágyazott GOSUB-oknál mértem: ez kb. 25-szörös volt a BASIC-hez képest.

A fordító ezek alapján Commodore 64-es programok Plus/4-re átírt változatainak fordítására kiválóan alkalmas. A direktívakezelés elsajátítása kezdő programozónak több időbe telhet, mint azoknak, akik már dolgoztak fordítóprogramokkal. Ezt a kellemetlenséget egy készülő előfordító fogja megszüntetni, amely a lefordítandó BASIC programot automatikusan fordítható állapotba alakítja.

Tervezés alatt áll egy 15 kbájtnál nagyobb programok fordítására alkalmas Plus/4-es fordító, amely a munkafájlokat kazettára írva hozná létre a P-kódú programot. Természetesen ez több időbe telik, mint a memóriában való fordítás. Ha igény lesz rá, TURBO MPC néven kerül majd forgalomba.

Egy bonyolult, sok feltételt tartalmazó program futási ideje BASIC-ben 5 óra volt, ami a lefordított program esetében majd 30 percre csökkent. Ekkora különbséget már mindenképpen meg lehet érezni!

Mivel a MINI fordító készítésénél sok olyan problémát meg kellett oldani, amely fellépne Commodore 64 és C-1530 esetében is, ha megfelelő igény lesz rá, a C64-es verzió elkészítése is szóba jöhet. A leendő fordító érdekessége, hogy a lefordított program a BASIC munkaterületből kevesebb helyet foglal, mint BASIC-ben, függetlenül a BASIC program nagyságától — ellentétben a közismert C64-es BASIC fordítókkal, ahol a szükséges RUN—TIME modul egysoros program esetén is a program elé generálódik, minimálisan 17 blokknyi helyet foglalva a BASIC munkaterületből.

BÁRÓ CSABA

A ROM-ban tárolt aritmetikai rutinok használata

TVC

A TV—Computeren az assembler hiánya miatt csak rövid gépi kódú programok írására vállalkozhatunk. A sokat használt ciklusmagok gépi kódban való megírásával azonban csökkenthető a futási idő, és olyan programok is futtathatók lesznek, melyek tisztán BASIC-ben megírva igen lassúak. A ROM adatmozgató és aritmetikai rutinjaival gépi kódú programunkat szubrutinhívások sorozatává rövidíthetjük. Ezen a módon tetszőlegesen bővíthető a beépített függvénykészlet is.

Amint a gép leírásából kiderül, a TV—Computer az aritmetikai műveleteket a BASIC veremben végzi, és ez a helye a beépített függvények közötti paraméterátadásnak is. A verem utolsó elemére az IY regiszter mutat. A programírás során — különösen ciklusoknál — mindig figyelni kell ennek helyes állítására.

A lebegőpontos számokat a gép háromféle számábrázolási formátum szerint kezeli, amit az ábrákon szemléltetünk. Az ábrázolás bájtonként történik. Egy mantisszabájt két BCD számjegyet tartalmaz.

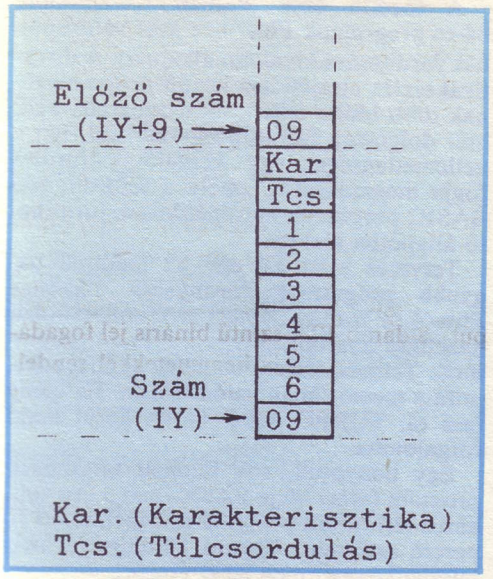
Az *1. ábrán* a verem látható. A számok magukban foglalják a túlcsordulásbájtot, amellyel együtt a verembe elhelyezett számok 14 decimális számjegy pontosságúak (egy bájt = két decimális számjegy).

Az ún. regiszter formátumú számoknál nincs túlcsordulásbájt *(2. ábra)*. A gép két beépített aritmetikai regiszterét (X és Y) a beépített függvények használják; a processzor HL és DE regiszterpárjainak megfelelő állításával több ilyen regiszter kialakítható a gép memóriájában.

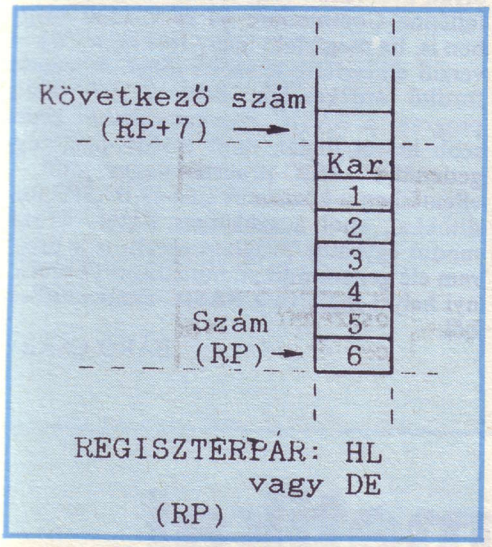
A *3. ábra* az X és Y belső aritmetikai regiszterek felépítését, a *4. ábra* a szimbólumtáblázatban (változótáblázatban) levő számok formátumát mutatja be. Ez utóbbiak már csak 10 decimális számjegy pontosságúak, ami egyben a megjelenítés pontossága is. A különböző formátumú számok közötti konverziót a ROM adatmozgató rutinjai végzik, a szubrutinhívások során erre nem kell ügyelni.

A szubrutinok vagy a szokásos módon, CALL utasítással, vagy táblázatból való kijelöléssel hívhatók. Ilyenkor a szubrutin táblázatbeli sorszámát kell megadni az alábbiak szerint:

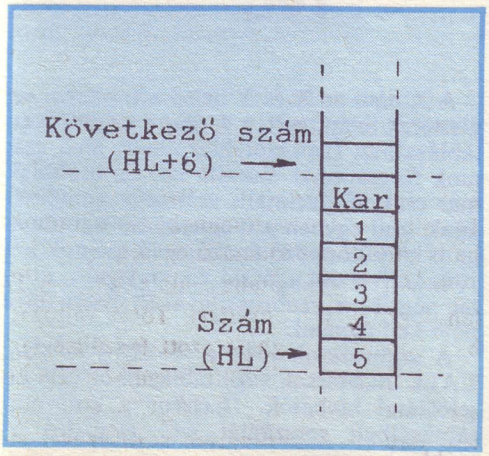
RST 18H	A táblázat aktivizálása
sorszám 1	Az 1-es sorszámú szubrutin végrehajtása
sorszám 2	A 2-es sorszámú szubrutin végrehajtása



1. ábra. Az aritmetikai verem



2. ábra. Egy tetszőlegesen kialakítható aritmetikai regiszter



3. ábra. Szimbólumtáblázat-elem

.
.
.

sorszám n Az n-edik sorszámú szubrutin végrehajtása után a szubrutinlista végét az jelzi, hogy a sorszám legnagyobb helyiértékű (128-as súlyú) bitje 1-est tartalmaz. A processzor az ezután következő kódot már utasításként értelmezi, és végrehajtja.

A hívható adatmozgató szubrutinokat a *táblázat* tartalmazza.

A szubrutin hívása	Hová	Honnan	Megjegyzés
CALL EAA2H	Verem (IY)	Regiszter (HL)	IY=IY−9
CALL EAC6G	Regiszter (DE)	Verem (IY)	IY=IY+9
CALL EAD5H	Regiszter (DE)	Verem (IY)	IY marad
CALL FA63H	Verem (IY)	Szimbólumtábla (HL)	IY=IY−9
CALL FB28H	Szimbólumtábla (HL)	Verem (IY)	IY=IY+9
RST 18H 06H	Verem (IY)	X regiszter	IY=IY−9
RST 18H 07H	Verem (IY)	Y regiszter	IY=IY−9
RST 18H 08H	X regiszter	Verem (IY)	IY marad
RST 18H 09H	Y regiszter	Verem (IY)	IY marad
RST 18H 0AH	X regiszter	Verem (IY)	IY=IY+9
RST 18H 0BH	Y regiszter	Verem (IY)	IY=IY+9

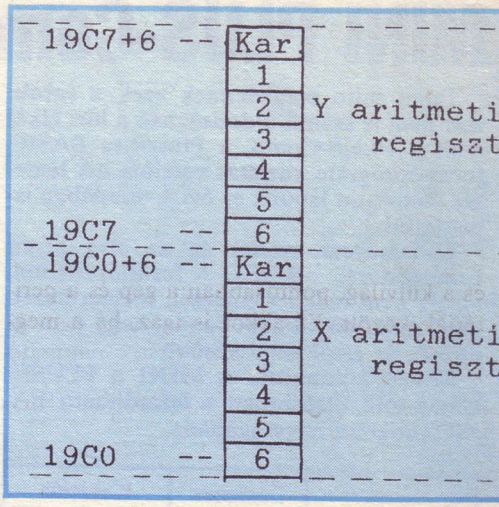
Példaként az alábbi programrészlet megcseréli az X és Y aritmetikai regiszterek tartalmát:

RST 18H
- 06H X regiszter tartalma a verembe (IY=IY−9)
- 07H Y regiszter tartalma a verembe (IY=IY−18)
- 0AH Veremből az X regiszterbe (IY=IY−9)
- 8BH Veremből az Y regiszterbe (IY=eredeti érték)

A 8BH hatásában megegyezik 0BH-val, de a szubrutinhívások végét jelzi.

1. lista

```
10 LOMEM 8000 : C=0 : D=0
20 FOR A=0 TO 7
30 READ B:POKE 7000+A,B:NEXT
40 C=VARPTR(B)
50 PRINT"Sorszám,Konstans"
60 FOR A=0 TO 39
70 POKE 7003,A: D=USR(7000,C)
80 PRINT A,B
90 IF INKEY$="" THEN 90
100 NEXT
110 DATA 229,223,133,39,225,
         195, 40,251
```



4. ábra. A beépített X és Y aritmetikai regiszterek

A táblázatban nem szereplő további adatmozgató rutin, az

RST 18H
0CH

hatására a veremben levő szám még egyszer rákerül a veremre, önmaga fölé, az IY pedig a másolt számra mutat.

Végül az utolsó adatmozgató rutin a ROM-ban tárolt, megadott sorszámú konstanst a verembe teszi. A sorszámok 0—39-ig terjedhetnek; a hozzájuk tartozó konstansok értéke az *1. listán* levő programmal határozható meg.

A következő programrészlet a veremben egymás fölé helyezi a 4-es, 23-as és a 16-os decimális sorszámú konstansokat:

RST 18H
- 05H A szubrutin sorszáma
- 04H A 4-es sorszámú konstans a verembe kerül (IY=IY−9)
- 05H A szubrutin sorszáma
- 17H A 23-as sorszámú konstans a verembe kerül (IY=IY−18)
- 85H A szubrutin sorszáma befejezéskor
- 10H A 16-os sorszámú konstans a verembe kerül (IY=IY−27)
- ... A processzor ezt a kódot már utasításként kezeli

A *2. lista* az USR függvény felépítését mutatja.

BOTTYÁN ZOLTÁN

2. lista

```
        1B58H   PUSH HL
                RST 18H
                85H
POKE 7003,A→    ...
                POP HL
                JP FB28H
```