A SZÖVEGSZERKESZTŐ

A szövegszerkesztőbe a monitorból az E paranccsal lépünk be. A monitorba visszalépni az ESC-pel lehet.

A szerkesztőbe csak a saját maga által előállított szövegfile-ok tölthetők be; ezek a file-ok más szövegszerkesztőben nem használhatók, nem nyomtathatók. (Ennek az az oka, hogy a sorokban egymás után álló szóközöket egy byteon tárolja. Így tömör, de máshol nem használható adatszerkezet jön létre.)

Egy sorban max. 58 karakter állhat. A szerkesztő üres sort nem fogad el, ezért minden sorban legalább egy, nem szóköz karakternek kell állnia.

A szerkesztett (módosított) sor akkor kerül be a memóriába, amikor a sorból kilépünk, ezért a RETURN gomb használata nem kötelező. A sor memóriába kerülését az jelzi, hogy sorszámot kap. A sorszámozás, átsorszámozás (sorbeszúráskor, - törléskor) automatikus. A keretszín megváltozása a memória betelését jelzi. Ilyenkor csak a túlcsorduló sor törlése után módosíthatunk a szövegben.

A képernyő alján és tetején scrollozás történik.

A BASIC editorban megszokott funkciók itt is müködnek, vagyis az INS, DEL, SHIFT DEL, CTRL I, CTRL K, CTRL N, a CTRL Y és a cursormozgatás.
A RETURN a következő sor 8. (mnemonik) poziciójára ugrik és C cursort állit be.

A kibövitett cursor-funkciók:

jobbra / balra		A a cursor a következő karakterre lép, szóközök esetén a következő tabulátor pozicióra.
SHIFT fel		A képernyö elsö sorára állás.
SHIFT le		A képernyö utolsó sorára állás.
SHIFT balra		Az 1. pozicióra állás.
SHIFT jobbra		Sorvégre állás(a következő üres pozicióra).
CTRL fel	(= CTRL E)	Lapozás visszafelé.
CTRL le	(= CTRL X)	Lapozás elöre.
ALT fel		A legelsö sorra állás.
ALT le		A legutolsó sorra állás.

Sorra állitó parancsok:

CTRL L	Sorszámot kérdez, majd rááll a megadott sorra.

```
48:
49:CTRL  F   : Bekéri a keresendő szót, majd a cursor sorától
50:           kezd keresni.Ha megtalálta rááll az adott sor-
51:           ra, egyébként csipog. A keresendő szó nem kez-
52:           ődhet és végződhet szóközzel!
53:CTRL  R   : Folytatólagos keresés a cursor sorától.
54:
55:                Blokk kezelő utasitások:
56:
57:  Lehetőség van a szöveg két blokkjelző (négy cursor négy-
58:zet) közé zárt soraival müveleteket végezni. Egyszerre
59:csak egy kijelölt blokk létezhet.
60:
61:CTRL  @   : Blokkjelző beszúrása.
62:           (a blokkjelzőt az öt tartalmazó sorral együtt
63:           lehet törölni.)
64:CTRL  A   : Az első blokkjelzőre állás, ha az létezik.
65:CTRL  Z   : A második blokkjelzőre állás, ha az létezik.
66:CTRL  -   : A kijelölt blokk törlése a blokkjelzőkkel
67:           együtt.
68:CTRL  C   : A kijelölt blokk bemásolása az adott sor elé.
69:           A blokkon belülre másolás tiltott!
70:CTRL  O   : A kijelölt blokk kinyomtatása sorszámok nél-
71:           kül. Ha a nyomtató helyén pl. magnó van, és
```

van irásra nyitott pufferelt file, akkor hagyományos szövegszerkesztöbe beolvasható állományt készithetünk.

Egyéb parancsok:

CTRL B : Hardcopy a képernyötartalomról.
CTRL U : UNDO. (A sor eredeti tartalmának visszaállitása, módositás után,ha a sorból még nem léptünk ki.)
: A cimkemezö (1-8. kar.) törlése, majd a következö sorra állás.
CTRL W : A teljes szöveg törlése, de csak a CTRL W háromszori lenyomása után.

A RENDSZER MEMÓRIAKIOSZTÁSA, RENDSZERPARANCSOK

A MNEMONIC IV. programnak két változata van, a BOTTOM a 19EFH-5BFFH, a TOP pedig 0BE21H-0FFFFH cimekre töltödik. (A további leirás a BOTTOM változatra vonatkozik, a TOP-hoz tartozó eltéréseket "[]" zárójelek között közöljük.)

A program I/O munkaterületnek a BASIC rendszerváltozók területét használja fel (pl.: BUFFER, COMMAND,stb.).

A maradék memória három részre osztódik fel:

- Kódpuffer: 5C00H-7FFFH [19EFH-39EEH].
  Alapállapotban az assembler ide helyezi a lefordított tárgykódot, és a monitor is itt használható.
- Szövegpuffer: 8000H-0DFFFH [39EFH-99EEH].
  A szövegszerkesztő tartalma található itt.
- Szimbólumtábla:0E000H-0FFFFH [99EFH-0BE20H].
  A fordítás során értéket kapott címkék táblázata.

A program BASIC ROM-ot nem használ, ezért a 2-3. lap [1-2. lap] szabadon lapozható.
A szövegszerkesztőhöz és a szimbólumtáblához tartozó RAM lapok be- és kilapozása automatikus.

Mint látható, a program bejelentkezés után 64k RAM-ot tételez fel, valamint fix pufferkiosztást vezet be. A program nem helyezhető át, de a pufferkiosztás igen. Erre szolgál a V parancs, amellyel pl. 32k-s gépen is használható a program. [természetesen csak a BOTTOM verzió!]

V n,nn : az n-edik rendszerváltozó értékét nn-re állítja, a puffer eredeti tartalma elveszik!
Figyelni kell a helyes paraméterezésre, az egymást átfedő pufferkiosztás a rendszer lefa-

gyását okozhatja!

V n : kiírja a változó aktuális értékét.

A 10 rendszerváltozóból az első 7 írható a V parancscsal, a többi 3 csak olvasható.

n értékei:

0: A kódpuffer kezdőcíme,
1: a kódpuffer vége,
2: a szövegpuffer kezdete,
3: a szövegpuffer vége,
4: a szimbólumtábla kezdete,
5: a szimbólumtábla vége,
6: bit 0=1 : szimbolikus debugger, disassembler bekapcsolása. (Ha egy utasítás visszafordításakor annak címe, vagy valamely paramétere egy létező szimbólum értékével azonos, akkor a számérték helyett a talált szimbólum nevét írja ki a disassemblált sorban.)
bit 1= bit2= 1: A szimbólumtábla rögzítése az assembler fordítás előtt. (ld. később részletesen!)
bit 3=1 : lépésenkénti nyomkövetéskor a re-

gisztertükör mindig a képernyö tetejére íródik, egyébként az adott cursorpozíció sorába.
bit 4 - bit 15 : makrobövítés céljára, jelenleg nem használt.

7 az utoljára kiadott ORG értéke. (Csak olvasható!)

8 a) az assembler fordítás után az elsö szabad byte fizikai címét veszi fel,
b) memória terület beolvasása után (ld. R utasítás) a beolvasott file utáni elsö byte fizikai címét tartalmazza a valtozó. (Csak olvasható!)

9 a címzési eltolás értéke. (Csak olvasható!)

A kódpuffer címzése eléggé speciális. Alaphelyzetben ugyanis csak a kódpuffer tartalmát írhatjuk-olvashatjuk, így elkerülhetö a többi puffer és a MNEMONIC IV program felülírása. A pufferen kivülre címzésre a "Kifutás a tárból!" hibaüzenet figyelmeztet.

A kódpufferben lehetöség van címzési eltolás használatára, erre szolgál az ORG nn utasítás.

Lássunk erre egy példát:

Betöltünk a kódpuffer elejére egy gépi kódú programot. Így a program fizikai kezdöcíme 5C00H. A programról tud-

juk, hogy valójában a 19EFH címen kezdödik, ezért kiadjuk az ORG 19EFH parancsot. A monitor ezután már a 19EFH logikai címen látja a kódpuffert. Ezután pl. a disassemblálás, dumpolás címhelyesn történik. Természetesen ez a program nem indítható el, mert a fizikai és a logikai címe különböző.

Az ORG nn utasitás a monitorban és az assemblerben is használható. Assembler fordítás során csak egy ORG lehet a programban, annak is az elején. Az ORG nn kiadása után a kódpuffer kezdetét az nn logikai címen látjuk.

ENT vagy ENT $ — A teljes 64K memória elérését teszi lehetővé, a címzési eltolást MEGSZÜNTETI. Az assemblerben és a monitorban is használható.

A BASIC-be a

Q — utasitással léphetünk ki. A kilépéskor lezáródnak a nyitott file-ok és meleg reset történik. A BASIC az 5C00H-0BFFFH-ig [19EFH-0BE20H-ig] terjedö memóriát kapja vissza. Az elözöleg az erre a terü-

letre lerakott adatok megsérülnek!

A MNEMONIC IV.-be az EXT 0 utasítással térhetünk vissza:

EXT 0	A felhasználói regisztertükör törlődik, a puffertartalmak megmaradnak.
EXT 0,11111	A felhasználói regisztertükör, a törésponttáblázat, a szimbólumtábla, a szövegszerkesztő tartalma, valamint a COM_DEV a REG_DEV és a BRK_DEV változók tartalma is törlődik. A V paranccsal definiált pufferkiosztás megmarad.
EXT 0,22222	Hideginditás; azonos a program újra betöltésével.

SZINTAXIS,RÖVIDITÉSEK

A monitor és az assembler max. 58 karakterből álló sorokat fogad el.

A sorok felépítése:

CÍMKEMEZŐ: Az utasításhoz rendelt címke helye, ha az létezik. A címke a jobbra mellette álló assembler utasítás memóriacímét veszi fel a fordításkor, értéke később nem változtatható meg. A címkenévnek a sor elején kell kezdődnie, előtte szóköz nem állhat. A szimbólumnév max. 7 karakter hosszú lehet, melyben betük, ékezetes betük és számok lehetnek, de az első karakter nem lehet számjegy! A címkenév nem lehet azonos regiszternévvel, vagy feltételszóval (C, NC, Z, NZ, PO, PE, P, M)! A címkenévben a kis- és nagybetük különbözőek, ezért ugyanaz a név kis- vagy nagybetüvel irva, más-más címkét jelöl.

A monitorban címke nem definiálható, de a már definiáltakra hivatkozhatunk.

UTASITAS: Z80 utasitás, monitorparancs, vagy assembler direktiva neve. Az utasitásnévben a kisbetük használata tiltott. Az assembler forrásszövegben az utasitások elött (a cimke és az utasitás között) legalább egy szóköznek kell állnia. A monitorban az utasitás elötti szóköz nem kötelező, de megenegedett.

PARAMÉTEREK: Az utasitáshoz tartozó első paramétert az utasitástól egy, vagy több szóköznek kell elválasztania. Egymástól a paramétereket vesszö választja el. A paraméterben álló regiszternevek és feltételszavak csak nagybetükkel irhatók!

MEGJEGYZÉS: A pontosvessző után álló feliratokkal a fordító és a monitor nem foglalkozik, azokat megjegyzésnek tekinti.

A csak megjegyzést tartalmazó sor használata megengedett.

```
264:
265:        Íme egy példa:
266:ALFA    CALL    NZ,BÉTA;         feltételes szubrutinhivás
267:
268:
269: Számkonstansok használata, és az alapértelmezés szerinti
270:                  számrendszer fogalma
271:-------------------------------------------------------------
272:
273:  A Mnemonic IV rendszer négyféle számrendszert ismer fel:
274: 1.) binárist        (2-es alapút):  xxxb, xxxB vagy %xxx;
275: 2.) oktálist        (8-as alapút):  xxxo vagy xxxO;
276: 3.) decimálist      (10-es alapút): xxxd, xxxD vagy .xxx;
277: 4.) hexadecimálist  (16-os alapút): xxxh, xxxH vagy #xxx;
278:   Az "xxx" a számjegyeket jelöli. Ha a  számjegyek előtt,
279:vagy után nincs módosítójel, akkor a szám az  alapértelme-
280:zett számrendszer szerint értékelődik ki!
281:   A MONITOR alapértelmezett számrendszere  a  HEXADECIMÁ-
282:LIS, az ASSEMBLER FORDÍTÓÉ DECIMÁLIS! Alapértelmezett  de-
283:cimális számrendszernél minden számot decimálisan,  egyéb-
284:ként hexadecimálisan ír ki a rendszer.
285:   A hexadecimális számok számjegyeket, és  "abcdef"  vagy
286:"ABCDEF" betűket tartalmazhatnak, és # jellel, vagy  szám-
287:jeggyel kell kezdődniük!
```

Így a #ffbc, a 0FFBCH és a 0ffbc alak helyes,az FFBC és az ffbch pedig hibás!

Alapértelmezett hexadecimális számrendszer esetén az xxxb, xxxB, xxxd és xxxD alak természetesen hexadecimális számot jelöl!

Rövidítések:

nn,mm : 16 bites aritmetikai kifejezés.

n,m : 8 bites aritmetikai kifejezés.

Az aritmetikai kifejezések egymástól müveleti jelekkel elválasztott, max. 16 bites tagokból állhatnak.

A kiértékelt kifejezés értéke 16 bitesre csonkitódik. A tagok és müveleti jelek közötti felesleges szóköz tiltott!

A lehetséges tagok:

$	:Az aktuális memóriamutató értéke.
Szimbólumnév	:az első fordítási menetben értéket kapott szimbólum (cimke, EQU konstans), vagy SET változó értéke.

```
312:  Regiszternév  :az elmentett felhasználói regiszterérték.
313:                 Csak monitorban használható, itt is  csak
314:                 kódot nem generáló parancsokban! (Ld. pl.
315:                 debugger parancsok.)
316:  Számkonstans  :lásd feljebb!
317:  ASCII konstans:"a", 'OK' stb. A " jel '-ok között írható
318:                 le, és viszont. Pl.: '"', "'" .
319:
320:                      A lehetséges müveleti jelek:
321:
322:(, )             : zárójelek.
323:                   (pl.: LD E,(n+m)*3 )
324:+               : összeadás, vagy pozitiv elöjel.
325:                   (pl.: LD HL,CIMKE+1 )
326:-               : kivonás,  vagy  negativ  elöjel  (kettes
327:                   komplemens képzése).
328:                   (pl.: LD HL,LAB2-LAB1;  LD A,-3 )
329:NOT             : egyes komplemens képzése.
330:                   (pl.: LD B,NOT 9 )
331:*, /, MOD      : szorzás, osztás és osztási maradék  kép-
332:                   zése.
333:                   (pl.: LD HL,DATA*6;  LD A,28 MOD 3 )
334:AND, OR, XOR   : bitenkénti és, vagy, kizáró vagy müvelet
335:                   (pl.:  LD A,n XOR m )
```

```
SHL, SHR        : bitenkénti balra, illetve jobbra lépte-
                  tés. A belépö bitek 0-át tartalmaznak.
                  (pl.:  LD A,2 SHL 3   : A-ba 16-ot tölt;
                  LD HL,83ACH SHR 4 : HL-be 083AH-t tölt.)
LOW, HIGH       : alsó, illetve felsö byte kiemelése.
                  (pl.:  LD A,LOW 83ACH : A-ba 0ACH kerül,
                  LD A,HIGH 83ACH : A-ba 83H kerül.)
<  vagy LT      ┐
>  vagy GT      │ Összehasonlitások. Ha a reláció igaz,
<= vagy LE      ├ akkor -1 -et, ha hamis, akkor 0-át ka-
>= vagy GE      │ punk eredményül. (A logikai müveletek a
=  vagy EQ      │ feltételes fordítást vezérlö utasitások-
<> vagy NE      ┘ ban használhatók fel leginkább.)

   A NOT, MOD, AND, OR, XOR, SHL, SHR, LOW, HIGH, LT, GT,
LE, GE, EQ és NE operátorok elött és után egy és csak egy
szóköz irása kötelezö!

   A müveletek végrehajtási sorrendje (prioritása):
1.:     (, )
2.:     *, /, MOD, elöjelek, NOT, LOW, HIGH, SHL, SHR
3.:     +, -
4.:     AND
5.:     OR, XOR
```

6.: <, >, <=, >=, =, <> (LT, GT,LE, GE, EQ, NE)

A MONITOR HASZNÁLATA

A monitorban találhatjuk a memória vizsgáló,- felüliró, I/O és debugger funkciókat, továbbá innen érhető el a szövegszerkesztő és az assembler fordító is.

A monitor sorassemblerként is müködik, azaz a beirt Z80 assembler utasitásokat azonnal lerakja a memóriába, de itt címkéket nem definiálhatunk.

A monitor full screen editor módban müködik, azaz a cursorral szabadon lehet mozogni, módositani a képernyön. A monitor az utasitássor első öt, inverz karakterét figyelmen kivül hagyja, 58 karakternél hosszabb logikai utasitássort nem fogad el!

Az utasitás végrehajtása után a következő sor elejére inverzben kiirja az új memóriamutató értékét ($) az aktuális számrendszer szerint.

? Számkonverziók:

-

```
384:        ?                 :  Az aktuális számrendszer cseré-
385:                           je. Nem decimálisból decimálisba,
386:                           decimálisból hexadecimálisba vált
387:        ?   nn            :  Az nn kifejezés értékének  dec.
388:                           és hex. kiírása. Ha a szám  ábrá-
389:                           zolható 8 biten  és/vagy  negatív
390:                           értékként, akkor úgy is kiírja.
391:
392: J nn                     :A memóriamtató ($) nn címre álli-
393:------                      tása.
394:
395:+                          : $:=$+1┐
396:á                                 ├  a cursor helyben marad,
397:-                          : $:=$-1┘ így a funkció ismételhető
398:
399: M                           Memória másolás:
400:---
401:        M nn,mm           : Az nn címtől mm címig tartó memó-
402:                            riarészt a  $  címre másolja. Az
403:                            átfedés megengedett!
404:                            mm>=nn !
405:
406:INS                         Beszúrás a memóriába:
407:---
```

```
408:        INS nn,mm       :  A $ cimtől nn-ig tartó memóriát
409:                           mm db. byte-tal feljebb helyezi.
410:                           A felszabaduló részt 0-val tölti
411:                           fel. nn>=$ !
412:
413: S nn                    :  A $ cimtől nn cimig tartó memó-
414:------                     riáról 24 bites ellenörző össze-
415:                           get ir ki. nn>=$ !
416:
417: C                        Memóriaterületek összehasonlitá-
418:---                        sa:
419:        C    nn         :  A $ cimen kezdödö memóriát ösz-
420:                           szehasonlitja az nn cimen kezdö-
421:                           dövel.  Eltérés esetén  kiirja a
422:                           talált cimeket ,az ottlévö byte-
423:                           okat,  majd  billentyülenyomásra
424:                           vár:
425:                            SPACE : tovább ellenöriz;
426:                            ESC   : kilép;
427:                            CTRL B: HARDCOPY a nyomtatón!
428:                           Egyezö blokkok esetén  egy  üres
429:                           sort ir ki.
430:
431: F                        Byte-sorozat keresése a memóriá-
```

```
---                          ban:
F n,"felirat",m,...        : Megkeresi az adatok első előfor-
                             dulását a $ címtől kezdve. Sike-
                             res  keresésкor  kiírja a talált
                             címet, majd  billentyűlenyomásra
                             vár :
                               SPACE  : tovább keres;
                               ESC    : kilépés
                               RETURN :  DUMP-ot  ad a talált
                                         címtől
                               egyéb  :  kilép, és a $-ba má-
                                         solja a talált  címet

L                            Disassemblálás:
---
        L                  : A $ címtől listáz
        L nn               : Az nn címtől listáz.
                             Egy sor kiírása után  billentyü-
                             lenyomásra vár:
SPACE                : újabb sort listáz
ESC                  : kilépés
^                    : kilépés, az utolsó listázott sor címe a
                       $-ba másolódik.
@                    : DB n listázás ki-bekapcsolása.
```

```
456:                    A bekapcsolt állapotot a keretszín jel-
457:                    zi.
458: INS               : Az EDITOR-ba listázás  ki-bekapcsolása.
459:                    Az így listázott sor a  szövegszerkesz-
460:                    tőbe is bekerül. A bekapcsolt állapotot
461:                    egy cursor-négyzet jelzi a sorban.
462: RETURN            : JP nn, JR nn,  CALL nn  utasitás esetén
463:                    a listázást az nn címen folytatja.
464:                     A CALL vagy  feltételes  ugró utasitás
465:                    címét megjegyzi (max. kb. 500-at).
466: DEL               : visszatér az előző,  a RETURN lenyomása
467:                    előtti címre, ha ott CALL vagy feltéte-
468:                    les ugró utasitás állt, azaz csak elá-
469:                    gazási pontra tér vissza.
470: CTRL B            : Hardcopy a nyomtatón.
471:
472:
473: D                            Memória listázása,felülirása:
474:---
475:         D                 : DUMP a $ címtől.
476:         D nn              : DUMP az nn címtől.
477:                    A képernyö tetején: a listázott 256 byte
478:                    kezdete,vége és az aktuális lapkiosztás.
479:                    A képernyö alján:  (mm)=nn;n  ahol
```

```
                    mm : a cursor pozíciója a memóriában;
                    nn : a cursor címén lévő 16 bites;
                    n  : a cursor címén lévő  8 bites ér-
                         ték.

                       A használható funkciók:
CTRL B             : hardcopy a nyomtatóra.
INS                :  ASCII karakterek 7. bitjének ki-be-
                      kapcsolása.
botkormány         : mozgás a memóriában :
     SHIFT-tel     : négyesével mozog;
     CTRL fel/le   : a megelőző/következő 240 byte.
ALT jobbra/balra   :  Grafikus dump szélességének válto-
                      tatása. A graf. dump a cursor  pozí-
                      citól kezdődik. A grafikus dump  se-
                      gítségével lehet a memóriában  ábrá-
                      kat (pl. sprite-okat) keresni.
RETURN             :  a cursor  vált az  ASCII és a hexa-
                      mező között.  A kijelölt cím  csak a
                      villogó cursor  helyén írható felül!
                      A hexadecimális mezőben az A-F szám-
                      jegyek csak nagybetűvel írhatók!
ALT RETURN         :  GRAPHICS mód csere. (csak ha a gra-
                      fikus dump be van kapcsolva.)
```

```
504:   CTRL ^            :  a kurzor helyén álló érték szerinti
505:                       címre ugrás.
506:   DEL               :  visszaállitja a felülirt cimek ere-
507:                       deti tartalmát az aktuális DUMP kép-
508:                       ernyön.
509:   ESC               : visszalép a monitorba (vagy  a  FIND
510:                       parancsba).
511:
512:
513:                         A monitor I/O - kezelö funkciói:
514:                         ================================
515:
516: I                       Az IN_TABLE olvasása,módositása:
517:---
518:        I n              : kiirja az IN_TABLE n. elemét.
519:        I n,m            : az IN_TABLE  n. elemébe  m érté-
520:                           ket ir.
521:
522: O                       Az OUT_TABLE olvasása, módositá-
523:---                     sa:
524:        O n;   O n,m     :    ua. mint az I-nél.
525:                        Az IN- és OUT_TABLE  módositása-
526:                        kor a program  nem  ellenörzi  a
527:                        beirt adat helyességét!
```

```
PORT                    I/O port írása,olvasása:
----
        PORT nn         : kiírja az nn portról olvasott
                          értéket.
        PORT nn,n       : az nn portra n értéket küld.
                           (Megjegyzés: a TVC beépített
                          portjai csak 8 bites portcímeket
                          használnak.)

 H                      : Hardcopy a nyomtatón.
---                       (Nem grafikus !)

 OS                       RST 30h,n rutin végrehajtása:
----
        OS n            : DE-be a $ fiz. címét, BC-be 0-át
                          tölt,majd végrehajt egy RST 48,n
                          utasítást.
        OS n,nn         : DE-be a $ fiz. címét, BC-be nn-t
                          tölt,majd RST 48,n.
                          Mindkét esetben kiírja a vissza-
                          kapott értékeket. A<>0 esetén
                          hibát jelez.
```

```
552: PAL                        paletta-értékek módosítása:
553:-----
554:        PAL                : kiírja az aktuális (elmentett)
555:                             palettaértékeket.
556:        PAL param.         : módosítja a paletta aktuális ér-
557:                             tékét a paraméterekben megadott
558:                             értékekkel.
559:       pl.:PAL 0           : a papírszint változtatja
560:           PAL 0,16,85,4:  mind a négy palettaszint módo-
561:                             sítja.
562:
563: CLS                        képernyő törlés:
564:-----
565:        CLS                :      -II-
566:        CLS 0              : GRAPHICS 2
567:        CLS 1              :   -II-   4
568:        CLS 2              :   -II-  16
569:
570: P                          memória lapozás:
571:---
572:        P                  : kiírja az aktuális lapkiosztást.
573:        P n                : lapozás az n érték szerint.
574:                             (csak a 2-3. [1-2.] lapra hajtó-
575:                             dik végre!)
```

```
576:
577:OPEN                       file  nyitása  írásra/olvasásra:
578:----
579:        OPEN W           : pufferelt file  nyitása  írásra,
580:        OPEN W,file-név  : név nélkül, illetve névvel.
581:        OPEN R           : file nyitása olvasásra név  nél-
582:        OPEN R,file-név  : kül, illetve névvel.
583:
584:CLOSE                      file zárása:
585:-----
586:        CLOSE R          : olvasott file lezárása.
587:        CLOSE W          : írt file lezárása.
588:
589: W                         memóriaterület mentése:
590:---
591:        W file-név       : A lefordított program  kimentése
592:                           (V 7 -től V 8-1 címig).
593:        W file-név,nn    : A $-tól nn címig  tartó  memória
594:                           kimentése.
595:        W file-név,SYM   : a szimbólumtábla kimentése.
596:        W file-név,TXT   : az assembler forrásszöveg kimen-
597:                           tése.
598:
599: R                         memóriaterület betöltése.
```

```
600:---
601:        R               : betöltés a kódpuffer elejére,
602:        R file-név      : név nélkül, vagy névvel. Csak
603:                          annyi byte-ot tölt be, amennyi
604:                          bele is fér a kódpufferbe. (V 8
605:                          értékét beállitja a file utáni
606:                          első byte fizikai címére!)
607:        R file-név,nn   : A $-tól nn címig tartó terület-
608:                          re töltés.  (V 8-at beállitja!)
609:        R file-név,SYM  : új szimbólumtábla betöltése; a
610:                          régi törlödik. Az esetleges elö-
611:                          zö tábla rögzitést feloldja!
612:        R file-név,TXT  : hozzátöltés a már meglévő for-
613:                          rásszöveghez.
614:
615:           Az R és a W az U3 RAM területén is használható,
616:        ilyenkor a képernyön zavaros ábra jelenik meg, de
617:        ez nem hiba! (A videomemória átmeneti adattároló-
618:        ként üzemel ilyenkor.)
619:        [A TOP nem tud az  U3 RAM  terülületére  tölteni!]
620:
621:\                       : VT-DOS kompatibilis floppymeg-
622:-                        hajtó esetén belépés a BASIC
623:                          CLI-be.
```

```
622:-                          hajtó esetén  belépés a  BASIC
623:                           CLI-be.
624:\parancs                   DOS parancs végrehajtása
625:--------                       pl.:  \DIR b:*.ASM
626:                           Munkaterületnek a  definiálha-
627:                           tó karakterek  mátrixát  hasz-
628:                           nálja fel.
629:
630:              A folytatás a HELP2 file-ban található.
631:            Betöltése előtt törölni kell ezt a  szöveget!
632:                        (háromszori CTRL W)
633:
```

A DEBUGGER FUNKCIÓK:

A debugger funkciók lehetővé teszik a gépi kódra lefordított programok kipróbálását, tesztelését. Használatuknál a következőket kell figyelembe venni:

- A debugger közös stacket használ az operációs rendszerrel, ezért az U0 RAM-ot kilapozni tilos, az SP csak az U0 RAM-ba mutathat!
- A tesztelni kívánt program logikai és fizikai címének egyeznie kell, ellenkező estben "Nem használható!" hibajelzést kapunk!
- A Mnemonic IV rendszer vektoros interrupt lekezelésére nincs felkészítve, ezért a tesztelendő programban nem lehet IM 2 utasítás!

X — Az elmentett regiszterkészlet olvasása, módosítása:

- X 0 : Az összes regisztert törli; az SP-t alaphelyzetbe állítja (5800 dec.), a PC-be a $ értékét tölti.
- X : Kiírja a teljes regisztertükröt; valamint a PC címen lévő utasítás nevét, kódját. A PC érté-

```
25:           két az aktuális számrendszerben, a regisztere-
26:           ket hexadecimálisan, az F-et és  F'-öt  biten-
27:           ként írja ki. Az F melletti, név nélküli  adat
28:           a PAGE érték; az F' melletti, név nélküli adat
29:           az I regiszter értéke.
30:  X x     : Kiírja az x értékét.
31:  X x,n   : Az x-be 8 vagy 16 bites értéket tölt.
32:           x fajtái:
33:           ═══════
34: AF BC DE HL IX IY AF' BC' DE' HL' SP PC regiszterpár,vagy
35:az A F B C D E H L XH XL YH YL A' F' B' C' D' E' H' L' I
36:és IFF regiszter, ahol:
37:           IFF =0: DI üzemmód,
38:           IFF<>0: EI üzemmód.
39:
40:  A monitorban az "x"-re hivatkozás megengedett.
41:           pl.: D HL
42:                D PC
43:                L DE, stb.
44:
45: B         Töréspontok definiálása, törlése:
46:---
47:
48:     - A tesztelendő programban max. 8 db. töréspont hasz-
```

```
49:       nálható.
50:     - Töréspontnak RST 0 utasitást használunk, ezért ez
51:       másra nem használható!
52:     - A rendszer csak a B utasitással definiált töréspon-
53:       tot ismeri fel, a memóriába direkt beirt RST 0 uta-
54:       sitás hatása a Mnemonic IV EXT 0 utasitással törté-
55:       nő újrainditásával azonos!
56:     - Töréspont csak RAM-ban használható!
57:     - Több byte-os utasitás esetén a töréspontot az első
58:       byte-ra kell helyezni!
59:     - Lehetőség van úgynevezett többszörös töréspont de-
60:       finiálására, azaz olyan töréspont készithető, ahol
61:       csak az mm-edik átfutáskor áll meg a program. Igy
62:       megszámlálható az adott ponton való áthaladások
63:       száma.
64:
65:     B      : Kiirja az egész törésponttáblázatot:
66:              1. oszlop  a töréspont cime,
67:              2. oszlop  a max. átfutásszám,
68:              3. oszlop  a még hátralévő átfutások száma.
69:
70:     B    0 : Törli az egész törésponttáblázatot.
71:     B   nn : Az nn cimre egyszeres töréspont definiálása.
72:     B nn,mm: Az nn cimre mm-szeres töréspont definiálása.
```

```
                (Vagy a már meglévő töréspont  átfutásszámá-
                nak módosítása.)
      B nn,0 : Az nn címen levő töréspont törlése a  táblá-
                zatból.

 G              A program  elindítása,  folytatása  töréspont
---             után:
        A töréspontok a G  utasítás kiadása után kerülnek a
  programba (az akkor aktuális  lapkiosztás  szerint!),  a
  töréspont táblázat alapján.  A  monitorba  visszatéréskor
  visszaáll az eredeti tartalom, azaz a töréspontok eltün-
  nek a programból, de a töréspont táblázatban megmaradnak.

    G   nn    : Szubrutinhívás az nn címre. A verembe visz-
                szatérési cím kerül.

    G         : Ugrás az aktuális PC-címre. (nem  a  $  cím-
                re!) A stack-be nem kerül visszatérési cím!
                Leginkább töréspont utáni folytatáskor  al-
                kalmazható.

 T              Lépésenkénti  programfuttatás.  RAM-ban  és
---             ROM-ban is használható.  Kiadásakor  töré-
                spontok nem  kerülnek  a  programba.  Az  RST
```

```
  97:                utasitások rutinjaiba nem lép bele.
  98:      T       : Lépésenkénti végrehajtás a  PC címtől. Min-
  99:                den lépés előtt kiirja a regisztertükröt, a
 100:                soronkövetkező utasitást, majd billentyü
 101:                lenyomására vár:
 102:                 ESC   : vissza a monitorba.
 103:                 CTRL  : utasitás végrehajtása,  a  szubru-
 104:                         tinhivásokba "belelép".
 105:                 ALT   : utasitás végrehajtása,  a  szubru-
 106:                         tinhivásokba nem lép  bele, azokat
 107:                         egy utasitásnak értelmezi.
 108:                 -     : a soronkövetkező utasitás átugrása
 109:
 110:      T nn    : A PC-címtől nn darab utasitás végrehajtása.
 111:                A regisztertükröt csak a lefutás  után irja
 112:                ki. A  billentyüzetet  most  is  figyeli! A
 113:;
 114:                program futási sebessége a  normálisnak kb.
 115:                1/2000-ed része !
 116:                   FIGYELEM az nn nem a kezdő  cim, hanem a
 117:                végrehajtandó utasitások száma !!!
 118:
 119:        Az alábbi utasitások mind az assembler forditóban,
 120:       mind a monitorban használhatók. Az adatok a $ címtől
```

```
121:      kerülnek lerakásra.
122:
123:LAB                      :  A szimbólumtábla elemeinek lis-
124:---                       tázása, a definiálás  sorrendjé-
125:                          ben. A listázás bármely billen-
126:                          tyü lenyomásával szüneteltethe-
127:                          tő, illetve folytatható, ESC le-
128:                          nyomására  megszakad.  Az érték
129:                          melletti jel a szimbólum tipusát
130:                          jelzi:
131:                           L : cimke,
132:                           < : EQU-val definiált konstans,
133:                           \ : SET-tel definiált változó.
134:
135:RADIX nn                  : Az aktuális számrendszer alapját
136:                          nn-re cseréli. nn  csak 2, 8, 10
137:                          vagy 16 lehet!
138:
139:DB n,"felirat",m,...      : 8 bites kifejezések és feliratok
140:DB 'szöveg',n,n..          lerakása a memóriába.
141:DW nn,mm,nn,mm,...       : 16 bites kifejezések lerakása  a
142:                          memóriába, alsó-,felsö byte sor-
143:                          rendben.
144:DS nn                   : A memóriába nn darab  0  byte-ot
```

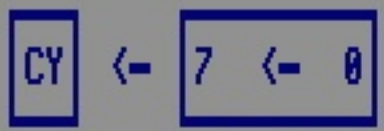
```
                          helyez.
D$ n,"felirat",m,...    : Elhelyezi az adatbyte-ok  számát
                          egy byte-on, majd magát az 'n,"f
                          elirat",m,...' adatokat. Igy le-
                          het pl. egyszerüen file-neveket
                          lerakni.
FILL nn                 : A memóriát az nn cimig  0  byte-
                          tokkal tölti fel.
FILL nn,n,'felirat',m,..: A memóriát az nn cimig  feltölti
                          az  n,'felirat',m,... adatokkal.
     A FILL és a DS utasitások nn paraméterében cimkene-
   vet ne használjunk, ez forditási fázishibához vezethet!

                AZ RST p UTASITÁSOK SZINTAKTIKÁJA

    A p az AKTUÁLIS SZÁMRENDSZERnek  megfelelö  szám, vagy
  használjuk a számrendszer jelöléseket!  (pl.  30H,  .48)
    Az RST 8 (hibaüzenet) és RST .48  (operációs  rendszer
rutin hivása) utasitások kétparaméteresek. A második para-
méter a funkciókód.
       Pl.: RST 8,1; "Not understood" hibajelzés
            RST .48,.33; egy karakter kiirása.
```

A nem szabványos Z80 utasítások használata

Mind a monitorban, mind az assemblerben elérhetjük az IX és az IY regiszterpárt regiszterenként is. Az így elérhető alsó regiszterek neve XL és YL, a felsőké XH és YH. Ezek a regiszterek a bitműveletek kivételével mindenhol használhatóak, ahol az L és H regiszterek használata megengedett. Pl.: LD A,XL; OR YH; ADC A,XH ; stb. A két indexregiszter egyazon utasításban nem használható. Ezért pl.: az LD XH,YL nemlétező utasítás. Lehetőség van még az SLL param. utasítás használatára is. Az SLL funkciója: a paraméter bitjet shifteli:



A paraméterek: A, B, C, D, E, H, L, (HL), (IX+d) és (IY+d) lehetnek. Pl.: SLL (HL); SLL C; stb.

AZ ASSEMBLER MÖKÖDÉSE

Az assembler két menetben fordít; az első menetben a

szimbólumtábla, a másodikban a kód készül el. A második menetre csak az első hibátlan lefutása esetén kerül sor. A fordítás I/O hiba, vagy a szimbólumtábla megtelése esetén azonnal megszakad.

A fordítás bármely gomb lenyomásával szüneteltethető, -folytatható, az ESC-pel bármikor kiléphetünk belőle.

A fordítás kezdetekor a már meglévő szimbólumtábla törlődik. A törlést a V 6 rendszerváltozó 1. és 2. bitjének 1-be állításával megakadályozhatjuk. A bitek beállításakori szimbólumtáblázat ezután már soha sem törlődik, csak a később definiált részek. A rendszer a 2. bit kinullázásával jelzi a szimbólumtábla alsó felének rögzítését. A teljes szimbólumtábla ismét törölhető a két bit 0-ázása után.

A fordítás végén az első szabad byte fizikai címe a V 8 változóba, logikai címe a $-ba kerül.

A A szövegszerkesztő tartalmának lefordítása.

A : A lefordított kód a memóriába (a kódpufferbe) kerül.

A 0 : Kód nem készül; szintaktikai ellenőrzésre használható.

A file-név:A második menetben a kódot pufferelt file-
ba tölti. File-ba fordításkor mindössze 64
byte hosszú kódpufferre van szükség!
A file első két byte-ja az ORG-címet tar-
talmazza, a többi byte maga a kód. Magnóin-
ításra "Magnófelvétel indul!" felirattal
figyelmeztet.File-ba fordítás előtt érdemes
ASM 0-val szintaktikai ellenőrzést végezni!

ASSEMBLER DIREKTÍVÁK:

A forrásszövegben található, az assembler müködését vezérlö parancsok:

MODE — A fordítás módjának megváltoztatása a fordítás közben:

- MODE 0 : "A 0" fordítási módot kapcsol be, azaz innentöl kezdve kód nem készül, csak szintaktikai ellenörzés történik.
- MODE C : Kód nem készül, a memóriában már bent lévö kódot összehasonlítja a forrásszöveg által elöírttal. Az eltérő sorokat kiírja.
- MODE N : Visszaállítja a MODE 0 vagy MODE C elötti fordítási állapotot.

```
241:
242:EQU             Számkonstans szimbólum definiálása, nem de-
243:---           finiálható újra.
244:címke EQU nn : a 'címke'-nevü, nn értékü konstans definiá-
245:              lása.Az nn kifejezésben címkenevet ne hasz-
246:              náljunk, fordítási fázishibához vezethet !
247:
248:SET           : Változó szimbólum definiálása.Az igy létre-
249:---            hozott szimbólum értéke  egy újabb SET uta-
250:címke SET nn   sitással átdefiniálható.
251:
252:END           : Befejezi a fordítást;  az esetleg hátralevö
253:---            forrásszöveget figyelmen kivül hagyja.Hasz-
254:              nálata nem kötelezö.
255:
256:INCLUDE file-név: A 'file-név' nevü file tartalmát betölti
257:---------------  a szövegpuffer még üres részébe  (ha van
258:              elég hely), majd lefordítja. File-vég után
259:              az igy betöltött file törlödik, a fordítás
260:              az INCLUDE után következö sorban folytató-
261:              dik. Az INCLUDE utasitások nem ágyazhatók
262:              egymásba! Magnóinditásra mindig figyelmez-
263:              tet.
264:                Az INCLUDE utasitás használatával olyan
```

```
265:              nagy programot is készithetünk,melynek for-
266:              rásszövege egy darabban nem férne be a szö-
267:              vegszerkesztőbe.
268:
269:LIST            Forditási listát készit a következő sortól.
270:----           Ha az A parancs kiadásakor müködökész nyom-
271:              tatót, vagy annak helyén más eszközt talál,
272:              oda -, ennek hiányában a képernyöre listáz.
273:              A lista oszlopai:
274:               - sorszám (dec.)
275:               - cim     (akt. számrendszer)
276:               - kód     (hexa.)
277:               - forrássor
278:     LIST    : Bekapcsolja a listázást.
279:     LIST nn : nn=0 esetében ki-, nn<>0 esetben bekapcsol-
280:               ja a listázást.
281:
282:IF  nn        : Feltételes forditási blokk  kezdete.  nn<>0
283:------        esetén fordit tovább.
284:
285:IFE nn        : Feltételes forditási  blokk  kezdete.  nn=0
286:------        esetén fordit tovább.
287:
288:ELSE          : Forditási mód cseréje.(fordit<->nem fordit)
```

```
289:----
290:
291:ENDIF        : Feltételes  fordítási blokk  vége. Innentől
292:-----          ismét minden lefordítódik.
293:                 FIGYELEM! A nem fordított  blokkokban  is
294:               történik szintaktikai ellenörzés !
295:
296:
297:             Áthelyezhető programok készítése
298:             ================================
299:
300:   Áthelyezhető (relokálható) az a program, amely a memó-
301:ria bármely  részére betöltve  müködőképes.  Mivel  a  TVC
302:operációs rendszere ilyen programok használatát nem támo-
303:gatja, és a pufferelетlen kazettás  file-kezelés  hatékony
304:relokáló loader használatát lehetetlenné  teszi,  ezért  a
305:relokálható programok hossza mindig nagyobb lesz,  mert  a
306:relokáláshoz szükséges adatokat is tartalmazniuk kell.
307:
308:   A fordítót a forrásszöveg elején  (az  ORG nn  helyett)
309:elhelyezett "címke RELOC" paranccsal utasíthatjuk relokál-
310:               ----------
311:ható program készítésére. A címke megadása kötelező!
312:
```

Az áthelyezhető programok írásának szabályai:

- Áthelyezhető programot nem lehet közvetlenül file-ba fordítani!
- relokálni csak a
  JP címke;
  JP feltétel,címke;
  CALL címke;
  CALL feltétel,címke;
  LD A,(címke);
  LD (címke),A;
  LD dd,címke;
  LD dd,(címke) és
  LD (címke),dd

  utasításokat lehet! A címke igen lényeges feltétel, ugyanis ha az előző utasítások számkonstansokat, EQU-val, vagy SET-tel definiált szimbólumokat tartalmaznak, akkor azok nem relokálódnak (fix címre mutatnak)!
- A "DW címke1,címke2,... címkex" utasítások nem relokálhatók, ezért az ilyen, ún. ugrótáblákból felvett értékekhez, amennyiben azok a programon belüli címeket tartalmaznak, mindig hozzá kell adni a program kezdőcímét! A relokálás után a program kezdőcíme a Start+2 címen lerakva megtalálható, ahol a "Start" a RELOC pa-

rancs címkéje.
- Az előző pontból következik, hogy egy ugrótábla nem tartalmazhat egyszerre programon belüli és kivüli címeket!
- Címkékhez hozzá lehet adni, illetve címkéből ki lehet vonni konstans értéket. Címkével más müveletet végezni tilos! (Két, ALFA és BÉTA címke távolságát kiszámitani a "táv SET BÉTA-ALFA" paranccsal lehet, a "táv" szimbólum értéke már szabadon felhasználható.)

A relokálható program látszólag a 0000H címre fordítódik (ORG 0000H), ám szerkezete jelentősen különbözik a hagyományosétól. A program egy 42 (dec.) byte-os relokáló rutinnal kezdődik, majd a "valódi" program következik. A már felsorolt utasitások után 2-2 byte adat kerül a kódba. A program igy, relokálás nélkül nem futtatható, és nem is fordítható vissza helyesen!

A "CODE" címre betöltött program BASIC-ből az

```
USR(CODE,CODE)
```

gépi kódból pedig az

```
LD   HL,CODE
CALL CODE
```

parancsokkal relokálható. A relokáló rutin ekkor négy dolgot végez el:

- a programban található adatok alapján elvégzi a címkehivatkozások átírását a "CODE" címhez képest,
- az adatbyte-okat NOP utasításokra cseréli,
- a CODE cím értékét elhelyezi a (CODE+2) címre, hogy a program ugrótáblát kezelő rutinjainak legyen honnan felvenniük az eltolási értéket,
- a relokáló szubrutint "kikapcsolja" a programból, így az soha többé nem futhat le. (A program így felszabaduló 5-42. byte-ja ezután bármilyen célra felhasználható.)

A relokálás végeztével a vezérlés visszakerül a relokáló rutin hívójához. Ezután a programkód már bármikor meghívható az USR(CODE), illetve a CALL CODE utasításokkal.

A rendszer bővíthetősége

A Mnemonic IV program a felhasználó részéről bővíthető, kiegészíthető.

A bővítések számára három belépési pont van. A belépési pontokhoz három 2-2 byte-os rendszerváltozó tartozik. A változóknak kell tartalmazni a felhasználó részéről megírt bővítőrutin kezdőcímét, hasonlóan a BASIC EXT utasításai-

nak ugrótáblájához. Alapesetben a kezdöcim 0 . Mindhárom bövitörutinnak RET utasitásal kell végződnie!

1.) A @ monitor utasitás, változója a COM_DEV
A felhasználó saját monitor utasitást, utasitásokat irhat, ezt "@ param1,param2" módon hivhatja meg. Az átadott paraméterstringek a STRING1 és a STRING2 cimekre kerülnek lerakásra a memóriába.
(Stringen egy, a memóriába lerakott feliratot értünk, első byte-ja a felirat további byte-jainak számát tartalmazza.)
A paraméterek megadása nem kötelező. A rutinból kilépéskor az A regiszter hibakódként értelmezett!

2.) A következő belépési pont a törespont lekezelő rutinban található. A bövités rutin megléte esetén a törespontnál csak akkor áll meg a tesztelendö program, ha a bövitésrutin NZ flagtartalommal tér vissza.
Igy lehetöség van 'intelligens' törespont készitésére Pl. egy adott memóriacim tartalmától, vagy a REG_TAB-ba elmentett regiszter tartalmaktól függö megállásra.

3.) Az utolsó belépési pont a REGVIEW rutinban található. Ez a rutin irja ki az elmentett regisztertartalmakat és

a PC cimen levö utasitást a képernyöre. Bövitésével elérhetö pl. a felhasználó részéröl fontosnak tartott memóriacimek tartalmának megjelenitése.

Az alábbiakban felsoroljuk a felhasználható rutinok listáját. Ahol nincs más megemlitve ott a paraméterek mindig fizikai cimet jelölnek, valamint a regiszterek visszatérö tartalma megváltozik.

Kiiró rutinok:

CRLF a cursort a következö sor elejére állitja.

PRINT a HL cimen kezdödö string kiirása, a 224 és nagyobb kódú karakterek helyett kód-223 db. space-t ir ki.

PRINTLN PRINT a sor elején.

PRINTAT PRINT a BC-vel megadott cursorpoziciótól.

INVERZ a PRINT inverz változata.

```
433:
434:SPACE   az A reg. számú space-t ir ki.
435:
436:FreeLin Ha a képernyöre irunk, a kiirt felirat sokszor nem
437:        felülirja, hanem maga alól "kitolja" a már ott lé-
438:        vö sorokat. A FreeLin rutin a  HL-ben adott  számú
439:        karakterhelyet felszabadit a képernyön, az  aktuá-
440:        lis sor kezdetétöl, a kiirás elött. Ajánlatos  16-
441:        tal osztható értéket adni!
442:
443:REGVIEW a regisztertükröt megjelenitö rutin.
444:
445:HEXBYTE a B reg. hexadecimális kiirása "H" nélkül.
446:
447:NUMERIC a HL értékét irja ki az aktuális számrendszer sze-
448:        rint.
449:
450:DECIMAL  a HL értékét irja ki decimálisan.
451:HEXANUM  a HL értékét irja ki hexadecimálisan, "H" nélkül.
452:
453:EQU_NUM  "string=nn" felirat kiirása; a stringre a HL  mu-
454:         tat, DE:=nn.
455:
456:
```

KONVERZIÓS RUTINOK:

FPC_RUT a logikai címből a címzési eltolás szerinti fizikai címet állítja elő: in/out=HL

LPC_RUT a fizikai címből a címzési eltolás szerinti logikai címet állítja elő: in/out=HL

FORM_16 a HL címen kezdődő 16 bites numerikus kifejezést tartalmazó string értékét a HL-be tölti, NZ flagtartalom esetén az A regiszter hibakóddal tér vissza.

FORM_8 ugyanez, csak 8 bites kifejezésre, az eredmény az L-be kerül.

LINE a HL címen kezdődő, C karakter hosszú 'n,n,"felirat",n,n,...' típusu szöveget a DE címre lerakott bináris stringgé alakitja. (ld. a DB utasitást!) Az esetleges hibakód az A-ba kerül.

ELLENÖRZÖ RUTINOK

```
481:
482:
483:CPHLDE   HL és DE összehasonlítása: 'CP HL,DE'. F az ered-
484:         ményt jelzi, HL és DE kezdőértéke megmarad.
485:
486:T_ADDR   input : HL=logikai cím;
487:         output: NZ=hibás cím; A=hibakód;  A=7: kifutás  a
488:         tárból, vagy A=11 : HL<$
489:
490:KIFUTÁS  input : HL=logikai cím;
491:         output: CY=kódpufferen  kívülre  címzés. (kifutás
492:         a tárból.) HL=változatlan.
493:
494:ST_END   output: CY=a stack mélysége kisebb mint 200 byte!
495:         Regisztert nem módosit!
496:
497:
498:                  KERESŐ RUTINOK
499:
500:
501:CÍMZŐ    egy HL címen kezdődő kétbyte-os adatokat  tartal-
502:         mazó táblázat A-adik elemét a HL-be  tölti  (első
503:         a 0. adat!). A DE az  adat 2.  byte-jának  címére
504:         mutat a visszatéréskor.
```

```
505:
506:FINDER0  egy n db. stringet tartalmazó táblázatban megke-
507:         resi a DE címen kezdödö stringet. A táblázat kez-
508:         detére a HL mutat. A táblázat végét egy 0 byte
509:         jelzi. Ha megtalálta, akkor A-ba a megtalált elem
510:         sorszáma-1; ha nem, akkor A-ba n kerül.
511:
512:
513:                    EGYÉB RUTINOK
514:                    ═════════════
515:
516:CLEAR    a HL címen kezdödö BC-db. byte feltöltése 0-val.
517:
518:CLEAR_A  a HL címen kezdödö BC-db. byte feltöltése az A
519:         regiszter tartalmával.
520:
521:IO_RUT   LOAD, SAVE, VERIFY az U2-U3+ lapon. A 3.lap hasz-
522:         nálata esetén a képernyön zavaros ábra jelenik
523:         meg. File nyitást/zárást a rutin meghivása elött/
524:         után kell végezni. [Csak a BOTTOM-ban létezö ru-
525:         tin!]
526:         Input : A=funkciókód
527:                 DE=terület kezdete
528:                 BC=terület nagysága
```

```
529:          Output: (HIBA)=hibakód.
530:
531:ASK        inverzben kiírja a HL címen kezdődő stringet,
532:          majd billentyű lenyomásra vár. "I","i","Y","y"
533:          lenyomása esetén NC; "N" vagy "n" esetén CY flag-
534:          tartalommal tér vissza.
535:
536:CURSOR     a cursor villogtatása billentyűlenyomásig. A ka-
537:          pott ASCII érték a C és az A regiszterekbe kerül.
538:          Nem írja ki a kapott karaktert!
539:
540:BEEP       kettős pittyegés, a HL es DE értékét megörzi.
541:
542:
543:                    A fontosabb változók
544:                    ====================
545:
546:Változó byte   leírás
547:------- ----   -------
548:
549:HIBA      1    Az IO_RUT rutin által visszaadott hibakód.
550:STRING1  57   I/O munkaterület.
551:STRING2  55   I/O munkaterület.
552:FPC       2   Az aktuális memóriamutató ($) fizikai címe.
```

```
553:LPC       2    Az aktuális memóriamutató ($) logikai címe.
554:EL        2    A címzési eltolás aktuális értéke.
555:               (EL = fizikai cím - logikai cím.)
556:INICTAB  8*2   A  V 0-V 7  rendszerváltozók alapértékei. A
557:               Mnemonic IV program betöltés  után, illetve
558:               EXT 0,22222-vel újraindításkor ebből a táb-
559:               lázatból veszi a kezdőértékeket.
560:RST_TAB  8*1   Az  RST 0  -  RST 56  utasítások hossza  az
561:               assembler, a  disassembler  és  a  debugger
562:               számára. Értékeik:
563:               80H : egybyte-os  (RST 0,  RST 16,  RST 32,
564:                     RST 40,  RST 56);
565:               81H : kétbyte-os (RST 8, RST 48)
566:               82H : többyte-os.  (RST 24)
567:REG_TAB 13*2   A debugger regisztertükör értékei; sorrend-
568:               ben:  PC, SP, BC, DE, HL, IX, IY, BC', DE',
569:               HL' ,AF, AF', I-IFF. (az IFF 2. bitje az EI
570:               állapotot jelöli).
571:COM_DEV   2    A @ utasítás kezdőcíme.
572:BRK_DEV   2    A töréspont kiértékelő rutin bővítéscíme.
573:REG_DEV   2    a REGVIEW rutin bővítéscíme.
574:
575:   Az itt leírt változók és  szubrutinok  címei a  BOT_SYM
576:[illetve a TOP_SYM] file-ban találhatók meg. A file  szim-
```

bólumtáblába töltése után rögzítsük a szimbólumtáblát!

A HIBAÜZENETEK MAGYARÁZATA

Hibakód	hibaüzenet / magyarázat

1: Kevés a memória!
A fordítás során megtelt a kódpuffer.

2: x/0!
Nullával osztás.

3: Ismeretlen parancs!
A Mnemonic IV. az utasítást nem ismeri.

4: Túlcsordulás!
Az aritmetikai kifejezés (vagy annak egy tagja) nem ábrázolható az adott szóhosszon. (8, illetve 16 biten).

5: Hibás kifejezés!
Az aritmetikai kifejezésben tiltott karakter található.

6: Hibás paraméterszám!
a) Az utasítás több, vagy kevesebb paramétert vár a megadottnál.

```
601:            b) A RELOC, EQU vagy SET utasitás cimke  nél-
602:               kül áll.
603:  7:  Kifutás a tárból!
604:            A kódpufferen kivülre hivatkozás.
605:  8:  Túl hosszú név!
606:            A cimke neve hosszabb hét karakternél!
607:  9:  Tele a tár!
608:            a) A szimbólumtábla megtelt.
609:            b) Az INCLUDE file nem fér be a szövegpuffer-
610:               be.
611:            c) Megtelt a töréspontt áblázat.
612: 10:  Elérhetetlen!
613:            A cim nem érhető el relativ cimzéssel!
614: 11:  Hibás paraméter!
615:            Az utasitás paramétere értelmetlen,  vagy ki-
616:            vül esik az értelmezési tartományon.
617: 13:  Hibás elnevezés!
618:            A cimkenévben tiltott karakter található!
619: 14:  Védett szó!
620:            A cimkenév regiszternévvel vagy  feltételszó-
621:            val azonos!
622: 15:  Nem használható!
623:            a) Cimzési eltolás van érvényben, a  debugger
624:               nem használható.
```

```
625:            b) Az editorba vagy  a  szimbólumtáblába  nem
626:               odaillő file-t próbáltunk betölteni.
627:            c) Az utasitás monitor parancs, az assembler-
628:               ben nem használható.
629: 16:  Újradefiniálás!
630:            A cimke már létezik!
631: 18:  Ismeretlen cim!
632:            Definiálatlan cimkére hivatkozás.
633: 20:  ???!
634:            Szintaktikai hiba, az utasitás nem választh-
635:            tó szét a paramétereitől, vagy csak cimke ta-
636:            lálható a sorban.
637: 21:  Nem cimkézhető!
638:            Assembler direktriva ( ORG,LIST,END,...) cim-
639:            kézése.
640:128-255: I/O hiba!
641:            Az operációs rendszer hibajelzései.
642:
643:       FÁZISHIBA !!!
644:               A két fordítási menet különböző hosszúságú
645:            kódot állitott elő. Ennek az az oka, hogy  az
646:            első menet során  valamelyik  'FILL nn',  'DS
647:            nn'  vagy 'cimke EQU nn'  utasitás nn kifeje-
648:            zésében még definiálatlan cimkére történt hi-
```

nn' vagy 'cimke EQU nn' utasítás nn kifejezésében még definiálatlan címkére történt hivatkozás.

12, 17, 19, 22-127: -
Nem használt hibakódok.

KÖSZÖNETNYILVÁNITÁS

Itt szeretném megköszönni Filó Gusztáv és Gyányi Sándor barátaim önzetlen segítségét, amit ezen leírás begépelésével, illetve az aritmetikai kifejezéseket kiértékelő rutin megírásával tettek nekem!

Mindenkinek jó munkát kívánok:

Bp. 1990. VI. 28.

Bata László
(1137 Bp. Jászai Mari tér 5. I/4.)