0

PRIMO ASSEMBLER 1.85.2

Használati útmutató

Kézzitette: Guthrod András
tanár
Dobó Katalin Gim.
Esztergom, 1985.V.

Az Assembler nemcsak assembler nyelvü programok forditására és szintaktikai ellenőrzésére, hanem az ezzel kapcsolatos járulékos feladatok ellátására is készült. Gondoskodik a forrásprogram szövegének előállitásáról, szerkesztéséről, annak kazettára való mentéséről illetőleg onnan történő visszamentéséről.
Az Assembler tartalmaz programbelövést elősegitő részeket, valamint egy önállóan használható monitor programot is.
A program előnyös tulajdonsága, hogy kis mérete folytán / 4.6k / valamennyi Primo változaton futtatható. Lehetőség van a BASIC interpreter és az Assembler egyidejü használatára is, azonban -a tár kisebb mérete folytán- ez a PRIMO A-32-es tipusra nem javallott.

A program betöltés után automatikusan indul. Ujrainditása a RESET gomb segitségével történhet.
Az Assembler a következő fő részekből áll:

- assembler forrásprogram szerkesztő
- assembler forrásprogram fordító
- programbelövő
- monitor
- perifériakezelő

A program nagybetüs parancsai egy betüből állnak, amit a prompt jel után kell szóköz kihagyása nélkül megadni. A parancsot és a paramétereket mindig a RETURN billentyüvel kell lezárni.
A Használati útmutatóban található memóriacimek és egyéb konstansok 16-os /hexadecimális/ számrendszerben értendők!
Az Assembler programba a felhasználói SP inicializálásával a 42DC cimen, inicializálás nélkül a 443Ø cimen lehet belépni.

1. A forrásprogram

A forrásfile sorai egyetlen assembler utasitást tartalmazhatnak. A sorokat az Assembler automatikusan sorszámozza, de csakis a szerkesztés megkönnyitése végett. Igy nem lehet ugró utasitas / JP / használatánál a sorszámmal megjelölni a célt /mint a BASIC-ben/!
A sor általában 4 mezőből áll:

CÍMKE: OPERÁTOR OPERANDUS;MEGJEGYZÉS

A mezők közötti elhatárolójelek -balról jobbra- a következők: ":" , SPACE , ";" .
Az első és/vagy a negyedik mező, vagy az első három mező elhagyható. A különböző esetekre példákat találhatunk a mellékelt mintaprogramokban.
Az operátorok és az operandusok /ún. mnemonikok/ a következő kiadványból választhatók ki:
Ipari Informatikai Központ által kiadott
Z 80-as sorozat VIII.rész: CPU utasitáskészlet
Két eltéréssel: 164. old. RLD utasitás helyett RLD (HL) ,
172. old. RRD utasitás helyett RRD (HL) .
Az operandusok lehetnek szimbólumok és konstansok. A szimbólumokkal és konstansokkal a négy alapmüveletet tartalmazó kifejezések alkothatók, amelyek az egész tipusú aritmetika szabályai alapján balról-jobbra értékelődnek ki a fordítás során.

```
LD A,2+3*3
LD B,3*3+2
```

egyenértékű, a LD A,ØF
LD B,ØB utasításokkal.

A forrásprogramban szereplő konstansokat háromfajta számrendszerben lehet megadni: hexadecimális, decimális és oktális. Az alapértelmezés hexadecimális. A decimális számok végén T-vel jelzünk, míg az oktális számok végén O-val kell jelezni!

Példával: 17T , 23T , 16ØØØT , 76O , 165O , 1F , 1B
decimális oktális hexadecimális

A forrásprogramban /de csakis ott!/ a 9-nél nagyobb hexadecimális számjeggyel kezdődő konstans elöl egy Ø számjegy nem maradhat el, mert ellenkező esetben "Nem def." hibaüzenetet kapunk.

Helyesen megadott hexadecimális konstansok:

18 , 1F , ØF , ØE8ØØ , 9654 , ØABCD

A forrásprogram aktuális /éppen szerkesztett vagy listázott/ sorára belső mutató mutat. Hasonlóan, mint a PRIMO BASIC-ben az "EDIT." , vagy a "LIST." parancsok.

A forrásprogram első tényleges utasítását egy ORG utasításnak kell megelőzni, a program végét pedig END utasítással kell lezárni. Ez az END azonban szerepét tekintve lényegesen különbözik a BASIC END utasítástól /lsd. később/!

2. Az Assembler parancsai

A parancsok egyetlen nagybetűből állnak.

A belső mutatót mozgató parancsok:

↑n : a mutató n sorral feljebb megy

↓n : a mutató n sorral lejjebb megy

T : a mutató a forrásprogram elejére áll

B : a mutató a forrásprogram végére áll

Pn : a mutatótól kezdve n sort listáz
Például a PF parancs /F=15T !/ tizenöt sort listáz a képernyőre.

Ixxxxx : a mutatótól kezdve hátrafelé megkeresi az xxxxx karakterlánc első előfordulási helyét, majd a mutatót erre a sorra állitja.

A szerkesztő parancsai:

E : Beszúrás a mutató elé. A beszúrás automatikus sorszámozással addig folytatódik, mig csak "." pont karaktert tartalmazó sort nem gépelünk.
Ezzel a paranccsal kell kezdeni a program irását is.

Zn : A mutatótól kezdve n sort töröl.

R : A mutató sorát egy új sorra cseréli.
A soron belüli utólagos javitásra nincs lehetőség!

Az assembler fordító parancsai:

A : A starttól az END utasitásig terjedő forrásprogramot leforditja. "Opció?" kérdésre
- üres válasz esetén nem készit forditási listát, de ez a leggyorsabb forditási mód.

- V-vel válaszolva a képernyőre készit forditási listát. Jó még a B,C,F,G,J,K,M,O,R,S,V,W,Ü betük bármelyike is.
- P-vel válaszolva a printerre készit forditási listát. Jó még az A,D,E,H,I,L,M,P,Q,T,U,Ö,É,Á betük bármelyike is.
  Ha nincs a géphez printer kapcsolva, akkor a gép hangjelzés közben végtelen ciklusba kerül, amiből a RESET gomb segitségével léphetünk ki.

R ? : Az előző forditáskor létrehozott szimbólumtáblát rendezi ABC szerint, majd kiirja az opcióban megadott eszközre.
A parancs után egy betüt irva csak a megadott betüvel kezdődő szimbólumokat irja ki.

K : Törli a forrásprogramot a memóriából. Itt jegyezzük meg, hogy a RESET nem törli a forrásprogramot.

H : Kiirja a forrásprogram start és end cimeit.
A forrásprogram területét a következő eljárással helyezhetjük át /amennyiben az 5800 start cim nem felel meg/.
A monitor segitségével a 43E4 cimtől kezdve forditott sorrendben irjuk be az új start cimet, majd hajtsunk végre egy K parancsot. Ellenőrizzük a müvelet helyességét a H parancs segitségével!

Programbelövő parancsok:

G : Vezérlésátadás a felhasználói programnak.
Start kérdésre az inditócimet, a Tpont kérdésre a töréspont cimét kell megadni.
Töréspont megadása esetén, ha a program a futás során eljut a Tpont-nál megadott cimre, akkor az

Assembler visszaveszi a vezérlést, elmenti a regiszterek tartalmait, majd egy X parancsot hajt végre.
Ismételt G parancs esetén a regiszterek kezdőértékeit innen olvassa be.

X : A regisztertartalmak megjelenitése a képernyőn. Vezérlésátadás esetén a felhasználói program az itt látható értékeket kapja a regiszterek kezdőértékeként. Az értékek csak töréspont végrehajtása esetén változnak meg.

A monitor parancsai:

Dnnnn : nnnn címtől kezdve 14-szer 8 byte-ot kiir a következő formában:

cim hexadecimális értékek ASCII

Az üres D parancs az előző dumpolást folytatja.

Annnn : Az nnnn cimü memóriarekesz tartalmát módosithatjuk. Üres válasz esetén továbblépés a következő memóriacimre. Kilépés csak "." pontot tartalmazó válasszal történhet.

C : Startcimtől a stopcimig terjedő memóriablokk áthelyezése új cimre.

F : Startcimtől a stopcimig feltölti a memóriát egy adat konstanssal.

3. A fordítóprogramnak szóló speciális operátorok

ORG nnnn : A forrásprogram elején elhelyezve a leforditott utasitások logikai kezdőhelyét határozza meg. A programban többször is alkalmazható.

END : A forrásprogram utolsó utasitásának ezt kell megadni, egyébként hibaüzenetet kapunk. A fordító az END utasitásig forditja a forrásprogramot.

LOAD nnnn : A leforditott tárgyprogram /object program/ fizikai kezdőcimét határozza meg és felszólit a a memóriába való beirásra. A load-olás a következő ORG-ig vagy az END-ig tart.
A LOAD-ban megadott cim nem feltétlen azonos az ORG-ban megadott cimmel!

DB : A DB után megadott több operandus értékét a tárgyprogramba forditja.

EQU : Egy szimbólumhoz értéket rendel.
Például:
```
STRING:EQU 7211
  TEXT:EQU STRING+12T
  HELY:EQU $
```
A $ jellel a forditáskor aktuális memóriarekesz cimére hivatkozhatunk. Alkalmazására a mellékelt mintaprogramokban találhatunk példát.

DW : A DW után megadott szimbólum értékét a tárgyprogramba forditja, a byte-okat forditott sorrendben, ahogy ez a Z-80 processzornak szükséges.

DS : A DS után megadott számú byte-ot üresen hagy a forcitáskor.

4. Hibaüzenetek

"Értelmetlen" : szintaktikai hiba

"End" : hiányzó END

"Foglalt" : A szimbólum foglalt szó.

"Org?" ; Hiányzó ORG utasitás.

"Opnd" : A mnemonikot helytelenül használtuk.

"Dupla symb." : Többször deklarált szimbólum.

"Nem def." : Nem definiált szimbólum vagy helytelenül megadott hexadecimális számot talált, amit megpróbált szimbólumnak értelmezni.

"Megtelt" : Megtelt a szimbólumtábla. A forrásprogramot magnóra kell rögziteni, majd a H parancsnál leirtak szerint hátrább kell helyezni a forrásprogramok területét és utána visszatölthető az eredeti forrásprogram.

5. Magnókezelés

A forrás és a tárgyprogramok magnóval rögzithetők és ellenőrizhetők.
A rögzitéshez megadott file név 12 karakterből állhat.
Az utána következő 4 karaktert az Assembler foglalja le file kiterjesztés rögzitése végett. Forrásprogram esetén a kiterjesztés ".TXT" , tárgyprogram esetén ".ASM".
Betöltéskor a név azonosságán kivül a kiterjesztés helyességét is figyeli az Assembler. Eltérés esetén "Tipushiba" üzenetet ad és folytatja a keresést.

Parancsok:

L	: Forrásfile olvasása magnóról. Név hiányában a következő ".TXT" tipusú file-t olvassa. Ha már van tárolva forrásprogram, akkor a meglévő után tölt /merge/.
LO	: Objectfile olvasása. A továbbiak megegyeznek a fentiekkel, csak itt cimek szerint tölt.
S	: Forrásfile irása kazettára.
SO	: Objectfile irása kazettára. Meg kell adni a gépi kódú program start és stop cimeit, majd a file nevet.
V	: File-ok ellenőrzése. Ugyanaz mint "L" illetve
VO	"LO",csak nem tölt a memóriába.

6. A BASIC interpreter és az Assembler egyidejü használata

Foglaljuk le - az első program segitségével - a tár alsó részét az Assembler-nek, a forrásprogramoknak és a gépi programoknak. A memória felső /magasabb cimü/ részét pedig átadjuk a BASIC interpreternek.

1.pr.

```
        ORG  6000
        LOAD 6000
ASMTOP: EQU  .....   ; ide irjuk az Assembler terü-
        LD IX,(1B)   ; letének a végcimét.
        LD HL,ASMTOP+2
        LD (40A4),HL
        DEC HL
        LD (HL),0
        CALL 1B4D
EXIT:   NOP
        END
```

Miután az első programot egyszer lefuttattuk -az EXIT cimére töréspontot helyezve- , máris inicializáltuk a BASIC-et. Ezt a programot azonban csak egyszer kell futtatni, mert minden egyes végrehajtásakor törli a BASIC programokat! Ezután a BASIC-be az 1A19 cimen léphetünk be a G parancs segitségével. Fontos azonban, hogy az IX indexregiszter 4042-öt tartalmazzon az átlépéskor, ezért erről az X parancs segitségével meg kell győződni! Ha ez a feltétel nem teljesül, akkor alkalmazzuk a 2. programot!

2.pr.

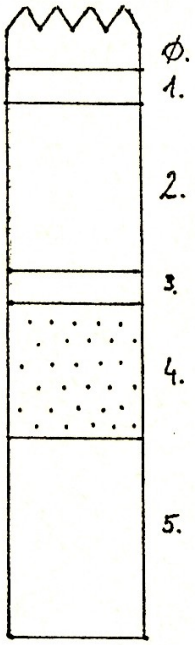
```
        LD IX,4042
        JP 1A19
```

A BASIC-ből a RESET gomb rövid lenyomásával léphetünk vissza az Assemblerbe.
Fontos, hogy a rendszerek közötti átlépések a rendszerek olyan melegindítását eredményezi, amely nem törli sem a BASIC programot, sem az assembler programot! A BASIC-be való átlépés után, azt teljes értékűként használhatjuk.

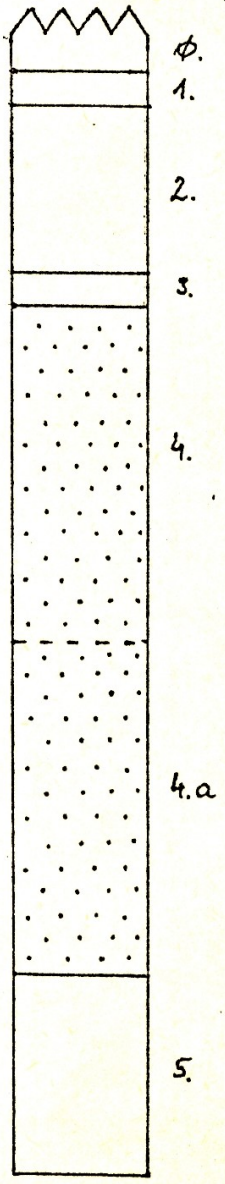
Ha valamilyen programhiba következtében megjelenik a BASIC hidegindítására utaló felirat /PRIMO BASIC SYSTEM..../, akkor csak az Assembler újratöltése után férhetünk ismét a memóriához. Ezt a kellemetlenséget elkerülendő, tanácsos a már meglévő programokat időnként mágnószalagra rögzíteni!

7. Javasolt memória felosztások

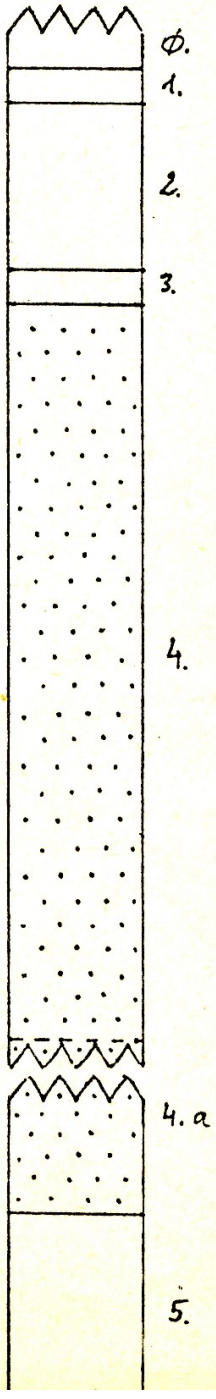
MILO A-32



A-48



A-64



A pirossal behatárolt területeken csak útmutatók alapján változtassunk!
Az egyes jelek jelentése:

	A-32	A-48	A-64
0. ROM	0000-3FFF 16 k	0000-3FFF 16k	0000-3FFF 16k
1. BASIC rendszer-változók	4000-42E8 0.75k	4000-42E8 0.75k	4000-42E8 0.75k
2. Assembler	42EC-5526 4.6k	42EC-5526 4.6k	42EC-5526 4.6k
3. Szimbólum-tábla	5527-57FF 0.75k	5527-57FF 0.75k	5527-57FF 0.75k
4. Szabad terület forrás pr. stb	5800-67FF 4k	5800-A7FF 20k	5800-E7FF 36k
4.a BASIC használat esetén:	--------- --------- --------- ---------	5800-7FFF 10k 8000-A7FF 10k	5800-AFFF szabad 22k B000-E7FF BASIC 14k
5. Video memória	6800-7FFF 6k	A800-BFFF 6k	E800-FFFF 6k

A 4-es pontban felsorolt értékek az útmutatóban leirtak szerint változtathatók, az itt közöltek csak javasolt értékek! Az Assembler a felhasználó stack pointerét automatikusan a szabad terület végére állitja. A rendszer stack-je az Assembler területen található!

6. Mintaprogramok

A mellékelt mintaprogramokban példákat találhatunk arra, miként kell visszatérni az Assemblerbe. Ahol nem találunk töréspontra való utalást, ott egy JP utasitással tér vissza a program.
A mellékelt programokban csak a jobb áttekinthetőség kedvéért kerültek a cimkék kiemelésre, begépeléskor nem szükséges a cimkemező kihagyása!

```
          ORG 6000
          LOAD 6000

          LD IX,(1B)          ; DCB cime
          CALL 1C9            ; Képernyőtörlés
          LD HL,STRING
          LD B,HOSSZ
UJRA:     LD A,(HL)
          INC HL
          CALL 15
          DJNZ UJRA
EXIT:     NOP
STRING:   DB "EZ EGY SZOVEG"
HOSSZ:    EQU $-STRING        ; Hossz számitása
          END
```

A program képernyőtörlés után a STRING cimtől elhelyezkedő szöveget irja ki. A programot a forditás után 6000-tól indithatjuk és EXIT értékére töréspontot elhelyezve futtathatjuk. Az EXIT értékét az R parancs segitségével kereshetjük ki.

HANGKÉPZÉS

```
ORG 6000
LOAD 6000

        LD L,5          ; Ismétlések száma
HANG    LD H,1FØF       ; Hangmagasság végértéke
        LD DE,0         ; Hangmagasság kezdőértéke
HANG1:  LD BC,1         ; Hang hossza
        CALL 3F68       ; BEEP rutin
        INC E           ; Hangmagasság csökkentése
        LD H,E
        CP H            ; Hangmagasság ellenőrzése
        JR NZ,HANG1
        DEC L           ; Ismétlésszámláló csökkentése
        JR NZ,HANG      ; Megvolt az 5 db ismétlés?
        ; Ujabb hangfajta
        LD L,5
HANG2:  LD DE,ØFF
HANG3:  LD BC,1
        CALL 3F68
        DEC E           ; Hangmagasság növelése
        JR NZ,HANG3     ; Ø a hangmagasság?
        DEC L
        JR NZ,HANG2

EXIT:   NOP             ; Töréspont helye
        END
```

6000 2E05
2600
1100FF
0101FF
CD683F
1C
F3
BC
20F5
2D
20ED
2E05
11FFFF
0101FF
CD683F
1D
20F7
2D
20F1
6026 FF

```
        ;%%%%%%%%%%%%%%%%%%%%%%
        ;% SCROLL JOBBRA %
        ;%%%%%%%%%%%%%%%%%%%%%%

        ORG 6ØØØ
        LOAD 6ØØØ

SOR:    EQU 3                  ;Eltolandó sorok száma

        LD IX,4Ø42             ;DCB címe
        LD B,Ø                 ;ELtolások száma (256!)
SCR1:   PUSH BC                ;Külső ciklus kezdete
        LD L,Ø
        LD H,(IX+8)            ;HL-be video kezdőcíme
        LD B,12T*SOR           ;Grafikus sorok száma
SCR2:   PUSH BC                ;Belső ciklus kezdete
        OR A                   ;Carry nullázása
        LD B,2Ø
SCR3:   RR (HL)                ;Legbelső ciklus kezdete
        INC HL
        DJNZ SCR3              ;Legbelső ciklus vége
        POP BC
        DJNZ SCR2              ;Belső ciklus vége
        POP BC
        DJNZ SCR1              ;Külső ciklus vége

EXIT:   NOP
        END
```

```
;%%%%%%%%%%%%%%%%%%%%%
;%     GEPTIPUS      %
;%  MEGALLAPITASA    %
;%%%%%%%%%%%%%%%%%%%%%

         ORG 6000
         LOAD 6000

         LD IX,4042          ;DCB cime
         CALL 1C9            ;CLS
         LD HL,STRING        ;STRING cimtől az első
         CALL 28A7           ; 0 byte-ig a videora
         LD A,(IX+8)
         CP 68               ; 6800?
         JR Z,A32
         CP 0A8              ; A800?
         JR Z,A48
         CP 0E8              ; E800?
         JR Z,A64
         LD HL,ERROR         ;Egyik érték sem fordult elő!
         JR VEGE
A32:     LD HL,STRING1
         JR VEGE
A48:     LD HL,STRING2
         JR VEGE
A64:     LD HL,STRING3
VEGE:    CALL 28A7
         LD HL,STRING4       ; SPACE,CHR$(1) , RETURN,..
         CALL 28A7
         CALL 25             ; Billentyülenyomásra vár
         JP 42EC             ; Assembler melegindítása

ERROR:   DB "PROGRAMHIBA!",0  ; Stringterület
STRING:  DB "AZ ON GEPE: "
         DB 2,4," PRIMO ",0
STRING1:DB "A-32",0
STRING2:DB "A-48",0
STRING3:DB "A-64",0
STRING4:DB 20,1,0D
         DB "TIPUSU",0

         END
```

```
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
;% BASIC KULCSSZAVAK %
;%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        CRG 6ØØØ
        LOAD 6ØØØ

MEZŐ:   EQU 7           ;Output mező szélessége
START:  EQU 165Ø        ;Kulcsszótábla kezdőcime
STOP:   EQU 1822        ;Kulcsszótábla végcime

        LD IX,4Ø42      ;DCB cime
        CALL 1C9        ;CLS
        LD HL,START     ;Mutató kezdőértéke
        LD DE,STOP      ;Mutató végértéke
        LD B,Ø          ;Ø-ik mező szélessége
RUN:    LD A,(HL)       ;Következő karakter beolvasása
        AND 7F          ;7.bit nullázása
        LD C,A          ;Karakter elmentése
        CP (HL)         ;Ø volt a 7.bit?
        INC HL          ;Mutató a következő karakterre
        JR Z,CONT       ;CONT, ha nincs vége a szónak
        CALL TABULA     ;Uj szó előtt tabulálás
        CALL PAUSE      ;Billentyüzet figyelése
        RST 18          ; HL-DE összehasonlitása
        JR NC,VÉGE      ; HL≥DE,akkor vége
CONT:   LD A,C          ;Kiirandó karakter visszatöltése
        CALL 15         ;Karakter a képernyőre      Display rutin
        DEC B           ;A mezőből elhasználtunk egy helyet
        JR RUN          ;Vissza a ciklus elejére
VÉGE:   CALL 25         ;Billentyülenyomásra vár    Input rutin
        JP 42EC         ;Assembler melegindítása

PAUSE:  CALL 1D         ;Ha van lenyomott billentyü, akkor   Inkey $
        JR NZ,PAUSE     ;addig vár, mig el nem engedik       rutin
        RET

TABULA: LD A,B          ;Ha B=Ø (a teljes mező be van tölt-
        OR A            ;ve) , akkor nem csinál semmit,
        JR Z,TABULA1    ;egyébként a maradékot szóközzel
        LD A,2Ø         ;tölti fel.
TABULA2: CALL 15                                    Display rutin
        DJNZ TABULA2
TABULA1: LD B,MEZŐ      ;B kezdőértéke
        RET
        END
```

6000
6047