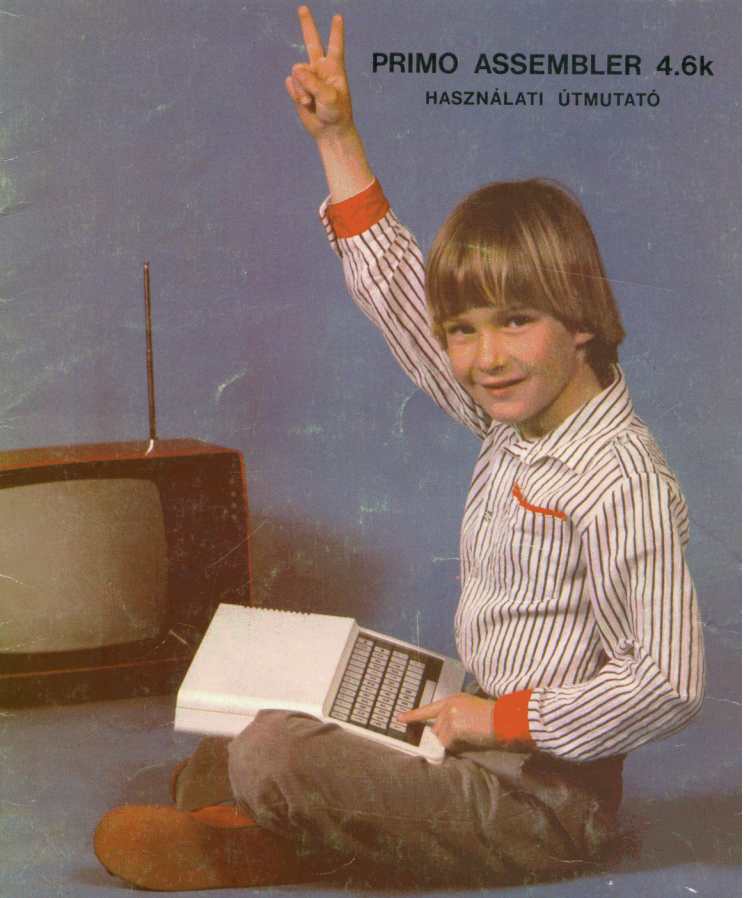
PRIMO FÜZETEK

PRIMO ASSEMBLER 4.6k

HASZNÁLATI ÚTMUTATÓ



COSY

EMO

PRIMO ASSEMBLER 4.6k

HASZNÁLATI ÚTMUTATÓ

MTA–SZTAKI COSY MŰSZAKI FEJLESZTŐ LEÁNYVÁLLALAT
BUDAPEST 1985

Készítette
GUTBROD ANDRÁS
tanár

Dobó Katalin Gimnázium
ESZTERGOM, 1985

Kiadja
az MTA–SZTAKI COSY Műszaki Fejlesztő Leányvállalat

Felelős kiadó
MÓRICZ SÁNDOR
igazgató

Megjelent a GRAFO KIADÓI IRODA gondozásában
1000 példányban, 2,5 (A/5) ív terjedelemben
265/85

VTV REPROTECHNIKA, BUDAPEST

Az Assembler nemcsak assembler nyelvű programok fordítására és szintaktikai ellenőrzésére, hanem az ezzel kapcsolatos járulékos feladatok ellátására is készült.
Gondoskodik a forrásprogram szövegének előállításáról, szerkesztéséről, annak kazettára való mentéséről, illetőleg onnan történő visszamentéséről.
Az Assembler tartalmaz programbelövést elősegítő 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 egyidejű használatára is, azonban – a tár kisebb mérete folytán – ez a PRIMO A–32-es típusra nem javallott.

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

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

A program nagybetűs parancsai egy betűből állnak, amit a prompt jel után kell szóköz kihagyása nélkül megadni. A parancsot és a paramétereket mindig a RETURN billentyűvel kell lezárni.
A Használati útmutatóban található memóriacímek és egyéb konstansok 16-os (hexadecimális) számrendszerben értendők!
Az Assembler programba a felhasználói SP inicializálásával a 42EC címen, inicializálás nélkül a 443ϕ címeken lehet belépni.

1. A FORRÁSPROGRAM

A forrásfile sorai egyetlen assembler utasítást tartalmazhatnak. A sorokat az Assembler automatikusan sorszámozza, de csakis a szerkesztés megkönnyítése végett. Így nem lehet ugró utasítás (JP) használatánál a sorszámmal megjelölni a célt (mint a BASIC-ben)!
A sor általában 4 mezőből áll:

CIMKE: 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 utasításkészlet
Két eltéréssel: 164. old. RLD utasítás helyett RLD (HL) ,
172. old. RRD utasítás helyett RRD (HL) .
Az operandusok lehetnek szimbólumok és konstansok. A szimbólumokkal és konstansokkal a négy alapműveletet tartalmazó kifejezések alkothatók, amelyek az egész típusú aritmetika szabályai alapján balról-jobbra értékelődnek ki a fordítás során.

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

Egyenértékű, az

```
LD A,ϕF
LD B,ϕB
```

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!

Például: 1ϕT , 23T , 16ϕϕϕT , 760 , 1850 , 1F , 18 ,
decimális oktális hexadecimális

A forrásprogramban (de csakis ott!) a 9-nél nagyobb hexadecimális számjeggyel kezdődő konstans elöl egy ϕ számjegy nem maradhat el, mert ellenkező esetben "Nem def." hibaüzenetet kapunk.
Helyesen megadott hexadecimális konstansok:

18 , 1F , ϕF , ϕE8ϕϕ , 9654 , ϕABCD

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 (lásd. 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 állítja.

A szerkesztő parancsai:

E : Beszúrás a mutató elé. A beszúrás automatikus sorszámozással addig folytatódik, míg csak "." pont karaktert tartalmazó sort nem gépelünk.
Ezzel a paranccsal kell kezdeni a program írását is.
Zn : A mutatótól kezdve n sort töröl.
N : A mutató sorát egy új sorra cseréli.
A soron belüli utólagos javításra nincs lehetőség!

Az assembler fordító parancsai:

A : A starttól az END utasításig terjedő forrásprogramot lefordítja. "Opció?" kérdésre

- üres válasz esetén nem készít fordítási listát, de ez a leggyorsabb fordítási mód.
- V-vel válaszolva a képernyőre készít fordítási listát. Jó még a B, C, F, G, J, K, N, O, R, S, V, W, Ü betűk bármelyike is.
- P-vel válaszolva a printerre készít fordítási listát. Jó még az A, D, E, H, I, L, M, P, Q, T, U, Ö, É, Á betűk bármelyike is.

Ha nincs a géphez printer kapcsolva, akkor a gép hangjelzés közben végtelen ciklusba kerül, amiből a RESET gomb segítségével léphetünk ki.

R : Az előző fordításkor létrehozott szimbólumtáblát rendezi ABC szerint, majd kiírja az opcióban megadott eszközre.
A parancs után egy betűt írva csak a megadott betűvel kezdődő szimbólumokat írja ki.
K : Törli a forrásprogramot a memóriából. Itt jegyezzük meg, hogy a RESET nem törli a forrásprogramot.
H : Kiírja a forrásprogram start és end címeit.
A forrásprogram területét a következő eljárással helyezhetjük át (amennyiben az 58ϕϕ start cím nem felel meg).
A monitor segítségével a 43E4 címtől kezdve fordított sorrendben írjuk be az új start címet, majd hajtsunk végre egy K parancsot. Ellenőrizzük a művelet helyességét a H parancs segítségével!

Programbelövő parancsok:

G : Vezérlésátadás a felhasználói programnak.
Start kérdésre az indítócímet, a Tpont kérdésre a töréspont címét kell megadni.
Töréspont megadása esetén, ha a program a futás során eljut a Tpont-nál megadott címre, 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 megjelenítése 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 kiír a következő formában:

cím hexadecimális értékek ASCII

Az üres D parancs az előző dumpolást folytatja.

Mnnnn : Az nnnn című memóriarekesz tartalmát módosíthatjuk. Üres válasz esetén továbblépés a következő memóriacímre. Kilépés csak "." pontot tartalmazó válasszal történhet.

C : Startcímtől a stopcímig terjedő memóriablokk áthelyezése új címre.

F : Startcímtől a stopcímig feltölti a memóriát egy adat konstanssal.

3. A FORDÍTÓPROGRAMNAK SZÓLÓ SPECIÁLIS OPERÁTOROK

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

END : A forrásprogram utolsó utasításának ezt kell megadni, egyébként hibaüzenetet kapunk. A fordító az END utasításig fordítja a forrásprogramot.

LOAD nnnn : A lefordított tárgyprogram (object program) fizikai kezdőcímét határozza meg és felszólít a memóriába való beírásra. A load-olás a következő ORG-ig vagy az END-ig tart. A LOAD-ban megadott cím nem feltétlen azonos az ORG-ban megadott címmel!

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

EQU : Egy szimbólumhoz értéket rendel.

Például:

```
STRING : EQU 7211
  TEXT : EQU STRING+12T
  HELY : EQU $
```

A $ jellel a fordításkor aktuális memóriarekesz címére hivatkozhatunk. Alkalmazására a mellékelt mintaprogramokban találhatunk példát.

DW : A DW után megadott szimbólum értékét a tárgyprogramba fordítja, a byte-okat fordított sorrendben, ahogy ez a Z–8ϕ processzornak szükséges.

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

4. HIBAÜZENETEK

"Értelmetlen" : szintaktikai hiba
"End" : hiányzó END
"Foglalt" : A szimbólum foglalt szó.
"Org?" : Hiányzó ORG utasítás.
"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 rögzíteni, majd a H parancsnál leírtak szerint hátrább kell helyezni a forrásprogramok területét és utána visszatölthető az eredeti forrásprogram.

5. MAGNÓKEZELÉS

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

Parancsok:

L : Forrásfile olvasása magnóról. Név hiányában a következő ".TXT" típusú file-t olvassa. Ha már van tárolva forrásprogram, akkor a meglévő után tölt (merge).

LO : Objectfile olvasása. A továbbiak megegyeznek a fentiekkel, csak itt címek szerint tölt.

S : Forrásfile írása kazettára.

SO : Objectfile írása kazettára. Meg kell adni a gépi kódú program start és stop címeit, 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 EGYIDEJŰ HASZNÁLATA

Foglaljuk le – az első program segítségével – a tár alsó részét az Assembler-nek, a forrásprogramoknak és a gépi programoknak. A memória felső (magasabb című) részét pedig átadjuk a BASIC interpreternek.

1. pr.

```
          ORG   6ϕϕϕ
          LOAD  6ϕϕϕ
ASMTOP :  EQU   .....   ; ide írjuk az Assembler területének
          LD IX, (1B)   ; a végcímét.
          LD HL,ASMTOP+2
          LD (4ϕA4), HL
          DEC HL
          LD (HL),ϕ
          CALL 1B4D
EXIT   :  NOP
          END
```

Miután az első programot egyszer lefuttattuk – az EXIT címére 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 címen léphetünk be a G parancs segítségével. Fontos azonban, hogy az IX indexregiszter 4ϕ42-őt tartalmazzon az átlépéskor, ezért erről az X parancs segítségével meg kell győződni! Ha ez a feltétel nem teljesül, akkor alkalmazzuk a 2. programot!

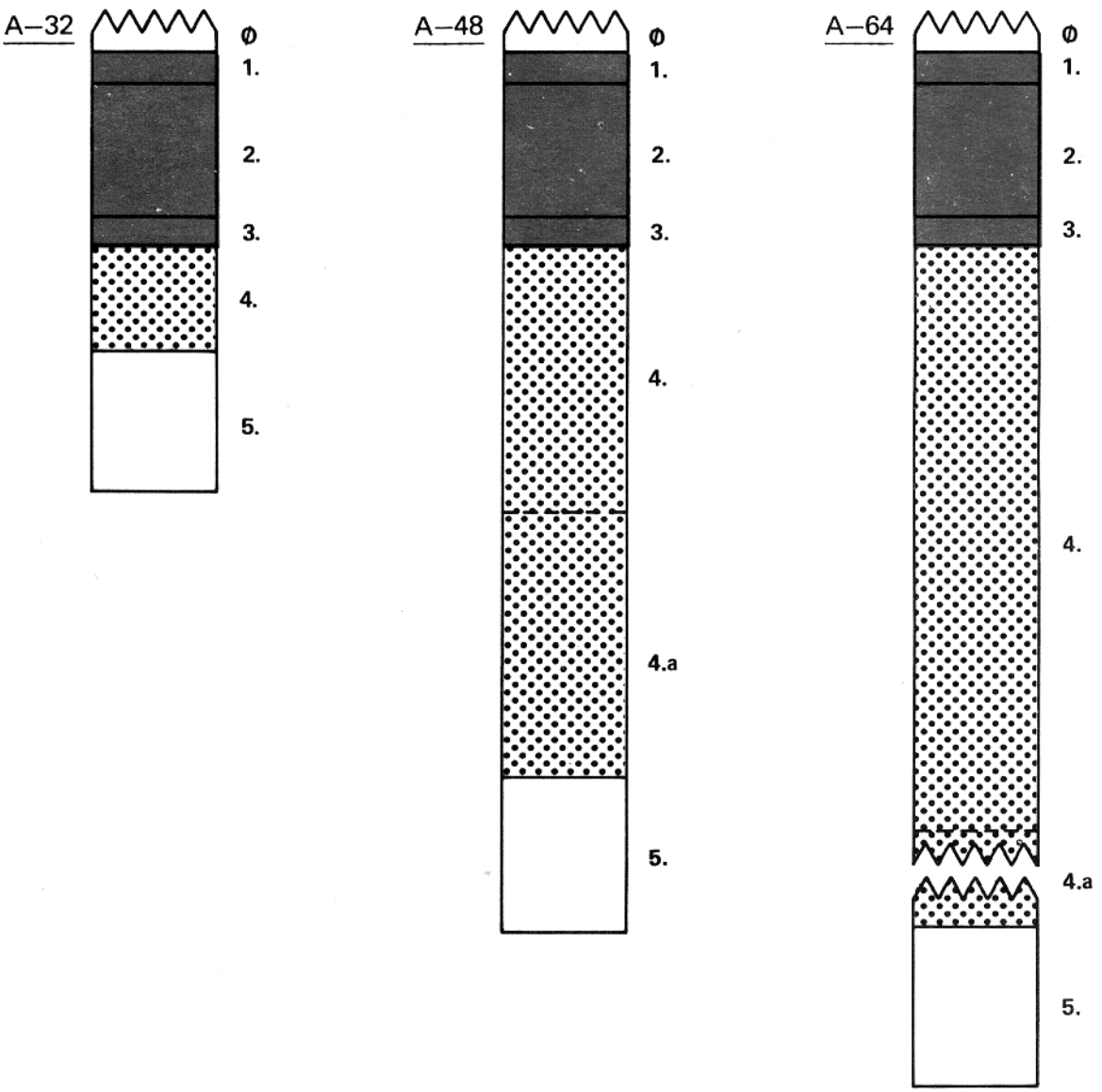
2. pr.

```
LD IX,4ϕ42
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 hidegindításra 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 magnószalagra rögzíteni!

7. JAVASOLT MEMÓRIA FELOSZTÁSOK



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

	A–32	A–48	A–64
ϕ. ROM	ϕϕϕϕ–3FFF 16k	ϕϕϕϕ–3FFF 16k	ϕϕϕϕ–3FFF 16k
1. BASIC rendszer-változók	4ϕϕϕ–42E8 ϕ.75k	4ϕϕϕ–42E8 ϕ.75k	4ϕϕϕ–42E8 ϕ.75k
2. Assembler	42EC–5526 4.6k	42EC–5526 4.6k	42EC–5526 4.6k
3. Szimbólum-tábla	5527–57FF ϕ.75k	5527–57FF ϕ.75k	5527–57FF ϕ.75k
4. Szabad terület forrás pr. stb.	58ϕϕ–67FF 4k	58ϕϕ–A7FF 2ϕk	58ϕϕ–E7FF 36k
4.a BASIC használat esetén:	—————— —————— —————— ——————	58ϕϕ–7FFF 1ϕk 8ϕϕϕ–A7FF 1ϕk	58ϕϕ–AFFF szabad 22k Bϕϕϕ–E7FF BASIC 14k
5. Video memória	68ϕϕ–7FFF 6k	A8ϕϕ–BFFF 6k	E8ϕϕ–FFFF 6k

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

8. MINTAPROGRAMOK

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

```
        ORG 6ϕϕϕ
        LOAD 6ϕϕϕ

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

A program képernyőtörlés után a STRING címtől elhelyezkedő szöveget írja ki. A programot a fordítás után 6ϕϕϕ-től indíthatjuk és EXIT értékére töréspontot elhelyezve futtathatjuk. Az EXIT értékét az R parancs segítségével kereshetjük ki.

HANGKÉPZÉS

```
        ORG 6ϕϕϕ
        LOAD 6ϕϕϕ

        LD L,5          ; Ismétlések száma
HANG:   LD H,17ϕT       ; Hangmagasság végértéke
        LD DE,ϕ         ; Hangmagasság kezdőértéke
HANG1:  LD BC,1         ; Hang hossza
        CALL 3F68       ; BEEP rutin
        INC E           ; Hangmagasság csökkentése
        LD A,E
        CP H            ; Hangmagasság ellenőrzése
        JR NZ,HANG1
        DEC L           ; Ismétlésszámláló csökkentése
        JR NZ,HANG      ; Megvolt az 5 db ismétlés?
        ; Újabb hangfajta
        LD L,5
HANG2:  LD DE,ϕFF
HANG3:  LD BC,1
        CALL 3F68
        DEC E           ; Hangmagasság növelése
        JR NZ,HANG3     ; ϕ a hangmagasság?
        DEC L
        JR NZ,HANG2

EXIT:   NOP             ; Töréspont helye
        END
```

SCROLL JOBBRA

```
        ORG 6ϕϕϕ
        LOAD 6ϕϕϕ
SOR:    EQU 3                   ; Eltolandó sorok száma

        LD IX,4ϕ42              ; DCB címe
        LD B,ϕ                  ; Eltolások száma (256!)
SCR1:   PUSH BC                 ; Külső ciklus kezdete
        LD L,ϕ
        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,2ϕ
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
```

GÉPTÍPUS MEGÁLLAPÍTÁSA

```
        ORG 6ϕϕϕ
        LOAD 6ϕϕϕ

        LD IX,4ϕ42          ; DCB címe
        CALL 1C9            ; CLS
        LD HL,STRING        ; STRING címtől az első
        CALL 28A7           ; ϕ byte-ig a videóra
        LD A, (IX+8)
        CP 68               ; 68ϕϕ?
        JR Z,A32
        CP ϕA8              ; A8ϕϕ?
        JR Z,A48
        CP ϕE8              ; E8ϕϕ?
        JR Z,A64
        LD HL,ERROR         ; Egyik érték sem fordult elő!
        JR VÉGE
A32:    LD HL,STRING1
        JR VÉGE
A48:    LD HL,STRING2
        JR VÉGE
A64:    LD HL,STRING3
VÉGE:   CALL 28A7
        LD HL,STRING4       ; SPACE,CHR$ (1) , RETURN, . .
        CALL 28A7
        CALL 25             ; Billentyűlenyomásra vár
        JP 42EC             ; Assembler melegindítása

ERROR:  DB "PROGRAMHIBA!",ϕ ; Stringterület
STRING: DB "AZ ÖN GÉPE: "
        DB 2,4," PRIMO ",ϕ
STRING1: DB "A-32",ϕ
STRING2: DB "A-48",ϕ
STRING3: DB "A-64",ϕ
STRING4: DB 2ϕ,1,ϕD
        DB "TÍPUSÚ",ϕ

        END
```

BASIC KULCSSZAVAK

```
        ORG 6ϕϕϕ
        LOAD 6ϕϕϕ

MEZÖ:   EQU 7           ; Output mező szélessége
START:  EQU 165ϕ        ; Kulcsszótábla kezdőcíme
STOP:   EQU 1822        ; Kulcsszótábla végcíme

        LD IX,4ϕ42      ; DCB címe
        CALL 1C9        ; CLS
        LD HL,START     ; Mutató kezdőértéke
        LD DE,STOP      ; Mutató végértéke
        LD B,ϕ          ; ϕ-ik mező szélessége
RUN:    LD A, (HL)      ; Következő karakter beolvasása
        AND 7F          : 7.bit nullázása
        LD C,A          ; Karakter elmentése
        CP (HL)         ; ϕ volt a 7.bit?
        INC HL          ; Mutató a következő karakterre
        JR Z,CONT       ; CONT, ha nincs vége a szónak
        CALL TABULA     ; Új szó előtt tabulálás
        CALL PAUSE      ; Billentyűzet figyelése
        RST 18          ; HL—DE összehasonlítása
        JR NC, VÉGE     ; HL≥DE, akkor vége
CONT:   LD A,C          ; Kiírandó karakter visszatöltése
        CALL 15         ; Karakter a képernyőre
        DEC B           ; A mezőből elhasználtunk egy helyet
        JR RUN          ; Vissza a ciklus elejére
VÉGE:   CALL 25         ; Billentyűlenyomásra vár
        JP 42EC         ; Assembler melegindítása

PAUSE:  CALL 1D         ; Ha van lenyomott billentyű, akkor
        JR NZ,PAUSE     ; addig vár, míg el nem engedik
        RET

TABULA: LD A,B          ; Ha B=ϕ (a teljes mező be van töltve),
        OR A            ; akkor nem csinál semmit,
        JR Z,TABULA1    ; egyébként a maradékot szóközzel
        LD A,2ϕ         ; tölti fel.
TABULA2: CALL 15
        DJNZ TABULA2
TABULA1: LD B,MEZÖ      ; B kezdőértéke
        RET
        END
```