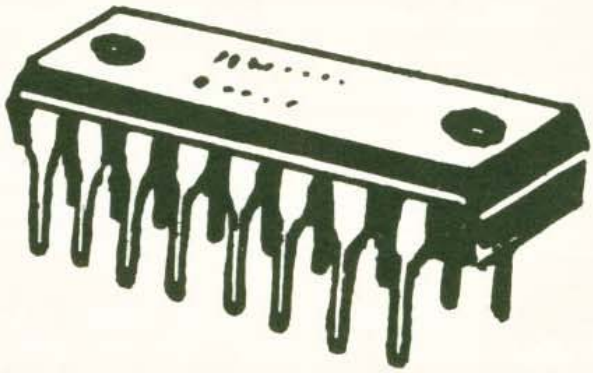
VIDEOTON



assembler

TVC

TVC ASSEMBLER

gépi kódú assembler program

1. Bevezetés

A TVC ASSEMBLER gépi kódú kétmenetes assembler program, amely hatékonyan támogatja a Z80 assembly nyelven írandó program szerkesztését és gépi kódra történő fordítását. Kezdők számára tanácsos némi szakirodalom tanulmányozása az assemblerprogram alkalmazása előtt.

A TVC ASSEMBLER néhány előnyös tulajdonsága:

- a TVC eredeti, teljes képernyős editorát használja a forrásprogram szerkesztéséhez
- a memóriában bárhol futó program készítésére alkalmas
- gyors
- lehetővé teszi makrók kezelését
- a tárgykód részére automatikusan helyet biztosít

2. A program betöltése

A program a

LOAD"TVC-ASSEMBLER" vagy LOAD

paranccsal tölthető be a memóriába. A betöltés végeztével a program bejelentkezik, és a gép parancs módba kerül, hogy a forrást betölthessük.

A forrás bevitele után EXT1 parancs hatására indul az assemblerprogram és megjelenik a képernyőn a menü.

A betöltés alatt ne nyomjuk meg a CTRL+ESC billentyűt, mert akkor a program betöltése megszakad.

3. A menü

Ebből választhatjuk ki a megfelelő billentyű (1-8) megnyomásával a kívánt funkciót.

Az alábbiak közül választhatunk:

1. INICIALIZÁLÁS

   Alapállapotba állítja a TVC ASSEMBLER-t: a létrehozott tárgykódot törli, de a forrást nem. Ha már egy előző assemblálás következtében van tárgykód a memóriában, és nem alkalmazunk INICIALIZÁLÁSt, akkor a következő assemblálászkor keletkező tárgykódot az előző után teszi.

2. HIBAELLENŐRZÉS, ASSEMBLÁLÁS

   Itt kéri az assemblálási opció megadását. Ezután ellenőrzi a forrást anélkül, hogy tárgykódot készítene. Hiba észlelése esetén hibajelzést ad, kilistázza a forrásnak azt a sorát, amelyben a hibát észlelte, és a gép parancs módba kerül, hogy a hiba javítható legyen.
   Hibátlan forrás esetén kiírja a HIBÁTLAN szöveget, helyet biztosít a tárgykód részére, és elvégzi az assemblálást. A helybiztosítást és az assemblálást a megadott opció szerint végzi el.

3. SZIMBÓLUMTÁBLÁZAT KIÍRÁSA

   Képernyőre vagy nyomtatóra írja a fő szimbólumtáblázat címkéihez tartozó értékeket decimálisan és hexadecimálisan egyaránt. Csak azokat a címkéket listázza ki, amelyeknek az értéke nem nulla. A listázás CTRL+P-vel felfüggeszthető.

4. SZIMBÓLUMTÁBLÁZAT KIMENTÉSE (SAVE)

   A fő szimbólumtáblázatot kimenti.

5. SZIMBÓLUMTÁBLÁZAT BETÖLTÉSE (LOAD)

   A kimentett fő szimbólumtáblázatot betölti.

6. KILÉPÉS FORRÁS BEVITELÉHEZ

   A gép parancs módba kerül, információt kapunk a forrás szerkesztését segítő rutinok hívásáról.

7. GÉPI KÓD KIMENTÉSE (SAVE)

Az előállított tárgykódot háttértárra menti.

8. RENUMBER

A forrás átsorszámozására szolgál.

A választott funkció végrehajtása után általában a menühöz tér vissza, kivétel a 6-os és hiba észlelése esetén a 2-es funkció.

A 4-es, 5-ös és 7-es funkció esetén a filenév legfeljebb 16 karakter hosszú lehet. Ha a megadott név éppen 16 karakter hosszú, akkor a 16. karakter beadásakor automatikusan elkezdődik a választott művelet végrehajtása. Ha a név rövidebb, a művelet a lezáró RETURN-re kezdődik el. Ezeknél a műveleteknél a CTRL+ESC megnyomása a menühöz való visszatérést eredményezi, miközben az éppen végzett művelet végrehajtását megszakítja. A nevet figyelmesen gépeljük be, mert nem javítható.

4.Forrásprogram szerkesztése

A forrás assembly nyelven írt szöveget és különböző utasításokat tartalmazhat, ezeket az assembler értelmezi, és segítségükkel előállítja a tárgykódot. A forrásprogramot parancs módban vihetjük be a memóriába billentyűzetről vagy háttértárról.

4.1. A forrás elhelyezkedése:

A forrást BASIC ! vagy REM sorok formájában kell elhelyezni. Sorszámként a 2000-9999 tartomány alkalmazható.

4.2. Szerkesztést segítő rutinok:

A program két gépi kódú rutint tartalmaz, amelyek a forrás szerkesztését segítik.

4.2.1. APPEND rutin:

Ha ezt a rutint az EXTO paranccsal beindítjuk, akkor a memóriában lévő forrás (BASIC program) vége utáni byte lesz a gép számára a BASIC programterület kezdete. Ez teszi lehetővé több részből álló forrásszöveg egymás után töltését a memóriába háttértárról.

Ezt a rutint az EXTO parancs ismételt alkalmazásával kapcsolhatjuk ki, ekkor visszaáll a bekapcsolás előtti állapot. A rutin bekapcsolt állapotában az EXT1 parancs, (amely az assemblerprogramot indítja) hatástalan. Ilyenkor egy EXTO paranccsal kapcsoljuk ki a rutint.

A rutin pillanatnyi allapotáról a LIST parancs alkalmazásával győződhetünk meg, amely az érvényes BASIC területen lévő programot listázza ki.

4.2.2. RENUMBER:

Ez egy forrásprogramot átsorszámozó rutin. Hibátlan forrásprogram esetén a forrásprogram sorszámai nem érdekesek, az assembler hibátlanul lefordítja a forrást gépi kódra. Ha viszont a forrásprogramban hiba van, a hibás sort képernyőre író rutin csak akkor működik helyesen, ha a sorok sorszámozása helyes. Ha billentyűzetről viszünk be egy forrást, akkor a gép editora gondoskodik arról, hogy a sorok a memóriában a sorszámozás sorrendjében helyezkedjenek el. Ha több forrást vagy makrókat töltünk be háttértárról, a helyes sorszámozásról gondoskodnunk kell. Csak az érvényes BASIC területen lévő forrást sorszámozza át.

A rutin kétféle módon hívható:

- menüből

- parancs módból szerkesztés közben.

1. Hívása menüből (8-as funkció):

Sorban megkérdezi az átsorszámozás paramétereit. Ha az első kérdésre csak a RETURN megnyomásával válaszolunk, akkor a beépített paraméterekkel végzi el a sorszámozást (a 2000-es, vagy ha az nincs, akkor a következő nagyobb sorszámú sortól kezdve, 2000-es új kezdősorszámmal, 10-es inkrementummal). Az inkrementum megadható maximális értéke 255, minimális értéke 1.

Olyan forrásprogramot, amelyben az első átsorszámozandó sor sorszáma 2000-nél kisebb, nem sorszámoz át.

Gondoskodik arról, hogy a maximális keletkező sorszám

9999 alatti legyen. Ha ez nem teljesül, egyesével csökkenti az inkrementumot addig, amíg sikerül végrehajtania a helyes átsorszámozást. Ebben az esetben jelzi, hogy inkrementumot változtatott. A felső határt a következőképpen vizsgálja: ha az utolsó sor után még egy sor lenne, annak új sorszáma nem haladhatná meg a 9999-et.

Amennyiben 1-re csökkentett inkrementummal sem tudja elvégezni az átszámozást, hibajelzést ad, és a megfelelő paramétert újra kéri.

2. Hívása parancs módból:

Az EXT5 parancs hatására az átsorszámozás a beépített paraméterekkel megtörténik. Hibajelzést ilyenkor nem kapunk.

Forrásprogramot a TVC ASSEMBLER betöltése nélkül is szerkeszthetünk, hiszen az szabályos BASIC program. Célszerű a sorszámozást 2000-től kezdeni, és a végén a szokásos módon kimenteni (SAVE "név").

Ha a forrás kisebb kezdősorszámmal kezdődik, ezt a következőképpen tudjuk átsorszámozni:

- a forrást kimentjük
- betöltjük a TVC ASSEMBLERt
- beírunk egy fiktív sort: 200[illegible]M (vagy 2000!)
- beadjuk a következő parancsot:

  EXT0:LOAD"név",

  és betöltjük az átsorszámozandó forrást.

- Az EXT0:EXT5 paranccsal átsorszámozzuk.
- a fiktív 2000-es sort törölhetjük.

4.3. Néhány egyszerű forrásprogram-művelet:

4.3.1. Forrásprogram listázása nyomtatóra:

- parancs módba kerülünk
- LIST

4.3.2. Forrásprogram kimentése:

- parancs módba kerülünk
- SAVE"név"

4.3.3. Forrásprogram betöltése

4.3.3.1. Memóriában lévő forrás törlésével:

- parancs módba kerülünk
- LOAD"név"

4.3.3.2. Összefűzés memóriában lévő forrással:

- parancs módba kerülünk
- EXTO:LOAD"név"
- EXTO

4.3.3.3. Több részből álló forrás egymás után töltése vagy több makró betöltése:

- parancs módba kerülünk
- Az előző pontban leírt műveleteket alkalmazzuk egymás után többször, amíg mindent be nem töltöttünk:

```
EXTO:LOAD"prog1"
EXTO:EXTO:LOAD"prog2"
EXTO:EXTO:LOAD"prog3"
.
.
.
EXTO
```

4.4. A szerkesztés általános szempontjai:

Ha a menü látható a képernyőn és a 6-os funkciót választjuk (KILÉPÉS FORRÁS BEVITELÉHEZ), parancs módba kerülünk.

A forrásprogramot az előzőekben leírtak szerint tölthetjük be háttértárról.

Mint már említettük, a forrást BASIC ! vagy REM sorokba kell beírni, a megjegyzések kivételével kötelezően nagybetűvel írva. Ennek érdekében a betöltés után a gép automatikusan CAPS LOCK módba kerül.

A REM vagy ! után nem kötelező szóközt tenni, de tetszés szerint tehető is. A címke és a mnemonik (utasítás neve) között ugyanez a helyzet. A mnemonik és (ha van) az operandus között legalább egy szóközt kell hagyni, az operandusokat szóköz nélkül, folyamatosan gépeljük be. Ha jónak látjuk, az editor tabulátorával (CTRL+I) tabulálhatunk is azokon a helyeken, ahol szóköz meg van engedve. A direktívák és operandusuk között nem szükséges szóköz, de megengedett akár több is.

Egy sorba több utasítás is kerülhet, ha azokat ; (pontosvessző) karakterrel választjuk el egymástól. Azt figyelembe kell venni, hogy a TVC BASIC egy sorának hossza max. 250 karakter. A ; közvetlenül az utasítás, illetve (ha van) az operandus után állhat, vagy tetszőleges számú szóköz előzheti meg. A sorban utolsónak álló utasítás után nem kell pontosvesszőt írni! Szintaktikailag a megjegyzésmező kezdetét jelző (csillag) karakter is utasításnak számít.

4.4.1. Forráskezdet és forrásvég jelzése:

A forrás kezdetét egy külön REM vagy ! sorban lévő, a REM vagy ! után közvetlenül álló kerek nyitó zárójel jelzi. A végét egy ugyanígy elhelyezett kerek csukó zárójel.

Ezekben a sorokban ne szerepeljen más utasítás.

A forráskezdet- és forrásvégjelzők az assembler számára jelzik a lefordítandó forrás határait. Ha egy forrásba több jelzőpárt írunk, akkor csak az elsőt veszi figyelembe!

Makró hivatkozás történhet a forrásvégjelző utánra is. Pl.

```
2000 !(
.
.       itt van a forrásszöveg
.
3000 !)
```

Csak a 2000 és 3000 sorok közötti forrást fordítja le, akkor is, ha a 3000 sorszámú sor után további forrássorok

vannak, kivétel a makródefiníció, amelynek törzsét a hivatkozás helyére fordítja.

Ha a forráskezdet vagy forrásvégjelzőt elfelejtettük kitenni, a 'kezdő zárójelet nem találom', illetve a 'záró zárójelet nem találom' hibajelzést kapjuk.

4.4.2. Megjegyzésmező:

Ebbe helyezhetjük el megjegyzéseinket. A megjegyzésmező kezdetét egy karakter jelzi, amely szintaktikailag utasításnak számít. Az assembler a *-tól kezdve a sor hátralévő részét nem veszi figyelembe, hanem a következő BASIC sorban folytatja a fordítást. Ezért a helyes működés érdekében a megjegyzésmező után már ugyanabban a sorban nem állhat utasítás. A makrók helyes működésének érdekében a megjegyzésmezőkben kerüljük a szögletes zárójelek alkalmazását!
Pl.:

```
2050 !   Memóriaterület vizsgálata
2055 !
2060 !       LD   HL,20000;   HL-ben a kezdőcím
2070 !       LD   A,(HL)  ;   karakterkód A-ba
```

4.4.3. Címkék

A címkék olyan szimbolikus nevek, amelyekhez egy-egy számérték van rendelve, és a címke nevének leírása helyettesíti az általa képviselt számértéket.

Egy címke kétféle módon kaphat értéket:

- címkéhez értéket rendelő direktíva (=) segítségével, vagy

- egy utasítás megjelölésével, amikor is a címke értéke a megjelölt utasítás első byte-jának memóriabeli címe lesz a futási helyen.

Ha egy utasítást egy címkével kívánunk megjelölni, akkor azt a megjelölendő utasítás mnemonikja elé írjuk : (kettőspont) karakterrel kezdve a címke nevét. Ha egy utasításban operandusként hivatkozunk egy címkére, akkor nem kell a : (kettőspont)! A kettőspontnak közvetlenül a címke neve előtt kell állnia. Pl.:

```
5000!:K114DS3;:K130  DS  14 ;:K106'TVC';LD B,3;LD HL,K114;:GO
 LD A,(K106);:K28 LD (HL),A;DJNZ GO
```

A fő szimbólumtáblázatban 256 címke használata megengedett. Ezek a K0-K255 jelű címkék. A másodlagos szimbólumtáblázatból újabb 256 címkét lehet használni. Ezeket G0-G255 jelöli. A két szimbólumtáblázat címkéit egyformán használhatjuk, azonban a másodlagos szimbólumtáblázat címkéinek értékei nem írathatók ki assemblálás után, nem menthetők ki háttértárra, és a szimbólumtáblázat törlése opció nem hat rájuk.

A címkéket hatékonyan tudjuk használni programunkban, mert:

- a fő szimbólumtáblázat elmenthető ill. visszatölthető

- van címkét definiáló direktíva (=)

- az előző fordítás eredményeként keletkezett fő szimbólumtáblázatot megőrizhetjük a memóriában, vagy törölhetjük. A másodlagos szimbólumtáblázat nem törlődik.

- a forrásban bárhol, ahol adatot vagy címet kell megadni (ill. makró nevében), alkalmazhatunk címkét is. Kivétel az IM és RST utasítás, valamint a bitműveleteknél a bitsorszám megadása.

A két szimbólumtáblázat címkéi között az '=' direktíva segítségével adhatók át a paraméterek.
Pl:

```
K10=G12
G0=K10
G2=K14- 1000+G0
```

A másodlagos szimbólumtáblázat a memóriában a 12647 címtől kezdve helyezkedik el, és minden címke értéke 2 byte-on van tárolva. A G0 címke értéke a 12647-12648 címen, a G1 a 12649-12650 címen, stb. található.

Az assembler nem ellenőrzi, hogy programban defininiált címkét használunk-e, vagy nem kétszeresen definiáltuk-e valamelyiket. Ezt magunknak kell ellenőriznünk. Törölt szimbólumtáblázat esetén a nem definiált fő szimbólumtáblázatbeli címkék értéke nulla. A gépi kódú program belépési pontjait célszerű egy-egy K jelű címkével megjelölni, így a fordítás után a szimbólumtáblázat kilistázásával megkaphatjuk a belépési pontok címeit.

4.4.4. Számok

Az assembler a számokat decimális számként értelmezi, kivéve, ha $ (dollár) jellel előjelöljük, amikor is hexadecimális számként kezeli. A számjegyeknek közvetlenül az $ jel után kell állniuk.

Megkötések:

- a címke nevében hexadecimális szám nem alkalmazható (pl. K$10 érvénytelen címke).

- az RST utasításokban csak hexadecimális szám alkalmazható, de $ előjelölés nélkül (pl.: RST 10 szabályos, RST $10 illegális). Az RST 08 és RST 00 utasításnál a vezető nulla kiírása kötelező.

- az IM utasításban tilos a számot $ jellel előjelölni, és vezető nullát sem lehet alkalmazni! Ugyanez vonatkozik a bitműveleteknél a bitsorszám megadására is. (pl.:IM $ 1 illegális). Az egyedüli helyes megadás: IM 1;IM 2;IM 3;BIT 2,A;stb...

4.4.5. Utasítások, operandusok

Az assembler általában a standard ZILOG mnemonikokat használja az utasításokban. Az ettől való eltéréseket itt jelezzük, illetve az assembler néhány tulajdonságáról is szólunk.

- Az assembler az utasításhoz tartozó operandusok (vagy ha nincs operandus, akkor az utasítás neve) után következő pontosvesszőig vagy sorvégjelző (FFH) karakterig minden karaktert figyelmen kívül hagy. Ezért ha egy sorba több utasítást írunk, és két utasítás közül kihagyjuk a pontosvesszőt, akkor a másodikat úgy tekinti, mintha ott sem volna! Emiatt a pontosvessző használatára fokozottan ügyeljünk! A sorban utolsóként álló utasítás után nem kell pontosvesszőt tenni.

- A standard EX AF,AF' helyett elfogadja az EX AF,AF formát is.

- AZ OUT (port),A és IN A,(port) utasításokat zárójel nélkül kell használni. Pl.: OUT 2,A;IN A,2

- Az RST és az IM utasítás esetén a 4.4.4. pontban leírtak érvényesek, azzal a kiegészítéssel, hogy a szám helyett

címkék sem állhatnak. Ugyanez vonatkozik a bitműveletekben a bitsorszámra is.

- A relatív ugró utasítások esetén (a JR utasítások valamint a DJNZ) az eltolási byte értéke (displacement) kétféleképpen adható meg:

  1. Címkével megjelöljük a címet, ahová ugrani kell, és a relatív ugró utasításban a címke nevének leírásával nivatkozunk rá. Pl.:

```
.
.
.
3000 !:K20   LD A,(HL)
.
.
3050 !       JR Z,K20
```

  2. Előjelesen megadjuk, hogy hány byte-tal kívánunk ugrani (a negatív előjel a visszafelé ugrást jelenti). Az ugrás számításának kezdetét szokásos módon, a relatív ugró utasítás utáni utasítás első byte-ja jelenti. Az eltolás -128...+127 közötti lehet, különben hibajelzést kapunk. Az előjelet mindenképpen ki kell tenni. A számérték, mint minden más esetben, lehet decimális, hexadecimális szám vagy címke. Nézzünk néhány helyes megadási módot:
  JR +10;JR -10;JR +$10;JR -$10;K12=10;JR +K12;JR Z,-K12;D
  JNZ -16;DJNZ -K12; stb.

- Index regisztereket (IX vagy IY) alkalmazó utasításoknál az eltolási érték az előző, 2. pont alapján, előjelesen adható meg.

- Az előzőekben megadott tiltások kivételével címkéket tetszőlegesen lehet címek, adatok, eltolások, portok helyett alkalmazni. Címkék használata makrónevekben is megengedett. Pl.: K100=2000;MK100 Itt MK100 ugyanaz, mintha M2000-t írtunk volna.
  - Adat, cím, port vagy eltolási érték esetén hibajelzést kapunk, ha nem a megengedett intervallumba eső értéket akarunk alkalmazni. ('Numerikus érték az intervallumon kívül'. Előjeles eltolás esetén lehet még 'Tiltott utasítás' hibajelzés is.)

4.4.6. Direktívák

A direktívák az assemblerprogram számára jelentenek utasítást. Egy részük tárgykódot is létrehoz.

4.4.6.1. Az = direktíva

Ez a szokásos EQU direktíva helyett használatos. Egy címkéhez lehet ennek segítségével értéket rendelni.

Formája: címke=OP1±OP2±OP3.....±OPn,

ahol OP1,OP2,OP3.....OPn lehet címke, decimális szám vagy hexadecimális szám. A '+' helyett '-' is állhat.

PL.: K20=1000+$100-K11+K4-$20-200

Ennek használatánál a következőre kell ügyelni: a műveleteket balról jobbra haladva végzi el. Ha bármely részeredménynél alul- vagy túlcsordulás következik be (0-nál kisebb vagy 65535-nél nagyobb részeredmény), akkor hibajelzést kapunk. Először a jobb oldal kiszámítását végzi el utána történik meg a címkéhez történő hozzárendelés. Így a következő utasítás a K4 címke értékét 2-vel megnöveli: K4=K4+2.

Ebben a direktívában a K-t és a G-t szám nélkül is elfogadja és K0-nak ill. G0-nak értelmezi.

4.4.6.2. A < direktíva

Ez a szokásos ORG direktíva helyett használatos. Ebben a leendő végrehajtási hely betöltési kezdőcímét adjuk meg az assemblernek, vagyis azt a címet, amelytől kezdve a keletkező tárgykódot be akarjuk tölteni, majd végre akarjuk hajtani.

Formája: < cím , ahol a cím lehet címke, decimális szám vagy hexadecimális szám.

Ez elvileg bárhol elhelyezhető a forrásprogramban, azonban értelmesen csak úgy tudjuk használni, ha csak egyszer szerepel, és csak olyan utasítások előzik meg, amelyek nem hoznak létre tárgykódot (például címkedefiniálás, makródefiníció). Ha elfelejtjük megadni, értéke egy előző assemblálás során megadott, a memóriában maradt érték lesz.

4.4.6.3. A DS direktíva

Formája: DS xxx , ahol xxx lehet címke, decimális vagy hexadecimális szám.

Az assembler a DS operandusaként megadott számú byte-ot átugrik fordítás közben.

4.4.6.4. A DB és a DW direktíva

Formája:

```
DB xxx
DW xxx
```

ahol az xxx címke, decimális vagy hexadecimális szám lehet.
A DB direktíva az operandusaként megadott egybyte-os értéket beírja a fordításkor soron következő byte-ba. Az operandusnak természetesen 0-255 közöttinek kell lennie.
A DW direktíva az operandusaként megadott értéket kétbyte-os számnak tekinti, és a Z80 elvárásának megfelelő sorrendben beírja a fordításkor soron következő két byte-ba (először az LSB-t, utána az MSB-t). Az operandusnak 0-65535 közé kell esnie.
Pl.: DW$1000 00-t és 10-et ír a két egymást követő byte-ba.

4.4.6.5. A ' (aposztróf) direktíva

Formája: 'szöveg' ('=SHIFT+1)

A szöveg karakterkódjait egymás után elhelyezi a memóriába. Ez csak önálló utasításként alkalmazható, egyéb utasítás operandusaként nem (pl. helytelen: LD A,'C').

Megkötések:

- A szöveg nem tartalmazhat ' (=SHIFT+1) ill. ; (pontosvessző) karaktert. Ez azonban azzal az előnnyel jár, hogy ha elfelejtjük kitenni a lezáró felsővesszőt, akkor a következő ; (pontosvessző) vagy sorzáró karakter (FF) is lezárja az utasítást.

- A szövegben csak 128-nál kisebb karakterkódú karakterek írhatók be, ugyanis a TVC a megjegyzéssorokban a 128 ill. ennél nagyobb kódú karakterek kódjából 128-at levon, és így tárolja. Így az ékezetes karaktereknél is fellép ez a probléma. Ekkor a karaktert a 'DB karakterkód' módon közvetlenül kell megadni. Pl. a BÁJT szó megadása:
:K10'B';DB128;'JT' . A K10 címke a "B" memóriabeli címét tartalmazza, amelyet a BÁJT szó kiíratásakor felhasználhatunk. A címke használata természetesen nem kötelező.

4.4.7. Makrókezelés

A TVC-ASSEMBLER program egy egyszerű makrókezelést is lehetővé tesz, amely nagymértékben segíti a program alkalmazhatóságát.

A makró egy olyan forrásprogramrész, amelynek törzsében szereplő utasításokat az assembler csak a rá való hivatkozás helyénél fordítja be a tárgyprogramba.

A makró neve egy M betű és egy közvetlenül utána álló szám, amelyet decimális ill. hexadecimális számként vagy címkeként adhatunk meg. A számnak a 0-65535 tartományba kell esnie, azaz a program 65536 különböző makrót tud megkülönböztetni. Érvényes makrónevek pl.: M10000;M$BAAA;MK10;MG100

A makróknak a forrás kezdetét jelző "(" után kell elhelyezkednie, azonban lehet a forrás végét jelző ")" utáni ! vagy REM sorokban is, így a forrásprogram elkészülte után csak be kell tölteni a szükséges makrókat, de nem szükséges azokat a forrásszöveg belsejébe szerkeszteni.

A makróra való hivatkozáshoz a forrásprogramban egyszerűen leírjuk a címke nevét, mint egy utasítást.

A makródefiníció kezdetét egy [(szögletes nyitó zárójel) jelzi, amely után a makró neve következik. A [és a makró neve között szóközök alkalmazása megengedett. Az [és a makró neve együtt szintaktikai szempontból egy utasításnak számít.

Sorrendben ezt követi a makró törzse, amely azokat az utasításokat tartalmazza, amelyeket a rá való hivatkozás helyére fordít az assembler.

A makródefiníció végét egy] (szögletes csukó zárójel) jelzi. Ez szintaktikai szempontból egy önálló utasítás.

Nagyon vigyázzunk arra, hogy a makródefiníció kezdetét és végét jelző jeleket mindig tegyük ki, ezek mindig párban legyenek, mert ellenkező esetben végzetes hibához vezethet, ill. nem a szándékunknak megfelelő tárgykód keletkezik.

A makró müködésének lényege az, hogy a hivatkozás helyére fordítja a hívott makró törzsében lévő utasításssorozatot. Ha a gyakran használt szubrutinokat makróként definiáljuk, és külön tároljuk háttértáron, akkor nem kell őket mindig begépelni, és megírni a forrásprogramban, hanem csak a makró nevét kell leírni és fordítás előtt a megfelelő makrókat betölteni.

Ha nem történt makróhivatkozás, akkor fordítás közben a makródefiníciókat egyszerűen átugorja.

Makrók nem ágyazhatók egymásba, azaz makródefiníció belsejében a makróhivatkozás tilos.

A szögletes zárójeleket más célra ne használjuk (pl. megjegyzésekben se).

Egy értelmetlen példán bemutatjuk a makrók használatának szintaktikáját:

```
2000 !(
2010 !<8000
2020 ![M0;LD B,0;XOR A;]
2060 ! LD B,5;INC HL;[M10;DEC DE;];EX DE,HL
2070 !XOR A;M1000;LD D,A;LD B,6
2080 !INC HL;:K101M1000;M10;M0
2100 !)
3000 ![M1000;LD A,(HL);OR A;RET Z;INC HL;]
```

Nézzük röviden, mi történik ennek fordítása során. Az első lefordított utasítás a 2060-as sorban lévő LD B,5 lesz, majd az INC HL. Az M10-es makró definíciója kimarad, és EX DE,HL következik, majd a XOR A. Ekkor következik a hivatkozás az M1000 makróra, így ennek törzse itt fordítódik le, majd a következő utasítással (LD D,A) folytatódik a fordítás. A 2080-as sorban hivatkozás történik egymás után mindhárom makróra, így azok törzsei a hívás helyére befordítódnak.

A fenti példában egy makró egy BASIC soron belül helyezkedik el, azonban erre semmilyen megkötés nincs.

5. A tárgykód mentése és betöltése

5.1. GÉPI KÓD KIMENTÉSE (SAVE)

Ez a funkció a menüből hívható. Az összes addig létrehozott tárgykódot elmenti a háttértárra a megadott névvel, az operációs rendszer rutinjainak segítségével. Ha nincs elmenthető tárgykód, azt üzenettel jelzi.

5.2. Elmentett tárgykód futási helyre töltése

5.2.1. Helybiztosítás, betöltés

Ez többféleképpen megvalósítható, most csak egy nagyon egyszerű megoldását írjuk le.

- A gépi kódot 6639 futási hely betöltési kezdőcímmel készítjük el (forrásban < 6639), és a tárgykódot kimentjük.
- Hideg resetet hajtunk végre.
- Kiadjuk a következő három parancsot:

```
LOMEM 6639+tárgykód hossza
POKE 5920,PEEK(5922)
POKE 5921,PEEK(5923)
```

- Betöltjük a LOAD"név" paranccsal a következő részben

leírt betöltő programot, és ennek segítségével betöltjük az elmentett tárgykódot. Kezdőcímnek 6639-et adunk meg.

5.2.2. Betöltő program

Most egy egyszerű betöltő program leírása következik, amelynek segítségével a kimentett tárgykód betölthető. Az operációs rendszer rutinjait használja az 1. sorszámú sorban lévő 3 byte-nyi gépi kód segítségével.

A betöltő program elkészítése:

- A resetet kétszer megnyomjuk (hideg reset).

- Beírjuk a következő BASIC sort:

```
1!123
```

- Begépeljük az alábbi parancsot:

```
POKE 6643,247:POKE 6645,201
```

- Ezután beírjuk a program többi részét:

```
 10 A=PEEK(5922)+256 PEEK(5923)
 20 INPUTPROMPT"KEZDOCIM: ":C
 30 INPUTPROMPT"HOSSZ   : ":H
 40 INPUTPROMPT"NEV     : ":N$
 50 POKE A+5,211
 60 POKE 37,A+4-256 INT((A+4)/256)
 70 POKE 38,INT((A+4)/256)
 80 EXT 2,0,VARPTR(N$)+1
 90 POKE A+5,210
100 EXT 2,0,C,H
110 CLOSE
120 END
```

- A kész programot kimentjük.

Használatkor RUN paranccsal indítjuk. A megadandó "HOSSZ"-t az assemblálás végén kiírt 'A program jellemzői'-ből olvashatjuk ki. Ugyaninnen kaphatjuk meg a 'KEZDŐCÍM'-et is ("<" direktívában megadott érték).

6. Néhány szó a program használatáról

A forrásprogram szerkesztése vagy betöltése után EXT1

paranccsal indíthatjuk a programot, és megjelenik a menü. Ha nem így történne, akkor az APPEND "bekapcsolt" állapotban van. Ekkor ki kell adni egy EXT0 parancsot, majd ismét egy EXT1-et.

Ha szükséges, INICIALIZÁLÁS-t alkalmazunk (ilyenkor minden előzőleg készített tárgykód törlődik!!).

Meghívjuk a HIBAELLENŐRZÉS, ASSEMBLÁLÁS funkciót, melynek hatására megjelenik az assemblálási opciókat tartalmazó táblázat. Két szempont figyelembevételével választhatunk opciót:

- a szimbólumtáblázatot törölje-e vagy ne,

- készítsen-e tárgykódot vagy ne.

Ha csak a RETURN-t nyomjuk meg, akkor a 0 opciót választjuk (a szimbólumtáblázat törlődik, és tárgykódot készít), amely a legtöbb esetben megfelelő.
Ha a hibaellenőrzés során a program hibát észlel, hibajelzést kapunk, ill. kilistázza a hibás forrássort, és parancs módba kerül. Hibátlan forrás esetén a megadott opció figyelembevételével elvégzi az assemblálást, majd a képernyőre írja a keletkezett tárgykód jellemzőit:

- a hosszát byte-okban,

- a futási hely betöltési kezdőcímét, amely a direktívában megadott érték,

- ha keletkezett tárgykód, annak elhelyezési kezdőcímét,

- a választott opciót.

A kiírt jellemzőket célszerű feljegyezni.

Az assembler a helybiztosítás előtt ellenőrzi, hogy a tárgykód elhelyezéséhez szükséges hely rendelkezésre áll-e. Ha nem, hibaüzenetet kapunk (NINCS ELÉG HELY), és nem végzi el az assemblálást. Ilyenkor az segít, ha a forrást kétfelé vesszük, és a memóriában egyszerre csak az egyik fele van benn. Ilyen több részből álló forrást, amelyben az éppen memóriában nem lévő forrásrészben meghatározott címkékre is hivatkozunk, a következőképpen assemblálhatunk:

1. Betöltjük a forrásprogram első részét, amelybe utolsó utasításnak (a forrásvégjelző elé) beírunk egy máshol nem használt címkével megjelölt fiktív utasítást:
:K255 DS0

A forrást kimentjük, és 3-as opcióval assemblálunk. Betöltjük a következő részt, és az elején megadjuk a K255 direktívát. Vigyázzunk, hogy csak ez az egy direktíva maradjon a forrásprogramban. Ha még van hatralévő betöltendő része a forrásprogramnak, akkor ennek a résznek a végére is teszünk az előzőkben leírtak szerint egy fiktív utasítást:

:G100 DSO

A forrást kimentjük, és 3-as opcióval assemblálunk. A fentieket egymás után minden részen megismételjük.

2. Újra sorban betöltünk minden részt, és 2-es opcióval assembláljuk részenként.

3. A keletkezett tárgykódot kimentjük.

Az 1. pontban csak a teljes szimbólumtáblázatot készítettük el, a másodikban ennek felhasználásával előállítottuk a tárgykódot.

A programmal 0000-FFFE (0-65534) címtartományban végrehajtható tárgykód készíthető. A program ellenőrzi, hogy ha a keletkezett tárgykódot a memóriában a direktívában megadott címtől kezdődően elhelyeznénk, akkor a tárgykód vége túlnyúlna-e a megengedett felső címhatáron. Ha igen, akkor szintén a 'NINCS ELÉG HELY' üzenetet kapjuk, és a program nem végzi el az assemblálást.

INICIALIZÁLÁS után a tárgykód mindig a 13183 (dec) címtől kezdődően tárolódik az assemblerprogram belsejében.

Célszerű az elkészített forrásprogramot mindjárt kimenteni.

Az olyan számértékeket (konstansokat), amelyekről úgy gondoljuk, hogy egy későbbi fordítás során szükség lehet a módosításukra, a forrásprogram elején rendeljük egy címkéhez, lássuk el a funkciójára vonatkozó megjegyzéssel, és a forrás többi részében ezt a címkét használjuk a szám helyett.

A program a betöltés után négyszínű üzemmódba állítja a számítógépet, ez azonban parancs üzemmódból a szokásos BASIC utasítások segítségével megváltoztatható.

A program színbeállítást nem végez, a felhasználó parancs módból beállíthatja a számára legkedvezőbb színösszeállítást.

Ha a forrásprogram valamely sorában elfelejtettük kitenni a sorszám után a REM-et vagy a ! jelet, akkor assemblláskor a 'záró zárójelet nem találom' hibajelzést kapjuk akkor is, ha valójában kitettük a forrásvégjelzőt. Ilyenkor meg kell keresni a hibás sort, és ki kell javítani. Kihasználhatjuk, hogy a BASIC a legtöbb ! vagy REM nélküli forrássort nem érti, és így a legtöbb esetben egy RUN parancs kiadásakor a hibás sor egy hibaüzenet kíséretében megjelenik a képernyőn. Ha ez nem vezet eredményre, akkor a forrás képernyőre listázásával (LIST parancs) keressük meg a hibát.

A TVC-ASSEMBLER használja az EXT0, EXT1, EXT4, EXT5, EXT6 utasításokat, ezért vigyázzunk, hogy az USRTAB ezeknek megfelelő címeinek tartalma ne változzék meg (pl. egy figyelmetlenül kiadott POKE parancs hatására).

Figyeljünk a következőre!

Ha nincs nyomtató csatlakoztatva a számítógéphez, ne akarjuk a szimbólumtáblázatot printerre listázni, mert végzetes hibához vezet! A számítógép ebből az állapotból csak hideg resettel hozható ki, és a memóriából minden törlődik.

A program nem végez mindig mindenre kiterjedő hibaellenőrzést, ezért figyelmesen bánjunk vele.

7.A program üzenetei

HIBAELLENŐRZÉS, ASSEMBLÁLÁS funkció:

- Kezdő zárójelet nem találom
- Záró zárójelet nem találom
- Tiltott címke
- Tiltott utasítás
- Numerikus érték az intervallumon kívül
- Relatív ugrás az intervallumon kívülre
- Tiltott macronév
- Macrot nem találom
- Macrozárót nem találom
- Nincs elég hely
- Hibátlan

GÉPI KÓDÚ KIMENTÉS (SAVE) funkció:

- Nincs tárgykód

RENUMBER funkció:

- Nem találom a kezdősort
- Inkrementumot változtattam
- Még 1 inkrementum esetén is 9999-nél nagyobb sorszám adódik
- Az új kezdősorszám minimális értéke 2000

8. Z80 utasítások

8 bites töltő utasítások:

LD r,r	LD r,(ind+dis)	LD A,(BC)	LD (BC),A
LD r,(HL)	LD (ind+dis),r	LD A,(DE)	LD (DE),A
LD (HL),r	LD (ind+dis),n	LD A,(nn)	LD (nn),A
LD A,I	LD A,R	LD I,A	LD R,A
LD r,n	LD (HL),n		

16 bites töltő utasítások:

LD rp,nn	LD ind,nn	LD rp,(nn)	LD ind,(nn)
LD (nn),rp	LD (nn),ind	LD SP,HL	LD SP,ind
PUSH rp	PUSH ind	PUSH AF	POP rp
POP ind	POP AF		

Cserélő utasítások:

EX (SP),HL	EX AF,AF' vagy EX AF,AF
EX (SP),ind	EX DE,HL
EXX	

Blokkmozgatás és keresés:

LDI	CPI	LDIR	CPIR
LDD	CPD	LDDR	CPDR

8 bites aritmetikai és logikai utasítások:

ADD A,r	ADD A,(HL)	ADD A,(ind+dis)	ADD A,n
ADC A,r	ADC A,(HL)	ADC A,(ind+dis)	ADC A,n
SUB r	SUB (HL)	SUB (ind+dis)	SUB n
SBC A,r	SBC A,(HL)	SBC A,(ind+dis)	SBC A,n
AND r	AND (HL)	AND (ind+dis)	AND n
XOR r	XOR (HL)	XOR (ind+dis)	XOR n

OR r	OR (HL)	OR (ind+dis)	OR n
CP r	CP (HL)	CP (ind+dis)	CP n
INC r	INC (HL)	INC (ind+dis)	
DEC r	DEC (HL)	DEC (ind+dis)	

16 bites aritmetikai utasítások:

ADD HL,rp	ADC HL,rp	ADD ind,BC	ADD ind,DE
ADD ind,SP	ADD IX,IX	ADD IY,IY	DEC rp
INC rp	DEC ind	INC ind	SBC HL,rp

Bitműveletek:

BIT b,r	RES b,r	SET b,r
BIT b,(HL)	RES b,(HL)	SET b,(HL)
BIT b,(ind+dis)	RES b,(ind+dis)	SET b,(ind+dis)

Rotáló és shiftelő utasítások:

RLC r	RRC r	RL r	RR r
RLC (HL)	RRC (HL)	RL (HL)	RR (HL)
RLC (ind+dis)	RRC (ind+dis)	RL (ind+dis)	RR (ind+dis)
SLA r	SRA r	SRL r	RLCA
SLA (HL)	SRA (HL)	SRL (HL)	RRCA
SLA (ind+dis)	SRA (ind+dis)	SRL (ind+dis)	RLA
RLD	RRA	RRD	

Ugró utasítások:

JP f,nn	CALL f,nn	RET f	JP nn
JR C,dis	JR NC,dis	JR NZ,dis	JR Z,dis
JP (HL)	DJNZ dis	JP (ind)	JR dis
CALL nn	RET	RETI	RETN
RST 00	RST 08	RST 10	RST 18
RST 20	RST 28	RST 30	RST 38

Input/output utasítások:

IN r,(C)	IN A,port	INI
INIR	IND	INDR
OUT (C),r	OUT port,A	OUTI
OTIR	OUTD	OTDR

Egyéb utasítások:

DAA	HALT	CPL	DI
IM 0	IM 1	IM 2	EI
NEG	CCF	SCF	NOP

A fenti táblázatban a nagybetűvel írottakat illetve a zárójeleket és a vesszőket ugyanígy kell leírni a forrásprogramban is. Ahol szóköz szerepel, oda legalább egy szóközt kell.tenni. A kisbetűvel írottak különféle jelölések, amelyek helyére az alábbiak szerint kell behelyettesíteni.

Az alkalmazott jelölések a következők:

r : az A, B, C, D, E, H, L regiszterek egyike.
rp : a BC, DE, HL, SP regiszterpárok egyike.
dis : egy egybyte-os eltolási érték, amelynek értéke -128...+127.
n : egy egybyte-os (8 bites) érték.
nn : egy kétbyte-os (16 bites) érték vagy cím.
ind : IX vagy IY indexregiszter.
b : bitsorszám (0...7)
f : C, NC, Z, NZ, PE, PO, M, P feltételek egyike.
port: egy portcím (0...255)

MEGAMICRO-PRESS 87418