FELHASZNÁLÓI KÉZIKÖNYV

az

OC–118 típusú 5 1/4"

FLOPPY–hoz

és

CS–118 típusú TÁPEGYSÉG–hez

IERT

FELHASZNÁLÓI KÉZIKÖNYV

az

OC–118 típusú 5 1/4"

FLOPPY–hoz

és

CS–118 típusú TÁPEGYSÉG–hez

IÉRT

FELHASZNÁLÓI KÉZIKÖNYV

AZ

OC-118 típusú 5 1/4"

FLOPPY-hoz

és

Cs - 118 típusú TÁPEGYSÉG-hez

TARTALOMJEGYZÉK

A TERMÉK típusa: 0C - 118

megnevezése: 5 1/4" FLOPPY DISK DRIVE(mágneslemezes adatrögzítő)

importálója: PIÉRT

forgalmazója: PIÉRT

MEEI BIZTONSÁGI MINŐSÍTŐ IRAT

száma: 212-01934

kelte: 1987. 11.10.

érvényességi ideje: visszavonásig

MEEI VIZSGÁLAT ALAPJÁUL SZOLGÁLÓ ÁLLAMI SZABVÁNY, HATÓSÁGI ELŐÍRÁS, EGYÉB ELŐÍRÁS: MSZ 91-85; 2/84. BkM-IpM rendelet; MSZ-KGST 1080-78

KIZÁRÓLAG A CS - 118 TÍPUSÚ TÁPEGYSÉGRŐL ÜZEMELTETHETŐ !

A TERMÉK típusa: Cs - 118

megnevezése: TÁPEGYSÉG

gyártója: ÚJ ÉLET MGTSZ, Sárisáp

forgalmazója: PIÉRT

MEEI BIZTONSÁGI MINŐSÍTŐ IRAT

száma: 212-02005

kelte: 1987. 11.24.

érvényességi ideje: visszavonásig

MEEI VIZSGÁLAT ALAPJÁUL SZOLGÁLÓ ÁLLAMI SZABVÁNY, HATÓSÁGI ELŐÍRÁS, EGYÉB ELŐÍRÁS: MSZ 91-85; 2/84. BkM-IpM rendelet; MSZ-KGST 1080-78

KIZÁRÓLAG AZ 0C-118 TÍPUSÚ FLOPPY DISK DRIVE ÜZEMELTETÉSÉRE HASZNÁLHATÓ!

1. OC - 118 típusú FLOPPY DISK DRIVE ismertetése

Az OC - 118 típusú FLOPPY DISK DRIVE (mágneslemezes adatrögzítő) a COMMODORE sorozatú személyi számítógépekhez kifejlesztett és gyártott periféria. Kompatibilis a COMMODORE 64 személyi számítógéppel, közvetlenül helyettesíti a COMMODORE 1541 típusú FLOPPY DISK-et, melynél - különösen adatbetöltési - és írási sebesség, valamint memóriapuffer tekintetében - jobb tulajdonságokkal is rendelkezik.

Programok, adatok mágneslemezen (diszken) történő rögzítésére és futtatására szolgáló készülék. 5 1/4"-os hajlékony diszk befogadására alkalmas, egyetlen mozgó író/olvasó fejet tartalmaz. Az írás/olvasás bytsoros. Egyszerű felépítésű és kezelhetőségű, viszonylag nagy kapacitású tárolóeszköz.

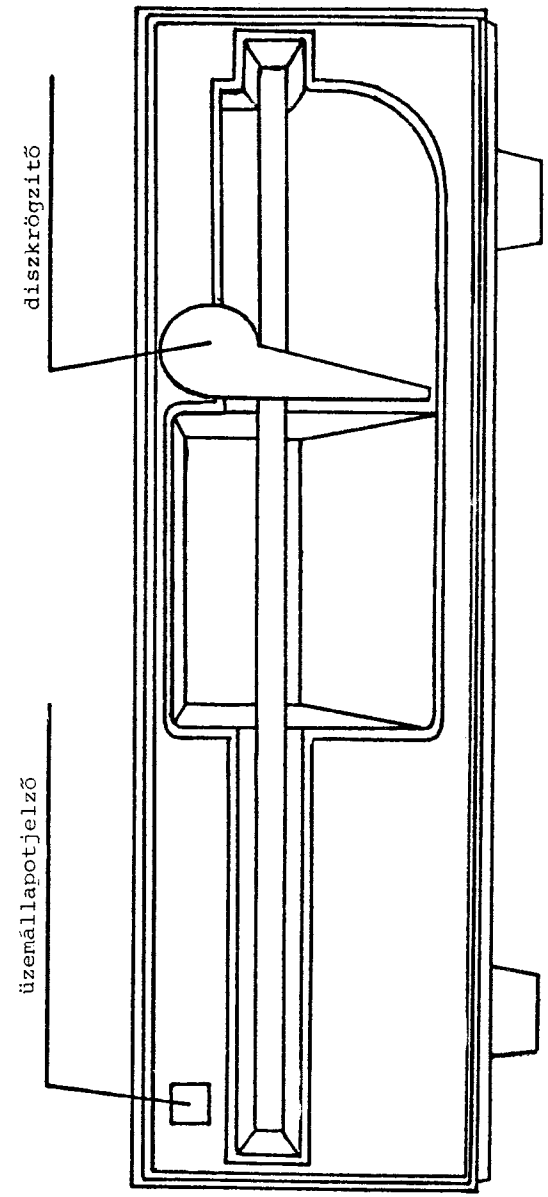
Aluminiumöntvényből készült váz hordozza valamennyi alkatrészt és szerkezeti elemet: a speciális meghajtómotort, a diszk rögzítését és az író/olvasó fej mozgatását szolgáló mechanizmust, az elektronikákat tartalmazó nyáklapokat, valamint a diszk bevezetését is szolgáló, megfelelően kialakított műanyag előlapot. A szerkezet fémlemezből kialakított házban nyert elhelyezést.

Kezelőszervek

A készülék műanyag előlapjának vízszintes nyílása szolgál a diszk bevezetésére. Az előlap bal felső sarkában helyezkedik el az üzemállapotot jelző LED dióda. (1. ábra), melynek jelzései:

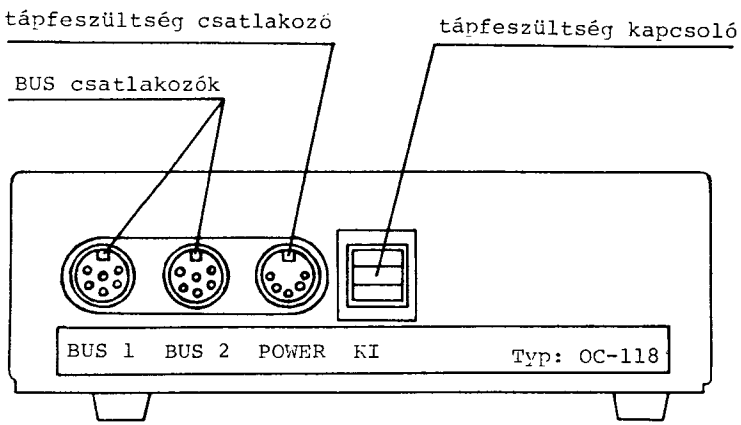
piros szín	-	bekapcsolt állapot
zöld szín	-	üzemkész állapot
villogó	-	hiba

Az előlap jobb oldalán találjuk a diszkrögzítő kart. Használaton kívül, vagy szállítás alkalmával ajánlatos a gyárilag mellékelt író/olvasófejet védő kartonlemezt visszahelyezni és a diszkrögzítő kar függőlegesre történő állításával rögzíteni.



1. ábra

A készülék hátlapján helyezkedik el - balról jobbra - 2 db BUS csatlakozó aljzat, a tápfeszültség csatlakozó aljzat és a tápfeszültség kapcsoló (2. ábra).



2. ábra

A fenéklapon lévő ablak mögötti két DIP kapcsoló a perifériaszám (készülékszám) megváltoztatására szolgál (lásd a 10. fejezetet).

A készülék fenéklapját négy csavar rögzíti, melyek közül egy (esetleg valamennyi) leplombálásra kerül. Megbontásuk a garanciális jogok elvesztését vonja maga után.

Meghibásodás esetén házilagos javitással saját érdekében ne kisérletezzen, a hibás készülék javítását bizza szakszervizre.

Műszaki adatok

Névleges feszültség	5V és 12V DC
teljesítményfelvétel	9W
Érintésvédelmi osztály	III.
Üzemi hőmérséklettartomány	+5 C° --- +40 C°
Mágneslemez méret	Ø 5 1/4"
Perifériaszám (készülékszám)	8
Beállítható perifériaszámok	9 ; 10 ; 11
Tárolóképesség	
lemezenként	174,8 Kbyte
könyvtárbejegyzések	144/lemez
szektor/sáv	17-21
byte/szektor	256
sáv	35
Átlagos működési idő	8000 óra
Befoglaló méretek	150 x 48 x 268 mm
Súly	kb. 2 kp
Szállítási helyzet	közömbös
körülmények	törékeny, nedvességtől óvni
Üzemeltetési helyzet	vízszintes
körülmények	normál
Csomagolás	hungarocell doboz + kartondoboz
Tartozékok	1 db BUS kábel
	1 db felhasználói kézikönyv

KIZÁRÓLAG A CS - 118 TÍPUSÚ TÁPEGYSÉGGEL ÜZEMELTETHETŐ !

2. CS - 118 típusú TÁPEGYSÉG ismertetése

A CS - 118 típusú TÁPEGYSÉG az OC - 118 típusú FLOPPY DISK DRIVE tápfeszültségigényének biztosítására szolgál. Formatervezett, kétrészes, ütésálló polisztirolból készült, szellőzőnyílásokkal ellátott dobozban nyert elhelyezést. Mechanikai felépítése egyszerű, a dobozaljba hat helyen támaszkodó nyák hordozza valamennyi alkatrészét. A nyákra csatlakoznak a be- és kimenő vezetékek s mechanikusan erre rögzített a hálózati kapcsoló, a biztosítótartók és a transzformátor is. A doboz hosszanti végein elhelyezett, lágy műanyagból készült törésgátlók - melyek feladata a csatlakozóvezetékek kis sugarú törésének és ezáltal a huzamos használat során bekövetkező vezetékszakadások megakadályozása - formailag is igazodnak a doboz külső kialakításához. A törésgátlók, valamint a csatlakozóvezetékek kicsúszás elleni rögzítésére a dobozfelek megfelelő végkiképzése szolgál.

A dobozfelek egymáshoz rögzítése önmetsző csavarokkal történik, melyek a gyártás utolsó fázisában leplombálásra kerülnek. Megbontásuk a garanciális jogok elvesztését vonja maga után. Az önmetsző csavarok alkalmazásából következően - mivel a műanyag ellendarabok csak néhány oldást-kötést viselnek el megbízhatóan - a garanciális javításokon és biztosítócseréken túlmenően - nem javasolt gyakori, esetleg indokolatlan megbontásuk.

A készülékben működés közben jelentős hő keletkezik. Ez a hőmennyiség részben a doboz alján és fedelén kialakított szellőzőnyílásokon a kéményhatás következtében átáramló levegővel, részben a felmelegedő burkolat hőleadásával távozik. Káros helyi túlmelegedések elkerülése érdekében kizárólag vízszintes helyzetben és kemény felületen üzemeltethető.

Gondoskodni kell arról is, hogy a szellőzőnyílások szabadon maradjanak.

Egyetlen működtető eleme a jelzőizzóval kombinált hálózati kapcsoló, melynek benyomott helyzete (ilyenkor világít a jelzőizzó) a bekapcsolt állapot.

A tápegység a 220 V-os hálózati feszültségből állítja elő a floppy üzemeltetéséhez szükséges stabilizált tápfeszültségeket. A hálózati csatlakozó utáni

lomha jelleggörbéjű B1 biztosító feladata meghibásodás esetén komolyabb károsodásoktól megvédeni a készüléket. A biztosító kiolvadása hibás alkatrészre, vagy a floppy meghibásodására utal. A K1 hálózati kapcsoló a floppy használatát - s egyben az egész számítógépes rendszerét is - teszi kényelmesebbé, lehetővé téve mind a tápegység, mind a floppy üzemen kívül helyezését a hálózati csatlakozó kihúzása nélkül.

A 220 V-os hálózati feszültségből a kimeneti feszültségeket előállító fokozatok számára a megfelelő változófeszültségeket az EI 66-os lemezekből készült, osztott csévetestű, vákuumimpregnált, biztonsági kivitelű Tr hálózati transzformátor állítja elő. Primer tekercset (1-2) és két szekunder tekercset (3-4 és 5-6) tartalmaz, melyek közül a 3-4 jelű szolgál a hálózati kapcsoló jelzőizzójának táplálására is (3.ábra).

A kimeneti feszültségeket előállító fokozatok azonos elvi felépítésűek, csupán a felhasznált alkatrészek tekintetében - a különböző kimenőfeszültségek és terhelhetőségek miatt - térnek el egymástól.

A 3-4 jelű szekunder tekercs által szolgáltatott kb. 8 V váltakozófeszültséget a D1-D4 diódákból felépitett Graetz híd egyenirányítja, majd a kapott lüktető egyenfeszültséget a C2 pufferkondenzátor szűri. A B2 lomha jelleggörbéjű biztosító szekunderköri védelmi célokat szolgál, kiolvadása hibás alkatrészre, vagy a floppy meghibásodására utal. A fokozat kimeneti egysége az IC1 hárompont stabilizátor. A C3 tantál kondenzátor feladata a stabilizátor nagyfrekvenciás gerjedésének megakadályozása.

Az 5-6 jelű szekunder tekercs által szolgáltatott kb. 14 V váltakozófeszültséget a D5-D8 diódákból felépített Graetz híd egyenirányítja, majd a kapott lüktető egyenfeszültséget a C5 pufferkondenzátor szűri. A fokozat kimeneti egysége az IC2 hárompont stabilizátor. A C6 tantál kondenzátor feladata a stabilizátor nagyfrekvenciás gerjedésének megakadályozása.

A C1 és C4 kondenzátorok szekunder oldali zavarszűrést végeznek.
Az 5 V-os fokozat és a 12 V-os fokozat közösített "-" kimenettel rendelkezik. A kimeneti csatlakozó forrasztásoldali bekötése a 3. ábrán található.

Az IC-k hütőlemezre szereltek, melyek feladata a keletkező, hővé alakuló teljesítmény leadása.

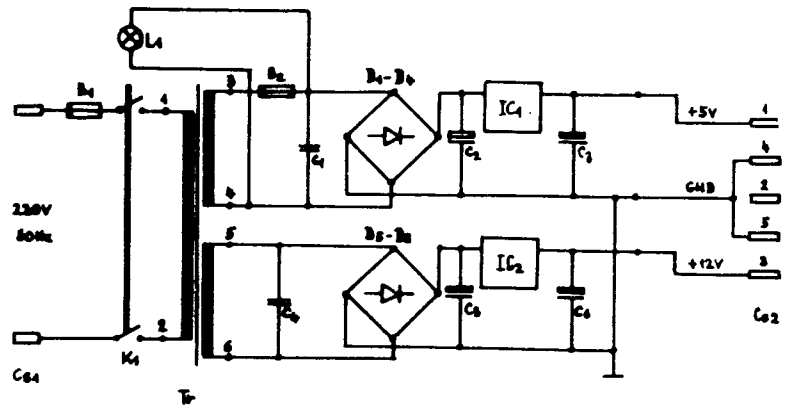
A készülék karbantartást nem igényel.

Műszaki adatok

Névleges feszültség	220 V AC
frekvencia	50 Hz
teljesítményfelvétel	25 VA
Érintésvédelmi osztály	II.
Védettségi fokozat	IP 20
Névleges kimeneti feszültségek	5V DC és 12V DC
terhelhetőségek	600 mA és 500 mA
Üzemi hőmérséklettartomány	+5 C^o --- +40 C^o
Befoglaló méretek	90 x 300 x 70 mm
Súly	kb. 1,2 kp
Szállítási helyzet	közömbös
körülmények	törékeny, nedvességtől óvni
Üzemeltetési helyzet	vízszintes
körülmények	normál
Csomagolás	habszivacs + kartondoboz

KIZÁRÓLAG AZ OC- 118 TÍPUSÚ FLOPPY DISK DRIVE ÜZEMELTETÉSÉRE HASZNÁLHATÓ !

A KÉSZÜLÉK BURKOLATÁNAK ELTÁVOLÍTÁSA ÉLETVESZÉLYES !
SÉRÜLT BURKOLATÚ KÉSZÜLÉKET ÜZEMELTETNI TILOS ÉS ÉLETVESZÉLYES !
BIZTOSÍTÓ- ÉS IZZÓCSERE KIZÁRÓLAG FESZÜLTSÉGMENTES ÁLLAPOTBAN ÉS CSAK SZAKEMBER ÁLTAL VÉGEZHETŐ !
A KÉSZÜLÉK KIMENETÉN LÉVŐ CSATLAKOZÓDUGÓT TILOS BÁRMILYEN HIRADÁSTECHNIKAI KÉSZÜLÉKHEZ (rádió, magnetofon, TV, stb.) CSATLAKOZTATNI (EZ A CSATLAKOZTATOTT KÉSZÜLÉK AZONNALI TÖNKREMENETELÉT OKOZZA) !



3. ábra.

Alkatrészjegyzék:

B1	T 200 mA üvegcsöves
B2	T 1 A üvegcsöves
K1	KN 241-352
Tr	EI 66/1205
L1	12 V ; 40 mA
C1	47 nF ; 40 V
C2	2200 MF ; 25 V
C3	1 MF ; 35 V
C4	47 nF ; 40 V
C5	2200 MF ; 25 V
C6	1 MF ; 35 V
D1-D8	1 N 4001
IC1	7805
IC2	7812
Cs1	Dz 1-22 flexó
Cs2	DKAS 05

3. ÜZEMBEHELYEZÉS

A következő fejezet elolvasása nélkul ne kapcsolja össze a floppyt sem a tápegységgel, sem a számítógéppel, továbbá tartsa be az üzembehelyezés sorrendjét. Az előírástól eltérő összekapcsolás vagy az üzembehelyezés sorrendjének felcserélése zavart idézhet elő a rendszerben és az egyes készülékek meghibásodásához is vezethet.

Helyezze kikapcsolt állásba a tápegység hálózati kapcsolóját, a floppy tápfeszultség kapcsolóját és ellenőrizze a számítógép kikapcsolt állapotát.

Dugaszolja be a tápegység tápfeszultség csatlakozóját a floppy hátlapján lévő tápfeszultség csatlakozó aljzatba (POWER).

Kapcsolja össze a floppyt és a számítógépet a BUS kábellel (a két BUS csatlakozó aljzat közul bármelyik felhasználható). Abban az esetben, ha printerrel vagy másik floppyval is rendelkezik, annak csatlakoztatására a szabadon maradt BUS aljzatot használja. Két floppy egyidejű használata esetén az egyik periféria számát (készulék számát) a 10. fejezetben leírtak szerint meg kell változtatni.

Csatlakoztassa a tápegységet 220 V-os hálózathoz, majd kapcsolja be a hálózati kapcsoló benyomásával (a bekapcsolt állapotot a kapcsolóba beépített izzó jelzi).

Fordítsa a diszkrögzítő kart vízszintes állásba, vegye ki a fejvédő kartont. Csúsztassa óvatosan be a diszket ütközésig, fordítsa vissza a diszkrögzítő kart függőleges állásba (a diszk cimkéje fölfelé, hosszúkás nyílása a floppy felé álljon; ekkor az írásvédő rovátka - kisméretű, négyzetalakú kivágás - baloldalon helyezkedik el).

Kapcsolja be a floppyt a hátlapon lévő tápfeszültség kapcsolóval.

Kapcsolja be a számítógépet.

Üzemen kívül való helyezéskor előzőek fordított sorrendjében járjon el.

Berendezéseinek és programjainak védelme érdekében feltétlenül tartsa szem előtt, hogy

- diszket kivenni vagy behelyezni kizárólag a tápfeszültség kapcsoló kikapcsolt állásában szabad (ekkor a LED nem világít),
- ha a floppyban diszk van és a LED világít, tilos a tápegységet kikapcsolni,
- feszültség alatt lévő készülékeket ne csatlakoztasson egymáshoz és csatlakozásokat se bontson meg,
- a számítógépet mindig utoljára kapcsolja be.

4. PROGRAMOK HASZNÁLATA

Az OC - 118 nagymértékben növeli a számítógéprendszerének hatékonyságát és lehetőségeit programozói szakértelmétől függetlenül.

Kérjuk tartsa szem előtt azt a tényt, hogy ez a kézikönyv az OC - 118 működésének tájékoztató leírása.
Mivel a könyv lépésenkénti utasításokat és a programok indítását tartalmazza, szükséges a BASIC nyelv, a számítógép, valamint a perifériák működtetésére vonatkozó parancsok ismerete.

Ne felejtse el, hogy nem szükséges egyszerre az egész kézikönyvet megtanulnia.

MEGJEGYZÉS: A formátum példákban a kisbetűvel írott szavakat az Ön által választott megfelelő szavakkal vagy számokkal kell helyettesíteni.

A kézikönyv a COMMODORE 64-es gépre vonatkozó utasításokat, illetve parancsokat tartalmazza. Az ettől eltérő COMMODORE tipusú gépekre (C=+4, C=16, C=116, C=128) a 11. fejezetben találhat példákat.

4.1 Programok indítása

Ha a lemezen lévő programot akarja használni, a következőket kell tennie:
- a programot tartalmazó lemezt helyezze be a meghajtóba (lásd 14. oldal),
- billentyűzze be a LOAD "programnév" parancsot,
- nyomja meg a RETURN billentyűt.

A lemez surrogó hangot ad, a képernyőn pedig ez jelenik meg:

SEARCHING FOR PROGRAMNÉV
LOADING
READY

A READY üzenet után:
- írja be a RUN parancsot,
- nyomja meg a RETURN billentyűt.

A program használható.

4.2 A LOAD parancs

Célja: A lemezről a programot a számítógép aktuális memóriájába tölti.
Formátum: LOAD "programnév", készülék # parancs #

A programnév egy karakterstring, azaz egy idézőjelek közé foglalt név, vagy egy adott stringváltozó tartalma.
A készülékszám a meghajtó áramköri lapján előre be van állítva. Értéke 8. Ha egynél több meghajtót használ, olvassa el a készülékszám megváltoztatásáról szóló fejezetet (10. fejezet).
A kézikönyv a 8-as készülékszám használatát feltételezi.
A parancsszám opcionális. Ha nem adja meg, vagy értéke 0, a program szabályosan betöltődik a számítógép BASIC programok számára felhasználható memóriaterületének kezdetére. Ha a szám 1, a program arra a memóriahelyre töltődik be, amelyről származik. Az 1-es parancsszámot főleg gépi nyelvű programok, karakterkészletek és más memóriafüggő alkalmazások esetén kell használni.

PÉLDÁK:
```
LOAD " TEST", 8
LOAD "Program #1", 8
LOAD "Mach Lang", 8,1
LOAD A$, J,K
```

FIGYELMEZTETÉS: A LOAD parancs amellett, hogy a programot a számítógép aktuális memóriájába tölti, kitörli az előzőleg a memóriában tárolt programot!

MEGJEGYZÉS: A stringek, a készülékszámok és a parancsszámok megjelenítésére változók is használhatók. Ezeket előzőleg a programban definiálni kell.

4.3. A lemez katalógusa

A katalógus a lemezen tárolt programok és file-ok listája. 144 könyvtárbejegyzés lehetséges. Egy-egy bejegyzés tartalmazza a file nevét, típusát, a használt blokkok listáját és a kezdő blokkot.

A lemezen 683 blokk található. A meghajtó író/olvasó feje a lemez bármely pontjára elmozdulhat és bármelyik egyedi adatblokkot elérheti.
Egy-egy adatblokk 256 byte információt tartalmaz.
A lemezmeghajtóban található egy DOS (Diszk Operációs Rendszer) nevű program, mely a blokkokat kezeli és azokat egy blokk elérési térképbe (BAM), illetve katalógusba szervezi.
A BAM ellenőrzési lista a blokkokról, amely minden alkalommal frissítésre kerül, valahányszor egy programot kiteszünk a lemezre (SAVE), vagy egy adatfile-t megnyitunk (OPEN).
Akárcsak a BAM, a katalógus is felfrissítésre kerül minden programmentéskor és filenyitáskor.
A BAM nem frissítődik mindaddig, míg a file-t le nem zárjuk. Minden a file-ban tartalmazott adat elvész, ha hibás a file lezárása.
A könyvtár ugyanúgy betölthető (LOAD) a számítógép memóriájába, mint például egy BASIC program.
Helyezze a lemezt a meghajtóba és gépelje be:

LOAD "$", 8

A gép válasza:

SEARCHING FOR $
FOUND $
LOADING
READY

Most a könyvtár az aktuális memóriában van. LIST parancsra a képernyőn is megjelenik.

A könyvtár BASIC programból való használatára vonatkozólag lásd a 6. fejezetet (GET# utasitás).

4.4 Globális karakterek

A betöltés megkönnyítésére lehetőség van arra, hogy bizonyos karaktereket definiáljon. Evvel a módszerrel az első - a megadott mintára illeszkedő - program töltődik be.

PÉLDÁK:	LOAD "*", 8	A lemezen lévő első file-t tölti be.
	LOAD "TE*", 8	Az első TE-vel kezdődő file-t tölti be.
	LOAD "TE??" 8	Az első olyan négybetűs file-t tölti be, amelyik TE-vel kezdődik.
	LOAD "T?NT", 8	Az első négybetűs file-t tölti be. A ? helyén bármilyen betű szerepelhet. Pl.: TINT, TENT stb.

A csillag azt jelenti, hogy a név többi részével nem kell törődni.
A kérdőjel bármilyen karakter helyett állhat.

A fenti módszer a könyvtárnak az aktuális memóriába való töltésekor is használható. Ez lehetővé teszi meghatározott programok listájának a vizsgálatát.
A folyamat egyezik a fentiekkel, kivéve a "$:" hozzáadását.

PÉLDA:	LOAD "$:T?ST*", 8	Az összes olyan filenevet betölti, amely a megfelelő első, harmadik és negyedik betűt tartalmazza.

4.5 SAVE

Célja: Az aktuális memóriában lévő programot átviszi a lemezre.
Formátuma: SAVE "programnév", készülék #, parancs #

Mint az előzőekben, a parancsszám itt is elhagyható.
Ha a lemezen már van ilyen nevű program vagy file, illetve nincs elegendő hely a lemezen, hibajelzés generálódik.
Helyhiány esetében más programokat kell törölni a lemezről, illetve más lemezt kell használni.

PÉLDA: SAVE"HOMEWORK", 8

4.6 SAVE AND REPLACE

Célja: Egy már létező file-t annak javított változatával helyettesít.
Formátuma: SAVE"@O: programnév", 8

Ha egy létező programot szerkeszt, amit el akar menteni ugyanazon név alatt, a SAVE AND REPLACE parancs ezt automatikusan elvégzi. Ha azonban a régi változatot meg kívánja tartani, akkor az új változatot más név alatt mentse ki.

PÉLDA: SAVE"@0:HOMEWORK",8

4.7 VERIFY

Célja: Az aktuális programot összehasonlítja a lemezen lévő programmal.
Formátuma: VERIFY "programnév", készülék #, parancs #

A VERIFY parancs a megadott paraméterekkel, byte-onként összehasonlítja a memóriában lévő programot a lemezen található programmal.

PÉLDA: VERIFY"RÉGI VÁLTOZAT",8

MEGJEGYZÉS A FILENEVEKKEL KAPCSOLATBAN:

A filenevek csak betűvel kezdődhetnek, számmal nem.
Üres karakterek használata megengedett.
A filenevek hosszára nincs korlátozás, míg a parancsok maximálisan 58 karakterből állhatnak.

5. DISZK PARANCSOK

Eddig a lemezmeghajtó használatának egyszerű módját sajátította el. A lemezzel való teljesebb kapcsolat érdekében diszk parancsok használatára van szükség.

5.1 OPEN

Célja: Létrehoz egy file-t a számítógép és a lemezmeghajtó közötti kommunikációs csatorna megnyitásával.
Formátuma: OPEN file #, készülék #, (parancs) csatorna #, szövegsztring

A file száma bármilyen 1 és 127 közé eső érték lehet.
A 128 és 255 közé eső számok is használhatók, de lehetőség szerint el kell kerülni ezeket, mert a PRINT# utasításban a "kocsi vissza" után soremelést okoznak.
A készülék száma általában 8.
A csatornaszám bármilyen 2 és 15 közé eső érték lehet.
A 0 és 1-es csatornákat az operációs rendszer használja, betöltésre és mentésre (LOAD, SAVE).
A 2-14 csatornák használhatók a file-okhoz való adatküldésre.
A 15-ös csatorna foglalt a parancsátvitel számára.

A szövegsztring egy olyan karaktersztring, amely a létrehozott file neveként használható.
File nem hozható létre mindaddig, amíg a filenév nincs definiálva a szövegsztringben.
Ha egy megnyitott file nyitására történik kísérlet, hibaüzenet generálódik. "FILE OPEN ERROR" (file nyitási hiba)

PÉLDÁK:
OPEN 5,8,5, "TEST" Egy TEST nevü file-t hoz létre.
OPEN 15,8,15,"I", Parancsot küld a diszkre a parancscsatornán keresztül.
OPEN A,B,C,Z$ A változókat definiálni kell.

5.2 PRINT#

Célja: Egy megnyitott file-t adattal tölt fel.

Formátuma: PRINT# file#, szövegsztring

Azonos módon működik a PRINT paranccsal, mindössze egy # jellel kell kiegészítenünk annak jelzésére, hogy nem a képernyőre, hanem a lemezjátszóra vonatkozik.

Adatcsatornával használva a PRINT# az információt a lemezmeghajtó pufferébe küldi, ahonnan az a lemezre töltődik. (LOAD)

Parancscsatornával használva a lemezmeghajtónak küld parancsokat. A parancsot idézőjelek közé kell írni.

PÉLDÁK: PRINT# 7,C$ — A 7 számú file-t a C$ szövegsztringgel tölti fel.

PRINT# 15,"I" — A parancs-csatornán keresztül diszk parancsot küld.

5.3 INITIALIZE

Célja: a lemezmeghajtót a bekapcsolás utáni állapotnak megfelelő alaphelyzetbe állítja.

Formátuma: OPEN 15,8,15, "I" vagy
OPEN 15,8,15: PRINT#15, "I"

Előfordulhat, hogy egy hibafelvétel a lemezen megakadályozza a további műveletek végrehajtását.

Az INITIALIZE visszaállítja a lemezmeghajtót alapállapotba.

5.4 NEW

Célja: A lemez formátumozása.

Formátuma: PRINT# 15, "NEW O: disknév, id#"

Ez a parancs törli a lemez teljes tartalmát. Használható új lemez formátumozásához, de egy már formátumozott lemez törlésére is.

A disknév a felhasználó kényelmét szolgálja.

Az id# egy kétjegyű alfanumerikus azonosító, mely bekerül a könyvtárba és a lemez minden blokkjába. Ha adatíráskor lemezcsere történik, a meghajtó felismeri ezt az id# ellenőrzése révén.

Ha csak a lemez törlésére - és nem az újra formátumozására - van szükség, az id# azonosítót el kell hagyni.

A NEW parancs időzítő- és blokkjelzőket helyez el, valamint létrehozza a könyvtárat és a BAM-ot.

PÉLDÁK: OPEN 15,8,15, "NEW O: TEST DISK, A1"
OPEN 15,8,15: PRINT#15, "N O: MY DISK, MY"
OPEN 15,8,15, "N O: NEW INFO" (törléskor id# nélkül)

5.5 SCRATCH

Célja: File törlése a lemezről.

Formátuma: PRINT# 15, "SCRATCH O: filenév"

A parancs egy vagy több file-t töröl a lemezről, helyet adva így új, vagy hosszabb file-oknak.

Egyidőben több file is törölhető.

PÉLDÁK: PRINT#15,"S O: TEXT" A TEXT nevű file-t törli.
PRINT#15,"SCRATCH O: TEXT, O: TEST, O: MUSIC"
Törli a TEXT, TEST, MUSIC nevű file-okat.

5.6 COPY

Célja: Másolatot készit a file-ról.

Formátuma: PRINT#15, "COPY O: uj filenév=0: régi filenév"

A COPY segítségével a lemezen tárolt programról, vagy file-ról másolat készíthető.

Az új filenév nem lehet azonos a régivel.

Egy új file-ba egyidejűleg több - maximálisan négy - file-t lehet másolni.

PÉLDÁK: PRINT#15,"C O: BACKUP=O : ORIGINAL"
PRINT#15,"COPY O:NEWFILE=O: OLD1,O: OLD2,0"
(Az OLD1 és OLD2 file-okat a NEWFILE-ba másolja)

5.7. RENAME

Célja: Egy file névének megváltoztatása.
Formátuma: PRINT#15, "RENAME0: új név=0: réginév"

A parancs segítségével egy, már a lemez könyvtárában lévő file neve változtatható meg.
A RENAME az éppen megnyitott file-okra nem használható.

PÉLDA: PRINT#15, "R O: GOODNAME=O: DUMBNAME"

5.8 VALIDATE

Célja: az értéktelen területeket eltávolítja a lemezről.
Formátuma: OPEN 15,8,15,"VO:"

Ha egy lemezen sok filementés és törlés történt, akkor az adatok között hézagok keletkeznek.

A VALIDATE parancs újraszervezi a lemezt, így a szabad területek ismét hasznossá válnak, a maximális tárolókapacitás érhető el.
Ez a parancs a megnyitott, de szabályosan le nem zárt file-okat is eltávolítja.

FIGYELMEZTETÉS! A VALIDATE parancs törli a tetszőleges elérésű file-okat (lásd 7. fejezet). Ha a lemezen ilyen file-ok vannak, ne használja ezt a parancsot.

5.9 A hibacsatorna olvasása

A DOS (Disk Operation System) nélkül nem lehetséges a hibacsatorna olvasása, mivel az INPUT# parancsot kell használni, amely programon kívül nem alkalmazható.
Ez az egyszerű program szolgál a hibacsatorna olvasására:

```
10 OPEN 15,8,15
20 INPUT#15, A$, B$, C$, D$
30 PRINT A$, B$, C$, D$
```


Az INPUT# használatakor maximálisan négy - a hibafelvételt leíró - változót olvas be a parancs-csatornáról. Az első, a harmadik és a negyedik helyen numerikus változó használható.

A bevitel szervezése:

Első: a hiba száma (a 0 jelentése: nincs hiba)
Második: a hiba leírása
Harmadik: a hiba előfordulási sávja
Negyedik: a sáv blokkja (szektora), melyben a hiba előfordult. A 18. sávon előforduló hibák a BAM és a könyvtár hibái.

5.10 CLOSE

Célja: az adatállomány lezárására szolgál.
Formátuma: CLOSE file#

Ha egy megnyitott file-ba nem kíván több adatot felvinni, akkor azt LE KELL ZÁRNI, különben minden adat elvész a file-ból.
Nagyon fontos, hogy az adatfile-ok a hibacsatorna (15) lezárása előtt le legyenek zárva. Ellenkező esetben a lemezmeghajtó zárja le ezeket, de a BASIC program még nyitottnak feltételezi.
A hibacsatornát elsőként kell megnyitni és csak az összes file lezárása után szabad lezárni.

MEGJEGYZÉS: Ha egy BASIC program hibaállapotra vezet, a BASIC-ben minden file lezáródik.
Nem így a lemezmeghajtóban. Ez nagyon VESZÉLYES!
Ilyenkor azonnal
CLOSE 15: OPEN 15,8,15 ; CLOSE 15
a teendő, mely újra inicializálja a meghajtót és elmenti az összes file-t.

6. SZEKVENCIÁLIS FILE-OK

A szekvenciális file-ok tárolása folyamatosan történik. Alapvetően három különböző típusú szekvenciális file használható:

- PRG Program file. Az egyetlen olyan szekvenciális file, mely programokat képes tárolni és olvasni.
- SEQ Szekvenciális file. Adatkezelés céljából.
- USR User file. Adatkezelés céljából.

6.1 OPEN

Célja: Szekvenciális file megnyitása.

Formátuma: OPEN file#, készülék#, csatorna#, "O: név, tipus, irány"

A filenév ugyanaz, mint az OPEN előző használatánál.

A készülék száma általában 8.

A csatorna száma 2-14 (adatcsatorna).

A könnyebb megjegyezhetőség érdekében érdemes a file és a csatorna számát azonosra választani.

A név a filenév, melyre WRITE (írás) file létrehozásakor nem használható globális karakter.

A típus a következők valamelyike:

- PRG programfile
- SEQ szekvenciális file
- USR user (felhasználói) file
- REL relativ file (a BASIC 2.0-ban nincs értelmezve)

Az irány READ (olvasás), vagy WRITE (írás).

Kezdőbetű használata elegendő.

PÉLDÁK:
```
OPEN 5,8,5, "O: DATA, S, R"
OPEN A,B,C, "O:TEXT, P,W"
OPEN A,B,C,"O:" +A$+ "U,W"
```
Az A$ stringváltozóval meghatározott WRITE file-t nyitja meg.
```
OPEN 2,8,2 "@O: PHONES, S, W"
```
A file régi változatát felülírja az újjal.

Ha egy file-t írásra, vagy olvasásra megnyitottak, az adatátvitelre három parancs használható:

PRINT#, INPUT#, GET#.

6.2 PRINT#

Célja: Adatkivitel egy megnyitott file-ba.

Formátuma: PRINT# file#, adatlista

A PRINT# utasitás ugyanúgy működik, mint a PRINT.

A fileszám az éppen megnyitott file száma.

Az adatlista változókat és/vagy idézőjelek közti szöveget tartalmaz.

Az egyszerűbb olvashatóság érdekében az adatok írását a következők figyelembevételével kell végezni:

- ha a tételeket vesszők választják el egymástól, a lemezen ezek helyére szóközök kerülnek,
- pontosvessző alkalmazása esetén a szóközök nem tárolódnak el,
- ha sem vessző, sem pontosvessző nincs az utasításban, a beírt adatok leírása egy "kocsi vissza" (CR) karakter lesz.

Mintaprogram:
```
10 A$="THIS IS A"
20 B$="TEST"
30 OPEN 8,8,8, "O:TEST ,S,W"
40 PRINT#8, A$,B$"OF THE DISK"
50 CLOSE8
60 END
```

A következő kép alakulna ki, ha láthatnánk az adatokat és azok elhelyezkedését a lemezen:

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
T H I S   I S   A
24  25 26 27 28  29 30 31 32 33 34 35
      T H E     D I S K CR eof   (end of file)
```

A vessző, a pontosvessző és a CR a lemezen tárolva speciális jelentéssel bír. Sztingen belül, vagy idézőjelek között használva szabályos karakterek-

ként kerülnek tárolásra. Mezők közötti elválasztóként a vessző hatására szóközök szúródnak be (általában memóriaveszteség), pontosvessző hatástalan marad, a CR hatására "kocsi vissza" tárolódik a lemezen.
Ezek fontos tudnivalók, ha a tárolt adatokat a GET# és az INPUT# parancsokkal akarja visszanyerni.

6.3 GET#

Célja: Byte-onkénti adatelérés a lemezről.
Formátuma: GET# file#, változólista

A GET# utasitás akkor hasznos, ha az aktuális adattartalom, vagy a struktúra nem ismert.
Például, ha a lemezen egy file megsérült.
Ha a file ismert és ép, az INPUT# hatásosabb.

Az adat byte-onként érkezik, CR-rel, vesszővel és más elválasztó karakterekkel együtt a GET# hatására.

A hibaüzenetek elkerülése érdekében biztonságosabb a karaktersztring változók használata.
A következő mintaprogram kiolvassa a file tartalmát (ez esetben a PRINT# utasítással létrehozott file-t).

```
10 OPEN 8,8,8, "TEST"
20 GET#8,A$:PRINT A$;
30 IF ST=0 THEN 20              (ST= státuszjel)
40 CLOSE 8
50 END
```

6.4 INPUT#

Célja: Lemezen tárolt adatok elérése (csoportokban)
Formátum: INPUT# file#, változó

A fileszám ugyanaz, mint a megnyitott file száma.
A változó karaktersztringeket vagy számokat jelenthet.

Egy adatcsoport olvasásához a csoport kezdetét és végét jelző elválasztó karakterekre van szükség.
Ezek a vessző, a pontosvessző és a CR, működésük azonos a PRINT# parancsnál látottakkal. A számok előtt egy szóköz is tárolásra kerül, mely pozitív számok esetén üres marad, negatív számok esetén negatív előjelet tartalmaz.

Mintaprogram:

```
10 OPEN 8,8,8, " O: DATAFILE, S.W"
20 FOR A=1 TO 10
30 PRINT#8,A
40 NEXT A
50 CLOSE 8
60 OPEN 2,8,2, "DATAFILE"
70 INPUT#2,B : PRINT B
80 IF ST=0 THEN 70
90 CLOSE 2
100 END
```

A program az 1-től 10-ig terjedő számokat a DATAFILE nevű szekvenciális file-ba írja.
A 70-es és a 80-as sorok beolvassák az adatokat a lemezről és kinyomtatják azt.

7. RELATIV FILE-OK

A relativ file-ok a lemezen található bármely adatot elérhetik - akárcsak a tetszőleges elérésű file-ok -, azonban nem kell a file-okat a saját programban kezelni.

A DOS kezeli az adatokat és figyeli a file-ok státuszát. A relativ file-ok elérése ezért lassúbb, de a kezelés kényelme kárpótol az időveszteségért.

A DOS nyomon követi a használt sávokat és szektorokat (blokkokat) és a rekordok közötti átfedést is megengedi egyik blokkból a következőbe. Ez mellékszektorok létrehozásával történik, amelyekben egy sormutató jelzi minden rekord kezdetét.
Egy file-ban 6 mellékszektor lehet. Ezek mindegyike legfeljebb 120 rekordra mutathat. Ezt alapul véve egy file 720 rekordot tartalmazhat. Mivel minden rekord 254 karakter hosszúságú lehet, egy file a teljes lemezt betölthetí. A blokk formátum az első két byte-ot tartalmazza, melyek a következő adatblokk sávját és szektorát határozzák meg. A következő 254 byte-en találhatók az aktuális adatok. Egy üres rekord első byte-ja FF (a csupa 1-es bitet tartalmazó byte hexadecimális megfelelője), a rekord ezután csak 00 byte-okat tartalmaz.

A mellékszektorok az összes mellékszektor elhelyezkedésére utalnak, nemcsak az adott mellékszektorral kapcsolatos adatblokk helyére.

7.1 A relativ file formátuma

BYTE	ADATBLOKK:	DEFINICIO
0,1		A következő adatblokk sávja és szektora.
2-256		254 adatbyte. Az üres rekord első byte-ja FF (csupa bináris 1-es), ezután a rekord végéig 00-k következnek. A részben feltöltött rekordok nullákkal (00) vannak feltöltve.

BYTE	MELLÉKSZEKTOR BLOKK:	DEFINÍCIÓ
0,1		A következő mellékszektor blokk sávja és szektora.
2		A mellékszektor száma. (0-5)
3		Rekordhossz.
4,5		Az első mellékszektor (0) sávja és szektora.
6,7		A második mellékszektor (1) sávja és szektora.
8,9		A harmadik mellékszektor (2) sávja és szektora.
10,11		A negyedik mellékszektor (3) sávja és szektora.
12,13		Az ötödik mellékszektor (4) sávja és szektora.
14,15		A hatodik mellékszektor (5) sávja és szektora.
15-256		120 adatblokk sáv- és szektor mutatói.

7.2 Relativ file-ok használata

A relativ file létrehozása a megnyitáskor (OPEN) történik. Lezárásig (CLOSE) ugyanez a file van használatban. Egy relativ file csak a SCRATCH paranccsal, vagy a lemez újraformátumozásával törölhető.
A SAVE parancsban használt "@" (SAVE and REPLACE) relativ file-okra nem alkalmazható.

A relativ file létrehozásának formátuma:

OPEN file #, készülék #, csatorna #, "O:név, L," +CHR$ (rl#) (rekordhossz)

PÉLDÁK: OPEN 2, 8, 2, "O:FILE, L"+CHR$ (100)
(rekordhossz 100)
OPEN F, 8, F, "O:" +A$+ ", L, "+CHR$ (O)

Létező relativ file megnyitásának formátuma:

OPEN file #, készülék #, csatorna #, "O:név"

PÉLDA: OPEN 2, 8, 6, "O: TEST"

Ebben az esetben a DOS megállapítja a szintaxisból, hogy relativ file-ról van szó.
Mindkét formátum alkalmazható a file-ok írására, vagy olvasására.

Minden más művelet előtt a file mutatót a megfelelő rekordra kell állítani!

7.3 POSITION

Célja: A file mutatót egy meghatározott rekordra állítja.
Formátuma: PRINT# file#, "P" CHR$ (csatorna#) CHR$ (rec# 10)
CHR$ (rec# hi) CHR$ (record pozíció)
Mivel 720 használható record van, két byte-ra van szükség a pozíció meghatározásához. (Egy byte 256)

A rec# 10 a cím alsó, a rec#hi a felső értéket tartalmazza.
A rec# = rec#hi * 256 + rec#10.
A rec# az a hely a rekordban, ahol az adatátvitel kezdődik.

PÉLDÁK: PRINT#15, "P"CHR$ (2) CHR$ (1) CHR$ (O)
PRINT#15, "p"CHRS (CH) CHR$ (R1) CHR$(R2)CHR$(P)

A következő mintaprogram egy relativ file-t hoz létre:

```
10 OPEN 15, 8, 15
20 OPEN 8, 8, 8, "O:TEST, L," + CHR$ (50)
30 PRINT#15, "p"CHR$(8)CHR$(O)CHR$(4)CHR$(1)
40 PRINT#8, CHR$(255)
50 CLOSE8: CLOSE15
```

Ez a program a TEST nevű relativ file-t hozza létre, mely 50 byte hosszú rekordokat tartalmaz.
A 30-as sor az 1024-es rekordban az első helyre mozgatja a mutatót.
(rec# = 256 * 4 + 0 = 1024)

A POINTER parancs a parancs-csatornán, az adatok az adatcsatornán kerülnek elküldésre. Az adatcsatorna a 8-as.
Mivel a rekord már nem létezik, hibaüzenet fog figyelmeztetni, hogy ne használjuk a GET#, vagy az INPUT# parancsokat.
Ha egy relativ file már létezik, megnyitható, bővíthető, vagy adatátvitel céljából elérhető.
A file ugyan bővíthető, de a rekordhossz nem változtatható meg.
A file bővítésére az is elegendő, ha az előző mintaprogram 30-as sorában nagyobb a rekordszám.

Létező relativ file-ba a következőképpen írhatók adatok:

```
 10 OPEN 15, 8, 15
 20 OPEN 2, 8, 6, "O:TEST"
 30 GOSUB 1000
 40 IF A=100 THEN STOP
 50 PRINT#15, "P"CHR$(6)CHR$(100)CHR$(O)CHR$(1)
 60 GOSUB 1000
 70 IF A=50 THEN PRINT#2, 1:GOTO 50
 80 IF A=100 THENSTOP
 90 PRINT#2, "123456789"
100 PRINT#15, "P"CHR$(6)CHR$(100)CHR$(O)CHR$(20)
110 PRINT#2, "JOHN QWERTY"
120 CLOSE2:CLOSE15
130 END
1000 INPUT#15, A, A$, B$, C$
1010 IF (A=50) OR (A  20) THEN RETURN
1020 PRINT "FATAL ERROR:";
1030 PRINT A, A$, B$, C$
1040 A=100:RETURN
```

A 10-es és a 20-as sorok megnyitják a parancs és az adatcsatornát.
A 30-as és a 40-es sor hibavizsgálatot végez.
Az 50-es sor a 100. rekordra mozgatja a file mutatót.

Mivel még egy rekord sem létezik, hibaüzenet generálódik.
A 60-as, 70-es és 80-as sorok hibaellenőrzést végeznek és létrehoznak 100 rekordot.
A 90-es sor 9 adatbyte-ot ír a 100-as rekord első 9 poziciójára.
A 110-es sor kinyomtatja az azon a helyen talált nevet.

Fontos, hogy az adatok szekvenciálisan íródjanak a rekordba, hogy a már ott lévő adatok ne rongálódjanak.
Ez a program visszaolvassa azokat az adatokat, amelyeket az előző program helyezett el a file-ban.

```
 10 OPEN 15,8,15
 20 OPEN 2,8,6, "O:TEST"
 30 GOSUB 1000
 40 IF A=100 THEN STOP
 50 PRINT#15,"P"CHR$(6)CHR$(100)CHR$(O)CHR$(1)
 60 GOSUB 1000
 70 IF A=50 THEN PRINT A$
 80 IF A=100 THEN STOP
 90 INPUT#2,D$: PRINT D$
100 PRINT#15,"P"CHR$(6)CHR$(100)CHR$(O)CHR$(20)
110 PRINT#2,E$: PRINT E$
120 CLOSE 2:CLOSE15
130 END
1000 INPUT#15,A,A$,B$,C$
1010 IF (A=50) OR (A(20)THEN RETURN
1020 PRINT "FATAL ERRCR"
1030 PRINT A,A$,B$,C$
1040 A=100: RETURN
```

A 90-es, 100-as és 110-es sorok beolvassák a rekordot, a tartalmát megjelenítik a képernyőn.
Az írási rutinban minden PRINT# utasítás után a lemezre küldött CR karakter a rekord valamennyi mezőjének elválasztó karaktereként működik.

Ha a file-t szekvenciálisan kell írni vagy olvasni, nincs szükség a mutató minden rekordhoz való beállítására.
A rekord mutató automatikusan az 1-es pozíción kezdődik, ha nincs más kijelölve.
A mezők írásakor ill. olvasásakor a mutató végighalad a rekordon.

8. TETSZŐLEGES ELÉRÉSŰ FILE-OK

Folytonos adatáramlás esetén a szekvenciális file-okkal igen kellemes dolgozni, néhány feladat azonban több rugalmasságot kíván.
Például, ha egy hosszú levelezési listát végig kellene nézni egy adott személy címének a megállapításához, az igen kellemetlen volna.
A tetszőleges elérési módszer kiválasztja a keresett adatot anélkül, hogy a teljes file-t végig kellene olvasni.
Kétféle file tipus van, mellyel elvégezhető ez a feladat.

1. Tetszőleges elérésü file
2. Relativ file

Ahol fontos a sebesség (gépi nyelvü programoknál), ott a tetszőleges elérésű file-okkal érdemes dolgozni. Itt az adatok elhelyezését a program kezeli.

Relativ file esetében a DOS végzi ezt.

Mivel a tetszőleges elérésű file-okat nem a DOS kezeli, ezért véletlenül is elmozdíthatók a lemezről.
A tetszőleges elérésű file-ok a lemez egy meghatározott helyére írt file-ok.
A lemez 35 koncentrikus körre - sávra - van felosztva úgy, hogy mindegyik sáv 17-21 szektort tartalmaz.

Sávok száma	Szektor tartomány	Szektorok száma
1 - 17	0 - 20	21
18 - 24	0 - 18	19
25 - 30	0 - 17	18
31 - 35	0 - 16	17

A lemez bármelyik blokkjára lehet írni illetve olvasni róla.
Meghatározható, hogy mely blokkok legyenek használhatóak.

8.1 OPEN

Célja: Tetszőleges elérésű adatcsatornát nyit meg.
Formátuma: OPEN file#, készülék#, csatorna#, "#"

Tetszőleges elérésű file-ok használatakor két csatornát kell megnyitni a lemezre:

1. a parancs-csatornát (15), melyen a parancsok küldhetők,
2. az adatcsatornát (2-14), mely az adatátvitelre szolgál.

A tetszőleges elérésű file-ok adatcsatornája a "#" jel, mint file-név kiválasztásával történik.
A parancs végén látható "#" egy 256 byte hosszuságú puffert jelöl ki a lemezen a kívánt adatblokk kezelésére.
A pufferek száma specifikálható.

PÉLDÁK: OPEN 5, 8, 5, "#" Nincs puffer kijelölés.
OPEN A, B, C, "#" Kijelölt puffer a 2-es.

8.2 BLOCK-READ

Célja: Meghatározott adatblokk leolvasása a lemezről.
Formátuma: PRINT# file#, "BLOCK-READ:" csatorna #, meghajtó #, sáv #, blokk # (A BLOCK-READ helyett B-R is állhat)

A file és a csatorna száma megegyezik a megnyitáskor szereplők számával.
A sáv és a blokk száma jelöli ki, hogy melyik 256 byte-os blokkot kell olvasni.
A parancs hatására a lemezegység a meghatározott adatblokkot a pufferterületre továbbítja.
Ezután az adat az INPUT#, vagy a GET#, paranccsal olvasható ki.
Csak a kijelölt blokk adatai kerülnek olvasásra, ebből is csak a ténylegesen használt byte-ok.

A mintaprogram a BLOCK-READ hatására kiolvassa az 5-ös sáv 9-es blokkját és a blokk tartalmát kiírja a képernyőre.

```
10 OPEN 15, 8, 15
20 OPEN 8, 8, 8, "#"
30 PRINT#15, "B-R:"8, 0, 5, 9
40 GET#8, A$
```

```
50 PRINT A$;
60 IF ST=0 THEN 40
70 PRINT "READ COMPLETE"
80 CLOSE 8 : CLOSE 15
```

8.3 BLOCK-WRITE

Célja: Egy adatblokkot a lemez meghatározott blokkjába ír.

Formátuma: PRINT#file#"BLOCK-WRITE:" meghajtó#, csatorna#, sáv#, blokk#

Rövidítése a B-W lehet. A parancs hatására az előzőleg a pufferben tárolt adatok a lemez meghatározott helyére íródnak.

A BLOCK-WRITE kiadása előtt az adatok a PRINT segítségével küldhetők a pufferbe az adatcsatornán keresztül.

A DOS ismeri a pufferben tárolt byte-ok számát.

A BLOCK-WRITE végrehajtásakor a byte-számot a blokk első byte-jába írja. Ez azt jelenti, hogy a blokk ténylegesen csak 255 byte-ot tartalmaz, mivel a blokk első byte-ja a byteszám részére foglalt.

A következő példa rutinja az előző mintaprogram BLOCK-READ-jével beolvasott blokkot írja ki.

```
10 OPEN 15,8,15
20 OPEN 8,8,8,"#"
30 FOR A=1 TO 32
40 PRINT#8, "TESTING"
50 NEXT
60 PRINT#15,"B-W:" 8,0,5,9
70 CLOSE 8: CLOSE 15
```

8.4 BLOCK-ALLOCATE

Célja: Meghatározza, hogy egy bizonyos blokk szabad-e és ha igen, azt felhasználásra kijelöli.

Formátuma: PRINT#15,"B-A:" csatorna #, meghajtó #, sáv #, blokk #

Mint korábban már említettük, a BLOCK-READ és a BLOCK-WRITE utasításoknál a DOS nem kezeli a lemezt.

A felhasználó a BLOCK-ALLOCATE utasítás használatával győződhet meg egy bizonyos blokk elérhetőségéről.

Ez lehetővé teszi a BLOCK utasítások használatát a file-okat tartalmazó lemezen.

A BAM vizsgálatával az utasítás meghatározza, hogy a kijelölt blokkot használták-e már.

A BAM mindig frissítődik, valahányszor a lemezen egy file rögzítésre kerül.

A BLOCK utasítások nem frissítik a BAM-ot - így nincsenek is elfogadva - míg egy BLOCK-ALLOCATE végrehajtásra nem került.

FIGYELMEZTETÉS: A VALIDATE parancs nem ismeri fel a tetszőleges elérésű file-okat, így ilyen file-okat tartalmazó lemezen nem használható.

Ha a BLOCK-ALLOCATE megállapítja, hogy a kijelölt blokk már használt, akkor hibajelzés (65) generálódik.

A hibaüzenet közli a legközelebbi szabad sáv és blokk számát a lemezen.

Ez a blokk azonban nem lesz használatra kijelölve, így a BLOCK-ALLOCATE utasítást újra ki kell adni.

A következő program lefoglal egy blokkot és ír bele.

Ha a blokk már használt, a következő - a hibaüzenetben jelzett - szabad blokkba ír.

```
10 OPEN 15,8,15:OPEN 8,8,8,"#"
20 PRINT#8,"THIS GOES INTO THE BUFFER"
30 T=5 : S=9
40 PRINT#15,"B-A: "O,T,S
50 INPUT#15,A,A$,B,C
60 IF A=65 THEN T=B:S=C: GOTO 40
70 PRINT#15,"B-W:" 8,0,T,S
80 PRINT" DATA WAS STORED IN TRACK: "T," SECTOR:"S
```

```
 90 CLOSE 8:CLOSE 15
100 END
```

A 20-as sor szöveggel tölti fel a puffert.
A 30-ás a 40-es sor ellenőrzi az 5-ös sáv 9-es blokkját, hogy szabad-e.
Az 50-es sor beolvassa a hibajeleket. Ha a blokk szabad, az adat ide kerül. Ha foglalt, a 60-as sor veszi az új sáv- és blokkszámot és lefoglalja az általuk meghatározott blokkot. Az adat az új blokkba kerül.
A 70-es és a 80-as sor a számítógépbe olvassa a sáv és blokkszámot, majd kiírja azokat a képernyőre.

8.5 BLOCK-FREE

Célja: Egy használt blokk felszabadítása.
Formátuma: OPEN 15,8,15, "B-F:"meghajtó #, sáv #, blokk #

Ez a parancs a BLOCK-ALLOCATE ellenkezője, amennyiben a rendszer számára felszabadítja a használni nem kívánt blokkot.
Egy kevéssé hasonlít a SCRATCH parancsra, mivel ténylegesen nem töröl semmit, csak felszabadítja a bejegyzést.
(Jelen esetben a BAM-ban.)

PÉLDA:
```
10 OPEN 8,8, "#"
20 OPEN 15,8,15, "B-F:"0,5,9
30 CLOSE 8 : CLOSE 15
```

Az 5-ös sáv 9-es blokkját szabadítja fel használatra.

8.6 BUFFER-POINTER

Célja: Blokkon belüli tetszőleges hozzáférés engedélyezése.
Formátuma: PRINT#15,"B-P:" csatorna #, helyi(byte#)

A puffer mutató ismeri azt a helyet, ahová az utolsó adat beíródott. Oda mutat, ahová a következő adat beolvasásra kerül. Ha a puffer mutató helyét megváltoztatjuk a pufferben, egy blokkon belüli egyedi byte-ok tetszőlegesen elérhetővé válnak.
Ez azt jelenti, hogy egy blokkot rekordokra lehet osztani.

PÉLDA: PRINT#15, "B-P:" 5,64 A mutatót a pufferben a 64. karakterre állítja.

8.7 Tetszőleges elérésű file-ok használata

A tetszőleges elérésű file-ok alkalmazásakor nem tudhatjuk, hogy melyek a használt blokkok. Ezek nyomonkövetésére a legegyszerűbb módszer az, ha a file-al párhuzamosan létrehozunk egy szekvenciális file-t is, mely a rekordok, a sávok és a blokkok elhelyezkedésének a listáját tartalmazza.
Ez azt jelenti, hogy minden véletlen elérésű file-hoz három csatornát kell megnyitni.

1. Parancs-csatornát
2. Tetszőleges elérésű adatok csatornáját
3. Szekvenciális file csatornáját

Ezzel egyidőben két puffert is kell használni.
A következő négy program a blokkon belüli tetszőleges elérést mutatja be.

Az A program 10 tetszőleges elérésű blokkot ír a szekvenciális file-ba.
A B program visszaolvassa ezt a file-t.
A C program 10 - egyenként négy rekordot tartalmazó - tetszőleges elérésű blokkot ír.
A D program visszaolvassa ezt a file-t.

A program: Szekvenciális file írása

```
 10 OPEN 15,8,15
 20 OPEN 5,8,5,"#"
 30 OPEN 4,8,4,"@0:KEYS, S, W"
 40 A$="Record CONTENS #"
 50 FOR R = 1 TO 10
 60 PRINT#5, A$",R
 70 T=1:S=1
 80 PRINT#15,"B-A:"0,T,S
 90 INPUT#15,A,B$,C,D
100 IF A=65 THEN T=C:S=D: GOTO 80
```

```
110 PRINT#15,"B-W:"5,0,T,S
120 PRINT#4,T","S
130 NEXT R
140 CLOSE 4:CLOSE 5: CLOSE 15
```

B program: Szekvenciális file olvasása

```
 10 OPEN 15,8,15
 20 OPEN 5,8,5,"#
 30 OPEN 4,8,4,"KEYS,S.R"
 40 FOR R=1 TO 10
 50 INPUT#4,T,S
 60 PRINT#15,"B-R:"5,0,T,S
 70 INPUT#5,A$,X
 80 IF AS      "Record Contens # OR X <> R THEN STOP
 90 PRINT#15,"B-F:"0,T,S
100 NEXT R
110 CLOSE 4: CLOSE  5
120 PRINT#15,"SO: KEYS"
130 CLOSE
```

C program: Tetszőleges elérésű file írása

```
 10 OPEN 15,8,15
 20 OPEN 5,8,5,"#"
 30 OPEN 4,8,4,"KEYS,S,W"
 40 A$="Record Contens #"
 50 FOR R=1 TO 10
 60 FOR L=1 TO 4
 70 PRINT#15,"B-P:"5: (L-1)*64
 80 PRINT#5,  A$","L
 90 NEXT L
100 T=1 : S=1
110 PRINT#15,"B-A:",0,T,S
```



```
120 INPUT#15,A,B$,C,D
130 IF A=65 THEN T=C:S=D:GOTO 110
140 PRINT#15,"B-W:"5,0,T,S
150 PRINT#/4,T",  "S
160 NEXT R
170 CLOSE 4:  CLOSE 5: CLOSE 15
```

D program: Tetszőleges elérésű file olvasása

```
 10 OPEN15,8,15
 20 OPEN 5,8,5,"#
 30 OPEN 4,8,4,"KEYS,S,R"
 40 FOR R=1 TO 10
 50 INPUT#4,T,S
 60 PRINT#15,"B-R:"5,0,T,S
 70 FOR L=1 TO 4
 80 PRINT#15,"B-P:"5,(L-1)*64
 90 INPUT#5,A$,X
100 IF A$     "Record Contens #" OR X=L THEN STOP
110 NEXT L
120 PRINT#15,"B-F:"0,T,S
130 NEXT R
140 CLOSE 4: CLOSE 5
150 PRINT#15,"SO:KEYS"
160 CLOSE 15
```

8.8 USER1

Célja: Egy teljes - 256 byte-os - blokkot olvas a lemezről a pufferba.

Formátuma: PRINT# file#, "U1:" csatorna #, meghajtó #, sáv #, blokk #

Az USER1 parancs majdnem azonos a BLOCK-READ paranccsal, de ez a puffer mutatót az olvasandó blokk végére állítja, így az egész blokk olvasásra kerül.

A USER1 rövidítése U1 és UA is lehet.

Mintaprogram:

Beolvassa az 5-ös sáv 9-es blokkját (a teljes 256 byte-ot) és megjeleníti a képernyőn.

```
10 OPEN 15,8,15:OPEN 8,8,8
20 PRINT#15, "U1:=8,0,5,9
30 GET#, A$:PRINT A$;
40 IF ST=0 THEN 30
50 CLOSE 8: CLOSE 15
60 END
```

8,9 USER2

Célja: A puffer mutató megváltoztatása nélkül egy adatblokkot ír a képernyőre.

Formátuma: PRINT#15, "U2:" csatorna#, készülék#, sáv# blokk#

A USER2 parancs hasonló a BLOCK-WRITE parancshoz.
A puffer tartalma a lemezre való írásakor azonban nem változtatja meg a puffer mutató helyét. Ez akkor hasznos, ha egy adatblokkot akar a pufferba olvasni és azt módosítani kívánja.
A USER2 parancs az adat újbóli lemezre való írására használható, ha a puffer mutatónak megfelelő adatot módosítottuk. Ekkor a puffer mutató a megfelelő helyen marad.
Ezt a BLOCK-WRITE használatakor nekünk kell beállítani.
A USER2 rövidítése U2 és UB is lehet.

Mintaprogram:

```
10 OPEN 15,8,15:OPEN 8,8,8
20 PRINT#15,"U1"8,0,5,9
30 PRINT#15,"B-P:"8,32
40 PRINT#8,"A"
50 PRINT#15,"U2:"8,0,5,9
60 CLOSE  8: CLOSE 15
70 END
```


A program a USER1 és a USER2 parancsokat használja.
A 20-as sor az 5-ös sáv 9-es blokkját olvassa a pufferba.
A 30-as sor a puffer mutatót a 32-es byte-ra állítja.
A 40-es sor a 32-es byte-ot az "A" karakterre változtatja.
Az 50-es sor a puffert visszaírja a lemezre.
Bár a puffer mutató megváltozott, a USER2 parancs biztosítja, hogy a lemez régi puffer mutatója ne változzon.

9. A DISK VEZÉRLŐ PROGRAMOZÁSA

Az FSD-1 saját mikroprocessort és memóriát tartalmazó periféria. Gyakorlott programozó számára nem gond a mikroprocessor és a memória kezelése, mely számos alkalmazási lehetőséget kínál.
Olyan rutinokat lehet tervezni, melyek a lemezmemóriában helyezkednek el és a mikroprocessor segítségével vezérlik a lemezmeghajtó működését. A lemezről származó DOS programok is hozzáadhatók.
A lemezmeghajtó 16K ROM-ot és 2K RAM-ot tartalmaz.
A programozó számára a leghasznosabb terület a 40000H és 5FFFH között elhelyezkedő puffer RAM.(A H a hexadecimális szám jelölése). Erre a területre gépinyelv szintű utasítások írhatók, melyeket a lemezvezérlő (mikroprocessor) végrehajt.
A memóriával kapcsolatos adatátvitel kezelésére a MEMORY parancsok szolgálnak.

Három alapvető MEMORY parancs létezik, melyek a USER parancsokkal egészülnek ki.

9.1 MEMORY-WRITE

Célja: Adatbyte átvitele a meghajtó memóriájába. (Max.34)
Formátuma: PRINT#15,"M-W"CHR$ (cim alsó byte-ja) CHR$ (cím felső byte-ja) CHR$ (# számú karakter) CHR$ (adat)

A MEMORY-WRITE parancs segítségével 34 adatbyte írható egyidőben a lemezvezérlő memóriájába.
A MEMORY-EXECUTE és a USER parancsok használhatók e kód futtatására.
Az alsó és felső byte-ok a hexadecimális cím megfelelői az aktuális memóriaterületen.
A byte-ok száma az átvitelre szánt byte-ok száma (decimálisan), maximális értéke 34.
Az adat a küldeni kívánt, hexadecimálisan kódolt utasítás decimális megjelenítése.

PÉLDA:

```
10 OPEN 15, 8, 15
20 PRINT#15, "M-W"CHR$ (o) CHR$ (112) CHR$ (3) CHR$ (169)
   CHR$ (8) CHR$ (96)
30 CLOSE 15
```

Ez a rutin 3 byte-ot ír a 7000H, 7001H és a 7002H helyekre. (256 * 112 + + 0 = 28672 = 7000H)

A három byte: 169 (A9H, a PAGE ZÉRO 0-s lap utasítás),
8 (8H, pozíció),
96 (60H, RETURN utasítás).

Végrehajtáskor a vezérlő akkumulátor-regisztere feltöltődik a 0008H pozíció tartalmával és a vezérlés visszakerül a lemezvezérlőhöz.

9.2 MEMORY-READ

Célja: Adatok beolvasása a meghajtó memóriájából.
Formátuma: PRINT#15 file #, "M-R" CHR$ (cím alsó byte-ja) CHR$ (cím felső byte-ja)

A MEMORY-READ parancs a cím alsó és felső byte-jai által a disk meghajtó memóriájában kijelölt byte-ot választja ki olvasásra.
A 15-ös csatornáról a GET# paranccsal olvasott következő byte, a kijelölt memóriahelyről származik.
Ezt mutatja be a következő példa, melyben az FFOOH és FFOAH (decimálisan 65280 és 65290) között elhelyezkedő, egymást követő 10 byte olvasása történik.

PÉLDA:

```
10 OPEN 15, 8, 15
20 FOR A = 1 TO 10
30 PRINT#15, "M-R"CHR$(A)CHR$(255)
40 GET#15, A$:PRINT ASC(A$ + CHR$(0)) ;
50 NEXT
60 CLOSED 15
```

MEMORY-READ esetén a hibacsatornán az INPUT# használata sajátos eredményre vezet.

A megoldás az, hogy a MEMORY parancs kivételével ilyenkor bármely más parancsot lehet használni.

Disk vezérlő memóriáját olvasó program:

```
10 OPEN 15,8,15
20 INPUT"LOCATION PLEASE"; A
30 A1 = INT (A/256) : A2 = A-A1*256
40 PRINT#15,+M-R"CHR$(A2)CHR$(A1)
50 FOR L = 1 TO 5
60 GET#15, A$
70 PRINT ASC(A$ + CHR$(0))
80 NEXT
90 INPUT"CONTINUE" ; A$
100 IF LEFT$ (A$,1)="Y" THEN 50
110 GOTO 20
```

9.3 MEMORY-EXECUTE

Célja: Program-végrehajtás a diszk memóriában.

Formátuma: PRINT#15 file #, "M-E"CHR$ (Cim alsó byte-ja) CHR$ (cím felső byte-ja)

Ha egy program betöltődött a diszk memóriába, legyen az akár a 16K ROM, akár a 2K RAM, a MEMORY-EXECUTE parancs címe meghatározza a program-végrehajtás kezdetét.

A parancs használata megköveteli, hogy a végrehajtandó program RTS utasítással záródjon, azaz a vezérlés visszakerüljön a DOS-nak.

A következő rutin egy RTS (Return from Subroutine) utasítást ír:

```
10 OPEN 15,8,15,"M-W"CHR$(0)CHR$(5); 1; CHR$(96
20 PRINT#15,"M-E"CHR$(0)CHR$(19): REM JUMPS TP BYTE? RETURNS
30 CLOSE  15
```


9.4 USER parancsok

A 8. fejezetben tárgyalt USER1 és USER2 parancsok mellett olyan USER parancsok is vannak, melyek a diszk meghajtó pufferének meghatározott helyére való ugrást okoznak.

Ez lehetővé teszi hosszabb rutinok írását, melyek a diszk meghajtó memóriájában működhetnek - egy ugrási táblával együtt - akár a BASIC-ben is.

USER PARANCS	FUNKCIÓ
U1 vagy UA	BLOCK-READ a puffer mutató megváltoztatása nélkül
U2 vagy UB	BLOCK-WRITE a puffer mutató megváltoztatása nélkül
U3 vagy UC	ugrás a 0500H címre
U4 vagy UD	ugrás a 0503H címre
U5 vagy UE	ugrás a 0506H címre
U6 vagy UF	ugrás a 0509H címre
U7 vagy UG	ugrás a 050CH címre
U8 vagy UH	ugrás a 050FH címre
U9 vagy UI	ugrás az FFFAH címre
U; vagy UJ	feszültség bekapcsolási vektor
UI+	Commodore 64 sebességének beállítása
U-	VIC 20 sebességének beállítása

PÉLDÁK:

```
PRINT#15,  "U3"
PRINT#15,"U"+CHR$ 50+Q
PRINT#15,"UI"
```

10. A KÉSZÜLÉKSZÁM MEGVÁLTOZTATÁSA

Mindegyik perifériának rendelkeznie kell egy készülékszámmal, hogy a számítógép azonosítani tudja, melyikkel akarunk adatátvitelt kezdeményezni.
Az OC-118 előre - hardver uton - beállított készülékszáma: 8, a meghajtó száma: 0.
A diszk a saját készülékszámát ismeri, egy hardver átkötés vizsgálata alapján - mely az áramköri lapon található - a RAM egy meghatározott területére írja ezt a számot.

A készülékszám két módszerrel, szoftver és hardver úton is megváltoztatható.
Hardver módszerrel akkor célszerű, ha várhatóan állandóan két vagy több meghajtó van használatban. Ilyenkor ez a módszer egyszerűbb és állandó megoldást jelent.

Szoftver módszerrel olyankor érdemes, ha időleges a két meghajtó használata. Az egyik lemezmeghajtó készülékszámát kell megváltoztatni.

10.1. Szoftver módszer

A készülékszám a 0077 H és 0078H helyekre vonatkozó MEMORY-WRITE utasítás végrehajtásakor változik meg.
A parancs akkor hajtódik végre, ha a parancs-csatorna meg van nyitva.

Formátuma: PRINT# file #, "M-W"CHR$(119)CHR$(0)CHR$(2)
CHR$ (cím + 32) CHR$ (cím + 64)

A cím az új készülék száma.

A következő példa a készülékszám 9-re való megváltoztatását mutatja be.

```
10 OPEN 15, 8, 15
20 PRINT#15, "M-W"CHR$(119)CHR$(119)CHR$(0)CHR$(2)CHR$(9+32)
   CHR$(9+64)
30 CLOSE 15
```

Először csak az egyik meghajtót kell bekapcsolni. A készülékszám megváltoztatása után a következőt és így tovább.

10.2 Hardver módszer

Az FSD-1 készülék számának megváltoztatása nagyon egyszerű. A meghajtó alján található DIP kapcsolókat kell a kívánt készülékszámnak megfelelően beállítani.

1. Kikapcsolni a lemezmeghajtót.
2. Megkeresni a DIP kapcsolót a meghajtó alján.
3. Kapcsolókat a kívánt készülékszámnak megfelelő kombinációba állítani. (Az "ON" állás a meghajtó hátlapja felé van)

Készülékszám:	8	9	10	11
1-es kapcsoló:	ON	OFF	ON	OFF
2-es kapcsoló:	ON	ON	OFF	OFF

4. A diszk meghajtó az új készülékszámmal használható.

11. A LEMEZEGYSÉG HASZNÁLATA C=16, C=116, C+4, C=128 GÉPEKEN

11.1 Hibacsatorna olvasása

Ellentétben a C=64-essel, ezeknél a gépeknél nem kell külön programot írni a hibacsatorna lekérdezéséhez. Itt ugyanis két BASIC változó, a DS és a DS$ tartalmazza a hibacsatorna üzenetét. A DS tartalmazza szám formában a hiba kódját, a DS$ pedig a teljes üzenetet string formában adja vissza.

PÉLDA:

```
10 OPEN 4,8,4, =kakukk"
20 IF SD > =20 THEN PRINT DS$: END
30 (a program többi része)
```

A program elején megnyitjuk az adatfile-unkat. Ha a file nem létezik, vagy más hiba történik, akkor kiíratjuk, hogy mi ez a hiba, egyébként folytatjuk a program végrehajtását. Ha DS 0 és 19 közé esik nincs hiba, csak ha 20-nál nagyobb.

11.2 Lemez formátumozása

A HEADER BASIC utasítás segítségével végezhető el.

PÉLDA: HEADER "PROGRAMOK", DO, Ixx

Készítünk egy új lemezt "PROGRAMOK" névvel. (A 8-as lemezegység 0-ás meghajtóján.)
Ixx kötelező, ha teljesen új a lemez, amit használunk.

11.3 FILE-OK TÖRLÉSE

Egyetlen BASIC utasítás van a file törlésére is, a SCRATCH.

PÉLDA: SCRATCH "PROBA*" Törli az összes PROBA...val kezdődő file-t.

11.4 A lemez újjászervezése

Ha egy lemezt már sokat használtunk, akkor az ugyanahhoz a file-hoz tartozó blokkok össze-vissza helyezkednek el a lemezen és lehetnek szabályosan le nem zárt file-ok, stb. Ekkor a lemezt a COLLECT BASIC utasítással szervezhetjük újjá.

11.5 File-ok átnevezése

Lehetőség van a file-ok nevének megváltoztatására is a BASIC-ből. Ez az utasítás a RENAME.

PÉLDA: RENAME "PROBA13" TO "VÉGLEGES"

A PRóBA13 nevű file nevét megváltoztattuk a VÉGLEGES file névre.

11.6 Lemezkatalógus listázása

A C=64-el szemben nem kell a meglévő programunkat törölni, ha meg akarjuk nézni a lemez tartalmát.
Ezeknél a gépeknél ugyanis a DIRECTORY utasítás a képernyőre listázza a lemez tartalomjegyzékét.
Az F3-as funkcióbillentyű végzi el.

11.7 Programok tárolása

Van közvetlen a lemezre vonatkozó programtárolási utasítás: DSAVE.

PÉLDA: DSAVE "program"

Elmenti a "program" név alatt az aktuális BASIC programot.
Ha már létezik a "program" nevű file és felül akarjuk írni, akkor használjuk ezt az alakját az utasításnak:

```
DSAVE "program"
```

Gépikódú programokat MONITOR üzemmódban, az S parancs segítségével tárolhatjuk a lemezen:

```
S "program" 08 0000 D000
```

Elmenti a memória 0000-től D000-ig terjedő tartalmát a 8-as lemezegységre.

```
S "@: program" 08 00000 D000
```

Itt a felülírásnál kell a kettőspont.

11.8. Programok beolvasása

Közvetlen betölthetők a programok a lemezről a DLOAD utasítással.

PÉLDA: DLOAD "program"

A program nevű file-t betölti a lemezről, mint BASIC programot.
A gépi kódú programok a LOAD "program", 8,1 utasítással tölthetők be, vagy a MONITOR üzemmód alatt az L "program"08 monitorparancs segítségével.

KIVÉTEL: C=128-as, ahol a BLOAD BASIC utasítás is használható. Ugyanitt használhatjuk a BSAVE parancsot gépi kódú programok kimentésére.

11.9 Programok ellenőrzése

A programok ellenőrzése ugyanúgy történik, mint a C=64-esnél a VERIFY paranccsal. Itt is kivétel a C=128-as, ahol alkalmazható a DVERIFY utasítás.

11.10 Relativ file-ok használata

C=128-as esetén van egy könnyítés, a RECORD BASIC utasítás.

PÉLDA:
```
10 OPEN 4,8,4, "retyezat"
20 RECORD#4,100,9
```

Megnyitjuk a már létező relativ file-t, majd pozícionáljuk a 100-adik record 9-edik byte-jára, ahonnan a program kezdi olvasni az adatokat.

"A" FÜGGELÉK

A parancsok listája

"B" FÜGGELÉK

A hibaüzenetek leírása

Valahányszor hibaüzenet generálódik, az OC-118 előlapján lévő LED villogni kezd.
A lemezmeghajtó csak kérésre küld hibaüzenetet a számítógépbe.

A következő rutin beolvassa a hibaüzenetet, majd megjeleníti azt a képernyőn.

```
10 OPEN 15,8,15
20 INPUT# 15,A,A$,B$,C$
30 PRINT A,A$,B$,C$
40 CLOSE 15
50 END
```

Hibaüzenetek listája és magyarázata

0: NINCS HIBA

1 Nem hibajelzés. A hibacsatorna olvasásakor fordul elő, a LED nem villog.

1: FILE-OK TÖRÖLVE
Nem hibaüzenet. Egy vagy több file törlése után a hibacsatorna olvasása eredményezi. A törölt file-ok száma is megjelenik.

2-19: NEM HASZNÁLT HIBAÜZENETEK

20: OLVASÁSI HIBA (A BLOKK FEJLÉC HIÁNYZIK)
A diszk vezérlő nem találja a keresett blokk fejlécét. Ennek oka egy hibás fejléc a lemezen, vagy egy nem létező szektorszám.

21: OLVASÁSI HIBA (A SZINKRON KARAKTER HIÁNYZIK)
A diszk vezérlő nem talál szinkronjelet a keresett sávon. Oka lehet: Író/olvasó fej helytelen állása, vagy a lemez nincs a meghajtóban, nincs formátumozva, illetve hibásan van behelyezve.
Hardver hibát is jelezhet.

22: OLVASÁSI HIBA (AZ ADATBLOKK HIÁNYZIK)
Hibásan felírt adatblokk olvasását, vagy ellenőrzését kérelmezte a diszk vezérlő. Ez a hibaüzenet a BLOCK parancsokkal kapcsolatosan fordul elő. Meg nem engedett sáv és/vagy szektorhívást jelez.

23: OLVASÁSI HIBA (ELLENŐRZŐ ÖSSZEG-HIBA AZ ADATBLOKKBAN)
Ez az üzenet azt jelzi, hogy egy vagy több adatbyte hibás. Az adat bekerült ugyan a DOS memóriába, de az összegellenőrzés hibát jelzett. Földelési problémára is mutathat.

24: OLVASÁSI HIBA (BYTE DEKODOLÁSI HIBA)
Az adat, vagy a fejléc beolvasásra került a DOS memóriába, de az adatbyte egy nem megengedett bitmintája következtében hardver hiba keletkezett.
A földelési problémára is mutathat.

25: ÍRÁSI HIBA (ÍRÁSI-ELLENŐRZÉSI HIBA)
Az üzenet akkor generálódik, ha a vezérlő eltérést fedez fel a felírt és a DOS memóriában lévő adatok között.

26: ÍRÁSVÉDELEM
A vezérlőhöz adatblokk-írási kérelem érkezett, mialatt az írásvédelem kapcsolója le volt nyomva. Általában ezt az okozza, hogy a lemezt írásvédő szalaggal ellátva akarták használni.

27: OLVASÁSI HIBA (ELLENŐRZŐ ÖSSZEG-HIBA A FEJLÉCBEN)
A keresett adatblokk fejléce hibás. A blokk nem került be a DOS memóriába.
Földelési problémára is mutathat.

28: ÍRÁSI HIBA (HOSSZÚ ADATBLOKK)
Egy adatblokk írása után a vezérlő a következő fejléc szinkronjelét keresi. A hibaüzenet akkor keletkezik, ha a szinkronjel felismerése egy előre meghatározott időn belül nem történik meg.

29: HIBÁS DISZK AZONOSÍTÓ
A vezérlőhöz nem inicializált, vagy hibás fejlécű lemez elérésére vonatkozó kérés érkezett. Akkor is jelentkezik, ha adatátvitel közben lemezcsere történt.

30: SZINTAKTIKAI HIBA (ÁLTALÁNOS SZINTAKSZIS)
A DOS nem tudja értelmezni a parancs-csatornán érkezett utasítást. Általában hibás filenév, vagy helytelenül használt minták okozzák.

31: SZINTAKTIKAI HIBA (ÉRVÉNYTELEN PARANCS)
A DOS nem ismeri fel a parancsot. A parancsnak az első pozíción kell kezdődnie.

32: SZINTAKTIKAI HIBA (HOSSZÚ SOR)
Az elküldött parancs 58 karakternél hosszabb.

33: SZINTAKTIKAI HIBA (ÉRVÉNYTELEN FILENÉV)
Az OPEN vagy a SAVE parancsban hibás volt a minta megadása.

34: SZINTAKTIKAI HIBA (HIÁNYZÓ FILENÉV)
A filenév kimaradt az utasításból, vagy a DOS nem ismerte fel. Általában a kettőspont hiányzik.

35-38: NEM HASZNÁLT

39: SZINTAKTIKAI HIBA (ÉRVÉNYTELEN PARANCS)
Akkor keletkezik, ha a parancs-csatornának küldött utasítást a DOS nem ismeri fel.

40-49: NEM HASZNÁLT

50: A RECORD HIÁNYZIK
Input vagy GET parancsokkal kisérlet történt az utolsó record utáni olvasásra a lemezen. Az üzenet akkor is előfordul, ha a relativ-file-ban a file vége utáni recordra pozícionálunk. Ha új record felvitelével bővíteni akarjuk a file-t (PRINT,) a hibajelzés figyelmen kivül hagyható. Az INPUT és GET utasítások e hiba után történő használata előtt újrapozícionálás szükséges.

51: TÚLCSORDULÁS A RECORDBAN
A PRINT utasitás túlnyúlik a record határon. Ez pedig információ vesztéssel jár, mivel a recordot lezáró karakterként kiküldött CR beleszámít a recordméretbe. Ez a hibaüzenet akkor fordul elő, ha a recordban lévő összes karakterek száma - beleértve a lezáró CR-t is - meghaladja a definiált értéket.

52: A FILE TÚL NAGY
Egy relativ file-on belüli recordpozíció jelzi, hogy lemeztúlcsordulás következik be.

53-59: NEM HASZNÁLT

60 AZ ÍRHATÓ FILE NINCS
Egy le nem zárt írható file-t megnyitottak olvasásra.

61: A FILE NINCS MEGNYITVA
Az elérni kivánt file a DOS-ban nincs megnyitva.
Néha ebben a helyzetben nem keletkezik hibaüzenet, a kérés egyszerűen nem teljesül.

62: A FILE HIÁNYZIK
A keresett file nincs az adott meghajtóban.

63: A FILE LÉTEZIK
A létrehozni kivánt filenév már szerepel a lemezen.

64: HIBÁS FILETÍPUS
A file típusa nem felel meg a keresett file könyvtári bejegyzésének.

65: NINCS BLOKK
Akkor jelentkezik, ha a lefoglalni kívánt blokk már foglalt. A paraméterek a legközelebbi szabad sávot a szektort jelzik. Ha a paraméterek értéke 0, az összes magasabb számú blokk foglalt.

66: ÉRVÉNYTELEN SÁV ÉS SZEKTOR
A DOS olyan sáv vagy szektor elérését kísérelte meg, mely a használt formátumban nem létezik. A következő blokk mutatójának olvasási problémáját is jelezheti.

67: ÉRVÉNYTELEN RENDSZERSÁV VAGY SZEKTOR
Ez a speciális üzenet érvénytelen rendszersávot vagy szektort jelöl.

68-69: NEM HASZNÁLT

70: NINCS SZABAD CSATORNA
A kért csatorna nem elérhető, vagy minden csatorna foglalt. A DOS-ban egyszerre legfeljebb 5 szekvenciális file lehet nyitva. A közvetlen elérésű csatornáknál 6 nyitott file lehetséges.

71: KÖNYVTÁR HIBA
A BAM nincs összhangban a belső számmal. Hiba adódott a BAM elhelyezésében, vagy felülíródott a DOS memóriában. A hiba a lemez újrainicializálásával javítható, ekkor a BAM visszaáll a memóriába. A javítás során néhány aktiv file lezáródhat.

72: A LEMEZ MEGTELT
A lemez blokkjai teljes felhasználásra kerültek, vagy a könyvtárbejegyzések száma elérte a 144-et.

73: DOS HIBA
A DOS1 és a DOS2 olvasáskor kompatibilis, íráskor azonban nem. A kétféle DOS különbözőképpen olvashat lemezeket, se az egyik változattal formátumozott lemezre - éppen az eltérő formátum miatt - a másik verzióval nem lehet írni. Ez a hibaüzenet jelenik meg mindannyiszor, ha írási kisérlet történik egy nem kompatibilis formátumú lemezre. Az üzenet előfordulhat a gép bekapcsolásakor is.

74: A MEGHAJTÓ NEM ÜZEMKÉSZ
Kisérlet történt a lemezmeghajtó elérésére olyankor, amikor abban nem volt lemez.

SZERVÍZELLÁTÁS

A készülék és a hálózati adapter zavartalan működéséért 1 év jótállást vállalunk.

Garanciális és azontúli javítást az

ISKOLA-SZÁMÍTÓGÉP SZERVÍZ

végzi.

Cím: 1077 Budapest VII. Baross tér 19.
Telefon: 428-999
Nyitvatartás: kedd és csütörtök 8^{30}-16^{30}-ig.

MINŐSÉGI BIZONYÍTVÁNY

Tanusítjuk, hogy az OC-118 típusú Floppy és a hozzátartozó CS-118 típusú tápegység a közölt jellemzőknek és a szabványoknak megfelel.

PIÉRT KERESKEDELMI VÁLLALAT

Készült a PIÉRT Vállalat
megbízása alapján
rota eljárással, A/4-es alakban
3050 pld-ban
Felelős vezető: Kepler László
BTI OTO Nytsz: 88.064