COMMODORE

DISC DRIVE
VC 1541

Návod k použití

Informace pro uživatele

Upozornění: Bylo ověřeno, že toto zařízení vyhovuje omezením pro výpočetní techniku třídy B, ve smyslu části J v oddílu 15 pravidel FCC. Pouze periferní zařízení (vstupní a výstupní zařízení počítače, terminály, tiskárny atd.) které průkazně vyhovují omezením třídy B smí býti připojeny k tomuto počítači. Práce s neověřenými periferními zařízeními bude pravděpodobně rušit příjem radiového a televizního signálu.

Toto zařízení, generuje a užívá energii o radiové frekvenci a pokud není řádně instalováno, tj. důsledně podle pokynů výrobce, může způsobovat poruchy radiového a televizního příjmu. Zařízení bylo jako takové typově ověřováno a bylo shledáno, že vyhovuje omezením třídy B části J předpisů FCC, které jsou vytvořeny tak, aby zajišťovaly rozumnou ochranu proti takovému rušení svou instalací na místě. Nelze ovšem zaručit, že při konkrétní instalaci nemůže k takovému rušení dojít. Pokud toto zařízení skutečně ruší televizní a radiový příjem, což můžete určit zapnutím a vypnutím tohoto zařízení, doporučujeme uživateli, aby se pokusil odrušit jedním nebo vícero opatřeními:

- přemístěte přijímací anténu
- jinak umístěte počítač vzhledem k přijímači
- zapněte počítač jinam do sítě tak, aby přijímač a počítač byly v jiné větvi sítě.
- změňte vzájemné postavení počítače s příslušenstvím

Pokud je to nutné, uživatel by měl záležitost konzultovat se zkušeným radiovým či televizním technikem.

Informace obsažené v manuálu by měli být zcela spolehlivé. Není předpokládána ovšem žádná zodpovědnost z případné nepřesnosti. Materiál v manuálu je pouze informací a může být kdykoliv změněn bez upozornění.

OBSAH

1 Všeobecný popis - úvod

Vítejte mezi uživatele nejrychlejšího, nejsnáze užívaného a nejefektivnějšího systému pro vytváření souborů (filů) na vašem počítači COMMODORE -Disketové jednotky 1541-II.Tento manuál byl sepsán, aby vám ukázal, jak dostat co nejvíce ze svého drajvu, ať už jste začátečník nebo pokročilý profesionál.

Pokud jste začátečník, tak vám prvních několik kapitol pomůže se základními pracemi, instalováním a operováním na disku. Až se zlepší vaše zručnost a znalosti programování, najdete více druhů použití pro vaši disketovou jednotku a další pokročilejší kapitoly se pro vás stanou mnohem cennější.

Pokud jste profesionál, tak vám tato instrukční knížka ukáže, jak projít všemi fázemi, abyste dosáhl všechny práce disketové jednotky, na které pomyslíte.

Nezáleží na úrovni vašich zkušeností, vaše disketová jednotka 1541 dramaticky zlepší všeobecné schopnosti vašeho počítačového systému.

Než se dostanete k detailní práci s 1541, měl byste si uvědomit několik důležitých věcí. Tento manuál je návodem pro uživatele, takže pokud se váš problém - hledaná informace netýká přímo disku, nebo disketové jednotky, budete muset použít svého manuálu uživatele COMMODORE 64 (VIC 20, Plus 4, COMMODORE 16), anebo příručky programátora, pokud chcete informaci o programování. Navíc, i když vám podáváme instrukce pro práci krok za krokem, měli byste se dobře seznámit s BASICem a instrukcemi (zvanými příkazy), které vám umožňují práce na vašich disketách a disketových jednotkách. Pokud ovšem chcete používat svoji disketovou jednotku pouze pro ukládání a loadování uceleného software, vložili jsme snadný a krátký úsek právě pro tuto práci.

Tak a teď dál s všeobecnou informací.

Příkazy pro diskovou jednotku jsou dávány v několika úrovních. Počínaje kapitolou 3, se naučíte jak účinkují příkazy,

které umožňují uchovávat a nahrávat programy na a z disku. Kapitola 4 vás naučí jak se příkazy na disk posílají a seznámí vás s příkazy pro provoz disku.

Kapitola 5 vám řekne, jak pracovat se sekvenčními fily. Jsou velmi podobné se svými obdobami na pásce, ale jsou mnohem rychlejší. Kapitola 6 přináší příkazy, které umožňují pracovat s fily s přímým (libovolným) vstupem, jak najít libovolné údaje na disku a dále jak organizovat disketu na stopy a bloky. Kapitola 7 popisuje zvláštní - relativní fily. Relativní fily jsou nejlepší metoda ukládání databází, zejména když jsou používány ve spojení se sekvenčními fily.

Kapitola 8 popisuje metody pro programování obvodů diskového řadiče na úrovni strojového kódu. A konečně závěrečná kapitola vám ukáže, jak lze změnit číslo diskové jednotky (zásahem do hardware nebo programově).

Nezapomeňte, že se skutečně není třeba naučit celý obsah této knížky najednou. První 4 kapitoly postačí k tomu, abyste začali pracovat a další dvě postačí pro naprostou většinu operací.

Poznání vaší diskové jednotky vás odmění v mnoha směrech - rychlost práce, spolehlivost a mnohem větší pružnost ve vašich datových aplikacích.

Specifikace

Diskový drive vám umožní uložit až 144 různých programů, nebo datových filů na jediné pružné minidisketě pro maximum 174 k paměti pro datové zpracování.

Do jednotky jsou vloženy obvody jak pro řadič disku, tak pro celý operační systém disku, vcelku 16 k RAM (čtecí) paměti a 2 k RAM (čtení i zápis - libovolný vstup). Tyto obvody činí z vaší diskové jednotky COMMODORE 1541-II "inteligentní" zařízení. To znamená, že zajišťuje svůj provoz, aniž by odebírala jakoukoliv paměť z vašeho počítače COMMODORE. Disk používá systému programově nezávislé linky. Závislá linka umožňuje disku zpracovávat příkazy, zatímco počítač sám již

řeší jiné úkoly. To výrazně zlepšuje celkovou průchodnost (vstup a výstup) systému. Diskety, které vytvoříte v tomto diskovém drivu (jednotce) lze číst a psát na diskových drivech COMMODORE 4040 a 2031 (kompatibilní). Proto lze diskety používat střídavě na kterémkoliv z těchto systémů. Navíc tato jednotka je schopna číst programy vytvořené na starších drivech COMMODORE 2040.

Disk 1541-II má dvojitý interface pro seriový bus. Tento bus byl vytvořen COMMODOREM. Signály tohoto busu připomínají paralelní IEEE - 488 interface používaný na počítačích COMMODORE PET až na to, že pro přenos dat se využívá jednoho drátu místo 8. Dvě přípojky na zadní straně umožňují, aby více než 1 zařízení sdílelo seriový bus současně. Je to vytvořeno propojením zařízení dohromady, při čemž každé je propojeno na další. Až 4 diskové drivy a dvě tiskárny mohou současně sdílet jeden bus.

Disková jednotka VC 1541-II komunikuje s počítači COMMODORE 64 a VIC 20 pomocí příkazů BASICu 2, s počítači COMMODORE 16 a Plus 4 s pomocí příkazů BASICu 3.5.

Specifikace pro VIC 1541-II
jednoduchý drive pružného disku

Paměť:

Celková kapacita	174848 bytů na disketu
Sekvenčně	168656 bytů na disketu
Relativně	167132 bytů na disketu
	65535 recordů (vět) na file
Záznamů directoria	144 na disketu
Sektorů na stopu	17 až 21
Bytů na sektor	256
Stop	35
Bloků	683 (664 volných)

Integrované obvody:

6502	microprocesor
6522 VIA 2ks	vstup, výstup, vnitřní časování
16k ROM	Diskový operační systém (DOS)
Buffer 6116	2k RAM

Rozměry:

výška	77 mm
šířka	184 mm
hloubka	256 mm

Požadavky na napájení:

napětí	220 a 240 V střídavých
frekvence	50 HZ
příkon	25 W

Medium:

diskety standartní mini 5 1/4 palce jednostraná s normální hustotou zápisu

2 Vybalení a připojení

Obsah krabice

Spolu s diskovou jednotkou 1541 byste měli najít šedou šňůru pro připojení na síť, černou šňůru pro seriový bus, uživatelský manuál a demonstrační disketu.
Síťová šňůra má konektor pro zadní stranu diskové jednotky na jedné straně a zástrčku do sítě na straně druhé. Kabel pro seriový bus má oba konce stejné. Je to 6-ti kolíková zásrtrčka (DIN), která připojuje počítače COMMODORE nebo další disketové drivy.
Prosím nedělejte nic, pokud jste nedokončili následující úsek.

Obr. 1 originálu
Door lever - západka dvířek
Drive indikater - indikátor drivu
Light: active/flash: error - Světlo: v činosti/blikání: chyba

Připojovací kabely

Vašim prvním krokem bude vzít správný kabel a zastrčit ho zezadu do přístroje. Nepodaří se vám zastrčit zástrčku obráceně. Jak je zastrčena do jednotky, můžete druhý konec zastrčit do sítě!!!. Jakmile disk v tomto okamžiku vydá nějaký zvuk, prosím vypněte ho ihned síťovým vypínačem na zadní straně.

Nezastrkávejte žádný jiný kabel do diskové jednotky pokud je pod proudem. Dále vezměte kabel pro seriový bus a připojte ho do jedné z obou zásuvek na zadní straně drivu. Vypněte počítač a zasunete druhý konec kabelu zezadu do počítače. Tak a to je vše.
Pokud máte tiskárnu, nebo ještě další diskovou jednotku připojte kabel do druhé zásuvky seriového busu (viz obr. 3B originálu). Pro návod k použití většího počtu drivů si současně pročtěte kapitolu 9. Když používáte poprve najednou několik drivů začněte pracovat pouze s jedním, dokud si na jednotku dobře nezvyknete.

Připojení na síť

Pokud jste všechna zařízení navzájem propojili, je čas zapnout síť. Je důležité zapínat jednotlivá zařízení ve správném pořadí!!!. Počítač by měl být vždy zapnut poslední. Pokud bude počítač zapnut poslední pak je vše v pořádku.

Především se před zapnutím přesvědčte, že jste vyjmuli diskety ze všech drivů!!!

Když už byly zapnuty všechny ostatní přístroje, pak lze bezpečně zapnout i počítač. Všechny ostatní přístroje pak vykonají své počáteční úkony. Spustí se motor tiskárny a tiskací hlava projde do poloviny řádky a vrátí se zpět. Disková jednotka 1541 rozsvítí své červené světlo (chybové) a pak zelené světlo zabliká, zatím co se na televizní obrazovce vytvoří počáteční obraz. Jakmile všechna světla přestanou blikat na drivu, je již bezpečné s drivem pracovat.

Obr. 4 originálu
Acces slot - přístupný otvor
Insert into drive - zastrč do drivu
Write protect notch - drážka pro ochranu zápisu
Když je drážka pro ochranu zápisu zakryta, nelze měnit obsah diskety (nelze zapisovat).

Vkládání diskety

Chcete-li otevřít dvířka drivu, dejte páčku do vodorovné polohy. Pokud je disketa v drivu, je vyhozena malou pružinkou. Vezměte disketu, kterou chcete vložit a umístěte ji do drivu nálepkou (lící) nahoru a otvorem napřed s drážkou ochrany proti zápisu na levé straně.

Lehce disketu zatlačte a když je zcela zastrčena ucítíte zaklapnutí a disketa už sama nevyskočí. Uzavřete páčkou dvířka tak, že ji posunete do svislé polohy až zaskočí na své místo. Nyní jste připraveni začít pracovat s disketou.

Nikdy nezapomeňte vyjmout disketu z drivu dřív, než vypnete nebo zapnete počítač. Nikdy nevyjímejte disketu pokud svítí zelené světlo. V takovém případě mohou být data na disketě zničena.

3 Užití programu

Zavádění hotových programů

Pro ty z vás, kdo se zajímá pouze o loadování hotových programů připravených a dostupných z disku, zde je vše co musíme udělat:

Užíváte-li diskový drive, opatrně vsuňte předem programový disk tak, že označení na disku je směrem nahoru a nejblíže k vám. Podívejte se, kde je malá drážka na disku (může být kryta malým kouskem pásky). Vkládáte-li disk správně, drážka je na levé straně. Jakmile je disketa uvnitř, otočte západku do svislé polohy. Pak napište na klávesnici počítače:

BASIC 2:
LOAD"název programu", číslo zařízení, povel
Příklad: LOAD"HOW TO USE",8

BASIC 3.5:
DLOAD"název programu", Dčíslo drivu, Ičíslo zařízení, povel
Příklad: DLOAD"HOW TO USE"
DLOAD předpokládá číslo zařízení 8 a drive 0 pokud nejsou uvedena explicitně.

Povel není povinný. Pokud není udán, nebo je roven 0, je program loadován normálně, to znamená, že se začne ukládat na začátku paměti, která je k dispozici pro programy BASIC. Je-li udáno číslo 1 , program bude zaveden přesně na to paměťové místo, ze kterého pochází. Povel 1 se používá především pro programy ve strojovém kódu.

Po napsání příkazu stiskněte RETURN. Disk zahučí a na vaší obrazovce se objeví:

```
SEARCHING FOR "HOW TO USE"
LOADING
READY
```

Jakmile se objeví READY a kursor začne blikat, napište RUN a váš předem připravený program je připraven k použití.
Poznámka: Můžete používat proměnné jako číslo zařízení a stringy jako názvy, pouze pokud jste je ve svém programu definovali.

Direktorium (seznam) diskety

Váš magnetofon pracuje jen v dané posloupnosti od začátku do konce. Nemůže přeskakovat, vynechávat, jít do zadu nebo zaznamenávat přes stará data.
Vaše disková jednotka je zařízení s libovolným výběrem. Čtecí a psací hlava diskové paměti se může přesunout na kterékoliv místo na disku a vyhledat i jediný blok údajů, který může obsahovat až 256 slabik informací.Na disketě je celkem 683 bloků informací.

Naštěstí se nemusíte starat o jednotlivé bloky údajů. V diskové jednotce máte program nazvaný DOS (diskový operační systém).Tento program si uchovává poznámky o blocích pro vaší potřebu. Organizuje je do BAM (mapa volných bloků) a direktoria.

Mapa volných bloků (BAM) je jednoduchý seznam všech 683 bloků na disketě. Je umístěna uprostřed diskety, na polovině cesty mezi vnitřní a vnější stopou. Po každém záznamu programu (SAVE) nebo uzavření souboru (CLOSE), je BAM aktualizována seznamem bloků které byly použity.

Direktorium je seznam všech programů a jiných souborů uložených na disketě. Fyzicky je seznam umístěn vedle BAMu. V direktoriu je možno udělat 144 záznamů, obsahující informace, jako jméno a typ filu, seznam použitých bloků a počáteční blok.

Direktorium je automaticky doplňováno pokaždé, když je zaznamenáván nový program, nebo je soubor otevřen pro psaní. Pamatujte si však, že BAM není aktualizován, dokud soubor není uzavřen (CLOSE), i když se direktorium opraví hned.

Direktorium (seznam) může být zaveden do paměti stejně jako program BASIC.

Vložte disketu do jednotky a na klávesnici napište:

BASIC 2:

```
LOAD"$",8
```

Po napsání příkazu stiskněte RETURN. Disk zahučí a na vaší obrazovce se objeví:

```
SEARCHING FOR "$"
FOUND $
LOADING
READY
```

Tím je direktorium v paměti vašeho počítače. Napište LIST a direktorium se vypíše na obrazovce.

BASIC 3.5:

```
DIRECTORY
```

Po napsání příkazu stiskněte RETURN. Direktorium se vypíše na obrazovce.

Chcete-li vytisknout direktorium na tiskárně napište na klávesnici tuto příkazovou řádku (v tomto případě má tiskárna číslo zařízení 4).

```
LOAD"$",8
OPEN 4,4: CMD 4: LIST
PRINT#4: CLOSE 4
```

Poznámka: Nedávejte mezeru mezi PRINT a # a nepokoušejte se zkrátit povel na ?#. Můžete však ke zkrácení použít P+SHIFT R Pro podrobnou informaci použijte manuál k tiskárně.

Návod jak prohlížet direktorium uvnitř programu v BASICu najdete v kap. 5 (v oddílu, který se zabývá příkazen GET#).

Srovnatelné vzory a žolíkové karty.

Při použití kazetového přehrávače můžete zavést každý program začínající na dané písmeno tak, že vynecháte písmena následující. Povel LOAD"T" najde první program na pásce,

který začíná písmenem T, povel LOAD"HELLO" najde první program začínající písmeny HELLO, například "HELLO THERE" atd.

Při použití diskové paměti se tato možnost nazývá srovnání vzorů a je pro ni stanoven zvláštní znak ve jménu souboru, který ji určuje. Hvězdička, která následuje za jménem programu říká diskové paměti, že chcete najít každý soubor začínající daným jménem.
Formát pro srovnatelné vzory:

LOAD"jméno souboru+*",8

Jinými slovy, chcete-li zavést první program na disketě, který začíná písmenem T použijte povel: LOAD"T*",8

Použijete-li pouze hvězdičku jako jméno souboru, disková paměť zavede do operační paměti poslední program, s nímž byl proveden výběr (bylo do něj psáno, nebo z něj čteno). Nebyl-li proveden výběr se žádným programem, bude použit ten, který je první na seznamu v direktoráři.

Pravděpodobně znáte žolíkové karty, kdy jedna karta může nahradit jakoukoliv jinou. Na diskové paměti 1541-II může otazník nahradit jakýkoli znak na disketě.

Jméno programu na disketě je porovnáno se jménem uvedeným v povelu LOAD, avšak znaky, na jejichž místě je otazník, se neporovnávají. Například, vydáte-li povel: LOAD"T?NT",8
mohou být zavedeny všechny programy jako TINT, TENT atd.

Při zavádění seznamu diskety se k nalezení daných programů používá jak srovnatelných vzorů, tak žolíkových karet.
LOAD"$0: TEST"
objeví se na seznamu pouze program TEST, pokud je na disketě.
Povel: LOAD"$0: T*"
vám vydá seznam všech programů začínajících písmenem T.

Povel: LOAD"$0: T?ST"
vydá všechny programy se čtyřmi písmeny, v jejichž jménu je první písmeno T a třetí a čtvrté písmeno ST.
Povel: LOAD"$0: T?ST*"
vydá jména všech délek se správným prvním, třetím a čtvrtým písmenem.

SAVE

K záznamu programu na disketu nepotřebujete nic jiného, než přidat číslo přístroje ke jménu programu.

BASIC 2:

SAVE"číslo drivu:jméno souboru",číslo zařízení
Poznámka: číslo drivu je nepovinné

BASIC 3.5:

DSAVE"jméno souboru", číslo drivu, číslo zařízení

Tento příkaz uloží program BASIC na disk. Za příkazem musí být uvedeno jméno souboru, sestávající maximálně ze 16 znaků. DSAVE předpokládá číslo zařízení 8 a číslo drivu 0 pokud nejsou uvedena explicitně.

Počítače COMMODORE 16 a Plus 4 mají zabudován monitor strojového jazyka, který se vyvolává příkazem MONITOR. Pro nahrávání (SAVE) z monitoru strojového jazyka je formát příkazu:
S"číslo drivu:jméno souboru", číslo zařízení, počáteční adresa, koncová adresa +1
Příklad: S"0: DOS 5.1",08,CC00,CF5A

Když jste nařídili diskové paměti, aby vám zaznamenala program, diskový operační systém musí vykonat několik úkonů.

Nejdříve se podívá na direktorium a zjistí, zda program tohoto jména již neexistuje. Dále ověří, zda je v direktoriu k dispozici místo pro dané jméno. Pak ověří na BAM, zda je

dostatek volných bloků, na kterých bude uložen program. Je-li vše v pořádku, program bude zaznamenán, není-li, začne blikat červené kontrolní světlo oznamující omyl.

SAVE AND REPLACE (SAVE s náhradou)

Když již program na disku existuje, je ho často nutné upravit a znovu uložit na disk. V tomto případě by bylo nepohodlné vymazat starou verzi programu a pak zaznamenávat novou.

Jsou-li prvními znaky jména programu @ (zavináč) a 0 následovaná dvojtečkou, diskový operační systém ví, že má nahradit každý starý program, který má dané jméno programem novým, který je nyní v operační paměti počítače. Drive překontroluje direktorium, aby našel starý program, pak zruší jeho vstup, zapíše nový vstup stejným jménem. Nakonec se program normálním způsobem uloží.

BASIC 2:

SAVE"@0:revidovaný program",8

Důvodem, proč se v povelu uvádí 0: je zachovat slučitelnost s jinými diskovými jednotkami COMMODORE, které mají více zabudovaných jednotek. V tomto případě číslo 0 nebo 1 by bylo použito k určení, která jednotka má být použita.

BASIC 3.5:

DSAVE"@revidovaný program"

VERIFY

Povel porovná program, který je momentálně v operační paměti s programem na disketě. Musíte přitom v povelu VERIFY uvést číslo přístroje. Počítač provádí srovnání programu slabiku za slabikou, včetně řádkových spojek, které mohou být rozdílné u různého složení paměti.

BASIC 2 a 3.5:

VERIFY"číslo draivu:jméno souboru", číslo zařízení, povel

Povel má stejný účel jako v příkazu LOAD. Neuvedené číslo drivu se předpokládá 0.

4 Diskové povely

OPEN a PRINT#

Doposud jsme se zabývali jednoduchými způsoby práce s diskovou jednotkou. Abyste mohli komunikovat s disketovou jednotkou v plném rozsahu, musíte se blíže seznámit s příkazy OPEN a PRINT# v BASICu (větší podrobnosti najdete ve svých manuálech k počítači). Měli byste se dobře seznámit s jejich použitím pro fily na mg kazetách, kde OPEN vytváří file a příkaz PRINT# ho naplňuje daty. Lze ho použít stejným způsobem pro disk, jak uvidíte v další kapitole. Mohou být ale použity také k nastavení povelového kanálu. Povelový kanál vám umožňuje výměnu informací mezi počítačem a diskovou pamětí.

OPEN číslo souboru, číslo přístroje, číslo kanálu,text

Číslo souboru může být každé číslo od 1 do 255. Toto číslo je pak použito v celém programu k určení, do kterého souboru je vstup činěn. Je však lepší vyhnout se číslům větším než 127, poněvadž tato čísla způsobují, že příkaz PRINT# automaticky generuje signál pro nový řádek po znaku Return. Tato čísla jsou míněna pouze k použití na nestandartních tiskárnách. Přístrojové číslo diskové paméti je zpravidla 8. Kanál může být kterékoliv číslo od 2 do 15. To platí o kanálu použitém pro komunikaci, čili spojení s diskem, přičemž kanály 0 a 1 jsou rezervovány pro operační systém na zavádění a záznam. Kanály 2 až 14 mohou být používány pro soubory dat a kanál č. 15 je povelový kanál. Text je řetězec, který je napsán do souboru stejně jako u povelu PRINT#. Je to prakticky pro vydání jednoho povelu přes kanál.

Příklad:

```
OPEN 15,8,15
OPEN 2,8,2
OPEN A,B,C,Z$
```

PRINT#

Povel PRINT# pracuje úplně stejně jako PRINT s tou vyjímkou, že data jdou do jiného přístroje a ne na obrazovku, v tomto případě do diskové paměti. Při použití s datovým kanálem vyšle povel PRINT# informace do vyrovnávací paměti v diskové paměti,odkud pak jdou na disketu. Při použití PRINT# s povelovým kanálem jsou povely vyslány do diskové paměti.

Formát pro vyslání diskových povelů:

```
OPEN 15,8,15,"povel"
```

nebo

```
PRINT#15,"povel"
```

NEW

Tento příkaz je nutný v případě, když je disketa používána poprvé. Příkaz NEW maže celou disketu, zaznamená časovací a blokové značky na disketu a vytváří BAM a direktorium. Příkaz NEW lze též použít pro vymazání direktoria již formátované diskety. Je to rychlejší než formátovat celý disk znovu.

BASIC 2:

```
PRINT#15,"NEW0:jméno diskety,ID"
```

nebo zkráceně:

```
PRINT#15,"N0:jméno diskety,ID"
```

formát pro vyčištění direktoráře:

```
PRINT#15,"N0:jméno diskety"
```

BASIC 3.5:

HEADER"jméno diskety,ID",Dčíslo drivu,Učíslo zařízení
kde číslo zařízení se předpokládá 8 pokud není uvedeno.

Kód ID je jakýkoli dvojznakový kód, který je uvedený nejenom na seznamu, ale na každém bloku po celé disketě. Když omylem vyměníte disketu při psaní dat, disková paměť se porovnáním identifikačních čísel dozví, že něco není v pořádku.

COPY

Tento povel vám umožňuje pořídit kopii programu nebo filu na diskové jednotce. Nebude provádět kopie z jedné jednotky na druhou (nejde-li o dvojitou diskovou paměť 4040), ale může okopírovat program pod jiným jménem na stejné diskové paměti.

BASIC 2:

```
PRINT# 15,"COPY0:nový soubor=0:starý soubor"
```

nebo zkráceně:

```
PRINT# 15,"C0:nový soubor=0:starý soubor"
```

Povel COPY se používá také ke spojení dvou až čtyř souborů na disketě:

```
PRINT# 15,"C0:nový soubor=0: starý soubor1, 0:starý
soubor2,0:starý soubor3, 0:starý soubor4"
```

BASIC 3.5:

```
COPY Dčíslo drivu, "starý soubor" To Dčíslo drivu,
"nový soubor", Učíslo zařízení
```

COPY předpokládá číslo zařízení 8 a drivu 0 pokud nejsou uvedena explicitně.

```
COPY"Originál" To "Kopie"
```

nebo:

```
COPY"Originál" To "Kopie",U9
```

RENAME

Tento povel vám umožňuje změnit jméno souboru,které je již v direktoráři. Jde o rychlou operaci, protože se mění pouze jméno.

BASIC 2:

```
PRINT# 15,"RENAME0:nové jméno=staré jméno"
```

nebo zkráceně:

```
PRINT# 15,"R0:nové jméno=staré jméno"
```

BASIC 3.5:

RENAME"staré jméno" TO "nové jméno", Dčíslo drivu, Učíslo zařízení

RENAME předpokládá číslo zařízení 8 a drivu 0 pokud nejsou uvedena.

Příklad:

RENAME"staré jméno" TO "nové jméno"

SCRATCH

Tento povel vymaže z diskety nežádoucí soubory nebo programy a uvolní prázdné bloky pro nové informace. Můžete vymazat programy jeden po druhém nebo v celé skupině použitím srovnatelných vzorů nebo žolíkových karet.

BASIC 2:

PRINT#15,"SCRATCH0:jméno"

nebo zkráceně:

PRINT# 15,"S0:jméno"

BASIC 3.5:

SCRATCH"jméno",Dčíslo drivu,Učíslo zařízení

nebo zkráceně:

SCRATCH"jméno"

Z obezřetnosti se počítač zeptá "ARE YOU SURE?".
Stiskněte Y aby se příkaz SCRATCH provedl, nebo jiné tlačítko, aby se operace zrušila.
SCRATCH předpokládá číslo zařízení 8 a drivu 0 pokud nejsou uvedena.

Když zkontrolujete chybový kanál po operaci SCRATCH, číslo obvykle rezervované pro číslo stopy nyní říká kolik filů bylo smazáno. Tak např. když vaše direktorium obsahuje programy KNOW a GNAW a vy použijete příkazu PRINT#15,"S0:?N?W", smažete oba programy. Když direktorium obsahuje programy TEST, TRAIN, TRUCK a TAIL a vy

použijete příkazu PRINT#15,"S0: T*" vymaže všechny tyto 4 programy.

INITIALIZACE

Někdy se stane, že na diskové paměti dojde k omylu a vy nemůžete vykonat operaci, kterou jste chtěli. Příkaz INITIALIZE vám vrátí disk do stejného stavu jako kdyby byl právě zapnut.

Příklad:

```
PRINT#15,"INITIALIZE číslo drivu"
```

nebo zkráceně:

```
PRINT# 15,"I0"
```

VALIDATE

Po častém a delším používání diskety může být její direktorář rozházen, dezorganizován. Když byly programy opakovaně mazány a zaznamanávány (SAVE, SCRATCH), mohou zanechat četné malé mezery na disku. Tyto bloky by již nikdy nebyly použity, poněvadž místo je příliš malé pro účelné využití. Příkaz VALIDATE zreorganizuje disketu tak, aby se dalo využít co nejvíce prostoru, který je na ní k dispozici.

Na disketě mohou být také soubory dat, které byly otevřeny, ale nikdy nebyly řádně uzavřeny CLOSEm. Povel VALIDATE shromáždí všechny bloky obsazené těmito soubory a dá je k dispozici diskové paměti, protože neuzavřené soubory se nedají normálně použít.

S povelem VALIDATE je spojeno určité nebezpečí. Při používání náhodilých souborů budou tímto povelem přidělené bloky odebrány.

Proto tento povel nesmí být použit v souvislosti s disketou, která obsahuje náhodilé soubory.

BASIC 2:

```
PRINT# 15,"VALIDATE0"
```

nebo zkráceně:

```
PRINT# 15,"V0"
```

BASIC 3.5:

```
COLLECT Dčíslo drivu,Učíslo zařízení
```

Příklad:

```
COLLECT D0
```

Ctení chybového kanálu

Bez pomocného programu diskového operačního systému neexistuje žádný způsob, jak přečíst diskový chybový kanál bez programu, protože potřebujete použít povel INPUT#, který nebude pracovat samostatně mimo program. Zde je jednoduchý program v BASICu pro čtení chybového kanálu:

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

kde A$ je číslo chyby
B$ je jméno chyby
C$ je stopa
D$ je blok

Kdykoliv provedete operaci INPUT#, z povelového kanálu čtete až čtyři proměnné, které popisují podmínky chyb. První, třetí a čtvrtá proměnná přicházejí jako čísla a můžete je INPUTem vložit do numerických proměnných, pokud chcete. První proměnná je číslo chyby, kde 0 značí bez chyby. Druhá proměnná je popis chyby, třetí proměnná je číslo stopy, na které došlo k chybě a poslední, čtvrtá proměnná je číslo bloku na této stopě (blok bývá nazýván také sektorem).

Chyby na stopě 18 mají co činit s BAM a direktoriem. Např. výpis READ ERROR na stopě 18, bloku 0 může znamenat že disketa nebyla naformátována.

CLOSE

Je velmi důležité, aby každý soubor byl po použití řádně uzavřen příkazem CLOSE. Zavření souboru způsobí, že

diskový operační systém řádně přidělí bloky do mapy volných bloků (BAM) a ukončí záznam do direktoria.
Když soubor neuzavřete, všechna data jsou ztracena.

Příklad:

CLOSE číslo filu

Musíte dát pozor, abyste neuzavřeli omylový kanál (kanál č. 15) dříve, než uzavřete CLOSEm datové kanály. Omylový kanál musí být otevřený jako první a zavřený jako poslední na všech souborech. Tím dosáhnete, že vám programy poběží bez potíží.

Zavřete-li omylový kanál dokud jsou ostatní soubory otevřeny, disková paměť je všechny uzavře, ale BASIC bude stále myslet, že jsou ještě otevřeny (dokud je řádně neuzavřete) a dovolí vám , abyste se pokoušeli na ně psát.

Poznámka:

Když vás program v jazyku BASIC zavede do chybového stavu, všechny soubory jsou v BASIC uzavřeny, ale přitom nejsou uzavřeny na diskové paměti. To je velice nebezpečná situace. Okamžitě musíte napsat příkaz OPEN 15,8,15,"I"
To vám znovu uvede disk do původního stavu a zabezpečí všechny vaše fily.

5 Sekvenční fily

OPEN

Sekvenční soubory pracují na diskové jednotce stejně jako na magnetické kazetě, pouze mnohem rychleji. Jsou omezeny svou sekvenční povahou, čili posloupnostním charakterem, což znamená, že musí být čteny od počátku do konce. Data jsou přenášena byt po bytu přes vyrovnávací pamět na magnetické media. Pro diskovou pamět jsou všechny soubory vytvořeny stejně, to znamená, že nevidí rozdíl mezi sekvenčními, programovými a uživatelskými soubory. Rozdíl je jen v tom, že pouze programové soubory mohou být loadovány, ale to je skutečně jediný rozdíl. Dokonce i seznam pracuje tímto způsobem, s tím rozdílem, že má jen čtecí charakter. Rozdílné jsou však relativní soubory.

Formát pro otevření sekvenčního souboru:

OPEN číslo souboru, číslo zařízení, číslo kanálu,"číslo drivu:jméno, typ, směr"

Číslo souboru je stejné jako u všech ostatních aplikací příkazu OPEN a je používáno v celém programu ve vztahu k danému souboru.

Číslo zařízení je obvykle 8.

čísla zařízení	název zařízení
0	klávesnice
1	magnetofon
2	RS 232
3	obrazovka
4-7	tiskárna
8-11	disková jednotka

Číslo kanálu je číslo datového kanálu, číslo od 2 do 14. Je vhodné používat stejné číslo pro kanál i pro soubor.

Jménem rozumíme jméno souboru (nelze používat srovná-

vání vzorku a žolíkových karet když tvoříte psací soubor).

Typ může být cokoli z níže uvedeného přehledu nebo alespoň první písmeno každého typu.

typ file	význam
PRG	programový
SEQ	sekvenční
USR	uživatelský
REL	relativní (není zahrnut do BASICu 2)

Směr musí být čtení nebo psaní (READ nebo WRITE), nebo alespoň první písmeno z těchto slov (R nebo W).
Příklad otevírání sekvenčních souborů:

```
OPEN2,8,2,"0: DATA,S,W"
OPEN8,8,8,"0: PROGRAM,P,R"
```

V prvním příkladě je jméno souboru DATA, v druhém PROGRAM. Typ souboru je v prvním příkladě sekvenční, ve druhém příkladě programový. Směr je v prvním příkladě WRITE, ve druhém příkladě READ. Když soubor existuje, můžete využít možnost výměny v povelu OPEN, podobně jako v povelu SAVE and REPLACE (záznam s výměnou), popsaném ve 3. kapitole, Jednoduše přidejte @0: před jméno souboru do příkazu OPEN.

Příklad sekvenčního souboru s možností výměny:

```
OPEN2,8,2,"@0: DATA,S,W"
```

PRINT# a INPUT#

Povel PRINT# pracuje přesně jako příkaz PRINT s tím rozdílem, že výstup je usměrněn na diskovou pamět. Důvod, proč jsme podtrhli slovo přesně je, že všechny formátovací schopnosti příkazu PRINT pokud jde o interpunkci platí také zde. Také to však znamená, že musíte být velmi opatrní při vkládání dat do souboru.

Formát pro psaní do souboru povelem PRINT#:

PRINT# číslo souboru, seznam dat

Výraz číslo souboru je z příkazu OPEN použitého pro vytvoření souboru.

Výraz seznam dat je stejný jako u normálního příkazu PRINT. Jde o výčet proměnných nebo o text uvnitř uvozovek nebo o obojí. Musíte však být zvláště opatrní při psaní dat tak, aby bylo co nejsnadnější je opět později přečíst.

Když u příkazu PRINT# použijete čárků "," pro oddělení jednotlivých položek na řádku, tyto položky budou odděleny několika mezerami, zatímco středník ";" tyto mezery nevytvoří.

Abychom lépe porozuměli, co se děje, uvádíme dále diagram sekvenčního souboru vytvořeného příkazem:

OPEN5,8,5,"0: TEST,S,W"

	eof														
znak	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

EOF znamená END OF FILE, tj. konec souboru. Data typu string (řetězcová data) vstupují do souboru byt po bytu, včetně mezer. Stanovme si, například, několik proměnných příkazem A$="HELLO"; B$="ALL"; C$="BYE". Zde je obraz souboru po příkazu PRINT# 5,A$, B$, C$:

	H	E	L	L	O	A	L	L	B	Y	E	CR	eof		
znak	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

CR znamená kód CHR$(13), což znamená návrat hlavy do výchozí polohy. Tento pokyn tvoří konec každého příkazu PRINT nebo PRINT#, pokud řádek nekončí čárkou nebo středníkem.

Poznámka:

Nedávejte mezeru mezi PRINT a #, a nepokoušejte se zkrátit povel na ?#. PRINT# jde zkrátit stisknutím P, SHIFT+R.

Formát pro příkaz INPUT#:

INPUT# číslo souboru, seznam proměných

Při použití INPUT# pro přečtení dat nemáte žádnou možnost vyjádřit, že nejde o jeden dlouhý řetězec. Potřebujete něco v tomto souboru, co by vám sehrálo roli oddělovače, (separátoru). Znaky, které lze použít jako oddělovače jsou např. CR, čárka nebo středník. CR může být snadno přidán použitím jedné proměnné na řádek v příkazu PRINT#, což bude mít za následek, že systém ho tam zařadí automaticky. Příkaz PRINT# 5,A$: PRINT# 5,B$: PRINT# 5,C$ zařadí CR při psaní za každou proměnnou, kterou bude psát za předpokladu správného oddělení v příkazu jako INPUT#5,A$,B$,C$. Jinak řádek jako Z$=",": PRINT#5,AZBZC$ udělá stejnou práci a na menším prostoru. Soubor po tomto řádku vypadá takto:

	H	E	L	L	O	,	A	L	L	,	B	Y	E	CR	eof
znak	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

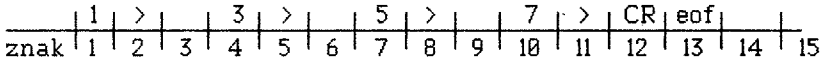
Dáme-li mezi proměnné čárky, způsobíme, že na disketě vznikne hodně volného prostoru. Příkaz jako PRINT# 5,A$,B$ vytvoří soubor, který vypadá takto:

	H	E	L	L	O							A	L	L	.		
znak	1	2	3	4	5	6	7	8	9	10	11	12	13	14	.	23	24

Vidíte, kolik volného prostoru je zde nevyužito.

Poučení z toho je: povel PRINT# používejte opatrně tak, aby se data dala řádně přečíst. Číselná data napsaná do souboru dostanou formu řetězce jako kdyby byla na ně před napsáním uplatněna funkce STR$. První znak bude prázdné místo je-li

číslo kladné, nebo bude znaménko mínus, je-li číslo záporné. Pak následuje číslo a posledním znakem je znak kurzor doprava. Tento formát dává dostatek informací, aby je příkaz INPUT# mohl přečíst jako samostatná čísla, i když je jich několik napsáno bez oddělovačů. Je to trochu ztráta místa, protože mohou zůstat dva nevyužité znaky, když jsou čísla kladná.
Zde je obraz souboru po vykonání příkazu PRINT# 5,1,3,5,7:



GET#

Příkaz GET# vybere data z diskety znak po jednom znaku.

Formát pro příkaz GET#:

GET# číslo souboru, seznam proměnných

Data přicházejí byt po bytu, včetně CR, čárky a dalších oddělovacích znaků. Je mnohem bezpečnější použít řetězcové proměnné při příkazu GET#, protože dostanete oznámení omylu v BASIC, byla-li přijata řetězcová data tam, kde mělo být číslo, ale ne obráceně.
Příklad:

```
GET# 5,A$
GET# 5,A$,B$,C$
GET# A,A
```

Poznámka: Můžete dostat najednou i více než jeden znak.

Příkaz GET# je velmi užitečný při zkoumání souborů s neznámým obsahem, jako například souboru, který mohl být poškozen zkušebním programem. Je jistější než INPUT#, protože počet znaků možných mezi oddělovači u proměnných v INPUTu je omezen. U příkazu GET# dostanete každý znak a můžete zkoumat také oddělovače, stejně jako ostatní data.

Příklad programu, který vám umožní prozkoumat každý soubor na disketě:

```
10    INPUT"jméno souboru"; F$
20    INPUT"typ filu"; T$
30    T$=LEFT$(T$,1)
40    IF T$<>"S" THEN IF T$<>"P"THEN IF T$<>"U" THEN 20
45    OPEN15,8,15
50    OPEN 5,8,5,"0:"+F$+","+T$+",R"
60    GOSUB 200
70    GET# 5,A$
80    IF ST<>0 THEN PRINT ST: STOP
90    PRINT ASC(A$+CHR$(0));
100   GOTO 70
200   INPUT#15,A$,B$,C$.D$
210   IF VAL(A$)>0 THEN PRINT A$,B$,C$,D$: STOP
220   RETURN
```

Poznámka k řádku 90:
Je-li přečten nulový znak, způsobí chybu funkce ASC.

Demonstrační program na sekvenční file originál str.54

Čtení direktoráře

Direktorář diskety může být čten jako běžný sekvenční soubor. Stačí použít $ jako jména souboru.
Napište:

```
OPEN 5,8,5,"$"
```

Nyní příkaz GET# pracuje na prozkoumání seznamu. Formát je zde stejný jako formát programového souboru: rozměry souborů jsou čísla řádků, jména jsou uložena jako znaky uvnitř uvozovek.

1541 BAM formát

(stopa 18,Sektor 0)

Byte	Obsah	Definice
0,1	18,01	Stopa a sektor prvního bloku direktoria
2	65	ASCII znak A označující formát 1541/4040
3	0	Nulový flag pro budoucí použití DOS
4-143		Bitová mapa dostupných bloků pro stopy 1-35
		1=volný blok 0=blok není k dispozici (každý bit představuje 1 blok)

1541 Záhlaví direktoria

(Stopa 18, Sektor 0)

Byte	Obsah	Definice
144-159		Jména disku
160-161	160	Shift mezera
162-163		ID disku
164	160	Shift mezera
165-166	50,65	ASCII znázornění pro 2A, což je verze DOS a typ formátu
167-170	160	Shift mezera
170-255	0	Nuly-není použito

POZOR: ASCII znaky se mohou objevit na některých disketách na pozici 180-191

1541 Direktory formát

(stopa 18, sektor 1)

Byte	Definice
0,1	Stopa a sektor dalšího bloku direktoria
2-31	Záznam pro file 1 *
34-63	Záznam pro file 2 *
66-95	Záznam pro file 3 *
98-127	Záznam pro file 4 *
130-159	Záznam pro file 5 *
162-191	Záznam pro file 6 *
194-223	Záznam pro file 7 *
226-255	Záznam pro file 8 *

* struktura jednotlivých záznamů v direktoriu

Byte	Obsah	Definice
0	128	Typ souboru porovnaný logicky OR
	+	pro označení řádně uloženého souboru
	typ	Typy:
		0=smazaný DEL
		1=sekvenční SEQ
		2=programový PRG
		3=uživatelský USR
		4=relativní REL
1,2		Stopa a sektor prvního bloku dat
3-18		Jméno souboru doplněné
		mezerami se shiftem
19-20		Pouze pro relativní fily: stopa
		a sektor první strany bloku
21		Pouze pro relativní fily: rozměr věty
22-25		Nepoužito
26-27		Stopa a sektor náhradního souboru
		při vydání povelu
		@ SAVE nebo @ OPEN
28-29		Počet bloků v souboru: nejnižší byte,
		nejvyšší byte

Sekvenční formát

Byte	Definice
0,1	Stopa a sektor dalšího sekvenčního bloku
2-256	254 bytů dat s CR jako oddělovači rekordů

Formát programového filu

Byte	Definice
0,1	Stopa a sektor dalšího programového souboru
2-256	254 bytů programu uloženého v CBM formátu paměti (s klíčovými slovy vyznačenými). Konec souboru je označen třemi nulovými byty.

6 Nahodilé soubory

Sekvenční soubory jsou výhodné pro práci s plynulým tokem dat, ale v některých případech se chce víc. Máte-li, například dlouhý seznam adres, nebude se vám chtít procházet celým seznamem, chcete-li jen jednu konkrétní adresu. Proto potřebujete nějakou metodu libovolného výběru, nějaký způsob dostat se ke kterémukoli zápisu uvnitř souboru bez toho, abyste museli číst celý soubor.

Na diskové paměti COMMODORE máme dva různé druhy souborů s libovolným výběrem. Relativní soubory popsané v další kapitole jsou vhodnější pro operace s daty, i když náhodilé soubory, o kterých budeme mluvit v této kapitole mají svoji platnost, zejména při práci ve strojovém jazyce.

Náhodilé soubory na diskové paměti COMMODORE nalézají jednotlivé 256-ti slabikové bloky dat uložené na disketě. Jak jsme se zmínili v první kapitole, na disketě jsou celkem 683 bloky, z nichž jsou k dispozici na prázdné disketé 664. Každý blok dat ve skutečnosti znamená jednu stopu a sektor stejného jména.

Disketa se dělí na stopy, které tvoří soustředné kruhy na jejím povrchu. Máme celkem 35 stop, počínaje stopou 1 na vnějším okraji diskety a konče stopou 35 uprostřed. Stopa 18 se používá pro direktory a diskový operační systém zaplňuje disketu ze středu k okraji.

Každá stopa se dále dělí na sektory. Protože na vnějších stopách je více místa, je na nich také více sektorů. Vnější stopy mají po 21 sektorech, zatímco uvnitř pouze 17.

Tabulka ukazuje počet sektorů na stopě:

číslo stopy	rozsah sektorů	celkově sektorů
1-17	0-20	21
18-24	0-18	19
25-30	0-17	18
31-35	0-16	17

Diskový operační systém obsahuje povely pro čtení a psaní přímo na každou stopu a sektor na disketě. Jsou také povely, kterými zjistíte, které bloky (stopy a sektory) jsou volné a povely, kterými označíte použité bloky.

Tyto povely se předávají přes povelový kanál (kanál č. 15) a řeknou diskové paměti, co má s daty dělat. Data musí být později čtena přes některý otevřený datový kanál.

Otevření datového kanálu pro libovolný výběr

Když pracujeme se soubory s libovolným výběrem, potřebujeme mít současně otevřeny 2 kanály k disku, jeden pro povely a druhý pro data. Povelový kanál je otevřený ke kanálu 15, stejně jako ostatní diskové povely, se kterými jste se dosud setkali. Datový kanál pro soubory s libovolným výběrem je otevřen zvolením znaku # jako jména souboru.

Formát příkazu OPEN pro data s libovolným výběrem:

OPEN číslo souboru, číslo zařízení, číslo kanálu, "#"
nebo
OPEN číslo souboru, číslo zařízení,"číslo vyrovnávací paměti+#"

Příklady otevření datového kanálu s libovolným výběrem:

```
OPEN 5,8,5,"#"
OPEN A,B,C,"#+2"
```

Poznámka:
1. U prvního povelu neudáváte vyrovnávací paměť.
2. U druhého povelu volíte vyrovnávací paměť č. 2.

BLOCK-READ

PRINT# číslo souboru,"U1";kanál;disková paměť; stopa; blok
nebo zkráceně:
PRINT# číslo souboru,"UA:"kanál;disková paměť; stopa; blok

Tento povel přemístí jeden blok dat z diskety na zvolený kanál. Po provedení této operace mohou příkazy INPUT# a GET# informace přečíst.
Příklad programu k přečtení bloku 2 na stopě 18 (usklad-ňující obsah v B$):

```
10 OPEN 15,8,15
20 OPEN 5,8,5,"#"
30 PRINT#15,"B-R";5;0;18;2
40 B$=""
50 FOR L=0 TO 255
60 GET#5,A$
70 IF ST=0 THEN B$=B$+A$: NEXT L
80 PRINT"KONEC"
90 CLOSE5: CLOSE15
```

Poznámka:
1. Na řádku 30: 5=číslo kanálu
0=číslo diskové paměti
18=stopa
2=sektor
Stopa a sektor dohromady tvoří blok.
2. Na řádku 70 bude sesbírán celý blok byt po bytu

BLOCK-WRITE

Povel BLOCK-WRITE je přesný protiklad povelu BLOCK-READ. Nejdříve musíte vyplnit datovou vyrovnávací pamět vašimi údaji a pak napíšete obsah vyrovnávací paměti na správné místo na disku.

Formát povelu BLOCK-WRITE:

PRINT# číslo souboru,"U2";kanál,disková paměť; stopa; blok
nebo zkráceně:
PRINT# číslo souboru,"UB:"kanál;disková paměť; stopa;blok

Když jsou data vložena do vyrovnávací paměti, ukazatel v diskovém operačním systému počítá, kolik znaků v ní je. Když vykonáte operaci BLOCK-WRITE, ukazatel je zaznamenán na disketě. To je důvod pro kontrolu ST na řádku 70 výše uvedeného programu: ST se stane nenulovým, když budete číst za znakem konec souboru.

Příklad programu k napsání dat na stopu 1, sektor 1:

```
10 OPEN 15,8,15
20 OPEN 5,8,5,"#"
30 FOR L=1 TO 50
40 PRINT#5,"TEST"
50 NEXT
60 PRINT#15,"B-W";5;0;1;1
70 CLOSE5: CLOSE15
```

BLOCK-ALLOCATE

Abychom mohli bezpečně používat náhodilé soubory společně s normálními, vaše programy musí prověřit BAM, aby našel volné bloky a pak změnit BAM tak, aby obsahoval informaci o tom, které bloky jste použili. Jakmile se bloky BAM upravili podle současného stavu, vaše soubory s náhodným přístupem budou bezpečné alespoň do té doby, dokud nevydáte povel VALIDATE.

Formát příkazu BLOCK-ALLOCATE:

PRINT# číslo souboru,"BLOCK-ALLOCATE";disková paměť; stopa;blok

nebo zkráceně:

PRINT# číslo souboru,"B-A";disková paměť;stopa;blok

Jak víte který blok je volný? Když se pokusíte použít blok, který není volný, diskový operační systém vám oznámí omyl číslo 65-NO BLOCK (není blok), a nastaví číslo stopy a bloku na

Jak víte který blok je volný? Když se pokusíte použít blok, který není volný, diskový operační systém vám oznámí omyl číslo 65-NO BLOCK (není blok), a nastaví číslo stopy a bloku na nejbližší volnou stopu a blok. Proto vždy, když chcete napsat nový blok na disketu musíte nejdříve tento blok přidělit, anglicky allocate. Není-li tento blok volný, přečtěte si číslo dalšího volného bloku z omylového kanálu a přidělte ho.
Příklad postupu při přidělování bloku:

```
10 OPEN 15,8,15
20 OPEN 5,8,5,"#"
30 PRINT#5,"DATA"
40 T=1: S=1
50 PRINT#15,"B-A";0; T; S
60 INPUT#15,A,B$,C,D
70 IF A=65 THEN T=C: S=D: GOTO 50
80 PRINT#15,"B-W";5;0; T; S
```

BLOCK-FREE

Povel BLOCK-FREE je opakem povelu BLOCK-ALLOCATE tím, že uvolní blok, který chcete použít kdekoli v systému. Vzdáleně připomíná povel SCRATCH, ale ve skutečnosti nevymazává z diskety žádná data, nýbrž pouze uvolní vstup, v tomto případě mapu volných bloků-BAM.

Formát povelu BLOCK-FREE:

PRINT# číslo souboru,"BLOCK-FREE";disková paměť; stopa; blok

nebo zkráceně:

PRINT# číslo souboru,"B-F";disková paměť;stopa;blok

Používání náhodilých souborů

Jediný problém, s nímž jste se doposud seznámili v souvislosti s náhodilými soubory je zjištění, že nemáte způsob, jak sledovat, které bloky na disketě byly použity. Proto nemůžete

rozlišit volné a obsazené bloky na mapě volných bloků-BAM. Nemůžete říci, zda obsahuje náhodilý soubor, část programu nebo vůbec nějaký sekvenční nebo relativní soubor.

Nejběžnější způsob jak sledovat, co je a není obsazeno, je vytvořit sekvenční soubor, který bude doprovázet každý náhodilý soubor. Používejte tento soubor pouze pro vedení soupisu vět, stop a bloků. To znamená, že jsou otevřeny tři kanály k disketě pro každý náhodilý soubor: jeden povelový kanál, druhý pro náhodilá data a třetí pro sekvenční data. To také znamená, že máte najednou otevřeny dvě vyrovnávací paměti, které zaplňujete současně.

Příklad programu psaní deseti bloků s náhodilým výběrem současně se sekvenčním souborem:

```
10 OPEN 15,8,15
20 OPEN 5,8,5,"#"
30 OPEN 4,8,4,"@0:keys,S,W"
40 A$="Obsah věty#"
50 FOR R=1 TO 10
70 PRINT#5,A$","R
90 T=1: B=1
100 PRINT#15,"B-A";0; T; B
110 INPUT#15,A,B$,C,D
120 IF A=65 THEN T=C: B=D: GOTO 100
130 PRINT#15,"B-W";5;0; T; B
140 PRINT#4, T, B
150 NEXT R
160 CLOSE4: CLOSE5: CLOSE15
```

Příklad programu čtení deseti bloků s náhodilým výběrem spolu se sekvenčním souborem:

```
10 OPEN 15,8,15
20 OPEN 5,8,5,"#"
```

```
30 OPEN 4,8,4,"Tlačítka,S,R"
40 FOR R=1 TO 10
50 INPUT#4,T,S
60 PRINT#15,"B-R";5; 0; T; S
80 INPUT#5,A$,X
90 IF A$<>"Obsah vět#" OR X<>R THEN STOP
110 PRINT#15,"B-F";0; T; S
120 NEXT R
130 CLOSE 4: CLOSE 5
140 PRINT#15,"S0: KEYS"
150 CLOSE 15
```

Poznámka:

1. Na řádku 50 (písmena T a S) najde program další použité stopy a sektory.
2. Na řádku 70 program prověří, zda jsou data v pořádku.

BUFFER-POINTER

Tento povel má za úkol sledovat, kam byla napsána poslední data. Současně je to i ukazatel určující, kde mají být čtena poslední data. Když změníme místo ukazatele ve vyrovnávací paměti, dostaneme libovolný výběr či vstup k jednotlivým slabikám uvnitř bloku. Tímto způsobem můžeme rozdělit každý blok na jednotlivé věty.

Vezměme si pro příklad seznam adres, adresář. Informace, jako jméno, adresa atd., zabere nejvíce 64 znaky. Mohli bychom rozdělit každý blok souboru s libovolným výběrem na čtyři samostatné věty a když budeme znát stopu, sektor a číslo věty, můžeme vstoupit do příslušné věty.

Formát povelu BUFFER-POINTER

PRINT# číslo souboru,"BUFFER-POINTER";kanál;místo
nebo zkráceně:
PRINT# číslo souboru,"B-P";kanál;místo

Příklad nastavení ukazatele na 64 znak ve vyrovnávací paměti:

```
PRINT#15,"B-P:"5;64
```

Dále uvádíme verze psaní a čtení s libovolným vstupem u výše uvedeného programu, upraveného pro práci s větami uvnitř bloků.

Příklad programu psaní deseti bloků s libovolným výběrem se čtyřmi větami v každém z nich:

```
10 OPEN 15,8,15
20 OPEN 5,8,5,"#"
30 OPEN 4,8,4,"@0: Keys,S,W"
40 A$="Obsah věty#"
50 FOR R=1 TO 10
60 FOR L=1 TO 4
70 PRINT#15,"B-P";5;(L-1)*64
80 PRINT#5,A$","L
90 NEXT L
100 T=1: B=1
110 PRINT#15,"B-A";0; T; B
120 INPUT#15,A,B$,C,D
130 IF A=65 THEN T=C: B=D: GOTO 110
140 PRINT#15,"B-W";5;0; T; B
150 PRINT#4,T,B
160 NEXT R
170 CLOSE4: CLOSE5: CLOSE15
```

Poznámka:

1. Na řádku 70 (L-1) znamená umístění na 0, 64, 128 nebo 192.
2. Na řádku 110-130 jsou nalezeny volné stopy a sektory.

Příklad programu čtení deseti bloků s libovolným výběrem se čtyřmi větami v každém z nich:

```
10 OPEN 15,8,15
20 OPEN 5,8,5,"#"
30 OPEN 4,8,4,"@0: 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
85 INPUT#5,A$,X
90 IF A$<>"Obsah vět#" OR X<>R THEN STOP
100 NEXT L
110 PRINT#15,"B-F";0; T; S
120 NEXT R
130 CLOSE 4: CLOSE 5
140 PRINT#15,"S0: KEYS"
150 CLOSE 15
```

USER 1, USER 2

Tyto uživatelské povely jsou určeny pro práci ve strojovém jazyku. Povely USER 1, USER 2 jsou zvláštní verzí povelů BLOCK-READ a BLOCK-WRITE s důležitou odlišností, kterou je způsob, jak povely USER 1 a USER 2 pracují s ukazatelem vyrovnávací paměti.

Povel BLOCK-READ přečte až 256 znaků, ale přestane číst, když ukazatel vyrovnávací paměti tohoto bloku řekne, že blok je ukončen. Povel USER 1 vykoná stejnou operaci jako povel BLOCK-READ, ale nejdříve přinutí ukazatel přejít na 255, aby mohl přečíst celý blok dat z diskety najednou.

Formát povelu USER 1:

PRINT# 15,"U1";kanál;disková paměť;stopa;blok
nebo zkráceně:

PRINT# 15,"UA:"kanál;disková paměť;stopa;blok
PRINT# 15,"U1:"kanál;disková paměť;stopa;blok
PRINT# 15,"U1:kanál;disková paměť;stopa;blok"

Povel BLOCK-WRITE napíše obsah vyrovnávací paměti do bloku na disketě společně s hodnotou ukazatele vyrovnávací paměti. Povel USER 2 napíše vyrovnávací paměť bez narušení hodnoty ukazatele vyrovnávací paměti, která je již uložena na daném bloku na disketě. To je užitečné, když blok má být přečten povelem BLOCK-READ, aktualizován pomocí povelů BUFFER-POINTER a PRINT# a pak napsán zpět na disketu povelem USER 2.

Formát povelu USER 2:
PRINT# 15,"U2";kanál;disková paměť;stopa;blok
nebo zkráceně:
PRINT# 15,"UB:"kanál;disková paměť;stopa;blok
PRINT# 15,"U2:"kanál;disková paměť;stopa;blok
PRINT# 15,"U2:kanál;disková paměť;stopa;blok"

7 Relativní soubory

Relativní soubory vám snadno umožní dostat se na přesně tu část dat, kterou ze souboru potřebujete. Jsou velmi dobré pro práci s daty, protože vám umožňují tvořit soubory ve větách a v polích uvnitř vět.

Diskový operační systém sleduje, které stopy a sektory jsou použity a dokonce umožňuje, aby věty přerostly z jednoho bloku do druhého. Je to možné, protože zřizuje vedlejší sektory (side sektors), čili sérii ukazatelů pro začátek každé věty. Každý vedlejší sektor může pojmout až 120 vět, přičemž soubor může mít až 6 vedlejších sektorů. V souboru může být až 720 vět a každá věta může mít 254 znaků, takže soubor může být stejně velký jako celá disketa.

Vytvoření relativního souboru

Když chceme použít relativní soubor poprvé, vytvoříme ho nejdříve příkazem OPEN a pak ho použijeme. Možnost náhrady či výměny nemáme a tlačítko @ soubor nevymaže ani nevytvoří nový. Soubor může být rozšířen, čten nebo se může do něj psát.

Formát příkazu OPEN pro vytvoření relativního souboru:

```
OPEN číslo souboru,číslo přístroje,číslo kanálu,"jméno,L,"
+CHR$(délka věty)
```

Příklady příkazů OPEN při tvoření relativních souborů:

```
OPEN 2,8,2."Soubor,L,"+CHR$(100)
OPEN F,8,F,A$+",L,"+CHR$(Q)
OPEN A,B,C,"TEST,L,"+CHR$(33)
```

Poznámka:
Znaky nebo čísla v povelu CHR$ určují délku věty.

Formát relativního souboru

Datový blok

Byte	Definice
0,1	Stopa a sektor dalšího bloku dat
2-255	254 bytů dat. Prázdné věty obsahují FF v první slabice následované 00 na konci věty, částečně naplněné věty jsou proloženy nulami (00)

	Blok vedlejšího sektoru
0,1	stopa a sektor dalšího bloku vedlejšího sektoru
2	číslo vedlejšího sektoru (0-5)
3	délka věty
4,5	stopa a sektor prvního vedlejšího sektoru (číslo 0)
6,7	stopa a sektor druhého vedlejšího sektoru (číslo 1)
8,9	stopa a sektor třetího vedlejšího sektoru (číslo 2)
10,11	stopa a sektor čtvrtého vedlejšího sektoru (číslo 3)
12,13	stopa a sektor pátého vedlejšího sektoru (číslo 4)
14,15	stopa a sektor šestého vedlejšího sektoru (číslo 5)
16-256	ukazatele stopy a sektoru na 120 datových bloků

Při výkonu se diskový operační systém nejdříve přesvědčí, zda soubor existuje. Když ano, nic se nestane. Je-diný způsob, jak vymazat starý relativní soubor je použít povelu SCRATCH, zatímco možnost náhrady, výměny (REPLACE) zde neexistuje.

Použití relativních souborů

Formát pro otevření relativního souboru, který již existuje, je jednoduchý:

OPEN číslo souboru,číslo přístroje,číslo kanálu,"jméno"

V tomto případě diskový operační systém automaticky ví, že jde o relativní soubor. Uvedený syntax a syntax, který jsme uvedli v předchozí části, umožní buď čtení nebo psaní souboru.

Abychom mohli číst nebo psát, musíme nejdříve, jako první operaci, umístit ukazatel souboru na správné místo věty.

Formát povelu na umístění ukazatele:

PRINT číslo souboru,"P"+CHR$(číslo kanálu+96)+CHR$(nejnižší číslo věty)+CHR$(nejvyšší číslo věty)+CHR$(místo)

Příklad povelu na umístění ukazatele:

```
PRINT#15,"P"+CHR$(98)+CHR$(3)+CHR$(0)
PRINT#15,"P"+CHR$(CH)+CHR$(R1)+CHR$(R2)
```

Dvojslabičný formát pro číslo věty je nutný, protože jedna slabika může uchovat pouze 256 různých čísel, zatímco my můžeme mít v souboru přes 700 vět. Nejnižší číslo věty obsahuje nejméně významnou část adresy, zatímco nejvyšší číslo věty je její nejdůležitější částí. To se dá přeložit do skutečného čísla věty vzorcem:

```
REC#=REC HI*256+REC LO
```

což znamená: číslo věty=nejvyšší číslo věty krát 256 plus nejnižší číslo věty

Předpokládejme že pracujeme s adresářem. Každá adresa se skládá z osmi údajů této skladby:

Jméno položky	délka
Jméno	12
Příjmení	15
První řádek adresy	20
Druhý řádek adresy	20
Město	12
Stát	2
PSČ	9
Telefon	10
Celkem	100

Tímto je určena délka věty. Možná, že budeme chtít větu prodloužit o další znak v každé položce, abychom umožnili oddělení, protože jinak by povel INPUT# vzal daleko delší část souboru než je třeba, jako kdyby šlo o sekvenční soubor. Proto stanovíme délku souboru se 108 znaky ve větě. U první věty dáme číslo 1, které představuje zatím největší číslo věty. Dále je uveden program, který to znázorňuje:

```
10  OPEN 1,8,15
20  OPEN 2,8,3,"0: MAILING LIST,L,"+CHR$(108)
30  GOSUB 900
40  PRINT#1,"P"CHR$(3+96)CHR$(1)CHR$(0)CHR$(1)
50  GOSUB 900
60  IF E=50 THEN PRINT#2,1: GOTO 40
70  INPUT#2,X
300 STOP
900 INPUT#1,E,B$,C,D
910 IF (E=50) OR (E<20) THEN RETURN
920 PRINT A; B; C; D: STOP: RETURN
```

Chyba č. 50, která je testována na řádce 60 je chyba RECORD NOT PRESENT (rekord není zapsán), a znamená, že věta ještě nebyla vytvořena. Psaní do věty vyřeší tento problém. Na tuto chybu si dávejte v programech dobrý pozor.

Zatím je tedy vytvořen soubor a první věta, ale nebyla vložena žádná věta. Dále je uvedena značně rozšířená verze předchozího programu, který umožňuje pracovat s adresářem, v němž jsou věty zakódovány v číslech.

Program pro čtení a psaní adresáře:

```
5   A(1)=12: A(2)=15: A(3)=20: A(4)=20: A (5)=12: A(6)=2:
    A(7)=9: A(8)=10
10  OPEN 1,8,15: OPEN 2,8,3,"0: Adresář,1,"+CHR$(108): GOSUB
    900
```

```
20  PRINT#1,"P"CHR$(3+96)CHR$(1)CHR$(0)CHR$(1): INPUT#2,X
30  INPUT"Ctení, psaní nebo konec"; J$: IF J$="K" THEN
    CLOSE2: CLOSE1: END
40  IF J$="P"THEN 200
50  PRINT: INPUT"Věta číslo"; R: IF R<0 OR R>X THEN50
60  IF R<2 THEN 30
70  R1=R: R2=0: IF R1>256 THEN R2=INT(R1/256): R1=R1-256*R2
80  RESTORE: DATA1,Jméno,14, Příjmení,30, Adresa1,51,
    Adresa2
90  DATA72,Město,85,Stát,88, PSC,98,t.číslo
100 FORL=1TO8: READA,A$: PRINT#1, "P" CHR$(3+96) CHR$(R1)
    CHR$(R2) CHR$(A): GOSUB900
110 ONA/50 GOTO50: INPUT#2,Z$: PRINTA$,Z$: NEXT: GOTO50
200 PRINT: INPUT"Císlo věty"; R: IF R<0ORR>5000 THEN200
210 IFR<2 THEN30
215 IFR>X THENR=X+1: PRINT: PRINT"číslo věty"R
220 R1=R: R2=0: IFR1>256THEN R2=INT(R1/256): R1=R1-256*R2
230 RESTORE: FORL=1TO8: READA,A$: PRINT#1, "P" CHR$(3+96)
    CHR#(R1) CHR$(R2) CHR$(A)
240 PRINTA$,: INPUTZ$: IFLEN(Z$)>A(L) THEN
    Z$=LEFT$(Z$,A(L))
245 PRINT#2,Z$: NEXT: X=R: PRINT#1, "P" CHR$(3+96) CHR#(1)
CHR$(0)
250 PRINT#2,X: GOTO200
900 INPUT#1,A,B$,C,D: IFA<20 THEN RETURN
910 IFA<>50 THEN PRINTA; B$, C; D: STOP: RETURN
920 IFJ$="C"THEN PRINTB$
930 RETURN
```

Tento program při vyjímání požaduje čísla vět. Nedovolí vám vyjmout údaj umístěný za ukazatelem konce souboru a pokud se za něj pokusíte psát, přinutí vás psát na další vyšší větu.

Složitější verze tohoto programu by sledovala položky podle určitého klíče a větám by přidávala indexy. Chtěli byste například najít větu podle jména nebo podle počátečních čísel PSČ. K tomu budete potřebovat zvláštní seznam klíčů a čísel vět, což by bylo uloženo pravděpodobně v sekvenčním souboru.

Když pracujeme s novým relativním souborem, který se nám brzy rozroste, je účelné a šetří čas, když vytvoříme větu na předpokládaném konci souboru. Jinými slovy, očekáváte-li, že soubor bude mít tisíc vět, vytvořte větu číslo 1000 hned na začátku, jakmile vytvoříte soubor.Tím přinutíte diskový operační systém, aby vytvořil všechny věty v daném prostoru, což urychlí jejich pozdější použití.

Příklad vytvoření velkého souboru:

```
OPEN1,8,15: OPEN2,8,2"REL,L"+CHR$(60)
PRINT#1,"P"CHR$(2+96)CHR$(0)CHR$(4)CHR$(1)
PRINT#2,"END"
CLOSE2: CLOSE1
```

Poznámka:
Na druhém řádku: indexy 0 a 4 u CHR$ znamenají: věta číslo 4 krát 256 plus 0 nebo 1024.

8 Programování řadiče diskety

Zkušený programátor může vytvořit programy a rutiny, které jsou uloženy na řadiči diskety a řídí práci s ní. Z diskety mohou být k tomu přidány rutiny diskového operačního systému. Tyto rutiny se přidávají obdobným způsobem jako podpůrné programy diskového operačního systému, které se vklíní do paměti.

BLOCK-EXECUTE

Tento povel zavede z diskety blok obsahující rutinu ve strojovém jazyku a začne ji vykonávat na pamětovém místě 0 ve vyrovnávací paměti, dokud nenarazí na povel RTS, což znamená RETURN FROM SOBRUTINE, česky Návrat z podprogramu.

Formát povelu BLOCK-EXECUTE:

PRINT#15,"B-E";číslo kanálu;číslo drivu;stopa;sektor
nebo:
PRINT#15,"B-E:";číslo kanálu;číslo drivu;stopa;sektor
PRINT#15,"B-E:číslo kanálu;číslo drivu;stopa;sektor"

MEMORY-READ

Na diskové paměti je 16k paměti ROM (permanentní) a 2k paměti RAM (s libovolným výběrem). Můžete získat přímý přístup k oběma, nebo k vyrovnávací paměti, kterou sestavil DOS v RAMu použitím příkazu pro paměť. Povel MEMORY-READ umožní zvolit slabiku ke čtení, a to přes chybový kanál.

Formát povelu MEMORY-READ:
PRINT#15,"M-R"CHR$(nejnižší slabika adresy)CHR$(nejvyšší slabika adresy)CHR$(počet čtených bytů)
nebo:
PRINT#15,"M-R:"CHR$ (nejnižší slabika adresy) CHR$ (nejvyšší slabika adresy) CHR$ (počet čtených bytů)

MEMORY-WRITE

Tento povel umožní psát až 34 slabik najednou do paměti řadiče diskety. Povely MEMORY-EXECUTE a uživatelské povely (USER) se používají ke zpracování tohoto kódu.

Formát povelu MEMORY-WRITE:

PRINT#15,"M-W" CHR$ (nejnižší slabika adresy) CHR$ (nejvyšší slabika adresy) CHR$ (počet čtených bytů) CHR$ (data1)

nebo:

PRINT#15,"M-W:" CHR$ (nejnižší slabika adresy) CHR$ (nejvyšší slabika adresy) CHR$ (počet čtených bytů) CHR$ (data1)

MEMORY-EXECUTE

Každá rutina v paměti diskového operačního systému, v paměti RAM nebo paměti ROM může být vykonána povelem MEMORY-EXECUTE.

Formát povelu MEMORY-EXECUTE:

PRINT#15,"M-E" CHR$ (nejnižší slabika adresy) CHR$ (nejvyšší slabika adresy)

nebo:

PRINT#15,"M-E:" CHR$ (nejnižší slabika adresy) CHR$ (nejvyšší slabika adresy)

Následuje příklad, užívající povelu M-W a M-E pro vykonání instrukce RTS:

```
10  OPEN 15,8,15
20  PRINT#15,"M-W" CHR$ (0) CHR$ (5)1 CHR$ (96)
30  PRINT#15,"M-E" CHR$ (0) CHR$ (5)
40  CLOSE 15
```

USER-Uživatelské povely

Vedle povelů USER1 a USER2, se uživatelské povely používají pro skoky na tabulku paměťových míst v paměti RAM disketové jednotky.

Uživatelský povel	Funkce
U1 nebo UA	BLOCK-READ bez změny ukazatele vyrovnávací paměti
U2 nebo UB	BLOCK-WRITE bez změny ukazatele vyrovnávací paměti
U3 nebo UC	skok na $0500
U4 nebo UD	skok na $0503
U5 nebo UE	skok na $0506
U6 nebo UF	skok na $0509
U7 nebo UG	skok na $050C
U8 nebo UH	skok na $050F
U9 nebo UI	1541 NMI
U; nebo UJ	1541 reset
UI+	nastav rychlost C-64
UI-	nastav rychlost VIC-20

Zavedení jiného skokového povelu do těchto míst, jako například JMP$0520, lze vytvořit další rutiny, které operují v diskové paměti dle tabulky skoků, dokonce i z BASIC.

Formát povelu USER:

```
PRINT#15,"U charakter"
```

9 Změna přístrojového čísla

Metoda pomocí programového vybavení.

Disková pamět zvolí přístrojové číslo tak, že se podívá na hardwerové číslo, které zapíše do sekce BAMu. Jakmile je operace skončena je jednoduché původní číslo přepsat číslem zařízení novým.

Formát pro změnu čísla zařízení:

```
PRINT# číslo souboru,"M-W" CHR$ (119) CHR$ (0) CHR$(2)
CHR$(adresa+31) CHR$ (adresa+64)
```

Příklad změny přístrojového čísla na číslo 9:

```
PRINT#15,M-W" CHR$ (119) CHR$ (0)  CHR$ (2) CHR$ (9+31)
CHR$ (9+64)
```

Máte-li více než jednu disketovou jednotku, je rozumné změnit číslo zařízení pomocí hardwerového (technického) vybavení.

Vypněte disketovou jednotku ze sítě a nastavte přepínače, které jsou umístěny na zadní straně jednotky, do takové polohy, která odpovídá vámi požadovanému číslu zařízení.

číslo zař.	sw1	sw2
8	zapnuto	zapnuto
9	vypnuto	zapnuto
10	zapnuto	vypnuto
11	vypnuto	vypnuto

Příloha A

Přehled diskových povelů
(obecný formát PRINT# č. souboru, příkaz)

Povel	česky	formát
NEW	Nový	"N
COPY	Kopie	"C0:nový soubor=0:původní soubor
RENAME	Přejmenuj	"R0:nové jméno=0:staré jméno
SCRATCH	Vymaž	"S0:jméno souboru
INITIALIZE	Označ	"I
VALIDATE	Odsouhlas	"V
DUPLICATE	Rozmnož - neplatí pro 1541	
BLOCK-READ	Blok-čti	"B-R"; kanál; disk.paměť; stopa; blok
BLOCK-WRITE	Blok-zapiš	"B-W"; kanál; disk.paměť; stopa; blok
BLOCK-ALLOCATE	Blok-přiděl	"B-A"; disk.paměť; stopa; blok
BLOCK-FREE	Blok-uvolni	"B-F"; disk.paměť; stopa; blok
BUFFER-POINTER	Vyr.paměť+ ukazatel	"B-P"; kanál; postavení
USER1 a USER2	Uživatelské povely	"Un"kanál; disk.paměť; stopa; blok
POSITION	Postavení	"P" CHR$ (č,kanálu+96) CHR$ (nejnižší číslo věty) CHR$ (nejvyšší číslo věty) CHR$ (pozice)
BLOCK-EXECUTE	Blok-výkon	"B-E"; kanál; disk.paměť; stopa; blok
MEMORY-READ	Paměť-čti	"M-R" CHR$ (nejnižší adresa) CHR$ (nejvyšší adresa)
MEMORY-WRITE	Paměť piš	"M-W" CHR$ (nejnižší slabika adresy) CHR$ (nejvyšší slabika adresy) CHR$ (počet čtených bytů) CHR$ (data1)
MEMORY-EXECUTE	Paměť-výkon	"M-E"; kanál; disk.paměť; stopa; blok
USER	Uživatelské povely	"Un"

Příloha B

Popis chybových hlášení DOS.

00: Vše v pořádku, není chyba

01: Oznámení o provedeném smazání (SCRATCH). Informuje o čísle souboru který byl vymazán. Není chyba.

20: READ ERROR (Není nadpis bloku na disketě).
Diskový řadič nemůže najít nadpis žádného bloku dat, protože buď bylo vydáno nepřípustné číslo sektoru, nebo byl nadpis zničen.

21: READ ERROR (Není synchronizační znak)
Diskový řadič nemůže zjistit synchronizační znak na dané stopě. Příčinou může být špatné nastavení čtecí a psací hlavy, nepřítomnost diskety, špatně naformátovaná nebo špatně nasazená disketa. Omyl může znamenat i závadu v technickém vybavení.

22: READ ERROR (Není blok dat)
Diskový řadič byl požádán přečíst nebo ověřit blok dat, který nebyl správně napsán. Toto oznámení se objeví ve spojitosti s blokovými povely a znamená nepřípustnou žádost na stopu nebo sektor nebo obojí.

23: READ ERROR (Omyl v kontrolním čísle dat)
Znamená omyl v jedné nebo více slabikách dat. Data byla přečtena do paměti DOSu, ale kontrolní číslo těchto dat je mylné. Toto oznámení může znamenat také uzemňovací problém.

24: READ ERROR (Omyl v dekódování slabiky)
Data nebo nadpis byl přečten do paměti DOSu, ale v technickém vybavení vznikl omyl v důsledku neplatného tvaru slabiky dat. Toto oznámení může znamenat také uzemňovací problém.

25: WRITE ERROR (Omyl při ověření psaní)
Diskový řadič zjistil neshodu mezi zapsanými daty a daty v paměti DOS.

26: WRITE PROTECT ON (Pokus psát při ochraně proti zápisu)
Diskový řadič byl požádán napsat blok dat v situaci, kdy je disk chráněn proti zápisu. To se stane s disketou, která má zalepený ochranný zářez.

27: READ ERROR (Omyl v kontrolním čísle nadpisu)
Diskový řadič zjistil omyl v nadpisu žádaného bloku dat. Blok nebyl zatím přečten do paměti diskového operačního systému. Toto oznámení může také znamenat uzemňovací problém.

28: WRITE ERROR (Příliš dlouhý blok dat)
Diskový řadič se po napsání bloku dat pokouší zjistit synchronizační znak dalšího nadpisu. Nenajde-li ho ve stanoveném čase, objeví se oznámení omylu. Příčinou omylu je špatný formát diskety (data překročí do dalšího bloku) nebo závada v technickém zařízení.

29: DISK ID MISMATCH (Neshoda ID diskety)
Diskový řadič byl požádán o výběr diskety, která nebyla naformátována. Oznámení se objeví i v případě, že disketa má špatný nadpis.

30: SYNTAX ERROR (Obecný omyl v syntaxu)
DOS nemůže rozeznat povel vyslaný do povelového kanálu. Obvykle je to způsobeno nepřípustným množstvím jmén souboru nebo nesprávně použitými vzory. Například v povelu COPY se na levé straně objeví dvě jména souboru.

31: SYNTAX ERROR (Neplatný povel)
DOS povel nezná. Povel musí začít na prvním místě.

32: SYNTAX ERROR (Dlouhý řádek)
Vyslaný povel má více jak 58 znaků.

33: SYNTAX ERROR (Neplatné jméno souboru)
Srovnání vzorů je v povelech OPEN a SAVE nesprávně použito.

34: SYNTAX ERROR (Dlouhý řádek)
Jméno souboru nebylo v povelu uvedeno nebo DOS ho jako jméno nerozeznal. Obvykle k tomu dojde vynecháním dvojtečky.

35: SYNTAX ERROR (Neplatný povel)
DOS nerozezná povel vyslaný do povelového kanálu (sekundární adresa 15)

50: RECORD NOT PRESENT (Věta není)
Povely INPUT# nebo GET# byla disketa přečtena tak, že byla překročena poslední věta. Oznámení se objeví také po nastavení do věty za znakem konce soubotu u relativních souborů. Chceme-li rozšířit soubor přidáním nové věty (pomocí povelu PRINT#), nemusíme si oznámení všímat. Ale nemůžeme se pokoušet o povely INPUT nebo GET dokud neprovedeme nové nastavení.

51: OVERLOW IN RECORD (Překročení délky věty)
Příkaz PRINT# překročí hranice věty. Informace je zkomolená. Protože vrácení do výchozí polohy, které je vysláno jako přerušovač věty, se počítá v délce věty, oznámení se objeví, jestliže celkový počet znaků ve větě (včetně posledního návratu do výchozí polohy) překročí stanovenou délku.

52: FILE TOO LARGE (Soubor příliš velký)
Postavení věty uvnitř relativního souboru naznačuje, že dojde k překročení délky věty.

60: WRITE FILE OPEN (Soubor otevřen pro psaní)
Oznámení se objeví, když soubor pro psaní, který ještě nebyl zavřen, je otvírán pro čtení.

61: FILE NOT OPEN (Soubor není otevřen)
Oznámení se objeví, když jde o výběr souboru, který ještě v diskové operační soustavě nebyl otevřen. V takové situaci může dojít k tomu, že oznámení se neobjeví a žádosti není jednoduše vyhověno.

62: FILE NOT FOUND (Soubor není)
Požadovaný soubor na určené disketě neexistuje.

63: FILE EXISTS (Soubor existuje)
Jméno souboru, které bylo právě vytvořeno, již na disketě existuje.

64: FILE TYPE MISMATCH (Zmatení druhů souboru)
Druh souboru neodpovídá druhu uvedenému v seznamu žádaného souboru.

65: NO BLOCK (Blok není)
Oznámení se objeví ve spojitosti s povelem B-A. Znamená, že blok, který má být přidělen, již byl přidělen. Parametry označují stopy a sektory, které jsou k dispozici pro nejvyšší číslo. Jsou-li parametry 0, pak všechny bloky vyšší než 0 jsou použity.

66: ILLEGAL TRACK AND SECTOR (Nepřípustná stopa nebo sektor)
DOS se pokusil o výběr stopy nebo sektoru, který neexistuje v použitém formátu. To může znamenat problém se čtením ukazatele pro další blok.

67: ILLEGAL SYSTEM T OR S (Nepřípustný systém stopy nebo sektoru)
Toto zvláštní oznámení znamená, že systém stopy nebo sektoru je nepřípustný.

70: NO CHANNEL (Kanál není k dispozici)
Požadovaný kanál není k dispozici nebo všechny kanály jsou použity. Pro diskový operační systém může být otevřeno nejvíce pět sekvenčních souborů najednou. Kanály s přímým výběrem mohou mít šest otevřených souborů.

71: DIRECTORY ERROR (Omyl v seznamu)
Mapa vnitřních bloků neodpovídá vnitřním počtům. Objevil se problém v přidělení místa v BAM nebo mapa byla v paměti DOSu přepsána. Abychom problém odstranili, je třeba znovu označit disketu (INITIALIZE) a tak dát mapu volných bloků v paměti do pořádku. Některé soubory mohou touto úpravou utrpět.

72: DISK FULL (Disketa je plná)
Buď jsou použity všechny bloky na disketě nebo je direktorium zaplněno až po svou mez.

73: DOS MISMATCH (73,CBM DOS V 2.6 1541)
DOS 1 a 2 jsou psány, ale nikoli čteny slučitelně. Diskety se mohou střídavě číst jedním nebo druhým DOSem, ale na disketu naformátovanou na jedné verzi se nedá psát druhou verzí, protože je formát rozdílný. Omyl se objeví vždy, když se pokusíte psát na disk, který byl naformátován neslučitelnm způsobem. Oznámení se může objevit také po zapojení do sítě.

74: DRIVE NOT READY (Disková paměť není připravena)
Byl učiněn pokus o výběr diskové paměti 1541 bez diskety v paměti.

Příloha C

Programy na demonstrační disketě

Demonstrační disketa - obsahuje řadu uživatelských programů. Je chráněna proti zápisu (zalepením bočního otvoru), aby nemohlo dojít k přepsání programů na ní uložených. V dalších odstavcích jsou jednotlivé programy stručně popsány.

C-64 WEDGE - Zkrácené psaní příkazů umožňuje program DOS 5.1. Chcete-li jej použít, nahrajte (LOAD) a spusťte (RUN) program C-64 WEDGE (nebo VIC-20 WEDGE, máte-li tento počítač), který DOS 5.1 automaticky nainstaluje. Pak lze na př. místo LOAD"$",8 a LIST zobrazit název a obsah diskety (adresář-direktory) jediným příkazem @$, navíc bez přepsání programu v počítači.

PRINTER TEST - tiskne znaky a zkouší grafické možnosti připojené tiskárny. Doporučeny jsou modely 1525, MPS 801 a MPS 803.

DISK ADDR CHANGE - změní programově adresu další diskové jednotky dle volby (na př. z 8 na 9).

VIEW BAM - umožňuje prohlížet tabulku BAM, která obsahuje informace o obsazených a volných sektorech diskety.

DISPLAY T&S - umožňuje prohlížet obsah bloků specifikovaných číslem stopy a číslem sektoru.

CHECK DISK - zapisuje a zkouší správnost zápisu ve všech blocích diskety. Závadné bloky označí do BAM, aby nebyly dále DOSem používány. Programy nebo data na disketě uložené jsou ovšem nenávratně ztraceny.

PERFORMANCE TEST - zkouší elektroniku i mechaniku disku v případě, že je podezření na závadu v diskové jednotce.

SEQ. A REL.FILE.DEMO - jsou dva příklady užití dvou dalších způsobů zápisu dat na disketu. Ilustrují také použitou techniku testu chybového kanálu po každém volání diskové jednotky z počítače. V případě počítače VIC-20 je nutné užít expansní destičku 3K RAM.

SD.BACKUP.C64 - umožňuje provoz jen s jednou disketovou jednotkou 1541 tím, že dává na obrazovku pokyny k výměnám zdrojové (čtené) a cílové (zapisované) diskety a čeká na jejich provedení. Obdobné programy jsou pro počítače C16 a +4.

PRINT.64.UTIL - dovoluje tisk obrazovky a/nebo výpis skalárních proměnných i když je v paměti nějaký BASIC program. Tyto dvě možnosti mohou být užity společně nebo zvlášť (je-li třeba zabránit kolisi s jiným programem ve strojovém kodu v RAM). Obdobně pro C16 a +4.

C-64 BASIC DEMO - provádí zjednodušený test počítače a jeho možností. Je vhodný jako příklad programování.

LOAD ADDRESS - umožňuje změnu nebo prohlídku dvou byte "zaváděcí adresy" programového file na disketě.

UNSCRATCH - umožňuje opět zpřístupnit file, který byl vymazán z adresáře nebo jehož adresář byl přepsán ("počmárán"). To lze ovšem jen tehdy, nebyly-li jeho bloky mezitím přepsány novými zápisy.

HEADER CHANGE - dovoluje změnit název diskety bez vymazání jejího dalšího obsahu.

Tento návod připravil **Commodore klub** v Praze na základě originálu.

Všechny textové i grafické části příručky byly zpracovány na počítači Commodore **C64** s příslušenstvím firmy Commodore.

První seznámení i další práci s počítači Commodore vám usnadní nejbližší klub výpočetní techniky v místě bydliště nebo přímo **Commodore klub** Praha (602. ZO Svazarmu, dr. Zikmunda Wintra 6, Praha 6).

Vytiskl : TOMOS Praha