COMMODORE 64

DODATKY

Příloha A

Rozšíření počítačového systému COMMODORE 64

Firma COMMODORE má na světovém trhu již dlouholetou tradici. Počítač COMMODORE 64 je jedním z nejlepších a nejrozšířenějších osmibitových počítačů. Má řadu příznivců po celém světě, existuje řada klubů uživatelů těchto počítačů. Bylo také vydáno mnoho knih a časopisů zabývajících se problematikou spojenou s COMMODORE 64.

COMMODORE 64 má nejen dokonalý servis po celém světě, ale i bohatou nabídku progamového vybavení a periferních zařízení, které lze k tomuto počítači připojit. Jeho využití je skutečně velmi široké, může se uplatnit jako textový editor pro psaní korespondence stejně jako inteligentní kartotéka pro třídění dat.

Základní systém COMMODORE 64 se skládá z počítače, televizního příjmače nebo monitoru a magnetofonu pro záznam programů a dat na kazetu. Chcete-li však využít svého systému dokonaleji, můžete nahradit magnetofon disketovou jednotkou, která ukládá i nahrává data i programy podstatně rychleji než magnetofon.

Periferní zařízení

VIC 1541 DISK DRIVE

Disketová jednotka VIC 1541 může uložit až 144 programových nebo datových souborů na pružný disk (disketu) standartního formátu 5.25 palce. Je to zařízení, se kterým počítač aktivně spolupracuje a které vpodstatě zvyšuje jeho operační paměť. Na jednu stranu běžné diskety je možno uložit až 174 000 znaků. Nespornou výhodou je i fakt, že disketová jednotka uschovává i nahrává data čtyřicetkrát rychleji než magnetofon. K počítači COMMODORE 64 je možno připojit současně až pět disketových jednotek.

VIC 1525 DOT MATRIX PRINTER

Tiskárna je jedním z nejdůležitějších zařízení každého počítačového systému. Pomocí tiskárny lze zaznamenat na papír výpisy programů, postupy a výsledky výpočtů, tisknout zprávy, dopisy, seznamy a podobně. Bez tiskárny není žádný systém kompletní.

Bodová tiskárna VIC 1525 může tisknout nejen všechny znaky, které je možno zadat z klávesnice COMMODORE 64, ale i libovolné znaky podle Všeho zadání. Standardně tato tiskárna tiskne rychlostí 30 znaků za sekundu s hustotou 12 znaků na palec. Pro tisk používá perforovaný papír o šířce 10 palců.

VIC 1526 BI-DIRECTIONAL DOT MATRIX PRINTER

Tato tiskárna má proti tiskárně VIC 1525 dvě výhody. Hlavní výhoda spočívá v tom, že tisková hlava tiskne jak při pohybu zleva doprava, tak i při svém návratu. Druhou výhodou je to, že i vlastní tisk probíhá rychleji než u VIC 1525. Tiskneme-li 80 znaků na řádek, vytiskne tiskárna VIC 1526

za minutu 45 řádků, při 40 znacích na řádek je to 78 řádků za minutu a při 20 znacích 124 řádků za minutu. Tiskárna VIC 1526 může tisknout jak na perforovaný papír šíře 10 palců, tak i na volné listy a to až se třemi kopiemi.

VIC 1520 PRINTER/PLOTTER

Souřadnicový zapisovač VIC 1520 umožňuje nejen kreslit různé grafy, výkresy a obrázky, ale může i sloužit jako standartní tiskárna. Umí napsat všechna malá i velká písmena, grafické symboly a libovolné znaky dle zadání. Píše rychlostí 14 znaků za sekundu. Velikost písma je možno nastavit v rozsahu 10 až 40 znaků na řádek. Volitelný je rovněž počátek a natočení souřadnic. Souřadnicový zapisovač pracuje s přesností 0.2 mm a kreslí rychlostí 60 mm/sec. Pro kreslení používá malá kuličková pera ve čtyřech různých barvách.

1701 COLOR MONITOR

Nejvhodnějším zobrazovacím zařízením, které plně využívá možností COMMODORE 64, je barevný monitor. Barevný monitor 1701 má obrazovku s vysokou rozlišovací schopností o úhlopříčce 14 palců. Pro reprodukci zvuku lze použít jak monitoru tak jakéhokoli HIFI zařízení, které lze k monitoru snadno připojit.

1311 JOYSTICK

Pákový ovladač (joystick) lze použít nejen pro ovládání různých her, ale také, ve spojení s příslušným programem, pro kreslení a malování.

Programové vybavení

1. Programy uživatelské

EASYFILE EFI 6440 (disketa)

EASYFILE je obsáhlý kartotékový systém pro COMMODORE 64, pomocí kterého lze ukládat, vyhledávat a třídit data. Vyhledávané informace lze dle přání vypsat na obrazovce nebo vytisknout na tiskárně.

CLUB MANAGER CMG 6440 (disketa)

Tento program je zvláště vhodný pro zpracování evidence členů klubů nebo jiných kolektivů. Dovoluje zaznamenat různé údaje o jednotlivých členech. Kromě osobních údajů lze u každého člena zaznamenávat například zapůjčení sportovních potřeb, jejich rezervaci, placení příspěvků, účast na různých akcích apod. Tento seznam je uchováván na disketě a je možno doplňovat, opravovat, třídit a vymazávat jeho jednotlivé položky. Samozřejmě je možno tento seznam kdykoliv vytisknout na tiskárně. Vzhledem k tomu, že tento program může spolupracovat s programem EASYFILE, je možno vytvořit jednotný dopis a doplnit jej o adresy uvedené v seznamu programu CLUB MANAGER. Jistě by se našlo pro tento program mnoho dalších uplatnění.

FUTURE FINANCE FFI 6440 (disketa)

FUTURE FINANCE umožňuje předpovědět zisk, pohyb hotovosti, úspory a vydání vašeho podniku. Zkrátka tento program vám pomůže lépe spravovat vaše peněžní konto.

EASYCALC ECL 6440 (disketa)

EASYCALC je speciální matematický program. Obsahuje trigonometrické, statistické a ostatní matematické funkce. Pomocí tohoto programu lze zpracovávat různé matematické úlohy, vynášet, kreslit a zpracovávat grafy a podobně.

EASY STOCK EST 6440 (disketa)

EASY STOCK je program usnadňující vedení jakéhokoliv skladu. V podstatě je to kartotéka umožňující zaznamenávat cenu libovolného zboží, jeho nákup a prodej. Program samozřejmě umožňuje jakékoliv opravy a úpravy jak u jednotlivých položek tak i u skupin zboží. Kartotéka se ukládá jako soubor dat na disketu, její jednotlivé celky lze kdykoliv vytisknout na tiskárně.

EASY SCRIPT ESC 6440 (disketa)

EASY SCRIPT je profesionální program, který vám pomůže snadno a rychle vytvořit, upravit a natisknout jakýkoliv text. Pomocí tohoto programu můžeme psát a tisknout dopisy, reportáže, rukopisy, prostě jakékoliv dokumenty. Texty se uchovávají na disketě nebo na kazetě a lze je kdykoliv upravit a vytisknout přesně podle vašich požadavků.

EASY SPELL ESP 6440 (disketa)

EASY SPELL je program, který kontroluje správný pravopis v textech vytvořených pomocí programu EASYSCRIPT. Obsahuje vlastní slovník, který je uložen na disketě a podle kterého porovnává jednotlivá slova textu.

2. Programy systémové

Tyto programy vám pomohou při tvorbě vlastních programů v BASICu i ve strojovém kódu. Pomohou vám jak při programování tak při hledání a opravě chyb.

SIMONS' BASIC SIB 6410 (zásuvný modul)

SIMONS' BASIC je program, který velmi ulehčí tvorbu jakéhokoliv programu. Umožňuje totiž při programování v BASICu používat některé speciální příkazy. Tyto příkazy je mosno rozdělit do tří skupin, na řídící, grafické a zvukové. Uveďme alespoň několik příkladů:

a) řídící příkazy
AUTO - automatické číslování řádků
RENUMBER - přečíslování programových řádků
KEY - definování významu funkčních kláves

b) grafické příkazy
HIRES - přepnutí do módu jemné grafiky
REC - kreslení pravoúhlých obrazců
CIRCLE - kreslení kružnic
PAINT - vyplnění obrazce barvou

c) řízení SPRITů
DESIGN - nakreslení sítě pro tvorbu SPRITů
MMOB - přesun SPRITů
DETECT - automatický záznam kolize dvou SPRITů

Další velkou výhodou SIMONS' BASICu je možnost strukturovat program podobně jako u jiných vyšších jazyků. To znamená, že je možno jednotlivým podprogramům (procedurám) přidělit jména a volat je těmito jmény.

PROC - pojmenování BASICové procedury
CALL - provedení volané procedury
EXEC - provedení procedury a návrat do hlavního programu
REPEAT UNTIL- příkaz pro opakování smyčky podmíněný splněním předepsané podmínky

SIMONS' BASIC obsahuje dále příkazy pro formátování obrazovky, posun textu, vstup, manipulaci se znakovými řetězci, převod z decimální soustavy do hexadecimální, binární a naopak. Kromě těchto obsahuje ještě příkazy pro snadnější odstranění BASICových chyb při ladění programu. Programy napsané SIMONS' BASICem je možno uložit na kazetu i na disketu.

ASSEMBLER TUTOR AST 6440 (disketa)
AST 6420 (kazeta)

ASSEMBLER DEVELOPMENT ASM 6440 (disketa)

Tyto programy velmi usnadňují programování ve strojovém kódu. Pro zápis programu se používá mnemotechnických zkratek pro jednotlivé instrukce mikroprocesoru 6502, které jsou pak překládány do strojového kódu. Programy obsahují všechny potřebné nástroje pro tvorbu programu, jeho uložení, nahrání, přeložení a spuštění.

PROGRAMMER'S UTILITIES UTL 6440 (disketa)

Tento soubor programů obsahuje velmi užitečné krátké programy umožňující například změnu čísla připojeného zařízení, kopírování programů z diskety na disketu, snadnější naprogramování SPRITů, změnu znaků, ovládání zvukových generátorů, formátování výpisu na obrazovce a podobně.

5. Programy pro zábavu

Firma COMMODORE pamatuje při tvorbě programů i na ty z vás, kteří máte rádi hry vyžadující logickou úvahu a kombinační schopnosti. U žádného z níže uvedených programů není důležitá rychlost, ale zato prověří vaši schopnost logicky uvažovat, kombinovat i správně a výhodně obchodovat.

LABYRINTH LBY 6420 (kazeta)

HIGH FLYER HFL 6440 (disketa)

RAIL BOSS RBO 6440 (disketa)

OCEAN RACER OCR 6440 (disketa)

Příloha B

Popis chybových hlášení DOS

Chybová hlášení s číslem menším než 20 můžeme ignorovat s výjimkou hlášení číslo 01, které informuje o tom, který soubor je mazán příkazem SCRATCH.

20 READ ERROR - hlavička bloku není nalezena
Na disketě nebyla nalezena hlavička uloženého datového bloku. Příčinou je buď chybné číslo sektoru nebo je hlavička poškozena.

21 READ ERROR - není synchronizační značka
Na disketě nebyla nalezena synchronizační značka na žádané sropě. Příčinou může být chybné nastavení kombinované hlavy jednotky, nepřitomnost diskety, nenaformátovaná nebo nevhodná disketa. Toto chybové hlášení také indikuje nějaké hardwarové opomenutí.

22 READ ERROR - datový blok není přítomný
Na disketě nebyl nalezen blok, který má být čten nebo verifikován. Příčinou je jeho chybný zápis.

23 READ ERROR - chyba v kontrolním součtu bloku
Toto chybové hlášení se objeví v případě, že jeden nebo několik bytů je na disketě chybně zapsáno. Správnost zápisu jednotlivých bloků je kontrolována tzv. kontrolním součtem, pokud tento nesouhlasí je to vyhodnoceno jako chyba. Toto hlášení může také indikovat nějaký podstatnější problém.

24 READ ERROR - chybně dekódovaný byte
Do paměti DOS byl z bloku dat nebo z hlavičky načten

neúplný nebo chybný byte. Toto hlášení zpravidla indikuje závažný problém.

25 **WRITE ERROR** - chyba při zápisu nebo verifikaci
Zapisovaná nebo kontrolovaná data nesouhlasí s daty v DOS paměti.

26 **WRITE PROTECT ON**
Toto hlášení upozorňuje na to, že disketa, na kterou chceme zapisovat je proti zápisu chráněna. Spínač chránící disketu před nežádoucím zápisem je aktivován,protože je na disketě přelepen ochranný zářez.

27 **READ ERROR** - chyba v kontrolním součtu hlavičky
Byla nalezena chyba v kontrolním součtu hlavičky bloku. Podobně jako v případě chyby v kontrolním součtu bloku, indikuje toto hlášení nějaký závažnější problém.

28 **WRITE ERROR** - dlouhý datový blok
Není nalezena synchronizační značka pro zápis hlavičky následujícího bloku. Tato značka byla smazána blokem předcházejícím. Ukazuje to buď na chybně naformátovanou disketu nebo na hardwarovou závadu v disketové jednotce.

29 **DISK ID MISMATCH**
Při čtení nebo zápisu nebyla nalezena hlavička diskety, disketa není inicializována. Toto hlášení se také objeví jestliže disketa má chybnou hlavičku.

30 **SYNTAX ERROR** - všeobecná chyba
DOS neumí provést zadaný příkaz. Bylo použito nepřípustné číslo ve jméně souboru nebo byl příkaz nesprávně použit.

31 **SYNTAX ERROR** - chybný příkaz
DOS nerozumí zadanému příkazu. Příkaz musí být zadán znovu.

32 SYNTAX ERROR - chybný příkaz
Příkaz je delší než 58 znaků.

33 SYNTAX ERROR - chybné jméno souboru
Chybně použitý příkaz OPEN nebo SAVE.

34 SYNTAX ERROR - není zadán soubor
Jméno souboru je vlevo od příkazu nebo DOS nezná takový příkaz.

39 SYNTAX ERROR - chybný příkaz
Toto hlášení se objeví, jetliže je použit příkazový kanál (sekundární adresa 15) a DOS nerozumí zadanému příkazy.

50 RECORD NOT PRESENT
Používáme-li pro čtení dat z diskety příkazu INPUT# nebo GET# a není-li na disketě příslušný záznam, objeví se toto chybové hlášení. Chyba se rovněž indikuje, jestliže jsou tyto příkazy použity po uzavření souboru.

51 OVERFLOW IN RECORD
Toto chybové hlášení se objeví a zápis příkazem PRINT# je ukončen, jestliže je překročen definovaný rozsah záznamu.

52 FILE TOO LARGE
Toto chybové hlášení se objeví v případě, že zapisovaný soubor je delší než je volná kapacita diskety.

60 WRITE FILE OPEN
Toto chybové hlášení se objeví, chceme-li zapisovat do souboru, který byl otevřen pro čtení.

61 FILE NOT OPEN
Toto chybové hlášení se objeví, jestliže chceme použít souboru, který není dosud otevřen. V některých případech není toto hlášení indikováno a žádost o přístup do souboru je ignorována.

62 FILE NOT FOUND

Toto chybové hlášení se objeví, jestliže žádaný soubor není na zvolené disketě.

63 FITE EXISTS

Na jedné disketě nesmí být dva soubory se stejným jménem. Toto chybové hlášení se objeví, jestliže má být uložen na disketu soubor, jehož jméno se již na disketě vyskytuje.

64 FILE TYPE MISMATCH

Toto chybové hlášení se objeví v případě, že nesouhlasí typ žádaného souboru s typem uvedeným v direktoráři diskety.

65 NO BLOCK

Tato zpráva se objevuje v souvislosti s použitím příkazu B-A. Sděluje, že blok, který je přiřazován už byl předtím přiřazen. Parametry udávají volnou stopu a sektor s nejbližším vyšším číslen. Jsou-li parametry 0 jsou všechny bloky s vyšším číslem použity.

66 ILLEGAL TRACK AND SECTOR

Systém DOS se pokusil vstoupit do stopy nebo sektoru,který vzhledem k použitému formátu neexistuje. To může být problém při čtení pointeru pro následující blok.

67 ILLEGAL SYSTEM T OR S

Tento speciální chybový výpis upozorňuje na zákázanou systémovou stopu nebo sektor.

70 NO CHANNEL - není volný kanál

Toto hlášení dává zprávu o tom, že žádný kanál není volný. Všechny kanály jsou použity.

V systemu DOS může být současně otevřeno maximálně 5 sekvenčních souborů. Kanály v přímém přístupu mohou mít 6 otevřených souborů.

71 DIRECTORY ERROR

Nesouhlasí kontrolní součet mapy volných bloků. Problém souvisí buď s umístěním mapy volných bloků nebo byla mapa volných bloků přepsána v paměti DOSu. Problém se odstraní novou inicializací diskety, která znovu uloží mapu volných bloků do paměti. Některé aktivní soubory mohou být ukončeny opravným zásahem.

72 DISK FULL

Toto chybové hlášení se objeví, jsou-li použity všechny bloky na disketě nebo je-li zcela zaplněn direktorář. Při použití disketové jednotky 1541 se objeví hlášení DISK FULL, jsou-li ještě dva bloky volné, aby bylo možno uzavřít rozpracovaný soubor.

73 DOS MISMATCH (73, CBM DOS V2.6 1541)

DOS 1 a 2 jsou kompatibilní při čtení, ale ne při zápisu na disketu. Diskety mohou být čteny pomocí kteréhokoliv DOSu, ale disketa naformátovaná na jedné versi nedovolí zápis druhou versí, vzhledem k rozdílnému formátu. Tato chyba se objeví kdykoliv se systém pokusí zapisovat na disk naformátovaný v nekompatibilním formátu. (Existuje uživatelský program, který umožňuje přepis z jednoho formátu na druhý.) Toto hlášení se může rovněž objevit po zapnutí.

74 DRIVE NOT READY

Toto chybové hlášení se objeví, jestliže byl učiněn pokus o přístup na jednoduchou disketovou jednotku 1541, ve které není zasunuta disketa.

Příloha C

COMMODORE 64 BASIC

Tato příloha obsahuje seznam všech příkazů, instrukcí a funkcí, které můžete použít na počítači COMMODORE 64. Nebojte se všechny příkazy vyzkoušet, nemůžete tím svůj počítač nikterak poškodit. Nezapomeňte, že programovat se každý naučí pouze experimentováním.

Tato příloha je rozdělena do následujících částí:

1. Proměnné a operátory
- druhy a jména proměnných
- matematické a logické operátory

2. Příkazy
- povely užívané v přímém módu k tvoření, opravě, uschování a nahrávání programů.

3. Instrukce
- povely užívané v programech

4. Funkce
- řetězcové a numerické funkce
- funkce pro tisk

Proměnné

COMMODORE 64 používá tři druhy proměnných. Jsou číselné proměnné (reálná čísla), celočíselné proměnné (integer) a řetězcové proměnné (string).

Jméno každé proměnné může být složeno z jednoho nebo dvou znaků. Prvním znakem musí být vždy písmeno. Druhým znakem může být i číslice.

Proměnné typu integer se označují značkou % (procenta) umístěnou za jménem. Stringové (řetězcové) proměnné mají za sým jménem uvedenou značku **$** (dolar).

Příklady:

proměnná typu real	A, A5, BZ
proměnná typu integer	A%, A5%, BZ%
proměnná typu string	A$, A5$, BZ$

Pole jsou skupiny proměnných téhož typu, které používají stejného jména, které je doplněno indexem. Index je číslo v závorce udávající pořadí proměnné v daném poli.

Velikost pole se definuje příkazem DIM, za kterým následuje jméno proměnné a rozměry pole, které jsou udávány v závorce. Pole může být jedno-, dvou- i třírozměrné.

Příklady:

A(7), BZ%(11), A$(50), PT(20)

Poznánka:

Existují tři jména proměnných, které využívá sám systém a mají svůj stálý význam. Jsou to:

ST - obsahuje hodnotu odpovídající vsledku naposled provedené vstupní nebo výstupní operace.

TI - obsahuje informaci o čase, její hodnota se zvyšuje o 1 každou 1/60 sekundy počínaje okamžikem zapnutí počítače.

TI$ - je řetězec obsahující šestici číslic, které udávají počet hodin, minut a sekund od zapnutí počítače. Tuto proměnnou lze nastavit na libovolnou hodnotu a využívat k měření času.

Příklad nastavení hodin na 10 hodin, 15 minut, 30 sekund:

```
TI$ = "101530"
```

Operátory

Aritmetické operátory se označují těmito znaky:

+	sčítání
-	odečítání
*	násobení
/	dělení
↑	umocňování

Použijeme-li v jednom výrazu více aritmetických operátorů, je výraz řešen podle jejich priority. To znamená, že první jsou řešeny mocniny, následuje násobení a dělení a na konec je řešeno sčítání a odčítání. Chceme-li tuto prioritu změnit můžeme použít kulatých závorek. Výraz uvnitř závorek má pak nejvyšší prioritu.
Pro porovnání je možno použít následující operátory:

=	je roven
<	menší než
>	větší než
<=	menší nebo roven
>=	větší nebo roven
<>	není roven

Dale je možno použít také logických operátorů:

AND (a) - obě podmínky musí být současně splněny
OR (nebo) - alespoň jedna podmínka musí být splněna
NOT (ne) - podmínka neplatí

Tyto operátory se zpravidla používají, jsou-li při větvení programu uvažovány dvě podmínky. Například:

```
IF A=B AND C=D THEN 100
IF A=B OR C=D THEN 100
```

Příkazy

CONT (continue)

Tohoto příkazu se používá k opětnému spuštění programu, který byl zastaven klávesou RUN/STOP nebo instrukcí STOP. Po zadání příkazu CONT pokračuje program od místa, kde byl přerušen.

Program nelze spustit příkazem CONT, bylo-li přerušení vyvoláno chybou nebo byla-li během přerušení opravena nebo doplněna některá programová řádka. V tomto případě se vypíše chybové hlášení CAN'T CONTINUE ERROR.

LIST

Příkaze LIST je možno nechat vypsat na obrazovku programové řádky. Podle zadání se vypíše celý program nebo jenom některé řádky.

LIST	výpis celého programu
LIST 10-	výpis od řádky 10 do konce
LIST 10	výpis řádky 10
LIST -10	výpis od začátku do řádky 10
LIST 10-20	výpis řádků od 10 do 20

LOAD

Tento příkaz slouží k nahrání programu z magnetofonu nebo disketové jednotky do počítače.

Zadáme-li pouze příkaz LOAD bude do počítače nahrán nejbližší program uložený na kazetě. Zadáme-li za tímto příkazem jméno programu v uvozovkách, bude nahrán z kazety program, jehož jméno jsme uvedli.

Za jménem programu může následovat číslice nebo proměnná oddělená čárkou, která udává číslo zařízení, ze kterého má být program nahrán. Nezadáme-li číslo zařízení, nahrává COMMODORE 64 ze zařízení 1 t.j. z magnetofonu. Chceme-li nahrávat program z disketové jednotky, zadáme číslo zařízení 8.

LOAD nahrává následující program z kazety
LOAD "HELLO"nahrává program HELLO z kazety
LOAD A$ nahrává z kazety program, jehož jméno je uloženo v proměnné A$
LOAD "HELLO",8nahrává program HALLO z diskety
LOAD "*",8 nahrává následující program z diskety

Jestliže chceme nahrávat program, který není umístěn na počátku paměti (program ve strojovém kódu), musíme zadat ještě sekundární adresu.

LOAD "M.CODE",1,1nahraje z kazety program M.CODE uloží jej na správné místo v paměti.

NEW

Tento program vymaže z paměti počítače všechny BASICové programy i všechny používané proměnné. Proto používejte tento příkaz velmi opatrně.

Příkaz NEW je možno použít i v programu. Pak se paměť počítače po skončení tohoto programu vymaže.

RUN

Tento příkaz spustí program, který je uložen v paměti počítače. Počítač začne vykonávat program od řádky s nejnižším číslem. Zadáme-li za příkazem RUN ještě číslo, začne počítač program vykonávat od této řádky označené tímto číslem.

RUN	startuje program od řádky s nejnižším číslem
RUN 100	startuje program od řádky 100
RUN X	nedovolený zápis příkazu. Není dovoleno používat v příkazu proměnné. Neexistuje-li zadaná řádka počítač ohlásí chybu.

SAVE

Pomocí tohoto příkazu je možno uložit program na kazetu nebo na disketu. Při ukládání na kazetu musíme počítat s tím, že počítač nemá možnost zjistit, zda ukládáním programu nemaže jiný dříve uložený program. Proto je třeba s kazetami zacházet velmi opatrně.

Zadáme-li pouze příkaz SAVE, uloží se na kazetu program, který máme právě v paměti. Výhodnější je však za příkazem SAVE uvést v uvozovkách jméno programu, abychom jej později mohli na kazetě identifikovat. Používáme-li příkazu SAVE v programu, můžeme místo jména programu uvést řetězcovou proměnnou, která toto jméno obsahuje.

Za jménem ukládaného programu můžeme (v případě disketové jednotky musíme) zadat číslo zařízení, na které program chceme uložit (1 = magnetofon, 8 = disketová jednotka). Ukládáme-li program na kazetu, můžeme příkaz ještě doplnit o další parametr oddělený čárkou. Je-li tento parametr 1, uloží se za program na kazetu značka konce pásky, u které pak bude končit případné vyhledávání programu.

SAVE uloží program na kazetu beze jména
SAVE "HELLO"uloží program pod jménem HELLO na kazetu
SAVE A$ uloží program na kazetu pod jménem, které obsahuje proměnná A$
SAVE "HELLO",8uloží proram HELLO na disketu
SAVE "HELLO",1,1uloží program HELLO na kazetu a za program uloží značku konec pásky

VERIFY

Po zadání tohoto příkazu počítač porovná program v paměti s programem uloženým na kazetě nebo na disketě a ohlásí případné chyby.

Zadáme-li pouze příkaz VERIFY porovná počítač obsah paměti s prvním nalezeným programem na kazetě bez ohledu na jeho jméno. Zadáme-li však jméno programu, počítač nejprve program vyhledá a potom porovná. Za jménem může být uvedeno ještě číslo zařízení, o kterém platí stejná pravidla jako u příkazů LOAD a SAVE.

VERIFY porovná s pamětí následující program na kazetě
VERIFY "HELLO" vyhledá na kazetě program HELLO a porovná jej s pamětí
VERIFY "HELLO",8 porovná s pamětí program HELLO uložený na disketě

Instrukce

CLOSE

Instrukce CLOSE uzavírá soubory otevřené instrukcí OPEN. Parametr za instrukcí udává číslo souboru, který má být uzavřen.

CLOSE 2 uzavírá soubor číslo 2

CLR

Instrukce CLR vymaže všechny použité proměnné. Program zůstává zachován. Instrukce CLR se provádí automaticky při zadání příkazu RUN.

CMD

Instrukce CMD odesílá všechny výstupy na předem zvolené zařízení (tiskárnu, disketovou jednotku nebo magnetofon). Po otevření souboru příkazem OPEN je tedy možno použít příkazu LIST pro výpis na tiskárnu místo na obrazovku.
Za instrukcí CMD musí následovat číslo nebo proměnná, která určuje soubor, do kterého má být zapisováno.

```
OPEN 1,4    otevírá soubor č.1 pro zařízení č.4 (tiskárna)
CMD         přepíná výstup z obrazovky na tiskárnu
LIST        výpis programu na tiskárnu
PRINT#1     přepíná výstup zpět na obrazovku
CLOSE 1     uzavírá výstupní soubor
```

DATA

Touto instrukcí se označují řádky, ve kterých jsou zapsána data pro instrukci READ. Tyto řádky mohou obsahovat číselné i znakové údaje. Znaková data nemusí být uzavřena v uvozovkách za předpokladu, že neobsahují čárku, dvojtečku ani mezeru. Jednotlivá data se oddělují čárkami. Pokud mezi dvěma čárkami není nic uvedeno, čte toto instrukce READ jako číselnou hodnotu 0 nebo jako prázdný řetězec.

```
10 DATA 12,14.5,"HELLO,MOM",3.14,PART1
```

DEF FN

Používá-li se v programu pro výpočet častěji jeden vzorec je možno jej definovat jako uživatelskou funkci instrukcí DEF. Za instrukcí DEF nasleduje jméno funkce složené z písmen FN a libovolného dvouznakového výrazu. Pro tento dvouznakový výraz platí stejná pravidla jako pro jména proměnných. Za jménem je v závorce uvedeno jméno proměnné, pro kterou bude funkce počítána. Následuje znaménko = (je rovno) a matematické vyjádření funkce.

```
10 DEF FNA(X)=12*(34.75-x/.3)
20 PRINT FNA(7)
```

Funkce se počítá pro X=7 (číslo uvedené v závorce, proto výsledkem tohoto výrazu bude 137.

DIM

Touto instrukcí se dimenzuje (nastavuje velikost) pole pro indexované proměnné. Pokud má toto pole méně než 11 prvků není potřeba jej dimenzovat. Mějte na paměti, že každé pole zabírá místo v paměti, proto nedimenzujte pole větší než potřebujete.

Počet proměnných obsažených v poli zjistíme, násobíme-li mezi sebou všechny jeho dimenze a přičteme 1 (pro index 0).

```
10 DIM A$(40), B7(15), CC%(4,4,4)
          A$(x)          41 prvek
          B7(x)          16 prvků
          CC%(x,x,x)     125 prvků
```

Jednou instrukcí DIM je možno dimenzovat i několik polí. Nikdy nedimenzujte jedno pole dvakrát.

END

Instrukcí END ukončujeme běh programu. V případě, že není v programu tato instrukce použita končí program na řádku s nejvyšším číslem.

FOR....TO....STEP

Instrukce FOR vytváří spolu s instrukcí NEXT programovou smyčku. To znamená, že část programu mezi těmito dvěma instrukcemi je možno libovolně opakovat. Instrukce FOR se používá v následujícím tvaru:

FOR X=A TO B STEP C

X = jméno proměnné
A = počáteční hodnota proměnné
B = konečná hodnota proměnné
C = krok

Během zpracování části programu uvnitř smyčky se k počáteční hodnotě proměnné přičítá velikost kroku tak dlouho, pokud proměnná nedosáhne předepsané konečné hodnoty. Není-li velikost kroku uvedena počítá se automaticky s hodnotou 1. Velikost kroku může být udána i záporným nebo desetinným číslem.

Příklad:

```
10 FOR L=1 TO 10 STEP .1
20 PRINT L
30 NEXT L
```

GET

Instrukce GET přiřazuje proměnné, jejíž jméno je uvedeno za touto instrukcí, hodnotu právě stisknuté klávesy. Není-li stisknuta žádná klávesa přiřazuje se proměnné hodnota 0 nebo prázdný znak (podle typu proměnné). Použijeme-li instrukci GET pro číselnou proměnnou a bude-li v okamžiku provedení této instrukce stisknuta klávesa písmene, ohlásí počítač chybu.

Instrukce GET se velmi často používá v následující smyčce:

```
10 GET A$: IF A$="" THEN 10
```

Program zůstává ve smyčce v řádku 10, pokud je řetězcová proměnná A$ prázdná, to je do doby, než bude stisknuta nějaká klávesa.

GET

Pro použití této instrukce platí stejná pravidla jako pro použiti instrukce GET. Rozdílem je pouze to, že instrukce GET # čte jednotlivé znaky ze souboru otevřeného instrukcí OPEN, který může být uložen na libovolném zařízení.

```
GET #1,A$
```

přiřadí proměnné A$ hodnotu přečtenou ze souboru 1.

GOSUB

Tato instrukce funguje podobně jako instrukce GOTO, ale počítač si zapamatuje místo v programu, na kterém tuto instrukci dostal. Dojde-li pak program k instrukci RETURN, vrací se zpět na toto místo odkud byl skok proveden. Výhodné je tuto instrukci používat v případě, že jednu část programu budeme využívat častěji. Vyvoláváme ji pak touto instrukcí z různých míst programu.

```
20 GOSUB 800
```

Za instrukcí GOSUB je číslo řádky, kde má program pokračovat.

GOTO nebo GO TO

Po této instrukci pokračuje program na řádku udaném za instrukcí. Pokud v programu taková řádka neexistuje, program se zastaví a počítač ohlásí chybu.

```
20 GOTO 800
```

IF ... THEN

Tato instrukce slouží k větvení programu. Pokud je splněna podmínka uvedená za IF program pokračuje instrukcí uvedenou za THEN. Zde může býy uvedena libovolná BASICová instrukce. Pokud podmínka splněná není pokračuje program na následující řádce.

Jako podmínka může být použita proměnná nebo vzorec. Za splněnou se považuje podmínka, jejíž výsledek není roven 0. Nejčastěji se jako podmínka používá výraz obsahující některý z logických operátorů (=,<,>,AND,OR.NOT).

```
10 IF X>0 THEN END
```

INPUT

Pomocí instrukce INPUT můžeme během programu přiřadit některé proměnné libovolnou hodnotu. Postoupí-li program na instrukci INPUT, zastaví se, vypíše obrazovku otazník a čeká na vstup. Nyní můžeme z klávesnice zadat libovolnou hodnotu,

kterou počítač přiřadí po stisknutí klávesy RETURN proměnné uvedené za instrukcí INPUT.

Pomocí jedné instrukce INPUT lze přiřadit různé hodnoty několika proměnným. Jejich jména uvedená za instrukcí musí být oddělena čárkou.

Instrukci INPUT lze doplnit i komentářem, který se uvádí v uvozovkách před jménem proměnné.

```
10 INPUT"UDEJTE SVE JMENO:"; A$
20 PRINT"ZADEJTE DEN A MESIC";: INPUT A,B
```

INPUT

Pro instrukci INPUT # platí stejná pravidla jako pro instrukci INPUT. Vstupní data jsou však čtena ze souboru otevřeného instrukcí OPEN podobně jako v případě instrukce GET #.

LET

Instrukcí LET je v programu přiřazována proměnným jejich hodnota.

```
10 LET A=5
20 LET D$="HELLO"
```

Vzhledem k tomu, že používání instrukce LET je nepovinné, používáme zpravidla pro přiřazení hodnoty proměnným následující zápis:

```
10 A=5
20 D$="HELLO"
```

NEXT

Tato instrukce se používá výhradně ve spojení s instrukcí FOR...TO. Narazí-li počítač na instrukci NEXT otestuje, zda hodnota proměnné již dosáhla konečné hodnoty. Pokud ne, přičte se k proměnné hodnota kroku a program se vrací na řádku s instrukcí FOR. Pokud je již konečná hodnota dosažena, program pokračuje následující instrukcí.

Za instrukcí NEXT je možno uvádět jméno proměnné (resp. proměnných), kterých se instrukce týká. Používáme-li instrukce NEXT pro více proměnných, oddělujeme jejich jména čárkou.

Vždy je nutné uzavřít posledně otevřenou smyčku jako první.

```
10 FOR X=1 TO 100
20 FOR Y=1 TO 20
30 PRINT X,Y
40 NEXT Y,X
```

ON

Instrukce ON se používá k několikanásobnému větvení programu. Za ON následuje vzorec, podle jehož výsledku se program větví. Za vzorcem následuje instrukce GOTO a seznam řádků, na kterých má program pokračovat.
Je-li výsledek roven 1, pokračuje program na řádku uvedeném v seznamu na prvním místě. Je-li výsledek roven 2 pokračuje na řádku uvedeném na druhém místě atd. Je-li výsledek roven 0 nebo je-li vyšší než je počet řádků v seznamu, pokračuje program na následující instrukci za instrukcí ON.

```
10 INPUT X
20 ON X GOTO 10,20,30,40,50,60
```

OPEN

Pomocí instrukce OPEN je možno komikovat s různými zařízeními (magnetofonem, disketovou jednotkou, tiskárnou a pod.).

Instrukce OPEN se zadává v následujícím formátu:

OPEN s,z,a

kde s je číslo otevíraného souboru, které se může pohybovat v rozsahu od 0 do 255. Na druhém místě se zadává číslo zařízení (z), se kterým chceme spolupracovat, a na posledním místě se zadává sekundární adresa.

číslo zařízení se zadává podle následují tabulky:

0 obrazovka
1 magnetofon
4 tiskárna
8 disketová jednotka

Sekundární adresa může být při komunikaci s magnetofonem 0 (pro čtení), 1 (pro zápis) nebo 2 (pro zápis ukončený značkou konec pásky). U disketové jednotky se používá sekundární adresy pro volbu řídicího kanálu a sekundární adresa při komunikaci s tiskárnou udává zpravidla formát tisku.

```
10 OPEN 1,0otevírá obrazovku jako zařízení
20 OPEN 2,8,8,"D" ... otevírá disketovou jednotku pro čtení a
   vyhledává soubor D
30 OPEN 3,4otevírá tiskárnu
40 OPEN 4,8,15otevírá datový kanál disketové jednotky
```

Současně s instrukcí OPEN se používají instrukce CLOSE, CMD, GET#, INPUT#,a PRINT#.

POKE

Instrukce POKE se používá k ukládání hodnot přímo do paměti. Za touto instrukcí následuje adresa a čárkou oddělená hodnota, která má být uložena.
Adresa se může pohybovat v rozsahu od 0 do 65535 a ukládaná hodnota v rozsahu od 0 do 255.

```
10 POKE 53281,0
20 S=4096*13
30 POKE S+29,8
```

PRINT

Instrukce PRINT je první instrukcí, se kterou přijde každý začátečník do styku. Je to instrukce pro výpis na obrazovku. Za instrukcí PRINT může následovat znakový řetězec, jméno proměnné, funkce nebo oddělovací znaménko.

Použijeme-li jako oddělovacího znaménka středník, bude následující výpis bezprostředně následovat. Použijeme-li čárku, bude následující výpis začínat na dalším tabulátoru. Tabulátory dělí obrazovku na čtyři části po deseti znacích. Nepoužijeme-li žádného oddělovacího znaménka bude výpis pokračovat na další řádce.

```
10 PRINT "HELLO"
20 PRINT "HELLO", A$
30 PRINT A+B
40 PRINT J;
60 PRINT A,B,C,D
```

Pro formátování výpisu na obrazovku se rovněž používají funkce POS, SPC a TAB.

PRINT #

Instrukce PRINT # se používá pro zápis dat na různá periférní zařízení. Za touto instrukcí následuje číslo souboru, který byl otevřen instrukcí OPEN. Za číslem souboru je čárkou oddělen seznam výstupních dat. Středník a čárka mají v tomto případě stejný význam jako u instrukce PRINT.

```
100 PRINT#1,"OBSAH DAT";a%,B1,C$
```

Instrukci PRINT # nelze psát ve zkráceném tvaru (?)!

READ

Instrukce READ přiřazuje, které jsou za ní uvedeny, postupně hodnoty uvedené v řádcích označených instrukcí DATA. Při tvoření těchto řádek je třeba věnovat pozornost pořadí dat, aby nebyly číselným proměnným přiřazovány znakové řetězce. V tom případě je pak program zastaven a počítač ohlásí chybu TYPE MISMATCH ERROR.

REM (remark)

Komentář uvedený za instrukcí REM počítač při provádění programu ignoruje. Instrukcí REM se oddělují poznámky, které slouží pro lepší orientaci v programu při jeho úpravách.

RESTORE

Tato instrukce vrací ukazatele (pointer) ukazující na další prvek v řádku DATA zpět na začátek. Potom mohou být data čtena vícekrát.

RETURN

Tato instrukce uzavírá část programu (podprogram), do které se počítač dostal instrukcí GOSUB. Po instrukci RETURN se tedy vrací na instrukci následující po GOSUB. Pokud je použita instrukce RETURN bez instrukce GOSUB, program se zastaví a počítač ohlásí chybu RETURN WITHOUT GOSUB RETURN.

STOP

Instrukce STOP zastaví program a na obrazovku se vypíše hlášení BREAK IN xxx, kde xxx je číslo řádku, na kterém byl program zastaven. V programu lze pokračovat zadáním příkazu CONT.

SYS

Instrukce SYS slouží ke spouštění strojových programů. Za instrukcí se uvádí adresa, od které má být program spuštěn.... např.: SYS 4096

WAIT

Instrukce WAIT zastaví průběh programu, dokud určitá adresa nebude mít žádanou hodnotu. Instrukce se zadává ve tvaru:

WAIT X,Y,Z

Písmeno X představuje adresu, jejíž obsah je porovnáván s hodnotou Z (EXCLUSIV OR). Poté je proveden logický součin s hodnotou Y (AND). Je-li výsledek roven 0 je tato adresa testována znovu. Není-li výsledek roven 0 program pokračuje.

Není-li zadána hodnota Z, EXCLUSIV OR se neprovádí.

ABS (x)

Funkce ABS vypočte absolutní hodnotu čísla x. Výsledek je vždy kladný nebo 0.

ATN (x)

Funkce ATN vypočte hodnotu úhlu, jehož tangenta je x.

COS (x)

Funkce COS vypočte cosinus úhlu x. Parametr x musí být zadán v úhlové míře.

EXP (x)

Funkce EXP vypočte x-tou mocninu matematické konstanty e (2.71827183)

FN aa (x)

Funkce FN vypočte výsledek vztahu aa zadaného instrukcí DEF FN pro hodnotu x.

INT (x)

Funkce INT vypočte celočíselnou část čísla x. Výsledek je tedy vždy menší nebo roven číslu x.

LOG (x)

Funkce LOG vypočte přirozený logaritmus x o základu e (viz funkce EXP). Při převádění na dekadický logaritmus se dělí LOG(10).

PEEK (x)

Funkce PEEK přečte z paměti obsah paměťového místa x. Adresa x musí být v rozsahu od 0 do 65535. Přečtená hodnota paměťového místa je v rozsahu od 0 do 255. Funkce PEEK je opakem instrukce POKE.

RND (x)

Funkce RND vygeneruje náhodné číslo v rozsahu od 0 do 1. Výhodné je pro první použití funkce RND zadat x=-TI. Dále pak za x můžeme dosazovat 0 nebo kladné číslo. Negativní parametr x vytváří totiž nové výchozí číslo pro generátor. Tentýž negativní parametr generuje vždy tentýž sled náhodných čísel, proto je výhodné jako první parametr použít proměnné TI.

Náhodné celé číslo v rozsahu od X do Y získáme podle vzorce:

```
N = INT(RND(1)*Y)+X
```

SGN (x)

Tato funkce zjišťuje znaménko čísla x. Je-li x kladné je výsledná hodnota 1, je-li x záporné je výsledkem -1. V případě, že x je rovno 0, je i výsledek roven 0.

SIN (x)

Funkce SIN vypočte hodnotu sinus pro úhel x. Parametr x je nutno zadat v úhlové míře.

SQR (x)

Funkce SQR vypočte druhou odmocninu čísla x. Při záporném parametru x ohlásí počítač chybu ILLEGAL QUANTITY ERROR.

TAN (x)

Funkce TAN vypočte hodnotu tangens pro úhel x. Parametr x je nutno zadat v úhlové míře.

USR (x)

Funkce USR se používá ke startování strojových programů z BASICu. Parametrem x se do strojového programu může přenést libovolná hodnota. Po skončení strojového programu je v proměnné x uložena hodnota použitelná v BASICovém programu. Strojový program se startuje od adresy uložené na paměťových místech 785 a 786.

Stringové (řetězcové) funkce

ASC (x$)

Funkce ASC převádí první znak řetězce x$ na kód ASCII.

CHR$ (x)

Funkce CHR$ převádí ASCII kód x na znak.

LEFT$ (x$,x)

Funkce LEFT$ vytváří znakový řetězec, který obsahuje prvních x znaků řetězce x$.

LEN (x$)

Funkce LEN udává počet znaků, které obsahuje řetězec x$.

MID$ (x$,x,y)

Funkce MID$ vytváří znakový řetězec, který obsahuje y znaků řetězce x$ počínaje x-tým znakem.

RIGHT$ (x$,x,y)

Funkce RIGHT$ vytváří znakový řetězec, který obsahuje x posledních znaků řetězce x$.

STR$ (x)

Funkce STR$ převádí číslo x do řetězcového tvaru.

VAL (x$)

Funkce VAL převádí řetězec x$ do numerického tvaru. Je to funkce inversní k funkci STR$. Do tvaru čísla je převedeny samozřejmě pouze znaky číslic.

```
10 X=VAL("123.456")X=123.456
10 X=VAL("12A 13B")X=12
10 X=VAL("RIU 017")X=0
10 X=VAL("-1.23.15.67")X=-1.23
```

Ostatní funkce

FRE (x)

Funkce FRE udává počet volných bytů paměti (hodnota x je nepodstatná).

POS (x)

Funkce POS udává číslo sloupce (0-39), na kterém se nachází kursor (hodnota x je nepodstatná).

SPC (x)

Funkce POS se používá společně s instrukcí PRINT. Parametr x udává počet znaků (pozic na obrazovce), o které se má kursor posunout před dalším výpisem.

TAB (x)

Funkce TAB se rovněž používá společně s instrukcí PRINT. Parametr x udává číslo sloupce, ve kterém má začít následující výpis.

Příloha D

Zkratky klíčových slov BASICu

Většinu klíčových slov BASICu (příkazů, instrukcí a funkcí) je možno zadávat ve zkráceném tvaru. Zkratka pro příkaz PRINT je otazník (?). Ostatní zkratky se tvoří zadáním prvních dvou nebo tří znaků slova, přičemž poslední znak se zadává se stisknutou klávesou SHIFT. I když zadáváme program pomocí zkratek, vypíší se po příkazu LIST klíčová slova v plném znění.

V příloze D anglické verse návodu je uvedena tabulka, která obsahuje seznam zkratek klíčových slov BASICu.

Slovník výrazů použitých v tabulce:
command = příkaz
abbreviattion = zkratka
looks like this on screen = zobrazení na obrazovce
none = neexistuje

Příloha E

Znakové kódy obrazovky

V příloze E anglické verse návodu je uvedena tabulka, která obsahuje kódy všech znaků, které je možno zobrazit na obrazovce.

Uvedené kódy se používají jako druhý parametr instrukce POKE, chceme-li pomocí této instrukce zobrazit nějaký znak na obrazovce. Jako první parametr se udává adresa, jejíž hodnota se může pohybovat od 1024 do 2023 (viz příloha G).

Počítač COMMODORE 64 má k dispozici dvě sady znaků. Jedna sada obsahuje velká písmena a grafické symboly, druhá obsahuje malá a velká písmena. Tyto sady znaků nelze však používat současně.

Sady znaků lze mezi sebou v přímém módu přepínat současným stisknutím kláves C and SHIFT. Z BASICového programu lze sadu velkých písmen vyvolat instrukcí POKE 53272,21 a sada malých písmen instrukcí POKE 53272,23.

Kterýkoliv ze znaků lze znázornit také v inversním módu (t.j. vyměnit barvu písma a barvu podkladu). V tomto případě je nutno ke kódu uvedeném v tabulce přičíst číslo 128.

Chceme-li tedy například znázornit v levém horním rohu plný kruh, použijeme instrukce POKE 1024,81.

Každý znak na obrazovce může být napsán jinou barvou, proto také každému místu na obrazovce odpovídá jedno místo v paměti, kde je uchována informace o barvě. Každá barva má svůj číselný kód. Barevné kódy jednotlivých znaků obrazovky jsou uloženy od adresy 55296 do adresy 56295 (viz příloha G).

Slovník výrazů použitých v tabulce:

set = sada

POKE = parametr pro funkci POKE

Pozn.:není-li ve sloupci SET 2 uveden žádný znak, jsou obě sady shodné.

Příloha F

Kódy ASCII a CHR$

V příloze F anglické verse návodu je uvedena tabulka, která obsahuje seznam všech znaků a jejich ASCII kódů.

Pro převod znaků na kódy ASCII a naopak se používá funkce ASC a CHR$. Pomocí funkce CHR$ lze také vysílat na obrazovku nebo jakékoliv jiné zařízení netisknutelné znaky (RETURN, RVS ON, barvy apod.).

Pozn.:

kódy	192 až 223	jsou stejné jako kódy	96 až 127
kódy	224 až 254	jsou stejné jako kódy	160 až 190
kód	255	je stejný jako kód	126

Příloha G

Mapa paměti obrazovky a barev

V příloze G anglické verse návodu jsou diagramy, ve kterých jsou uvedeny adresy jednotlivých znakových pozic obrazovky a paměťových míst pro záznam barvy znaků.

Slovník výrazů použitých v tabulce:
screen memory map = mapa paměti obrazovky
color memory map = mapa paměti barev

Kódy barev, které lze instrukcí POKE zapsat do paměti barev znaků:

0	černá	8	oranžová
1	bílá	9	hnědá
2	červená	10	světle červená
3	tyrkisová	11	šedá 1
4	fialová	12	šedá 2
5	zelená	13	světle zelená
6	modrá	14	světle modrá
7	žlutá	15	šedá 3

Příloha H

Odvozené matematické funkce

V příloze H anglické verse návodu je uvedena tabulka, která obsahuje seznam matematických funkcí a jejich zápis v jazyce BASIC

Slovník výrazů použitých v tabulce:
funtion = funkce
Basic equivalent = ekvivalentní výraz v BASICu

Příloha I

Konektory pro vstupní a výstupní zařízení

V příloze I anglické verse návodu jsou uvedeny nákresy zapojení jednotlivých konektorů počítače COMMODORE 64.

Slovník výrazů použitých v tabulce:

Game I/O = řídicí vstupy pro hry
Cartridge slot = konektor pro zásuvný modul
Audio/Video = zvukový a obrazový výstup
Serial I/O (Disk/Printer) = sériový vstup a výstup (disketová jednotka, tiskárna)
Modular Output = modulovaný výstup pro TV
Cassette = konektor pro magnetofon
User Port = uživatelský vstup/výstup

Pin = kontakt
Type = signál
Note = poznámka
Control Port = řídicí vstup/výstup
JOY = joystick (pákový ovladač)
BUTTON = tlačítko
LP (light pen) = světelné pero
luminance = jas
GND (ground) = zem
audio out = zvukový výstup
audio in = zvukový vstup
video out = obrazový výstup
cassette motor = motor magnetofonu
cassette read = čtení z magnetofonu
cassette write = zápis na magnetofon
cassette sense = indikace funkce magnetofonu

Příloha J

Ukázky programů

V příloze J anglické verse návodu je uvedeno několik programů, na kterých si můžete procvičit získáné znalosti o programování v BASICu.

Pozn.: V posledním uvedeném programu (PIANO KEYBOARD) jsou použity následující klávesy:

programová řádka	klávesa
100	SHIFT CLR/HOME, CTRL 9, CTRL), SHIFT B
150	CRSR DOWN
240	CRSR UP
500	f1
510	f3
520	f5
530	f7
540	f2
550	f4
560	f6
570	f8
590	SHIFT CLR/HOME

Příloha K

Převod cizích programů na BASIC COMMODORE 64

Pokud vlastníte programy psané v jiné versi jazyka BASIC než je použita v počítači COMMODORE 64, je nutné je před spuštěním upravit. V následujících řádcích vám dáme několik typů, které vám mohou tyto úpravy usnadnit.

1. Vynechte všechny instrukce, které dimenzují délku řetězce. Instrukci DIM A$(i,j), která dimensuje pole pro i řetězců o délce j, je nutno změnit na DIM A$(i).

2. Pro spojování několika řetězců používají některé verse BASICu čárky (,) nebo apostrofu ('). Tyto znaménka je nutno zaměnit za znaménko plus (+), které je jedinným operátorem pro spojování řetězců v BASICu V2.0.

3. BASIC používaný počítačem COMMODORE 64 používá pro vytváření dílčích řetězců funkcí MID$, RIGHT$ a LEFT$. Některé verse BASICu používají výrazu A$ (i) pro načtení i-tého znaku z řetězce nebo A$ (i,j) pro vytvoření dílčího řetězce, který obsahuje znaky od i-té do j-té pozice řetězce A$.

```
A$(i) = X$      -> A$ = LEFT$(A$,i-1)+X$+MID$(A$,i+1)
A$(i,j) = X$    -> A$ = LEFT$(A$,i-1)+MID$(A$,j+1)
```

4. Má-li se několika proměnným přiřadit stejná hodnota, dovolují některé BASICy použít tvaru:

```
10 LET B=C=0
```

Vzhledem k tomu, že by COMMODORE 64 interpretoval druhé znaménko = jako logický operátor, přiřadil by proměnné C hodnotu 0 a proměnné B hodnotu -1. Proto je nutno napsat uvedený vztah takto:

```
10 B=0: C=0
```

5. U některých versí BASICu se oddělují instrukce použité na jedné řádce opačně skloněným znaménkem pro dělení. Verse BASICu V2.0 povoluje používat k oddělení instrukcí pouze dvojtečku (:).

6. Některé verse jazyka BASIC mají speciální funkce pro maticové operace. Tyto funkce je nutno při úpravě programu přepsat pomocí instrukce FOR....NEXT.

Příloha L

Popis chybových hlášení

Tato příloha obsahuje všechna chybová hlášení, která se mohou objevit na obrazovce a vysvětluje příčinu jejich vzniku.

BAD DATA
Z otevřeného souboru jsou místo numerických dat přijata data ve formě znaků (string).

BAD SUBSCRIPT
Index indexové proměnné přesáhl rozsah daný příkazem DIM.

BREAK
Program byl zastaven klávesou RUN/STOP.

CAN'T CONTINUE
Nelze vykonat příkaz CONT, protože program nebyl dosud spuštěn příkazem RUN nebo byla od posledního spušění opravena některá jeho řádka.

DEVICE NOT PRESENT
Výstupní zařízení není dostupné (připojeno, zapnuto) při použití příkazů CLOSE, CMD, OPEN, PRINT#, INPUT# nebo GET#.

DIVISION BY ZERO
V programu je matematický výraz, který obsahuje dělení číslem 0.

EXTRA IGNORED
Při vstupu pomocí příkazu INPUT bylo zadáno příliš mnoho znaků. Akceptovány budou pouze data zadaná jako první.

FILE NOT FOUND
Vyhledávaný soubor nebyl nalezen.

FILE NOT OPEN
Před použitím příkazu CLOSE, CMD, PRINT#, INPUT# nebo GET# nebyl soubor otevřen příkazem OPEN.

FILE OPEN
Soubor, který má být otevřen příkazem OPEN, je již otevřen.

FORMULA TO COMPLEX
Výraz je příliš složitý. Znakový řetězec je třeba rozdělit na dva menší. U matemetického výrazu je použito příliš mnoho závorek.

ILLEGAL DEVICE NUMBER
V programu je použit příkaz s nesprávným číslem vstupního nebo výstupního zařízení.

ILLEGAL DIRECT
Byl použit v přímém módu příkaz, který smí být použit pouze v módu programovém.

ILLEGAL QUANTITY
Argument funkce nebo příkazu leží mimo povolený rozsah.

LOAD
Program nebyl z kazety nebo z diskety správně nahrán.

MISSING FILE NAME
Bylo použito příkazu LOAD nebo SAVE pro sériový port (např. pro disketovou jednotku) bez udání jména souboru.

NEXT WITHOUT FOR
Byl použit příkaz NEXT bez příkazu FOR.

NOT INPUT FILE
Byl použit příkaz INPUT nebo GET pro čtení dat ze souboru, který je označen jako soubor výstupní.

NOT OUTPUT FILE
Byl použit příkaz PRINT pro zápis do souboru, který je označen jako soubor vstupní.

OUT OF DATA
Všechna data pro příkaz READ byla již vyčerpána.

OUT OF MEMORY
Kapacita paměti RAM je vyčerpána pro zápis programu i pro proměnné.

OVERFLOW
Byla překročena hranice číselného rozsahu, který je ± 1.70141884 E+38.

REDIM'D ARRAY
Byl použit příkaz DIM pro pole, které již bylo dimensováno. Pole může být dimensováno příkazem DIM nebo je dimensováno automaticky v okamžiku, kdy je poprvé použijeme.

REDO FROM START
Při použití příkazu INPUT byl místo čísla zadán znakový řetězec. Vstup je možno zadat znovu, aniž by se program přerušil.

RETURN WITHOUT GOSUB
Byl použit příkaz RETURN bez předešlého zadání příkazu GOSUB.

STRING TOO LONG
Použitý znakový řetězec má více než 255 znaků.

?SYNTAX ERROR
Použitý příkaz nemá odpovídající tvar (chybí závorky, uvozovky ap.) nebo je chybně zapsán (chybí písmeno ap.).

TOO MANY FILES
Příkazem OPEN je otevřeno více než 10 souborů.

TYPE MISMATCH
Při použití příkazu pro manipulaci se znakovými řetězci bylo jako argumentu použito číslo přesahující rozsah daného řetězce.

UNDEF'D FUNCTION
Byla použita uživatelská funkce bez přechozí definice příkazem DEF FN.

UNDEF'D STATEMENT
Bylo použito příkazu RUN, GOTO nebo GOSUB s udáním čísla řádku, který neexistuje.

VERIFY
Program uložený na kazetě nebo na disketě nesouhlasí s programem v paměti.

Příloha M

Nastavovací hodnoty not

V příloze M anglické verse návodu je uvedena tabulka, která obsahuje jména not, jejich kmitočty a hodnoty, které mají být zadány do příslušných registrů obvodu SID.

Slovník výrazů použitých v tabulce:
musical note = hudební nota
cycles/sec. = kmitočet
oscillator freq = obsah registru
high = horní (mocnější) byte
low = dolní byte

Nastavení filtrů

ADRESA	FUNKCE
54293	mezní frekvence (0 - 7)
54294	mezní frekvence (0 - 255)
54285	filter ozvěny (bit 4 - 7) filter pro hlas 3 (bit 2) filter pro hlas 2 (bit 1) filter pro hlas 1 (bit 0)
54296	HIGH-PASS filter (bit 6) BAND-PASS filter (bit 5) LOW-PASS filter (bit 4) nastavení hlasitosti (bit 0 - 3)

Příloha N

Literatura

V příloze N anglické verse návodu je uveden seznam doporučené literatury dostupné v zahraničí.

Příloha O

Mapa SPRITových registrů

V příloze O anglické verse návodu je uvedena tabulka, která obsahuje adresy a význam jednotlivých SPRITových registrů.

Slovník výrazů použitých v tabulce:

X component = souřadnice X
Y component = souřadnice Y
light pen = světelné pero

enable = zvolen
expand = rozšíření
screen character memory = znaková paměť obrazovky
interupt request = žádost o přerušení
interupt request masks = maska žádosti o přerušení
background sprite priority = priorita zobrazení SPRITu a pozadí
multicolor sprite select = výběr barvy SPRITu
sprite-sprite collision = kolize SPRITů
sprite-background collision = kolize SPRITu a pozadí

border color = barva rámečku
background color = barva pozadí
sprite color = barva SPRITu

Příloha Q

Nastavení a řízení zvuku na COMMODORE 64

V příloze Q anglické verse návodu je uvedena tabulka, která obsahuje adresy a význam jednotlivých registrů pro řízení zvuku.

Pro ovládání zvuku se používá BASICová instrukce POKE ve tvaru:

POKE adresa registru, obsah

Při nastavování registrů nesmíme zapomenout na to, že jeden byte může obsahovat i více parametrů zvuku. Musíme tedy nastavované hodnoty správně sečíst.

POKE 54272 + 12, 5*16 + 7

V uvedeném příkladu nastavujeme dobu nasazení a útlumu hlasu 2, kde:

54272	- základní adresa SIDu
12	- číslo registrů
5*16	- hodnota nasazení (5) posunutá o 4 bity vlevo (*16)
7	- hodnota dozvuku

Nezapomeňte, že je nutno také nastavit hlasitost instrukcí POKE 54296,A , kde A je hlasitost, která může být nastavena v rozsahu od 0 do 15 a platí pro všechny tři hlasy.

Slovník výrazů použitých v tabulce:

VOLUME CONTROL = řízení hlasitosti
TO PLAY A NOTE = hraná nota
HIGH FREQUENCY = horní kmitočet
LOW FREQUENCY = dolní kmitočet
WAVEFORM = tvar vlny (kmitu)
PULSE RATE = šířka pulsu (pouze u pravoúhleho pulsu)

ATTACK = nasazení
DECAY = útlum
SUSTAIN = výdrž
RELEASE = dozvuk

TRIANGLE = trojúhelníková (vlna)
SAWTOOTH = pilová (vlna)
PULSE = obdélníková (vlna)
NOISE = šum

Pozn.: Nastavení obalové křivky tónu (nasazení, útlum, výdrž a dozvuk) musí být nastaveny vždy dříve než tvar vlny.

Příloha R

6581 SOUND INTERFACE DEVICE (SID)

Specifikace obvodu

6581 SOUND INTERFACE DEVICE je integrovaný obvod obsahující tři nezávislé generátory zvuku. Tento integrovaný obvod dokáže splupracovat s mikroprocesorem 6510 a ostatními mikroprocesory této řady. SID může být velice přesně řízen v širokém rozsahu kmitočtů. Vyznačuje se velmi jednoduchým programovým ovládáním a širokou škálou zvuků s různým zabarvením a hlasitostí.

Popis:

3 tónové oscilátory s rozsahem 0 až 4 k Hz
4 různe tvary signálu (vln)
- trojúhelníkový
- pilový
- obdélníkový
- šumový

3 amplitudové modulátory s rozsahem 48 d B
3 generátory obalové křivky tónu
- exponenciální odezva
- poměr nasazení: 2ms až 8s
- poměr útlumu: 6ms až 24 s
- hladina výdrže: 0 až max. hlasitost
- poměr dozvuku: 6ms až 24s

Synchronizovaný oscilátor
Kruhová modulace

Dále je v příloze R anglické verse návodu je uvedena tabulka, která obsahuje adresy a význam jednotlivých registrů obvodu SID.

Příloha S

Příkazy a instrukce pro disketovou jednotku a tiskárnu

Následující příkazy a instrukce jsou určeny pro řízení disketové jednotky a tiskárny kompatibilní s COMMODORE 64.

CLOSE

typ: instrukce vstupu/výstupu

formát: CLOSE <číslo souboru>

Tato instrukce uzavírá datový soubor nebo otevřený komunikační kanál. Uvedené číslo souboru se musí shodovat s číslem souboru uvedeným za instrukceem OPEN, kterým byl soubor otevřen.

Pokud pracujeme se záznamovými zařízeními podobnými disketové jednotce , ukončí instrukce CLOSE přepis souboru. Když není vykonán, soubor je uložen na disketě nekompletní v nepoužitelné podobě. Instrukci CLOSE je možno u některých zařízení vynechat, ale snižuje se tím možnost otevření dalších souborů.

Další podrobnosti najdete v návodu pro přídavné zařízení.

Příklad:

```
10 CLOSE 1
20 CLOSE X
30 CLOSE 9*(1+J)
```

CMD

typ: instrukce vstupu/výstupu

formát: CMD <číslo souboru> [,řetězec;

Implicitně je určena obrazovka jako primární výstupní zařízení. Tato instrukce určí jako výstupní zařízení soubor, který byl otevřen instrukceem OPEN. Tímto zařízením může být disketová jednotka, tiskárna, magnetofon nebo jakékoliv jiné vstupní/výstupní zařízení.

Za instrukcí CMD je možno uvést řetězec, který je do zařízení předán společně se souborem a lze jej využít například jako titulku při výpisu na tiskárně.

Pokud je tato instrukce v činnosti, provádějí se výpisy PRINT nebo LIST na zvoleném zařízení ve stejném formátu jako na obrazovce.

Přepnutí výstupu zpět na obrazovku se provede instrukcí PRINT# a přenosu dat se ukončí instrukcí CLOSE.

Pozor! Jakákoliv systémová chyba (SYNTAX ERROR) ukončí přenos na zvolené zařízení a přepne výstup zpět na obrazovku.

Příklad:

```
OPEN 4,4: CMD 4: "TITLE": REM VYPIS PROGRAMU NA
TISKARNU
PRINT#4: CLOSE 4: REM UKONCENI VYPISU A UZAVRENI
KANALU

10 OPEN 8,1,4,"TEST": REM VYTVORENI SOUBORU SEQ
20 CMD 8: REM VYSTUP SOUBORU NA MAGNETOFON MISTO
   NA OBRAZOVKU
30 FOR L=1 TO 100
40 PRINT L: REM ZAPISUJE L NA KAZETU
50 NEXT
60 PRINT#1: REM PREPNUTI ZPET NA OBRAZOVKU
70 CLOSE 1: REM UZAVRENI SOUBORU
```

GET

typ: instrukce vstupu/výstupu

formát: CLOSE <číslo souboru>,<jméno proměnné>

Tato instrukce čte jednotlivé znaky ze vstupního zařízení nebo z určeného souboru otevřeného instrukcí OPEN. Znaky musí být v souboru odděleny čárkou nebo znakem RETURN (ASCII kód 13). Jestliže není žádný znak přijat přiřadí se numerické proměnné hodnota 0. Je-li použita řetězcová proměnná zůstane prázdná ("").

Použijeme-li jako vstupní zařízení TV obrazovku (zařízení #3), bude instrukce GET # číst jeden znak po druhém a posouvat kursor vpravo. Poslední znak v řádku bude nahrazen znakem RETURN.

Příklad:

```
5  GET# 1,A$
10 OPEN 1,3: GET# 1,Z7$
20 GET# 1,A,B,C$,D$
```

INPUT

typ: instrukce vstupu/výstupu

formát: INPUT# <číslo souboru>,<jméno proměnné>

Tato instrukce přiřadí uvedené proměnné hodnotu přečtenou z otevřeného souboru. Hodnoty jednotlivých proměnných musí být od sebe odděleny znakem RETURN (ASCII kód 13), čárkou (,), středníkem (;) nebo dvojtečkou (:). Jakým způsobem se oddělovací znaky zadávají je popsáno v popisu instrukce PRINT#.

Bude-li instrukcí INPUT# přiřazován číselné proměnné znakový řetězec, ohlásí počítač chybu BAD DATA. Touto instrukcí je možno řetězcové proměnné přiřadit maximálně 80 znaků. Bude-li délka řetězce delší než 80 znaků objeví se chybové hlášení STRING TOO LONG.

Použijeme-li jako vstupní zařízení obrazovku (zařízení č.3), budę proměnné uvedené za instrukcí INPUT# přiřazena ta řádka obrazovky, na které se nachází kursor. Tento se posune na následující řádku.

Příklad:

```
10 INPUT #1,A
20 INPUT #2,A$,B$
```

LOAD

typ: příkaz

formát: LOAD "<jméno souboru>",<číslo zařízení> [,<adresa>]

Tento příkaz slouží k nahrání programu z disketové jednotky do počítače.

Jako první parametr se zadává jméno souboru (programu), které musí být uvedeno v uvozovkách. Jako druhý parametr se zadává číslo zařízení, ze kterého budeme nahrávat. Disketová jednotka má standartně číslo 8.

Příkaz LOAD použitý v přímém módu uzavře všechny otevřené soubory a vymaže věechny proměnné. Použijeme-li však příkazu LOAD z programu zůstanou všechny proměnné beze změny a žádaný program se nahraje za stávající. Tímto způsobem lze spojovat různé programy.

Neznáme-li jméno programu uloženého na disketě můžeme je nahradit hvězdičkou ("*"). Počítač pak nahraje do paměti program, který je na disketě uložen na prvním místě.

Nenajde-li počítač soubor, jehož jméno je uvedeno za příkazem LOAD, na disketě, objeví se chybové hlášení ?FILE NOT FOUND.

Použijeme-li sekundární adresy 1, bude program umístěn v paměti na stejné místo, odkud byl na disketu uložen. Nezadáme-li sekundární adresu, bude program umístěn na počátek paměti vyhrazené pro BASIC.

Příklad:

LOAD A$,8 nahraje program, jehož jméno je v proměnné A$
LOAD "*",8 nahraje program, který je na disketě jako první
LOAD "FUN",8 nahrává program FUN z diskety
LOAD "GAME",8,1 nahraje program GAME a uloží jej na původní místo v paměti.

OPEN

typ: instrukce vstupu/výstupu

formát: OPEN <číslo souboru>,<číslo zařízení>,[<adresa>,
"<jméno souboru>,<typ>,<mód>"]

Instrukce OPEN otevírá komunikační kanály pro čtení a zápis na různé periférní zařízení.

Tato instrukce musí obsahovat číslo otevíraného souboru, které se může pohybovat v rozsahu od 0 do 255 (doporučujeme 0 až 127), a číslo zařízení, se kterým chceme spolupracovat. Zadané číslo souboru je pak nutné používat ve všech instrukcích, které se tohoto souboru týkají (CLOSE, CMD, INPUT#, GET# a PRINT#). čísla jednotlivých zařízení jsou uvedena v následující tabulce:

0 - obrazovka
1 - magnetofon
4 - tiskárna
8 - disketová jednotka

Sekundární adresa 1 až 14 je u disketové jednotky vyhrazena pro manipulaci s datovými soubory. Adresa 15 otevírá příkazový kanál pro systém DOS.

Jméno souboru může obsahovat maximálně 16 znaků.

Všechny soubory u kterých není uveden typ jsou zaznamenány jako soubory typu PROGRAM. Kromě toho lze na disketě uchovávat sekvenční a relativní soubory (SEQUENTIAL, RELATIVE).

U sekvenčních souborů lze zadat rovněž mód záznamu. Mohou to být soubory určené pouze pro zápis (W = write) nebo soubory určené pro čtení (R = read).

Všechna chybová hlášení, se ktarými se můžemem setkat při práci se soubory jsou uvedena v příloze B.

Příklad:

```
10 OPEN 2,8,4,"DISK-OUTPUT,SEQ,W" ... otevírá pro zápis
   sekvenční soubor na disketě
10 OPEN 50,0otevírá vstup z klávesnice
10 OPEN 130,4otevírá výstup na obrazovku
10 OPEN 1,2,0,CHR$(10) otevírá kanál pro RS 232
10 OPEN 1,4,0,"STRING"  přepíná tiskárnu na velká písmena
10 OPEN 1,4,7,"STRING"  přepíná tiskárnu na malá písmena
10 OPEN 1,8,15,"COMMAND" otevírá příkazový kanál pro
   disketovou jednotku
```

PRINT #

typ: instrukce vstupu/výstupu

formát: PRINT # <číslo souboru> [<proměnná>]
[<,/;><proměnná>]...

Instrukce PRINT # se používá pro zápis dat do souborů otevřených instrukcí OPEN.

Za číslem souboru je čárkou oddělen seznam výstupních dat. Středník a čárka mají v tomto případě stejný význam jako u instrukce PRINT. Takto uložená data lze číst instrukcí INPUT# včetně oddělovacích znamének. Tato oddělovací znaménka mohou být zadána i ve formě ASCII kódu.

Příklad:

```
10 OPEN 1,8,4,"MY LIFE"
20 R$ = CHR$(13): REM RETURN -> R$
30 PRINT# 1,1;r$;2;r$;3;r$;4;r$;5: REM KAZDA CIFRA NA
   NOVE RADCE
40 PRINT# 1,6: REM CARKA = TABULATOR
50 CLOSE 1
```

SAVE

typ: příkaz

formát: SAVE "<jméno souboru>",<číslo zařízení> [,<adresa>]

Pomocí tohoto příkazu je možno uložit program na disketu jako soubor typu PROGRAM. Místo jména souboru lze použít řetězcové proměnné, ve které je jméno souboru uloženo.

Příklad:

SAVE "FUN",8uloží program FUN na disketu
SAVE A$,8uloží na disketu program pod jménem, které obsahuje proměnná A$

SAVE a REPLACE (nahrazení)

typ: příkaz

formát: SAVE "@0:<jméno souboru>",<číslo zařízení>

Tato verse příkazu SAVE dovoluje přepsat již existující soubor na disketě.

Pokud bychom se pokusili uložit na disketu příkazem SAVE dva programy se stejným jménem, vypíše počítač chybu. Použijeme-li ale tuto versi příkazu SAVE bude původně uložený program přepsán novým.

Příklad:

```
SAVE "@0: PROGRAM",8
```

VERIFY

typ: příkaz

formát: VERIFY "<jméno souboru>",<číslo zařízení>

Po zadání tohoto příkazu počítač porovná program v paměti s programem uloženým na disketě a ohlásí případné chyby. Tento příkaz používáme pro ověření správnosti záznamu příkazem SAVE. Lze jej použít jak v přímém módu tak v programu.

Místo jména souboru můžeme zadat jméno řetězcové proměnné, která obsahuje jméno souboru, nebo prázdný řetězec ("").

Příklad:

```
9000SAVE "ME",8
9010VERIFY "ME",8
```

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