

129

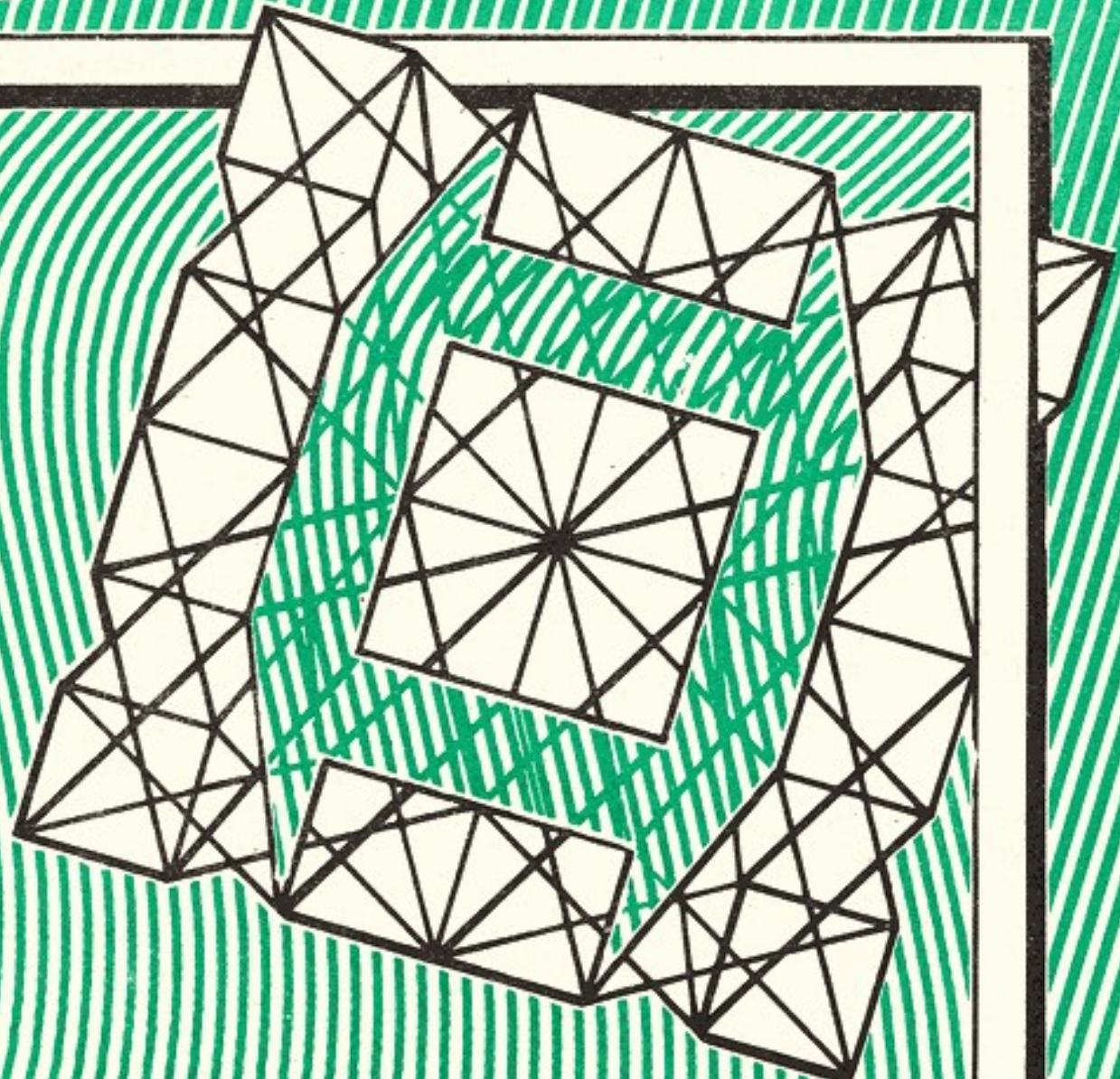


602

ATARI

4

88



Uživatelské programy

ATMAS II a ATMAS III

Program poskytuje široké možnosti pro práci s assemblerem mikroprocesoru 6502.

Nářízení programu

Po zavedení první části programu pomocí START + OPTION stlačíme A. Jakmile se zavede druhá část programu, stiskneme B a dostaneme se do režimu EDITORU, který je vyhrazen pro práci s texty.

V systému TURBO zavádime program obvyklým způsobem.

EDITOR

Na obrazovce se v levém horním rohu nachází čítač, jehož hodnota závisí na množství vloženého textu a okamžité pozici kurzoru.

Druhý zleva je čítač udávající velikost volné paměti, rezervované pro text.

Vpravo je ukazatel velikosti kopirovacího registru.

Zcela vpravo je místo vyhrazené pro hlášení systému (OK, E-Error...)

POHYB KURZORU

Pohyb je ovládán podobně jako v BASICu, klávesami TAB, šipky vlevo, vpravo, nahoru, dolů. Navíc přibývají pouze povely:

CTRL D - kurzor se přemístí na konec textu

CTRL E - kurzor se přesune na začátek textu

Po celou dobu je kurzor v INSERT modu, tj. posunuje před svou pozici celý text směrem doprava vždy o jedno místo a tím ne-překrývá znaky.

POVELY

Povely EDITORU jsou zadávány současným stlačením klávesy <Control> a potřebné další. V dalším textu je tato skutečnost zapsána jako CTRL X, kde X je název další klávesy.

CTRL G - opakování příkazového řádku

CTRL F - uzavření kopirovacího registru

CTRL J - vložení obsahu kopirovacího registru do textu

CTRL K - vymazání kopirovacího registru

CTRL R - otevření kopirovacího registru

CTRL P - přechod do režimu MONITORU

CTRL T - zviditelnění kontrolních znaků

CTRL V - přepínání do modu 1/2 / celá stránka textu

CTRL Y - překlad zdrojového textu do strojového kódu. Po ukončení překladu se na obrazovce objeví hlášení o úspěšnosti překladu. Návrat do režimu EDITORU se provede stiskem libovolné klávesy. Pokud nebyl překlad úspěšný, zastavi se kurzor v textu na místě první nalezené chyby.

PŘÍKAZY

Příkazy se zapisují na dolní, příkazový řádek.

Po stisku ESC se na začátku řádku objeví znak \$, za nějž se zapisuje příslušný příkaz a případný operand bez závorek. Příkaz se nakonec uzavře opět stiskem ESC a mohou následovat další příkazy, nebo vykonání příkazového řádku stiskem ESC.

Je-li třeba příkaz zapsaný v příkazovém řádku opakovat, stačí stisknout CTRL G.

- B - kurzor o jednu pozici zpět
- F - kurzor o jednu pozici vpřed
- D - vymazání znaku vlevo od kurzoru
- T - vymazání znaku vpravo od kurzoru
- E - vymazání kopirovacího registru
- G - vložení obsahu kopir. registru do textu (jako CTRL J)
- H(hex) - vložení ASCII znaku do textu. V přík. řádku bez závorek
- I(řetěz) - vložení řetězce znaků do textu
- S(řetěz) - hledání řetězce od pozice kurzoru
- J - opakování přík. řádku (jako CTRL G, ale programově)
- K - vymazání celého textu
- M - návrat do DOS
- U - start programu na \$ A800
 - nastavení tabulační šířky (0-9)
- L1 - výpis textu (zdrojového programu) na tiskárnu
- WC: - uložení textu na mgf.
- RC: - čtení textu z mgf. a jeho uložení od pozice kurzoru

PSEUDOINSTRUKCE

ORG - definice počáteční adresy. Program by definováním adresy měl vždy začínat.

EQU - každou adresu nebo proměnnou instrukci je možné nahradit symbolem např. LDA #WORL a WORL pak kdekoliv v programu pomocí EQZ definovat

EPZ - totéž jako EQU, ale pro nultou stránku paměti

DFB - slouží pro vložení byte do určitého místa programu, je-li třeba vložit byte a nikoliv instrukci

DFW - vložení slova. Totéž jako DFB, ale umožňuje vložit dva byte

ASC - vložení ASCII textu, umožňuje vložit do programu libovolně dlouhý řetězec

MACRO - vytvoření makroinstrukce. Často se opakující programové sekvence mohou být pomocí MACRO definovány a v programu se pak používá pouze symbolický název makroinstrukce

MEND - ukončení makroinstrukce

;(středník) - odděluje programové instrukce od komentářů používaných pro dokumentaci programu

Všechny číselné hodnoty mohou být jak v hexadecimálním, tak i v desítkovém vyjádření. Mezi desítkovou číselnou hodnotou nebo symbolickým názvem a instrukcí musí být mezera. Před číselnou hodnotou v hexa tvaru musí být znak \$.

M O N I T O R

PŘÍKAZY

V závorkách textu jsou uvedeny tvary příkazů.

Na rozdíl od EDITORU je nutno všechna čísla zadávat výhradně v hexadecimálním tvaru !

D - disassembler. V prvním sloupci se nachází adresa, v dalších pak kódy instrukcí, symbolická vyjádření instrukci s absolutními hodnotami operantu a příslušejícími ASCII znaky. Zpět do MONITORU stlačením RESET

M - prohlížení, zobrazování paměti (od, do, druh zařízení)

C - změny v paměti (od). Zpět stlačením RESET

F - FILL, naplnění paměťového bloku zadanou hodnotou (od, do, hodnota)

B - přesun bloku paměti (od, do, velikost)

S - SAVE, uložení bloku paměti na periferní zařízení (od, do, inicializační adresa, druh zařízení a příp. jméno souboru)

L - LOAD, zavedení dat ze zařízení do paměti (druh zařízení příp. jméno souboru)

G - GOTO, start strojového programu (od)

E - návrat do EDITORU, mimo to se lze z jakéhokoliv vykonávaného příkazu vrátit také stlačením RESET.

R O Z D Ě L E N Ī P A M Ě T I

Deci Hexa (\$)

0-1151 0-47F - Tato část paměti není vhodná pro uložení programu. Při dobré znalosti systému ji lze využít pro vlastní proměnné

1152-1535 480-5FF - Prostor využitelný pouze nebude-li využíván BASIC

1536-1791 600-6FF - Libovolně využitelná část paměti, vhodná zejména pro programy, data a proměnné

1792-2463 700-99F - Je-li připojena disk. jednotka, je zde uložen DOS, jinak zaváděcí program BLC. V případě potřeby lze příkazem F, MONITORU tuto oblast vymazat a využít ji pro své programy

2464-10239 9A0-27FF - Prostor libovolně využitelný, vhodný pro strojové programy, data i proměnné

10240-19711 2800-4CFF - část paměti, kde je uložen ATMAS II. Většina zásahů do této oblasti končí havárií nebo špatnou funkcí programu!

19712-26175 4000-63FF - Libovolně využitelný prostor

26176-43585 6400-AA41 - Do této části ukládá ATMAS text zapsaný v EDITORU. Při dobré znalosti systému lze její vyšší část využít

43586-48190 AA42-BC3E - Další část paměti využívaná ATMASem, vhodnější je do této oblasti nezasahovat

48191-49151 BC3F-BFFF - Obrazová paměť, paměť ROM-BASIC je zcela odstavena

49152-65535 C000-FFFF - Paměť ROM

P R O G R A M O V Á N ī

Plocha obrazovky EDITORU je rozdělena na tři pomyslné sloupce. Do prvního se vždy zapisuje návěst (label). Je to symbolické vyjádření čísla, ať už adresy nebo operandu. Návěst může obsahovat až osm znaků, avšak pouze číslic nebo písmen bez mezer.

Po stisku tabulátoru (i v případě, že návěst chybí) lze zapisovat instrukce mikroprocesoru a případně adresy nebo operandy instrukci v absolutní i symbolické podobě, nebo klíčová slova EDITORU.

Třetí sloupec začíná středníkem. Zde se ukládají poznámky k textu nebo popisy kroků. Třetí sloupec, stejně jako první, není povinný. Následuje RETURN, jehož stisk rádek ukončí.

První rádek programu by měl vždy začinat klíčovým slovem ORG, které definuje počátek prostoru pro ukládání přeloženého programu:

```
ORG $600; hexa vyjádření
```

```
ORG 1536; deci vyjádření
```

připadně

ORG STRT; symbolické vyjádření. STRT pak musí být v programu ještě definováno. Definování může být na libovolném místě programu:

```
STRT EQU $300; hexa
```

```
QWERT EQU 120; deci
```

Pro vložení byte, který není instrukčním kódem, se používá:

```
DFB "A"; vloží ATASII hodnotu znaku A
```

```
DFB 65; hodnotu lze vložit i přímo
```

a pro vložení slova, které není instrukcí strojového programu:

```
DWF "AB"; jako u byte
```

```
DWF 6566
```

Pokud je potřeba, aby program obsahoval text, využívá se pseudoinstrukce ASC ve tvaru:

```
ASC "Znění potřebného textu"
```

Pozn.red.: Z tiskových důvodů je v předcházejícím odstavci textu pro značku apostrofu (SHIFT 7) použita značka uvozovek.

Obdobou BASIC příkazů GOSUB a RETURN jsou v ATMASu příkazy MACRO a MEND. Nejprve se uvede symbolický název podprogramu, pak pseudoinstrukce, dále instrukce a na závěr opět pseudoinstrukce. MACRO může být definováno kdekoli v programu. Při volání sekvence postačí vložit do programu symbolický název:

```
LJK MACRO; začátek podprogramu s názvem LJK
```

```
... ; instrukce
```

```
... ; instrukce
```

```
...
```

```
MEND ; konec podprogramu s názvem LJK
```

L A D Ě N ī P R O G R A M U

Tento odstavec nepatří k nutným informacím, pomůže však zejména začínajícím zájemcům, protože obsahuje některé drobné rady pro programování.

Pro odláďování programů existuje kromě programátorské zkušenosti a intuice jen jediný prostředek: instrukce BRK, vložená

do programu. BRK zastavi vykonávaný program a na obrazovku se vypíši všechny registry mikroprocesoru. Je to vlastně vložená programová zarázka, podobně jako v BASICu STOP.

Pokud při překladu hlásí překladač trvale chybu a už si nevíme rady, je možno zkousit zviditelnit kontrolní znaky (TAB, RETURN) stlačením CTRL-T a ověřit si zdali některý z nich nechybi.

Pokud potřebujeme do programu vložit text nebo instrukci, která bude často volána, je zejména v začátcích sestavování programu výhodné do sloupce návěšt zapsat u začátku příslušného podprogramu vhodnou zkratku a tu pak v případě potřeby volat. Nemusíme si tím pamatovat či vyhledávat potřebnou adresu, která se většinou při tvorbě programu mění.

A T M A S III - doplňky

Tento program obsahuje všechny předchozí příkazy. Navíc disponuje následujicimi instrukcemi nebo příkazy:

RT (name) - TURBO čtení zdrojového textu

WT (name) - TURBO zápis zdrojového textu

Mezi "name", kterým je libovolný název a RT nebo WT musí být zachována mezera jednoho znaku

RM 0-4 Ramdisk čtení zdr.textu

WM 0-4 Ramdisk zápis zdr.textu

SELECT - skok do BASICu

\$27BB - adresa restartu programu

; - není nutno používat pro oddělení komentářů

Ze zápisníku programátora

TŘÍDÍCÍ KOMB AJN TRIKO 1.0

J. Skála, S. Vohnický

Velmi často se setkáváme s potřebou abecedně či čiselně seřadit určitou danou posloupnost údajů. S touto zdlouhavou prací nám může pomoci program TRIKO 1.0. I když by se tak někomu mohlo podle názvu zdát, nejedná se o program s tematikou oděvních výrobků. Třídici kombajn TRIKO nám pomůže abecedně, případně i čiselně setřidit rozsáhlou posloupnost údajů v poměrně velmi krátkém čase. Posloupnost 162 položek o průměrné délce 12 znaků dokáže za použiti ATARI BASICu setřidit za 95 sekund. Při použiti TURBOBASICu XL samozřejmě ještě daleko rychleji. Lvi podíl na této rychlosti setřídění má jednak použity algoritmus SHELLSORT, jednak způsob ukládání údajů do paměti.

Shellovo třídění je nejrychlejším z jednoduchých třídicích algoritmů, mezi které patří "přímé zatřídování", třídění "přímým výběrem" a třídění "přímou výměnou" (bublinové třídění). Existují samozřejmě ještě rychlejší třídici algoritmy, avšak ty potřebují ke své práci paměť navíc (jako zásobník). Mimo Shellova třídění jsou všechny jednoduché třídici algoritmy vhodné hlavně pro "dotřídění" souboru, případně pro setřídění krátkých posloupnosti. Jsou velmi neefektivní v případech, kdy je třeba jednotlivé členy

posloupnosti zařazovat do velké vzdálenosti od jejich stávající pozice.

Tuto nevýhodu odstraňuje právě Shellovo třídění. V prvním průchodu porovnává a v případě nutnosti provádí i vzájemnou výměnu prvků o velkých vzdálenostech. V dalších průchodech tuto počáteční vzdálenost postupně zmenšuje, a v posledním průchodu pracuje s dvojicemi sousedících prvků. Poslední průchod je tedy klasickým bublinovým tříděním (dotříděním předtříděné posloupnosti).

PŘÍKLAD:

Mějme nesetříděnou posloupnost čísel:

44 55 12 42 94 18 06 67.

V prvním průchodu stanovíme jako počáteční vzdálenost prvků. 4. V dalším průchodu ji zmenšíme na polovinu, tedy 2, a v posledním průchodu budeme pracovat se vzdáleností 1.

1.průchod

44		94	
18		55	
06		12	
42		67	

Pořadí je nyní: 44 18 06 42 94 55 12 67

2.průchod

06	44	94	12
18	42	55	67
06	44	94	12
18	42	55	67
06	12	44	94
18	42	55	67

Pořadí: 06 18 12 42 44 55 94 67

3.průchod

06	18	12	42	44	55	94	67
06	12	18	42	44	55	94	67
.
.

Konečné pořadí: 06 12 18 42 44 55 67 94.

V příkladu jsme použili jako počáteční vzdálenost 4 a tu jsme pak stále půlili, až jsme došli ke vzdálenosti 1, tedy posloupnost vzdáleností byla geometrická, s kvocientem 2. V některých příručkách programování se doporučují i jiné posloupnosti vzdáleností. V programu TRIKO je použita posloupnost:

1,4,13,40,121,...

tj. vzdálenosti se počítají podle vztahu:

$$v(i+1) = 3 \cdot v(i-1) + 1, \quad v(1) = 1.$$

Vice informací o Shellově třídění a dalších třidicích algoritmech lze najít v publikaci: Niklaus Wirth - ALGORITMY A ŠTRUKTÚRY ÚDAJOV, která v tomto roce vyšla ve slovenském nakladatelství ALFA.

Druhým faktorem, který se významně podílí na rychlosti třídění programu TRIKO, je způsob ukládání položek. Jednotlivé

položky se ukládají do textového řetězce těsně za sebou, zároveň s informací o délce (počtu znaků) položky. Nevznikají tedy mezery mezi jednotlivými položkami, jak je to běžné při ukládání s pevně nastavenou délkou položky, a významně se šetří paměť. Část této ušetřené paměti je v zájmu jednoduššího prohledávání řetězce věnována na uložení informaci o poloze počátku položky v řetězci. V programu je k tomuto účelu vyhrazeno pole A(i). Zde jsou uloženy "ukazatele" na počátky položek v tom pořadí, v jakém byly položky vkládány. Při vlastním třídění pak nejsou zaměňovány mezi sebou přímo položky v textovém řetězci, ale pouze ukazatele na počátky příslušných položek v poli A(i). To výrazně šetří čas při vlastním třídění. Je totiž časově výhodnější přesouvat v paměti pouze 6 byte (tolik v paměti zabírá jedna hodnota v poli A(i)), než například 40.

Práci s programem TRIKO 1.0 není třeba obširně popisovat. Ihned po spuštění se zobrazí nabídka (menu). Stiskem určité klávesy zvolíme požadovanou činnost. Volbu LOAD si důkladně rozmyslime, protože nahráním nového datového souboru automaticky přicházíme o data v paměti počítače. Stejně tak při ukončení práce programu. Při stisku klávesy BREAK dojde k přerušení běhu programu, data jsou ve většině případů zachována. Do programu se nejlépe vrátíme příkazem GOTO 80.

Program třídí abecedně (dle kodu ATASCII) do vzestupné posloupnosti. Potřebujeme-li posloupnost sestupnou, změníme směr nerovnosti na řádku 340. Budeme-li chtít třídit podle číselných údajů, musíme zachovat stejný počet míst u všech čísel. Doplníme proto nižší čísla do plného počtu míst zleva nulami.

Přestože je program koncipován ke třídění jednorozměrných vektorů, lze jím setřídit i pole dvojrozměrná. Při vkládání položek vyhradíme první počet znaků kličovému údaji (podle kterého chceme třídit) a zbytek znaků, které máme v položce k dispozici, rezervujeme na druhý údaj.

Příklad:

Máme seznam jmen určitého počtu osob a jejich stáří. Tyto osoby chceme setřídit do vzestupné posloupnosti podle jejich věku. Pro věk tedy vyhradíme první tři znaky položky a hned za něj budeme uvádět jména:

030	Martin,
015	František,
101	Alois,
007	Lukáš, atd.

Po setřídění dostaneme vzestupnou posloupnost uspořádanou podle věku a teprve u osob se stejným věkem bude přihlédnuto k abecednímu pořadí.

Doufáme, že předkládaný program TRIKO 1.0, jehož listing najdete dále, vám bude alespoň trochu užitečný, a že programátorům-začátečníkům pomůže objasnit některé partie teorie a praxe třídění dat a tvorby programů.

Výpis programu je uveden s kontrolními kódy (viz Zpravodaj č.3)
J. Skála

P.S. Blížší podrobnosti k programu najdou zájemci v komentáři.

```

RI 10 REM ****
KH 20 REM *      TRIdici Kombajn   *
HY 30 REM *      verze 1.0       *
VA 40 REM *      1988 Vohnicky, Skala   *
EY 50 REM *      ATARI klub 602   *
RN 60 REM ****
PA 70 OPEN #1,4,0,"K":SETCOLOR 1,0,8:SET
      COLOR 2,0,0
MG 80 ? CHR$(125):POKE 752,1
GH 90 ? "*****"
NM 100 ? "* T R I D I C I   K O M B A J
      N   "
ZG 110 ? "*****"
IT 120 POSITION 15,8:? "1.VKLADANI":POSIT
      ION 15,10:? "2.TRIDENI":POSITION 15,12
      :? "3.TISK":POSITION 15,14:? "4.SAVE"
GP 130 POSITION 15,16:? "5.LOAD":POSITION
      15,18:? "6.KONEC"
DE 140 GET #1,SEL:IF SEL<49 OR SEL>55 THE
      N 140
GH 150 IF SEL<50 OR SEL>52 THEN 170
IJ 160 IF Q<=0 THEN GOSUB 940:GOTO 80
AF 170 ON (SEL-48) GOTO 180,290,390,510,6
      10,760
OJ 180 ? CHR$(125):IF N<=0 THEN GOSUB 800
      :GOSUB 850
XP 190 ? :? "!!!!PRI VKLADANI POLOZEK POUZ
      IVEJ JEDENDRUH PISMEN. BUD JEN MALA,
      NEBO JENVELKA!!!";
WG 200 ? "MAX. DELKA POLOZKY JE: ";N;" ZN
      ."
LM 210 POCPOL=AA-Q:IF POCPOL<=0 THEN 870
IW 220 POKE 752,0
KE 230 ? :? "ZBYVA ";POCPOL;" VOLNYCH POL
      OZEK!"
OI 240 ? :? " ZADEJ POLOZKU KE TRIDENI: "
DH 250 AS="":INPUT AS:IF AS="" THEN 80
PT 260 B=LEN(Z$):C=LEN(AS):Z$(B+1,B+1)=CH
      RS(C)
IS 270 Z$(LEN(Z$)+1)=AS:Q=Q+1:A(Q)=B+1
MS 280 GOTO 210
CM 290 IF Q<2 THEN ? CHR$(125):POSITION 8
      ,5:? "MALO POLOZEK KE TRIDENI!":POSITI
      ON 2,22:GOSUB 910:GOTO 80
QC 300 ? CHR$(125):POSITION 14,5:? "T R I
      D I M !":P=0
VC 310 S(1)=1:FOR L=1 TO 9:S(L+1)=S(L)*3+
      1:NEXT L
RT 320 P=P+1:IF S(P+2)<Q THEN 320
NW 330 FOR I=P TO 1 STEP -1:S=S(I):FOR J=
      S+1 TO Q:L-S:A=A(J):B=A(L)

```

MQ 340 IF Z\$(A+1,A+ASC(Z\$(A,A)))>Z\$(B+1,B
 +ASC(Z\$(B,B))) THEN 360
 QV 350 POSITION 10,8:? L;" ":POSITION 3
 0,8:? S;" ":A(L+S)=B:L=L-S:IF L>0
 THEN B=A(L):GOTO 340
 OH 360 A(L+S)=A:NEXT J:NEXT I:POSITION 5,
 5:POSITION 11,5:? "*****SETRIDENO!*****"
 CN 370 POSITION 6,8:? "POCET TRIDENYCH PO
 LOZEK: ";Q
 SR 380 GOSUB 910:GOTO 80
 VL 390 ? CHR\$(125):?:? " VYSTUPNI ZA
 RIZENI...(E/P):";:INPUT WS
 DT 400 IF WS<>"E" AND WS<>"P" THEN 390
 PY 410 IF WS="P" THEN X=55:? ?:? "Zadej na
 zev souboru (Max.40 znaku)":?:? " ";:I
 NPUT A\$:GOTO 430
 XW 420 X=21
 ZF 430 CLOSE #2:TRAP 890:OPEN #2,8,0,WS:T
 RAP 40000:POKE 752,1:SETCOLOR 1,0,8:SE
 TCOLOR 2,0,0
 NX 440 ? #2;A\$:#2
 GH 450 FOR L=1 TO Q
 WS 460 A=A(L):A\$=Z\$(A+1,A+ASC(Z\$(A,A)))
 VH 470 ? #2;L,A\$
 VS 480 IF L/X<>INT(L/X) AND Q>L THEN 500
 SA 490 ? :GOSUB 910
 NR 500 NEXT L:CLOSE #2:GOTO 80
 TR 510 ? CHR\$(125):A\$="":? " VYSTUPNI
 ZARIZENI (D/C/T/M): "
 DP 520 POSITION 2,5:? " ";:INPUT
 A\$:IF A\$="" THEN 80
 DI 530 IF A\$(1,1)="D" OR A\$(1,1)="C" OR A
 \$(1,1)="T" OR A\$(1,1)="M" THEN 560
 VO 540 IF A\$="" THEN 80
 NT 550 GOTO 510
 NX 560 TRAP 890:OPEN #2,8,0,AS:TRAP 40000
 :AS=""
 WE 570 ? #2;N:#2;AA:#2;NM:#2;O:#2
 :LEN(Z\$)
 AF 580 FOR I=1 TO Q:#2;A(I):NEXT I
 QA 590 FOR I=1 TO LEN(Z\$):#2;Z\$(I,I):NE
 XT I:CLOSE #2
 FS 600 POSITION 4,5:? " U L O Z E N O
 !":GOSUB 910:GOTO 80
 NM 610 POSITION 9,22:? " POTVRD KLAVESOU
 'L'!:GET #1,SEL:IF SEL=ASC("L") THEN
 640
 NO 620 POSITION 9,22:? " POTVRD KLAVESOU
 'L'!:GET #1,SEL:IF SEL=ASC("L") THEN
 640
 LU 630 ? CHR\$(125):GOTO 80
 OE 640 ? CHR\$(125):CLR :DIM A\$(20)

```

CG 650 A$="" :? :? "          VSTUPNI ZARIZENI
(D/C/T/M)"
DY 660 POSITION 2,5 :? "          " ;:INPUT
A$:IF A$="" THEN 80
SR 670 IF A$(1,1)="D" OR A$(1,1)="C" OR A$(1,1)="T" OR A$(1,1)="M" THEN 700
FY 680 IF A$="" THEN CLR :GOTO 80
QE 690 GOTO 650
WL 700 TRAP 890:OPEN #2,4,0,AS:TRAP 40000
:CLR
QT 710 INPUT #2;N:INPUT #2;AA:INPUT #2;NM
:INPUT #2;Q:INPUT #2,B
WD 720 GOSUB 850
FJ 730 FOR I=1 TO Q:INPUT #2;X:A(I)=X:NEX
T I
PK 740 FOR I=1 TO B:INPUT #2,A$:Z$(I,I)=A
$:NEXT I
RO 750 POSITION 4,5 :? "      N A H R A N O
!":GOSUB 910:GOTO 80
KM 760 POSITION 9,22 :? "    POTVRD KLAVESOU
'E'!"
MQ 770 GET #1,SEL:IF SEL=ASC("E") THEN 79
0
MF 780 ? CHR$(125):GOTO 80
FP 790 POKE 752,0:GRAPHICS 0:CLOSE #1:CLO
SE #2:CLR :? CHR$(125):END
LH 800 ? :? "ZADEJ MAXIMALNI DELKU POLOZK
Y (1-40)":TRAP 420:INPUT N:TRAP 40000
PM 810 IF N<=0 THEN 420
CR 820 IF N>40 THEN N=40
JU 830 F=FRE(0)-N-2068:AA=INT((F-6)/(7+N)
):NM=AA*(N+1)
ZM 840 RETURN
VV 850 DIM AS(N+1),WS(1),S(10),A(AA),Z$(N
M)
ZQ 860 RETURN
EJ 870 ? CHR$(125):POSITION 9,8 :? "JIZ NE
ZBYVA VOLNA PAMET!"
SW 880 GOSUB 910:GOTO 80
BA 890 TRAP 40000 :? CHR$(125):POSITION 4,
12 :? "ZVOLENE ZARIZENI NENI PRIPOJENO!
"
SH 900 GOSUB 910:GOTO 80
HZ 910 POSITION 7,22 :? "          STLAC KLA
VESU ":GET #1,SEL
ZJ 920 RETURN
SN 930 GISUB 910:GOTO 80
LX 940 ? CHR$(125):POSITION 13,7 :? "CHYBN
A VOLBA!":POSITION 6,13 :? "NEBYLY VLOZ
ENY ZADNE POLOZKY!"
LS 950 GOSUB 910:RETURN

```

JAK NA TO ?

V naší pravidelné rubrice vám tentokrát nenabízíme manuál na některou z her, myslíme si však, že námi uvedený návod upoutá stejně. Patří totiž k zajímavému a docela užitečnému programu. Takže, hurá do toho...

V E R I F Y K A T A L O G I I

Autor: Petr Vohnický <PEVO>

Program prohledá celou vloženou kazetu, programy nahrané systémem TURBO 2000 a TURBO UNIVERSAL verifikuje a jejich názvy, stav, délku, systém nahrávky a pořadí na kazetě uloží do paměti. Odtud jsou pak listovány na určená periferní zařízení. Jeho kapacita je min. 660 názvů. Při dosažení této hranice je uživatel upozorněn na možnost zablokování systému. Klávesa RESET vraci program do menu aniž by bylo cokoliv smazáno.

Po natažení Verify katalogu II se objeví základní menu programu s touto volbou:

- "K" - Nová kazeta; program žádá název kazety (max. 21 znaků).
- "X" - Konec; ukončení činnosti a skok do SELF TESTU.
- "I" - Load názvů; nahraje dříve uložené názvy ze zařízení
 - C - standart s krátkými mezerami
 - CC - standart s dlouhými mezerami
 - T - TURBO 2000, bloky 1024 byte
 - M - ramdisk
 - K - klávesnice, bez zobrazení psaného textu
 - E - jako K se zobrazením

Je-li v paměti název kazety či programu, je základní menu rozšířeno o následující volby:

- "J" - Pokračovat v kazetě; neuzavírá obsah strany kazety, pokračuje v nahrávání a číslování
- "L" - List názvů; listuje názvy na zařízení: C,CC,T
(nepovinný název),P,M,E,S
- "N" - Smazání názvů; vymaže paměť a vrátí se do základního menu

U všech zařízení je nepovinná dvojtečka. Je-li používáno zařízení E a J jako vstupu, je nutno pamatovat na to, že prvních 8 znaků bude bráno jako hlavička (1,2 byte - \$3000 = délka; 3,4,5 byte = počítadlo kazety; 6,7,8 byte = počítadlo celkem).

Způsob použití:

Po vložení první kazety použijeme volbu "K". Totéž učiníme rovněž tehdy, chceme-li změnit název kazety. Touto volbou nejsou smazány dříve nahrané názvy. Kazetu nastavenou na začátek vložíme do mgf. a stiskneme PLAY. Pokud práci Verify katalogu přerušíme stiskem RESET, můžeme pokračovat volbou "J". Na konci kazety se stiskem RESET vracíme do rozšířeného menu.

List na E,S lze zastavit stiskem CTRL 1.

Slabikář ATARIsty

Kazetová paměť, co to je?

Po vypnutí počítače je obsah vnitřní paměti nenávratně ztracen. Těžko si lze představit, že bychom pokaždé program znova přepisovali pomocí klávesnice. A to nemluvím o souborech dat, které jsou převážně mnohem rozsáhlejší než vlastní program. Proto je nezbytná vnější paměť.

Nejjednodušší a nejlevnější možnost uchování programů a dat poskytuje magnetofonová kazeta. Zápis i čtení probíhá, z technického hlediska, stejně jako u běžných hudebních nahrávek. Použití speciálního mgf. ATARI však zajišťuje uživateli komfortní obsluhu včetně programovatelného zastavení a spuštění mgf. a kvalitnějšího záznamu. Toto vše je samozřejmě možno vyřešit i u normálního mgf. pomocí interfejsu, ale práce již vyžaduje poměrně hluboké odborné znalosti systému.

Nevýhodou této vnější paměti je hlavně seriová registrace informací a také malá rychlosť záznamu. Počítač sice dokáže vyhledat požadovaný oddíl, ale hledání spočívá v kontrole všech načtených informací a jejich následném posouzení stejnou rychlostí jakou byl pořízen záznam. Je proto výhodnější používat kazety s kratší páskou, neboť výměna kazety zabere mnohem méně času než prohledání kazety s delší páskou.

S "kazetovou pamětí" spolupracuje ATARI prostřednictvím obvodu POKEY. Zápis i čtení informací jsou realizovány pomocí procedur vnitřního operačního systému (OS) mikropočítače. OS zapisuje informace v blocích o stejné délce rychlosť 600 baudů při standartním záznamu (600 bitů/sec).

Obvod POKEY rozlišuje přenášený bajt v následujícím pořadí: 1.bit = startbit (logická 0), dále 8 bitů dat a nakonec stopbit (logická 1). Bajt je vysilán i čten vždy od nejnižšího bitu. Pro logickou "1" je vysilán kmitočet 5327 Hz, pro logickou "0" - 3995 Hz. Celý obraz vysílaného bajtu vypadá následovně:

```
st 0 1 0 1 1 0 0 1 sp
startbit      data          stopbit
```

Každý blok záznamu se skládá ze 132 bajtů a obsahuje: 2 znaky ukazatelů, kontrolní bajt, 128 bajtů dat a bajt kontrolního součtu.

Znaky ukazatelů mají šestnáctkový kód 55. Spolu s bity startu a stopu má každý ukazatel délku 10 bitů a slouží k měření rychlosť přenosu. Jak již bylo řečeno, je teoretická rychlosť 600 baudů. Skutečná rychlosť však stálá není a závisí na kolísání otáček motorku mgf., tření pásky kazety a podobně. Proto je měřena odpovídající procedurou SIO. Po zjištění skutečné rychlosť záznamu jsou ihned odpovídajícím způsobem nastaveny potřebné obvody uvnitř počítače. Timto způsobem může být měněna rychlosť přenosu dat od 318 až do 1407 baudů. Měření rychlosť přenosu je prováděno pro každý blok čtený z mgf. pásky.

Kontrolní bajt má vždy jednu ze tří možných hodnot:

- \$FC ukazuje, že blok obsahuje přesně 128 bajtů dat

- \$FA ukazuje, že blok je zaplněn pouze částečně, obsahuje méně než 128 bajtů dat. Tento případ může nastat pouze před koncem souboru. Skutečný počet bajtů v bloku je pak uveden v posledním bajtu před kontrolním součtem, t.j. ve 128. bajtu bloku dat.

- \$FE ukazuje na poslední blok dat souboru a obsahuje 128 nulových bajtů.

Bajt kontrolního součtu obsahuje počet všech ostatních bajtů v bloku, včetně obou ukazatelů. Součet je prováděn tzv. kruhovým výpočtem: s každým zapsaným bajtem je vždy doplněn rovněž bit přenosu. To je příčinou hlavních problémů. OS, který zapisuje data na pásku kazety, musí ponechat dostatečně dlouhé přestávky, aby začátky bloků nebyly v průběhu nahrávání vynechávány.

Pro odlišení jednoho bloku od druhého vytváří procedura obsluhy mgf. meziblokové mezery (IRG). Délka IRG je závislá na módu použitém pro zápis dat. Existují dva druhy IRG. Normální (dlouhé) meziblokové mezery jsou určeny ke čtení zápisu s překladem. Páska se vždy po zápisu bloku zastaví. Může se stát, že počítač svoje výpočty provede dostatečně rychle a pak k zastavení pásky nedochází.

V módu zkrácených meziblokových mezer se páška nezastavuje ani při zápisu ani při čtení. Všeobecně se tento mód používá k ukládání BASIC programů příkazem CSAVE.

Soubory dat jsou na mgf. pášce ukládány v tzv. rekordech (záznamech). Počet rekordů je dán délkou pásky. Rekord se dále dělí na pole a může jich mít libovolné množství, avšak všechny rekordy musí mít stejný počet polí. Pro označení konce souboru zapisuje počítač speciální rekord.

Jedno vážné ohrazení však přece jen platí pro všechny rekordy dat: na mgf. pášce musí být zaznamenány sekvenčně, tzn. že musí začínat od začátku a pokračovat až do konce, nelze nic měnit, doplnit či vynechat.

Počítač data do i z mgf. nezpracovává bajt po bajtu, ale po 128 bajtových blocích. Rekord může obsahovat více nebo méně než jeden blok. Lidově řečeno program se nestará o délku bloků, to obstará do jisté míry počítač automaticky sám. Zde je nutno podotknout, že s načtením rekordů delších než blok zpět do počítače mají zejména začátečníci potíže. Takovéto rekordy se totiž musí načítat jiným způsobem než je proveden zápis. Jinak dojde k "zahlcení" a počítač hlásí chybu typickou pro tento případ: ERROR 137. (viz Manuál ATARI)

Se všemi periferiemi komunikuje ATARI pomocí vstupně/výstupních kanálů, kterých má 8 a jsou číslovány 0 - 7. Pro spojení musí program nejprve otevřít některý kanál. Jestliže poté program zadá výstup dat, proudí data tímto kanálem na určené zařízení. Podobně je tomu i u vstupních operací.

BASIC rezervuje kanály s čísly 0,6,7 pro určité operace.

Kanál č. 0 je trvale rezervován pro Editor a okamžitě vykonávané vstupní instrukce z klávesnice. Rovněž jednoduché instrukce INPUT a PRINT využívají kanálu 0.

Kanál 7 je používaný pro některé instrukce tiskárny a zápis a čtení programů.

Speciální grafické instrukce užívají kanál 6.

Pro potřebu programů jsou k dispozici bez omezení kanály 1-5. Kanály 6,7 lze použít pouze za určitých podmínek.

Instrukce OPEN otevřívá kanál pro jedno určité vnější zařízení. Následující vstupní/výstupní instrukce mají tedy k tomuto zařízení přístup a mohou s ním přes tento kanál komunikovat.

Tvar instrukce OPEN je následující: OPEN # k, a, t, název kde

k = číslo kanálu a musí platit, že: $0 < k < 7$

a = označuje druh operace: 4 - vstup

8 - výstup

12 - oba současně

Podmínkou pro "a" je, že musí mít smysl. Nelze např. otevřít klávesnici pro výstup.

t = je parametr závislý na užitém zařízení. Pro "E" a "K" je ignorován, pro "S" rozlišuje grafický a textový mód. Pro "C" určuje délku meziblokové mezery (0-dlouhé, 128-krátké) název = název zařízení, viz manuál ATARI

Kanál zůstává otevřený až do konce programu, nebo jeho zavření instrukcí CLOSE. Pokud program končí instrukcí END, jsou uzavřeny všechny kanály. Je-li však přerušen instrukcí STOP, stisknutím BREAK nebo hlášením chyby ERROR..., všechny kanály zůstávají otevřené. Tato situace nastává zejména při odlaďování programů, kdy často přerušujeme vykonávání programu pro různé opravy.

Instrukce CLOSE# uzavírá pouze jeden kanál a proto je nutné přesně specifikovat o který se jedná. Např.: CLOSE# 1, CLOSE# 2 a pod.

Informace uložená na mgf. pásku se stane dostupnou otevřením zařízení a zůstane dostupnou až do zavření příslušného kanálu. Magnetofon však nemůže být současně otevřen pro vstup i výstup.

Pokud program narazí na instrukci OPEN, signalizuje počítač tuto skutečnost zvukem. Jeden signál je pro vstup (LOAD), dva pro výstup (SAVE). Následně, po přesném nastavení mgf. pásky, čeká na stlačení libovolné klávesy. Teprve poté se roztočí mgf. páška a počítač odečte nebo zapiše 20ti sekundový zaváděcí tón. Po tuto dobu není žádná jiná procedura vykonávána.

Důležité je uzavření nepoužívaného zařízení, zejména zařízení otevřeného pro zápis. Toto opomenutí může způsobit ztrátu dat. Bufer magnetofonu může být totiž částečně naplněn daty a uzavřením zařízení se aktivuje zápis dat z tohoto buferu do posledního bloku na kazetě. Jestliže nebude kanál uzavřen, nebude nikdy obsah takto částečně zaplněného buferu nahrán na pásku.

Data mohou být zaznamenávána pomocí instrukce PRINT# nebo PUT. Obě instrukce vysílají data do otevřeného kanálu a je lhodné, jaké zařízení je k němu připojeno.

Instrukce PRINT# vysílá číselné i textové hodnoty v ATASCII kódu. K rozdělení pozice kódů je lépe používat středník než čárku. Čárka je sice povolena, avšak způsobuje doplnění dodatečnými mezerami.

Každá samostatná hodnota vysílaná na mgf. pásku má končit znakem EOL (End Of Line). I když je to často používáno, neměla by hodnota končit středníkem nebo čárkou.

Každá instrukce PUT vysílá numerickou hodnotu od 0 do 255. Zabírá stejně místa jako jeden ATASCII znak.

Instrukce INPUT# odečítá hodnoty zapsané na mgf. pásku instrukcí PRINT#. Kanál pro ni určený musí být otevřen pro vstup dat do počítače. Instrukce interpretuje data jako kódy ATASCII. Pokud když narazi na EOL (kód ATASCII 155), převede znaky načtené od posledního EOL do následující proměnné podle tabulky proměnných. Je-li proměnná numerická, INPUT# překládá načtené znaky do numerických hodnot.

Instrukce GET načítá numerické hodnoty. Program pak musí rozhodnout, jak načtený znak interpretovat. Například lze hodnotu považovat za ATASCII kód a zobrazit jej pomocí funkce CHR\$.

No a to by myslím na dnešek stačilo.... že?

š.v.

Na závěr ještě trochu adres, pro pokusy s obrazem při programování v BASICu:

- 88,89- ukazatel začátku obrazové paměti OS
- 82- levý okraj obrazu pro GR.0 (obyčejně 2)
- 83- pravý okraj (obyčejně 39)
- 94,95- okamžitá pozice kurzoru
- 85,86- sloupec pozice kurzoru
- 84- řádek pozice kurzoru
- 93- hodnota znaku na pozici kurzoru

No a kdo chce vidět jak vypadá Display List, může zkoušet následující krátký program:

```

10 DL= PEEK(560)+256*PEEK(561)
20 FOR I= 1 TO 31
30 PRINT DL+I,PEEK(DL+I)
40 NEXT I

```

ZO SVAZAR M

ATARI KLUB

OLOMOUC

ev. č. 88ST-16.7.-724

poč. listů 16

Na tomto čísle spolupracovali: P.Leden, J.Skála, P.Vohnický,
T.Bělik

=====

ATARI 602, technický zpravodaj pro mikroelektroniku a výpočetní techniku. Vydává 602. ZO Svazarmu pro potřeby vlastního aktuivu, zodpovědný redaktor Vohnický Š. Adresa redakce: 602. ZO Svazarmu Wintrova 8 160 41 Praha 6, tel.: 341409. Povoleno ÚVTEI pod ev. číslem 87006. Cena 9 Kčs dle ČČÚ č. 1030/202/86.