1-2/1990

KLUB MIKROELEKTRONIKY

ATARI®

ZPRAVODAJ

OLOMOUC

HIT'90

NEW POCKET COMPUTER ATARI PORTFOLIO

MINIPC

ST vs. MAC

Mac v prachu ST

Snad je to v naší krvi...

Stalo se samozřejmostí, že lidé potřebují neustále hodnotit a srovnávat věci, které vidí kolem sebe, která z nich je rychlejší nebo výkonnější. Platí to i v případě výpočetní techniky. Na počítač se již nedíváme s úctou podle objemu prostoru, který zabírá, ale podle výkonu a koncepce.

Jedním ze standartů, který se používá pro srovnání počítačové "síly", je rychlost vykonávání programu. Stejný program ve stejném programovacím jazyku se "protáhne" různými počítači. Měří se čas, který potřebuje počítač k vyřešení úkolu. Naměřené hodnoty se sestaví do tabulky a tak získáme přehled o přibližném výkonu počítače.

V roce 1981 americký časopis BYTE poprvé zveřejnil programy na testování rychlosti počítačů, tzv. benchmark testy. Jedním z nejuznávanějších testů se stal program pro vyhledávání prvočísel, který je v odborné literatuře znám pod názvem "Prime Number Benchmark". Tento srovnávací program (= benchmark test) je vlastně jednoduchá procedura, která nalezne všechna prvočísla mezi čísly 3 a 16381. Samotná metoda pochází z 3. století před naším letopočtem; je to vlastně upravené "Síto" matematika Eratosthena.

Síto kvalifikace

Jak dopadl počítač ST?

Pro testování jsme použili počítač Atari 520 ST a programovací jazyk C. Stopovali jsme, jak dlouho trvá běh programu. Výsledek se objevil za 3.8 s. Tato rychlost je srovnatelná s minipočítači pracujícímu pod operačním systémem UNIX! (Podívejte se na tabulku 1.) Minipočítač Z8000 Z-Lab Zeus UNIX s jazykem C "vyhodil" výsledek až za 4.8 s. Další minipočítač, Z8001 5.5 Mhz, pracující s jazykem C pod OS UNIX dosáhl času 1.97 s. Na druhém konci stupnice pak najdeme mikropočítače, kterým trvalo zpracování testu od 15.7 s (Digital BASIC na mikroprocesoru Z80) až po neuvěřitelných 5115 s (1 hodina 25 minut), které dosáhl CP/M Z80 v jazyku COBOL. Z výsledků vidíme, že 8-bitové Atari zdaleka nepatří do "starého železa"...

ST vs. Mac

Apple Macintosh je relativně nejbližší počítači Atari ST, protože oba používají stejný mikroprocesor 68000. Nejrychlejší čas, který Mac dosáhl, byl 7 s a nejpomalejší 13 s. Tyto rozdíly vznikly z použití dvou typů programovacího jazyka C. Atari ST tak zanechal svého protivníka ve svém stínu!

Upozorňuji, že tento i jiné srovnávací testy nezávisí na rychlosti vstupně/výstupních operací. Jsou počítače, které bleskurychle zpracují daný problém, ale potom celé odpoledne zapisují výsledky na disketu. Jiné počítače pracují sice pomaleji, ale o to svižněji zapisují výsledky, a proto kompletace celého úkolu je dříce hotova.

Předpokládejme, že bychom si vybrali testovací program, který by třídil soubory čísel na disketách. Vlastní rychlost disketové jednotky a jejího obslužného programu by pak byla stejně důležitá jako okamžitá rychlost počítače. Velké sklony ke zpomalení práce počítače ukazují tiskárny. Například Atari 520 ST může vykonat více než 10 miliónů operací za dobu, kterou potřebuje normální tiskárna k návratu svého vozíku.

Existují však prostředky, jak zrychlit běh testovacího programu na určitém typu počítače. Využijeme většinou hardwarové vlastnosti, které nám počítač poskytuje. Uživatelé 8-bitových počítačů Atari jistě znají způsob, jak zrychlit činnost programů až o 30 procent. Stačí jen programově vypnout obrazovku...

Pouze 2 milióny?

Jak je rychlý rychlý? Poslední dobou je stále zábavnější sledovat recenze počítačů v časopisu Elektronika, kde jeden počítač chválí za vykonání 8 miliónů operací za sekundu a druhému se posmívají za to, že zvládne "jen" 2 milióny operací za sekundu. A přiznejme si - jak dlouho by nám trvalo provedení 2 miliónů operací jen s pomocí tužky a papíru?

Jiří Hrdlička
GIA Software

Tabulka 1 - "Prime Number Benchmark" Test

Počítač	OS	Jazyk	Čas (s)
68000 Atari 520 ST	TOS	C Digital CP/M 68K	3.8
68000 Apple Macintosh	-	C Manx	7
68000 Apple Macintosh	-	C Hippo L2	13
Z8001 5.5 MHz	UNIX	C	1.97
Z8000 Z-Lab	Zeus UNIX	C	4.8
Z80	CP/M	Digital BASIC	15.7
Z80	CP/M	Microsoft COBOL	5115
6502 Atari 800 XL	OS Rev.B	Action! display off	12.2
		display on	17.9
6502 Atari 800 XL	OS Rev.B	BASIC	389
6502 Atari 800 XL	OS Rev.B	BASIC XL	214

Action! software

Input/Output v jazyku Action!

Knihovna jazyku Action! na disketě obsahuje celkem devět procedur typu I/O (vstup/výstup). Tento počet se však může zdát malý, zejména pro náročného programátora, a tedy nám nezbývá nic jiného, než se pokusit tento počet rozšířit.

Procedury již existující v jazyku Action! jsou: Print, Input, Put, Get, Open, Close, Note, Point a XIO. Tyto procedury pracují na stejném principu jako jím odpovídající příkazy v jazyku Atari BASIC. Chybí zde však procedury pro práci s disketovou jednotkou (Atari BASIC je konec konců také nemá), které jsou již imlementovány v jazyku Turbo BASIC XL. Pokusme se tedy vytvořit potřebné procedury v jazyku Action!, což by nám umožnilo dokonalejší využití tohoto vynikajícího programovacího jazyka.

Základní procedury

Prvních pět procedur získáme velmi snadno pomocí procedury XIO. V Atari BASICu existuje stejná instrukce XIO, která byla již mnohokrát popsána v různých publikacích. Uvedené parametry pro XIO, kterými se zde budeme zabývat, platí pro všechny programovací jazyky počítače Atari, ve kterých se tato procedura nachází.

Tímto způsobem vytvoříme procedury pro změnu názvu souboru (Rename), vymazání programu (Delete), ochranu a uvolnění souboru (Lock/Unlock) a také formátování diskety (Format). Ve všech případech využíváme pro provedení těchto operací kanálu 5 IOCB, který se jen velmi zřídka využívá v programech. Upozorňuji pouze na proceduru Format, která formátuje disketu na tzv. rozšířenou hustotu (medium density) s kapacitou 1010 sektorů. Pro formátování diskety v jednoduché hustotě (single density) o kapacitě 707 sektorů je třeba změnit hodnotu v proceduře Format z 254 na 253 (viz XIO).

```
;Input/Output
;Action!

;změna názvu disketového souboru
;příklad: Rename("D1:OLD.ACT,NEW.ACT")
```

```
PROC Rename(BYTE ARRAY filename)
  XIO(5,0,32,0,0,filename)
RETURN

;vymazání souboru z diskety
;příklad: Delete("D1:PROGRAM.ACT")

PROC Delete(BYTE ARRAY filename)
  XIO(5,0,33,0,0,filename)
RETURN

;ochrana souboru na disketě
;příklad: Lock("D1:PROGRAM.ACT")

PROC Lock(BYTE ARRAY filename)
  XIO(5,0,35,0,0,filename)
RETURN

;uvolnění ochrany souboru na disketě
;příklad: UnLock("D1:PROGRAM.ACT")

PROC UnLock(BYTE ARRAY filename)
  XIO(5,0,36,0,0,filename)
RETURN

;formátování diskety
;příklad: Format("D1:")

PROC Format(BYTE ARRAY drive)
  XIO(5,0,254,0,0,drive)
RETURN
```

Přenos bloku

Při práci s disketovou jednotkou se často stane, že je třeba zapsat nebo přečíst značné množství bytů, které jsou postupně umístěny v paměti (nebo v bloku paměti). Jako příklad si můžeme uvést načtení nebo zápis obrazové paměti. Velice jednoduchá, ale málo efektivní, je metoda, kdy pomocí smyčky For-Next využijeme dvojici procedur Peek/Put pro zápis a dvojici Get/Poke pro čtení. V Atari BASICu využíváme krátké strojové programy, které zrychlují celou činnost. Struktura jazyka Action! však dovoluje dosáhnout stejného výsledku elegantním způsobem.

Nejdříve musíme nadefinovat proceduru CIO, která nám umožní přístup do operačního systému počítače Atari. Potom napíšeme proceduru Block, která řídí přenos dat umístěním správných hodnot do bloku IOCB a která vyvolává proceduru CIO.

Nyní již můžeme uložit procedury BGet a BPut pro přenos bloku dat. Všimněte si, že procedura BGet byla napsána jako funkce. Díky tomu po jejím vyvolání obdržíme délku čteného bloku. Kromě toho je tento způsob v souladu s konvencí jazyka

Action!, kde všechny procedury čtení jsou funkcemi (např. Input a Get). U obou procedur BGet i Bput jsou parametry při jejich vyvolání tyto: číslo kanálu IOCB, adresa přenášeného bloku a jeho délka. Potřebný kanál se samozřejmě musí otevřít (pomocí OPEN) před vyvoláním procedur.

```
;základní procedura I/O

PROC CIO=$E456(BYTE acum,xreg)

;pomocná procedura přenosu

CARD FUNC Block(BYTE kan,scrn,
                CARD adr,buflen)
  TYPE IOCB= BYTE id,num,cmd,stat
             CARD badr,padr,blen
             BYTE a1,a2,a3,a4,a5,a6
  IOCB POINTER iptr
  kan== $07
  iptr=$340+(kan LSH 4)
  iptr.cmd=scrn
  iptr.blen=buflen
  iptr.badr=adr
  CIO(0,kan LSH 4)
RETURN(iptr.blen)

CARD FUNC BGet(BYTE kan CARD adr,length)
  CARD temp
  temp=Block(kan,7,adr,length)
RETURN(temp)

;zápis daného bloku
;příklad: BPut(1,32768,1024)

PROC BPut(BYTE kan CARD adr,length)
  Block(kan,11,adr,length)
RETURN
```

Procedury pro disketu

Díky snadnému přístupu k systémovým procedurám umožňuje jazyk Action! bezprostřední provádění operací na celých sektorech diskety. K tomu je třeba nadefinovat proceduru DSK, která vyvolává DSKINT. Normální vyvolání pomocí tabulky skoků operačního systému (adresa $E453) dovoluje práci pouze s disketovou jednotkou č.1. Využil jsem přímého vyvolání na adrese $C6B3, což však nemusí fungovat u počítačů se starší nebo jinou verzí operačního systému (OS typ B).

Druhou (pomocnou) procedurou je Sector, která určuje parametry v bloku kontroly zařízení BCD (podobně jako pracuje procedura Block pro IOCB).

```
;základní disketová procedura
```

```
PROC DSK=$C6B5(BYTE acum)

;pomocná disketová procedura

BYTE FUNC Sector(BYTE drv,scrn
                 CARD adr,secnum)
  TYPE DCB= BYTE dev,num,cmd,stat
            CARD badr BYTE tout,unus
            CARD cntr,aux
  DCB POINTER dptr=$300

  drv== $07
  dptr.cmd=scrn
  IF scrn=$50 or scrn=$57
    THEN dptr.stat=$80
    ELSE dptr.stat=$40
  FI
  dptr.aux=secnum
  dptr.badr=adr
  drv==+$30
  DSK(drv)
RETURN(dptr.stat)

;čtení sektorů diskety
;příklad: GetSec(1,$169,1536,2)

BYTE FUNC GetSec(BYTE station
                 CARD nsek,adr,lsek)
  CARD i
  BYTE temp

FOR i=nsek TO nsek+lsek-1
  DO
    temp=Sector(station,82,adr,i)
    adr==+128
  OD
RETURN(temp)

;zápis sektorů bez verifikace
;příklad: PutSec(1,4,$4000,5)

PROC PutSec(BYTE station
            CARD nsek,adr,lsek)
  CARD i
FOR i=nsek TO nsek+lsek-1
  DO
    Sector(station,80,adr,i)
    adr==+128
  DO
RETURN

;zápis sektorů s verifikací
;příklad: WrtSec(1,4,$4000,5)
```

```
PROC WrtSec(BYTE station
            CARD nsek,adr,lsek)
  CARD i
FOR i=nsek TO nsek+lsek-1
  DO
    Sector(station,87,adr,i)
    adr==+128
  OD
RETURN

;přečtení statutu disketové jednotky
;příklad: DrvStat(1)

BYTE FUNC DrvStat(BYTE station)
  BYTE temp
  temp=Sector(station,83,0,0)
RETURN(temp)
```

Nyní již můžeme snadno napsat vlastní procedury, které zabezpečí čtení a zápis sektorů. V těchto procedurách musíme použít smyčku, která umožní přenos několika následných sektorů při jednom vyvolání procedury. Procedura zápisu sektorů je zde ve dvou variantách, a to buď s verifikací zápisu (WrtSec), nebo bez verifikace zápisu (PutSec). Parametry těchto procedur jsou: číslo disketové jednotky, číslo prvního sektoru, adresa bufferu paměti a počet přenášených sektorů.

Závěrem programu je uvedena ještě jedna jednoduchá procedura, která slouží k přečtení statutu disketové jednotky. Pozor, tato procedura není totožná s instrukcí STATUS v Atari BASICu, a proto má i jiný název. Po jejím vyvolání je status disketové jednotky o délce 4 byte umístěn do registru DVSTAT (adresa 746) a lze jej přečíst instrukcí Peek. Parametrem této procedury je číslo disketové jednotky. Na rozdíl od procedur pro přenos bloků procedury GetSec a DrvStat vracejí vždy zpět status operace - tedy číslo 1.

Podle časopisu BAJTEK 4/89, str. 6-7
přeložil ing. Jeroným Liška
volně upravil Jiří Hrdlička

INPUT / OUTPUT

Bloky řízení vstupu/výstupu

Všechny vstupně-výstupní operace v počítači ATARI 800 XL/XE a 130 XE jsou prováděny přes tzv. bloky řízení V/V (IOCB - Input/Output Control Block). Tyto bloky jsou očíslovány od 0 do 7 a nacházejí se na třetí stránce paměti od adresy 832. Po zapsání odpovídajících údajů na toto místo se přejde do procedury obsluhy V/V zařízení CIOMAIN (adresa 58454). To lze snadno provést i z úrovně BASICu.

Některé řídící bloky (0,6,7) používá operační systém. Je třeba znát jejich určení, aby jich bylo možno plně využívat při práci s počítačem. IOCB 0 je určen pro obsluhu editoru (klávesnice a obrazovka), IOCB 6 je používán pro výstup na obrazovku v některých grafických režimech, IOCB 7 slouží k operacím LOAD, SAVE, LIST, CLOAD, CSAVE a LPRINT. Z toho vyplývá, že můžeme používat bloky 6 a 7 jen tehdy, jestliže nepoužíváme v programu ty instrukce, které jich využívají.

Každý blok řízení V/V zabírá 16 bytů paměti, které mají daný význam. Po řadě to jsou:

Adresa v IOCB	Název	Význam
0	IOCBCHID	Identifikační číslo kanálu V/V. Označuje pořadí. Je nastaveno OS
1	IOCBDSKN	Číslo disketové jednotky.
2	IOCBCMD	Kód příkazu. Na tuto adresu je třeba zapsat kód příkazu, který má být proveden. Může to být:
Pro všechna zařízení		/ 3 -Open | 5 -Get record | 7 -Get characters | 9 -Put record | 11-Put characters | 12-Close L' 13-Status

```
  Pouze                          / 14-Special
   pro     . . . . . . . .      |  17-Draw line
obrazovku                         L' 18-Draw line with right fill

                                 / 32-Rename file
  Pouze                         |  33-Delete file
   pro                          |  35-Lock file
disketovou . . . . . . . .      |  36-Unlock file
 jednotku                       |  37-Point
                                |  38-Note
                                 L' 254-Format
```

3	IOCBSTAT	Status. Po provedení procedury CIOMAIN je umístěna hodnota 1 (SUCCESS), nebo kód chyby.
4,5	IOCBBUFA	Adresa bufferu. Adresa 1.bytu vyrovnávací paměti dat.
6,7	IOCBPUTB	Počáteční adresa procedury Put Byte pro dané zařízení. Nastavuje OS.
8,9	IOCBBUFL	Délka bufferu.
10-13	IOCBAUXn	Pomocné hodnoty, závislé na prováděné operaci (n=1-4)
14	IOCBNUM	Číslo IOCB*16.
15	IOCBCHAR	Pomocný registr pro zasílání dat při zápisu bez bufferu.

Buňky paměti, označené IOCBAUXn, jsou využívány uživatelem pouze během operace OPEN a odpovídají hodnotám, které jsou dány instrukcí OPEN v BASICu. Do IOCBAUX1 lze zapsat následující hodnoty

4 -čtení (mimo tiskárnu a obrazovku)
6 -čtení directory (pouze disketová jednotka)
8 -zápis (mimo klávesnici)
9 -zápis na konec souboru (pouze disketová jednotka)
12-čtení i zápis
13-čtení i zápis z obrazovky (pouze editor)

Jiné hodnoty (8,12,24,28,40,44,56,60) jsou dovoleny pouze pro obrazovku a označují různé grafické režimy.

Hodnota, zapsaná v IOCBAUX2, je důležitá pouze pro obrazovku a magnetofon. Pro obrazovku může být v rozsahu 0-8 (15) a označuje grafický režim v BASICu. Pro magnetofon mohou být použity hodnoty 0 nebo 128, které označují délku mezery mezi bloky na pásce (0- dlouhá mezera jako LIST "C:", 128- krátká mezera jako CSAVE).

Všechny uvedené hodnoty lze zapsat instrukcemi POKE nebo pomocí procedury ve stojovém kódu. Taková procedura může být provedena následujícím způsobem:

```
<CIO>
A$="h"  Vd<CTRL+.>":A$(3,3)=CHR$(IOCB*16):A=USR(ADR(A$))
```

Poznámka:
Podtržené znaky nutno vypsat inverzně.
V závorce <...> se nachází znak .

Zdrojový text procedury:

```
104          PLA
162 ...      LDX #IOCB*16
32 86 228    JSR CIOMAIN
96           RTS
```

V druhém řádku procedury je zapsáno do registru X mikroprocesoru číslo IOCB, násobené 16. Vyžaduje to procedura CIOMAIN. Nyní uvedu několik příkladů. Budu používat uvedenou proceduru CIO.

Př.1: CLOSE #1

```
BASIC:IOCB=1                    ASS:LDX #16
      POKE 832+IOCB*16+2,12         LDA #12
      GOTO CIO                      STA 832,X
                                    JSR 58454
```

Př.2: OPEN #3,4,0,"C:"

```
BASIC:B$="C:"                   ASS:LDX #48
      AH=INT(ADR(B$)/256)           LDA #3
      AL=ADR(B$)-AH*256             STA 834,X
      IOCB=3                        LDA #AL
      POKE 832+IOCB*16+2,3          STA 836,X
      POKE 832+IOCB*16+4,AL         LDA #AH
      POKE 832+IOCB*16+5,AH         STA 837,X
      POKE 832+IOCB*16+8,2          LDA #2
      POKE 832+IOCB*16+9,0          STA 840,X
      POKE 832+IOCB*16+10,4         LDA #4
      POKE 832+IOCB*16+11,0         STA 842,X
      GOTO CIO                      LDA #0
                                    STA 841,X
                                    STA 843,X
                                    JSR 58454
```

Př.3:Zápis obrazové paměti v grafice 0 na kazetu po otevření IOCB 5.

```
BASIC:IOCB=5                    ASS:LDX #80
      POKE 832+IOCB*16+2,11         LDA #11
      POKE 832+IOCB*16+4,PEEK(88)   STA  834,X
      POKE 832+IOCB*16+5,PEEK(89)   LDA 88
      POKE 832+IOCB*16+8,224        STA 836,X
      POKE 832+IOCB*16+9,3          LDA 89
      GOTO CIO                      STA 837,X
                                    LDA #224
                                    STA 840,X
                                    LDA #3
                                    STA 841,X
                                    JSR 58454
```

(224+3*256=992 -počet bytů obrazové paměti v režimu 0)

Woicech Zientara
BAJTEK č.2/87, str.8
Přeložil: Ing. František Komín

Přenos strojových programů do BASICu

Užitečnost strojových programů spolupracujících s programy v BASICu byla již mnohokrát vysvětlována. Rozhodující je především rychlost. Zde nejde o zrychlení několika málo procent, jako např. při použití řízení přímého přístupu do paměti (DMA) příkazem POKE 559,0 (a zpět POKE 559,34). Použitím podprogramů ve strojovém kódu se dosahuje mnohonásobného zrychlení. Některé činnosti nelze ani bez použití strojového programu provést. To jsou akce, prováděné tzv. pseudoparalelně s hlavním programem, buďto během VBI (Vertical Blank Interrupt - přerušení při vykreslení celého obrazu - 50 krát za sekundu), nebo DLI (Display List Interrupt - přerušení po vykreslení každého řádku obrazu).

Na jednoduchém příkladu (viz listing 1) lze provést porovnání rychlosti BASIC-programu a strojového programu. Uvedený program provede popsání celé obrazovky jediným znakem. Program provede tuto činnost dvakrát a porovnává rychlost obou způsobů. Zde je zrychlení více než stonásobné, ale výsledek je ještě poněkud ovlivněn vlastním výpočtem času, který je (pro jednoduchost) proveden v obou případech v Basicu.

Časem se někteří programátoři rozhodnou vylepšit svoje programy strojovými podprogramy. Ve většině příruček k assemblerům je popsáno, kam mají tyto podprogramy umístit a jak je startovat. Zopakujme tedy tři základní způsoby:

1) Krátké relokabilní (přemístitelné) programy lze umístit přímo do funkce USR, volající podprogram:
   X=USR(ADR(".....").parametry),
kde namísto teček jsou zapsány ASCII znaky odpovídající strojovému programu. Při tomto způsobu nelze použít strojový program, obsahující kód 155 ($9B) nebo 34 ($22), neboť tyto kódy odpovídají ASCII znakům RETURN a uvozovka.

2) Delší relokabilní programy (viz např. SORT 1.0 - zveřejněný také v tomto čísle AZ) lze umístit do řetězcové proměnné:

```
DIM M$(délka)
M$(1)="......."
M$(80)="......."
atd.
X=USR(ADR(M$),parametry)
```

3) Nejčastější umístění strojových programů je na předem zvolených adresách. Pro programy nepřesahující 256 byte je to obvykle 6. stránka paměti, tj. od adresy 1536 ($0600) do 1791 ($06FF). Pro delší programy se obvykle vyhrazuje místo nad vlastním BASIC-programem snížením horní hranice RAMTOP (adr.106 - $6A) příkazem

```
POKE 106,PEEK(106)-n:GRAPHICS 0,
```

kde číslo "n" určuje počet stránek paměti, které si rezervujeme pro strojový program. Bližší informace viz /1/ .

Zbývá zvolit vhodný assembler, tj. pomocný program, který nám usnadní tvorbu a testování (ladění) strojového podprogramu. Krátký prográmek lze překódovat ručně, za pomoci kódovací tabulky, ale u delších programů by to bylo nesmyslné plýtvání energií, nehledě na možnost vzniku těžko nalezitelných chyb.

Pro uživatele počítače ATARI s magnetofonem (těch je velká většina) lze doporučit překladač ATMAS III (Atari makroassembler), nebo alespoň starší verzi ATMAS II. Verze ATMAS III je z předchozí odvozena, v Atari klubu Litvínov ji však doplnili tři velice užitečné vlastnosti:

1) Do tabulky vstupně/výstupních zařízení je doplněno zařízení "T", umožňující zápis a čtení kazety v režimu TURBO 2000, a to jak v editoru ATMAS, tak v BASICu.
2) Do tabulky vstupně/výstupních zařízení je doplněno zařízení "M", tzv. RAMDISK o velikosti 14 kB, využitelný pro zápis a čtení oblasti paměti skryté pod adresami paměti ROM a jinak běžně nevyužívané. Pozor! Lze zapsat jen jednou. Dalším zápisem se původní obsah přepíše.
3) Nejužitečnější vlastností je možnost přechodu z Editoru nebo Monitoru ATMASu III do BASICu a naopak, bez ztráty zdrojového textu, nebo programu v BASICu. Tento přechod stiskem klávesy SELECT umožňuje testování programu přímo ve vlastním BASIC-programu. V případě nesprávné činnosti je možný návrat ke zdrojovému textu v ATMASu a jeho úpravy.

Použití překladače ATMAS II je dostatečně popsáno v příručce /2/. Programátor sestaví a odladí pomocí překladače strojový program. Potřebuje ho ale trvale připojit ke svému BASIC-programu. Nejčastěji se to provádí pomocí DATA-řádků,

ze kterých se potom "stroják" v cyklu načítá do příslušné paměti nebo řetězcové proměnné. To představuje delší prodlevu při prvním spuštění programu. Druhým způsobem je transformace strojového kódu na ASCII znaky a jejich naplnění přiřazovacím příkazem přímo do řetězcové proměnné.

Program pro samočinný přenos strojového kódu do BASICových DATA-řádků je uveden jako listing 2. Program vytvářející samočinně přiřazovací příkazy pro řetězcovou proměnnou M$ je uveden jako listing 3. V případě, že strojový program obsahuje kódy 155 nebo 34, nelze je, jak bylo dříve uvedeno, přímo přiřadit do řetězcové proměnné. Program zajistí tuto činnost přerušením přiřazovacího příkazu a použije přiřazení pomocí funkce CHR$(155), případně CHR$(34). Oběma programům je třeba pouze zadat počáteční adresu, na které je umístěn přeložený strojový program (viz instrukce ORG, povinně použitá v ATMASu) a zadat délku (počet bytů) programu.

Převáděcí programy zajistí v samoreturnovacím módu editaci potřebných BASIC-řádků. Ty se potom přenesou do vlastního programu příkazem LIST. Zde se nabízí možnost využití RAMDISKu (LIST"M:", od, do), dále vymazání paměti NEW a zpětné načtení programu ENTER "M:" .

Rovněž uživatelé překladače ATMAS II mají možnost uvedené programy využít, mají-li k dispozici TURBO OPERATING SYSTEM, známý jako TOS 4.1 . Program ATMAS II nahrají volbou L (LOAD) z menu TOSu a na dotaz RUN? odpovědí "Y" program spustí. Mají nadále k dispozici TURBO 2000 jako zařízení "D:jméno" i RAMDISK jako zařízení "D8" - pro zápis zdrojového textu (nebo i přeloženého programu). Po sestavení programu v Editoru ATMASu, jeho překladu do strojového kódu (CTRL-Y) a případném odladění v Monitoru ATMASu (funkce G), je možno přejít do BASICu. Na rozdíl od ATMAS III je to však možné jen jednou a není možné se úspěšně vrátit! Přechod do Basicu bez ztráty dat (přeloženého strojového programu) v paměti se provede následovně:

- v Monitoru se provede start TOSu, který je stále v paměti:

```
G
GOTO: 0B03
```

- ze zobrazeného menu TOS 4.1 se zvolí B (přepnutí na Basic) a na dotaz INIT? je nutno odpovědět "Y"
- nápis READY oznamuje přítomnost interpretu BASIC. Nyní je možno natypovat, případně nahrát z magnetofonu (LOAD"D:") jeden z převáděcích programů (listing 2 nebo 3).

Podobný způsob přenosu dat do Basicu je možný z programu EASMD (Edit Assembler Debug). Program je nutné zavést operačním systémem TOS 4.1. Před provedením překladu (kompilace) příkazem ASM v režimu Editoru je třeba zadat volbu:

```
.OPT OBJ
```

Tato volba povolí vygenerování strojového kódu od adresy zadané direktivou *= adresa. Příkaz DOS pro přechod do DOSu v tomto případě zajistí přechod do TOSu. Další postup je shodný s přenosem z ATMASu.

Programy uvedené jako listing 2 a 3 jsou pro kontrolu bezchybného natypování vytisknuty s kontrolními kódy TYPO II

(viz časopis ATOM č.3/1989 nebo Zpravodaj Atari-klubu Olomouc č.3-4/1988). Je vhodné je zapsat na kazetu příkazem:

1) Je-li použit TURBOBASIC 1.5X, BASIC-DOS (nepoužívat menu LOAD SAVE VERIFY QUIT), nebo BASIC-ATMAS III, použijeme příkaz: SAVE "T:název"

2) Je-li použit Basic pod operačním systémem TOS 4.1., použijeme příkaz: SAVE "D:název"

Ing.Petr Válka - Atari klub Svazarmu Brno

Literatura:

/1/ Pavel Dočekal: Adresy paměti počítačů ATARI 600XL/800XL (vydal AK Jimramov)
/2/ Ing.Václav Fajta: Základy programování mikroprocesoru 6502 (vydal AK Praha)
/3/ Ing.Petr Válka: Turbo 2000 Operační systém verze 4.1 (vydal AK Brno)
/4/ Mark Chasin: Programování v asembleru pro počítače ATARI (překlad Ing.Karel Šmuk, AK Praha)

Listing 1 - program TEST.BAS Part: 1

```
IX 0 REM  POROVNANI STROJAK - BASIC
JD 10 DIM M$(35):? "↰"
KE 20 GOTO 200:REM  Po prvním RUN se zde vytvoří jiný řádek
QE 30 TIME0=PEEK(20)+PEEK(19)*256+PEEK(18)*65536
GK 40 X=USR(ADR(M$),34)
QI 50 TIME1=PEEK(20)+PEEK(19)*256+PEEK(18)*65536:FOR I=1 TO 500
      :NEXT I
ZH 60 ? "↰"
DE 70 VR=PEEK(88)+256*PEEK(89):POKE 82,0:POSITION 0,0
QZ 80 TIME2=PEEK(20)+PEEK(19)*256+PEEK(18)*65536
KX 85 FOR I=1 TO 960:? "B";:NEXT I
NO 100 TIME3=PEEK(20)+PEEK(19)*256+PEEK(18)*65536
IG 110 POKE 82,2
BQ 120 ? "↰↓↓ Tisk strojakem = ";(TIME1-TIME0)/50;" sec"
TI 130 ? "↓ Tisk v Basicu  = ";(TIME3-TIME2)/50;" sec"
TJ 140 ROZ=(TIME3-TIME2)/(TIME1-TIME0)
LQ 150 ? "↓ Basic je ";INT(ROZ*1000)/1000;" krat pomalejsi"
OI 190 END
AA 200 REM  Po vytvoření řádku 20 můžete následující řádky vym
       azat
WY 210 CLR :DIM M$(34):FOR I=1 TO 34:READ A:M$(I)=CHR$(A):NEXT
       I
AW 220 DATA 104,165,88,133,203,165,89,133,204,160,0,162,0,104,1
       04,145,203,224,3,208,4,192,192,240,8,200,208,243
DP 230 DATA 230,204,232,208,238,96
XG 240 ? "↰↓↓20 M$=";CHR$(34);M$;CHR$(34):? "↓STISKNI RETURN":P
       OSITION 0,0:STOP
```

Listing 2 - program ATMBAS.BAS Part: 1

```
DA 1 REM  PRENOS DAT: ATMAS3 -> BASIC
WH 2 REM  Ing.Valka  AK-Brno
GG 10 M=256:REM delka stroj.programu
TU 20 Z=1536:REM adresa ulozeni programu
RK 30 I=0
KZ 40 ? "↰":POSITION 2,3
KI 50 IF 7*I<>M THEN ? 20000+I;" DATA ";
FN 60 FOR J=0 TO 6
YR 70   IF 7*I+J>M-1 THEN POP :? "◀":GOSUB 100:? "↰":END
DY 80   ? PEEK(7*I+J+Z);",";:NEXT J
EH 90 ? "◀":GOSUB 100:I=I+1:GOTO 40
KK 100 ? "↓CONT":POSITION 0,0:POKE 842,13:STOP
SE 110 POKE 842,12:RETURN
```

Listing 3 - program MEMSTR.BAS Part: 1

```
IA 1 REM  PREVOD Z PAMETI DO RETEZCE
CP 2 REM  Ing.Valka  AK-Brno
NN 10 B=256:REM Velikost programu
DS 20 ZAC=1536:REM Zacatek programu
JD 30 A=1:D=1:C=20000:Z=0
HX 40 GOSUB 100:GOSUB 150
TM 50 IF K=34 OR K=155 THEN GOSUB 160:GOSUB 130:GOSUB 110:GOSUB
       140:GOSUB 100:GOTO 50
CO 60 IF Z=0 THEN Z=1:? CHR$(34);
AB 70 ? "↳";CHR$(K);
ZK 80 D=D+1:IF D=91 THEN ? CHR$(34):GOSUB 110:GOSUB 140:GOSUB 1
      00:D=1:GOTO 50
SC 90 GOSUB 140:GOTO 50
RZ 100 ? "↰":POSITION 2,3:? C;"M$(";A;")=";:C=C+1:Z=0:D=1:RETUR
       N
BA 110 ? "↓↓↓":? "CONT":POSITION 0,0:POKE 842,13:STOP
SG 120 POKE 842,12:RETURN
GJ 130 ? "CHR$(";K;")":RETURN
EA 140 A=A+1:IF A>B THEN GOSUB 160:? "↰":END
YF 150 K=PEEK(A+ZAC-1):RETURN
GX 160 IF Z=1 THEN ? CHR$(34):GOSUB 110:GOTO 100
ZL 170 RETURN
```

Program IBM klávesnice

Pomocí tohoto programu se vaše klávesnice Atari stane téměř standartní klávesnicí IBM!

Jak vytvořit vlastní klávesy?
Nejdříve je nutné definiční tabulku klávesnice přenést na bezpečné místo v paměti. Já jsem si vybral 149. stránku paměti RAM. V programu to řeším příkazem MOVE DPEEK(121),38144,192. Původní tabulka je od adresy 64337. Teď již stačí změnit hodnoty v naší nové tabulce. Jak? Postup si ukážeme na jednoduchém příkladu, např. chceme změnit třeba A na B:

1. zjistíme si interní kód klávesy A
   1000 PRINT PEEK(764):GOTO 1000 <RETURN>
   GOTO 1000 <RETURN>
2. zjistíme si ASCII kód nového znaku
   PRINT ASC("B") <RETURN>
3. zapíšeme na adresu (v našem případě) 149 krát 256 plus interní kód klavesy, ASCII hodnotu nového znaku.
   POKE 149*256+63,66
4. ukazatel na adresách 121,122 změníme na stránku 149.
   DPOKE 121,149*256 <RETURN>

Poznámka :
Všechny příklady i program jsou v Turbo BASICu, proto při zkoušení v tomto jazyku nezapomeňte na uvedení příkazu:
POKE 8329,254

Mnoho zdaru při práci na nové klávesnici.
Podle BAJTKU 5,6/89 zpracoval Petr Pohanic

Listing 1 - program IBMKEY.BAS Part: 1

```
PH 100 REM Podprogram pro klavesnici IBM
YD 110 MOVE DPEEK(121),149*256,192:RESTORE 130
IW 120 FOR I=0 TO 32:READ A,B:POKE 149*256+A,B:NEXT I
QN 130 DATA 92,125,156,157,158,130,154,132,152,129,179,142,181,
       143,157,144,155,145,91,94,115,38,117,42,118
OZ 140 DATA 95,54,45,119,43,55,61,14,28,15,29,78,91,79,93,6,30,
       7,31,70,39,71,35,134,64,124,47,60,42,39,43
JX 150 DATA 103,45,96,60,98,62,108,255,172,254
ZJ 160 RETURN
```

Trigonometrické funkce rychleji

Ian Finlayson

Osmibitové Atari patří mezi nejlepší počítače ve své třídě, ale to neznamená, že je po všech stránkách perfektní. Jeho velkou slabinou jsou pomalé matematické a trigonometrické funkce v BASICu. Hledal jsem způsob, jak se tomu vyhnout, když chci urychlit svůj program.

Jedním z řešení je vykonat všechny výpočty během inicializace na začátku programu. Vyjde to lépe, než kdyby měl program během celého chodu pracovat pomalu.

Jako příklad jsem si vybral funkci SIN. Podprogram je jen na řádcích 3800 a 3801. První řádek zjišťuje, zda už bylo pole nastaveno, aby se běh programu zbytečně nezpomaloval reinicializací. Ve druhém řádku programu je hodnota SIN(A) přepočítávána ve stupních od 0 do 90 a výsledek je ukládán do pole SSIN(A). Je to tak jednoduché! Teď mohu používat SIN pro každý přepočítaný SSIN, ale jen pro hodnoty, které jsem předtím nastavil. V tomto případu pro 0-90 stupňů. SSIN(1.4) vrátí hodnotu rovnou SIN(1) nejbližšímu celému číslu, hodnota mimo rozmezí 0-90 způsobí chybové hlášení.

Pro ilustraci vykreslí program dvě sady soustředních kružnic a sinusoid v grafickém režimu 8. Při kreslení prvních sad se využívá funkce SIN, při druhých SSIN. Jasně vidíte, že křivka při použití SSIN se vykreslí dvakrát rychleji (mimo inicializaci).

Pokud nebudete využívat celý rozsah funkce SIN (nebo jiné trigonometrické funkce), stačí změnit hodnotu jen na rozsah, který budete potřebovat. To zkrátí inicializaci po spuštění programu.

Viki Babic
přeloženo z časopisu PAGE 6/33

Listing 1 - program TRIGFN.BAS Part: 1

```
EI 1 REM ****************************
JX 2 REM **                        **
DN 3 REM **     GRAPHICS DEMO      **
JZ 4 REM **                        **
WS 5 REM **    by Ian Finlayson    **
KB 6 REM **                        **
EO 7 REM ****************************
NN 8 REM
NO 9 REM
CY 10 GRAPHICS 8:X=80:Y=80:R=70:COLOR 1:GOSUB 31800
NP 20 FOR R=20 TO 70 STEP 10:FOR A=0 TO
45:XX=R*SIN(A):YY=R*SIN(90-A)
DV 30 PLOT X+XX,Y-YY:PLOT X+XX,Y+YY:PLOT X-XX,Y-YY:PLOT
X-XX,Y+YY
KC 40 PLOT X+YY,Y+XX:PLOT X-YY,Y+XX:PLOT X+YY,Y-XX:PLOT
X-YY,Y-XX:NEXT A:NEXT R
LO 50 FOR R=20 TO 70 STEP 10:X=240:FOR A=0 TO
45:XX=R*SSIN(A):YY=R*SSIN(90-A)
DY 60 PLOT X+XX,Y-YY:PLOT X+XX,Y+YY:PLOT X-XX,Y-YY:PLOT
X-XX,Y+YY
KF 70 PLOT X+YY,Y+XX:PLOT X-YY,Y+XX:PLOT X+YY,Y-XX:PLOT
X-YY,Y-XX:NEXT A:NEXT R
RJ 80 FOR R=20 TO 70 STEP 10:PLOT 0,80
CC 90 FOR X=1 TO 90 STEP 3:DRAWTO X*8/9,80-R*SIN(X):NEXT X
EZ 100 FOR X=91 TO 180 STEP 3:DRAWTO
X*8/9,80-R*SIN(180-X):NEXT X
IW 110 FOR X=181 TO 270 STEP 3:DRAWTO
X*8/9,80+R*SIN(X-180):NEXT X
FC 120 FOR X=271 TO 360 STEP 3:DRAWTO
X*8/9,80+R*SIN(360-X):NEXT X:NEXT R
MQ 130 FOR R=20 TO 70 STEP 10:PLOT 0,80:FOR X=1 TO 90 STEP
3:DRAWTO X*8/9,80+R*SSIN(X):NEXT X
JB 140 FOR X=91 TO 180 STEP 3:DRAWTO
X*8/9,80+R*SSIN(180-X):NEXT X
MV 150 FOR X=181 TO 270 STEP 3:DRAWTO
X*8/9,80-R*SSIN(X-180):NEXT X
LY 160 FOR X=271 TO 360 STEP 3:DRAWTO
X*8/9,80-R*SSIN(360-X):NEXT X:NEXT R
OE 170 END
EA 31793 REM
XH 31794 REM ****************************
LB 31795 REM **  PODPROGRAM PREPOCTU  **
TW 31796 REM **    FUNKCE SIN(A) NA   **
RK 31797 REM **       SSIN (A)        **
YB 31798 REM ****************************
FE 31799 REM
SM 31800 IF SSIN THEN RETURN
EU 31801 DEG :DIM SSIN(90):FOR A=1 TO 90:SSIN(A)=SIN(A):NEXT
A:RETURN
```

Grafik-Editor

Existuje několik způsobů, jak kreslit s počítačem, např. v Atari BASICu příkazy PLOT, DRAWTO atd., nebo speciálními grafickými editory, jako jsou programy DESING MASTER, KOALA MICROILUSTRATOR... Pro složitější obrázky je to však nevhodné. Ty se musí nakreslit na milimetrovém papíru a přenést do počítače. A právě k tomu je náš program.

Pracuje v GR. 15, a proto na 1 obrazový bod na obrazovce připadají 2 body na papíře. V GR. 15 se používají 4 barvy.

Ovládání programu je následující:

Klávesy :	Význam :
<0> až <3>	udělá bod barvy 0 až 3
<space>	jako <0>
<return>	přechod na nový řádek
<clear>	smazání obrazovky přechod do levého horního rohu
<insert>	přechod do levého horního rohu
<->	pohyb nahoru
<=>	pohyb dolů
<+>	pohyb vlevo
<*>	pohyb vpravo
<S>	save obrázek
<L>	load obrázek
<D>	přechod do DOSu
<E>	konec

LOAD a SAVE využívá všechny HANDLERy, které jsou k dispozici (normálně je to C: a D:, nebo také T: pro T2000).

Chceme-li přejít do DOSu, případně do TOSu, stiskeme <D>. Pokud nemáte žádný DOS v paměti, vypíše se zpráva >DOS není<. Jinak se vás počítač zeptá >Přechod do DOSu. Myslíte to vážně? (a/n)<. Ovšem pozor! Přejdete-li do DOSu, zničí se vám obsah obrazovky a po ukončení práce v DOSu počítač přejde do Atari BASICu.

Stisknutím <E> se vás počítač zeptá >Skončit? (a/n)<. Volbou A program ukončíme.

Petr Tomášek

Listing MiSe - program GREDIT.BAS Part: 2

```
FB 1310 POKE 904,0:POKE 905,25
TH 1320 A=USR(ADR("hhh3LV0"),64)
OL 1330 CLOSE #4
RS 1340 GOTO 15
JM 1400 IF PEEK(10)=35 AND PEEK(11)=242 THEN ? "¶ DOS neni !!0¶
        ":GOTO 15
OL 1410 ? "¶0 Skok do DOSu."
CA 1420 ? "MYslis to vazne ?[a/n]"
LI 1430 GET #1,A:A$=CHR$(A)
ZE 1440 IF A$="A" OR A$="a" THEN GRAPHICS 0:DOS :END
BO 1450 ? "¶":GOTO 15
LI 2000 TRAP 2000
BX 2010 A$="4":GOTO 110
```

Listing MiSe - program GREDIT.BAS Part: 1

```
HE 5 TRAP 2000
AT 10 GRAPHICS 15:DIM A$(1):DIM N$(20):OPEN #1,4,0,"K"
TB 11 ?
UR 15 POKE 752,1:? " GRAFIK-EDITOR V 1.0 "
ZP 16 ? " by Petr Tomasek "
NG 20 GET #1,A:A$=CHR$(A):Y=Y+1:IF Y>159 THEN Y=0:X=X+1:IF X>15
      9 THEN 300
VF 30 IF A$="-" THEN GOTO 220
EH 35 IF A$="=" THEN GOTO 240
WB 40 IF A$="*" THEN GOTO 20
SW 45 IF A$="+" THEN GOTO 200
HR 50 IF A$="<" THEN GRAPHICS 15:A$=">"
HO 55 IF A$=">" THEN X=0:Y=0:GOTO 11
SH 60 IF A$=CHR$(155) THEN GOTO 260
GA 65 IF A$="S" THEN 1000
EG 70 IF A$="L" THEN 1200
MZ 75 IF A$="E" THEN 300
CL 80 IF A$="D" THEN 1400
DU 105 IF A$=" " THEN A$="0"
AP 110 IF A$<"0" OR A$>"3" THEN ? "{ CHYBA!!!}":GOTO 15
TX 120 COLOR VAL(A$):PLOT Y,X
PU 140 GOTO 20
VD 200 Y=Y-2:IF Y<0 THEN Y=159:GOTO 220
PP 210 GOTO 20
WZ 220 X=X-1:Y=Y-1:IF X<0 THEN X=159
PT 230 GOTO 20
UZ 240 X=X+1:Y=Y-1:IF X>159 THEN 300
PX 250 GOTO 20
XJ 260 Y=0:X=X+1:IF X>159 THEN 300
QB 270 GOTO 20
HR 300 ? "{ SKONCIT ?[a/n]"
MZ 310 GET #1,A:A$=CHR$(A)
AM 320 IF A$="A" OR A$="a" THEN POKE 752,0:GRAPHICS 0:END
MD 330 IF A$="N" OR A$="n" THEN X=0:? "{":GOTO 15
MV 340 GOTO 310
EE 1000 ? "{@ SAVE  OBRAZKU "
OK 1004 POKE 752,0
DN 1005 ? "Vloz <Zar:jmeno>:";:INPUT #16,N$
OE 1006 POKE 752,1:?
WK 1020 OPEN #4,8,128,N$
YM 1060 POKE 898,11
AB 1080 POKE 900,PEEK(88):POKE 901,PEEK(89)
FT 1090 POKE 904,0:POKE 905,25
SX 1100 A=USR(ADR("hhh!LV@"),64)
OB 1110 CLOSE #4
RI 1120 GOTO 15
ZB 1200 ? "{@ LOAD OBRAZKU"
OO 1204 POKE 752,0
DR 1205 ? "Vloz <Zar:jmeno>:";:INPUT #16,N$
OI 1206 POKE 752,1:?
HR 1249 POKE 764,255
UT 1250 OPEN #4,4,128,N$
YB 1270 POKE 898,7
ZJ 1300 POKE 900,PEEK(88):POKE 901,PEEK(89)
```

SORT 1

ing. Petr Válka, AK Brno

Program SORT 1 je třídící program pro počítače ATARI 800XL/XE a 130XE. Strojový program je určen pro velmi rychlé třídění dat v Basicu. Vzhledem k tomu, že jazyk ATARI Basic neumožňuje definici indexovaných řetězcových polí, je soubor záznamů zapisován do jediného řetězce a umístění v řetězci určuje "index" záznamu.

Program SORT 1 používá známý algoritmus, tzv. SHELL SORT (porovnávací třídění), kterým dokáže setřídit např. 2000 záznamů během necelých 30 vteřin. Při volání strojového programu se zadává adresa počátku řetězce-souboru, délka jednoho záznamu a třídící pole (tj. začátek a konec třídícího pole v záznamu). Program SORT 1 USR je proveden jako demonstrační program, do kterého se zapisují např. účastníci sportovní soutěže: startovní číslo, jméno, příjmení, bodový zisk a pod. Podle volby setřídění lze potom vypsat startovní listinu (podle startovních čísel), jmenný seznam (abecedně) nebo výsledkovou listinu (podle bodového zisku).

Strojový program je relokabilní a je umístěn v řetězcové proměnné, dlouhé 234 B. Startuje se přímo na adrese uložení této proměnné. Šestá stránka paměti tak zůstává volná k dispozici pro případný další strojový program uživatele.

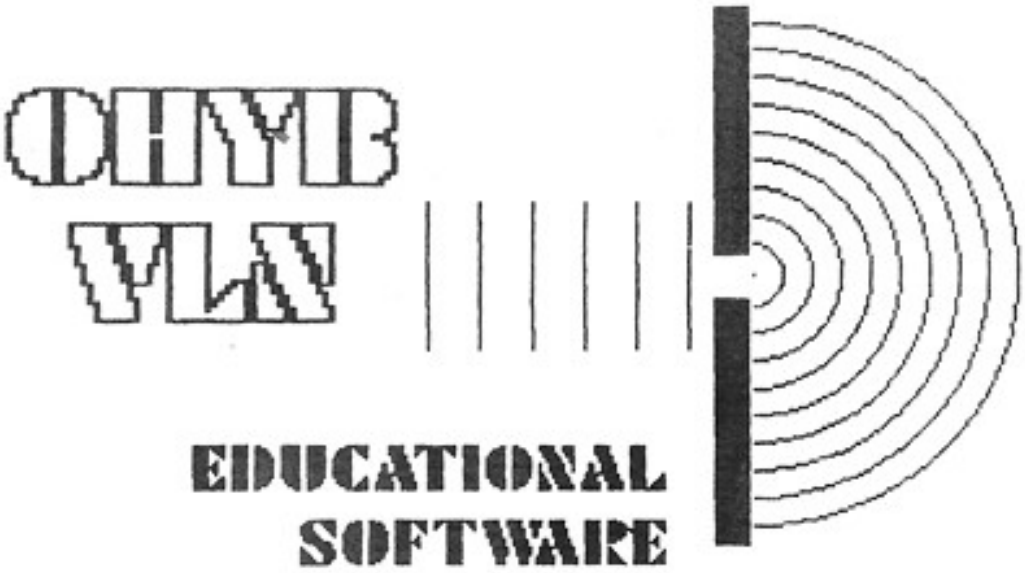
Pro použití s vlastním programem uživatele lze zapsat tři programové řádky obsahující strojový program příkazem LIST na kazetu, disketu, nebo RAMdisk a příkazem ENTER potom přieditovat k vlastnímu programu. Pro natypování programu je k dispozici verze SORT 1 ZDR (zdrojová), která obsahuje strojový kód v DATA-řádcích a rutinu pro samočinné přenesení do řetězcové proměnné. Další programový řádek potom zajistí odmazání již zbytečných DATA-řádků, přenosové rutiny a na závěr vymaže sám sebe.

Listing 1 - program SORTZDR.BAS Part: 2

```
AU 1060 DATA 229,212,133,234,165,211,229
YX 1061 DATA 213,197,215,240,4,176,174
PK 1062 DATA 144,170,165,234,197,214,176
QM 1063 DATA 166,144,162
XB 1064 DIM M$(234):RESTORE 1030
LF 1065 FOR I=0 TO 2:? "{"
TD 1066   POSITION 2,3:? 30020+10*I;" M$(";I*78+1;",";(I+1)*78;
          ")= ";CHR$(34);
HH 1067   FOR J=1 TO 78:READ A:? "E";CHR$(A);:NEXT J:? CHR$(34)
RB 1068   POSITION 2,7:? "N.I:POKE 842,12"
RE 1069   POSITION 0,0:POKE 842,13:STOP
TK 1070   ? "{":FOR I=1030 TO 1070:POSITION 2,2:? I:? "N.I:POKE
           842,12":POSITION 0,0:POKE 842,13:STOP
GH 30000     REM ....... SORT 1.0 .........
NL 30010     DIM M$(234)
DT 30050     RETURN
```

Listing 1 - program SORTZDR.BAS Part: 1

```
OW 10 REM [redacted]
CW 20 REM [redacted]
BA 30 REM
DG 40 DIM D$(1000):D$(1)=" ":D$(1000)=D$:D$(2)=D$:GOSUB 30010:?
      "↰"
EZ 50 ? :? "Delka polozky ";:INPUT DL
DO 60 DIM P$(DL):IP=1
CM 70 ? "Zadavej polozky, ukonci * "
KJ 80 ? IP,:INPUT P$
YS 90 IF P$<>"*" THEN D$(1+(IP-1)*DL,IP*DL)=P$:IP=IP+1:GOTO 80
VP 100 ? :? "Pocatek trideneho pole (od 0) ";:INPUT ZP
LL 110 ? "Konec trideneho pole ";:INPUT KP
JI 120 X=USR(ADR(M$),ADR(D$),IP,DL,ZP,KP)
AE 200 FOR I=1 TO IP
AU 210   ? D$(1+(I-1)*DL,I*DL)
FW 220 NEXT I
OH 230 ? :? "Tridit, prohlizet, konec (T/P/K) ":OPEN #1,4,0,"K:
      "
MI 240 GET #1,ODP:IF ODP=ASC("T") THEN CLOSE #1:GOTO 100
UW 250 IF ODP=ASC("P") THEN CLOSE #1:GOTO 200
IK 260 CLOSE #1:END
CI 1030 DATA 104,104,133,221,104,133,220
GC 1031 DATA 104,133,211,133,213,104,133
EC 1032 DATA 210,133,212,104,104,133,222
HO 1033 DATA 104,104,133,224,104,104,133
QV 1034 DATA 226,70,213,102,212,208,5
CD 1035 DATA 165,213,208,1,96,162,1
UD 1036 DATA 134,214,202,134,215,165,214
GA 1037 DATA 133,216,165,215,133,217,24
ZE 1038 DATA 165,212,101,216,133,218,165
JC 1039 DATA 213,101,217,133,219,162,2
XD 1040 DATA 169,0,149,228,149,229,56
JB 1041 DATA 181,216,233,1,133,232,181
JO 1042 DATA 217,233,0,133,233,165,222
BE 1043 DATA 133,234,160,8,70,234,144
AE 1044 DATA 13,24,165,232,117,228,149
UA 1045 DATA 228,165,233,117,229,149,229
DN 1046 DATA 6,232,38,233,136,208,232
TS 1047 DATA 24,165,220,117,228,149,228
NI 1048 DATA 165,221,117,229,149,229,224
VW 1049 DATA 0,240,10,162,0,240,186
VE 1050 DATA 144,143,176,159,176,165,164
XA 1051 DATA 224,177,228,209,230,208,7
EC 1052 DATA 200,196,226,208,245,240,40
WR 1053 DATA 144,38,160,0,177,228,72
VZ 1054 DATA 177,230,145,228,104,145,230
NZ 1055 DATA 200,196,222,208,241,56,165
DY 1056 DATA 216,229,212,133,216,165,217
DS 1057 DATA 229,213,133,217,208,205,165
SB 1058 DATA 216,201,1,176,199,230,214
XL 1059 DATA 208,2,230,215,56,165,210
```



Atari simuluje vlnové jevy

Jan Golla

U všech 8-bitových počítačů Atari můžeme použít několika vlastností, které vytváří zajímavé grafické efekty. Mezi ně patří i tzv. page-flipping (přepínání obrázků uložených v paměti RAM). Touto metodou jsem dosáhl pěkných výsledných efektů při zobrazování ohybu, odrazu a lomu vln. Jednotlivé fáze pohybu jsem uložil do paměti počítače a potom jsem je promítal na obrazovku v rychlém sledu po sobě (na podobném principu pracuje i film). Jednotlivé obrázky se od sebe velmi málo liší, což při jejich rychlém zobrazování vyvolává zdání pohybu.

Tak jsou v programu řešeny všechny jevy kromě části "ohyb jedné vlny". Během inicializace počítač postupně ukládá do své paměti jednotlivé obrázky. V hlavní části programu pak dochází k jejich přepínání. Pomocí kláves F (=Fast - rychleji) a S (=Slow - pomaleji) volíme nejvhodnější rychlost pozorování jevů. Upozorňuji, že nejdéle (asi 140 sekund) trvá inicializace obrazů při simulaci ohybu vln. Programy jsou napsány v jazyku Atari BASIC a jsou opatřeny kontrolními kódy TYPO II.

Doufám, že tyto simulační programy naleznou uplatnění ve výuce na školách všech typů a že ukazují přednosti počítačů Atari proti používaným školním mikropočítačům.

Přeložil a upravil: ing. Jeroným Liška
Jiří Hrdlička

Vzhledem k nedostatku místa ve Zpravodaji otiskneme třetí program v následujícím čísle. Děkujeme za pochopení.

Redakce AZ Olomouc

Listing 1 - program WAVE1.BAS Part: 1

```
HX 10 REM ┌──────────┐
DD 20 REM │ OHYB VLN │
JH 30 REM └──────────┘
CS 40 REM NEW version!
BY 50 REM - educational software
FC 60 REM - for all ATARI XL/XE
BN 70 REM - by GIA Software
SR 80 REM - (c) 1990
SG 90 REM Part 1
ES 100 GRAPHICS 2:POSITION 5,4:? #6;"ODRAZ VLNY":POKE 657,8
ZT 110 ? "←STISKNI LIBOVOLNOU KLAVESU":POKE 764,255:POKE 755,0:
       SETCOLOR 2,0,0
ZQ 120 IF PEEK(764)=255 THEN 120
CS 130 GRAPHICS 0:SETCOLOR 2,0,0:POKE 82,2:POKE 83,38
OR 140 POSITION 2,4:? "PROGRAM UKAZUJE ODRAZ VLNY POD":? "ZVOLE
       NYM UHLEM"
LH 150 TRAP 130
XR 160 POSITION 2,7:? "ZADEJ UHEL DOPADU (0-70)";:INPUT K:DEG :
       OP=10
VY 170 IF K>70 OR K<0 THEN 130
MT 180 POSITION 2,10:? "CEKEJ ASI 30 SEKUND...":POKE 755,0
DU 190 FOR O=1 TO 750:NEXT O
PJ 200 POKE 106,160
MG 210 FOR E=1 TO 5
WS 220   POKE 106,160-E*16
IJ 230   GRAPHICS 6:COLOR 1:POKE 559,0
AW 240   SI=SIN(K):CO=COS(K)
IA 250   POKE 82,6:POKE 755,0:POKE 657,0
WJ 260   ? "                                    "
VA 270   ? "KLAVESA [F] - RYCHLEJI (FAST)":? "KLAVESA [S] - POM
         ALEJI (SLOW)"
DL 280   ? "KLAVESA [ESC] - NOVY UHEL";
GU 290   POKE 710,0
LD 300   PLOT 80,1:DRAWTO 80,80:SETCOLOR 2,0,0:POKE 709,0
ZZ 310   FOR PZ=-10 TO 110 STEP 5
MT 320     TRAP 350
VW 330     PLOT 80+(PZ+E)*SI-12*CO,80-(PZ+E)*CO-12*SI
EM 340     DRAWTO 80+(PZ+E)*SI+12*CO,80-(PZ+E)*CO+12*SI
UX 350   NEXT PZ
BH 360   FOR PZ=-10 TO 80 STEP 5
LK 370     TRAP 400
ZK 380     PLOT 80-(PZ-E)*SI+12*CO,80-(PZ-E)*CO-12*SI
MA 390     DRAWTO 80-(PZ-E)*SI-12*CO,80-(PZ-E)*CO+12*SI
UO 400   NEXT PZ
EW 410   FOR FA=78 TO 83 STEP 0.05:TRAP 440
CO 420     PLOT 80-FA+SI+14*CO,80-FA*CO-14*SI
FV 430     DRAWTO 80-FA*SI-14*CO,80-FA*CO+14*SI
GL 440   NEXT FA
EQ 450 NEXT E
RC 460 GRAPHICS 6:POKE 559,34
PT 470 FOR EK=135 TO 87 STEP -16:POKE 561,EK
```

Listing 1 - program WAVE1.BAS Part: 2

```
AU 480   FOR AS=1 TO OP
GW 490     POKE 710,0
DB 500     IF PEEK(764)<255 THEN GOTO 560
LY 510   NEXT AS
KE 520 NEXT EK
PT 530 GOTO 470
MC 540 REM REGULACE RYCHLOSTI
GP 550 POKE 710,0
RL 560 IF PEEK(764)=28 THEN CLR :GOTO 130
VJ 570 IF PEEK(764)=62 THEN OP=OP+1
RV 580 IF PEEK(764)=56 AND OP>2 THEN OP=OP-1
ZH 590 POKE 764,255:GOTO 480
```

Listing 2 - program WAVE2.BAS Part: 2

```
EP 520 ? "RIZENI RYCHLOSTI:":? :? "SNIZENI - STISKNI KLAVESU S
       (SLOW)":? "ZVYSENI - STISKNI KLAVESU F (FAST)"
YM 530 ? :? "PRO ZASTAVENI PROGRAMU STISKNI":? "KLAVESU ESC"
GT 540 POSITION 9,18:? "PROBIHA INICIALIZACE"
HP 550 POSITION 8,20:? "CEKEJTE ASI 140 SEKUND..."
LQ 560 FOR RB=1 TO 6000:NEXT RB
HV 600 REM OHYB RADY VLN
MO 610 FOR ST=1 TO 5
DQ 620   GRAPHICS 6+16:SETCOLOR 2,0,0:COLOR 1:POKE 559,0:POKE 1
         06,160-ST*16
UF 630   FOR G=10 TO 15:PLOT G,20:DRAWTO G,75:NEXT G
GD 640   FOR P=95 TO 100:PLOT P,1:DRAWTO P,45:PLOT P,50:DRAWTO
         P,95:NEXT P
PP 650   FOR F=9+ST TO 94+ST STEP 5:PLOT F,20:DRAWTO F,75:NEXT
         F
PU 660   FOR RK=1+ST TO 72 STEP 6
IA 670     FOR N=2 TO 57:XO=MH(N)*RK+100:YO=AK(N)*RK+47
CA 680       TRAP 690:PLOT XO,YO:PLOT XO,95-YO
SV 690     NEXT N:NEXT RK:NEXT ST
PY 700 POKE 764,255:POKE 77,0:POKE 764,255
NC 710 POKE 708,40:POKE 709,202:POKE 559,34
LS 720 FOR EK=135 TO 64 STEP -16:POKE 561,EK
BL 730   FOR OP=1 TO OP1:IF PEEK(764)<255 THEN 810
HG 740   NEXT OP:NEXT EK
PA 750 GOTO 720
LX 800 REM REGULACE RYCHLOSTI
BW 810 IF PEEK(764)=62 OR OP1<0 THEN OP1=OP1+0.04
ME 820 IF PEEK(764)=28 THEN END
XF 830 IF PEEK(764)=56 OR OP1>50 THEN OP1=OP1-0.4
XZ 840 POKE 764,255:GOTO 740
```

Listing 2 - program WAVE2.BAS Part: 1

```
HX 10 REM ┌──────┐
DD 20 REM │ OHYB VLN │
JH 30 REM └──────┘
CS 40 REM NEW version!
BY 50 REM - educational software
FC 60 REM - for all ATARI XL/XE
BN 70 REM - by GIA Software
SR 80 REM - (c) 1990
ST 90 REM Part 2
OE 100 DIM MH(60),AK(60):POKE 566,158:POKE 764,255:POKE 83,37
CB 110 GRAPHICS 2:SETCOLOR 2,0,0:POKE 755,0:POSITION 6,4
ZW 120 ? #6;"OHYB VLN"
FP 130 POKE 657,8:? "STISKNI LIBOVOLNOU KLAVESU"
BY 140 IF PEEK(764)=255 THEN 140
AG 150 GRAPHICS 0:SETCOLOR 2,0,0:POSITION 2,5:POKE 752,1
LF 160 POSITION 2,1:? "PROGRAM MA DVE CASTI:"
RC 165 POSITION 2,2:? "-----------------------":?
YW 170 ? :? "CAST 1: OHYB JEDNE VLNY":? "CAST 2: OHYB RADY VLN"
JN 180 POSITION 2,9:? "KLAVESA [ESC] ZPUSOBI PRECHOD":? "DO DRU
       HE CASTI"
OM 190 POSITION 2,15:? "PROBIHA INICIALIZACE PRVNI CASTI"
RD 200 POSITION 2,17:? "CEKEJTE ASI 30 SEKUND..."
BG 210 FOR MT=1 TO 1000:NEXT MT
FM 300 REM OHYB JEDNE VLNY
OW 310 FOR KO=6.3 TO 8 STEP 0.03:N=N+1:MH(N)=SIN(KO):AK(N)=COS(
       KO):NEXT KO
FS 320 BS=PEEK(106)-16:POKE 106,BS:POKE 559,0:POKE 54279,BS
AI 330 MS=BS*256+1024+72
BZ 340 FOR X=MS TO MS+109:POKE X,128:NEXT X:FOR ST=0 TO 5
DZ 350   POKE 106,160-ST*16:GRAPHICS 6+16:COLOR 1:POKE 559,0:GO
         SUB 470
YE 360   RK=10+ST*10:FOR N=2 TO 52:XO=MH(N)*RK+100:YO=AK(N)*RK+
         47
OV 370     TRAP 390
SM 380     PLOT XO,YO:PLOT XO,95-YO
TJ 390   NEXT N:NEXT ST
AF 400 GRAPHICS 6+16:POKE 53277,14:POKE 559,62:GOSUB 470
LC 410 POKE 704,8:FOR PK=60 TO 148 STEP 0.8:POKE 53248,PK
BT 420   FOR RM=1 TO 7:IF PEEK(764)=28 THEN POKE 53248,0:GOTO 4
         90:NEXT RM
DS 430 NEXT PK:POKE 704,0
AQ 440 FOR GD=151 TO 64 STEP -16:POKE 561,GD
AR 450   FOR PZ=1 TO 14:IF PEEK(764)=28 THEN POKE 53248,0:GOTO
         490
PO 460   NEXT PZ:NEXT GD:GOTO 410
GH 470 FOR P=95 TO 100:PLOT P,1:DRAWTO P,45:PLOT P,50:DRAWTO P,
       95:NEXT P
YW 480 FOR KZ=10 TO 15:PLOT KZ,20:DRAWTO KZ,75:NEXT KZ:RETURN
SQ 490 GRAPHICS 0:SETCOLOR 2,0,0:POKE 752,1:OP1=20:POSITION 9,4
XX 500 ? "CAST 2: POHYB RADY VLN"
VX 510 POSITION 9,5:? "----------------------":?
```

MIKRONOTES verze 3

Doplňky k návodu verze 1

1. Program Mikronotes 3 umožňuje zavedení DOSem nebo TOSem. Tím je použitelný i pro uživatele disketové jednotky, aniž by museli obětovat pro každý soubor jednu disketu (viz návod pro verzi 1). Zavedení programu pomocí TOS 4.1 doplňuje možnost vstupů a výstupů (FORM.TISK, ZAPS, DATA ...) na zařízení D - magnetofon TURBO 2000, nebo D8 - RAMdisk. Při použití DOSu nebo TOSu si program sám zajistí ukládání dat od 32. stránky paměti, namísto od 6. stránky při zavedení zavaděčem TURBO 2000. Paměť pro soubor je potom jen 31,5 kB namísto 38 kB.

2. Při vstupu dat se program Mikronotes 3 dotazuje:

OPAKOVAT POLE? A/N

Po odpovědi "A" je třeba písmenem D označit ta pole, která se mají při vstupu dat duplikovat z předchozího záznamu. Ukončení označování polí tlačítkem START. Označená pole lze ovšem přepsat, nová hodnota je dále duplikována.

3. Při volbě SUM nebo PRU žádá program ještě kriteria výběru (viz PROHLEDÁVÁNÍ). Výpočet součtu nebo průměru numerických polí je potom proveden jen pro záznamy, odpovídající výběru. Při stisku START bez zadání kriterií výběru se provedou matematické operace se všemi záznamy.

4. Při volbě ZAPS (zápis na magnetofon, nebo disketu) se program dotazuje:

SOUBOR / DATA

Při volbě "S" je zapsán celý soubor jako ve verzi 1. Volba "D" provede zápis samotných dat (bez popisu formuláře, formátovacích povelů ap.). Zápis se provádí od aktuálního (naposledy zobrazovaného) záznamu až do konce souboru.

5. Nová funkce DATA v MENU 1 umožňuje připojení dat k existujícímu souboru, nebo formuláři v počítači. Předpokladem je stejná struktura (formulář). Data se připojí opět v místě aktuálního záznamu, tento záznam je přepsán prvním záznamem přihrávaných dat. Pro sřetězení dvou souborů je třeba doplnit jeden záznam (doplní se na konec) a potom použít funkci DATA.

6. Pro volbu FORM.T. při zvoleném zařízení P (tiskárna) se program dotazuje na horní okraj a počet řádků na stránku (včetně horního okraje). Po vytisknutí zadaného počtu řádků odřádkuje tiskárna do 72 řádků a program čeká na stisk SELECT - pokračování tisku, nebo START - ukončení tisku.

7. Pro formátovaný tisk (FORM.T.) lze zvolit i zařízení C

(magnetofon standard) nebo D (disketa, případně magnetofon TURBO, byl-li program zaveden pomocí TOS 4.1) nebo D8 (RAMdisk). Tato funkce umožňuje provádět změny struktury "formuláře", aniž by uživatel pozbyl data v souboru:
a) změna pořadí polí - při volbě ZMENA FORMATU označit pole číslem 0 v požadovaném pořadí. Provést "tisk" na kazetu, nebo disketu. Upravit formulář pro změněné pořadí polí a funkcí DATA přihrát data k formuláři.
b) prodloužení polí. Zadáním čísla 1 až 9 lze prodloužit pole při formátování "tisku". Další postup je shodný s bodem a)
c) vypuštění polí - při formátování se vypouštěná pole přeskočí. Dále viz bod a)
d) zkrácení polí se provede nadvakrát. Nejprve se data zapíší funkcí ZAPS s volbou DATA. Nový formulář se upraví ze starého rozdělením příslušného pole (klávesami CTRL-INS). Data se přihrají k upravenému formuláři. Druhá etapa je vlastně vypuštění pole viz bod c).
e) přidání nových polí se provede prodloužením předchozího pole o 1 až 9 mezer. Nový formulář však namísto prodloužení má nové pole 1 až 9 znaků. Nově přidaný název pole nemá na soubor vliv.
Další úpravy se provádějí kombinací, nebo opakováním uvedených činností.

8) Možnosti třídění byly rozšířeny:

/ - vzestupné třídění podle ASCII kódů znaků
  - sestupné třídění podle ASCII kódů znaků
< - vzestupné třídění abecední (nerozlišuje velká/malá písmena)
> - sestupné třídění abecední

9) Možnosti výběru byly rozšířeny:

| - výběr podle ASCII kódů znaků
! - výběr abecední
  - výběr podle ASCII kódů ve všech polích záznamu.

10) V titulní obrazovce jsou kromě volby 1. 2. a 3. možné tyto další volby:

B - změna barvy obrazovky (16 barev)
O - změna odstínu (8 odstínů až do inverze)
CTRL-D - přechod do DOSu, případně do TOSu. Návrat z TOSu volbou G na adrese $100.

3x s BAREVNOU GRAFIKOU

128 barev v GR.0

Program COL128.BAS je napsán v jazyku Atari BASIC a je opatřen kódy TYPO II. Využívá přerušení display listu. Vlastní rutina je ve strojovém kódu, BASIC slouží pouze k vložení procedury do 6. stránky paměti. Před inicializací strojové procedury je třeba vypnout 7 bit adresy ANTICu (informace o přerušení DLI). Dále na paměťové buňky 512 a 513 vložíme adresu začátku procedury DLI. Příkazem POKE 54286,192 "žádáme" počítač o vydání povolení pro DLI.

Atari RAINBOW

Program COLMOVE.BAS je napsán v Atari BASICu a je opatřen kontrolními kódy TYPO II. Umožňuje získat známou "duhu" 8-bitových počítačů Atari. "Duha" je znázorněna vodorovnými barevnými pohyblivými pruhy (256 barev). Strojová rutina se umístní do 6. stránky paměti RAM. Příkazy POKE jsou podobného typu jako u programu COL128.BAS.

Statická "DUHA"

Program COLGR2.BAS je napsán v Atari BASICu a je opatřen kódy TYPO II. Umožňuje "stínování" textu i pozadí v grafickém módu 2. Řádek s textem (řádek 150) můžeme samozřejmě libovolně modifikovat. V programu je použita procedura, která mění registry barev COLBAK a COLPFO. Samotná procedura je velmi jednoduchá a je složená ze dvou částí. První část zvyšuje obsah COLBAK a snižuje obsah COLPFO, druhá pracuje přesně naopak. Tímto způsobem dosahujeme výsledného efektu "stínování" barev. Změnám se zda meze nekladou, proto můžeme v řádku 140 změnit příkaz POKE DL+1,112+128 na POKE DL+2,112+128 a tím dostaneme opačné umístění barev. Můžeme také modifikovat hodnoty na adresách 1944 a 1942. První z nich řídí startovací registr COLBAK, který normálně obsahuje hodnotu 0 (černá barva). Když zde ale zapíšeme násobky 16 (16,32,48,...), získáme jinou barvu pozadí. Analogicky postupujeme i u registru COLPFO, kde měníme obsah buňky 1942. Normálně je zde hodnota 14 (bílá barva). Můžeme sem zapisovat hodnoty podle vztahu 16*N+14, kde N je kód barvy. Tento program můžeme po malé úpravě spojit s předchozími programy

ing. Jeroným Liška
Jiří Hrdlička

Listing 1 - program COL128.BAS Part: 1

```
SJ 10 FOR A=1536 TO 1554
RE 20 READ X:POKE A,X:NEXT A
ZY 30 POKE 39980,130:POKE 512,0:POKE 513,6:POKE 54286,192
YI 40 DATA 72,138,72,162,0,142,10,212,142,24,208,232
UQ 50 DATA 232,208,246,104,170,104,64
```

Listing MiSe - program COLMOVE.BAS Part: 1

```
TZ 10 FOR I=1536 TO 1562
DU 20 READ X:POKE I,X:NEXT I
AR 30 POKE 39970,240:POKE 512,0:POKE 513,6:POKE 54286,192
LR 100 DATA 72,138,72,152,72,166,200,160,192
YN 110 DATA 141,10,212,142,24,208,232,136,208,246,230,200
RK 120 DATA 104,168,104,170,104,64
```

Listing MiSe - program COLGR2.BAS Part: 1

```
FP 100 GRAPHICS 2+16
GZ 110 FOR A=1536 TO 1610
CN 120 READ X:POKE A,X:NEXT A
FR 130 DL=PEEK(560)+256*PEEK(561)
TJ 140 POKE DL+1,112+128:POKE 512,0:POKE 513,6:POKE 54286,192
XS 150 FOR A=0 TO 9 STEP 3:POSITION 3,A:? #6;"ATARI 130 XE":POS
       ITION 3,A+1:? #6;"GRAPHICS II":NEXT A
OG 160 GOTO 160
ND 170 DATA 72,138,72,152,72,162,14,160,0,169,12,133,201,169,8
YP 180 DATA 133,200,141,10,212,140,26,208,142,22,208,200,200,20
       2
IE 190 DATA 202,198,200,208,239,169,8,133,200,136,136,232,232
ET 200 DATA 141,10,212,140,26,208,142,22,208,198,200,208,239
EQ 210 DATA 152,24,105,16,168,138,24,105,16,170,198,201,208
HX 220 DATA 200,104,168,104,170,104,64
```

Pomocí dvou programů si můžete nahrát z magnetofonu do počítače různé zvuky, zpomalit je nebo zrychlit, vybrat si určité úseky, řadit je za sebe, opakovat je a nahrát si výsledný soubor jako samostatný program. Oba programy využívají celou paměť RAM (i pod ROM), a proto je pro správnou funkci programu nutné mít alespoň 64 kB RAM (ATARI 800 XL a novější).

SoundDigitalizer 1.0

Výběr jakosti

Po nahrátí programu zavaděčem ZXL (oba programy jsou v relativním formátu) se objeví název programu, číslo verze a nápis "Vyber si jakost". Výhodu mají tentokrát majitelé magnetofonů bez příposlechu, kteří při výběru jakosti slyší počítačovou verzi nahrávky. Jakost se volí pohybem joysticku 1 nahoru a dolů. Nejkvalitnější je jakost 001, která však zaplní za sekundu asi 2200 bytů a do paměti se tak vejde jen asi 27 sekund záznamu (SAMPLER MUSIC hraje 28 sekund). Nejméně kvalitní je jakost 000, která je použitelná snad jen při záznamu velmi hlubokých jednoduchých zvuků, zato se ale do paměti vejde asi 6 minut 40 sekund. Číslo jakosti vlastně udává, jak dlouho má program čekat po každém průchodu smyčkou. Obecně platí, že čím jsou v nahrávce vyšší frekvence, tím musíme použít kvalitnější jakost. Použitelné jakosti jsou asi od 1 do 50, počáteční nastavení je 25. Volbu jakosti ukončíme stiskem tlačítka na ovladači.

Digitalizace

Počítač třikrát zahouká a po stisku klávesy se začne nahrávka zapisovat do paměti.

Při digitalizaci i přehrávání jsou zakázána všechna přerušení, tedy i od procesoru ANTIC, a tím je vyloučeno zobrazování. Interface pro TURBO předělá všechny zvuky na obdélníkový signál, proto mohou být některé zvuky (např. bubny) dost podstatně zkresleny. Převodem na obdélníkový signál se také zesilují šumy, je tedy třeba mít kvalitní nahrávku.

Menu

Když se zaplní celá volná paměť (58 kB - $E800 bytů), objeví se menu. Vybíráte si stiskem čísla 1-5, všechny činnosti můžete ukončit stiskem tlačítka RESET, dostanete se zpět do menu. Po stisku RESET se ale systémem automaticky nastaví DL a obrazová paměť v oblasti $BC20-$BFFF, takže se smaže max. 7 sekund nahrávky (při jakosti 0). Tuto nepříjemnou vlastnost systému se zatím nepodařilo obelstít.

Stiskem 1 si můžete přehrát celou paměť tak, jak byla zdigitalizována.

Po stisku klávesy 2 si můžete vybrat z nahrávky určitou část (úsek). Paměť se začne přehrávat pomaleji a stiskem klávesy SHIFT určíme začátek úseku. Pohybem joysticku doleva se začne paměť přehrávat znovu od začátku. Opětovným stiskem SHIFTu určíme konec úseku a joystickem doleva se paměť začne přehrávat znovu od začátku úseku.

Volby 3 a 4 jsou použitelné až po výběru úseku. Stiskem klávesy 3 si můžeme nahrát vybraný úsek na kazetu ve formátu TURBO 2000 a stiskem 4 si ho můžeme přehrát.

Stiskem klávesy 5 se dostaneme opět na výběr jakosti. Zpět se můžeme dostat stiskem RESET bez ztráty dat (kromě oblasti $BC20-$BFFF) . Této vlastnosti lze využít při změně rychlosti. Níže popsaná "finta" dá podobný výsledek, jako když na gramofonu nastavíte rychlejší nebo pomalejší otáčky.

Zrychlení nebo zpomalení

Běžným způsobem zdigitalizujte zvuk a zapamatujte si zvolenou jakost. Po naplnění paměti stiskněte klávesu 5. Počítač se vás opět ptá na jakost. Pokud chcete větší rychlost než původně, zadejte kvalitnější jakost než při předchozí digitalizaci, jinak nějakou méně kvaliní. Nemusíte ani mačkat FIRE a stiskněte RESET. Tím jsme vlastně programu nalhali, že má záznam nahraný v jiné jakosti, a podle toho bude při přehrávání dělat pauzy.

Příklad: Chceme zpomalit záznam zdigitalizovaný v jakosti 20. Běžným způsobem jej zdigitalizujeme. Až se objeví menu, stiskneme klávesu 5. Zvolíme větší číslo jakosti, např. 35. Po stisku RESET pracujeme se záznamem jako normálně.

Při volbě "SAVE úsek" program zapíše jakost do 2.byte hlavičky (typ). Tak můžete jakost (a tím i rychlost) jednoduše změnit např. v SUPERMONu.

Chyba v podprogramu SAVE

Až po uveřejnění programu byla zjištěna chyba v rutině pro nahrávání na kazetu. Program totiž využívá téměř celou paměť RAM, tedy i pod operačním systémem ($C000-$FFFF). Při nahrávání z této oblasti je třeba přeskočit oblast periferních obvodů ($D000-$D7FF). Pro tento přeskok jsem rutinu pro SAVE opravil, ale při přeskoku se ještě musí snížit délka o 2 kB. Rutinu pro LOAD v SoundStudiu jsem této chybě přizpůsobil, ale úseky, ve kterých se přeskakovaly periferní obvody, můžeme kopírovat pouze přes program SUPERMON, který při ZX-LOAD nehledí na délku uvedenou v hlavičce.

SoundStudio 1.0

Menu
---

1) LOAD usek - běžné LOAD ve formátu TURBO 2000. V rutině je zabudován příposlech i pro magnetofony, které jinak fungují bez příposlechu. Do paměti můžete nahrát max. 40 úseků (pokud se tam vejdou).

2) Seznam useku - při této volbě program ve dvou sloupcích vypíše úseky, které máme v paměti. U každého úseku je jeho pořadové číslo, název a jakost.

3) Prehrat usek - po stisku klávesy 3 se počítač zeptá na číslo úseku. Zadáváme běžným způsobem přes klávesnici, zadávání ukončíme stiskem RETURN nebo SPACE BAR. Jestliže jsme zadali neexistující číslo úseku, počítač se ptá znovu.

4) LOAD soubor - touto volbou načteme do paměti dříve vytvořený soubor.

5) Edit soubor - po stisku klávesy 5 se dostaneme do editoru. Ve čtyřech sloupcích je vypsáno pořadové číslo, číslo úseku a počet opakování. V první kolonce bliká kurzor. Pohybujeme jím pomocí kurzorových šipek (bez CONTROL) a čísla zadáváme stejně jako u volby 3. Chceme-li např. 4krát opakovat úsek číslo 5, napíšeme do prvního řádku (nebo kteréhokoliv jiného) : 05 (RETURN, šipka doprava) 04
Do menu se vrátíme stiskem ESC.

6) Prehrat soubor - po stisku klávesy 6 počítač přehraje vytvořený soubor.

7) SAVE soubor - i zde platí chyba popsaná na konci popisu pro SoundDigitalizer.

8) Nahrat soubor - po této volbě počítač 4krát zahouká a vytvořený soubor si můžete nahrát jako zvuky na magnetofon (např. pro Sinclairisty, kteří puknou závistí). Pomocí equalizeru můžete nahrávku tak přefiltrovat, že nebude mít daleko k originálu.

9) Vytvorit program - díky této funkci získáte program, který si můžete nahrát přímo do zavaděče. Autoři standardních zavaděčů ale nepočítali s programy, které by využívaly i paměť RAM pod ROM, a proto při nahrávání nechali ROM zapnutou (i když ji nepotřebují) a nepočítali ani s přeskokem periferních obvodů. Z toho vyplývá jeden důsledek - správně budou fungovat jen ty programy, u kterých byla volná paměť alespoň $3800 bytů, nezasáhly tedy pod ROM.

0) Restart - po stisku 0 se počítač ještě jednou zeptá a pak zruší všechny úseky, tedy i celý soubor.
Dole je napsáno, kolik úseků máme právě nahraných a kolik nám ještě zbývá volné paměti (na začátku $E000 - 56 kB).

Všechny funkce lze přerušit stiskem RESET, ovšem za cenu ztráty paměti v oblasti $BC20-$BFFF.

Příjemnou práci s programy SoundDigitalizer a SoundStudio přeje

Myroush Trochta, HardSoft, Ltd, s.p.
Valašské Meziříčí

SVAZARM BRNO

PROSTOROVE GRAFY

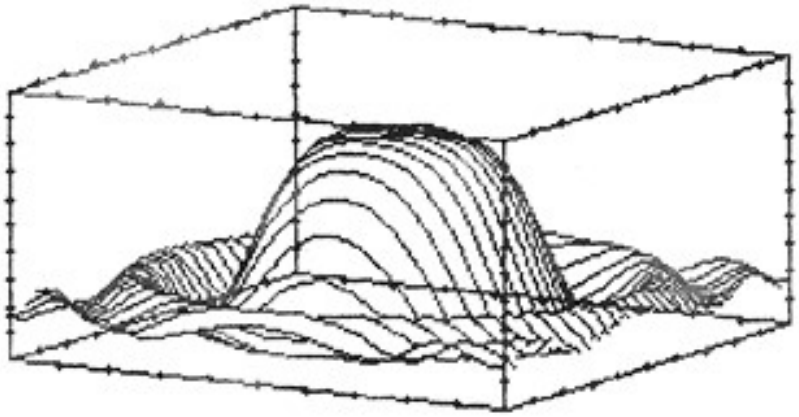
Popis programu

Program PROSTOROVE GRAFY je určen k využití grafických schopností počítačů ATARI 800XL/XE a 130XE. Po zadání funkce z=f(x,y) se provádí výpočet bodů zborcené plochy. Podle volby zadání a zvoleného počtu řezných rovin jsou vykreslovány křivky řezu plochy. Řezy je možno volit podélně, příčně, nebo obojí. Kromě toho lze potlačit vykreslování neviditelných čar. Volitelný je úhel natočení souřadných os a úhel nadhledu.

Program kreslí v grafickém režimu GRAPHICS 8, který má nejvyšší rozlišitelnost (320 x 192 bodů). Graf je možno doplnit popisem funkce a zakreslením souřadných os.

Příklad výstupu z programu PROSTOROVE GRAFY:

```
GRAF FUNKCE   Z=SIN(X*X+Y*Y)/(X*X+Y*Y)
                                X:-3     3
                                Y:-3     3
                                Z:-0.3171.2
```



Grafický výstup je možno zapsat na výstupní zařízení (magnetofon, nebo disketu) pro případné vytisknutí programem Hardcopy, nebo z textového editoru Čížek 3, jak ukazuje právě tento popis.

Program PROSTOROVE GRAFY je naprogramován v Turbobasicu. Byl upraven z polského originálu, zveřejněného v časopisu Bajtek. Úpravy programu byly provedeny v Atari klubu Svazarmu Brno.

ELEKTRO HOBBY S POČÍTAČEM ATARI

Vytvoření spínacího obvodu s jedním tranzistorem

Zájemci o elektroniku nebo výpočetní techniku, kteří často přichází do styku s číslicovými obvody, mi potvrdí, že jsou "terorizováni" různým propojováním přístrojů, čili konstruováním různých interfaců. Ale aby mohli tyto nadmíru užitečné věci (zvláště pro počítače) vytvářet, musí si rozumět i s vnějšími obvody: s relé, s motory, žárovkami, LED diodami, atd. Není to tak zlé, když pracujeme se standartními výstupy TTL. Ale co máme dělat, když máme řídit motor o výkonu 5 HP pomocí invertoru CMOS? Samotná metoda je jednoduchá (spínač), výpočet je složitější. V tomto článku se budeme zabývat návrhem spínače s tranzistorem. Výsledkem řešení jsou normalizované hodnoty odporů v zapojení. Výpočet je doplněn programem v Atari BASICu. Mezi vlastnosti tohoto programu patří i to, že po provedených výpočtech nám předá hodnoty normalizované řady odporů (E24).

Před zahájením jakýchkoliv výpočtů si ujasněme situaci. Uvědomme si, že k vytvoření spínacího obvodu nám stačí jeden tranzistor, který zvládne tuto funkci. Jestliže naše řídící zařízení je PIA 6520 nebo PIO Z80 nebo něco podobného, a chceme ovládat třeba NC soustruh, kde je motor o 5HP, budeme potřebovat několik tranzistorů ke zhotovení řídící jednotky, která bude schopna ovládat dostatečně velké relé a toto potom vlastní motor.

Jestliže však potřebujeme zapínat a vypínat LED diodu, můžeme použít tranzistorový řídící obvod. Ten dostaneme použitím předřadného rezistoru, i když tento způsob neumožní LED diodě svítit plným jasem.

Pokud bude pro nás tuto práci vykonávat obvod s jedním tranzistorem, musíme si vybrat správný typ tranzistoru. Zdvojnásobme napětí a proud potřebný v zátěži a podle těchto parametrů vyhledejme v katalogu tranzistor. Raději jich vyhledejme hned několik pro případ, že by byly nedostupné nebo příliš drahé.

Každý tranzistor má stejnosměrný zesilovací činitel nazývaný BETA. Naneštěstí se hodnoty BETA liší i mezi tranzistory stejného typu a navíc jsou také závislé na zátěži a teplotě.

My vezmeme průměrnou hodnotu BETA - tj. hodnotu mezi minimální a maximální hodnotou uvedenou v katalogu. Pokud nemůžeme určit hodnotu BETA, použijíme následující pravidlo: pro malovýkonnový tranzistor vezmeme BETA=100 a pro výkonnový tranzistor vezmeme BETA=10. Toto nám umožní odhadnout hodnotu předřadného rezistoru v obvodu báze (viz R1 na obr. 1). Protože proud báze je definován jako proud kolektoru děleno BETA, obdržíme hodnotu proudu báze vydělením velikosti proudu kolektoru 100 nebo 10. Ale než začneme hledat hodnotu odporu R1, musíme znát hodnotu odporu R2.

Vlastní výpočet

Abychom vypočítali hodnoty odporů ve spínacím obvodu (obr. 1) s minimální námahou, začneme postupovat od zátěže. Musíme znát, jaký proud protéká zátěží a jaký odpor má zátěž. Z těchto hodnot potom vypočítáme úbytek napětí U(L). Podle Kirchoffova zákona platí:

U(CC) = U(L) + U(R2) + U(CE),

kde U(CE) je úbytek napětí mezi kolektorem a emitorem T1 (viz obr. 1). Toto napětí je většinou kolem 0.3 V, takže ho můžeme ignorovat. Tímto zjednodušením a upravením původní rovnice dostaneme tento vztah:

U(R2) = U(CC) - U(L).

My však potřebujeme znát odpor R2. Ten vypočítáme pomocí Ohmova zákona. Opět upravíme stávající rovnici a dostaneme:

R2 = (U(CC) - U(L)) / I(L).

Jestliže jako zátěže použijeme LED diodu, můžeme za U(L) dosadit hodnotu 1.5 V. Pokud nemáme LED diodu zapojenu, U(L) musí být rovno I(L) * R(L). V tomto případě vypočítáme hodnotu R2 z upraveného původního vztahu:

R2 = (U(CC) - (I(L) * R(L))) / (I(L)).

K výpočtu hodnoty odporu R1 využijeme hodnotu BETA. Protože platí vztah BETA = I(C) / I(B), můžeme vyjádřit proud kolektoru I(B) jako I(C) / BETA.

Podle Theveninovy poučky víme, že I(CC) se musí rovnat I(L), z čehož vyplývá vztah pro I(B) = I(L) / BETA. Nyní opět využijeme Kirchoffův zákon a vyjádříme vstupní napětí U(IN) tímto vztahem:

U(IN) = U(R1) + U(BE),

kde U(BE) je úbytek napětí mezi bází a emitorem T1. Celý problém zjednodušíme tím, že budeme považovat U(BE) za nulové, takže úbytek napětí na R1 se bude rovnat vstupnímu napětí. Proto může být hodnota odporu R1 vyjádřena tímto vztahem:

R1 = U(IN) / I(B).

Protože I(B) můžeme vyjádřit ve vztahu k proudu kolektoru, obdržíme úpravou tento vztah:

R1 = U(IN) / (I(C) / BETA)

Neznáme hodnoty odporů R1 a R2 jsme tedy vyjádřili vztahy se známými hodnotami. Teď nastává čas vzít všechny známé údaje, ty dosadit do rovnic a rovnice vyřešit. A to nás vede k využití malé výpočetní techniky. Zde nám může hodně prospět náš počítač Atari, který nám tak podstatně zkrátí čas potřebný k vytvoření spínacího obvodu.

Pomocí programu TYPO II natypujte program do počítače. Program je napsán v jazyku Atari BASIC. Před spuštěním programu doporučuji tento nahrát na kazetu nebo na disketu (D:TSCD.BAS - Tranzistor Switching Circuit Design).

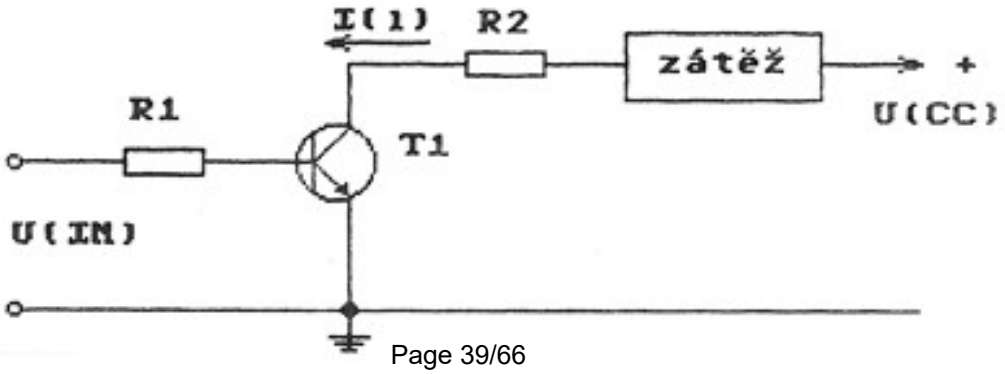
Samotný program vám poskytne několik zajímavých možností, které snad časem oceníte. Například může být využit k vytvoření spínacího obvodu s jakýmkoliv řídícím napětím. V případě, že rádi využíváte TTL, program automaticky nastavuje hodnotu vstupního napětí U(IN) na 2.4 V. Podobně při ovládání LED diody program si sám nastaví tyto hodnoty: proud zátěže 15 mA a úbytek napětí na LED diodě 1.5 V.

Původně byl program napsán v jazyku MBASIC-80, který jsem "přeložil" do jazyku Atari BASIC. Program je velmi jednoduchý, a proto ho můžete využít na jakémkoliv typu počítače s dialektem jazyka BASIC. Hlavní rutiny jsem přepsal jako podprogramy pro jednodušší integraci těchto rutin do vašich programů. Řádky 1000-1210 operují s vloženými údaji a provádějí výpočet hodnot odporů R1 a R2; řádky 2000-2320 vybírají konečnou normalizovanou řadu odporů E24. Řádky 110-190 řídí hlavní rutinu, která následuje po řádku 1000. Řádek 130 nastaví návěští, které využívá rutina výpočtu standartních hodnot. Z toho vyplývá, že data umístěná na řádcích 2050-2110 se nemusí v průběhu dalších výpočtů znovu načítat. Řádek 140 "čistí" obrazovku; závisí na typu jazyka a počítače.

A to je vše, co jsem chtěl říci o otázce výpočtu hodnot odporů v jednoduchém spínacím obvodu. Dříve než se pustíte do práce, raději se činnost programu zkontrolujte. Zde vás upozorním, že program také vypíše i hodnotu I(B) ve vašem obvodu, což slouží ke kontrole výpočtu. Jestliže použijete jako řídící obvod CMOS a I(B) bude větší než 1 mA, máte na vybranou. Buď použijete Darlingtonovo zapojení nebo přidáte další tranzistor pro "předzesílení". Jesliže si vyberete první možnost, tj. Darlingtonovo zapojení, můžete program používat bez jakýchkoliv změn.

ComputerDigest, 8/1986
Jiří Hrdlička
GIA Software

Obr. 1



Listing 1 - program STCD.BAS Part: 1

```
FT 10 REM ┌──────────────────────┐
GV 20 REM │ Switching transistor │
WW 30 REM │    circuit design    │
GG 40 REM └──────────────────────┘
BZ 50 REM - for all Atari XL/XE
FT 60 REM - Atari klub Olomouc
BS 70 REM - (c) 1990 by GIA Software
MU 80 REM Pozor!   Program neni osetren
TN 90 REM proti spatnym vstupnim udajum.
FY 100 DIM YESNO$(1)
KX 110 FIRST=0
JC 120 GRAPHICS 0:SETCOLOR 2,0,0
JL 130 POKE 752,1
MX 140 LIST 10,90
AV 150 FOR TIME=1 TO 750:NEXT TIME
BE 160 ? CHR$(125)
JF 170 POKE 752,0
QY 180 GOSUB 1000
SV 190 ? :? "Dalsi vypocet (A/N) ?":INPUT #16;YESNO$
VI 200 IF YESNO$="A" OR YESNO$="a" THEN GOTO 160
KJ 210 GRAPHICS 0:END
WS 1000 REM Udaje pro spinaci obvod
FY 1010 ? CHR$(125)
TA 1020 ? :? "Zadej napeti zdroje U(CC) [V]: ";:INPUT #16;UCC
IV 1030 ? :? "Zadej hodnotu BETA tranzistoru: ";:INPUT #16;BETA
KF 1040 ? :? "Je zatez LED dioda (A/N)? ";:INPUT #16;YESNO$
DI 1050 IF YESNO$="A" OR YESNO$="a" THEN UL=1.5:IL=15:GOTO 1100
AP 1060 ? :? "Zadej odpor zateze R(L) [Ohm]: ";:INPUT #16;RL
NE 1070 ? :? "Zadej proud zateze I(L) [mA]: ";:INPUT #16;IL
XN 1080 UL=IL*1.0E-03*RL
UT 1100 IL=IL*1.0E-03
YH 1110 ? :? "Ovlada TTL tento obvod (A/N)? ";:INPUT #16;YESNO$
BE 1120 IF YESNO$="A" OR YESNO$="a" THEN UIN=2.4:GOTO 1150
EE 1130 ? :? "Zadej ridici napeti U(IN) [V]: ";:INPUT #16;UIN
OU 1150 IB=IL/BETA
AO 1160 R1=UIN/IB
XU 1170 RIN=R1:GOSUB 10000:R1=ROUT
OU 1180 R2=(UCC-UL)/IL
ZL 1190 RIN=R2:GOSUB 10000:R2=ROUT
LD 1200 ? :? :? "  Vypoctene hodnoty:":? "  ------------------"
        :?
JG 1210 ? "  R1 = ";R1;" Ohm"
KO 1220 ? "  R2 = ";R2;" Ohm"
OF 1230 ? "I(B) = ";IB*1000;" mA"
AP 1240 RETURN
PJ 10000 REM Normalizovana rada E24
MV 10010 REM vstup = rin
JJ 10020 REM vystup = rout
HG 10030 REM sf = meritko
RX 10040 REM Nasleduje 25 hodnot:
KP 10050 DATA 1.0,1.1,1.2,1.3,1.5,1.6,1.8
```

Listing 1 - program STCD.BAS Part: 2

```
UV 10060 DATA 2.0,2.2,2.4,2.7
CD 10070 DATA 3.0,3.3,3.6,3.9
EH 10080 DATA 4.3,4.7
DT 10090 DATA 5.1,5.6
FM 10100 DATA 6.2,6.8
VO 10110 DATA 7.5,8.2,9.1,10.0
MZ 10120 IF FIRST=1 THEN 10210
RN 10130 FIRST=1
AU 10140 AMAX=25
QD 10150 DIM A(AMAX)
IM 10160 RESTORE 10050
PH 10170 FOR I=1 TO AMAX
AP 10180 READ X
RX 10190 A(I)=X
FP 10200 NEXT I
LL 10210 X=RIN
PD 10220 SF=0
UK 10230 IF X<=A(AMAX) THEN 10270
NW 10240 X=X/10
ZA 10250 SF=SF+1
AO 10260 IF X>A(AMAX) THEN 10230
NV 10270 I=1
CA 10280 IF X<=A(I) THEN 10310
HS 10290 I=I+1
ZL 10300 IF X>A(I) THEN 10280
JE 10310 IF X-A(I-1)<=A(I)-X THEN X=A(I-1):GOTO 10330
JY 10320 X=A(I)
ED 10330 ROUT=X*10^SF
DW 10340 RETURN
```

PÁNOVÉ PROMINOU, ALE NA TÉMA "COMPUTERIZACE" NAŠÍ SPOLEČNOSTI PROMLUVÍ MŮJ SYN...

NÁSLEDKY VÝPADKU ELEKTRICKÉ SÍTĚ JSOU VŽDY NEPŘÍJEMNÉ!

ATARI KLUB OLOMOUC VÁM NABÍZÍ

Atari klub Olomouc pronajímá náhradní zdroj (viz AZ Olomouc 1-2/1989), který je použitelný pro všechny typy 8-bitových počítačů Atari XL/XE. Počítač si uchová při náhlém výpadku elektrického proudu program nebo data až 3 hodiny. Zdroj byl již několikrát s úspěchem vyzkoušen při pořádání mnoha soutěží a závodů. Informace na telefonním čísle (068) 32666.

Provádím opravy zdrojů (i zalitých) pro všechny 8-bitové počítače Atari XL/XE. Obraťte se, prosím, na tuto adresu:

Petr Ondra
Koněvova 38
783 51 Olomouc-Droždín

TURBO 2000
HARDWARE & SOFTWARE

RUTINA T-BACIL

Potřebný hardware a software:
- ATARI XL/XE, magnetofon s úpravou TURBO 2000
- zavaděč TURBO 2000, BASIC PREVADIČ

Všichni uživatelé mikropočítačů ATARI XL,XE zajisté znají ty programy, které se na kazetě v systému TURBO 2000 skládají ze dvou bloků (např. "STRIP POKER SUZI"), nebo i více bloků (např. "GOONIES"). U těchto programů je zajímavý ten fakt, že pro zavedení druhého bloku (popř. dalších bloků) do paměti počítače není nutné znovu nahrávat TURBO zavaděč. Je to provedeno tak, že předešlý blok "znovunastartuje" zavaděč, který je už uložen v paměti...

Protože jsem potřeboval tohoto efektu dosáhnout také v některých svých basicových programech, vytvořil jsem si krátkou rutinu ve strojovém jazyce, kterou jsem si nazval T-BACIL. Provedením této rutiny jako podprogramu je možno tedy v basicovském programu kdykoliv znovunastartovat zavaděč a zavést tak do paměti libovolný další blok v TURBU 2000.

Jedinou podmínkou, která zde musí být splněna, je, že při znovunastartování musí být zavaděč v paměti a musí být nepoškozen! Z toho plyne, že basic program, ve kterém voláme rutinu T-BACIL, musí být na kazetě ve formě normálního TURBO bloku, který se zavádí normálním TURBO zavaděčem (TURBO 2000, Universal TURBO, atd.). Do této formy náš program dostaneme pomocí programu "BASIC PREVADIČ" (viz referenční kazety AK Olomouc). Protože TURBO zavaděč je většinou uložen v paměti v šesté stránce, neměli bychom ve svém programu šestou stránku používat, abychom zavaděč nepoškodili (bohužel většina strojových podprogramů pro basic je umísťována právě do této šesté stránky, takže se bez nich budeme muset obejít...).

Rutina T-BACIL je velmi krátká - je to vlastně jediný basic řádek:

```
30000 GRAPHICS 0:TBACIL=USR(ADR(".................."))
```

Tečky mezi uvozovkami budou ve skutečnosti různými grafickými znaky (nelze zapsat použitým textovým editorem...), které jsou vlastně jen jedním ze způsobů uložení strojové rutiny v basic programu. Pro rutinu je použit řádek č. 30000, aby se dala pomocí příkazu ENTER přihrát k libovolnému basic programu. Potom použitím příkazu GOTO 30000 možno kdykoliv nastartovat TURBO zavaděč.

Nyní si uvedeme praktický návod na jedno z možných použití rutiny T-BACIL.

Titulek před libovolný program v TURBU 2000

Protože grafické znaky se velice nepříjemně opisují, použijeme raději generátor pro získání rutiny T-BACIL. Zapněte počítač a opište pečlivě následující program. Dříve než jej spustíte, nahrajte si jej na kazetu (např. příkazem CSAVE)!!!

```
5 REM *** GENERATOR RUTINY T-BACIL ***
6 REM nahraje na kazetu rutinu, kterou
7 REM  je mozno pak zavest ENTER"C:"
8 REM ---------------------------------
10 DATA 51,48,48,48,48,32,71,82,65
20 DATA 80,72,73,67,83,32,48,58,84
30 DATA 66,65,67,73,76,61,85,83,82
40 DATA 40,65,68,82,40,34,104,169
50 DATA 255,141,1,211,169,32,141
60 DATA 252,2,169,2,133,9,108,12
70 DATA 0,34,41,41,155,-1
80 GRAPHICS 2:SETCOLOR 2,0,0:POKE 752,1
90 RESTORE 10:POSITION 5,3
100 ? #6;"GENERATOR          RUTINY ";
110 ? #6;"TBACIL      ***************"
120 OPEN #1,8,0,"C:"
130 READ A:IF A>-1 THEN PUT #1,A:GOTO 130
140 CLOSE #1
150 ? "   <START>  - ZNOVU VYGENEROVAT"
160 ? "   <OPTION> - KONEC..."
170 IF PEEK(53279)=6 THEN 80
180 IF PEEK(53279)<>3 THEN 170
190 GRAPHICS 0:NEW
```

Máte-li tento program nahrátý na kazetě, spusťte jej příkazem RUN. Počítač dvojím zavrčením oznámí, že chce nahrávat na kazetu. Najděte volné místo na kazetě, stiskněte PLAY+RECORD, stiskněte libovolnou klávesu (tj. potvrzení připravenosti pro záznam) a počítač nahraje na kazetu rutinu T-BACIL (možno zavést do paměti příkazem ENTER"C:").

Příkazem NEW vymažte paměť a opište následující program:

```
5 REM *** TITULEK K INT.KARATE ***
10 POKE 54018,52:GRAPHICS 18
20 POSITION 4,4:? #6;"INTERNATIONAL"
30 POSITION 7,6:? #6;"KARATE"
40 FOR W=0 TO 2000:NEXT W
50 IF PEEK(53279)=7 THEN 30000
60 POKE 54018,60:GOTO 50
```

Nyní k tomuto programu pomocí příkazu ENTER"C:" přihrejte z kazety rutinu T-BACIL (vygenerovanou předtím GENERATOREM) a dostanete následující program:

```
5 REM *** TITULEK K INT.KARATE ***
10 POKE 54018,52:GRAPHICS 18
20 POSITION 4,4:? #6;"INTERNATIONAL"
30 POSITION 7,6:? #6;"KARATE"
40 FOR W=0 TO 2000:NEXT W
50 IF PEEK(53279)=7 THEN 30000
60 POKE 54018,60:GOTO 50
30000 GRAPHICS 0:TBACIL=USR(ADR(".................."))
```

To je vlastně hotový titulek před hru INTERNATIONAL KARATE. Teď je ještě nutné dostat tento program do normálního TURBO bloku. Nahrajte tento program na kazetu příkazem CSAVE. Zaveďte do paměti program "BASIC PREVADIČ" (přes START+OPTION). Ten vypíše zprávu "STISKNI PLAY+RETURN". Nastavte kazetu na začátek programu "TITULEK K INT.KARATE" (musí být nahrán pomocí CSAVE!) a stiskněte RETURN. Po načtení programu požaduje "BASIC PREVADIČ" jméno. Zvolte si tedy jméno (max. 10 znaků) - např. ">>I.KARATE" a stiskněte RETURN. Počítač dvakrát zavrčí, nachystejte kazetu, stiskněte PLAY+RECORD a po stisku libovolné klávesy se na kazetu nahraje TURBO blok titulku. Za něho skopírujte hru "INT.KARATE" v TURBU 2000 a již můžete vyzkoušet výsledek:

Zaveďte do paměti obvyklým způsobem TURBO zavaděč - nejlépe "Universal TURBO" (ať už z kazety nebo z kartridže) a spusťte zavádění od bloku s názvem ">>I.KARATE". Po zavedení tohoto bloku se na chvíli zobrazí úplný název hry (a o to nám vlastně v tomto případě šlo...) a poté se znovunastartuje zavaděč a zavede se druhý blok ("INT.KARATE" - vlastní hra) a hra se normálně spustí. Zobrazení názvu po prvním bloku je možné prodloužit podržením tlačítka START,SELECT nebo OPTION. Po jejich uvolnění zavádění pokračuje...

Ještě o funkci rutiny T-BACIL:

V rutině je použit následující strojový program:

```
104,            PLA
;obvyklé u podprogramů pro basic
169,255,        LDA #255
141,1,211,      STA 54017
;vypnutí basic-ROM
169,32,         LDA #32
141,252,2,      STA 764
;simuluje stisk libovolné klávesy nutný pro spuštění zavaděče
169,2,          LDA #2
133,9,          STA 9
;nastaví  příznak  úspěšného vykonání kazetového BOOTu (jakoby
byl zavaděč právě načten z kazety...)
108,12,0        JMP(12)
;nepřímý skok "přes DOSINI" na start adresu zavaděče
```

Pozn.: na adresách "DOSINI" registru - tj. 12,13 bude start adresa zavaděče, dokud nebude nahrazena nějakou novou

start adresou. Basic programy převedené BASIC PŘEVADĚČEM nemění obsah DOSINI, takže lze tyto programy řetězit třeba do nekonečna a neustále znovu startovat zavaděč...

Možnosti použití rutiny T-BACIL:

Záleží jen na fantazii, jak využívat tuto jednoduchou, ale užitečnou rutinu, takže snad jen stručný nástin - titulky k programům, úvodní obrázky, předřazené návody, postupné zavádění jednotlivých fází programu (např. jednotlivých obtížností hry), ukončení programu s přechodem k okamžitému zavedení úplně jiného programu atd...

Ještě podobné rutiny SO-BACIL a S-BACIL:

Protože některé užitečné programy jsou zatím ještě stále jen ve standartním (pomalém) systému záznamu, mohou se hodit také tyto rutiny, které v podstatě fungují stejně jako T-BACIL, jen s tím rozdílem, že SO-BACIL zavede strojový program normálně zaváděný přes "START+OPTION" a S-BACIL zavede strojový program normálně zaváděný přes "START".
Generátory jsou podobné generátoru rutiny T-BACIL:

```
5 REM *** GENERATOR RUTINY SO-BACIL ***
6 REM nahraje na kazetu rutinu, kterou
7 REM  je mozno pak zavest ENTER"C:"
8 REM ---------------------------------
10 DATA 51,48,48,48,48,32,83,79,65,65
20 DATA 67,73,76,61,85,83,82,40,65,68
30 DATA 82,40,34,104,169,0,133,8,133,9
40 DATA 169,1,141,233,3,169,255,141,1
50 DATA 211,169,32,141,252,2,76,0,196
60 DATA 34,41,41,155,-1
70 GRAPHICS 2:SETCOLOR 2,0,0:POKE 752,1
80 RESTORE 10:POSITION 5,3
90 ? #6;"GENERATOR          RUTINY ";
100 ? #6;"SOBACIL     ****************"
110 OPEN #1,8,0,"C:"
120 READ A:IF A>-1 THEN PUT #1,A:GOTO 120
130 CLOSE #1
140 ? "   <START>  - ZNOVU VYGENEROVAT"
150 ? "   <OPTION> - KONEC..."
160 IF PEEK(53279)=6 THEN 70
170 IF PEEK(53279)<>3 THEN 160
180 GRAPHICS 0:NEW
```

```
5 REM *** GENERATOR RUTINY S-BACIL ***
6 REM nahraje na kazetu rutinu, kterou
7 REM  je mozno pak zavest ENTER"C:"
8 REM ---------------------------------
10 DATA 51,48,48,48,48,32,83,66,65
20 DATA 67,73,76,61,85,83,82,40,65,68
```

```
30 DATA 82,40,34,104,169,0,133,8,133,9
40 DATA 169,1,141,233,3,169,32,141,252
50 DATA 2,76,0,196,34,41,41,155,-1
60 GRAPHICS 2:SETCOLOR 2,0,0:POKE 752,1
70 RESTORE 10:POSITION 5,3
80 ? #6;"GENERATOR          RUTINY ";
90 ? #6;"SBACIL       ***************"
100 OPEN #1,8,0,"C:"
110 READ A:IF A>-1 THEN PUT #1,A:GOTO 110
120 CLOSE #1
130 ? "    <START>  - ZNOVU VYGENEROVAT"
140 ? "    <OPTION> - KONEC..."
150 IF PEEK(53279)=6 THEN 60
160 IF PEEK(53279)<>3 THEN 150
170 GRAPHICS 0:NEW
```

V rutině SO-BACIL je použit tento strojový program:

```
104,            PLA
;obvyklé u podprogramů pro basic
169,0,          LDA #0
133,8,          STA 8
133,9,          STA 9
;nastavení  indikátoru  teplého  startu a nastavení indikátoru
BOOT do výchozího stavu (vlastně nasimuluje zapnutí počítače)
169,1,          LDA #1
141,233,3,      STA 1001
;simulace stisku klávesy START
169,255,        LDA #255
141,1,211,      STA 54017
;vypnutí basic-ROM (tato část je u rutiny S-BACIL vypuštěna)
169,32,         LDA #32
141,252,2,      STA 764
;simuluje stisk libovolné klávesy
76,0,196        JMP 50176
;skok na příslušné místo do operačního systému...
```

ing. Maceček Václav
1989 Atari klub Uničov

Psaní v GR. 2

Program byl napsán pro Atari XL/XE v jazyku Atari BASIC. Umožňuje psát přímo z klávesnice na obrazovku v grafickém módu 2, kde jsou písmenka několikrát větší než v základním grafickém módu 0. Proto je lze lépe číst i z větší vzdálenosti. Program je určen pro malé děti, které se chtějí učit poznávat písmenka, psát jednoduchá slova a vytvářet vzory na obrazovce. Základní červenou barvu písmen můžeme měnit klávesou CAPS na světle zelenou, klávesou INVERSE na tmavě zelenou. Kurzor je v podobě krátké čárky. Po zaplnění obrazovky (10 řádků x 18 sloupců) opět naskočí prázdné pole. Mazání znaků se provádí klávesou DELETE. Výpis programu je opatřen kontrolními kódy TYPO II.

B. Barnet, Brno

Listing MiSe - program WRITEGR2.BAS Part: 1

```
LB 10 REM PSANI V GR.2
PL 20 DIM A$(10)
ZE 30 ? "{"
PO 40 GRAPHICS 2+32+16
ER 50 SETCOLOR 4,6,0
VN 60 POSITION 1,1
YV 70 ? #6;"_"
GQ 80 FOR J=1 TO 10
LH 90 FOR I=1 TO 18
FW 100 OPEN #7,4,0,"K:":GET #7,KEY:CLOSE #7
CJ 110 IF KEY=126 AND I=1 AND J=1 THEN POP :GOTO 30
IW 120 IF KEY=126 AND I>1 THEN POSITION I-1,J:? #6;"_ ":I=I-1:G
       OTO 100
ZJ 130 IF KEY=126 AND I=1 THEN POSITION 18,J-1:? #6;"_":POSITIO
       N 1,J:? #6;" ":J=J-1:I=18:GOTO 100
GG 140 IF KEY=155 AND J=10 THEN GOTO 30
MT 150 IF KEY=155 THEN POSITION I,J:? #6;" ":J=J+1:I=1:POSITION
        I,J:? #6;"_":GOTO 100
DT 160 A$=CHR$(KEY)
JV 170 POSITION I,J
SW 180 ? #6;A$
LF 190 IF I=18 THEN POSITION I+1,J:? #6;" ":POSITION 1,J+1:? #6
       ;"_":GOTO 220
VJ 200 POSITION I+1,J
VC 210 ? #6;"_"
FW 220 NEXT I
GI 230 NEXT J
QF 240 GOTO 30
```

N O V I N K A

TURBO 2000 s OBRAZEM

Vážení uživatelé osmibitových počítačů značky ATARI. V tomto příspěvku Vás chci seznámit se zrychleným přenosem dat mezi počítačem a magnetofonem (datasetem), který jistě ještě neznáte.

Po zapnutí počítače je nastavena přenosová rychlost pro komunikaci s datsetem na 600 Bd. Je ro takzvaný standartní přenos s předepsanými bloky dat a kontrolními byty v každém bloku. Tato rychlost je samozřejmě po určité době nedostačující. Jisté zrychlení přenosu bylo vyřešeno pražským systémem T2000, autor Richter, a jeho verzemi. Tímto se vyřešilo pouze zrychlení nahrávání programů, zvláště her. Pokud chcete uživatelsky komunikovat mezi počítačem a datasetem, přichází v úvahu použít pouze turbo operační systém (TOS), ale ten vylučuje použití disketové jednotky. Proto jsem navrhl zrychlený systém, který nepotřebuje ani úpravu zakoupeného datasetu a který nevylučuje ani spolupráci s disketovou jednotkou. Prakticky veškeré programy, až na výjimky, pracují stejně jako při přenosové rychlosti 600 Bd, nic není zkráceno nebo jinak upraveno.

V Atari BASICu veškeré příkazy jako LIST"C:", SAVE"C:", CSAVE"C:", PUT#N, CLOAD, GET#N, SAVE"D1:xxx" atd. pracují stejně jako při zapnutí počítače. Zrychlený přenos mnou vyvinutý se provádí při zapnutém obraze, obraz nebliká, pruhy nemusí kazit obrazovou část televizoru a dodržují předepsané bloky pro přenos jako při přenosu 600 Bd.

Celý program pro systém T2000 s obrazem je na dvou programujících řádcích v BASICu (ř. 1 a 2). Po spuštění programu pomocí RUN se provede následující:

1) Překopíruje se ROM OS do RAM pod OS.
2) V takto upravené RAM se při vypnutí ROM OS provede úprava přenosové rychlosti u magnetofonu (8 Bytů).
3) Přepíše se program pro vypínání datasetu v mezerách při příkazech LIST"C:" a podobných příkazech NOP.

Tímto je T2000 s obrazem instalováno v BASICu. Můžeme přepínat do rychlosti 600 Bd příkazem POKE 54017,253 nebo RESET tlačítkem, pokud nám to prgram dovolí (není zablokováno) a do rychlosti 2000 Bd se přepíná příkazem POKE 54017,252.

Je to pouze vypnutí nebo zapnutí ROM OS při povoleném BASICu. Na stejné adrese i vypíná nebo zapíná ROM BASIC (např. pro potřeby použití ramdisku pod ROM BASICu). Uživatelé ATARI 130 XE na stejné adrese pomocí jednotlivých bitů mohou ještě připojovat zvolené 16 kB bloky přídavné paměti 64 kB.

Po aktivaci systému T2000 s obrazem se program smaže příkazem NEW. Systém přenosu T2000 s obrazem je v paměti do vypnutí počítače, nebo do doby přepsání jiným programem. V assembleru, případně v jiném jazyku, se přepíná následujícím způsobem:

```
a)  600 Bd  -  LDA #255
               STA 54017     (RTS)
```

b) 2000 Bd - LDA #254
STA 54017
(RTS)

Pár připomínek a poznatků k T2000 s obrazem:

a) Překopírování OS z ROM do RAM jsem nevymyslel já ani nikdo v ČSSR, vše bylo publikováno v zahraničním tisku již v roce 1986.

b) Úprava přenosové rychlosti v ROM je dostupná všem z Peek and Poke (2.část o ROM OS).

c) V RAM OS není žádná jiná úprava, tudíž každý uživatel si může jinou úpravu (např. v znakové sadě, nebo v rutině pro tiskárnu) upravit sám.

d) Při dalších soukromých úpravách v RAM OS je třeba dodržet kompatibilitu s ROM OS tzn., aby rutina na tiskárnu BT 100 nebo tiskárnu EPSON byla na stejném místě jako původní pro ATARI tiskárnu a další případy.

e) Tento zrychlený přenos není nikde chráněn žádným patentem nebo zlepšovacím návrhem (jak se u nás stalo zvykem) a tudíž jej může každý použít podle vlastního zvážení ve svých programech.

f) Podobným způsobem lze změnit i přenosovou rychlost pro disketovou jednotku (až do 100 kB). Disketáři si to jistě vyzkoušejí.

g) Pokud se nahraje A.C.E., není třeba používat Turbo Basic. Sám ho nepoužívám, i když v něm programovat umím. Nesetkal jsem se totiž s případem, který bych nemohl v normálním Basicu řešit stejně rychle jako v Turbo Basicu.

Na základě Vašich připomínek k T2000 s obrazem je možné v některém dalším Zpravodaji zveřejnit některé další programy, jako například:

a) Zavedení BASIC programu nebo programu ve strojovém kódu do počítače v Basicu příkazem ENTER"C:". Po prvním bloku vypíše až 38 znaků názvu programu na horní řádek obrazovky. Krátké mezery a automatické spuštění programu po nahrátí.

b) Kopírovací program v Basicu pro přenosy v 600 Bd i v 2000 Bd.

c) Podprobnější popis a možnosti T2000 s obrazem.

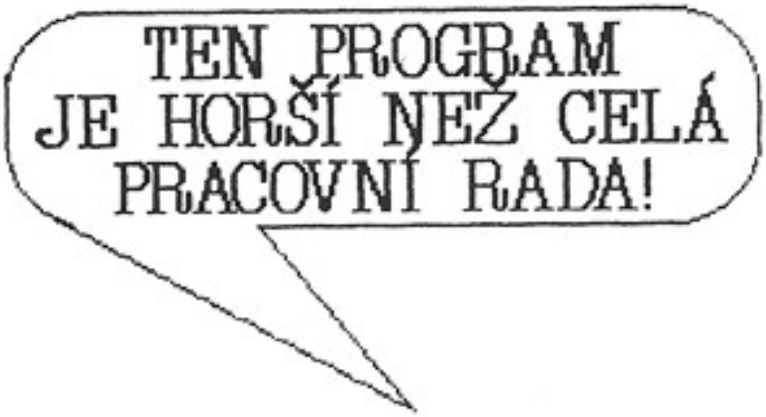
Pokud se někomu nebude líbit shodnost názvu T2000 s názvem pražského T2000, tak podotýkám, že pražské turbo má přenosovou rychlost asi 2170 Bd a je to pouze turbo zavaděč zrychlených programů a ne turbo přenos. Při nahrávání obrázků v T2000 s obrazem je přitom nahrávání v GR.2 rychlejší než při použití T.O.S. a obraz je vidět.

Veškeré dotazy a připomínky budu přijímat přes redakci olomouckého zpravodaje. Pro zavádění her i nadále doporučuji používat pražské T2000. Uživatelům tohoto nového zrychleného systému přeji mnoho úspěchů

Břetislav Korhoň

Listing 1 - program T2000.BAS Part: 1

```
YY 0 REM T 2000 (c) KORHON 1988 OLOMOUC
RI 1 A=USR(ADR("[illegible]
     [illegible](h◆"))
BJ 2 A=USR(ADR("h[illegible]
     [illegible]◆"))
WJ 3 REM PO SPUSTENI PROGRAMU POMOCI RUN        JE NASTAVENA RYC
     HLOST 2000 BAUD.       PREPINANI RYCHLOSTI 600 BD NEBO
XG 4 REM 2000 BD SE PROVEDE ZPUSOBEM :          A)600 BAUD
                                BASIC = POKE 54017,253
ZQ 5 REM . ASSEMBLER= LDA # 255                                STA
      54017 / RTS           B)2000 BAUD
QF 6 REM .  BASIC = POKE 54017,252                ASSEMBLER= LD
     A # 254                                STA 54017 / RTS
LB 7 REM ORIGINALNI DATASETY SE NEMUSI            UPRAVOVAT!PRENOS
      JE S OBRAZEM
ZL 8 ? "↰":LIST 3,7:NEW
```



Řádek 1

INV)	INV f	CTRL+N	INV f
CTRL+,	INV L	INV T	INV L
INV CTRL+E	INV p	INV -	INV f
K	CTRL+L	CTRL+A	INV N
INV CTRL+E	INV SHIFT+5	INV S	INV p
INV M	INV L	)	CTRL+L
INV)	INV I	ESC CTRL+DEL	INV SHIFT+5
INV SHIFT+8	INV P	INV CTRL+M	INV N
INV CTRL+E	INV P	CTRL+A	INV I
INV L	INV m	INV S	INV P
INV)	INV)	INV)	INV P
SHIFT+8	INV X	INV SHIFT+8	INV m
INV CTRL+E	INV CTRL+E	INV CTRL+E	INV)
INV N	INV L	INV N	INV X
INV SP	INV P	INV)	INV CTRL+E
CTRL+,	INV g	SHIFT+8	INV N
INV 1	CTRL+H	INV CTRL+E	INV P
INV K	x	INV L	INV g
INV CTRL+Q	INV -	INV 1	h
INV M	CTRL+N	INV K	INV CTRL+M
INV H	INV T	INV CTRL+Q	CTRL+N
INV P	H	INV M	INV T
INV y	INV)	INV H	(
INV f	CTRL+,	INV P	h
INV N	INV CTRL+M	INV y	CTRL+.

Řádek 2

h	CTRL+B	INV 9	INV m
INV)	INV CTRL+M	INV CTRL+M	INV CTRL+M
CTRL+A	CTRL+V	INV SHIFT+3	ESC CTRL+TAB
INV CTRL+M	INV n	INV k	INV m
CTRL+X	INV)	INV CTRL+M	INV SP
INV n	CTRL+X	INV CTRL+G	CTRL+H
INV CTRL+M	INV CTRL+M	INV SHIFT+=	INV)
INV CTRL+N	CTRL+R	INV CTRL+M	INV j
ESC CTRL+DEL	INV n	A	INV CTRL+Y
INV CTRL+M	INV)	ESC CTRL+2	CTRL+C
INV (	-	INV CTRL+M	INV l
INV k	INV CTRL+M	INV CTRL+W	INV CTRL+Y
INV CTRL+M	INV CTRL+P	INV m	INV CTRL+I
INV a	ESC CTRL+DEL	INV)	INV m
INV SHIFT+=	INV)	INV)	INV CTRL+H
INV CTRL+M	CTRL+,	INV CTRL+M	INV P
F	INV CTRL+M	INV CTRL+V	INV w
ESC CTRL+2	INV CTRL+R	INV m	INV CTRL+M
INV CTRL+M	ESC CTRL+DEL	INV CTRL+M	CTRL+L
ESC SHIFT+INS	INV)	ESC SHIFT+DEL	INV L
INV m	INV CTRL+V	INV m	INV CTRL+M
INV CTRL+M	INV CTRL+M	INV)	CTRL+M
CTRL+T	INV CTRL+T	INV j	INV l
INV n	ESC CTRL+DEL	INV CTRL+M	CTRL+.
INV)	INV)	INV CTRL+X	

Animace na ATARI XL/XE

(c) 1990 by Dora Software
Milan Taláček
AK Jihlava

Potřebujete doplnit Váš program efektní hlavičkou? Potřebujete upoutat pozornost na propagační akci nebo na přehlídce? Použijte animaci! Pokud nevíte, jak na to, může Vám napovědět tento článek.

V Turbo Basicu je nejvýhodnější použít k "animaci" grafických znaků a příkazu TEXT, který umožní jemný nebo hrubší posun objektu ve zvoleném grafickém módu. Grafické znaky je třeba změnit tak, aby představovaly požadovaný objekt. V následujícím programu je tímto objektem malý panáček, který se pohybuje od pravého okraje obrazovky do jejího středu; přitom "hýbe" nohama a horní polovina těla zůstává nezměněna.

```
4 DIM TRANS$(120),A$(2)
5 GR=8+16
6 EXEC TRANS:REM zmena CTRL znaku
7 POKE 709,14:POKE 710,130:POKE 712,130:REM barvy
8 COLOR 1
9 X=319:Y=60:REM pocatecni pozice
10 REPEAT
20   X=X-2:EXEC KROK1
30   EXEC KRESLIVRSEK
35   A$(1)=CHR$(3):A$(2)=CHR$(6)
40   TEXT X,Y+16,A$
50   X=X-2:EXEC KROK2
60   EXEC KRESLIVRSEK
65   A$(1)=CHR$(7):A$(2)=CHR$(8)
70   TEXT X,Y+16,A$
80   X=X-2:EXEC KROK3
90   EXEC KRESLIVRSEK
95   A$(1)=CHR$(9):A$(2)=CHR$(10)
100   TEXT X,Y+16,A$
110 UNTIL X<150:REM do prostred obrazovky
120 GET KEY
121 END
130 REM ============================
140 PROC KRESLIVRSEK
145   A$(1)=CHR$(1):A$(2)=CHR$(4)
150   TEXT X,Y,A$
155   A$(1)=CHR$(1):A$(2)=CHR$(5)
160   TEXT X,Y+8,A$
170 ENDPROC
180 REM ============================
190 PROC KROK1
200   SOUND 0,110,12,4:PAUSE 2:CLOSE
210 ENDPROC
220 REM ============================
```

```
230 PROC KROK2
240   SOUND 0,80,12,4:PAUSE 2:CLOSE
250 ENDPROC
260 REM ============================
270 PROC KROK3
280   SOUND 0,90,12,4:PAUSE 2:CLOSE
290 ENDPROC
300 REM ============================
310 PROC TRANS
320   ORIG=152
321   CHSET=152*256
322   GRAPHICS GR
323   POKE 559,0
324   POKE 756,152
326   RESTORE 330
328   FOR I=1 TO 33:READ A:TRAN$(I,I)=CHR$(A):NEXT I
330   DATA 104,104,133,205,104,133,204,104,133,
       207,104,133,206,104,104,170,160,0,177,204,145
332   DATA
206,136,208,249,230,205,230,207,202,208,242,96
340   IF PEEK(CHSET+101)=22 THEN 370
350   X=USR(ADR(TRANS$),57344,CHSET,4)
355   RESTORE 380
360   FOR A=0 TO 79:READ B:POKE 520+A+CHSET,B:NEXT A
370   POKE 559,34
375 ENDPROC
379 REM data pro figurku
380 DATA 0,0,0,0,0,0,0,1
390 DATA 62,68,197,137,145,231,7,14
400 DATA 44,15,20,4,54,76,32,31
410 DATA 0,0,0,0,8,20,232,16
420 DATA 206,8,14,16,224,64,64,64
430 DATA 64,64,192,128,128,128,128,128
440 DATA 44,15,20,6,5,29,34,63
450 DATA 64,64,64,64,48,136,16,224
460 DATA 44,15,20,6,4,28,32,63
470 DATA 64,64,64,64,176,136,144,224
```

Procedura TRANS změní grafické znaky CTRL+A až CTRL+J. Ve smyčce na řádcích 10 až 110 se potom vykreslují znaky představující stylizovaného panáčka. Procedura KRESLIVRSEK vykresluje znaky horní poloviny těla. Mění se pouze postavení nohou. Procedury KROK1-3 zajišťují zvukové efekty při "chůzi".

V předcházejícím příkladu se pohyboval pouze malý objekt. Kdybychom však v Turbo Basicu chtěli pohybovat s něčím větším, zjistíme, že iluze pohybu již není tak dokonalá, protože vykreslování trvá déle a výsledný obrázek se nám jeví jako rozmazaný. S tím už bohužel nic nenaděláme. Zde nám pomůže pouze strojový kód.

V assembleru je možné díky velké rychlosti vykreslovat obrázek bod po bodu, resp. byte po bytu. Následující podprogram vykreslí na obrazovku obdélník 11x46 byte, tj. 88x46 pixelů v gr.módu 8 nebo 44x46 pixelů v gr.módu 7, apod.

V Basicu stačí vyvolat grafický mód, zvolit barvu a můžeme kreslit. Zde ale potřebujeme navíc display list pro zvolený grafický mód. V podprogramu je jeho začátek stanoven na adresu $3000. Data vlastního obrázku nebo lépe obrázků jsou uložena od adresy $4000. Aby data zabrala co nejmenší místo v paměti, jsou "naskládána" za sebou, tzn. že první byte prvního řádku je na adr. $4000, pak následuje dalších deset byte prvního řádku prvního obrázku a na adrese $400B je první byte druhého řádku prvního obrázku, atd. Data jsou pravděpodobně největším úskalím animace. Vyžadují rozfázování pohybu vykreslení v některém grafickém programu a převedení "obdélníku" dat, ve kterém je Váš objekt, na řadu čísel. Dále je třeba vědět, kam se budou obrázky vykreslovat. V tomto příkladě je zvolen začátek obrazové paměti, která je určena v display listu, na adresu $5000.

```
0010            *=$2000 ;začátek vlastního podprogramu
0030 CH         =$02FC
0040 COUNT      =$2FFF
0050 DLISTL     =$0230
0060 DLISTH     =$0231
0070 NO1        =$2021
0080 NO2        =$2020
0090 NO3        =$2024
0100 NO4        =$2023
0110 ;------------------------
0120 ;nastavení počátečních hodnot
0130            LDA #$00
0140            STA NO2
0150            STA NO4
0160            STA COUNT
0170            STA DLISTL
0180            TAX
0190            TAY
0200            LDA #$30
0210            STA DLISTH
0220            LDA #$40
0230            STA NO1
0240            LDA #$50
0250            STA NOš
0260 ;------------------------
0270 HVE        LDA $4000,X ;začátek dat
0280            STA $5000,Y ;obrazová paměť
0290            CPX #$FF
0300            BEQ BSUB ;pokud by mělo dojít k přetečení
0310            INX      ;jinak další byte
0320            CPY #0A  ;počet sloupců-1 (v tomto případě
pro $0B=©11)
0330            BEQ CSUB ;vykreslen celý řádek obrázku?
0340            INY
0350            JMP HVE
0360 ;------------------------
0370 CSUB       LDY #$00 ;další řádek
0380            LDA NO4
```

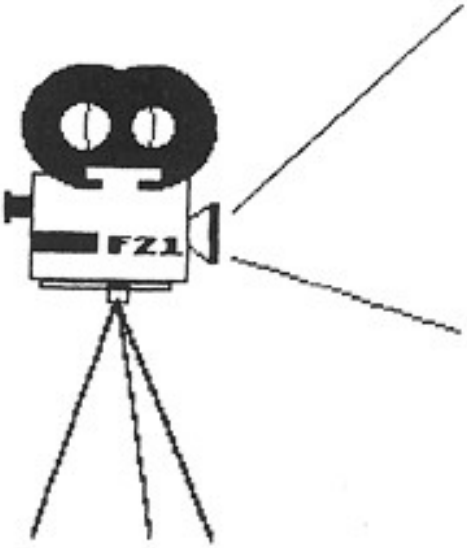
```
0390          CLC
0400          ADC #$10 ;první část ze $28=@40 - počet
byte na řádek pro celou obrazovku ve zvoleném gr. modu
0410          BCS FSUB
0420 VLN      CLC
0430          ADC #$18 ;druhá část
0440          BCS GSUB
0450 TEC      STA NO4  ;zvětší zač. obr. paměti
0460          INC COUNT ;zvětší čítač řádků
0470          LDA COUNT
0480          CMP #$2E ;pokud je obr. celý (v tomto př.
$2E=@46 řad)
0490          BEQ ESUB ;pak konec
0500          JMP HVE  ;jinak další řádek
0510 ;----------------------
0520 FSUB     INC NO3  ;při přetečení
0530          JMP VLN  ;přičti další
0540 ;----------------------
0550 GSUB     INC NO3  ;při dalším přetečení
0560          JMP TEC  ;a ulož
0570 ;----------------------
0580 BSUB     LDX #$00 ;když množství dat přesahuj
$FF=@255
0590          INC NO1
0600          JMP KOL  ;vrať se do hl.prg.
0610 ;----------------------
0620 ESUB     LDA #$FF ;konec podprogramu
0630          STA CH
0640 KEY      LDA CH   ;čekej na stisk klávesy
0650          CMP #$FF
0660          BEQ KEY
0670          LDA #$FF
0680          STA CH
0690          RTS
```

Tento podprogram obrázek vykreslí. Čeká na stisk klávesy a vrací se do hlavního programu. Čekání na stisk klávesy je vlastně nahrazení zpomalovací smyčky, protože bez ní by pohyb obrázku byl příliš rychlý pro lidské oko. Smyčka s klávesou navíc umožňuje pohled na rozfázovaný pohyb. Rychlost pohybu je možné řídit hodnotami na adresách $02D9 a $02DA (KEYDEL a KEYREP).

V hlavním programu potom musíme změnit NO1 a NO2 (určuje počátek dat obrázku). Uvedený podprogram však neumožňuje, aby se obrázek pohyboval po obrazovce, pouze pohyb uvnitř "obdélníku" dat. (Posun obrázku o 8 bitů je příliš hrubý.) Je tedy vhodnější např. ke znázornění grimas obličeje nebo jiným "statickým" obrázkům.

Použitá literatura:

Atari 800XL důležité adresy systému a jejich použití při programování v Basicu, Miloš Bayer a kol., 1988



Nej, nej, nej...

Na trhu se objevil nejnovější, nejmenší, nejlehčí a nejlacinější počítač světového standartu IBM PC, samozřejmě z konstručních dílen slavné ATARI. Nazývá se ATARI PC FOLIO ("Portfolio") a nebylo by na něm nic zvláštního, nebýt jeho rozměrů na hranicích konstrukčních možností. Jde totiž o krabičku velikosti zápisníku: délka 20 cm, šířka 10 cm a výška 3 cm. A váha i s bateriemi je pouhých 450 gramů.

Nový výrobek byl zkonstruován ve shodě s moderními trendy ve vysoce rozvinutých zemích, žádajícími přenosné počítače, které umožní práci v každých podmínkách. Tyto počítače mívají malou obrazovku z tekutých krystalů, napájení z baterií a několik uživatelských programů v paměti ROM. Některé umožní i připojení modemu a takto výměnu dat s jinými počítači nebo bankami informací.

ATARI PC FOLIO je postaven na procesoru Intel 80C88. Procesor pracuje s hodinami o frekvenci 4,91 MHz. Spolupracuje s RAM o 128 KB paměti, kterou je možno používat jako celek nebo libovolnou část používat jako Ramdisk. ROM má 256 KB a obsahuje standartní procedury BIOS dle standartu IMB PC a procedury pro obsluhu displeje, klávesnice a systém zápisu a uchování informací. Navíc je v ROM instalováno bohaté uživatelské oprogramování.

Folio má malý displej z tekutých krystalů. Pracuje podle karty MDA (monochromatická grafická karta) typického IBM PC. Obsáhne 40 znaků textu v 8 řádcích o rozlišitelnosti 240 na 64 bodů. Procedury obsluhy displeje dovolují ale i 80 znaků v 8 řádcích. Displej Folia obsahuje asi 1/6 celého formátu obrazu a abychom mohli kontrolovat celou plochu, můžeme posunovat okénko displeje po obraze. Samozřejmostí je možnost změny kontrastu pomocí programových příkazů z klávesnice, a to v 8 stupních.

Počítač má klávesnici, připomínající kalkulačku. Obsahuje všechny znaky písmen i číslic, klávesy pro pohyb kurzoru, ESC, CTRL, ALT, obsažené ve standartu PC. Nemá ale klávesy funkční a pole numerických kláves, umístěných obvykle po pravé straně kláves alfanumerických. Klávesy funkční se vyvolávají současným stisknutím speciální klávesy a klávesy číselné, odpovídající číslu příslušné funkční klávesy. Speciálmí klávesa je vtipně označena znakem ATARI. Klávesy číselné jsou nad klávesami písmennými. Některé klávesy mají i tři funkce, které se vyvolají stiskem tří kláves současně, podobně, jako to bylo u Spectra. Klávesnice je vyrobena z kvalitní umělé hmoty.

ATARI PC FOLIO nemá disketovou stanici. Zápis a úschova dat jsou možné jednak do Ramdisku, a nebo na speciální karty vnější paměti, pro které je zabudován konektor na levé boční stěně počítače. Karty jsou vybaveny pamětí 32 nebo 128 KB na principu pamětí EPROM, dovolující jeden zápis a libovolný počet čtení. Nový zápis je možný po vymazání stejně jako u pamětí EPROM.

Počítač má jeden konektor pro osazení rozšiřujícího modulu. Základní rozšiřující moduly jsou:

- modul RAM až do 640 KB
- modul interfejsů seriového a paralelního pro připojení tiskárny a modemu.

Další rozšíření je možné k přenosu dat do nebo z počítačů standartu IMB PC. Jiným modulem je karta pro zápis a čtení z již uvdených pamětí EPROM. Počítač nemá přípojku pro vnější monitor ani možnost připojení jiné klávesnice.

Folio je vybaven bohatým oprogramováním. Všechny tyto programy jsou v ROM a jsou kdykoliv přístupné. V ROM je uložen operační systém MS DOS verse 2.11, vypracovaný firmou Microsoft pro počítače třídy laptop. Mimo operační systém jsou v ROM tyto programy:

- ovládání rozšiřujících modulů
- elektronická kalkulačka
- elektronická sekretářka, hlídající důležité informace
- telefonní seznam
- adresář
- stoletý kalendář, sledující informace v reálném čase
- textový editor
- program pro výměnu dat mezi počítači
- tabulkový program.

Za zmínku stojí zvláště program tabulkový. Umožní práci se soubory, tvořenými pomocí známého Lotusu 1-2-3 ve versi 1.0 i 2.1. Program textový editor má vyhledávací funkci, funkci výměny bloků, výměny znaků, jakož i tisk při použití rozšiřujícího modulu.

Počítač je nepájený bateriemi typu R-6 nebo ze sítě pomocí adaptéru. Při vypnutí sítě se automaticky přepíná na bateriový provoz a navíc se po minutě vypne displej, když se nevykonává žádná operace na obrazovce. Stiskem libovolné klávesy se displej rozsvítí. Jeden komplet baterií dovolí 160 <!> hodin práce.

Počítač ATARI PC FOLIO je výrobkem, který jistě přivítají manažeři, obchodníci a novináři. Dovoluje pracovat kdekoliv. Folio má vzhledem k minimálním rozměrům i svoje omezení, týkající se hlavně malého displeje a malé osazené RAM. Jeho místo je v práci s texty a menšími soubory dat. Pro uživatele je možnost bohatého rozšíření jistě vítána.

Počítač ATARI PC FOLIO je zajímavý i svou cenou: zásilkové obchodní domy Quelle i Neckemann v NSR jej nabízejí ve svých katalozích za 700,- DM.

A.Rosípal
AK Bruntál

directed by Jiří Hrdlička

Vážení příznivci MiSe!

S potěšením konstatuji, že tato rubrika získala stálé místo v záplavě zajímavých článků. Koncem roku 1989 jsme dostali do redakce první ohlasy a také první programy. Zpravodaje AK Olomouc tak získávají na zajímavosti, protože drobné triky se hodí téměř všem. Těším se proto na Vaše příspěvky v novém ročníku AZ Olomouc.

Test magnetofonu

Na redakční stůl se nám dostal zajímavý program z PLR (autor Pawel Koziel), který dokáže testovat rychlost a rovnoměrnost posuvu pásku v magnetofonu (datasetu). Program využívá proceduru CIO, která zde provádí korekci rychlosti přenosu dat před načtením každého bloku dat.

Po spuštění programu (RUN) je nejprve proveden zápis dat na "vzorový" mgf. pásek (volba 1). K tomu je nutno použít kvalitní magnetofon (dataset), u kterého kolísání posuvu pásku nepřekročuje 15 procent a střední rychlost posuvu pásku je rovna jmenovité rychlosti - 4.76 cm/s. Máme-li již "vzorově" nahraný pásek, přistoupíme k vlastnímu měření (volba 2).

Během nahrávání připraveného pásku se na TV obrazovce objeví tyto údaje - změna rychlosti posuvu a její okamžitá hodnota. Po ukončení nahrávky "vzorku" se vytiskne hodnota průměrné (střední) rychlosti posuvu a její odchylka v procentech. Na obrazovce uvidíme 3 vodorovné čáry. Prostřední čára ukazuje střední rychlost posuvu pásku, zbývající dvě určují toleranční pole pro nerovnoměrnost posuvu (větší nebo rovné 0.5 procentům).

Vlastní program je napsán v jazyku Atari BASIC a opatřen kódy TYPO II. Na řádcích 10-30 je menu programu. Řádky 40-50 provádí zápis "vzorku". Hlavní blok programu je na řádcích 60-100. Zde se "vzorek" nahrává, provádí se výpočet okamžité a střední rychlosti posuvu pásku, vykreslí se čára pro střední rychlost. Řádky 110-150 zabezpečují výpočet a vykreslení tolerančního pole a "kosmetickou" úprava výsledků.

Listing MiSe - program TESTMGF.BAS Part: 1

```
HX 10 REM ┌──────┐
WN 20 REM │ TEST MGF │
JH 30 REM └──────┘
BY 40 REM - for all Atari XL/XE
BB 50 REM - NEW! version
SP 60 REM - (c) 1990
BN 70 REM - by GIA Software
HK 80 REM Please, read instructions
HB 90 REM before you run program.
PA 100 OPEN #1,4,0,"K:"
UZ 110 GRAPHICS 0:POKE 752,1:? :? :? " TEST MAGNETOFONU ":? :?
       "1. ZAPIS":? "2. TEST"
ZB 120 TRAP 110:GET #1,B
WO 130 ON B-48 GOTO 200,300
JZ 200 ? CHR$(125):SETCOLOR 2,0,0:? "ZAPIS"
MA 210 OPEN #2,8,0,"C:"
TC 220 FOR I=1 TO 50
WC 230 POSITION 8,1:? 51-I,:POKE 61,127
AL 240 PUT #2,1
WZ 250 NEXT I:RUN
UO 300 GRAPHICS 8:COLOR 1:SETCOLOR 2,0,0
SS 310 POKE 752,1:? "TEST"
KD 320 OPEN #2,4,0,"C:"
IP 330 X=6:S=0:PLOT 0,84
CQ 340 POKE 656,0:? "Okamzita rychlost posuvu        £m/s ";
AF 350 FOR I=1 TO 50:GET #2,B:POKE 61,128
XW 360 V=((PEEK(750)+256*PEEK(751))/1480)*4.76
SU 370 GOSUB 500
QA 380 DRAWTO X,84+400*(4.76-V)
KX 390 X=X+6:S=S+V:POKE 656,0:POKE 657,28:? V;" "
FU 400 NEXT I
QX 410 V=S/50:GOSUB 550
TU 420 ? CHR$(125):? "Stredni rychlost posuvu ";V;" cm/s "
YB 430 P=V:V=(V-4.76)*100/4.76:GOSUB 500
HB 440 ? "(odchylka ";V;"%) ";
TO 450 V=1.005*P:GOSUB 500:GOSUB 550:V=0.995*P:GOSUB 500:GOSUB
       550
LH 460 ? "   [SPACE] ↵ menu ";:GET #1,B:RUN
YS 500 V=V*100:V=(INT(V)+((V-INT(V))>0.5))/100
ZD 510 RETURN
UO 550 Y=84+400*(4.76-V)
LZ 560 PLOT 0,Y:DRAWTO 315,Y
ZP 570 RETURN
```

Mikromonitor 64 kB

Milan Dadok
Milan Dostál (cartridge)

Program Mikromonitor je určen pro práci s assemblerem počítače ATARI. Tento program, jako ostatně mnoho jiných, prošel několika pracovními úpravami, z nichž některé se dostaly do oběhu dříve, než byl dokončen a tím neobsahují všechny instrukce zde popisované.Popisovaná je cartridgová verze. Existuje také verze 2.2E (kazetová), která ve spojení s programátorem umožňuje programování pamětí Eprom. Obsahuje několik jiných příkazů C,V,E. Všechny verze jsou navzájem "kompatibilní" a tím souhlasí příkazy.

Vlastní P Ř Í K A Z Y

Příkaz (+adresa)	provede od zadané adresy
D 0600	disassemblace
M 0600	hexa výpis
A 0600	ATASCII výpis (i inverzní A)
@ 0600	binární výpis

Psát do paměti lze zapsáním znaku (uvidíte jej po použití některého z předchozího příkazů) před adresu, na kterou chcete ukládat.

K FE	uloží hodnotu FE na $D301 (může být 00-FF).
$0600	převod na deci (#1536)
#1536	převod na hexa ($0600)
+ 0600 1000	hexa součet ($1600)
- 1600 0600	hexa rozdíl ($1000)
P	přehled paměti (naskočí 0 stránka, kurzor na 0001, zadáním např. 06 skočí přímo na 0600, posun celá stránka: CLEAR (dozadu), INSERT (dopředu), posun po bytech editačními klávesami, hodnoty lze přímo přepsat.
Z	zapne MOTORCONTROL (podruhé vypne)
Z inv.	zapne COMMAND (podruhé vypne)
J 0600 0700 00	uloží hodnotu 00 od 0600 do 0700 (může být 00-FF)
B 0600 0700 1000	blok transfer od, do, kam
B 0600 0700 1000 2	totéž, ale odzadu
W 0600 0700 1000	blok transfer s přečíslováním 3 byte instrukcí uvnitř transform. bloku
W 0600 0700 1000 2	totéž, ale provede od zadu
W 0600 0700 1000 3	blok transfer s přečíslováním 3 byte instrukcí jak uvnitř bloku tak všech dalších v paměti, které do transform. oblasti směřují (pozor na BNE, BEQ) (Pozor na delší dobu provádění.)

(Číslice 2 nebo 3 se umísťují dvě mezery za adresy.)

F 00 0A 0B 0C — hledání v paměti (najde až 12 byte)
BRK — skok na instrukci vyvolá výpis registrů a nastavení příznaků
[— otevření tiskárny
] — uzavření tiskárny

Je nadefinováno rozhraní Centronics na joystick-porty + strobe = command (viz AZ Olomouc 1-2/1988). Na tiskárnu lze poslat vše (i "zblbnout") včetně hlaviček. Přerušení tisku - klávesa ESC. Tisk není definován jako zařízení. Závorky v inverzi - tiskárna typu Atari.

L 1000 0300 C — load (600 Bd), kam, délka, zařízení
L/RETURN — vytiskne délku nahrávky
S 1000 0300 C — save (600 Bd), odkud, délka, zařízení
X 1 03 1000 0300 04 04 C:
operace CIO: č. kanálu, příkaz, odkud, délka, $035A, $035B, zařízení: (případně název), zadá-li se odkud: ???? - převezme, XXXX - nezmění.
I — příkaz pro nahrávání v TURBU
(Zadefinováno UNITURBO+ čili rychlost do 6000 Bd.)
Vytiskne hlavičku a ptá se na adresu, kam program uložit (zadat podle volby či hlavičky), adresa ukládací, délka, startovací adresa a RETURN.
Např.: ?0700/RETURN nahraje program od zadané adrsy

R 30 0700 03 0700 0100 0700
Kde: 30 rychlost (2000)
0700 odkud
03 ident. byte
0700 ukládací adresa
0100 délka
0700 startovací adresa
N — název (10 byte) možnost změny
N název12/RETURN vypíše nový název
Změna hlavičky nejjednodušeji:
R/RETURN - BREAK, vypíše OK, najet kurzorem, provést změny, potvrdit RET. skok do Atari BASICu, jinak možné přes RESET (podle nastavených vektorů)

Další možnosti cartridgového Mikro monitoru poskytují volby přes tyto tlačítka:

OPTION — maže paměť po 9FFF
SELECT — zachovává paměť až po adresu 03FF, kterou uloží pod systém na adr. D800 (možnost podívat se na HATABS a kanály před RESETEM)
Snadno se podíváte přes: K FE a pak M DB00
START — obrazovka zůstane celá černá. Čeká totiž na byte, kam DL uložit. (normálně ukládá na 9C20/9C40)

Start certridge M.monitoru 64 kB WS.

Drž tlačítko RESET (POUZE U 800XL) a do počítače zasuň cartridge. U počítače typu XE zasouváme ZÁSADNĚ ve vypnutém stavu. Tuto aktivuj stlačením tlačítka cartridge a teď vol možnost startu přes konzolu. Pokud chceš start přímo, pouze pusť RESET, jinak drž zvolené tlačítko konzoly a pusť RESET. Tím máš aktivován M.monitor.

Co lze s tímto programem v cartridgi provádět, to už je každému, kdo zná alespoň trochu assembler ATARI a jeho paměť, celkem jasné. Nejenže se lze "nabourat" do libovolného programu, ale lze okamžitě začít tvořit.

Poznámka:
Hovoříme-li o Mikro monitoru 64 kB, tvrdím, že právě tato cartridge je tou ..... . Program v popisované cartridgi zabírá v paměti počítače cca 100 byte (používá oblasti nevyužité paměti stacku apod.) + obraz, který jak již bylo popsáno, lze umístit "kamkoli". Připojování cartridge k systému zajišťuje MHB 4013. Zkušenosti s cartridgí jsou velmi dobré, umí toho tolik, že ostatní jsem přestal používat. S touto cartridgí se majiteli dostává do rukou velmi "nebezpečná zbraň". Při troše šikovnosti nám neunikne žádný sebelépe zabezbečený program proti "vniknutí" a zkopírování.

Doplnění o návod na obsluhu 16 kB CARTRIDGE

Tato verze obsahuje jak Mikro monitor (v první části EPROM), tak i sadu zaváděčů, kopírku, DOS a TOS 4.1 (v druhé části EPROM).

Obsluha i start Mikro monitoru je stejná jako u 8 kB verze. Aktivace zaváděčů se provádí z Mikro monitoru pomocí příkazu 2 x ESC a RETURN. Obsluha zaváděčů je velmi jednoduchá a nepotřebuje komentář. V TOSu a v DOSu je nadefinován tisk přes paralelní rozhraní joystick porty.

Popis programovacího přípravku:

Programování se provádí za pomocí známého programu Mikro monitor 2.2E doplněného o program ovládající programovací přípravek.

Mikro monitor 2.2E je rozšířen o tyto příkazy:

V r načtení obsahu paměti EPROM zasunuté do přípravku, přičemž r má význam:
  r=2 2 kB EPROM
  r=4 4 kB EPROM
  r=8 8 kB EPROM
Q r totéž co V r, ale je doplněné o komparaci s $FF (test vymazání paměti).

Za pomocí obou výše uvedených příkazů se rovněž obsah EPROM přepíše od adresy $6000 výše, a to v rozsahu snímané EPROM.

Příklad: Chceme přepsat obsah celé paměti 2716 (2kB):
V 2 a RETURN
- obsah EPROM se přepíše od adresy $6000 po $67FF

C xxxx yyyy zzzz - tímto příkazem se provádí komparace dvou paměťových oblastí.
xxxx - začátek první oblasti, kterou chceme komparovat
yyyy - začátek druhé oblasti, kterou chceme komparovat
zzzz - délka prováděné komparace

Příklad: Chceme porovnat obsahy dvou pamětí 2732
Pomocí V 2 načteme obsah první paměti do oblasti $6000-$6FFF, pomocí příkazu B 6000 6FFF 4000 přesuneme obsah do oblasti $4000 - $4FFF, vyměníme paměti v přípravku a opakujeme příkaz V 2.
Nyní můžeme provést vlastní komparaci příkazem C 4000 6000 1000 a RETURN.
V případě nestejného obsahu se vypíše první nesouhlasný byte první oblasti, jinak OK.

E xxxx mmmm zzzz r - programování paměti EPROM
xxxx adresa zdroje dat
mmmm adresa začátku programování dat v EPROM
zzzz délka

Příklad: Chceme programovat paměť 2732. Data máme připravena od adresy $6000 - $7000.
Píšeme: E 6000 0000 1000 4 a RETURN
- probíhá programování.

MIKROMONITOR

```
00 :14 12 85 55 68 69 00 85 56 68 85 0E D0 0E 85 70 :
10 :68 85 66 68 85 61 68 85 62 68 85 63 68 85 64 68 :
20 :85 65 6C 55 00 A5 0C C5 0E F0 D2 A8 F0 6A A0 00 :
30 :24 70 10 03 20 1B BC A5 61 30 06 F0 8F A9 F0 30 :
40 :0E 38 E9 90 30 09 D0 49 98 10 46 A5 62 30 14 AA :
50 :06 66 08 90 03 20 4D B9 A5 62 28 08 6A 66 63 E8 :
60 :D0 F8 28 85 64 A5 63 85 65 A9 80 85 0E 60 A5 0E :
70 :30 0B 24 66 30 1B A0 80 20 1D 10 30 04 A5 64 30 :
80 :10 4C 04 B8 A5 0E 30 03 20 1B 10 A6 65 A5 64 F0 :
90 :42 4C 48 B2 A5 0E 10 37 A0 00 84 61 84 62 84 63 :
A0 :84 66 A5 64 10 05 85 66 20 72 0C A2 90 A5 64 D0 :
B0 :08 A2 88 A5 65 F0 18 84 65 30 06 CA 06 65 2A 10 :
C0 :FA 85 62 A5 65 85 63 86 61 84 64 84 65 84 70 A9 :
D0 :00 85 0E 60 20 B0 0B F0 B8 A0 00 B1 62 4C 28 11 :
E0 :20 81 10 A5 3C 18 69 14 0A 20 D5 AF 4C BE 0C A5 :
F0 :0E 30 04 46 66 10 07 A5 64 10 03 20 72 0C 4C BE :
```

Obsah AZ Olomouc 1-2/1990

Redakční oznámení:
!!! PROGRAMATORI - POZOR !!!

Hledáme autory článků - SOFTWARE i HARDWARE 8-bitových počítačů Atari XL/XE. Neváhejte a pošlete článek do naší redakce. Přivítáme i překlady ze zahraniční literatury. Vždy však uvádějte, na základě jaké publikace nebo časopisu jste článek zpracovali. Honorář máme možnost vyplácet v rámci směrnice pro hospodaření ZO Svazarmu.

Již druhým rokem zpracováváme veškeré informace, které se objeví v našem zpravodaji, na počítačích Atari. Po mnoha zkušenostech s Vašimi příspěvky Vás prosíme o následující:

Příspěvky zasílejte pokud možno napsané v programu Čapek nebo Čížek (uveďte verzi) a uložené na magnetofonové kazetě (nejlépe C-60) nebo disku. Raději svůj příspěvek uložte 2x nebo 3x za sebou, totéž platí i v případě programu. Programy netypujte do textových editorů. Příspěvky nahrávejte ve STANDARTU (!), tj. 600 baudů pro dataset nebo DOS 2.5 pro disk. Značně tím urychlíte průběh zpracování jednotlivých čísel Zpravodaje a pomůžete zlepšit namáhavou práci redakční kolektivu Atari Zpravodaje Olomouc.

Zásilky posílejte na adresu olomouckého Atari klubu a označte je nápisem ZPRAVODAJ.

Naše adresa: Atari klub Olomouc
PS 137
772 13 OLOMOUC

Zpravodaj AK Olomouc 1-2/1990
NEPRODEJNÉ, odběr vázán na příspěvek.

Odpovědný redaktor: ing. Dobromil Pavlík
Odborný redaktor: Jiří Hrdlička

Neprošlo jazykovou úpravou.
Předáno do tisku: únor 1990

Tisk povolen OK ONV Olomouc, 0380500387
Tisk: MTZ Olomouc