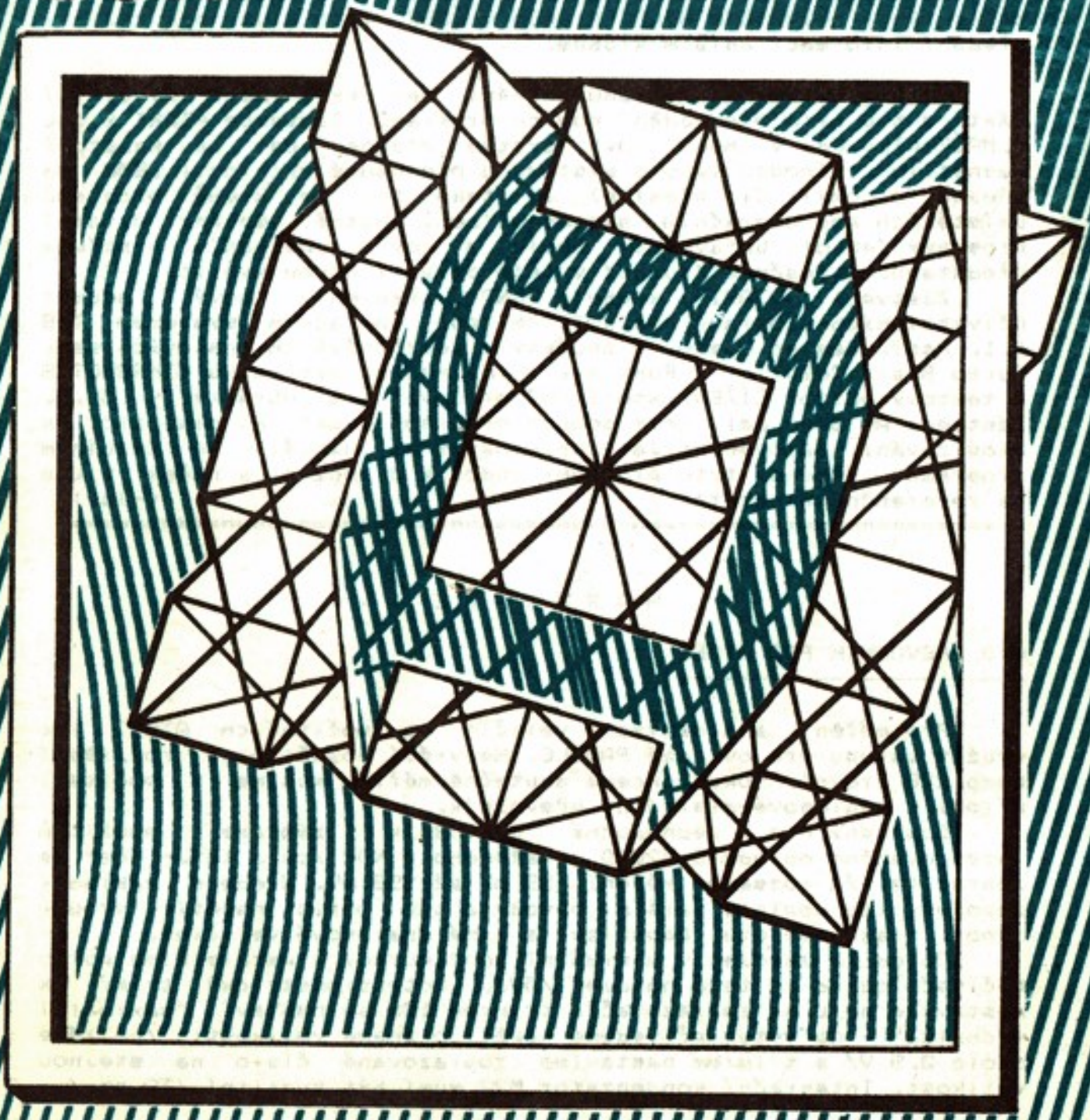
ATARI

1

89



Z P R Á V Y V Ý B O R U

Ve dnech 25. až 27. listopadu se v Liberci konalo setkání zástupců ATARI klubů z celé ČSSR. Setkání zorganizoval liberecký ATARI klub při tamější pobočce ČSVTS. Hlavním účelem setkání bylo dohodnout se na systému předávání informací a literatury mezi jednotlivými kluby tak, aby co nejširší počet členů všech klubů měl přístup ke všem dostupným informacím, aby bylo známo jaká literatura, či jaký hardware je připravován a kde.

Dohodnutý systém předávání informací bude zastřešovat ATARI klub PRAHA (487. ZO Svazarmu). Na jeho adresu budou zasílány všechny informace ze všech ostatních ATARI klubů, tj. i z těch nejmenších skupin. ATARI klub PRAHA tyto informace rozmnoží a zašle 14-ti největším ATARI klubům v ČSSR. Tyto kluby pak zajistí předání informací dalším klubům.

Dalším bodem na programu setkání pak byla méně oficiální část, a tou bylo předvádění nových programů. Zástupce ATARI klubu TLMAČE předvedl v chodu nový systém přenosu dat na kazetový magnetofon. Výhodou tohoto systému o přenosové rychlosti 9600 Bd (nejde o omyl či překlep) je také to, že kromě programů umístěných pod operačním systémem není nutné všechny ostatní programy jakkoli upravovat. Spolu s novým operačním systémem představuje tlmačské "turbo" v této oblasti novou kvalitu.

Zástupci ATARI klubu BRNO předvedli nový soubor uživatelských programů pracujících pod operačním systémem TOS 4.1. Např. Design Master s množstvím nejrůznějších znakových sad, Turbo Basic Compiler s Runtime, velmi hezkou databanku MIKRONOTES a textový editor ČÍŽEK, který umí pracovat i s obrázky v GR.8. Zástupci AK Brno dali celý soubor programů a dat k dispozici k okopírování. Zároveň přijali objednávky na manuály ke zmíněným programům. Všechny tyto programy budou k dispozici v našem klubu na referenčních kazetách. (-js-)

==========

H A R D W A R E

A/D PŘEVODNÍK PRO ATARI

Pro měření analogových veličin na počítačích ATARI lze využít vstupu pro ovladač PADDLE. Nejvyšší dosažitelné rozlišení bude 228 úrovní. Pokud chceme skutečně měřit, musíme k počítači připojit analogově-digitální převodník.

Dostupný a jednoduchý převodník získáme použitím integrovaného obvodu C 520 D dováženého z NDR /cena 165.- Kčs/ se zobrazitelným rozsahem měření -99 mV až 999 mV. Uvedené základní zapojení lze doplnit dalšími obvody a tak měřit napětí, proud, odpor, tlak, teplotu, kapacitu, a jiné analogové veličiny.

Obvod je zapojen v minimální konfiguraci k paralelnímu portu počítače odkud je také napájen /5V/. Potenciometrický trimr 5k nastavuje nulu na zobrazovači, trimrem 10k se nastaví "maximální výchylka" - na vstup přivedeme napětí o známé velikosti /nejlépe okolo 0.9 V/ a trimrem nastavíme zobrazované číslo na stejnou velikost. Integrační kondenzator M22 musí být kvalitní /TC,ap./.

Výstup převodníku je multiplexovaný - využívá čtyřbitové datové a tříbitové adresové sběrnice. Program proto nejprve kontroluje, zda adresa /bity 4,5,6 portu A = vývody 5,3,4 IO/ obsahují správnou adresu, t.j., že je nejprve nízká logická úroven na vývodu LSD /Jednotky/. Program přečte port jednou, podruhé, obě čtení porovná a jsou-li totožná, hodnota přečtená z portu se maskuje /zachová se pouze datová část - dolní 4 bity/ a výsledek se zapíše do paměti na adresu jednotek /RAMJ/. Totéž se opakuje pro desítky /RAMD/ a stovky /RAMS/. Teprve pak další část programu obstará převod uložených dat na číslice a jejich zobrazení. Zároveň je ošetřeno přetečení údaje a zobrazení záporného znaménka.

Komentovaná část programu řeší pouze čtení správné hodnoty a její uložení do paměti. Celý program ve strojovém kódu pracuje jako voltmetr s citlivostí 1 V při zobrazení měřeného napětí v rozsahu -99mV až 999mV s přesností na poslední platnou číslici /t.j. 0.1 procenta/.Hexadecimální výpis funkčního programu pracujícího jako jednorozsahový voltmetr najdete na strane 5. Do počítače jej můžete uložit například pomocí programu DEBUG MONITOR, nebo jiného monitoru od adresy 0600H (1536). Startovací adresa je 07A0H (1952)

Nevýhodou pro některé aplikace je relativní pomalost tohoto převodníku. V pomalejším režimu asi 5, v rychlejším 80 měření za sekundu. (-ok1uxx-

LITERATURA - Šandera Josef, ing. / Připojení převodníku A/D C 520 k mikropočítači - Amatérské rádio A/8-1985 strana 300.
- Katalog polovodičových součástek TESLA.

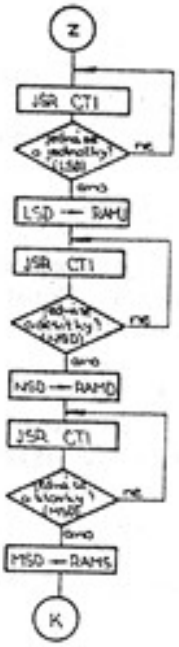
==========

KLÁVESA =BREAK= nemusí sloužit pouze k přerušování běhu programů v jazyku BASIC či jiných programů. Může nám pomoci i při práci s magnetofonem a standardním způsobem nahrávání dat na kazetu (rychlost 600 Bd). Před vlastním zaváděním dat do počítače musíme nastavit kazetu na začátek záznamu a potom odstartovat nahrávání stiskem libovolné klávesy. Pokud kazetu nenastavíme co nejpřesněji na začátek záznamu, ozvou se z reproduktoru televizoru po určité době nám všem dobře známé pazvuky, které signalizují chybu při zavádění dat. Buď byl zaváděcí tón příliš krátký, nebo na něj počítač čekal příliš dlouho.

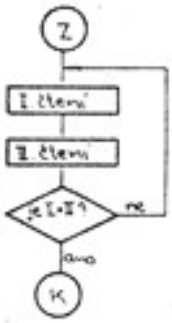
Odstartujeme-li zavádění dat ne libovolnou, ale zcela určitou klávesou, totiž klávesou BREAK, počítač již nebude trvat na přesné délce zaváděcího tónu. Můžeme si potom dovolit nastavit kazetu téměř na konec zaváděcího tónu. K chybě při nahrávání tentokrát nedojde, a my ušetříme několik sekund.

Bohužel nelze tento způsob startování magnetofonu použít přímo v režimu BASIC. V tom případě dojde k přerušení běhu nahrávací procedůry. Klávesou BREAK můžeme tedy startovat magnetofon v případě nahrávání přes tlačítka START a OPTION, nebo v těch případech, kdy stisk klávesy BREAK není vyhodnocen jako příkaz přerušení běhu programu.

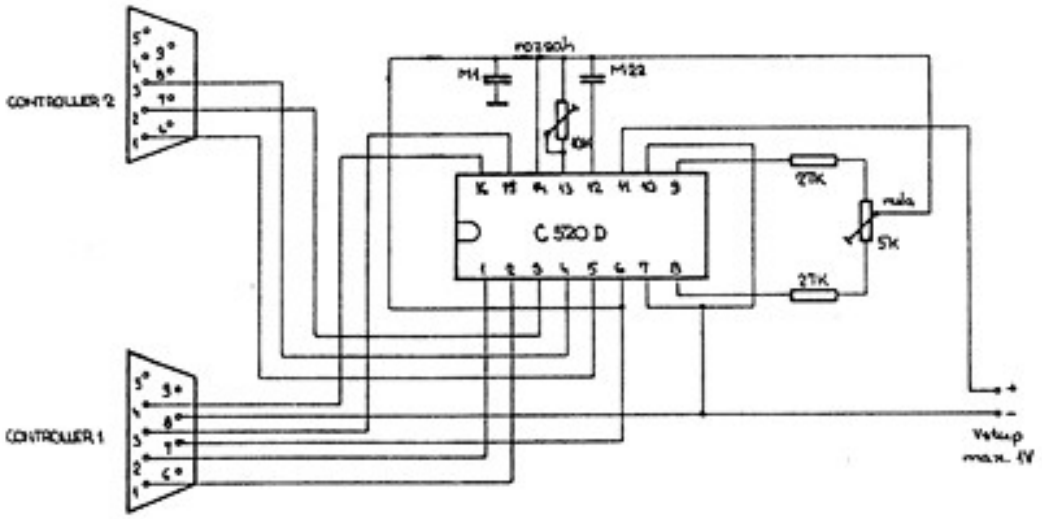
```
0000          04          .TITLE "*************************************"
0000          05          .TITLE "A/D prevodnik C 520 D /-99 az +999 mV"
0000          06          .TITLE "C 1987 ************************ OK1UXX"
0000          07          .TITLE " "
0000          08          .TITLE " "
0000          10          *=   $0710
D300          20 PORT     =    $D300      ;adresa portu
073A          30 RAMJ     =    $073A      ;pamet jednotek
073B          40 RAMD     =    $073B      ;pamet desitek
073C          50 RAMS     =    $073C      ;pamet stovek
073A          60 RAM      =    RAMJ
0710 A900     0100 C520   LDA  #$00       ;nulovani stradace
0712 EA       0110        NOP            ;*
0713 EA       0120        NOP            ;* rezerva
0714 EA       0130        NOP            ;*
0715 A000     0140        LDY  #$00      ;citac vahy cislice
0717 204007   0150 JED    JSR  CTI       ;cteni portu
071A B0FB     0160        BCS  JED       ;skok zpet,nejsou-li jednotky
071C 203007   0170        JSR  MASK1     ;zapis jednotek
071F 204007   0180 DES    JSR  CTI       ;cteni portu
0722 2A       0190        ROL  A
0723 B0FA     0200        BCS  DES       ;skok zpet,nejsou-li desitky
0725 202F07   0210        JSR  MASK2     ;zapis desitek
0728 204007   0220 STO    JSR  CTI       ;cteni portu
072B 2A       0230        ROL  A
072C 2A       0240        ROL  A
072D B0F9     0250        BCS  STO       ;skok zpet,nejsou-li stovky
072F C8       0260 MASK2  INY            ;vyssi vaha cislice
0730 8A       0270 MASK1  TXA
0731 290F     0280        AND  #$0F      ;maskovani hornich 4 bitu
0733 993A07   0290        STA  RAM,Y     ;zapis byte do pameti
0736 60       0300        RTS            ;konec podprogramu cteni
0737 EA       0310        NOP
0738 EA       0320        NOP
0739 EA       0330        NOP
073A EA       0340        NOP            ;pamet jednotek
073B EA       0350        NOP            ;pamet desitek
073C EA       0360        NOP            ;pamet stovek
073D EA       0370        NOP
073E EA       0380        NOP
073F EA       0390        NOP
0740 AE00D3   0400 CTI    LDX  PORT      ;cteni portu do X
0743 EC00D3   0410        CPX  PORT      ;porovnani poru a X
0746 D0F8     0420        BNE  CTI       ;skok zpet pri rozdilu obou cteni portu
0748 8A       0430        TXA
0749 2A       0440        ROL  A
074A 2A       0450        ROL  A
074B 60       0460        RTS
074C          0500        .TITLE "*************************************"
```



CTI:



```
$0600 $00 $06 $00 $06 $A0 $07 $18 $60 $70 $70 $70 $42 $00 $08 $02 $02 $02 $41 $08 $06 $00 $00 $00 $00
$0618 $00 $00 $00 $00 $00 $00 $00 $00 $A9 $3C $8D $02 $D3 $A9 $08 $8D $30 $02 $A9 $06 $8D $31 $02 $A9
$0630 $00 $8D $CB $02 $A9 $0F $8D $C5 $02 $A9 $04 $8D $C6 $02 $20 $50 $07 $60 $60 $60 $00 $00 $00 $00
$0648 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00
$0660 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00
$0678 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00
$0690 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00
$06A8 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00
$06C0 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00
$06D8 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00
$06F0 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $10 $11 $12 $13 $14 $15 $16 $17
$0708 $18 $19 $0D $0B $00 $00 $00 $00 $A9 $00 $EA $EA $EA $A0 $00 $20 $40 $07 $B0 $FB $20 $30 $07 $20
$0720 $40 $07 $2A $D0 $FA $20 $2F $07 $20 $40 $07 $2A $2A $B0 $F9 $CB $8A $29 $0F $99 $3A $07 $60 $EA
$0738 $EA $EA $05 $05 $05 $EA $EA $EA $AE $00 $D3 $EC $00 $D3 $D0 $FB $8A $2A $2A $60 $00 $00 $00 $00
$0750 $20 $10 $07 $20 $60 $07 $20 $70 $07 $90 $F5 $60 $00 $00 $00 $00 $A2 $03 $BC $39 $07 $B9 $00 $07
$0768 $9D $96 $08 $CA $D0 $F4 $60 $00 $18 $AD $1F $D0 $29 $07 $C9 $07 $1B $F0 $01 $38 $60 $00 $00 $04
$0780 $00 $11 $10 $13 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00
$0798 $00 $00 $00 $00 $00 $00 $00 $00 $A9 $3C $8D $02 $D3 $A9 $08 $8D $30 $02 $A9 $06 $8D $31 $02 $A9
$07B0 $00 $8D $44 $02 $8D $C8 $02 $A9 $0F $8D $C5 $02 $A9 $04 $8D $C6 $02 $A9 $A0 $85 $02 $A9 $07 $85
$07C8 $03 $A9 $02 $85 $09 $20 $50 $07 $EA $EA $A9 $0A $8D $97 $08 $8D $98 $08 $8D $99 $08 $EA $4C $DD
$07E0 $07 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00
$07F8 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $21 $0F $24 $00 $70 $72 $65 $76 $6F $64 $6E $69
$0810 $6B $00 $00 $23 $00 $15 $12 $10 $00 $24 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00
$0828 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $66 $6F $72 $00 $21 $34 $21 $32 $29 $00 $6D
$0840 $61 $64 $65 $00 $62 $79 $00 $2F $2B $11 $35 $38 $38 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00
$0858 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $0D $00 $00 $00 $00 $00 $00 $00 $00 $00 $00
$0870 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00
$0888 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $00 $35 $00 $1A $00 $0A $0A $0A $00 $6D $36 $00 $00 $00
```



Port A

7	6	5	4	3	2	1	0
L	M	S	D	C	B	A	

Z E Z Á P I S N Í K U P R O G R A M Á T O R A

Neboj se přerušení (2)

V minulé části bylo popsáno přerušení, vyvolané programem ANTIC-u. S vytvářením obrazu je spojeno ještě jedno přeřušení, mnohem zajímavější a poskytující více možností. Je to přerušení, provázející svislou synchronizaci obrazu, nazývané VBLK, někdy také VBLI nebo VBI (Vertical BLanK Interrupt).

Během vytváření obrazu na obrazovce monitoru nebo televizoru je po dosažení pravého dolního rohu přesouván elektronový paprsek do levého horního rohu. V té době je operačním systémem vykonáváno přerušení VBLK. K tomuto systémovému programu může uživatel doplnit svoji vlastní procedůru o délce až 2 kbyty !

KROK 1 (radky 10 - 20)

Napíšeme ukázkovou procedůru, která bude každou sekundu měnit barvu okraje obrazovky. Tuto procedůru zavedeme do paměti.

```
198 206       DEC 206
165 206       LDA 206
208  16       BNE END
173 200   2   LDA COLBAK$
 24           CLC
105   2       ADC #2
141 200   2   STA COLBAK$
141  26 208   STA COLBAK
169  50       LDA #50
133 206       STA 206
 76  98 228   JMP EXITVBLK
```

Tato procedura nejprve zmenší o 1 hodnotu paměťového místa s adresou 206 a vloží ji do akumulátoru. Je-li různá od nuly, následuje skok na konec procedury. Protože přerušení VBLK je systémem vykonáváno každou 1/50 sekundy, dosáhneme tak vykonaní zbytku naší procedury pouze 1x za 50 přerušení, tj. každou sekundu. Potom je vložena hodnota příslušného registru barvy do akumulátoru, zvětšena o 2 a zapsána zpět do registru barvy. Abychom zabránili nedorozumění, zapisujeme ji zároveň do ukládacího registru (COLBAK) i do stínového registru (COLBAK$). Nakonec je obnovena hodnota zpoždění (50) na adrese 206. Nyní bychom měli použít instrukci RTI - návrat z přerušení. Protože však operační systém musí vykonat ještě několik operací, provedeme na tomto místě skok na systémové návěští EXITVBLK (výstup z VBLK).

KROK 2 (řádky 30 - 40)

Procedura doplněná do VBLK vyžaduje zvláštní způsob spouštění, aby nedošlo ke zhroucení systému. Provádí se to krátkým programem, který musíme také zapsat do paměti .

```
104           PLA
160   0       LDY #0
```

```
162   6       LDX #6
169   7       LDA #7
 76  92 228   JMP SETVBLK
```

Do registru Y zapisujeme nižší byte adresy naší procedury přerušení, do registru X vyšší byte (0 + 6*256 = 1536) a do akumulátoru kód změněné procedury (7 - přerušení uživatele po přerušení VBLK). Potom provedeme skok do systémové procedury SETVBLK (nastavení VBLK). Počáteční instrukce PLA je nutná vzhledem ke způsobu práce příkazu USR v BASIC-u.

KROK 3 (řádek 50)

Zbývá jen zapsat počáteční hodnotu do paměťového místa s adresou 206, vyvolat naší proceduru nastavující přerušení a pak povolit přerušení VBLK (bit 6 v registru NMIEN - 54286).

A to je celý program v BASIC-u :

```
10 GRAPHICS 0 : FOR I=1536 TO 1560 : READ A : POKE I,A : NEXT I
20 DATA  198,206,165,206,208,16,173,200,2,24,105,2,141,200,2,141,
26,208,169,50,133,206,76,98,228
30 FOR I=1561 TO 1570 : READ A : POKE I,A : NEXT I
40 DATA 104,160,0,162,6,169,7,76,92,228
50 POKE 206,50 : I=USR(1561) : POKE 54286,6
```

Dříve popsané přerušení vyvolané svislou synchronizací (VBLK) bylo účinné pouze jedenkrát za 50 volání, tj. odměřování času bylo realizováno přímo v proceduře.
Operační systém ATARI má k dispozici pět časovačů čítajících dolů, tj. zmenšujících svůj obsah po každém přerušení VBLK. Vynulování čítačů TIMER1 a TIMER2 způsobí vyvolání procedury přerušení. Vynulování časovačů TIMER3 - TIMER5 nastaví na 0 ukazatele TIMERSIGnal. Protože TIMER1 je používán operačním systémem a zasahování do jeho činnosti vyžaduje velmi dobrou znalost systému, využijeme v naší ukázkové proceduře TIMER2.

KROK 1 (řádek 10-20)

Napíšeme ukázkovou proceduru, která bude každou sekundu měnit barvu okraje pozadí a umístíme ji v paměti.

```
173 200   2     LDA COLBAK$
 24             CLC
105   2         ADC #2
141 200   2     STA COLBAK$
141  26 208     STA COLBAK
169  50         LDA #50
141  26   2     STA TIMCOUNT2
 96             RTS
```

Procedura ukládá hodnotu z barvového registru do akumulátoru, zvětší ji o 2 a znovu zapíše do barvového registru. Potom obnoví hodnotu zpoždění (50) v registru TIMCOUNT2 (538). Nyní je místo instrukce RTI (návrat z přerušení) použita instrukce RTS (návrat z podprogramu). Je to způsobeno tím, že obsluha přerušení časovačů je vyvolávána jako podprogram z přerušení svislou

synchronizací (VBLK).

KROK 2 (řádek 30)

Nastavíme počáteční hodnotu časovače v obou registrech TIMCOUNT2 (538 a 539).

KROK 3 (řádek 40)

Změníme vektor ukazující na proceduru obsluhy přerušení tak, aby ukazoval na naši proceduru (TIMER2VKT - TIMER2VeKTor, 552 a 553) a povolíme přerušení VBLK (bit 6 v registru NMIEN - 54286).

Celý program v BASICu vypadá takto:

```
10 GRAPHICS 0 : FOR I=1536 TO 1553 : READ A : POKE I,A : NEXT I
20 DATA 173,200,2,24,105,2,141,200,2,141,26,208,169,50,
141,26,2,96
30 POKE 539,0 : POKE 538,50
40 POKE 552,0 : POKE 553,6 : POKE 54286,64
```

Posledním přerušením, které chceme popsat, je přerušení vyvolané programem ve strojovém kódu. Uvedeme je bohužel bez příkladu (byl by zbytečně dlouhý). I bez něj snad bude popis srozumitelný.
V souboru instrukcí mikroprocesoru 6502 je obsazen příkaz BRK. Je to příkaz k provedení přerušení. Užívá se nejčastěji při testování programu, napsaných ve strojovém kódu. V takovém případě vypisuje procedura přerušení na obrazovce nebo tiskárně stav všech registrů mikroprocesoru. Samozřejme tato procedura musí být rovněž napsána uživatelem.
Ještě musíme vědět, kde bude počítač hledat proceduru přerušení BRK. Adresa této procedury je zapsána v registru VBREAK (518, 519 - Vector BREAK). Na adrese 518 je uložen nižší byte adresy procedury a na adrese 519 vyšší byte. Stačí ho změnit tak, aby ukazoval na začátek naší procedury a počítač ji po každé instrukci BRK vykoná.

Samozřejmě má operační systém ATARI mnohem více ruzných přerušení. Jsou používána operačním systémem a zásahy do nich vyžadují dobrou znalost počítače a jeho systému. Z toho důvodu jsme se omezili na popis přerušení, která je možno aplikovat v uživatelských programech. (Bajtek 4-5/87, překlad H.V.

==========

TURBO BASIC ještě jednou

V prvním loňském čísle našeho zpravodaje byl uveden stručný přehled příkazu jazyka TURBO BASIC. Při přípravě tohoto přehledu jsme jako podklady použili několika příruček, jak českých, tak i cizojazyčných. Bohužel, jsou v těchto příručkách a tedy i v našem přehledu uvedeny některé udaje chybné. Za tyto nesrovnalosti se čtenářům omlouváme a v následujících řádcích uvádíme správný význam některých příkazů.

značka labelu /#/:
přiřazuje název programovému řádku, podobně jako příkaz PROC. Název tohoto řádku je možné použít jako parametr v příkazech TRAP, RESTORE a GO# (nikoli GOTO).

DEL:
tento příkaz nevymaže program. Jeho použití bez dvou parametrů má za následek chybové hlášení. Správné použití: DEL c1,c2, kde c1<c2.

DEL c.r.:
jako DEL

DIM n:
použití tohoto příkazu je stejně jako v ATARI BASICu. Dimenzuje vektory, matice a retězcové proměnné. Stejnou proměnnou lze dimenzovat v programu pouze jednou (bez předchozího použití příkazu CLR). Opakované dimenzování stejné proměnné má za následek chybové hlášení.

CIRCLE x,y,r:
kreslí jednu kružnici se středem o souřadnicích x a y, a s poloměrem r.

CIRCLE x,y,a,b:
kreslí jednu elipsu se středem o suořadnicích x a y, a s poloosami a, b. Poloosa a je rovnoběžná s osou x, poloosa b s osou y.

RND(n):
použití je stejné jako v ATARI BASICu.

RAND(n):
dává celé náhodné číslo z intervalu (0:n). Odpovídá příkazu INT(RND(0)*n).

(-js-)

==========

O tom, zda upravovat hry tak, aby měl hráč k dispozici nekonečně mnoho životů, se občas vedou živé diskuse. Faktem však je, že v některých případech je taková úprava velmi účelná, chce-li si hráč prohlédnout i závěr hry a při tom u ní nestrávit neúměrně mnoho času. Proto v našem zpravodaji návod na takové úpravy her občas uveřejníme.

Dnes popíšeme úpravu hry THE DIAMONDS od firmy ENGLISH SOFTWARE. Program je v paměti počítače umístěn od adresy $0F95 (3989) a zabírá $297F (10623) bajtu. Startovací adresa je $1443 (5187). Počáteční počet životů je umístěn na adrese $14E9 (5353). Na adrese 1560H (5472) je umístěna strojová instrukce snižující v případě nezdaru obsah adresy $14E9. Stačí tedy tuto strojovou instrukci přepsat tak, aby ke snižování počtu životů nedocházelo, což docílíme zapsáním tří instrukcí NOP (kód $EA-234) od adresy $1560. Provedeme to pomocí některého z monitorů, do kterého program nahrajeme od adr. $0F95. Po provedení úprav si hru můžeme buď odstartovat, nebo uložit na kazetu.

U Ž I V A T E L S K É P R O G R A M Y

KAREL ML - 1987

Mezi majiteli počítačů ATARI se rozšířilo několik verzí programovacího jazyka KAREL, chybí však návody k jejich správnému používání. Požádali jsme autora jednoho z překladačů, aby pro náš zpravodaj takový návod ke svému programu zpracoval.

Jazyk KAREL slouží k výuce strukturovaného programování především pro mladé programátory. Princip a koncepce překladače KAREL ML vychází z interaktivního kursu, pořádaného 602.ZO Svazarmu.

Nahrávání překladače

Překladač jazyka KAREL ML je napsán v jazyce TURBO BASIC. Je tedy nutné nejprve nahrát tento jazyk. Nahrávka KAREL ML v systému Turbo 2000 je určena k nahrávání do verze TURBO BASIC DOS 1.5 nebo 2.0 (po příkazu DOS). Po krátkém zobrazení úvodního obrázku přejde překladač KAREL ML do interpretačního režimu.

Zápis příkazu

V interpretačním režimu se všechny příkazy vepisují do spodní části obrazovky pod "město". Maximální délka příkazu (slova) je 15 znaků . Uvnitř slova nesmí být mezera. Pokud napíšeme slovo, které je již překladači známé (stačí zkratka zakončená tečkou), příkaz je vykonán. Pokud napíšeme slovo neznámé, překladač jej přenese do levé editační části a umístí pod něj slovo ZNAMENÁ. Pak očekává sekvenci příkazů, které se mají vykonávat během slova, jež jsme uvedli jako první. Zde již očekává pouze známá slova; neznamá do definice nezařadí, stejně jako slova odporující syntaxi jazyka KAREL. Např. ..KDYŽ JE POLOŽ.., slovo POLOŽ do definice nepatří a tak jej ignoruje. Na konec definice nového slova je nutno napsat příkaz KONEC. Slovo KONEC má více významů a mimo jiné i ukončuje definici. Pokud použijeme v definici cykly a podmínky, jsou v levé části graficky odděleny tak, že celý zápis podmínky nebo cyklu je posunut o jedno místo vpravo až ke slovu KONEC. To usnadňuje orientaci v hloubce ponoření do vnitřního zásobníku. Zde platí zásada, že každá podmínka, resp.cyklus má právě jeden KONEC. Všechny příkazy je nutno zadávat jednotlivě (víceslovné příkazy po jednotlivých slovech), každý příkaz zakončit stisknutím RETURN.
Správně napsané a ukončené nové slovo je zařazeno do slovníku jazyka KAREL a může být od tohoto okamžiku používáno, tj.přímo prováděno nebo zařazováno do definic nových slov. Je-li do slovníku definováno více příkazů se stejným názvem, je používána vždy nejnovější definice, neboť slovník je překladačem prohledáván odzadu.

Slovo můžeme ze slovníku vymazat příkazem CHYBA. Překladač se zeptá na rušené slovo a po jeho napsání jej ze slovníku vymaže. Pozor však na to, že zároveň s rušeným slovem jsou vymazána také všechna slova, následující ve slovníku za ním, tj. všechna slova definovaná po definici rušeného příkazu !

Spouštění příkazu

Zadáním příkazu OBSAH získáme seznam známých slov, tj. příkazů, které lze přímo provádět nebo použít v definicích nových slov. Provádění známého slova spustíme tím, že jej napíšeme (celé nebo zkratku zakončenou tečkou) a potvrdíme stisknutím RETURN. KAREL je dokonale osetřen proti chybám a tak jej máme neustále na očích. Pokud chceme jeho činnost během vykonávání příkazu zastavit, stiskneme klávesu HELP.

Režim MĚSTO

Použijeme-li příkaz MĚSTO, překladač se přepne do režimu, kdy se může KAREL pohybovat po městě pomocí klávesnice. V tomto režimu snadno umísťujeme značky a pohybujeme postavičkou. Po opuštění tohoto režimu (stisknutím RETURN) značky i KAREL zůstávají na svých místech a KAREL je znovu ovladatelný příkazy ze slovníku.

Rekurze

KAREL ML může provádět i rekurzivní volání příkazu, tj. použití daného slova uvnitř jeho definice, čímž dojde k jeho opakovanému provádění znovu od počátku. Přitom je nutno brát v úvahu hloubku zásobníku definovaného v překladači (200 zpětných adres). Po přeplnění tohoto zásobníku KAREL přeruší provádění příkazu a vrátí se do interpretačního režimu.
I během provádění rekurzivního příkazu lze činnost přerušit klávesou HELP.

Základní příkazy

Jsou to příkazy, které jsou obsaženy ve slovníku překladače KAREL ML vždy po jeho spuštění. Při opakováném spouštění překladače (např. po použití RESET nebo BREAK) příkazem RUN zůstává zachována pouze tato část slovníku, nově definovaná slova jsou vymazána. To lze obejít spuštěním překladače příkazem GOTO 40.

KROK : provede krok o jedno pole ve směru, do kterého je otočen

VLEVO-VBOK : otočí se o 90 stupňů vlevo.
POLOŽ : položí jednu značku na místo, kde stojí. Na jedno místo lze položit až 999 značek, jejich počet není zobrazován.
ZVEDNI : zvedne jednu značku z místa, kde stojí.
SEVER : otočí se na sever (nahoru).
JIH : otočí se na jih (dolů).
VYCHOD : otočí se na východ (doprava).
ZÁPAD : otočí se na západ (doleva).

Pozn. - příkazy SEVER ... ZÁPAD mohou být také součástí podmínek (viz dále).

OPAKUJ : cyklus s opakováním. Počet opakování je zadaván jako součást definice (max. 999).
KDYŽ : navestí podmínkového tělesa
KDYŽ - (podmínka) - (tělo) - KONEC

KDYŽ - (podmínka) - (tělo A) - JINAK - (tělo B) - KONEC
DOKUD : navěstí podmínkového tělesa
DOKUD - (podmínka) - (tělo) - KONEC, nebo nekonečný cyklus
DOKUD - (podmínka) - (tělo A) - JINAK - (tělo B) - KONEC
JINAK : viz popis příkazu KDYŽ, DOKUD
JE : součást podmínky, např. KDYŽ JE ZEĎ ... nebo DOKUD JE ZNAČKA ...
NENÍ : součást podmínky, např. KDYŽ NENÍ SEVER ... nebo DOKUD NENÍ ZEĎ ...
ZEĎ : součást podmínky. KAREL testuje, zda stojí před obvodovým čtvercem, který tvoří zeď kolem města.
ZNAČKA : součást podmínky. KAREL testuje, zda stojí na značce.
SEVER, JIH, VÝCHOD, ZÁPAD : součást podmínky. KAREL testuje, zda je natočen do příslušného směru.
KONEC : ukončuje definici nového slova nebo cyklus, resp. podmínkové těleso. Platí zásada, že každá podmínka má právě jeden KONEC. Tento princip je také zdůrazněn způsobem zápisu, posouváním vnořených úrovní vpravo.
MĚSTO : přepne do režimu, ve kterém je postavička KAREL přímo ovladatelná z klávesnice. Zruší se stiskem RETURN.
CHYBA : je-li použito během definice nového slova, zápis se přeruší a právě definované slovo se z paměti vymaže. Je-li použito v interpretačním režimu, překladač se zeptá na slovo, které má být zrušeno a vymaže jej ze slovníku, ovšem včetně všech slov, následujících ve slovníku za ním, tj. všech slov definovaných po definici rušeného slova!
ROZKLAD : vypíše strukturu (definici) námi definovaného příkazu (slova), jehož označení zadáme. Nelze vypsat strukturu základních příkazu. Slovo ROZKLAD nelze zařadit do definice nového slova.
OBSAH : vypíše aktuální obsah slovníku překladače KAREL ML, včetně základních příkazů. Slova vypisuje dle jejich pořadí ve slovníku.

Pozn.: Příkazy OPAKUJ, KDYŽ, DOKUD, JE, NENÍ, ZEĎ, ZNAČKA nelze použít v interpretačním režimu.

Vzory slov s podmínkou, resp. cyklem :

ZMĚNA (znamená) KDYŽ JE ZNAČKA ZVEDNI JINAK POLOŽ KONEC KONEC
KE-ZDI (znamená) DOKUD NENÍ ZEĎ KROK KONEC KONEC
10KROK (znamená) OPAKUJ (kolikrát) 10 KROK KONEC KONEC
?KROK (znamená) KDYŽ NENÍ ZEĎ KROK KONEC KONEC

Pozn.: definice jsou uvedeny v řádce pouze pro úsporu místa; zadávají se po jednotlivých slovech !

Úpravy překladače

Majitelům starších verzí překladače KAREL ML doporučujeme doplnit si do programu v jazyce TURBO BASIC tyto nové řádky:

```
2215 POCET=0
2255 IF POCET=0 THEN 2210
2285 POCET=0
6015 QQ=0
```

Dále si proveďte následující změny:

```
2225 ..... IF (VS=155 AND POCET<>0) THEN 2270
2235 ..... GOTO 2210
2250 ..... GOTO 2210
```

(Autor překladače a návodu M. Žemlička, upravil a doplnil H.V)

* * *

Použití příkazu CREATE v programu SOUND MACHINE TURBO

Pomocí následujícího programu můžeme snadno využít ve vlastních programech hudbu, napsanou v SOUND MACHINE TURBO.

Hudbu uložíme na kazetu pomocí příkazu CREATE. Potom do počítače nahrajeme TURBO O.S., který inicializujeme příkazem -C- a povelem -YES-. Hudbu pak nahrajeme pomocí tohoto programu.

Okamžitě po nahrání se automaticky spustí. Využívá se VBI, takže hudba nijak nezdržuje náš program. Dá se vypnout příkazy POKE 7966,1:FOR I=0 TO 3:SO.I,0,0,0:N.I.
Opět zapnout ji můžeme příkazem POKE 7966,0.

```
FE 10 REM = BAJTEK 8/88 =
SA 20 REM = DESIFRATOR =
RE 30 POKE 82,0:POKE 752,1:? CHR$(125)
BQ 40 ? :? :? "              NAHRAVANI"
HK 50 OPEN #1,4,0,"T:"
LU 60 FOR I=1 TO 15:GET #1,Z:NEXT I
FG 70 TRAP 100:A=7944
SD 80 A=A+1:GET #1,X:POKE A,X:GOTO 80
YJ 100 IF PEEK(195)=136 THEN 130
PQ 110 ? :? :? "   CHYBA ";PEEK(195):END
XH 130 POKE 7944,104:P=USR(7944):POKE 796
   6,0
```

POZOR! Po nahrání každého 1kB bloku počítač zpracovává data, a tak chvíli trvá, než opět spustí motor datarekorderu pro nahrání dalšího bloku ! (podle Bajtku 8/88 upravil M.H)

Další "nesmrtelnost", kterou vám nabízíme se týká hry ROBIN HOOD, též od firmy ENGLISH SOFTWARE.

Hru nahrajeme pomocí vhodného monitoru nejlépe od adresy $07E1 (2017). Na této adrese je normálně umístěna. Zabírá v paměti $347F (13439) bajtu a startuje se od adresy $2C23 (11299). Pokud hru nahrajeme na jiné místo, nepůjde odstartovat, a budeme muset dále uvedené adresy přepočítat.

Počáteční počet životů je umístěn na adrese $2DA5 (11685). Normálně máme k dispozici 3 životy. Pokud nechceme mít nekonečně mnoho životů, avšak 3 nám nestačí, změníme hodnotu na adrese $2CC1 (11457) ze 3 na požadovaný počet. Chceme-li mít životů nekonečně mnoho přepíšeme hodnotu na adrese $350E (13582) z $CE (206) na $AD (173). Tím zařídíme, že se v případě nezdaru nebude snižovat obsah paměti na adrese $2DA5. Takto upravenou hru je nejlepší znovu uložit na kazetu.

S L A B I K Á Ř A T A R I S T Y

X I O

Při čtení příruček o ATARI Basicu, dodávaných spolu s počítačem při jeho koupi, narazíme v seznamu příkazů jazyka na příkaz téměř záhadného názvu: XIO. Vše, co se o něm dozvíme z těchto příruček je napsáno ve dvou větách:"Používá se při operacích s disketou (viz příručka DOS) a při práci s grafikou. Působí jako univerzální operace vstupu/vystupu." Pokud se o tomto příkazu chcete dozvědět více, jistě přivítáte následující překlad článku Wojciecha Zientary z polského časopisu BAJTEK.

XIO je zkratkou anglického názvu "eXtended Input/Output command", což v českém překladu znamená: rozšířený příkaz vstupu/výstupu. To již něco naznačuje. Důkladnější popis příkazů a jeho použití zahájíme uvedením jeho syntaxe:

XIOČíslo,#kanal,parametr1,parametr2,"zařízení:"

Číslo za písmeny XIO je kód operace vstupu/vystupu.Kanál - je číslo kanálu IOCB. Parametr1 a parametr2 jsou pomocné číselné parametry. Zařízení - název periferního zařízení (pro disketu se ještě uvádí název souboru).
V závislosti na žádaném kódu může příkaz XIO vykonávat různé operace vstupu/výstupu. Nejdříve popíšeme základní operace vstupu/výstupu a potom operace XIO pro jednotlivá zařízení.

Základní operace vstupu/vystupu

Všechny základní operace XIO mají své ekvivalenty v jiných příkazech vstupu/výstupu ATARI Basicu:

XIO3	OPEN	otevření kanálu IOCB
XIO5	INPUT	čtení bloku bajtu
XIO7	GET	čtení jednoho bajtu
XIO9	PRINT	zápis bloku bajtu
XIO11	PUT	zápis jednoho bajtu
XIO12	CLOSE	uzavření kanálu IOCB
XIO13	STATUS	čtení stavu IOCB.

S vyjímkou XIO3 jsou ve všech ostatních příkazech ignorovány hodnoty pomocných parametrů a zadávají se rovny nule. V příkazu XIO3 (OPEN) určuje hodnota parametr1 druh přístupu do periferního zařízení. Hodnota parametr2 pro zařízení "C:" (0 nebo 128) určuje délku mezibloková mezer při nahrávce, pro zařízení "S:" reprezentuje číslo grafického módu (0 az 8). Dále jsou uvedeny možné hodnoty pro parametr1 a jejich význam v závislosti na periferním zařízení:

"C:" 4-čtení, 8-zápis,
"D:" 4-čtení, 6-čtení adresáře, 8-zápis nového souboru,
9-zápis na konec souboru, 12-současné čtení a zápis,
"E:" 8-zápis na obrazovku, 12-zápis na obrazovku a čtení
z klávesnice,

13- zápis na obrazovku a čtení z obrazovky,
"K:" 4-čtení
"R:" 5-současné čtení, 8-zápis bloku, 9-současný zápis, 13-současný zápis a čtení,
"S:" zde může být zadána hodnota i součtem několika hodnot 4-čtení z obrazovky, 8-zápis na obrazovku, 16-vytvoření textového okna, 32-obsah obrazovky zůstane nezměněn.

Protože v ATARI Basicu je většina operací XIO proveditelná pomocí jiných příkazů a značně jednodušeji, používají se XIO příkazy pro vykonání základních vstupně/výstupních operací velmi řídce. Jsou více pouhou zajímavostí ATARI Basicu než jeho přínosem.

Grafické operace

Dvě z operací XIO mohou být používány výlučně pro spolupráci s obrazovkou. Jsou to:

XIO17 - zobrazuje přímky (obdoba DRAWTO)
XIO18 - vybarvuje obrazec (jako funkce FILL v grafických programech)

První z nich, jak je výše uvedeno, odpovídá příkazu DRAWTO a proto se o ni dále zajímat nebudeme. Druhá operace svůj ekvivalent v ATARI Basicu nemá. Protože to je ze všech nejvíce pouzivaná operace XIO, uvedeme si její syntaxi zvlášť:

XIO18,#6,0,0,"S:"

Použití XIO18 není úplně jednoduché, a proto si je ukážeme na příkladu - v závěru článku je uvedený demonstrační program. Nejdříve musíme zobrazit křivky, které z pravé strany ohraničují vybarvovaný prostor. Potom musíme umístit kurzor do místa, ze kterého chceme začít vybarvovat. To můžeme provést pomocí PLOT, nebo DRAWTO. V demonstračním programu to vykonávají řádky 90 a 100. Do registru FILLDAT s dekadickou adresou 765 zapíšeme číslo použité barvy a příkazem POSITION umístíme kurzor na konec vybarovaného prostoru (řádek 110). Nyní již můžeme vyvolat operaci XIO18 (řádek 120). V řádcích 20 až 80 jsou náhodně vybíraná místa zobrazovaní a probíhá zde neustálá změna barvy. Po spuštění programu se na obrazovce vybarví obdélník. Další obdélníky získáme opakovaným tisknutím tlačítka START.

Operace XIO 18 vybarvuje každou řádku obrazovky až do té doby, než narazí na bod jiné barvy, než má pozadí. Nelze tedy vybarvovat již dříve vybarvené obrazce.

Diskové operace

Třetí skupinou operací XIO jsou operace pro spolupráci s disketovou jednotkou. Při této práci je obvyklé načíst DOS a používat jeho funkci. Operace XIO nám umožní používat takové funkce z Basicu. Ve všech operacích jsou ignorovány hodnoty parametr1 a parametr2, a je nutné, aby jejich hodnoty byly rovny nule. Nyní si uvedeme seznam diskových operací XIO:

XIO32	RENAME	přejmenování souboru,
XIO33	DELETE	zrušení souboru,
XIO35	LOCK	zabezpečení souboru,
XIO36	UNLOCK	ruší zabezpečení souboru,
XIO37	POINT	nastavení hlavičky,
XIO38	NOTE	čtení polohy hlavičky,
XIO254	FORMAT	formatování diskety,
XIO253	Format SINGLE	formatování diskety o jednoduché hustotě (jednotka 810).

Operace XIO37 a XIO38 jsou ekvivalentní příkazům POINT a NOTE v ATARI Basicu a vyžadují předchozí otevření kanálu IOCB příkazem OPEN nebo XIO3.

Zde uvedený přehled operací XIO není zdaleka úplný. Existuje další skupina vstupně/výstupních operací, které mohou být vykonávány prostřednictvím příkazu XIO. Jedná se o operace přenosu dat přes seriový interfejs RS 232. Protože je mezi majiteli počítačů ATARI velmi málo těch, kteří interfejs RS 232 využívají, záměrně jsme tuto skupinu operací XIO pominuli.

Závěrem uvádíme krátký demonstrační program na použití operace XIO18. Při přepisování do počítače nazadávejte dvojice písmen před čísly programových řádků. Jsou to kontrolní kódy vygenerované programem TYPO II. (BAJTEK 3/88, překlad -js-)

```
NM 10 GRAPHICS 31
KH 20 X1=INT(RND(0)*120)+40
OB 30 Y1=INT(RND(0)*162)+30
PI 40 X2=INT(RND(0)*140)
XE 50 IF X2>=X1 OR X2<X1-60 THEN 40
SM 60 Y2=INT(RND(0)*180)
BT 70 IF Y2>=Y1 OR Y2<Y1-60 THEN 60
DE 80 C=C+1:IF C>3 THEN C=1
FU 90 COLOR C:PLOT X1,Y1
LW 100 DRAWTO X1,Y2:DRAWTO X2,Y2
BS 110 POKE 765,C:POSITION X2,Y1
RZ 120 XIO 18,#6,0,0,"S:"
FU 130 ON (PEEK(53279)=6)+1 GOTO 130,20
```

==========

Na přípravě tohoto čísla se podíleli: P. Leden, J. Vyskoč, M. Hájek a OK1UXX.

ATARI 602, technický zpravodaj pro mikroelektroniku a výpočetní techniku. Vydává 602. ZO Svazarmu pro potřeby vlastního aktivu. Zodpovědný redaktor: J. Skála. Adresa redakce: 602. ZO Svazarmu, Wintrova 8, Praha 6, 160 41. Telefon: 341 409. Povoleno UVTEI pod ev. číslem 87006. Cena 9 Kčs dle CCU c. 1030/202/86.
Náklad 500 výtisků. Praha, prosinec 1988