ČERVEN 1993

FUN with Commodore

časopis uživatelů Commodore 64/128 6. číslo

BASIC

DYSP GRAFIKA

MIDI interface

Programy MSE

EPROMER

PLOTTERY A C64

64

Výherci slosování
I./IV.1993

OBSAH

Úvodní slovo 1

Basic 2

Assembler na C64 4

DYSP grafika 6

MIDI interface 9

Přehled Basiců pro C64 10

EXOS 12

Programy MSE 14

Představujeme:

EPROMER 18

PLOTTERY A C64 20

Comotronic News 23

Výherci slosování – I./IV.1993 23

CO BUDE V PŘÍŠTÍM ČÍSLE?

Basic – pokračování
Assembler – pokračování
Tipy a triky
Vyhodnocení čtenářské ankety
Programujeme DEMO

Představujeme:
GEOS 1.3
ČStext
Comotronic News
Dopisy čtenářů
atd.

ÚVODNÍ SLOVO

Vážený čtenáři,

jistě vás nepotěšila situace ve vydávání FUNu, kde jsme již 2 čísla za předpokládaným edičním plánem, nemluvě o tom, že dlouhé čekání, zda a kdy další číslo přijde nikomu na radosti nepřidá. Nepřidá na radosti ani vám, ani nám. Věřte, že z něj radost nemáme a hledáme cesty, jak zajistit, aby FUN vycházel pravidelně. Zadali jsme tisk časopisu i katalogů jiné tiskárně a sestavili s ní nový ediční plán, který, jak doufáme, bude dodržován.

Dlouhá pauza mezi letošním 3. a 4. číslem (nepočítáme zvláštní číslo, které je navíc), nás vedla k zařazení nové řady článků od pana Billy z Hlohovce, jehož programy pro C64 mají, jak jsme měli sami možnost se přesvědčit, špičkovou úroveň a budou pro vás určitě velice zajímavé.

Další změna, která nás naprosto netěší, nastala ve slovenském zastoupení. Spolupráce s firmou PRINGTON nepřinesla našim slovenským zákazníkům očekávané služby a proto jsme využili naší dlouholeté spolupráce s firmou ERKA Bratislava, P. O. BOX 23, 835 32 Bratislava 35, a zahájili objednávkovou a dobírkovou službu pro Slovensko přes ni. Současně se v Bratislavě začaly prodávat naše výrobky v Prodejně výpočetní techniky v Ihriskové ulici 2, 835 32 Bratislava 35.

Takže přes stále nové problémy, způsobené vznikem dvou nových států v srdci Evropy a dalšími změnami se situace postupně normalizuje v tom lepším smyslu slova. Jsem přesvědčen, že tentokrát to pocítíte i vy, čtenáři FUNu.

Redaktor 6. číslu FUNu
Jiří Kouřil
Šumperk, červen 1993

Práce s řetězcovými proměnnými

Pro práci s řetězcovými proměnnými využívá Basic V2.0 příkazy RIGHT$, MID$ a LEFT$. Související jsou příkazy VAL a LEN. Jak spolu souvisí a jak je v programech racionálně využívat si naznačíme v několika dále uvedených příkladech.

1. Určování délky řetězce bez prázdných znaků

Zjišťování délky řetězce, nebo správněji řečeno počtu znaků v řetězcové proměnné je velmi častý úkol, který je nutný při tvorbě programů. Pokud jde jen o absulutní určení počtu znaků v proměnné, není nic jednoduššího, než použít příkaz LEN. Ale v mnoha případech obsahuje řetězec na začátku nebo na konci „prázdné" znaky, odpovídající stisku mezerníku, které z různých důvodů nechceme do délky řetězce započítat, nebo bychom se jich dokonce rádi zbavili. Pak musíme kromě příkazu LEN využít i další.

a. Odstranění prázdných znaků na začátku řetězce

K řešení tohoto úkolu použijeme příkaz LEFT$ k načtení prvního znaku testovaného řetězce do pomocné proměnné P$. Nejdříve si však zjistíme délku řetězce a uložíme ji do proměné D (řádek 20). Pak si uložíme první znak zkoumaného řetězce do proměnné P$ (řádek 30). Dále zjistíme, zda je znak v P$ prázdný znak nebo jiný. Pokud je prázdný, odečteme jej od testovaného řetězce a vrátíme test k načtení nového prvního (dříve druhého) znaku testovaného řetězce do pomocné proměnné P$ (řádek 40). Tento postup opakujeme tak dlouho, až narazíme na první „neprázdný" znak. To je signálem, že na začátku řetězce již žádné prázdné znaky nejsou a je možno proces odstraňování prázdných znaků na začátku řetězce ukončit.
Praktické provedení programu pro odstraňování prázdných znaků na začátku řetězce je následující.

```
10 REM *** ODSTRANENI PZ NALEVO ***
20 D = LEN(A$)
30 P$ = LEFT$(A$,1)
40 IF P$ = " " THEN A$ = RIGHT$(A$,D-1):GOTO 20
50 REM *** POKRACOVANI PROGRAMU ***
```

b. Odstranění prázdných znaků na konci řetězce

Pokud jste si výše uvedený programový blok podrobnějiť prostudovali, je vám již jasné, jak se bude postupovat při odstraňování prázdných znaků na konci řetězce. Bude to zcela obdobné, jen s tím rozdílem, že do pomocné proměnné P$ nebudeme ukládat levý krajní znak řetězce, ale pravý krajní. Proto musíme prohodit funkce LEFT$ a RIGHT$.

```
60 REM *** ODSTRANENI PZ NAPRAVO ***
70 D = LEN(A$)
80 P$ = RIGHT$(A$,1)
90 IF P$ = " " THEN A$ = LEFT$(A$,D-1):GOTO 20
99 REM *** POKRACOVANI PROGRAMU ***
```

Pokud na takto „očesaný" řetězec nasadíme funkci LEN pro zjištění jeho délky, nemusíme mít obavy, že například při dále popisovaném centrovaném výpisu na obrazovku nám text ujede nalevo nebo napravo, jak se to jinak stává.

1. Centrovaný výpis na obrazovku.

Úroveň programátora se pozná nejen podle toho, co program umí, ale i podle toho, jak program „vypadá". Tím mám na mysli vzhled menu, výpisů na obrazovku, úpravu obrazovky, přehlednost toho, co necháme na obrazovku vypsat. Úpravnému vzhledu výpisů v mnoha případech pomůže, když je například nadpis vycentrovaný na střed obrazovky. K tomu je nutno spočítat kolik znaků text, který má být ve středu obrazovky vypsán tvoří, vydělit počet dvěma, výsledek odečíst od dvaceti a dostaneme tak počet prázdných míst, která musí být před textem, aby byl jeho výpis na obrazovku vystředěn. Tolik prázdných míst před textem pak vepíšete do příkazu PRINT"...a je to. Existuje však elegantnější postup, který, pokud jej použijeme jako podprogram, vystředí jakýkoliv texový řetězec. Postup si vysvětlíme na následujícím výpisu programu:

```
100 A$= "TEXT PRO VYPIS"
120 D=LEN(A$)
130 V=(39-T)/2
140 PRINTTAB(V)A$
```

V řádku 100 je definován znakový řetětec, který chceme na obrazovku centrovaně (vystředěně, symetricky vůči středu) vypsat.
V řádku 120 je do proměnné D uložená délka znakového řetězce ve znacích pomocí funkce LEN která slouží právě k tomuto účelu.
V řádku 130 je spočítáno, v jaké vzdálenosti od kraje se má textový řetězec vypsat, aby byl centrován a výsledek je uložen do proměnné V.
V řádku 140 je pak znakový řetězec A$ vypsán na obrazovku ve vzdálenosti V znaků od kraje s pomocí funkce TAB, která definuje, v jaké vzdálenosti od kraje se má výpis na obrazovku umístit.

Všimněte si, že funkce TAB prodlouží příkaz PRINT o 6 znaků "TAB(V)". Pokud má být výpis umístěn dále než uvedených 6 znaků od kraje, znamená tedy funkce TAB i úsporu paměťového prostoru, což vás začne zajímat při sestavování dlouhých programů, zpracovávajících velké počty proměnných.

Spojování znakových řetězců

Když píšeme program v Basicu, je maximální délka logického řádku 80 znaků. Řetězcová proměnná ale může mít délku až 255 znaků. Jak ale této maximální délky dosáhnout, když zápis v programovém řádku včetně čísla řádku a příkazu PRINT je omezen na uvedených 80 znaků?
Jde to jednoduše spojováním řetězců například následujícím postupem.

```
100 A$="PRVNI CAST TEXTU KTERY CHCEME SPOJIT DO JEDNOHO RETEZCE"
110 B$="DRUHA CAST TEXTU KTERY CHCEME SPOJIT DO JEDNOHO RETEZCE"
120 C$="TRETI CAST TEXTU KTERY CHCEME SPOJIT DO JEDNOHO RETEZCE"
130 D$="CTVRTA CAST TEXTU KTERY CHCEME SPOJIT DO JEDNOHO RETEZCE"
140 V$=A$+B$+C$+D$
150 PRINT V$
```

V prvních čtyřech řádcích 100 – 130 jsme definovali čtyři samostatné znakové řetězce o délce cca 60 znaků. Jejich spojení do jednoho znakového řetězce zajišťuje řádek 140, ve kterém je definována nová řetězcová proměnná, ve které jsou čtyři dříve definované sečteny do jedné, obsahující téměř 240 znaků.
Řádek 150 slouží k výpisu nově definovaného součtového řetězce na obrazovku. Jak zjistíte po spuštění tohoto krátkého programu, lze jednoduchým programovým řádkem jako je řádek 150 zajistit výpis téměř šestiřádkového textu!
Tento postup se dá s výhodnou použít k urychlení výpisu na obrazovku, definování mezer v textu o délce třeba 100, 150 znaků bez komplikovaného použití příkazu PRINT"", k vytváření tabulek a všelijakých efektů na obrazovce.

Dělení znakových řetězců.

Klasickými příklady, kdy je třeba rozdělit znakový řetězec je třeba oddělení jména od příjmení nebo rozdělení čísla na celou a desetinnou část.
Při tomto dělení se postupuje tak, že v celém řetězci hledáme určitý znak, v našem prvním případě mezeru, kterou je odděleno jméno od příjmení a ve druhém desetinnou tečku.

Příklad 1

```
100 A$="DVE SLOVA"
110 N=1
120 B$=MID$(A$,N,1)
130 IF B$=" " THEN 160
140 N = N + 1
150 GOTO 120
160 A1$=LEFT$(A$,N)
170 A2$=RIGHT$(A$,LEN(A$)–N)
180 PRINT A1$
190 PRINT A2$
```

V řádku 100 je definován znakový řetězec "DVE SLOVA". V řádku 110 je nastaveno počítadlo znaků v řetězci na první znak (N=1).
V následujících řádcích 120 – 150 je ve smyčce zkoumán znakový řetězec A$ na výskyt mezery " ".
V tomto procesu hraje důležitou roli řádek 120, kde je pomocí řetězcové funkce MID$ z řetězce A$ vybrán jeden znak, uložen do proměnné B$ a ta je v řádku 130 otestována, zda není rovna mezeře. Pokud není, zvýší se počítadlo znaků o 1 a program se vrátí na řádek 120, kde je do proměnné B$ uložen další znak. To se opakuje tak dlouho, až se narazí na znak mezery " ". V tomto okamžiku programový řádek 130 zajistí skok na řádek 160. Tento a následující řádek pak slouží k definici nových řetězcových proměnných A1$ a A2$, do kterých budou uložena samostatně dvě slova, obsažená v původním řetězci A$. Slouží k tomu opět řetězcové funkce LEFT$ a RIGHT$, které oddělí definovaný počet znaků zleva (N) a zprava (N–1). Zprava se odděluje N–1 znaků proto, že první znak v pravé části je mezera, kterou do řetězce nechceme. Přesvědčte se o tom sami tím, že "–1" vynecháte!
Stejnou funkci, jako příkazy LEFT$ a RIGHT$ zajistí i příkaz MID$. Dokazuje to i dále uvedená náhrada příkazů LEFT$ a RIGHT$ v řádcích 160 a 170:

```
160 A1$=MID$(A$,1,N)
170 A2$=MID$(A$,N+1)
```

Příklad 2

```
100 A=12345.788
110 A$=STR$(A)
120 N=1
130 B$=MID$(A$,N,1)
140 IF B$=". " THEN 170
150 N = N + 1
160 GOTO 120
170 A1$=LEFT$(A$,N–2)
180 A2$=RIGHT$(A$,N+1)
190 PRINT A1$
200 PRINT A2$
```

V tomto příkladě jsme navíc použili funkci STR$, která převádí číselnou proměnou na řetězcovou. Do proměnné B$ jsme tentokrát zapsali znak ".", desetinnou tečku, kterou musíme v čísle najít, abychom mohli oddělit celočíselnou a desetinnou část čísla. Jak z řádků 170 a 180 vyplývá, je uložení klasické řetězcové proměnné do proměnné odlišné od uložení čísla změněného na řetězcovou proměnnou funkcí STR$.
Pokud použijeme v tomto případě funkci MID$, budou řádky 170 a 180 vypadat následovně:

```
170 A1$=MID$(A$,2,N-2)
180 A2$=MID$(A$,N+1)
```

Je to dáno jiným uspořádáním mezer před znakovým řetězcem. Věnujte prosím této skutečnosti pozornost a důkladně si ji prozkoumejte! V programátorské praxi se vám to vyplatí.

Dělení znakového řetězce na tři a více částí

Praktickým příkladem využití dělení znakového řetězce na tři části je práce se systémovými hodinami. Systémové hodiny TI$ zobrazují čas ve tvaru hhmmss v jednom řetězci. Pro praktické využití není toto zobrazení vhodné a nejčastěji se tento řetězec upravuje pro výpis na obrazovku do tvaru hh:mm:ss. Praktickou konverzi původního tvaru do tohoto zajistí například následující program:

```
100 H$=LEFT$(TI$,2)
110 M$=MID$(TI$,3,2)
120 S$=RIGHT$(TI$,2)
130 C$=H$+":"+M$+":"+S$
140 PRINT CHR$(147)
150 PRINT C$
160 FOR X=1 TO 500: NEXT
170 GOTO100
```

V řádcích 100 – 120 jsou z řetězce TI$ odděleny části odpovídající hodinám, minutám a sekundám. Poté je definován nový řetězec C$, do kterého jsou s mezinaky ":" zřetězeny hodiny, minuty a sekundy.
V řádku 140 je vymazána obrazovka, řádek 150 vytiskne čas ve tvaru hh:mm:ss. Následuje prodleva cca 500 milisekund, protože je zbytečné opakovat zobrazování častěji než 1x za sekundu, když řetězec TI$ kratší časy neobsahuje. Pro nezasvěcené zde uvedu, že ze systémových hodin, které odečítají čas po 1/60 sekundy je možno v případě potřeby „vytáhnout" i časy kratší. To ale není možné v Basicu, který je pro takové účely příliš pomalý. Navíc nepřesnost sítového kmitočtu, jak víte, stejně neumožňuje přesné měření.

(pokračování)

(JK)

ASSEMBLER NA C64

(5.pokračování)

SED, CLD

Oba tyto příkazy slouží k práci s dekadickým příznakem. Příkaz SED (SEt Decimal) je určen pro nastavení příznaku. Opačné příkaz CLD (CLear Decimal) maže dekadický příkaz. Pokud je dekadický příznak nastaven, pracuje procesor v dekadickém módu, jinak v normálním, binárním. Pro úplnost se o dekadickém módu zmíníme ještě na konci kapitoly.

SEI, CLI

Pomocí příkazů SEI (SEt Interrupt) a CLI (CLear Interrupt) se nastavuje a maže příznak přerušení. Pokud je příznak přerušení nastaven, jsou všechna přerušení zablokována. Jedinou výjimku zde tvoří přerušení NMI (Non Maskeable Interrupt) Se zmíněnými typy přerušení se ještě budeme zabývat ve speciální kapitole.

CLV

Pomocí příkazu CLV (CLear oVerflow) se maže příznak přetečení. Příkaz k nastavení tohoto příznaku neexistuje, protože by neměl praktický význam.
A ještě si uvedeme kódy popsaných příkazů pro práci se stavovým registrem.

CLC : $18	CLD : $D8	CLI : $58	CLV : $B8
SEC : $38	SED : $F8	SEI : $78	

Na tomto místě se ještě vrátíme k dekadickému módu, abychom si jej důkladně objasnili.
Kromě takzvaného binárního módu, který již dobře známe, může být procesor využíván ještě v jiném druhu provozu. Tento druh provozu je tzv. dekadický mód. Tento mód se nastavuje velice zřídka, například v matematických programech a dalších, kde jde o vyšší

přesnost výpočtu. Z tohoto důvodu je dekadický mód většinou programátorů ve strojovém kódu vzdálený.
V dekadickém módu lze jedním bajtem zobrazit číslo od 0 do 99. To je možno proto, že bajt je rozdělen na dvě části, tzv. nibbly. Pomocí 4 bitů lze však zobrazit jen čísla od 0 do 15. V dekadickém módu to nevadí, protože se stejně nastavují jen čísla 0 – 9. V dekadickém módu představují 4 spodní bity jednotky a 4 horní bity desítky. Pokud má dolní nibble hodnotu například 3 a horní třeba 7, má bajt hodnotu 73, pokud je aktivní dekadický mód. Dekadický mód je nicméně registrován příkazy ADC a SBC. Je také možné čísla v dekadickém formátu sečítat a odčítat.
Seřazením více bajtů je možno vytvářet čísla s libovolným počtem míst. Pokud se bude pracovat s kladnými i zápornými čísly, musí se pro předznaménko rezervovat jeden nibble. Při desetinných číslech se musí ještě registrovat místo, kde je desetinná tečka umístěna.
Když si uvedené ještě jednou shrneme, lze zkráceně říci, že:
v dekadickém módu lze jedním bajtem zobrazovt čísla jen v rozsahu 0 – 99. Jeden půlbajt (nibble) přitom představuje číslo mezi 0 a 9.
Z toho vyplývá, že tento mód nevyužívá všechny možné kombinace bitů.

K tomu příklad:

Binární číslo %01100111 má v dekadickém módu následující hodnotu:

	MSB				LSB			
Bit:	0	1	1	0	0	1	1	1
Hodnota:	0	4	2	0	0	4	2	1
Celkem:			6				7	

Dekadická hodnota našeho čísla v dekadickém módu je tedy 67.

Jak funguje dekadické sčítání a odčítání ilustrují následující příklady:

1.

%01110000	7	0	příznaky před/po
+ %00100000	+ 2	0	C=0; N=0; V=0
= %10010000	= 9	0	C=0; N=1; V=1

2.

%10010000	9	0	příznaky před/po
+ %00100000	+ 2	0	C=0; N=0; V=0
= %00010000	= 1	0	C=1; N=1; V=0

3.

%01000100	4	4	příznaky před/po
– %01010101	– 5	5	C=1; N=0; V=0
= %10001001	= 8	9	C=0; N=1; V=0

Jak se však dají jednotlivé výsledky v příkladech interpretovat?

V obou prvých příkladech je přenosový příznak před sčítáním nastaven na 0, tedy vymazán. Při odčítání (příklad 3) se přenosový příznak nastaví. Průběh je tedy stejný, jako při normálním, binárním sčítání nebo odčítání.

To zásadní, co se změnilo, je spojovací schema. Neprobíhá po jednotlivých bitech, ale po číslech. Interně se nejdřív sečtou dvě dekadická čísla tvořená spodními nibbly. Výsledek tohoto sčítání se zobrazí opět čtyřmi spodními bity. Pokud při tomto součtu dojde k přetečení, bude toto při sčítání obou horních polovin bajtů zohledněno. Pokud při součtu horních nibblů dojde k přetečení, převezme jej příznak přetečení. Příznak přetečení tak představuje stovku.

Odčítání probíhá analogicky. To znamená, že podtečení se vždy odečte od vyššího místa. Pokud dojde k podtečení u obou odčítaných dekadických čísel, příznak přetečení se vymaže. S výsledkem se pak musí zacházet jako se záporným číslem (příklad 3).

Nyní ještě pár slov k funkci negativního příznaku a příznaku přetečení. Jak již je z uvedených příkladů vidět, je příznak přetečení nastaven vždy, když dekadické číslo nejde nastavit pomocí 3. bitů, t.zn., při nastavení 7. bitu.

Jak se ale počítá s absolutními hodnotami, když dojde k podtečení?

Nic jednoduššího! V binárním módu obrátíme všechny bity a přidáme 1. Přesně tak to taky funguje v dekadickém módu:

Jako první se provede logické spojení EX–OR a dekadickou hodnotou 99. V dekadickém módu to vypadá následovně:

Bit:	7	6	5	4	3	2	1	0
Hodnota:	1	0	0	1	1	0	0	1

Nebo v hexadecimálním vyjádření je to $99. K výsledku tohoto spojení se přidá 1 a již máme absolutní výsledek našeho odčítání. Provedeme si pro názornost popsaný postup s třetím příkladem.

Výsledek:	%	1	0	0	0	1	0	0	1
EXOR :	%	1	0	0	1	1	0	0	1
Výsledek:	%	0	0	0	1	0	0	0	0
plus 1:	%	0	0	0	0	0	0	0	1
Výsledek:	%	0	0	0	1	0	0	0	1

Výsledek je tedy –11.

Pokud nezamýšlíte programovat v dekadickém módu, pak nevadí, když si nyní pod tímto pojmem nedokážete nic představit. Pokud ale jednou dekadický mód využijete, pak vám nebude pochopení výše uvedeného postupu činit problémy.

Příkazy pro posun ASL, LSR, ROL, ROR

Kromě příkazů logických a arltmetických existují ještě příkazy pro posun pro zpracování jednotlivých bajtů. S jejich pomocí je možno například obsah akumulátoru posunout o 1 bit vlevo nebo vpravo. Přitom se rozlišují dva typy příkazů pro posun. Za prvé jsou to příkazy, které posunují obsah registrů vlevo nebo vpravo a na uvolněné místo vsunou 0. U tohoto typu příkazů pro posun se „vystrčený" bit přesune do příznaku přenosu. Pokud je „vystrčený" bit 1, pak se příznak přenosu nastaví a pokud je roven 0, pak se příznak přenosu vymaže. Podíváme se nyní na schema posunu vlevo podrobněji.

(JK)

pokračování

DYSP GRAFIKA

Dále uvedený listing programu je ve formátu TURBO–ASS. V rutině YJUMP jsou pevně stanoveny pozice sprajtů v ose Y. Hodnoty jsou jednoduše čteny z tabulky sinusu, uložené od adresy $4000 (dek. 16384) a jedna po druhé přepisovány do adres $D001, $D003, $D005 atd. Pro každý sprajt se zde používá zvláštní číselný registr. Z hlediska programování je to poněkud složitější postup, ale vyplatí se s ohledem na získaný rastrový čas. Po pevném stanovení všech pozic začíná hlavní práce rutiny. Všechny sprajty jsou posunuty o 4 bajty vlevo (od ST0). Od značky ST4 se pak zkoumá, zda nedošlo k podtečení pozice sprajtu. Pokud ano, vytáhne se odpovídající MSB z tabulky a spojí se s $D010 (dek. 53264). Potom je číslo aktuálního sprajtu uloženo do registru X, aby aby s ním mohl být později situován další znak v předpokládaném sprajtu. Nyní se vytáhne další bajt ze skrolovaného textu a přezkouší se na přerušení ($FF). Pokud je tento bajt různý od $FF, pak bude vytažený znak zvětšen o offset bloku sprajtů (v tomto případě o $C0) a zkopírován do odpovídajícího ukazatele bloku sprajtů ($07F8). Současně se pozice tohoto sprajtu nastaví na $58. Ten se pak nachází na pravém okraji a může se opět s novým obsahem přesunout nalevo. K využití rutiny ve vlastních programech je nutno dbát následujícího:

1. rutinu je nutno naskakovat vždy pomocí JSR "DYSP".
2. rutina očekává znaková dat sprajtů od adresy $3000. Ta musí začínat vždy prázdným sprajtem.
3. Skrolovaný text bude čten od adresy $4300 a musí začínat bajtem pro přerušení $FF.
4. Tabulka sinusu je očekávána od adresy $4000.

Naskakovací adresa hlavní rutiny je $4080, dekadicky 16512. Pro profesionála jistě nebude problémem změnit skrolovaný text, znakovou sadu sprajtů, tabulku sinusu nebo rychlost upravit podle vlastní potřeby. A nyní vlastní výpis

```
 *=4000
SINTAB  .BYTE $14,$14,$14,$15,$16,$16
        .BYTE $16,$17,$18,$19,$1A,$1B
        .BYTE $1C,$1E,$20,$21,$22,$24
        .BYTE $26,$29,$2C,$2E,$30,$33
        .BYTE $36,$39,$3C,$3F,$42,$45
        .BYTE $48,$4B,$4E,$51,$54,$57
        .BYTE $5A,$5D,$60,$64,$68,$6C
        .BYTE $70,$6C,$68,$65,$60,$5D
        .BYTE $5A,$57,$54,$51,$4E,$4B
        .BYTE $48,$45,$42,$3F,$3C,$39
        .BYTE $36,$33,$30,$2E,$2C,$29
        .BYTE $26,$24,$22,$21,$20,$1E
        .BYTE $1C,$1B,$1A,$19,$18,$17
        .BYTE $16,$16,$16,$15
;-------------------------------------------
          *= $4080

BLOCK =  $07F8     ;UKAZATEL NA BLOK SPRAJTU
SPRMC =  $D01C     ;SPRITE MULTICOLOR RE-
GISTR
SPRON =  $D015     ;SPRAJTY ZAP/VYP.
SPRDT = $4000 ;TABULKA SINUSU

HELP0 =  $16E5
HELP1 =  $16E6
YREG0 =  $16E7
YREG1 =  $16EA     ;HODNOTA SINU SPRAJT#0
YREG2 =  $16EB     ;HODNOTA SINU PRAJTU#1
```

```
YREG3 = $16EC    ;HODNOTA SINU SPRAJT#2
YREG4 = $16ED    ;HODNOTA SINU PRAJTU#3
YREG5 = $16EE    ;HODNOTA SINU SPRAJT#4
YREG6 = $16EF    ;HODNOTA SINU PRAJTU#5
YREG7 = $16F0    ;HODNOTA SINU SPRAJT#6
YREG8 = $16F1    ;HODNOTA SINU PRAJTU#7

         SEI
         LDA #>START  ;IRQ VYPNUTO
         LDY #<START  ;VLASTNI PROGRAM
         STA $0315    ;ODKLONENI
         STY $314

         LDA #$00     ;PRIPRAVIT VSECHNY
         STA YREG0    ;REGISTRY
         STA $D020
         STA $D021
         LDA #<SCRTXT ;
         STA THB+1    ;NASTAVENI ZACATKU
         LDA #>SCRTXT ;TEXTU PRO SKROLOVANI
         STA THB+2    ;LDX #$00
         TXA
         L1 STA XREG,X ;ULOZIT OSM RUZNYCH
         INX          ;HODNOT SINUSU DO
         CLC          ;OSMI REGISTRU
         ADC #$0A
         CPX #$08
         BNE L1
         LDA #$01     ;PRIPUSTNY JEN
         STA $D01A    ;REGISTR IRQ

         JSR SPRIN    ;INICIOVAT SPRAJTY

         LDA #$7F     ;PORT DMA
         STA $DC0D    ;PREPNOUT
         CLI
ENDLESS  JMP ENDLESS  ;NE ZPET DO BASICU
START    LDA #$37     ;CEKAT NA RASTROVOU
ST1      CMP $D012    ;POZICI $37
         BNE ST1      ;LDA #$00 ;VYMAZAT
         STA $3FFF    ;GHOSTBAJT
         LDA #$1B     ;HRANICE Y ZAPNUTA
         STA $D011    ;
         JSR DYSP     ;HLAVNI RUTINA
         LDA #$F9     ;CEKAT NA RASTROVOU
ST2      CMP $D012    ;POZICI $F9
         BNE ST2      ;
         LDA $FF      ;GHOSTBAJT VYPLNIT
         STA $3FFF    ;
         LDA #$13     ;HRANICE Y VYPNUTA
         STA $D011    ;
         JMP $EA31    ;STARE IRQ
```

```
DYSP     JSR YJUMP    ;RUTINA Y-SINUS
         LDX #$0E     ;
ST0      LDA $D000,X  ;
         SEC          ;
         SBC #$04     ;VSECHNY SPRAJTY
         STA $D000,X  ;SKROLOVAT O $04
         DEX          ;NALEVO
         BPL ST0      ;
         LDX #$0E     ;ZACIT SPRAJTEM
         LDY #$07     ;$07 ($0E = X-POZICE
ST4      LDA $D000,X  ;SPR.7, $07 = MSBTAB
         CMP #$FC     ;SPR.7) A
         BNE ST3      ;TESTOVAT NA MSB
         LDA $D010    ;SPOJIT S MSB
         EOR MSTAB,Y  ;POKUD
         STA $D010    ;SPRPOS NENI MENSI
         AND MSBTAB,Y ;NEZ $00
         BEQ ST3      ;POKUD ROVNO $00,
                      ;POTOM DALSI SPRAJT
         STX HELP0    ;POZNAMENAT CISLO
         STY HELP1    ;AKTUALNIHO SPRAJTU
         TYA          ;CISLO SPRAJTU (0-7)
         AND #%011111111 ;VYTAHNOUT A HIBIT
         TAX          ;VYJMOUT, CISLO
                      ;SPRAJTU DO X
         INC YREG0    ;BAJTOVY CITAC
         LDY YREG0    ;PRO TEXT
THB      LDA SCRTXT,Y ;NATAHNOUT TEXTBYTE
         CMP #$FF     ;POKUD PRERUSOVACI
         BNE THB2     ;BAJT PAK OPET
         LDA #>SCRTXT ;ZACATEK TEXTU
         STA THB+$02  ;NASTAVIT
         LDA #<SCRTXT ;OPET NA $3800
         STA THB+$01  ;A CITAC BAJTU
         STA YREG0    ;NASTAVIT NA $00
         JMP THB3     ;POKRACOVAT
                      ;DALSIM SPRAJTEM
THB2     LDY YREG0    ;POKUD SE LOWBAJT
         CPY #$FF     ;CITACE BAJTU
         BNE THB1     ;ROVNA $FF PAK
         INC THB+$02  ;HAJBAJT + 1
THB1     CLC          ;K VYTAZENEMU ZNAKU
         ADC #$C0     ;PRIDAT OFFSET BLOKU
         STA BLOCK,X  ;SPRAJTU
                      ;A DO PRISLUSNEHO
                      ;SPR.BLOKU
                      ;ZAPSAT UKAZATEL
THB3     LDX HELP0    ;VYTAHNOUT CISLO SPRAJTU
         LDA #$58     ;A NASTAVIT ZPET NA $58
         STA $D000,X  ;(MSB NASTAVEN)
         LDY HELP1    ;CISLO SPRAJTU
                      ;VYTAHOVAT
ST3      DEX          ;AZ JE VSECH 8 BAJTU
         DEX          ;PREZKOUSENO
```

```
        DEY             ;A ZAPSANO
        BPL ST4         ;
        RTS             ;
YJUMP   LDX XREG4       ;BAJTOVY CITAC
        CPX #$52        ;HODNOTY SINUSU
        BNE YJ1A        ;A PREZKOUSET ZDA
        LDA #$00        ;ZDA BYLA JIZ
                        ;ZPRACOVANA
        STA XREG4       ;CELA TABULKA
        JMP YJ2         ;
YJ1A    LDA SPRDT,X     ;POKUD NE PAK
                        ;VYTAHNOUT
        STA $D001       ;X-TY PRVEK A ULOZIT
        INC XREG4       ;DO Y-POS SPR0
YJ2     LDX XREG5       ;BAJTOVY CITAC
        CPX #$52        ;HODNOTY SIN2
                        ;VYTAHNOUT
        BNE YJ2A        ;A PREZKOUSET ZDA
                        ;UZ BYLA
        LDA #$00        ;ZPRACOVANA CELA
        STA XREG5       ;CELA TABULKA
        JMP YJ3         ;
YJ2A    LDA SPRDT,X     ;POKUD NE PAK
                        ;VYTAHNOUT
        STA $D003       ;X-TY PRVEK
                        ;A ULOZIT JEJ
        INC XREG5       ;DO Z-POS SPR0
YJ3     LDX XREG5       ;VIZ YJ2
        CPX #$52        ;
        BNE YJ3A        ;
        LDA #$00        ;
        STA XREG6       ;
        JMP YJ4
        RTS
YJ3A    LDA SPRTD,X
        STA $D005
        INC XREG6
YJ4     LDX XREG7
        CPX #$52
        BNE YJ4A
        LDA #$00
        STA XREG7
        JMP YJ5
YJ4A    LDA SPRDT,X
        STA $D007
        INC XREG7
YJ5     LDX XREG8
        CPX #$52
        BNE YJ5A
        LDA #$00
        STA XREG8
        JMP YJ6
YJ5A    LDA SPRDT,X
        STA $D009
        INC XREG8
YJ6     LDX XREG9
        CPX #$52
        BNE YJ6A
        LDA #$00
        STA XREG9
        JMP YJ7
YJ6A    LDA SPRDT,X
        STA $D00B
        INC XREG9
YJ7     LDX XREGA
        CPX #$52
        BNE YJ7A
        LDA #$00
        STA XREGA
        JMP YJ8
YJ7A    LDA SPRDT,X
        STA $D00D
        INC XREGA
YJ8     LDX XREGB
        CPX #$52
        BNE YJ8A
        LDA #$00
        STA XREGB
        RTS
YJ8A    LDA SPRDT,X
        STA $D00F
        INC XREGB
        RTS
MSTAB   .BYTE $01,$02,$04,$08,$10,$20,
        .BYTE $40,$80

;-----------------------------------------
SPRIN   LDX #$07
        LDA #$C0
SPR0    STA BLOCK,X
        DEX
        BPL SPR0

        LDA #$FF
        STA SPRMC
        STA SPRON

        LDA #$02
        STA $D025
        LDA #$07
        STA $D026
        LDX #$07
        LDA #$0A
SPR1    STA $D027,X
        DEX
        BPL SPR1

        LDA #$80
```

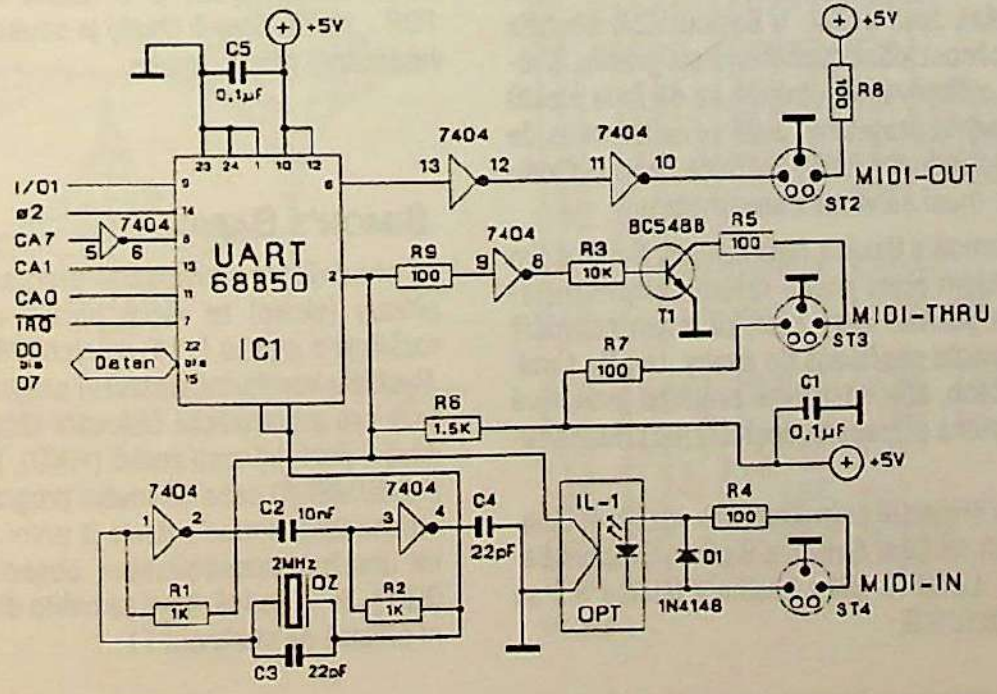
```
        LDX #$0F
SPR2    STA $D000,x
        DEX
        DEX
        BPL SPR2

        LDX #$00
        LDA #$30
SPR3    STA $D000,X
        CLC
        ADC #$2C
        INX
        INX
        CPX #$0A
        BNE SPR3

        LDA #%11100000
        STA $D010
        LDX #$00        LDA #$08
SPR4    STA $D00A,X
        CLC
        ADC #$2C
        INX
        INX
        CPX #$06
        BNE SPR4

        LDA #$FF
        STA $D01B
        RTS
;-------------------------------------------------
        *= $4300
SCRTXT  .BYTE $00,$01,$01,$01,$01,$01
        .BYTE $01,$01,$01,$01,$01,$01
        .BYTE $01,$01,$01,$01,$01,$01
        .BYTE $01,$01,$01,$01,$01,$01
        .BYTE $01,$01,$01,$01,$00,$FF
;-------------------------------------------------
```

konec
(jk)

MIDI HARDWARE

V únorovém čísle FUNu byl uveřejněn článek popisující normu MIDI. Jednu z možností praktického provedení MIDI interfejsu uvádí připojené schema. Zapojení podle obr. bylo realizováno na oboustranné desce plošných spojů o rozměrech 10 x 7 cm ze součástek, které je možno zakoupit v obchodech KTE Electronic a GM Electronic. Modul, jehož ústředním členem je obvod UART 68850 se připojuje do expanzního portu C64, odkud je také zajištěno jeho napájení. Interfejs komunikuje s okolím přes vývody MIDI–IN, MIDI–THRU, MIDI–OUT, osazené konektory DIN 5.

Firma Comotronic nemá v současné době možnost zajistit vývoj software, nutného pro uživatelské využití tohoto často žádaného hardware. K dispozici jsou jen programové moduly pro řízení a testování interfejsu. Program MIDITEST, sloužící k testu komunikace s klá-



vesnicí si po spuštění dotáhne ještě z diskety program MIDIFILE a provede test vysílání dat na klávesnici a příjem stejných dat z klávesnice.
MIDIFILE navíc umožňuje i z Basicu pomocí příkazů SYS a PEEK testovat stisk kláves.
Program RECIEVETEST zobrazuje na obrazovce data přijímaná z klávesnice včetně stavových bajtů.
Uvedené programové bloky ve strojovém kódu jsou základními kameny, na nichž je již možno vystavět uživatelskou nádstavbu. Pokud budete mít zájem se tvorbou potřebného software zabývat, firma Comotronic je ochotna Vám poskytnout kompletní interfejs MIDI dle výše uvedeného schema včetně popsaných programů na disketě za symbolickou cenu 380,– Kč. Pokud bude zájemců více, uvažujeme o vypsání soutěže s atraktivními cenami a u vítězného programu zajistíme autorovi jeho komerční využití.

PŘEHLED BASICŮ PRO C64

To, že se v lidské společnosti vyvinula řada různých jazyků má mnoho vědecky doložitelných příčin. Jednou z nich jsou jistě rozdílné možnosti a mentalita daného národa (jeho hrdost), druhou pak velké vzdálenosti mezi skupinami obyvatelstva, které jsou ještě umocněny geografickými bariérami.

Zcela jiné příčiny vedly k různosti počítačových jazyků. Také zde zafungovala odlehlost jednotlivých vývojářů software, spíše se však projevila jejich pýcha. Jen málo z nich bylo ochotno kopírovat už hotový produkt, což bylo ke škodě v kompatibilitě jednotlivých programů. Navíc speciální problémy vyžadovaly pro co nejjednodušší vyjádření možností počítače, speciální řešení. Tak postupně vznikla celá řada programovacích jazyků: Basic, Pascal, C nebo Fortran. V rámci jednotlivých jazyků se vyvinula řada „dialektů". Zvláště nápadná je popsaná situace u Basiců. Začátečník se ihned po uvedení C64 do chodu setká s Basicem V2.0. Počítače C16, C116, Plus 4 hovořily jazykem Basic 3.5 a krátce po té přišel C128 s Basicem V7.0 Počítače tak dostaly to co potřebovaly.

Vraťme se však zpět k C64. V Basicu V2.0 chyběla uživatelům možnost jednoduchého řízení grafiky. Kvůli problémům s časováním obvodů se dá řada efektů řešit jen strojovými programy. Jistě se nelíbí ani to, že k nakreslení jednoduché linky, kružnice nebo byť jediného bodu, se musí sáhnout k assembleru.

Proto vznikl Simon's Basic a řada dalších. Spolu s tím se objevil problém zcela jiného ražení. Jakými metodami určit, jak poznat, který z jazyků je ten nejlepší? Testování zpravidla nepřipadá do úvahy. Uvažte sami. Kdo má tolik času, aby intenzivně prohlížel jednotlivé produkty. V mnoha případech postačí znát pouze soubor příkazů.

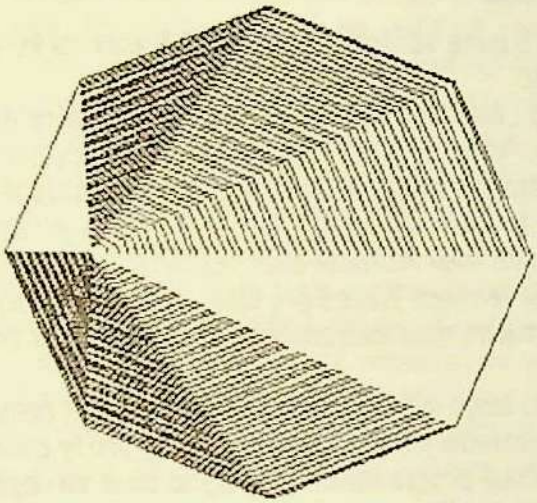
Nabízíme vám srovnání nejznámějších rozšíření Basiců použitelných na C64, Simon's Basicu, Graphic Basicu, Exbasicu Level II, Ultra Basicu a Basicu 3.5 se standardem Basic 2.0.

BASIC 2.0

Programovací jazyk Basic 2.0 rozhodně uživateli C64, zvláště začátečníkovi, práci s počítačem nijak neusnadní. Umožňuje zadefinovat tři druhy proměnných (real, integer, string) a provádět s nimi matematické a logické operace (sčítání, odečítání, násobení, dělení, umocňování a porovnání – je roven, je větší, je menší, menší nebo roven, větší nebo roven, není roven). Dále je vybaven příkazy potřebnými pro editování programu, jeho ládování a ukládání na externí paměťová média, jakož i pro čištění obrazovky a paměti. Většinou se tyto příkazy používají v přímém módu. Další velkou skupinu tvoří příkazy, které se vyskytují v příkazových řádcích a jsou nosným jádrem všech programů. Mimo to má řadu matematických funkcí a funkcí s řetězcovými proměnnými. Úplně mu chybí příkazy pro jemnou grafiku a programování zvukových efektů. Grafika, zvuk a sprajty se musí konstruovat složitě pomocí příkazů PEEK a POKE. Jazyku chybí možnost systematického, komfortního programování. Konstrukce smyček je omezena na použití příkazu FOR...NEXT. Stejně chudý je soubor příkazů pro rozhodování i příkazy skoku.

SIMON'S BASIC

Je vhodný pro universální programátory. Obsahuje příkazy (skoro) ze všech oblastí využití C64. Basic rozšiřuje o stovku (114) nových příkazů.
Rychlé a komfortní sestavení programů umožňují příkazy pro automatické číslování řádků (AUTO), vyhledávání posloupností znaků (FIND), přečíslování řádků (RENUMBER) nebo trasování programu (TRACE). Mimochodem, Simon's Basic je první z programů, který mi umožnil bezproblémově obsadit funkční klávesy (KEY). Je výhodné, když se místo dlouhého vypisování příkazů dá stisknout F1...



```
*** EXPANDED CBM V2 BASIC ***

30719 BASIC BYTES FREE

READY.
AUTO10,10

READY.
10 HIRES1,6
20 FORX=5TO100STEP2
30 ARC40+X*.8,100,315,90,45,X*1.2,X,1
40 ARC40+X*.8,100,135,270,45,X*1.2,X,1
50 NEXTX
60 ARC120,100,0,360,45,120,100,1
70 GOTO70
```

Výkonnost programů zvyšují i příkazy pro strukturované programování, známé z vyšších jazyků. Sem patří příkazy pro tvorbu lokálních i globálních proměnných (LOCAL,GLOBAL), vnitřních procedur (PROC LABEL, EXEC PROC, END PROC), skoky na určitý řádek či návěští (C GOTO X, CALL LABEL), podmíněné skoky a smyčky (EXIT IF, REPEAT UNIT). Dále obsahuje funkce a skoky podmíněné chybou (ON ERROR GOTO).

Netušené možnosti otevírá uživatelům C64 Simon's Basic v oblasti konstrukce grafiky, se vším, co k tomu patří. Najdete v něm příkazy pro tvorbu sprajtů (MOBSET, MMOB...), pro jemnou grafiku v zobrazení HIRES 320x200 bodů (HIRES, PLOT, REC, CIRCLE, ANGLE, LINE...). Simon's Basic má příkazy nejen pro sestavení grafiky, ale i pro její vytištění tiskárnou (COPY, HRDCOPY). Nechybí ani příkazy pro syntézu zvuků a pomůcky pro kontrolu polohy joysticku, světelného pera a paddle.

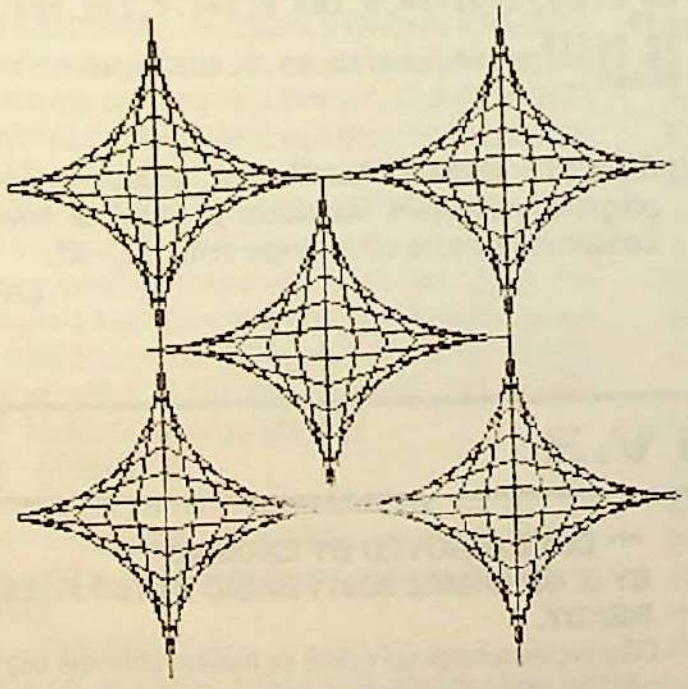
Přirozeně, žádný program není dokonalý, tedy ani Simon's Basic ne. Vytknout se mu dá značná pomalost při provádění grafických operací.

Graphic Basic

Tento graficky orientovaný Basic vznikl zhruba ve stejné době jako Simon's Basic a přestože se jedná o kvalitní program, nikdy nedosáhl jeho rozšíření.

Obsahuje přes 100 rozšiřujících příkazů. Po natažení a odstartování základního programu zůstává v paměti místo pro cca 20 kB vlastních programů.

G. Basic využívá obrazovku ve třech režimech, text (25x40 znaků), HIRES (320x200 bodů) a multicolor (160x200 bodů). Všechny tři režimy lze kombinovat. Podobně jako Simon's Basic (SB), obsazuje i Graphic Basic (GB) funkční klávesy (F1: Run, F3: LIST, F5:



```
10 SZ=9:REM MULTI DIAMOND DRAW BY
20 RESET:REM JAY STEVENS
30 CLEAR:REM 11/3/83
40 HIRES:C=1:C(1)=2:C(2)=5:C(3)=6
50 HIRESCOLOR WHITE ON BLACK
60 BACKGROUND BLACK:R=1
70 C=C+1:IFC>3THENC=1
80 COLOR C(C)
90 COLOR 15*RND(8)+1
100 SETORIGIN 0,0
110 IFR=1THEN S=50:E=50
120 IFR=2THEN S=50:E=155
130 IFR=3THEN S=160:E=50
140 IFR=4THEN S=160:E=155
150 IFR=5THEN S=105:E=105
160 W=50:LL=W*2+5
170 X=S:Y=E
180 LINE X,Y
190 FORA=WTO0 STEP-SZ
200 N=J+SZ
210 X=X+A:Y=Y+N:LINE TO X,Y
220 J=N

230 X=X+A:Y=Y-N:LINE TO X,Y
240 X=X-A:Y=Y-N:LINE TO X,Y
250 X=X-A:Y=Y+N:LINE TO X,Y
260 X=X+SZ:LINE X,Y
270 NEXT:J=0
280 X=S+LL:LINE X,Y
290 X=S-(SZ/2):Y=E:LINE TO X,Y
300 R=R+1:IFR<6THEN70
310 FORT=1TO1000:NEXT
320 GOTO320

READY.
```

DIR...). V grafice nechybí příkazy pro definování barvy obrazovky a rámečku (BACKGROUND, BORDER). Samozřejmostí jsou příkazy pro rozsvícení určitého bodu, nakreslení přímky. Obdélníky lze definovat zadáním levého dolního rohu a délkou strany nebo levým dolním a pravým horním rohem. Pro kružnice, elipsy, mnohoúhelníky je příkaz CIRCLE. Zadáním příslušných parametrů se rozhodne o kreslení daného obrazce. Program umožňuje zadefinování měřítek a jejich změnu (SCALE).

Shodně se SB má i GB pro definování, kopírování sprítů, pro kontrolu jejich pohybu a kolizí, řadu příkazů. Program nezapomněl ani na možnost tvorby zvuků. Dále má příkazy pro definování vlastní znakové sady a práci s ní. Shodně jako SB umožňuje strukturované programování.

EXBASIC LEVEL II

V počtu příkazů není zdaleka tak bohatý jako Simon's Basic. Je však účinný při sestavování programů. Obsahuje příkazy pro automatické číslování řádků, skrolování nahoru a dolů, obsazení funkčních kláves a řadu dalších. K dispozici jsou i elementární příkazy pro grafiku (pseudografika) a zvuk, jakož i funkce pro zaokrouhlování a výpočet maxima a minima. Obsahuje i několik příkazů pro strukturované programování (IF...THEN...ELSE).

ULTRABASIC 64

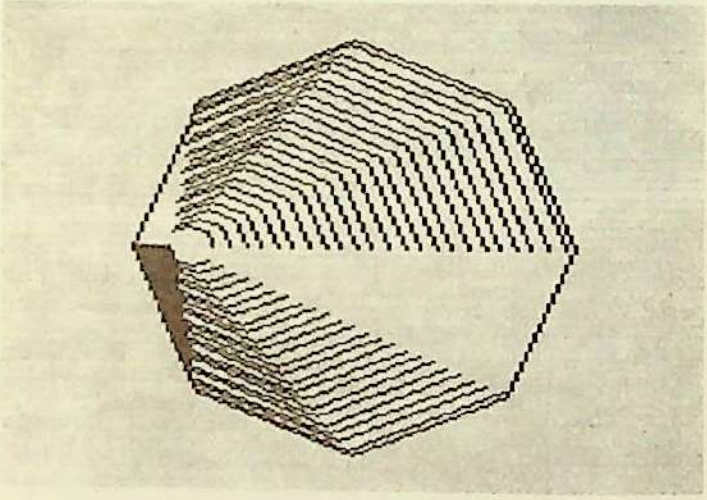
Ultrabasic je z řady těch programovacích jazyků, které rozšiřují základní Basic 2.0 převážně o grafické příkazy. Obsahuje příkazy pro grafiku typu želva – Turtle grafik (TURTLE, TUP, TDOWN, MOVE, TURN atd.) V kombinaci s funkcí pro ovládání joysticku, se dají sestavovat programy pro kreslení na obrazovku pomocí joysticku. Grafiku podporují i další příkazy (DRAW, BOX, BLOCK, CIRCLE, DOT). TIC usnadní orientaci na obrazovce, formou rastru zobrazí popis os X a Y. Ultrabasic podporuje i tvorbu sprajtů. Po provedení každého grafického příkazu zůstává, na rozdíl od Simon's Basicu, grafika zapnuta. Kromě grafických příkazů má Ultrabasic i příkazy pro vytváření hudby.

BASIC 3.5

Basicu 3.5 jsme věnovali samostatný článek na konci roku 1992.

Basic 3.5 „umí" téměř vše, co předchozí Basicy dohromady. Však taky byl vymyšlen renomovanými programátory firmy COMMODORE a byl mazistupněm ve vývoji BASICu V7.0 pro C128. Asi i proto je těmi, kdo se s ním měli možnost blíže seznámit, označován za nejlepší ze všech Basiců pro C64, na čemž se podílí především jeho mnohostrannost a optimální výběr příkazů.

A abyste o tento výborný programovací nástroj nebyli ochuzeni, zařadili jsme jej do naší nabídky pro ty z vás, kteří se věnují programování a chtějí to dělat co nejlépe a nejefektivněji.



```
10 GRAPHIC3,1
20 FORX=5TO50STEP2
30 CIRCLE1,40+X*.8,100,X,X*1.7,315,90,0,
45
40 CIRCLE1,40+X*.8,100,X,X*1.7,135,270,0
,45
50 NEXTX
60 CIRCLE1,80,100,50,85,0,360,0,45
READY.
```

BASIC 3.5 na disketě/kazetě spolu s řadou demo–programů a kvalitním manuálem je v nabídce firmy Comotronic s.r.o. za uživatelskou cenu 125,– Kč.

(JV)

EXOS V.3

Exos V.3 je jeden z nejrychlejších floppy–speederů, který je znám. Vedle 14x rychlejšího nahrávání, dostanete k dispozici ještě vysoce výkonný operační systém s dalším rozšířením.

Operační systém se ohlásí světle šedým písmem na černém pozadí:

***** C64 IMPROVED BY EXOS V3 *****
BY J. SCHMMEL 38911 BASIC BYTES FREE
READY.

Díky rychlé resetovací rutině se hlášení objevuje okamžitě po zapnutí počítače. Loader urychluje nahrávací rutinu (LOAD) 12x až 14x. Jaké urychlení se využije,

záleží na poloze stop, obsazených daným programem na disketě. Programy uložené na vnějších stopách diskety, tedy stopách s nízkými čísly, budou natahovány rychleji než stopy vnitřní, s vyššími čísly. Ostatně, rozdíl zde činí asi 20%. Program, dlouhý 200 bloků, uložený na vnějších stopách diskety, se natahuje asi 12 s. Pokud ovšem program leží blízko středu diskety, trvá natahování cca 14,5 s.

Během natahování programu je obrazovka odepnuta, ohmatávání klávesnice zůstává aktivní. Všechny informace o stisknutých klávesách se přenášejí do vyrovnávací paměti. Můžete tedy zadat jak LIST, tak i RUN. Ihned po dokončení loadingu se zvolené příkazy provedou. Proces lze kdykoli přerušit stiskem klávesy <RUN/STOP>.

Loader je kompatibilní s velkou částí programů provozovaných na C64. Bezvadně natahuje i vícedílné programy, avšak všemocný není. Pokud vzniknou při nahrávání programů problémy, dá se loader odpojit příkazem <CTRL A>. Při použití příkazu LIST se dá skrolovaná obrazovka zpomalit nejen klávesou <CTRL>, ale i klávesou <CBM>.

Exos obsazuje nejčastěji používanými příkazy funkční klávesy. Pro uživatele to znamená, že již nadále nemusí pracně vypisovat dané příkazy, ale stačí mu, aby stiskl příslušnou klávesu.

Funkční klávesy

< F1 > – LIST < RETURN > – odpovídá zadání basicovského příkazu List, prohlédne zapsaný program.

< F3 > – RUN < RETURN > – okamžitě startuje program uložený v paměti počítače.

< F5 > – LOAD""

– odpovídá natipování návěští LOAD. První uvozovky se píší automaticky. Ve spojení s klávesou <F7> dokáže natáhnout program do počítače každý. Stiskem <F7> vyrobte na obrazovce directory diskety. Najeďte kurzorem na začátek řádku s názvem programu a stiskněte <F5>. Stisk klávesy zajistí napsání příkazu LOAD před název programu. Po stisku <RETURN> se potom příkaz provede. Na rozdíl od normálního systému se programy natahují absolutně (LOAD"xxx",8,1). Pokud stojíte o natažení programu standardním způsobem, musíte zapsat LOAD"xxx",8. Místo normálního hlášení se objeví "LOADING TO ADR". Tím se míní adresa, na kterou se program uloží.

< F7 > – LOAD "$",8

– bez ztráty programu natáhne directory.

< F2 > – SYS 32768 < RETURN > – pokud je na této adrese uložen strojový program, stiskem klávesy se odstartuje.

< F4 > – SYS 49152

– vypíše na obrazovku příkaz SYS 49152 a čeká na potvrzení příkazu klávesou < RETURN >. Teprve potom je odstartován z tohoto paměťového místa program (pokud je na něm k dispozici).

< F6 > – SAVE""

– slouží k uložení programu. První uvozovky se píší automaticky shodně jako v případě příkazu pod klávesou < F5 >. Další uvozovky a ,8 nemusí již být uvedeny, Exos je bere jako by byly.

< F8 > – CLOSE: OPEN 7,8,15,"

– Otevírá povelový kanál k floppy. Po uvozovkách zadejte požadovaný příkaz.

Vedle obsazení funkčních kláves se dají některé funkce vyvolat stiskem klávesy < CTRL > spolu s některou z kláves.

Programovací pomůcky

< CTRL + K > – Chybový kanál

– na obrazovce ukáže hlášení poruchy, která se vyskytla.

< CTRL + O > – OLD

– vrací zpět program určený příkazem NEW k vymazání, ev. po resetu počítače ztracený program (OLD). Program je potom přirozeně plně editovatelný a listovatelný. Rutina OLD se se při vracení programu zpět řídí aktuální startovací adresou v paměti Basicu. Proto se mohou vracet zpět i programy, které mají posunutý počátek Basicu.

< CTRL + F > – vyšší frekvence IRQ

– zvýší IRQ frekvenci floppydiskové jednotky a tím i rychlost hlavičky. V praxi to znamená, že všechny disketové příkazy, které jsou náročné na pohyby hlavičkou se provádějí rychleji: př. S(cratch), V(alidate) atd.

< CTRL + X > pokračování LIST

– prohlíží program v basicu od naposled zpracovávaného řádku dál. Pokud objevíte při prohlížení programu chybu, zastavíte prohlížení, chybu odstraníte a stiskem < CTRL + X > spustíte další prohlížení, aniž byste museli znovu zadávat příkaz. Ne vždy se však dá rekonstruovat adresa naposled zpracovávaného řádku. V takovém případě začíná prohlížení programu od začátku.

< CTRL + Z > LIST mínus 50 řádků

– pracuje shodně s příkazem < CTRL + X >, pouze počátek prohlížení je posunut o 50 řádků před naposled zpracovávaným řádkem.

< CTRL + U > ukládání RAM pod ROM Basic

– zdá se to být nelogické, ale vysvětlení je snadné. Je-li v paměti program, který zabírá na disketě místo větší než 151 bloků, leží jeho konec v paměti pod ROM Basic. Při pokusu natáhnout delší program se objeví hlášení OUT OF MEMORY ERROR. S tímto hlášením se ovšem můžete setkat i v případě, kdy koncová adresa natahovaného programu je vyšší než koncová adresa paměti Basicu. Nyní si představte, že tento program máte uložit příkazem SAVE. Objeví se následující problém: Za pamětí Basicu následuje ROM–Basic. Pod ROM–Basic je však ještě RAM. Zapisovat

do ní je však možno pouze z Basicu. Při čtení se však nedostanou stejné hodnoty, ale obdrží se hodnoty z Basic–ROM. Tvrzení se dá lehce dokázat. Zadejte POKE 45000,2 a potom PRINT PEEK (45000). Dostanete číslo, které nebude rovno 2. Avšak zpět k našemu programu. Jeho konec leží v RAM pod ROM–Basic. Uložte program pomocí rutiny SAVE. Jednotlivé bajty programu se čtou přesně jako příkazem PEEK. Budou se tedy číst hodnoty z ROM–Basic a budou se ukládat na disketu. Na disketě potom bude chybět konec programu, který se nacházel v části paměti pod ROM–Basic. Stiskem < CTRL + U > se exosová rutina SAVE přetvoří tak, že bude správně ukládat i obsah zmíněné části RAM. Příkazem SAVE se nyní mohou ukládat i programy delší než 151 bloků. Současným stiskem < RUN/STOP + RESTORE > lze tuto funkci opět vypnout.

< CTRL + A > vypnutí turboloadru

– po této kombinaci kláves se odepíná rychlé nahrávání a pracuje se opět se standardní rutinou. Tato funkce tedy dovolí zpracovat i kolizní programy, které nejsou s urychlovačem exosu kompatibilní. K vrácení do výchozího stavu postačí stisk < RUN/STOP + RESTORE >.

RAM – Floppy

Nejedná se sice o žádné plnohodnotné RAM–floppy, jaké se dá obvykle najít u velkých počítačových systémů, ale dovoluje využít 24 kilobajtů paměti, která normálně nemá v C64 pod Basicem žádnou funkci. Pomocí RAM floppy mohou být programy psané v Basicu ukládány do 4 různých paměťových rozsahů RAM a z nich mohou být natahovány zpět. Rozsahy 1 až 3 mohou být používány současně, rozsah 0 lze použít pouze samostatně.

< CTRL + W > ukládání do RAM floppy

– ukládá program v Basicu do RAM–floppy. Volba žádaného rozsahu RAM–floppy se děje malým menu:

(0)	20k	$B0 – FF
(1)	4k	$B0 – BF
(2)	4k	$C0 – CF
(3)	4k	$D0 – FF

Jednoduše se stiskne odpovídající cifra a funkce se v žádaném rozsahu provede. Stiskne-li se jiné tlačítko, funkce RAM–floppy se přeruší.

< CTRL + L > ládování z RAM–floppy

– natáhne určitou část RAM–floppy do pracovní paměti

< CTRL + V > výměna aktuálního basicovského programu s RAM–floppy

Ukládání obsahu obrazovky

Systém Exos zpracovává 4 různé obrazovkové paměti, které leží v rozsazích od $A000 do $AFFF (40960–45055) pod ROM–Basic.

< CBM + F1 > a < CBM + F3 > – pomocí kombinace daných kláves se dá obsah obrazovky uložit do jednoho ze dvou bufferů. Při používání tohoto příkazu dejte pozor, neboť RAM–floppy a paměť obrazovky používají stejný adresový rozsah.

Všeobecné poznámky.

Při používání příkazů LOAD a SAVE se nemusí zadávat žádné číslo přístroje. Programy se automaticky nahrávají ev. ukládají s adresou přístroje 8 a sekundární adresou 1 (,8,1).

Exos urychluje a usnadňuje práci s disketovou jednotkou. Rutiny pro práci s datassettem přišly zkrátka. Tato skutečnost se dá tolerovat, neboť systém Exos lze odpojit a k dispozici zůstane originální systém Commodore.

(JV)

PROGRAMY MSE

Nikdo není dokonalý. Každý počítačový fanda, lhostejno, zda úplný zelenáč nebo ostřílený profík, dělají při opisování programů chyby. Jejich pozdější odhalení může činit skutečné problémy. Proto byly vyvinuty programy Checksummer a MSE. Prvý program slouží pro opisování programů v basicu, další pak pro zpracování listingů ve strojovém kódu.

CHECKSUMMER V3

Nejprve opište připojený listing. Zvláště pozorně opište řádky s daty. Opsaný program uložte na kazetu či disketu. Vlastní práci s programem začněte následovně:

1. Odstartujte program Checksummer – RUN, RETURN
2. Pokud se na obrazovce objeví hlášení, že program je aktivován, potom jste při opisování neudělali žádnou chybu.
3. Pro vymazání programů v basicu klidně použijte příkaz NEW, Checksummer zůstane zachován.
4. Checksummer můžete otestovat. Zadejte prosím následující řádek a potvrďte RETURN:

1 REM

V levém horním rohu obrazovky se objeví v lomené závorce kontrolní součet. Musí činit <63>. Programu je přitom jedno, kolik mezer uděláte a zda jste napsali 1REM nebo 1 REM. O přesný počet mezer jde v přípa-

dě, že jste je udělali mezi uvozovky. Kontrolní součet se vypíše vždy, když je aktivován Checksummer. Napíšete řádek v basicu a potvrdíte jej stiskem klávesy RET. Při bezchybném opisování musí hodnota uvedená na obrazovce, souhlasit s hodnotou uvedenou v lomené závorce na konci řádku v listingu zpracovávaného programu (viz MSE 1.1). Výpisy programů mohou vypadat následovně:

```
5  PRINT CHR$(14)                          <242>
10 PRINT "{CLR}"                           <245>
20 PRINT "{4DOWN,2SPACE}
   TEST{SPACE,BLUE,6SPACE}"                <...>
```

V řádku 10 po příkazu PRINT následuje výraz CLR ve složené závorce. Znamená to, že máte stisknout SHIFT a CLR/HOME. V řádku 20 je situace obdobná, vyžaduje se nejprve 4 krát stisk SHIFT a kurzor dolů, potom stisk 2 krát mezerník (SPACE). Dále připojte slovo TEST a řádek ukončete stiskem 2 krát SPACE, CTRL a 7 (BLUE) a opět 6 krát SPACE. Někdy se ve výpisech vyskytují podtržená písmena. Narazíte-li na takové, stiskněte SHIFT a dané písmeno. Podobně u písmen s pruhem stiskněte klávesu C= a dané písmeno. Ve složených závorkách se často objevují následující symboly:

{DOWN}	klávesa kurzor dolů (CRSR UP/DOWN)
{UP}	kurzor nahoru, tj (SHIFT+ CRSR UP/DOWN)
{CLR}	SHIFT+ druhá klávesa zprava v horní řadě (CLR/HOME)
{DEL}	klávesa v pravém horním rohu (INST/DEL)

```
1 PRINT""
10 PRINT:PRINT"CHECKSUMMER C64"
11 PRINT"prosim cekejte..."
12 FOR I=828 TO 864:READ A:POKE I,A:PS=PS+A:NEXTI
13 IF PS<>5765 THEN PRINT"PREKLEP V RADCICH 20 AZ 22":END
14 SYS 828:PS=0:FOR I=58464 TO 58583:READ A:POKE I,A:PS=PS+A:NEXT I
15 IF PS<>16147 THEN PRINT"PREKLEP V RADCICH 22 AZ 30":END
16 POKE1,53:POKE 42289,96:POKE42290,228
17 PRINT" checksummer je aktivovan"
18 PRINT" vypnout : poke 1,55 nebo"spc(27)"<run/stop+restore>"
19 PRINT:PRINT" ZAPNOUT : POKE 1,53"
20 DATA169,0,133,254,162,1,189,93,3,133,255,160,0,177,254
21 DATA145,254,136,208,249,230,255,165,255,221,95,3,208,238,202
22 DATA16,230,96,160,224,192,0,160,2,169,0,170,133,254,177
23 DATA95,240,40,201,32,208,3,200,208,245,133,255,138,41,7
24 DATA170,240,14,72,165,255,24,42,105,0,202,208,249,133,255
25 DATA104,170,232,165,255,24,101,254,133,254,76,111,228,192,4
26 DATA48,219,198,214,165,214,72,162,3,169,32,157,1,4,189
27 DATA212,228,32,210,255,208,12,0,92,72,32,201,255,170,104
28 DATA144,1,138,96,202,16,228,166,254,169,0,32,205,189,169
29 DATA62,32,210,255,104,133,214,32,108,229,169,141,32,210,255
30 DATA76,128,164,9,60,18,19

READY.
```

{INST} SHIFT + klávesa v pravém horním rohu (INST/DEL)
{HOME} druhá klávesa zprava v horní řadě (CLR/HOME)
{RIGHT} klávesa v pravém dolním rohu (CRSR RIGHT/LEFT)
{LEFT} SHIFT + klávesa v pravém dolním rohu (CRSR RIGHT/LEFT)
{BLACK} klávesa CONTROL + 1

Ostatní klávesy barev předpokládají stisk klávesy CONTROL a kláves s číslicemi, resp. klávesy C= dohromady s klávesami s číslicemi.

MSE

MSE slouží k opisování programů ve strojovém kódu. V prvé řadě musíte opsat tzv. "MSE–LOADER". Ten vyrobí na disketě nebo kazetě vlastní program MSE.
Před opisováním musíte bezpodmínečně zadat pár příkazů v přímém módu:
POKE 44,32 : POKE 8192,0 : NEW
Teprve nyní můžete začít s opisováním. Program se opravuje sám. Pokud se přepíšete v řádcích s daty, objeví se na závěr chybové hlášení. Nejčastější chybou bývá zapomenutá čárka mezi dvěma číselnými údaji, může to být i překlep a namísto čárky bývá uvedena tečka. Bezchybnou práci vám bezesporu usnadní použití programu Checksummer. Kontrolní součty najdete v lomených závorkách na konci každého řádku. Listing nemusíte opsat najednou. Práci můžete kdykoli přerušit. Nesmíte však zapomenout vždy před každým novým natažením programu zadat výše uvedené pouky.
Pokud jste vše správně opsali, odstartujte program a pak jej uložte na kazetu či disketu pod názvem MSE 1.1. Hotový MSE 1.1 ládujte podle potřeby jako normální program a startujte jej příkazem RUN.

Pozn. Majitelé datasetů musí v řádku 343 namísto 8 zadat 1 !

PRÁCE S MSE.

Nejprve chce MSE vědět název zpracovávaného programu. Potom musíte zadat startovací a koncovou adresu programu. Všechny požadované údaje najdete v prvém řádku otiskovaných výpisů.
Pokud chcete natáhnout program z kazety nebo diskety, proto abyste jej mohli opravit, resp. abyste dále mohli pokračovat v psaní, zadejte po dotazu na startovací adresu L. Potom musíte stisknout D (disk) nebo T (tape), podle toho, kterou externí paměť použijete. V případě chyby, kdy program zadaného názvu na disketě není, se objeví hlášení I/O ERROR. V takovém případě stiskněte (RUN/STOP RESTORE) a jednoduše zopakujte zadání RUN.
Při opisování postupujte přesně podle zadaných znaků a číslic. Uděláte-li při opisování chybu, MSE ji okamžitě odhalí brumem a hlášením. Po stisku klávesy RETURN můžete klávesou DEL chybu korigovat. Opsaný program uloží MSE automaticky na kazetu či disketu.
U delších listingů je nemyslitelné, abyste je opsali kompletně najednou. Práci můžete kdykoli přerušit a stiskem CTRL + S uložit neúplný program na médium. Nezapomeňte si ovšem poznamenat, na kterém místě jste v opisování skončili. Později zadejte po naládování první části programu CTRL + N. Na následující otázku na startovací adresu uveďte číslo řádku, ve kterém jste skončili opisování. Stisk kláves CTRL + M umožní kdykoliv kontrolu opsaného díla (listing). Stisk SPACE způsobí další prohlížení, RUN/STOP je přerušuje. Majitelé tiskárem mohou nechat listing pomocí CTRL + P vytisknout. Pomocí CTRL + L se dá program znovu natáhnout do počítače.

(JV)

```
100 REM DIESES PROGRAMM ERZEUGT DEN          <210>
110 REM MSE V1.1 AUF DISKETTE.               <030>
120 REM BESITZER EINER DATASETTE             <178>
130 REM MUESSEN DIE '8' AM ENDE VON          <145>
140 REM ZEILE 343 IN EINE '1' AENDERN!       <176>
150 REM                                      <212>
200 IF PEEK(44)<>32 THEN PRINT"{CLR}SIE HA
    BEN VERGESSEN, DIE POKES EINZUGE- BEN!
    ":END                                    <050>
240 PRINT"{CLR)":DIM H(75):FOR I=0 TO 9      <042>
250 H(48+I)=I:H(65+I)=I+10:NEXT:Z=1000       <136>
260 FOR I=2048 TO 3755 STEP 20:PRINT"{HOME
    }ICH LESE ZEILE "Z                       <253>
261 FOR N=0 TO 19:READ A$:IF LEN(A$)<>2 TH
    EN 900                                   <062>
262 IF PEEK(63)+PEEK(64)*256<>Z THEN 600     <011>
270 H=ASC(LEFT$(A$,1)):L=ASC(RIGHT$(A$,1))   <199>
280 D=H(H)*16+H(L):S=S+D:POKE I+N,D           <165>
290 NEXT:READ V:IF S<>V THEN 900             <139>
300 S=0:Z=Z+1:NEXT:R=PEEK(2111):H=PEEK(210
    6)                                       <126>
301 POKE 53280,R:POKE 53281,H:POKE 646,R:P
    RINT"{CLR}DIE DATA-ZEILEN SIND FEHLERF
    REI!"                                    <080>
302 PRINT"SIE KOENNEN NUN DIE FARBEN DES M
    SE"                                      <209>
303 PRINT"EINSTELLEN.":PRINT"{2DOWN,SPACE.
    RVSON}DRUECKEN SIE <1>, <2> ODER <9>     <205>
304 PRINT"{DOWN,2SPACE)<1> - RAHMEN-/SCHRI
    FTFARBE                                  <013>
305 PRINT"{2SPACE)<2> - HINTERGRUNDFARBE     <233>
306 PRINT"{DOWN,2SPACE)<9> - FARBEN UEBERN
    EHMEN                                    <158>
307 PRINT"{2DOWN}FARBE <1>:"R:PRINT"FARBE
    <2>:"H                                   <066>
308 GET A:IF A=0 THEN 308                    <210>
309 IF A=1 THEN R=(R+1)AND 15                <098>
310 IF A=2 THEN H=(H+1)AND 15                <086>
311 IF A=9 THEN 340                          <217>
312 GOTO 301                                 <034>
340 POKE 2106,H:POKE 2111,R                  <153>
342 POKE 631,19:POKE 632,13:POKE 198,2       <135>
343 PRINT"{CLR}SAVE"CHR$(34)"MSE V1.1"CHR$
    (34)",8                                  <091>
344 POKE 43,1:POKE 44,8:POKE 45,172:POKE 4
    6,14:END                                 <140>
600 PRINT"{CLR,RVSON}SIE HABEN ZEILE"Z"{LE
    FT,SPACE}VERGESSEN!":A=PEEK(646)AND 15   <124>
610 POKE 646,PEEK(53281)AND 15:PRINT"LIST"
    Z-2"-"Z+2:POKE 646,A                     <224>
620 GOTO 920                                 <082>
900 PRINT"{CLR,RVSON}SIE HABEN EINEN TIPPF
    EHLER GEMACHT!":A=PEEK(646)AND 15        <154>
910 POKE 646,PEEK(53281)AND 15:PRINT"LIST"
    Z:POKE 646,A                             <173>
920 POKE 631,19:POKE 632,17:POKE 633,13:PO
    KE 198,3:END                             <126>
1000 DATA 00,0B,08,0A,00,9E,32,30,36,31,00
    ,00,00,A2,08,A9,36,85,A4,A9, 1247        <119>
1001 DATA 08,85,A5,A9,00,85,A6,A9,B0,85,A7
    ,A9,00,B1,A4,91,A6,C8,D0,F9, 2886        <054>
1002 DATA E6,A5,E6,A7,CA,D0,F2,A9,36,85,01
    ,4C,00,B0,20,D1,B1,A9,00,8D, 2781        <096>
1003 DATA 21,D0,A9,0F,8D,20,D0,8D,86,02,A9
    ,B3,A9,74,20,FF,B1,A0,B3,A9, 2679        <089>
1004 DATA B9,20,FF,B1,A0,00,20,CF,FF,99,01
    ,02,C8,C9,0D,D0,F5,88,F0,D2, 2912        <217>
1005 DATA C0,11,00,02,A0,10,8C,00,02,20,EA
    ,B1,A0,B3,A9,CF,20,FF,B1,20, 2327        <045>
1006 DATA 8E,B4,85,FC,85,62,20,8E,B4,85,FB
    ,85,61,20,A7,B4,D0,20,A0,B3, 2864        <199>
1007 DATA A9,E5,20,FF,B1,20,8E,B4,85,69,20
```

```
      ,8E,B4,85,5F,20,A7,B4,D0,0A,  2624   <091>
1008 DATA A5,61,C5,5F,A5,62,E5,60,90,06,20
      ,43,B3,4C,3A,B0,A9,AA,A0,00,  2379   <167>
1009 DATA EA,EA,E6,FB,D0,02,E6,FC,20,3F,B2
      ,90,EF,4C,FB,B4,A2,02,86,58,  3190   <041>
1010 DATA A9,A6,A0,9D,20,F2,B1,20,E4,FF,F0
      ,FB,C9,30,90,0C,C9,47,B0,08,  2970   <231>
1011 DATA C9,3A,90,0B,C9,41,B0,07,C9,14,D0
      ,0F,4C,0B,B1,20,D2,FF,A6,58,  2322   <121>
1012 DATA 95,F7,C6,58,D0,D2,60,AE,8D,02,F0
      ,26,C9,0C,D0,03,4C,0B,B6,C9,  2685   <057>
1013 DATA 13,D0,03,4C,8B,B5,C9,0D,D0,03,4C
      ,BA,B4,C9,10,D0,03,4C,68,B5,  2282   <225>
1014 DATA C9,0E,D0,06,20,5F,B4,4C,64,B1,4C
      ,92,B0,A5,F9,20,02,B1,0A,0A,  2132   <208>
1015 DATA 0A,0A,85,F9,A5,F8,20,02,B1,05,F9
      ,60,C9,3A,90,02,69,08,29,0F,  1950   <0^2>
1016 DATA 60,A6,59,E0,08,90,1F,A6,58,E0,02
      ,B0,06,20,D2,FF,4C,8E,B0,C6,  2509   <188>
1017 DATA 59,A0,14,A9,92,20,F2,B1,CA,D0,FA
      ,84,57,68,68,4C,8B,B1,A6,D3,  2891   <197>
1018 DATA E0,08,B0,03,4C,92,B0,20,D2,FF,A6
      ,58,E0,02,90,09,C6,59,20,D2,  2468   <049>
1019 DATA FF,C6,58,D0,F9,4C,8E,B0,48,4A,4A
      ,4A,4A,20,59,B1,68,29,0F,C9,  2419   <035>
1020 DATA 0A,90,02,69,06,69,30,4C,D2,FF,A2
      ,FC,9A,20,D1,B1,20,48,B2,20,  2261   <073>
1021 DATA EA,B1,20,9F,B2,A5,FC,20,4E,B1,A5
      ,FB,20,4E,B1,20,ED,B1,A9,3A,  2860   <148>
1022 DATA A0,20,20,F2,B1,A9,00,85,59,20,8E
      ,B0,20,ED,B1,A4,59,20,EF,B0,  2530   <233>
1023 DATA 91,FB,C8,84,59,C0,08,90,EC,20,10
      ,B2,A9,12,20,D2,FF,20,8E,B0,  2657   <105>
1024 DATA 20,EF,B0,C5,FF,F0,0D,20,43,B3,A9
      ,14,A0,14,20,F2,B1,4C,A2,B1,  2665   <034>
1025 DATA A9,92,20,D2,FF,20,33,B2,20,E0,B2
      ,20,3F,B2,90,9F,4C,8B,B5,A9,  2648   <123>
1026 DATA 93,20,D2,FF,A2,00,A9,03,9D,00,D8
      ,9D,00,D9,9D,00,DA,9D,00,DB,  2476   <237>
1027 DATA E8,D0,EF,60,A9,0D,2C,A9,20,4C,D2
      ,FF,20,D2,FF,98,4C,D2,FF,20,  2965   <160>
1028 DATA E4,FF,F0,FB,60,84,5D,85,5C,A0,00
      ,B1,5C,F0,06,20,D2,FF,C8,D0,  3100   <077>
1029 DATA F6,60,A5,FB,85,5A,A0,00,84,5B,B1
      ,FB,18,65,5A,85,5A,90,02,E6,  2606   <156>
1030 DATA 5B,06,5A,26,5B,C8,C0,08,90,EC,A5
      ,5A,65,5B,85,FF,60,18,A5,FB,  2467   <219>
1031 DATA 69,08,85,FB,90,02,E6,FC,60,A5,FB
      ,C5,5F,A5,FC,E5,60,60,A0,B3,  3106   <183>
1032 DATA A9,FB,20,FF,B1,A0,01,B9,00,02,20
      ,D2,FF,CC,00,02,C8,90,F4,A9,  2692   <098>
1033 DATA 14,ED,00,02,AA,20,ED,B1,CA,D0,FA
      ,A5,62,20,4E,B1,A5,61,20,4E,  2457   <060>
1034 DATA B1,20,ED,B1,A5,60,20,4E,B1,A5,5F
      ,20,4E,B1,EA,EA,EA,EA,EA,EA,  3122   <190>
1035 DATA EA,EA,24,5E,10,01,60,A9,12,20,D2
      ,FF,A2,28,20,ED,B1,CA,D0,FA,  2703   <087>
1036 DATA A9,92,4C,D2,FF,A5,D6,C9,16,B0,01
      ,60,A9,A0,85,A4,A9,78,85,A6,  2945   <204>
1037 DATA A9,04,85,A5,85,A7,A2,13,A0,27,B1
      ,A4,91,A6,88,10,F9,CA,F0,19,  2671   <208>
1038 DATA 18,A5,A4,69,28,85,A4,90,02,E6,A5
      ,18,A5,A6,69,28,85,A6,90,E0,  2503   <251>
1039 DATA E6,A7,4C,B6,B2,A9,91,4C,D2,FF,A9
      ,0F,8D,18,D4,A9,00,8D,05,D4,  2776   <000>
1040 DATA A9,F7,8D,06,D4,A9,11,8D,04,D4,A9
      ,32,8D,01,D4,A9,00,8D,00,D4,  2413   <126>
1041 DATA A0,80,20,09,B3,A9,10,8D,04,D4,60
      ,A2,FF,CA,D0,FD,88,D0,F8,60,  2914   <240>
1042 DATA A9,0F,8D,18,D4,A9,2D,8D,05,D4,A9
      ,A5,8D,06,D4,A9,21,8D,04,D4,  2385   <119>
1043 DATA A9,07,8D,01,D4,A9,05,8D,00,D4,A0
      ,FF,20,09,B3,A9,20,8D,04,D4,  2250   <078>
1044 DATA A9,00,8D,01,D4,8D,00,D4,60,38,20
      ,F0,FF,8A,48,98,48,18,A0,06,  2179   <175>
1045 DATA A2,18,20,F0,FF,A0,B4,A9,0A,20,FF
      ,B1,20,12,B3,20,E4,FF,F0,FB,  2931   <093>
1046 DATA A2,1D,A9,14,20,D2,FF,CA,D0,FA,68
      ,A8,68,AA,18,4C,F0,FF,0D,0D,  2704   <088>
1047 DATA 0D,20,20,20,20,20,20,20,4D,41,53
      ,43,48,49,4E,45,4E,53,50,52,  1144   <216>
1048 DATA 41,43,48,45,20,2D,20,45,44,49,54
      ,4F,52,20,0D,0D,20,20,20,20,  1023   <038>
1049 DATA 20,20,20,20,56,4F,4E,20,4E,2E,4D
      ,41,4E,4E,20,26,20,44,2E,57,  1128   <206>
1050 DATA 45,49,4E,45,43,4B,00,0D,0D,0D,20
      ,20,20,50,52,4F,47,52,41,4D,  1102   <117>
1051 DATA 4D,4E,41,4D,45,20,3A,20,00,0D,0D
      ,20,20,20,53,54,41,52,54,41,  1073   <095>
1052 DATA 44,52,45,53,53,45,20,3A,20,24,00
      ,0D,0D,20,20,20,45,4E,44,41,  1014   <129>
1053 DATA 44,52,45,53,53,45,20,20,20,3A,20
      ,24,00,92,01,01,50,52,4F,47,  1136   <228>
1054 DATA 52,41,4D,4D,20,3A,20,00,12,20,20
      ,2A,2A,2A,20,46,41,4C,53,43,  1024   <027>
1055 DATA 48,45,20,45,49,4E,47,41,42,45,20
      ,2A,2A,2A,20,20,92,00,0D,0D,  1058   <098>
1056 DATA 2A,2A,2A,20,45,4E,44,45,20,2A,2A
      ,2A,00,13,01,20,20,12,44,92,  916    <153>
1057 DATA 49,53,4B,20,4F,44,45,52,20,12,54
      ,92,41,50,45,0D,00,13,20,20,  1151   <035>
1058 DATA 49,2F,4F,20,2D,20,46,45,48,4C,45
      ,52,00,20,D1,B1,20,48,B2,A0,  1606   <012>
1059 DATA B3,A9,CF,20,FF,B1,20,8E,B4,85,FC
      ,20,8E,B4,85,FB,C5,61,A5,FC,  3207   <251>
1060 DATA E5,62,90,23,A5,FB,C5,5F,A5,FC,E5
      ,60,B0,19,20,A7,B4,D0,14,60,  2860   <112>
1061 DATA 20,A7,B4,F0,0C,85,F9,20,A7,B4,F0
      ,05,85,F8,4C,EF,B0,68,68,20,  2749   <086>
1062 DATA 43,B3,4C,5F,B4,20,CF,FF,C9,4C,D0
      ,09,20,D1,B1,20,48,B2,4C,0B,  2372   <046>
1063 DATA B6,C9,0D,60,A9,00,85,5E,20,5F,B4
      ,20,EA,B1,20,0D,B5,24,5E,30,  2042   <120>
1064 DATA 05,20,E4,FF,F0,FB,20,E1,FF,F0,26
      ,20,9F,B2,24,5E,10,09,20,4E,  2405   <198>
1065 DATA B5,20,0D,B5,20,60,B5,20,33,B2,20
      ,3F,B2,90,D7,A0,B4,A9,26,20,  2190   <207>
1066 DATA FF,B1,20,E4,FF,C9,0D,D0,F9,A9,00
      ,85,5E,A5,61,85,FB,A5,62,85,  3056   <240>
1067 DATA FC,20,E0,B2,4C,64,B1,A5,FC,20,4E
      ,B1,A5,FB,85,FF,20,4E,B1,A9,  3003   <221>
1068 DATA 20,A0,3A,20,F2,B1,A0,00,20,ED,B1
      ,B1,FB,20,4E,B1,C8,C0,08,90,  2566   <070>
1069 DATA F3,20,ED,B1,24,5E,30,03,A9,12,2C
      ,A9,20,20,D2,FF,20,10,B2,A5,  2190   <059>
1070 DATA FF,20,4E,B1,A9,92,20,D2,FF,4C,EA
      ,B1,A9,FF,65,B8,85,B9,A9,04,  3073   <029>
1071 DATA 85,BA,20,C0,FF,A2,FF,4C,C9,FF,20
      ,CC,FF,A9,FF,4C,C3,FF,20,5F,  3315   <189>
1072 DATA B4,A9,60,85,5E,20,4E,B5,20,48,B2
      ,A2,24,A9,2D,20,D2,FF,CA,D0,  2596   <111>
1073 DATA FA,20,EA,B1,20,EA,B1,20,60,B5,4C
      ,C1,B4,20,B8,B5,A6,5F,A4,60,  2812   <015>
1074 DATA A9,61,20,D8,FF,B0,0A,20,B7,FF,29
      ,BF,D0,03,4C,FB,B4,A9,01,20,  2577   <201>
1075 DATA C3,FF,20,68,B6,A0,B4,A9,4F,20,FF
      ,B1,20,F9,B1,4C,FB,B4,20,68,  2921   <237>
1076 DATA B6,A9,37,A0,B4,20,FF,B1,20,F9,B1
      ,A2,08,C9,44,F0,06,A2,01,C9,  2717   <213>
1077 DATA 54,D0,F1,A9,01,A8,20,BA,FF,A0,00
      ,E0,01,F0,1A,A9,40,8D,20,02,  2403   <101>
1078 DATA A9,3A,8D,21,02,B9,01,02,99,22,02
      ,C8,CC,00,02,90,F4,C8,C8,D0,  2162   <127>
1079 DATA 0C,B9,01,02,99,20,02,C8,CC,00,02
      ,D0,F4,98,A2,20,A0,02,4C,BD,  2018   <025>
1080 DATA FF,20,B8,B5,A5,BA,C9,08,90,33,A6
      ,B9,86,57,A9,01,20,C3,FF,A9,  2800   <022>
1081 DATA 60,85,B9,20,C0,FF,B0,28,A5,BA,20
      ,B4,FF,A5,B9,20,96,FF,20,A5,  2911   <053>
1082 DATA FF,85,61,A5,90,4A,4A,B0,13,20,A5
      ,FF,85,62,20,AB,FF,A5,57,85,  2663   <214>
1083 DATA B9,A9,00,20,D5,FF,90,03,4C,A3,B5
      ,86,5F,84,60,A5,BA,C9,01,D0,  2639   <131>
1084 DATA 0A,AD,3D,03,85,61,AD,3E,03,85,62
      ,4C,FB,B4,A9,13,20,D2,FF,A2,  2300   <120>
1085 DATA 1C,20,ED,B1,CA,D0,FA,60,00,00,00
      ,00,00,00,00,00,00,00,00,00,  1230   <143>
```

© 64'er **Listing 2. Der MSE-Lader**

SUPER EPROMER

Měli jsme možnost prakticky odzkoušet a porovnat epromery Phantom X (REX), ALCOMP a Tiny Epromer (Conrad). Z testu vyšel jednoznačně jako vítěz Tiny Epromer. Vypaluje nejběžnější typy pamětí 2764, 27128 a 27256 s programovacím napětím 12,5 V a 21 V. V nejrychlejším módu vypaluje paměti s kapacitou 8kB za 5 s.

Super epromer vychází z Tiny Epromeru. Je možno jej použít pro vypalování pamětí 2732, 2764, 27128, 27256, 27512 a jejich pinově kompatibilních CMOS verzí. Jeho silnou stránkou je především vysoká rychlost vypalování dat. Ovládací software umožňuje volit ze tří algoritmů programování. Nejrychlejší algoritmus shodně s Tiny Epromerem vypálí 8kB za 5 s. V závislosti na počtu prázdných bajtů se dá tento čas ještě zkrátit. Vysokou rychlost vypalování oceníte zvláště při zpracovávání pamětí 27512 s kapacitou 64 kB. Dodávané programové vybavení obsahuje i rutiny ke čtení již vypálených pamětí eprom, verifikaci dat, kontrolu vymazání paměti a také modulgenerátor, pomocí nějž se dá sestavit jednoduchý 8kB eprom modul s jedním nebo více programy ve strojovém kódu nebo Basicu.

Vypalovací algoritmy

1. Turbo Speed.

Tímto algoritmem může být 8 kB paměť vypálena za cca 5 s. V případě, že se mezi daty vyskytne větší počet prázdných bajtů, je vypalovací rychlost ještě vyšší. Tato hodnota řadí Super epromer mezi nejrychlejší epromery provozované na C64 vůbec. Každý bajt je vypalován nejméně 0,1 ms, přičemž první impuls trvá 0,05 ms. Podle potřeby je však možno prodloužit délku vypalovacího cyklu až na 50 ms.

Vedle časové úspory má tento algoritmus ještě jednu výhodu. Šetří životnost pamětí. Protože paměť není při vypalování dlouho elektricky (tepelně) namáhána, prodlouží se počet cyklů, kdy může být opakovaně mázána a programována.

Nevýhodou je snížená stálost zapsaných dat. Proto by měl tento algoritmus být využíván tehdy, když je zápis možno po několika měsících obnovit, nebo pro zkušební vypalování při vývoji kdy se počítá s opakovaným mazáním pamětí. Jak dlouho fakticky data v paměti vydrží lze jen těžko odhadnout, neboť vliv má použitá technologie výroby pamětí a výrobní tolerance.

2. Fast & Secure.

Tento algoritmus je rovněž velmi rychlý. Vypálí 8kB paměti za méně než 11 s. Vyznačuje se však podstatně vyšší stálostí uložených dat. Eprom je vypalována vždy pětinásobkem nutného času, přičemž jeden impuls je dlouhý 0,15 ms. Pokud byl bajt správně zapsán, je jeho stálost pojištěna následným vypalovacím impulsem čtyřnásobné délky. V případě, že bajt není správně vypálen, nastaví program delší impuls a postup opakuje. Délka impulsu může u zvlášť „odolných" pamětí dosáhnout až předepsaných 50 ms. Tento algoritmus se doporučuje pro běžné použití.

3. Standard 50 ms.

Je nejpomalejším algoritmem (8kB za 7 min), zároveň však, pokud se týče stálosti dat, nejjistějším. Všechny bajty jsou vypalovány tak, jak je předepsáno výrobcem, t.j. 50 ms. Použití algoritmu je vázáno na požadavky na extrémní stálost dat. Mimochodem, některé starší typy pamětí se ani jinak vypalovat nedají.

Pro všechny typy pamětí platí, že bajt do kterého má být uložena hodnota FF, se nevypaluje. Jedná se totiž o vymazaný bajt. Proto dochází u eprom s velkým podílem těchto bajtů k dalšímu zkracování doby vypalování. Všechny bajty se verifikují, což vede k okamžitému odhalení a ohlášení chyb. Epromer umožňuje samozřejmě i novou kontrolu dat zapsaných do paměti (Verify). Tato funkce se hodí aplikovat už v jednou uvedeném příkladu, když si nejsme jisti obsahem pamětí. Pokud se při verifikaci zjistí chyba, je možno učinit pokus o její odstranění opakovaným vypálením. Během vypalování pamětí se mohou vyskytovat dva druhy chyb. Buď bit nemůže být vypálen, pokus lze zopakovat, ev. se dá použít delší vypalovací algoritmus, nebo je nastaven špatný bit. V takovém případě je nutno eprom znovu vymazat,tedy nastavit všechny bity do výchozího stavu $FF.

Při každém kontaktu s eprom (čtení, verifikace, vypalování) se na obrazovce objeví červený rámeček. Během vypalování některých pamětí svítí červená dioda LED, umístěná vedle patice pro eprom. V této době raději nemanipulujte s eprom. Mohlo by to vést k jejímu zničení.

Vypalování pamětí

Po připojení epromeru k počítači se natáhne a odstartuje program. Automaticky se dotáhne strojová rutina a objeví se hlavní menu. Funkčními klávesami F1 až F8 můžete vybrat mezi různými funkcemi:

"F1" = Vypalování pamětí
"F2" = Test, zda eprom je prázdná
"F3" = Volba typu paměti a vypalovacího algoritmu

"F4" = Čtení obsahu paměti
"F5" = Datové soubory – directory
"F6" = Verifikace eprom
"F7" = Zpět do Basicu
"F8" = Tools

V prvém kroku se klávesou F3 zvolí typ paměti a jeden ze tří vypalovacích algoritmů. Kurzorovými klávesami najedete na požadovaný typ paměti, s příslušným vypalovacím napětím a volbu potvrdíte klávesou Return. Shodně, kurzorovými klávesami, vyberete algoritmus vypalování. Pro skok zpět do hlavního menu použijete klávesu F1.

Další postup záleží na tom v jaké formě máte připravena data k vypalování. Nejobvyklejší je disketa s datovým souborem nebo eprom jejíž kopii je žádoucí vytvořit. Při vytváření kopie eprom se do patice epromeru vloží správně orientovaný čip a zvolí se funkce F4. Objeví se další menu, s funkcemi:

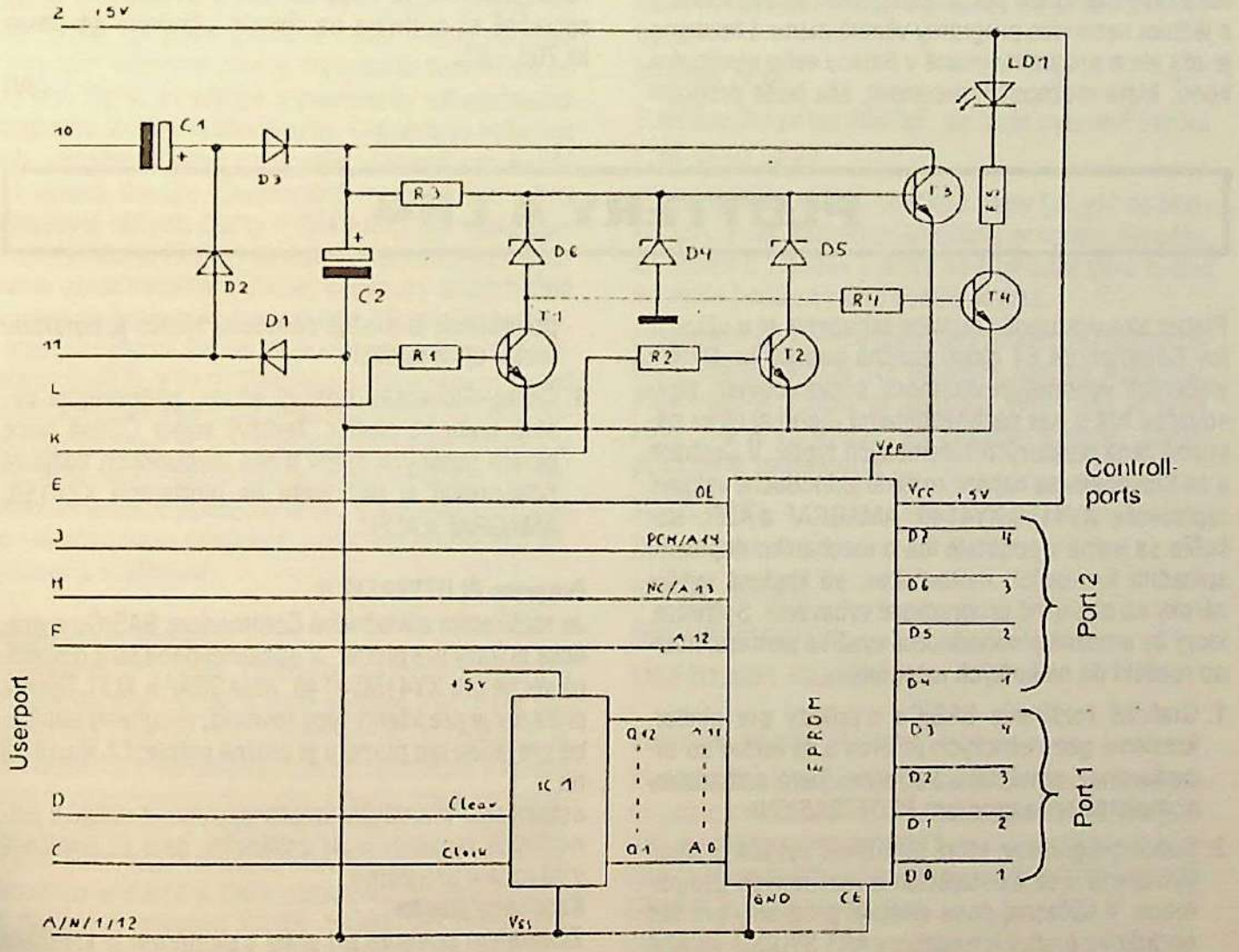
"F3" = obsah eprom načíst do paměti počítače
"F5" = číst jednotlivé bajty
"F1" = zpět do hlavního menu

Po volbě F3 se počítač zeptá na počáteční adresu a po jejím zadání navrhne koncovou adresu. Obě adresy, počáteční i koncová, se dají samozřejmě změnit. Nezapomeňte, že adresy se vztahují k eprom. Pokud chcete číst paměť od začátku, zadáte $0000. Po zadání počáteční adresy paměti počítače, do které se mají ukládat vyčítaná data z eprom a koncové adresy, ještě zbývá potvrdit kontrolní otázku, že jste si skutečně jisti svým počínáním a obsah paměti přečíst. Klávesou F1 se vrátíte zpět do hlavního menu. Nyní se již může konečně vypálit vlastní eprom.

Zvolíte F1 pro vypalování a následně F3 (vypálení obsahu paměti) nebo F5 (vypalování jednotlivých bajtů). Následuje opět určení počátečních a koncových adres eprom a dané oblasti paměti počítače, ze které se data berou.

Pro všechny případy si po ukončeném vypalování pomocí F6 proveďte kontrolu, zda je paměť skutečně dobře vypálena a daný datový soubor uložte na disketu.

Ukládání datových souborů na disketu i jejich zpětné natahování se volí z menu pod klávesou F5. Menu zajišťuje následující funkce:



"F1" = zpět do hlavního menu
"F2" = příkaz floppy (viz manuál VC 1541)
"F3" = directory
"F4" = poruchový kanál
"F5" = natažení programu
"F7" = uložení programu

Stisknete tedy F7, zapíšete název datového souboru, číslo přístroje (paměťového média) a počáteční a koncovou adresu oblasti paměti, v níž leží data. Po nezbytné kontrolní otázce, zda jste si jisti, se data uloží na disketu a opět se vrátíte do shodného menu. Stiskem F3 se dá zkontrolovat, zda se datový soubor skutečně uložil. Známým způsobem přejdete zpět do hlavního menu a pokračujete v práci.

MODULGENERÁTOR

Z hlavního menu je možno volbou F8 přejít do menu Tools a klávesou F3 potom zvolit funkci modulgenerátor. Program Modulgenerátor je na programové disketě k Super epromeru, proto je nutno tuto disketu mít vloženu v drajvu.

Modulgenerátor slouží k tvorbě programových modulů 8kB délky, které lze umístit do cartridge, připojitelné do expanzního portu. Zhotovovat se dají moduly s jedním nebo více programy včetně menu. Lhostejno je zda jde o programy psané v Basicu nebo strojovém kódu. Máte možnost rozhodnout, zda bude program v modulu startován jako basicový (RUN) nebo jako strojový (JSR,JMP). Počáteční adresa se automaticky přebírá z diskety. Programy v Basicu se startují vždy od adresy $0801.

Hotový modul zpracovaný modulgenerátorem je uložen v paměti počítače od adresy $8000. Příkazem v přímém módu SYS 64738 se dá odstartovat a otestovat. Z praktických zkušeností víme, že ne všechny programy se dají tímto modulgenerátorem zpracovat. Pokud je vše v pořádku, uložte modul jako datový soubor na disketu.

Zvláštnosti

- software Super epromeru spolupracuje s FC II i FC III.
- stisk RUN/STOP způsobí vždy skok do hlavního menu. Tato funkce je důležitá neboť některé podprogramy nelze jinak přerušit (vypalování nebo čtení).
- pro paměti 2732 a paměti 27512 se volí v menu typů pamětí 27256. Vlastní typ paměti se potom určí polohou kódových přepínačů na epromeru.

Hotový přístroj je v nabídce firmy Comotronic s.r.o. společně se software na disketě a českým návodem za 795,– Kč.

(JV)

PLOTTERY A C64

Plotter ako výstupné kresliace zariadenie si u užívateľov Commodore 64 získal značnú popularitu. Okrem viacerých výhodných vlastností súradnicových zapisovačov hrá u nás nezanedbateľnú úlohu aj nízka nákupná cena niektorých tuzemských typov. V Čechách a na Slovensku sa najviac rozšírili jednoduché valcové zapisovače XY4150/XY4140, AMAGRAF a ALFI. Nakoľko sa jedná v podstate iba o mechaniku doplnenú spínačmi krokových motorčekov, sú kladené vyššie nároky na obslužné programové vybavenie. Software, ktorý by umožnil plnohodnotné využitie plotteru, možno rozdeliť do niekoľkých kategórií:

1. Grafické rozšírenie BASICu o príkazy pre plotter, kreslenie geometrických útvarov a čs textov so zabudovanou emuláciou tlačiarne. Tieto požiadavky kompletne spňa program PLOTTBASIC II.
2. Súbor programov, ktoré umožňujú vytlačiť grafiku, vytvorenú v už existujúcich a rozšírených programoch. V súčasnej dobe existujú programy pre tlač obrázkov z grafických editorov ART STUDIO, KOALA PAINTER, STAR PAINTER a VIDCOM64. Za určitých podmienok je možné pomocou týchto programov kresliť aj vo farbách.
3. Česko–Slovenský textový editor, podporujúci výstup textu na plotter. Textový editor ČStext bude okrem mnohých typov u nás dostupných tlačiarní podporovať aj tlač textu na plotteroch XY4150, AMAGRAF a ALFI.

Program PLOTTBASIC II

Je rozšírením základného Commodore BASICu o grafické príkazy pre plotter. V súčasnej dobe sú k dispozícii verzie pre XY4150/4140, AMAGRAF a ALFI. Syntax príkazov je pre všetky typy rovnaká, programy napísané pre jeden typ plotteru je možné preniesť a spustiť aj na
ostatných. Pri zadávaní kresliacej plochy existujú u jednotlivých zariadení malé odlišnosti, tieto sú popísané v návode k programu.

Kresliaca plocha

Základným pojmom pri práci s plotterom je kresliaca plocha. Na nej sa odohráva všetko kreslenie, všetky

presuny písacej hlavy. Je dôležité, aby pri práci s programovým vybavením bolo možné pohybovať perom len v určitej, vopred nadefinovanej oblasti. Grafická jednotka XY4150 má zabudované spínače ľavého a pravého dorazu, preto pri vybehnutí z povolenej plochy nehrozí poškodenie zariadenia. Inak je to u plotterov AMAGRAF a ALFI, ktoré spínače dorazu nemajú. Preto je treba, aby pri chybnom užívateľskom programe základné vybavenie vždy ustrážilo vopred nadefinovanú plochu. Veľkosť plochy pre kreslenie zadáva užívateľ pri začatí práce s programom príkazom PAPER. Obdžnik, v ktorom je povolené kreslenie je určený ľavým horným a pravým dolným rohom. Kresliacu plochu je treba znovu zadať pri zmene formátu použitého papiera. Ak je treba len vymeniť papier, stačí zadať príkaz NP, ktorý zabezpečí vysunutie papiera a zachovanie polohy kresliacej plochy vzhľadom na nový papier.

Definícia okna, orezávanie

Program PLOTTBASIC II má zabudované plnohodnotné orezávanie kresliacej plochy. V praxi to znamená asi toľko, že každá úsečka, alebo aj zložitý útvar budú počas kreslenia podrobené výpočtu, ktorý zistí, či nejaká časť útvaru neleží mimo definovanú plochu. Ak áno, budú vyrátané spoločné body medzi útvarom a stranami kresliacej plochy. Nakreslí sa teda len to, čo má byť vidieť a nedôjde k prerušeniu užívateľského programu ani k posunu kresby. Orezávanie vyžaduje veľa výpočtov, ktoré sa musia vykonať pre každú vykreslenú úsečku. Štandardné matematické rutiny pohyblivej rádovej čiarky v C64 sú už pre tento účel trochu pomalé. Preto sme do programu vyvinuli špeciálne goniometrické funkcie, pomocou ktorých C64 orezávanie bez problémov zvládne. Základným oknom, v ktorom funguje kreslenie, je okno nadefinované príkazom PAPER. V PLOTTBASICu existuje ešte možnosť nadefinovať menšie okno príkazom WSET. V takto definovanom okne je tiež povolené kreslenie, grafiku je možné obmedziť len na určitú oblasť kresliacej plochy. Okno definované príkazom WSET podlieha operáciám so súradnicovou sústavou, preto ho možno otáčať, posúvať a zväčšovať.

Súradnicové sústavy

Program PLOTTBASIC II ponúka možnosť programovať grafiku pre plotter v karteziánskej a polárnej súradnicovej sústave. Po inicializácii PLOTTBASICu je nastavená karteziánska sústava so začiatkom v ľavom dolnom rohu kresliacej plochy. Písacia hlava sa pohybuje v smere osi x, papier v smere osi y. Príkazom XYSET nadefinujeme posunutie, zväčšenie a otočenie súradnicovej sústavy. Osi x,y aktuálnej sústavy je možné zviditeľniť príkazmi XAXIS, YAXIS. Pri niektorých aplikáciách je vhodnejšie použiť polárnu sústavu súradníc. Prepínanie medzi sústavami zaisťuje príkaz POL.

Čiarová grafika

Základnou jednotkou čiarovej grafiky je úsečka. Z úsečiek sa skladajú zložitejšie útvary a znaky. Pri vytváraní užívateľských programov pre plotter potrebujeme niekoľko základných spôsobov posunu a kreslenia úsečky. Absolútny posun so zdvihnutým alebo spusteným perom, relatívny posun a kreslenie nezávislej úsečky. Program PLOTTBASIC II poskytuje tieto príkazy:

MOVE x,y

Príkaz presunie pero plotteru z okamžitej polohy do bodu so súradnicami (x,y). Pero je po celý čas presunu zdvihnuté, úsečka sa nenakreslí.

DRAW x,y

Vykreslenie úsečky z okamžitej polohy pera do bodu so súradnicami (x,y). Pero sa na začiatku presunu spustí a na konci zdvihne.

RMOVE dx,dy

Relatívny presun bez kreslenia z okamžitej polohy (x,y) do bodu (x+dx,y+dy). Hodnoty dx,dy môžu byť kladné, nulové aj záporné. Ak sa niektorý z parametrov dx,dy rovná nule, posun v príslušnom smere sa nevykoná.

RDRAW dx,dy

Podobne ako príkaz RMOVE, ale bude sa kresliť úsečka.

LINE x1,y1,x2,y2

Vykreslenie nezávislej úsečky z bodu (x1,y1) do bodu (x2,y2). Pred začiatkom kreslenia program vypočíta, ku ktorému z bodov úsečky sa nachádza pero bližšie a z tohto bodu sa začne úsečka kresliť.

Prepnutím súradnicovej sústavy príkazom POL na polárnu získajú príkazy pre kreslenie úsečiek nový tvar. Parameter R je dĺžka sprievodiča bodu, alfa je uhol, ktorý zviera sprievodič so základnou osou. Syntax príkazov je nasledovná:

MOVE R,alfa

DRAW R,alfa

RMOVE dR,dalfa

RDRAW dR,dalfa

LINE R1,alfa1,R2,alfa2

Kružnice, elipsy, oblúky ...

PLOTBASIC II podporuje kreslenie oblúkov komfortnými príkazmi. Na plynulé vykresľovanie kriviek sa využívajú špeciálne goniometrické funkcie, multitasking a vyrovnávacia pamäť. Príkazy pre oblúky pracujú v karteziánskej aj polárnej sústave.

CIRCLE r,x,y
Nakreslí kružnicu s polomerom r a stredom v bode

o súradniciach (x,y).
ELLIPSE a,b,x,y

Vykreslenie elipsy, ktorej hlavná poloos má dĺžku a, vedľajšia poloos má dĺžku b a stred sa nachádza v bode (x,y).

ARC beta1,beta2,a,b,x,y

Príkaz nakreslí oblúk elipsy od uhlu beta1 po uhol beta2. Oblúk pochádza z elipsy s parametrami a,b,x,y.

ANGL x,y,a,b,beta

Vykreslenie sprievodiča elipsy s parametrami x,y,a,b, prislúchajúceho uhlu beta.

NPOLY n,a,b,x,y

Vytvorí n–uholník vpísaný do elipsy s parametrami a,b,x,y.

V polárnej sústave:

CIRCLE r,R,alfa
ELLIPSE a,b,R,alfa
ARC beta1,beta2,a,b,R,alfa
ANGL R,alfa,a,b,beta
NPOLY n,a,b,R,alfa

Texty v grafike

Okrem základných geometrických obrazcov je potrebné kresliť aj texty. PLOTTBASIC podporuje kreslenie textov rôznymi znakovými sadami s využitím kompletnej českej a slovenskej diakritiky. Parametre textu komplexne pokrývajú príkazy TSIZE a TSMER. Príkazom TSIZE definujeme šírku a výšku znakov, medzeru medzi znakmi, medzeru medzi riadkami, počet obťahov znaku, diferenciu medzi obťahmi a proporcionalitu písma. Príkaz TSMER umožňuje nastaviť smer kreslenia textu a uhol sklopenia znakov. Samotné vykreslenie textu sa realizuje príkazom TEXT s udaním pozície ľavého dolného rohu textu. Text sa zadáva v úvodzovkách, alebo môže byť aj obsahom reťazcovej premennej. Príkaz TEXT pracuje v karteziánskej aj polárnej sústave. V polárnej sústave je možné pomocou tohto príkazu s minimálnym programátorským úsilím dosahovať zaujímavé efekty. Pri kreslení českého a slovenského textu program sám rozpoznáva interpunkciu pred veľkým a malým písmenom, ako aj nezmyselnú interpunkciu. Umiestnenie interpunkcie je pre každé české a slovenské písmeno individuálne. Znakové sady sú riešené ako dynamické údajové štruktúry. Dĺžka znakovej sady je úmerná zložitosti znakov, v pamäti sa môže naraz nachádzať viac znakových súborov, najviac však osem. Kapacita vyhradená pre znakové sady je 12kB. Znakovú sadu nahráme z diskety príkazom LFONT. Program prehľadá automaticky všetky mechaniky. Pokiaľ sa hľadaná znaková sada nenájde, program pokračuje v kreslení naposledy nastavenou znakovou sadou. Ak sa sada nájde, ale už sa nezmestí celá do pamäti, príde k vymazaniu najkratšej znakovej sady, ktorá uvoľní dostatočný pamäťový priestor pre donahrávanie požadovanej sady. Príkaz LFONT je možné použiť v dialógovom režime, ale aj kdekoľvek v užívateľskom programe. Užívateľský BASIC–program preto môže počas svojho behu dynamicky meniť znakové sady. Ak sa vyskytne pri požiadavke o nový font chyba, nedôjde k prerušeniu behu aplikačného programu, ale sa pokračuje v kreslení predtým definovanou sadou. K programu sa dodáva niekoľko hotových fontov. Okrem toho má užívateľ možnosť dokúpiť si ďalšie fonty, alebo si ich vytvoriť programom FONT EDITOR. Znakové sady z programu PLOTTBASIC II používa aj textový editor ČStext v prevedení pre plottre.

Plotter ako tlačiareň

Na tlačenie veľkého množstva textu sa plotter v zásade nehodí, napriek tomu niektorí užívatelia majú záujem o emuláciu tlačiarne na plotteri. V amatérskych podmienkach je možné plotter využiť aj týmto spôsobom, najmä na výpisy listingov, directory a kratších textov. PLOTTBASIC II poskytuje kompletnú emuláciu tlačiarne pod číslami zariadenia 4 a 5. Pod akým číslom zariadenia bude plotter–tlačiareň komunikovať, nastavíme príkazom DEVC. Pre tlačiareň je potrebné štandardne otvoriť komunikačný kanál príkazom OPEN. Programy v BASICu, písané pre tlačiareň budú za určitých podmienok normálne pracovať. Pre tlačenie textu je možné zvoliť si znakovú sadu, ako aj smer písania a parametre znakov. Počet znakov na riadok a počet riadkov na stranu sa nastavuje príkazom TPAGE. Po vypísaní textovej stránky program vysunie papier a čaká na stlačenie klávesy. Na rozdiel od tlačiarne umožňuje plotter písanie textovej stránky ľubovoľným smerom. Odsek môže byť natočený, dajú sa vytvárať rôzne efekty, napríklad zrkadlové písmo.

Multitasking

Základné rutiny pre ovládanie mechaník plotterov sú v programe PLOTTBASIC II riešené moderným a zaujímavým spôsobom. Zistilo sa, že pri ovládaní pomalých mechaník sa veľa času spotrebuje na čakanie procesoru medzi jednotlivými krokmi zapisovača. Ak dáme vykresliť úsečku, procesor v určitých intervaloch dodáva mechanike pokyny o stave cievok v krokových motoroch a o stave písacej hlavy. Medzi krokmi zapisovača je ale dosť času na to, aby výpočty predbiehali kreslenie. Preto sme zaviedli krokovanie prerušením a definovali sme vyrovnávaciu pamäť o veľkosti 8000 krokov zapisovača. Všetky pohyby plotteru sa vykonávajú v pozadí, využíva sa takzvaný multitasking. Užívateľský program a všetky výpočty prebiehajú v popredí a sú len prerušované obslužným behom pre

krokovanie plotteru a vyprázdňovanie vyrovnávacej pamäti. Vyrovnávacia pamäť je realizovaná ako spoločný buffer pre obidva behy, ktorých ukazatele sa v bufferi neustále predbiehajú. Vlastné zablokovanie výpočtov a čakanie užívateľskej aplikácie môže nastať až vtedy, keď ukazovateľ výpočtov v bufferi dobehne ukazovateľ krokovania. V praxi sa to môže stať len pri jednoduchých posunoch, kde výpočet prebieha veľmi rýchlo. Ak sa potom napríklad kreslí krivka náročná na výpočty, program má v bufferi kus „nadbehnutý" a pomer sa vyrovnáva. Pri kreslení kriviek pomocou príkazov obsiahnutých v PLOTTBASICu sa jedná o výpočty pomocou špeciálnych rutín. Výpočty sú dosť rýchle na to, aby sa krivky vykresľovali plynule. Inak je to pri programovaní užívateľských kriviek v BASICu. Výpočty nie sú dosť rýchle na plynulé kreslenie. Preto sme zaviedli do bufferu zarážku. Zarážka sa aktivuje príkazom HALT1. Spôsobí to, že všetky príkazy pre plotter sa nevykonávajú hneď, ale sa iba ukladajú do bufferu pre 8000 krokov. Po zadaní príkazu HALT0 sa zarážka uvoľní a krivka sa plynule vykreslí. Ak sa krivka skladá z viac ako 8000 krokov, buffer sa automaticky vyprázdni, vykreslí sa plynule časť krivky a opäť sa automaticky nastaví zarážka.
Pri používaní multitaskingu sa plotter navonok chová, ako keby mal vlastnú inteligenciu. Po zadaní grafického príkazu alebo textu je počítač „skôr hotový" ako plotter, kurzor už bliká a plotter ešte kreslí. Výhodou krokovania prerušením je aj to, že počas vykonávania grafických príkazov nebliká obrazovka ako u niektorých programov, ktoré využívajú na časovanie strojové cykly procesoru.

Záver

Cieľom tohto článku je zoznámiť užívateľov C64 o možnostiach využitia plotterov a ich programového vybavenia. Programy, spomínané v tomto článku je možné si objednať u firmy Commotronic. Prípadní záujemcovia o uvedenú problematiku sa môžu obrátiť priamo na adresu:

Branislav BILLA
Za poštou 9/A
Hlohovec
920 01

Pozn. redakce:

Tento příspěvek byl do FUNu zařazen mimo pořadí proto, že software, připravované panem Billou pro uživatele C64, kteří vlastní nebo mají možnost získat levné plotry a tiskárny považujeme dnes za velice aktuální.
Seriál článků o software pana Billy bude pokračovat v dalších číslech FUNu popisem ČS textu, souboru programů pro tisk grafiky a FONT EDITORU.

COMOTRONIC NEWS

Výsledky slosování zákazníků za I./IV. 1993

Ve slosovnání, které provedl nás služební C64 metodou výběru náhodných čísel se štěstí přiklonilo k těmto našim zákazníkům:

1. cenu – disketovou jednotku VC 1541–II získal
Miloslav Mlčkovský
Na pískovně 649
Liberec

2. cenu – DTP program PRINTFOX získal
Václav Trojan
Bavorova 991
Strakonice

3. cenu – Final Cartridge III získal
Petr Hrda
Budovatelů 383
Poběžovice

Šťastným výhercům blahopřejeme a ostatním zákazníkům přejeme, aby se na ně štěstí usmálo v dalším kole.

PROGRAMOVÉ NOVINKY

Pro konec léta připravujeme k distribuci výborné programy pana Billy z Hlohovce, které představují první, opravdu profesionální programy pro C64 z česko–slovenské produkce.

PLOTBASIC II

Program firmy BILLA je určen pro spolupráci se všemi dostupnými plotry, které jdou připojit k C64, například ALFI, XY 4150, Minigraf, atd.
Dokáže při rýsování otáčet libovolně souřadné osy, pracuje i v polárních souřadnicích. Libovolná velikost a sklon písma jsou samozřejmostí. Bude vybaven 8 – 10 různými fonty pro psaní, samozřejmě s ČS diakriti-

kou. Využívá paměť C64 jako buffer, do kterého ukládá až 8000 kroků plotru, takže můžete dále pracovat na počítači, zatím co plotr maluje. Pracuje v módu OPEN – obdoba tiskárny, nebo TEXT, kdy se dá použít jako editor.
Toto a mnoho dalších funkcí Vám bude k dispozici na kazetě i disketě s podrobným manuálem v ceně cca 250,– Kč!!

ČStext

Zcela nový, původní a originální textový editor s ČS diakritikou, určený pro všechny dostupné devítijehličkové tiskárny od D100 po EPSON LX400. Další drivery jsou ve vývoji. Klávesnice tohoto editoru je definována přesně podle psacího stroje. Protože tiskne v grafickém módu a znaky jsou vektorově definovány, je možno volit libovolnou velikost písma u všech pěti dodávaných základních fontů. Rutiny TURBO DATA pro DISK i TAPE jsou samozřejmostí, stejně jako všechny běžné funkce textových editorů vč. vyhledávání a přesunů. Bude k dispozici do září. Předpokládaná cena kazetové i disketové verze s podrobným manuálem bude cca 300,– Kč!!

Pozor!

Jako příslušenství bude dodáván program podporující připojení PC klávesnice k C64 s využitím znakové klávesnice stejně jako u PC! Program + redukce pro připojení klávesnice budou dodávány od října 93.

Dalším doplňkem ČS TEXTu bude disketa (kazeta) s dalšími fonty, stovkami nejrůznějších ikon a znaků, které budete moci využít pro obohacení svých textů. Bude se dodávat od září.

FONTEDITOR

Pro ty, kteří si budou chtít vytvářet vlastní znakové sady nebo ikony (obrázky), bude k dispozici program, v němž je lze snadno vytvářet a využívat v obou výše uvedených programech. Bude k dispozici společně s ostatními programy v září.

GRAFICKÉ HARDCOPY

Pro uživatele, kteří potřebují s pomocí plotru nebo tiskárny zhotovit grafickou hardcopy , za jistých podmínek i barevnou, bude sloužit soubor programů pro všechny běžné plotry a alespoň osmijehličkové tiskárny od ALFI před D100 po STAR LC20. Možnost volby kazetové a disketové verze bude samozřejmostí.

(jk)

Předplatné časopisu FUN with Commodore

Cena za jedno číslo v předplatném na minimálně 5 čísel je 10,– Kč. Ročník 1993 bude mít celkem 10 čísel. V současné době máme ještě v omezeném množství k dispozici všechna vyšlá čísla, která je možno dokoupit za cenu 5,– Kč za číslo z ročníku 1992 a 10,– Kč za číslo z ročníku 1993. Cena jednotlivých čísel je 15,– Kč.
Předplatné se platí složenkou na adresu firmy Comotronic, na záda poukázky pro příjemce napište FUN 93 – počet čísel. Kopii nebo druhý kontrolní ústřižek zašlete společně s objednávkou na adresu firmy Comotronic. Pokud jste zaplatili i starší čísla, budou vám odeslána společně s dalším novým číslem, které vyjde.

Adresa redakce: Dolnomlýnská 2, 787 01 Šumperk * Autoři čísla: Jaroslav Vančura a Jiří Kouřil
Podávání novinových zásilek povoleno Oblastní správou pošt v Ostravě pod č.j. 2882/92–P1 ze dne 14.12.1992
Podávanie novinových zásielok povolené SP š.p. ZsRP Bratislava č.j. 613–PD–1993 zo dňa 1. apríla 1993
FUN with Commodore – časopis Comotronic klubu pro uživatele počítačů Commodore 64/128.
3/93 – 6. číslo
Fotosazba REPROtisk J. Kotinský * Tisk: Vegaprint Šumperk

Znakové sady pro Printfox

K používání znakových sad v programech Printfox, Eddison a Pagefox není jistě třeba obsáhlých komentářů. Znakové sady ve větším či menším množství jsou součástí těchto programů. Volají se, jak jistě víte, příkazem 'z' ve formátovacím řádku. Ke znakům se musí potom ještě příkazy 'h' a 'v' dodefinovat správná vzdálenost mezi znaky a řádky. Zatímco vzdálenost řádků se řídí zcela vaší chutí a charakterem dokumentu, pro vzdálenost znaků existují doporučené hodnoty zaručující harmonický obraz písma. Přirozeně i tyto hodnoty se dají alternativně měnit, v několika případech se to však nedoporučuje. Příkladem může být znaková sada Script, která dává uspokojivý výsledek jen při 'h=-8'.

Možná vás zarazilo záporné znaménko vzdálenosti znaků některých znakových sad. Je tomu tak u kurzív (šikmého písma). Aby písmo působilo uceleným dojmem, je potřeba každý znak mírně podsunout svému sousedu. Proto záporná vzdálenost mezi znaky.

Pokud se vám bude jevit vaše bohatství znakových sad jako nedostatečné, nezbude vám než sáhnout po znakovém editoru Characterfox. Umožní vám úpravu už existující znakové sady, respektive vytvoření vlastní znakové sady. Mimochodem, od firmy Scanntronik lze získat až 5 oboustranně nahraných disket se znakovými sadami.

Pro úplnost ještě uvádím některé ze znakových sad:

1 Printfox - česká sada

10 Printfox - česká sada

20 Printfox - česká sada

21 Printfox - česká sada

30 Printfox - česká sada

32 Printfox - česká sada

40 Printfox - česká sada

70 Printfox - česká sada

Vydává Comotronic klub Šumperk