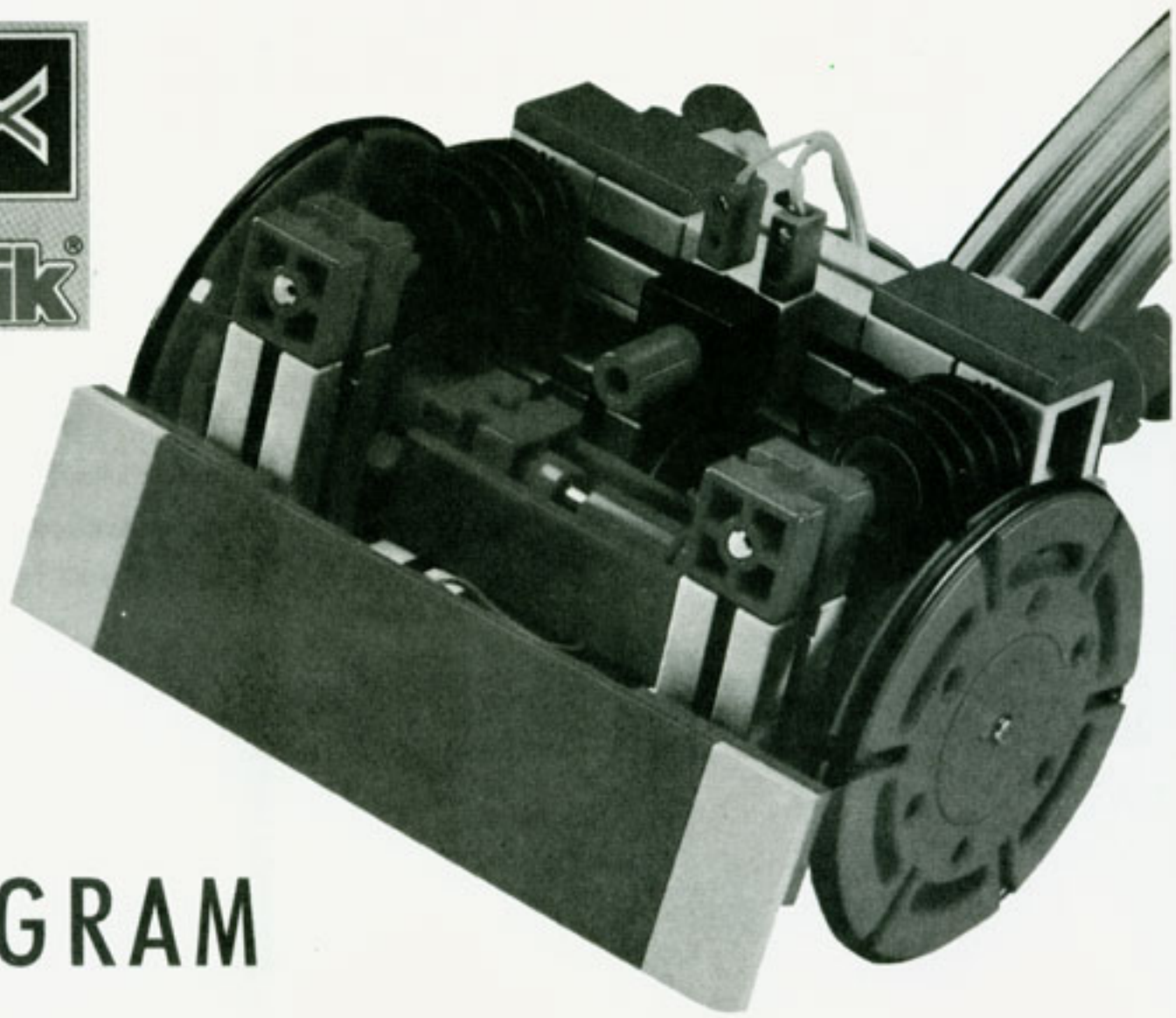
1988/7
cena 12 Kčs

Mikro

báze

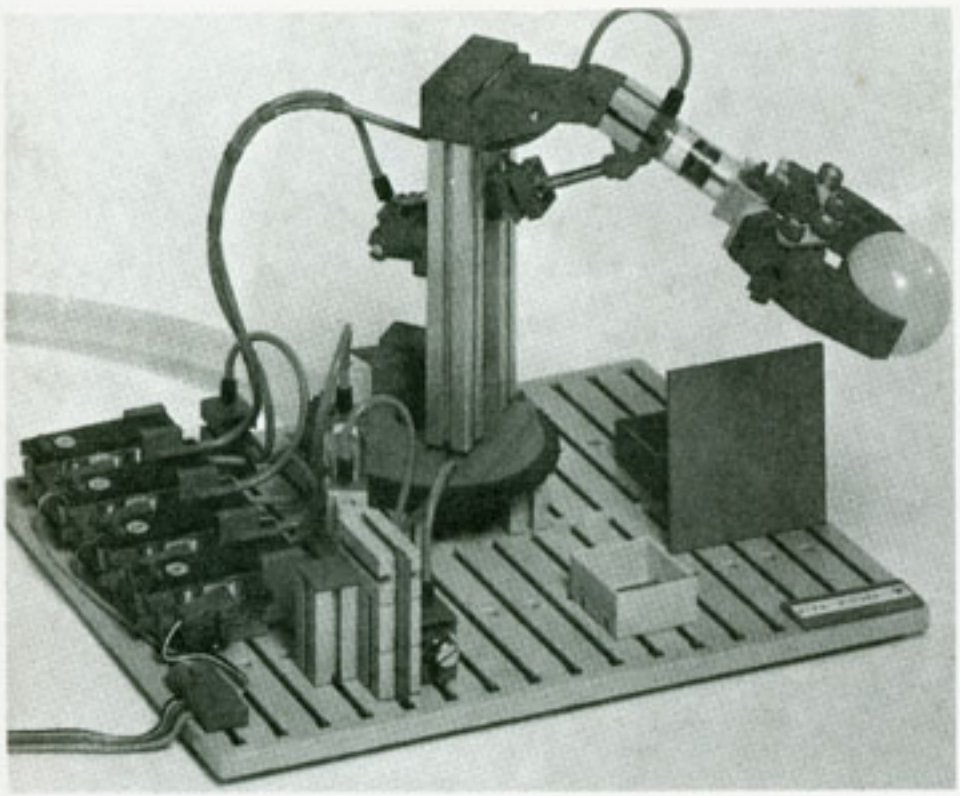
technický
zpravodaj
svazarmu
pro zájemce o
mikropočítače

CVK fischer fischertechnik®

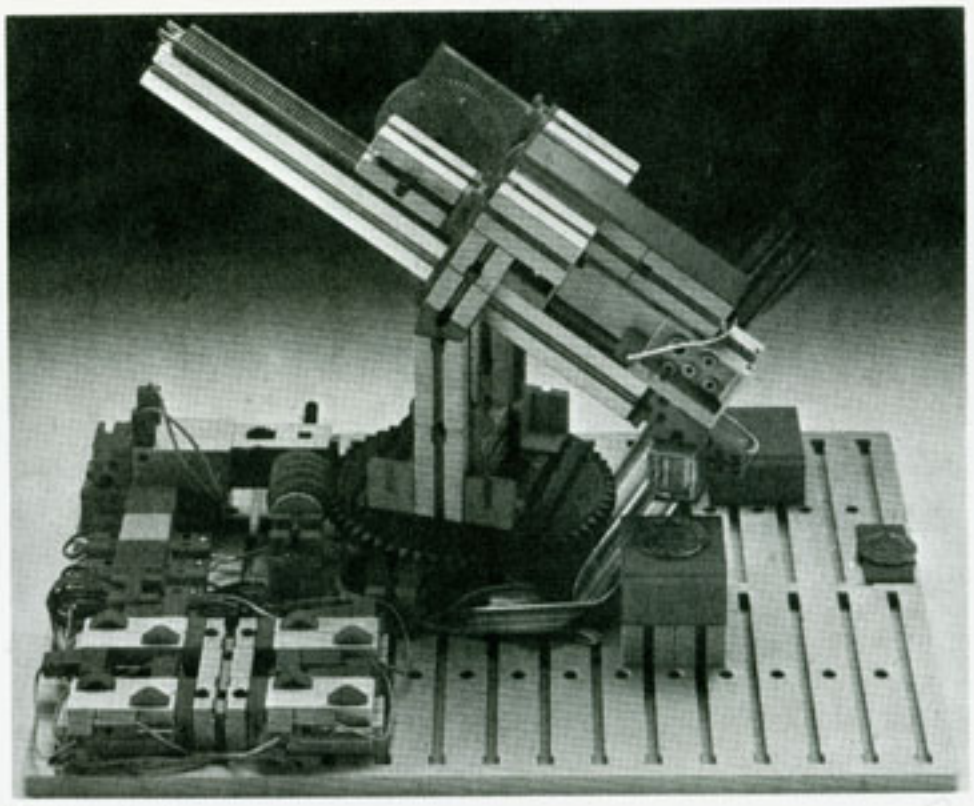


Firma Fischer vystavovala v uplynulém roce několikrát v ČSSR svoje výrobky. Proto jich několik chceme představit i vám. Jde o stavebnice modelů nejrůznějších robotů včetně potřebné řídicí elektroniky.

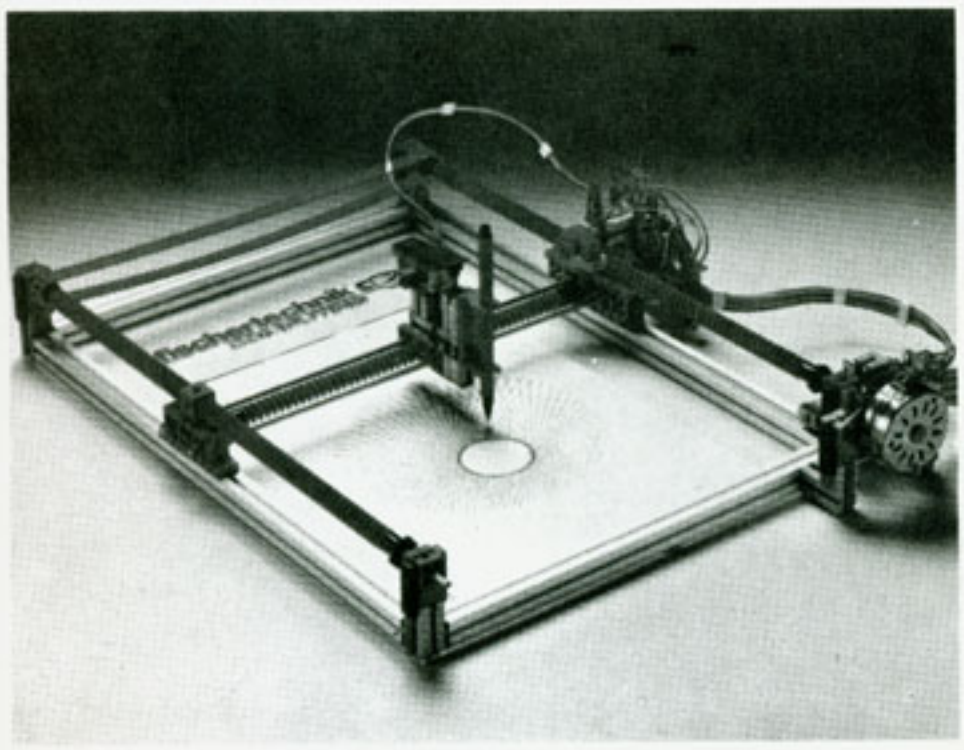
ŠKOLNÍ PROGRAM



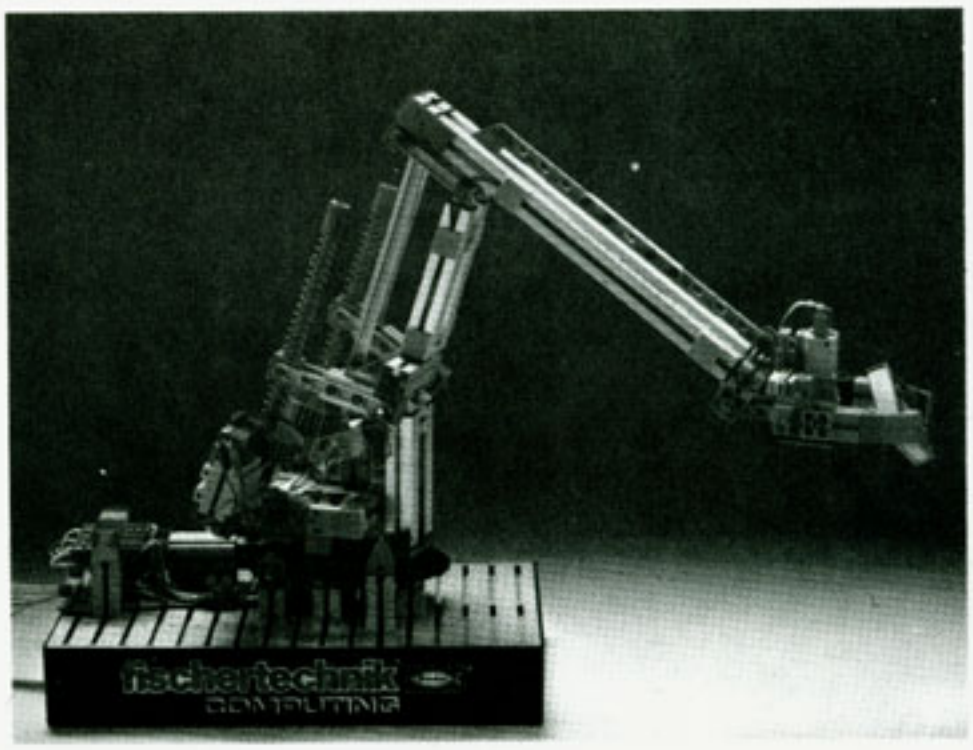
Pneumatický robot „PN-ROB"



Vyučovací robot



Plotter/scanner



Tréninkový robot

1988/7

Mikro báze

technický zpravodaj svazarmu pro zájemce o mikropočítače

OBSAH

K čemu vlastně...? 1
Rozhovor s R.Pecinovským 2
Kudy, kudy, kudy cestička? 4
Počítačová science fiction (1) 4
Ošetřování hard disku 8
Dešifrátor Basicu 10
Tisk s D-WRITERem 12
Kalkulátor ZX Spectra (1) 13
Univerzální interface MIREK 16
CPU versus paměť 22
Interface pro Atari 24
Číselnicový terminál 25
Univerzální tisková rutina 26
Amstrad CPC: systémové rutiny (1) .. 23
Metakomunikace (1) 30
Programová nabídka Mikrobáze 32

Technický zpravodaj Svazarmu pro zájemce o mikropočítače. Vydává 602. ZO Svazarmu ve spolupráci s redakcí časopisu Amatérské radio. Povoleno ÚVTEI pod evidenčním číslem 87 007. Zodpovědný redaktor ing. J. Klabal, sestavil ing. A. Myslík. Redakční rada: P. Horský, ing. J. Klabal, ing. P. Kratochvíl, J. Kroupa, ing. A. Myslík, ing. J. Truxa. Ročně vyjde 10 čísel, cena výtisku 12 Kčs podle ČCÚ a SCÚ č.1030/202/86. Roční předplatné 120 Kčs. Objednávky přijímá a zpravodaj rozšiřuje 602. ZO Svazarmu, Wintrova 8, 160 41 Praha 6.

SVAZARM

602.ZO

&

Amatérské RADIO

VIII. SJEZD SVAZARMU 1988

K ČEMU VLASTNĚ?

Už jste si toho taky všimli? Je tu nová vlna. Nikoli přílivu, ale odlivu. Odlivu zájmu. Hned vysvětlím.

Mám známého, který mě pořád vidí vysedávat u počítače. Před rokem se rozhodl věnovat část devizáku na nákup ZX Spectra. Když se vrátil coby novopečený majitel počítače, dal jsem mu všechny kazety, co jsem měl. Ať si ozkouší, co a jak. I to peklo herního poblouznění. Pár měsíců chodil s červenýma očima ve fosforeskujícím obličeji. Jako většina z nás, než jsme si řekli: "Tak a dost! Už žádnou hru! Apage, satanas!"

Přinesl mi všechny kazety zpět a ptá se: "Hele, nemáš něco, co by k něčemu bylo?" Vysvětlil jsem mu, co a jak - jazyky, slovní procesory, databanky a podobně. Za měsíc se objevil: "Je to pěkný, hlavně ten hudební program, taky ten malovací, ale nic z toho mi nijak zvlášť k ničemu není. Prostě nemám pro to žádný využití." On je totiž kuchař. Psát si recepty do databanky je přece jen nesmysl. Jednak je má v hlavě a jinak se kdykoli může podívat do spousty knih s barevnými obrázky. Zkusil Basic. Zrovna v televizi běžel ten bratislavský kurs. Zase z toho nic nebylo: "K tomu bych ještě potřeboval video, abych si to mohl pouštět sem a tam. Teď jsem jednou vynechal a už jsem mimo." Mých několik xeroxových svazků mu pomoci nemohlo - neumí anglicky. Učebnici Basicu SNTL napřed hodil do kouta, pak ji odnesl do antikvariátu.

Jednou se stavil, chvíli mlčel a pak to z něj vypadlo: "Hele, nevíš, kdo by koupil Spectrum?" Trochu mne zaskočil. Skoro jsem měl pocit, že za to nějak sám můžu, že na tom mám svou vinu. Ne snad že by si místo Spectra chtěl koupit něco lepšího, on se ho zbavoval jako fenoménu. Zřejmě jsem měl pro něj i pro ostatní, co dospěli do stejného bodu zvratu, udělat víc... Měl jsem přinejmenším vytvořit aspoň deset perfektních programů denně. Měl jsem pro jejich prodej vystavět rozsáhlou distribuční síť s několika sty prodejen. Měl jsem postavit několik výrobních linek, ze kterých by do té sítě denně sjížděly tisíce různých počítačů, monitorů, tiskáren, hard disků... Měl jsem měsíčně vydat nejméně dvacet knih o tom všem. Měl jsem zaplavit trh modemy. Měl jsem založit a naplnit sto velkých databank, na které by se každý mohl svým modemem napojit. Měl jsem zajistit výuková centra. A školy, které lidi učí po drátě. Měl jsem...

I kdybych byl superman, nic z toho bych nezvládl ani omylem. A v tom to právě je. Na to totiž není třeba žádný superman, ale normální lidi, kteří vědí, o co jde, kterým se do cesty nestaví anachronické zátarasy byrokratických supermanů. Integrované poznání, um, nadšení z předem možného postupu kupředu by dokázalo s integrovanými obvody udělat zázraky. Izolovaný počítač, který nemůže napojit svou sběrnici na statisíce jiných, je stejně atomizován, jako člověk, který po zbytečných pokusech a nedobrých zkušenostech zaleje své osobní hranové konektory kanagomem dezintegrujícího nezájmu a netečnosti. Atomizován sedí u svého atomizovaného počítače a v té izolované idylce si jen tak pro sebe bastlí. A když na to nemá buňky, dá si inzerát "Prodám počítač".

Co jsem to vlastně chtěl? Jo...jen, že mám známého kuchaře, který si tady jednoho dne nad svým počítačem řekl: "K čemu mi to vlastně...?"

-elzet-

HOVORY O PROGRAMOVÁNÍ

(Dokončení rozhovoru s R.Pecinovským)

Co považuješ při tvorbě programu za nejtěžší?

Fázi určení koncepce. Tedy co má program umět a jak se má s uživatelem bavit. Někomu to může připadat přehnané. Ale ať si zkusí sednout před "pécéčko", projede si v něm dvacet různých slovních procesorů a pokusí se s nimi domluvit. Vzápětí pochopí, že věc není jednoduchá, že některé programy mají dobrou, jiné špatnou koncepci.

Třeba já používám Norton Editor, který má pouhých 30 kilo, což je v porovnání s řadou jiných hrozně málo. Kromě několika drobností má ale všechno, co potřebuji, žádné zbytečnosti navíc. Norton je nejjednodušší, nejkratší, nejlíp se mi s ním dělá. Je rychlý, protože je napsán v assembleru; ty druhé většinou v Céčku. Když potřebuji nějakou speciální operaci, kterou Norton nemá, provedu ji na procesoru, který jinak ignoruji. Dnes se za jeden z nejlepších slovních procesorů považuje Word Perfect. Jde s ním dělat hrozně moc operací. Jenže je má do sebe tak nějak podivně zapasované, mnohé z nich jsou až nepříjemně pomalé.

Koncepce prodává program. Vem si Turbo Pascal firmy Borland. I když jiné programy mohou produkovat kratší a rychlejší kód, každý radši sedí u "borlandu", protože je to koncepčně vymakaný, integrovaný systém. Když se objevilo Turbo C, šel Microsoft se svým Céčkem ke dnu a musel zabrat, než se mu podařilo přijít s ekvivalentem, který se tváří jako Turbo C. Proč? Protože pro uživatele je nejdůležitější, aby se mu s programem dobře dělalo, i když je třeba méně efektivní. A to je věc koncepce. Proto ji považuji za nejobtížnější.

A taky proto tvrdím, že my nemáme šanci se prosadit na světovém trhu se softwarem. Nemáme měřítka srovnání, nejsme "v tom". Tady je spousta kodérů, ale minimum koncepčních programátorů, skutečných architektů programu. Koncepce, jako první fáze tvorby, se jaksi "nenosí". A ani tu druhou fázi, tedy tvorbu datových struktur, u nás moc lidí neovládá.

V čem vidíš klady datových struktur?

Když je dobře navrhneš, odpadne ti spousta práce s programováním. Třeba teď jsem dělal nový standard pro klávesnici "pécéčka". Postupně jsem zjistil, že vlastně programuji tabulkami. Původní program, který ošetřuje klávesnici IBM PC, je plný fint a skoků. Pokud bych ho v tomto duchu měl rozšiřovat na českou klávesnici, neskončil bych ani za půl roku. Použitím tabulek se celý program výrazně zpřehlední. A i když programuješ v assembleru, výsledkem je čistá struktura.

Dá se říct, že datové struktury programují za tebe. Wirth se jimi zabývá ve své knize Algoritmy + datové struktury = programy. Vyšla teď ve slovenském překladu. Ale ani z téhle tlusté publikace všechno není tak zřejmé, jako když se tím zabýváš přímo v nějaké aplikaci. Datová struktura převezme na sebe spoustu práce, kterou bys jinak vynaložil na algoritmizování.

A třetí fáze tvorby programu?

Právě algoritmizování problému. To je to nejjednodušší a taky to první, čím se při výuce programování začíná.

Přejdu hned k metodice výuky. Program Karel je dobrý tím, že nemá žádnou syntaxi, děti se při stavbě programu zabývají jen algoritmizováním problému. To, že algoritmus není schopen života bez dat, je v Karlovi velice šikovným způsobem potlačeno. Dítě totiž při prvním doteku s počítačem ještě nemůže datové struktury plně pochopit. Tak ho učíš jen tu algoritmizaci.

V té souvislosti jsem si ověřil užitečnost kopenogramů, které jinak spousta lidí pomlouvá. Když se mi někoho podařilo přesvědčit, aby program řešil pomocí kopenogramů, a bez dalších mezifází je opsal do zdrojáku, chytlo ho to, a už od této metody neutekl. Neznám člověka, který by s kopenogramy vyřešil větší problém a pak se od nich odvrátil. Vždy už u nich zůstal.

Kopenogramy pokrývají celou oblast od Karla přes Pascal až po Prolog. Podle současných znalostí o programování se dá říct, že by dětem mohly vydržet až do programátorského důchodu. Při výuce s kopenogramy se děti vlastně učí určitému jazyku, který už nebudou muset nikdy v životě měnit. Pochopitelně se mohou objevit nové přístupy a všechno bude najednou jinak. Ale soudobé programování je kopenogramy prakticky plně postižitelné.

Když děti vymyslí program v kopenogramech, Karel pak vlastně hraje roli vyučovací pomůcky, na které si děti ověří, zda to, co vymyslely, je správně. Proto je třeba, aby v tom nebyla syntaxe, aby děti do Karla jen přepsaly, co si vymyslely. Karel samozřejmě syntaxi má, ale řídí si ji konverzací s uživatelem, který tak není pod jejím přímým tlakem.

A jak vyučovat datové struktury?

Pro tuhle výuku se zatím nepodařilo vytvořit nějakého takového Karla. Nezbylo, než přejít na Pascal, který je celosvětovým výukovým standardem. Pascal jako vyučovací jazyk je navržen perfektně a zatím se nic lepšího neobjevilo. Napadlo mne, že když už se někdo zajel na Karla, zkusím ho naučit přepisovat do syntaxe Pascalu to, co v Karlovi vytvořil. Tak pochopí souvislosti a půjde rychle dopředu. Lidem se při tom přechodu líbí i nová práce s komfortnějším editorem, než jaký má Karel. A že si se zdrojovým textem mohou všelijak hýbat, přenášet kousky, používat dopředné reference, prostě že je toho hodně povoleno. Pochopení přepisu karlovských algoritmů do syntaxe Pascalu jde skutečně svižně kupředu.

Pak se může přejít na datové struktury. Protože žáci už nemají problémy se syntaxí, můžeš jim rozumně, per partes, předložit další oblast. Tak je postupně učíš jednu věc za druhou. Nehrneš to na ně všechno najednou, jako se to dělá v některých učebnicích. V nich to jde jedním hustým proudem - syntaxe, algoritmizace, data... Jedno přes druhé, od ničeho k ničemu, nic se nedodělá pořádně do konce.

Jak učíš koncepci?

Na to není metodika. Dělám to tak, že vyberu koncepčně dobře navržené programy a ty pak předvádím. Tady bych se chtěl zmínit o některých klucích, kterým jde zpočátku všechno hrozně rychle, až mají pocit, že kroužek, do kterého začali chodit, je brzdí ve vývoji. Odejdou a hrajou si s počítačem doma sami. To je cesta nikam. Prokázalo se mi to při jedné soutěži. Jeden takový malý, odpadlý génius si se soutěžním úkolem vůbec nedokázal poradit. Zatímco ostatní, kteří se výuky v kroužku pravidelně účastnili, program napsali bez potíží. Tenhle malý démon, ke kterému ještě před časem všichni zbožně vzhlíželi, před nimi zcela propadl.

Nakolik si myslíš, že by lidé, kteří se učí programovat, měli vědět, co se děje v kompilátoru, interpreteru, v hardwaru?

Já to dětem neříkám, a myslím, že tím nijak nestrádají. A když, tak jim řeknu jen základní informace o vstupu, výstupu, že je paměť smazatelná a nesmazatelná, protože na tohle chtě nechtě během praktické výuky narazíš. Všechny implementační věci by se jim měly říkat na úrovni toho, kam ses dostal s výkladem.

V tom bych ti rád oponoval. Myslím, že lidé, zabývající se programováním, by měli mít určitý vhled do věci hned od začátku. Mám za to, že tu jde o zlom myšlení vůbec. O nový, funkčně strukturální pohled nejen na počítač, ale na všechno kolem sebe. Je to asi jako kvalitativní skok z newtonovské do kvantové fyziky. Zdá se mi, že učit programovat v rámci karteziánského myšlení celou věc zamlžuje, i když žáci nakonec budou schopni programy psát. Jenže budou mimo širší souvislosti.

Já učím děti od páté třídy nahoru. Ony chápou toho Karla jako robota, který si to vše sám vymyslí. Na téhle úrovni všechno zvládnou takřka levou zadní. A to včetně rekurze, aniž by potřebovaly mít do toho počítače vhled. Když pak přejdou mezi pokročilé a začnou se zabývat daty, provádím s nimi praktické aplikace. Zkušený programátor nedělá věci neefektivně, nejde nesmyslnými cestami, protože je svými zkušenostmi a znalostmi už předem vyloučí. Děti ale mohou snadno na takovou cestu sejít. Když vidím, že někdo něco dělá úplně podivně, pak mu z toho vnitřku něco prozradím. Když to pochopí, příště se špatné cestě vyhne.

Když chceš třídit data, musíš o tom samozřejmě něco vědět. Donedávna jsem si myslel, že nejpomalejším tříděním je bubble sort. Ale v Prologu, který vymýšlí program za tebe, jde zadat ještě pomalejší třídění, zvané naivní. Vyplývá samozřejmě ze špatného zadání, které může zplodit ten, kdo o třídění nic neví.

Může se stát, že někomu nepřipadá něco tak jasné, dokud se nepodívá víc dovnitř. Ti kluci, co jsem je učil, dělali různé soutěže, vyhrávali je, ale co nebylo nutné k vědění, jsem jim neříkal.

Spisovatel nemůže psát bez znalosti výstavby literárního útvaru, bez znalosti lidské psychiky atd. Jinak plodí jalovosti, kterými jsme tak obklopeni. Nebo herec. Ryzí "přírodáci" už dnes skoro nejsou možní. Na akademii herce učí všemu možnému, aby dokázali svou expresí vyjádřit charakter postavy. Myslím si, že u programování je to totéž. Když lidi učíš jen příkazy, aby ta želva někam šla nebo nešla, a nestojí to na ničem, než na tomhle, zdá se mi to být málo. Myslím, že tady je i zdroj určité ztráty průzračnosti a soudnosti, zaviněné úzkým pohledem.

Člověče, není tomu tak. Vezměme si třeba toho spisovatele - o teorii pozadí tvorby má smysl se učit, když už si k psaní přičichl. Na školu, kde se tohle dá studovat, nevezmou, respektive by neměli vzít nikoho, kdo se psaním už nějak nezabýval. Člověk dokáže docenit podstatu věci teprve tehdy, když už něco sám udělal a může to porovnat s tím, co udělali jiní, i s tím, co říká teorie. Já jsem si při učení ověřil, že těžko můžeš někomu říkat, aby psal tak či onak. Ani z teoretických knih nikdo není hned moudrý.

Každý se nejvíc učí vlastní zkušeností, příkladem. I ti herci a spisovatelé. Někdo, kdo je dobrý, jim ukáže, jak na to jde on. A oni díky svému talentu pochopí, o co jde. Já jsem zatím nepřišel na lepší metodu, než ukazovat těm klukům, jak to dělám já, protože v tom momentu jsem pořád ještě lepší než oni. Jako když se slabší šachista učí od dobrého. Teorie ti spoustu věcí dá, pomůže ti vyhnout se základním nesmyslům, ale na mistra se musíš vyhrát sám. Nemůžeš se na něj vyteoretizovat.

To máš jako s výukou cizích jazyků. Když si jen přečteš učebnici, budeš slabým pasívním znalcem jazyka. Abys ho opravdu ovládl, musíš ho aktivně používat v ústním i písemném projevu, musíš přečíst hromadu knih. Pak do toho jazyka začneš pronikat. To platí i v programování.

V čem vidíš hlavní přednost a nedostatek vyučujícího počítače?

I když by se děti měly brzy začít učit s počítačem, neměl by jim nahrazovat styk se živým člověkem. Počítač dítě naučí penzum znalostí lépe než učitel. Ale děti se kromě látky učí i sociálním kontaktům a postojům. Ty, které se učí jen s počítačem, se pak neumějí pohybovat mezi lidmi. Program může děti "naučit násobilku", ale těžko je může vychovávat.

Myslíš si, že děti, kterým práce s počítačem vyloženě nejde, nebaví je stejně jako dějepis nebo matematika, budou v budoucí společnosti nějak handicapované?

Časem se bude stále více prohlubovat rozdíl mezi programátory a uživateli. V současné době většina uživatelů zároveň programuje. V budoucnu bude skoro každý uživatelem, programátorem se stane jen ten, koho to bude opravdu bavit. Podle mne nebudou děti, které by práce s počítačem nebavila. Budou děti, které nebude bavit programování. Každé dítě však radši párkrát ťukne do klávesnice, než by hledalo něco ve slovníku nebo v jiných knihách. Lidé, kteří dnes odmítají mít s počítačem cokoli společného, se počítače bojí. To u dětí nepřipadá v úvahu - ty se ho nebojí. Takže ve všech běžných aplikacích, kde nám počítač bude ulehčovat život, nebudou problémy. Ty ovšem nastanou těm, kteří nebudou schopni si na počítači něco dotáhnout do konce, jít trochu dál, než kam nějaký aplikační program momentálně dovoluje jít. Všichni budoucí výzkumníci, vědci, inženýři, doktoři, kteří budou počítač hodně používat, mu občas budou nuceni něco "dořict". Ne snad na úrovni profesionální programátorské práce, ale v rámci specifičtějšího uživatelského přístupu.

Třeba když u nás administrativa dostane příkaz, že má z databanky vypisovat nový formulář, který má vypadat tak a onak, přijdou za námi, abychom jim to v té databance nějak připravili. Uděláme nový formát výpisu a je to. Ale ne všude budou programátoři hned vedle v kanceláři a ne vždy budou mít čas.

Rozmouval -elzet-

KUDY, KUDY, KUDY CESTIČKA?

Nikoli pro folklórního Jeníčka, ale pro náš časopis Mikrobáze. Aneb - co by mělo tvořit jeho obsah, aby si v něm každý našel něco svého, ale i o něco víc, než jen něco málo svého. Mluvil jsem o tom s různými lidmi. Diskuse byly fandovsky zavilé. Někdo požaduje, aby se psalo převážně o ZX Spectru, jiný se svatosvatě zapřísahá, že smysl má psát jen o IBM PC, jiní horují pro Atari, Amstrady... Každý podle toho, co má nebo na čem dělá. Když se na to podívám z pohledu časopisu, musel by být rozbit na počítačové klany jako třeba polský Bajtek. Nevím, jaký názor na to máte vy, ale mně se takové rozbití vůbec nelíbí. Nelíbí se mi, že tam pro mne jsou dvě stránky, které mne zrovna v tom kterém čísle ani nemusejí zaujmout. To pak listuji, jestli by se někde přece jen nenašlo něco, co by...jenže všude klany a klany počítačů, o jejichž vnitřnostech vůbec nic nevím a nikdy vědět nebudu. Přece si kvůli Bajtku nekoupím pět počítačů.

Podle mého názoru by se časopis také neměl pokoušet nahrazovat specializovanou knihovnu, protože ji ani nahradit nemůže. A to i tam, kde není co si do té knihovny koupit. Není vinou těžko se rodících počítačových časopisů, že SNTL v konzistentní součinnosti s ostatními vydavatelstvími tak zdařile ignoruje desetitisíce uživatelů různých mikropočítačů i programovacích jazyků. Proluka se dohání roztříštěnou námahou v pobočkách společenských organizací. Jakkoli jde o činnost záslužnou, z celospolečenského hlediska je investovaná námaha z velké části jalová, protože zbytečně mnohonásobná, a na každého, kdo by její výsledky potřeboval, se ani zdaleka nedostane.

Když čtu zahraniční časopis cele zaměřený na ZX Spectrum nebo Amstrad CPC, musím mít za zády referenční databanku, kterou jsou (opět zahraniční) knihy o těchto počítačích. Pisatel článku samozřejmě nemůže popisovaný detail rozvést do absolutního pochopitelna, protože by tak vlastně napsal novou knihu. Ovšem časopis cele zaměřený na jeden typ počítače je opět něco zcela jiného než promixovaný substrát klanů, kde se počet stránek věnovaných mému počítači pohybuje od dvou k nule.

Mám za to, že počítačový časopis, určený lidem všech možných profesí a zaměstnání, do nichž počítač vstupuje jako pomocný nástroj, by m.j. měl rozšiřovat základní orientaci v přepestré škále problematiky počítačového světa. Měl by poskytovat komplementární, holistický pohled na celou oblast. A měl by orientovat nejen technicky, ale i lidsky, na což se tak podivně rádo zapomíná.

Prakticky s každým, kdo zná měsíčník Byte, jsem se shodl na tom, že ze všech časopisů tohoto druhu zvolil nejzajímavější cestu. Jistě - má nepoměrně víc stránek než Mikrobáze. Co je však podstatné - když chci vědět, co je to neuronová síť, multitasking, architektura mikroprocesorů atd. apod., v Bytu najdu čtivou odpověď sestavenou z různých pohledů několika specialistů. Tím se hned nestanu odborníkem na dané téma, ale dozvím se poměrně dost na to, abych věděl, o co ve své podstatě jde. Orientuji se, což mi svým způsobem pomáhá i ve vlastní práci. Baží-li někdo po hardwarových konstrukcích, Ciarciův koutek ho jimi zaplaví. Informace o soft/hardwarových novinkách, o práci s nimi, o výsledcích jejich testů apod. tvoří další velký díl časopisu. Ani kodéři nepřijdou zkrátka. O orientaci, kterou poskytuje Byte svou rozsáhlou inzercí, nemluvě - ovšem to už je z poněkud jiného světa (smetanovsky řečeno - tato symfonie je pohříchu novosvětská).

K otázce po směru cesty mám tento návrh: Nechť se časopis Mikrobáze vedle prezentace zájmových soft/hardwarových činností stane i sondou do nekonečných hlubin a šíří computerového světa. Ať poskytne svým čtenářům i obecnější orientaci jak v detailu, tak v oborech se širší platností. Dnešní svět počítačů už zdaleka není nějakou izolovanou disciplinou. Naopak - mohli bychom o něm říci, že je interdisciplinárním katalyzátorem. Pokusme se společně vstupovat do tohoto dobrodružného světa poznání, abychom si rozšířili svůj pohled, který nám pomůže objevit řadu dosud skrytých a o to překvapivějších souvislostí. Nebojme se občas jít až k modravě mlžným obzorům, za nimiž Kolumbovi námořníci tušili nekonečnou propast. Ale čiňme tak formou nám všem srozumitelnou - našimi průvodci budiž měkký jazyk mateřštiny a názornost zobrazení.

Nezapomeneme samozřejmě ani na tvrdojazyčné kodéry, konstruktéry a matematiky. Zvláště těm posledním by měl být určen jejich vlastní koutek (nikoli klan!). A když se (nedejbože) stane, že někteří čtenáři v nějakém čísle z oné tvrdojazyčné oblasti nenajdou nic pro sebe, ostatní stránky se budou vždy obracet i k nim.

Z návrhu vysvítá, že jde o založení nového výrazu, nové tradice časopisu. Zpočátku budeme trochu zápasit s nedostatkem tiskové plochy, ale její rozšíření už je přislíbeno. Kolem návrhu bylo hodně redakčního štěkotu i mňoukání. Obojí však mělo jeden cíl - najít nejlepší podobu Mikrobáze. Ale abych tu neštěkal jen já - jaký názor na věc máte vy?

-elzet-

POČÍTAČOVÁ SCIENCE FICTION (1)

Na SOFSEMu 83 vystoupili **RNDr.J.Hořejš,CSc. a Ing.J.Franek** s netradičně pojatou přednáškou. Obsahuje mnoho zajímavých myšlenek pnoucích se nejen k fikcím budoucí, ale už i současné symbiózy člověk/počítač, která mnohé literární fikce rychlými kroky překonává. Věříme, že vám ideje, extenze a metateze autorů vedle obveselení přinesou nejedno hlubší zamyšlení nad zásadní otázkou - Qou vadis, člověče počítačový?

1.Místo úvodu

Člověk vždy dával přednost tvůrčí práci před rutinou. Aby nemusel pěstovat kozy, trmácel se Atilla za každého počasí přes půl světa a tvořivě si podroboval jeden národ za druhým. Tvořit totiž vždy znamená i ovládat. Einstein vytvořil teorii relativity a umožnil lidem ovládnout atom. Ani Atillovi ani Einsteinovi se však nepodařilo ovládnout myšlení svých poddaných - národy si myslely své a atomy pravděpodobně ani to. Teprve člověk současnosti má před sebou šanci - možnost vytvořit otroka, který mu bude sloužit nejen tělem, ale i duší. Má to ale dva háčky. Jednak je takový sluha do značné míry jen v představě (jednou nohou se dotýká země, ale ještě na ní pevně nestojí), jednak jde o to, aby to tak nakonec nebylo lepší - už první roboti z roku 1921 se v tomto smyslu zrovna moc nepovedli a koukali odkoukat od lidí trochu jiné vlastnosti než jen poslušnou pracovitost. Takže nevíme, na čem jsme - možná, že to, co bude sloužit, nebude ochotno iniciativně myslet, a to, co bude trochu víc myslet, nebude ochotno sloužit. Nebo to bude všechno naopak. Fantazii se meze nekladou; to je ostatně důvod, proč se trochu proběhneme po jejích pláních.

2.Důvodová zpráva

Domníváme se, že vždy a všude stojí za to občas zpochybnit příliš realistický a střízlivý postoj, který snadno vede do stadia, v němž spolu s fantazií zahyne i reálná "představivost, která je důležitější než vědomosti" (A.Einstein). Ostatně "fantastické představy mají původ ve skutečnosti a naopak nejvěrnější představy o skutečnosti musejí nutně oživovat dech fantazie" (V.I.Lenin).

Je zřejmé, že s počítačem je v každém případě možné vyhrát si víc než třeba s turbínou...a hra je první schůdek fantazie. Dialektická jednota vědy a umění nám tak dává panem et circenses (chléb a hry), v nichž se každý podle svých chutí může uplatnit jako duchovní gladiátor, příslušník nobility, i jako prostý, leč pyšný člen plebsu.

Konkrétněji: Fantazie umožňuje osvobodit se z tradice a uvažovat i jiné logicky konzistentní "možné světy", z nichž některé se mohou stát reálnými dřív, než se nadějeme. K.Blažek spatřuje jeden z úkolů scifi v profylaxi člověka před "kosmickým šokem" - a k tomu se můžeme dopracovat i bez zelených mužíčků. Známé jsou i folklórní výroky fyziků. N.Bohr se táže: "Je vaše teorie dostatečně fantastická na to, aby mohla být pravdivá?" A J.B.S.Haldene píše: "Mám podezření, že vesmír je nejen podivnější než předpokládáme, ale že je podivnější než jsme schopni předpokládat." Věříme, že v tomto smyslu patříte (alespoň) mezi virtuální vyznavače scifi, které O.Neff charakterizuje jako "lidi specifického ražení, kteří spojují prožitky umělecké tvorby s potřebou intelektuální spekulace".

A ještě konkrétněji: scifi může vynést na světlo střípky úvah, které odrážejí zárodky reálných filozofických, společenských a dokonce i technických problémů; ty se mohou někdy i poskládat v zrcadla nastavená budoucnosti. Scifi tak zahrnuje i prvky futurologie jakožto docela seriózní "vědy o projektech, programování a prognostice sociálních a technických podmínek budoucnosti". Není vyloučeno, že některé z dále rozebíraných fantazií mají už teď své reálné protějšky v tajných laboratořích velmocí i že se může opakovat situace z roku 1944, kdy Cleve Cartmill v časopise Astounding Science Fiction uveřejnil povídku Deadline, popisující řadu podrobností o atomové bombě a jejím použití. A měl pak nepříjemnosti s FBI, ačkoli o tajném projektu Manhattan vůbec nevěděl...Hirošima pak překvapila celý svět s výjimkou čtenářů zmíněného časopisu.

Ve všech svých úvahách jsme natolik zatíženi současností, že jakékoli odhady budoucnosti budou zřejmě zatíženy chybami. Tyto chyby budou tím fantastičtější, čím je odhad reálnější vzhledem k momentálnímu stavu vědomostí. Nebo řečeno obráceně - čím fantastičtější je odhad, tím reálnější mohou být šance jeho uskutečnění.

To je ovšem myšlenka, vystupující v mnoha parafrázích scifi i futurologie. Abychom v tom nebyli sami, odvoláme se ještě na klasika české scifi, Josefa Nesvadbu: "...je mnohem bezpečnější předpovědět i ten nejfantastičtější vynález, než cokoli vyloučit, o čemkoli pochybovat. Vynález vždycky může být objeven, negativní tvrzení může být často usvědčeno z malověrnosti. Je vlasně jistější být snílkem než skeptikem..."

Obrázek symbolizuje potenciální vývojovou linii symbiózy člověka a počítače. Navštívíme teď některé milníky budoucnosti spolu s autory scifi. Z celé záplavy povídkové, dramatické, filmové, ba i lyrické tvorby jsme vybrali k dostaveníčku jen několik - ale takových, z nichž je možno jako z Ezopových bajek vyvodit některá poučení či otázky.

Čtenáři vždy předkládáme krátký abstrakt, úryvek či přepis, v němž explicitně upozorňujeme na některé ideje (označené I), které v dalším komentujeme a "domýšlíme" (takové extenze jsou vyčleněny identifikátorem E), popř. extrahujeme a zobecňujeme do jistých (meta)tezí (T). V těchto komentářích zmiňujeme i myšlenky či epické zápletky z dalších literárních děl, na jejichž samostatný popis či rozbor již nezbylo místo; klíčové písmeno těchto dodatků je A. Otazník v některých odkazech svědčí o tom, že člověk si někdy zapamutuje spíš nápad než jeho zdroj.

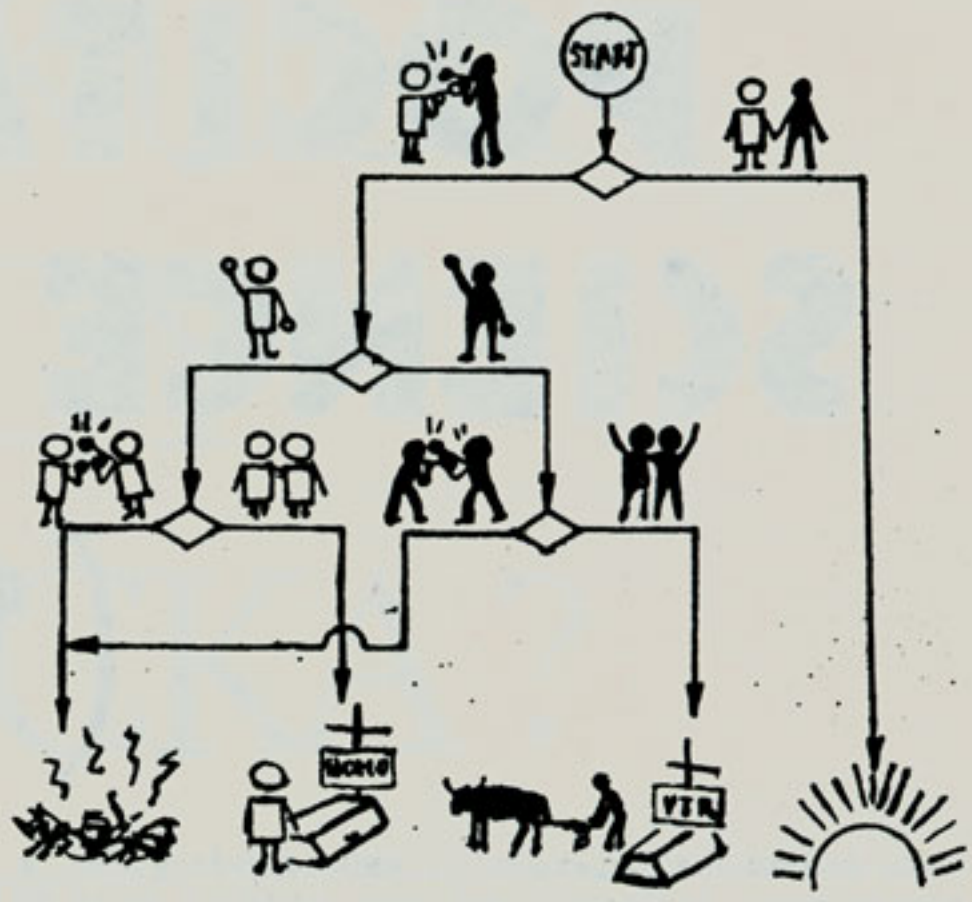
Jako základní kritérium uspořádání jsme nakonec zvolili míru fantastičnosti, resp. abstrakce, což by mělo umožnit i gradaci textu. Do jistého stupně toto hledisko koreluje i s chronologií - naděje na překvapení roste s časem a snad každý z nás tak trochu čeká na Godota, který slibuje tím víc, čím je jeho příchod nejistější a posunutý kamsi dál do budoucna.

I. Začneme skupinou povídek, poukazujících na problémy, jichž se můžeme dočkat i my. Již dnešní či nastupující výpočetní technika může vyvolat společenské důsledky, jejichž fantastičnost je jen realtivní a záleží od jiných než technologických faktorů. Tento oddíl scifi proto představuje ponejvíc společenskou kritiku, slib či varování a vyslovuje se k otázkám, jejichž pochopení nevyžaduje žádnou dnes nepředstavitelnou technickou konstrukci či zvláštní myšlenkový trénink.

II. Stanou-li se kdy počítače neodmyslitelným atributem života společnosti, začnou se také stávat jejími více či méně plnoprávnými členy, začnou vystupovat jako osobnosti. Bude-li to zatím jen dohodnutá hra a simulace, nebo začne-li se skutečně konstituovat nový typ stvoření se specifickými nároky ve sféře komunikace, myšlení, emocí, práva, sociálního zařazení apod., to je ovšem otevřená otázka. Právě tak problém, jak se v takové koexistenci změní postavení člověka. Různé odpovědi potkáme ve druhé skupině povídek; spadají do ní např. všechny ty, které čerpají náměty z tzv. Asimovových zákonů, i ty, co uvažují hybridní systémy člověk-stroj propojené nejen softwarově. V řadě povídek je popisována již cílevědomá snaha bytostí, odvozených ze současných počítačů; ta může přitom jít souběžně, ale může se i postavit do protikladu k současným hodnotám lidské pospolitosti. I zde je ještě v principu možné, že i taková snaha je (třeba nepřímo a ne-chtíc) "naprogramována". V každém případě se ale již předpokládá vznik nové kvality, nového typu poznávajícího a hodnotícího subjektu.

III. Další skok v dávce fantazie je spojen s představami inteligencí, které překonávají standardní formy chápání střízlivého programátora přinejmenším tak, jako obecná teorie relativity obzor řadového pracovníka v živočišné výrobě (většina povídek přitom vyvolává nostalgickou závist vůči životnímu stylu posledně jmenovaných). Přitom jde o formy podstatně rozlišné po stránce technologické (např. životy na jiné než běžné biologické bázi) či gnozeologické (jiné smysly, schopnosti jinak vnímat prostor a čas, akceptovat jiné logické systémy, jiné životní hodnoty apod.).

IV. Konečně na samém vrcholu tohoto uspořádání stojí literární výtvory, navozující potenciální pocit existence entit, které obvykle charakterizujeme téměř nic neříkající předponou vše- (všemohoucí vševědoucí apod.). Představované entity přitom svou obsáhlostí obvykle překonávají i

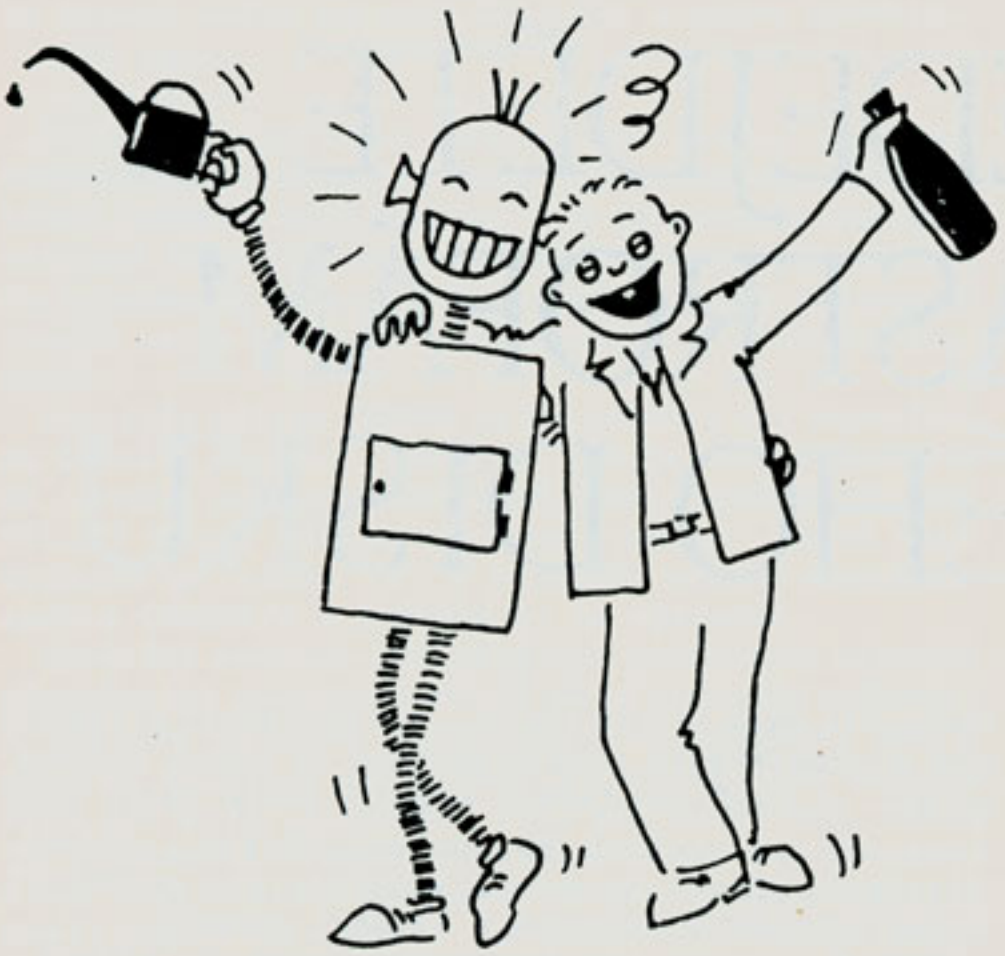


klasické religiózní představy "Nejvyšší Bytosti". Zde se často ocitáme na půdě jakési kosmologické mystiky vyššího řádu. Na rozdíl od středověké mystiky je logika někdy přesahována způsobem natolik absurdním, že už není dost dobře možné to jemně ironické mávnutí rukou, kterým snadno vybruslíme z osidel scholastiky, proti jejímuž "credo quia absurdum est" lze snadno vznést dostatek námitek právě proto, že to, v co požaduje věřit, není dostatečně absurdní.

(-1-) R.A.Lafferty: What's the Name of the Town? (Jak se jmenovalo toto město?) Osoby: Smirnov - nejdůležitější vědec, Valerie - zvědavá vědečka, Epikt(étos) - stroj cca 6. generace, ostatní - lidé výzkumného týmu. Místo: Institut, který není tlačen žádným plánem (T1).

Valerie obstarává nepodstatnou epiku, během níž vyjde najevo: Smirnov zadal Epiktovi úkol "najít něco, o čem nevíme, zda existuje, na základě absence informace", přičemž "není vyloučeno, že najdeme odpověď dřív než otázku" (I2). Tomuto zadání (I3) se Epikt vysmál, zatímco následující zpřesněnou specifikaci akceptoval: "Mám pocit, že se snažím vzpomenout na něco, na co jsem byl donucen zapomenout (I4). Zkusme to zrekonstruovat právě na základě této předpokládané snahy zničit všechny stopy (I5)." Zásoben kvanty encyklopedické informace Epikt postupně zjišťuje některé nesrovnalosti. Tak např. v textu o indiánském kmeni Chibcha se v mnoha řádcích neříká téměř nic, při sledování historie jazzu se objeví proluka, jako kdyby byl v jistém období celý jeho kus vytržen i s kořeny atd. atd. Z tisíců takových indikací tvoří Epikt hypotézu: ze všech informačních zdrojů bylo v r.1980, tedy před 20 lety, vymazáno slovo CHICAGO, a to proto, že bylo "vymazáno" i město samo velmi nepřirozenou katastrofou takového typu, že sama vzpomínka na ni by lidstvo traumatizovala. Proto byl urychleně vyvinut stroj, který - za pomoci rozhlasových a jiných vln - vymazal vzpomínky v mysli lidí. Jeho vynálezcem (a to je další pointa povídky) byl právě Smirnov, který se se všemi, včetně Epikteta, v závěru povídky noří do blahé nevědomosti, neboť hypnóza opět začíná fungovat. Ještě si však stačil uvědomit, že zapomínání obstarává přístroj, kolem kterého denně chodí v domnění, že jde o odepsané zařízení, které patří do šrotu.

T1 * Některé problémy je možno vyřešit jedině tehdy, když žádné nejsou zadány * Kdyby v Insti-



tutu dělali ASŘ, mohli by si dovolit popsaný experiment: objevit to, co má být zapomenuto a zapomenout to podruhé (zřejmě jsou věci, které stojí za to zapomenout víckrát). Neplánované objevy bývají ovšem často zajímavější a důležitější i mimo oblast scifi. Otázka je - na základě jakých kritérií může společnost někomu přisoudit právo volného výzkumu, aby se - statisticky vzato - ještě rentoval.

I2 * Některé problémy je možné vyřešit i tehdy, když nejsou zadány * Formulační podoba I2 s T1 je vypočítána na efekt, obsahová je podstatně menší. Jde o to, upustit od požadavku přesné specifikace úlohy. Jako je současná doba charakteristická posunem od programovacích jazyků (JAK) ke specifikačním (CO), tak bude budoucnost od počítačů nepochybně požadovat i pomoc při formulaci otázek (JAKÉ CO). Má-li počítač simulovat činnost lidského mozku, bude muset zahrnout i etapu nejasného "tušení souvislostí", pomoci řešit i ten nejdůležitější problém - stanovení problému samotného.

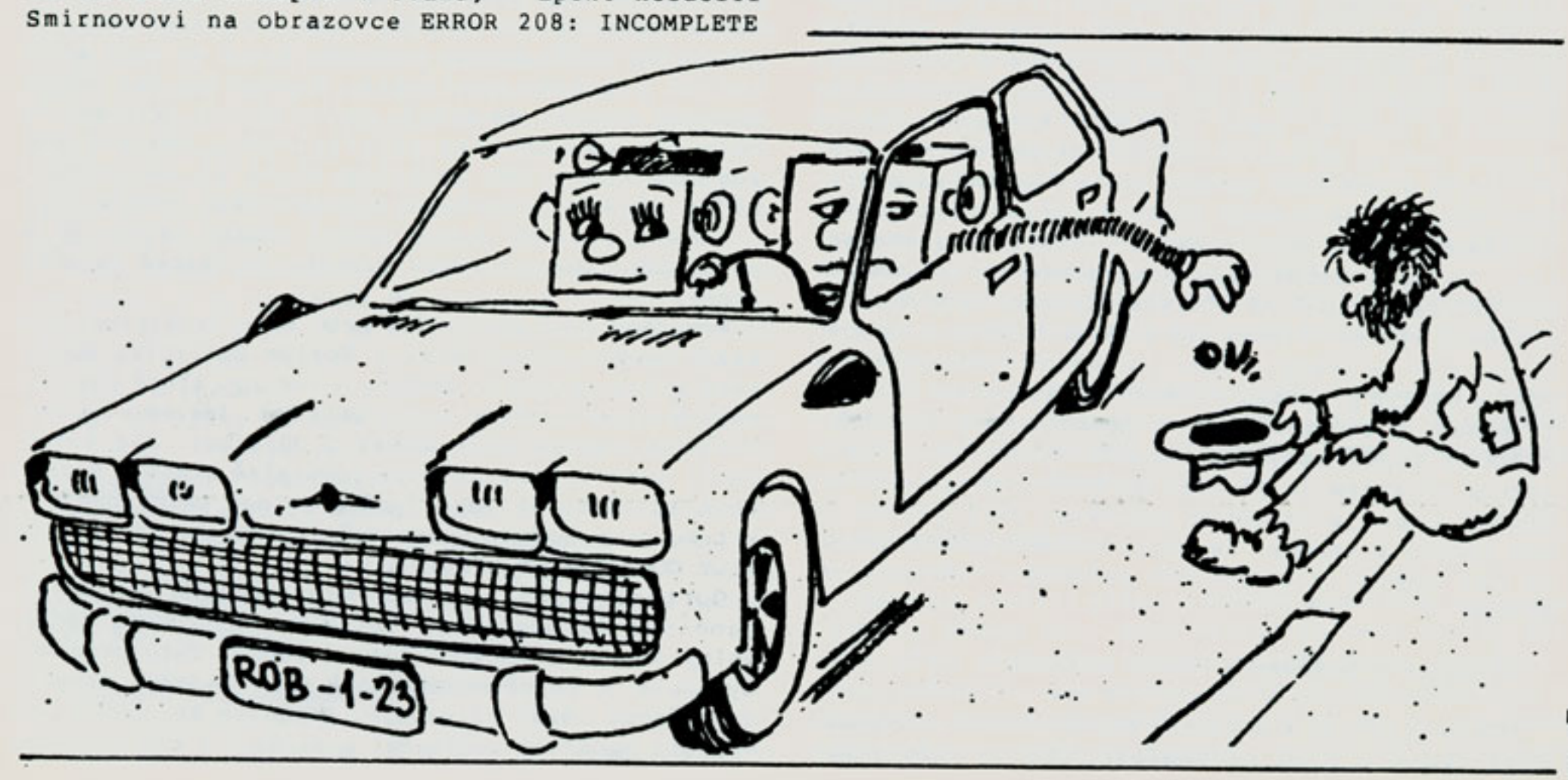
E3 * Reakce počítače - mohou být - budou - měly by být - neměly by být - antropomorfní (nehodící se vezměte alespoň v úvahu) * Epikt nesdělil Smirnovovi na obrazovce ERROR 208: INCOMPLETE SPEC., ale vysmál se mu. Z povahy činnosti stroje vyplývá, že může jít (a řekněme, že jde) pouze o simulaci emocí, dosažitelnou již prostředky současnými. Až na některé studentské produkty, se běžný software snaží budit dojem lidského partnera. Zato literární úroveň systémových vzkazů je beznadějně fádní, psychlogicky nestimuluje, neodlišuje důležitější vzkaz od zanedbatelné banality - FATAL ERROR 235: SEMICOLON MISSING (fatální chyba 235: chybí středník) versus WARNING 236: FIRE IN THE ROOM (upozornění 236: místnost je v plamenech) - a primitivní počítačový systém od "sofistikovaného" (zatímco z prvních několika vět prohozených se spolucestujícím můžete odhadnout jeho kulturní styl bez ohledu na námět rozhovoru, počítač zůstává v tomto smyslu amorfní).

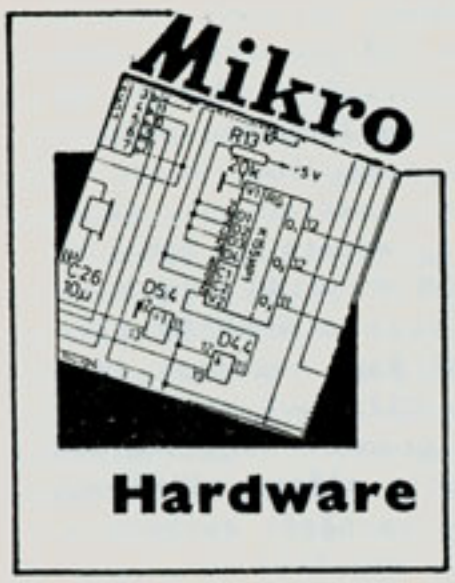
I4 * Zapomínání informace může být těžší než její zapamatování * Tato skutečnost je evidentní u člověka - nejen že neumíme provést účinný selektivní výmaz informací (ačkoli by to někdy bylo velice žádoucí z důvodů terapeutických, pracovních i společenských), my ani nevíme, proč to nejde (proč nová informace - vstupující za týchž podmínek, týkající se formálně i obsahově téhož subjektu apod. - nepřepíše starou). U jednoduchých počítačů lze výmaz provést logicky i fyzicky velice snadno. Není však vyloučeno, že u složitých informačních systémů s mnoha vzájemnými vazbami a složitou strukturou distribuované paměti se setkáme se situací analogickou situaci člověka - že operace DELETE bude (aspoň z hlediska robustnosti systému) představovat těžší úkol než WRITE či UPDATE (srv.s I5).

I5 * Potlačená informace zůstává informací * Totální výmaz informace v dostatečně složitých systémech nemusí být vůbec proveditelný - po informaci zůstane stopa ve formě "černé díry", či filmového negativu. Výmazem bitu 1 dostaneme bit 0, nikoli ne-bit. Myšlenka z povídky - vytváření "pozitivní" informace na základě negativních stop dřívější zničené - může už v současnosti připouštět docela seriózní formalizaci a zasluhovala by rozvážení (výzva pro specialisty v teorii informace a praktiky v otázkách vyhledávání?!). Je ostatně možné, že tady tušíme narození dítěte, které už delší dobu běhá po světě.

T6 * Než něco z počítačového sálu vyhodíte, zjistěte, jestli to nedělá něco jiného (třeba i žádoucího), než co by to podle vás dělat mělo (a nedělá) * (Může se týkat třeba i operátorek).

- pokračování -





PŘEDEJDĚTE KATASTROFÁM PEVNÉHO DISKU

Měsíčník PC World 11/86 přinesl několik pravidel péče o hard disk (zkr. HD) spolu s informacemi o pěti programech, které umožňují tato pravidla dodržovat.

Rychlost a snadnost práce s HD jsou jejími známými atributy, ale udržovat HD uspořádaný tak, aby pracoval co nejefektivněji, není snadná záležitost.

Ať je cena magnetických médií jakákoli, vaše data jste nezískali lacino a stojí za to předejít poruchám HD a potížím s vadnými sektory či operačním systémem. Stálá ostražitost je cena, jaká se platí za laciný HD. Ale ať už jde o typ výprodejní nebo nejlepší typ firmy IBM, dobrá práce s HD začíná jeho preventivní údržbou.

Naštěstí máme celou řadu pomocníků - PC Tools, Norton Utilities, Mace Utilities, Disk Optimizer, Safety Net. Většina těchto programů má implikovány možnosti, jako je ukrytí názvů souborů, změna dat a vyhledávání textu. Ale hlavním úkolem těchto programů je něco jiného. Žádný z nich sice nemá vlastnosti Direc Tree nebo lDIR, protože to jsou nástavby operačních systémů, ale nicméně jsou výborné pro dodržení dobrých návyků při práci s větším množstvím dat.

Článek se zaměřuje na rady, jak využít vlastnosti produktů dvou různých kategorií - jedny ulehčují některé problémy práce s HD, další pomáhají obnovit data v případě jejich "ztráty".

Zlatý pořádek

Občas neškodí, když si uděláte pořádek na svém psacím stole, zbavíte se starých papírů, uspořádáte dokumenty, dáte na místo tužky a pera... Takový pořádek je jednou za čas třeba udělat i s HD. Zamyslete se nad tím, zda je opravdu nutné mít okamžitý přístup k datům, uloženým přede dvěma a více lety. Vymažte data pro vaši práci nepotřebná, zkontrolujte soubory s krycími jmény jako XXX, XYZ apod. Takové záznamy nečiní HD spolehlivější; naopak mohou způsobit omyly a ztěžovat práci. Kopie souborů na disketách jsou dobrým zvykem. Jestliže se vám však podařilo nashromáždit hory souborů určených k archivaci tak, že jsou promíchány s programovými soubory a starými daty, použití příkazu DOS COPY nedává nijak příjemný výhled. Přenos dat z HD na disketu a vymazání dat z HD je práce na několik hodin. Pomocí příkazu COPY a DELETE programu PCTOOLS to zabere jen nějakých 15 minut. Místo psaní názvů jednotlivých souborů v PCTOOLS stačí kurzorem označit vybrané soubory a jedním příkazem je přenést do nového místa. Příkaz DELETE pracuje obdobným způsobem. Celý proces je tak rychlý a snadný, že se rozhodně vyplatí jednou za čas udělat v souborech pořádek, "uklidit ve sklepě".

Udržujte v pořádku i názvy souborů

DOS nemá příkaz k seřazení názvů souborů v seznamu. Když žádný z nich nevymažete, DOS je bude stále řadit dle data jejich zápisu. Ale postupným přidáváním a vymazáváním nových souborů se tento pořádek ztratí. I když uspořádání souborů podle názvů nebo času zápisu neučiní HD spolehlivějším, udržení jistého pořádku je dobrou pomůckou pro snadnější orientaci při vyřazování a třídění souborů.

Pořadí, podle kterého budete soubory třídit, záleží jen na vás - někdo bude jména souborů třídit podle abecedy vzestupně, jiný sestupně. Programy můžete řadit tak, že nejdřív budou v seznamu soubory s příponou COM a EXE a soubory bez přípony na konci. Některé programy nabízejí možnost řazení dle velikosti souboru nebo data zápisu. Protože jsou řazeny pouze názvy souborů, nikoli samy soubory, operace je nesmírně rychlá.

PCTOOLS i Mace Utilities mají možnost třídění adresáře. Ale Norton Utilities nabízí ještě praktičtější přístup. Tento samostatný program příkazem DIRSORT A-N uspořádá jména souborů asi za 5 sekund. Příkaz DIRSORT promítne i pomocné menu.

Pravidelně reorganizujte svůj HD

Sektory jsou jednotky paměti HD. Malý soubor může být obsažen v jediném sektoru, velký může zabrat i stovky sektorů. DOS určuje umístění dat na sektorech tak, jak jdou postupně za sebou - tak se může stát, že soubor zabírající pět sektorů může být umístěn zčásti na třech přilehlých sektorech a zbývající dva sektory mohou být úplně jinde; tak bude soubor rozházen po celém HD. To se stává především u souborů, které časem narůstají. Nekontinuita uložení dat se samozřejmě projeví prodloužením času přístupu a nutí hlavy HD k mnoha zbytečným pohybům. To může ovlivnit jeho spolehlivost. Roztříštěnost souborů zjistíte příkazem DOS chkdsk/path/*.* pro každý nižší seznam souborů, které ve svém aplikačním programu používáte. Rozmístění dat všech seznamů (textů, databází, kalkulačních tabulek) by mělo být čas od času zkontrolováno. Některé 80K soubory mohou být umístěny až na 23 sektorech. Žádný příkaz operačního systému však nemůže toto umístění změnit; a kopírování obsahu celého HD a přemístění dat na novou disketu je časově náročný proces.

Programy Disk Optimizer nebo Mace Utilities, ale také Norton Speed Disk z Norton Utilities mohou tuto práci usnadnit. Obsahují rutiny, které percentuálně vyjádří účelnost uložení dat. Základní optimalizační proces může trvat i 30 minut, ale každá další optimalizace už bude podstatně kratší. Neznamená to však, že tak získáte volnou paměť navíc - k tomu jsou určeny komprimační programy ze skupiny FILE COMPRESSION UTILITY.

Optimalizační programy mají opravdu minimální paměťové nároky. Disk Optimizer potřebuje jeden volný sektor a Mace Utilities žádný. Tato pracovní symfonie je úplně automatická a po spuštění nevyžaduje žádný zásah uživatele. Nemusíte se obávat ani vypnutí napájení počítače, protože i s tím programy

počítají; data jsou vždy na bezpečném místě. Tak je proces možno kdykoli přerušit a znova spustit.

Holá fakta

Přidejte příkaz CHKDSK do svého souboru AUTOEXEC.BAT. Nejen že bude identifikovat soubory, které nemají kontinuitu, ale provede i analýzu seznamů souborů, umístění dat a struktury souborů na kterémkoli médiu. Tento příkaz bez parametrů rovněž najde a opraví určité chyby v umístění souborů.

Jestliže příkaz CHKDSK najde sektory dat nepatřící do žádného souboru, zeptá se, zda je chcete sestavit do nového souboru: CONVERT LOST CLUSTERS TO NEW FILES? Tento příkaz nevytvoří soubor, dokud není doplněn příkazem /F. V případě vašeho souhlasu zadáte CHKDSK/F a počkáte si na výsledek.

Opuštěné sektory jsou většinou pozůstatky programů přerušených narychlo nebo nějakým nevhodným způsobem. V takovém případě CHKDSK zachytí fragmenty dat v souboru FILExxxx CHK v ROOT directory, kde xxxx je 0000 pro první fragment, 0001 pro druhý atd. Potom je jednoduché fragmenty zkontrolovat, ev. vymazat, a tak získat místo na disku. CHKDSK rovněž předchází nepříjemnostem; patří do souboru AUTOEXEC.BAT, takže je používán automaticky - chybová hlášení jsou prvním příznakem vad HD.

Hledání místa k parkování

HD zaparkujte vždy předtím, než vypnete počítač. Záznamové a čtecí hlavy plují nad povrchem otáčejícího se HD na vzduchovém polštáři. Když počítač vypnete - ať již úmyslně nebo ne - hlava se položí na citlivý povrch HD. Výsledné tření způsobuje opotřebení a víří mikroskopické částečky. I když není pravděpodobné, že se hlava dotkne vždy stejného místa, větší počet takových vypnutí může poškodit HD a tím i uložená data.

Disky se strunou mívají mechanismus, který při vypnutí počítače automaticky položí hlavu na vyhrazené místo - je to tzv. autopark. Oproti HD, které používají krokové motory pro nastavení polohy hlavy, jsou dražší. Krokový motor totiž nemůže oddálit hlavu při vypnutí, proto je nutné hlavu zaparkovat před každým vypnutím počítače.

Další důvod parkovaní - při přenášení počítače s hlavou spočívající na povrchu HD se všechny pohyby a vibrace transportu na ně přenášejí a hlava poškozuje povrch spolu s daty. Uvolněné částečky povrchu HD pak po jeho spuštění víří a dále poškozují hlavu.

Správné použití parkovacího programu prodlouží životnost HD a podstatně omezí vznik potíží při práci s daty. Ze všech nebezpečí pak zůstává jen problém náhlého vypnutí přívodu proudu. Ale u nových programů už je zajištěno i automatické zálohování dat v předem nastavitelných intervalech.

Jedno upozornění, týkající se parkovacích programů: vždy používejte jen program určený přímo vašemu typu počítače a HD! Například IBM XT Advanced Diagnostic použitý na počítači AT může za určitých podmínek způsobit úplnou ztrátu diskových dat.

Pravidelně kontrolujte, kolik vadných sektorů na HD máte. I ty nejlepší HD obsahují výrobní vady - určitý počet sektorů je navždy nepoužitelný. Takové sektory si DOS zanese do své tabulky během formátování a nedovolí do nich data zapisovat. Bohužel i na nejlepším HD bude počet vadných sektorů časem narůstat. Jistým příznakem blížící se katastrofy je rychlé přibývání vadných sektorů.

Takové katastrofě můžeme předejít právě dodržováním výše uvedených pravidel a také označením částečně poškozených sektorů ještě před zápisem dat. Programy PCTOOLS, NORTON UTILITIES, MACE UTILITIES se vadným sektorům vyhýbají.

Náprava při vymazávání dat

Naučte se obnovovat data omylem "vymazaná", ale na HD stále ještě fyzicky přítomná. Když dáte pokyn k výmazu dat příkazem ERASE operačního systému, data nejsou skutečně vymazána. Pouze bylo přepsáno první písmeno názvu souboru zvláštním znakem a sektory souboru byly označeny jako volné pro zápis dalších dat. Teprve až zápisem nových dat se stará přepíší.

Výše uvedené programy umožňují najít data označená jako vymazaná a obnovit je v automatickém nebo manuálním režimu. Ale i zde mohou být případná omezení.

V automatickém režimu se obnovovací program snaží sloučit sektory do nového souboru. Většinou to provede vyhovujícím způsobem. Při manuálním režimu můžete sami rozhodnout o přiřazení sektorů souborům. Je pravděpodobné, že pokud tak neučinil automatický režim, manuálním to rovněž nezvládneme.

Safety Net přistupuje k obnovování dat jiným způsobem, protože data nevymazává, je tedy snadnější je obnovit. Sídlí v RAM (RAM resident) a většinou se spouští z AUTOEXEC.BAT, zachycujícím příkazy ERASE a DEL operačního systému. Operační systém se přitom přesvědčí, zda k vymazání došlo. Ve skutečnosti je soubor nedotčen a zůstane ukryt a nepoškozen na vámi zvolenou dobu. Jestliže si za tři týdny vzpomenete, že jste omylem vymazali tucet souborů, všechny mohou být rychle a snadno obnoveny pomocí menu Safety Net. Nemusíte přitom doplňovat ani první písmeno názvu souboru nebo mít obavu o chybějící nebo poškozené sektory.

Tento způsob "obnovování" souborů má pouze jednu vadu: u některých programů se "špatným chováním", které nejdříve uvolní paměť RAM, se stane, že zmizí i program Safety Net. Za takových okolností samozřejmě nemůžeme na jeho ochranu spoléhat.

V případě, že chcete některé soubory opravdu vymazat, Safety Net poskytuje i možnost uvolnění místa. A vždy můžete pracovat i s klasickým příkazem DOS operačního systému.

Je dobré být na možnost vlastního omylu při práci s HD předem připraven a naučit se s takovými programy zacházet v normálním i automatickém režimu. Nejprve si to vyzkoušejte na jednom souboru a postupně na dalších. Automatický režim totiž bude obnovovat sousedící sektory, což nemusí souhlasit s rozložením souborů. Pozor také při použití textových editorů. Ty totiž mohou zanechat např. znak uzavírající textový soubor a tento znak může znemožnit obnovení souboru.

Může se stát, že budete nuceni obnovit startovací sektor BOOT (první sektor pevného disku při formátování), který obsahuje malý, ale životně důležitý program pro zavedení operačního systému do paměti RAM. Poškození tohoto záznamu znemožní spuštění systému. Ostatní data HD přitom mohou být nepoškozena. Počítač přivedete k životu systémovou disketou z disketové jednotky. Ale i to je nejisté. Další variantou je přeformátování celého HD. To však znamená napřed archivovat celý HD.

Přestože k poškození sektoru BOOT dojde jen zřídka, stát se to může. Program Mace Utilities obsahuje soubor, který může záznam rekonstruovat za předpokladu, že tuto eventualitu ovládáte.

Stejný program, který umožní obnovení vymazaného souboru na HD, obsahuje dostatek informací k opravení vadného záznamu BOOT.

Když zvolíte RESTORE BOOT z výběru Mace Utilities, další činnost programu je automatická. To je rozhodně lepší, než zálohování celého HD s jeho formátováním a konfigurací systému.

Generální úklid

Náhodné použití příkazu FORMAT dosud vždy znamenalo ztrátu všech dat. Mace Utilities nebo Safety Net Vás před následky takové nehody mohou uchránit. Oba požadují, aby soubor AUTOEXEC.BAT obsahoval i program, který zabezpečí životně důležité informace adresáře ve zvláštním souboru. Formátování s operač-

ním systémem DOS je téměř stejné jako vymazání souboru; data zůstávají, ale zmizí jejich pojmenování. Oba obslužné programy mají zvláštní soubor, který obsahuje informace o adresách a jménech souborů. Na základě těchto informací může být rekonstruován obsah HD včetně posledně aktualizovaných dat.

Tato část programu nepracuje na Bernouli Box HD 20 MB, i když může být použita u verze s kapacitou 10 MB. Na doplnění programu autoři pracují. Safety Net naproti tomu nemůže být použit na žádném typu HD firmy Bernouli Box.

I když uvedené programy mohou být velmi užitečné, pouhé pomyšlení, že si jejich činnost vyzkoušíme na aktivním HD plném dat, může být poněkud nepříjemné. Uvedený soubor programů je samozřejmě jen prvním krokem k dobrému využití HD vašeho počítače. Pečujte o HD jako by to byl ten nejkrásnější automobil, vždyť jeho data mají mnohdy ještě větší cenu.

Programy uvedené kategorie najdete dnes už i mezi programy typu Public Domain (neomezené copyrightem).

Překlad: RNDr.R.Havlík

DEŠIFRÁTOR BASICU PRO ZX SPECTRUM

Většina programů ZX Spectra začíná blokem Basicu, který pak řídí další činnost počítače. Mnoho programátorů tyto úvodní bloky zabezpečuje proti BREAKu, listování, obecně řečeno proti editaci.

Často je účelné přizpůsobit program svým potřebám použití. Pak musíme zjistit, jak je program zatajen. Způsobů zatajení je mnoho. Dešifrovat takovou blokádu často zabere mnoho času. Dešifrátor tuto činnost výrazně usnadňuje. Nejprve přeložte assemblerový program a strojový kód zapiště na kazetu. Programový blok pak kdykoli načtěte příkazy

```
CLEAR 62999: LOAD ""CODE
```

a spusťte příkazem RANDOMIZE USR 63000. Ihned poté se smaže obrazovka a vypíše se požadavek načtení basicové části dešifrovaného programu. Bloky, které nemají basicovou hlavičku, jsou odmítnuty. Pokud jste předtím měli v paměti basicový program, bude vymazán.

Přečtená hlavička Basicu je zobrazena včetně délky bloku a spouštěcí řádky Basicu. Podle údajů z hlavičky Dešifrátor upraví potřebné systémové proměnné a udělá místo pro uložení bloku, který je načten jako bezhlavičkový, proto nedojde k jeho automatickému spuštění.

Po stisku jakéhokoli tlačítka (kromě BREAKu) se aktivují další funkce Dešifrátoru (lze je volat opakovaně příkazem RANDOMIZE USR 63003). Na obrazovce probíhá výpis basicových řádek. Zobrazuje se číslo řádky, její délka a adresa prvního příkazu řádky. Kódy znaků, které způsobovaly ochranu, jsou nahrazeny znakem "." (tečkou). Každé číslo je zobrazeno v hranatých závorkách - tentokrát jeho skutečná hodnota, se kterou počítač pracuje. Na chybnou řádku jsme upozorněni; listování programem můžeme přerušit stiskem BREAKu při dotazu "scroll?".

Ing. Jiří Vacek

```
10 ; ***************************
20 ; *   DESIFRATOR BASICU     *
30 ; ***************************
40
50 ; 13.7.1987 - VERZE 3.1
60 ;       JRM pro AF
70
80         ORG   63000
90
100        JP    LOAD
110        JP    LIST
120
130 TEXT1:
140        DEFB 13
150        DEFM " Adresa: "
160 TEXT9:
170        DEFB 13
180        DEFM "Radek: "
190 TEXT8:
200        DEFB 13,13,13
210        DEFM "Chybna delka "
220        DEFM "nasledujiciho "
230        DEFM "radku"
240        DEFB 13,13
250        DEFM "STOP = BREAK  "
260        DEFM "  ENTER = CONT"
270        DEFB 13,13
280 TEXT1
290        DEFB 13,13
300        DEFM "     Zaloz "
310        DEFM "kazetu s "
320        DEFM "BASICem :"
330        DEFB 13,13
340 TEXT2:
350        DEFM "Nazev: "
360 TEXT3:
370        DEFM "Delka: "
380 TEXT4:
390        DEFM "Start: "
400 TEXT5:
410        DEFM "Basic: "
420 TEXT6:
430        DEFM "CHYBA - DALSI "
440        DEFM "POKUS (BASIC "
450        DEFM "1/1)"
460        DEFB 13
470 TEXT7:
480        DEFB 13,13,13
490        DEFM "     K L A V E"
500        DEFM " S U   !!!"
510 BCPIS:
520        CALL #2D2B
530        CALL #2DE3
540        RET
550 WTKEY:
560        LD    DE,TEXT7
570        LD    BC,BCPIS-TEXT7
580        CALL  #203C
590 WAIT   CALL  #028E
600        INC   E
610        JR    Z,WAIT
620        RET
```

```
DCHHL
        LD    DE,TEXT6
        LD    BC,TEXT7-TEXT6
        CALL  #203C
        JR    HLAVA
ZPRAVA
        LD    DE,TEXT8
        LD    BC,TEXT1-TEXT8
        CALL  #203C
        RET
SMAZ
        CALL  #0D6B
        LD    A,2
        CALL  #1601
      RET

: DESIFRATOR BASICU
: ------------------

LOAD
        CALL  SMAZ
        LD    DE,TEXT1
        LD    BC,TEXT2-TEXT1
        CALL  #203C
NEWBAS
        LD    DE,(#5C53)
        LD    HL,(#5C59)
        DEC   HL
        CALL  #19E5

HLAVA
        LD    IX,BAFF
        LD    DE,#0011
        XOR   A
        SCF
        CALL  #056A
        JR    NC,DCHHL
        LD    A,(BAFF)
        CP    0
        JR    NZ,DCHHL
UKAZ
        LD    DE,TEXT2
        LD    BC,TEXT3-TEXT2
        CALL  #203C
        LD    DE,BAFR+1
        LD    BC,10
        CALL  #203C
        CALL  NL
        LD    DE,TEXT3
        LD    BC,TEXT4-TEXT3
        CALL  #203C
        LD    BC,(BAFR+11)
        CALL  BCPIS
        CALL  NL
        LD    DE,TEXT4
        LD    BC,TEXT5-TEXT4
        CALL  #203C
        LD    BC,(BAFR+13)
        CALL  BCPIS
        CALL  NL
        LD    DE,TEXT5
        LD    BC,TEXT6-TEXT5
        CALL  #203C
        LD    BC,(BAFR+15)
        CALL  BCPIS
MISTO
        LD    HL,(#5C53)
        LD    BC,(BAFR+11)
        CALL  #1655
        LD    HL,(#5C53)
        LD    DE,(BAFR+15)
        ADD   HL,DE
        LD    (#5C4B),HL
        LD    IX,(#5C53)
        LD    DE,(BAFR+11)
        LD    A,#FF
        SCF
        CALL  #056A
        CALL  WTKEY


LIST
        CALL  SMAZ
        LD    HL,(#5C53)
        LD    DE,(#5C4B)
PROH
        PUSH  HL
        PUSH  DE
        LD    DE,(#5C4B)
        AND   A
      SBC   HL,DE
        POP   DE
        POP   HL
        JP    NC,KONEC
        CALL  #19B8
: HL-ADRESA 1.BYTE RADKU
:          (VSTUP I VYSTUP)
: VYSTUPNI PARAMETRY:
: DE - ADRESA 1.BYTE DALSIHO
:      RADKU
: BC - DELKA 1. RADKU
        PUSH DE
        PUSH BC
        PUSH HL
        DEC   DE
        LD    A,(DE)
        CP    #0D
        CALL  NZ,ZPRAVA
CISRAD  LD    A,2
        CALL  #1601
        LD    DE,TEXT9
        LD    BC,8
        CALL  #203C
        POP   HL
        LD    B,(HL)
        INC   HL
        LD    C,(HL)
        INC   HL
        PUSH  HL
        CALL  BCPIS
        CALL  SPACE
        CALL  SPACE
        LD    DE,TEXT7
        LD    BC,7
        CALL  #203C
        POP   HL
        LD    C,(HL)
        INC   HL
        LD    B,(HL)
        INC   HL
        PUSH  HL
        CALL  BCPIS
        LD    DE,TEXT10
        LD    BC,10
        CALL  #203C
        POP   HL
        LD    B,H
        LD    C,L
        PUSH  HL
        CALL  BCPIS
        CALL  NL
        POP   HL
        POP   BC
        DEC   BC
        DEC   BC
        DEC   BC
        DEC   BC
PRIKAZ
        LD    A,(HL)
        CP    13
        JR    Z,PIS
        CP    #0E
        JR    Z,CISLO
        CP    32
        JR    C,TECKA
PIS     RST   #10
DAL     INC   HL
        PUSH  HL
        LD    DE,(#5C4B)
        AND   A
        SBC   HL,DE
        POP   HL
        JR    NC,KONEC
        DEC   BC
        LD    A,C
        OR    B
      JR    NZ,PRIKAZ
        POP   HL
        JP    PROH
TECKA
        LD    A,"."
        JR    PIS
CISLO
        CALL  SPACE
        LD    A,"["
        RST   #10
        INC   HL
        DEC   BC
        PUSH  BC
        PUSH  HL
        CALL  #33B4
        CALL  #2DE3
        LD    A,"]"
        RST   #10
        POP   HL
        POP   BC
        LD    A,4
MINUS
        INC   HL
        DEC   BC
        DEC   A
        JR    NZ,MINUS
        JR    DAL
SPACE
        LD    A,32
        RST   #10
        RET
NL
        LD    A,13
        RST   #10
        RET
KONEC
        RST   #8
        DEFB  #FF
        BAFR  DEFS 17
```

OVLÁDACÍ PROGRAM PRO D-WRITER

Pražská 666. ZO Svazarmu poslední dobou vyvíjí velmi záslužnou činnost v oblasti mikroelektroniky. Její kurs uživatelů osobních počítačů nemá svým rozsahem a kvalitou u nás obdobu. Náplní kursu je výuka ve 4 základních oblastech nasazení osobní výpočetní techniky: textové editory, tabulkové procesory, grafické editory a databáze. K těmto tématům uživatel obdrží kazetu se 4 programy a příslušnou dokumentaci provedenou na profesionální úrovni.

D-WRITER je velmi kvalitní textový editor pro ZX Spectrum. Po všech stránkách výrazně převyšuje Tasword. Do manuálu pro D-WRITER se vloudilo několik nepřesností, které nakonec znemožňují to hlavní - vložit do editoru ovladač tiskárny a provádět tak základní funkci textového editoru - přenos dat na tiskárnu.

Kapitola 6 manuálu - Instalace tiskárny - uvádí postup uložení vlastního ovladače:

1. Nahrát instalaci (příkazem LOAD"")
2. Volbou 'Instalovat' opustit instalaci
3. Klávesou BREAK přerušit program
4. Příkazem LOAD "název" CODE x nahrát ovladač
   x může být jedna z adres E400H, E800H...
5. Spustit program příkazem RUN

Právě bod 5 vrátí uživatele zpět do editoru bez toho, že by došlo ke skutečnému přesunu jím zavedeného ovladače z uvedených adres na adresu pracovní (B300H). Stane se tak proto, že funkce EXTENDED MODE+B (návrat do Basicu) funguje správně pouze v hotovém "holém" D-WRITERu. V jeho instalační verzi končí zaseknutím programu. Pátý bod je nutno změnit na: Vrátit se zpět do instalace příkazem GOTO 40. Poté si v hlavním menu zvolíme správný ovladač, volbou 'Instalovat' opustíme instalaci a můžeme si hotovou instalaci s našimi ovladači či jen "holý" D-WRITER se zavedeným ovladačem uložit na kazetu.

Další velké úskalí spočívá ve vlastním ovladači. Většina uživatelů ZX Spectra používá interfejs několikrát zveřejněný v AR s MHB 8255A, který má jinou adresaci, než je použita v ovladačích nabízených instalačním menu (pro tiskárny D 100, STAR GX10, ROBOTRON a Schneider). Uživateli nezbyde, než sáhnout po manuálu a podle vzorové ukázky začít tvořit vlastní rutinu. K tomu, aby tato snaha byla korunována úspěchem, uvádím několik poznámek.

V ukázce uvedená subrutina pro zjištění stavu tiskárny je spolu se svým popisem nejednoznačná:

```
STATUS    XOR A
          INC A
          RET
```

Je napsána tak, že při dotazu na stav tiskárny vždy odpoví, že je tiskárna připravena. Po RET zůstává v A jednička. Indikátor Z je tedy vynulován, což nesouhlasí s popisem v manuálu, že při Z=0 tiskárna není připravena. Pokusem jsem zjistil, že STATUS hlásí trvalou připravenost tiskárny (Z=0).

Vlastní tisková rutina uvnitř D-WRITERu (vně ovladače) navíc STATUs nevolá důsledně pro každý vyslaný znak. Pak se stává, že se při komunikaci ztrácejí řídící kódy (ESC x), na tiskárně nelze nastavit jiný druh tisku, nebo jsou ignorovány řídící kódy CR, LF a tisknutý text je špatně tabelován. Test stavu tiskárny je nutno provádět již v subrutině pro vyslání znaku (vyšli) tak, jak je to uvedeno v rutině, jejíž výpis přikládám. Tento ovladač používám pro tiskárnu D 100 s interfejsem IRPR. Je beze změny použitelný i pro tiskárny s interfejsem CENTRONICS. Obvod 8255A pracuje v módu 1 (handshaking), potvrzovací signály jdou přes port C - PC7 jako OBFA (data platná, odpovídající vstup do tiskárny je STB), PC6 jako vstup signálu ACK. Rutina má 170 bajtů a je mnohem jednodušší než dodávané ovladače, které kromě rutiny pro Robotron mají délku nad 500 bajtů. Pro tisk češtiny používám v D 100 upravený generátor znaků, takže pro vyslání 1 českého písmena stačí vyslat 1 bajt (kódy C1H..FAH). V ukázce ovladače z manuálu je uvedeno, že místo každého znaku české abecedy je možné vyslat až 4 znaky. Není však jasné, k čemu slouží počáteční jednička v každé čtveřici (viz listing manuálu). Jisté je, že pokud ji neuvedete, pak je tiskárna při tisku českých písmen ignoruje. Zkoušel jsem vysílat posloupnost např. E1H,0,0,0, ale k cíli nevedla. V manuálu jsou též přehozena data pro "ú" a "ů" ve velké i malé sadě písmen. Připomínám, že znak české abecedy, který se má tisknout, je dán pořadím čtveřic dat.

Věřím, že jsem mnoha uživatelům D-WRITERu ušetřil trochu práce, a pokud přehlédnou drobnosti, jako že v menu Instalace nejde navolit první ovladač (D 100) a v menu Řízení tisku je číslo tisknuté stránky o 1 vyšší, než má být, pak věřím, že se D-WRITER stane jedním z jejich nejoblíbenějších programů.

Stanislav Rejman

```
;DRIVER pro tiskarnu D-100
;
;osetrena ceska abeceda
;
;adresace 8255:
PCW     EQU  127
PA      EQU  31
PC      EQU  95
;
        ORG  #B300
        JP   vysli
        JP   status
        JP   init
;-------------------------------
        DEFM "D-100 IRPR"
        DEFB 1,#E1,0,0 ;A
        DEFB 1,#E3,0,0 ;C
        DEFB 1,#E4,0,0 ;D
        DEFB 1,#F7,0,0 ;E'
        DEFB 1,#E5,0,0 ;Ev
        DEFB 1,#E9,0,0 ;I
        DEFB 1,#EE,0,0 ;N
        DEFB 1,#EF,0,0 ;O
        DEFB 1,#F2,0,0 ;R
        DEFB 1,#F3,0,0 ;S
        DEFB 1,#F4,0,0 ;T
        DEFB 1,#F5,0,0 ;U'
        DEFB 1,#EA,0,0 ;Uo
        DEFB 1,#F9,0,0 ;Y
        DEFB 1,#FA,0,0 ;Z
        DEFB 0,0,0,0   ;nic
        DEFB 1,#C1,0,0 ;a
        DEFB 1,#C3,0,0 ;c
        DEFB 1,#C4,0,0 ;d
        DEFB 1,#D7,0,0 ;e'
        DEFB 1,#C5,0,0 ;ev
        DEFB 1,#C9,0,0 ;i
        DEFB 1,#CE,0,0 ;n
        DEFB 1,#CF,0,0 ;o
        DEFB 1,#D2,0,0 ;r
        DEFB 1,#D3,0,0 ;s
        DEFB 1,#D4,0,0 ;t
        DEFB 1,#D5,0,0 ;u'
        DEFB 1,#CA,0,0 ;uo
        DEFB 1,#D9,0,0 ;y
        DEFB 1,#DA,0,0 ;z
        DEFB 0,0,0,0   ;nic
;-------------------------------
vysli   PUSH AF
        OUT  (PA),A
ack     IN   A,(PC)
        AND  08
        JR   Z,ack
        POP  AF
        RET
;-------------------------------
status  XOR  A
        INC  A
        RET
;-------------------------------
init    LD   A,#A0
        OUT  (PCW),A
        LD   A,#0D
        OUT  (PCW),A
        RET
```

KALKULÁTOR ZX SPECTRA (1)

Slovo kalkulátor nám zpravidla vybaví obraz elektronické kalkulačky, více či méně komplikované a dokonalé. U kalkulátoru Spectra je to rozhodně jinak. Jeho podprogramy patří mezi nejvýkonnější v rámci operačního systému Spectra a dokáží mnohem víc, než jen "kalkulovat". Hlavní program CALCULATE je krátký (335BH-33A1H), ale se svými podprogramy (33A1H-386DH) tvoří kompaktní, potentní celek. Organicky jsou k němu přičleněny aritmetické podprogramy (2D4FH-335AH).

Pochopení činnosti a využití kalkulátoru značně rozšiřuje možnosti a variace tvorby vlastních programů ve strojovém kódu. Můžeme ho přivolat pomocí instrukce CALL nebo JP, ale nejpraktičtější je volba instrukce RST 0028H.

Po zadání RST 0028H se okamžitě přivolá program CALCULATE (335BH):

```
0028 FP-CALC    JP 335BH,CALCULATE     Přímý skok
```

CALCULATE je nejuniverzálnější programem romky ZX Spectra. Jeho pomocí můžeme realizovat prakticky všechny funkce, které má Basic ZX Spectra zabudované, včetně rozhodování a skoků. Prostřednictvím kalkulátoru můžeme dělat podmíněné i nepodmíněné skoky, vyvolávat další strojové programy a podprogramy, manipulovat s řetězci, provádět logické operace apod. Podprogram CALCULATE pracuje s pětibajtovými hodnotami, které reprezentují buď číselnou hodnotu nebo parametry řetězce.

Celá činnost kalkulátoru podporovaná činností zásobníku kalkulátoru a kalkulátorových pamětí. Tyto paměti tvoří souvislý, třicetibajtový úsek paměti RAM, rozdělený do šesti pětibajtových bloků, reprezentujících kalkulátorové paměti mem-0 až mem-5. Počáteční adresa této oblasti je uložena v systémové proměnné MEM (5C68H). Při inicializaci systému je počáteční adresa oblasti kalkulátorových pamětí shodná s adresou systémové proměnné MEMBOT (5C92H), jejíž délka je právě 30 bajtů. Změnou adresy v systémové proměnné MEM můžeme definovat novou oblast kalkulátorových pamětí. Je třeba mít na paměti, že některé podprogramy - např. PRINT-FP (2DE3H) - v některých částech používají pro určení adresy mem-0 až mem-5 obsah systémové proměnné MEM, zatímco v jiných částech předpokládají jejich umístění do oblasti MEMBOT. A tak kdyby oblast kalkulátorových pamětí byla posunutá a použili bychom např. podprogram PRINT-FP, takový pokus o zobrazení čísla by dal chybný výsledek.

Adresa začátku zásobníku kalkulátoru je v systémové proměnné STKBOT (5C63H) a adresa jeho vrcholu je v systémové proměnné STKEND (5C63H). Když je zásobník kalkulátoru prázdný, adresy jeho začátku a vrcholu jsou shodné. Zásobník kalkulátoru je dynamickou částí paměti - jeho obsah i adresa jeho vrcholu se často mění. Je to dáno

různými vlivy ROM-ky. Z toho důvodu je správnější používat termín aktuální vrchol zásobníku kalkulátoru, resp. adresa aktuálního vrcholu zásobníku kalkulátoru. Každá nově uložená položka na vrchol jeho zásobníku zvýší adresu vrcholu o hodnotu 5. Když některá z operací vrchol sníží, adresa vrcholu zásobníku kalkulátoru se sníží zase o hodnotu 5. Kalkulátor pracuje s poslední hodnotou uloženou na jeho vrchol a s předposlední hodnotou, uloženou "pod" poslední hodnotou. Pro lepší pochopení mechaniky zásobníku kalkulátoru to vysvětlím názorněji:

Předpokládejme, že počáteční adresa zásobníku kalkulátoru je 6991H a že zásobník kalkulátoru je prázdný. V tom případě je adresa jeho aktuálního vrcholu shodná s adresou jeho začátku. Řekněme, že jsme po sobě do zásobníku kalkulátoru uložili tři hodnoty: AAAAH a FFFFH. Bude to vypadat následovně:

```
(STKBOT) = 6991H - 00  ... počáteční adresa zásob-
           6992H - 00      níku kalkulátoru; v
           6993H - AA      prvních pěti bajtech je
           6994H - AA      uloženo číslo AAAAH
           6995H - 00

           6996H - 00  ... počáteční adresa před-
           6997H - 00      poslední hodnoty, re-
           6998H - 00      prezentující druhé vlo-
           6999H - 00      žené číslo, tj. 0000H
           699AH - 00

           699BH - 00  ... počáteční adresa po-
           699CH - 00      slední hodnoty, v je-
           699DH - FF      jíchž pěti bajtech je
           699EH - FF      uloženo číslo FFFFH
           699FH - 00

(STKEND) = 69A0H ..... adresa aktuálního
                       vrcholu zásobníku
                       kalkulátoru
```

Je též užitečné vědět, že mnoho podprogramů romky adresuje počáteční adresu poslední hodnoty v zásobníku kalkulátoru do reg. HL a adresu aktuálního vrcholu zásobníku kalkulátoru do reg. DE. Ale pozor - není to pravidlem!

Požadované operandy se do zásobníku ukládají buď pomocí některých operací kalkulátoru nebo specializovaných podprogramů romky; např.:

CALL 2D28H,STACK-A - jednobajtové číslo z reg. A změní na jeho pětibajtový ekvivalent a uloží jej na vrchol zásobníku kalkulátoru.

CALL 2AB1H,STK-ST-0 - na vrchol zásobníku kalkulátoru uloží postupně tyto hodnoty:

1. bajt = 00
2. bajt = obsah reg.E
3. bajt = obsah reg.D
4. bajt = obsah reg.C
5. bajt = obsah reg.B

Tento podprogram se zpravidla používá k uložení parametrů řetězce na vrchol zásobníku kalkulátoru. Před jeho přivoláním musí obsahovat reg.DE počáteční adresu příslušného řetězce a reg.BC délku tohoto řetězce.

CALL 2AB6H,STK-STORE - plní tutéž funkci jako předcházející podprogram, s tím rozdílem, že do 1. bajtu nevloží hodnotu +00, ale obsah reg.A.

CALL 33B4H,STACK-NUMBER - na vrchol zásobníku kalkulátoru uloží postupně 5 bajtů, jejichž počáteční adresa je v reg.HL:

1. bajt = (HL)
2. bajt = (HL+1)
3. bajt = (HL+2)
4. bajt = (HL+3)
5. bajt = (HL+4)

CALL 24FBH,SCANNING - před přivoláním podprogramu musí systémová proměnná CH-ADD obsahovat adresu prvního znaku vyhodnocovaného výrazu. Výraz se vyhodnocuje jako posloupnost ASCII kódů, z nichž je složen, a musí být ukončen kódem CR. Když je výsledkem vyhodnocení číslo, uloží se na vrchol zásobníku kalkulátoru. Když je výsledkem vyhodnocení řetězec, na vrchol zásobníku kalkulátoru se uloží jeho parametry, přičemž samotný výsledný řetězec bude uložen do pracovní oblasti.

Pro opačný postup, t.j. přenos poslední hodnoty ze zásobníku kalkulátoru (resp. i předposlední hodnoty) do příslušných registrů, můžeme použít třeba:

CALL 2DD5H,FP-TO-A - floating-point číslo (-FFH až +FFH), uložené jako poslední hodnota, přesune do reg.A - nejdříve se vytvoří absolutní hodnota tohoto čísla a zaokrouhlí se na nejbližší celé číslo.

CALL 2314H,STK-TO-A - provede tutéž operaci jako předcházející podprogram, ale navíc do reg.C uloží hodnotu +01, když bylo číslo kladné, nebo hodnotu +FFH, když bylo číslo záporné.

CALL 1E85H,TWO-PARAM - do reg.A přesune poslední hodnotu ze zásobníku kalkulátoru (-FFH až +FFH). Když je číslo záporné, vytvoří se nejdřív jeho dvojkový doplněk. Současně se do reg. BC přesune číslo uložené jako předposlední hodnota v zásobníku kalkulátoru. Toto číslo musí být nezáporné, v rozpětí 0000 až FFFFH. Prvním krokem při přenosu obou čísel je jejich zaokrouhlení na nejbližší celé číslo.

CALL 2DA2H,FP-TO-BC - zaokrouhlí floating-point číslo, uložené jako poslední hodnota v zásobníku kalkulátoru, vytvoří jeho absolutní hodnotu a uloží je do reg.BC. Číslo musí být v rozpětí od -FFFFH až +FFFFH. Do reg.A se zároveň uloží nižší bajt výsledku.

CALL 2307H,STK-TO-BC - floating-point číslo, tvořící poslední hodnotu v zásobníku kalkulátoru, přenese do reg.B a číslo uložené jako předposlední hodnota, do reg.C. Obě čísla musí být v rozpětí -FFH až +FFH. Prvním krokem je zaokrouhlení čísel a vytvoření jejich absolutní hodnoty. Současně se do reg.E uloží hodnota +01, když bylo číslo, přenášené do reg.C nezáporné, nebo hodnota +FFH, když bylo číslo záporné. Totéž platí pro reg.D při přenášení čísla do reg.B.

CALL 2D7FH,INT-FETCH - přenese floating-point číslo typu 'small integer' (pětibajtové číslo v rozsahu -FFFFH až +FFFFH včetně, uložené do 3. a 4. bajtu v pořadí nižší/vyšší bajt), tvořící poslední hodnotu v zásobníku kalkulátoru, do reg.DE

Pozn.: Před přivoláním podprogramu musí reg.HL obsahovat počáteční adresu přenášeného čísla!

CALL 2BF1H,STK-AEDCB - jednotlivé bajty poslední hodnoty v zásobníku kalkulátoru se přenesou následovně:

1. bajt do reg.A
2. bajt do reg.E
3. bajt do reg.D
4. bajt do reg.C
5. bajt do reg.B

Tento podprogram se používá jako příprava k zobrazení řetězce, jehož parametry jsou uložené jako poslední hodnota v zásobníku kalkulátoru. Po jeho ukončení bude reg.DE obsahovat počáteční adresu příslušného řetězce a reg.BC jeho délku. Řetězec můžeme potom zobrazit pomocí instrukce CALL 203CH,PR-STRING.

CALL 2DE3H,PRINT-FP - do aktuální PRINT-pozice zobrazí číslo uložené v zásobníku kalkulátoru jako poslední hodnota. Rozsah a formát zobrazovaného

čísla se řídí stejnými pravidly, jaké známe z Basicu Spectra. Po ukončení podprogramu číslo ze zásobníku kalkulátoru "zmizí" a vrchol zásobníku kalkulátoru se sníží o 5.

Operace kalkulátoru můžeme rozdělit do čtyř základních skupin:

1. manipulační operace - např. vzájemná záměna poslední a předposlední hodnoty, "zdvojení" poslední hodnoty apod.
2. binární operace - operace s poslední a předposlední hodnotou (sčítání, odečítání, násobení apod.)
3. unární operace - operace s poslední hodnotou; např. výpočet exponenciální funkce, logaritmu, negace atd.
4. jiné operace - např. rozhodovací operace, výpočet Čebyševových polynómů apod.

Operace, které bude kalkulátor vykonávat, jsou určené definovanými bajty za instrukcí RST 0028H. Tyto bajty - tzv. literály - jsou buď ofsety určující podprogram požadované operace, nebo parametry požadované operace. Posledním definovaným bajtem musí být vždy 38H, což je signál pro ukončení výpočtu. Následující bajt se bude interpretovat už jako instrukce mikroprocesoru. Některé operace však nelze realizovat zadáním literálu v režimu kalkulátoru (např. 'val', 'e-to-fp'). Realizujeme je tak, že reg.B obsadíme jejich ofsetem a následně vložíme CALL "adresa příslušného podprogramu". Jiné operace sice realizujeme přes kalkulátor, ale před přivoláním kalkulátoru musíme reg.B obsadit jejich ofsetem (např. rozhodovací operace). Kalkulátor můžeme pochopitelně přivolat i přímo, pomocí instrukce CALL 335BH,CALCULATE. Vstup pomocí CALL 335EH, GEN-ENT-1 se používá tehdy, používáme-li reg.B jako čítač (následně se ovšem hodnota čítače přesune do systémové proměnné BREG, která převezme funkci "čítače"). Vstupní bod pomocí CALL 3362H,GEN-ENT-2 se používá tehdy, když se hodnota čítače uloží do systémové proměnné BREG přímo (resp. tehdy, když se požaduje, aby se obsah systémové proměnné BREG programem CALCULATE nezměnil).

Pro komentovaný popis činnosti kalkulátoru se běžně používá následující formát:

```
adresa  RST 0028H,FP-CALC    07          komentář
```

Manipulační operace kalkulátoru

```
offset, název          činnost
-------------------------------------------
```

```
        DEFB 31H,duplicate   07,07
        DEFB 38H,end-calc    07,07
```

V uvedeném příkladu se předpokládá, že před přivoláním kalkulátoru byla na jeho vrchol uložena hodnota 07. Definovaný bajt 31H je ofsetem operace 'duplicate', která "zdvojí" vrchol zásobníku kalkulátoru. Adresa aktuálního vrcholu se zvýší o 5 bajtů. Ve formátu výpisu se "narůstání" vrcholu znázorňuje vždy zleva doprava (viz šipka). Literály se zpravidla uvádějí hexadekadicky.

A0,stk-zero uloží na vrchol hodnotu 0
A1,stk-one uloží na vrchol hodnotu 1
A2,stk-half uloží na vrchol hodnotu 1/2
A3,stk-pi/2 uloží na vrchol hodnotu PI/2
A4,stk-ten uloží na vrchol hodnotu 10 d

C0,st-mem-0 kopíruje posl.hodn.do kalk.paměti 0
C1,st-mem-1 kopíruje posl.hodn.do kalk.paměti 1
C2,st-mem-2 kopíruje posl.hodn.do kalk.paměti 2
C3,st-mem-3 kopíruje posl.hodn.do kalk.paměti 3
C4,st-mem-4 kopíruje posl.hodn.do kalk.paměti 4
C5,st-mem-5 kopíruje posl.hodn.do kalk.paměti 5
Pozn.: při operacích C0 až C5 se obsah zásobníku kalkulátoru ani adresa jeho vrcholu nemění.

E0,get-mem-0 kopír.číslo z kalk.pam.0 na vrchol
E1,get-mem-1 kopír.číslo z kalk.pam.1 na vrchol
E2,get-mem-2 kopír.číslo z kalk.pam.2 na vrchol
E3,get-mem-3 kopír.číslo z kalk.pam.3 na vrchol
E4,get-mem-4 kopír.číslo z kalk.pam.4 na vrchol
E5,get-mem-5 kopír.číslo z kalk.pam.5 na vrchol

Pozn.: Při operacích E0 až E5 se obsah kalkulátorých pamětí nezmění. Vrchol zásobníku kalkulátoru se zvýší o 5. Při používání kalkulátorových pamětí ve vlastních strojových programech je třeba určité opatrnosti, protože některé kalkulátorové operace tyto paměti (mem-0 až mem-3) využívají (např. pro výpočet trigonometrických funkcí, umocňování, 'get-argt'). Z ostatních podprogramů romky používají tyto paměti hlavně:

```
PRINT-FP (2DE3H)        - mem-3 až mem-5
BEEP (03F8H)            - mem-0
příkaz DRAW             - všechny
příkaz CIRCLE           - všechny
příkaz FOR-NEXT         - mem-0 až mem-2
```

01,exchange - vzájemná záměna poslední a předposlední hodnoty zásobníku kalkulátoru.

02,delete - sníží adresu aktuálního vrcholu zásobníku kalkulátoru o 5 ("odstraní" tak poslední hodnotu; novou poslední hodnotou se stává původně předposlední hodnota).

**19H,usr-$** - před přivoláním operace musejí být jako poslední hodnota uloženy parametry řetězce X$. Řetězec X$ musí být jednoznakový; povolená ASCII hodnota znaku je 41H-55H (písmena "A" až "U"), 61H-75H (písmena "a" až "u") a 90H-A4H (kódy UDG písmen "a" až "u"). Výsledkem, který nahradí původní poslední hodnotu, je adresa prvního z osmi bajtů, které reprezentují příslušný UDG znak. Např. když je systémová proměnná UDG nastavena na adresu 6000H a před přivoláním operace bude poslední hodnotou zásobníku kalkulátoru 91H (ASCII kód UDG znaku "b"), bude výsledkem 6008H, což je adresa prvního z osmi bajtů, které vytvářejí znak "b" v oblasti UDG.

1AH,read-in - snímá kanál přes přenosovou linku, jejíž číslo je uloženo v reg.A. Výsledkem je řetězec, jehož parametry se uloží na vrchol zásobníku kalkulátoru. Když vybraný kanál neposkytl žádný signál, výsledkem bude prázdný řetězec.

2BH,peek - před přivoláním operace musí být jako poslední hodnota v zásobníku kalkulátoru adresa v rozpětí 0000H-FFFFH. Po provedení operace nahradí původní poslední hodnotu jednobajtový obsah zadané adresy.

2CH,in - snímá vstupní port, jehož adresa je uložena jako poslední hodnota v zásobníku kalkulátoru. Sejmutá jednobajtová hodnota nahradí původní poslední hodnotu.

31H,duplicate - "zdvojnásobí" poslední hodnotu vytvořením její kopie na aktuální vrchol zásobníku kalkulátoru, jehož adresa se zvýší o 5.

3DH,re-stack - transformuje 5-bajtové číslo typu 'small integer' na jeho ekvivalent typu 'full floating-point'. Transformovaná hodnota přepíše původní poslední hodnotu. Adresa aktuálního vrcholu zásobníku kalkulátoru se nezmění.

MUDR.Peter Ziman,CSc., Ing.Stanislav Kmec

(pokračování)

UNIVERZÁLNÍ INTERFACE MIREK

Doba vášnivého opojení Manic Minery, Jetapcy, Under Worldy apod. nenávratně minula. Osobní počítače se stále častěji využívají jako pomocníci v zaměstnání nebo v domácnosti při tvorbě a úpravách textů, kreslení obrázků, při vedení domácí agendy. K těmto aplikacím, ale i k řadě dalších nutně potřebujeme tiskárnu.

To nebývá tak neřešitelným problémem - nevlastníte-li ji sami, pak ji máte v zaměstnání, v klubu vědeckotechnické činnosti mládeže, kabinetu elektroniky Svazarmu nebo u přítele. Daleko větší problémy vznikají při připojení tiskárny k počítači a potřebné úpravě stávajícího programového vybavení.

Proto aktiv střediska pro mládež a elektroniku ÚV SSM společně s Mikrokomputer klubem 666. ZO Svazarmu (pošt. schránka 64, 169 00 Praha 6) pro vás připravil metodický materiál o připojování periferních zařízení k počítači ZX Spectrum. Materiál má název Univerzální interfejs MIREK.

Ve své první části se zabývá popisem počítače jako celku, poté podrobněji seznamuje s jeho zdroji. Protože se ve druhé části zabývá připojováním periferních zařízení, musíte znát volné adresové pozice vstupních a výstupních obvodů. Těm je věnována další kapitola. Na závěr popisu počítače Spectrum podrobně seznamuje s jeho hranovým konektorem.

V další části naznačuje způsob, jak připojovat periferní obvody v československých podmínkách. Popisuje konstrukci univerzální periferní stavebnice MIREK k mikropočítači ZX Spectrum. Stavebnice proto, že ne každému se hodí nebo ne každý má chuť stavět zařízení, které prakticky nevyužije. Nabízí vám tedy možnost postavit si jenom tu část, kterou budete potřebovat.

Pro ty, kdož metodický materiál "Univerzální interfejs MIREK" již nesehnali, uvádím velmi stručně jeho popis. Celková koncepce je prostá. Mirek se skládá ze dvou základních částí - redukce a vlastních periferních obvodů. Takto členěná konstrukce umožňuje použití nejen snadno dostupných konektorů, ale i stavbu účelových a tudíž jednoduchých periferních desek. Metodický materiál popisuje konstrukci redukce, dvou periferních desek pro připojování vnějších zařízení po paralelních linkách a jejich aplikace, připojení křížového ovladače (joystick), myši a několika typů tiskáren.

1. Redukce

Redukce má za úkol převést signály z přímého konektoru WK46580 s roztečí kontaktů 1/10" (2.54 mm) na dvojici konektorů WK46512.

Základní technické údaje redukce:

- Dvě lineárně propojené pozice pro vnější obvody
- Priorita přerušení určena pozicí desky v redukci
- Jedna pozice s úplnou množinou signálů
- Napájecí zdroj 5V/500 mA
- Možnost vestavění tlačítka RESET
- Možnost připojení i k počítačům SHARP, Schneider, MSX a Sord
- Rozměry redukce (š,v,h) 97,5 mm,50 mm,82,5 mm

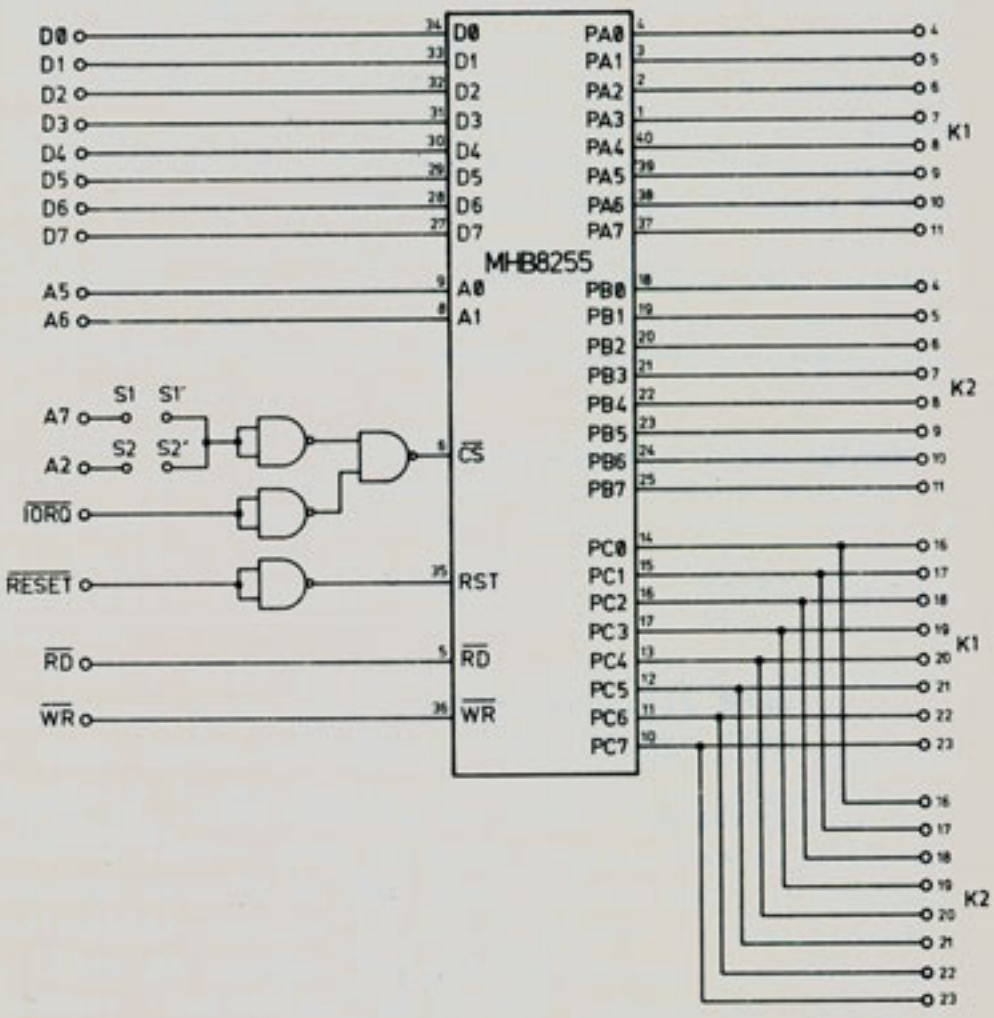
2. Deska PIO-1

Deska PIO-1 slouží k připojování periferních zařízení po paralelní osmibitové lince. Současně automaticky generuje všechny signály pro řízení přenosu dat mezi deskou a příslušnou periférií (handshake). Přenos dat mezi deskou a počítačem je možno řídit pomocí přerušení.

Základní technické údaje desky:

- Dvě na sobě nezávislé obousměrné osmibitové brány
- Čtyři druhy provozu u každé z nich:
  - Výstup bajtu
  - Vstup bajtu
  - Obousměrná sběrnice (pouze u brány A)
  - Vstup a výstup bitu
- Ve všech druzích provozu lze řídit přenos dat pomocí přerušení
- Automatické generování sektoru přerušení
- Výběr desky adresami A7 nebo A2
- Všechny vstupy a výstupy jsou plně kompatibilní s TTL logikou
- Jedno napájecí napětí 5V a jednofázové hodiny
- Rozměry desky (šířka, výška): 87,5 mm, 110 mm

Obě výstupní brány jsou vyvedeny na konektor WK46512 (tentýž jako je použit v redukci). Obsazení jeho kontaktů je následující:



Obr.1 Schéma zapojení desky PIO-2

kontakt č.	signál	charakteristika	tiskárna
1	+5V	napájení	
2	ASTR	vstup, aktivní 0	
3	ARDY	výstup, aktivní 1	
4	A0	V/V, 3 st.	Výstup/D0
5	A1	V/V, 3 st.	Výstup/D1
6	A2	V/V, 3 st.	Výstup/D2

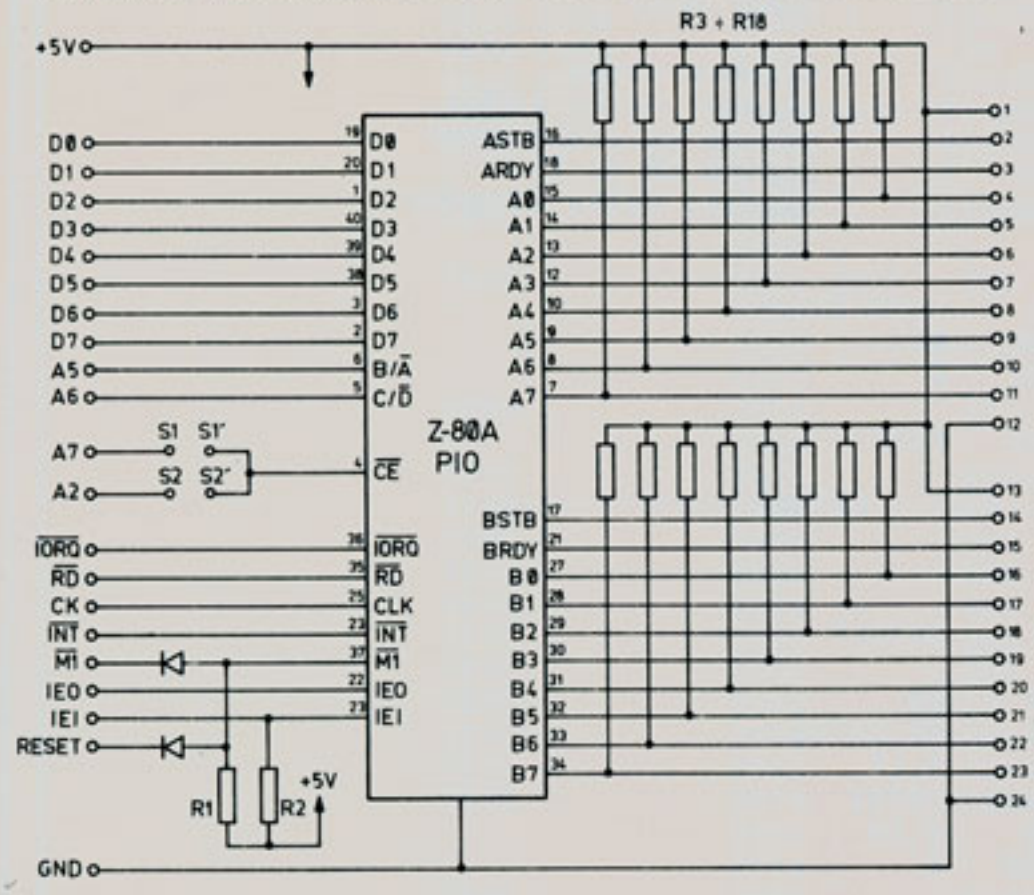
7	A3	V/V, 3 st.	Výstup/D3
8	A4	V/V, 3 st.	Výstup/D4
9	A5	V/V, 3 st.	Výstup/D5
10	A6	V/V, 3 st.	Výstup/D6
11	A7	V/V, 3 st.	Výstup/D7
12	GND	napájení	GND
13	+5V	napájení	
14	BSTR	vstup, aktivní 0	
15	BRDY	výstup, aktivní 1	
16	B0	V/V, 3 st.	Vstup/Busy
17	B1	V/V, 3 st.	Vstup/Error
18	B2	V/V, 3 st.	
19	B3	V/V, 3 st.	
20	B4	V/V, 3 st.	
21	B5	V/V, 3 st.	
22	B6	V/V, 3 st.	
23	B7	V/V, 3 st.	Výstup/Strobe
24	GND	napájení	

zjistíte, že pro obě desky vystačíte pro stejnou tiskárnu s jediným kabelem!

Další důležitou záležitostí je adresace desek. Protože lze připojit desky dvě, má každá z nich možnost volby adresové pozice. Následující tabulka uvádí adresy jednotlivých bran v závislosti na drátové propojce určující pozici desky v adresovém prostoru.

spojka	brána A		brána B		řídící slovo A		řídící slovo B	
	Hex	Dek	Hex	Dek	Hex	Dek	Hex	Dek
S1 - S1'	1F	31	3F	63	5F	95	7F	127
S2 - S2'	9B	155	BB	187	DB	219	FB	251

První tři sloupečky tabulky definují obecné zapojení výstupního konektoru desky. Třetí ukazuje jednu z aplikací. Jde o zapojení kabelu pro tiskárnu s rozhraním Centronics. Podíváte-li se na zapojení konektoru desky PIO-2 s obvodem MHB8255,



Obr.2 Schéma zapojení desky PIO-1

3. Deska PIO-2

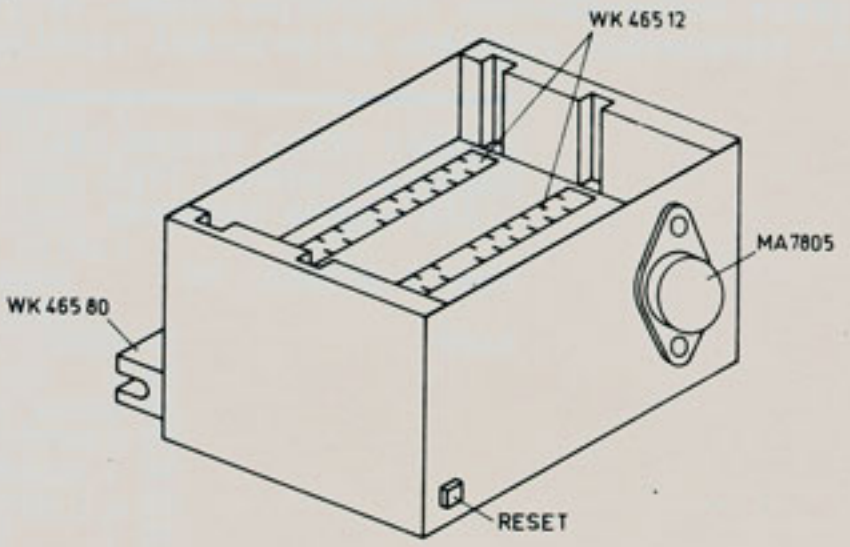
Deska PIO-2 slouží k připojování periferních zařízení po paralelní osmibitové lince. Obsahuje tři univerzální brány, které mohou být naprogramovány jako vstupní nebo výstupní.

Základní technické údaje desky:

- Tři obousměrné osmibitové brány
- Tři druhy provozu: jednoduché vstupy a výstupy
  potvrzované vstupy a výstupy
  obousměrná sběrnice
- Výběr desky adresami A7 nebo A2
- Všechny vstupy a výstupy jsou plně kompatibilní s TTL logikou
- Jedno napájecí napětí 5V
- Rozměry desky (šířka, výška): 87,5 mm, 110 mm

I tato deska má možnost volby adresové pozice. Zde jsou opět konkrétní adresy jednotlivých bran desky PIO-2 pro obě možné polohy drátové propojky.

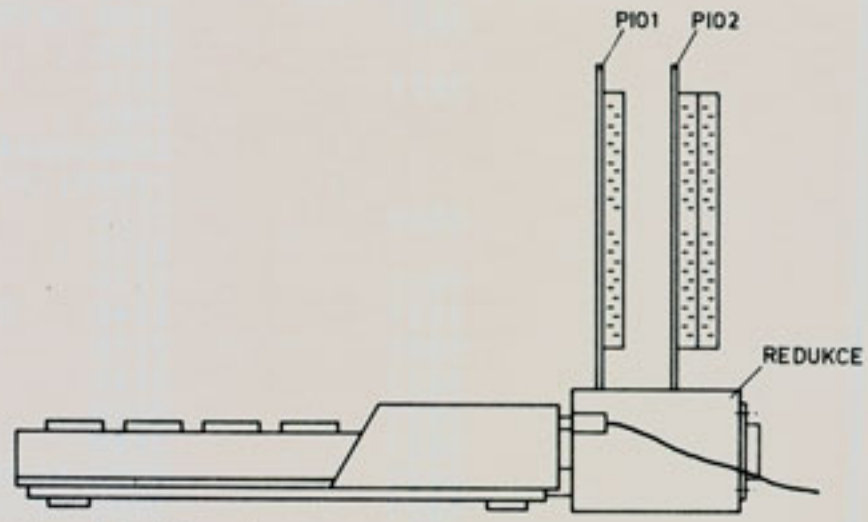
spojka	brána A		brána B		brána C		říd.slovo	
	Hex	Dec	Hex	Dec	Hex	Dec	Hex	Dec
S1 - S1'	1F	31	3F	63	5F	95	7F	127
S2 - S2'	9B	155	BB	187	DB	219	FB	251



Obr.3 Axonometrický pohled na redukci zezadu

Tři univerzální brány A, B a C jsou vyvedeny na dva konektory WK46512 tak, aby kabel k libovolné periferii byl připojitelný k oběma deskám PIO-1 i PIO-2. Zapojení konektorů je následující:

kontakt č.	konektor A	konektor B	charakteristika
1	+5V	+5V	napájení
2	NC	NC	
3	NC	NC	
4	PA0	PB0	V/V, 3 st.
5	PA1	PB1	V/V, 3 st.
6	PA2	PB2	V/V, 3 st.
7	PA3	PB3	V/V, 3 st.
8	PA4	PB4	V/V, 3 st.
9	PA5	PB5	V/V, 3 st.
10	PA6	PB6	V/V, 3 st.
11	PA7	PB7	V/V, 3 st.
12	GND	GND	napájení
13	+5V	+5V	napájení
14	NC	NC	
15	NC	NC	
16	PC0	PC0	V/V, 3 st.
17	PC1	PC1	V/V, 3 st.
18	PC2	PC2	V/V, 3 st.
19	PC3	PC3	V/V, 3 st.
20	PC4	PC4	V/V, 3 st.
21	PC5	PC5	V/V, 3 st.
22	PC6	PC6	V/V, 3 st.
23	PC7	PC7	V/V, 3 st.
24	GND	GND	napájení



Obr.4 Celkový pohled na ZX Spectrum s redukcí a deskami PIO-1 a PIO-2 zboku

Metodický materiál, který vydal Mikrokomputer klub 666. ZO Svazarmu, má za cíl vysvětlit uživatelům mikropočítače ZX Spectrum, že i tento malý počítač lze využívat stejně jako jeho větší kolegy. Na první pohled se zdá nesmyslné, chtít od počítače, který byl od svého počátku určen malým dětem na hraní, něco víc. Podle odhadů je však v ČSSR zastoupen asi 150000 kusy, což je největší množství počítačů jednoho typu u nás vůbec. Československé počítače se s tímto počtem nemohou vůbec měřit.

Kromě tohoto metodického materiálu se Mikrokomputer Klub 666. ZO Svazarmu zaměřuje i na další aplikace Univerzálního interfejsu MIREK. Je jím např. zpracování problému připojování rychlých vnějších pamětí. Řešení tohoto problému ukazuje materiál ZX-FLOPPY. Jedná se o podrobný popis a stavební návod ke konstrukci řadiče floppy diskové jednotky s obvodem I8272 a úpravy ZX Spectra na operační systém CP/M autora ing. Tomáše Krejči. Tento metodický materiál můžete získat na dobírku na výše uvedené adrese.

Dále popisované programové vybavení řeší problém okamžité aplikace Univerzálního interfejsu MIREK v připojování několika typů tiskáren (Epson, Star, DZM180, Consul 2111, MPT 80 a další) a ukazuje řízení elektronické myši pod přerušením. Toto programové vybavení vychází z dlouhodobých zkušeností autorů při práci s Univerzálním interfejsem MIREK.

Ing.Petr Šimůnek, Ing.Daniel Jenne

```
*HISOFT GENS3 ASSEMBLER*
Copyright HISOFT 1983
  All rights reserved

Pass 1 errors: 00

9689                10 XXXX   EQU   $
                    20 ;
                    30 ;
                    40 ;=======================================
                    50 ;     Univerzalni ovladac tiskarny
                    60 ;
                    70 ;            M I R E K
                    80 ;
                    90 ;          (c)DaJe, 1987
                   100 ;=======================================
                   110 ;
                   120 ;
                   130 ;
                   140 ;
                   150 ;---------------------------------------
                   160 ;
0000               170 PIO    EQU   0     ; 1=Z80 PIO   0=I8255
                   180 ;
0000               190 DZM180 EQU   0     ; typ tikarny
0001               200 CENRON EQU   1
0000               210 CONSUL EQU   0
                   220 ;
0001               230 ADR2   EQU   1     ; 1=A2  0=A7
                   240 ;
0000               250 TOKEN  EQU   0     ; klicova slova
0000               260 LINE   EQU   0     ; hlidani delky radku
0000               270 UPPER  EQU   0     ; prevod malych pismen
0001               280 CRTOLF EQU   1     ; prevod CR na LF
0001               290 PAGE   EQU   1     ; hlidani delky stranky


5B18 32445B       1880 PAGE1  LD    (NUMLIN),A
5B1B F1           1890        POP   AF
5B1C              1900 PAGE2  EQU   $
                  1910 ;
5B1C              1920        END   PAGE
                  1930 ;
                  1940 ;--------------- CRTOLF -----------
                  1950 ;
                  1960
                  1970 ;
0001              1980        IF    CRTOLF
                  1990 ;
5B1C FE0D         2000        CP    CR
5B1E 2007         2010        JR    NZ,CRTO1
5B20 3E0A         2020        LD    A,LF
5B22 CD275B       2030        CALL  CRTO1
5B25 3E0D         2040        LD    A,CR
5B27              2050 CRTO1  EQU   $
                  2060 ;
5B27              2070        END   CRTOLF
                  2080 ;
                  2090 ;-------------- CONSUL ---------------
                  2100 ;
0000              2110        IF    CONSUL
                  2120 ;
5B27              2130        PUSH  AF
5B27              2140        OUT   (DATA),A
5B27              2150        IN    A,(CTRL)
5B27              2160        AND   STROBE!-1
5B27              2170        OUT   (CTRL),A
5B27              2180        IN    A,(CTRL)
5B27              2190        AND   ERROR
5B27              2200        JR    NZ,EXIT
5B27              2210 WBUSY1 IN    A,(CTRL)
5B27              2220        AND   BUSY
5B27              2230        JR    Z,WBUSY1
```

```
5B27                2240          IN    A,(CTRL)
5B27                2250          OR    STROBE
5B27                2260          OUT   (CTRL),A
5B27                2270 WBUSY2   IN    A,(CTRL)
5B27                2280          AND   BUSY
5B27                2290          JR    NZ,WBUSY2
5B27                2310          AND   STROBE!-1
5B27                2320          OUT   (CTRL),A
5B27                2330 EXIT     POP   AF
5B27                2340          RET
                    2350 ;
5B27                2360          END   CONSUL
                    2370 ;
                    2380 ;-------------- DZM180 ------------
                    2390 ;
0000                2400          IF    DZM180
                    2410 ;
5B27                2420          PUSH  AF
5B27                2430          CPL
5B27                2440          OUT   (DATA),A
5B27                2450          IN    A,(CTRL)
5B27                2460          AND   STROBE!-1
5B27                2470          OUT   (CTRL),A
5B27                2480          IN    A,(CTRL)
5B27                2490          AND   ERROR
5B27                2500          JR    Z,EXIT
5B27                2510 WBUSY    IN    A,(CTRL)
5B27                2520          AND   BUSY
5B27                2530          JR    NZ,WBUSY
5B27                2540          IN    A,(CTRL)
5B27                2550          OR    STROBE
5B27                2560          OUT   (CTRL),A
5B27                2570 EXIT     POP   AF
5B27                2580          RET
                    2590 ;
5B27                2600          END   DZM180
                    2610 ;
                    2620 ;---------- CENTRONICS -------------
                    2630 ;
0001                2640          IF    CENRON
                    2650 ;
5B27 F5             2660          PUSH  AF
5B28 D39B           2670          OUT   (DATA),A
5B2A DBDB           2680 WBUSY    IN    A,(CTRL)
5B2C E601           2690          AND   BUSY
5B2E 20FA           2700          JR    NZ,WBUSY
5B30 DBDB           2710          IN    A,(CTRL)
5B32 E67F           2720          AND   STROBE!-1
5B34 D3DB           2730          OUT   (CTRL),A
5B36 F680           2740          OR    STROBE
5B38 D3DB           2750          OUT   (CTRL),A
5B3A F1             2760 EXIT     POP   AF
5B3B C9             2770          RET
                    2780 ;
5B3C                2790          END   CENRON
                    2800 ;
5B3C                2810 INIT     EQU   $
                    2820 ;
                    2830 ;--------------- PIO -------------
                    2840
0000                2850          IF    PIO
                    2860 ;
5B3C                2870          PUSH  AF
5B3C                2880          LD    A,-1
5B3C                2890          OUT   (PIOAC),A
5B3C                2900          XOR   A
5B3C                2910          OUT   (PIOAC),A
5B3C                2920          DEC   A
5B3C                2930          OUT   (PIOBC),A
5B3C                2940          LD    A,%00001111
5B3C                2950          OUT   (PIOBC),A
5B3C                2960          POP   AF
5B3C                2970          RET
                    2980 ;
                    2990 ;
5B3C                3000          ELSE
                    3010 ;
5B3C F5             3020          PUSH  AF
5B3D 3E83           3030          LD    A,%10000011
5B3F D3FB           3040          OUT   (CSW),A
5B41 F1             3050          POP   AF
5B42 C9             3060          RET
                    3070 ;
5B43                3080          END   PIO
                    3090 ;
                    3100 ;
                    3110 ;
                    3120 ;
5B43 50             3130 NUMCHR   DEFB  MAXLEN
5B44 32             3140 NUMLIN   DEFB  MAXLIN
                    3150 ;
                    3160 ;
                    3170 ;
```

```
C48C            3180 YYYY   EQU   $-XXXX

Pass 2 errors: 00

ADR2    0001    BUSY    0001
CENRON  0001    CONSUL  0000
CR      000D    CRTO1   5B27
CRTOLF  0001    CSW     00FB
CTRL    00DB    DATA    009B
DZM180  0000    ERROR   0002
EXIT    5B3A    FF      000C
INIT    5B3C    LF      000A
LINE    0000    LO      5B06
MAXLEN  0050    MAXLIN  0032
NUMCHR  5B43    NUMLIN  5B44
PA      009B    PAGE    0001
PAGE1   5B18    PAGE2   5B1C
PB      00BB    PC      00DB
PIO     0000    PIOAC   00DB
PIOAD   009B    PIOBC   00FB
PIOBD   00BB    STROBE  0080
TOKEN   0000    UPPER   0000
WBUSY   5B2A    XXXX    9689
YYYY    C4BC

Table used:   455  from  2000
```

```
                300 ;
                310 ;
                320 ;----------------------------------------
                330 ;
000D            340 CR      EQU   13
000A            350 LF      EQU   10
000C            360 FF      EQU   12
                370 ;
0050            380 MAXLEN  EQU   80    ; delka radky (LINE)
0032            390 MAXLIN  EQU   50    ; delka stranky (PAGE)
                400 ;
0001            410 BUSY    EQU   1     ; definice umisteni
0002            420 ERROR   EQU   2     ;  ridicich signalu
0080            430 STROBE  EQU   X80
                440 ;
                450 ;---------------- ADR2 ---------------
                460 ;
0001            470         IF    ADR2
                480 ;
009B            490 PIOAD   EQU   X9B
00BB            500 PIOBD   EQU   XBB
00DB            510 PIOAC   EQU   XDB
00FB            520 PIOBC   EQU   XFB
                530 ;
009B            540 PA      EQU   X9B
00BB            550 PB      EQU   XBB
00DB            560 PC      EQU   XDB
00FB            570 CSW     EQU   XFB
                580 ;
009B            590 DATA    EQU   X9B
00DB            600 CTRL    EQU   XDB
                610 ;
9689            620         ELSE
                630 ;
9689            640 PIOAD   EQU   X1F
9689            650 PIOBD   EQU   X3F
9689            660 PIOAC   EQU   X5F
9689            670 PIOBC   EQU   X7F
                680 ;
9689            690 PA      EQU   X1F
9689            700 PB      EQU   X3F
9689            710 PC      EQU   X5F
9689            720 CSW     EQU   X7F
                730 ;
9689            740 DATA    EQU   X1F
9689            750 CTRL    EQU   X3F
                760 ;
9689            770         END   ADR2
                780 ;
                790 ;----------------------------------------
                800 ;
5B00            810         ORG   X5B00
                820 ;
5B00 C3065B     830         JP    LO
5B03 C33C5B     840         JP    INIT
                850 ;
                860 ;----------------------------------------
                870 ;
                880 ;
5B06            890 LO      EQU   $
                900 ;
                910 ;---------------- TOKEN -----------
                920 ;
0000            930         IF    TOKEN
                940 ;
```

```
5B06                950         PUSH  HL
5B06                960         PUSH  DE
5B06                970         PUSH  BC
5B06                980         PUSH  AF
5B06                990         CP    X80
5B06               1000         JR    NC,GRAF
5B06               1010 KONEC   CALL  OUTA
5B06               1020 KONEC1  POP   AF
5B06               1030         POP   BC
5B06               1040         POP   DE
5B06               1050         POP   HL
5B06               1060         RET
                   1070 ;
5B06               1080 GRAF    CP    XA5
5B06               1090         JR    NC,KEY
5B06               1100         LD    A," "
5B06               1110         JR    KONEC
                   1120 ;
5006               1130 KEY     SUB   XA5
5B06               1140         LD    DE,X95
5B06               1150         PUSH  AF
5B06               1160         CALL  XC41
5B06               1170         JR    C,PRIN
5B06               1180         LD    A," "
5B06               1190         CALL  OUTA
5B06               1200 PRIN    LD    A,(DE)
5B06               1210         AND   X7F
5B06               1220         CALL  OUTA
5B06               1230         LD    A,(DE)
5B06               1240         INC   DE
5B06               1250         ADD   A,A
5B06               1260         JR    NC,PRIN
5B06               1270         POP   DE
5B06               1280         CP    X48
5B06               1290         JR    Z,TRSP
5B06               1300         CP    X82
5B06               1310         JR    C,KONEC1
5B06               1320 TRSP    LD    A,D
5B06               1330         CP    3
5B06               1340         JR    C,KONEC1
5B06               1350         LD    A," "
5B06               1360         JR    KONEC
                   1370 ;
5B06               1380 OUTA    EQU   $
                   1390 ;
5B06               1400         END   TOKEN
                   1410 ;
                   1420 ;---------- LINE ----------------
                   1430 ;
0000               1440         IF    LINE
                   1450 ;
5B06               1460         PUSH  AF
5B06               1470         LD    A,(NUMCHR)
5B06               1480         INC   A
5B06               1490         LD    (NUMCHR),A
5B06               1500         CP    MAXLEN
5B06               1510         JR    NZ,NOLF
5B06               1520         XOR   A
5B06               1530         LD    (NUMCHR),A
5B06               1540         LD    A,CR
5B06               1550         CALL  OUTA1
5B06               1560 NOLF    POP   AF
                   1570 ;
5B06               1580 OUTA1   EQU   $
                   1590 ;
5B06               1600         END   LINE
                   1610 ;
                   1620 ;---------- UPPER --------------
                   1630 ;
0000               1640         IF    UPPER
                   1650 ;
5B06               1660         CP    "§"
5B06               1670         JR    C,NORM1
5B06               1680         CP    " "+1
5B06               1690         JR    NC,NORM1
5B06               1700         SUB   32
5B06               1710 NORM1   EQU   $
                   1720 ;
5B06               1730         END   UPPER
                   1740 ;
                   1750 ;---------- PAGE --------------
                   1760 ;
0001               1770         IF    PAGE
                   1780 ;
5B06 FE0D          1790         CP    CR
5B08 2012          1800         JR    NZ,PAGE2
5B0A F5            1810         PUSH  AF
5B0B 3A445B        1820         LD    A,(NUMLIN)
5B0E 3D            1830         DEC   A
5B0F 2007          1840         JR    NZ,PAGE1
5B11 3E0C          1850         LD    A,FF
5B13 CD1C5B        1860         CALL  PAGE2
5B16 3E32          1870         LD    A.MAXLIN
```

CPU VERSUS PAMĚŤ

Odběratelé měsíčníku BYTE byli obšťastněni 13. číslem coby prémií za své celoroční předplatné. Toto zvláštní vydání je takřka cele věnováno problematice IBM PC. Mark L. Van Name se v článku Keeping up with the CPU zabývá komunikací CPU s vnitřní pamětí počítačů:

Všechny inzeráty na počítače s 32-bitovým CPU 80386 vytrubují dvě věci - cenu a rychlost procesoru. Napřed to bylo 16, pak 20, nyní už 25 MHz. Jak se ale můžete přesvědčit pohledem do testovacích tabulek (benchmarks), ne vždy je uváděná rychlost procesoru rozhodující pro výslednou rychlost práce celého systému. Procesory jsou dnes už tak rychlé, že zahlcují systémovou paměť. Proto se její architektura stává jedním z rozhodujících faktorů efektivity využití potence stále rychlejších procesorů.

Dokud procesory běžely tak asi do 10 MHZ, bylo vše bez problémů; dynamické ramky jim bohatě stačily. To už ale neplatí. Když máte systém s rychlým procesorem a pamětí, která za ním pokulhává, celý systém na to doplácí. Přehnaně řečeno asi tak, jako kdybyste v počítači s 80386 měli jako paměť jen jednotku floppy. Čekací doba pomalejší paměti RAM samozřejmě není tak velká jako u diskety. Jenže procesor je s ní ve styku prakticky neustále. A tak se také dostává do stavů čekání (wait states) na to, až jej paměť dožene, a bude mu schopna poskytnout potřebná data nebo je naopak přijmout.

Rychlost procesoru

Nejdůležitějším aspektem je doba cyklu (cycle time) procesoru. To je doba, za niž 80386 provede svou nejrychleší instrukci. U 16-MHz 80386 je to 62,5 ns. Tato informace pro samotnou rychlost, s jakou procesor provádí operace, ještě není dostačující. Některé instrukce - jako NOP - provede během jednoho cyklu. Ale jiné mu zaberou dva i více cyklů. Jednoduché operáce mezi registry (register-to-register) jako např. CMP (compare, porovnej) a ADD (součet) trvají 2 cykly. Jsou i takové, které spolknou 9 cyklů - třeba MUL (násobení).

Zatím jsme mluvili jen o instrukcích, které si procesor odbyde interně. Jak programátoři vědí, takových instrukcí je v programu menšina. Větší část z nich čte z paměti nebo do ní zapisuje (není bez zajímavosti, že čtení paměti je v programech podstatně víc než zápisů). Instrukce se protahují o dobu přístupu do paměti. Když ADD vyžaduje příjem dat z paměti, trvá 2-6 cyklů v případě, že je výsledek součtu ukládán do registru. Je-li zapisován zpět do paměti, operace už zabere 7 cyklů.

Z toho vyplývá, že celý systém bude plně efektivní jen tehdy, bude-li doba přístupu do paměti harmonizovat s rychlostí procesoru. Pokud bude paměť pomalejší, bude se procesor dostávat do stavů čekání. Každý cyklus, po který procesor jen čeká na paměť - wait state - práci systému zpomaluje. Když reklama hlásá "0 WAIT STATE!", znamená to, že výrobce vyřešil architekturu paměti nabízeného počítače tak, že mikroprocesor může jet (skoro) pořád naplno.

Rychlost paměti

Protože 80386 potřebuje k zaměření se na paměť typicky 2 cykly, znamená to, že by mu paměť měla

během nich umět odpovědět - tj. za dobu 2 x 62,5 ns = 125 ns. Takhle to vypadá, že 100-ns DRAM (dynamická RAM) i 120-ns DRAM by mohla být vyhovující. Bohužel není. Uvedené hodnoty uvádějí dobu přístupu (access time), tj. čas, za který je paměťový čip připraven k požadované činnosti. Jenže při každém kontaktu procesoru s pamětí se určitá její adresovaná část občerstvuje (recharge) ještě předtím, než dojde k vlastnímu přenosu dat mezi ní a procesorem. Pro výpočet doby trvání tohoto občerstvení neexistuje jednoduchá formulka. Bývá to obvykle jen o něco méně, než kolik je vlastní doba přístupu. U 100-ns DRAM je to asi 90 ns.

To ale pořád ještě není všechno. Cyklicky, nezávisle na práci mikroprocesoru (ale tak, aby se mu nepletly do cesty), speciální občerstvovací obvody udržují vytrácející se náboje v paměťových buňkách na potřebné úrovni. Tyto občerstvovací (refresh) cykly prodlužují celkovou dobu komunikace o dalších 6 až 12 procent.

Doba cyklu 100-ns DRAM je tedy ve skutečnosti 190 ns plus refresh.

Problém

Když porovnáte rychlost 16-MHz 80386 (125 ns) a 100-ns DRAM (190 ns + refresh), vidíte, že rychlost paměti je nevyhovující. Za tuto diferenci pak budete platit čekacími stavy. U procesorů 20-MHZ a 25-MHz 80386 se efektivita využití jejich schopností ještě zhorší. To je podstata problému, se kterou se potýkají současní tvůrci počítačů s vysokou operační rychlostí. Mohou se dát jednou ze šesti cest - smířit se s pomalejší DRAM, užít rychlejší DRAM, statické RAM (SRAM), nebo se přiklonit k některé ze tří speciálních architektur paměti: prokládané (interleaved, čte se interlivd), stránkované (paged, čte se pejdžd) a rychlé vyrovnávací (cache, čte se keš).

Jak na to?

Nejjednodušší je s daným stavem (čekacím) se smířit. Výsledkem je levný produkt, který však nevyhovuje stále náročnějším potřebám současných aplikací.

Použití rychlých pamětí DRAM vede k prudkému vzrůstu ceny systému. Mnohem podstatnější problém je ale v tom, že pro 20-MHz 80386 s cyklem 100 ns a pro 25-MHz 80386 (80 ns) paměti DRAM, které by těmto procesorům stačily, ani neexistují. Když použijeme nejrychlejší dostupné 80-ns DRAM, tak i pro 16-MHz CPU s cyklem 125 ns budou o něco pomalejší - jejich cyklus jde nad 140 ns.

SRAM mají dobu přístupu v podstatě shodnou s dobou cyklu. Srovnáme-li oba typy pamětí se shodnými dobami přístupu, SRAM jsou ve skutečnosti skoro dvakrát rychlejší. Navíc nepotřebují refresh. 100-ns SRAM tedy 16-MHz CPU plně vyhoví (až do rozsahu cca 1 MB). Problém je však v ceně. SRAM jsou mnohem dražší než paměti DRAM. To je důvod, pro který SRAM žádný z předních výrobců počítačů s 80386 nedává do hlavní paměti.

Prokládaná paměť

Podstatou je prokládání (prolínání) doby cyklu v několika mezi sebou přepínaných bankách paměti DRAM. Banky se občerstvují v různých momentech, proto odpadá nutnost občerstvení při každém jednotlivém přístupu procesoru do paměti. Každá banka obsahuje každou N-tou adresu, kde N je počet bank. V nejjednodušších PC s 80386 jsou banky dvě - v jedné jsou liché, ve druhé sudé adresy. Pokud jde procesor adresami pamětí sekvenčně, vše probíhá bez čekacích stavů. Odskoky s sebou přinášejí 50-procentní pravděpodobnost, že skočí do banky, která bude připravena (a naopak). Když bude program potřebovat přístup postupně do dvou adres, z nichž obě budou v jedné bance, nastane stav čekání. Např. Tandy 400 běží rychleji se dvěma bankami 1 MB paměti než s jednou.

Stránkovaná paměť

Paměť DRAM je rozdělena do stránek (typicky po 2K). Když program běží v jedné stránce, nevznikají čekací stavy. Ty se objeví vždy při přechodu do jiné stránky. Tomu je nutno podřídit programovou strukturu - např. smyčky by měly zásadně být jen uvnitř jedné stránky. Počítač Compaq 386/16 dosáhl s tímto typem paměti hodnoty 0,8 čekacího stavu při použití 100-ns DRAM.

Caching

je architektura přinášející kompromis mezi paměťmi SRAM a DRAM. Cache je rozsahem malá paměť s velmi krátkým cyklem. Současné nejrychlejší systémy s 80386 používají cache 32K nebo 64K s paměťmi 25-ns až 35-ns SRAM. Cache hraje roli určitého bufferu, možno říci i stránky, se kterou je procesor v kontaktu. Kdykoli je třeba provést operace, jejichž instrukce či data jsou uloženy v DRAM, jsou převedeny do cache. Tak se samozřejmě cache velmi rychle zaplní. Pak jde o to rozhodnout, co dát odtud pryč. Nejčastěji se používá metoda LRU (Least Recently Used) - z cache jsou odstraněny instrukce/data, která procesor nepoužil nejdelší dobu. Podobně jako u stránkované paměti, je i zde nutno funkci cache podřídit strukturu programu, abychom se vyhnuli zbytečně častým přenosům mezi DRAM a SRAM.

Efektivita cache je dána dobou, po kterou v ní má procesor, co potřebuje, bez nutnosti obracet se na hlavní paměť. Vyjadřuje se v procentech. Obvod Intel 82385, který řídí paměť cache, slibuje 95 procent. Většina předních výrobců nejrychlejších počítačích jej používá. Některé řídící obvody zásadně zapisují do hlavní paměti, i když třeba místo pro zápis je v cache. 82385 je i v tom dokonalejší. Když má objekt zápisu v cache, zapíše do něj bez ztráty času a dokonce mezitím, co se procesor zabývá svými záležitostmi, převede obsah zápisu na určené místo hlavní paměti DRAM za pomoci procesu DMA. Může se ovšem stát, že dojde k DMA zápisu do hlavní paměti, jejíž obsah pak nebude odpovídat obsahu adekvátního místa v cache. Pokud by tuto hodnotu procesor použil, došlo by k nepříjemnostem. 82385 to řeší tak, že kdykoli obvody DMA zapisují do hlavní paměti, zjistí, zda nemá nějaké zrcadlové dvojče v cache. Když ano, patřičné místo v cache označí jako invalidní. Pokud se na ně procesor později zaměří, pozastaví jej, odskočí pro správnou hodnotu do hlavní paměti, svou invalidní opraví a vše může pokračovat.

Jak dál?

Vývoj ukazuje, že paměti SRAM, které by byly největším garantem nulových čekacích stavů rychlých procesorů, nemají pro svou vysokou cenu v počítačích typu PC perspektivu. Uvedené architektury paměti mají své háčky a nezaručují čistý nulový stav čekání. Ze všech uvedených se nejvíc prosazuje paměť cache, pro jejíž řízení jsou vyvíjeny další nové obvody.

Přeložil -elzet-

UNIVERZÁLNÍ INTERFACE PRO ATARI 800 XL

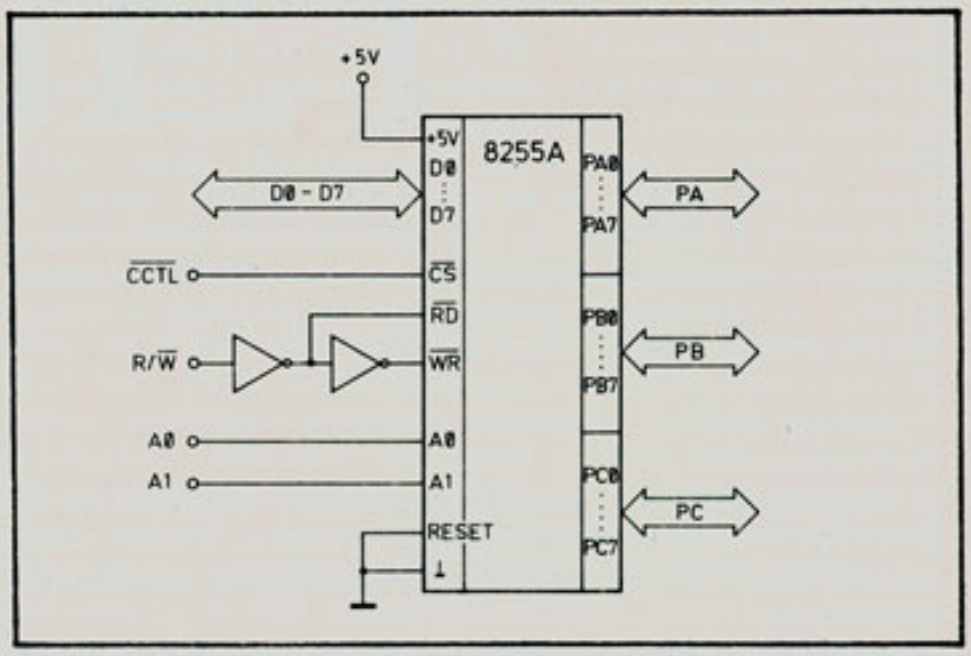
Po čase snad každý vážnější zájemce o výpočetní techniku narazí na problém připojení nějakého zařízení ke svému počítači. Já osobně jsem byl postaven před problém připojení programátoru pamětí EPROM ke svému ATARI 800 XL. Použití joystickových konektorů nepřipadalo pro malý počet signálů v úvahu a žádné jiné zapojení jsem v dostupné literatuře nenašel. Proto jsem vytvořil následující zapojení.

Při návrhu jsem kladl požadavek na jednoduchost, snadnou reprodukovatelnost a využití všech možností, které může ATARI poskytnout. Celkové schéma zapojení je na obr. 1. Obvod je umístěn do adresového prostoru D500H-D5FFH, který je normálně nevuyžit. Jakýkoli pokus o přístup do této oblasti však generuje signál CCTL v konektoru cartridge. Proto je obvod připojen přes tento konektor. Na něm není vyveden signál RESET, takže tento vstup obvodu je stále připojen na log. 0. Obvod je třeba nulovat vysláním stavového slova CW. Je aktivován signálem CCTL, který je připojen na vstup CS. Směr toku dat je ovládán signály RD a WR, získanými ze signálu R/W konektoru.

Protože je tento výstup mikroprocesoru už zatěžován několika vstupy obvodů v počítači, k dekódování signálů jsem raději použil obvodu řady LS. Signál R/W je veden na oddělovací invertor, z jehož výstupu je veden signál RD a opětovným invertováním je získán signál WR. Ostatní signály nebylo třeba dělit, protože jsou zatěžovány jen jedním vstupem. Vstupy AC a A1 jsou připojeny na adresovou sběrnici na stejnojmenné signály. Datová sběrnice je připojena bez jakýchkoli úprav.

Ovládání obvodu je velmi jednoduché. Na adresách D500H (54528) až D502H (54530) jsou umístěny brány A-C a na adrese D503H (54531) je registr stavového slova CWR. Interface je možno ovládat jak z Basicu, tak ve strojovém kódu. Protože se obvod nenuluje signálem RESET, je třeba jej nulovat zapsáním CW a pak teprve přenášet data.

Zapojení na univerzální desce jsem s počítačem propojil vícežilovým kabelem ukončeným přímým konektorem. Ten byl vyleptán na oboustranné desce kuprextitu. Na obr. 2 je umístění signálů v konektoru počítače pro připojení cartridge. Spičky označené černým trojúhelníčkem jsou v zapojení použity.

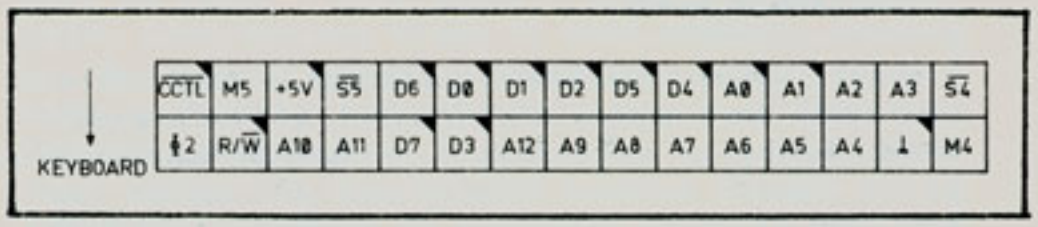


Obr. 1 - Celkové schéma zapojení

Obvod funguje na první zapojení. O jeho bezchybnosti se lze přesvědčit přepnutím obvodu do módu 0 a nastavením bran jako výstupních. Po zapsání jakékoli hodnoty do brány bychom po opětném přečtení (např. v Basicu pomocí PEEK) měli obdržet stejnou hodnotu.

Obvod pracuje již delší dobu naprosto bez závad a je možno jej doporučit každému, kdo má zájem o rozšíření možností svého počítače.

Zdeněk Polách



Obr. 2 - Zapojení konektoru cartrige v počítači

ČÍSELNICOVÝ TERMINÁL PRO ZX SPECTRUM

Popsané zapojení umožňuje jednostrannou komunikaci člověk - počítač formou zadávání dekadických číslic z telefonní číselnice. Je využitelé v případech, kdy se můžeme obejít bez monitoru (např. pro ovládání zvětšovacího přístroje z fotokomory, provádění korekcí řízeného vytápění od kotle ústředního topení apod.).

Základem řešení je číselnice, kterou můžeme získat třeba z vyřazeného telefonního přístroje. Číselnice dává na dvou vývodech (typicky zelený a žlutý) impulsy s klidovým stavem - sepnuto. Počet rozepnutí je určen hodnotou zvolené číslice. Aby bylo připojení k mikropočítači co nejjednodušší, použil jsem interface Kempston pro joystick, který je ve vybavení téměř každého majitele Spectra. Vlastníme-li příslušný devítivývodový konektor, napojíme vývody číselnice na špičky 4 a 8, tedy pro "pohyb doprava". Na polaritě přitom nezáleží. Není-li potřebný konektor k dispozici, můžeme si opatřit vlastní joystick jakýmkoli jiným vhodným konektorem a připojovat pak číselnici k němu. Jednu špičku tohoto konektoru napojíme na společný vodič (signálovou zem) a druhou špičku na plošku určenou pro pohyb doprava. Schéma připojení pro jeho jednoduchost neuvádím.

Obslužný program tohoto "terminálu" pro ZX-Spectrum je řešen procedurou DEJZNAK v Pascalu (kompilátor HP4TM161). Procedura je po aktivaci ve stavu očekávání akce. Jí může být stisknutí tlačítka na klávesnici počítače nebo příchod impulsu od připojené číselnice. V prvním případě procedura vrátí volajícímu programu prostřednictvím parametru "ZNAK" ASCII reprezentaci stisknutého tlačítka a ve druhém parametru znak 'K'. Ve druhém případě jsou spuštěny dekódující procedury počítající počet impulsů číselnice. Je-li přijatá číslice uznána za bezchybnou, obsahuje parametr "ZNAK" ASCII reprezentaci vytočené číslice a parametr "TYP" znak 'T'. Byla-li dekódována chyba, je obsah parametru "ZNAK" nedefinován a parametr"TYP" obsahuje znak 'E'. Za chybový stav je považováno rovněž rozpojení linky (přetržení). V tomto případě je vracen chybový kód pravidelně asi jedenkrát za sekundu. Procedura je ve výpisu 1.

Použití procedury v praxi a reprezentace obdržených znaků je záležitostí programu pro konkrétní aplikaci. Pro ilustraci použití procedury je ve výpisu 2 uveden krátký program, který na obrazovce opisuje znaky přicházející z klávesnice nebo číselnice. Každý znak je doplněn místem původu (K nebo T) a dvěma mezerami. Program se ukončí po vstupu znaku '!' z klávesnice počítače.

Ing.Jan Schlossarek

Výpis 1 - Procedura DEJZNAK

```
procedure DEJZNAK  (var ZNAK,TYP:char);
{=====================================}
{Procedura vraci ZNAK vyslany z klaves-
nice nebo z telefonni ciselnice. TYP
oznacuje: 'K' - vstup z klavesnice,
          'T' - vstup z ciselnice,
          'E' - pripadny chybovy stav.}

const TRM=31; SPI=chr(1); {ZX Spectrum}
      ZPOZDI=599;
var   POCET,DELKA : integer;
      ZNAK1       : char;

function CEKEJ : char;
{-------------------------------------}
{Cekej na startovaci impuls nebo v pri-
pade vstupu z klavesnice vrat znak.   }
const TST   = 59;
var   ZNAK2 : char;
      J     : integer;
begin{CEKEJ}
   repeat
      J:=0; ZNAK2:=inch;
      while (inp(TRM)<>SPI) and (J<TST)
            do J:=succ(J)
   until (ZNAK2<>chr(0)) or (J=TST);
   CEKEJ:=ZNAK2
end{CEKEJ};

function PULS : integer;
{-------------------------------------}
{Zmer delku jednoho pulsu z ciselnice }
const TMX=9999; TCH=999; TMN=99;
var   J : integer;
begin{PULS}
   J:=0;
   while (inp(TRM)<>SPI) and (J<TMX)
         do J:=succ(J);
   if J>TCH then PULS:=9{Prerus. linka}
   else if J<TMN
           then PULS:=0{Rusivy impuls}
           else PULS:=1{Regularni imp}
end{PULS};

function KONEC : boolean;
{-------------------------------------}
{Hledej konec prenosu cislice z TRM   }
const TKO=799;
var   J : integer;
begin{KONEC}
   J:=0;
   while (inp(TRM)=SPI) and (J<TKO)
         do J:=succ(J);
   if J<TKO then KONEC:=false
           {Pokracuje dalsi impuls}
           else KONEC:=true
           {Konec prenosu cislice }
end{KONEC};

procedure BPR (A,B:integer);
{-------------------------------------}
begin{BPR}
   inline(#DD, #6E, #2, #DD, #66, #3,
          #DD, #5E, #4, #DD, #56, #5,
          #CD, #B5, #3, #F3))
end{BPR};

procedure BP (F:integer; D:real);
{-------------------------------------}
{Generuj zvukovy signal               }
begin{BP}
   BPR(entier(F*D),
       entier(487500/F-30.125))
end{BP};
```

```
      {*Telo procedury DEJZNAK*}
begin{DEJZNAK}
   POCET:=0; ZNAK1:=CEKEJ;
   if ZNAK1=chr(0) then
   begin      {Vstup z ciselnice}
      TYP:='T';
      repeat
         DELKA:=PULS;
         if DELKA=1 then
                    POCET:=succ(POCET)
                    else if DELKA=9
                         then TYP:='E'
      until KONEC or (TYP='E');
      if POCET=10 then POCET:=0;
      ZNAK:=chr(POCET+48); BP(1046,0.1)
   end else
   begin      {Vstup z klavesnice}
      TYP:='K';ZNAK:=ZNAK1;BP(523,0.1);
      for POCET:=1 to ZPOZDI do
   end
end{DEJZNAK};
      {*Konec procedury DEJZNAK*}
{-------------------------------------}
```

```
program OPISUJ;
{-------------------------------------}
{Over cinnost procedury DEJZNAK.      }
var   VSTUP,ZDROJ : char;
   .
   .
{Zde vloz deklaraci procedury DEJZNAK.}
   .
   .
begin{OPISUJ}
   repeat
      DEJZNAK (VSTUP,ZDROJ);
      write(VSTUP,ZDROJ,'  ')
   until VSTUP='!'
end{OPISUJ}.
```

Výpis 2 - Program OPISUJ

UNIVERZÁLNÍ TISKOVÁ RUTINA PRO ZX SPECTRUM

Spectristé nevlastnicí tiskárnu často stojí před problémem, kde vytisknout programový výpis, obsah databanky apod. Možnosti bývají různé - jednou je k dispozici tiskárna DZM 180 jindy VIDEOTON či SEIKOSHA GP 100A, EPSON atd. apod. Uvedená rutina "T-3.1" dokáže otestovat a obsluhovat libovolnou tiskárnu s paralelním interfacem Centronix.

Připojení tiskárny

Připojení tiskárny je realizováno prostřednictvím programovatelného obvodu MHB 8255A. V podstatě lze použít libovolný univerzální interface s tímto obvodem - viz např. (1) nebo (2). Řídící slovo CW má hodnotu 130 pro nastavení kanálů PA a PC jako výstupních a PB jako vstupního. Kanál PA je použit pro přenos dat a PC pro vysílání signálu STROBE. Kanálem PB se čtou signály BUSY a ACK. Při softwarovém ovládání tiskáren je vhodné pro odpověď její připravenosti používat signál BUSY, neboť v porovnání se signálem ACK má delší dobu trvání. Tak nevznikají problémy s detekcí přítomnosti impulsu s ohledem na rychlost programu ve strojovém kódu. U některých tiskáren však signál BUSY "chybí" - pak je automaticky použit ACKn. Časové průběhy komunikace s tiskárnou jsou k dispozici v příslušné literatuře.

Funkce softwaru

"T-3.1" je relokovatelný blok strojového kódu 230 bajtů dlouhý. Tak se vejde i do tiskového bufferu počítače od adresy 5800H. Volací část INIT provede inicializaci systémové proměnné STRMS, naprogramuje PIO 8255A, otestuje signál ACKn a podle jeho stavu nastaví pomocný bajt DETEKTOR. Jím se rozhoduje, zda se mají data a řídící signály vysílat pozitivně nebo negativně. Proto před inicializací musí být tiskárna zapojena. Ve druhé části "T-3.1" jsou modifikovány adresy volání tiskové rutiny a nastaven registr IX.

"T-3.1" se ovládá basicovými příkazy LLIST a LPRINT s možností použití tabelačních příkazů "TAB" a "," (čárka) a je propustný pro všechny ovládací kódy tiskáren. Grafické kódy (128-164) jsou tisknuty jako "*" - neznámý znak. Počet znaků na řádek lze volit příkazem POKE start.adresa+206,(počet). "T-3.1" je vhodný k použití v programech D-TEXT, GENS, MONS a celé řadě dalších.

Literatura

1. Soldán J.: Interfejs s MHB 8255A pro mikropočítač ZX Spectru AR A6/1985
2. Formánek P.: Tiskárna D 100 a ZX Spectrum, AR A7/1987
3. Firemní literatura TESLA Piešťany k.p.
4. Logan J., O'Hara F.: The Complete Spectrum ROM Disassembly,1983
5. Firemní literatura MERA-BLONIE, PLR
6. Firemní literatura VIDEOTON, MLR 1986

Ing.Pavel Jáneš

```
;----------------------
;Univerzální
;relokovatelná
;tisková rutina
;----------------------
;Délka řádky:
;  POKE START+206,POCET
;----------------------

        ORG  23300

CW      EQU  130
PA      EQU  #1F
PB      EQU  #3F
PC      EQU  #5F
CWR     EQU  #7F
STAT    EQU  #5CC5

INIT    LD   A,CW
        OUT  (CWR),A
        LD   H,B
        LD   L,C
        LD   DE,TOK
        ADD  HL,DE
        LD   (STAT),HL
        DEC  HL
        IN   A,(PB)
        BIT  7,A
        JR   NZ,NAST
        LD   (HL),1
        XOR  A
        JR   NUL
NAST    LD   (HL),0
        LD   A,#FF
NUL     OUT  (PC),A
        RET

SAT     DEFB 0,0,0

;SAT+0...INDIK.TAB
;SAT+1...POCITADLO
;SAT+2...DETEKTOR

ZACAT   PUSH AF
        LD   DE,(STAT)
        LD   HL,ADTISK
        ADD  HL,DE
        PUSH HL
        POP  BC
        LD   HL,N7
        ADD  HL,DE
        LD   (HL),C
        INC  HL
        LD   (HL),B
        LD   HL,N8
        ADD  HL,DE
        LD   (HL),C
        INC  HL
        LD   (HL),B
        DEC  DE
        DEC  DE
        DEC  DE
        PUSH DE
        POP  IX
        LD   A,(IX+0)
        CP   23
        JR   Z,TAB
        POP  AF
        CP   23
        JR   Z,NAVR
        CP   6
        JR   Z,CARKA
        CP   12
        JR   NZ,ENTER
        LD   (IX+1),0
        JR   TISK
ENTER   CP   13
        JR   NZ,DAL
KON     LD   (IX+1),0
        LD   A,13
A7      CALL TISK
        LD   (IX+1),0
        LD   A,10
        JR   TISK
DAL     CP   32
        JR   NC,PISM
        JR   TISK
PISM    CP   127
        JR   NC,GRAF
        JR   TISK
GRAF    CP   165
        JR   NC,TOKEN
        LD   A,42
        JR   TISK
TOKEN   SUB  165
        CALL #0C10
        RET
NAVR    LD   (IX+0),23
        DEC  (IX+1)
        RET
TAB     POP  AF
        DEC  A
        SUB  (IX+1)
        JR   TAM
CARKA   LD   A,(IX+1)
        LD   B,A
        ADD  A,16
        AND  #F0
        SUB  B
TAM     LD   B,A
MEZ     LD   A,32
A8      CALL TISK
        DJNZ MEZ
        LD   (IX+0),0
        RET
TISK    PUSH AF
        LD   A,(IX+2)
        CP   1
        JR   Z,ACK
BUSY    IN   A,(PB)
        BIT  0,A
        JR   NZ,BUSY
        POP  AF
        OUT  (PA),A
        XOR  A
        OUT  (PC),A
        LD   A,#FF
ZPET    NOP
        OUT  (PC),A
        LD   A,(IX+1)
        INC  A
        CP   65
        JR   NC,KON
        LD   (IX+1),A
        RET
ACK     IN   A,(PB)
        BIT  7,A
        JR   NZ,ACK
        POP  AF
        CPL
        OUT  (PA),A
        LD   A,#FF
        OUT  (PC),A
        XOR  A
        JR   ZPET

ADTISK  EQU  TISK-ZACAT
N7      EQU  A7+1-ZACAT
N8      EQU  A8+1-ZACAT
TOK     EQU  ZACAT-INIT
```

ARMSTAD/SCHNEIDER

TABULKA SYSTÉMOVÝCH PROCEDUR

č.	adresa vektoru	CPC 464 - 664 - 6128

------- O b s l u h a k l á v e s n i c e -------

00 BB00 — 19E0 - 1B5C - 1B5C

Incializace obsluhy klávesnice.
Vstup: -
Výstup: mění AF, BC, DE, HL.

01 BB03 — 1A1E - 1B5C - 1B5C

RESET obsluhy klávesnice.
Vstup: -
Výstup: mění AF, BC, DE, HL.

02 BB06 — 1A3C - 1BBF - 1BBF

Obsluha klávesnice čeká na zadání znaku.
Vstup: -
Výstup: při CY=1 akumulátor obsahuje kód stisknuté klávesy.

03 BB09 — 1A42 - 1BC5 - 1BC5

Přečte znak z klávesnice. Procedura testuje, jestli se na klávesnici neobjevil nějaký znak hned po jejím vyvolání.
Vstup: -
Výstup: jestliže se znak objevil, je CY=1 a akumulátor obsahuje kód tohoto znaku: v opačném případě mění akumulátor.

04 BB0C — 1A77 - 1BFA - 1BFA

Uchovává znak do příštího vyvolání předcházející procedury.
Vstup: akumulátor obsahuje znak na uschování.
Výstup: -

05 BB0F — 1ABD - 1C46 - 1C46

Ukládá znakový řetězec s odpovídajícím kódem.
Vstup: reg.B - kód, který se má přiřadit znakovému řetězci; reg.C - délka řetězce a HL - jeho adresa.
Výstup: jestliže přiřazení proběhlo, CY=1; je-li řetězec dlouhý nebo špatný kód, CY=0; mění A, BC, DE, HL.

06 BB12 — 1B2E - 1CB3 - 1CB3

Čte znak ze znakového řetězce. Znaky v řetězci se číslují od 0.
Vstup: v reg.A kód řetězce, v L číslo znaku.
Výstup: když byl znak načten, reg.A a CY = 1; když byl špatný kód nebo znakový řetězec není dost dlouhý, CY=0 a reg.A je změněn; mění DE.

07 BB15 — 1A7B - 1C04 - 1C04

Přidělení bufferu pro řetězce funkčních kláves.
Vstup: DE - adresa bufferu, HL - jeho délka.
Výstup: je-li vše v pořádku, CY=1, když ne, CY=0; mění A, BC, DE, HL.

08 BB18 — 1B56 - 1CDB - 1CDB

Čeká na znak zadaný z klávesnice.
Vstup: -
Výstup: CY=1, reg.A obsahuje znak stisknuté klávesy.

09 BB18 — 1B5C - 1CE1 - 1CE1

Testuje, jestli vystupuje znak z klávesnice.
Vstup: -
Výstup: když byla stisknuta klávesa, CY=1 a reg.A obsahuje znak této klávesy; jinak CY=0.

10 BB1E — 1CBD - 1E45 - 1E45

Zjišťuje, jestli je stisknuta určitá klávesa. Rovněž umožňuje rovněž číst joystick.
Vstup: v reg.A - číslo testované klávesy.
Výstup: Když klávesa není stisknuta, Z=1, když je stisknuta, Z=0; mění se A, HL; C obsahuje stav kláves SHIFT a CTRL.

11 BB21 — 1BB3 - 1D38 - 1D38

Zjišťuje, zda je zapojen CAPS LOCK.
Vstup: -
Výstup: v reg.L - stav SHIFT, v reg.A - stav CAPS LOCK (0 pro vypnuto a FFH pro zapnuto).

12 BB24 — 1C5C - 1DE5 - 1DE5

Zjišťuje stav joysticků.
Vstup: -
Výstup: v reg.A a H - stav joysticku 1, v reg. L stav joysticku 0; značení bitů jako u basicové funkce JOY.

13 BB27 1D52 - 1ED8 - 1ED8

Nastavuje kód, který má být vygenerován při stisku určité klávesy (bez užití SHIFT nebo CTRL)
Vstup: reg.A - číslo klávesy, reg.B - ASCII kód pro tuto klávesu.
Výstup: mění AF, HL.

14 BB2A 1D3E - 1EC4 - 1EC4

Poskytuje kód odpovídající určité klávese.
Vstup: reg.A - číslo klávesy.
Výstup: reg.A - ASCII kód této klávesy; mění F, HL

15 BB2D 1D57 - 1EDD - 1EDD

Nastavuje kód, který má být vygenerován při stisku určité klávesv současně se SHIFT.
Vstup: akumulátor obsahuje číslo klávesy, B obsahuje ASCII kód pro tuto klávesu.
Výstup: mění AF, HL.

16 BB30 1D43 - 1EC9 - 1EC9

Poskytuje kód odpovídající určité klávese stisknuté současně se SHIFT.
Vstup: reg.A - číslo klávesy.
Výstup: reg.A - ASCII kód této klávesy; mění HL.

17 BB33 1D5C - 1EE2 - 1EE2

Nastavuje kód, který má být vygenerován při stisku určité klávesy současně s CTRL.
Vstup: reg.A - číslo klávesy, reg.B - ASCII kód této klávesy.
Výstup: mění AF, HL.

18 BB36 1D48 - 1ECE - 1ECE

Poskytuje kód odpovídající určité klávese stisknuté současně s CTRL.
Vstup: reg.A - číslo klávesy.
Výstup: reg.A - ASCII kód této klávesy; mění HL

19 BB39 1CAB - 1E34 - 1E34

Nastavuje/ruší autorepeat klávesy.
Vstup: reg.A - číslo klávesy; má-li být autorepeat povolen, reg.B=FFH, má-li být zakázán, reg.B=0.
Výstup: mění AF, BC. HL.

20 BB3C 1CA6 - 1E2F - 1E2F

Zjišťuje, zda zadaná klávesa má povolen autorepeat.
Vstup: reg.A - číslo klávesy.
Výstup: je-li autorepeat povolen, Z=1, není-li povolen, Z=1; nastavuje CY na "0" a mění AF, HL.

21 BB3F 16C9 - 1DF6 - 1DF6

Nastavuje čas zpoždění před prvním opakováním a čas mezi dvěma opakováními.
Vstup: reg.H - zpoždění před prvním opakováním, reg.L - rychlost opakování. Časy se udávají v padesátinách sekundy.
Výstup: -

22 BB42 1C69 - 1DF2 - 1DF2

Přečte rychlost opakování a čas čekání před prvním opakováním.
Vstup: -
Výstup: reg.H - zpoždění před prvním opakováním (1/50 sec), reg.L - rychlost opakování.

23 BB45 1C71 - 1DFA - 1DFA

Nastaví mechanismus procedury BREAK.
Vstup: reg.DE - adresa procedury obsluhy BREAK, reg.C - adresa paměti ROM zvolené pro tuto proceduru.
Výstup: mění AF, BC, DE, HL.

Pozn.: Tento mechanismus může být neutralizován vyvoláním následující procedury.

24 BB48 1C82 - 1E0B - 1E0B

Neutralizuje mechanismus procedury BREAK.
Vstup: -
Výstup: mění AF, Hl.

25 BB4B 1C90 - 1E19 - 1E19

Obnovuje provádění procedury BREAK, když byla nastavena procedura č. 23 (BB45).
Vstup: -
Výstup: mění AF, HL.

Obsluha textu

26 BB4E 1078 - 1070 - 1070

Inicializuje textový režim.
Vstup: -
Výstup: mění AF, BC, DE, HL.

27 BB51 1088 - 1080 - 1084

RESET textového režimu.
Vstup: -
Výstup: mění AF, BC, DE, HL.

28 BB54 1451 - 1455 - 1459

Umožňuje umístění znaku/písmena na obrazovce v textovém řežimu.
Vstup: -
Výstup: mění AF.

29 BB57 144B - 144E - 1452

Zakazuje umístění znaku/písmena na obrazovce.
Vstup: -
Výstup: mění AF.

30 BB5A 1400 - 13FA - 13FE

Posílá na obrazovku v textovém režimu znak nebo provede řídící kód (tj. znaky 0-1FH).
Vstup: reg.A - znak posílaný na obrazovku.
Výstup: -

31 BB5D 1344 - 1331 - 1335

Posílá na obrazovku v textovém režimu znak nebo grafický symbol odpovídající řídícímu kódu.
Vstup: reg.A - znak posílaný na obrazovku.
Výstup: mění AF, BC, DE, HL.

32 BB60 13AB - 13AB - 13AC

Čte z obrazovky znak na pozici kurzoru.
Vstup: -
Výstup: když je znak rozpoznán, CY=1 a reg.A obsahuje tento znak; v opačném případě CY=0 i reg.A=0.

33 BB63 13A7 - 13A4 - 13A8

Nastavuje ON/OFF mechanismu obsluhy grafických znaků.
Vstup: reg.A=0 pro OFF, jinak ON.
Výstup: mění AF.

34 BB66 120C - 1204 - 1208

Nastavuje rozměr aktuálního textového okna.
Vstup: reg.H - sloupec levých rohů,
reg.D - sloupec pravých rohů,
reg.L - horní řádek
reg.E - dolní řádek
Výstup: mění AF, BC, DE, HL.

35 BB69 1256 - 124E - 1252

Čte rozměr aktuálního okna.
Vstup: -
Výstup: když okno obsahuje celou obrazovku, CY=0, jinak CY=1; v obou případech je:
reg.H - číslo levého sloupce
reg.D - číslo pravého sloupce
reg.L - číslo horního řádku
reg.E - číslo dolního řádku mění se A.

36 BB6C 1540 - 154B - 154F

Maže aktuální okno (CLS).
Vstup: -
Výstup: mění AF, BC, DE, HL.

37 BB6F 115E - 1156 - 115A

Určuje sloupec pozice kurzoru.
Vstup: reg.A - číslo sloupce kurzoru.
Výstup: mění AF, HL.

38 BB72 1169 - 1161 - 1165

Určuje svislou pozici kurzoru.
Vstup: reg.A - číslo řádky kurzoru.
Výstup: mění AF, HL.

39 BB75 1174 - 116C - 1170

Určuje pozici kurzoru.
Vstup: reg.H - číslo sloupce
reg.L - číslo řádky.
Výstup: mění AF, HL.

40 BB78 1180 - 1178 - 117C

Čte pozici kurzoru.
Vstup: -
Výstup: reg.H - číslo sloupce
reg.L - číslo řádky polohy kurzoru
reg.A - počítač posuvu řádek (scroll).

41 BB7B 1289 - 1282 - 1286

Povoluje promítnutí kurzoru.
Vstup: -
Výstup: mění AF.

42 BB7E 129A - 1293 - 1297

Zakazuje promítnutí kurzoru.
Vstup: -
Výstup: mění AF.

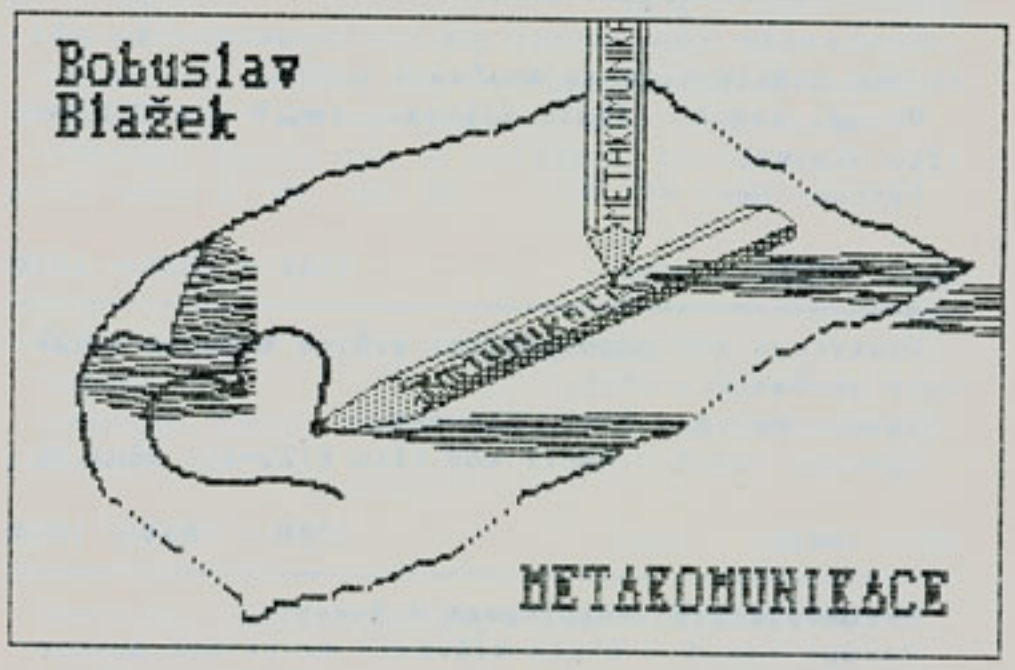
43 BB81 1279 - 1272 - 1276

Povoluje systému promítnout kurzor.
Vstup: -
Výstup: -

Bajtek 1987
Překlad Martin Novotný

(pokračování)

META KOMUNI KACE (1)



(pokračování)

Když někdo něco říká, ukazuje, píše, ale i poslouchá nebo čte, komunikuje. Některé z komunikátů se přitom mohou týkat vlastního komunikování, například "Nic jsem neslyšel". A to je pak **metakomunikace**.

Scéna malých počítačů u nás je zvláštní. Moc se na ní nekomunikuje: nemá své obecně známé a uznávané mluvčí, nemá své veřejné forum, nemá svůj časopis. Je skryta v několika klubech a ještě četněji rozptýlena po domácnostech. Zato má svou poměrně rozbujelou metakomunikaci: o světě malých počítačů hovoří a svá stanoviska publikují lidé stojící mimo něj: politici, celníci, novináři, pracovníci jednotných zemědělských družstev nebo ředitelé škol. Modalita, ve které se tato prazvláštní metakomunikace vede, je převážně normativní: říká se, co se nesmí, co se musí a co by se mělo. Uživatel zůstává v pozadí jak ta němá tvář.

A přitom právě malé počítače jsou komunikačním nástrojem par excellence: můžete jimi ztvárňovat a sdělovat své myšlenky, vytvářet a rozšiřovat své grafické projevy, komponovat a posílat dál svá hudební díla, můžete se díky nim stát uživateli

mezinárodních databank a prostřednictvím počítačových sítí nabízet světu nejrůznější služby. Můžete - ale ne u nás.

Tato rubrika svým způsobem bude rozmnožovat onu již tak rozrostlou vrstvu metakomunikace bez komunikace. Bude se v ní metakomunikovat o tom, jak se jinde komunikuje o malých počítačích. Když ale metakomunikace, tak důkladná. Nebudeme tu tedy jenom přebírat zprávy odjinud, ale skutečnosti, o kterých se dozvíme, budeme srovnávat s naší domácí realitou. A budeme se zamýšlet nad tím, jak se jinde o světě malých počítačů myslí a píše.

Nemá ovšem smyslu si hrát sebeklamnou hru. Tato rubrika nebude žádný "přehled zahraničního tisku" - to by musel být takový tisk u nás běžně k dostání. Výběr nejenom že nebude reprezentativní, ale ani záměrný - dal by se charakterizovat jako "co život dal".

Tento nedostatek se ale může svým způsobem změnit ve výhodu. Místo abychom měli problém volby způsobený přemírou informace, naším úkolem naopak bude využít každého fragmentu asi tak, jako to činí detektivové nebo archeologové.

Podobenka časopisu "64'er"

Tento měsíčník je uživatelský časopis, který se omezuje na jednu značku a z ní na čtyři typy: Commodore 64, 128, 16 a Plus/4. Vychází v nakladatelství Markt und Technik v Haaru u Mnichova a má 176 stran. Toto nakladatelství s více než 500 zaměstnanci a ročním obratem přes 100 miliónů marek vydalo od roku 1976, kdy vzniklo, přes 400 knih (ročně jich přibude kolem 80) a je vydavatelem 14 dalších časopisů: v USA Business Software, Micro/Systems Journal a Software Tools, v Evropě (NSR a Švýcarsko) Amiga, Computer Personlich, Deckblatt, Design and Elektronik, Happy-Computer, Markt und Technik, PC Magazin, PC Magazin PLUS, Power Play, ST Magazin a 68000er. Z názvů časopisů vyčteme část orientace (68000 je čip). Další značky, které jsou předmětem zájmu firmy, najdeme jako záhlaví kapitol katalogu: Apple, Atari a Schneider.

"Čtyřiašedesátka" je z těchto časopisů nepochybně nejpopulárnější, a to proto, že C 64 udělal v NSR kariéru. Prodalo se jich tam zatím 1,5 miliónu (přičemž v celém světě jich má být 10 miliónů). Cena vlastního počítače včetně diskety s grafickým programem GEOS už dlouho činí 299,- marek. Šéfredaktor "64'er" ho nazývá nejúspěšnějším počítačem všech dob a tiskový mluvčí německé pobočky firmy Commodore tvrdí, že je to počítač, pro který existuje snad nejvíc softwaru.

Kdybychom tedy začali se srovnáváním, dalo by se říci, že oblibě tohoto počítače odpovídá u nás popularita Sinclairu ZX Spectrum.

C 64 je osmibitový počítač o kapacitě RAM 64 kB. Přes dětinskost této číslice dokáží do této světničky programátoři vtěsnat celé zástupy. Typické písíčkové hry s grafikou při svém startu tak oslňující jako třeba "Defender of the Crown" se zanedlouho objevují ve verzi pro C 64 a přitom s ochuzeními překvapivě malými. V grafických programech si vybíráte z vytahovacích menu, programy pro editování textu nabízejí uživatelský přepych včetně 80 znaků na řádku a hudba, jakou lze ze tří hlasů tohoto malého syntezátoru vyloudit, dosud nutí fandy zařazovat ji do svých sbírek počítačových hudebních zázraků.

Když si tak po sobě na obrazovce svého C 64 čtu předchozí řádky, nabývám dojmu, že do mě vstoupil duch - povětšinou mladičkých - redaktorů tohoto časopisu. Jejich postoj k rodnému počítači lze totiž charakterizovat jako nezlomně nadšený. Působí na vás jako gumoví panáčci neustále pružně nadskakující nezdrženlivou touhou si už zase pohrávat se svým milovaným přístrojkem. Můžeme tomu říkat věrnost firmě nebo prostě reklamní unisono, nicméně i ten konzumní Západ rozlišuje reklamu chytrou od hloupé stejně jako vkusnou od nevkusné. A nutno říci, že "čtyřiašedesátka" se čte s gustem. Všechna čísla časopisu, která se mi podařilo získat, byla podobně našlapaná, všechna jakoby exaltovaná samou radostí z toho, co se zase od minula podařilo vymyslet. Je to jakási nakažlivá vibrace, ze které se neprobouzíte s nepříjemnou kocovinou, ale jež se stává dlouhodobým zdrojem vašeho stále se prohlubujícího a zjemňujícího zájmu. Nedělají z vás sektáře, dělají z vás amatéra v někdejším smyslu toho slova, tedy milovníka.

Jako většina časopisů tohoto nakladatelství má i tenhle na obálce rámeček o konstantní barevnosti - zde je to azurová - a uvnitř pestrý obrázek. Bývá to kombinace barevné fotografie s iluzivním kresleným obrázkem, často jde o vír objektů zanořený do imaginárního prostoru. Motivem je hlavní článek čísla, ostatní důležité články jsou oznámeny v titulní stránce nápisy.

Jedno z největších lákadel, ale zároveň i největší bluff, to jsou výpisy z programů. I když vám s nimi pomáhá při přepisování pomocný program (odlišné kontrolní součty v řádcích signalizují překlep), rozsah programů psaných většinou ve strojovém kodu činí tento dar danajským. Nemáš trpělivost, zato ale dost peněz? Kup si disketu se všemi programy z čísla - číslo stojí 6,50, disketa k němu 29,90 marek.

Bluff číslo 2 je kvalita těchto programů. Pravda, za 30 marek se žádný slušnější software nedostane. A tady je to dokonce celá kupa programů. Ty malé, před těmi opravdu klobouk dolů. Jsou to "tipy a triky" většinou od čtenářů, pozoruhodné objevy v nevyčerpatelném bludišti čtyřiašedesátky. Ty velké sice také umějí leccos, ale bez oku lahodící grafiky a hlavně bez uživatelského komfortu, který práci s programem mění v slastnou zábavu.

Redakce získává většinu programů od čtenářů. Oběti do svého doupěte láká za pomoci soutěží o ceny. Tisíc nebo dva tisíce marek, to nejsou ani v NSR malé peníze, ale jako honorář například za zbrusu nový grafický program dostat týdenní mzdu, to je opravdu skromné - tím spíš když k tomu redakce nádavkem získává zásobník nápadů a přehled o rozvíjejících se mladých talentech mezi čtenáři.

Čísla mají některé stálé tématické okruhy jako Aktuality, Výpisy programů, vícedílné Kurzy, Test softwaru nebo Test hardwaru. Do každého z nich se vejde i více článků. Jako rubriky jsou označeny například Dopis vydavatele nebo Forum čtenářů.

Nemalou část čísla zabírá inzerce, firemní i individuální. Cenové rozpětí za jeden a tentýž výrobek od různých dodavatelů vede českého člověka k prostoduché dedukci, že alespoň jeden z těch inzerentů s krajními cenami klame: buď je podvodník ten s cenou dumpingovou, nebo ten s cenou nestoudně vysokou. Pohyb cen zatím pořád dolů vyvolává zase zvláštní nervozitu v podobě dedukce: když počkám ještě chvíli, budu si za stejné peníze moci koupit ještě víc. Problém je, že tyto naše pseudoracionální úvahy nás odvádějí stále dál od reality. Malé peníze se platí většinou firmičkám, které nabízejí vysoké procento zmetků. A investovat do dobře vybraného zařízení zase může znamenat začít s ním včas vydělávat.

V čem je časopis neúnavný a nevídaně kritický, to jsou testy hardwaru a jakési dotazníky, jimiž jsou porovnávány programy určitého zaměření. Tady se zhodnocuje obrovská zkušenost formou jak obecně přístupnou, tak hlavně obecně prospěšnou. Je to vlastně kultivace čtenářů, do značné míry vzájemná, nikoli jen padající shůry.

PHDr.Bohuslav Blažek

PROGRAMOVÁ NABÍDKA

KAZETOVÉ PAMĚTI ROM PRO POČÍTAČE ATARI

Ve spolupráci s JZD Český ráj Podůlší, které je výrobcem těchto zásuvných modulů (cartridge), pod softwarovým zajištěním 602. ZO Svazarmu se postupně dostává do prodeje první kolekce programů pro počítače ATARI 800 XL, 800 XE a 130 XE.

Kazetové paměti ROM představují rychlou vnější přídavnou, pevně naprogramovanou paměť na bázi křemíkového mikročipu. Umožňují zavést velmi rychle do počítače program bez použití magnetofonových kazet či disketové jednotky. Používání programových kazet spoří čas a zvyšuje spolehlivost zavedení programu. V případě aplikace turbo zavaděčů lze k časovým úsporám připočíst navíc i úsporu paměťovým médií.

V době uzávěrky tohoto rukopisu nebyly ještě známy přesné ceny, orientačně lze počítat s cenovou úrovní asi 300 Kčs za jeden modul.

ZM01
Slouží ke čtení a spuštění programů v kódu ZX (Turbo 2000) pomocí upraveného datarekordéru Atari 1010, XC 12, Phonemark Atari apod. Implantovaný program ZXL-M umožňuje zavádění souborů nejen v absolutním, ale i v relativním formátu, které původní zavaděč Turbo 2000 neumožňoval nahrávat. Zaváděcí rychlosti jsou 1 a 3 (dle monitoru SUPERMON to odpovídá přibližně 2000 a 4000 Bd). Přechodem na práci v turbo systémech ušetří uživatel ve srovnání se standardním postupem asi 90 % kapacity záznamového média.

ZM02
Čte a spouští programy v kódu ZX (Turbo 2000) bez speciálního datarekordéru Atari. Prostřednictvím nahrávací šňůry, dodávané se zásuvným modulem, lze počítač připojit ke sluchátkovému výstupu běžného magnetofonu. Zaváděcí rychlosti jsou opět 1 a 3, také další možnosti jsou shodné s modulem ZM01.

ZM03
Program umožňuje komunikaci počítačů Atari přes normalizované sériové rozhraní RS 232 s širokou paletou periferních přístrojů i s většinou počítačů standardu IBM PC. Rychlost komunikace je volitelná v rozsahu 50 až 950 Bd.

ZM04
Modul s implantovaným programem Centronics pro komunikaci počítačů Atari přes toto normalizované paralelní rozhraní. Ideální pro připojení většiny standardních mozaikových tiskáren.

ZM05
Modul k rychlému a přehlednému měření délky telefonních rozhovorů a k okamžitému výpočtu ceny hovorného.

ZM101
Čte a spouští programy v kódu ZX (Turbo 2000) pomocí upraveného datarekordéru Atari 1010, XC 12, Phonemark Atari apod. Umožňuje zavádění programů v absolutním i relativním formátu rychlostí 0 až 5 (podle monitoru SUPERMON 1500 až 6000 Bd). Proti ZM01 je modul vybaven kopírovacím programem.

ZM102
Modul slouží ke čtení a spouštění programů v kódu ZX (Turbo 2000) bez speciálního datarekordéru Atari. Prostřednictvím šňůry dodávané s modulem lze počítač připojit ke sluchátkovému výstupu běžného magnetofonu. Ostatní vlastnosti jsou obdobné modulu ZM 101; proti funkčně obdobnému typu ZM02 je vybaven kopírovacím programem.

ZM103
V modulu je implementován monitor pro pořizování nahrávek turbo souborů pomocí kazetopáskového záznamníku. Upravená verze monitoru SUPERMON 2.2 se vyznačuje vysokým uživatelským komfortem. Lze přenést nahrávky programů pořízené na počítači ZX Spectrum na Atari (nefunkčně) a naopak.

ZM104
Program SCREEN COPY znásobuje možnosti počítače pro ty, kteří nejsou zručnými programátory a neumějí si sami upravit existující programy pro výstup na tiskárnu. Program je určen pro tiskárny Atari 1029 nebo Seikosha GP 500 AT.

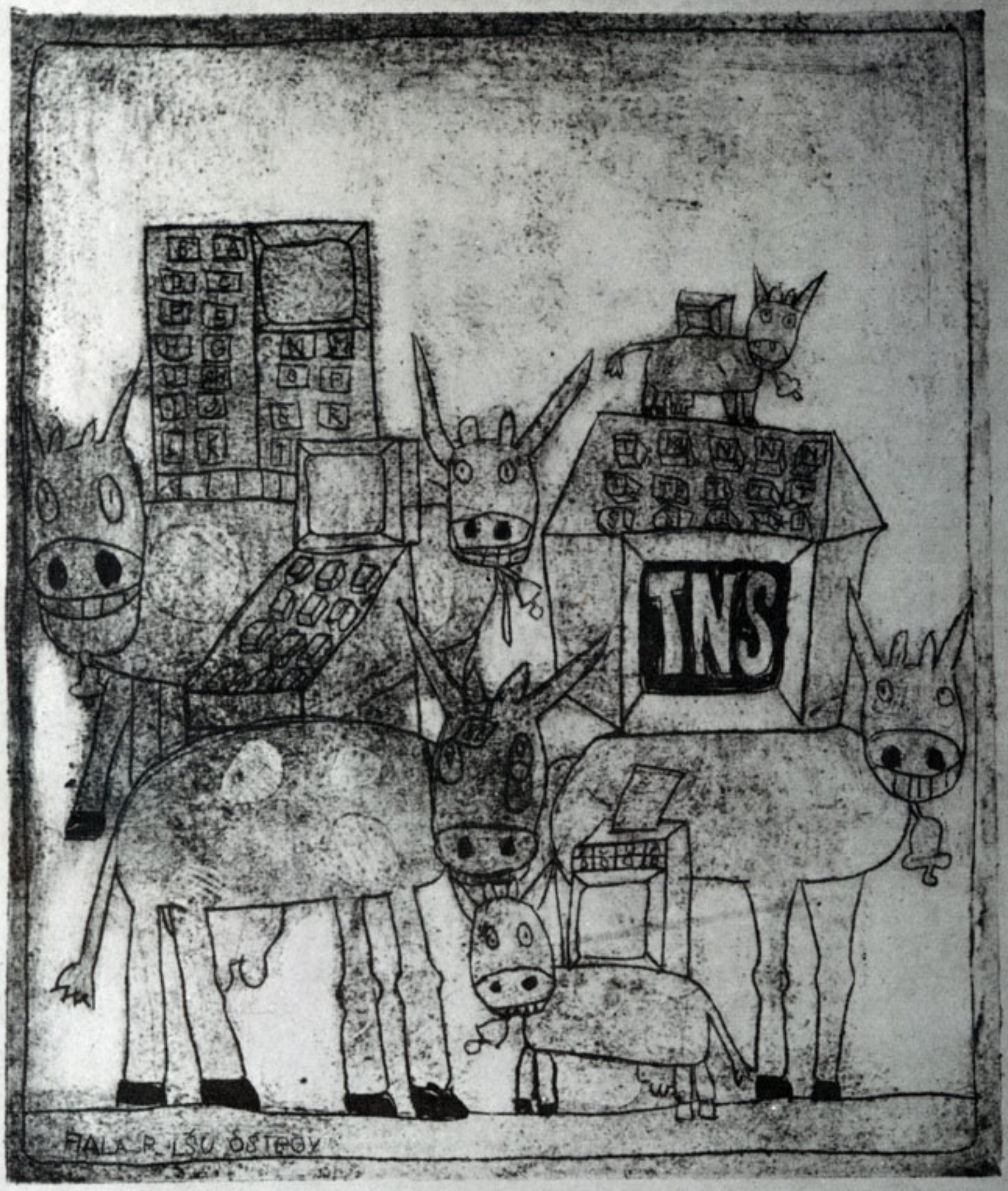
ZM105
Turbo D je zatím nejdokonalejší systém rychlého nahrávání používající vlastní kazetopáskový operační systém (slovenská verze). Systém nevyžaduje úpravu standardních záznamů, používá 1024bajtové bloky, umožňuje pořizovat jištěné nahrávky s dvojitým záznamem. V případě výstytu chyby stačí magnetofon vrátit jen o kousek a opakovat čtení bloku, nikoli celého souboru.

ZM201
TOLSTOJ - textový editor se znakovou sadou azbuky. Obdoba textového editoru Speed Script a Čapek. Umožňuje výpis na obrazovku v režimu osmdesáti znaků na řádek, tiskne azbuku na mozaikových tiskárnách Atari 1029 a Seikosha GP 500 AT.

ZMX01
Vstupně výstupní port s obvodem MHB8255. Modul umožňuje realizovat velké množství dříve vyřešených aplikací propojení mikropočítačů s periferními zařízeními. Otázka měření, regulace a řízení procesů už nemusí být doménou počítačů na bázi mikroprocesoru 8080 a Z80. S modulem ZMX01 to jde i pomocí počítačů Atari.

■ ■ ■

Nabídky z minulých čísel zůstávají v platnosti



DĚTI A POČÍTAČE

Obrázek Petra Fialy z výtvarné soutěže JZD AK Slušovice

ENTEC 160

ENTEC 320

ENTEC je jedna z firem, jejíž zboží – personální počítače a jejich příslušenství – dováží do Evropy firma Ensch z Lucembourgu a prodává je i v Československu (zastupovaná podnikem Media).

ENTEC 386