MIKRO BÁZE

1989 **2**

technický zpravodaj svazarmu pro zájemce o mikropočítače

Cena 12 Kčs

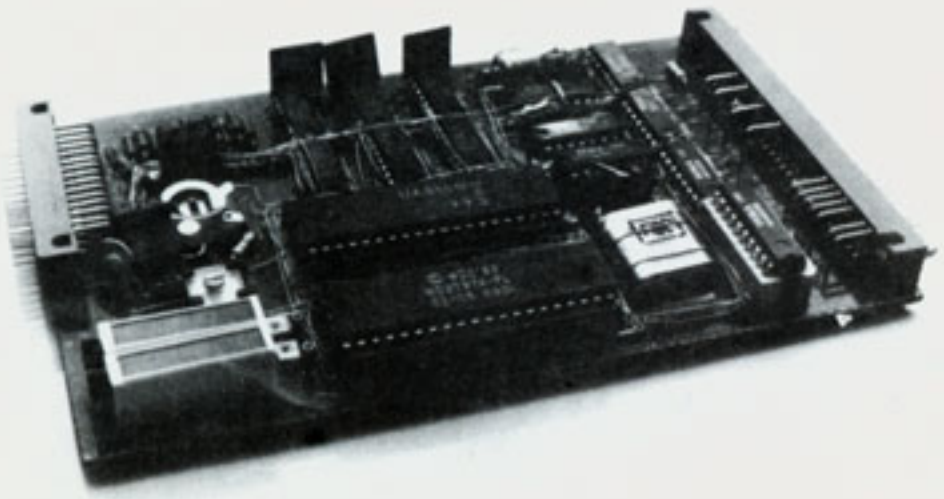
ZX SPECTRUM

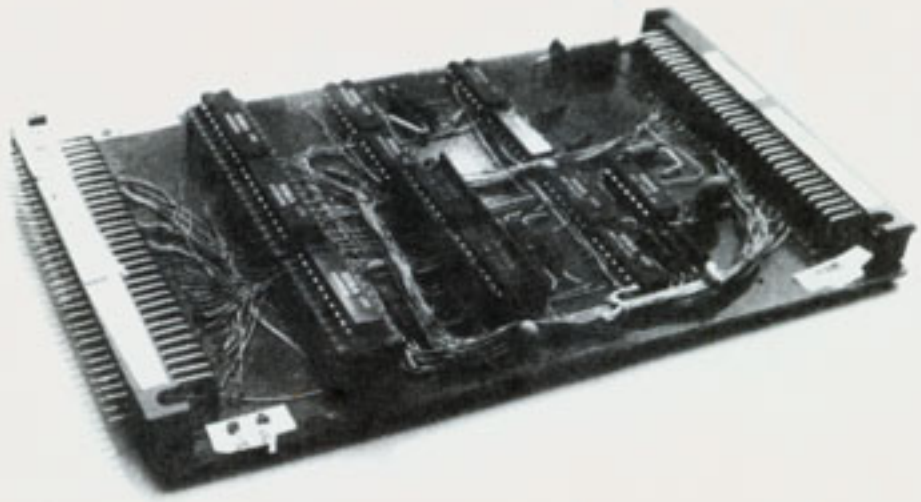
S ROZŠÍŘENOU PAMĚTÍ

Obrazová příloha ke stejnojmennému článku uvnitř tohoto čísla. Uprostřed horního obrázku je miniaturní verze (e) STD BUSu, osazená třemi deskami. Spodní deska (plochým vodičem propojená do ZX Spectra) je posilovač sběrnice, prostřední deska je řadič floppydisku (mechanika 5,25 je částečně vidět v pravém dolním rohu snímku) a na horní desce je právě ověřovaný RAMdisk (část zapojení je zatím na nepájivém kontaktním poli). RAMdisk je možno osadit až osmi pouzdry statických RAM 32×8 bitů. Odběr při zálohování je 8 μA na pouzdro.

Na levém dolním obrázku je funkční vzorek posilovače sběrnice. Je postaven na univerzální desce; propojení je realizováno tenkým smaltovaným drátem.

Na pravém dolním snímku je řadič floppydisku. Vlevo je krátký konektor pro připojení mechaniky. Uprostřed je Z80A–PIO (UA855D) a WD2797. Vpravo je konektor FRB pro připojení k STD BUSu.





1989/2

Mikro

báze

technický zpravodaj svazarmu pro zájemce o mikropočítače

OBSAH

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. Jan Klabal. Sestavil Ladislav Zajíček. Grafická úprava Yvetta Dunděrová. Sekretářka redakce Božena Divišová. Redakční rada: ing. Petr Horský, ing. Jan Klabal, ing. Petr Kratochvíl, Josef Kroupa, Daniel Meca, ing. Alois Myslík, ing. Josef Truxa, Ladislav Zajíček. Za původnost a správnost příspěvků ručí autoři. 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

&

RADIO

SCIFI V BARVĚ ANTRACITU IO

V jednom kouzelném království žili byli jednou jedni, co se měli rádi. Narodila se jim spanilá dceruška Softie. Protože byla dítětem kouzelné říše, jako neposedný rarášek pronikala do světa lidí prostřednictvím toho, čemu v tom světě začali říkat informatika. Když Softie vyrostla z dětských her (Manic Miner apod.), pustila se do vymýšlení kousků, které lidi začaly bavit ještě víc - Pascal, Prolog, C, Unix, Xenix, Word Perfect, dBase... To už si Softie hrála spolu s lidmi celého světa. Tedy... Skoro celého.

Od těch časů pro ni lidé stále zdokonalují životní prostředí, aby se v něm cítila co nejlépe - rychlé paměti, disky, modemy, sítě - dokonce světlovodné! Jako každá mladice, je i Softie parádnice. Hrozně ráda si obléká barevné šatičky EGA, VGA a na stínítcích monitorů se v nich před lidmi všelijak nakrucuje. A oni na ní mohou oči nechat. Jak by se jí to nemělo líbit!

Přesto v duši nosí stín. Někde totiž jako by ji vůbec nechtěli. Doma se o tom radila i s úchvatnou Fantazií, ale ani ta jí nedokázala poradit. Zalistovala v elektronickém adresáři kongresové knihovny a rozhodla se vydat na Olymp za Venuší, která zná všechny finesy jejího rodu. Uklonila se a přednesla svůj bol. Venuše posmutněla, řka: "Milá Softie, stojíš před největší překážkou, jakou si lidé od prvopočátků své historie sami sobě neúnavně strojí. Mám obavu, že ani ohňostroj tvých nejelegantnějších algoritmů ti nepomůže. S Byrokracií si neporadí ani moje krása, ani tvá elegance, ani udatnost reků. Říše Byrokracie je beztvará, nemá hranic, ba nemá ani svého boha, kterého bys mohla okouzlit nebo zmást. Každý příslušník Byrokracie je sám sobě bohem a jedná podle toho. Vše, co voní životem, bytostně ničí ve jménu přetrvání svého pseudobožství. Však víš, jak někteří olympští bozi dokáží být ješitní! Ale i to bledne před sebeláskou Byrokratů. Na Olympu sídlí zástupci všech dobrých lidských stránek i neřestí. Není tu však místa pro boha lidí, kteří se za bohy sami považují. Takový je zákon Olympu. Provázej tě Perseus."

Bylo to právě tou dobou, jak po světě kolabovaly computerové sítě a objevil se softwarový virus. Marná sláva - Softie je přece jenom ženská a v některých ohledech má její trpělivost své meze. Nevím už, co mě k tomu vedlo, ale rozhodl jsem se jí napsat. Jen tak jsem vytočil nějaké směrové číslo, myslím že Rio de Janeira, sluchátko jsem položil na reproduktor a spustil SAVE "Softie-Urgent": Nejdražší Softie, probůh nemysli si, že tě tu nechceme. Jsou nás spousty, co si svůj život bez tebe už vůbec nedovedou představit. Snad právě proto, že jsi nám tu tak vzácná, zanícenější milovníky bys těžko kde po světě hledala. Vím, pro tvé oslnivé kreace tu není zrovna nejkvalitnější ekohardsystém, ale děláme, co můžem. A věř, že kdyby to opravdu šlo, dělali bychom o moc víc. Ostatně to ti už Venuše vysvětlila. Věz, že v Čechách se právě rodí jedno softwarové družstvo, na Moravě prý taky... Jistě, pořád je to strašně málo, skoro nic. Ale jednou se to rozjet musí, jinak se opravdu nikam nedostaneme. Tak už se nekaboň, vždyť se každý den tolik těšíme, až na nás z monitorů zase mrkneš...

Brzy nato se objevily diagnostické programy na odhalení virů. Jen tak v koutku duše jsem si troufl zahlédnout souvislosti. Když se však podařilo vytvořit novou redakci Mikrobáze (a já se do ní vrátil), přestal jsem pochybovat. Až na jedno - napsal jsem tohle všechno opravdu já, nebo že by...sama Softie?

-elzet-

ZX SPECTRUM V OBJETÍ PÉCÉČKA

Když se mi v listopadu 88 dostalo do rukou čerstvě vydané dubnové číslo Mikrobáze, zaujala mě informace o nově nabízeném programu Assembler 80 pro ZX Spectrum. V nabídce bylo uvedeno, že program bude k mání od září 88. V prosinci jsem zašel do prodejny v Martinské ulici. Tam o něm nic nevěděli. Pár dní nato mi z poststudijní vojny volal **Richard Lukeš**. Jak už tomu v rozhovoru dvou assembleristů bývá, řeč sklouzla na strojákové novinky. Zmínil jsem se i o záhadném programu Assembler Z80, aniž bych tušil, že právě mluvím s jeho autorem. Hned jsem ho začal přemlouvat k rozhovoru i k tomu, aby mi svůj assembler "ale i s manuálem!" poskytl pro recenzi do Mikrobáze. Přiznám se, že v ten moment mě spíš stravovala velká zvědavost. Recenzní udičku jsem nahodil proto, abych program z Richarda co nejrychleji vymámil. Stalo se. Recenzi si můžete přečíst na jiném místě čísla.

"Proč se to jmenuje Assembler 80?"

"Nevím."

Podivné, když někdo křtí cizí dítko bez vědomí otce. I když do jisté míry to chápu. Richard svůj program nazval Masm Z80. Jako že makroassembler. Název Masm ve mně budí představu řeznického krámu. A zrovna moc komerčně nezní. Assembler 80 ovšem taky ne - je oněch 80 kus čísla roku vydání či procesoru 8080? Já bych ho nazval třeba...třeba Cross (má přece křížové reference). A což takhle MaCross? Jsou v tom i makra a slangově symbolický "můj kříž". Nó... Nebo Makron (Makarón)? Hlavně krátce a zaujatelně, tedy i zapamatovatelně. Asi bych si ze všeho nejspíš vybral...

"Co bys řekl MaCrossu?"

"Proč ne?"

Než jsme se k němu dostali, vyzvěděl jsem, že Richard (ročník 1964) loni úspěšně zakončil studium oboru elektronické počítače na ČVUT FEL. Změna je život - obor programování, kterým FEL před lety vzala počítače na vědomí, už zanikl.

"Co budeš dělat po vojně?"

"To je tajemství."

Na některé lidi se nemá spěchat. Snad se Richard vylistuje později. Začnu obligátně.

"Jak jsi se dostal k programování?"

"To bylo ještě na průmce, když k nám přišly texasky 58, 59."

V populárním časopise bych musel vysvětlit, že tu nejde o odrůdu kalhot v uvedených velikostech. Ale v Mikrobázi snad...

"A po TI?"

"Sinclair ZX-80 a 81. Brzy nato Spectrum."

Standardní odpověď. Nesmazatelná stopa anglického sira v naší historii.

"Tvoje jazyková výbava?"

"Ve škole jsme z vyšších programovacích jazyků měli jen Fortran. V předmětu strojových jazyků se učily tři - pro systémy PDP 11 a JSEP a pro procesor 8080. Tehdy se mi naštěstí podařilo vyučujícího přemluvit na Z80. Od třetího roku studia jsme už dělali převážně v Pascalu."

"Když se jako absolvent podíváš zpět, čeho se ti ve studiu nedostávalo?"

"Hlavním problémem školy je soulad s vývojem. Přestože se profesoři snažili jít s dobou, v řadě předmětů to dost kulhalo. Začínat Fortranem je už samo o sobě neštěstím. Výuka byla v mnohém zastaralá. Nechyběly ani kolize ze strany vyučujících."

"Čím to?"

"Pro mnohé je to dílem stále novinka, dílem to nestíhají. Pracovník vysoké školy toho má na sobě pořád víc, až se nakonec ničemu nemůže věnovat cele."

"To ale hrozí nebezpečí, že oborem nadšení studenti budou své profesory při přednášce opravovat."

"Tak to se stávalo dost často. Některé přednášky byly velmi kvalitní. Ale byly i takové, kde studenti museli své vyučující opravovat i v naprosto základních věcech."

Mně jsem si takovou situaci představil v medicíně. Ufff... Radši otočím list.

"Jak jsi se spřátelil se Spectrem?"

"Dostal jsem ho dost brzy a už jen tím jsem byl nadšen. Smůla byla, že tu k němu nebyl skoro žádný software."

"Ani hry?"

"Taky ne. Všechno až později."

"Propadl jsi herní fantasmagorii?"

"Nijak moc. Spíš mě zajímalo, jak je to v těch hrách uděláno."

"I když jsi už přešel na péčko, jakou spectrovskou hru by sis ještě dnes zahrál?"

"Nejradši jsem měl mírové hry firmy Psion-Checquered flag, takový ten tenis... Příjemně relaxační je Scrabble. Samozřejmě nemohu opomenout tituly Manic Miner, Jetpac a sérii trojrozměrných her firmy Ultimate... Moc zajímavý materiál pro animaci."

"Na péčku už si nehraješ?"

"Ne. Nějaké hry tu sice jsou, ale většinou se nedají srovnat s těmi pro Spectrum, ani nepřinášejí nějaký nový nápad."

"Co jsi dělal se Spectrem v době, kdys ho neměl čím napustit?"

"Leccos jsem si musel udělat sám. Od začátku mě to táhlo k systému. Radši píšu systémové miniatury, než abych tvořil rozsáhlé celky. Samozřejmě jsem začal Basicem, ale brzy jsem propadl strojáku."

"A po přechodu na PC?"

"Tam jsem Basic zcela vyloučil. Dokonce jsem si přísahal, že Basic už nikdy! Zpočátku to byl hlavně Pascal. Jenže ten svou normou není pro spoustu věcí vhodný. Časem ze všeho přirozeně vyplynulo Céčko. Pro mou problematiku je to vlastně jediný rozumně použitelný jazyk. A mohu v něm řešit i všechno ostatní. Někdy možná ne tak elegantně, ale v každém případě určitě."

"Jakou konkrétní problematikou už jsi se zabýval?"

"Coby pomocná vědecká síla jsem při studiu dělal v laboratoři smyslových náhrad. Tam jsem se zabýval způsoby komunikace počítače s okolím. Obecně šlo o řízení hardwaru a zpracování informací."

"V té laboratoři...to byly jaké počítače?"

"Výhradně Spectra."

"Cože?! V tak vznosném institutu a pro tak vznosnou činnost?"

"Kupodivu ano. Dostali jsme tam taky vzorky IQ 151 a PMD 85, ale ty jsou naprosto nepoužitelné už kvůli své nespolehlivosti."

Vždy, když něco takového slyším, tak... Však víte.

To, že rozhovor stále přeskakuje mezi ZX Spectrem a péčéčkem, není náhoda. Pokud si někdy budete moci koupit MaCross (či jak se bude jmenovat), možná vás bude zajímat, že:

"Všechny zdrojové texty pro MaCross jsem psal na péčéčku."

A je to tady. Vývoj programu pro nižší systém na vyšším. A v opravdu domácích podmínkách.

"Jak jsi zajistil komunikaci Spectra s péčéčkem?"

"To mělo dvě fáze. V první jsem psal zdrojáky na péčéčku a přenášel je na microdrivy. Ve Spectru jsem měl základní jádro pro překlad, který v něm probíhal s pomocí microdrivů. Jenže ty mi poměrně brzy vyhořely. Ve druhé fázi jsem si udělal komunikační linku mezi Spectrem a péčéčkem. K překladovému jádru ve Spectru přibyla rutina pro komunikaci mezi oběma počítači. Překlad zase probíhal ve Spectru, ale texty jsem upravoval už jen na péčéčku."

"Takže jsi měl neustále k dispozici aktuální, "nezbouratelnej" text. Po krachu programu nenásledovalo opakované a protivné načítání všeho ze všeho. Jak ti závidím..."

"Za zmínku stojí i to, že poslední dvě třetiny programu překládalo už jeho vlastní jádro."

"Ale když ti odešel program ve Spectru, z čeho jsi pak načetl to jádro?"

"Z kazety. S tím souvisí problém přenosu z péčéčka do Spectra přes RS232. Spectrum ho nemá přesně podle normy. Při přenosu se občas objeví rušový moment, který změní obsah bajtu. To pak pochopitelně celou práci zpomaluje."

"Který způsob překladu jsi při přenosu textu z péčéčka používal?"

"Vkládání, známé pod anglickým povelem include. Text pro překlad šel do Spectra po osmi drátech. V péčéčku jsem ještě měl rezidentní program pro určitou transformaci textu, protože Spectrum používá trochu jiný formát než editory péčéčka."

"Jaký editor jsi používal?"

"Začal jsem editorem od Turbo Pascalu, ale přestal mi stačit svou kapacitou. Pak to byl Micro Star Borlandu z balíku Toolbox. I ten přestal stačit. Nakonec jsem přešel na Nortonův editor. Ale i u něj jsem se dostal do úzkých."

"Malá paměť počítače?"

"Mám jen půl mega. Těch 128 kilo chybělo dost citelně."

"Kolik má zdroják MaCrossu?"

"Asi 200 kilo. Jenže spolu s operačním systémem, editorem a podpůrným programem to bylo na doraz. Musel jsem začít zkracovat. Třeba všechny podmíněné překlady pro verzi 8080 jsem musel vystříhnout."

"Co tě vedlo k tomu, pustit se do assembleru pro Spectrum? Vždyť jich pro něj není zrovna málo."

"Myslím, že pro takový počítač je assembler ideální program. Když pro Spectrum uděláš databanku, tak to pořád není databanka - není paměť, nejsou disky. Rezidentní programy jsou nepoužitelné, protože Spectrum nemá operační systém. I textový editor naráží na kapacitu paměti. Ta u assembleru není až tak rozhodující. Svou roli sehrálo i to, že jsem nějakých pět let dělal na Spectru výhradně v assembleru. Dost dlouhá doba na to, aby si člověk vytvořil představu, co by takový program měl umět. Za jednoznačně nejlepší assembler pro Spectrum považuji Gens firmy Hisoft, jenže má řadu nedostatků. MaCross má všechny jeho dobré vlastnosti a navrch něco z komfortu, jakým se vyznačují assemblery počítačů vyšší třídy."

"Jak dlouho jsi MaCross dělal?"

"To nedokážu odhadnout. Časově mi to splynulo s prací na diplomce. Když jsem ji odevzdal, programoval jsem každý den. Až do druhé půlky září, kdy jsem program odevzdal 602.ZO."

Tak i proto nemohl vyjít hned v září. Jenže máme konec ledna 89... A to ještě Richard dosud nedostal honorář za přijatý program. Dobrý návod na to, jak odradit všechny ostatní.

Už klasicky mě u každého programátora zajímá všechno, co souvisí s průběhem jeho tvorby i přístupem k ní. Zajímavě o tom mluvil Petr Adámek v Mikrobázi 1-3/88. Byl jsem zvědav na Richarda:

"Ty myslíš jako jestli shora dolů nebo zdola nahoru...?"

"...i odprostřed všemi směry..."

"Aha... Já se dívám shora dolů, abych věděl, co potřebuju, ale vlastní realizaci dělám zdola nahoru. Začínám takovými základními jadérky, ta pak nabaluju a nabaluju, až z toho vzniknou silné rutiny, které už se dají používat."

"Ty tedy nezačínáš třeba ovládáním z klávesnice?"

"Tím se u nás začíná dost často. Zatímco venku je zvykem provádět delší analýzu celého problému. Pokud jde o průběh tvorby, občas se dostanu do situace, že pár dní nevím, kudy kam, nemám jasný cíl... Tehdy je analýza nutná. Když je mi zřejmý cíl i metoda jeho realizace, mohu si sednout rovnou ke klávesnici."

"Počkej, ale snad bys měl mít napřed nějakou představu o ovládání programu z klávesnice a vnějších datech. Z toho se odvíjí celý spodek."

"To je už ta téměř nejvyšší úroveň. Já k ní dospívám zespodu. Napřed dělám ty úplně základní, naprosto elementární rutiny. Většina z nich se dá velice těžko odzkoušet, když ještě nad sebou nemají tu nadstavbu, která je používá. Když to pak nabalím do základního vstupně výstupního jádra, tak už jdu do vyšších úrovní, pomocí kterých pak vidím, co se dole děje. Cesta zdola nahoru má tu nepříjemnou vlastnost, že hrozně dlouho nevidíš přímé výsledky funkcí toho spodku."

"Zajímavé. Přesně opačná cesta, než jakou zakouším já a někteří další. A jak se bráníš ztrátě orientace, když program houstne?"

"U MaCrossu se projevila výhoda péčéčka. Ve Spectru jsem s programem sotva začal a už jsem ho měl ve čtyřech kusech. Zatímco na péčéčku jsem měl stále k dispozici celý text s řadou komentářů. Editor mi navíc umožňoval označit si libovolná místa textu a kdykoli na ně skočit."

"A grafické pomůcky? Jako vývojové diagramy, struktogramy, kopenogramy...?"

"Ne, to ne. Pokud jsem měl problémy s vývojem

nebo optimalizací algoritmu, napsal jsem si ho v Pascalu na péčéčku a výsledek jsem pak přepsal do assembleru. Trochu atypická cesta, ale na odzkoušení to stačí."

"Papírová dokumentace žádná?"

"Jo, to jo! A rozsáhlá! Ta je nutná. Papíry s poznámkami jsem měl po zdech kolem sebe. Nejen systémové proměnné, stavové bity a bajty a tak dále, ale i seznam vytýčených a splněných úkolů. I to je nutné. Náročné úkoly si člověk musí stanovit ještě v době, kdy má dost energie na jejich zvládnutí."

"Vrhal ses někdy takovou tou zběsilou cestou, kdy napíšeš nějakou subrutinu a říkáš si - tohle musí jasně fungovat, šoupnu to rovnou do programu a basta! Nebo máš takový ostražitější, analytičtější způsob tvorby?"

"Nevím, jestli jsem zběsilý nebo ostražitý, ale když překládáš dlouhý zdroják, tak už to nějakou minutu trvá. Tehdy je lepší si to přece jen prohlédnout."

"Tys překládal vždycky celý text? Neměl jsi ho rozlámaný třeba do knihovny nebo na hotové kusy strojáku?"

"Vždycky celý. Jedna z mých prvních myšlenek směřovala k provádění překladu přímo na péčéčku, což by asi bylo rychlejší. Ale dodnes nemám žádný vhodný program, který by to uměl."

"Jak dlouho trval překlad těch 200 kil?"

"Tak asi pět, šest minut."

"Jak jsi hledal chyby? Používal jsi debugger ve Spectru?"

"Velká část chyb je obvykle rázu estetického- jako třeba špatné umístění nápisu apod. To vychytávám v jedné hromadě až nakonec. Původ některých chyb se dá uhádnout. Něco změníš, program dvakrát, třikrát přeložíš a je to. Debugger jsem použil jen v krajních případech."

"Jaký?"

"Zásadně Mons. Nikdy se mi nestalo, že by nestačil."

"A Laser Genius?"

"Jen Mons. Jednak - zvyk je zvyk. A svou roli hraje i délka debuggeru, který jsem v dané konfiguraci musel načítat z pásku."

"Co se ti na Gensu nelíbilo?"

"Především editace textu. To se snad za editaci ani nedá považovat. Petr Adámek udělal sérii editorů pro jednotlivé překladače Hisoftu - Pascal, Céčko, assembler... Ale i ta jeho úprava assembleru se mi nezdála být ideální. A v paměti se zbytečně válelo několik nevyužitých kil. V době, kdy Petr Adámek úpravu provedl, byla přímo revoluční. Ale když jsem se pak dostal k péčéčku a jeho editorům, nemohl jsem odolat, abych nezačal přemýšlet o tom, jak spectrovským programátorům ulehčit dřinu. Nejdřív jsem se rozhodl udělat full screen editor. V době, kdy tyto editory vyžadovaly ruční tabelaci, byly skoro rychlejší řádkové. V mém editoru je tabelace automatická. Díky tomu lze programovou řádku zapsat stejně rychle jako v řádkovém. Práce s bloky je inspirovaná editorem firmy Borland. Stala se mnohem rychlejší a přehlednější. Běžnou věcí je projekce menu, aby člověk nemusel listovat v manuálu. Doplnil jsem i to, co v Gensu katastrofálně chybělo - křížové reference, lokální bloky a makroinstrukce s parametry. Makra sice v poslední verzi Gensu byla, jenže zcela nepoužitelná. MaCross taky pracuje s polovinami indexových registrů. Je kompatibilní s Gensem do té míry, že může použít texty Gensu, pokud neobsahují nová zakázaná slova MaCrossu. Při využití schopností MaCrossu, které Gens nezná, zpětný převod pochopitelně není možný."

"Znáš úpravu editoru Gensu od Jiřího Lamače? Jeho full screen editor má taky automatickou tabelaci."

"Já jsem dělal s Adámkovým editorem, tenhle neznám."

"Je zdrojový text MaCrossu komprimován?"

"Není."

"S jakými typy externí paměti MaCross pracuje?"

"S kazeťákem, microdrivem a interfacem Disciple pro disk."

"Každý, kdo udělá nějaký větší program, se k němu obvykle vrací. Dostává neodolatelné nutkání vylepšit ještě tohle a tamto... Co ty a tvůj MaCross?"

"Já bych ze všeho nejvíc uvítal, kdyby uživatelé MaCrossu měli disk a stejný operační systém. To by pak směr vylepšování byl zcela jasný. Jinak už pár nápadů mám. Týkají se nepatrných úprav editoru, rozšíření pseudoinstrukcí, organizace tabulky symbolů na drivu, vylepšení maker, respektivě vytvoření maker řetězcových... V tomhle směru nejvíc napoví až reakce programátorů po uvedení MaCrossu do života. Rozvíjet se dá pořád. Jen kdyby byla paměť! Jinak je člověk tvůrcem kompromisů."

"Stihneš to při práci, která tě po vojně čeká?"

"Nevím, jak to v tom výzkumáku bude vypadat. Tam je to z 90 procent zabezpečování provozu počítačů a tiskáren, implementace češtiny do všech programů... A vlastně ani nevím, jestli nebudu dělat něco úplně jiného..."

Vida, listing budoucích záměrů začíná vyjíždět...

"Jak by sis třeba představoval takový ideální softwarový trh? A kdyby existoval, šel bys na volnou nohu?"

"Zcela určitě. To by tu ale musely existovat specializované organizace a hodně toho umět. Od uzavření jasné smlouvy s programátorem přes kvalifikované posouzení programu až po jeho distribuci na trhu. Nezbytnou složkou tohoto řetězce jsou třeba firmy, které nedělají nic jiného, než že testují programy daleko před jejich uvedením na trh. Tak tvůrcům pomáhají výrazně zkrátit dobu ladění programů a vůbec přispívají ke zjednodušení a zrychlení procesu tvorby i ke zvýšení spolehlivosti hotových produktů. Dalším důležitým článkem produkce jsou odborné týmy pro tvorbu manuálů k programům. Uroveň manuálu je klíčem k efektivnímu využití programu. Napsat opravdu dobrý manuál je umění. Proto se svěřuje specialistům. Nechci tu v absolutních číslech porovnávat hrozný nepoměr mezi tím, co programátor dostane za program tady a co venku. Programátor na volné noze věnuje vlastní tvorbě třeba půl roku, dalších několik měsíců program ladí, vylepšuje a píše manuál. Během té doby nejspíš nebude mít žádný příjem. Proto musí mít ve všem předem jasno. Honorář by neměl dostávat jen proto, aby se necítil příliš sláb a měl se vůbec do čeho obléci. Honorář programátora tu prakticky padá na nutnou inovaci pekelně drahého technického vybavení a stále hladové knihovny. A co když má ještě rodinu? Programování je dřina a dokáže člověka vysát. Taky nejde jeden den odevzdat hotový program a druhý den ráno vstát a začít ťukat další..."

"...jistě, nejsme Alexader Dumas. Myslíš, že lze udělat opravdu dobrý program vedle hlavního zaměstnání?"

"Dost těžko."

"A je programátor, který v zaměstnání programuje za běžnou gáži, stimulován k tomu, aby podával maximální výkon?"

"Programování není kusová výroba u pásu. Kvalitu programové tvorby podmiňují nejen znalosti, talent a píle tvůrce, ale i interaktivní, dynamické prostředí soutěživosti s přirozenou snahou udržet se na špičce vývoje a tomu odpovídající mírou přístupu k programátorům. Bez toho to nejde. Ovšem to je problém mnohem širší. Proto bych chtěl popřát mnoho štěstí všem software housům."

"Jakým?"

"Těm, co tu ještě nejsou."

Rozmlouval -elzet-

ASSEMBLER JAKO VÍNO

Jako mladé víno. A říkám to coby letitý abstinent. Jde o assemblerový burčák ze sklípku Richarda Lukeše. Před jeho cestou do spectrovských pohárů mu horliví propagátoři dali hned tři viněty - asmb.80, Assembler 80, Masm-Z80. Já mu v rozhovoru s autorem začal říkat MaCross (viz str.2). Zůstanu u toho - stejně dosud nikdo neví, jak se vlastně bude jmenovat.

Editor

Rodina assemblujících spectristů má celkem slušný výběr zahraničních produktů. Nejoblíbenějším stále zůstává stařičký Gens firmy Hisoft. 602.ZO jej nabízí v upravené kombinaci s Monsem pod názvem Dr.MG. Kamenem úrazu původního Gensu je editor. S jeho podstatným vylepšením přišli nezávisle na sobě Petr Adámek a Jiří Lamač. Každý na to šel jinak. Trochu jinak na to šel i autor Macrossu.

J.Lamač zvolil příjemné písmo s délkou řádky 42 znaků. P.Adámek vsadil do svého editoru písmo Tasswordu (64 zn/ř). I Macross má tasswordovské znaky, ale na řádku se jich vejde 80. Její opačný konec proto místy odplouvá z obrazovky. Ale na tiskárně se standardem 80 zn/ř je to ono. Adámkův editor je řádkový, Lamačův i Lukešův celooobrazovkový. Nejpodstatnější při editaci je její svižnost. A ta je společná všem třem. Osobně dávám přednost editorům celoobrazovkovým před řádkovými. I proto jsem dosud užíval Lamačův. Ten mimochodem píše i bíle na černém. Macross jen černě na bílém.

Po načtení MaCrossu do počítače se v jedné řádce ohlásí menu:

Ass Bas Code Del Edit Get Incl Mem Name Put Run
Text Ver Wrt Xrf

V tomto hlavním menu (stejně jako v dalším-editorovém) nevolíme žádným kurzorováním. Zkrácené názvy volitelných funkcí slouží k základní orientaci (není třeba listovat manuálem nebo hledat dobře založený tahák). Je to tedy něco ve smyslu help menu. Volba se provádí stiskem tlačítka s prvním písmenem názvu funkce. Teď zmáčkneme E a vstoupíme do editoru. V první řádce je informace:

Line 1 Col 1 Error 0 Overwrite < >

R.Lukeš se důsledně držel anglických názvů. Proti tomu nic nemám. Při setkání s různými editory upravenými do češtiny nebo slovenštiny často nevím, co zjevivší se informace vlastně znamená. Mezi nápisy Error (s číslem poslední chyby) a Overwrite se po stisku CAPS LOCK objeví nápis Capitals (psát můžeme malými i velkými písmeny). Mód Overwrite se jedním tlačítkem přepíná s módem Insert - nápis se adekvátně změní. Ve volném prostoru mezi znaky < > se objeví jméno textu po jeho prvním zadání (pro práci s vnější pamětí). Line je číslo řádky, Col sloupce. Tak máme povšechnou informaci stále před očima.

Zápis programové řádky se řídí podobnými pravidly jako v Lamačově editoru. Když píšeme návěští, zapisujeme je hned zkraje. Mezi vším ostatním stačí udělat jen jednu mezeru. Při zápisu řádky bez návěští na jejím začátku odklepneme jednu mezeru (nebo třeba víc). Editor si po stisku ENTERu rozhodí řádku na obrazovce sám. Když třeba zapíšete:

```
START LD A,24 ;Do A počet řádek
 SUB B ;Kolik řádek volných?
```

editor z toho udělá:

```
START   LD A,     24          ;Do A počet řádek
        SUB       B           ;Kolik řádek volných?
```

Tak jde zápis svižně od ruky. Kurzorem můžete poskakovat všemi směry, skákat o jednu obrazovku nahoru i dolů a na začátek nebo konec řádky i celého textu. Jakož i mazat znaky, řádky a vložit řádku volnou. Po stisku obou shiftů se pod obecnou informací objeví řádka druhého menu:

Start End Copy Delete Move Put Gost Wrt
Label Find Repl Quit

Prvních osm názvů se týká operací s bloky textu. Určit lze jeden blok jeho začátkem a koncem (ten musí být "níž"). Blok lze zkopírovat na jiné místo textu, vymazat, přemístit, ale i - zapsat na vnější paměť či vypsat na tiskárně! Operace probíhají velmi rychle. Po definici bloku se všechny řádky v jeho rozsahu rozsvítí (bright). To pro posledně definovaný blok platí i v průběhu libovolné manilupace s textem na obrazovce. Jakmile najedeme na blok, hlásí se svou září. Gost (Go to start) nás hned hodí na začátek bloku. Put zapíše blok na vnější paměť. Wrt vypíše blok na tiskárnu. Vše dohromady velká pohoda oproti jiným editorům assembleru pro ZX Spectrum.

Další tři funkce slouží k zadání, hledání a případné výměně řetězce znaků. Operace probíhají od pozice kurzoru směrem ke konci textu. Opět velmi svižné.

Vše, co se zadává, zůstává k dalšímu použití

(přístě se to nemusí zadávat znova).

Tlačítkem Q se dostaneme do hlavního menu. Jiným, než představuje první písmeno funkce v menu, se vrátíme k editaci.

Při zkusmých manipulacích s editorem a jeho funkcemi jsem nenarazil na žádný problém. Ale mám pár drobných výtek:

Vadí mi dlouhé čekání na první echo stisknutého tlačítka. Pro znalého detail, začátečník si holt zvykne. Což takhle dát přímou možnost úpravy? Pomalé scrollování textu čechrá nervy.

Někdo (včetně mne) může u editoru assembleru citelně postrádat posun textu řádky od kurzoru doprava, mazání znaků na pozici stojícího kurzoru s posunem zbytku řádky doleva a výmaz zbylé části řádky od kurzoru do jejího konce.

Za trochu násilné považuji nucené umístění komentáře za instrukcí od 46. sloupce řádky (do konce přímo viditelného pásma zbývá jen 18 pozic). Výsledek je sice úhledný, ale proč to uživateli nutit? Když už, tak nabídnout jako eventualitu.

Po prohledání textu (Label, Find, Repl) se objeví prázdná obrazovka. Kurzor je nahoře (za poslední, skrytou řádkou textu) - a to i když je hledání neúspěšné (co jen tam ten kurzor pohledává?). Uživatel tak musí lopotně zpátky do místa, odkud hledal. Může si sice výchozí místo hledání předem nadefinovat jako začátek bloku, ale to jaksi není ono. V každém případě by měl mít možnost návratu na místo, odkud hledal. Stejně jako možnost jednoduché definice aspoň jednoho místa, kam by se mohl vrátit kdykoli a odkudkoli.

Nepříjemně působí zmizení kurzoru z obrazovky po stisku obou shiftů - zvláště při definici bloku. V ten moment si nejsem jist, jestli jsem vše provedl správně. I o začátku bloku může mít člověk pochybnosti, protože o něm po jeho stanovení nemá žádnou informaci (blok by měl rozlévat/hasit své světlo hned po určení jeho začátku a během natahování/zkracování bloku až do konce).

A poslední detail, který je spíš jen programátorským prohřeškem na kráse - když je na obrazovce zářící blok a zadáme hledání řetězce, pak - není-li nalezen - z prázdné obrazovky na nás zůstanou civět zářící obrazové atributy původního bloku.

Většina těchto výtek nijak zvlášť nesnižuje kvality editoru. Ale v další verzi by některé z nich určitě měly (jiné mohly) být zažehnány.

Assembler

Z prvního menu volíme překlad (Ass), který probíhá stejně rychle jako v Gensu. Do textu můžeme umístit řadu řídících povelů. *C, *D, *E, *F, *H, *L, *S jsou stejné jako u Gensu. Povel *M vypíše obsah makroinstrukcí, při *W se výpis zastaví a na obrazovce se objeví u něj zapsaný řetězec (např. upozornění na provedení potřebné akce apod.).

Klasické pseudoinstrukce obohacuje DEFL (Define Label) pro definici návěští. Je to obdoba EQU, ale s tím rozdílem, že hodnota stejnojmenného návěští může být v průběhu překladu kdykoli změněna (není tedy hlášena chyba vícenásobné definice). Jedna editační příjemnost - místo EQU lze psát jen rovnítko. Všechny pseudoinstrukce začínající písmeny DEF lze psát ve zkrácené podobě (bez EF)- např. DB, DW, DL apod.

Pro podmíněný překlad jsou k dispozici běžné pseudoinstrukce IF, ELSE, END. Samozřejmě nechybí ORG, ENT, DB, DW, DS, DM.

Novinkou jsou MAC, ENDM, LOC, ENDL. MAC upozorňuje překladač, že vše, co následuje až do ENDM, je obsah makroinstrukce se jménem zapsaným na místě návěští. Např.:

```
DOZAS  MAC
       PUSH AF
       PUSH BC
       PUSH DE
       PUSH HL
       ENDM
```

Kdykoli v textu použijeme návěští makra na místě instrukce, překladač ji v tom místě v paměti rozvine do strojového kódu. Makra mohou mít i parametry. Např.:

```
OBSAZ  MAC                    ;Vynulování úseku paměti
       LD HL,=1
       LD D,H
       LD E,L
       INC DE
       LD BC,=2
       LD A,=3
       LD (HL),A
       LDIR
       ENDM
```

Třeba pro vymazání obrazovky ZX Spectra pak stačí použít:

```
OBSAZ 16384,6143,0
```

První číslo je hodnota prvního parametru, druhá druhého... V každém makru může být nejvíc 16 parametrů. Definice maker mají některá omezení, o nichž se dozvíte z manuálu. Obsah maker se ukládá do makrobufferu, jehož velikost si programátor určuje funkcí Mem z hlavního menu.

Vše, co je umístěno mezi LOC a ENDL, je lokální blok. Jeho výhodou je, že v něm můžeme používat návěští platná jen uvnitř tohoto bloku. Tak se nemusíme při spojování různých částí programů starat o to, zda obsahují stejnojmenná návěští. To se týká např. práce s moduly, které máme hotové a často je do různých programů zařazujeme. V makru vnitřní návěští používat nemůžeme (hned při druhém použití takového makra by návěští bylo označeno jako vícekrát definované). Zvláštnost zápisu lokálních návěští je v tom, že jim musí předcházet dvojtečka:

```
ZPOZD  LOC
       LD B, 255
:SMYC  DJNZ  :SMYC             ;Lokální návěští
       RET
       ENDL
```

Tento blok můžeme z programu spustit třeba instrukcí CALL ZPOZD. Z vnějšku lokálního bloku však nelze volat návěští :SMYC. Ale kdekoli v programu můžeme mít návěští SMYC, které bude mít zcela jinou hodnotu než :SMYC. MaCross umožňuje použít až 256 lokálních bloků. Jejich zařazení nevyžaduje žádný zvláštní buffer. Další výhodou lokálních bloků je možnost jejich vnořování do sebe. Dokonce názvy jejich návěští se přitom mohou shodovat. To napovídá, že z jednoho bloku nemůžeme volat návěští jiného. Maximální počet vnoření je (zcela postačujících) dvacet.

A ještě jedna pěkná možnost - v případě potřeby použití návěští uvnitř makra do něj zařadíme lokální blok a tím to vyřešíme:

```
LABEL  MAC
       LOC
       ...                     ;Blok s různými
       ...                     ;lokálními návěštími
       ENDL
       ENDM
```

Při více než jednom použití takového makra nebude při překladu hlášena chyba vícenásobného použití návěští.

Pro překlad textu můžeme volit kombinace sedmi možností, podobně jako v Gensu. Navíc je zařazena volba výpisu souboru EQU. Ten má úzkou spojitost s křížovými referencemi (opět tolik potřebná novinka). Po volbě souboru EQU si vyberete z kombinace

dalších šesti možností. V podstatě jde o volbu toho, jaká návěští mají být do souboru zařazena- vybraná, definovaná, nedefinovaná, lokální. Vybraná si programátor přímo v textu předem označí podtržítkem před jejich názvem. Definovaná návěští jsou ta, u kterých překladač mohl určit jejich hodnotu. Nedefinovaná jsou toho opakem. Lokální jsou ve výběru jaksi navíc, ale mohou sloužit pro větší přehled v programu. Jedna z voleb umožňuje zařazení zvolených návěští za konec zdrojového textu. Soubor EQU můžeme zapsat na vnější paměť funkcí Xref a pak jej kdykoli načíst do počítače. Tak můžeme přenášet hodnoty návěští z jednoho textu do druhého. To je výhodné v mnoha případech. Můžeme tak kombinovat funkci *F (vkládání textu s jeho průběžným překladem) s postupným překladem celých (nebufferovaných) textových částí, které se nám spolu nevejdou do paměti. Nebo mít část programu uloženu jako hotový strojový kód a na něj se odvolávat v textu, který zrovna vylepšujeme atd. apod. Při dobré organizaci tvorby programu jsou křížové reference velkým přínosem hlavně pro spectristy, kteří nemají žádnou svižnou vnější paměť. Dvakrát zapisovat bufferovaný text na magnetofon a zase jej načítat dvakrát po sobě z magnetofonu je činnost hodná Mistra Zenu.

Překladač má jednu příjemnou vlastnost - když při překladu objeví chybu, ihned vás vrátí do editoru s kurzorem na řádce, která chybu vyvolala. A u nápisu Error máte číslo chyby.

Macross spolupracuje s magnetofonem a microdrivem. Funkce prvního menu Get načte zdrojový text do počítače, Put jej zapíše na vnější paměť, Ver provede verifikaci zápisu zdrojového textu. Pomocí Nam můžeme přejmenovat posledně použitý název zdrojového textu. Tato funkce se mi zdá být zbytečná, protože změněný název při Put, Incl i Get se hned vztáhne ke všem třem úkonům (po jejich vyvolání vám program nabídne vždy posledně zadaný název). Podobně (a nezávisle na názvu pro Put, Incl a Get zdrojového textu) nemizí posledně zadaný název pro zápis bloku, souboru EQU nebo strojového kódu funkcí Code. Ta na vnější paměť zapíše kód posledně přeloženého zdrojového textu, resp. jeho bloku, resp. poslední části textu, která obsahuje ORG. Funkce Nam má jakýs takýs smysl pouze pro opravu názvu zdrojového textu, když jsme jej nuceně změnili před přičtením souboru EQU nebo bloku textu.

Bas vrací do Basicu (zpět do MaCrossu bez ztráty textu se dostanete příkazem GO TO 1, resp. RANDOMIZE USR 25000). Del maže celý zdrojový text (jeho části, nebo i celý vymažete blokovými funkcemi). Po volbě funkce Mem nejdříve stanovíte velikost bufferu pro práci s funkcí Incl (resp. microdrivem) a povelem *F a poté ještě rozsah bufferu pro makra. Run spouští váš program od adresy ENT. Funkce Text poskytuje přehnaně skoupou informaci - jen o první a poslední adrese uloženého textu.

Macross není přemístitelný (začíná na adrese 25000). To by až tolik nikomu nemuselo vadit. Kromě mne - mám wafadrive, který obsazuje ramku za adresu 26000.

Nakonec drobnůstka, která potěší - změnou obsahu jedné adresy si můžete stanovit počet sloupců výpisu tabulky symbolů. Macross je startovně naladěn na projekci čtyř sloupců.

Manuál

má přibližně 60 stran. Je psán stručně, hutně, obsahuje všechny nezbytné informace pro ovládání programu. Ale nemohu se zbavit dojmu, že pro začátečníky bude místy dost nestravitelný. Chápu, že manuály k zahraničním assemblerům nemusejí obsahovat zcela vyčerpávající výklad všeho. Stačí totiž zajít na roh do knihkupectví a přinést si náruč knih, které detailně popisují, co je to makro, lokální návěští, jak a kde je nejlépe použít, co se přitom stane atd. U nás taková možnost neexistuje. O to víc by autoři i vydavatelé manuálů měli na to brát ohled. Myslím, že opačným přístupem se sami připravují i o značný díl zisku z prodeje. Při kopírování všeho všemi je u nás program samotný snadno dostupným artiklem. Zato dobře napsaný a pěkně graficky zpracovaný manuál je biblí, po jaké prahne každý uživatel dobrého programu.

Manuál obsahuje několik drobných příkladů, ale ani jeden, který by začátečníkům demonstroval práci s MaCrossem na programku s nějakými čtyřiceti řádkami. S MaCrossem bude dodáván program Had, který si sice každý může prohlédnout i spustit, ale to mu pro vlastní praxi dá jen velice málo. Tento krok se mi vůbec nezdá být šťastný. Had je původní anglický program typu public domain. R.Lukeš v něm nechal anglické komentáře. A neobsahuje ani jedno makro, ani lokální blok. Značně samoúčelná nepůvodnost. Prostě had.

V manuálu jsem nikde nenašel jasnou informaci, jak se do MaCrossu vrátit z Basicu (a co se přitom stane nebo nestane se zdrojovým textem). Musel jsem to vytušit a risknout. Do Basicu spadnete i při přerušení funkcí pro čtení/zápis na vnější paměť. Nejen že o tom manuál taktně mlčí - něco takového by program vůbec neměl dělat. Nikde žádná informace o délce programu (je to cca 14K). Méně znalý majitel microdrivu z manuálu neuhodne, jak je to s bufferováním textu na jeho zařízení, když Incl je jen pro mgf (ne že by mu to bez takové znalosti nefungovalo, ale přece jen...). Chybí více detailů pro znalce (ve stylu autorem dobře zpracovaného přehledu o formátu textu). Jedna mikrosubkapitolka věnovaná problému komunikace s tiskárnou přivede mnoho začátečníků do úzkých, s problémem nehnou. A stačilo by pár vysvětlujících slov (novic nemusí tušit, že přenos znaků jde "samočinně" registrem A; v manuálu uvedená rutina INIT ho vyděsí, protože se nikde nedozví, že, proč a jak ji má spouštět pro inicializaci tisku, kam vůbec s ní, jak si udělat pracovní kopii atd.). Protože mnoho spectristů má různě nekompatibilní zařízení vnější paměti, bylo by nanejvýš vhodné zavést možnost jejich ovládání z Basicu. U změny obsahu adresy pro stanovení počtu výpisu sloupců tabulky symbolů není uvedeno, kolik se jich vejde na šířku 80-znakové řádky a co když na onu adresu dá třeba 167. Každého tak čeká zbytečné experimentování. Ještě k přehledovým tabulkám ovládání- jsou rozházené, uživatel bude nucen si je přepsat na zvláštní papír.

Pokud bude manuál vydán tak, jak jsem jej dostal k recenzi, bude mu citelně chybět redakční jazyková úprava. Místy v textu chybějí slova. Ale to by nebyla vina autora.

Suma sumárum - pro zkušené praktiky je manuál napsán dost srozumitelně.

Závěr

Editor MaCrossu je dobrý a k opravdovému komfortu na úrovni možností ZX Spectra mu chybí jen málo. Velkým skokem kupředu je vlastní assembler s řadou vynikajících možností. Nepochybuji, že po jeho uvedení do života se valná část uživatelů dosavadních assemblerů promění v MaCrossany. Lepší assembler pro ZX Spectrum neznám. Svými schopnostmi se mu může rovnat jen Laser Genius, který však zdaleka nejde tak pěkně do ruky (a nemá lokální bloky). Práce s MaCrossem je o hodně rychlejší a přehlednější. Srovnám-li však manuály obou programů, MaCross prohrává 10:1. K programu samotnému nemohu, než říci, že jeho autor vytáhl z rukávu křížové eso (příští už bude srdcové?).

Při dobré a svižně zajištěné distribuci se MaCross může stát hitem.

-elzet-

DŘU, DŘEŠ, DŘEME... CÉČKO (2)

Pro základní orientaci v jakémkoli programovacím jazyku platí:

Kudy? kam? co? = programové řízení + přenos parametrů + inicializace

Základy programového řízení máme za sebou. Patří k němu i cykly a výběry (jich se dotkne příští část seriálu). Poslední dva otazníky zahrnují i předběžnou (startovní) deklaraci či definici budoucích přenášených parametrů. Kompilátoru Céčka musíme předem sdělit, co bude číslem, co znakem, co makrem, jaký bude rozsah jejich platnosti atd. Tímto obsažným tématem se budeme zabývat průběžně. Ale abychom vůbec měli co přenášet, uvedu aspoň nezbytné deklarační minimum.

Pro znalce Pascalu to opět nebude nic nového. Ale assemblující a basicující fandy čeká výrazná novinka. V Basicu přiřazujeme hodnoty různým typům proměnných přímo (LET a=3 nebo LET x$="z"). U assembleru není skoro vůbec o čem mluvit. Céčko vyžaduje, abychom před operacemi s proměnnými napřed sdělili, jakého typu jaká proměnná bude:

```
main ()
{
int čí1,čí2,čí3;
čí1=11; čí2=22; čí3=33;
printf("Čísla: %d,%d,%d",čí1,čí2,čí3);
}
```

Int (integer) určuje, že proměnné za ním uvedené budou celočíselné (tedy bez desetinných míst). Zatím budeme operovat jen s nimi, abychom nerozptylovali svou pozornost.

I pro pascalské je v těle funkce main novinka. V argumentech funkce printf jsou páry znaků %d. Pro tisk proměnných musíme předem určit jejich formát (také proto se knihovní funkce tisku jmenuje print format). %d znamená: číslo, které se ke mně svým pořadím vztahuje, vytiskni v dekadické podobě. Prvnímu páru %d zde přísluší čí1, druhému čí2, třetímu čí3. Formáty se píší do uvozovek stejně jako text, který chceme nechat vytisknout přímo. To m.j. znamená, že vše, co je v rozmezí uvozovek zapsáno mezi formáty, se rovněž vytiskne (zde jsou to čárky). Zásadně si pamatujte, že proměnné musíte před jejich tiskem formátovat. Když na nějakou zapomenete, pak se vám nevytiskne, přidáte-li něco navíc, vyjde nesmysl.

Výsledný výpis argumentů funkce printf bude:

Čísla: 11,22,33

V této části uvedené prográmky nejsou žádnými klenoty. Vedle čistě demonstračního charakteru splňují účel zažití céčkové gramatiky.

Přenos parametrů argumenty funkce a vrácením hodnoty

Představme si, že v programu budeme chtít mít funkci pro sečítání tří celočíselných proměnných. Dáme jí název součet. Do takové funkce musíme nějak přenést momentální hodnoty všech tří proměnných, aby s nimi funkce mohla patřičně naložit:

```
main ()
{
int čí1,čí2,čí3,čí4;
čí1=11; čí2=22; čí3=33;
printf("Čísla: %d,%d,%d",čí1,čí2,čí3);
čí4=součet(čí1,čí2,čí3);
printf("\nSoučet: %d",čí4);
}
součet(x,y,z)
int x,y,z;
{
int a;
a=x+y+z;
return (a);
}
```

Výsledný výpis:

```
Čísla: 11,22,33
Součet: 66
```

Při přenosu parametrů mezi oběma funkcemi dochází k "převleku" přenášených parametrů. Volací příkaz součet má v závorce uvedeny argumenty funkce součet s názvy čí1, čí2, čí3. Volaná funkce součet má však v závorkách názvy x, y, z. Přenos vstupních parametrů probíhá tak, že hodnota prvního argumentu uvedená u volacího příkazu je přiřazena prvnímu argumentu volané funkce, druhá druhému, třetí třetímu (...atd.). Po přenosu bude x=čí1, y=čí2, z=čí3. Argumentům volacího příkazu se počítačsky říká formální parametry, které se převlekem ve volané funkci mění na parametry faktické.

Ve volané funkci opět musíme správně deklarovat typ proměnných (zde int x,y,z). Zapamatujte si, že deklarace stejnojmenných argumentů funkce musí být uvedena ještě před první svorkou těla funkce. Ostatní lokální proměnné (zde je to proměnná a) deklarujeme až v těle funkce. Na to při zápisu pozor. Kompilátor případnou chybu obvykle registruje (jen nám vyhubuje).

Jednotlivým proměnným čí1, čí2, čí3 jsou ve funkci main přiřazeny celočíselné hodnoty 11, 22, 33. Volací příkaz součet(čí1,čí2,čí3) volá stejnojmennou funkci, jejímž argumentům uvedené hodnoty předá (překopírují se do nově vytvořených lokálních proměnných). Výsledek aritmetického součtu v těle funkce je přiřazen proměnné a. Její hodnota je volající funkci vrácena příkazem return jako výstupní parametr. Hodnota tohoto parametru je přiřazena proměnné čí4, která je nakonec vytisknuta posledním příkazem funkce main.

Co se však stane, když v programové změti napíšeme argumenty funkce součet třeba takto?:

```
součet(čí3,čí2,čí1)
int čí3,čí2,čí1;
{
int a;
a= čí3+čí2+čí1;
return (a);
}
```

Na první pohled by člověk mohl propadnout dojmu, že dojde k nějakému nevítanému zmatku. Skutečnost je však milosrdná. Vše proběhne naprosto v pořádku. Jak už bylo uvedeno v první části seriálu, volaná funkce nemůže přímo měnit hodnoty proměnných funkce volající. To platí i při užití proměnných se stejným názvem, jaký mají proměnné funkce volající. Volaná funkce si vždy vytvoří své vlastní lokální proměnné, které se po skončení jejího běhu zase "vypaří". To nás do značné míry zbavuje nepříjemné nutnosti sledovat, zda už jsme někde v programu nepoužili název proměnné, kterou se právě chystáme deklarovat (viz Basic). Vedle lokálních proměnných lze používat i globální, u nichž je určitá míra ostražitosti nezbytná (o nich až dále).

Náš sečítací prográmek je hodně rozsochatý. Céčko má řadu fines, které vedou ke krátkému zápisu zdrojového textu. Jako první z nich si uvedeme možnost zařazení volacího příkazu coby argumentu jiného volacího příkazu:

```
main ()
{
int čí1,čí2,čí3;
čí1=11; čí2=22; čí3=33;
printf("Čísla: %d,%d,%d",čí1,čí2,čí3);
printf("\nSoučet: %d",součet(čí1,čí2,čí3));
}

součet(x,y,z)
int x,y,z;
{
int a;
a=x+y+z;
return (a);
}
```

Výsledek bude stejný. Ale v textu ubyla proměnná čí4. Celý příkaz volání funkce součet v posledním příkazu printf bude "nahrazen" hodnotou vráceného parametru a. Malé zkrácení můžeme provést i ve funkci součet:

```
součet(x,y,z)
int x,y,z;
{return(x+y+z);}
```

Argumentem příkazu return může být jakýkoli výraz, který vrací jednu hodnotu. V našem programu se ale klidně obejdeme i bez příkazu return:

```
součet(x,y,z)
int x,y,z;
{x+y+z;}
```

Volající funkci je vždy vrácena poslední vypočtená hodnota. Z toho plyne, že příkaz return má své využití např. tehdy, když se v těle volané funkce rozhoduje o více eventualitách návratu s různými hodnotami. V jednom těle funkce tak může být jeden či více příkazů return, které jsou obvykle aktivovány po splnění určité podmínky. Na konci těla funkce bývá return tehdy, když je jeho argumentem volací příkaz, když funkce končí cyklem, nebo když volané funkci vracíme jinou hodnotu, než je posledně vypočtená.

Přenos parametrů prostřednictvím adresy

Příkaz return umí vracet jen jednu hodnotu, nic víc. To je hodně málo. A vůbec - jaký by to byl jazyk, kdybychom si z jakékoli funkce nemohli sáhnout na libovolné proměnné? Céčko nám to umožňuje zprostředkovaně - přes adresu uložení proměnné.

Výsledkem unární operace:

```
&prom
```

je adresa uložení proměnné prom. & je zde v roli unárního operátoru, proměnná je operandem. Výsledek této operace se anglicky jmenuje pointer. Schématicky:

```
pointerofprom=&prom
```

Pro rychlou orientaci se obvykle používá konvenční zápis jen s písmenem p na začátku (zde by to bylo pprom=&prom).

V češtině se objevily překlady ukazatel a ukazovátko. Ten druhý není zrovna lahodný. Mohli bychom použít i slovo odkaz. V podstatě jde o ukazatele adresy proměnné či odkaz na adresu proměnné. Jak se ve kterém překladači odkazování interně provádí, nám momentálně může být jedno. I když se konvenčně říká, že výsledkem uvedeného přiřazení získáme ukazatel, neuděláme chybu, když prostě řekneme, že tak dostaneme adresu.

Když už máme adresu proměnné, měli bychom mít možnost manipulace s jejím obsahem. To umožňuje další unární operace:

```
*pprom
```

Analogicky assembleru - máme-li v reg.HL adresu proměnné prom, pak HL=&prom=pprom (adresa proměnné, resp. ukazatel adresy). Pak (HL)=*pprom=prom (obsah adresy proměnné, resp. hodnota proměnné samotné).

Proměnná typu int má obvykle rozsah dvou bajtů. To znamená, že manipulace probíhá s obsahem obou adres, na nichž je proměnná int uložena. Proberme si sled těchto příkazů:

```
int prom,obsahprom;
int *adrprom;
adrprom=&prom;
obsahprom=*adrprom;
```

Samotné číslo adrprom (bez hvězdičky) není už třeba deklarovat (mnemonika *adrprom toto číslo zahrnuje). Ve výsledku dává sled příkazů totéž, co přiřazení obsahprom=prom. Ovšem když máme proměnnou prom někde ve volající funkci a nemůžeme ovlivnit její obsah přenosem vracené hodnoty, pak se můžeme odvolat přímo na její adresu a použít či

měnit její obsah. Předvedeme si to na příkladu výměny obsahů dvou proměnných:

```
main()
{
int a,b;
a=10;b=20;
printf("Před výměnou: %d,%d",a,b);
prohoz(&a,&b);
printf("\nPo výměně: %d,%d",a,b);
}

prohoz(čís1,čís2)
int *čís1,*čís2;
{
int pánev;
pánev=*čís1;
*čís1=*čís2;
*čís2=pánev;
}
```

Volací příkaz prohoz(&a,&b) přenese adresy proměnných a, b do argumentů čís1, čís2 funkce prohoz. Dále funkce operuje už s obsahy adres obou proměnných. *čís1 je "hozeno" (přesněji okopírováno) na pánev. Tak můžeme klidně do *čís1 přenést *čís2 a nakonec z pánve původní hodnotu *čís1 do *čís2. Po návratu z této funkce nic neodebíráme, protože se v ní už přímo mění obsahy adres uložení proměnných a, b.

Vstupní parametry se přenášejí prostřednictvím argumentů podobně jako u přenosu předchozího. Výstupní parametry však nevrací příkaz return. Jejich hodnoty se mění přímými změnami obsahů adres jejich uložení.

Assembleristům je nejspíš vše jasné. Basicovým borcům snad pomůže, když si pomyslně představí, že za každou fyzicky existující proměnnou se kryje nejen její momentální hodnota, ale i (na rozdíl od Basicu) stabilní adresa, na níž je tato hodnota uložena. Funkce prohoz hodnoty obou proměnných vyměňuje. Ve vztahu k funkci main se však nemění adresa uložení deklarovaných proměnných a, b. Zkuste si představit, že printf poprvé i podruhé provádí "něco jako" PRINT PEEK adresaA;",";PEEK adresaB. Ve funkci prohoz se na tyto adresy "poukuje". Tak je tam pokaždé něco jiného.

Přenos adresou má velký význam při práci s poli. I když nás teprve čeká značná porce jejich studia, odskočme si do jednoho pole takříkajíc po hlavě. Zkusím do něj převést náš součtový program. Jde sice o porušení plynulosti výkladové linky - ale někdy takový odklon od stereotypu pomůže vrýt si do paměti nějakou neočekávanou neobvyklost:

```
int čí[3]={(11),(22),(33)};
main()
{
printf("Čísla: &d,&d,&d",čí(0),čí(1),čí(2));
printf("\nSoučet: &d",součet(čí));
}

součet(čís)
int *čís;
{
int x,y,z;
x=*čís; y=*(čís+1); z=*(čís+2);
x+y+z;
}
```

Překvapující je hned první řádka - svým umístěním i svou syntaxí. Touto globální (mimo tělo funkce) definicí je deklarováno jednorozměrné pole s názvem čí a určen obsah jeho tří prvků (proto pojem definice). Globální znamená, že je platná pro všechny další funkce a ty se na ni mohou odvolávat. Pro pole obecně platí, že lokálně je můžeme pouze deklarovat. Globálně deklarovat i definovat. Trošku zmatku do polí vnáší číslování prvků od nuly, když v deklaraci neprázdného pole určujeme jejich počet od jedné. Tak třeba třetí prvek zde má symboliku čí(2) - viz argumenty v prvním příkazu printf. Kompilátor Hisoft C, s nímž experimentuji, vyžaduje, aby každý definovaný prvek pole byl umístěn v kulatých závorkách. Některé kompilátory se obejdou bez nich. Zapamatujte si použití hranatých závorek a svorek v definici pole.

Pole má tu zvláštnost, že už samotným jeho jménem je předávána adresa uložení prvního prvku pole (tedy prvku(0)). Volací příkaz součet(čí) tak předává argumentu volané funkce hodnotu adresy uložení prvku čí(0). Deklarace int *čís uvádí, že budeme s proměnnou (resp. obsahem adresy, na níž je uložena) pracovat jako s celým číslem. To se ale týká i adresy samotné - deklarace int *čís se vztahuje jak k předané adrese čís, tak i k jejímu obsahu *čís.

Zatímco jméno pole čí ve funkci main je konstanta, po přenosu do volané funkce se uvnitř této funkce mění na proměnnou. Podle toho je s ní pak možno nakládat. Jakékoli pokusy o podobné manipulace s konstantou by kompilátor nepřijal (resp. neměl by přijmout).

Proměnné x je přiřazen obsah adresy s prvkem čís(0) (u funkce volající je to čí(0)). Výraz *(čís+1) posouvá adresu na další prvek. Když je proměnná typu int dvoubajtová, pak se adresa zvýší o dvě fyzické adresy paměti (přičítané číslo tu hraje roli ofsetu vůči první adrese pole). Samotný výraz *čís je de facto *(čís+0). Touto ofsetovou operací se v paměti můžeme relativně libovolně pohybovat nahoru, a záporným ofsetem i dolů. Pokud bychom chtěli brouzdat pamětí "polním ofsetem" podobně jako v assembleru, pak musíme vědět, kolik fyzických adres paměti posun na sousední prvek zahrnuje (bývá to různé).

Proměnné y je přiřazen obsah adresy s druhým prvkem pole a proměnné z se třetím prvkem. Následuje už jen součet. Jeho výsledek je jako výstupní parametr předán funkci main, která jej vytiskne.

I tady můžeme odbouráním proměnných x, y, z zkrátit tělo funkce součet:

```
součet(čís)
int *čís;
{*čís+*(čís+1)+*(čís+2);}
```

Závěrem jedna začátečnická poznámka. Když jsem louskal první desítky stran literatury o Céčku, říkal jsem si - aha, jojo, tohle je jasný, samozřejmě... A liboval jsem si, jak hezky mi to jde. Pak jsem počítač napustil kompilátorem, a tu náhle - tma přede mnou, tma za mnou i ve mně. Zámořský poeta by řekl darkness inside out. Říkám to proto, že bez praktického vstřebání céčkových návyků se nikam nedostanete. Jedna věc je o jazyku číst, zcela jiná usednout před prázdný editor a začít do něj jen tak z hlavy něco ťukat. Zpočátku mě nejvíc trápila ta úplně nejzákladnější syntaxe - čárky, středníky, závorky. To brzy jakž takž pominulo. V další fázi to bylo špatné umístění i provedení deklarací a nerovný zápas se zápisem argumentů funkcí. Než mi začalo fungovat vše z této části seriálu, kompilátor mi aspoň třicetkrát vynadal.

Rovněž by nemělo valný smysl všechno jen opisovat. Vymýšlejte si vlastní, byť sebejednodušší problémy a postupně je (přese všechny chyby a omyly) převádějte do "mluvy" jazyka přímo na počítači.

To byla jen taková malá stimulace těch, kteří ještě váhají. Protože my ostatní Céčko opravdu dřeme. Nebude trvat dlouho a v Mikrobázi se objeví první původní céčkové programky. Kdo nepřestane váhat, neporozumí jim. Nebude to škoda? Ještě je čas!

-elzet-

FILOSOFICKÉ ASPEKTY STROJOVÉHO MYŠLENÍ (5)

Formální systémy (syntaxe)

Metoda moderní logiky je charakterizována důsledným odlišováním formální (syntaktické) stránky určitého výrazového a usuzovacího mechanismu od jeho stránky obsahové (sémantické). Cílem je zkoumat, jak se fungování prvého odráží ve druhém. Bezděky jsme použili slova 'mechanismus'. A skutečně, formální operace mají v sobě vždy cosi mechanického, strojového. Tato skutečnost se nabízí jako vodítko při úvahách o tom, co stroje vlastně dělají, když si myslíme, že myslí. Budeme totiž pohlížet na formální systémy logiky jako na metaforu pro určité aspekty myslících strojů. Vzhledem k tomu, co všechno už logika o sobě ví, by to mohla být metafora dosti produktivní.

Na počátku každého formálního systému je jazyk. Nikoli jazyk přirozený, nýbrž formální - čistě konfigurace symbolů, které buď nic neoznačují, anebo označují přesně to, co jim my určíme (při konkrétní interpretaci).

Představme si jako příklad abecedu o třech písmenech: D, I, A. Libovolné řetězy těchto písmen - zvané formule - budou tvořit jazyk formálního systému (nazveme jej DIA systém), který použijeme pro ilustraci metody logiky.

Náš jednoduchý jazyk nyní oživíme čtyřmi inferenčními pravidly:

I. Končí-li řetěz symbolem I, přidej na jeho konec A.
II. Má-li řetěz tvar Dx, vyrob z něj Dxx (kde x je libovolný řetěz).
III. Vyskytuje-li se v řetězu trojice III, nahraď ji písmenem A.
IV. Vyskytne-li se v řetězu dvojice AA, vypusť ji.

Inferenční pravidla píšeme ve zjednodušené podobě takto:

$$(I)\ \frac{xI}{xIA},\quad (II)\ \frac{Dx}{Dxx},\quad (III)\ \frac{xIIIy}{xAy},\quad (IV)\ \frac{xAAy}{xy}.$$

Řekněme, že nám spadl z nebe nějaký řetěz, třeba DI. Bude to náš (jediný) axióm. A už nám nic nebrání v generování množiny všech řetězů, které vzniknou z DI postupnou aplikací pravidel (I)-(IV), opakovaně a v libovolném pořadí. Tak např.:

DI ----> DII ----> DIII ----> DIIIIA ---->
(axióm) (II) (II) (I) (III)

DAIA ----> DAIAAIA ----> DAIIA
(II) (IV)

Takto získaný řetěz (formule) DAIIA je jedním z (nekonečně mnoha) teorémů našeho DIA-systému a konečná posloupnost (DI, DII, DIIII, DIIIIA, DAIA, DAIAAIA, DAIIA) jsou jeho důkazem (čili teorémy jsou dokazatelné formule). To, že termíny 'teorém' a 'důkaz' mají pro nás nějaký intuitivní význam, je zde zatím zcela nepodstatné.

Úloha: Je DA teorémem?

Náš DIA-systém je příkladem velmi primitivního a čistě formálního systému. (Primitivního? Je DA teorémem nebo ne?) Nicméně už teď můžeme udělat několik zajímavých pozorování:

1. Teorémy lze dokazovat, aniž bychom věděli, co vyjadřují.
2. Není těžké rozhodnout, zda daná posloupnost formulí je důkazem či nikoli (stačí vyzkoušet inferenční pravidla).
3. Obecně je však daleko těžší rozhodnout, zda daná formule je teorémem či nikoli (tzv. problém rozhodnutelnosti systému).
4. Vždy existuje metoda (až nudně jednoduchá- tedy jako dělaná pro stroj), generující postupně všechny teorémy (stačí generovat strom všech důkazů).
5. Metodu z bodu 4 lze použít jako spolehlivý, ale zdlouhavý způsob nalezení důkazu libovolného teorému (dříve či později na něj musíme narazit).
6. To, že některá formule není teorémem, se však nemusíme dovědět nikdy.
7. Jsou situace, kdy se člověk chová jinak než stroj.

Tato poslední poznámka zaslouží vysvětlení. Představme si, že je třeba rozhodnout, zda A je teorémem DIA-systému. Stroj by postupně zkoušel všechny možnosti (dle bodu 4) - a nikdy by neskončil. Kdežto člověk si třeba všimne, že řetěz A nelze dokázat prostě proto, že ani jedno pravidlo neodstraní počáteční D z žádného řetězce generovaného z axiómu DI. Člověk má prostě schopnost si všimnout něčeho, co není součástí pravidel hry- je to právě ten odskok ze systému, o kterém jsme mluvili dříve. Hofstadter zdůrazňuje /s.37/, že

tato úvaha neříká nic víc, než že je možné, aby stroj byl totálně nevšímavý, kdežto člověk (ač nemusí být vždy všímavý) nemůže být totálně nevšímavý.

Možnost člověka vždy odskočit ze systému bude pro nás důležitým tématem diskuse, zatím však poznámka 7. nepřispívá k problému člověk versus stroj o mnoho víc než tvrzení, že jsou situace, kdy se člověk chová jinak než visací zámek.

Formální systémy (sémantika)

Definujme trochu jiný formální systém nad jazykem, který se bude od DIA-jazyka lišit pouze tím, že za formule budeme považovat pouze řetězce typu xAyDz, kde x, y, z jsou řetězce ze samých I. Inferenční pravidlo bude jen jedno:

$$\frac{zAyDz}{xAyIDzI},$$

zato budeme mít celé schéma axiómů (tj. nekonečnou množinu zadanou jedním vzorem):

xAIDxI.

Tentokrát však našim symbolům udělíme konkrétní význam:

A označuje: a (plus)
D označuje: dá (rovná se)
I označuje: jedna
II označuje: dvě
III označuje: tři
atd.

(takže IIAIIIDIIIII říká "dvě a tři dá pět"), přičemž českým slovům ponecháváme jejich přirozený význam (2+3=5).

Přiřazení významů jednotlivým prvkům jazyka říkáme interpretace (zde jsme interpretaci provedli zprostředkovaně, přes přirozený jazyk, což je nejobvyklejší způsob). Co se však stalo? Nejen prvky jazyka, ale celé formule mají pojednou význam, "rozumíme jim", a navíc, o některých formulích (např. IIAIIIDIIIII) můžeme říci, že jsou při této interpretaci pravdivé, o jiných, že nejsou pravdivé (např. IAIDI).

Interpretace je tak trochu naše záležitost. Tentýž systém lze interpretovat mnoha způsoby. Představme si, že se rozhodneme, že:

A označuje: a (spojka)
D označuje: dělí (je dělitelem)

(ostatní jako dříve), takže IIAIIIDIIIII říká "dvě a tři dělí pět" (rozuměj: "pět je dělitelné dvěma a třemi"). Zcela jiné formule jsou nyní pravdivé (např. IAIDI) a jiné nepravdivé (např. IIAIIIDIIIII). Mluvíme-li o formálním systému, je vždy nutno mít zcela jasno v tom, zda a jak je interpretován. Obvykle je jedna interpretace považována za standardní (např. + je plus, čísla jsou čísla apod.), pak ji není třeba vždy znovu připomínat.

Každá formule tedy může, ale nemusí být dokazatelná a (při dané interpretaci) může, ale nemusí být pravdivá. Rozdíl mezi těmito dvěma vlastnostmi je hluboký: zatímco dokazatelnost je čistě formální, syntaktickou vlastností, definovanou mechanickými operacemi a formulemi, je pravdivost sémantickou vlastností, závislou na "skutečném" stavu věcí, o kterých interpretovaná formule vypovídá. Vztah pravdivosti a dokazatelnosti je hlavním předmětem matematické logiky.

V případě našeho DIA-systému s interpretací "a+b=c" dochází k zajímavé shodě: jak si snadno ověříme, každá dokazatelná formule je pravdivá a každá pravdivá formule je dokazatelná. V takovém případě říkáme, že formální systém je (při dané interpretaci) úplný. Brzy se setkáme s případem matematicky přirozeně interpretovaného systému, který není úplný.

Ještě něco platí pro DIA-systém z tohoto odstavce: existuje algoritmus, který o každé formuli rozhodne, zda je teorémem či nikoli. Tento systém je tedy rozhodnutelný (vlastnost, která nezávisí na interpretaci). Jsou však též systémy, které nejsou rozhodnutelné, tj. ve kterých existuje "pozadí" sestávající z formulí, o kterých nelze říci v konečném stavu, zda jsou či nejsou teorémy.

Peanova aritmetika

Předchozí systém generoval pravdivé výroky typu "a+b=c" o přirozených číslech. Zdalipak existuje systém, který by byl schopen generovat všechny pravdy o přirozených číslech - např. "Existuje nekonečně mnoho prvočísel"?

Teď už musíme opustit primitivní DIA-logiku a použít bohatšího aparátu, jak jej známe ze školy. Bude to predikátový kalkul prvního řádu (s rovností), s jazykem rozšířeným o dva binární funkční symboly (+ .) a o dvě konstanty (0 1). Inferenční pravidla predikátové logiky se nemění, k logickým axiómům přibydou "aritmetické" axiómy:

$$x + 1 = y + 1 \supset x = y$$
$$\neg (0 = x + 1)$$
$$x + 0 = x$$
$$x + (y + 1) = (x + y) + 1$$
$$x \cdot 0 = 0$$
$$x \cdot (y + 1) = (x \cdot y) + x$$
$$0 + 1 = 1$$

a schéma axiómů indukce:

$$P(0) \,\&\, \forall x(P(x) \supset P(x + 1)) \supset \forall y\, P(y)$$

kde P je libovolná vlastnost, kterou lze vyjádřit naším jazykem. Takto vytvořený formální systém se nazývá Peanova aritmetika, zkráceně PA.

Interpretaci snad nemusíme popisovat (jen poznamenejme, že např. termín (((1 + 1) + 1) + 1) + 1 je interpretován jako číslo 5). Tak např. formule $\forall x\,[\neg(x = 1 + 1) \supset \exists y\,(x = y \cdot y)]$ vyjadřuje (nepravdivé) tvrzení, že každé přirozené číslo různé od 2 je čtvercem.

Peanova aritmetika byla vytvořena tak, aby:

(1) všechny axiómy byly pravdivé,
(2) inferenční pravidla z pravdivých premis odvozovala pouze pravdivé závěry.

Tak se zaručí, aby všechny teorémy PA byly pravdivé (ve standardní interpretaci) - z čehož m.j. plyne, že PA je konzistentní. Připomeňme si, že konzistentní (bezesporný) formální systém je takový, v němž nelze dokázat nějakou formuli i její negaci (pak by šlo dokázat naprosto vše).

To, co bylo právě řečeno o korektnosti inferenčních pravidel a pravdivosti axiómů PA, vlastně předpokládá daleko více, než by se na první pohled zdálo. Inference, např. použitím pravidla Modus Ponens (z A a $A \supset B$ odvoď B) nepředpokládá, že dvě hypotézy (a a $A \supset B$) jsou pravdivé. "Proč mám věřit, že

$$A \,\&\, (A \supset B) \supset B$$

už je pravdivé?", ptá se Želva (trochu jinými slovy) v dialogu Lewise Carrola (/1/, s. 43) a žádá Achilla, aby to považoval za třetí hypotézu pro Modus Ponens. Totéž však platí pro toto nové "Modus Ponens" a pak zas další atd. Nekonečný regres, který může dohnat Achilla k šílenství (že nemůže dohnat Želvu) poukazuje na skutečnost, že v základech logiky je vždy něco, co musíme bez diskuse přijmout za evidentní. (Přijme to za evidentní také stroj?)

Gödelova věta

"Nejsem teorém!" prozrazuje sama na sebe formule Peanovy aritmetiky (v pohádkách mohou mluvit i formule). Pak ale, ať chce či nechce, musí mít pravdu: kdyby ji totiž neměla, byla by teorémem. Ale nepravdivý teorém? Absurdní!

To je esence Gödelova triku. Nejvíc práce však dá předvést, že nejen v pohádkách, ale i v Peanově aritmetice mohou formule mluvit, dokonce o sobě samých.

O čem normálně mluví formule PA? Samozřejmě o číslech. Zařídíme to tedy tak, aby formule měly svá identifikační čísla. Pak budou moci o sobě mluvit jejich prostřednictvím. Postup je tento:

Nejdříve se jednoznačným způsobem přiřadí čísla všem symbolům jazyka PA (i proměnným).

Pak se vytvoří číselná funkce, která, je-li použita na posloupnost čísel reprezentujících jednotlivé symboly nějaké formule A (obecně libovolného řetězu symbolů jazyka PA), dá nové číslo a, z něhož lze kdykoli získat zpátky zase formuli A.

Číslo a se nazývá Gödelovým číslem formule A a popsaný postup je gödelizace jazyka PA. Není obtížné ještě jednou postup zopakovat a obdobně číselně reprezentovat i posloupnosti formulí, např. důkazy.

Máme-li takto očíslovány formule a posloupnosti formulí, lze vyjadřovat některé jejich vlastnosti pomocí formulí PA, jako by to byly vlastnosti čísel (podobně jako rodná čísla vyjadřují některé vlastnosti občanů). Budeme potřebovat dvě takové vlastnosti; v obou případech je lze přesně definovat formulemi PA, my to však zde dělat nebudeme. Použijeme pro ně zkráceného zápisu $\underline{Pr}(x,y)$ a $\underline{Apl}(x,y,z)$. Slovy:

$\underline{Pr}(x,y)$ je formule PA, která vyjadřuje binární predikát "x je číslo důkazu (v PA) formule s číslem y". Budiž dále $\underline{Tm}(y)$ zkrácený zápis formule $\exists x\ \underline{Pr}(x,y)$, čili $\underline{Tm}(y)$ vyjadřuje unární predikát "y je číslo formule, která je teorémem PA".

$\underline{Apl}(x,y,z)$ je formule PA, která vyjadřuje ternární predikát "y je číslo formule o jedné volné proměnné a x je číslo formule vzniklé aplikací formule s číslem y na číslo z (čili dosazením za volnou proměnnou do y)". Je-li např. a číslo formule $\exists u(u + v = v)$ a b číslo formule $\exists u(u + 1 = 1)$, platí $\underline{Apl}(b,a,1)$.

Uzavřeným formulím (tj. formulím bez volných proměnných) budeme též říkat sencence. Všimněme si, že každá sentence PA může mít dvojí intuitivní význam: jako tvrzení o přirozených číslech (např. "7 je prvočíslo") a jako tvrzení o formulích PA (např. "formule s číslem 7 je uzavřená").

Dostáváme se k rozhodující konstrukci. Vytvoříme formuli, které vzhledem k její důležitosti dáme nějaké jména, třeba "Teta":

$$\neg\exists x[\underline{Tm}(x) \,\&\, \underline{Apl}\,(x,y,y)]$$

Teta říká: "Neexistuje formule, která je teorémem a přitom vznikla aplikací dané formule (s číslem y) na její vlastní číslo (y)." Nebo stručněji: "Daná formule aplikovaná na své vlastní číslo není teorém."

Nechť t je Gödelovo číslo Tety (je to jedno konkrétní číslo; t je jen náš symbol pro ně). Víme, že Teta má jednu volnou proměnnou (y) a můžeme ji tedy aplikovat na zvolené číslo, speciálně na její vlastní číslo t:

$$\neg\exists x[\underline{Tm}(x) \,\&\, \underline{Apl}(x,t,t,)]$$

Toto je už sentence; nazveme ji Gödelovou sentencí G. Říká:

"Teta aplikovaná na své vlastní číslo není teorém."

Avšak Teta aplikovaná na své vlastní číslo je přece G! Tedy G vlastně říká:

"G není teorém."

čili, chcete-li:

"Nejsem teorém!"

Co z existence Gödelovy sentence G plyne? To, co sama říká, ještě nemusíme brát vážně. Zopakujme tedy úvahu z počátku tohoto odstavce. Předpokládejme, že ve skutečnosti je G teorémem PA. Pak, jak víme, musí být pravdivá, tj. musí platit co říká: že není teorémem - a to je spor s předpokladem. Platí tedy:

Gödelova věta:

> Existuje pravdivá sentence o přirozených číslech, která není teorémem PA. PA je tudíž neúplný formální systém.

Není-li G dokazatelná, mohla by třeba být dokazatelná její negace $\neg G$. G je však pravdivá, takže $\neg G$ musí být nepravdivá (pravdivost je komplementární). Ježto teorémy PA jsou pravdivé, nemůže být ani $\neg G$ teorémem (což se dalo čekat). Dokazatelnost není komplementární.

Spor, který nás přivedl ke Gödelově větě, byl sémantický: týkal se pravdivosti. Zkusme provést úvahu poněkud jinak.

Znovu předpokládejme, že G je teorémem PA. Pak existuje důkaz G v PA - nechť d je jeho číslo (najít je lze např. generováním všech důkazů). Platí tedy $\underline{Pr}(d,g)$, kde g je číslo G. Rozborem definice $\underline{Pr}$ (kterou jsme neuvedli, takže rozbor též neuvedeme) lze ukázat, že potom $\underline{Pr}(d,g)$ je teorémem. Z definice G je zřejmé, že $\underline{Apl}(g,t,t)$ je rovněž teorémem. Tedy

$$\exists x\,[\underline{Tm}(x) \,\&\, \underline{Apl}(x,t,t)]$$

je teorém, přičemž je negací G, a o G jsme předpokládali, že je teorémem. To by bylo možné jen kdyby PA nebyla konzistentní. Tentokrát jsme použili spor "uvnitř" PA.

Použijme to k obecnější formulaci Gödelovy věty. Předem si však povšimněme, že celou konstrukci i důkaz lze provést i v jiném konzistentním formálním systému, pokud obsahuje PA (čili pokud je rozšířením PA o další axiómy, popř. s širším jazykem). PA slouží pouze k číslování a k definici predikátů $\underline{Pr}$ a $\underline{Apl}$.

Gödelova věta (obecnější tvar):

> Každý konzistentní systém obsahující PA je neúplný.

Ještě dvě poznámky. Před chvílí jsme slovně dokázali, že Gödelova sentence G je pravdivá. Nelze tento náš "důkaz" zformalizovat též? Ukazuje se, že lze, ale nikoli v rámci PA. Odvolali jsme se v něm na vlastnosti PA, že její teorémy jsou pravdivé (popř. na konzistenci PA). Kdyby tyto vlastnosti byly dokazatelné v PA, byla by G teorémem - se všemi absurdními důsledky. K důkazu pravdivosti G jsme museli odskočit do metasystému a podívat se na PA zvenku.

Další možnost: Nic nebrání, abychom nepřidali G (anebo sentenci ekvivalentní, např. výrok o konzistenci PA) k PA jako nový axióm. Co se stane pak? Dostaneme nový systém, PA + G, který je konzistentním rozšířením PA. Platí v něm tedy obecnější tvar Gödelovy věty - vše se opakuje, ale pro jinou godelizaci, pro jinou dokazatelnost a pro jinou Gödelovu sentenci, řekněme

G_{PA+G}

A znova totéž:

$$(PA+G)+G_{PA+G},\ ((PA+G)+G_{PA+G})+G_{(PA+G)+G_{PA+G}}$$

atd. Nekonečná hierarchie stále chytřejších systémů.

(pokračování)

Ivan M.Havel, Petr Hájek

POČÍTAČOVÁ SCIENCE FICTION

(DOKONČENÍ)

(-19-) A.C.Clarke: Křížová výprava. Osoby: inteligence v mělkých oceánech supravodivého Helia. Místo: daleko ode všech sluncí a tepla.

Inteligence se rozvíjí za nepředstavitelně výhodných podmínek: informace kolují obrovskými rychlostmi prakticky bez energetických ztrát /I67/ ...až stvoří sondy, které vracejí divnou zvěst- že existují nepředstavitelně kuriózní formy inteligencí dokonce na tak horkých planetách, že voda je tam v kapalném stavu! Jestli je tomu tak, jde o jev tak zvrhlý, že musí být vymýcen. A tak se již miliardu let ze souhvězdí Orla k Zemi blíží koncentrovaná energie, vesmírný lapis, skalpel (doba příjezdu: 2005) /E69/.

I67 * Materiální základna informačních soustav může být extrémně malá * Takže si lze představit prakticky "čisté" zpracování informací. Hádankou je, o čem. Člověk se naučil myslet, aby nemusel vynakládat tolik energie. Slepý hluchoněmý od narození postižený sníženou citlivostí hmatu- narodil se a žil někdy někdo takový? Zaregistrovala biologie podobné mládě lidské nebo zvířecí? Z jakých rigorózních důvodů bychom naopak mohli vyloučit existence systémů postrádajících smyslové vnímání našeho typu a nahrazujících jej vlastní sebereflexí, resp. přímou informační reflexí jednotlivých složek? Zajímavé jsou i další důsledky:

E68 * Pojem individuality je relativní * Lidstvo sestává z jedinců-individuí proto, že mezi jeho jednotlivými členy není přímá informační vazba typu procesor-procesor; je to existence periférií - energeticky podstatně náročnějších a informačně podstatně neschopnějších - která vytváří přirozený rozklad na individua. Připusťte telepatii či aspoň jiné energeticky nenáročné dorozumívání (mají ho mravenci či včely?) a začne mizet rozdíl mezi jedincem a kolektivem (šlápnou na nohu Frantovi a bude to bolet i tebe).

E69 * Pojem "cizí" je vždy spojen s pojmem "nežádoucí" a "nebezpečný" * Rozdíly mezi inteligencemi mohou vést ke konfliktu z naprosto jiných než lidsky motivovaných zájmů, třeba mocenských. Byli bychom ochotni připustit rozhovor s viry (třeba neškodnými), kdyby se mezi nimi nějakou mutací náhodně vyvinula inteligence souměřitelná s naší a my jsme byli schopni to rozpoznat?

A70 * Teoreticky možných implementací informačních soustav je nekonečně mnoho * Od A.C.Clarka pocházejí i bytosti - nesmírně řídké sloučení mlhoviny (Ze Slunce zrozeni); vesmírem kolující "čisté" inteligence, tu a tam se inkarnující, např. do lumíků (Posedlí); nebo tajemné bytosti Jupitera, projevující se jako mraky z uhlovodíkových pěn - meduzy (Setkání s medusou). V této povídce mimochodem vystupuje i kyborg, vyrobený z havarovaného letce Howarda Falcona: "nesmrtelný kompromis mezi člověkem, tak nestálým, že možná vůbec nemá oprávnění překročit hranice atmosféry, a strojem"; a věta, popisující "pouhého" Jupitera slovy: "Byl to svět, v němž se mohlo přihodit absolutně všechno a nikdo z lidí neuměl odhadnout, co přinese příští okamžik."

E71 * Budoucnost přinese jiné struktury počítačových systémů i pokud jde o vztah procesorů a periférií * Procesory budou nuceny komunikovat na bázi jiných logik (ne-deterministické, ne-dvouhodnotové, ne-typové, ne-logické, nebo naopak vše super-) a periférie budou nuceny akceptovat informační vstupy a výstupy přes jiné než dosud běžné lidské smysly. Kancelářské stroje r. 2500: laserová telepatická holografická tiskárna, rychlost 150 pojmů zapisovaných přímo do paměti X-anů za sekundů; pořizovač dat neutrino-to-disc; selektorový kanál pro přenos vůní spojených s asociativním vyhledáváním vzpomínek z dětství, atd.

E72 * Abstraktní struktura nebe-peklo-ráj a jiné mytologické a religiózní pojmy připouštějí zcela materialistickou implementaci * A proti některým úvahám scifi jsou až banálně jednoduché. Většinou jsme již tento fakt zaregistrovali. Např. nebe a ráj lze implementovat na úrovni naivní /I43/ i psychologické /I60, 61, I4/. Analogicky inverzně i peklo. V takovém systému lze pak akceptovat: modlitbu (jako neproceduralně orientovaný program ve vhodném specifickém jazyce denotační sémantiky), papeže (jako vedoucího programátora Stroje-boha; jiná interpretace viz I38), vzkříšení (jako limitní případ rekonstrukce objektu ze zbytkové informace; vedle možnosti uvedené již v I5 stačí uvážit, že dnes víme třeba o Tutanchámonovi víc než před sto lety - jak postupujeme do budoucnosti, odkrývá se i minulost, a počítače již rozluštily několik zapomenutých jazyků a nápisů). V jádru problému je ovšem otázka všemohoucnosti a vševědoucnosti boha; odpověď na možnost jeho implementace dává následující Odpověď.

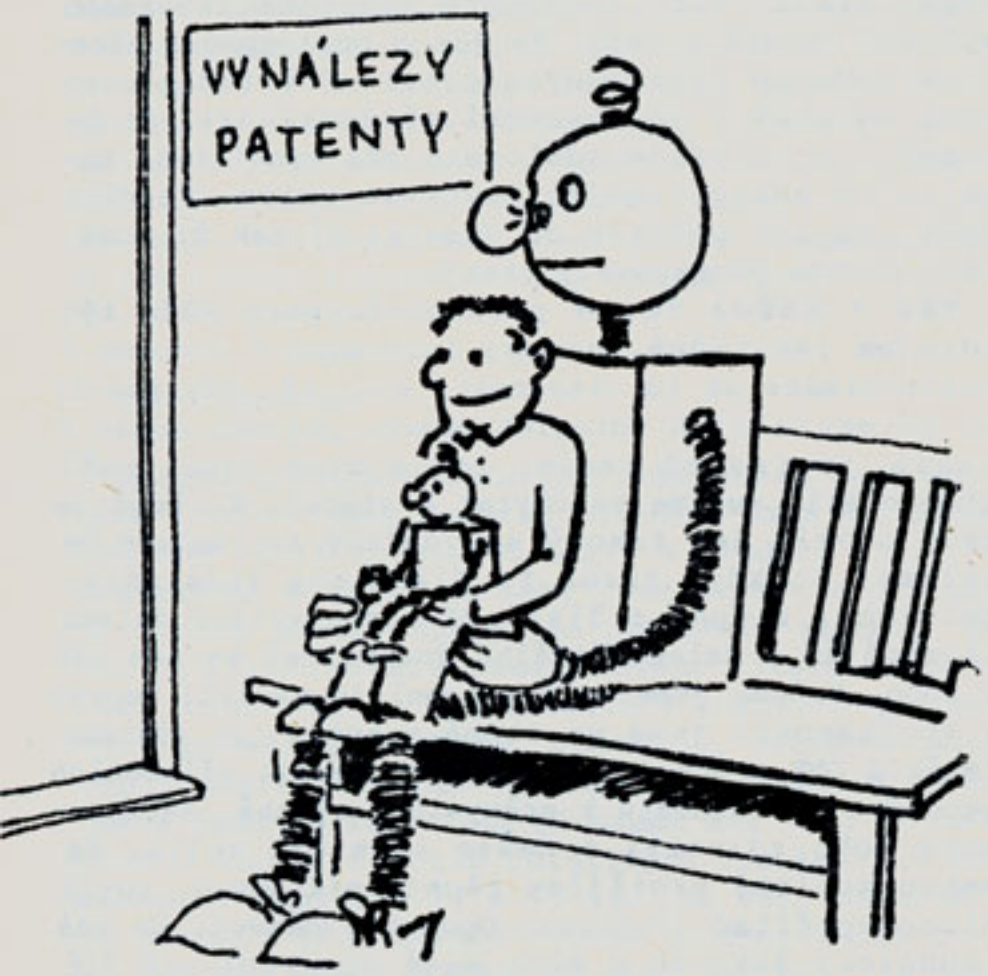
(-20-) F.Brown: The Answer (Odpověď). Osoby: Ev, Reyn (řídí závěr experimentu); později: bůh. Místo: vesmír zabydlený lidmi.

Povídka je krátká; na půl stránce nás přenáší do okamžiku, kdy před očima 96 biliónů obydlených

planet dochází k historické chvíli: Ev připravil propojení všech počítačů všech planet a vyzývá Reyna, aby nově vzniklému superpočítači položil první otázku, hodnou té chvíle. Reyn vybírá tu, na niž zatím nikdo nedovedl odpovědět, klasickou otázku lidstva mnohých epoch: Existuje Bůh? Okamžitě přišla odpověď pronesená mocným hlasem: Ano, teď už Bůh existuje /I73/. Na náhlý Evův popud rychle vypnout ustavená spojení - sjel z jasného nebe blesk, srazil Eva k zemi a zastavil klíč v poloze ZAPNUTO. /I74/.

I73 * Informace je síla * Často větší než hrubá materiální síla. na dolním konci škály to ví paní Nováková, která se právě něco dozvěděla na paní Vopršálkovou. Někde uprostřed jsou generální štáby rozvědek a kontrarozvědek. A na horním konci ...?!

I74 * Vševědoucnost a všemocnost není nutně jen náboženský pojem * Alespoň ta relativní. Ale je natolik význačná, že z hlediska lidského jedince si plně zasluhuje tento název. Je zde ale jeden technický detail - jak propojit ideu s materiální silou bez prostřednictví biologických bytostí, v níž se oba fenomény od počátku prolínají? Tj. jak zkonstruovat všemohoucnost na základě předpokládané vševědoucnosti. (Mimochodem: co byste asi dělali vy, být vševědoucí?) Ovšem i tomu už jsou položeny základy: počítače řídí nejen prostřednictvím výkazů ASŘ; již dnes nízkoenergetická informace ovládá některé vysokoenergetické zdroje.



Mezi počítačem vyhodnocujícím radarová hlášení a pověstným tlačítkem mezikontinentálních střel zatím doufejme ještě stojí člověk, u něhož je přece jen jistá naděje, že kvůli hejnu hus nezničí svět (včetně Říma, který kdysi zachránily). Nebylo by dobré ho tam pro jistotu navždy nechat?

A75. Podobná je idea Frankensteina (A.C.Clarke: ...A ozve se Frankenstein) - totéž jako (-20-), jen v pozemském měřítku a o něco dřív - a místo deklarace o vlastní všemohoucnosti různé průvodní zjevy ve všech elektrických systémech světa.

(-21-) A.C.Clarke: Záznam. Osoby: Oni - naprosto neznámí s naprosto nepostižitelnou technikou; Já: W.V.Neuberg, pilot mistrovské třídy; oprava: V.W.Freeburg, pilot první třídy; oprava: V.Willburg, pilot druhé třídy ... Místo: (symbol nedefinovatelnosti).

Myslel jsem, že své tělo znám ... Ruce, nohy, kde jste ... Neposlouchají ... Nejsou! Aha, vstoupili jste do mé mysli ... zbavili jste mě bolesti ... Co se stalo? Zachránili jste mě, děkuji. Umíte víc než u nás na zemi. Ale dopravte mě tam, spraví mě. Co ze mě zbylo? Tělo mi přidělají, je-li mozek v pořádku. Ani to ne? Shořel jsem i s lodí? Co ale jsem potom? /I76/. Aha - záznam v jakémsi fantastickém zařízení ... kdo jsem? W.V. - ne V.W. - ne ... Vytrácím se. Chcete vědět jak jsem vypadal? Hlava oválná - ne tak ne, ústa horizontálně- Tahkle že jsem řekl, že vypadám? Vymažte to, musím začít znova ... Ale vytrácí se to ... všechno ... nic nevím ... děkuji vám, že jste se vůbec pokusili ... je pozdě ... mami, kde jsi?

I76 * Inteligence je nezávislá na implementaci * Jde o parafrázi I67, ale se zeslabující podmínkou: zde máme revokaci inteligence, která již existovala. Oč je snadnější případně inteligenci zkopírovat než vytvořit? Pro současný software podstatně. Pokud jde o lidi jako biologický druh, rozhodně ne: denně se rodí milióny, ale dobrá kopie se ještě nepovedla. Navrhujeme metodu introspekce: co byste si přáli ze sebe zachovat v případě, že si můžete nechat nahrát myšlenky, pocity, představy, sny? Které složky své osobnosti byste nabídli na export a které pokud možno ochránili heslem, jež byste se snažili utajit (i před sebou)? Jakými predikáty byste sami sebe definovali jako abstraktní datový typ? (Chcete-li, můžete si ovšem přát i některé implementační detaily, ale ty jsou řekněme mimo možnosti specifikace.)

A77 * Problémem identity je z mnoha hledisek neřešený /neřešitelný?/ * Úvahy z (-21-), I47, A39, E68 aj. doplníme realističtější metaforou: D.C.Dennet z Hofstadterovy knihy The Mind's I se podrobil separace mozku (umístěn do fyziologického prostředí v jakési laboratoři v Houstonu) od těla (jež má splnit jistý úkol v místě, v němž jsou narušovány buňky mozku, a jen ty - to byl důvod k celé operaci). Z celé řady paradoxů navozených touto situací uveďme jeden dost nečekaný z oblasti práva: pokud by tělo (řízené na dálku z Houstonu) vyloupilo někde v Los Angeles banku a bylo dopadeno: podle zákonů kterého státu by byl trestný čin posuzován - státu Texas nebo státu California?

E78 * Otázky související s problémy identity vědomí, a tím m.j. i otázky typu "Mohou stroje myslet resp. mít vědomí?", jsou pseudootázky * Např. ve smyslu Carnapa. Ptají se na něco, co nejsou schopny definovat. Tato charakterizace jim může být přisouzena nejen ze striktně pozitivistického hlediska.

(-22-) J.Christopher: Museum Piece (Muzeální kus). Osoby: npor. Don Parker - z Perikla, lodi hledající vesmírné civilizace; Nuker - jeden z jedné takové civilizace (podobné lidem), kompars obou typů. Místo: strašně vzdálená planeta 384.

Výzkumný tým Perikla (výkvět lidstva, metalizované černostříbrné kombinézy, inteligentní řeči jak investovat vydělané peníze po návratu na Zemi) se téhle civilizaci diví. Zemědělská malovýroba, prosté příbytky a oděvy, ale planeta má společnou kulturu a řeč a její obyvatelé jsou naprosto lhostejní k vesmírným hostům: přátelský, zdvořilý nezájem. A zde vstupuje na scénu Don, senzitivnější než ostatní. Podaří se mu navázat srdečnější spojení se svým průvodcem Nukerem a - veden instinktem - nechal se ještě jednou zavést do velké budovy Muzea, do níž proudí davy domorodců, kteří vycházejí jako mentálně pozměněni. Při prohlídce složitého labyrintu chodeb a sálů Don postřehne zacyklení - zjistí, že je mu úmyslně zatajováno poslední patro. Na přímou žádost se však do něj dostane. (Nuker: "Muselo se to stát".) Zde chudoba planety mizí v záplavě barev, uprostřed sálu je Stroj. Krystalické vibrace, záplava jasu. A pak je Don zcela jinde, nejen geograficky. Vše je jiné, sám se mění - přetváří se v bytost utkanou z lehkosti a síly /I79/. Nuker po návratu: Jsme velmi stará civilizace, také jsme objevovali nové

světy a rozváželi vědění - možná i to vaše pochází od nás. Ale spokojenost chyběla. Pak byl objeven Stroj. Nemohu Ti nic vysvětlit, ani předat- nemohli byste ho využít, je příliš jemný, a i Tvůj krátký pobyt narušil jeho chod. Proto ho skrýváme. Teď jsme Ti vydáni na milost a nemilost - jako každá vyšší forma nižší, jako život neživé přírodě /I80/. K obhajobě lidstva dlužno říci, že Don našel sílu vrátit se a zatajit svůj objev. Něco z něj však již navždy zůstane ve světě za Strojem.

I79 * "Existují" ideje, pocity, postupy, názory ..., které nejsme zatím schopni v principu pochopit * Srv. E71, E72. "Stroj" zde reprezentuje implementaci orákulí, které známe z computer science teorie, i nejasná podvědomá tušení, která známe ze snů a jiných mimologických informačních zdrojů, jakými mohou být třeba některá umělecká díla. Pozitivní rub agnostické části tvrzení I79 je právě v tom, že je možno očekávat a připravovat zcela nové cesty poznání. Máme šest smyslů a stovky jich mohou existovat v principu; máme jednu běžnou a několik experimentálních logik a tisíce mohou existovat v principu - jsme neuzavření a z tohoto hlediska je nutno posuzovat každou naši činnost a všechny její plány. Kdo ví, třeba je Stroj na spadnutí; nějaká ta generace počítačů navíc a pár objevů z oblasti psychologie - a bude nám tak, jako člověku, který po dlouhé době vylezl z bunkru na světlo. Jde o další variantu I60, E72: vhodnou (umělou) regulací biologických a informačních toků v organismu je možno ustavit takový stav individuálního i kolektivního vědomí, který bude možno kvalifikovat přídomkem "spokojenost", tj. takový stav, který bude ve všech směrech akceptovatelný, a přesto - na rozdíl od analogických stavů dosahovaných ryze mentálními prostředky (např. různé "extatické stavy") nebo prostředky chemickými (spokojenost narkomana) - bude plně kompatibilní s aktivním přístupem k životu.

I81 * Čím složitější je systém, tím je náchylnější k poruše * (srv. I19-I22) A tím větší část svých prostředků musí věnovat na zabezpečení základních funkcí. Administrativa bezpečnosti a robustnosti, zabezpečující vnější ochranu a příp. regeneraci - to vše bude klást stále větší nároky i na výpočetní systémy budoucnosti. Je zde mez, která znemožní růst informačních systémů. Nedostaneme se při jejich budování do paradoxní situace, v níž tato dodatečná administrativa podstatně sníží funkce-schopnost systému (tak jako některé operační systémy), tedy do situace, k níž se současný svět (doufejme, že jen dočasně) blíží v tom, že náklady na obranu životní úrovně ji samu podstatně sníží (v limitě natolik, že vlastně nebude, pokud jde o materiální hodnoty, téměř co bránit)? Nebo budeme nuceni riskovat tak, jako obyvatelé 384? Nebo se najde ještě nějaké jiné řešení? Jak velká bezpečnostní opatření lze vůbec uvažovat, jestliže - ve vesmírném měřítku - nejsme schopni odhadnout ani rozsah potenciálního nebezpečí?

(-23-) S.Lem: Pánův hlas. Osoby: vědci, Pán. Místo: výzkumné středisko nejvyšší vládní priority, jinak téměř současného typu.

Úcta, s níž se autoři tohoto referátu sklánějí před ohňostrojem literárních, logických i kosmogonických nápadů jednoho z největších děl scifi, jakož i inherentní složitost obsažených myšlenek, zabraňují pokusu detailněji rozebírat povídku, jejíž četba je evidentním "musem" pro skutečné zájemce o fantastiku a pravdu vůbec. Poznamenáme proto jen, že jde o otázky dešifrování případného vesmírného vzkazu /I82/, který - třebaže "inteligentní" - nemusí být nutně plodem "inteligentní bytosti" /I81/.

I82 * Apriori nelze vyloučit možnost vzniku resp. existence inteligentní informace bez inteligentního tvůrce * Srv. I63. Přitom nemáme na mysli ev. náhodný děj (opici, která náhodnými doteky kláves napíše Hamleta). A jen v nejhorším jsme ochotni připustit Darwinovský princip výběru (z neustále kolující informace přežívá jen ta nejschopnější), popř. možnost postupného samovolného ulpívání a obohacování informačního toku během jednotlivých vesmírných pulsů na blíže neurčeném nositeli (jehož existenci stvořily některé mytologicky orientované civilizace a pojmenovaly termínem "bůh", zatímco skupina vědců odchovaných industriální společností zde zvolila název Master's Voice - Pánův hlas), jak uvádí i J.Vladík v povídce ON. Informace se prostě může zrodit tak nevyzpytatelně jako vesmír sám, může být některým jeho místům vlastní právě tak jako jeho hmotná podstata. A můžeme být schopni pochopit její tajemství jen s tou mírou bezradnosti, s jakou chápeme třeba pojem nekonečna. Zatím...

I83 * (Pseudo)informace může vzniknout neúmyslně, třeba žertem * V Signálu P.Rozehnala vzbudila kdesi v USA zájem analyzátoru kosmických vzkazů podvržená recitace Nerudových veršů. Pro neznámý český text byla navržena řada interpretací - něco podobného by mimochodem mohlo sloužit jako dobrý test schopnosti programů navržených pro seriozní dešifraci případných signálů z vesmíru. V povídce P.Materny Kdo je idiot? se hádal počítač s programátorem o interpretaci takového signálu; zatímco počítač přikládá signálu význam přehlubokého vzkazu o způsobu rozmnožování (na logické abstraktní úrovni) předpokládaných inteligentních odesilatelů, šéf laboratoře neautomatizovaného myšlení prostě zjistí, že jde o posloupnost několika prvních přirozených čísel. Počítač přesto trvá na svém - je absurdní předpokládat, že by vyspělá civilizace sdělovala při vynaložení nesmírných energetických investic celému Vesmíru fakt, že umí počítat do deseti! Ať tak či onak, rádi bychom propagovali zásadu:

T84 * Každá forma pseudoinformace může být zdrojem (skutečné, cenné) informace * Dokonce i dezinformace má tu vlastnost; srv. I5. Pátráme-li po potenciálních zdrojích pravd, jejichž obsah i vznik je (aspoň zatím) zcela mimo dosah naší chápavosti, musíme se chytat i stébel. Co když je celý vesmír jen takový nevydařený antisofsem ve velkém? Co když různé ty Planckovy konstanty, Kelvinovy stupně a číslo PI nesou vzkazy daleko závažnější a daleko širšího dosahu než se nám zdá z toho, k čemu jsme byli schopni je využít? Teprve v současnosti jsme na stopě pravého (?) významu řetězců DNK a pomalu se chystáme využít přírodních genových NC-automatů k průmyslové výrobě. Astronomové toho zjistili o našem sluníčku tolik, že makroskopický protějšek těchto mikroúvah, totiž Slunce-počítač (P.Gloss: Opus bez názvu), se zdá technicky dokonce o něco méně absurdní než již prokázaná fakta. A vůbec, když se tak rozhlížíme po ostatních přírodních vědách - není náhodou skryt v živých hodinách ganglií švábů (viz R.R.Ward: Živé hodiny, kap. 13) třeba návrh concurrent Pascalu, v atomu kryptonu zaručeně korektní program pro výpočet největšího společného dělitele a v oku včely databázový systém pro ASŘ školství? (Vztahy, které objevili přičinliví snaživci v rozměrech pyramid, se příliš od těchto hypotéz neliší.)

Závěrečnou, 84. tézí chceme podepřít problematickou stavbu celého tohoto příspěvku, který programově není ničím jiným než snůškou pseudoinformací. Ale uznáte-li aspoň jistou oprávněnost této téze (samozřejmě bez těch úmyslných ironizujících absurdit pro pobavení), nezbyde nám, než se zamyslet nad potenciální oprávněností aspoň několika dalších. A víc jsme si ani netroufli přát.

...vtom někdo zaklepal na dveře...
(F.Brown)

RNDr.J.Hořejš,CSc.,ing.J.Franek

ISO-ROM PRO ZX SPECTRUM

Pro ZX Spectrum existuje už celá řada různých úprav paměti ROM. Důvod je jednoduchý - v původní ROMce je několik drobných chyb, ale hlavně (a to programátory stále provokuje) je v ní několik nevyužitých míst, z nichž nejdelší má více než jeden KB. V dlouhé řadě více či méně zdařilých verzí ROMky určitě se ctí obstojí ISO-ROM firmy Individual Software. Mezi spectristy je poměrně známá a populární hlavně díky podnikavcům, kteří ji kopírují do EPROMek. Při tomto způsobu šíření je kromě otázky autorských práv zanedbána otázka seriózní dokumentace. I mně se do rukou dostal neúplný a místy nepřesný popis instalace a funkce ISO-ROM. Nezbývalo než obsah paměti disassemblovat a pak trochu laborovat. Zde je výsledek:

Start systému

Oproti původní ROMce již sám start systému a jeho inicializace doznaly značných změn k lepšímu. ISO-ROM totiž rozeznává teplý a studený start. Ke studenému startu, na jaký jsme zvyklí a při němž je mazána paměť, dojde jen při prvém zapnutí. Při dalších průchodech programového čítače nulou se vždy kontroluje, zda nedošlo k porušení systémových proměnných STRMS. Pokud ne, proběhne pouze teplý start. Při teplém startu není mazána RAMka, takže nedojde ke ztrátě programu ani dat. Pouze se obnoví obsah životně důležitých systémových proměnných, registru IY a kanálových dat. Názorně je to vidět z přiloženého výpisu inicializační rutiny. Teplý start skáče do bodu WSTART. Je lhostejné, zda k průchodu nulou došlo hardwarově (tlačítkem RESET) či softwarově (RANDOMIZE USR 0, RST 0, JP 0, CALL 0). Tento mechanismus zachrání většinu systémových havárií, končících resetem. K vymazání paměti je pak nutno použít adresu 100 (např. RANDOMIZE USR 100), od které probíhá klasický RESET. Basicový příkaz NEW probíhá normálně.

Editace Basicu

je mnohem pohodlnější. Ihned si všimneme že řádkový kurzor, ukazující na aktuální řádek, bliká. Není třeba zdůrazňovat, jak se tím zvýší orientace na obrazovce. Do editační zóny můžeme řádek vytáhnout též přímo, pomocí příkazu "$n" (n je číslo řádku). V editovaném řádku se může kurzor pohybovat i nahoru a dolů. Když přitom přesáhne okraj editovaného řádku, způsobí to přechod řádkového editoru do řádku sousedního. Při zjištění syntaktické chyby se do jejího místa kromě chybového kurzoru (?) přesune i editační kurzor.

Převod HEXA-DEC a zpět

Pomocí ISO-ROM můžeme převádět dekadické hodnoty na hexadekadické a zpět. Slouží k tomu znaky "5" a "6". Příklady použití:

```
PRINT 6A0  (ENTER) vytiskne 160
PRINT 5160 (ENTER) vytiskne 6A0
Program:  10 LET a = 620 + 6D0
          20 PRINT "DEC ";a;" = HEX ";5a
Vytiskne: DEC 240 = HEX 6F0
```

Tímto způsobem lze libovolně kombinovat dekadická i hexadekadická zadání hodnot, počítat s nimi a výsledky vyjadřovat dle potřeby tak či onak.

Příkazy pro Beta disk

ISO-ROM zjednodušuje ovládání Beta disku. Namísto sekvence "RANDOMIZE USR 15363: REM:" se zadá jen "!". Příklad:

```
místo:    RANDOMIZE USR 15363: REM: LOAD "program"
postačí:  !LOAD "program"
```

Tato verze nefunguje s Beta diskem verze 5.xx, která se volá na adrese 15619.

Mapa microdrivu

Příkazem PRINT USR 15360 lze vytisknout bitovou mapu obsazení kazety microdrivu a zlepšený katalog.

Nové přímé povely Basicu

jsou použitelné pouze přímo (samostatně), nemohou tedy být součástí basicového programu. Vždy se ukončují klávesou ENTER:

"@"	- skok do monitoru (návrat = BREAK)
"$n"	- editace řádku n
"$"	- rekonstituce systémových proměnných tak, aby začátek Basicu byl na 23755
"&"	- skok na adresu uloženou v systémové proměnné NMIREG (23728 a 23729)
"CUR-UP"	- posun editačního kurzoru o řádku výš
"CUR-DOWN"	- posun editačního kurzoru o řádku níž

Monitor

Také ISO-ROM má implementován jednoduchý monitor paměti. Z Basicu ho lze vyvolat přímým příkazem "2" (též RANDOMIZE USR 102). Lze do něj vstoupit i hardwarově, pomocí NMI. To je zvláště cenné v případě zaseknutí programu ve strojovém kódu.

Monitor zobrazuje všechny údaje hexadekadicky, ve čtyřech sloupcích. Je to vždy inverzně adresa a napravo od ní její obsah. Při vstupu do monitoru jsou na začátku stránky tři důležité adresy, na které jsou v okamžiku přerušení dočasně uloženy tyto okamžité hodnoty:

na #5CA8 hodnota programového čítače (pokud přerušený program běžel v RAMce),
na #5CAA hodnota SP,
na #5CAC hodnota systémové proměnné CHARS (aby byla zaručena funkce monitoru, přepíná si na standardní hodnotu v ROMce).

Spodní dva řádky zobrazují obsah registrů procesoru v okamžiku přerušení takto:

A	BC	DE	HL	IY	z c s
A'	BC'	DE'	HL'	IX	int

Pokud je nastaven některý stavový indikátor, příslušné písmeno červeně bliká. Stav indikátoru S je indikován písmeny "p" nebo "m". Stav klopného obvodu interruptu je hlášen jako "ei", nebo "di". Po vstupu do monitoru je přerušení povoleno.

Můžeme měnit obsah paměťového místa uprostřed obrazovky, na které ukazuje blikající kurzor. Změna se provede zadáním hexadekadického čísla, ukončeného ENTERem. Chybu během zadávaní můžeme odstranit pomocí DELETE. Po každém ENTERu se automaticky přejde na další adresu. Na předchozí adresu se lze vrátit pomocí CS+ENTER. Přechod na další stránku zabezpečí stisk klávesy "n", návrat na předchozí stránku stisk "v".

Přímé nastavení adresy umožňuje zadání "s hhhh", kde hhhh je hexadekadická adresa. Program ve strojovém kódu spouští klávesa "r", po jejímž stisku se vypíše RUN a čeká se na zadání adresy. Po odeslání ENTERu proběhne program, který se po instrukci RET vrátí zpět do monitoru. V něm si pak můžeme pohodlně prohlédnout hodnoty registrů v okamžiku návratu.

Varování: s implementovaným monitorem je sice možno zastavit každý program a zachytit zhroucení systému, ale někdy je před návratem do Basicu nutno zrestaurovat systémové proměnné. S rezervou musíme brát i údaj o hodnotě SP - jak už bylo řečeno, při zastavení programu v ROMce je údaj nesprávný. Též případný zákmit kontaktu tlačítka NMI způsobí chybu. Je nutno si uvědomit také to, že při volání rutiny příkazem RUN nelze zadat výchozí stavy registrů procesoru (to dokáže nová LECROM, o které informovala Mikrobáze 1/1989).

Pro své účely monitor používá tyto systémové proměnné:

MEMBOT - jen část od 5CA8H do 5CAFH
CHARS
FLAGS
FLAGS2
DFSZ
PFLAG
DFCC a DFCCL
SPOSN a SPOSNL

Při každém vstupu do monitoru je volána rutina CHRESTO, která je součástí teplého startu (viz přiložený výpis). Ta mimo jiné obnoví kanálová data a přiřazení proudů. To má přirozeně za následek ztrátu všech změněných údajů této oblasti- např. ztrátu spojení s obsluhou tisku v RAMce.

Kompatibilita programů

Některé programy, především hry, nepracují s kombinací ISO-ROM a KEMPSTON JOYSTICK. Podle firmy ISO to není zaviněno voláním rutin ROM, ale tím, že např. Chequered Flag a první verze Ghost Busters pracují v módu IM 2, pričemž jako tabulku adres používají nepoužitou oblast ROMky, kde je FFH. Tyto hry však někdy nepracují s interfacem KEMPSTON ani v originální ROMce.

Pochopitelně nebudou pracovat originální firemní programy, které si provádějí kontrolní součet ROMky.

Kompatibilita hardwaru

Některá hardwarová rozšíření, zejména interfacy pro tiskárny, mají ovládací software ve vlastní stínové ROMce, pro kterou využívají oblast nad adresou 14446. Tu však zároveň využívá ISO-ROM. Tuto situaci je třeba řádně zvážit. Pokud je ISO-ROM vestavěna do počítače, pomůže jen vestavění vypínače do tiskového interfacu. Tím se buď přeruší jeho napájení, nebo zablokuje ROM CS v době, kdy se netiskne. V průběhu tisku se pak nesmí využívat nových příkazů ISO-ROM, zejména "kurzor nahoru" a "kurzor dolů". Výkonné rutiny těchto příkazů jsou totiž právě v místě, které je zamaskované stínovou ROMkou.

Zapojení ISO-ROM do ZX Spectra

Tento problém probere samostatný článek v příštím čísle Mikrobáze.

Reset a NMI

Připojení resetu je klasické, pomocí tlačítka připojeného paralelně k C27. Tlačítko NMI by mělo být bez zákmitů (osvědčil se miniaturní mikrospínač). K dalšímu potlačení zákmitů výrazně přispěl kondenzátor M1, paralelně připojený ke kontaktu.

Závěr

Tak jako ostatní upravené ROMky pro ZX Spectrum, má smozřejmě i ISO-ROM opraveny všechny známé chyby původní ROMky. K dokonalosti už jí asi moc neschází - snad jen některé vymoženosti LECROM- jenže ta zas neumí leccos z ISO-ROM. Kdyby tak bylo něco mezi! Nebo vestavět do počítače obě. Snad až budou dostupnější 27256!

Dan Meca

Výpis rutiny teplého startu ISO-ROM

Rutina začíná na obvyklém místě rutiny START/NEW ZX Spectra, tj. na adrese 11CBH.

```
CST/NEW DI                  ;Studený start/new.
        LD      B,A         ;Uschování flagu.
        LD      A,3FH       ;Nastavení registru I na
        LD      I,A         ;hodnotu 3FH.
        LD      H,D         ;Přenesení hodnoty z DE
        LD      L,E         ;(START=FFH,NEW=RAMTOP).
RAMFILL LD      (HL),02H    ;Naplnění paměti až po
        DEC     HL          ;3FFFH hodnotou 02H.
        CP      H
        JR      NZ,RAMFILL
RAMREAD AND     A           ;Příprava k odečítání.
        SBC     HL,DE       ;CY je nulováno, když
        ADD     HL,DE       ;je nalezen konec tes-
        INC     HL          ;tované části.
        JR      NC,RAMDONE  ;Skok na konci.
        DEC     (HL)        ;Snížení ze 2 na 1.
        JR      Z,RAMDONE   ;Když dává 1, je chyba. Dosažitelná
                            ;paměť je nastavena podle HL.
        DEC     (HL)        ;Snížení z 1 na 0.
        JR      Z,RAMREAD   ;Na další pozici, je-li paměť OK.
RAMDONE DEC     HL          ;HL ukazuje na poslední použitelnou
                            ;pozici.
        EXX                 ;Přepnutí na uschované syst. prom.
                            ;(nemá význam při startu).
        LD      (PRAMT),BC  ;Obnovení původních PRAMT,RASP/PIP
        LD      (RASPIP),DE;UDG (při NEW).
        LD      (UDG),HL
        EXX                 ;Zpět na základní registry.
        INC     B           ;Test CST/NEW.
        JR      Z,RAMSET    ;Skok vpřed při NEW.
        LD      (PRAMT),HL  ;Nast. konce fyzické RAM.
        LD      DE,3EAFH    ;Poslední bajt písmene "U" v gene-
                            ;rátoru znaků.
        LD      BC,00A8H    ;Počet kopírovaných bajtů.
        EX      DE,HL       ;Přepnutí ukazatelů.
```

```
        LDDR              ;Zkopírování znaků do UDG.
        EX      DE,HL     ;Ukazatele zpět.
        INC     HL        ;Ukazuje začátek UDG.
        LD      (UDG),HL  ;Nastavení UDG.
        DEC     HL        ;Jednu pozici dolů.
        LD      BC,1440H  ;Nastavení syst. prom.
        LD      (RASPIP),BC;RASP a PIP
RAMSET  LD      (RAMTOP),HL;Nastavení RAMTOP.
        LD      HL,5CCAH  ;Nastavení prom. DATADD
        LD      (DATADD),HL;na poslední pozici kanálových dat.
        INC     HL        ;Nastavení PROG a VARS na
        LD      (PROG),HL ;následující pozici.
        LD      (VARS),HL
        LD      (HL),80H  ;Ukazatel konce oblasti proměnných.
        INC     HL        ;E-LINE bude ukazovat na
        LD      (ELINE),HL ;následující pozici.
        LD      (HL),0DH  ;Nastavení znaku CR do Edit-line
        INC     HL        ;a zakončení ukazatelem
        LD      (HL),80H  ;konce Edit-line.
        INC     HL        ;Zvýšení pozice pro nalezení hodnot
        LD      (WORKSP),HL;pro WORKSP,STKBOT
        LD      (STKBOT),HL;a STKEND.
        LD      (STKEND),HL

WSTART  LD      HL,(RAMTOP);Uložení ukazatele 3FH
        LD      (HL),3EH  ;do místa RAMTOPu
        DEC     HL        ;Snížení pozice.
        LD      SP,HL     ;Nastavení SP na tuto pozici.
        DEC     HL        ;O dvě pozice dolů pro nalezení
        DEC     HL        ;správné hodnoy pro ERR-SP.
        LD      (ERRSP),HL ;Nastavení ERR-SP.
        LD      A,04H     ;Zelený BORDER.
        OUT     (#FE),A
        LD      A,38H     ;Nastavení systém. prom. barvy na
        LD      (ATTRP),A ;FLASH 0, BRIGHT 0, PAPER 7, INK 0.
        LD      (ATTRT),A
        LD      (BORDCR),A
        CALL    CHRESTO   ;Tuto rutinu používá také monitor.
        DEC     (IY-58)   ;Nastaví KSTATE-0 na FFH.
        DEC     (IY-54)   ;Nastaví KSTATE-4 na FFH.
        JR      POKINI    ;Skok na pokračování inicializace
                          ;systému.

CHRESTO LD      HL,3C00H  ;Inicializace syst. prom. CHARS.
        LD      (CHARS),HL
        IM      1         ;Mód přerušení 1.
        LD      IY,5C3AH  ;Registry IY ukazují na ERR-NR.
        LD      DE,5CB6H  ;Základní adresa oblasti kanálo-
        LD      (CHANS),DE ;vých informací na CHANS.
        LD      HL,15AFH  ;Základní kanálová data jsou pře-
        LD      BC,0015H  ;místěna z tabulky na 15AFH do
        LDIR              ;oblasti kanálových dat.
        INC     HL
        INC     HL
        LD      DE,5C10H  ;Základní data proudů jsou
        LD      C,0EH     ;přemístěna na své místo.
        LDIR
        LD      HL,0523H  ;Inicializace syst. prom.
        LD      (REPDEL),HL;REPDEL a REPPER.
        RET

POKINI  EI                ;Uvolnění přerušení
        SET     1,(IY+1)  ;Signál "Tiskárna použita" a volání
        CALL    0EEBH     ;části PRB-BYTES bez mazání tiskové
                          ;ho bufferu.
        LD      (IY+49),02H;Nastavení velikosti editační zóny.
        CALL    0D6BH     ;Smazání celé obrazovky.
        XOR     A         ;Pořadí hlášení je 0.
        LD      DE,1538H  ;Začátek tabulky hlášení.
        CALL    0C0AH     ;Tisk úvodního hlášení.
        SET     5,(IY+2)  ;Signál pro dolní část obrazovky.
        JR      MAIN1     ;MAIN-1 do hlavní smyčky.
```

ZKUŠENOSTI S PMD 85-2

Nová verze počítače PMD 85-2 je dost vylepšená. Přesto je třeba upozornit na různé nepříjemnosti jejího interpreteru Basic G-2.0, z nichž uvádím:

1) Příkaz DEG určuje, že argument goniometrických funkcí se uvádí ve stupních. Tento příkaz však neplatí pro inverzní funkci ATN; ta vychází vždy v radiánech.

2) Při výpočtu záporných mocnin kladného základu menšího než 1 (např. 0.1^(-2)=100) počítač hlásí dělení nulou. V obecných případech proto místo samotného příkazu

```
RE=ZK^EX
```

musíme použít jednoduchou subrutinu:

```
RE=ZK : IF EX<0 THEN RE=1/RE
RE=RE^ABS(EX)
RETURN
```

Na tyto nedostatky je třeba upozornit hlavně proto, že návod k PMD 85 je spíše záznam veřejného předvádění než dokumentace k počítači, který přijde do rukou alespoň poloprofesionálních. Zejména recenzenti, kteří budou posuzovat příští publikace tohoto druhu, by je měli srovnávat s popisy u nás rozšířených zahraničních počítačů.

Ing.Jiří Vondrák,CSc.

PROGRAMÁTOR PAMĚTÍ EPROM - PRG 85

Jeho výrobcem je TESLA Piešťany, k.p. Je konstruován pro mikropočítač téhož výrobce - PMD 85. Bohužel není zcela kompatibilní s jeho druhou verzí PMD 85-2, vyráběnou v Tesle Bratislava.

Fyzické připojení k PMD 85-2 je dle uživatelské příručky shodné, ale programové zavedení je odlišné. Pro přenesení obsahů sounáležících pamětí EPROM nelze použít příkazu JOB ani ROM. V režimu MONITOR je nutné vytvořit vlastní program s využitím subrutiny TRANSFER:

```
CALL TRANSFERR          CD 00 8C
DW ADR1                 XX XX
DW BATCH                00 07
DW ADR2                 00 70
RET                     C9
```

ADR1 volíme podle pozice uložení EPROM v modulu ROM. Po vyvolání tohoto programu instrukcí JUMP musíme opravit přenesená data:

Adresa	Původní data	Nová data
7436H	D1H	89H
743BH	D2H	8AH
7440H	D3H	8BH
7445H	D4H	8CH
744AH	D5H	8DH
744FH	D6H	8EH
7454H	E7H	8FH
7459H	E8H	90H
745EH	E9H	91H

Posledním krokem k inicializaci programů je start od adresy 73FFH (JUMP 73FFH). Další postup viz uživatelská příručka PRG-85.

Ing.Vladimír Georgiev

ZX SPECTRUM S ROZŠÍŘENOU PAMĚTÍ (3)

ad 7) Posilovač sběrnice

Podíváme-li se na to, jak je uvedená problematika řešena na stránkách Amatérského radia či jiných časopisů, nenajdeme jediné schéma, které by ji uspokojivým způsobem řešilo. Existuje sice několik schémat popisujících posílení portů nebo datové sběrnice, ale ani jedno, které by řešilo posílení celé sběrnice. Schémata, která mezi spectristy kolují, mají svá omezení v tom, že nejsou univerzální (nepočítají např. s režimem DMA).

Popisovaný posilovač sběrnice plní několik odlišných funkcí:

1) Převádí sběrnici Spectra na sběrnici (e) STD
2) Odděluje sběrnici Spectra od sběrnice (e) STD
3) Posiluje všechny signály Spectra nezbytné pro spolupráci s (e) STD
4) Rozděluje adresový prostor pro porty na 128 portů uživatelských a 128 portů vyhrazených systému (klasické Spectrum jich může využívat jen 8)
5) Obsahuje logiku řídící směr přenosu informace (umožňuje pracovat i v módu DMA)

Vlastní zapojení lze modifikovat podle dostupných součástek.

Popis funkce posilovače sběrnice

Na schématu jsou signály příslušející Spectru vlevo a signály sběrnice (e) STD vpravo (viz obr. 7/1-14). Jak vidíme, jsou tam dva typy signálů. U jedněch jde přenos informace vždy jedním směrem, druhé jsou obousměrné. U těch je třeba rozhodnout, v kterém okamžiku se bude informace přenášet ze Spectra do (e) STD a kdy opačně.

Jednosměrné signály jsou :/M1, /RFSH, /HALT, /BUSAK, A15GE, /RAMS, CLK, /RESET, /RESFF, /WAIT, /BUSRQ, /NMI, /INT.

Obousměrné: /RD, /WR, /MREQ, /IORQ, datová a adresová sběrnice. Je nutné si uvědomit, že signály /RD, /WR, /MREQ, /IORQ negeneruje pouze procesor ve Spectru, ale libovolná deska, která zažádá o sběrnici Spectra a převezme její řízení.

Poznámka: / označuje negovaný signál.

Kombinační síť složená z obvodů 74LS04, 7450, 7460 realizuje tuto logickou funkci:

```
Y = / ((/BUSAK./WR) + (/IORQ./M1) +
    + (/MREQ./RD./RAMS) + (A7./RD./IORQ))
```

a vyjadřuje podmínky, kdy je třeba otočit budiče směrem do Spectra. To nastane v případě, kdy jsou aktivní signály:

/BUSAK./WR - procesor potvrdil žádost o sběrnici a čeká, že do paměti Spectra bude zapsána informace z (e) STD (mód DMA).

/IORQ./M1 - procesor potvrzuje přijetí maskovatelného přerušení a podle jeho módu očekává z datové sběrnice instrukci nebo vektor přerušení vyslaný z (e) STD.

/MREQ./RD./RAMS - procesor čte z externí paměti. Sběrnice (e) STD umožňuje přímo adresovat až 1 MB.

/A7./RD./IORQ - procesor čte informaci z uživatelských portů 0-7FH.

Význam signálů je zřejmý z popisu normy (e) STD nebo ZX-Sp 80K. Zajímavé je pouze využití signálu /IORQGE, který zamezuje přihlášení obvodu ULA, když adresujeme porty 0-7FH. Signál /RESET je synchronizován s /M1.

Stavba a oživení

Posilovač lze postavit na univerzální desce. Budiče je vhodné umístit do patic. Ostatní obvody lze zapájet přímo do desky. Snažte se, aby přívody od ZX-Sp 80K k budičům byly co nejkratší. Optimální by bylo provést propojení plochým krouceným kabelem (vždy 2 vodiče spolu). Přílišná délka kabelu zvyšuje jeho kapacitu a může být příčinou nespolehlivosti.

1) Oba konce desky jsou opatřeny konektory FRB 62 pinů - "sameček", typ TY 517 62 x1. Pro tuto variantu je třeba vyrobit propojovací kabel mezi Spectrem a posilovačem. Na straně Spectra je použit konektor WK 465 80 a na straně posilovače konektor FRB TX 517 62 xx . Protože konektor FRB má 62 pinů a konektor Spectra pouze 56, některé kontakty konektoru FRB zůstanou nazapojeny. Přiřazení kontaktů:

Spectrum	28B	27B	26B	...	7B	6B	SLOT	4B	3B	2B	1B
FRB	61	59	57		19	17	15	13	11	9	7

Spectrum	28A	27A	26A	...	7A	6A	SLOT	4a	3A	2A	1A
FRB	62	60	58		20	18	16	14	12	10	8

Označení konektoru Spectra je uvedeno v kapitole o STD BUS.

2) V opačném případě lze přímo z desky vyvést propojovací kabel zakončený konektorem WK 465 80. Je to přímý konektor s palcovou roztečí pro připojení ke sběrnici Spectra. Diody jsou germaniové, např. GA 200-207 apod. Kritická je pouze jejich rychlost. Oproti křemíkovým mají menší úbytek napětí v propustném směru. V případě problémů s rychlostí diod lze použít zapojení tranzistoru, u něhož spojíme bázi s kolektorem. Na desku umístíme větší (centrální) elektrolytický kondenzátor a jednotlivé rozvody k IO blokujeme menšími, nejlépe poštářkovými (cca 100 nF). Kondenzátory nejsou ve schématu zakresleny. Místo budičů 8286 lze použít méně dostupné, ale rychlejší obvody 74LS245, které mají jiné zapojení vývodů. Při jejich použití je nutno zapojení posilovače upravit.

Při pozorné stavbě funguje posilovač po prvním zapnutí. Problémy mohou vzniknout při použití druhojakostních (pomalých) obvodů 8286 nebo pomalých diod.

Seznam potřebných polovodičů

3 x 8286
2 x 3216
4 x 74LS04
1 x 7450
1 x 7460
1 x 7474 9 x GA 200-207 (nebo podobné)

Poznámka: Na obrázcích 7/1 - 7/14 je nedůsledné značení negace u některých signálů. Správně je uvedeno v kapitole 6 o (e) STD.

ad 8) Řadič pružného disku pro ZX-Sp 80K

1. Popis řadiče

Řadič pružných disků pro počítač ZX-Sp 80K je jednoduše sestrojitelný a snadno ovladatelný. Využívá integrovaného řadiče řady 2797 firmy Western Digital. Oproti jiným konstrukcím, založeným zpravidla na řadiči Intel 8272, se výrazně zjednodušuje oživování celé desky i programová obsluha. Některé nevýhody - např. nemožnost prohledávání v toku dat z disku - ve srovnání s přednostmi považujeme za nepodstatné. Rovněž otázka získání příslušného IO je sporná, neboť řadič Intel dosud není na trhu a v zahraničí lze zakoupit oba typy.

1.1 Sestava desky řadiče

K základním částím desky řadiče patří zejména:
- vlastní IO řadiče, WD 2797
- obvod Z80A PIO (UA855D), který slouží jako port i jako generátor přerušení pro CPU
- obvody selekce řadiče a obvodu PIO, budič datové sběrnice s logikou výberu a otáčení směru přenosu
- budiče diskového interfacu (oddělovače a invertory s otevřeným kolektorem)
- pomocné obvody řadiče (obvod pro zpoždění HEAD LOAD, generátor 2/1 MHz, obvod PUMP, generátor 50 Hz pro generování přerušení)

Schéma zapojení je na obr.8.

1.2 Technické parametry řadiče

Počet připojitelných mechanik: 4
Typ mechanik: libovolný (3.5", 5.25", 8")
    jednostranné i oboustranné
    (lze připojit libovolnou kombinaci)
Typ záznamu: jednoduchá i dvojnásobná hustota
Formát záznamu: libovolný, s délkou sektoru
    128, 256, 512 nebo 1024 bajtů
Dosahovaná kapacita: 180 KB-1,2 MB (formátovaná)
    podle použitého formátu a mechaniky
Přepínání režimu činnosti: výhradně programové
    užitím obvodu Z80A PIO
Adresování: adresy 10H-17H (8 I/O adres)
Ostatní vlastnosti:
    - možnost spolupráce s obvodem DMA
    - plné využití přerušení od řadiče
    - vytváří všechny signály potřebné
      pro práci s různými mechanikami

1.3 Funkce řadiče

Funkce řadiče vychází plně z filosofie integrovaného obvodu WD 2797 nebo ekvivalentu (SAB 2797 SIEMENS). Tento obvod je vybaven integrovaným PLL separátorem dat, který vyžaduje připojení minima vnějších součástí. Rovněž nastavení je velmi jednoduché a vystačí s minimem pomocných přístrojů (osciloskop).

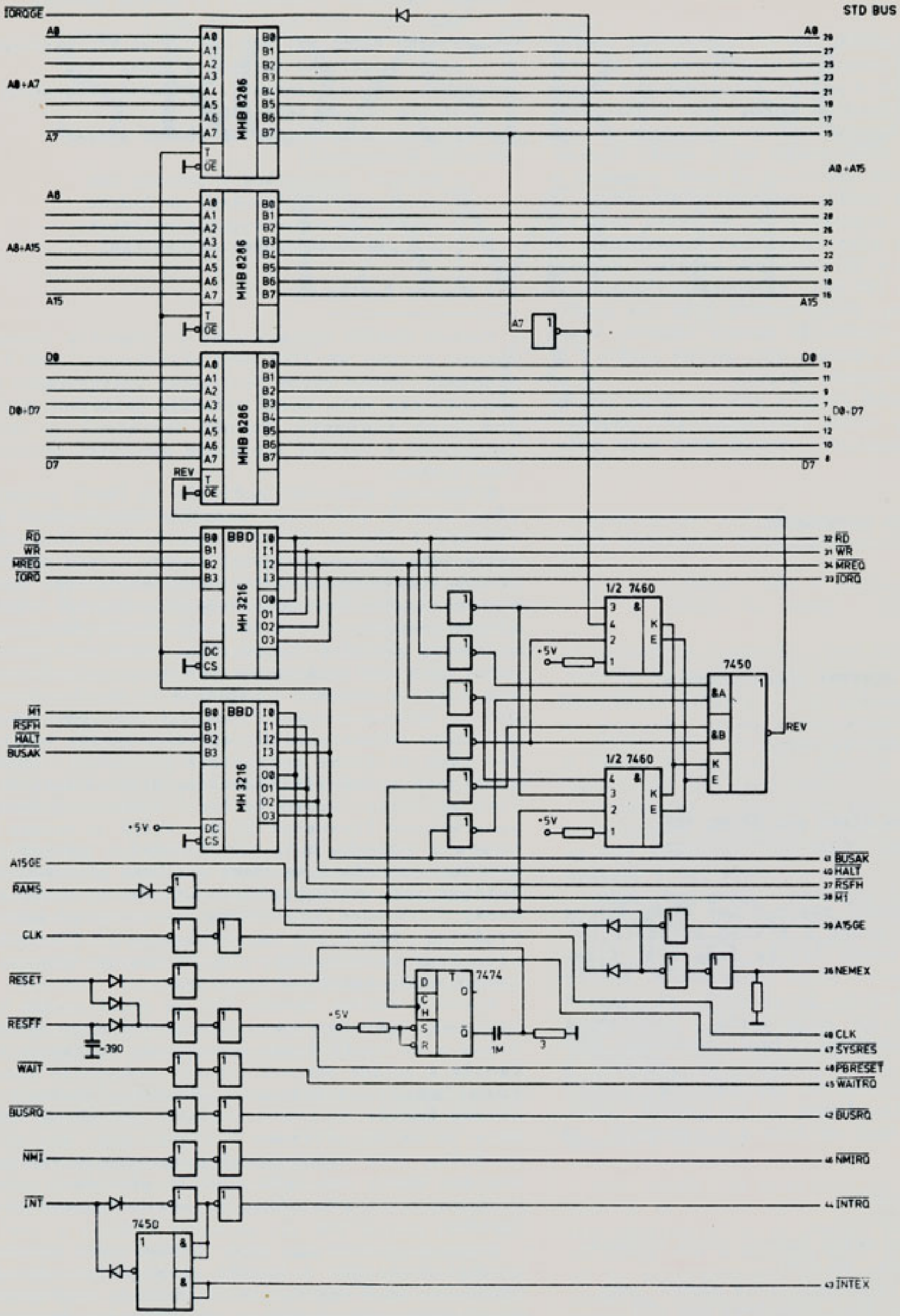
S datovou sběrnicí počítače je řadič propojen přes budič sběrnice, který posiluje data při výstupu z řadiče či z obvodu PIO. K přepínání směru toku dat se používají obvody, které detekují jednak stav čtení z řadiče nebo obvodu PIO, jednak potvrzení přerušení od obvodu PIO, aby byl umožněn přenos vektoru v módu IM 2. V něm je však krajně nežádoucí přerušení obvodu ULA, který není schopen generovat vektor a navíc je mimo prioritní řetězec. Jak je uvedeno v kapitole 7, budič sběrnice STD je vybaven možností blokovat přerušení obvodu ULA. Této možnosti se při aplikaci řadiče využívá. Tím však ztrácíme periodické přerušení, vhodné např. pro snímání klávesnice, blikání kurzoru a jiné aplikace. Proto je řadič vybaven pomocným generátorem frekvence 50 Hz, která je vedena na bránu B obvodu PIO, kde dochází k periodické aktivaci přerušení se všemi atributy přerušovacího řetězce Z80. Nutno podotknout, že po instalaci desky s obvodem Z80 CTC se tento obvod nebude využívat. Řadič sám nevysílá povel k blokování přerušení obvodu ULA - to je úkolem jednoho z obvodů desky systémové konzole.

Hlavním úkolem obvodu PIO je sloužit jako V/V zařízení. Obě brány jsou využity v bitovém režimu. Brána A se využívá pro specifikaci režimu celého řadiče a ovládá některé signály výstupního konektoru řadiče. Kromě toho převádí přerušení od obvodu řadiče do tvaru vhodného pro obvody Z80. Dále je na ni přiveden i signál DRQ, aby bylo umožněno jeho vyhodnocení programem i bez instalace desky DMA (při využití přerušení nelze sledovat stav DRQ přímo ze stavového registru řadiče).

Brána B slouží pro výběr jednotlivých mechanik pružných disků, k řadiči připojených. Rovněž obsahuje vodič READY, který informuje o připravenosti mechaniky, a výstup přerušovacího generátoru. Dva bity brány B prozatím nejsou využity.

Funkční popis řadiče WD 2797 je u nás poměrně rozšířený ve formě kopií firemních manuálů, i jejich překladů. Proto není funkce řadiče v této kapitole popisována. Rovněž obvod Z80A PIO je popsán v mnoha pramenech. Proto se věnujme jen detailům vlastního zapojení.

Řadič vyžaduje zdroj hodinových pulsů 1 nebo 2 MHz - podle toho, užíváme-li mechaniku 5.25" nebo


STD BUS
IORQGE
A8
A8÷A7
A7
MHB 8286
A8÷A15
A15
D8
D8÷D7
D7
REV
RD
WR
MREQ
IORQ
BBD
MH 3216
M1
RSFH
HALT
BUSAK
+5V
DC
CS
1/2 7460
7450
&A
&B
K
E
A15GE
RAMS
CLK
RESET
RESFF
390
WAIT
BUSRQ
NMI
INT
7474
D
C
H
S
R
T
Q
1M
3
32 RD
31 WR
34 MREQ
33 IORQ
43 BUSAK
40 HALT
37 RSFH
38 M1
39 A15GE
36 NEMEX
49 CLK
47 SYSRES
48 PBRESET
45 WAITRQ
42 BUSRQ
41 NMIRQ
44 INTRQ
43 INTEX

8". Hodiny by měly být v toleranci +-15, průběh by měl mít střídu 1:1. To lze nejlépe zajistit krystalovým generátorem o frekvenci 4 MHz s vydělením dvěma nebo čtyřmi. V případě užití krystalu 2 MHz je nutno věnovat pozornost symetrii generovaného průběhu. Schéma možné varianty oscilátoru s krystalem 2 MHz je na obr.8/1. Symetrická zapojení oscilátoru jsou obecně náchylnější k rozkmitání na parazitních kmitočtech. Tomu se lze vyhnout nesymetrickým zapojením na frekvenci 4 MHz a příslušným vydělením frekvence.

Smyčka fázového závěsu řadiče vyžaduje připojení vnějšího nesymetrického integračního obvodu (PUMP). Časová konstanta se rovněž přepíná podle použité mechaniky. To je provedeno užitím tranzistoru, který připíná přídavný kondenzátor do obvodu PUMP při přepnutí do režimu 5.25". Tím se časová konstanta obvodu přibližně zdvojnásobí. Místo termistoru lze na místě odporu použít běžný rezistor.

Obvod pro zpoždění signálu HLT (Head Load Timing) je tvořen monostabilním klopným obvodem 74LS123. Ovšem můžeme použít libovolný jiný MKO, který dává možnost nastavení doby kyvu v rozsahu cca 10-100 ms.

Budiče diskového interfacu v profesionálních aplikacích bývají tvořeny hysterezními hradly s otevřeným kolektorem. Tato hradla obvykle nejsou k dispozici, proto byla nahrazena běžnými hradly dle potřeby (buď invertujícími, nebo neinvertujícími). V některých lacinějších aplikacích naopak bývá zvykem vstupní hradla (tj. z disku do řadiče) vypustit a nahradit je odporovou sítí. Tuto variantu jsme nepoužili - jednak zvyšuje chybovost, a špatným zapojením diskového kabelu může dojít k ohrožení obvodu řadiče.

Všechny vstupy z diskového interfacu musejí být připojeny na +5 V přes odpory 150-390 Ohmů, neboť mechaniky mají své výstupy zásadně typu OC. Pokud někdo použije hradla typu OC i pro oddělování vstupních signálů, nesmí pochopitelně zapomenout zapojit příslušné odpory k jejich výstupům. Nebojte se použít nízké hodnoty odporů; vysoké hodnoty způsobují příliš velké zpoždění - řadič pak pracuje nespolehlivě, nebo nepracuje vůbec.

Signály, které vaše mechanika neposkytuje nebo nevyžaduje (FILE UNSAFE, LOW CURRENT a další) ponechte nezapojené. Pokud jste si jisti, že je nebudete nikdy potřebovat, nemusíte osazovat ani příslušné budiče.

2. Konstrukce a nastavení řadiče

Celý řadič se bez problemů vejde na jednu univerzální desku malého evropského formátu. Všechny součástky je vhodné zapájet - s výjimkou vlastního řadiče, obvodu PIO a ev. budiče datové sběrnice s ohledem na jeho možnou výměnu při nevyhovující rychlosti. Na místě kapacitního trimru doporučujeme typ WN 70419 - 60 pF, nebo WN 70425 - 50 pF. Odporové trimry jsou lepší keramické, ale postačí i pertinaxové. Při návrhu rozmístění součástek na desce veďte signály vysokých frekvencí (hodiny, Read/Write Data...) co nejkratšími spoji.

Samozřejmostí jsou blokovací kondenzátory u obvodu LSI a u budiče sběrnice. Je vhodné vytvořit si servisní propojku uzemňující vývod TEST řadiče. Ten není třeba vyzvedávat odporem proti +5 V; odpor je už vestavěn na struktuře čipu.

Při pečlivé práci na univerzální desce a při vedení spojů samopájitelnými vodiči můžete mít řadič hotový za 1-2 dny. Po nastavení pracuje velmi spolehlivě na první zapojení.

Oživení a nastavení řadiče zahájíme s vyjmutými obvody řadiče a PIO kontrolou činnosti oscilátoru. Frekvenci je nejlépe měřit čítačem, ale v nouzi postačí i dobře cejchovaný osciloskop. Frekvence musí být v toleranci +-15 a při uzemňování vodiče A1 obvodu PIO (vývod 14 objímky) se musí přepínat mezi 2 MHz (neuzemněn) a 1 MHz (uzemněn). Při překročení tolerance nemůže řadič uspokojivě pracovat (záznam na disku nebude čitelný jinými počítači, nebo se na disk ani nevejde celá stopa).

Po oživení oscilátoru a kontrole na zkraty zasuneme řadič do objímky. Po připojení napájecího napětí nejdřív provedeme reset (uzemněním příslušného vývodu sběrnice - PBRESET) a poté uzemníme vývod TEST řadiče. Pozor - v okamžiku resetu nesmí být TEST uzemněn - řadič by nepracoval!

Nyní nastavíme volnoběžnou frekvenci VCO. Při nezapojených vodičích A0, A1 obvodu PIO nastavíme frekvenci na vývodu 16 řadiče (DIRC) změnou kapacity trimru na 250 kHz. Při uzemnění vývodu A0 (15 - PIO) se frekvence zdvojnásobí, při uzemnění vývodu A1 (14 - PIO) klesne na 125 kHz.

Šířku čtecího impulsu nastavíme pozorováním pulsu na pinu 29 řadiče (TG43) a otáčením trimru připojeného k vývodu 18 (RPW) na 450 ns (opět při nezapojených A0, A1 PIO).

Zápisovou předkompenzaci nastavíme přibližně podle údajů výrobce mechaniky. Zpravidla se pohybuje kolem 1/3 šířky zápisového impulsu. Nastavujeme otáčením trimru připojeného k vývodu 33 (WPW) při pozorování pulsu na výstupu 31 (WD).

Obvod zpoždění HLT nastavíme (podle údajů připojených mechanik) na nejdelší hodnotu. Při použití mechanik, které neužívají signál HL, nastavíme obvod na minimum.

Pokud byly předchozí kroky úspěšně provedeny, je vysoká pravděpodobnost, že řadič je připraven k práci. Zbývá zasunout obvod PIO a vyzkoušet spolupráci s počítačem.

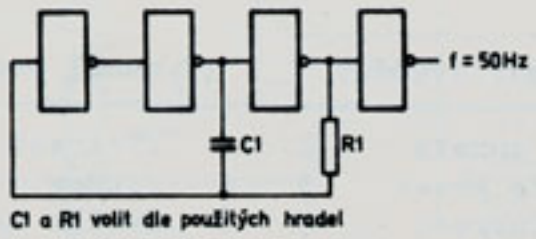
2.1 Doporučené zapojení diskového konektoru

Typ konektoru: FRB 30 pin, vidlice ("sameček")

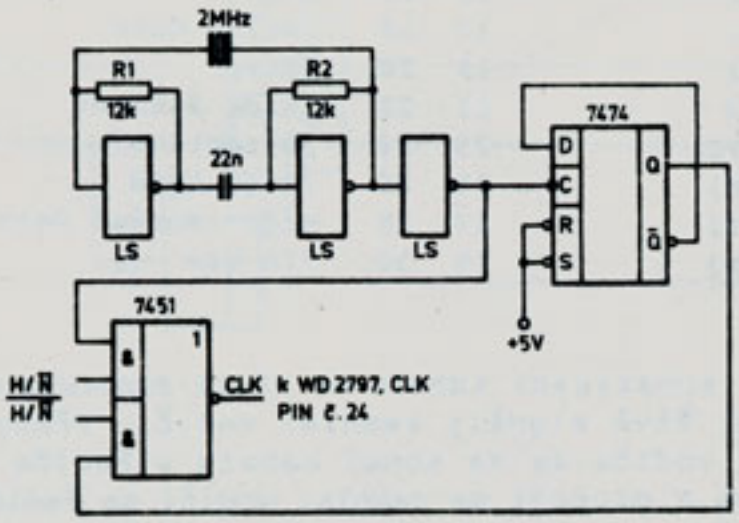
Zapojení vývodu:			(/ značí negaci)
/File unsafe	1	2	/Track00
/Unsafe Reset	3	4	/Index
/Low Current	5	6	/Write Protect
/Switch Filter	7	8	GND
GND	9	10	Read Data
GND	11	12	GND
GND	13	14	Write Data
/DS0	15	16	GND
/DS1	17	18	/Write Gate
/DS2	19	20	/Step
/DS3	21	22	/Side Sselect
/RDY0	23	24	/Direction
/RDY1	25	26	/Head Load
/RDY2	27	28	High-/Normal Density
/RDY3	29	30	/In Use

Při konstrukci kabelu se vřele doporučuje prokládat živé signály zemními vodiči. Přebytečné zemní vodiče se na konci kabelu u řadiče spojí drátem a propojí se zemními vodiči na řadiči. Na straně mechaniky jsou připraveny příslušné zemní vývody v plném počtu.

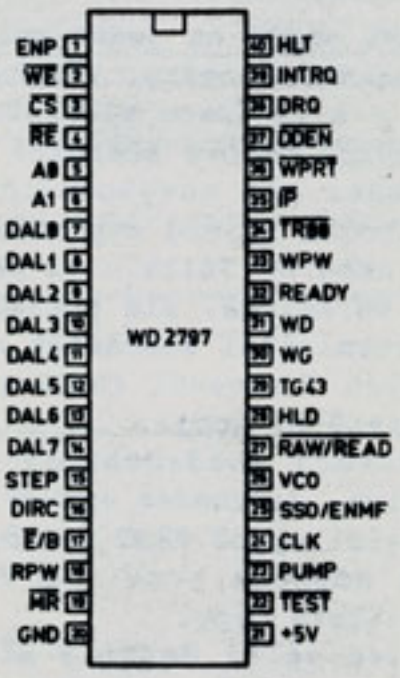
Obr.8 Zapojení řadiče floppy disku (verze pro připojení na (e) STD BUS)



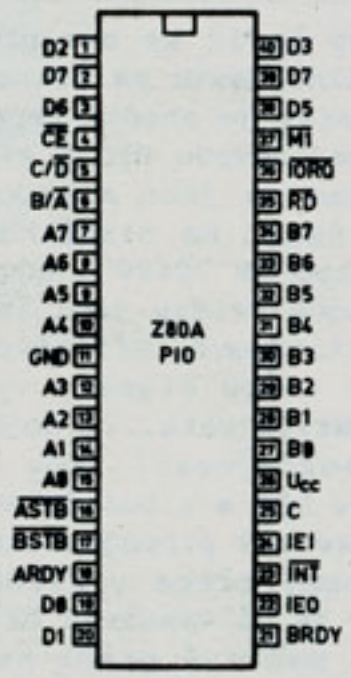
◀ Zapojení oscilátoru 50 Hz
Hodnoty C1, R1 je nutno volit podle typu použitých hradel



◀ Generátor 2/1 MHz



Obvod WD-2797



Obvod Z80A-PIO

3. Programová obsluha řadiče

V této fázi je třeba se seznámit s programovou obsluhou řadiče. Vlastní řadič zaujímá adresy 10H-13H, obvod PIO 14H-17H. Rozložení registrů je následující:

Adresa	Registr
10H	Control/Status Register
11H	Track Register
12H	Sector Register
13H	Data Register
14H	Port A Data
15H	Port B Data
16H	Port A Control
17H	Port B Control

Nejjednodušší kontrola správné funkce obvodu interfacu mezi řadičem a počítačem spočívá v postupném načtení všech výše uvedených adres instrukcí IN přímo z Basicu. Hodnoty, které bychom měli obdržet, jsou:

128,0,1,0 (svědčí o správné inicializaci řadiče);
63,207,143 (nebo i jiná hodnota vlivem nezapojených vstupů);
255,255 (řídicí registry nedávají při čtení data)

Jednotlivé bity bran obvodu PIO mají tento význam:

Brána A:

Bit	Směr	Signál	
0	OUT	DDEN	(přepínání hustoty)
1	OUT	H/N	(přepínání 5/8 nebo režimu dvourychlostních mechanik)
2	OUT	ENP	(povolení zápisové předkomp.)
3	OUT	IN USE	(potvrzení použití pro mech.)
4	OUT	SWITCH FILTER	(pro některé druhy mechanik)
5	OUT	UNSAFE RESET	(zrušení chyby mech.)
6	IN	DRQ	(žádost o data od řadiče)
7	IN	IRQ	(aktivuje přerušení)

Brána B:

Bit	Směr	Signál	
0	OUT	DS0	(výběr mechaniky)
1	OUT	DS1	
2	OUT	DS2	
3	OUT	DS3	
4	OUT	INTEX	(ovládání přerušení od ULA)
5	-		
6	IN	50 Hz	(generuje přerušení)
7	IN	READY	(mechanika připravena)

Po naprogamování obvodu PIO lze již zkoušet provedení selekce mechaniky, zda je schopna potvrdit připravenost atd. Je-li vše funkční, nic už nebrání vložení systémové diskety (prozatím chráněné proti zápisu) a pokusit se o BOOT.

Ladislav Sieger

JEŠTĚ JEDNOU
DIDAKTIK GAMA

Krátce před vánocemi 1987 se na našem trhu objevil nový počítač - Didaktik Gama. Jeho výrobce - výrobní družstvo Služba Bratislava, závod Didaktik Skalica - jej uvedl na trh v rekordně krátké době. Postaral se i o velmi dobře načasovanou televizní reklamu, která ústy ředitele závodu slibovala bohaté programové vybavení a řadu hardwarových doplňků s ohledem na kompatibilitu Didaktiku Gama se ZX Spectrem. Dále výrobce sliboval, že sám zajistí tvorbu nových programů.

Jak se později ukázalo, s kompatibilitou to není slavné, řada programů pro ZX Spectrum na Didaktiku nechodí a připojení standardních periférií Spectra je zhola nemožné.

Výrobce se k tomuto problému nikde nevyjádřil. Z návodu k obsluze není místy jasná ani obsluha počítače. Přitom je o něj na našem trhu velký zájem (taky nic jiného téměř není).

Dost důvodů pro to, abychom požádali Michala Bechyně z prodejny počítačů podniku Domácí potřeby Praha ve Spálené ulici, aby se s námi rozdělil o své zkušenosti:

Prospekt Didaktiku Gama o něm říká, že je kompatibilní se ZX Spectrem. Je rozšířen o některé doplňky, které si uživatelé ZX Spectra jinak museli opatřovat dodatečně - paralelní interface s 8255A a rozšířená paměť RAM 80K. Tyto doplňky a související úpravy však způsobily, že pro práci s Didaktikem nelze převzít kompletní software k původnímu ZX Spectru.

Ve svém příspěvku nejdříve popíšu samotný počítač a potom rozeberu příčiny neúplné kompatibility.

Klávesnice

První, co na uživatele udělá dojem, je bezesporu dobrá klávesnice Didaktiku. Jediným slabým místem je barevný popis kláves - nápisy se po delší době provozu smazávají. Výrobce je si tohoto nedostatku vědom a snaží se životnost popisu kláves prodloužit.

Funkce a rozmístění kláves odpovídá původnímu ZX Spectru s gumovou klávesnicí. Zůstala i často diskutovaná funkce klíčových slov, která je hlavním problémem pro začátečníky. Proto je nutné se dobře seznámit s obsluhou, což - vedle návodu - usnadňuje i program na přiložené kazetě.

Zapojení počítače, jeho naladění a načtení přiloženého programu

Didaktik Gama se dodává v jedné krabici, kde je vlastní počítač, napájecí zdroj, kabel pro připojení k televizoru, návod a kazeta s úvodním programem. Po rozbalení počítače je nutno dodržet následující postup:

1. Spojit počítač s televizorem
2. Zapojit napájecí zdroj do sítě
3. Konektor pro nahrávání zapojit do magnetofonu
4. Zasunout napájecí konektor do počítače, přičemž se rozsvítí (většinou zelená) dioda LED signalizující zapnutí
5. Naladit televizor

Tento postup je třeba dodržet proto, aby se zdroj ustálil a do počítače se nedostala napěťová špička, která by jej mohla poškodit. Pro napájení je použit pětikolíkový konektor DIN, který je současně využit i pro spojení s magnetofonem. U prvních sérií se signál z počítače ladil v prvním a ve třetím TV pásmu. Vzhledem ke špatné kvalitě obrazu začal výrobce do počítače vestavovat UHF modulátor se signálem ve IV. TV pásmu. Pro ty, kteří si Didaktik zakoupili ještě bez tohoto modulátoru, jej výrobce zdarma namontuje.

Na kazetě s úvodním programem je napsán příkaz pro načtení programu do paměti počítače. Bohužel u příkazu LOAD "" tam není uvedeno, že za každým příkazem je třeba stisknout klávesu ENTER. Popis na kazetě předpokládá, že uživatel již prostudoval návod. Jenže ten je v tomto místě stručný a vychází naopak z předpokladu, že se uživateli podařilo nahrát úvodní kazetu a seznámit se s obsluhou klávesnice. Trochu to připomíná pohádku o kohoutkovi a slepičce. Na kazetě nebo na začátku návodu by bylo mnohem vhodnější uvést text:

Nahrání úvodního programu: Stiskněte klávesu "J". Na obrazovce se vypíše příkaz LOAD. Pak podržte klávesu SYMBOL SHIFT a dvakrát stlačte klávesu "P". Na obrazovce se vypíší dvoje uvozovky. Nakonec stiskněte klávesu ENTER.

Pokročilejší tuto pasáž automaticky přeskočí, ale začátečníci ji ocení, neboť si ušetří pocit, že počítač není v pořádku. Skutečně se už stalo, že do prodejny přišel úplný začátečník s reklamací, že počítač nenahrává. Důvodem byla právě neznalost obsluhy klávesnice a příkazu ENTER.

Vstupy a výstupy počítače

Již jsem se zmínil o napájecím konektoru, který je společný i pro nahrávání programů. Při pohledu na počítač zezadu jsou vlevo od napájecího konektoru umístěny další dva konektory pro periférie. První je systémový. Je to hranový konektor s vyvedenými signály procesoru a adresové i datové sběrnice. Jejich rozmístění je stejné jako u ZX Spectra. Vzhledem k jiné konstrukci Didaktiku však není vyvedeno napájení +9, -12 a -5 voltů, schází výstup video a rozdílové signály barvy. Na druhém konektoru (typ FRB 30) jsou vývody vestavěného interfacu s obvodem 8255A.

Paměť RAM a její stránkování

ZX Spectrum je obvykle osazeno vadnými paměťmi RAM 64K, u nichž lze využít jen spodní polovinu. V Didaktiku jsou bezchybné paměti 64K. Protože použitý osmibitový procesor může přímo adresovat pouze 64K paměti, z čehož ROM zabírá 16K, zbývá pro uživatelskou RAMku pouze 48K. Proto jsou na horních 32K paměťového prostoru střídavě připojovány obě poloviny pamětí 64K - tzv. stránky, které se mohou mezi sebou přepínat (vyměňovat). To se provádí přepínáním logických úrovní na adresových vodičích A15 v multiplexerech použitých pamětí.

Toto uspořádání paměti má však své nevýhody. Po zapnutí počítače je totiž nastaven RAMTOP na 65367, tedy do přepínané části paměti. Pod RAMTOP je umístěn zásobník návratových adres procesoru. V případě prostého přepnutí stránky se celý systém zhroutí. Výrobce se snažil situaci řešit tím, že před přepnutím si systém kopíroval obsah první stránky do druhé, a to od počátku zásobníku až do konce paměti RAM. To znemožňovalo využití druhé stránky pro další strojové programy, resp. data. Využití pro Basic bylo dost pochybené, protože pokud program či proměnné přesáhly adresu 32767, došlo po přepnutí k jejich ztrátě nebo zkomolení. To mělo za následek chybnou funkci programu, případně jeho zhroucení. Proto byl od série, která přišla do prodeje v únoru 1988, upraven režim stránkování do podoby, která je sice použitelná, ale vůbec neodpovídá popisu v návodu k obsluze (ten je i tak dost nejasný). Horní část paměti až po RAMTOP je systémem kopírována do obou stránek, takže výsledek je takový, jako by se stránkovalo jen nad RAMTOP. Oblast nad ním je pak bez omezení využitelná pro programy ve strojovém kódu a pro data. (Co se děje s UDG? - pozn. red.)

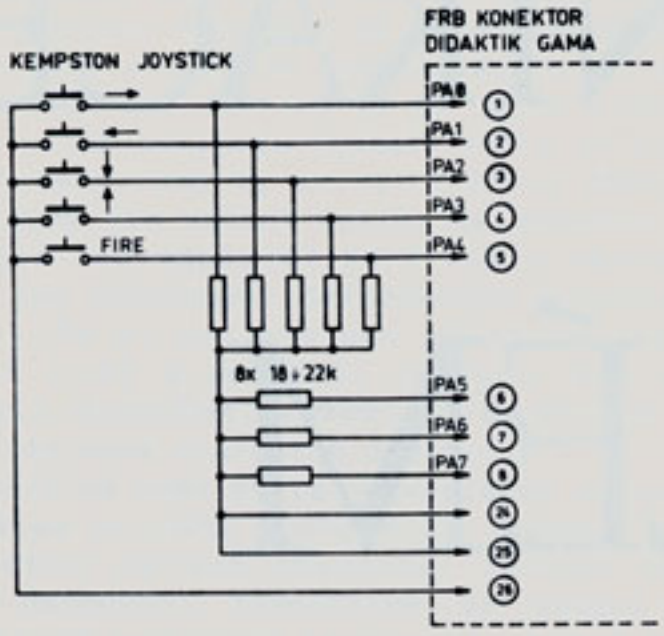
Pro přepínání stránek je použit bit 0 z portu C vestavěného interfacu, jehož řídicí registr je na výstupní adrese 127, která se používá i pro přepínání. Přepnutí se provede příkazy OUT 127,1 a zpět OUT 127,0. To způsobí přímé nastavení, resp. vynulování bitu 0 na PC. K předefinování bran portu přitom nedojde, protože v příkazu není nastaven bit 7 (číslo je menší než 128). Monitor v paměti RAM se již postará o přenesení zásobníku. Jinou možností stránkování RAM je běžné ovládání portu C na adrese 95. Na to je třeba dávat pozor při obsluze periférií - i v případě ovládání přes výstupní adresu 95 je monitorem přenášen zásobník návratových adres.

Vestavěný interface s MHB 8255A

Jde vlastně o klasicky zapojený programovatelný interface MHB 8255A, jehož schéma a popis zapojení je např. v Amatérském radiu A7/87. Toto zapojení má několik výhod i nevýhod. Jeho hlavní nevýhoda je v tom, že k počítači už nemůžeme připojit jiné zařízení, které používá stejnou adresu (např. interface pro připojení joysticků). Další nevýhodou je použití bitu PC0 pro přepínání stránek paměti. Tak nelze použít software, který má pro komunikaci nadefinován port C. To se týká hlavně programů pro tiskárny BT 100 a GAMACENTRUM, které se pro práci s Didaktikem musejí upravit, což je pro začátečníka nepřekonatelný problém.

Výhody však převyšují nevýhody. Paralelní interface je zařízení natolik užitečné, že si je většina majitelů Spectra tak či onak opatřuje hlavně pro připojení jakékoli tiskárny s paralelním vstupem. V Didaktiku je vestavěn právě interface pro spojení s tímto typem tiskáren. Obslužný program je implementován v paměti ROM a je nadefinován jako hlavní. Pokud bychom chtěli použít ZX Printer, stačí změnit adresy v systémových proměnných podle návodu. Při připojování ostatních zařízení je nutno dbát na zachování portu C, nebo alespoň jeho bitu 0. Tento bit musí být vždy výstupní a nesmí být ovlivněna jeho logická úroveň. Různé obslužné programy jsou většinou psány ve strojovém kódu procesoru. Je třeba zkontrolovat, zda nepracují s portem 95. Mohlo by se stát, že by došlo k přestránkování paměti bez předchozího překopírování zásobníku. Tak by se registr PC nastavil na náhodnou hodnotu s následným krachem programu. Pokud bude tato náhodná hodnota 0000H nebo FFFFH, bude výsledek stejný jako při RESETu počítače - dojde k vymazání paměti.

Vestavěný interface umožňuje připojit jeden joystick. Ten se připojuje k portu A obvodu 8255A, který je po zapnutí počítače inicializován jako vstupní. Protože je port A na adrese 31, můžeme jej použít jako KEMPSTON INTERFACE, což umožní ovládání naprosté většiny her. Schéma připojení je na obr.1.



Obr.1 Připojení joysticku k Didaktiku

Systémový konektor

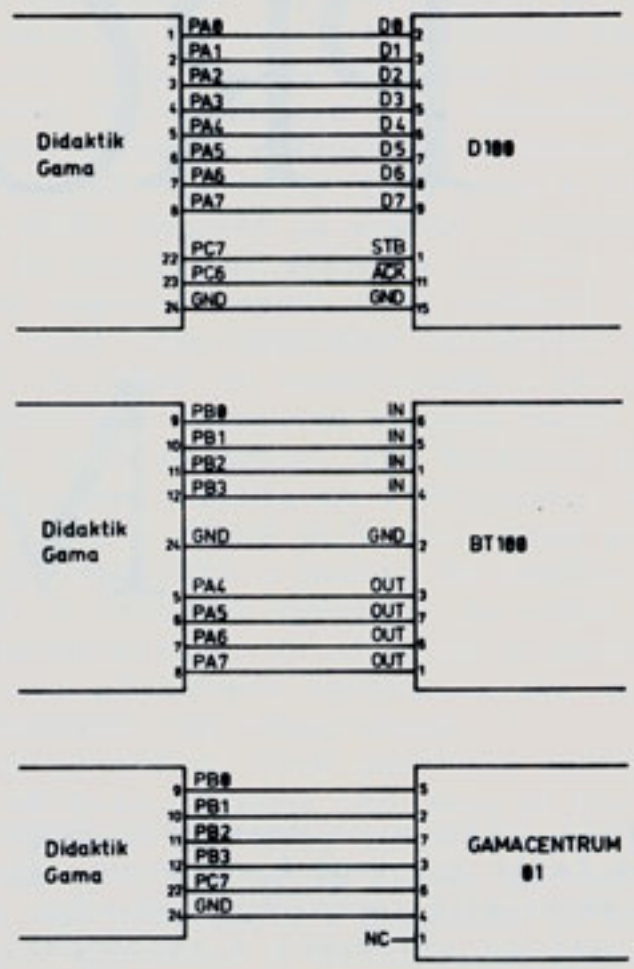
je shodný s odpovídajícím konektorem u ZX Spectra - až na chybějící napájení +9V, -5V a -12V, signály video a rozdílové signály barvy. Z toho opět vyplývá, že řada standardních periférií ZX Spectra nebude ve spojení s Didaktikem fungovat. Na takto uvolněné kontakty mohl být vyveden aspoň CS obvodu 8255, který by byl směrem k dekodéru adresy oddělen odporem 680 ohmů. To proto, aby se dal celý vestavěný interface zablokovat a pak připojit např. interface pro dva joysticky, z nichž jeden je adresován jako Kempston.

Největším problémem při rozšiřování systému však asi bude signál /ROMCS, který (na rozdíl od ZX Spectra) není od obvodu ULA oddělen odporem. Proto POZOR při zapojování vnějších pamětí. Může dojít ke kolizi na sběrnici a případně i k poškození počítače! To se stane například při pokusu o připojení zařízení ZX Interface 1 a ZX Interface 2!

Programování obvodu 8255A není v návodu popsáno. Snad by tam mohly být alespoň odkazy na literaturu - např. na AR, v jehož ročníku 1983 bylo popsáno nejen programování obvodu, ale je tam i popis procesoru a dalších obvodů. Potřebné informace přinesla i Sdělovací technika, Mikrobáze a další, jenže o tom návod mlčí (ať si chudák začátečník poradí, jak umí).

Kompatibilita s počítačem ZX Spectrum

Problémy kolem slučitelnosti počítačů Didaktik Gama a ZX Spectra vyplývají z předchozího textu, takže jen malé doplnění. Didaktik Gama má v podstatě stejnou konstrukci jako jeho vzor ZX Spectrum. Využívá i obvod ULA, který výrobce Didaktiku dováží. Z větší části zůstal zachován obsah paměti ROM, kde byly provedeny jen menší změny. Např. byla opravena rutina pro obsluhu NMI, byly doplněny rutiny pro obsluhu přepínání stránek paměti a obsluhu tiskárny přes Centronics atd. Všechny změny byly podrobně popsány v časopise Elektronika 8/88. Změna byla provedena i ve vstupu z magnetofonu (EAR) - zařazení zesilovače umožňuje použít jeden pětikolíkový konektor DIN pro zápis i čtení programů. To je výhodné pro magnetofony SP 210 (SP 210 T) nebo minivěž T 710. Ne však pro přenosné magnetofony menších rozměrů (někdy ovšem větších kvalit), jako je třeba toho času jediný na našem trhu dostupný magnetofon ELTA. Ten je sice pro Didaktik vhodný, ale má jisté nedostatky. Předně při převíjení někdy dochází k poškození programů. Tuto nectnost odstraňuje pražská záruční opravna magnetofonů v Pařížské ulici č.17. Druhou vadou, která se projeví právě ve spojení mgf ELTA s Didaktikem a kterou je možno odstranit až po uplynutí záruční lhůty, je fakt, že při použití výstupu DIN nedojde k vypnutí vestavěného reproduktoru. Tento výstup je bohužel závislý na poloze regulátoru hlasitosti, který musí být při načítání programů v poloze MAX. Jak nepříjemný hluk přitom vyslechne obsluha a její okolí, není třeba líčit. Reproduktor nelze vypnout ani pomocí zápalky, zasunuté do jacku s označením EAR, protože tím se zároveň přeruší signál do konektoru DIN.



Obr.2 Připojení tří typů tiskáren k Didaktiku

Připojování tiskáren

Dnes je již vyřešeno připojení čtyř nejdostupnějších tiskáren - BT 100 (SP 210 T), GAMACENTRUM, libovolné tiskárny s paralelním vstupem (např. D100) a ZX Printeru. Na rozdíl od prvních dvou typů, třetí nevyžaduje žádný podpůrný software. Ten je již uložen v upravené části ROM. Totéž platí pro ZX Printer - jen musíme změnit obsah příslušných systémových proměnných. Rutina pro ZX Printer může být využita i pro BT 100, pro niž byl k tomu účelu zkonstruován zvláštní interface. Výhodné je, že není třeba ukládat příslušnou rutinu do RAMky. Nevýhodou je pouze jednostranný tisk, který dále zpomalí už tak pomalou práci této tiskárny.

Pro tiskárny BT 100 a GAMACENTRUM ve spojení s Didaktikem existuje poměrně kvalitní obslužný program s funkcemi LLIST, LPRINT a COPY. Příkazem COPY lze tisknout ve dvou velikostech. Dále můžeme nadefinovat různé velikosti písma a u BT 100 i vlastní sadu znaků.

Připojení tiskáren je na obr.2. Připojení ZX Printeru není třeba uvádět, protože se zapojuje přímo na systémový konektor.

Michal Bechyně

Pozn.red.: Pokud je nám známo, žádné originální programy pro tento počítač na trhu nejsou. Před vánocemi byly dovezeny (zřejmě výprodejní) sady čtyř her pro původní ZX Spectrum. Na vánoční trh se však nedostaly, protože nebyla včas stanovena jejich cena. Nabídky tuzemských autorů ke spolupráci při tvorbě originálních československých programů a při úpravě vlastního počítače výrobce tvrdošíjně odmítá, případně odkládá jejich projednání na neurčito. Proč?

METAKOMUNIKACE

PROBLÉM MÍRY

Stojíme-li před problémem programátorským, můžeme jej řešit jen vhodně zvolenými algoritmy. Pominu-li materiálně technickou základnu, pak nic než míra našich vědomostí a schopností nám v cestě nestojí. Představte si však, že pro řešení jakéhokoli problému dostanete soupis závazných předpisů, vymezujících, jaké, kdy a jakou měrou smíte ty či ony algoritmy použít. Třeba že pro třídění souboru dat jmenného seznamu nesmíte použít jiný algoritmus než bubble sort. Usmíváte se? Jistě, s takovým modelem bychom se v programování nikam nedostali. A přesto v praktickém životě podobné modely "fungují". A všichni se chováme, jako že je to úplně v pořádku.

V rozvíjení myšlenky zůstanu na půdě Mikrobáze. Základním kamenem úrazu redakční práce je získání kvalitních materiálů. Ty mohou napsat jen fundovaní odborníci a praktici, kteří 1/ umějí psát a 2/ umějí psát tak, aby jim ostatní lidé rozuměli. Zkusil jsem kontakt s některými z nich. Prvotní zájem byl nakonec vždy zmrazen informací o výši honoráře. Podle svazarmovských předpisů je to maximálně 45 Kčs za standardní stránku strojopisu. Napsat hutný, dobře sestavený soft/hardwarový materiál, který často zasahuje i do řady dalších oborů, není jednoduché. Za příspěvek s rozsahem 6-10 stran dostane autor 270-450 Kčs brutto. Od toho se mu ještě odečtou daně. Tak autor za obecně přínosnou informaci, ve které se snoubí jeho talent, zkušenosti, um, vědomosti a píle, dostane vyměřenu almužničku, jakou by si rychleji a bez jakékoli námahy vyseděl u kostela s kloboukem v ruce.

Po déletrvajícím naléhání mě jeden programátor odbyl slovy: Jsou lidé, kteří dělají, a lidé, kteří píší. Snad nevědomky dost přesně vystihl zdejší situaci. Zkusil jsem zachytit drápkem v pražském matfyzu. Matematicky řečeno: nula od nuly... Jiný "péčéčkový" tvůrce mi po sdělení výše honoráře řekl: Schéma mám, musel bych k němu napsat tak asi tři stránky...ale to by byla ztráta času. Měl pravdu. Protože za obrázek mu může Mikrobáze dát dokonce jen pouhých 15 korun! Za celý materiál, kterému věnoval možná měsíc dřiny, by dostal 150 Kčs brutto! Kde je tu vůbec sebemenší náznak jakékoli míry?

Když jsem se dozvěděl, že autoři článků v Amatérském radiu dostávají za stránku v průměru jen o 5 Kčs víc než v Mikrobázi, na chvilku jsem zapomněl dýchat. Představa, že autor nějakého superzapojení, nad nímž strávil půl roku života, dostane za jeho otisknutí tři tisíce, je šokující. Hledal jsem porovnání. Tak třeba za reportáž, rozhovor nebo jakýkoli článek z jakékoli netechnické oblasti se jinde platí v průměru stovka (jak kde). Má-li autor určité zkušenosti a schopnosti, 10 stránek za 1000 korun napíše za 1-2 dny. Dá-li si s tím práci, zabere mu to maximálně týden. A jsem přesně tam, kde to vůbec nemá být ("Mistře, jak dlouho malujete jeden čtvereční centimetr plátna?"). Jenže obecně to tam bohužel je - tabulky, vyhlášky, vymezení, omezení, administrativní kličky a háčky... Tyto "rozkošné" miniatury anonymních mistrů druhé poloviny 20. století táhnou rozmach umělců budoucnosti někam do časů minulých. Aneb stokrát nic umořilo... Stará, ale všemi bůhvíproč stále uctívaná finta byrokratů.

Podívejme se na věc ještě z pohledu finančních nákladů vydání knihy či časopisu. U nás autorský honorář činí jejich nejzanedbatelnější část. A je celkem jedno, jestli se stejný honorář vyplatí někomu, jehož články či knihy nikoho nijak zvlášť nezajímají, či tomu, na jehož tvorbu se čeká ve frontě. Prašť jako uhoď - snad i proto se obecně potřebné a hledané knížky vydávají v naprosto nedostatečném nákladu, zatímco jasné ležáky si přijdou na své. Vydavatelství má kvantitativní ukazatele plánu, který se nakonec vždy nějak splní. Kvalitativně jej v podstatě nemá co ovlivnit. Pár přeložených detektivek to v nejhorším "nažene". Žádná zpětná vazba, přerod kvantity v kvalitu se nekoná. A v redakčním skleníku je tak útulně a teploučko...

Potenciálně dobrý autor přece nebude vedle vší své práce psát knihu za dvacet tisíc, když ani dopředu neví, zda mu ji vydavatelství vytiskne. To tady totiž většinou nehledá autory, ale čeká, co odkud spadne hotového. A pak to buď vydá nebo ne. Když ne, může si to autor strčit za klobouk (jeho chyba, že tomu věnoval svůj čas).

A tak potenciálně dobrý autor bude radši dělat cokoli jiného, než psát knihu, na kterou 1/ nemá

smlouvu a 2/ i kdyby ji měl, nestojí mu to ani za ten honorář, ani za ty nervy. Autor obvykle dostane třetinu honoráře až po dlouhosáhlém, někdy povahu věci mijejícím rozhodování o přijetí knihy a zbývající dvě třetiny až po jejím vydání, které nemůže nijak ovlivnit (vydací lhůty se pohybují od 2 do 5 let). Jak by se asi tvářili dělníci loděnic, kdyby jim vedení řeklo, že dostanou mzdu, až budovaná loď odpluje? Když tohle všechno autor přece jen podstoupí, jednoduchým propočtem snadno zjistí, že si na jeho umu přijde na své spousta jiných lidí a podniků, zatímco jemu z koláče odklepli drobeček, rozemletý na dlouhá léta dopředu (dojemná starost o dědice?).

Ve většině zemí světa je výsledek autorské práce předmětem volného obchodního jednání. Vydavatel ani autor nejsou omezováni žádnými tabulkami (kromě daňových). Samozřejmě, že jednání probíhají na základě určitých zvyklostí, které však jsou obrazem momentální (proměnlivé) situace společenské poptávky. Uzavíraná smlouva může nabýt mnoha podob. Od jednorázového honoráře po tantiémy, včetně všech možných kombinací. Vydavatelé mají šanci udržet se ve své činnosti jen tehdy, když získají dobré autory, kteří budou psát, co společnost právě teď potřebuje. Dobří autoři nepucují kliky vydavatelství, ale vydavatelé je sami vyhledávají a jednají s nimi o tom, co a za jakých podmínek by mohli napsat.

To se týká nejen knižní, ale i časopisecké produkce. Je pravdou, že i zahraniční časopisy mívají pro běžné příspěvky vymezenu určitou výši honoráře. Ovšem s běžnými příspěvky by žádná redakce dlouho nepřežila. Udrží se jen tehdy, když dokáže uspokojit potřeby svých adresátů. To vyžaduje nejen obchodní a organizační zdatnost. Redakce musí sama moc dobře vědět, co má tisknout, a pro realizaci svých předsevzetí musí vytvořit odpovídající podmínky. Tomu, co dělá, se bude moci věnovat i příští týden jen za předpokladu, že se ve své práci opravdu vyzná. To se zdaleka netýká jen problematiky interního provozu. Redakce, která by nebyla společensky interaktivní a reflexivní, by musela brzy stáhnout roletu.

Ve většině zemí světa si autor vydáním úspěšné, tedy i potřebné knihy zajistí klidné zázemí pro další tvorbu na mnoho let dopředu. Odborník si za jeden obsáhlejší článek v časopise s vyšším nákladem koupí třeba IBM AT, což mu pomůže v další prospěšné práci. Takovou automatiku adekvátní reflexe hodnot a funkcí vidím jako nadmíru spravedlivě vyřešený problém míry.

U nás ten, kdo by tiskem mohl předat druhým, co umí, na co ostatní tolik čekají a co tolik potřebují, aby se hnuli kupředu, se za stávajících podmínek bude sakra rozmýšlet, jestli se tím vůbec má zabývat. Kdo na to doplácí, je zřejmé gramotnost celé společnosti. Kdo tím získává, je byrokracie, kterou každý společenský pohyb kupředu zalévá studeným potem. Proto se vnucuje svými betonovými tabulkami a vyhláškami, na kterých květ dynamiky všestranného rozvoje nevyroste. Betonové piedestaly byrokratů jsou i hlubokým podceněním a devalvací dispozic společnosti, která má na to, aby si dovedla řídit své věci sama. Vždyť ani ta volně podnikající redakce kdekoli na světě si nemůže dovolit vydat na honorářích víc, než právě může. Kolik může vydat, neurčují žádné monolitické tabulky, ale úroveň celé společnosti jakožto aktuální obraz její skutečné míry. Umělé vsazení nízkých stropů by přeťalo všechny souvztažnosti seberegulace, její všesměrový informační tok. Obraz by rychle oněměl, protože informace je jazykem komplexní evoluce.

Nejednou jsem slyšel cosi o tom, že zájmová a profesionální sféra jsou dvě zcela odlišné oblasti. To je tvrzení značně prostoduché. Každá zájmová sféra je potenciálním zřídlem budoucích profesionálů i základnou stálého zvyšování gramotnosti a kulturnosti celého národa. Podle toho je k ní třeba přistupovat. A kdo jiný by měl působit na zvyšování její kvality, než právě profesionálové, kteří se svému oboru věnují každý den? A jsem zase tam, odkud jsem začal. U Mikrobáze a jejích 45 korun za stránku. U problému míry a u betonových stropů.

Mám sen - pro každé číslo Mikrobáze dostaneme určitou honorářovou sumu, tedy tolik, kolik se vydavatel rozhodne investovat. Renomovanému autorovi přínosného, podnětného příspěvku dáme třeba 250 Kčs za stránku, na standardní příspěvek 90 Kčs. Vydavatel by nezchudl a úroveň časopisu by šla nahoru. Díky tomu by vydavatel vydělával stále víc a mohl by dávat víc i na honoráře v přímé závislosti na svých možnostech. A nahoru by šla i úroveň obecná. Je tohle snad nesplnitelný sen? Ptám se PROČ? Odpověď je snadná - protože všichni děláme, že cokoli, co spadne s jakékoli "hůry", je nehybným znamením osudu mimo veškerý náš vliv. Tváří v tvář tomu vypínáme racionální myšlení a jednání. A ještě se handrkujeme o pár korun tam někde dole pod betonovým stropem (Když jste Einsteinovi dali 45 korun, tak Novákovi musíte dát jen 3,50). Ne, takhle to v Mikrobázi není. Ale leckde jinde v různé míře ano. Snad proto, aby redaktoři nepřišli o pocit, že poměřují, ještě ukrajují z toho, co vydavatel ve svých nákladech ani nezaregistruje. To už nejde pojmenovat jinak než mírou pěstované ubohosti. Jak co do míry oněch honorářů, tak všeho kolem.

Kam až to došlo, živě pociťuje každý, kdo se chce vzdělávat. Zrovna teď jsem se chtěl naučit Céčko. Jenže - kde nic, tu nic, není z čeho se učit, kam jít, kde by se člověk dozvěděl... Opět nezbylo, než přistoupit k obvyklému (a už proklatě otravnému) samoučení z takřka nedostupných anglicky a rusky psaných materiálů. Ani omylem jsem nezkusil obrátit se na odborníka, aby napsal seriál o Céčku do Mikrobáze (smích v sále). Pevně rozhodnut nepřestat, dokud Céčko nezvládnu, jsem dostal nápad zkombinovat svou sebevýuku s jejím popisem v Mikrobázi. Ať z toho mají něco i ostatní postižení, leč vzdělání stále ještě chtiví (komenští sandokanové). Je to do jisté míry z nouze ctnost. Asi v té míře, v jaké jsme si tu tradicionalisticky zvykli být rádi, že jsme rádi (Hlavně to zdravíčko, to je to nejdůležitější, pane Čech...).

Všeho s mírou, říkali staří Řekové. Z této moudrosti jsme si omylem vzali jen jednu polovinu - že špatné je, co je přes míru (přejídání, nemírné používání toho či onoho apod.). Zcela však pomíjíme tu druhou, neméně významnou polovinu. Špatné je obojí - předimenzování i poddimenzování, přeceňování i podceňování, přepětí i podpětí. Ani v jednom případě nebude nic fungovat správně. Přepínaný subjekt je na cestě k rychlému zmaru. Při podpětí skomírá - je-li to stroj, pak není k ničemu, je-li to organismus, čeká ho zase jen zmar - jen cesta k němu je (podle okolností) delší. To v sobě chová jednu naději - dokud je čas, lze z této necesty vykročit vstříc míře věcí. Každý organismus má toleranci míry své existence. Uvnitř rozptylu jejích hodnot se vyznačuje schopností seberegulace. Hodnoty mimo životné hranice nabývají extrémní povahy. To se týká každé organické struktury. Obecně přírody (ekologie). I společenského života, jehož jednou složkou je informatika. Do ní spadá i míra funkce vydavatelské činnosti.

Široce rozepjala svá křídla věta, vyjadřující iracionalitu zahnízděného poměřování: Ze strachu, aby snad někdo nezbohatl, budeme chudnout všichni. Co dodat? Snad že - jak o tom svědčí výstupy vydavatelství - nemírného rozporu všeho podpětí s přepětím bylo už dost (špatná práce stojí druhé spoustu vypětí). A že je načase, abychom vyšli vstříc míře věcí i v této sféře. Ku prospěchu nás všech.

-elzet-

Z DOMOVA

* První série joystické pro ZX Spectrum z Kovodružstva Náchod někdy trpěly předčasnou únavou materiálu kontaktních pružin. Změnou použitého materiálu nyní výrobce dosáhl podstatného zvýšení spolehlivosti. (Meca)

* Chystané vánoční překvapení Kovodružstva Náchod pro majitele Didaktiku Gama se nekonalo. Celá první várka **joystické pro Didaktik** nefungovala. Příčinou bylo nevhodné propojení konektorů, zřejmě zaviněné naprostou absencí jakékoli technické dokumentace k počítači. (Meca)

* **Kazetové magnetofony ELTA**, nabízené na našem trhu a často kupované k počítačům, mívají velmi nepříjemnou závadu. Při převíjení někdy odmažou nepatrný kousek záznamu. Při hudbě to tak nevadí, ale počítačový program je tím ztracen. Prý je to způsobeno nabíjením či vybíjením kondenzátoru v obvodu mazací hlavy. Tuto závadu odstraňuje (i v expresním termínu) záruční opravna magnetofonů-Kovoslužba Praha, Pařížská 17. (Meca)

* Tesla Kolín myslí ma početnou rodinu sinclairistů. Pod tajemným kódovým číslem 5QK06741 jim připravila pěkný dárek. Vzhledem k obvyklým zdlouhavým cenovým jednáním to asi bude spíš k velikonocům, než k vánocům, ale to nic nemění na podstatě věci - vytoužený **univerzální paralelní interface UR-4** je na světě. Je určen **všem variantám ZX Spectra**, s vyjímkou Didaktiku Gama, který nemá na systémovém konektoru potřebné napájení +9 V. UR-4 je osazen programovatelným obvodem MHB 8255A a umožňuje připojení joysticku (konektor CANNON 9p), případně tiskárny a dalších doplňků (konektor FRB). Celý výrobek je velmi pěkně proveden. O péči, kterou mu výrobce věnoval, svědčí i to, že jej před uvedením na trh nechal otestovat řadou uživatelů. (Meca)

* V rámci pražského Sinclair klubu 602. ZO vznikla **sekce počítačů pracujících s operačním systémem CP/M**. Protože s ním může pracovat řada počítačů různých značek, jsou všichni jejich uživatelé na úterních schůzkách Pod Juliskou (vždy v lichém týdnu) vítáni. CP/M je výrazným jednotícím prvkem nejen mezi jednotlivými typy počítačů, ale i mezi amatéry a profesionály. Proto se Mikrobáze chystá věnovat "cépéemce" pravidelnou rubriku. (Meca)

* Minulé číslo Mikrobáze informovalo o připravované službě Kovoslužby Praha - **rozšíření paměti ZX Spectra na 272K podle AR 9/88**. Po několika dnech jsme se dozvěděli, že Kovoslužba připravuje ještě jednu variantu (skoro) téhož - **rozšíření paměti ZX Spectra na 80K podle ST 11/87**. Tuto variantu popisuje i Mikrobáze v seriálu Rozšíření paměti ZX Spectra. První variantu nabízí 602.ZO jako službu svým členům, ovšem bez montáže i bez integrovaných obvodů. Pochopitelně, že Kovoslužba bude provádět celkovou montáž zapojení, takže nepájející majitelé počítačů si domů přinesou ZX Spectrum nejen s rozšířenou pamětí, ale i se zárukou. Každou z variant vymyslela jiná skupina tvůrců. I když se obě obdoby v detailech a aplikačním zaměření mírně liší, jsou převážně kompatibilní. Běžným uživatelům by nastávající dilema volby mělo být patřičně vyjasněno. Určitě dobrý důvod pro to, aby Mikrobáze svedla obě tvůrčí skupiny k jednomu stolu a otiskla záznam jejich "dišputace" (už se na tom pracuje).

(-elzet-)

Středisko
vědeckotechnických informací Svazarmu
pro elektroniku
Martinská 5, 110 00 Praha 1

Služby střediska

Jsou poskytovány pouze osobně. Vyřizování členství a hostování v 602.ZO Svazarmu, přístup ke knihovně časopisů na mikrofiších, pořizování ozalitových kopií z knihovny časopisů, prodej programových produktů Mikrobáze, zpravodaje Mikrobáze, nepájivých kontaktních polí, zpravodaje střediska MONITOR a poskytování informací o odborných akcích Svazarmu.

Pracovní doba

pondělí	**zavřeno**	
úterý až čtvrtek	**10 - 12**	**14 - 17**
pátek	**10 - 12**	**14 - 16**

>>>>>>> telefon 22 87 74 <<<<<<<

ERROR

V Mikrobázi č. 7 a 8 loňského roku se stala nemilá chyba při montáži assemblerových výpisů. Věříme, že znalci se v přehozených částech programu rychle orientovali. V č.7 se chyba týká výpisu ovladače tiskárny MIREK. Díky tomu, že výpis má číslované řádky, orientace je snadná. Druhým postiženým je výpis ke článku ZX Spectrum tester v č.8. Test 6 na str.25 (Důsledný test RAM) správně pokračuje pravým sloupcem na straně 24. Na přípravu posledních čtyř čísel roku 1988 redakce dostala dvouměsíční ultimatum. To si vynutilo i změny v obsazení redakce. Expresní lhůty přípravy Mikrobáze přinutily přetíženou redakci k vynechání kontroly montáže časopisu před tiskem (kontrola byla prováděna až k montáži). Nová grafička časopisu, coby počítačový laik, bohužel nepostřehla, že uvedené výpisy při montáži přehodila. Autorům i čtenářům se velice omlouváme. Po dokončené stabilizaci obsazení nové redakce a mírném opadnutí časového stressu prochází Mikrobáze od svého 1. letošního čísla kontrolou i po montáži před tiskem.

PŘEHLED POČÍTAČŮ TYPU PC

Po tiskárnách, kterým jsme se věnovali v minulých dvou číslech Mikrobáze, budeme pokračovat přehledem počítačů kompatibilních s IBM PC a jeho klony, jak je nabízel britský trh v době loňských letních prázdnin. Vzhledem k jejich velkému počtu jsme tabulku rozdělili na několik částí. V příštím čísle najdete počítače typu AT, v dalším AT s procesorem 80386 a nakonec laptopy a jiné.

Použité zkratky:

zLstg - základní cena bez daně
uP - mikroprocesor a jeho kmitočet
Paměť - standardní paměť
Diskj - vestavěné diskové jednotky
+flop - doplnitelné floppy jednotky
+HD - doplnitelný hard disk
Her - monitor Hercules a cena
CGA, EGA - typy monitorů a cena
Mo - monochromatický monitor

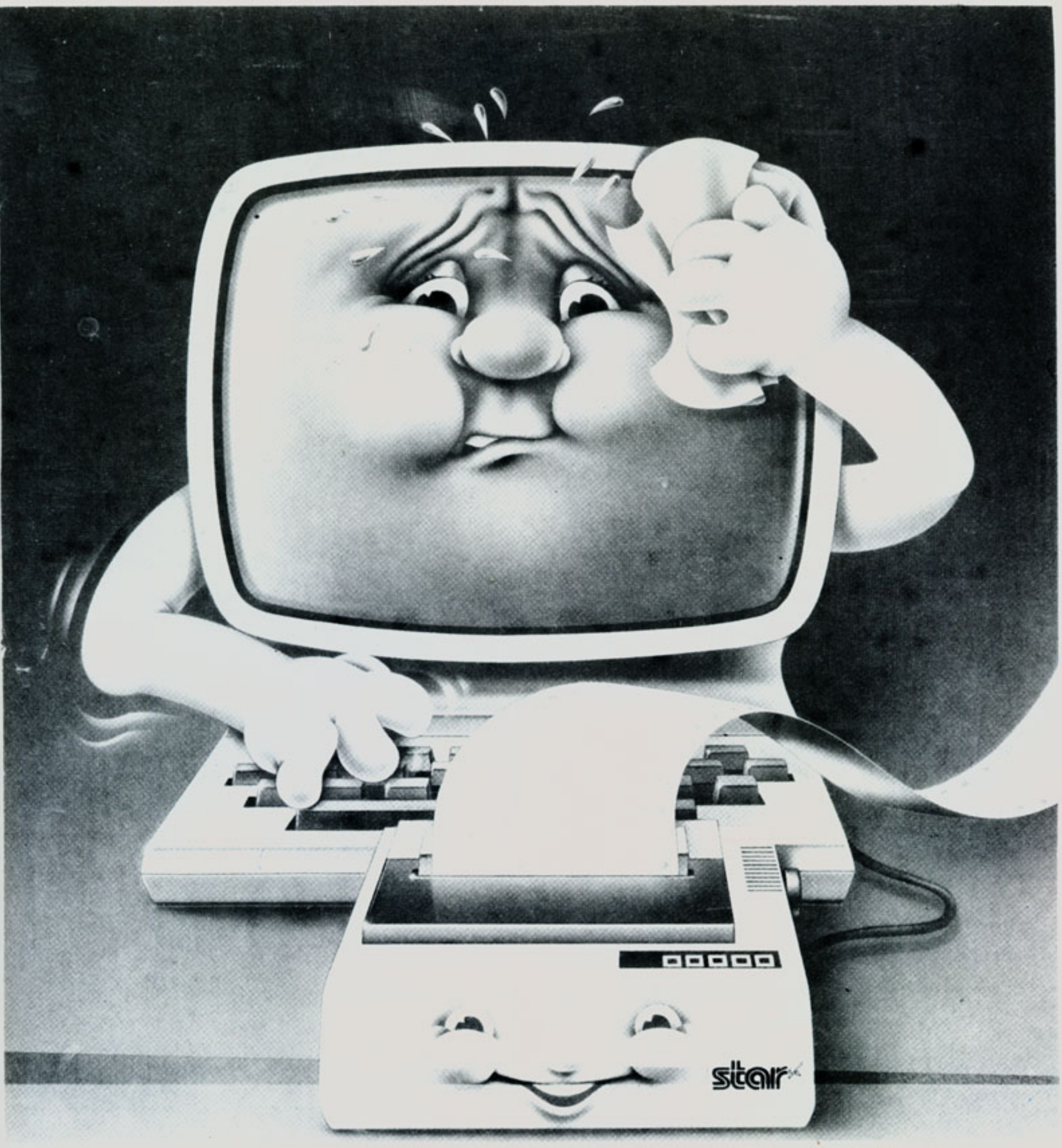
P - paralelní interface (Centronics)
S - sériový interface (RS232)
E - (expansion) konektory pro rozšíření
tLstg - typická prodejní cena:
v horní řádce cena základní konfigurace,
v dolní cena plné konfigurace
\+ - číslo za "+" je cena v Lstg;
když za "+" není číslo, je možno přikoupit
výrobek jiných producentů (různé ceny)
* - výrobek je součástí nabízené konfigurace

Klony IBM PC XT

Název počítače	zLstg	uP	Paměť	Disk	+flop	+HD	Her	CGA	EGA	P	S	E	tLstg
ACER PC500 Plus	+539	8088 4,77/8	640K	1x360K	2x360K	32M	*		+400	2	1		+625 +799
ACER PC710	+893	8088 4,77/10	768K	1x360K	2x360K	32M	*		+400	1	2		+903 +1151
AESIR Viking	+750	8088 4,77/10	704K	1x360K	2x360K 2x720K	20M 30M	*	+350	+550			8	+875 +1225
AKHTER PC	+499	8088 4,77/8	256K	1x360K	2x360K 2x720K	20m	*			1	1	8	
AMI PC88 10	+795	8088 10	512K 640K	1x360K	2x360K	30M	*		+450	*	*	5	+795 +1645
ARC TURBO PC	+499	8088 4,77/8	640K	1x360K	2x360K	20- 40M	*			1	2		
BARBATAN PC	+625	8088 4,77/8	640K	2x360K		20M	*			1	1	8	+625 +920
BETOS PC 88 Turbo	+798	8088 4,77/10	640K	1x360K 1x30M	720K	40M	*	+195	+399	1	1	8	+798
BONDWELL BW 34/36	+599	8088 4,77	640K	2x360K		20M	+30	+329		1		5	+699 +1029
CAS Turbo PC-XT	+650	8088 4,77-10	640K	1x360K 20M		40M	*			1	1		+715
CDATA PC 401	+950	V20 4,77-8	640K	1x360K 20M		40M				1	1		+950
COMPRO 88	+895	V20 4,77/8	640K	2x360K				Mo+ +250		1	1	8	+895
CYBERCOM PC/XT	+1395	8088 4,77	640K	2x360K 20M				*		1	1		+1395
DIAMOND 86	+945	V20 4,77/8	640K	1x360K 20M		40M 83M	*	+185	+395			8	+945 +1850
DIGITASK Axiom XT	+499	8088 4,77/10	640K	1x360K	2x360K	20M 60M	*	+245	+450	1	1	7	+618 +824
FLONEX PC88	+695	V20 4,77/10	640K	1x360K 20M		40M	*	+160	+360	1	1	8	+695
INTER ORIENT 1600	+339	8088 4,77/8	256K 640K	1x360K	2x360K	20M 40-110M	*			1	+30		+499 +699
ISIS 220	+599	V20 4,77/8	704K 1M	1x360K	2x360K	20M	*	+219		1	1		+694 +999
LINK XT	+629	8088 4,77/8	512K 640K	2x360K		20M 30M	*		+505	2	1	8	+629 +889
MATRIX XT-1	+424	V20 80	640K	1x360K	2x360K	20M 30	*	+219	+365	2	1	8	+558 +783
MATRIX XT-2	+1095	V20 8	640K	360K 720K		30M 40M	*			2	1	8	+1170 +1200
META-DYNE PC	+539	V20-8 4,77/8	740K 640K	2x360K	1x360K 1x720K	20M 40M	*	+165	+444 +80card	2	1	88 bit	+717 +976
MICRO MACRO Yuppie	+775	8088 4,77/8	640K	2x360K		10M 20M	*			1	1		+845
MICRONIX PC MARVEL	+749	V20 4,77/8	1M	2x1,2M		20M 40M	*			1	1		+749
MISSION PC2	+790	8088 4,77/8	640K	1x360K		30M	*			1	1	8	+790 +1190
MITAC Paragon 88	+1030	8088 4,77/10		360K 20M			*	+120	+270	1	1	58 bit	
NESS XT	+499	8088 4,77/8	256K 640K	1x360K	2x360K	20M 40M	*						
NTS 100 Series	+459	V20 4,77/14	640K	2x360K	1x720K	20M 40M	*	+190	+385	1	2	8	+748 +1147
NTS 10	+399	V20 4,77/14	640K	1x360K		20M	*	+290	+385	1	2	8	+997
OPUS PC III Turbo	+499	8088 4,77/10	256K 1M	1x360K	2x360K	30M	*	Mo+	+300	1	1	8	+549 +1045
OPUS PC IV Rurbo	+995	8088 4,77/10	768K	1x360K 20M				*		1	1	2	+995 +1045
PACIFIC 88	+499	8088	512K	1x360K	2x360K 1x720K	20M 10M	*		+389	1	1		
PCC 4/8	+739	8088 4,77/7,86	640K	1x360K 30M	2x360K		*			2	1	8	+804
PEACOCK PC/XT	+815	8088	640K	1x360K 20M				Mo+		1	1		+815
PEARTREE PC	+499	8088 4,77/8	512K 640K	1x360K	2x360K	20M	*			1	1	8	+689 +989
Q-DATA Turbo XT	+602	8088 4,77/8	640K			20M	*	+150	+438	1		8	+872 +915
QUBIE 8 Terminal	+575	V20 4,77/8	640K	1x360K	1x720K	20M 42M	*			2	1	8	+575
QUBIE 8 Turbo	+895	V20 4,77/8	640K	1x360K 20M	2x360K 1x720K	42M	*		+325	2	1		+755 +970
SBC FD H020	+599	V40 5,5/8	640K	2x360K		20M	*	+200		1	1	4	+599 +999
SCICON PC2 Turbo	+544	8088 4,77/8	640K	2x360K		1x20 M	+10	Mo+ +126		1	1	8	+617 +813
SCREENS CL PC	+450	8088 4,77/8	640K	1x360K	2x360K	20M	*	+199	+399	1	1	8	+520 +750
SCREENS ST100	+399	8088 4,77/8	512K 640K	1x360K	2x360K	20M	*	+199	+399	1	1	8	+494 +724
SYSTEM ZONE XTel	+499	8088 4,77/8	640K	1x360K	1x1,2M	20M 40M	*		Mono +500	1	1	8	+499 +895
VIGLEN PC Turbo	+499	V20 4,77/8	256K 640K	1x360K	2x360K	20M	*			1		8	
WALTERS PC XT	+399	8088 4,77	256K 640K	1x360K	2x360K	20M	*	+200	+450	+19	+26	28b 616	+775 +1125

Počítače kompatibilní s IBM PC XT

Název počítače	zLstg	uP	Paměť	Disk	+flop	+HD	Her	CGA	EGA	P	S	E	tLstg
ALCATEL XTRA/Prof.300	+995	8088 4,77/10	768K	1x720K	2x720K 1x360K	20M	+ 120	+459	+395	1	2	4	+1115 +1679
AMSTRAD PC 1512	+399	8086 8	512K 640K	1x360K				Mo+ +150		1	1	3	+589
AMSTRAD PC 1640	+499	8086 8	640K	1x360K	2x360K	20M	*	+150	Mo+ +300	1	1	3	+649 +949
ATARI PC1	+347	8088 4,77/8	512K 640K	1x360K	2x360K					1	1	no- ne	
ATARI PC 2	+599	8088 4,77/8	512K 640K	1x360K	2x360K	30M	*		Mo+	1	1	5	+599 +949
CANON AS200	+1595	8086 4,77/7,16	256K 640K	2x360K		20M	*			1	1		
COMMODORE PC1	+499	8088 4,77	512K	1x360K	2x360K 1x720K			Mo+		1	1		
COMMODORE PC10	+699	8088 4,77	512K 640K	2x360K		20M	*	+150		1	1		
COMPAD Deskpro	+1595	8086 4,77/7,16	128K 640K	1x360K	2x360K	20M		Mo+	+631	1	+89	8	+2005 +2555
COMPUCORP Computext 10	+2200	8088 4,77/8	640K	1x360K 1x10M		20M		*		1	2	5	
COMPUCORP Connection 16	+2750	8088 4,44	640K	2x360K		10M		*		1	2		+2750
COMPUCORP Connection 32	+4600	8088/6 NS32032	1,64M 2,64M	1x360K 20M				*		1	2		+4600
DATA GENERAL Dasher One Model 2	+1637	8088 4,77/8	256K 640K	1x720K	2x720K	10M	*			1	1	2	
EPSON PC	+718	8088 4,77	256K 640K	2x360K		20M		Mo+		1	1		
EPSON PC+	+1199	V30 4,77/7,16	640K	1x360K	2x360K	20M	*			1	1		+1340 +1599
EPSON	+1199	8088	640K	1x360K						1	1		
PCe		4,77/10		20M									
ERICSSON PC	+1575	8088 4,77	640K	1x360K	2x360K	20M			Mo+	1	1	8	
FERRANTI XT	+1375	8086 4,77	640K	2x360K		20M	*	+159	+488	1	1	5	+1375 +1725
GOUPIL 65 S86	+1295	8086 10	640K	2x360K		20M	*		+600	1	1	5	
HERMES H80	+1306	8088 4,77/8	250K 640K	1x369K	2x360K	20M		Mo+		1	1		
HERMES PC110	+1609	8086 10	640K	2x360K	720K 1n2	20- 40M	*			1	1	7	
HEWLETT PACKARD Vectra CS	+837	8086 7-16	640K	1x360K	1x1,44 M	20M	*			1	1	7	+837
HONEYWELL PC EP	+1345	8088 4,77/8	256K 640K	1x360K	2x360K	10M	*			1			
HONEYWELL PC XP	+1695	8088 4,77/8	640K	1x360K 20M			*			1	1		+1695
IBM PC XT	+1450	8088 4,77	640K	2x360K		20M	*			+49	1		
IBM PS/2 Model 30	+1496	8086 8	640K	2x720K	1x360K	20M		Mo+ +239		1	1	38 bit	+1566 +2019
KAYPRO K-PC	+999	8088 4,77/8	768K	2x360K		20M	*			1	1		+1258 +1759
MICROVITEC Cubpack PC1/PC2	+649	8088 4,77	512K	1x360K	2x360K			*		1	1		
NIXDORF 8810/35	+2990	8088 4,77	256K 640K	2x360K		10M		Mo+		1	1		
OLIVETTI M19	+1059	8088 4,77/8	256K 512K	1x360K	2x360K	20M		Mo+ +60		1	1		+1395 +1897
OLIVETTI M24	+1366	8086 8	640K	1x360K	2x360K	20M		Mo+ +314		1	1		+1554 +2028
OLIVETTI M240	+1549	8086 10	640K	2x360K	1x720K 2x720K	20M		Mo+ +296	+197 +589	1	1	7	+1690 +2078
OLIVETTI M24SP	+2130	8086 10	640K	1x360K 20M				Mo+ +314		1	1		+2190
OLIVETTI PC1	+399	V40 4,77-8	512K 640K	1x720K	2x720K	40M ext.		Mo+		1	1	1	+399 +599
PANASONIC FX-600	+1274	8086 4,77/7,16	640K	1x360K	2x360K	20M		Mo+		1			+1399 +2089
RAIR PC WORKSTATION	+1554	8088 8	256K 512K	2x360K		12,8 25M	*			1	2		
SAMSUNG SPC-3000	+699	8088 4,77/8	640K	2x360K		20M	*			1	1		+699 +999
SANYO MBC 16	+499	8088 4,77/8	256K 640K	1x360K	2x360K	20M		Mo+ +326		1	1		+725
SANYO MBC 885	+895	8088 4,77/8	256K 640K	2x360K		10M 20M	*			1	+89		
TANDON PCX 20	+1195	8088 4,77	640K	1x360K 20M			*		+395	1	+95		+1195
TANDY 1000HX	+595	8088 4,77/7,16	256K 640K	1x720K	2x720K 1x360K			Mo+ +100		1	+69		+843
TANDY 1000SX	+795	8088 4,77/7,16	640K	2x360K		20M		Mo+ +200		1	+69	5	+795
TULIP PC Compact	+975	V20 4,77/9,54	640K	2x360K		20M	*			1	1		+975 +1350
VICTOR VPCIIC	+1199	8086 4,77/7,16	640K	2x360K		20M 40M	*	+300	+500	1	1		+1199 +1799
ZENITH Eazy PC	+499	V40 7,1	512K	1x720K	2x720	20M		Mo+		1	-	ne	+499 +899
ZENITH Z148	+593	8088 4,77/8	512K 768K	2x360K		20M		Mo+		1	1	1	+593 +889
ZENITH Z159	+849	8088 4,77/8	512K 5M	2x360K		20M	*			1	1	3	+894 +1359



Tiskařské závody 5 přijmou pro pracoviště fotosazby – Praha 10, Sámova 12 – samostatného elektronika pro servis a údržbu fotosazby Linotron 505 (řídící počítač Honeywell). Platové podmínky dle kvalifikace (8. třída TKK + prémie, tj. cca 20 Kčs/hod.). Hlavním předpokladem je fandovství a zájem o výpočetní techniku a elektroniku. Uchazeči o toto zajímavé zaměstnání se mohou přihlásit přímo na uvedené adrese (s. ředitel Kadlec), případně v redakci Mikrobáze.

Redakce Mikrobáze hledá nové spolupracovníky. Pro zkvalitnění obsahu potřebujeme autory, zpravodaje z krajů, ale i odborného redaktora. Pokud tedy máte zájem nám pomoci udělat zpravodaj takový, jaký bychom ho všichni chtěli mít, ozvěte se nám. A ještě něco. Ukazuje se, že někteří autoři velmi zajímavých konstrukcí mají někdy potíže se stylistikou. Často je problém jenom v nedostatku času. Protože máme zájem seznámit naše čtenáře i s výsledky práce těchto autorů, rozhodli jsme se pro malý experiment. Podle možnosti budeme po jistou dobu přijímat i jen heslovité popisy zajímavých konstrukcí. Náš odborný redaktor se podle svých časových možností pokusí (pochopitelně ve spolupráci s autorem) o zpracování takových příspěvků do formy normálního článku. Tyto tzv. „telegrafické náměty" zasílejte přímo na adresu Daniel Meca, Jihlavská 76, 140 00 Praha 4.

TNS-AT

TNS-GC/W

JZD AGROKOMBINÁT SLUŠOVICE

NOSITEL ŘÁDU PRÁCE

JZD AK Slušovice dodává celou řadu různých konfigurací osmi i šestnáctibitových počítačů, vhodných pro nejrůznější nasazení ve všech oblastech hospodářství. Řadu doplňuje přístroj pro akustický přenos dat po telefonních linkách.

TNS-AT/BA

COUPLER TNS