NR INDEKSU 353965
PL ISSN 0860-1674

Bajtek

MAGAZYN KOMPUTEROWY

NR 1 (59)'91 CENA 7600

Z MIKROKOMPUTEREM NA TY

ZACZYNAMY:

KLAN EDUKACJI

KUPIŁEM ATARI:
— i co dalej?

MONTE CARLO

80KB — ZABAWA NA 102!

ZŁOTA 10'90

DODATKOWA PAMIĘĆ RAM
W CPC 6128

MUSIC BY MAC

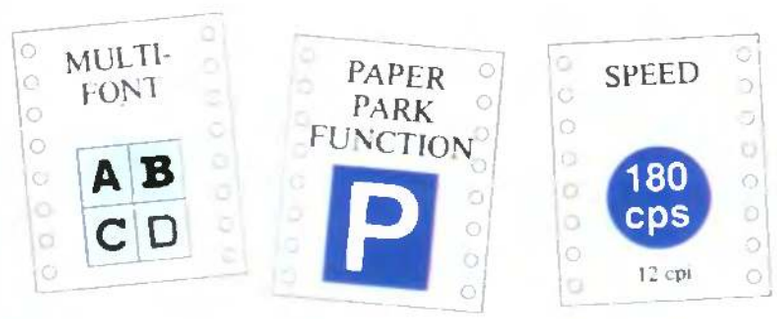
JOYSTICK NA KAŻDĄ OKAZJĘ

DRUKARKA STAR LC-20

— to nowa, szybsza LC-10

- Prędkość druku: 180 zn./sek.
- Jakość druku: standard oraz NLQ
- Traktor pchający
- „Parkowanie" papieru
- Automatyka oddzierania papieru
- Interfejs Centronics

Cena 2.500.000 (orientacyjna cena detaliczna)



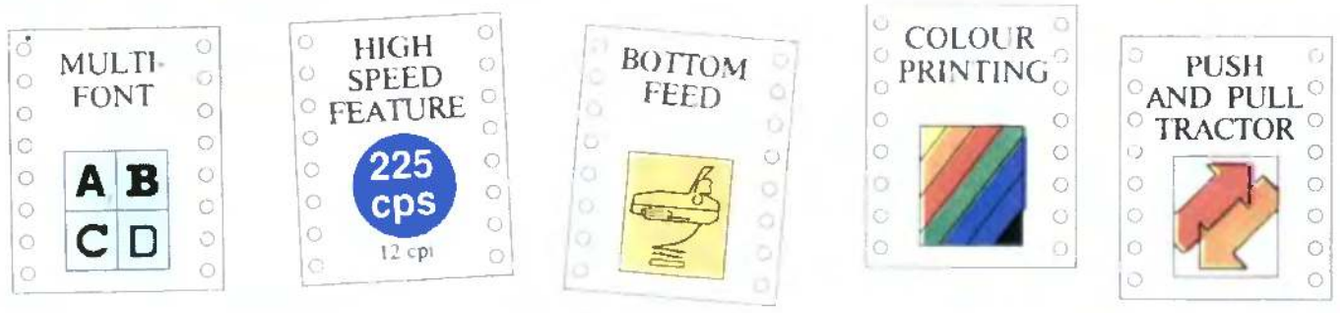
DRUKARKA LC-200

— Star znów ustanawia nowy standard!

- Max. prędkość druku: 225 zn./sek.
- Druk kolorowy
- Możliwość podawania papieru od dołu
- Traktor pchający i ciągnący
- „Parkowanie" papieru
- Automatyka oddzierania papieru
- Interfejs Centronics

Cena 3.900.000

(orientacyjna cena detaliczna)



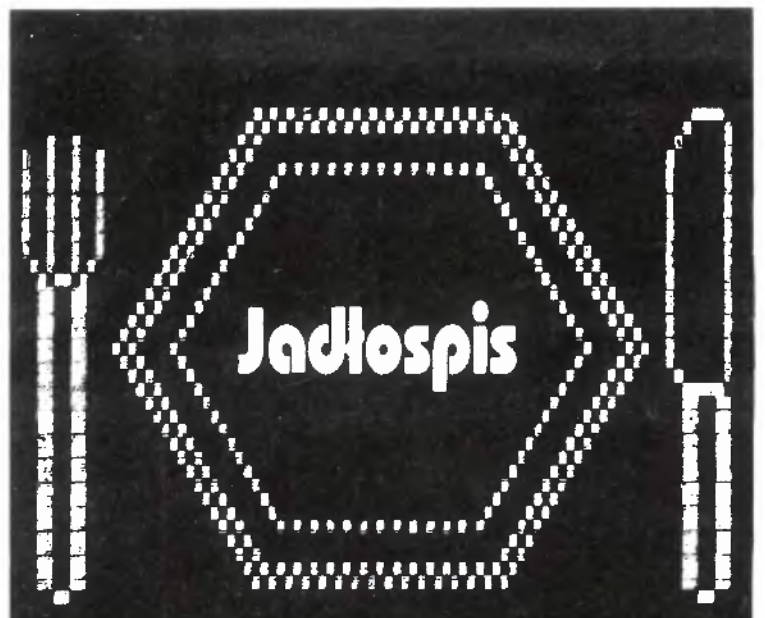
star

Twoja drukarka

ABC DATA WARSZAWA

Przedstawicielstwo w Polsce
ABC Data Warszawa
ul. Waliców 13

tel. 24-11-43
24-78-35
tx. 816-423



— Porządkujemy numerację

Rok 1990 był ciężkim okresem dla całej prasy. Do horendalnych podwyżek cen papieru i druku doszedł rozpad instytucji–molocha, jaką jest pozostająca w długiej likwidacji RSW. W branży pism komputerowych był to rok szczególnie trudny: przestały ukazywać się na rynku dwa tytuły, a pewien znany i zasłużony miesięcznik zrobił się „półrocznikiem".

Problemy te nie ominęły również naszego pisma. Przez osiem pierwszych miesięcy ubiegłego roku, z powodu nieudolności naszgo poprzedniego wydawcy, ukazały się tylko dwa zeszyty „Bajtka" o łączonej numeracji. Decyzja Komisji Likwidacyjnej pozostawiła naszemu zespołowi redakcyjnemu niewielki margines.

Przez cztery miesiące mogły ukazać się i ukazały wydane przez nas samodzielnie cztery numery, także o łączonej numeracji. Częściową reczasowy na wyrównanie tych zaległości.

kompensatą wobec Czytelników było zwiększenie objętości „Bajtka" o 25% przy pozostawieniu tej samej ceny z początku ubiegłego roku.

Pierwszy tegoroczny numer, który mają Państwo przed sobą, liczy 44 strony i ma odrobinę lepszą okładkę. Niestety musieliśmy podnieść cenę pisma, nie tylko z tego powodu. Cała kalkulacja „Bajtka" sporządzona była przy założeniu minimalnych zysków i pewnych ulg, których jak na razie, nie udało się nam uzyskać. Mimo tych problemów mamy nadzieję, że nie będą konieczne następne podwyżki ceny pisma i że kolejne numery „Bajtka", już nie łączone, będa się ukazywać regularnie co miesiąc.

W zeszłym roku, zgodnie z zapotrzebowaniem Czytelników, wprowadziliśmy Klan IBM, w tym roku — kierując się podobnymi przesłankami — dodaliśmy 3-stronicowy Klan Edukacji. Liczymy na pomoc Państwa w jego redagowaniu. Zadania i problemy jakim ma być poświęcony ten nowy Klan, zostały przedstawione w artykule pod tytułem „Zaczynamy".

W Klanie ZX Spectrum możemy polecić materiał red. Magdziaka dotyczący procedur niskiego poziomu dla stacji FDD. Jest on pewną kontynuacją „Operacji dyskowych ...", które zdominowały ubiegłoroczny Klan Amstrad. Podobny tekst dla małego Atari obiecał także p. Zientara. Lżejsze, ale równie ciekawe informacje, tym razem o grach wykorzystujących rozszerzenie pamięci dla ZX'a, znajdują się w artykule pt. „Zabawa na 102 przy 80K". W Klanie Amstrad więcej materiałów dla posiadaczy CPC oraz informacje o nowej nakładce na system operacyjny dla Joyce'a.

Dla osób zniechęconych trochę wielkością i zawartością Klanu Commodore mamy dobrą wiadomość: od następnego numeru „Bajtka" Klan wraca do poprzedniej objętości 3—4 stron. Kolega Klaudiusz Dybowski, wspomagany nowymi siłami, bierze się znowu do roboty.

W Klanie IBM m.in. informacje o skanerach i kolejnych pożytkach płynących z posługiwania się MS Windows. Osobom zainteresowanym bardziej rozrywkowym wykorzystaniem najszybszych IBM'ów można polecić tekst pt. „Wolno, wolniej ..." Z materiałów znajdujących się poza Klanami chcemy zwrócić uwagę Państwa na artykuł poświęcony zastosowaniom układu 8255, będącego swoistym oknem na świat dla komputerów nie tylko serii ZX Spectrum.

Rewelacją końca roku 1990 jest nowy produkt pana Jobsa — NeXT Station Color. W stosunku do poprzednich modeli różni się m.in. wprowadzeniem grafiki kolorowej, szybszym procesorem (Motorola 68040/25) i zewnętrzną stacją dyskietek 3.5" o pojemności 2.8 MB. Nie bez znaczenia jest też stosunkowo niska cena tego sprzętu. Z kolei, znana Czytelnikom „Bajtka", amerykańska firma Microway wprowadza nowe „szybkościowe" produkty oparte na superkości, jaką jest układ Intel i860. Ciekawostką w dziedzinie software'u jest pojawienie się języka Modula-3, ukierunkowanego na prostotę i bezpieczeństwo programowania.

O tym, że komputer bez oprogramowania jest kupą złomu, pisaliśmy wielokrotnie. Sami, zresztą na własnej skórze, przekonujemy się o tym od momentu rozpoczęcia prenumeraty wydawanych przez nas czasopism. Pewne opóźnienia i potknięcia w tej dziedzinie, za które bardzo przepraszamy zainteresowane osoby, wynikają z niedoskonałości przyjętych dotychczas metod i napisanych programów. Mamy nadzieję, że rozwiążemy te problemy w najbliższym czasie, zapewniając wygodną i regularną prenumeratę naszych tytułów.

Jarosław Młodzki

PRENUMERATA

Przez cały czas przyjmujemy zamówienia na prenumeratę wszystkich trzech wydawanych przez nas tytułów. Prenumeratę można zamówić na dowolną liczbę egzemplarzy dowolnych tytułów. W tym celu należy wpłacić odpowiednią kwotę na nasze konto:

Spółdzielnia „Bajtek"
Bank „Agrobank S.A."
479994-1834-131
ul. Grochowska 262
04-398 Warszawa

Wstępny koszt rocznej prenumeraty jednego egzemplarza „Bajtka" wynosi 66000 zł, „Top Secret" (6 numerów) 33000 zł, „Mojego Atari" (6 numerów) 30000 zł. Jeśli cena któregoś z tytułów zostanie podniesiona, wpłata będzie potraktowana jako przedpłata i niezbędna będzie dopłata. O jej konieczności będziemy informować w tym miejscu. Prosimy o bardzo staranne wypełnianie przekazów długopisem, drukowanymi literami i podawanie na odwrocie wszystkich trzech informacji o tym, jakiego okresu i których tytułów dotyczy prenumerata — nie wiadomo, który odcinek dostarczy nam poczta. Niewyraźnie lub błędnie wypełnione przekazy mogą być źródłem opóźnień i błędów w realizacji Waszych zamówień. W przypadku kontynuacji prenumeraty prosimy o podanie swojego numeru.

Uwaga: po otrzymaniu przekazu wysyłamy pocztą najwcześniejszy możliwy numer. Liczba wysłanych numerów będzie zgodna z zamówieniem. W razie błędnej realizacji prenumeraty, błędu w adresie lub nieotrzymania pierwszego numeru zamówionych pism w ciągu dwóch tygodni od momentu ukazania się ich w kioskach, prosimy o kontakt w celu umożliwienia nam wyjaśnienia sprawy.

Bajtek MAGAZYN KOMPUTEROWY

Redakcja:
ul. Wspólna 61,
00-687 Warszawa,
tel. 21-12-05

Redaguje Kolegium:
Marcin Borkowski, Marek Czarkowski, Łukasz Czekajewski, Janusz Jarmoch, Jarosław Młodzki (Redaktor Naczelny), Waldemar Nowak, Maciej Pietraś, Marcin Przasnyski, Wanda Roszkowska, Wojciech Zientara.

Sekretarz redakcji: *Marcin Przasnyski*

Szefowie Klanów:
Amstrad — *Jonasz Mayer*
Atari — *Wojciech Zientara*
Commodore — *Klaudiusz Dybowski*
Edukacja — *Tadeusz B. Mańk*
Co Jest Grane — *Łukasz Czekajewski*
IBM — *Marcin Borkowski*
Micro Magazyn — *Janusz Jarmoch*
Spectrum — *Maciej Pietraś*

Stali współpracownicy:
Grzegorz Bujanowski
Jarosław Burczyński
Piotr Kos
Robert Magdziak
Grzegorz Ostapiuk
Andrzej Pilaszek
Mieczysław Płacheta
Marek Sawicki
Piotr Sumara
Michal Szokoło
Stanisław Winiecki

Opracowanie graficzne: *Wanda Roszkowska*
Redaktor stylistyczny: *Maria Radzimińska*
Zdjęcia: *Leopold Dzikowski*

Wydawca:
Spółdzielnia „Bajtek"
ul. Wspólna 61
00-687 Warszawa

Skład i Druk:
Prasowe Zakłady Graficzne w Ciechanowie
Fotoskład: *Grażyna Kurzątkowska*
Montaż: *Grażyna Ostaszewska*
Korekta: *Maria Krajewska, Teresa Rutkowska*

Nakład: 107 tys. egz. Zam. 84260.

MICRO magazyn

COMPAKT 386

Compakt 386, niepozornie wyglądający przenośny komputer firmy Comcen, przewyższa pod względem szybkości działania i pojemności pamięci niejedną potężną maszynę liczącą. W jego wnętrzu kryje się mikroprocesor Intel 80386—20, 2 MB pamięci RAM (z możliwością rozszerzenia do 10 MB), pamięć podręczna 64 KB (cache memory) oraz stacja dysków elastycznych 3.5", 1.44 MB. Ale to nie wszystko. Taka wyliczanka nie zaskoczy dziś nikogo, kto interesuje się laptopami Toshiby, Atari czy Macintosha. Znawcy tematu będą poszukiwali czegoś naprawdę zaskakującego. Niespodziankę sprawiają nam dopiero dwa inne zainstalowane w komputerze napędy. Któż mogłby przypuszczać, że w skrzyneczce o wymiarach 228x178x406 mm skrywają się urządzenia zdolne do przechowywania aż 1 gigabajta (1.000.000.000 bajtów) danych?

Z tej niewyobrażalnej liczby bajtów prawie połowa pochodzi z twardego dysku o pojemności 512 MB. Pozostała część, około 530 MB, przypada na stację dysków optycznych CD-ROM. Producentem napędu CD-ROM jest dobrze znana klientom naszej Baltony firma Hitachi, jeden z czołowych producentów urządzeń audio-video.

Dla postronnego obserwatora stacja CD-ROM może w pierwszej chwili wyglądać jak zwykła stacja dysków elastycznych 5.25". Ma ona jednak kilka ciekawych dodatków: wskaźniki kontrolne, przycisk EJECT wysuwający dysk i miniaturowe gniazdko do podłączenia słuchawek stereo lub wzmacniacza Hi-Fi. Napęd CD-ROM można wykorzystywać jako pamięć masową komputera albo zwykły odtwarzacz płyt kompaktowych. Co więcej, muzyka może być zapisana na tym samym dysku, co dane i programy. Jeden z dysków CD-ROM zawiera nawet katalog wszystkich klasycznych płyt CD wraz z nazwami utworów, nazwiskami wykonawców i autorów, fragmentami recenzji z magazynów muzycznych, a nawet krótkimi urywkami melodii.

Oprogramowanie dostępne na CD-ROM liczy już ponad 2900 tytułów. Są wśród nich programy public domain, użyteczne programy inżynierskie, dane dla programów DTP.

Prawdziwą atrakcją są jednak słowniki i encyklopedie. Na CD-ROM zapisano już kompletny słownik „The Oxford English Dictionary" z etymologią, wymową i objaśnieniami oraz „Whittaker's Books in Print", zawierający szczegóły każdej książki aktualnie drukowanej w Wielkiej Brytanii. CD-ROM okazał się również bardzo atrakcyjnym sposobem popularyzacji publikacji naukowych z dziedziny chemii, medycyny, farmacji i informatyki. Umożliwiają one niezwykle szybkie i łatwe wyszukiwanie żądanej informacji. Firma Microsoft publikuje w ten sposób wszystkie opisy i podręczniki do pakietów produkowanego przez nią oprogramowania i systemów operacyjnych.

(J)

Na podst. Personal Computer World

AMNESTIA DLA PIRATÓW

XTree Software, wydawca popularnego programu XTree ogłosił pod koniec ubiegłego roku propozycję amnestii dla piratów oprogramowania tej firmy. Kierownictwo XTree dało szansę przyznania się do grzechu i zapłacenia niewielkiej sumy pieniędzy, za którą winowajca otrzymuje aktualną wersję programu wraz z instrukcją. Właściciel pirackiej kopii powinien wysłać wydruk jednej ze stron programu oraz $20. Firma zobowiązała się do dostarczenia najnowszej wersji XTree (cena rynkowa $70) a także obiecała, że nie oskarży użytkownika o naruszenie praw autorskich.

Podobną ofertę ma zamiar zaproponować piratom firma Kyocera Unison, producent programu PrintMaster Plus. Żadne z przedsiębiorstw nie jest zainteresowane wytoczeniem setek, a może tysięcy procesów o piractwo. Propozycja amnestii została opracowana z myślą o zaoszczędzeniu wydatków na koszty sądowe i z nadzieją na uzyskanie dodatkowych wpływów. Piractwo jest jednym z głównych problemów firmy software-owych, który istnieje od momentu pojawienia się tej gałęzi przemysłu.

Pomysł amnestii jest uznawany za nowatorski. Ani XTree, ani Kyocera nie były pewne, jak zareagują użytkownicy pirackiego oprogramowania. Jest jednak nadzieja, że w świadomości użytkowników utrwali się przekonanie o nielegalności piractwa i jego szkodliwości dla samych odbiorców oprogramowania.

(J)

Na podst. Personal Computer World

Monitor portretowy

Na pierwszy rzut oka odnosimy wrażenie, że jakiś dowcipniś zrobił nam kawał. Nikt normalny nie przekręci przecież na bok monitora, który od dawna spokojnie stoi przy komputerze. Proponuję jednak przyjrzeć się troszkę uważniej. Urządzenie pracuje w prawidłowej pozycji!

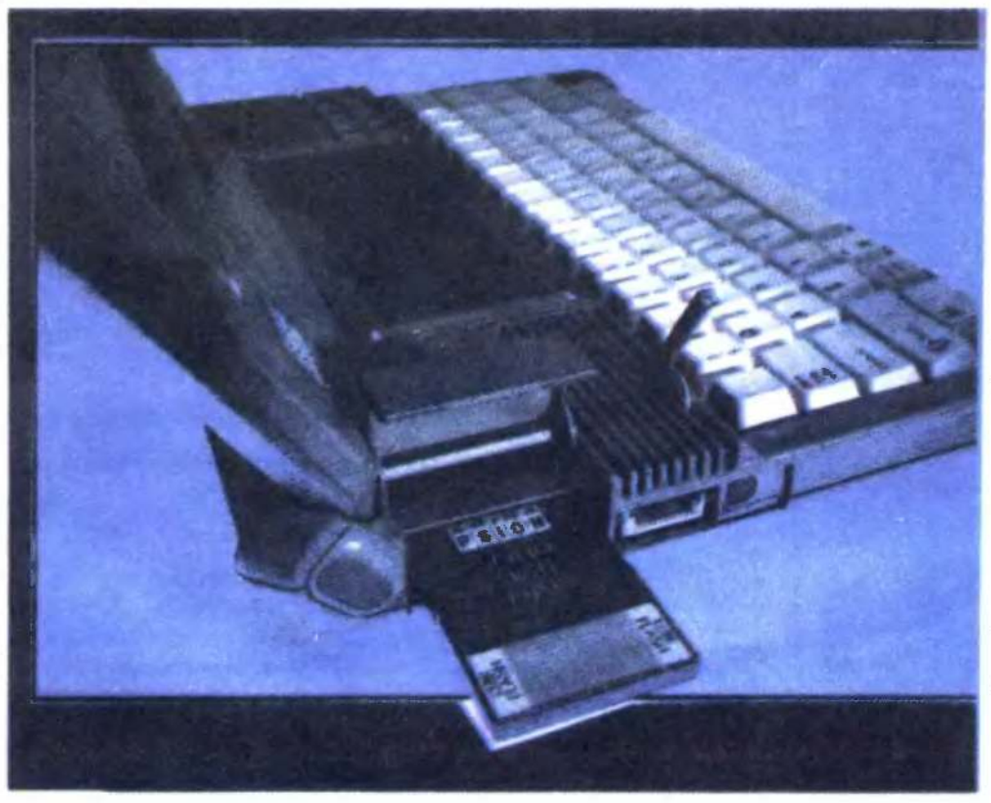
Apple Macintosh Portrait Monitor ma przekątną 15", to znaczy około 38 cm. Można na nim wyświetlić pełny arkusz A4, łącznie z charakterystycznym dla Macintosha menu i ramkami. Urzędnicy i inżynierowie, którzy rzadko mają do czynienia z formularzami i rysunkami mniejszego formatu, mogli dotychczas tylko marzyć o ekranie mieszczącym całą stronę tekstu.

Na monitorze prawie nie dostrzegamy migotania. Zawdzięczamy to nowoczesnej technologii. Częstotliwość odtwarzania obrazu, która wynosi 75 Hz, jest o kilka Hz większa, niż w innych typowych monitorach. Wzrok użytkownika siedzącego przed takim ekranem męczy się znacznie wolniej, a więc pracuje się o wiele bezpieczniej i przyjemniej. Prawie całkowicie pozbyto się zniekształceń obrazu, które tak często obserwujemy przy krawędziach monitorów przeciętnej jakości. Konstruktorzy zastosowali bowiem nowoczesny kineskop o prostokątnych narożach.

Monitor jest wyposażony w trzy złącza ADP (Apple Desktop Bus), które umożliwiają bezpośrednie przyłączenie klawiatury, myszy lub innych urządzeń. Długi, bo aż 2 metrowy kabel zasilający, ułatwia użytkownikowi komputera wygodne zorganizowanie miejsca pracy.

Pełnostronicowy monitor znakomicie spełnia typowe wymagania użytkowników Macintosh II. Nadaje się on szczególnie do wykorzystania w biurach, przy przetwarzaniu tekstów i korzystaniu z programów Desktop Publishing.

(J)



DYSKI STAŁE

Producenci komputerów przenośnych i kieszonkowych mieli spore problemy z dobraniem odpowiedniego nośnika, na którym będą przechowywane dane. Tradycyjne stacje dysków elastycznych i twarde dyski są w niektórych przypadkach zbyt duże i nie mieszczą się w zminiaturyzowanych urządzeniach.

Firma Psion, wspólnie z Intel i Microsoft, opracowała nowy standard nośnika informacji, przeznaczony dla najmniejszych komputerów. Tym nowym medium są dyski stałe (Solid State Disk — SSD). Wyglądem przypominają one kartę kredytową. Produkuje się je w trzech odmianach: ROM, RAM lub EPROM. Karty RAM mają pojemność 64, 128 lub 256 kB i są zasilane z własnej, miniaturowej baterii litowej. Dyski EPROM nie potrzebują żadnego źródła energii do podtrzymania zapisanych danych. Kasowanie zapisu, w odróżnieniu od konwencjonalnych EPROMÓW, nie wymaga zastosowania specjalnych urządzeń ultrafioletowych. Dane mogą być wymazane sygnałami elektrycznymi bezpośrednio przez system operacyjny. Pojemność kart EPROM wynosi 128, 256 lub 512 kB.

Dyski stałe współpracują z komputerem poprzez interfejs szeregowy. Złącze znajduje się na krawędzi karty. Szybkość transmisji danych jest zbliżona do szybkości transmisji w twardych dyskach o czasie dostępu 65 ms.

Karta zawierająca układy ROM, RAM i EPROM kojarzy się z kasetą pamięci stałej, tzw. cartridge. Dlaczego więc nazwano ją dyskiem stałym. Zdecydowała o tym organizacja zapisu danych. Pamięć cartridge'a zastępuje pewien obszar pamięci adresowanej przez procesor. Dyski stałe zachowują się tak jak zwykła dyskietka. Danymi zarządza dyskowy system operacyjny. Programy zapisywane są w postaci zbiorów.

(J)

NOWA DRUKARKA STAR-a!

LC-20 to nowa drukarka firmy Star Micronics, mająca zastąpić popularną LC-10. „Dwudziestka" jest nie tylko szybsza (180 zn/s), lecz ma też nowocześniejszą obudowę i wygodniejszy panel sterujący.

Poza tym LC-20 posiada wiele funkcji, zwykle oferowanych przez sprzęt droższy, np. parkowanie papieru na wstędze, automatyka oddzierania stron, traktor pchający oraz półautomatyka podawania pojedyńczych kartek.

Poważną zaletą LC-20 są wbudowane polskie znaki wg standardu Mazovii, co poważnie dystansuje inne drukarki tym bardziej, że „elcetka" kosztuje tylko 2.5 mln zł. Standardowy interface Centronics pozwala na współpracę z każdym popularnym u nas komputerem.

(KM)

POŻEGNANIE Z MYSZĄ?

Mysz, spopularyzowana w latach siedemdziesiątych przez Apple-Macintosh, jest dziś standardowym wyposażeniem większości komputerów osobistych. O jej popularności zadecydowały walory ergonomiczne: znakomicie nadaje się do sterowania ruchem kursora na ekranie, wybierania opcji menu, piktogramów, przesuwania i zmiany wielkości okien. Mysz ma jednak jeden mankament. Musi być przesuwana po twardym podłożu, najlepiej po specjalnym dywaniku z tworzywa sztucznego. Jej ruch znacznie ogranicza wolną powierzchnię biurka. Mankament ten ogranicza zastosowanie myszy w komputerach przenośnych, które nie zawsze są eksploatowane w komfortowych warunkach pracowni albo biura.

Wad, które psują krew użytkownikom myszy, nie ma inne, działające na podobnej zasadzie urządzenie. Nie zdobyło ono jeszcze tak dużej popularności, ale prawdopodobnie w niedalekiej przyszłości znajdzie wielu zwolenników. Najtrafniej można je określić jako mysz odwróconą do góry brzuchem. Kulka jest umieszczona tym razem na górze. Sterowanie czynnościami programu jest znacznie szybsze i precyzyjniejsze. Dotychczas przejechanie kursorem do bardzo odległego punktu było dość kłopotliwe. Mysz trzeba było w takiej sytuacji przesunąć

parę razy, a kulkę można obracać w zamierzonym kierunku bez żadnych przerw.

Nowe urządzenie jest produkowane w dwóch zasadniczych wersjach. Pierwsza odmiana, wbudowana na stałe przy klawiaturze komputera, nazywa się Trackball. To rozwiązanie przyjęło się już w komputerach przenośnych. Trackball zastosowano na przykład w laptopie Atari Stacy.

Druga, ruchoma wersja, bardziej przypomina wyglądem tradycyjną mysz. Producenci oferują ją pod różnymi nazwami: Fast Trap, Mouse Track, Tracker.

Nie wiadomo jeszcze jakie słowo przyjmie się w języku polskim dla określenia tych urządzeń. Niemcy już od dawna stosują oryginalny, angielski termin Trackball. Duże szanse ma również dźwięcznie brzmiący Tracker. Przypuszczam, że zadomowi się on także w naszym żargonie komputerowym. A może ktoś wymyśli jakąś polską nazwę?

(J)

PROSCAN — ręczny skaner do PCW

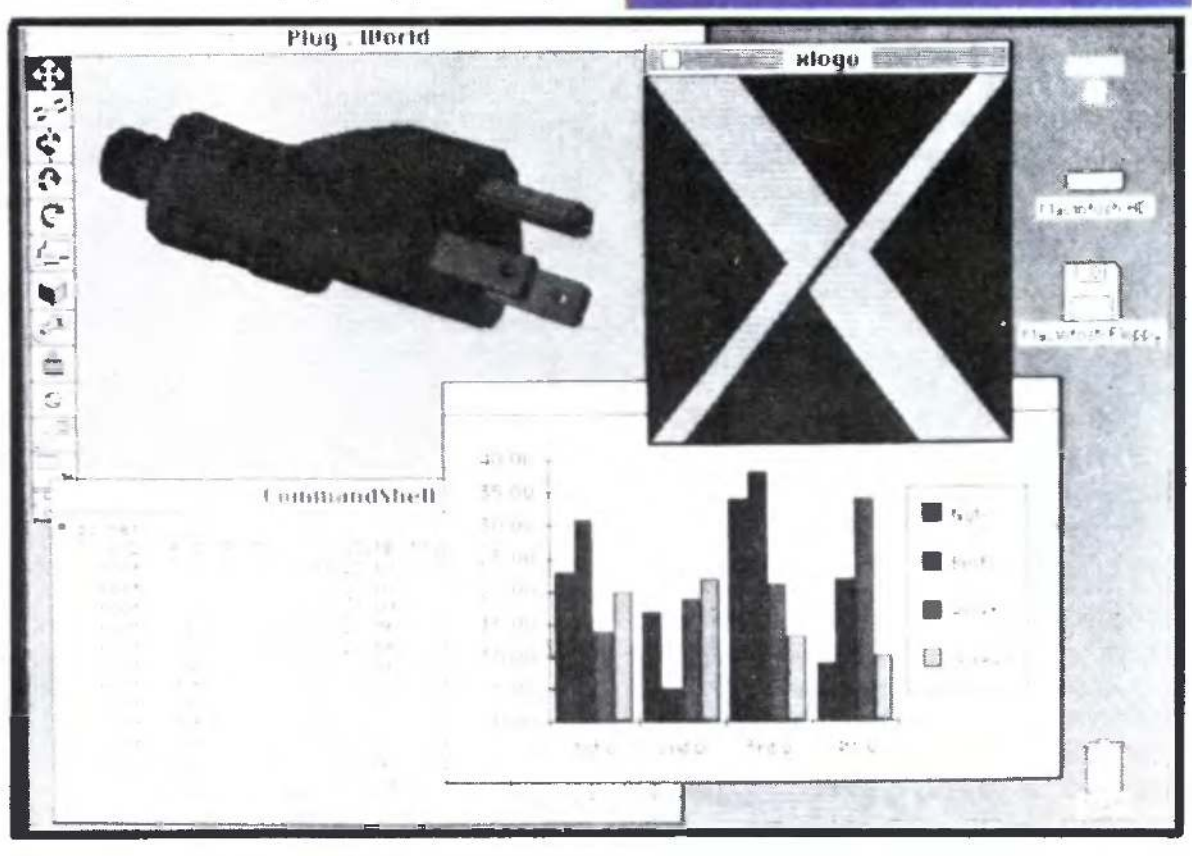
Angielska firma Creative Technology we współpracy, ze znaną Czytelnikom Bajtka firmą SCA Systems, wypuściła na rynek jesienią 1990 roku urządzenie o nazwie ProScan. Jest to ręczny skaner o maksymalnej rozdzielczości 400 dpi (dots per inch — punkty na cal) do komputera Amstrad PCW.

Razem ze skanerem dostarczany jest interface i oprogramowanie pozwalające na prostą obróbkę skanowanych obrazów. Możliwa jest współpraca z drukarkami różnych typów (9-, 24-igłowe, laserowe i strumieniowe — InkJet).

Wykorzystanie programu Flipper 2 Plus pozwala na jednoczesne uruchomienie programów Proscan i Micro Design II z możliwością wymiany danych między nimi.

Cały zestaw — skaner, interface i oprogramowanie — kosztuje w Anglii 179 funtów.

(JM)



SYSTEM OPERACYJNY A/UX

Dialog z komputerem powinien być prowadzony w sposób możliwie prosty, przypominający naturalny język, którym na codzień posługuje się użytkownik. Oczywiście idealnym rozwiązaniem byłoby nauczenie komputera ludzkiej mowy, ale do tego jeszcze daleko. Na razie za najlepsze rozwiązanie uważane są ikony, czyli obrazki symbolizujące czynności wykonywane przez program. Koncepcję porozumiewania się przy pomocy graficznego systemu ikon rozwinęła firma Apple.

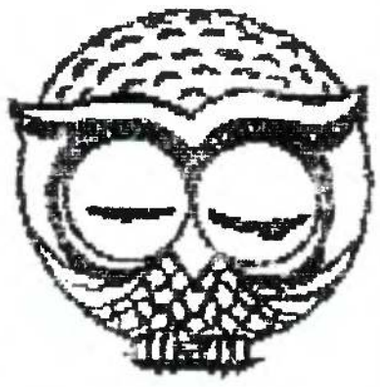
Od niedawna graficzne środowisko towarzyszy również systemowi operacyjnemu UNIX. Apple oferuje ulepszoną wersję systemu, którą oznaczono symbolem A/UX 2.0. Stanowi on implementację UNIX, wzbogaconą o możliwość komunikowania się przy pomocy ikon. Nowa wersja A/UX umożliwia pełniejsze wykorzystanie zalet oprogramowania komputerów Macintosh. Gerhard Jorg — dyrektor Apple Computer GmbH w Monachium — uważa, że połączenie zalet koncepcji realizowanej w oprogramowaniu Macintosha z potęgą UNIX będzie w odczuciu użytkowników rewolucyjnym krokiem.

A/UX 2.0 przejął z Macintosha charakterystyczny sposób porozumiewania się z użytkownikiem, a więc czytelny układ menu i ikon. Pozwala on użytkownikowi jednocześnie korzystać z kilku aplikacji UNIX, systemu X-Window i biblioteki Macintosha. Prościej mówiąc komputer pracuje wielozadaniowo, a efekty poszczególnych programów można obserwować na ekranie jednocześnie w kilku różnych oknach.

A/UX jest przeznaczony dla następujących komputerów osobistych: Macintosh SE/30, Macintosh II, Macintosh IIx, IIcx, IIci i najnowszym IIfx. Minimalny rozmiar pamięci RAM pozwalający uruchomić system wynosi 4 MB. Oprogramowanie systemowe jest dostarczane na taśmie, dyskietkach lub dyskach optycznych.

(J)

PO DZWONKU



Zaczynamy!

Człowiek uczy się przez całe życie — tę prawdę znamy wszyscy. Wiemy również, że czym skorupka za młodu nasiąknie..., i staramy się jak najwięcej nauczyć nasze dzieci. Współczesny świat jest jednak zupełnie inny niż świat naszego dzieciństwa i naszej młodości (to były czasy...). Na naszych oczach, a nawet za sprawą nas samych, dokonała się rewolucja. Rewolucja informatyczna. Nie trzeba być osobą w starszym wieku, by pamiętać świat, w którym komputerów nie było, lub zajmowały wielkie pokoje, nie potrafiąc prawie nic. A dziś? Nie musimy się nawet bardzo rozglądać, by znaleźć w pobliżu jakąś klawiaturę. Jeśli tempo komputeryzacji nie spadnie, to za kilka lat życie bez tej maszynki będzie bardzo uciążliwe. Tak jak dziś trudno egzystować bez elektryczności, telefonu, samochodu. Dlatego niezbędne jest, by nasze dzieci „nasiąkały" informatyką.

Do szkół (polskich) komputery przenikają z wielkimi oporami. Dzieje się tak pewnie dlatego, że nauczyciele są zapracowani jak mało kto. Na lekcji nie można wypić herbaty, przeczytać gazety czy zadzwonić do znajomych. Tu się ciężko pracuje, będąc cały czas pod czujnym ostrzałem dwudziestu lub więcej par oczu, a każde potknięcie czy rozkojarzenie nauczyciela zostanie natychmiast zauważone i skomentowane. Tworzenie zaś zasad posługiwania się komputerem w procesie edukacji zajmuje tak wiele czasu, że żaden nauczyciel temu nie podoła. W Polsce niestety nauczyciel musi te zasady tworzyć, gdyż gotowych wzorców jest bardzo mało.

Dlatego też chcemy w „Bajtku" stworzyć dział poświęcony edukacji. Chcemy, by nauczyciel nie musiał zaczynać od czytania instrukcji obsługi i poznawania budowy komputera. By każdy nauczyciel w szkole mógł od zaraz (jeśli oczywiście ma tam komputer) z niego skorzystać wiedząc niewiele ponad to, jak się go włącza i uruchamia program.

W dziale tym będziemy prezentować istniejące oprogramowanie dydaktyczne (jest go w Polsce — wbrew pozorom — całkiem dużo, tylko nie wiadomo jak do niego dotrzeć). Będziemy się starali, by każdy program był przedstawiany przez nauczyciela, który już używał go na lekcji i zna jego przydatność. Niestety, z przyczyn technicznych nie będziemy mogli sami zajmować się kolportażem oprogramowania — choć być może zmieni się to w przyszłości — ale zawsze będziemy podawali możliwości zakupu.

Polityka tego działu będzie zmierzała do tego, by uzyskać sprzężenie zwrotne między szkołami i firmami tworzącymi oprogramowanie. Oczekujemy na Państwa sugestie dotyczące potrzebnych programów i opinie dotyczące istniejących. Sugestie te przekażemy znanym nam firmom, a Państwu z kolei zaprezentujemy ich nowe wytwory, spełniające wasze oczekiwania. My, jako „przekaźnik", będziemy ze swojej strony dążyć do podniesienia jakości technicznej programów przez pokazywanie sprawdzonych wzorów, dążenie do standardów współpracy z użytkownikiem, prezentację światowych i polskich kierunków rozwoju oprogramowania i opinii specjalistów od oprogramowania edukacyjnego.

Prócz programów komercyjnych chcemy na naszych łamach stworzyć forum programów tworzonych przez amatorów. Programy te będziemy publikować w postaci listingów, a ich autorami będą nauczyciele-zapaleńcy, uczniowie i wszyscy ci, którzy chcą i mogą pisać takie programy. Zapraszamy wszystkich do współpracy. Przed prezentacją każdy z programów zostanie wykorzystany zgodnie z jego przeznaczeniem, a jego oceny dokona nauczyciel, który go używał.

Przed nauczycielem, który chce zacząć używać komputer, staje duża bariera sprzętowa. Co kupić, który komputer będzie najlepszy dla naszych celów, jaka konfiguracja będzie konieczna, a co będzie użytecznym „bajerem" — to pytania, na które trudno znaleźć odpowiedź osobie bez przygotowania. Dlatego też chcemy przedstawiać różne dostępne w Polsce komputery. Opisy te będą miały formę krótkiego kompendium wiedzy i przydatności danego modelu. Zaczniemy (za miesiąc) od popularnego ZX SPECTRUM, potem inne, bardziej lub mniej popularne modele.

Kierunki rozwoju, opinie fachowców-dydaktyków, nowinki światowe to inny temat, którym chcemy zająć się na naszych łamach. Chcemy pokazywać efektowne doświadczenia i niekonwencjonalne wykorzystanie komputera w edukacji, a także propozycje wspomagania edukacji przez komputer, które jeszcze nie znalazły rozwiązania (np. scenariusze programów).

Tak w skrócie przedstawiają się nasze zamiary. Co z tego uda się nam zrealizować — pokaże czas. Liczymy również na pomoc Państwa. Prosimy o uwagi i propozycje. Ten klan „Bajtka" jest poświęcony Państwu, a możemy go dobrze prowadzić tylko wtedy, gdy spełnimy Wasze oczekiwania. Listy i telefoniczne uwagi bardzo nam pomogą prowadzić ten dział.

Zaczynamy! Już dawno po dzwonku!

T.B. Mańk

UWAGA!

W tym numerze znajdziecie Państwo tekst pt. „8255 — Okno na świat". Radzimy przeczytać go uważnie, gdyż w jednym z najbliższych numerów będziemy przedstawiali serię doświadczeń szkolnych możliwych do przeprowadzenia przy udziale tego interface'u.

(red.)

JAK TO ROBIĄ INNI?

rozmowa z prof. Uniwersytetu Londyńskiego, przewodniczącym GIREP, dyr. Center for Educational Studies, Paulem Blackiem

— W jaki sposób komputery wykorzystywane są w angielskich szkołach?

— W naszym ogólnonarodowym programie mamy nowe wymagania dotyczące techniki informacji (*nauka o sposobach przekazywania informacji, w odróżnieniu od informatyki — przyp. red.*). Uważamy, że technika informacji nie powinna być wykładana jako odrębny przedmiot, ale że trzeba ją wykorzystywać na wszystkich lekcjach. W szczególności będziemy wymagali, by nasi uczniowie posługiwali się gotowymi programami nie tylko w przedmiotach przyrodniczych i technicznych, ale także w przedmiotach humanistycznych. Próbujemy kłaść nacisk na to, by uczniowie nauczyli się posługiwać komputerami i oprogramowaniem w różnych aspektach życia. I tak np. umiejętność wykorzystania edytorów tekstu może być pomocą na lekcjach angielskiego lub przy prezentacji obowiązkowych projektów (*prace semestralne wymagane w szkołach angielskich — przyp. red.*) czy publikacji własnych przez uczniów. Znajomość baz danych może pomóc w poznawaniu miejscowej historii lub umożliwić zbieranie danych eksperymentalnych dotyczących środowiska. Uczniowie powinni mieć również doświadczenie w modelowaniu, to znaczy w wykorzystaniu symulacji, czy w używaniu prostych modeli. Mogą wówczas opracowywać proste układy, w których komputer jest używany do sterowania. Wszystkie te aspekty są uwzględnione w naszym programie.

— Jakie komputery są wykorzystywane w szkołach angielskich?

— Większość naszych szkół ma różne modele BBC, niektóre ZX SPECTRUM (znany w Polsce). Jednak wachlarz się rozszerza, bo wiele firm chce wejść ze swoim sprzętem na chłonny rynek szkolny. Jednak te dwa typy są najpopularniejsze, ze względu na to, że mają poparcie rządowe.

— A jak wygląda sprawa oprogramowania?

— Przez ostatnie siedem lat obowiązywał nas ministerialny program, który miał doprowadzić do powstania oprogramowania dla ludzi pracujących w szkołach, współpracujących ze szkołami lub dla instytucji takich jak moja, gdzie zaprasza się nauczycieli, by pomagali w produkcji oprogramowania. Możemy być zatem pewni, że oprogramowanie jest zgodne z zasadami dydaktyki i programami nauczania. Te podstawy bardzo nam teraz pomagają, gdyż dzięki temu mamy duży zapas dobrego oprogramowania dostępnego dla szkół. Naszym głównym problemem jest przekonanie nauczycieli do wykorzystywania komputerów na swoich lekcjach, by nabrali do nich zaufania i potrafili w naturalny sposób włączyć je w tok nauczania.

— Jak w Waszym kraju odbywa się szkolenie nauczycieli do pracy z komputerami?

— Nauczyciele są do tego przygotowywani już w trakcie studiów, tak w collegeach, jak i w ośrodkach kształcenia nauczycieli. Szkolenie to dotyczy wszystkich — niezależnie od kierunku. Także na moim wydziale wszyscy studenci uczą się wykorzystywać komputery i oprogramowanie w procesie dydaktycznym. Lokalne władze szkolne mogą przygotowywać nauczycieli również na kursach dokształcających. Na moim uniwersytecie prowadzimy kursy doskonalące dla dydaktyków. W swoim środowisku stają się wtedy ekspertami i mogą uczyć innych nauczycieli, ale takich szkoleń nigdy dość...

— Jak uczniowie podchodzą do wykorzystania komputerów w edukacji?

— Większość z nich to entuzjaści. Wielu uczniów ma komputery w domu i nie są one dla nich czymś obcym. Myślę, że jesteśmy jednym z krajów mających najwyższe nasycenie komputerami w domu, a jedynym problemem jest to, że chłopcy dominują i starają się odepchnąć dziewczęta od klawiatury. Dzieje się tak prawdopodobnie dlatego, że rodzice uważają komputer za coś nieodpowiedniego dla dziewcząt. Nauczyciele muszą to zatem sami regulować, tak by wszyscy uczniowie, niezależnie od płci, nabrali przekonania do komputerów i wprawy w ich użytkowaniu.

— Czy szkoły angielskie są połączone ze sobą siecią komputerową?

— Robimy takie eksperymenty. Szkoły wymieniają informacje, np. na temat lokalnej historii. Prowadzone były międzyszkolne badania radioaktywności i występowania kwaśnych deszczy. Mamy sieć szkolną, pokrywającą nasz kraj od Anglii do Walii.

— Czy w szkołach angielskich istnieje informatyka jako odrębny przedmiot?

— W niektórych tak, jednak nie we wszystkich. Staramy się wprowadzać ten przedmiot do szkół, ale jako przedmiot fakultatywny (opcjonalny). Komputery wykorzystujemy we wszystkich szkołach, więc uczniowie i tak muszą mieć podstawowe informacje na ich temat. Podstawy programowania są włączone w nauczanie algebry i robotyki oraz nauk technicznych. Oczywiście, gdy ktoś wybierze informatykę jako przedmiot, musi się na tym skoncentrować i dobrze ją poznać.

— Jaką, Pańskim zdaniem, drogą powinna pójść komputeryzacja szkolnictwa w Polsce?

— Myślę, że dobry jest kierunek, w którym my zdążamy: traktowanie komputera jako podstawowego i najważniejszego narzędzia pomocniczego w nauczaniu i myśleniu w ogóle. Polecałbym drogę prowadzącą przez powstanie wysoce wyspecjalizowanego oprogramowania dydaktycznego i komercyjnego, a także wykorzystywanie w nauczaniu arkuszy elektronicznych, baz danych, edytorów tekstu itp. Wydaje się, że lepiej jest, gdy uczniowie poznają możliwości wykorzystania komputerów niż informatykę i komputery, choć i ta metoda ma z pewnością swoje zalety.

— Dziękuję za rozmowę.

Rozmawiał T.B. Mańk

PS. prof. Black gościł w Polsce w dniach 5—8 XI 1990 r. z okazji sympozjum poświęconego edukacji, zorganizowanego przez UW.

Uczniowie dzielą się na dwie grupy: tych, którzy fizyki nie lubią, i tych, którzy fizyki bardzo nie lubią. Istnieją co prawda wyjątki, ale fizyka z natury rzeczy zajmuje się ogólnymi prawami.

ROZPAD

Taki stan zmusza nauczycieli fizyki do wzmożonego wysiłku. Utrzymanie uwagi klasy wymaga od nich bądź efektownego machania rękami przy wyprowadzaniu zależności, wybuchowych doświadczeń, bądź jednego i drugiego. Można też wprowadzić na lekcję komputer (jest to w Polsce nowość mogąca utrzymać dzieci w stanie zaciekawienia) i przy jego pomocy nauczać.

Prezentowany obok program pozwala na ułatwienie pracy przy wprowadzaniu prawa rozpadu promieniotwórczego. W listopadzie 1990 roku została z jego wykorzystaniem przeprowadzona lekcja w warszawskim LO im. Modrzewskiego. Oto, jak ona wyglądała:

Lekcja składała się z trzech części: wstępnego pokazu (symulacji rzeczywistej), części teoretycznej będącej omówieniem wniosków z pokazu i komputerowej symulacji procesów statystycznych.

Pokaz był typowy dla tej lekcji: seria rzutów monet. Zakładamy, że „reszka" eliminuje monetę z gry, zostają te z „orłem". Wyniki przedstawiane były na tablicy w formie histogramu. Sugestie, że atomy można traktować jak monety (atom, który uległ przemianie, wypada z gry), uczniowie przyjęli bez oporów.

Z pomocą histogramu łatwo można było wprowadzić takie podstawowe pojęcia, jak stała rozpadu (jako prawdopodobieństwo rozpadu w jednostce czasu — tu oczywiście równe 1/2) czy czas połowicznego zaniku.

Na tym etapie zostało wprowadzone i rozwiązane równanie określające zanik próbki w czasie. Jednak rozwiązanie tego problemu (krzywa logarytmiczna) nie przystawało do narysowanego histogramu. Zwykle na tym etapie kończyłam wykład mówiąc, że dla większych próbek krzywa ta lepiej odzwierciedla dane teoretyczne, co uczniowie musieli przyjąć na wiarę. Teraz jednak mogłam pójść dalej i wykorzystać program komputerowy.

Program ma trzy opcje, z których pierwsza jest właściwie prezentacją na ekranie wykonanego dopiero co pokazu. Mamy szesnaście atomów i „ręcznie" losujemy stan rozpadu. Dokonujemy tego przez kolejne naciskanie klawisza. Atom, który się rozpadł, znika z ekranu. Po każdym odcinku czasu (w pokazie jest to kolejny rzut monetami, w programie losowanie przeprowadzone dla wszystkich atomów) rysowany jest histogram liczby pozostałych atomów. Procedura taka pozwala na przyswojenie przez uczniów faktu, że to, co dzieje się na ekranie, odzwierciedla rzeczywistość. Przeprowadzenie losowania dla różnych czasów połowicznego rozpadu pozwala dodatkowo zrozumieć różnice w historii bryłek atomów różnych pierwiastków.

Gdy uczniowie już zaakceptowali możliwość symulacji rozpadu jąder w przedstawiony sposób, mogłam pokazać im to, do czego właśnie konieczny był komputer: symulację rozpadu dla bryłki składającej się z dużej liczby atomów (w tym wypadku 10000).

Druga opcja programu — czyli właśnie symulacja bryłki prezentuje się graficznie w sposób zbliżony do pierwszej: tu też mamy atomy (prezentowane przez ciemne punkty, które znikają w miarę rozpadu), a po każdej jednostce czasu rysowany jest histogram. Łatwo wykazać, że fluktuacje, tak widoczne przy dziesięciu lub szesnastu atomach, przestają odgrywać rolę znaczącą. Ciemna na początku „bryłka" rozjaśniała się w trakcie kolejnych losowań (atomy wypadają z gry jak w pokazie z monetami), a histogram przypomina wykres. Ten fragment programu najbardziej zainteresował moich uczniów. Wielokrotne powtórzenia pozwalają na porównanie historii bryłki przy różnych czasach połowicznego rozpadu i podział atomów na długo- i krótkożyciowe. Ciąg logiczny wykładu i podobieństwo graficzne obu opcji spowodowały, że uczniowie uwierzyli w możliwość dokładnego odtworzenia histogramu przez krzywą logarytmiczną.

Na zakończenie lekcji zaprezentowałam (również na ekranie) przykładowe rozpady konkretnych pierwiastków. Program umożliwia wybór jednego z czterech lub wpisanie własnego pierwiastka. Tu już nie ma prezentacji atomów za pomocą punktów, a dane są przedstawione tak, jak w książce: za pomocą wykresu.

Po wybraniu pierwiastka widzimy wykres liczby atomów w zależności od czasu, a jednocześnie nałożony zostaje wykres teoretycznej funkcji, wyprowadzonej na początku lekcji. Zaobserwowane różnice dają możliwość wprowadzenia pojęcia fluktuacji.

Na wykresie zaznaczane były czasy, po jakich zostaje połowa pierwotnej liczby atomów. Czasy te można było porównać z danymi tablicowymi (co też zrobiliśmy), a wykres przypominał rysunek z podręcznika, z rozdziału poświęconego temu tematowi.

Program ma pewne błędy i ograniczenia: najważniejsze z nich to rozbieżności między krzywą teoretyczną a symulacją, pojawiające się dla pierwiastków o krótkim czasie połowicznego zaniku i brak możliwości nakładania na siebie wykresów pierwiastków o różnych czasach połowicznego rozpadu. Niewątpliwym zaś walorem jest łatwość obsługi. Nawet nie przygotowany do pracy z komputerem nauczyciel może go użyć na lekcji bez ujawnienia swej komputerowej niewiedzy i uatrakcyjnić (spójrzmy prawdzie w oczy) nudne dla uczniów lekcje fizyki.

E. Wojtania

Program „ROZPAD" przeznaczony jest do wspierania lekcji poświęconej statystycznemu prawu rozpadu. Wykorzystując go można unaocznić losowość rozpadu jąder atomowych i wykazać, że losowość ta prowadzi wprost do krzywej logarytmicznej.

Program posiada trzy opcje. W pierwszej z nich odbywa się losowanie dla każdego spośród 16 danych atomów. Losowanie rozpoczyna naciśnięcie klawisza [ENTER]. Jeśli wylosowana liczba jest większa od stałej rozpadu (która może być interpretowana jako prawdopodobieństwo rozpadu w jednostce czasu), atom nie rozpada się. Po losowaniu wszystkich atomów przedstawiany jest histogram liczby atomów w danej chwili. Losowanie powtarzane jest dziesięciokrotnie, co pozwala na omówienie charakteru rozpadu i tworzonego histogramu. Po ostatnim losowaniu należy jeszcze raz wcisnąć [ENTER], by wrócić do głównego MENU.

Opcja druga przedstawia w identyczny z zewnątrz sposób rozpad 10000 atomów. Ciemne punkty oznaczają atomy, które się nie rozpadły. Losowanie jest automatyczne i trwa do czasu wciśnięcia klawisza [ESC]. Jednocześnie tworzony jest (jak poprzednio) histogram liczby atomów.

Trzecia opcja przedstawia wykres zależności liczby atomów od czasu na tle teoretycznej zależności:

$$N=N_0e(-\lambda \cdot t);\ (N_0=10000)$$

wyprowadzonej na lekcji.

Możemy w niej wybrać jeden z czterech wprowadzonych pierwiastków z ich rzeczywistymi czasami połowicznego rozpadu lub wprowadzić dane innego. Na wykresie zaznaczane są również momenty przekroczenia połowy pierwotnej liczby atomów, co umożliwia wyjaśnienie reguł rządzących krzywą logarytmiczną.

Opcje wybierane są przez wciśnięcie klawisza z liczbą odpowiadającą jej numerowi.

Czas połowicznego rozpadu, który musimy podać na początku pierwszej i drugiej opcji, wpisujemy pamiętając, że część całkowitą oddzielamy od ułamkowej kropką, a kończymy klawiszem [ENTER]. Dobre wyniki daje wprowadzanie czasu rozpadu z zakresu 1 do 10 (ew. potęgę liczby dziesięć można traktować jako jednostkę).

Program kończymy przez wybranie liczby 0.

Uruchomienie programu nie jest możliwe bez kompilatora TURBO PASCAL, w wersji co najmniej 4.0. Użycie innych kompilatorów jest możliwe po niewielkich „kosmetycznych" przeróbkach związanych z używaniem trybu graficznego i funkcją **readkey**, która dotyczy czytania klawiatury.

Program wymaga obecności w aktywnej kartotece sterownika karty graficznej, w jaką wyposażony jest komputer (pliki z rozszerzeniem BGI dostarczane z pakietem TURBO PASCALA).

T.B.M.

```
{*************************************************************}
{**********   (C)  TM 1990  Warszawa   ***********************}
{*************************************************************}

uses crt, graph;

type
    pierwiastek = record                          {*DEFINICJA ATOMU*}
                        nazwa : string[15];       {*              *}
                        T12   : real;             {*CZAS POL. ROZP *}
                        jedn  : string[10];       {*jednostka osi  *}
                  end;

const
     dane:array[1..5] of pierwiastek=
                         ((nazwa:'URAN 238'   ;T12:4.5; jedn:'10^9  lat'),
                          (nazwa:'THOR 232'   ;T12:1.4; jedn:'10^10 lat'),
                          (nazwa:'NEPTUN 237';T12:2.2; jedn:'10^6  lat'),
                          (nazwa:'AKTYN 235' ;T12:7.1; jedn:'10^16 lat'),
                          (nazwa:'nieznany'  ;T12:1.0; jedn:'________'));

     TmpPierw : pierwiastek = (nazwa:'.........'; T12:1.0; jedn:'...');

procedure startGrafiki;                   {*URUCHOMIENIE TRYBU GRAFICZNEGO*}
var                                       {*NIEZBEDNE W TURBO PASCALU      *}
   GraphDriver,graphMode:integer;         {*NA KOMPUTERZE KLASY PC         *}

begin
     DetectGraph(graphDriver,GraphMode);
     InitGraph(Graphdriver,graphMode,'');
end;


function wybierzPierwiastek:byte;         {*FUNKCJA WYBIERAJACA PIERWIASTEK*}
var                                       {*SPOSROD PIECIU ZADEKLAROWANYCH *}
```

```
   poz:byte;                          (*       W OPCJI TRZECIEJ          *)
   i:integer;                         (* lambda - stala rozpadu          *)
                                      (* tau    - sredni czas zycia      *)
begin
     clrscr;
     writeln('nazwa':18,'T 1/2':20,'lambda':20, 'tau':10);
     for i:=1 to 5 do
         begin
              writeln;
              with dane[i] do
                   writeln(i:3, nazwa:15, T12:15:2, ' [',jedn:10,']',
                        ln(2)/T12:12:2, T12/ln(2):10:2);
         end;
     writeln; writeln;
     write('PODAJ NUMER PIERWIASTKA (LUB PODAJ DANE NIEZNANEGO)');
     repeat
          {$I-}                    (* ZABEZPIECZENIE PRZED PODANIEM *)
          readln(poz);             (*    NIEPRAWIDLOWYCH DANYCH     *)
          {$I+}
     until (IOresult=0) and (poz in [1..5]);
     case poz of
          1..4 : wybierzPierwiastek:=poz;
            5 : begin
                     with dane[5] do
                          begin
                               write('PODAJ NAZWE ');  readln(nazwa);
                               repeat
                                    {$I-}
                                       write('PODAJ T1/2 '); readln(T12);
                                    {$I+}
                               until IOresult=0;
                               write('JEDNOSTKA T1/2 ? ');readln(jedn);
                          end;
                     wybierzPierwiastek:=5;
                end;
     end;(*CASE*);
end;


procedure losowanie;
var
   bryla : array[ 1..4, 1..4] of boolean;
   i,j   : byte;
   T12   : real;                  (* PIERWSZA OPCJA PROGRAMU    *)

    czas  : byte;                  (* LOSOWANIE DLA 16 "ATOMOW" *)
    ilosc : byte;
    los   : real;
    znak  : char;



begin
     for i:=1 to 4 do
         for j:=1 to 4 do
             bryla[i,j]:=true;
     clrscr;
     repeat
          write('PODAJ T1/2 (losowanie dla 16 atomow) ');
          {$I-}
               readln(T12);
          {$I+}
     until IOresult=0;
     clrscr;
     writeln('  1 2 3 4'); writeln;
     for i:=1 to 4 do
         begin
              write(i,' ');                          (*RYSOWANIE OBRAZU*)
              for j:=1 to 4 do  write('X ');         (*"BRYLKI ATOMOW" *)
              writeln; writeln;
         end;
     writeln('T1/2    = ',T12:4:2); writeln;
     writeln('lambda = ',ln(2)/T12:4:2); writeln;
     writeln('tau     = ',T12/ln(2):4:2);
     czas:=0;
     while czas <= 10 do
          begin
               sound(400); delay(500); nosound;
               gotoxy(1,20); writeln('CZAS: ',czas);
               inc(czas);
               ilosc := 0;
               for i:=1 to 4 do
                   for j:=1 to 4 do           (*POWSTAWANIE I RYSOWANIE*)
                       if bryla[i,j] then     (*       HISTOGRAMU        *)
                          begin
                               inc(ilosc);
                               gotoxy(35+czas,20-ilosc);write('I');
                          end;

               (*LOSOWANIE*)




               for i:=1 to 4 do
                   for j:=1 to 4 do
                       if bryla[i,j] then
                          begin
                               gotoxy(1,17);
                               writeln('ATOM Z POZYCJI (',i,',',j,')');
                               randomize;
                               los := random;
                               write('wylosowano:',los:5:2);
                               if los < ln(2)/T12
                                  then
                                      begin
                                           writeln(' rozpada sie !!!');
                                           bryla[i,j]:=false;
                                           gotoxy(i*2+1,j*2+1); write(' ');
                                      end
                                  else writeln(' nie rozpada sie');
                               znak := readkey;
                               if znak=#27 then exit;
                          end;

          end;
     znak:=readkey;
end; (*LOSOWANIE*)



procedure rozpad;
var
   bryla : array[1..100,1..100] of boolean;
   T12   : real;
   lambda: real;                      (*OPCJA DRUGA PROGRAMU*)
   i,j   : integer;                   (*PROBKA 10000 "ATOMOW"*)
   znak  : char;                      (*lambda=LN(2)/T1/2    *)
   czas  : integer;
   ilosc : integer;
   strtmp: string[10];



begin
     clrscr;
     repeat
          clrscr;
          write('PODAJ T1/2 (losowanie dla 10000 atomow) ');
          {$I-}
               readln(T12);
          {$I+}
          until IOresult=0;
          lambda:=ln(2)/T12;
          randomize;
          startGrafiki;
          clrscr;                                   (*"PUDELKO" Z ATOMAMI*)
          line(0,0,0,101);      line(0,101,101,101);
          line(101,101,101,0); line(0,0,101,0);


          str(T12:4:2,strtmp)         ; outtextxy(1,120,' T1/2 = '+strtmp);
          str(ln(2)/T12:4:2,strtmp)   ; outtextxy(1,140,'lambda = '+strtmp);
          str(T12/ln(2):4:2,strtmp)   ; outtextxy(1,160,'   tau = '+strtmp);


     for i:=1 to 100 do
         for j:=1 to 100 do
             begin
                  bryla[i,j]:=true;
                  putpixel(i,j,1);


             end;
     czas:=0;
     ilosc:=10000;
     while czas < 200 do
           begin
                inc(czas);
                line(200+czas,150,200+czas,150 - ilosc div 80);
                for i:=1 to 100 do
                    for j:=1 to 100 do
                        if bryla[i,j]
                           then
                               if random < lambda then
                                  begin
                                       bryla[i,j]:=false;
                                       dec(ilosc);
                                       putpixel(i,j,0);
                                  end;



                if czas=200 then znak:=readkey;   (*STOP PO 200 OKRESACH  *)

                if keypressed                     (*LUB                    *)
                   then begin                     (*PO WCISNIECIU KLAWISZA*)
                             closegraph;exit;
                        end;
           end;
     closeGraph
end;(*ROZPAD*)


procedure przyklady;
var
   bryla : array[1..10000] of boolean;
   czas  : integer;
   i     : integer;
   pierw : pierwiastek;          (*OPCJA TRZECIA PROGRAMU*)
   lambda: real;                 (*PRZYKLADY ROZPADU     *)
   znak  : char;
   ilosc : integer;
   tmp   : integer;
   polowa: integer;
   tmpstr: string[20];



begin
     pierw:=dane[wybierzPierwiastek];
     for i:=1 to 10000 do bryla[i]:=true;
     lambda:=ln(2)/pierw.T12;
     startGrafiki;                                  (*RYSOWANIE EKRANU*)
     line(18,2,20,0);       line(20,0,20,170);
     line(20,170,250,170);  line(250,170,248,172);
     outtextxy(1,1,'N');
     outtextxy(200,180,pierw.jedn);
     line(20,170-10000 div 80,17,170-10000 div 80);



     line(200,20,220,20);outtextxy(230,15,pierw.nazwa);
     circle(210,30,2); outtextxy(230,25,'N=N0exp(-lambda*t');
     czas:=0;
     ilosc:=10000;
     polowa:=ilosc;
     randomize;
     while czas < 40 do
           begin
                if keypressed then begin
                   closeGraph;
                   exit
                  end;
                tmp:=ilosc;
                circle(20+czas*5,170-round(10000*exp(-lambda*czas)) div 80,2);
                for i:=1 to 10000 do
                    if bryla[i] then
                       if random<lambda    (*LOSOWANIE*)
                          then
                              begin
                                   dec(ilosc);
                                   bryla[i]:=false;
                              end;
                line(20+czas*5,170,20+czas*5,165);
                inc(czas);
                if ilosc < (polowa div 2)
                   then
                       begin                        (*RYSOWANIE KRZYWEJ TEORETYCZNEJ*)
                            setlinestyle(dottedln,0,1);
                            line(20+czas*5,170,20+czas*5,170-ilosc div 80);
                            line(20,170-ilosc div 80, 20+czas*5,170-ilosc div 80);
                            setlinestyle(solidln,0,1);
                            polowa:=ilosc;
                       end;

                line(20+(czas-1)*5,170-tmp div 80, 20+czas*5, 170-ilosc div 80);
                   end;
           end;
     znak:=readkey;  (*STOP PO WCISNIECIU KLAWISZA*)
     closeGraph;
end;

var
   znak:char;



begin
     repeat
          clrscr;
          writeln;
          writeln('1 - losowanie nieautomatyczne dla 16 atomow'); writeln;
          writeln('2 - losowanie automatyczne dla 10000 atomow'); writeln;
          writeln('3 - przyklady konkretnych rozpadow');          writeln;
          writeln('0 - KONIEC');
          znak:=readkey;
          case znak of
               '1' : losowanie;
               '2' : rozpad;
               '3' : przyklady;
          end; (*CASE*)
     until znak='0';
end.
```

JĘZYK "C"

DLA NAJMŁODSZYCH CZYLI DZIESIĄTY WIECZÓR Z CZARNOKSIĘŻNIKIEM

Dziś Czarnoksiężnik postanowił dokończyć pracę nad komputerowym notatnikiem. W tym celu musiał poznać „urodę" kompilatora „Deep Blue C", który miał zamiar użyć. Można go zastosować do poważniejszych prac inżynierskich lub informatycznych oraz do tworzenia gier.

Czarnoksiężnik zmodyfikował bibliotekę podprogramów, która jest standardowym wyposażeniem kompilatora DBC. Mógł to zrobić, gdyż biegle władał „zaklęciami" języka „C". Wszystkie podprogramy napisane są z użyciem angielskich słów kluczowych. Poprzestając na znajomości polskich słów, skazujemy się na brak możliwości czytania i modyfikowania programów napisanych w „normalnym" języku. Dziś, kiedy biegle władamy polskimi „zaklęciami", nie powinno nas przerażać działanie z użyciem angielskich słów kluczowych. W tym celu przytoczymy cały zbiór ‹predef. h› i od tej pory będziemy „czarowali" po angielsku.

```
#define tekst char
#define majster main()
#define pisz printf
#define czytaj scanf
#define rzeczywista float
#define calkowita int
#define precyzyjna double
#define dluga long
#define powtarzaj for
#define robgdy while
#define wykonuj do
#define dopoki while
#define gdy if
#define przeciwnie else
#define cdn continue
#define wybierz switch
#define klucz case
#define gotowe break
#define reszta default
#define teraz goto
#define oddaj return
#define czytajwiersz gets
```

Zamieszczony poniżej opis powstał na podstawie informacji zapisanych w zbiorze DOC.COM, który rezyduje na dyskietce z kompilatorem DBC. Oryginalny opis angielski można uzyskać wywołując ten zbiór jako dyrektywę.

Kompilator DBC stosuje pewien standard rozszerzeń nazw zbiorów. Wymieńmy je wszystkie:

NAZWA. C — wersja źródłowa programu „NAZWA" w „C",

NAZWA. CCC — wersja biblioteczna programu „NAZWA",

NAZWA. LNK — zbiór zawierający informacje dla kompilatora, które programy w wersji bibliotecznej mają stanowić cały program,

NAZWA. COM — zbiór wynikowy po kompilacji. Komputer może go wykonywać jak każdy inny program binarny,

PRASM.ASM — program PRASM w assemblerowej wersji źródłowej,

PRASM.OBJ — program biblioteczny powstały z assemblerowej wersji źródłowej programu PRASM.ASM.

Stosując ten standard można w odpowiednich momentach pisać tylko same nazwy zbiorów (bez rozszerzeń). Kompilator sam dołączy do nich stosowne, zgodne ze standardem rozszerzenie.

Pierwsza strona dyskietki zawiera dodatkowo zbiory:

CC.COM — program translujący wersję źródłową w języku „C" (.C) na wersję biblioteczną (.CCC),

CLINK.COM — program łączący wersje biblioteczne programów (.CCC i .OBJ) w jeden program binarny (.COM) według listy zamieszczonej w zadeklarowanym zbiorze (.LNK),

DOC.COM — program, dzięki któremu można uzyskać oryginalny opis w języku angielskim.

Zbiory te stanowią główną część kompilatora DBC. Mając je do dyspozycji, możemy pisać już w języku „C", ale będzie to praca żmudna.

Oto zbiory, które ułatwią i przyśpieszą pracę:

AIO.CCC
GRAPHICS.CCC
PMG.CCC
PRINTF.CCC
DBC.OBJ
— pięć bibliotek podprogramów

AIO.C
GRAPHICS.C
PMG.C
PRINTF.C
— cztery wersje źródłowe podprogramów

Do kompletu dołączono dwa przykładowe zestawy zbiorów potrzebnych do utworzenia programów binarnych:

X.C
X.CCC
X.LNK
X.COM
— przykładowy program ilustrujący współpracę z dyskiem

BOUNCE.C
BOUNCE.CCC
BOUNCE.LNK
BOUNCE.COM
— program ilustrujący tworzenie grafiki PMG.

Zanim poznamy obsługę kompilatora DBC, trzeba wymienić jego możliwości.

Przede wszystkim akceptuje on niepełny zestaw typów zmiennych. Do znanych mu typów należą:

1. **char** (tekst), **int** (całkowita)
2. wskaźniki do zmiennych typu **char** i **int**
3. jednowymiarowe tablice zmiennych, których typ wymieniono w punktach 1. lub 2. Wobec zmiennych strukturalnych, unii oraz tablic wielowymiarowych nasz kompilator zachowuje się tak jak analfabeta wobec książki.

Kilka braków można sztucznie uzupełnić. Tablice wielowymiarowe można zamienić na jednowymiarowe, zmieniając kilka indeksów na jeden, według odpowiedniego wzoru. Dla dwu wymiarów ma on postać:

I = i1*dim(i2) + i2

Dla trzech wymiarów ma on postać:

I = i1*dim(i2)*dim(i3) + i2*dim(i3) + i3

gdzie **dim(i)** jest maksymalną wartością indeksu **i**.

Dla tablic o większej liczbie indeksów (wymiarów) wzór jest analogiczny. Dokładnie tak samo porządne kompilatory przeliczają indeksy tablic wielowymiarowych na adres elementu w pamięci.

Podczas poprzednich spotkań poznaliśmy liczby (zmienne) typu float (rzeczywiste, zmiennoprzecinkowe). Kompilator DBC potrafi działać na tych obiektach, czyni to jednak w sposób niejawny. Temat ten omówimy później.

Brak zmiennych strukturalnych można uzupełnić tworząc tablicę wskaźników do zmiennych będących polami struktury. Oczywiście, dostęp do poszczególnych pól będzie możliwy nie dzięki operatorowi selekcji (kropka), ale poprzez indeksowany element tablicy potraktowany jako wskaźnik do zmiennej. Niestety, autor tego cyklu nie znalazł sposobu tworzenia dynamicznie takich „struktur". Dzięki temu zbędny okazał się operator „sizeof".

Wcześniej wspomniane było, że podprogramy w DBC mogą zwracać wartość tylko typu **int** (całkowita). Kolejnym ograniczeniem jest to, że liczba argumentów podprogramu może wynosić maksymalnie 126. Oprócz tego zmienne typu **char** (tekst) można traktować jako jednobajtowe liczby bez znaku. Liczby te mają zakres od 0 do 255.

Należy pamiętać też o tym, że długość jednej linii wersji źródłowej nie może przekroczyć 79 znaków oraz, że łańcuchy znaków nie mogą być przedłużane do następnej linii logicznej.

Ze względu na brak niektórych specyficznych dla „C" znaków, wprowadzono zastępcze kombinacje. Tak więc, zamiast nawiasów klamrowych wprowadzono odpowiednio $(i $), natomiast zamiast operatora uzupełnienia jedynkowego ~ (tylda) wprowadzono $-.

Warto jeszcze wspomnieć o jednym, niestandardowym rozszerzeniu, w jakie został wyposażony kompilator DBC. Jest nim instrukcja **ASM adres**, która powoduje skok pod wskazany, bezwzględny adres w pamięci. Nie należy mylić jej z podprogramem **USR (...)**, który wchodzi w skład jednej z bibliotek i zostanie omówiony w następnym odcinku.

Przystąpimy teraz do tworzenia programu. Najpierw przy pomocy dowolnego edytora kreujemy tekst programu źródłowego. Ważne jest, żeby tekst ten został zapisany na dysku jako zbiór o nazwie z rozszerzeniem .C. Tworzymy także zbiór, zawierający nazwy wszystkich bibliotek, które utworzą nasz program. Jego nazwa może być identyczna jak tego, który zawiera program źródłowy, lecz powinna mieć standardowe rozszerzenie. LNK. W nazwach wyspecyfikowanych zbiorów bibliotecznych można pominąć standardowe rozszerzenie .CCC.

Teraz możemy utworzyć wersję biblioteczną. W tym celu należy uruchomić program **CC.COM.** Poinformuje on nas, kto jest jego autorem oraz zapyta o nazwę zbioru do kompilacji.

Jeśli nazwa kompilowanego zbioru ma standardowe rozszerzenie (.C) to można je pominąć; w takim przypadku program sam wie, jakiego zbioru ma szukać. Na tym polega wygoda stosowania standardowych rozszerzeń nazw.

Po odnalezieniu zbioru na dysku program poinformuje nas, że przetworzy go na zbiór o identycznej nazwie, ale ze standardowym rozszerzeniem .CCC. Każdy inny komunikat świadczy o kłopotach.

Następuje translacja. W trakcie mogą pojawić się informacje o ewentualnych błędach.

Po zakończeniu translacji program ponownie zapyta o zbiór do kompilacji. Naciśnięcie klawisza ‹RETURN› bez podania nazwy zbioru spowoduje zakończenie pracy programu **CC** i powrót do DOS-u.

Teraz kolej na program **CLINK.COM.** Po wywołaniu on także przedstawi się po czym zapyta:

Link program, Duplicate file or Quit.

Trzeba wcisnąć jedną z trzech liter:

L — gdy chcemy połączyć biblioteki w jeden program,

D — gdy chcemy skopiować gotowy program,

Q — gdy chcemy zakończyć działanie programu CLINK.

Wybór potwierdzamy wciśnięciem klawisza <RETURN>.

W przypadku opcji **L** program zapyta o nazwę zbioru zawierającego listę bibliotek, które mają utworzyć program. Następnie poinformuje nas, że programy biblioteczne z listy zawartej w zbiorze **NAZWA.LNK** po połączeniu, utworzą program **NAZWA.COM**.

Na koniec kilka uwag praktycznych. Przed przystąpieniem do pracy radzimy przygotować specjalną, roboczą dyskietkę. Proponujemy umieścić na pierwszej stronie system operacyjny **DOS XL** ver.2.35L, wszystkie zbiory biblioteczne oraz programy **CC.COM** i **CLINK.COM**. Wykluczy to możliwość zniszczenia ostatniej kopii tych zbiorów. Ze względu na możliwość przetwarzania wsadowego proponujemy na dyskietkę roboczą zapisać także ciało dyrektywy zewnętrznej **DO** oraz zbiór **STARTUP.EXC**, który będzie zawierał wywołania poszczególnych programów. Teraz kompilacja będzie przebiegała prawie automatycznie. Jeśli inteligentnie zdefiniujecie zbiór **.EXC** to uwierzycie w starą prawdę, że do roboty są komputery — człowiek jest do myślenia.

Mieczysław Płacheta

KUPIŁEM KOMPUTER:

I CO DALEJ?

Takie pytanie przewija się w bardzo wielu listach przychodzących do naszej redakcji. Kłopoty zaczynają się niemal natychmiast po wyjściu ze sklepu.

Po rozpakowaniu zakupionego sprzętu stwierdzamy, że w pudełku oprócz komputera znajduje się zasilacz, przewód i instrukcja. Podobna jest także zawartość pudełka ze stacją dysków, lecz ponadto jest tam jeszcze dyskietka z systemem operacyjnym. Natomiast w komplecie z magnetofonem znajduje się tylko instrukcja.

Łączenie

Pierwszym problemem jest zwykle połączenie i podłączenie systemu. Czynność ta jest dokładnie opisana w instrukcji, lecz z moich obserwacji wynika, że instrukcja obsługi jest zupełnie zbędna — wszyscy tak dobrze znają się na komputerach, iż niemal nikt jej nie czyta. Zamiast od tego zacząć, lepiej przecież zrobić to po swojemu, a potem ponarzekać na firmę, sklep i wszystkich świętych.

Jak zatem powinno przebiegać prawidłowe postępowanie? Ustawiamy komputer w wybranym miejscu w pobliżu telewizora lub monitora i przyłączamy do niego wszystkie niezbędne przewody, które łączą go z zasilaczem, telewizorem (lub monitorem) i magnetofonem lub stacją dysków (w tym przypadku dodatkowo jeszcze przewód zasilający stację) oraz ewentualnie z joystickiem. Wtyczki wszystkich przewodów są tak skonstruowane, że nie można ich pomylić — niewłaściwe połączenie jest więc wykluczone. Jeżeli jakaś wtyczka pasuje do więcej niż jednego gniazda, to oznacza, że można ją włożyć do dowolnego z nich.

Prawdziwy problem może stanowić tylko połączenie z telewizorem (monitorem). Konieczne jest tu pewne wyjaśnienie dotyczące zasady działania telewizora. Składa się on z dwóch podstawowych części: odbierającej sygnał (obraz i dźwięk) i wyświetlającej. W uproszczeniu można powiedzieć, że telewizor składa się z odbiornika i monitora (wiadomo teraz, jaka jest różnica pomiędzy telewizorem i monitorem — ten drugi nie ma odbiornika). W pudełku komputera znajduje się przewód łączący komputer z odbiornikiem telewizora. Należy go przyłączyć do gniazda antenowego telewizora (ZAMIAST anteny).

Komputery Atari umożliwiają jednak ominięcie odbiornika i korzystanie bezpośrednio z telewizora. Uzyskuje się w tej sposób lepszą jakość obrazu i dźwięku. Konieczne jest w tym celu zdobycie odpowiedniego przewodu, gdyż nie ma go w zestawie. Przewody takie można kupić w sklepach ze sprzętem radiowo-telewizyjnym, lecz najpierw trzeba sprawdzić, jakie wtyczki są potrzebne. Do gniazda w komputerze potrzebna jest standardowa wtyczka DIN, powszechnie stosowana w krajowym sprzęcie elektroakustycznym. W telewizorze (lub monitorze) przewód ten przyłącza się do gniazda monitorowego. Jest ono oznaczone jako MONITOR, VIDEO lub AUDIO IN/VIDEO IN. Możliwe są tu cztery rodzaje wtyczek: DIN, SCART (Eurozłącze), dwa wtyki CINCH lub CINCH i BNC. Najłatwiej rozpoznać złącze SCART, gdyż jest ono szerokie i ma dwa rzędy styków. CINCH jest bardzo podobny do gniazda antenowego w komputerze, a DIN jest identyczny z gniazdem MONITOR. Złącze BNC jest okrągłym złączem bagnetowym o średnicy pośredniej między DIN i CINCH. Na przykład, 14-calowe telewizory SANYO mają podwójny wtyk CINCH, a 21-calowe wtyki CINCH i BNC. Natomiast polski telewizor Westa ma złącze SCART. Podobne złącza występują w monitorach, lecz niektóre z nich mają tylko gniazdo RGB (przypominające wtyczkę joysticka) — do takiego monitora nie można przyłączyć komputera Atari.

Przy okazji odpowiem na jeszcze jedno pytanie, które coraz częściej pojawia się w listach. Sygnał przekazywany przez komputer z gniazda antenowego jest technicznie identyczny z sygnałem odbieranym z anteny, zaś sygnał z wyjścia monitorowego jest identyczny z takim sygnałem otrzymywanym z telewizora (przez wyjście video). Możliwe jest więc zapisanie na kasecie magnetowidowej przebiegu odtwarzanego programu lub stworzenie własnej czołówki do filmu nagranego na tej kasecie. Niestety, nie można mieszać tych sygnałów (w prosty sposób), więc czołówka musi być nagrana przed filmem, gdyż inaczej skasuje jego część.

Programy

Następnym problemem jest wczytanie programu z kasety lub dyskietki. Pierwszymi programami są zwykle gry, które wczytują się i uruchamiają automatycznie po włączeniu komputera. Dla zrozumienia sposobu ich odczytu wystarczy wiedzieć, jakie zadanie spełnia wciśnięcie klawiszy konsoli (rząd pięciu wydzielonych klawiszy) przy włączaniu zasilania. Klawisz ‹OPTION› służy do wyłączenia wewnętrznego Basica — jest to konieczne w przypadku prawie wszystkich programów napisanych w języku maszynowym, czyli dla większości programów.

Normalnie komputer próbuje odczytywać program ze stacji dysków — dlatego musi ona być włączona PRZED komputerem. Wciśnięcie klawisza ‹START› sygnalizuje, że odczyt ma być wykonany z magnetofonu zamiast ze stacji. Komputer sygnalizuje wówczas pojedynczym buczeniem konieczność wciśnięcia klawisza ‹PLAY› w magnetofonie. O wykonaniu tej czynności należy powiadomić komputer przez uderzenie dowolnego klawisza.

Podczas odczytu programu z magnetofonu niekiedy występują błędy (przy korzystaniu ze stacji przypadki takie są bardzo rzadkie). Magnetofon jest urządzeniem bardzo mało dokładnym i często komputer nie rozpoznaje, że wystąpił błąd. Jedynym sposobem skontrolowania poprawności odczytu jest słuchanie dźwięku emitowanego podczas transmisji. Już po kilku próbach łatwo odróżnia się dźwięk poprawny od wskazującego błąd. Trzeba w takim razie powtórzyć całą operację odczytu. Jeżeli na skutek błędu na ekranie pojawił się SELF-TEST, to nie jest konieczne wyłączanie komputera. Trzeba wtedy wcisnąć odpowiednie klawisze (zwykle ‹START› i ‹OPTION› i na krótko nacisnąć ‹RESET› — efekt będzie taki sam jak po wyłączeniu i ponownym włączeniu komputera.

Drugą grupę stanowią programy, które trzeba odczytać i uruchomić „ręcznie" (zalicza się tu także większość programów przepisanych z „Bajtka"). W przypadku magnetofonu trzeba dla ich uruchomienia włączyć komputer (nie wciskając żadnych klawiszy), a po ukazaniu się napisu **READY** wpisać z klawiatury **CLOAD** i nacisnąć ‹RETURN›. Odczyt programu przebiega w sposób opisany wcześniej, aż do ponownego komunikatu **READY**. Teraz należy napisać **RUN** i ponownie nacisnąć ‹RETURN›.

Odczyt takiego programu z dyskietki jest nieco bardziej skomplikowany. Przede wszystkim trzeba pamiętać, aby w chwili włączania komputera w stacji dysków znajdowała się dyskietka zawierająca DOS, który umożliwia komunikację między komputerem i stacją. Po wyświetleniu napisu **READY** wpisuje się polecenie **RUN „D:nazwa"** i naciska ‹RETURN›. „nazwa" oznacza tu nazwę, pod którą

program został zapisany na dyskietce. Sposób nadawania nazw poszczególnym programom jest opisany w instrukcji obsługi stacji i w licznych książkach wymienionych w końcowej części tego artykułu.

Wspomniałem wcześniej o odczytywaniu programów przepisanych z „Bajtka". A jak je zapisać? Operacja zapisu jest zbliżona do operacji odczytu — inne są tylko polecenia, które trzeba podać komputerowi. Dla magnetofonu jest to **CSAVE** (w skrócie **CS.**), a dla stacji dysków — **SAVE „D:nazwa"** (w skrócie **S.** itd.). Przy zapisie na magnetofon konieczność wciśnięcia klawiszy ‹RECORD› ‹PLAY›, jest sygnalizowana podwójnym dźwiękiem, co potwierdza się przez uderzenie dowolnego klawisza w komputerze.

Powyższe informacje dotyczą jedynie programów napisanych w Atari Basic. Jeżeli chcemy przepisać program w innym języku (Turbo Basic, Action!, Pascal, Logo itd.), to konieczne jest „nauczenie" komputera tego języka. Polega to na wczytaniu odpowiedniego programu, co wykonuje się identycznie jak w przypadku gier, gdyż i gra, i język są programami komputerowymi. Jedynymi wyjątkami są programy (języki i gry) umieszczone w modulu ROM (cartridge'u). Moduł taki wkłada się do odpowiedniego gniazda w komputerze PRZED jego włączeniem, a po włączeniu program zgłasza się automatycznie zamiast Basica.

Literatura

Podane wyżej informacje można znaleźć w wielu książkach wydanych w Polsce. Przede wszystkim jednak polecam użytkownikom Atari nowe pismo — wydawany przez Spółdzielnię „Bajtek" dwumiesięcznik „Moje Atari", który jest w pewnym stopniu kontynuacją wcześniejszych dwóch wydań specjalnych „Tylko o Atari". Natomiast dla osób zainteresowanych wyłącznie grami komputerowymi Spółdzielnia „Bajtek" wydaje specjalne nowe pismo „Top Secret".

Wróćmy jednak do książek. Podstawową literaturą każdego posiadacza Atari powinna być seria książek poświęconych tym komputerom i wydanych przez SOETO. Jest to przede wszystkim przeznaczony dla każdego „Poradnik użytkownika Atari XL/XE" oraz podręcznik dla początkujących programistów — „Podstawy programowania w Atari Basic". Dla bardziej doświadczonych programistów przewidziane są książki „Poradnik programisty Atari" i „Języki Atari" (dwie części). Zaawansowani programiści, piszący programy w języku maszynowym i „grzebiący" w systemie operacyjnym komputera znajdą dokładny jego opis w czterech książkach pod wspólnym tytułem „Mapa pamięci Atari XL/XE". Wymienione książki bywają w różnych księgarniach na terenie Polski. Można je również nabyć (bez marży księgarskiej — o 20% taniej) w Wydawnictwie SOETO w Warszawie.

Wojciech Zientara

INSTRUKCJA INACZEJ

Przy tworzeniu programu stajemy często przed problemem wprowadzenia danych w oknie ekranowym.

Zwykła instrukcja INPUT w Basicu powoduje odczytanie, oprócz wprowadzonego tekstu, także części ramki ograniczającej okno.

Prezentowana procedura — znajdująca się w programie demonstrującym jej działanie — umożliwia ograniczenie liczby danych odczytywanych z ekranu, a więc pozwala na odczyt z okna ekranowego. Poza wyznaczonym polem odczytu zawartość ekranu nie ulega zmianie. Można jednak dzięki niej odczytywać tylko dane tekstowe. W przypadku wartości liczbowych konieczna jest więc późniejsza zamiana za pomocą funkcji VAL.

Wywołanie procedury następuje przez podanie zastępującej INPUT instrukcji

I=USR(39424,ADR(A$),KOL,DLUGOSC, WIERSZ)

gdzie:

ADR(A$) — adres zmiennej tekstowej, do której będzie wpisany wprowadzany tekst (odpowiada to zwykłej instrukcji INPUT A$);

KOL — numer kolumny ograniczającej pole odczytu danych z lewej strony, czyli wielkość lewego marginesu pola (od 0 do 39);

DLUGOSC — długość pola odczytu (od 0 do 40);

WIERSZ — numer wiersza zawierającego pole odczytu (0–23).

Dodatkowym efektem działania procedury jest miganie kursora podczas wprowadzania danych. Powyższa procedura nie jest relokowalna i zajmuje obszar poniżej pamięci obrazu dla trybu zero. Przy zmianie trybu graficznego można więc zniszczyć niechcący procedurę.

Andrzej Zieliński

```
IF  10 REM Input
CL  15 REM Andrzej Zielinski
AJ  20 REM Copyright (c) Bajtek
BJ  25 REM
KD  30 DIM A$(40):A$(40)=" "
DH  40 FOR I=39424 TO 39851:READ A:POKE I,
    A:NEXT I
VD  50 KOL=7:DLUGOSC=10:WIERSZ=15
FN  60 POSITION KOL-1,WIERSZ:? CHR$(27);CH
    R$(127):POSITION KOL+DLUGOSC,WIERSZ:?
    CHR$(27);CHR$(126)
IQ  70 A=USR(39424,ADR(A$),KOL,DLUGOSC,WIE
    RSZ)
VX  80 ? :POKE 752,0:? A$(1,10)
FI  90 DATA 104,104,133,204,104,133
JP  100 DATA 203,104,104,133,82,104
BB  110 DATA 104,56,233,1,133,206
CW  120 DATA 24,101,82,133,83,104
YK  130 DATA 104,133,84,141,240,2
QD  140 DATA 165,82,133,85,169,30
VN  150 DATA 32,122,155,169,159,32
IO  160 DATA 122,155,169,31,32,122
AY  170 DATA 155,169,159,32,122,155
EE  180 DATA 142,240,2,169,110,141
CX  190 DATA 40,2,169,155,141,41
AW  200 DATA 2,169,2,141,26,2
RN  210 DATA 169,12,162,16,157,66
LS  220 DATA 3,32,86,228,169,3
BU  230 DATA 157,66,3,169,169,157
YP  240 DATA 68,3,169,155,157,69
LQ  250 DATA 3,169,4,157,74,3
RT  260 DATA 32,86,228,76,45,155
RO  270 DATA 169,7,162,16,157,66
HQ  280 DATA 3,169,0,157,72,3
MK  290 DATA 157,73,3,32,86,228
EE  300 DATA 133,205,169,0,141,26
KW  310 DATA 2,165,205,201,155,208
ZL  320 DATA 44,165,83,141,240,2
BO  330 DATA 133,85,160,0,152,72
HQ  340 DATA 169,31,32,122,155,104
GY  350 DATA 168,165,93,141,250,2
JZ  360 DATA 32,123,247,145,203,200
XJ  370 DATA 165,85,197,83,208,230
SV  380 DATA 169,2,133,82,169,39
PQ  390 DATA 133,83,96,201,125,208
EV  400 DATA 3,76,143,155,201,28
IN  410 DATA 240,107,201,29,240,103
KB  420 DATA 201,156,208,3,76,143
MW  430 DATA 155,201,157,208,3,76
SA  440 DATA 143,155,201,255,208,32
DI  450 DATA 56,165,83,229,85,168
PO  460 DATA 240,19,136,240,8,177
TW  470 DATA 94,200,145,94,136,208
UY  480 DATA 245,165,93,200,145,94
SD  490 DATA 169,0,168,133,93,76
GY  500 DATA 45,155,201,127,208,11
DJ  510 DATA 165,85,197,83,208,95
RB  520 DATA 169,31,76,99,155,201
AT  530 DATA 254,208,52,56,160,0
MK  540 DATA 165,83,229,85,240,13
ZY  550 DATA 133,205,200,177,94,136
YZ  560 DATA 145,94,200,196,205,208
WO  570 DATA 245,169,0,145,94,168
UF  580 DATA 177,94,133,93,76,45
BJ  590 DATA 155,169,31,141,240,2
DT  600 DATA 32,122,155,169,0,141
RC  610 DATA 240,2,169,30,76,99
FU  620 DATA 155,165,85,197,83,208
CD  630 DATA 28,165,205,201,30,240
OB  640 DATA 24,201,31,240,20,201
KO  650 DATA 126,240,16,32,219,241
YW  660 DATA 41,127,133,93,160,0
WO  670 DATA 145,94,76,45,155,165
QM  680 DATA 205,32,122,155,169,15
HS  690 DATA 141,26,2,76,108,154
YL  700 DATA 177,94,73,128,145,94
IK  710 DATA 169,15,141,26,2,96
EX  720 DATA 72,169,11,162,0,157
KD  730 DATA 66,3,169,0,157,72
ZI  740 DATA 3,157,73,3,104,32
VT  750 DATA 86,228,96,165,83,133
JG  760 DATA 85,169,31,32,122,155
HW  770 DATA 160,0,152,133,93,145
CA  780 DATA 94,200,196,206,48,249
SG  790 DATA 145,94,76,102,155,75
MC  800 DATA 58,155
```

MIGAJĄCY KURSOR

Wielokrotnie już był poruszany temat migania kursora. Chciałbym pokazać jeszcze jedno rozwiązanie tego problemu — procedurę włączaną z poziomu Basica.

Procedura ta jest zawarta w wydrukowanym obok programie demonstracyjnym. Posiada ona dwa warianty uruchomienia: dla normalnego redagowania treści programu i dla wprowadzania danych do programu za pomocą instrukcji INPUT. W porównaniu z poprzednic prezentowanymi rozwiązaniami nie pozostawia ona na ekranie śladów i nie powoduje migania znaków wyświetlanych w negatywie.

Miganie kursora jest uruchamiane instrukcją

```
PI  10 REM Migajacy kursor
CL  15 REM Andrzej Zielinski
AJ  20 REM Copyright (c) Bajtek
BJ  25 REM
EC  30 FOR I=1536 TO 1596:READ A:POKE I,A:
    NEXT I
WF  40 A=USR(1536,0):REM miganie kursora w
     trakcie edycji programu
AD  50 STOP
OJ  60 A=USR(1536,1):? "a=";:INPUT A:REM m
    iganie kursora podczas wprowadzania da
    nych-komenda INPUT
US  70 A=USR(1536,0)
FP  80 DATA 104,186,138,56,233,10
SI  90 DATA 133,203,104,104,240,4
HJ  100 DATA 198,203,198,203,169,26
AH  110 DATA 141,40,2,169,6,141
CD  120 DATA 41,2,169,15,141,26
IN  130 DATA 2,186,228,203,208,24
EA  140 DATA 165,34,201,5,208,18
GI  150 DATA 165,47,201,155,240,12
YK  160 DATA 165,35,201,1,208,6
YT  170 DATA 177,94,73,128,145,94
NF  180 DATA 96
```

I=USR(1536,I)

gdzie I musi mieć wartość 0 dla redagowania programu, a wartość 1 dla wprowadzania danych przez INPUT. Procedura ta jest relokowalna i może być umieszczona w innym dowolnie wybranym miejscu pamięci.

Andrzej Zieliński

Każdy użytkownik Atari, który zapoznał się z grafiką graczy i pocisków, zdaje sobie sprawę, jak szerokie może być jej zastosowanie. Niestety, Atari Basic nie został wyposażony w instrukcje obsługi duszków. Niedopatrzenie to popełnił również twórca Turbo Basica XL.

Wady te w znacznym stopniu eliminuje przedstawiany program. Aby uzyskać duszka na ekranie, należy:

1. Wczytać program "PLAYER GRAPHICS" (listing 1) i uruchomić go instrukcją RUN.
2. Uruchomić własny program zawierający instrukcje, które wykorzystują grafikę graczy i pocisków.

Działanie programu polega na umieszczeniu w tablicy urządzeń systemu operacyjnego dodatkowego urządzenia o nazwie "A:". Urządzenie to obsługuje grafikę graczy i pocisków. Aby wykorzystać to urządzenie, należy najpierw przygotować dane duszka umieszczone w zmiennej tekstowej. Następnie można otworzyć IOCB dla urządzenia "A:" przy użyciu instrukcji.
OPEN #N,A,B,"A:":GRAPHICS C:POKE 559,D
gdzie:

N jest numerem IOCB, który będzie wykorzystany dla P/MG (1—5);
A oznacza rozdzielczość i może przyjmować wartości: 0 dla rozdzielczości jednowierszowej i 1 dla dwuwierszowej (nie ma to nic wspólnego z nazwą urządzenia "A:", którą zawsze wpisujemy jako literę);
B jest związane z wybranym trybem graficznym i musi mieć wartość 0 dla trybów znakowych lub 1 dla trybów bitowych;
C jest numerem trybu graficznego, w którym pracuje cały obraz;
D jest niezbędne do prawidłowej pracy procedur duszków i przyjmuje wartości 42 dla rozdzielczości dwuwierszowej i 58 dla jednowierszowej.

Po tych przygotowaniach można już w bardzo prosty sposób stosować duszki. Wystarczy napisać bowiem

POSITION X,Y:? #N;NR;A$

gdzie X i Y to współrzędne duszka, N jest numerem IOCB (jak wyżej), NR określa numer gracza (0—3), A$ zaś jest zmienną tekstową, która zawiera dane tego gracza. Wyświetlenie duszka w innym miejscu powoduje automatycznie wymazanie go na starej pozycji. Bardzo łatwo uzyskać w ten sposób ruch obiektu (także pionowy!). Jest jednak jeden szkopuł. Wymazywanie obszaru pamięci przeznaczonego dla gracza, przed jego ponownym wyświetleniem, powoduje nieprzyjemne miganie podczas ruchu obiektu. Można temu łatwo zapobiec wpisując

POKE 40744,96

Trzeba jednak pamiętać o samodzielnym wymazywaniu duszka na starej pozycji (dla ruchu pionowego). Można to uzyskać przez dopisanie kilku zer na końcach jego danych. Pierwotny tryb pracy jest przywracany przez instrukcję

POKE 40744,166

Przykładem zastosowania tej techniki oraz realizującej ją procedury jest gra "MINI ARKANOID" (listing 2). Aby uzyskać prawidłowe jej działanie, trzeba najpierw wczytać i uruchomić "PLAYER GRAPHICS", a dopiero potem właściwą grę.

Michał Skowroński

LISTING 1

```
    0 REM PLAYER GRAPHICS
GX  1 REM Michal Skowronski
OI  2 REM (c) 1990, Bajtek
HZ  3 REM
NI  10 POKE 106,158:GRAPHICS 0
WY  20 FOR A=40448 TO 40754
GO  30 READ X:POKE A,X
IH  40 NEXT A
FY  50 POKE 812,65:POKE 813,0:POKE 814,158
WV  60 END
YU  100 DATA 11,158,115,158,112,158,144,15
KU  8,112,158
    110 DATA 112,158,189,74,3,240,38,169,1
LP  54,133
    120 DATA 1,133,204,169,155,133,206,133
XJ  ,208,169
    130 DATA 0,133,0,133,205,169,128,133,2
JN  03,133
    140 DATA 207,169,152,133,106,141,7,212
RE  ,169,127
    150 DATA 133,209,76,92,158,169,148,133
VV  ,1,169
    160 DATA 149,133,204,169,150,133,206,1
NO  69,151,133
    170 DATA 208,169,0,133,0,133,203,133,2
ZJ  05,133
    180 DATA 207,169,144,133,106,141,7,212
VQ  ,169,255
    190 DATA 133,209,189,75,3,240,4,169,14
YN  4,133
    200 DATA 106,169,2,133,29,141,29,208,1
SY  69,1
    210 DATA 141,111,2,160,1,96,169,34,141
LV  ,47
    220 DATA 2,169,158,133,106,169,0,141,2
VH  9,208
PY  230 DATA 141,0,208,141,1,208,141,2,208
    ,141
BO  240 DATA 3,208,160,1,96,164,85,166,29,
    224
WK  250 DATA 2,208,104,201,48,208,23,165,0
    ,141
WK  260 DATA 45,159,165,1,141,46,159,32,40
    ,159
EC  270 DATA 169,0,133,29,140,0,208,160,1,
    96
NU  280 DATA 201,49,208,23,165,203,141,45,
    159,165
IW  290 DATA 204,141,46,159,32,40,159,169,
    203,133
YD  300 DATA 29,140,1,208,160,1,96,201,50,
    208
JU  310 DATA 23,165,205,141,45,159,165,206
    ,141,46
NM  320 DATA 159,32,40,159,169,205,133,29,
    140,2
MN  330 DATA 208,160,1,96,165,207,141,45,1
    59,165
QR  340 DATA 208,141,46,159,32,40,159,169,
    207,133
IM  350 DATA 29,140,3,208,160,1,96,201,155
    ,208
RZ  360 DATA 11,169,2,133,29,133,84,133,85
    ,160
WE  370 DATA 1,96,72,160,0,177,29,141,35,1
    59
FC  380 DATA 160,1,177,29,141,36,159,166,8
    4,104
BX  390 DATA 157,232,3,230,84,96,166,209,1
    69,0
PX  400 DATA 157,232,3,202,208,250,96
```

LISTING 2

PLAYER GRAPHICS

```
NE  0 REM MINI ARKANOID
OI  1 REM Michal Skowronski
HZ  2 REM (c) 1990, Bajtek
NI  3 REM
BH  4 GRAPHICS 0:POSITION 5,10:? "Stopien
    trudnosci (1-3) ";:INPUT ST
RM  5 DIM A$(5),B$(18):A$="(■)(♥":B$="♥♥♥♥
    ♥♥♣88♣♥♥♥♥♥♥"
DJ  10 OPEN #1,0,15,"A":GRAPHICS 23:POKE 5
    59,58:COLOR 1
FW  19 REM ----------PLANSZA--------------
LC  20 PLOT 0,0:DRAWTO 159,0:DRAWTO 159,95
    :DRAWTO 0,95:DRAWTO 0,0
IX  30 FOR XR=5 TO 150 STEP 10
ZR  40 FOR YR=9 TO 39 STEP 6
ZZ  50 C=2:GOSUB 500:NEXT YR:NEXT XR
GS  60 COLOR 3:FOR Y1=8 TO 86 STEP 3
WL  70 PLOT 4,Y1:DRAWTO 155,Y1:NEXT Y1
AS  80 FOR X1=4 TO 155 STEP 10
FE  90 PLOT X1,9:DRAWTO X1,85:PLOT X1+1,9:
    DRAWTO X1+1,85:NEXT X1
AT  129 REM ----------GRA---------------
WY  130 POKE 40744,96:POKE 704,26:POKE 705
    ,15:VY=-6:VX=3:X=122:Y=182:POKE 53256,
    4-ST-(ST>1):PX=79:PY=200:K=4
FQ  135 POKE 53278,0:TRAP 135:GOTO 165
GV  140 POSITION PX,INT(PY):? #1;0;A$
AM  145 IF PY>150 THEN PY=PY-0.01-0.02*(ST
    -1)
EG  150 IF X+VX>200 OR X+VX<48 THEN VX=-VX
WD  155 IF Y+VY<30 OR Y+VY>210 THEN VY=-VY
VG  156 IF K>0 THEN K=K-1:POKE 53278,0
OL  157 IF PEEK(53260)=2 AND K=0 THEN VY=-
    VY:VX=SGN(VX)*INT(RND(0)*5+1):K=4
FN  160 IF Y>205 THEN 300
OF  162 Z=PEEK(53253)
IZ  163 IF Z=6 OR Z=2 THEN VY=-VY:K=1:XR=I
    NT((X-50)/10)*10+5:YR=INT((Y-24)/6)*3:
    C=0:PK=PK+1:GOSUB 500
HH  165 X=X+VX:Y=Y+VY:POSITION X,Y:? #1;1;
    B$
JN  170 IF STICK(0)=11 AND PX>52 THEN PX=P
    X-6
KM  180 IF STICK(0)=7 AND PX<194 THEN PX=P
    X+6
NQ  190 GOTO 140
WM  299 REM ----------KONIEC--------------
BD  300 POKE 40744,166:? #1;1:? #1;0:CLOSE
     #1:GRAPHICS 0:POSITION 10,10:? "Zdoby
    les ";PK;" punk";:P=PK-(INT(PK/10)*10)
ZY  303 IF P<5 AND P>1 THEN ? "ty":GOTO 31
    0
XN  304 ? "tow"
AQ  310 IF PEEK(764)=255 THEN 310
FI  320 POKE 764,255:RUN
XO  499 REM --------DRUK CEGIEL------------
CF  500 COLOR C:PLOT XR+1,YR:DRAWTO XR+8,Y
    R:PLOT XR+1,YR+1:DRAWTO XR+8,YR+1
ZF  520 RETURN
```

NOWE WYKORZYSTANIE SITA ERATOSTENESA

W dotychczasowych zastosowaniach sito Erastotenesa było wykorzystywane jedynie do generowania liczb pierwszych z przedziału ograniczonego od góry liczbą naturalną N, natomiast do sekwencyjnej generacji liczb pierwszych zalecano nieefektywną metodę sprawdzania podzielności liczb naturalnych przez mniejsze od nich liczby pierwsze.

Zagadnienie sekwencyjnej generacji liczb pierwszych metodą sita Erastotenesa okazuje się problemem nie tak bardzo złożonym, jak wydawało się dotąd, ponieważ korzystając z własności liczb złożonych można je wyeliminować opierając się na następujących prostych zasadach:

1. Każda liczba złożona N może być przedstawiona w postaci iloczynu co najmniej dwóch liczb pierwszych lub w postaci k-tej potęgi liczby pierwszej, gdzie k=2,3,4...
2. O ile dana liczba złożona N nie jest kwadratem liczby pierwszej, to zawiera co najmniej jeden czynnik mniejszy od N.

Na tej podstawie można stwierdzić, że jeśli p_m jest n-tą z kolei liczbą pierwszą, to wszystkie liczby złożone w przedziale $[p^2_{m-1}, p^2_m]$ można wyeliminować czynnikami mniejszymi od p_m. Z uwagi na oczywistą nierówność $p_{m+1} < p^2_m$, liczba p_{m+1} jest znana i operację powyższą można powtórzyć dla przedziału $[p^2_m, p^2_{m+1}]$ za pomocą liczb pierwszych mniejszych od p_{m+1} itd., aż do momentu wygenerowania żądanej z góry ilości liczb pierwszych, co wykorzystano w programie 1.

W programie tym tablica A(N) jest tablicą liczb nieparzystych ze względu na oszczędność pamięci. Z uwagi na to, że w programie 1 do odczytu liczb pierwszych zastosowano maskę 2*3 (pętla w wierszu 12) należało podać pozycję liczby 5 w tablicy A(N) (pozycje — tablica p(C)) oraz krok (tablica K(C)) dla liczb 5 i 7. Liczba N — wymiar tablicy A(N) musi być postaci N=30*k-1, gdzie k=1,2,3... Wraz ze wzrostem wartości N rośnie szybkość działania programu, np. dla Atari 130XE czas wygenerowania 5000 kolejnych liczb pierwszych w zależności od N przedstawia się następująco:

N	T "
29	20'12"
299	10'39"
2999	9'34"
4499	9'31"

Współczynniki skali to zmienna W dla tablic A(N) i P(C) oraz zmienna B dla liczb. Znacznik H tablicy

```
FB    0 REM SEKWENCYJNY GENERATOR
VG    1 REM LICZB PIERWSZYCH
TE    2 REM Wojciech Przybyl
IA    3 REM (c) 1990, Bajtek
NJ    4 REM
WF    5 ? CHR$(125):? :? "  Podaj wymagana l
      iczbe":? "    liczb pierwszych ";:INPU
      T Z:N=4499:C=600:M=C+1:S=2:D=3:B=-1:?
RF    6 A=13:H=1+RND(S):DIM A(N),K(C),P(C):P
      (0)=13:K(0)=5:K(1)=7:LPRINT "1 - 2":LP
      RINT "2 - 3"
TM    7 FOR J=1 TO C:G=K(J):R=0.5*(G*G+1)-W:
      P(J)=R+G
MA    8 G=N:IF N>R THEN G=R:A(G)=H
NR    9 FOR T=0 TO J-1:IF G>=P(T) THEN FOR I
      =P(T) TO G STEP K(T):A(I)=H:NEXT I:P(T
      )=I
EQ    10 NEXT T:FOR D=D TO G STEP 2:ON A(D)<
      >H GOSUB A:D=D+1:ON A(D)<>H GOSUB A:NE
      XT D
BE    11 IF D>N THEN W=W+D:R=R-D:B=B+D+D:H=H
      +1:FOR I=0 TO J:P(I)=P(I)-D:NEXT I:D=0
      :GOTO 8
NV    12 NEXT J:? :? "  Za mala wartosc C -
      wymiar":? "   tablic K(C) i P(C)":END
UK    13 K(S-2)=B+D+D:IF S>M THEN A=14
TN    14 S=S+1:LPRINT S;" - ";B+D+D:IF S<Z T
      HEN RETURN
```

```
MY    0 N=4499:M=N+1:S=2:D=3:C=600:B=-1:A=4:
      H=1+RND(Z):Z=S:DIM A(N),K(C),P(C):P(0)
      =13:K(0)=5:K(1)=7:? "1 - 2":? "2 - 3"
PA    1 FOR J=1 TO C:G=0.5*(K(J)*K(J)+1)-W:P
      (J)=G:IF G>N THEN G=N:GOSUB 3:W=W+M:G=
      P(J)-M:P(J)=G:B=B+M+M:T=0:D=T:H=H+1
XE    2 FOR T=T TO J:FOR I=P(T) TO N STEP K(
      T):A(I)=H:NEXT I:P(T)=I-M:NEXT T:GOSUB
       3:NEXT J:END
EB    3 FOR D=D TO G STEP 2:ON A(D)<>H GOSUB
       A:D=D+1:ON A(D)<>H GOSUB A:NEXT D:RET
      URN
YT    4 K(S-2)=B+D+D:IF S>601 THEN A=5
GV    5 S=S+1:? S;" - ";B+D+D:Z=Z+1:IF Z>19
      THEN Z=0:POKE 764,255:?
GD    6 ON PEEK(764)>254 GOTO 6:RETURN
TM    7 FOR J=1 TO C:G=K(J):R=0.5*(G*G+1)-W:
      P(J)=R+G
```

A(N), zmieniany po każdorazowym przejściu tej tablicy, może być wybrany w sposób dowolny, ale żeby po zatrzymaniu programu (np. z powodu błędu) nie było konieczne wyzerowanie tablicy A(N), zastosowano zmianę początkowej wartości H, wykorzystując funkcję RND.

Program 2, działający na tych samych zasadach, ale szybciej, jest przeznaczony dla tych, którzy nie posiadają drukarki, a ponadto wystarczy im ilość liczb pierwszych ograniczona do około 1250000 kolejnych czynników. Ze względu na uproszczenia w programie tym nie można zmieniać wymiarów tablic, ponieważ maksymalna wielkość kroku k (600) nie może być większa od N. Program 2 generuje po 20 kolejnych liczb pierwszych wraz z ich numerami porządkowymi tak, aby jednorazowy wydruk mieścił się całkowicie na ekranie. Na początku tego programu oraz po przejrzeniu kolejnych 20 liczb wystarczy nacisnąć dowolny klawisz, aby na ekranie pojawiła się następna porcja czynników.

Uwaga! Obydwa programy działają w sposób półciągły i nie należy przejmować się przerwami, koniecznymi do przygotowania kolejnej serii liczb pierwszych.

Wojciech Przybył

KOLORY W DLI

Zainspirowany artykułem „Grafika w DLI" („Bajtek" 3/88) oraz pchełką „TĘCZA" — przedruk „Komputera" z „Antic" — (notabene ten drugi nie spełnił chyba oczekiwań Czytelników), napisałem poniższy programik, wykorzystujący bardzo prostą procedurę wykonywaną podczas DLI i ukazujący możliwe do uzyskania efekty kolorystyczne w systemowo najuboższym pod tym względem trybie 8, który jednak ze względu na rozdzielczość często się stosuje.

Cały programik jest „radosną twórczością" i powstał w wyniku pewnych poszukiwań, w których przede wszystkim zrezygnowałem z posługiwania się rejestrem VCOUNT ($D40B = 54283), przy którego użyciu można uzyskać bardzo dokładne efekty „pasów", tak często obserwowane ostatnio w wielu grach. Postawiłem w większym stopniu na improwizację, bez niewolniczego śledzenia przez program określonych komórek pamięci, od którego jednak całkowicie uciec się nie da. Uzyskany efekt niech ocenią sami Czytelnicy.

Opis programu

Wiesze 10—30 — deklaracje zmiennych oraz modyfikacja programu ANTIC-a w trybie 8+16 (bez okna tekstowego).

Wiersz 40 wczytuje program maszynowy dla przerwania DL oraz procedurę utrzymującą tryb graficzny (od adresu 1536 — ostatnia używana komórka — 1563).

Wiersze 50—80 tworzą rysunek mający uwypuklić możliwe do uzyskania efekty (może być oczywiście zastąpiony innym, samodzielnie obmyślonym).

Wiersz 90 wczytuje i umieszcza na dole ekranu napis.

Wiersz 100 nie wymaga chyba komentarza.

Wiersz 110 zapisuje wektor DLI adresem programu do wykonania oraz ustawia bit 7 w rejestrze NMIEN (bit 6 jest ustawiony), zezwalając na wykonanie DLI.

Wiersz 120 utrzymuje tryb 8+16. Przyciśnięcie dowolnego klawisza spowoduje powrót do trybu GRAPHICS O lub wykonanie dalszej części programu, jeśli takowa zostanie dopisana.

Wiersz 130 — procedura przerwania DL. Dla uzyskania odmiennych efektów można zastąpić liczbę 29 jakąkolwiek inną z zakresu 0—255.

Wiersze 131 i 141 zawierają listing asemblera z opisem. Mogą naturalnie zostać opuszczone przy przepisywaniu programu.

Wiersz 150 — dowolny napis o długości do 20 znaków.

Sławomir Słomiński

```
EZ    1 REM KOLORY W DLI
TH    2 REM SLAWOMIR SLOMINSKI
XO    3 REM COPYRIGHT (C) BAJTEK
NJ    4 REM
YO    10 DIM A$(20):A$(1)=" ":A$(20)=" ":A$(
      2)=A$:A=220:B=50:N=319:DR=80:MR=40:Z=1
      00:M=1:W=0:X=0:S=0:DEG
GZ    20 GRAPHICS 24:POKE 710,0:COLOR M:DL=P
      EEK(560)+256*PEEK(561):FOR K=DL TO DL+
      3:POKE K,PEEK(K)+128:NEXT K
FS    30 POKE DL+198,6:FOR I=DL+6 TO DL+198
      STEP 4:POKE I,PEEK(I)+128:NEXT I:NAP=P
      EEK(DL+100)+256*PEEK(DL+101)+97*40-1
XB    40 READ C:IF C<>-1 THEN POKE 1536+X,C:
      X=X+1:GOTO 40
CY    50 FOR I=0 TO N STEP 320/191:PLOT I,W:
      DRAWTO N,W:W=W+1:NEXT I:M=2
UH    60 COLOR M:H=A+DR:PLOT H,B
WZ    70 FOR I=0 TO 360 STEP Z:X=DR*COS(I):Y
      =MR*SIN(I):DRAWTO A+X,B+Y:NEXT I:IF Z<
      170 THEN Z=Z+4:GOTO 70
RR    80 N=N+1:IF N<321 THEN A=100:B=140:Z=1
      00:M=1:GOTO 60
RG    90 READ A$:FOR I=1 TO 20:POKE NAP+I,AS
      C(A$(I))-32:NEXT I
DZ    100 SOUND 0,0,0,0:FOR T=15 TO 0 STEP -
      1:SOUND 0,38,10,T:FOR D=1 TO 5:NEXT D:
      NEXT T
AN    110 POKE 512,0:POKE 513,6:POKE 54286,1
      92
YI    120 A=USR(1554)
HA    130 DATA 72,24,173,27,6,105,29,141,27,
      6,141,010,212,141,24,208,104,64
YM    131 REM *=$0600,PHA,CLC,LDA $061B,ADC
      #$1D,STA $061B,STA $D40A,STA $D018,PLA
      ,RTI-procedura przerwania DL
PB    140 DATA 104,173,252,2,201,255,240,249
      ,96,-1
TO    141 REM *=$0612,PLA,LDA $02FC,CMP #$FF
      ,BEQ -7,RTS - utrzymanie grafiki 8+16
      (zamiast linii typu 120 GOTO 120)
KE    150 DATA KOLORY W TRYBIE 8+16
```

FUNKCJE KĄTÓW WIELOKROTNYCH

Chciałbym zaprezentować program, który pomógł mi podczas nauki trygonometrii w III klasie.

Przyjmując wartość jednej z czterech funkcji trygonometrycznych za znaną, można obliczyć wartości funkcji kątów wielokrotnych: sin nx, cos nx, tg nx i ctg nx, nie tylko dla każdego n naturalnego, ale także dla n należącego do zbioru liczb rzeczywistych. Na przykład, aby obliczyć wartość ctg 123.75x znając sin x, wystarczy podać numer funkcji (tu 1), wprowadzić jej wartość i podać wielokrotność n (w tym przypadku n=123.75). Wyniki wyświetlane są niemal natychmiast. Wykonanie podobnego działania metodą tradycyjną (czyli kartka, ołówek i funkcje specjalne) jest wręcz samobójstwem.

Arnold Adamczyk

```
UZ    0 REM Funkcje katow wielokrotnych
QR    1 REM Arnold Adamczyk
HZ    2 REM (c) 1990, Bajtek
NI    3 REM
SV    10 GRAPHICS 0:SETCOLOR 2,0,0:DEG
MQ    20 ? "   Funkcje katow wielokrotnych"
SA    30 ? :? " 1  sin x"
AM    40 ? " 2  cos x"
OV    50 ? " 3  tg  x"
BA    60 ? " 4  ctg x":?
DM    70 ? "Podaj nr znanej Ci funkcji"
GX    80 INPUT NR:IF NR<1 OR NR>4 THEN 10
TI    90 ? "Podaj wartosc tej funkcji"
SK    100 INPUT X
UX    110 IF NR<3 AND ABS(X)>1 THEN 10
ZQ    120 ? "Podaj wielokrotnosc kata"
UR    130 INPUT N:?
WE    140 ON NR GOSUB 210,230,250,260
HB    150 X=X*N
ZT    160 ? "sin ";N;"x =";SIN(X)
VH    170 ? "cos ";N;"x =";COS(X)
BP    180 IF COS(X)<>0 THEN ? " tg ";N;"x =
      ";SIN(X)/COS(X)
SY    190 IF SIN(X)<>0 THEN ? "ctg ";N;"x =
      ";COS(X)/SIN(X)
XT    200 ? :GOTO 20
AP    210 IF ABS(X)=1 THEN X=90*X:RETURN
QK    220 X=ATN(X/SQR(1-X*X)):RETURN
YB    230 IF ABS(X)=1 THEN X=-90*(X-1):RETUR
      N
PE    240 X=90-ATN(X/SQR(1-X*X)):RETURN
DJ    250 X=ATN(X):RETURN
AX    260 X=90-ATN(X):RETURN
```

MONTE CARLO

Tytuł sugeruje pokera lub ruletkę, ale nie chcemy rozbijać banku w kasynie gry. Mowa będzie o czymś zupełnie innym — o poważnych obliczeniach, wykonywanych w sposób na pierwszy rzut oka niepoważny.

Co to za poważne obliczenia? Na przykład liczenie wartości niezwykle skomplikowanych całek, rozwiązywanie zadań związanych z ruchem pojazdów (np. sterowanie światłami na skrzyżowaniach ulic), symulowanie różnych naturalnych procesów, a także wyznaczanie różnych stałych. Co to za niepoważna metoda? Zgodnie z tytułem — metoda Monte Carlo. Nie wnikając głęboko w różne klasyfikacje, jakie można by w tym miejscu zastosować, będziemy nazywać metodami Monte Carlo wszystkie te sposoby rozwiązywania zadań, które bazują na wykorzystaniu generatorów liczb losowych. O przykładowym generatorze i o symulacji naturalnego procesu (była nim dyfuzja) już pisałem („Bajtek" 3–4/90 i 5–6/90). Dzisiaj zajmiemy się wyznaczaniem stałej.

Na pytanie, czy korzystając z generatora liczb losowych można wyznaczyć liczbę pi (3.141592753589...), większość z Was zapewne zakreśli palcem charakterystyczne kółko na czole. Tymczasem istnieją przynajmniej dwa sposoby rozwiązania tego zadania. Jeden z nich polega na symulowaniu starego doświadczenia, w którym rzuca się igłę na poliniowany papier. Jeśli odległość między liniami jest równa podwojonej długości igły, stosunek całkowitej liczby rzutów do liczby rzutów, w których igła przecinała którąś z linii, jest równy właśnie pi. Ten sposób jest jednak dość trudny do zaprogramowania; zastosujemy więc inną, znacznie prostszą metodę.

Wykorzystamy w tym celu odrobinę geometrii analitycznej. Weźmy kwadrat, którego bok będzie miał długość równą 2, a środek znajdzie się w punkcie (0,0). Wylosujmy (korzystając z generatora liczb losowych) dowolny punkt leżący wewnątrz tego kwadratu. Jeżeli odległość tego punktu od punktu (0,0) (czyli od środka kwadratu) jest mniejsza lub równa jeden (co można łatwo sprawdzić), to leży on nie tylko wewnątrz kwadratu, ale także wewnątrz koła o promieniu równym jeden i środku w punkcie (0,0). Wylosujmy N punktów. Wśród nich N_k znalazło się wewnątrz koła (sprawdziliśmy to badając ich odległość od punktu (0,0)). Pole kwadratu jest równe 4, a pole koła — jeśli szanse każdego punktu wewnątrz kwadratu na to, że zostanie wylosowany, są takie same — $S_{koła}=4*N_k/N$ (oczywiście tylko w przybliżeniu, radzę to zrozumieć przed czytaniem dalej). Wiemy równocześnie z geometrii, że $S_{koła}=\pi r^2=\pi$ (bo w naszym przypadku r=1). Po porównaniu obu wzorów otrzymamy proste równanie $\pi=4*N_k/N$. Tym sposobem możemy wyznaczyć liczbę pi, korzystając z losowania punktów wewnątrz kwadratu. Robi to bardzo krótki program, przedstawiony na wydruku 1.

Wprawdzie punkty nie są losowane wewnątrz całego kwadratu, tylko w jego części leżącej w pierwszej ćwiartce układu współrzędnych, ale nie ma to żadnego ubocznego wpływu na wynik, tak samo jak fakt, że nie liczymy w programie odległości, ale jej kwadrat. Procedura **randomize** służy do inicjowania generatora liczb losowych, funkcja **random(i)** zwraca losową liczbę należącą do przedziału **0...i−1**. Przyjrzyjmy się teraz przykładowym wynikom (błąd jest oszacowany na podstawie tzw. prawa wielkich liczb):

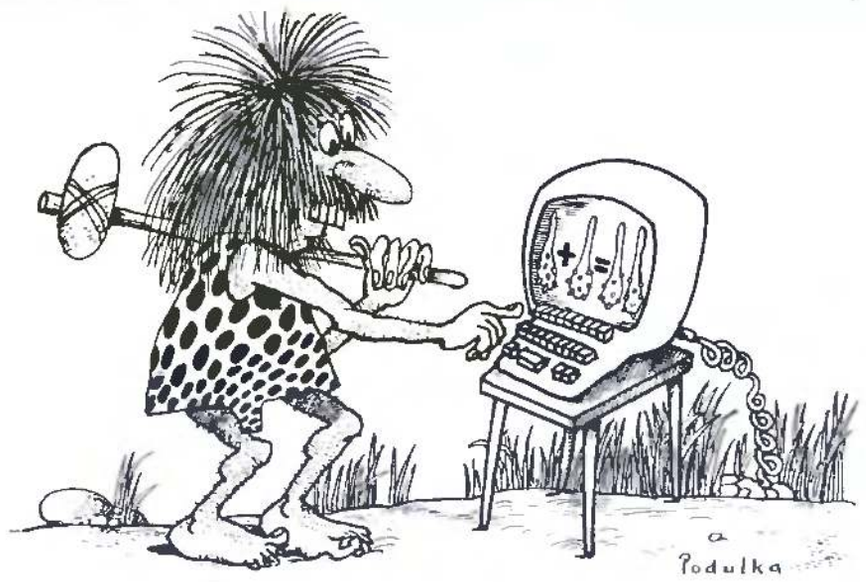
Liczba prób	Przybliżenie π	Błąd
10	2.7	±0.47
100	3.20	±0.15
1000	3.164	±0.05

Trudno je nazwać dokładnymi, ale wartość 3.164 może już okazać się wystarczająco dobra do wielu zastosowań. Zauważcie, że zwiększenie ilości losowań powoduje poprawę wyniku — można udowodnić, że błąd wyznaczenia liczby π w prezentowany sposób jest odwrotnie proporcjonalny do pierwiastka z liczby losowań, czyli po nieskończenie wielu losowaniach powinno się udać wyznaczyć π z dowolnie dużą dokładnością. Widać to na rysunku, na którym jest zaznaczona wartość liczby pi po określonej liczbie losowań. Powoli wynik zbliża się do pożądanej przez nas wartości.

Oczywiście nikt zajmujący się w sposób poważny komputerami, nie będzie wyznaczać akurat liczby pi w opisany sposób. Istnieją inne, znacznie wydajniejsze sposoby, pozwalające wyznaczyć w ciągu kilku minut tysiące cyfr na komputerze klasy IBM PC XT. Po co więc cała wrzawa wokół metod Monte Carlo?

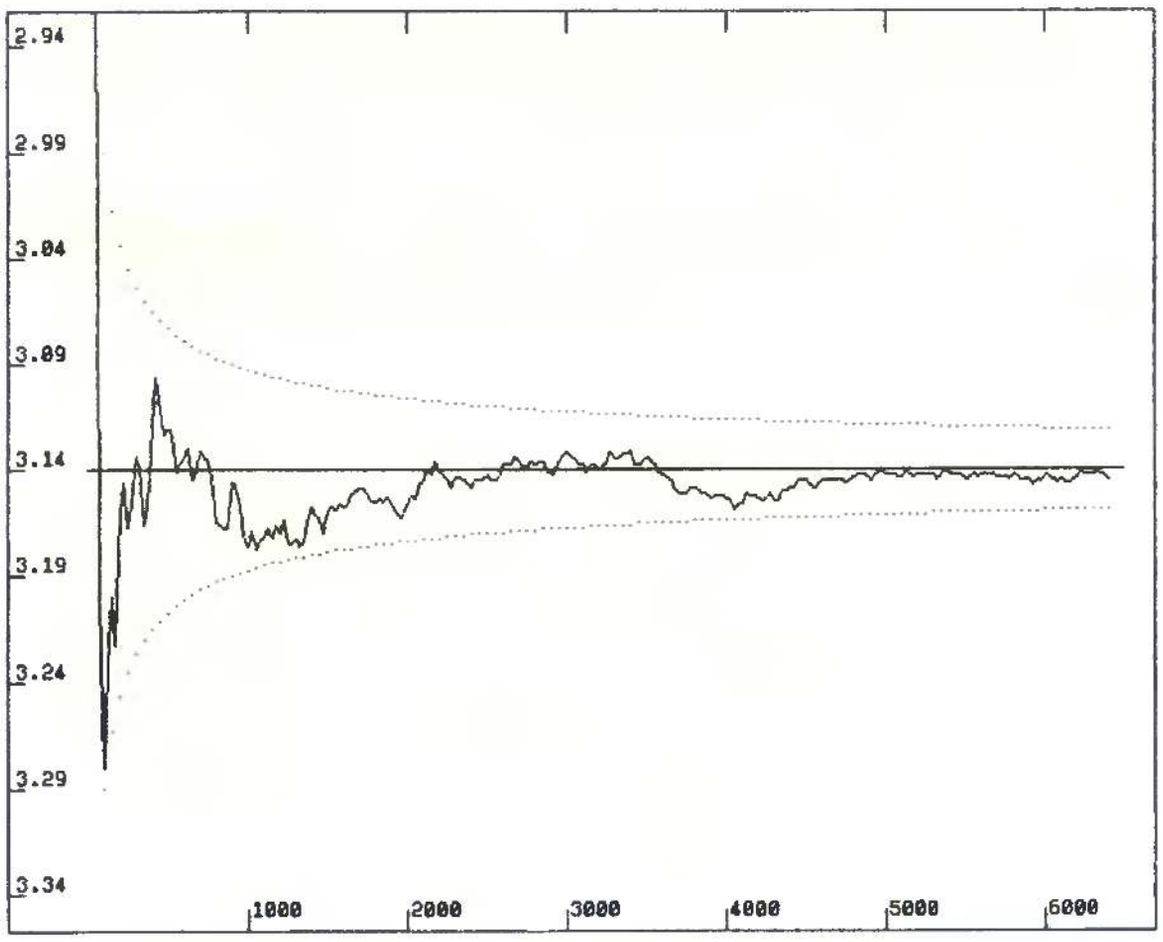
Istnieją po temu co najmniej dwa powody. Po pierwsze, pewnych zadań praktycznie nie daje się inaczej rozwiązać, czy to ze względu na ich charakter, którego nieodłączną cechą jest losowość, czy też ze względu na niesłychany stopień komplikacji zagadnienia. Często metody Monte Carlo pozwalają na wykonanie obliczeń wielokrotnie mniejszym kosztem niż w przypadku metod klasycznych. Wprawdzie wynik jest zawsze obarczony pewnym błędem, ale kilkuprocentowy błąd nie ma większego znaczenia. Istnieją zresztą metody pozwalające na zmniejszenie błędu. (Zauważcie, że przy wyznaczaniu liczby pi ważniejsze jest dokładne pokrycie losowanymi punktami okolicy brzegu koła niż jego środka. Przez losowanie punktów w sposób nierównomierny, a odpowiednio dopasowany do zagadnienia, można zmniejszyć błąd wyniku, jednak wzór, z którego będziemy liczyć pi, nie będzie już tak prosty i łatwy do zrozumienia, bo musi uwzględniać nierównocenność punktów wewnątrz kwadratu).

Po drugie, choć wynik pojedynczych obliczeń może być daleki od właściwego, mamy pewność, że średnia z wielu obliczeń nie jest obciążona błędem systematycznym. O co tu chodzi? Pomożemy sobie prostym przykładem. Liczbę pi można wyznaczyć doświadczalnie, mierząc obwód i średnicę koła, a następnie dzieląc je przez siebie. Jeśli do mierzenia obwodu użyjemy bardzo grubego sznurka, jego długość będzie zawsze większa od rzeczywistego obwodu koła (kto nie wierzy — niech sprawdzi!). Choć po kilku pomiarach okaże się, że stosunek mierzonego obwodu do średnicy jest niemal stały i że błąd pojedynczego doświadczenia jest bardzo mały, wyznaczona liczba pi będzie trochę za duża. Mówimy wówczas, że popełniliśmy błąd systematyczny. Otóż, w metodach Monte Carlo (dopóki nie zajmują się nimi matematyczno-informatyczni analfabeci) błąd systematyczny nie ma prawa się pojawić, co daje im przewagę nad niektórymi klasycznymi metodami, o których wiadomo, że dają zawyżone (bądź zaniżone) wyniki.



Ktoś może teraz zadać pytanie — czy zamiast losować punkty, nie byłoby lepiej pokratkować kwadrat i zamiast losowanych punktów użyć tych, które leżałyby na środku (albo na wierzchołkach) kratek? Pytanie bardzo dobre, a odpowiedź na nie brzmi — nie. Wyobraźcie sobie, że po pokratkowaniu kwadratu macie na jego powierzchni sto równomiernie rozmieszczonych punktów i policzyliście za pomocą tej siatki liczbę pi. Teraz chcecie zwiększyć dokładność i zwiększyć liczbę punktów o 20. Gdzie mają być położone? Nie da się ich w systematyczny sposób tak rozmieścić, żeby nie było na powierzchni kwadratu miejsc, w których będzie ich nieco więcej niż gdzie indziej. Znaczy to, że punkty nie są rozmieszczone równocennie i nie możemy już skorzystać z prostego równania $\pi=4*N_k/N$. Jest jeszcze jeden powód, czysto matematyczny (i ważniejszy). Żeby oszacować błąd, jaki popełniamy wyznaczając szukaną wielkość, musimy znać prawdopodobieństwo wylosowania każdego punktu — może ono być dowolnie małe, ale nie może być zerowe. Tymczasem w przypadku systematycznego wyboru niektóre punkty mają prawdopodobieństwo wylosowania równe jeden, a pozostałe zero (bo losowania tak naprawdę nie ma). Nie potrafimy więc oszacować popełnionego błędu i cała praca idzie na marne. Dlatego też trzeba punkty losować, a nie wybierać w sposób systematyczny.

Marcin Borkowski




```
program Monte_Carlo;

const
  Nlos     =  100;  { Określa ilość losowań. }

var
  N, Nk         : integer;
  rndx, rndy    : real;  { Losowane współrzędne punktu. }

begin
  Nk:=0;
  randomize;
  for N:=1 to Nlos do
  begin
    rndx:=random(maxint)/(maxint-1); { Maxint to predefiniowana }
    rndy:=random(maxint)/(maxint-1); { stała, równa 32768. }
    if sqr(rndx)+sqr(rndy)<=1 then Nk:=Nk+1  { x²+y²=r² }
  end;
  writeln('Po ',Nlos,' losowań pi=',4.0*Nk/Nlos:6:4)
end.
```

PODGLĄDACZ (KODU MASZYNOWEGO)

Czy pamiętacie procedurę do śledzenia pracy programu publikowaną w „Bajtku" nr 11/87? Proponujemy dziś podobną, lecz przeznaczoną do śledzenia pracy kodu maszynowego.

Adres aktualnie wykonywanej instrukcji asemblera odczytujemy z rejestru **PC** mikroprocesora. Nie jest to proste, z uwagi na to, że nie ma bezpośrednich komend odczytujących rejestr **PC**.

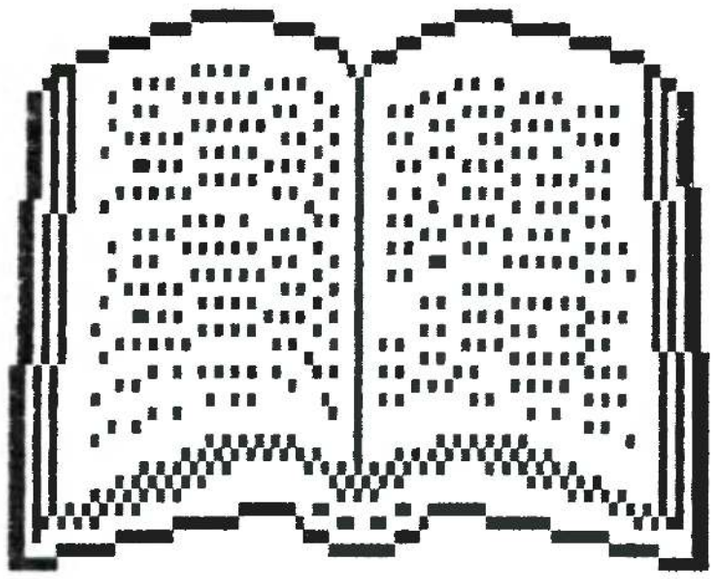
Nasz „śledź" będzie wykorzystywał drugi tryb przerwań, przez co nie można zmieniać rejestru **I**, trybu, ani zawartości tablicy przerwań.

Obok zamieszczone są dwa listingi: pierwszy jako tekst dla GENS-a, drugi w postaci programu w BASIC'u. Jeżeli chcemy skorzystać z pierwszego — należy wpisać go na asembler GENS i skompilować. Otrzymany kod wynikowy można ewentualnie nagrać. Listing drugi sam wpisze do pamięci odpowiednie wartości i nagra je na taśmę. Początek i długość kodu wynikowego: **65116,184**.

Aktywacja procedury następuje od etykiety **INIT**, dezaktywacja zaś — od **DISABL**. Z poziomu BASIC-a są to odpowiednio **RANDOMIZE USR 65271** i **RANDOMIZE USR 65292**.

Wynikiem działania procedury jest wyświetlanie z prawej strony ekranu adresów wykonywanych aktualnie instrukcji. Podawane są one szesnastkowo.
Należy pamiętać, że adresy te są przybliżone ze względu na cykl przerwania **IM 2** a szybkość wykonywania instrukcji kodu maszynowego.

BRØMBA & GEN MARTINEZ



```
  1 DATA 34,176,92,225,229,245,197,213,58,129,1598
  2 DATA 92,60,50,129,92,254,22,32,11,17,759
  3 DATA 28,64,237,83,19,255,175,50,129,92,1132
  4 DATA 237,91,19,255,124,31,31,31,31,205,1055
  5 DATA 185,254,19,124,205,185,254,19,125,31,1401
  6 DATA 31,31,31,205,185,254,19,125,205,185,1271
  7 DATA 254,42,19,255,203,28,203,28,203,28,1263
  8 DATA 1,32,0,237,74,203,20,203,20,203,993
  9 DATA 20,34,19,255,209,193,241,42,176,92,1281
 10 DATA 195,56,0,230,15,135,229,33,215,254,1362
 11 DATA 6,0,79,9,70,35,78,197,225,6,705
 12 DATA 8,126,18,35,20,16,250,122,214,8,817
 13 DATA 87,225,201,61,128,61,136,61,144,61,1165
 14 DATA 152,61,160,61,168,61,176,61,184,61,1145
 15 DATA 192,61,200,62,8,62,16,62,24,62,749
 16 DATA 32,62,40,62,48,33,28,64,34,19,422
 17 DATA 255,24,2,92,254,175,50,129,92,62,1135
 18 DATA 254,237,71,237,94,201,237,86,62,63,1542
 19 DATA 237,71,201,0,0,0,0,0,243,13,765
8000 CLEAR 5e4: LET a=65116
8010 FOR n=1 TO 19
8020 LET s=0
8030 FOR m=0 TO 9
8040 READ w: POKE a,w: LET s=s+w: LET a=a+1
8050 NEXT m: READ w: IF w<>s THEN  PRINT "Popraw
linie ";n: STOP
8060 NEXT n
8070 SAVE "pc_view.1"CODE 65116,184
```

```
 10 *C-
 20 *D+
 30
 40          ORG   #FE5C
 50
 60
 70 START    LD    (#5CB0),HL
 80          POP   HL
 90          PUSH  HL
100          PUSH  AF
110          PUSH  BC
120          PUSH  DE
130          LD    A,(#5C81)
140          INC   A
150          LD    (#5C81),A
160          CP    #16
170          JR    NZ,OMIN
180          LD    DE,#401C
190          LD    (VAR),DE
200          XOR   A
210          LD    (#5C81),A
220 OMIN     LD    DE,(VAR)
230          LD    A,H
240          RRA
250          RRA
260          RRA
270          RRA
280          CALL  DRUK
290          INC   DE
300          LD    A,H
310          CALL  DRUK
320          INC   DE
330          LD    A,L
340          RRA
350          RRA
360          RRA
370          RRA
380          CALL  DRUK
390          INC   DE
400          LD    A,L
410          CALL  DRUK
420          LD    HL,(VAR)
430          RR    H
440          RR    H
450          RR    H
460          LD    BC,#20
470          ADC   HL,BC
480          RL    H
490          RL    H
500          RL    H
510          LD    (VAR),HL
520          POP   DE
530          POP   BC
540          POP   AF
550          LD    HL,(#5CB0)
560          JP    #38
570
580 DRUK     AND   #0F
590          ADD   A,A
600          PUSH  HL
610          LD    HL,#FED7
620          LD    B,#0
630          LD    C,A
640          ADD   HL,BC
650          LD    B,(HL)
660          INC   HL
670          LD    C,(HL)
680          PUSH  BC
690          POP   HL
700          LD    B,#08
710 POWT     LD    A,(HL)
720          LD    (DE),A
730          INC   HL
740          INC   D
750          DJNZ  POWT
760          LD    A,D
770          SUB   #08
780          LD    D,A
790          POP   HL
800          RET
810
820 TABL     DEFB  #3D,#80,#3D,#88
830          DEFB  #3D,#90,#3D,#98
840          DEFB  #3D,#A0,#3D,#A8
850          DEFB  #3D,#B0,#3D,#B8
860          DEFB  #3D,#C0,#3D,#C8
870          DEFB  #3E,#08,#3E,#10
880          DEFB  #3E,#18,#3E,#20
890          DEFB  #3E,#28,#3E,#30
900
910 INIT     LD    HL,#401C
920          LD    (VAR),HL
930          JR    OMIN2
940          DEFB  #5C,#FE
950 OMIN2    XOR   A
960          LD    (#5C81),A
970          LD    A,#FE
980          LD    I,A
990          IM    2
1000         RET
1010
1020 DISABL  IM    1
1030         LD    A,#3F
1040         LD    I,A
1050         RET
1060
1070 VAR     DEFB  #00,#00
```

▲ **LISTING 1**
◀ **LISTING 2**

NAPĘDY 5'25" RAZ JESZCZE

Artykuł w numerze II (7–8) „Bajtka" traktujący o podłączaniu napędu pięciocalowego do stacji TIMEX/UNIPOLBRIT FDD wzbudził zainteresowanie wielu Czytelników. Niniejszym postaram się podać garść dodatkowych informacji, mam nadzieję pomocnych przy tej, tak na pozór prostej, operacji.

Po pierwsze. Stacja dysków TIMEX FDD 3000 (wyposażona w „duży" sterownik z dwoma kwadratowymi układami scalonymi) steruje liniami DRIVE SELECT nie bezpośrednio z układów scalonych lecz poprzez diody. Powoduje to problemy, szczególnie przy współpracy z napędami wyposażonymi w oddzielny rezystor „podciągający" ten sygnał do plusa.
We wspomnianym artykule polecano wyjąć z podstawki tzw. drabinkę rezystorową, która zawiera na ogół 8 rezystorów „podciągających" sygnały wejściowe do plusa. Cóż mają jednak zrobić nabywcy nowoczesnych napędów firm TEAC, CHINON, PANASONIC itp., wyposażonych w drabinkę w postaci 9-nóżkowego „paska", wlutowanego bezpośrednio w płytkę? Odpowiadam: nic. Nie trzeba wyrywać tej drabinki, wartość bowiem rezystorów w niej zawartych jest tak dobrana, by nie powodować nadmiernego obciążenia sterownika. Należy natomiast sprawdzić, czy nie ma dodatkowego rezystora włączonego pomiędzy wspólną ścieżkę biegnącą od zwrotek wyboru napędu do któregoś z układów scalonych na płytce napędu. Zworki te wskazują napęd poprzez wybranie jednego z czterech sygnałów DRIVE SELECT ze złącza Shugart. Zatem cztery (na ogół dolne) nóżki tych zworek biegną do odpowiednich sygnałów na złączu, a pozostałe cztery są połączone i stanowią właściwy sygnał uaktywniający napęd. Jeżeli ten sygnał jest „podciągany" do plusa rezystorem o wartości mniejszej niż około 300Ω, to należy wytropić ten rezystor i wymienić go na inny, o wartości ok. 2kΩ.

Druga sprawa. W napędach diodka świecąca podłączona jest na ogół nie do masy, tak jak zakładają autorzy w/w artykułu, lecz do plusa (najprościej sprawdzić to omomierzem).
Wystarczy więc wykorzystać drugą bramkę z układu 74LS02 jako inwerter i po kłopocie.

Po trzecie. Niektóre stacje starszych typów wyposażone są w solenoid (elektromagnes) opuszczający głowicę w odpowiedzi na sygnał HEAD LOAD. Kłopot w tym, że FDD nie ma tego sygnału. W jego miejscu (w oryginalnej dokumentacji technicznej) widnieje sygnał o nazwie IN USE, sygnalizujący wykonywanie operacji odczytu/zapisu sektora na dysku. Znamy efekt jego działania z charakterystycznego migotania diodki świecącej napędu 3-calowego. Stacja z elektromagnesem w dobrej wierze wali i łomocze głowicą w tą i z powrotem, powodując — między innymi — znaczne (3-, 4-krotne) wydłużenie czasu trwania operacji dyskowych i ból uszu.
W związku z tym radziłbym zrezygnować z tego elektromagnesu (zrezygnować — to znaczy wyrwać go ze stacji z korzeniami).

Po czwarte... Nie. „Poczwartego" na razie nie będzie. Jeśli spotkam się z jakimiś nowymi problemami, dotyczącymi omawianego tematu, nie omieszkam podzielić się z Czytelnikami doświadczeniami.

Stanisław Winiecki

ZABAWA NA 102 PRZY 80 KB

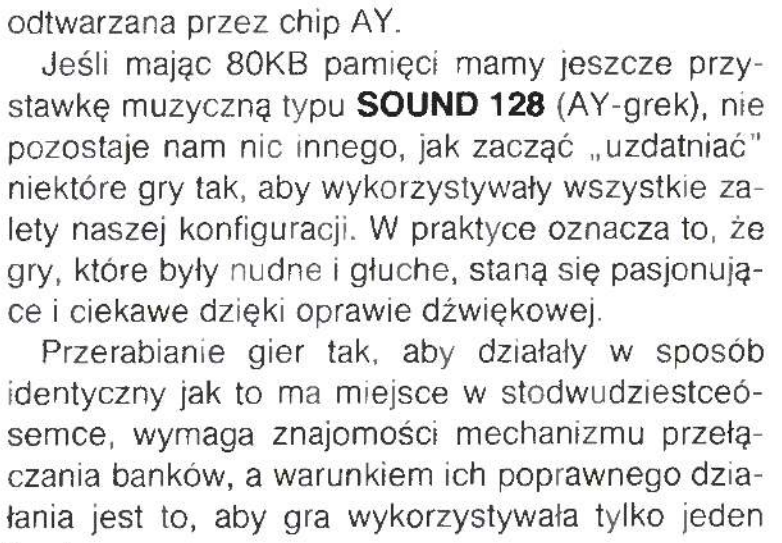
W poprzednich numerach „Bajtka" zamieszczone były opisy rozbudowy pamięci do 80KB, poniższy artykuł jest niejako ich dopełnieniem.

Jesteśmy szczęśliwymi posiadaczami **Spectrum z 80KB** pamięci RAM i dopiero teraz zastanawiamy się do czego możemy wykorzystać dodatkowe 32768 bajtów. Dodatkowego banku możemy używać jako Ramdysku, obszaru roboczego itd. Najlepszym i najbardziej cieszącym rozwiązaniem będzie jednak zbliżenie naszej starej, poczciwej trumny do **Spectrum 128**. Co miałem na myśli przez stwierdzenie „zbliżyć"? Otóż istnieje wiele programów (czytaj: gier) wykorzystujących dodatkowe banki pamięci w **Spectrum 128**. W dodatkowym banku przeważnie jest przechowywana muzyka odtwarzana przez chip AY.

Jeśli mając 80KB pamięci mamy jeszcze przystawkę muzyczną typu **SOUND 128** (AY-grek), nie pozostaje nam nic innego, jak zacząć „uzdatniać" niektóre gry tak, aby wykorzystywały wszystkie zalety naszej konfiguracji. W praktyce oznacza to, że gry, które były nudne i głuche, staną się pasjonujące i ciekawe dzięki oprawie dźwiękowej.

Przerabianie gier tak, aby działały w sposób identyczny jak to ma miejsce w stodwudziestceósemce, wymaga znajomości mechanizmu przełączania banków, a warunkiem ich poprawnego działania jest to, aby gra wykorzystywała tylko jeden bank.

Jak wiemy, nasz dodatkowy bank możemy przełączać rozkazami **OUT 255,32** i **OUT 255,0**; zakładając, że podstawowym bankiem jest bank 0. Instrukcje te przełączają górne 32768 bajty pamięci, a w **Spectrum 128** instrukcją **OUT 253, x** przełączane jest tylko 16KB od adresu 49152 — wszystkie banki w **128** zaczynają się od tego właśnie adresu. Modyfikacja kodu programu będzie polegała na zmianie procedur przełączających banki, lecz samo zamienienie **OUT 253 (32765)** na **OUT 255 (32767)** nie wystarczy.



Rys. 1

Co mogłoby się stać, jeśli przełączylibyśmy banki tak jak w **Spectrum 128**, mając jednak tylko **80KB**. Przełączyłoby się całe 32KB pamięci, co oznaczałoby „zniknięcie" danych z obszaru między adresami 32768 a 49152. Proszę przeanalizować **rysunek 1.**

Zniknięcie to nie stanowi problemu wtedy, gdy procedury przełączające banki są poniżej adresu 32768 (z uwagi na mechanizm przełączania banków w Spectrum 128 procedury te zawsze są poniżej adresu 49152). Teraz proszę sobie wyobrazić, że procedura przełączająca bank jest pod adresem: 40000. Po instrukcji OUT... następuje przełączenie banku, rejestr **PC** pozostaje bez zmian, lecz następuje pozorna zmiana zawartości komórek, z których dane (instrukcje) będzie pobierał procesor według wskazań **PC**. Jest to pozorna zmiana, bo tak naprawdę zawartości tych komórek nie zostały zmienione, tylko podmienione. Podmienione zostało 32KB od adresu 32768 do 65535. Można próbować to wytłumaczyć inaczej: z chwilą wykonania OUT'u procesor przechodzi do następnej w kolejności komendy w pamięci, z tym tylko, że już w drugim banku.

Wyjście z tej sytuacji jest bardzo proste: należy przepisać 16KB od adresu 32768 wzwyż do drugiego banku, wtedy po przełączeniu procesor w drugim banku odnajdzie kolejne rozkazy, będące tymi samymi co kolejne rozkazy w banku pierwszym. W istocie jest to symulacja przełączania jednego banku, tak jak to ma miejsce w **Spectrum 128**. Efektem tej operacji jest jakby przełączenie tylko górnych 16384 bajtów pamięci — dolne pozostają bez zmian.

Takie przepisanie połowy banku na ogół wystarczy. Czasami bywają perfidne gry, które mają stos w obszarze między 32768 a 49152. Nie jest to utrudnienie, jeśli w trakcie odwołania się do drugiego banku stos jest nie używany, lecz jeśli jest używany — to mamy problem. Rozwiązaniem jest zmiana ustawienia wskaźnika stosu w obszar wolnej pamięci — poniżej adresu 32768. Programy zwykle na samym początku, przy inicjacji ustalają, gdzie ma być stos. Jeśli robią to w trakcie działania programu albo jeszcze łączą zmianę stosu z przełączaniem banków — wtedy mamy 98% pewności, iż dana gra nigdy nie będzie możliwa do przystosowania dla naszej konfiguracji.

Na **listingu 1** przedstawiłem tekst źródłowy procedury przepisującej dolne 16KB z banku pierwszego do banku drugiego — przełączanego poleceniem **OUT 255,32.**

Jak przerabiać gry? Jaka jest technika ich modyfikacji? Postaram się to wyjaśnić na przykładzie gry **„DIZZY II".**

Zasadniczym problemem jest rozpoznanie, czy program korzysta z jednego, czy z większej liczby banków — wszak mamy możliwość symulacji tylko jednego. W celu dowiedzenia się czegokolwiek o programie należy go załadować do pamięci. Następnym krokiem jest wgranie monitora-debuggera typu **MAD** lub **FOX**, który cały znajduje się w pamięci ekranu, czyli nie zajmuje pamięci, w której mogłaby się znajdować gra. Po uruchomieniu monitora przeszukujemy pamięć w celu odnalezienia liczby **32765.** Jest to adres służący **Spectrum 128** do przełączania banków. W przypadku **„DIZZY II"** znajdujemy:

```
29171 F5       PUSH AF
29172 3E16     LD   A,#16
29174 18Ø3     JR   29179
29176 F5       PUSH AF
29177 3E1Ø     LD   A,#1Ø
29179 Ø1FD7F   LD   BC,32765
29182 ED79     OUT  (C),A
29184 F1       POP  AF
29185 C9       RET
29186 3ABØ5C   LD   A,(23728)
29189 A7       AND  A
2919Ø C8       RET  Z
29191 CDF371   CALL 29171
29194 CDØ3CØ   CALL 49155
29197 C3F871   JP   29176
292ØØ Ø63C     LD   B,#3C
```

Procedura zaczynająca się od adresu 29171 przełącza bank na bank, w którym jest muzyka (standardowe odwołanie do muzyki: CALL 49155), a procedura od 29176 przywraca bank podstawowy. W tym tylko miejscu znaleziony został adres **32765**, prawdopodobnie jest to standardowa procedura, do której odwołuje się program za każdym razem, gdy chce przełączyć bank. Poza takim sposobem przełączania banków (jedną podprocedurą) istnieje mniej eleganckie: przełączanie banku zachodzi w kilku (kilkunastu!) miejscach programu.

Analizując dalej: od adresu 29186 sprawdzana jest komórka pamięci (23728) — prawdopodobnie oryginalny loader do programu sprawdzał, na jakim typie komputera został uruchomiony i w zależności od tego wpisywał tam odpowiednią wartość. Jeśli będzie tam 205, to gra nawet nie będzie próbowała przełączać banków. Natomiast jeśli testowanie wy-

LISTING 1

```
10 *C-
20 *D+
30 ;    SYMULACJA BANKU 128
40
50
60
70 SAVE    LD    BC,16384
80         LD    HL,32768
90 SALOP   LD    D,(HL)
100        LD    A,32
110        OUT   (255),A
120        LD    (HL),D
130        XOR   A
140        OUT   (255),A
150        INC   HL
160        DEC   BC
170        LD    A,B
180        OR    C
190        JR    NZ,SALOP
200        RET
```

```
    5 REM 30 spacji

   10 DATA 1,0,64,33,0,128,86,62,32,211,617
   20 DATA 255,114,175,211,255,35,11,120,177,
32,1385
   30 DATA 241,201,0,0,0,0,0,0,0,0,442
 8000 LET a=23760
 8010 FOR n=1 TO 3
 8020 LET s=0
 8030 FOR m=0 TO 9
 8040 READ w: POKE a,w: LET s=s+w: LET a=a+1
 8050 NEXT m: READ w: IF w<>s THEN  PRINT "Po
praw linie ";n: STOP
 8060 NEXT n
```

LISTING 2

padnie poprawnie, to zostanie włączony drugi bank komendą **CALL 29171,** odwołanie do muzyczki **CALL 49155** oraz włączenie standardowego banku łącznie z powrotem **JP 29176**.

Zmiany, jakie należy wprowadzić, to po pierwsze zmiana **OUT'ów**. Sprowadza się to do zamienienia wartości #16 (22 dziesiętnie), będącej pod adresem 29173, na wartość 32 — dlatego, że **OUT 255,32** włącza nasz drugi bank. Wpisujemy jeszcze wartość 255 pod adres 29180, co daje w efekcie końcowym pod adresem 29179 instrukcję **LD BC,32767** — jest to nasz parametr dla **OUT'u**. Spytacie, dlaczego **32767** zamiast **255**. Dlatego, że i tak liczy się młodszy bajt adresu [**OUT(C),A**], a wpisywanie do komórki 29181 zera byłoby tylko zbędną czynnością.

Zmienianie wartości #10 w procedurze przywracającej bank właściwy też jest zbędne, bo wszystkie liczby mniejsze od 32 działają tak jak zero; np.: **OUT 255,30** jest tożsamy z **OUT 255,0**.

Aby modyfikowana przez nas gra działała poprawnie, należy wykonać kolejno podane niżej czynności:

— wgrać do banku drugiego moduł muzyki; najczęściej jest to plik o długości od 4 do 6 kilobajtów. Oczywiście adres ładowania nigdy nie może być niższy od 49152. Adresy ładowania najlepiej sprawdzać w oryginalnych loaderach;

— po przełączeniu na bank standardowy wgrać główny moduł gry;

— wpisać poprawki omawiane wyżej;

— przepisać za pomocą procedury z listingu 1 dolną połowę banku pierwszego do drugiego;

— uruchomić grę.

Gra powinna teraz zachowywać się tak jak na Spectrum 128, w przypadku zawieszenia się komputera bądź innych problemów należy wszystkie czynności (łącznie z przeglądaniem programu głównego) powtórzyć. Jeśli cykl ten za około dwudziestym razem nie przyniesie oczekiwanych rezultatów, oznacza to prawdopodobnie, że gra nie może być przerobiona „na 80KB".

Postępowanie w przypadku **„DIZZY II"** jest analogiczne, przy czym polecałbym wpisanie programu z **listingu 2** i uruchomienie go. W pierwszą linię zostanie wpisana procedura maszynowa z listingu 1. Po zakończeniu pracy programu w linii **5** widoczne będą „śmieci"; linie od **10** do **8060** możemy usunąć. Należy nagrać ten programik, a właściwie jedną linię, bo może się kiedyś przydać.

Do tak stworzonej linii dopisujemy następne i otrzymujemy mniej więcej taki loader:

```
 5 REM kod maszynowy; śmieci
 6 REM DIZZY 2 ver 80K by BRØMBA
10 CLEAR 24499: OUT 255,32: LOAD "MUZY-
   KA"CODE 49152
20 OUT 255,0: LOAD "GŁÓWNYPROG"CODE
30 POKE 29173,32: POKE 29180,255: POKE
   23728,255
40 RANDOMIZE USR 23760: REM uruch. kodu
   masz. z linii 5
50 RANDOMIZE USR 24832: REM start gry
```

Gra omawiana tutaj jest wyprodukowana przez **CodeMasters** — firmę, która pisze gry ładnie i przejrzyście. Prawie wszystkie gry oznaczone ich charakterystycznym symbolem dają się przerobić „na 80KB". Niestety inni autorzy nie tylko nieprzyzwoicie bawią się stosem, lecz nawet ich muzyczki posiadają zmieniane dynamicznie dane w obszarze najbardziej dla nas niekorzystnym — gdy zachodzą tam zmiany w czasie przełączania banków — czyli od 32768 do 49152.

Omówiony wyżej sposób postępowania jest przykładowy. Zdając się na inteligencję Czytelnika mniemam, iż postępując analogicznie nie będzie miał trudności z przerabianiem gier.

Maciej Pietraś

Oto gry, które udało mi się przerobić dla 80KB:

Advanced Pinball Simulator
Altered Beast
Aufwiedersehen Monty
Barbarian Quest
BMX Freestyle Simulator
Bubble Bobble
Death Stalker
Death Wish III
Eliminator
Energy Warrior
Fantasy World Dizzy
Fast Food
Foxx Fights Back
Fruit Machine Simulator II
Gemini Wing
Ghostbusters II
Hydrofool
Indiana Jones III
KGB Superspy
Marauder
Monte Carlo Casino
Monty Free
Operation Gunship
Out Run
Overlander
Pacmania
Passing Shot
Peter Pack Rat
Power Boat Simulator
Saboteur II
Samurai Warrior
Shinobi
Skills Soccer
Star Wars Droids
Street Cred Boxing
Street Soccer
Subway Vigilante
Super Scramble Simulator
Task Force
Terramex
Thunderbirds 1..4
Tiger Road
Time Scanner
Toobin
Treasure Island Dizzy
Untouchables
Venom Strikes Back
War Machine
Wizball
Xenon
Zub

JĘZYK MASZYNOWY CZ. IV

Kontynuujemy wprowadzenie do programowania w języku maszynowym Z80 na Spectrum. Czas na przedstawienie zapisu zmiennych w pamięci, za polem programu.

1. Zmienna prosta tekstowa (typ 2)
bajt 1 — typ zmiennej i numer litery deklarującej
bajty 2, 3 — długość zmiennej (liczba bajtów zajęta na zmienną od czwartego do ostatniego bajtu opisu)
Przykład: deklaracja **a$="b"**
zapis: 65 — 010 00001 = 64+1 czyli typ 2 oraz litera "a"
1, 0 — długość tekstu (1+256*0)
98 — treść zmiennej, kod litery "b"
* Długość zapisu zmiennej w pamięci zależy tylko od wielkości zmiennej. Oznaczenie zmiennej może być literą z zestawu 26 liter, po której następuje znak dolara. Pierwszy bajt przyjmuje wartość z przedziału od 65 (01000001) do 90 (01011010). Charakterystyczną cechą tej zmiennej jest to, że posiada ona zapisaną długość.

2. Zmienna prosta numeryczna o nazwie jednoznakowej (typ 3)
bajt 1 — typ zmiennej i numer litery deklarującej
bajty 2–6 — miejsce na zapis wartości zmiennej w postaci uproszczonej (młodszy i starszy bajt na poz. 4 i 5) lub zmiennoprzecinkowej.
Przykład: deklaracja **a=24116**
zapis: 97 — 011 00001 = 96+1 czyli typ 3, litera "a"
0, 0 — niewykorzystane
52, 94 — młodszy i starszy bajt liczby 24116
0 — niewykorzystane
* Długość zapisu zmiennej jest stała i wynosi sześć bajtów. Gdy argumentem jest liczba ujemna, na pozycji trzeciego bajtu znajdzie się liczba 255, a pozycje 4 i 5 reprezentować będą uzupełnienie liczby do dwóch. W tym przypadku dla −24116 byłaby to liczba 41420=65536−24116; czyli 204, 161.
* Pierwszy bajt przyjmuje wartości od 97 (01100001) do 122 (01111010).

3. Tablica numeryczna (typ 4)
bajt 1 — typ zmiennej i numer litery deklarującej
bajty 2, 3 — długość zmiennej (od czwartego bajtu opisu włącznie)
bajt 4 — liczba wymiarów tablicy (n)
bajty 5, 6 — pierwszy wymiar
bajty 7, 8 — drugi wymiar
bajty n, n+1 — n-ty wymiar tablicy
bajty n+2..n+k — zestaw pięciobajtowych bloków reprezentujących poszczególne liczby wypełniające tablicę; liczba bloków równa jest 1*2*3*..*n
Przykład: deklaracja **DIM a(2)**
zapis: 129 — 100 00001 = 128+1 czyli typ 4 i litera "a"
13, 0 — długość opisu tablicy (od czwartego bajtu włącznie)
1 — liczba wymiarów tablicy
2, 0 — pierwszy wymiar tablicy
0, 0, 0, 0, 0 — pierwsza liczba w tablicy
0, 0, 0, 0, 0 — druga liczba w tablicy
Zadeklarowano tablicę dwuliczbową, stąd w pamięci zarezerwowane 10 bajtów.
* Całkowita długość zapisu zmienia się i zależy od liczby zadeklarowanych wymiarów. Pierwszy bajt przyjmuje wartości od 129 (10000001) do 154 (10011010).

4. Zmienna prosta numeryczna o nazwie wieloznakowej (typ 5)
bajt 1 — typ zmiennej i numer pierwszej litery nazwy zmiennej
bajty 2..k — kody kolejnych znaków nazwy, oprócz znaku ostatniego
bajt k+1 — kod ostatniego znaku powiększony o 128 (znacznik końca nazwy zmiennej)
bajty k+2..k+6 — standardowy zestaw pięciu bajtów wartości zmiennej
Przykład: deklaracja **abc=2**
zapis: 161 — 101 00001 = 161+1 czyli typ 5 i litera "a"
98 — kod litery "b"
227 — kod litery "c" powiększony o 128 (99+128)
0, 0 — niewykorzystane

2, 0 — wartość zmiennej
0 — niewykorzystane

* Całkowita długość zapisu jest również zmienna i zależy od liczby użytych do oznaczenia zmiennej znaków. Pierwszy bajt opisu przyjmuje wartości od 161 (10100001) do 186 (101111010)

5. Tablica tekstowa (typ 6)

bajt 1 — typ zmiennej i numer litery deklarującej
bajty 2, 3 — długość opisu; jak zwykle
bajt 4 — liczby wymiarów tablicy (n)
bajty 5, 6 — pierwszy wymiar
bajty 7, 8 — drugi wymiar
bajty n, n+1 — n-ty wymiar tablicy
bajty n+1..n+k — obszar zarezerwowany na kody poszczególnych znaków wypełniających tablicę. Liczba zajętych bajtów równa jest iloczynowi wymiarów.

Przykład: deklaracja **DIM c$(3,2,8)**
zapis: 195 — 110 00011 = 192+3 czyli typ 6 i litera "c"
55, 0 — długość opisu zmiennej
3 — liczba wymiarów tablicy
3, 0 — pierwszy wymiar
2, 0 — drugi wymiar
8, 0 — trzeci wymiar tablicy
32, 32, 32 itd. — 48 bajtów (3*2*8) wypełnionych kodami 32 (spacja) i przeznaczonych na treść tablicy.

* Długość zmiennej w pamięci nie jest stała; zapisana jest w drugim i trzecim bajcie. Zaś bajt pierwszy przyjmuje wartości od 193 (11000001) do 218 (11011010).

Cechą charakterystyczną tablic tekstowych jest stała długość poszczególnych pól, równa wielkości następnego zadeklarowanego wymiaru. Oznacza to, że tekst krótszy wpisywany do tablicy dopełniany jest spacjami, dłuższy zaś obcinany (zasada łoża Prokrusta). Własność ta może sprawiać kłopoty przy porównywaniu krótszych łańcuchów znakowych z kolejnymi połami tablicy, gdyż oprócz zawartości porównywana jest również liczba znaków. Problem nie występuje dla tablic numerycznych.

6. Zmienna sterująca pętlą FOR-NEXT (typ 7)

bajt 1 — typ zmiennej i numer litery deklarującej.
bajty 2..6 — pięć bajtów przeznaczonych na zapis wartości początkowej zmiennej sterującej, która będzie w miarę wykonywania pętli zwiększana lub zmniejszana. Po zakończeniu pętli zapisana jest tu wartość końcowa powiększona o 1.
bajty 7..11 — jw., lecz na zapis wartości końcowej. Wartość ta nie ulega zmianie.
bajty 12..16 — jw., lecz na zapis zadeklarowanego kroku pętli (STEP). Jeśli nie zadeklarowano kroku, to przy charakterze narastającym pętla realizowana jest z krokiem 1, malejącym zaś nie jest wykonywana.
bajty 17, 18 — numer linii wykonywanej w zakresie pętli FOR-NEXT.
bajt 19 — numer pętli w danej linii powiększony o 1 (dotyczy pętli deklarowanych po dwukropku, w tej samej linii).

Przykład: deklaracja **78 FOR n=1 TO 5** (po realizacji)
zapis: 238 — 111 01110 = 224+14 czyli typ 7 i litera "n"
0, 0, 6, 0, 0 — wartość końcowa po realizacji (5+1)
0, 0, 5, 0, 0 — zadeklarowana wartość końcowa
0, 0, 1, 0, 0 — krok pętli, przyjęty za 1
78, 0 — numer linii (w przykładzie — 78)
2 — liczba pętli w linii powiększona o 1

* Długość zapisu zmiennej sterującej wynosi zawsze 19 bajtów. Pierwszy bajt przyjmuje wartości od 225 (11100001) do 250 (11111010).

* W przypadku deklaracji w jednej linii większej liczby pętli, opisy kolejnych zmiennych sterujących występują jeden po drugim.

* Po wyjściu z pętli (GOTO, GOSUB), w miejscu wartości chwilowej (poz. 2–6) zapisana zostaje wartość zmiennej sterującej w momencie wyjścia, gdyż zmienia ją dopiero NEXT.

Zadeklarowane w programie zmienne zapisywane są w specjalnie na to przeznaczonym miejscu, zwanym polem zmiennych. Obszar ten znajduje się bezpośrednio za programem, a adres początku wskazuje zmienna systemowa **VARS** (23628/9). Zmienne zapisywane są w polu zmiennych w trakcie działania programu, a więc w sposób dynamiczny i w kolejności ich występowania.

Koniec pola zmiennych można ustalić na podstawie zmiennej systemowej **ELINE** (23641/2), wskazującej adres początku tzw. bufora systemowego. Wartość **ELINE** należy jednak zmniejszyć o 2, gdyż separatorem w/w obszarów jest kod 128, będący znacznikiem końca pola zmiennych.

Materiał ten stanowi podstawę do zrozumienia działania wspomnianej procedury NEXT-ONE, do której przejdziemy w części następnej cyklu.

Piotr Sumara

ZAGLĄDAMY DO DYSKIETEK INNYCH KOMPUTERÓW

Wraz z pojawieniem się na rynku stacji dysków FDD 3000 użytkownicy komputerów ZX Spectrum otrzymali możliwość pracy w najpopularniejszym systemie operacyjnym dla komputerów 8-bitowych CP/M 2.2.

Mimo ogromnych możliwości, jakie daje praca pod tym systemem, zainteresowanie użytkowników jest niewielkie. Na taką sytuację ma głównie wpływ brak jakiegokolwiek oprogramowania użytkowego dostarczanego razem ze stacją i trudności z zakupem.

Możliwość przenoszenia plików między FDD 3000, a innymi komputerami pozwoli więc nie tylko korzystać z bogatego oprogramowania systemowego, ale także na transfer zbiorów tekstowych.

Wspomnianą wymianę danych można osiągnąć „ucząc" stację FDD 3000 czytać i zapisywać dyskietki z innych komputerów, o innym formacie zapisu. Oczywiście operacje te będą dotyczyły pojedynczych sektorów, a nie całych plików.

Czytanie i zapisywanie sektorów nie może odbywać się przez podprogramy zawarte w **BIOS**, gdyż moduł ten nie jest na tyle uniwersalny. Dodatkowe procedury czytania i zapisywania odwołują się zatem bezpośrednio do kontrolera dyskowego i portów systemowych. To pozorne skomplikowanie (w końcowym efekcie pozwala) na obsługę dyskietek o dowolnym rozmiarze sektora.

Jedynym warunkiem jest to, aby dyskietka była sformatowana w systemie **MFM.** Można będzie zatem czytać dyskietki komputerów np. **Amstrad, IMB, Spectravideo, Brain.**

Wszystkie podprogramy zostały napisane w asemblerze, jednak dla wygody użytkowników wbudowane zostały do procedur **Turbo Pascala,** co niewątpliwie uczyni je bardziej uniwersalnymi (Listing 1). Ich poprawna praca jest możliwa jedynie z oryginalną wersją modułu **BIOS** firmy TIMEX. Blok procedur warto zapisać na dysku jako osobny zbiór biblioteczny pod nazwą DYSK.SYS. Krótki program podany na końcu (Listing 2) ilustruje ich użycie.

Omówienie działania procedur:

Procedura **SetDrive** (drive) przygotowuje napęd „drive" do pracy z pozostałymi procedurami. Dodatkowo przesuwa głowicę dysku na ścieżkę 0. Wywołanie tej procedury jest niezbędne przed wywołaniem którejkolwiek z pozostałych. Daną wejściową „drive" we wszystkich wywołaniach jest numer napędu dla "A"=0, "B"=1 itd.

Procedura **ResetDrive** (drive) odtwarza pierwotny stan napędu dyskowego. Należy ją wykonać po zakończeniu pracy z napędem wybranym przez SetDrive. W trakcie pracy z opisywanymi procedurami nie wolno odwoływać się do niego zwykłymi funkcjami Bios.

Procedura **SectorSize** (drive, secnum, error, size) zwraca rozmiar (size) i numer (secnum) pierwszego napotkanego sektora na bieżącej ścieżce. Rozmiar sektora będzie jedną z czterech wartości, tj. 128, 256, 512, 1024 bajty, gdyż tylko takie wartości są akceptowane przez kontroler dysku. Rozpoznawanie nieznanych dyskietek z pomocą tej procedury, będzie niewątpliwie łatwiejsze. Zmienna „error" podaje kod błędu i będzie równa zero, gdy wywołanie zakończy się pomyślnie.

Procedura **Reads** (drive, track, sector, where, error) odczytuje z dysku jeden sektor o dowolnej długości i numerze „sector". Odczytane dane umieszcza w pamięci od adresu „where" w górę. Postać danych wejściowych jest następująca:
track — od 0 do 255,
sector — od 0 do 255,

Kody błędu:
0 — wywołanie pomyślne,
4 — dane utracone (np. przez wyjęcie dysku podczas pracy),
16 — sektor nie znaleziony,
255 — błąd ustawienia głowicy.

Procedura **Writes** (drive, track, sector, where, error) zapisuje na dysku jeden sektor o parametrach identycznych jak w Reads. Dodatkowy kod błędu o wartości 64 sygnalizuje sprzętowe zabezpieczenie dyskietki.

Opierając się na powyższym zbiorze procedur można już napisać program przenoszący pliki z dyskietek obcego formatu. Wiadomości ogólne o tym, jak system CP/M przechowuje pliki, zainteresowany czytelnik znajdzie w [1]. Organizacja fizyczna dyskietek, jak: wielkości sektorów, położenie katalogu itp. została opisana w [2].

Czytelników, którzy nie chcą pisać samodzielnie takich programów informuję, że redakcja dysponuje programami do przenoszenia plików z komputerów **Amstrad** i **IBM.**

Robert Magdziak

Literatura:
[1] R. Świniarski, *System operacyjny CP/M*, WNT 1988.
[2] J. Mayer, *Operacje dyskowe w systemie CP/M Plus*, „Bajtek" 1–2 1990.

LISTING 1

```
{************************************************************}
{   Plik DYSK.SYS       (c) R.M. 1990                        }
{ Procedury dyskowe niskiego poziomu:                        }
{ 1. Procedura SetDrive (drive:byte) ;                       }
{ 2. Procedura ResetDrive (drive:byte) ;                     }
{ 3. Procedura SectorSize (drive:byte;var secnum,error:byte  }
{                          var size:integer) ;               }
{ 4. Procedura Reads  (drive,track,sector:byte;              }
{                      where:integer; var error:byte) ;      }
{ 5. Procedura Writes (drive,track,sector:byte;              }
{                      where:integer; var error:byte) ;      }
{************************************************************}
```

```
Procedure SetDrive(drive:byte);
{************************************************************}
{ Procedura przygotowuje dany naped do pracy .               }
{ Wywolanie konieczne przed uzyciem innych procedur.         }
{ dane : drive 0=A,1=B itd.                                  }
{************************************************************}
begin

 inline($3A/DRIVE/$3C/$47/$3E
      /$5F/$D3/$E0/$3E/$7F/$7/$10/$FD/$E6/$5F/$D3/$E0
      /$3E/$5/$D3/$C0/$6/$13/$10/$FE/$DB/$C0/$CB/$47/$20/$FA
      /$3A/drive/$3C/$47/$3E/$DF/$D3/$E0/$3E/$7F/$7/$10/$FD
      /$E6/$DF/$D3/$E0/$21/>$31/$36/$0)
end; { procedury SetDrive }


Procedure ResetDrive(drive:byte);
{************************************************************}
{ Procedura odtwarza pierwotny stan napedu. Wywolanie        }
{ konieczne po zakonczeniu pracy.                            }
{ dane : drive 0=A,1=B itd.                                  }
{************************************************************}
begin
  inline($3A/drive/$CD/$FA07/$DB/$C1/$77);

end; { procedury ResetDrive }


Procedure SectorSize(drive:byte;var secnum,error:byte
                                ;var size:integer     );
{************************************************************}
{ Procedura zwraca rozmiar pierwszego napotkanego  sektora   }
{ na ostatnio uzywanej sciezce i dodatkowo podaje numer      }
{ tego sektora.                                              }
{ Dane wejsciowe: drive 0=A,1=B itd.                         }
{ Dane wyjsciowe:                                            }
{ size= rozmiar sektora w bajtach                            }
{ error= 0 gdy Ok ,inne wartosci gdy blad CRC                }
{ secnum= numer napotkanego sektora                          }
{************************************************************}
var
      s:byte;
 buffer:array[1..7] of byte;

begin
   s:=1;
   repeat
   inline(
     $CD/$E2/$FB/$ED/$56/$FB/$3A/drive/$3C/$47/$3E/
     $5F/$D3/$E0/$3E/$7F/$07/$10/$FD/$E6/$5F/$D3/$E0/
     $CD/*+29/$21/$30/$00/$77/$3A/drive/
     $3C/$47/$3E/$DF/$D3/$E0/$3E/$7F/$07/$10/$FD/$E6/
     $DF/$D3/$E0/$CD/$FE/$FB/$18/
     $12/$0E/$C3/$21/buffer/$3E/$C4/$D3/$C0/$DB/$2F/
     $17/$30/$FB/$ED/$A2/$18/$F7);
   error:=mem[$30];
   s:=s+1;
   until (error=0) or (s=3);
   secnum:=buffer[3];
   s:=buffer[4];
   case s of
     0:size:=128;
     1:size:=256;
     2:size:=512;
     3:size:=1024;


   end;
end; { procedury SectorSize }

Procedure Reads(drive,track,sector:byte;
                where:integer; var error:byte);
{************************************************************}
{ Procedura czyta z dysku jeden sektor o dowolnej dlugosci   }
{ Odczytane dane umieszcza w pamieci od adresu 'where'       }
{ Postac danych:                                             }
{ drive :  0=a ,1=b itd.                                     }
{ track :  0<=track<=255                                     }
{ sector:  0<=sector<=255                                    }
{ Kody bledu:    0=Ok                                        }
{                4=Data lost                                 }
{               16=Record not found                          }
{              255=Bad seek                                  }
{************************************************************}
 var
    i:byte;

 begin
   i:=1;
 repeat
 inline($DB/$C1/$21/>$2F/$77/$21/>$30/$36/$0/$CD
 /$FBE2/$3A/>$31/$D3/$C1/$3A/DRIVE/$3C/$47/$3E/$5F
 /$21/$FF/$A6/$D3/$E0/$3E/$7F/$07/$10/$FD/$E6/$5F/$A6/$D3/$E0
 /$3A/SECTOR/$D3/$C2/$DB
 /$C1/$21/TRACK/$BE/$28/$14/$3A/TRACK/$D3/$C3/$CD
 /*+55/$E6/$BF/$A7/$28/$07/$21/>$30/$36/$FF
 /$18/$0D/$2A/WHERE/$CD/*+46/$A7/$28/$04/$21
 />$30/$77/$3A/DRIVE/$3C/$47/$3E/$DF/$D3/$E0/$3E/$7F/$07/$10
 /$FD/$E6/$DF/$D3/$E0/$CD/$FBFE/$DB/$C1/$32/>$31/$18/$1B
 /$3E/$15/$ED/$56/$FB/$D3/$C0/$18/$FE/$3E/$80/$0E/$C3/$ED/$56
 /$FB/$D3/$C0/$DB/$2F/$17/$30/$FB/$ED/$A2/$18/$F7
 /$3A/>$2F/$D3/$C1);
 error:=mem[$30];  i:=i+1;
 until (error=0) or (i=3)
end; { procedury Reads }

Procedure Writes(drive,track,sector:byte;
                 where:integer; var error:byte);
{************************************************************}
{ Procedura zapisuje na dysk jeden sektor o dowolnej dlugosci}
{ Dane pobiera z pamieci od adresu 'where'                   }
{ Postac danych:                                             }
{ drive :  0=a ,1=b itd.                                     }
{ track :  0<=track<=255                                     }
{ sector:  0<=sector<=255                                    }
{ Kody bledu:    0=Ok                                        }
{                4=Data lost                                 }
{               16=Record not found                          }
{               64=Write protect                             }
{              255=Bad seek                                  }
{************************************************************}
 var
    i:byte;

 begin
   i:=1;
 repeat
 inline($DB/$C1/$21/>$2F/$77/$21/>$30/$36/$0/$CD
 /$FBE2/$3A/>$31/$D3/$C1/$3A/DRIVE/$3C/$47/$3E/$5F
 /$21/$FF/$A6/$D3/$E0/$3E/$7F/$07/$10/$FD/$E6/$5F/$A6/$D3/$E0
 /$3A/SECTOR/$D3/$C2/$DB
 /$C1/$21/TRACK/$BE/$28/$14/$3A/TRACK/$D3/$C3/$CD
 /*+55/$E6/$BF/$A7/$28/$07/$21/>$30/$36/$FF
 /$18/$0D/$2A/WHERE/$CD/*+46/$A7/$28/$04/$21
 />$30/$77/$3A/DRIVE/$3C/$47/$3E/$DF/$D3/$E0/$3E/$7F/$07/$10
 /$FD/$E6/$DF/$D3/$E0/$CD/$FBFE/$DB/$C1/$32/>$31/$18/$1B
 /$3A/>$2F/$D3/$C1);
 error:=mem[$30];  i:=i+1;
 until (error=0) or (i=3)
end; { procedury Writes }

 { koniec pliku dysk.sys }
```

LISTING 2

```
program przyklad;
{************************************************************}
{ Demonstracja dzialania procedur ze zbioru DYSK.SYS         }
{************************************************************}
var
   sector,track,drive,
   error,secnum       :byte;
   size,j,i           :integer;
   dr                 :char;
   buff               :array[1..1024]of byte;

{$i dysk.sys }

begin
 clrscr;
 write('podaj nazwe napedu (A/B)? ');
 readln(dr);   drive:=ord(upcase(dr))-65;
 write('podaj numer sciezki ? ');
 readln(track);
 write('podaj numer sektora ? ');
 readln(sector);
 writeln('wloz dyskietke do napedu, nacisnij enter..');
 repeat until keypressed;

 setdrive(drive);   { przygotowanie napedu }

 sectorsize(drive,secnum,error,size); { sprawdzenie dysku }

 if error<>0 then
   begin
     writeln('dysk uszkodzony, lub nie w systemie MFM !');
     resetdrive(drive);
     halt;
   end;
writeln('wielkosc sektora nr:',secnum,'  ',size,' bajtow');
writeln('nacisnij enter..');
repeat until keypressed;
                                              { czytanie sektora }
reads(drive,track,sector,addr(buff),error);

if error<>0 then
   begin
      writeln('blad odczytu nr:',error);
      resetdrive(drive);
      halt;
   end;
 i:=1;
 for j:=1 to size do
   begin
     write('bajt nr:',j:5,' wartosc:',buff[j]:4,' znak:');
     if buff[j] in [32..127] then writeln(chr(buff[j]))
                             else writeln('.');
     if i=22 then
        begin
          readln;
          i:=1;
        end;
     i:=i+1;
   end;

  resetdrive(drive);     { zakonczenie pracy }

end.  { programu przyklad }
```

THROUGH THE TRAP DOOR

Gruby, obleśny Berk znów ma kłopoty. Poprzednio nieźle się obłowił; zabrał całe złoto z zamku Ducha i jadł i pił długie tygodnie. Brzuch mu urósł jeszcze bardziej, przybyły dwa podbródki, a nos zwisł jak ogórek. Zabrany Duchowi pałac rozwinął się i wypiękniał.

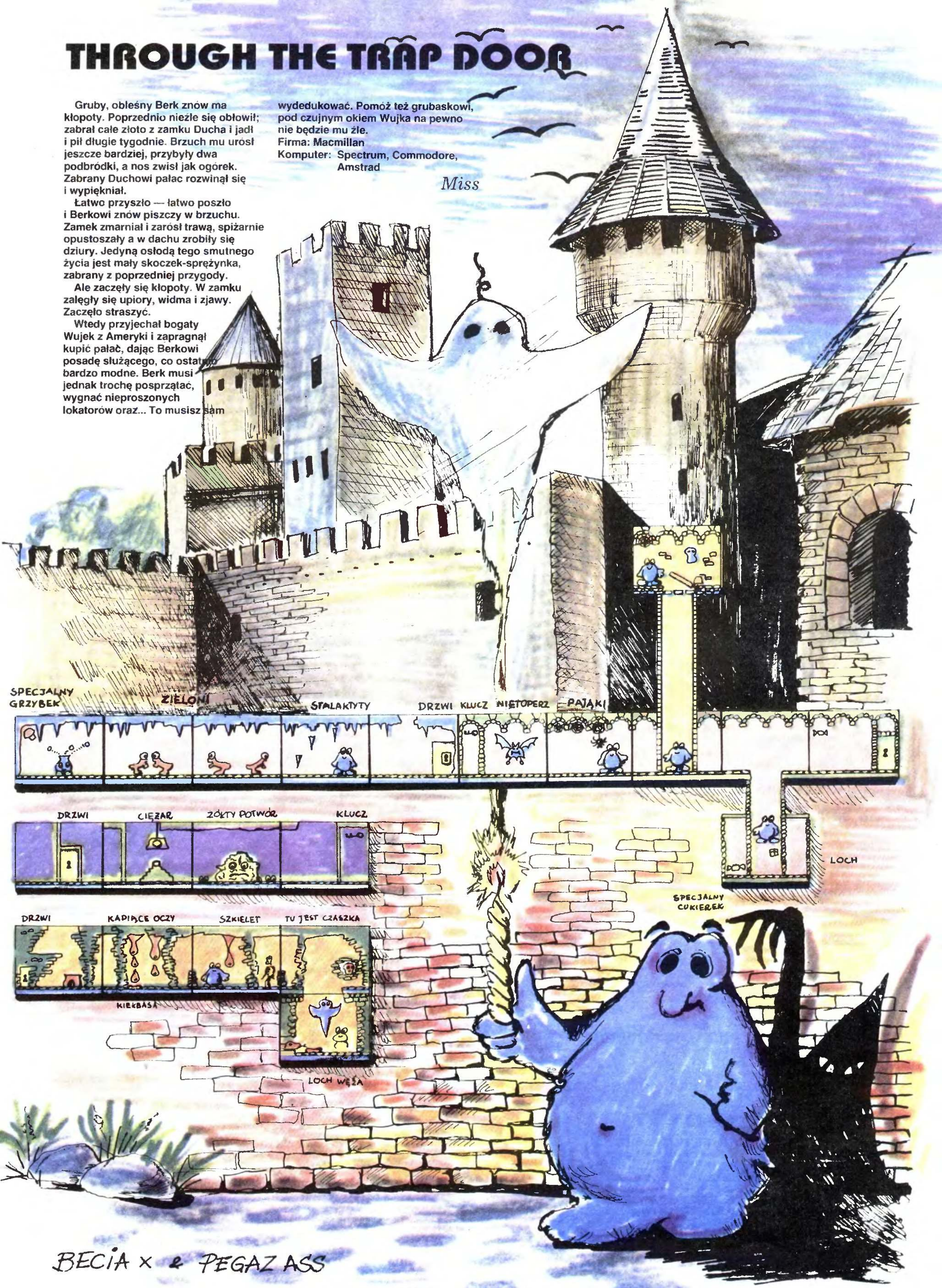
Łatwo przyszło — łatwo poszło i Berkowi znów piszczy w brzuchu. Zamek zmarniał i zarósł trawą, spiżarnie opustoszały a w dachu zrobiły się dziury. Jedyną osłodą tego smutnego życia jest mały skoczek-sprężynka, zabrany z poprzedniej przygody.

Ale zaczęły się kłopoty. W zamku zalęgły się upiory, widma i zjawy. Zaczęło straszyć.

Wtedy przyjechał bogaty Wujek z Ameryki i zapragnął kupić pałac, dając Berkowi posadę służącego, co ostatnio bardzo modne. Berk musi jednak trochę posprzątać, wygnać nieproszonych lokatorów oraz... To musisz sam wydedukować. Pomóż też grubaskowi, pod czujnym okiem Wujka na pewno nie będzie mu źle.

Firma: Macmillan
Komputer: Spectrum, Commodore, Amstrad

Miss



BECIA x & PEGAZ ASS

KUPIĘ — SPRZEDAM — ZAMIENIĘ

Każdy, kto przyśle do nas wycięty z Bajtka kupon (odbitek nie będziemy honorować), może zamieścić krótkie ogłoszenie. Maksymalna długość ogłoszenia — piętnaście słów razem z adresem, drobne odchylenia do zaakceptowania. Ogłoszenie może dotyczyć sprzedaży, kupna lub zamiany komputera i akcesoriów — wszelkiego typu urządzeń zewnętrznych, programów i literatury, używanych i nowych, pod warunkiem, że oferta dotyczyć będzie pojedynczych sztuk. Ogłoszenia drukować będziemy kolejno w miarę ich napływania. Zastrzegamy sobie prawo niewydrukowania oferty, w razie podejrzeń o próbę sprzedaży hurtowych ilości towaru, oraz prawo do zmiany zasad akceptowania ogłoszeń. Piszcie na nasz adres, z dopiskiem na kopercie — Kupię-Sprzedam-Zamienię.

JAK REKLAMOWAĆ SIĘ W BAJTKU?

Spółdzielnia "Bajtek" oferuje Państwu obsługę reklamową na łamach pism "*Bajtek*", "*Moje Atari*", "*Top Secret*".

Ceny: 1 cm^2 - 12 tys.zł.
cała strona - 10 mln zł.
kolor, okładka - do 100% drożej

Format ogłoszenia: wielokrotność 20 cm^2

Zlecenia przyjmuje:
Biuro Reklamy Spółdzielni "Bajtek"
00-687 Warszawa, ul.Wspólna 61
tel. 21-12-05

Bajtek • MOJE Atari • TOP SECRET — *Z TOBĄ PRZEZ CAŁY ROK!*

STYCZEŃ						LUTY						MARZEC					
P		7	14	21	28	P		4	11	18	25	P		4	11	18	25
W	1	8	15	22	29	W		5	12	19	26	W		5	12	19	26
Ś	2	9	16	23	30	Ś		6	13	20	27	Ś		6	13	20	27
C	3	10	17	24	31	C		7	14	21	28	C		7	14	21	28
P	4	11	18	25		P	1	8	15	22		P	1	8	15	22	29
S	5	12	19	26		S	2	9	16	23		S	2	9	16	23	30
N	6	13	20	27		N	3	10	17	24		N	3	10	17	24	31

KWIECIEŃ						MAJ						CZERWIEC					
P	1	8	15	22	29	P		6	13	20	27	P		3	10	17	24
W	2	9	16	23	30	W		7	14	21	28	W		4	11	18	25
Ś	3	10	17	24		Ś	1	8	15	22	29	Ś		5	12	19	26
C	4	11	18	25		C	2	9	16	23	30	C		6	13	20	27
P	5	12	19	26		P	3	10	17	24	31	P		7	14	21	28
S	6	13	20	27		S	4	11	18	25		S	1	8	15	22	29
N	7	14	21	28		N	5	12	19	26		N	2	9	16	23	30

LIPIEC						SIERPIEŃ						WRZESIEŃ					
P	1	8	15	22	29	P		5	12	19	26	P		2	9	16	23	30
W	2	9	16	23	30	W		6	13	20	27	W		3	10	17	24	
Ś	3	10	17	24	31	Ś		7	14	21	28	Ś		4	11	18	25	
C	4	11	18	25		C	1	8	15	22	29	C		5	13	19	26	
P	5	12	19	26		P	2	9	16	23	30	P		6	14	20	27	
S	6	13	20	27		S	3	10	17	24	31	S		7	15	21	28	
N	7	14	21	28		N	4	11	18	25		N	1	8	15	22	29	

PAŹDZIERNIK						LISTOPAD						GRUDZIEŃ					
P		7	14	21	28	P		4	11	18	25	P		2	9	16	23	30
W	1	8	15	22	29	W		5	12	19	26	W		3	10	17	24	31
Ś	2	9	16	23	30	Ś		6	13	20	27	Ś		4	11	18	25	
C	3	10	17	24	31	C		7	14	21	28	C		5	12	19	26	
P	4	11	18	25		P	1	8	15	22	29	P		6	13	20	27	
S	5	12	19	26		S	2	9	16	23	30	S		7	14	21	28	
N	6	13	20	27		N	3	10	17	24		N	1	8	15	22	29	

REKLAMUJ SIĘ U NAS!

Atari Turbo 2000 F

Nowy system transmisji danych z magnetofonem przyspieszony do 6700 bodów.

Komplet:

— cartridge
— oprogramowanie
— przeróbka magnetofonu
— instrukcja obsługi
— 12 miesięcy gwarancji

Instalacje wykonujemy na poczekaniu.

Interfejs do zwykłego magnetofonu

Duży wybór oprogramowania w standardzie TURBO-2000.

Informacja:
Tel. 33-40-91

Korespondencja:
MUEL ul. Cząstkowska 30,
01-678 Warszawa.

B82

ATARI 800 XL, 65 XE, 130 XE

Sprzedaż wysyłkowa
gier i programów użytkowych
na kasetach i dyskietkach.
Również w systemie TURBO 2000
Wszystkie nowości!!!
Instrukcje i literatura.
Dla zainteresowanych rachunki.

ANWIKOL

03-721 Warszawa ul. Jagiellońska 3/28.

B81

Programy C-16, C-116, C PLUS 4, C-64 wysyłam pocztą. Katalog gratis po otrzymaniu koperty zwrotnej.
Nagrywanie programów komputerowych
ul. Lenina 104/3
58-304 Wałbrzych

B77

COMMODORE 64

Programy tanio!!
Katalogi gratis! koperta zwrotna

COMMOSOFT

13-230 Lidzbark
ul. Ogrodowa 3

B83

ATARI XE,XL SPECTRUM, TIMEX
NISKIE CENY PROGRAMÓW NA TAŚMIE I DYSKU
INTERFEJSY TURBO I „AY" DO SPECTRUM
CARTRIDGE DO ATARI — GRY I UŻYTKI
INFORMACJA — ZAADRESOWANA KOPERTA + ZNACZEK
05-220 ZIELONKA SKR. POCZ. 9/2

B80

MICROMAN

oferuje na miejscu lub wysyłkowo:

1. Sprzęt komputerowy i akcesoria.
2. Programy i literaturę do komputerów: Atari XL/XE/XT, Commodore 16/116/ +4/64/125/ Amiga, Spectrum,/Timex.
3. Oprogramowanie na cartrig'a
4. Przystawki „UNIVERSAL TURBO" do magnetofonów firmowych Atari umożliwiające zapis i odczyt programów w systemie Blizzard oraz Turbo 2000
5. Naprawy zasiłaczy, magnetofonów, klawiatur w komputerach Atari Commodore Spectrum.

Informacje: (na miejscu lub koperta zwrotna),
zamówienia:
— 40-181 Katowice ul. Osikowa 66
tel. 585-106
— 44-200 Rybnik ul. Wiejska 19 tel. 233-56

Firmowe punkty sprzedaży:
— Katowice ul. Plebiscytowa 31
— Rybnik D.H. „Hermes" I piętro
— Bielsko-Biała Plac Wolności 9

B88

ATARI, COMMODORE

— naprawy magnetofonów i zasilaczy do komputerów
— montaż systemu Turbo — Rom — Plus w magnetofonach Atari
(około 80 — 100 gier na kasecie C60, Feud w 1 min. 12 sek, również praca w „Blizzardzie" — licencja programowa firmy „Atares" z Chorzowa, Kartridże systemowe).
Honorujemy gwarancję firmową na przerobione magnetofony.
— Montaż wejść monitorowych w OTV turystycznych.

„PLUS"

Kraków ul. Mochnackiego 67 tel. 33-23-12
godz. 10 – 18 w sob. 9 – 13
Punkty przyjęć:
Tarnów ul. Traugutta 7/10 tel. 33-15-41
środy 16 – 18
Rzeszów ul. Rejtana 43/6 tel. 548-82
środy 10 – 14

B87

SKLEPY — GIEŁDY
DYSKIETKI

najtańsze, najwyższej jakości
oferuje

HURTOWNIA MEGABAJT II

skr. poczt. 28
03-912 Warszawa 33
tel. 17-76-16

Dostawy natychmiastowe

Ponadto oferujemy kartridże i interfejsy dla komputerów ATARI XL/XE i COMMODORE (m.in. SPARTA DOS X, BASIC XE/XL, ACTION, CENTRONICS, TURBO AST, X, BLACK BOX, FINAL II, FINAL III, ACTION PLUS) oraz programy i instrukcje dla ATARI.

Prowadzimy także sprzedaż detaliczną.

B90

KAŻDE PAŃSTWA ZAMÓWIENIE ZREALIZUJEMY

LHK

LHK Electronics Service S.A
81-736 Sopot
ul. Architektów 1A
*** tlx.512742 BSP PL**
PC/XT/AT/386/LAP

w dowolnych konfiguracjach, po atrakcyjnych cenach proponujemy: HD 20,30,40,80 MB, monitors 14":amber, paper-white, EGA
Zestawy do odbioru TV-SAT dla 4,8,16 użytkowników

LHK

TWOIM TRAFNYM WYBOREM

Uwaga! Odbiorcom hurtowym
udzielamy do 12% rabatu

NOWOŚĆ!

DYSKIETKA CZYSZCZĄCA NAPĘDY GŁOWIC KOMPUTEROWYCH

Dyskietka czyszcząca zapobiega zużywaniu się głowicy, przedłuża trwałość komputera, eliminuje wszelkie zanieczyszczenia bez ryzyka uszkodzenia komputera. Dyskietka może być używana wielokrotnie, posiada wymienne wkłady i wystarcza na kilka miesięcy pracy komputera.
Cena promocyjna 25000 zł.

Zamówienie

zamawiam ..... sztuk * 25000 zł
Nazwisko/firma .....
adres .....
wysyłka za zaliczeniem pocztowym

LHK prowadzi regenerację taśm do wszystkich typów drukarek
Regenerując taśmę — oszczędzasz pieniądze
Usługę realizujemy w 48 godzin — wysyłka za zaliczeniem pocztowym.

ZAGRAJMY W TO JESZCZE RAZ

```
LADY TUT              2392,50  L
                    * 4566,176 L
                    * 2847,99    ?
LANCER LORDS         16424,60  L
LASER STRIKE         16475,173 L
LASER ZONE            4971,177 L
LAST MISSION          7927,12
                sys     14848 L
LAZERFORCE           31611,197 L
                     11783,205 E
                sys     40680
LAZY JONES            2971,9   L
LE MANS               2547,x   X
LEGEND OF KAGE       13611,95
                sys      2344 L
LIGHTFORCE           13344,205 L
                    *40725,0   L
                    *14235,234
                     14236,234 L
                sys      2061
                    *11547,5
                sys      6713 L
LIVING-STONE         29744,173
                     31026,173
                sys     16384
LOCO                 26944,77  L
LODERUNNER            7892,255 S
                      2710,44  L
                      7636,x   LI

M

MAGGOT MANIA          2532,4   L
MANIC MINER          16432,5   L
                      1624,x   LI
                sys     16640
                    *16571,173
                    *16423,173
                sys     16384
                    *16423,5   Run
MATRIX                7629,238
                      7683,238 L
                    * 7983,238 L
MEGA APOCALYPSE      22589,1
                wiecej'gwiazd
                     21841,170
                     21842,72  dzwiekd
                     32417,173 S  1st
                     32509,173 S  2nd
                sys     22562
                     26070,12
                     32898,234
                     32899,234
                     32900,234
                     32901,234
                     32902,234
                     32903,234  szybkosc
MEGAZONE              6252,1-255 LI
MERMAID MADNESS      21244,208
                     21250,240
                     21290,234
                     21291,234
                         2,54
                sys     16384
METROCROSS           13501,234
                     13502,234
                      4356,11
                sys      4096 T
MINER                 9450,173 L
                    * 2652,165 L
                    * 2471,255 L
                      16424,x   LI
MINER 2049'ER         9450,173 L
                    * 2652,165 L
                    * 2471,255 L
                    *11510,255 L
                    * 2599,184
                      2600,17
                      2652,184
                      2653,18  L
MISSION ELEVATOR      2822,1-255 LI
MONSTER ATTACK       19228,234
                     19229,234 L
MONTEZUMAS REVENGE   36190,230 L

MONTY MOLE            9100,205 S
                    *39273,234
                     39274,234 L
MONTY ON THE RUN      3994,255 L
                    * 8926,205 L
MOON BUGGY           24151,173 L
                    *30624,173 L
                    *17388,173 L
MOON PATROL           2872,255 L
                    * 8300,5   L
                    * 2872,200 L
MOTOR MANIA           8646,255 L
MR. MEPHISTO         25922,44  L
                    *26432,44  L
                    *26372,0   L
                     40800,x   LI
MR. ROBOT            11518,255 S
                     11558,44  L
MRS PACMAN            8090,173 L
                    * 8091,173 L
                    * 7616,245 L
MUTANT CAMELS        11639,255 L
MUTANT MONTY         21647,173
                     21226,234
                     21227,234
                sys     22039 L
MUTANTS              14615,128
                    * 9273,230
                    * 9273,234
                      9274,234 L
                sys      4096 L
MYTICAL MISSION      23194,173
                sys      8192

N

NEMESIS              10239,166 S

NEMESIS               5975,234
                      5976,234
                      5977,234
                sys      5779 statek
NEPTUNE               7870,60  L
NEPTUNES DAUGHTER     7904,173 L
                    * 7909,173 L
                    * 7922,173 L
                    * 7927,173 L
                    * 7940,173 L
                    * 7945,173 L
                    * 9533,234 L
                    * 9521,234
                      9522,234
                      9533,234 L
                    * 7870,x   LI
                    * 7870,60  I
NOMAD                 5571,197 L

O

O'RILEYS MINE         9521,0   S
                      9523,170 L
                    * 9251,44    ?
OCEAN DECATHLON       9450,173 L
OERM                 14864,0   L
OMEGA RACE            6300,230 L
ORPHEUS AND-
  -THE UNDERWORLD    18870,234
                     18871,234 L
                     10118,x   LI
                sys      9728
OUT RUN              34711,234
                     34712,234
                     34713,234 T
                     33393,173 kolor
                sys     38045
```

Podsumowując sześć zeszłorocznych notowań Listy Przebojów tradycyjnie wyłoniliśmy dziesiątkę gier 1990 roku. Zasady punktowania jak przed rokiem — za pierwsze miejsce dziesięć, za drugie dziewięć punktów itd.

W tym numerze „Bajtka" brak już notowania Listy. Została ona przeniesiona w rozwiniętej formie do „Top Secret", którego trzeci numer pojawi się na rynku z końcem lutego.

1. BATMAN — Ocean

Cztery części przygód człowieka-nietoperza kryją się pod jedną nazwą „Batman". Trudno więc dokładnie powiedzieć, która z nich była tą ulubioną. Ale postacią minionego roku był na pewno BATMAN!

2. THE UNTOUCHABLES — Ocean

Gra oparta również na świetnym filmie, produkt tej samej firmy, co zdobywca I miejsca. Szybka akcja, kulturalna grafika i fortepianowe ragtime'y — oto, co dobarwia tę grę. Opis w pierwszym numerze „Top Secret" z października '90.

3. IRON LORD — Ubi Soft

Młody magnat wraca do kraju z wygnania. Na miejscu zastaje sieć intryg skorumpowanej rodziny oraz wrogie wojska. Musi zmontować rewolucję, pokonać nieprawego władcę i wygnać najeźdźcę z kraju. Gra to w sumie standardowa, lecz wykonana w niecodziennym klimacie. Recenzja w trzecim numerze „Top Secret".

4. HOSTAGES — Infogrames

Legendarna gra, znana ze świetnej grafiki i muzyki, ostatnio pojawiła się na Spectrum! Nie ma jej tylko na małe Atari, lecz powinna pojawić się w tej wersji czym prędzej. Szturm SAS na opanowaną przez terrorystów ambasadę dostarcza wrażeń porównywalnych z odpytką z biologii w klasie matfiz.

5. INDIANA JONES III — Ocean

Ostatnia Wyprawa Krzyżowa, tym razem w wykonaniu Harrisona Forda. Święty Graal — biblijny kielich krwi Chrystusowej, poszukiwany przez tysiąclecia staje się obiektem pożądania faszystów. Indiana Jones i jego ojciec mają przeszkodzić świętokradztwu i zapewnić wieczny spokój Graalowi.

6. RED HEAT — Ocean

To także reminiscencja głośnego filmu, tym razem ze Schwarzeneggerem i Belushim w rolach głównych. Policjanci — rosyjski i amerykański razem rozbijają międzynarodowy gang narkotyczny. Gra uproszczona do niemożliwości, w akcji tylko prawe ramię Arnolda.

7. ALIENS — Activision/Electric Dreams

Słynna wyprawa ślicznej Sigourney Weaver przeciwko pladze kosmicznych stworów — Obcych. Gra wieloetapowa, przekładana scenami i dialogami z filmu, ostatnio pojawiła się również na Spectrum. Mapa i opis w „zerowym" numerze „Top Secret".

8. ROBBO — Laboratorium Komputerowe Avalon

Rewelacyjna polska gra, niejako połączenie Sokobana i Boulder Dasha. Niecodzienny pomysł i przyzwoite wykonanie to efekty ciężkiej pracy programistów Avalonu, którzy zręcznie wypełniają lukę na rynku gier ośmiobitowego Atari.

9. FIGHTER BOMBER — Activision

Symulator lotu jak wszystkie, ale... Jest w nim kondensat wieloletnich doświadczeń w opracowywaniu gier symulacyjnych. Walka powietrzna, akrobacje, lądowania. Kompleksowy opis w trzecim, najbliższym numerze „Top Secret".

10. WAR IN MIDDLE EARTH — Melbourne House

Trylogia Tolkiena w odświeżonej skomputeryzowanej formie. Elementy taktyczne i zręcznościowe oraz dokładne odtworzenie Wojny z kart książki. Bilbo Baggins, czarodziej Gandalf, elfy, krasnoludy. Opis w „Top Secret" nr 1.

Pilnie szukam następujących gier na CPC 464: PIT STOP, TAPPER, PACMAN, KAPERS, SKYFOX. W zamian oferuję WINTER GAMES, GRAND PRIX i wiele innych gier. **Marcin Chabros, ul. 3-go Maja 7/13, 08-110 Siedlce**

Szukam pilnie gier na Atari: OUT DAN, GUERILLA WAR, SHOIN, KUNG-FU, BARBARIAN, GRYZOR. W zamian inne gry, opisy i wymiana doświadczeń. **Rafał Popek, ul. Zachodnia 53/5, 91-063 Łódź**

Mam Atari 65 XE i magnetofon firmowy. Szukam gier: THE TRAIN, STREET FIGHTER, RENEGATE. W zamian 10 gier. Proponuję wymianę oprogramowania. **Marek Pluta, ul. Noniewicza 10a/90, 16-400 Suwałki**

Poszukuję gier RAMBO, COMMANDO, BARBARIAN, GLADIATOR, UNIVERSAL HERO wraz z opisami na Atari. W zamian inne ciekawe gry z opisami. **Piotr Sondej, ul. Wita Stwosza 50/52, 80-308 Gdańsk-Oliwa**

Pilnie poszukuję gier na Spectrum TRAPDOOR i KING'S QUEST. Proponuję książki: „Przewodnik po ZX Spectrum", „Świat mikrokomputerów" a także program „Grafika" na Spectrum. Wersja kasetowa. **Krzysztof Łański, ul. Kłodnicka 62b/15, 41-706 Ruda Śląska**

Jak zrobić małe litery na komputerze Commodore 64? Głowię się nad tym od dawna — może ktoś mi pomoże? **Cecylia Ormaniec, ul. Łużycka 2a/13, 41-902 Bytom**

Jestem posiadaczem Commodore 16. Proszę o jakieś wiadomości o grach: PIN-POINT, SABOTEUR, POWER-BALL, RAIDER. Interesują mnie też nieśmiertelności. **Wojciech Horbulewicz, ul. Nowa 17a/6, 41-908 Bytom 8 Miechowice**

Mam Atari 65 XE z magnetofonem XC12. Proszę o grę BARBARIAN oraz kod, który ją uruchamia. Proszę również o mapę do gry PITFALL II. W zamian oferuję programy kopiujące. **Daniel Śniegoń, ul. Stawowa 27c/118, 41-400 Mysłowice**

Poszukuję gier: BARBARIAN I i II, GAME OVER I i II, DESTROYER, COMMANDO. W zamian inne. Mam Atari 65 XE. **Emanuel Olszewski, ul. Żmigrodzka 26/10, 51-120 Wrocław**

Bardzo proszę o opisy do gier: ASTERIX, TERRAMEX, ZORRO, WILLY IN PARADISE na Spectrum+ oraz NEW YORK CITY, GREEN BERET, TARZAN na Atari 800 XL. **Bartosz Roguski, ul. Stachiewicza 29/52, 31-303 Kraków**

DODATKOWA PAMIĘĆ RAM W CPC-6128

Komputer Amstrad CPC-6128 wyposażony jest w 128K pamięci RAM. Procesor Z-80 „widzi" maksymalnie 64K. Po co więc dodatkowe kilobajty? CPC 6128 był projektowany jako komputer półprofesjonalny, konieczne więc było umożliwienie korzystania z „cywilizowanego" systemu operacyjnego. Na taki system wybrano CP/M Plus firmy Digital Research.

Wymaga on około 64 kilo na swój kod. Gdyby uruchomić go na komputerze wyposażonym w 64K pamięci, na program użytkownika zostałoby... kilkaset bajtów! Stąd 128K. Jednakże dodatkowe 64K pamięci można wykorzystać do różnych celów, takich jak: RAMDYSK, pamięć bazy danych, przechowywanie obrazków, prawdziwe okna i wielu innych, zależnych tylko od potrzeb i umiejętności, a nie tylko jako pamięć dla systemu CP/M Plus. Przykładami mogą być: TASWORD 6128 (z pomocą którego piszę ten tekst, mieści on w pamięci „tylko" 65276 znaków — zgadnijcie gdzie!), MASTERFILE III, cała seria programów typu RAM-DYSK, Bank Manager... Chyba wystarczy.

PRIMO: Nie wszystko na raz!

Pamięć CPC-6128 podzielona jest na dwa banki po 64K. Jest to spowodowane typem procesora i układów współpracujących — mogą one obsługiwać co najwyżej 64K RAM. Podłączenie całych 128K na raz nic by nie dało, połowa pamięci i tak byłaby niedostępna. Tak więc podzielono ją i zamontowano układ przełączający. Wybór konfiguracji należy do programu: chce 128K? To niech sobie włączy! Nie chce? To nie! We wszystkie kłopoty związane z przełączaniem pamięci pakujemy się całkowicie dobrowolnie, i jeśli coś nie wyjdzie, to (niestety!) będzie można winić tylko siebie.

Nagłe przełączenie całego obszaru pamięci miałoby „mało zabawne" konsekwencje: zniknąłby program, stos i inne obszary pamięci niezbędne dla pracy systemu operacyjnego oraz programu użytkownika. Więc dodatkowa pamięć jest dostępna tylko dla programów pisanych przez fachowców i/lub geniuszy? Nic takiego! Po prostu nie przełącza się całych 64K na raz. Na raz można dostać ćwiartkę dodatkowej pamięci, czyli 16K — akurat na jeden obrazek. Ten kawałek dodatkowej pamięci zastępuje odpowiedni obszar normalnej pamięci. Obszar zmian zaczyna się od adresu #4000 (16384) i kończy na #7FFF (32767). Jest to oczywiście 16K, ale za to jakie! Pięć szesnastokilobajtowych bloków do wyboru.

SECUNDO: Tylko ten i żaden inny!

Włączając blok dodatkowej pamięci w "pole widzenia" procesora musimy jednoznacznie określić, który to ma być blok. Numery bloków są zakodowane w sposób jednoznaczny i zrozumiały: 0 — pamięć standardowa, 4 — dodatkowy blok nr 0, 5 — dodatkowy blok nr 1, 6 — dodatkowy blok nr 2 i 7 — dodatkowy blok nr 3. Można zauważyć, że bloki dodatkowe mają kod równy "numer bloku plus 4". Wynika to z tego, że dodatkowa pamięć wybierana jest przez bit numer 2 o wadze 4. Stąd bajt konfiguracji można opisać w sposób następujący: "11000WNN", gdzie W to bit wybierający dodatkową pamięć, a para NN to numer bloku tej pamięci. Dwie jedynki na początku to sygnał, że zmieniamy konfigurację pamięci. Jeśli wybieramy pamięć z drugiego banku, to W=1, a NN=numer bloku: jeśli pamięć z pierwszego banku, to W=0 i NN=0. A co jeśli W=0 i NN=(powiedzmy) trzy? Otóż oznacza to jedną z "dzikich" konfiguracji używanych przez CP/M Plus: bloki z pierwszego i drugiego banku są wtedy pomieszane.

TERTIO: Jak!?

Przełączanie pamięci jest proste jak drut (ściślej: jeszcze prostsze). Należy załadować do akumulatora bajt konfiguracji i wywołać procedurę KL RAM CONFIGURATION o adresie #BD5B. Procedura ta ustawia konfigurację pamięci i zapamiętuje ją. Jeśli ktoś chce bezpośrednio, to proszę bardzo: do rejestru B ładujemy #7F (127) a do rejestru C żądaną konfigurację i wykonujemy rozkaz OUT (C), C — spowoduje to przesłanie nowej konfiguracji do układu VGA (Video Gate Array), który jest odpowiedzialny (między innymi) za konfigurację pamięci. Potem mamy już żądaną konfigurację. Należy pamiętać o dwóch zastrzeżeniach: 1 — "nasz" program musi znajdować się poza obszarem #4000—#7FFF (16384—32767), 2 — po zakończeniu zadania powinno się przywrócić konfigurację standardową.

Dlaczego tylko w asemblerze? Bo asembler jest najwłaściwszym językiem do takich machlojek. Poza tym, zakładam, że pamięć w pierwszym banku (dla BASIC-a) nie może zostać ograniczona, czyli przełączenia i kopiowanie bloków muszą odbywać się w sposób niewidoczny dla interpretera! Jakiekolwiek dane czy kawałek programu znajdujący się w obszarze podmiany musi pozostać nieuszkodzony. Tylko procedura w asemblerze może tego dokonać: "mignąć" blokiem dodatkowej pamięci na czas przepisywania i przywrócić normalną konfigurację. Wyłączony na ten czas BASIC "niczego nie zauważy" i nie padnie.

QUARTO: Przykład

Podam prosty przykład PROGRAMiku przechowującego obrazki w drugim banku. Może on zapamiętać cztery obrazki i na żądanie odtworzyć je na ekranie. Listing nr 1 jest w asemblerze, czyli tylko dla tych, którzy się na tym cokolwiek znają. Należy wpisać ten programik do asemblera (polecam Laser Genius) a następnie zasemblować, zapisując sobie adresy procedur SAVE i LOAD. Teraz, aby zapamiętać obrazek, należy użyć rozkazu "CALL ‹adres procedury SAVE›,numer obrazka" (w BASIC-u) lub "LD A,kod bloku" i "CALL SAVE1" w asemblerze. Dla odczytu będą to odpowiednio "CALL ‹adres procedury LOAD›,numer obrazka" oraz "LD A,kod bloku" i "CALL LOAD1".

Dla tych, którzy na sam dźwięk słowa "asembler" dostają dreszczy, mam prostszy przykład: Listing nr 2 zawiera trzy kawałki BASIC-a. Pierwszy instaluje kod maszynowy (a jednak!) pozwalający przełączać banki oraz szybko kopiować bloki po 16K (BASIC potrzebowałby ponad minuty na przepisanie bloku). Dwa następne kawałki to procedury zapamiętywania (1000—1040) i odtwarzania (2000—2040) obrazka. Za pomocą tych procedur można zapamiętać cztery obrazki (tak samo jak za pomocą procedur asemblerowych z listingu nr 1). Do wiadomości ciekawskich podaję: procedura w kodzie maszynowym współpracująca z BASIC-em jest znacznie uproszczoną wersją procedury z listingu 1, umieszczona jest w "dziurze" (nieużywanym kawałku pamięci) &BE80-&BEFF. Wreszcie listing nr 3 zawiera program typu SLIDE SHOW (po polsku: pokaz slajdów), używający procedur z listingu nr 2. Program ten wyświetla na zmianę jeden z czterech wcześniej załadowanych obrazków. Uwaga: należy wpisać procedury z listingu nr 2 i dopiero do nich dołączyć program z listingu nr 3. Osobno nie będzie działać

QUINTO: BASIC-u mój kochany!

W "Komputerze" nr 5/90 ukazał się artykuł Tadeusza Panasiewicza pt. „Przełączanie banków RAM w Amstradzie CPC 6128". Artykuł przedstawia sprawę w sposób raczej skrótowy, a poza tym, choć doradzana przez jego autora metoda przełączania banków jest prosta i szybka, to jest ona również śmiertelnie niebezpieczna. Można w ułamku sekundy stracić cały program i wszystkie dane. Otóż pan Panasiewicz proponuje użycie rozkazu "OUT &7F00,kod bloku" z poziomu BASIC-a. Można i tak. Jeśli jednak używa się tej metody, trzeba ograniczyć pamięć na BASIC do 16K rozkazem "MEMORY &3FFF". Poza tym, jak sam to zauważył, przepisywanie bloku o długości 16 kilo przez program w BASIC-u trwa dosyć długo (jak na mój gust, znacznie za długo). Dlatego też, wydaje mi się, że mój sposób jest lepszy. Kod maszynowy zrobi to szybciej i bezpieczniej, zostawiając całą pamięć (41 kilo zamiast 16) na program w BASIC-u i inne dane. Wybór metody zależy od upodobań programisty (czyli Ciebie, Czytelniku) i konkretów programu. Uważam jednak, że BASIC nie nadaje się do takich zadań jak bezpośredni dostęp do sprzętu, jest po prostu zbyt wolny. Kombinacja BASIC-a z asemblerem, którą proponuję, jest szybsza niż BASIC i łatwiejsza w użyciu niż asembler.

SEXTO: Już koniec?!

To już wszystko na temat wykorzystania dodatkowej pamięci w CPC-6128. Reszta należy do Was, drodzy Czytelnicy. Co zrobicie z tą „wiedzą tajemną" to już Wasza sprawa. Zastosowań jest wiele i każdy znajdzie odpowiednie dla siebie. Odtąd nie będę zwracał uwagi na usprawiedliwienia typu „A bo mi zabrakło pamięci..." Uff, skończyłem. Teraz Wy się będziecie martwić „co z tym zrobić".

POST SCRIPTUM: W następnym odcinku...

W następnym odcinku rzecz jeszcze lepsza czyli „RSX dla każdego!" (chodzi oczywiście o dodatkowe komendy BASIC-a, a nie o rozszerzenia systemu CP/M). Później „Wgląd w AMSDOS, czyli co programiści z Locomotive Software ukryli przed profanami", skąd dowiecie się jak sformatować dyskietkę, wczytać i zapisać sektor i nie tylko! W dalszej przyszłości: „Praktyczne wykorzystanie przerwań" czyli jak zrobić zegarek, lub jak podzielić ekran na kawałki bez użycia młotka i piły. A o tym, co będzie dalej, to nawet najstarsi górale nie wiedzą. Ale ja tu zostaję i może coś wymyślę...

Skargi, wnioski, pochwały i krytyki proszę przesyłać na adres redakcji. Uprasza się o nieprzysyłanie bomb zegarowych oraz anonimów.

Nadmiernie ciekawym podaję źródła mojej wiedzy: Wypatroszony BANK MANAGER oraz "CPC INTERN" (Data Becker).

Michał Szokoło

LISTING 1: PROCEDURY ASEMBLEROWE

```
KLRAM:  EQU #BD5B ; ADRES PROCEDURY PRZEŁĄCZAJĄCEJ

SAVE:   CALL    SCRNUM ;NUMER ZAPAMIETYWANEGO OBRAZKA
SAVE1:  CALL    KLRAM   ; PRZEŁĄCZ
        LD      DE,#4000
        LD      HL,#C000
        LD      BC,#4000
        LDIR            ; PRZESYŁAMY 16K
        XOR     A       ; AKUMULATOR NA ZERO
        CALL    KLRAM   ; KONFIGURACJA STANDARD
        RET             ; POWROT

LOAD:   CALL    SCRNUM
LOAD1:  CALL    KLRAM
        LD      DE,#C000
        LD      HL,#4000
        LD      BC,#4000
        LDIR
        XOR     A
        CALL    KLRAM
        RET

SCRNUM: LD      A,E     ; W DE PARAMETR
        AND     3       ; NUMER 0-3
        OR      #C4     ; USTAW 11000011NN
        RET             ; W AKUMULATORZE BAJT
                        ; KONFIGURACJI
```

LISTING 2: PROCEDURY BASIC-OWE

```
10 REM instalacja procedury przełączania i przepisywania
20 REM (C)1990 M. Szokoło & "BAJTEK"
30 DATA 3E,00,CD,5B,BD,21,00,C0,11,00,40,01,00,40,00
40 DATA ED,B0,AF,C3,5B,BD
50 FOR A=0 TO 20:READ A$:POKE &BE80+A,VAL("&"+A$):NEXT
60 REM procedura zainstalowana
70 STOP

1000 REM Zapamiętanie obrazka w drugim banku
1010 REM Zmienna "obr" zawiera numer bloku
1020 obr=(obr AND 3)OR &C4:POKE &BE81,obr
1030 POKE &BE8E,0:CALL &BE80
1040 RETURN

2000 REM Odtworzenie obrazka na ekranie
2010 REM Zmienna "obr" zawiera numer bloku
2020 obr=(obr AND 3)OR &C4:POKE &BE81,obr
2030 POKE &BE8E,&EB:CALL &BE80
2040 RETURN
```

LISTING 3: POKAZ SLAJDOW

```
 10 REM SLIDE SHOW czyli POKAZ SLAJDOW
 70 REM w linii 90 dane nazwy obrazków do załadowania
 80 REM z dysku. Na końcu numer trybu graficznego
 90 DATA screen1.scr,screen2.scr,screen3.scr,screen4.scr,1
100 DIM scr$(4):FOR a=1 TO 4:READ scr$(a):NEXT A
110 READ tryb:MODE tryb
120 INK 0,0:INK 1,3:INK 2,16:INK 3,25 :REM ustaw kolory
130 FOR a=1 TO 4
140 LOAD scr$(a),&C000 :REM ładujemy obrazek
150 obr=a:GOSUB 1000   :REM i zapamiętujemy go
160 NEXT a
170 REM wchodzimy do głównej pętli
180 obr=1
190 WHILE obr<>-1 :REM ustawienie obr na -1 kończy pokaz
200 GOSUB 2000    :REM pokaż obrazek
210 obr=obr+1:IF obr=5 THEN obr=1
220 a$="":WHILE a$="":a$=INKEY$:WEND :REM czekaj na klawisz
230 IF a$="K" or a$="k" THEN obr=-1 :REM K=KONIEC
240 WEND :REM zamknięcie pętli
250 END
260 REM koniec głównej części programu
270 REM
```

PROCEDURY SYSTEMOWE AMSTRADA
część II

PAKIET SYSTEMU OPERACYJNEGO (KERNEL)

Pakiet KERNEL zawiera procedury obsługi RSX-ów (dodatkowych komend), przerwań, zdarzeń (events) oraz zegara systemowego.

KL CHOKE OFF — „MAŁY RESET" SYSTEMU — #BCC8

Kasuje wszystkie przerwania i zdarzenia (z wyjątkiem systemowych), obszary danych ROM-ów rozszerzających i RSX-y. Przygotowuje do MC BOOT PROGRAM.
Wejście: nie ma.
Wyjście: B zawiera kod aktywnego ROM-u, C zawiera kod ROM-u dla programu w pamięci RAM. DE zawiera adres startowy aktywnego ROM-u. AF i HL skasowane.

KL ROM WALK — INICJALIZUJ ROM-Y ROZSZERZAJĄCE — #BCCB

Inicjalizuje ROM-y rozszerzające o numerach 0-15 (w kolejności 15...0), przydziela im obszary danych, podłącza RSX-y do listy itp.
Wejście: DE zawiera adres pierwszego wolnego bajtu RAM.
HL zawiera adres ostatniego wolnego bajtu.
Wyjście: HL i DE zawierają uaktualnione dane, AF i BC skasowane.

KL INIT BACK — INICJALIZUJ ROM ROZSZERZAJĄCY — #BCCE

Jak KL ROM WALK ale dla jednego ROM-u.
Wejście: HL i DE jak KL ROM WALK, C zawiera numer ROM-u.
Wyjście: jak KL ROM WALK, ale C niezmienione.

KL LOG EXT — PODŁĄCZ RSX — #BCD1

Instaluje RSX (Resident System Extension).
Wejście: HL zawiera adres 4 wolnych bajtów na podłączenie do listy RSX-ów, BC zawiera adres tablicy skoków.
Wyjście: DE skasowane.
Format tablicy skoków:
Adres tablicy nazw komend (2 bajty)
skoki do procedur (po 3 bajty)
Tablica nazw składa się z nazw komend pisanych dużymi literami. Ostatni znak każdej nazwy musi mieć ustawiony najstarszy bit. Bajt o wartości 0 oznacza koniec tablicy.
Uwaga: nazwy nie powinny być dłuższe niż 16 znaków. Żadna z tablic nie może leżeć pod ROM-em.

KL FIND COMMAND — PODAJ ADRES PROCEDURY RSX-A — #BCD4

Podaje adres wywołania komendy RSX.
Wejście: HL zawiera adres nazwy RSX-a.
Wyjście: A, B i DE zawsze skasowane. Jeśli CY=1 to komenda została znaleziona i HL zawiera jej adres a C — kod ROM-u. Jeśli CY=0 to nie znaleziono komendy. HL i C skasowane.

KL NEW FRAME FLY — STWÓRZ PRZERWANIE „FRAME FLYBACK" — #BCD7

Przygotowuje i podłącza przerwanie „ramki" (powrotu promienia elektronów) do kolejki przerwań.
Wejście: HL zawiera adres bloku przerwania (9 bajtów), DE — adres procedury, B — klasę zdarzenia a C — kod ROM-u.
Wyjście: AF, DE i HL skasowane.

KL ADD FRAME FLY — PODŁĄCZ PRZERWANIE „FRAME FLYBACK" — #BCDA

Podłącza blok przerwania powrotu ramki do kolejki przerwań.
Wejście: HL zawiera adres bloku przerwania.
Wyjście: AF, DE i HL skasowane.

KL DEL FRAME FLY — SKASUJ PRZERWANIE „FRAME FLYBACK" — #BCDD

Usuwa blok przerwania powrotu ramki z kolejki.
Wejście: HL zawiera adres bloku przerwania.
Wyjście: AF, DE i HL skasowane.

KL NEW FAST TICKER — STWÓRZ PRZERWANIE „FAST TICKER" — #BCE0

Jak KL NEW FRAME FLY, ale dla tzw. szybkiego zegara (przerwanie następuje 300 razy na sekundę)

KL ADD FAST TICKER — PODŁĄCZ PRZERWANIE „FAST TICKER" — #BCE3

Podłącza przerwanie szybkiego zegara do kolejki.
Parametry jak KL ADD FRAME FLY.

KL DEL FAST TICKER — SKASUJ PRZERWANIE „FAST TICKER" — #BCE6

Odłącza przerwanie szybkiego zegara.
Parametry jak KL DEL FRAME FLY.

KL ADD TICKER — PODŁĄCZ PRZERWANIE ZEGAROWE — #BCE9

Podłącza przerwanie zwykłego zegara (50 razy na sek.) do kolejki.
Parametry jak KL ADD FRAME FLY.

KL DEL TICKER — SKASUJ PRZERWANIE ZEGAROWE — #BCEC

Usuwa przerwanie zwykłego zegara z kolejki.
Parametry jak KL DEL FRAME FLY.

KL INIT EVENT — PRZYGOTUJ BLOK ZDARZENIA — #BCEF

Przygotowuje blok zdarzenia.
Wejście: HL zawiera adres bloku, DE — adres procedury, B — klasę zdarzenia a C — kod ROM-u.
Wyjście: AF, BC, DE i HL skasowane.

KL EVENT — UAKTYWNIJ ZDARZENIE — #BCF2

Aktywuje zdarzenie (umieszcza blok w kolejce zdarzeń).
Wejście: HL zawiera adres bloku zdarzenia.
Wyjście: AF, BC, DE i HL skasowane.

W następnym odcinku dokończenie pakietu systemu operacyjnego (KERNEL), oraz pakiet kodu maszynowego (MACHINE CODE).

Michał Szokoło

SZTUCZKI I KRUCZKI (AMSDOS)

W pierwszym odcinku cyklu porad fachowych znajduje się mały, ale przydatny programik do odkasowywania pomyłkowo skasowanych plików. Nie raz (i nie dwa, ale znacznie więcej razy) zdarzyło mi się skasować jakiś zupełnie niewinny plik. Potem musiałem „grzebać" na dysku aby go odkasować (zmarnowane pół godziny). Ponieważ zdarza się to wszystkim, w tym także osobom, które nie potrafią „grzebać" na dysku (lub nie mają czym), napisałem program „Odkasuj!".

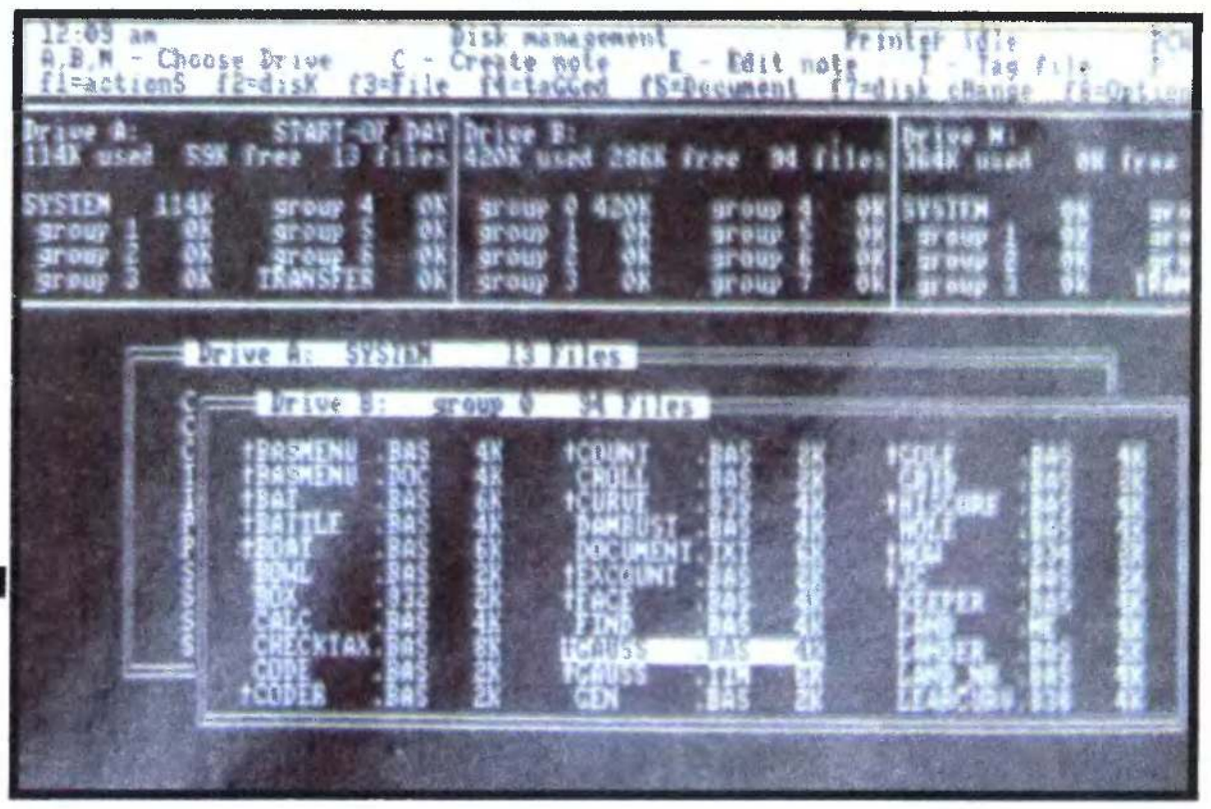
Programik ten jest naprawdę BARDZO krótki, szczególnie, biorąc pod uwagę jego skuteczność, dochodzącą do 90% (niestety, czasem na dysku jest taki śmietnik, że nie potrafi on skutecznie zadziałać). Drugą jego zaletą jest to, że nie trzeba ręcznie „grzebać" na dysku, wszystko jest automatyczne (i to w 92% robione przez system!), a zadanie użytkownika ogranicza się do wpisania nazwy pliku, wybranej z wyświetlonej listy. Przed uruchomieniem programu trzeba tylko ustawić numer użytkownika dla odzyskiwanych programów (rozkazem :USER).

Uwaga: gwarantuje 95% skuteczność pod warunkiem, że między skasowaniem pliku a uruchomieniem programu „Odkasuj!" nic nie było zapisywane na dyskietkę.

(M. Sz.)

```
100 '
110 ' Odkasuj!
120 ' (c)1990 Arnold Isp
130 '
140 MODE 2 : CALL &BC02
150 PAPER 0 : PEN 1
160 PRINT CHR$(24);" ODKASUJ! ";CHR$(24):
170 PRINT " (c)1990 by Arnold Isp"
180 PRINT
190 INPUT "Który dysk (A/B) : ",d$
200 d$=UPPER$(LEFT$(d$,1))
210 IF d$<>"A" AND d$<>"B" THEN 230
220 |DRIVE,@d$
230 au=PEEK(&B8E8)+256*PEEK(&B8E9)+1
240 IF au=0 OR au=65535 THEN ERROR 32
250 usr=PEEK(au) : POKE au,&E5
260 |DISK : CAT
270 PRINT "Nazwa pliku (bez ? i *), ";
280 INPUT "RETURN jeśli koniec: ",f$
290 IF f$="" THEN 330
300 f1$=STR$(usr)+":"+f$
310 |REN,@f1$,f$ : f$=""
320 GOTO 260
330 POKE au,usr : PRINT
340 PRINT "Do widzenia!" : PRINT
350 END
```

PCW SuperDOS



Rys. 1 Ekran programu PCW SuperDOS

Większość posiadaczy Amstrada PCW, posługujących się Locoscriptem dość rzadko korzystała z systemu operacyjnego CP/M Plus. Częściowym usprawiedliwieniem tego faktu była niewielka „przyjazność" tego systemu w stosunku do tego, co oferował dostarczany razem z komputerem edytor tekstów.

W przypadku MS-DOS'a (system operacyjny IBM PC) problem ten został rozwiązany przez wprowadzenie na rynek programów — nakładek typu XTREE, Norton Commander lub 1DirPlus. Na podobne nakładki dla PCW trzeba było czekać trochę dłużej. Dopiero w roku 1990 angielska firma Encyclasoft zaprezentowała program o nazwie SuperDOS, wykazujący duże podobieństwo funkcjonalne do wspomnianych nakładek. Z kolei układ ekranu i sposób komunikacji z użytkownikiem bardzo przypomina w działaniu Locoscript.

Zasadnicze funkcje programu są bardzo zbliżone do tych, którymi dysponuje opisywany wcześniej w Bajtku program NSweep. Różnice dotyczą przede wszystkim wyglądu ekranu i wygody posługiwania się tymi narzędziami.

Obserwując podstawowy ekran programu SuperDOS (ryc. 1) możemy wyróżnić trzy główne okna. W pierwszym, górnym oknie znajduje się logo programu z krótką ściągawką opisująca znaczenie klawiszy funkcyjnych. Okno środkowe zawiera ogólne informacje o zajętości poszczególnych napędów (A:, B: i M:). Najniższa, ale o największym rozmiarze, część ekranu wykorzystana jest na dane o plikach znajdujących się na dyskiecie w konkretnym napędzie.

Posługując się klawiszami sterującymi ruchem kursora i klawiszem F4 można wybierać poszczególne pliki do operacji typu kopiowanie, zmiana nazwy lub usunięcie. Z kolei klawisz F1 pozwala na wykonanie wskazanego programu, co jest znacznie prostsze niż wpisanie jego nazwy i naciśnięcie klawisza ENTER.

SuperDOS umożliwia też tworzenie, edycję i drukowanie niewielkich (2-5KB) plików tekstowych. Jest to użyteczne przy adresowaniu kopert lub sporządzaniu krótkich notatek. W najnowszej wersji programu dodano możliwość bezpośredniego uruchamiania programów w języku BASIC poprzez wskazanie odpowiedniego pliku.

Nie ma jednak róży bez kolców. Opisywany program oprócz wielu zalet ma też kilka wad. Jedną z nich są spore wymagania pamięciowe — min. 512 KB. Szybkość działania programu, napisanego przy pomocy Hisoft Pascala też nie jest rewelacyjna. Dla osób zaawansowanych pewnym problemem może być ograniczenie liczby wyświetlanych użytkowników (ang. users) do pierwszych ośmiu (0–7).

Jednak mimo wszystko SuperDOS jest programem godnym polecenia wielu właścicielom Amstrada PCW, zainteresowanym wydajniejszym wykorzystaniem ich komputera przy pracy w systemie CP/M Plus. Potencjalnych nabywców może trochę zniechęcić wysoka cena tego produktu wynosząca 30 funtów.

JM

DRUKOWANIE EKRANU

PCW

Dużą wygodą komputera Amstrad PCW jest możliwość wydrukowania w każdej chwili dowolnej zawartości ekranu. Jedyną wymaganą operacją jest jednoczesne naciśnięcie klawiszy ‹EXTRA› i ‹PTR›.

Czasami jednak zachodzi potrzeba wykonania wydruku przy pomocy własnego programu bez angażowania użytkownika komputera do naciskania co pewien czas dwóch klawiszy. Proste rozwiązanie tego problemu przedstawia procedura PrintScreen umieszczona w przykładowym programie. Wykorzystując nieudokumentowane odwołanie do modułu XBIOS, procedura ta pozwala na programowy wydruk ekranu.

Jonasz Mayer

```
Program Software_Screen_Dump_Demo;
{*****************************************}
{          (c) JM 1990                    }
{  Przykladowe zastosowanie procedury     }
{          PrintScreen                    }
{*****************************************}

procedure PrintScreen;
{*****************************************}
{ Wydruk calego ekranu - programowy odpo- }
{ wiednik nacisniecia klawiszy  EXTRA PTR }
{*****************************************}
begin
  inline ($cd/$5a/$fc/$72/$14);
end; { of print screen }

{*****************************************}
var
    i : integer;

begin

 clrscr;
 for i := 1 to 5
 do writeln ('software screen dump';50);

 printscreen;

end.
{*****************************************}
```

Drukowanie według ekranowego generatora znaków

Pewną nieprzyjemną cechą Amstrada PCW jest zróżnicowanie istniejące między zestawem znaków dostępnym na ekranie i drukarce.

Generator ekranowy pozwala na wyświetlenie 256 różnych znaków. Kody poniżej 32 uzyskuje się poprzedzając je kodem 27 (ESC). W przypadku drukarki, pracującej zgodnie ze standardem Epson FX80, sytuacja jest inna. Możliwe jest wydrukowanie znaków o kodach w zakresie 32-127. Znaki o kodach 161-255 są drukowane jako pochyłe (italic). Dostępne są też niektóre znaki z zakresu 0-32 i 128-160. Nie jest także możliwe w prosty sposób definiowanie własnych znaków generatora drukarki.

Częściowym rozwiązaniem tych problemów jest wykorzystanie trybu graficznego drukarki i drukowanie tekstów według ekranowego generatora znaków. Ten ostatni może być zmieniany (Bajtek 5/88).

Na listingu 1 (plik BDOS5.MAC) przedstawiono program w asemblerze, który umożliwia wygenerowanie RSX'a zmieniającego odwołanie do piątej funkcji modułu BDOS (List Output). Generacja RSX'a odbywa się w sposób opisany w Bajtku 12/89 i polega na wykonaniu następujących poleceń:

M80 = BDOS5
LINK BDOS5 [OP]
RENAME BDOS5.RSX=BDOS5.PRL
GENCOM BDOS5 [NULL]

Polecenie **PIP** pozwala na skopiowanie dowolnego pliku tekstowego na drukarkę. Wykonanie przed tą operacją komendy **BDOS5** umożliwi drukowanie zbioru w sposób graficzny przy wykorzystaniu ekranowego generatora znaków. Jeśli zamierzamy częściej stosować tę opcję, to możliwe jest dołączenie RSX'a do programu PIP.

Korzystanie z tej możliwości przy wydrukach w programach pascalowych wymaga dalszej pracy. Kompilator Turbo Pascala przy dostępie do drukarki odwołuje się do funkcji modułu BIOS. Zmiana tego stanu rzeczy, tzn. korzystanie z odwołań do modułu BDOS, a w dalszej konsekwencji drukowanie tekstów w trybie graficznym, jest możliwa dzięki procedurom zawartym w pliku BDOS5.INC (listing 2).

Jeśli po dołączeniu tego pliku do własnego programu wywołamy procedurę OWNLST z argumentem TRUE, to wszystkie wydruki będą w trybie graficznym. Powrót do zwykłego trybu tekstowego zapewni wywołanie z opcją FALSE. Warunkiem koniecznym jest uruchomienie RSX'a BDOS5 lub dołączenie go przy pomocy polecenia **GENCOM** do własnego programu lub kompilatora Turbo Pascala.

W następstwie opisanych działań możemy w łatwy sposób uzyskać na drukarce polskie litery, kody rameł i inne specjalne znaki w napisanych przez siebie programach pascalowych.

Jonasz Mayer

LISTING 1 Plik BDOS5. MAC

```
; ***********************************************
; *                                             *
; *    Plik BDOS5.MAC    (C) JM 1988            *
; * RSX przechwytujacy odwolanie do funkcji     *
; * modulu BDOS o numerze 5 (LIST OUTPUT).      *
; * Drukowanie kazdego znaku odbywa sie w trybie *
; * graficznym. Dolaczenie tego RSX'a do programu *
; * PIP.COM powoduje, ze instrukcja             *
; *    PIP LST:=nazwa_pliku                     *
; * wydrukuje plik w trybie graficznym posługując *
; * sie ekranowym generatorem znakow. Znaki o kod. *
; * ponizej 32 sa poprzedzane kodem 27 (ESC).   *
; * Wersja na komputer AMSTRAD PCW 8256/8512.   *
; *                                             *
; ***********************************************
screen equ 00E9H
xbios  equ 0FC5AH
RSG    equ 0B800H  ; adres generatora
,z80               ; znakow
cseg
; rsx header
ds     6
jp     start
bdos:  jp     0
dw     0
db     0FFH
db     0
db     'BDOS 5  '
db     0,0,0
start: ld     a,c
cp     05
jp     nz,bdos
begin: ld     a,(escFlag)
cp     01          ; czy byl <ESC> ?
jp     nz,test
xor    a
ld     (escFlag),a
jp     enter
test:  ld     a,e
cp     27          ; czy jest <ESC> ?
jp     nz,cont
ld     a,1
ld     (escFlag),a
ret
cont:  cp     32          ; czy byla spacja ?
jp     nc,enter
cp     0AH         ; czy byl <LF> ?
jp     z,terminate
cp     0DH         ; czy byl <CR> ?
jp     z,terminate
cp     0CH         ; czy byl <FF> ?
jp     z,terminate
ret
enter: ld     a,(counter)
ld     b,0
ld     c,a
ld     hl,chars
add    hl,bc
ld     (hl),e
inc    a
ld     (counter),a
; test counter
cp     120
ret    nz
ld     e,0AH    ; e = (LF)
terminate:
ld     a,(counter)
ld     hl,pdata
cp     0
jp     z,term2
push   de
ld     bc,expand
call   xbios
dw     screen
translate:                ; tlumaczenie
ld     de,sdata    ; skad
ld     hl,pdata    ; dokad
ld     b,a
cloop: push   bc
ld     b,8
oloop: push   de
push   hl
ld     a,(de)
ld     b,8
iloop: rlc    a
rl     (hl)
inc    hl
djnz   iloop
pop    hl
inc    de
pop    bc
djnz   oloop
ld     bc,8
add    hl,bc
pop    bc
djnz   cloop
; end of translate
pop    de
term2: ld     (hl),e      ; e=terminator
ld     a,(counter)
ld     l,a
ld     h,0
add    hl,hl
add    hl,hl
add    hl,hl       ; * 8
ld     (TRNL),hl
ld     bc,5
add    hl,bc
ld     (TOTAL),hl
ld     de,ccb      ; wywolanie
ld     c,70H       ; funkcji
call   bdos        ; BDOS 70H
xor    a
ld     (escflag),a
ld     (counter),a ; zerowanie licznika
ret
expand: ld     hl,chars
ld     de,sdata
ld     b,a
chloop: push   bc
push   hl
push   de
ld     e,(hl)
ld     d,0         ; de = charNo
ld     hl,RSG
ld     b,8
r:     add    hl,de
djnz   r
pop    de
ld     bc,8
ldir
pop    hl
inc    hl
pop    bc
djnz   chloop
ret
; ***** data
chars:  ds     120     ; bufor znakowy
ccb:    dw     buf     ; tablica CCB dla BDOS 70H
total:  dw     0       ; calkowita dlugosc ciagu
buf:    db     27,'L'  ; tryb graficzny
TRNL:   dw     0       ; liczba bajtow w trybie gr.
pdata:  ds     8       ; poczatek bufora drukarki
sdata:  ds     960     ; bajty znakow wg gen. ekr.
escflag:db     0       ; znacznik <ESC>
counter:db     0       ; licznik znakow
end
; ***********************************************
```

LISTING 2 Plik BDOS5. INC

```
(***********************************************)
(*                                             *)
(*      Plik BDOS5.INC  (C) JM 1988            *)
(*                                             *)
(*  Uzywana przez kompilator Turbo Pascala funkcja*)
(* LstOut korzysta z funkcji modulu BIOS przy   *)
(* drukowaniu znaku instrukcja write (lst,znak). *)
(* Zdefiniowana w pliku procedura OwnLST umozliwia*)
(* wykorzystanie funkcji modulu BDOS do tego celu.*)
(* Dolaczenie RSX'a generowanego przez plik     *)
(* BDOS5.MAC pozwala drukowac w trybie graficznym *)
(* wszystkie znaki generatora ekranowego.       *)
(*                                             *)
(***********************************************)
const
    DefaultLST ; integer = 815; (* standartowy  *)
                  (* adres procedury LstOut *)
procedure InitLST;
(***********************************************)
(* procedura ustala adres standartowej procedury  *)
(* drukowania znaku.                           *)
(***********************************************)
begin
    defaultLST := LstOutPtr;
end; (* InitLST *)

procedure Bdos5 (c : char);
(***********************************************)
(* procedura definiuje drukowanie znaku na drukar-*)
(* ce przez piata funkcje modulu BDOS.         *)
(***********************************************)
begin
    BDOS (5,ord(c));
end; (* bdos 5 call *)

procedure OwnLST (Switch : boolean);
(***********************************************)
(* Procedura wybiera miedzy standartowa obsluga   *)
(* drukowania znaku (modul BIOS) i obsluga poprzez*)
(* modul BDOS.                                 *)
(***********************************************)
begin
    Case Switch
    of false : LstOutPtr := DefaultLST;
       true  : LstOutPtr := Addr (Bdos5);
    end;
end; (* OWN LST *)

(***********************************************)
```

MUSIC by MAC

JAK TO ROBIĄ INNI ?

Apple'owskie komputery z serii Macintosh zaprojektowane były z myślą o użytkowniku z różnym stopniem zaawansowania w technice komputerowej i różnymi zainteresowaniami. Ci, którzy chcieli liczyć dostali szybki i sprawny komputer z bogatym oprogramowaniem językowym, graficy i projektanci — mnóstwo pamięci i trójwymiarową grafikę, humaniści — świetne edytory tekstów, najmłodsi — moc mądrych gier i programów edukacyjnych, wreszcie Ci zainteresowani muzyką dostali swoją prywatną orkiestrę.

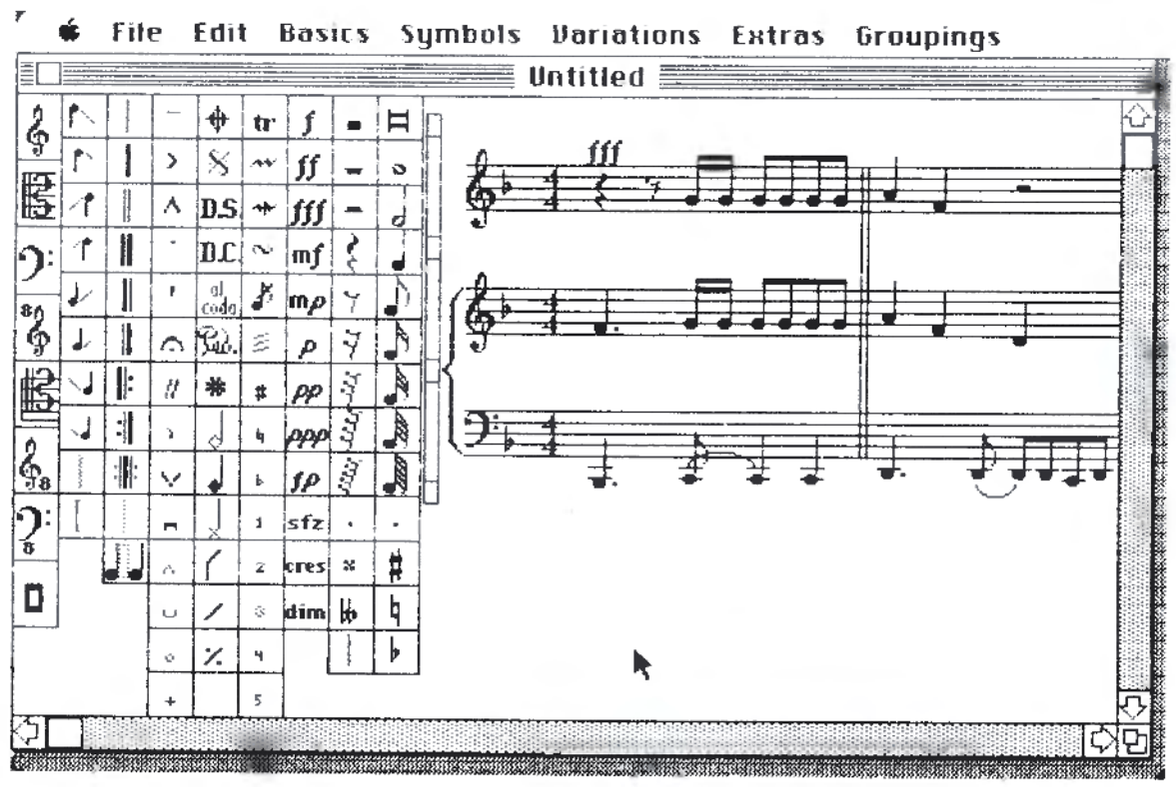
Tworzenie muzyki jest tak proste, jak obsługa tego komputra. Wszystkie właściwe zestawy programów muzycznych na Mac'a są podobnie zaprojektowane. Składają się prawie zawsze z Edytora, Odtwarzacza i programu pomocniczego do tworzenia instrumentów. Pominę już tu możliwości podłączenia przez interface MIDI syntezatora, choć i tę możliwość oferuje większość programów.

Edycja, czyli wpisywanie nut, odbywa się tak, jak na papierze. Długość utworu ograniczona jest jedynie przez pamięć komputera. Do dyspozycji mamy obecnie osiem kanałów (na rysunkach sześć, gdyż mamy starsze wersje programów), choć sam komputer wyposażony jest tylko w cztery: sprytnie je rozbito i ludzkie ucho nie jest w stanie wychwycić różnic. W najbardziej profesjonalnym programie muzycznym możliwe są wszystkie tonacje, wszystkie wartości nutowe, wszystkie... Ten program to Professional Composer, którego używa już w Polsce jeden z chórów do zapisu głosów. Każdy kanał może być odgrywany przez inny instrument. Instrumenty możemy robić nawet sami. Ciekawie jest eksperymentować z różnymi kształtami krzywej dźwięku oraz zmieniając zawartość wyższych harmonicznych wpływać na barwę głosu. Jednak tak tworzone instrumenty są typowo elektroniczne i nawet najlepiej podrobione skrzypce nie brzmią jak prawdziwe. Jest na to jednak rada, można mianowicie wgrywać do komputera przez mikrofon i specjalny interface dźwięki prawdziwych instrumentów, które komputer zapisuje sobie cyfrowo. I już brzmi jak powinien. Jednak do tego potrzeba kosztownego sprzętu (z jednym z programów dostarczany jest na szczęście fabrycznie plik prawie trzystu instrumentów). Inny program dostarcza także zbiór motywów muzycznych oraz akompaniamentu, do wykorzystania w całości lub częściowo we własnych utworach. Przygotowaną partyturę, możemy oczywiście pięknie wydrukować przy pomocy drukarki mozaikowej lub laserowej.

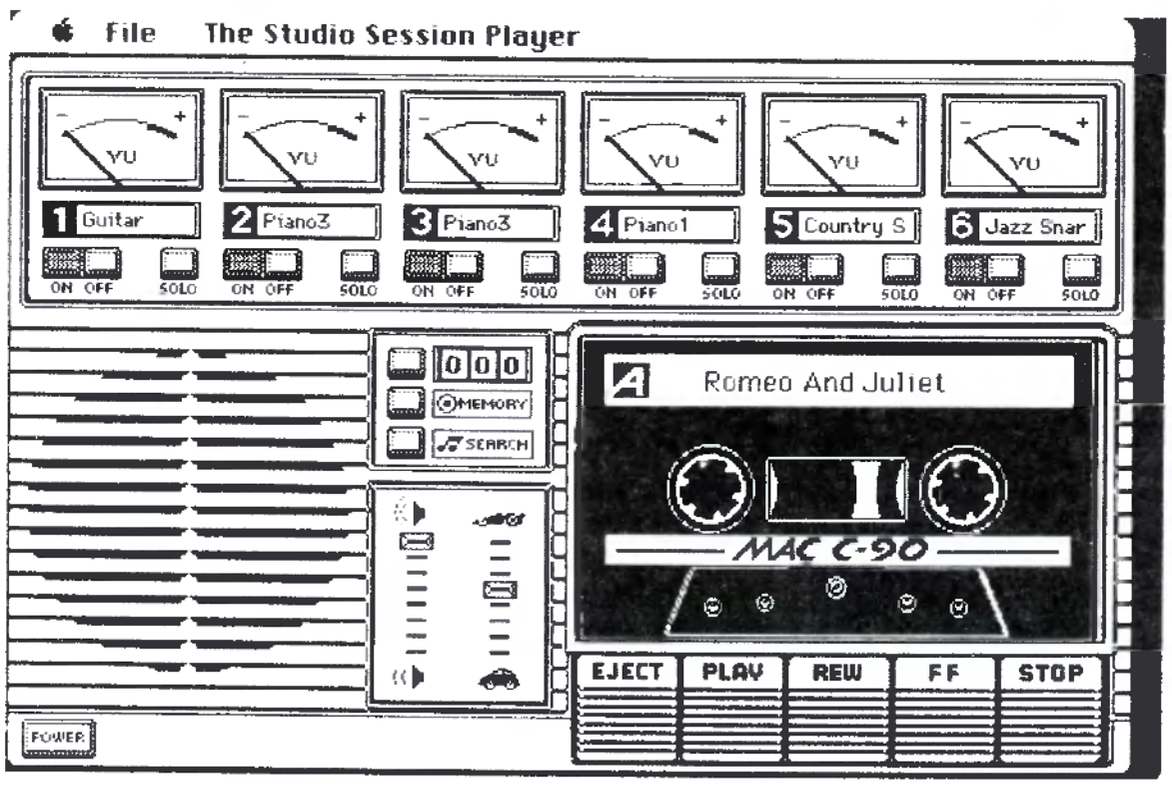
Teraz możemy już posłuchać gotowego dzieła bądź bezpośrednio z Edytora, bądź przez często ciekawie zrobiony Odtwarzacz, który dodaje połysku i sprawia, że cała procedura tworzenia muzyki wygląda profesjonalnie. Jeden z programów pozwala też skomponowaną w innym programie muzykę odtworzyć w ciekawej scenerii. Na przykład muzykę country w takiej jak na rysunku (postacie są animowane i naprawdę grają swoje partie). Niestety, dopiero nowsze Macintosh'e mają wyjście stereofoniczne; do starszych, by uzyskać ten efekt, potrzeba MIDI. Brak stereofonii nieco pogarsza końcowy efekt. Ale i tak muzyka odtworzona przez wzmacniacz i głośniki robi wrażenie.

Wojtek Rzążewski

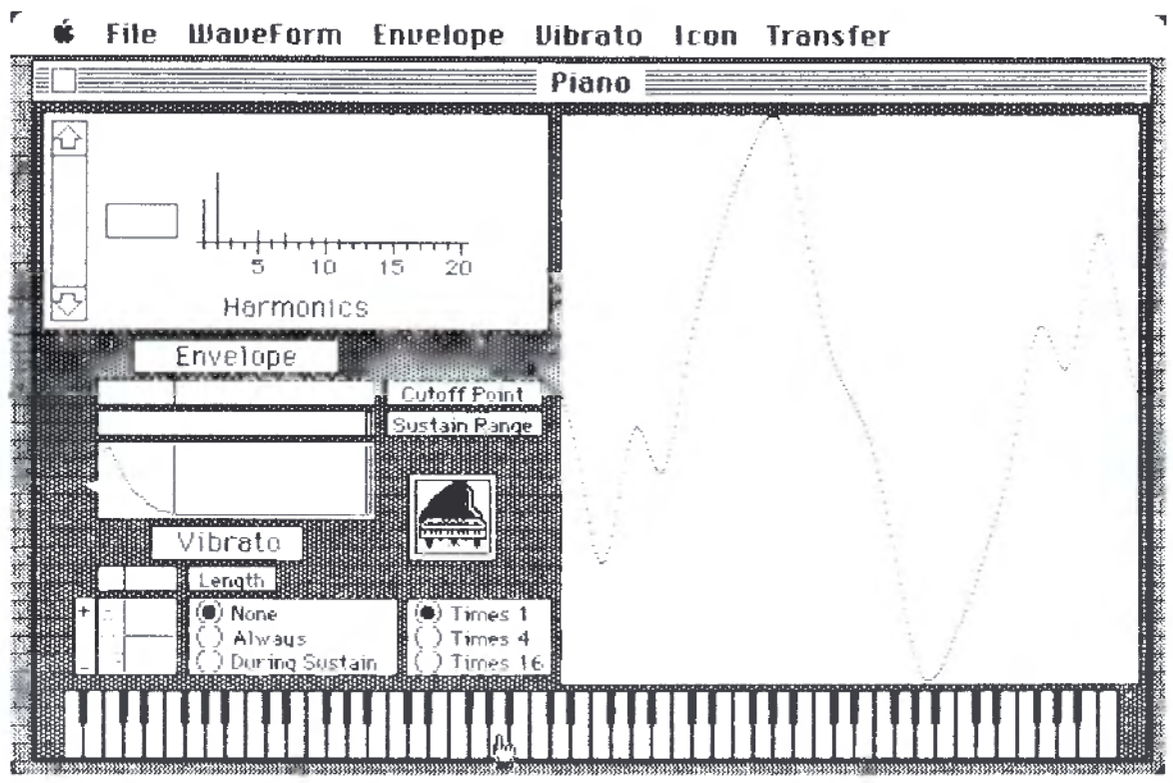
W artykule użyłem własności różnych programów: Studio Session, Jam Session, Professional Composer, Music Construction Set, z których tylko dwa pierwsze są zgodne. Mimo to wszystkie mają podobne cechy tyle, że mniej rozbudowane.



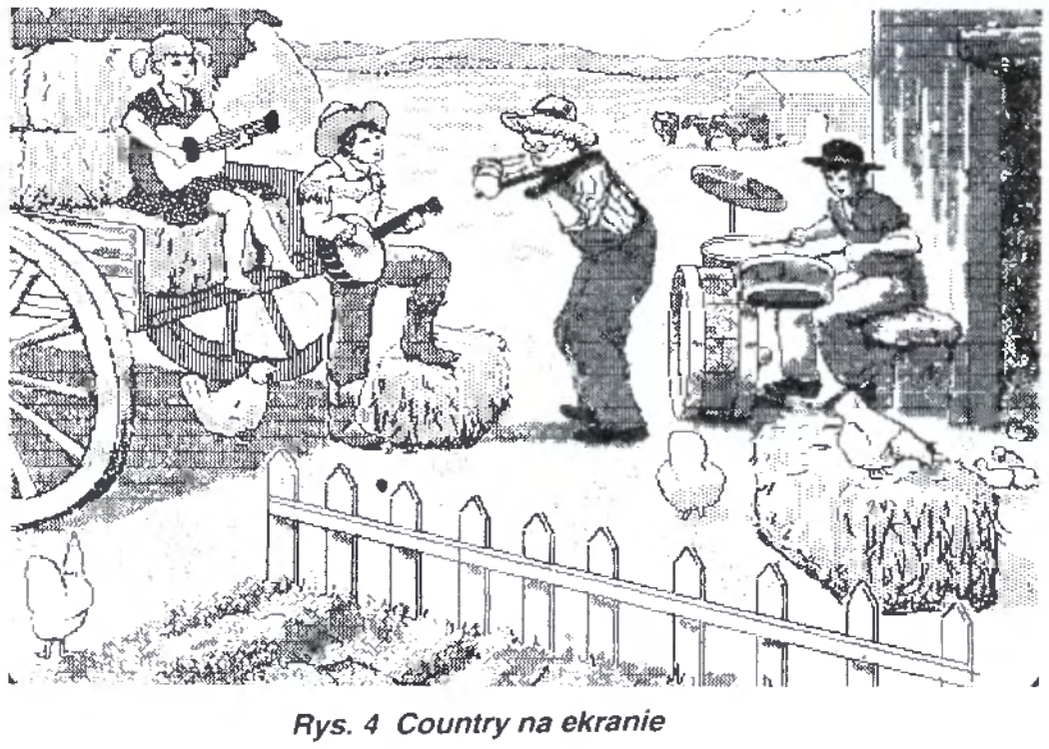
Rys. 1 Edytor



Rys. 2 Odtwarzacz



Rys. 3 Edytor instrumentów



Rys. 4 Country na ekranie

Nie mamy Macintoshów, w Polsce też nie jest ich wiele. Są za to śliczne i miłe, i życzymy każdemu posiadania Mac-a.

TEST !

JOYSTICK NA KAŻDĄ OKAZJĘ

Dzięki uprzejmości p. Marka Tokarskiego z firmy TAL otrzymaliśmy do przetestowania garnitur najnowszych joysticków, które właśnie zdobywają nasz rynek.

W odróżnieniu od spotykanych w naszych sklepach i na giełdach joysticków, te modele odznaczają się świeżością i nowatorstwem konstrukcji. Wszystkie produkowane są przez koncern **Quickjoy,** a montowane w niedalekich Chinach.

Po wstępnym zapoznaniu się z całą dwunastką nie byliśmy jeszcze w stanie wyróżnić faworyta. Od czasów legendarnego QuickShota firma Quickjoy poczyniła niebywałe postępy technologiczne. Dawno minęły już czasy pękających przyssawek blaszek i blokujących się FIRE-ów. Także wygląd zewnętrzny znacznie się poprawił; joysticki złapały linię i manualną funkcjonalność.

Oprócz typowych zabijaków powstały modele do zadań specjalnych. Mamy więc wspaniały wolant, przeznaczony specjalnie do symulatorów lotniczych, wielki, ale efektowny. Absolutnym hitem jest **InfraRed** (joystick na podczerwień). Pozbawiony on jest kabla można więc posługiwać się nim w dowolnie wielkim pokoju. Być może przyda się on również pedagogom do pokazów komputerowych na odległość.

Niekonwencjonalnym rozwiązaniem jest **Superboard V** — wyposażony w 8 FIRE-ów, cztery przełączniki i wielofunkcyjny zegar. Pomyślano specjalnie o leworęcznych, różnicując działanie przycisków FIRE: lewo-prawo, góra-dół.

Zawstydził nas za to **TopStar** — wszystko mu widać! Jest bowiem przezroczysty i w trakcie pracy można obserwować, co dzieje się w środku. Konstrukcja wewnętrzna jest absolutnie nietypowa; zamontowane są tam cztery amortyzatory kompensujące siłę uderzenia, specjalny metalowy rdzeń usztywniający całość drążka oraz wielkie mikroprzełączniki zamontowane „na wsuwki", bez lutowania. **TopStar** odznaczał się ogromną wytrzymałością; można nim popełnić prawdziwe morderstwo. Drążek jest nie do ukręcenia i nie do złamania.

Dla wielbicieli strzelanin i symulatorów przeznaczony jest **MegaBoard** — wyposażony w ogromne przyciski FIRE oraz ciekłokrystaliczne ekrany odmierzające czas, pokazujące godzinę i datę.

Superciekawostką jest przełącznik **Slow Motion,** spowalniający reakcje układów elektronicznych joysticka na poruszenia drążkiem. Pozwala to np. na prowadzenie precyzyjnych walk powietrznych. **Slow Motion** zamontowany jest w **MegaBoard** i **TopStar.**

Producenci nie zapomnieli również o dzieciach i posiadaczach małych dłoni. **Quickjoy Junior** i **Juniorstick** mają niewielkie uchwyty, wręcz niewygodne dla redakcyjnych oprawców.

Pozostałe joysticki, nie tak specjalistyczne i ekstrawaganckie, prezentowały również wysoki poziom. Istnieją pewne różnice w ergonomii, jednak trudno dostrzegalne. Ułożenie uchwytu, przycisków FIRE, kształt podstawy — te elementy dają się poznać naprawdę dopiero po dłuższej znajomości.

Testowane joysticki przeszły u nas prawdziwą drogę krzyżową. Męczyliśmy je równo trzy tygodnie, zarówno na sucho, jak i w akcji. W użyciu były programy: PrintFox (program graficzny, Commodore), WetBeaver (superdynamiczny tenis, Amiga), F-29 Retaliator (symulator lotu, Amiga) i Carrier Command (gra strategiczna, lecz użyta tylko do sprawdzenia precyzji sterowania strzałką kursora, Amiga).

Oprócz tego badaliśmy:

— ergonomię,

— sprawność i regularność działania Auto-Fire,
— „równouprawnienie" kierunków działania,
— luzy, elastyczność i samopowracalność drążka do położenia początkowego,
— precyzję działania,
— dostępność i opór przycisków FIRE i przełączników specjalnych,
— siłę przyssania do stołu, oraz gładkich powierzchni,
— tendencję do przesuwania się po blacie.

Pod uwagę braliśmy też długość i elastyczność kabli (możliwe oddalenie od ekranu i bezpieczeństwo spadku ze stołu) oraz kształt i łatwość operowania wtyczkami.

Zaglądaliśmy również do środka tym bardziej, że dwa joysticki w trakcie prób poległy i chcieliśmy ustalić przyczynę zgonu (błąd konstrukcji, staranność montażu itp.). Część konstrukcji była szczególnie nieprzyjazna dla majsterkowiczów ze względu na wielość śrub, kabelków i zatrzasków.

W oczy (a raczej uszy) rzuciła nam się różna hałaśliwość, tzn. odgłosy wydawane podczas zaciętego używania.

Prawie wszystkie joysticki zostały dodatkowo wyposażone w przełącznik w tryb pracy CPC 464, jak wiadomo odbiegającego od ogólnie przyjętego standardu.

Do zapoznania się z dokładnymi wynikami testów dla poszczególnych modeli zapraszamy do lektury trzeciego numeru pisma „Top Secret", zapowiadanego na styczeń 1991 roku.

Waldemar Nowak
Marcin Przasnyski

W trakcie testów otrzymaliśmy od p. Trojaka z zaprzyjaźnionej firmy MATT specjalne, blisko dwumetrowe przedłużacze kabli joysticków. Są one wyposażone w normalną wtyczkę z jednej strony i jej dokładne przeciwieństwo z drugiej. Pozwala to na znaczne oddalenie się od monitora, ułatwia manipulowanie joystickiem i oszczędza gniazdo komputera. W naszym przypadku przedłużacze uchroniły nas od ciągłego gmerania wtyczkami z tyłu komputera. Małe, a cieszy! Gratulujemy pomysłu.

Potęga Action Replay

Po sukcesie FINAL'a III, na polskim rynku pojawił się nowy moduł o nazwie ACTION REPLAY. Jest to najbardziej wszechstronny jak dotąd cartridge przeznaczony do współpracy z C64, a także najdroższy (300—400 tys. na giełdzie). W porównaniu do FINALA II i III ma on znacznie poszerzoną paletę dostępnych opcji. Poniżej przedstawiamy skrótowy opis możliwość ACTION REPLAY'a:

KLAWISZE FUNKCYJNE
zarezerwowane podobnie jak w FINAL III.

MAGNETOFON:
Normalne TURBO (np. ABC), oraz specjalny system przyspieszający 5—6 razy szybciej niż tradycyjne turbo. Tu też kryje się zasadnicza różnica w stosunku do zachodniego oryginału bowiem polska wersja wyposażona jest też w standardowe TURBO „bez pasków" kompatybilne z TURBO ROM.

DYSK:
Przyspieszenie 20-krotne odczytu i zapisu, w trybie WARP 25-krotne.

ASSEMBLER:
Monitor pamięci oraz monitor dyskowy.

FREEZE:
Zgrywanie zawartości pamięci na dysk lub taśmę w jednej całości (file) bez loadera
Zapamiętywanie screenów (ze spritami) i ich zapis w jednym z sześciu formatów (np. Koala Painter)
Drukowanie ekranu.
Biblioteka sprite'ów.
Wyłączanie kolizji sprite'ów.

BASIC TOOLKIT:
Rozszerzenie o bardzo użyteczne komendy: OLD, DELETE, LINESAVE, MERGE, APPEND, AUTOBOOT, PLIST, SLIST, ON/OFF, COPY, BACKUP

INNE:
Podręczne bardzo szybkie kopiery (file'owy i całodyskowy), zgrywanie obrazów wysokiej rozdzielczości, przenoszenie programów nagranych w systemie NOVALOAD na dysk.

UWAGA!
Obecnie na polskim rynku znajdują się dwie wersje ACTION'a: niemiecka i angielska. Warto pamiętać, że rozkazy w wersji niemiecko językowej są odpowiednikami angielskich:
AUS = OFF
AN = ON
znak funta = DSAVE
/ = DLOAD
"ucho słonia" = DOS
Na koniec krótkie podsumowanie:

Zalety:
1. łatwa obsługa
2. szybkość
3. niekolidowanie z programem
4. najszybsze kopiery
5. wiele rozszerzeń BASIC'a
6. powiększony monitor assemblera
7. i wiele innych

Wady:
1. wysoka cena
2. turbo do magnetofonu "bez pasków"
3. brak udogodnień takich jak "okienka" w FINAL III

The Naturat

Print AT(X,Y)

Basic V2.0 w C64 przyprawia często początkujących programistów o mdłości. Ot chociażby umieścić jakiś wyraz w określonej części ekranu (nie posługując się "spacjami"). Myślę że przynajmniej połowa początkujących miałaby z tym trudności. A i bardziej zaawansowanym nie uśmiecha się perspektywa wpisywania ciągów „negatywów" w cudzysłowie. Myślę, że poważnie ułatwi wam życie ten oto programik. Zajmuje ona zaledwie 151 bajtów i rozszerza basic Commodore'a o komendę PRINT AT. Służy ona właśnie do wypisywania różnych rzeczy w dowolnym miejscu ekranu. Program wywoływany jest po RESET rozkazem SYS 40820.
Nagrodą za pracowite wklepanie tego listingu będzie poważna wygoda w pracy.
Przyjemnej roboty!

The Naturat

```
100 REM ************************************
110 REM **                                  **
120 REM **        PRINT AT[X,Y]             **
130 REM **                                  **
140 REM **    WRITTEN BY T.BISSHOFF         **
170 REM **                                  **
180 REM **                                  **
181 REM **    POPRAWIŁ : THE NATURAT        **
190 REM ************************************
210 :
220 FOR A=0 TO 132:READ B:POKE 40820+A,B
230 S=S+B:C=C+1:IFC<7THEN NEXT:C=-1
240 READ P:IF P<>S THEN PRINT "BLAD W LINI
DATA";Z+290:STOP
250 Z=Z+10:IF C>0 THEN C=0:S=0:NEXT
260 :
270 POKE 55,112:POKE56,159 :SYS40820:END
280 :
290 DATA 169,127,141,8,3,169,159,776
300 DATA 141,9,3,96,32,115,0,396
310 DATA 201,153,240,6,32,121,0,753
320 DATA 76,231,167,165,122,141,60,962
330 DATA 3,165,123,141,61,3,169,665
340 DATA 0,170,168,32,115,0,217,702
350 DATA 241,159,208,8,200,192,3,1011
360 DATA 240,19,76,154,159,173,60,881
370 DATA 3,133,122,173,61,3,133,628
380 DATA 123,32,121,0,76,231,167,750
390 DATA 32,115,0,32,158,183,224,744
400 DATA 40,144,5,162,14,76,58,499
410 DATA 164,134,211,32,253,174,32,1000
420 DATA 158,183,224,25,144,3,76,813
430 DATA 196,159,134,214,32,108,229,1072
440 DATA 32,121,0,201,93,240,5,692
450 DATA 162,11,76,58,164,169,153,793
460 DATA 32,124,0,76,231,167,65,695
470 DATA 84,91,0,255,0,0,255,685
```

SŁÓW KILKA O STACJACH DYSKÓW

Gdy masz już dość własnego magnetofonu do Commodore, programy nagrane na różnych poziomach ustawienia głowicy powodują lęki nocne. Pisk ładowania gier przyprawia Twoich rodziców o wściekliznę, a brat lub siostra kasują Ci programy nagrywając zamiast nich odgłosy rybek z akwarium. Masz już wszystkiego powyżej uszu. Wtedy właśnie oświadczasz wyraźnie przerażonym rodzicom, że chcesz mieć stację dysków.

I wreszcie nadchodzi ten dzień, w którym to właśnie uzyskujesz pieniądze w ilości wystarczającej na zakup stacji. Co dalej?

Jeżeli kupujesz stację dysków, obojętnie czy to w BALTONIE czy na giełdzie, musisz wykonać minimum 5 czynności:

1. Włączyć stację i posłuchać pracy silnika. Szum powinien być cichy i równomierny.

2. Włożyć nową, czystą dyskietkę dobrej klasy i sformatować ją. Po ukończeniu tej operacji, oglądany pod światło nośnik dyskietki; nie powinien posiadać żadnych rys.

3. Nagrać na dyskietkę kilka programów i parokrotnie je załadować.

4. Załadować program nagrany na innej stacji.

5. Sprawdzić czy stacja nie była rozkręcana. Jeśli była, warto sprawdzić jej stan wewnątrz.

Gdy już szczęśliwie przebrniesz przez te testy, możesz być dumny ze swojej stacji dysków. Należy jednak pamiętać, że jest ona urządzeniem bardzo delikatnym i czułym na wstrząsy, a zatem polewanie herbatą, formatowanie kartki itd. nie wpływa dodatnio na jej pracę. Dyskietki należy wkładać spokojnie (bez dopychania widelcem), a klucz zamykać i otwierać bez nerwów.

Po tej porcji rad i przestróg kilka wiadomości o stacjach dysków dostępnych na polskim rynku:

1. VC 1541 (stary typ) — stacja duża, ciężka i pancerna. Jej główną wadą jest to, że posiada wbudowany zasilacz powodujący wzrost ciepła w urządzeniu. Na minus zaliczyć też trzeba tzw. klapkę. Dłuższy czas użytkowania powoduje „wyrobienie się" kieszeni, co może być przyczyną błędnego odczytu.

2. VC 1541 (nowszy typ) — podobnie jak wyżej, różni się jednak od poprzedniczki tym, że posiada inny napęd z kluczem. Zmodyfikowano w niej też nieco układy.

3. VC 1541 II — zmniejszone wymiary, lekka, zasilacz oddzielnie, dość słaba budowa. Wbudowany przełącznik numeru urządzenia.

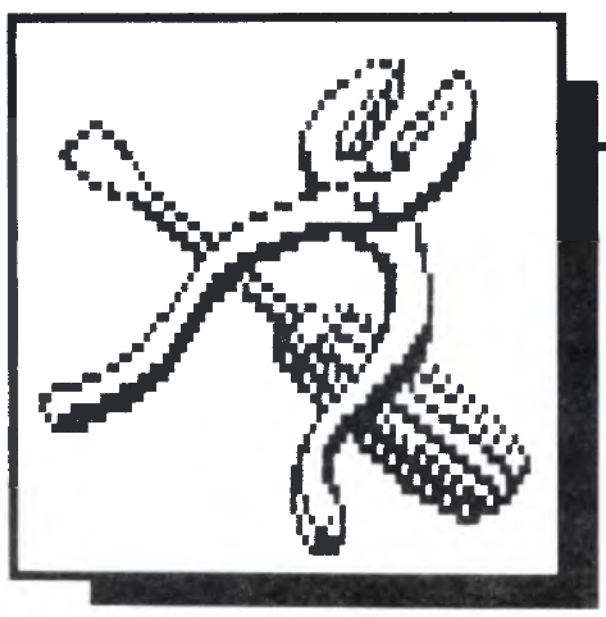
4. VC 1571 — stacja nadająca się głównie do Commodore 128. Używana z C64 nie pracuje z niektórymi programami. Dość mocna budowa, dwie głowice, przełącznik numerów urządzenia, zasilacz (niestety) w środku.

5. OCEANIC — stacja z budowy dość podobna do VC 1541 II lecz troszkę mocniejsza.

6. TURBO — najlepsza jak dotąd stacja dysków. Zasilacz na zewnątrz, bardzo solidna budowa, stacja troszkę szybsza od poprzedniczek.

Sądzę, że te kilka wiadomości pomogą Ci w zakupie i użytkowaniu stacji dysków. Wybór pozostawiam Tobie.

Piotr Liszewski



8255 — okno na świat

Jedną z niezbędnych części każdego systemu mikrokomputerowego są układy we-wy. Przez nie system komunikuje się z otoczeniem, np. wyświetla informacje na ekranie monitora, czyta klawiaturę, zapisuje i czyta dane z magnetofonu lub stacji dysków, wysyła dane do drukarki. W większości komputerów te możliwości zapewnia producent instalując odpowiednie interfejsy w obudowie komputera.

Niestety praktycznie nie dotyczy to ZX Spectrum. Dołączenie drukarki lub jakiegokolwiek urządzenia wyposażonego w złącze szeregowe lub równoległe wiąże się z zakupem intefejsu. A co zrobić, jeśli chce się przy pomocy komputera sterować różnymi urządzeniami (zapalać i gasić światło, włączać i wyłączać silniki, sterować reklamą świetlną, wykonać prosty system alarmowy z pamięcią, podłączyć niestandardową drukarkę lub ploter)? Trzeba posiadać uniwersalny układ we-wy i trzeba umieć go oprogramować.

Bardzo popularnym programowanym układem we-wy jest układ 8255 firmy Intel (polski odpowiednik — MCY 7855). Układ ten posiada 24 we-wy umożliwiające równoległe dwukierunkowe przesyłanie danych w trzech trybach (0, 1, 2). 24 we-wy są zgrupowane w trzech ośmiobitowych rejestrach zwanych portami i oznaczanych literami A, B i C. Wysyłając odpowiednie słowo sterujące do specjalnego rejestru sterującego RS możemy zaprogramować każdy port jako wejściowy lub wyjściowy. Dla uproszczenia zajmiemy się tylko pracą układu 8255 w trybie 0. Rys. 1 przedstawia interfejs zrealizowany w oparciu o układ 8255.

bit	port	wartość
D7	—	— "1"
D6	—	— "0" dla trybu 0
D5	—	— "0" dla trybu 0
D4	— PA7-PA0	— "0"-wy, "1"-we
D3	— PC7-PC4	— "0"-wy, "1"-we (grupa II)
D2	—	— "0" dla trybu 0
D1	— PB7-PB0	— "0"-wy, "1"-we
D0	— PC3-PC0	— "0"-wy, "1"-we (grupa I)

Tab.1 Budowa słowa sterującego wysyłanego do rejestru RS.

Noga	Znaczenie	Noga	Znaczenie
1	— PA3	21	— PB3
2	— PA2	22	— PB4
3	— PA1	23	— PB5
4	— PA0	24	— PB6
5	— RD	25	— PB7
6	— CS	26	— +zasilania
7	— masa	27	— D7
8	— A1	28	— D6
9	— A0	29	— D5
10	— PC7	30	— D4
11	— PC6	31	— D3
12	— PC5	32	— D2
13	— PC4	33	— D1
14	— PC0	34	— D0
15	— PC1	35	— Reset
16	— PC2	36	— WR
17	— PC3	37	— PA7
18	— PB0	38	— PA6
19	— PB1	39	— PA5
20	— PB2	40	— PA4

Tab.2. Opis wyprowadzeń układu 8255

Sposób tworzenia słowa sterującego.

Porty programuje się w dwóch grupach. Do pierwszej grupy należy port PA i część portu PC (PC4-PC7), do drugiej — port PB i część portu PC (PC-PC4). Korzystając z Tab. 1 można łatwo zaprogramować sposób pracy każdego portu. Załóżmy, że chcemy aby port PA pracował jako wejście, porty PB i PC jako wyjścia. Odpowiednie słowo będzie miało postać 10010000. Aby PA pracował jako wy, a PB i PC jako we należy wysłać do RS słowo 10001011 (sprawdź!).

Wiadomo już co należy wysłać, aby ustawić dowolnie porty. Ale gdzie należy wysłać słowo sterujące? Każdy port PA, PB, PC i rejestr sterujący ma swój własny adres. W naszym przypadku:

Port PA — #1F (31 D) (00011111)
Port PB — #5F (95 D) (01011111)
Port PC — #9F (159 D) (10011111)
RS — #DF (223 D) (11011111)

Dlaczego takie adresy? Spójrzmy na schemat interfejsu na rys. 1. Układ 8255 reaguje na sygnały sterujące z mikroprocesora tylko wtedy, gdy zostanie uaktywniony (wybrany) podaniem "0" logicznego na wejście **CS.** W naszym przypadku nastąpi to tylko wtedy, gdy jednocześnie sygnał **IORQ** i A5 ZX Spectrum będzie miał wartość logiczną "0". Wejścia adresowe A0 i A1 układu 8255 przy pomocy których wybiera się port A, B, C lub rejestr RS są połączone z liniami adresowymi A7 i A6 ZX Spectrum. Wybrano linie adresowe A5, A6 i A7 ponieważ A0 — A4 są zajęte przez system ZX Spectrum.

Teraz wystarczy napisać OUT 223, BIN 10001011 aby wysłać słowo sterujące do rejestru RS ustawiając port PA jako wy, a PB i PC jako we. Konfigurację portów można zmienić wysyłając inne słowo sterujące zestawione wg Tab. 1. Uwaga! Po wyzerowaniu układu 8255 (np. po włączeniu zasilania) wszystkie porty pracują jako wejścia!

Jeśli napiszemy teraz OUT 31,0 to wszystkie wyjścia portu PA będą miały niski poziom logiczny. Wysłanie OUT 31,255 (inaczej OUT 31,BIN 11111111) ustawi wszystkie wyjścia PA w wysoki poziom logiczny. Jeśli chcemy by np. bit 3 PA miał wysoki poziom logiczny to należy wysłać OUT 31,BIN 00001000.

Ciekawą właściwość posiada port PC. Można indywidualnie ustawiać każdy bit tego portu wysyłając odpowiednie słowo sterujące do rejestru RS (nie do portu PC!). Słowo to tworzymy przy pomocy Tab. 3. Uwaga!!!!! Wyjścia każdego portu można obciążyć tylko jednym standardowym wejściem TTL, to znaczy, że do każdego wyjścia 8255 może wpłynąć max. prąd 1.6 mA. Należy więc stosować wzmacniacze, z których najprostszy to tranzystor.

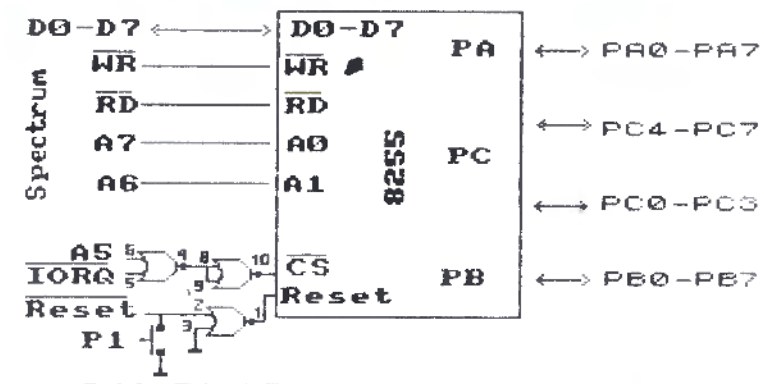
Rys. 2 przedstawia sposób dołączenia przełącznika do 8255. Ponieważ z poziomu BASIC-a bezpośrednio nie można testować poszczególnych bitów portu więc pozostałe wejścia tego portu też muszą być dołączone przez oporniki do masy jak na rysunku. Oczywiście można do nich dołączyć przełączniki. Można teraz przy pomocy instrukcji *IN adres portu* czytać stan przełączników.

Programując w asemblerze możemy łatwo testować dowolny bit portu.

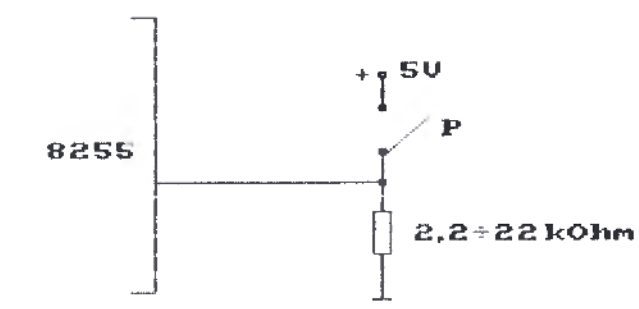
```
      IN   A,(31)
      BIT  4,A
      JP   NZ,TAK
NIE   ...
      ...
TAK   ...
      ...
```

W podanym przykładzie sprawdzany jest 4 bit portu PA i jeśli równy jest "1" to następuje skok do etykiety TAK. W przeciwnym wypadku wykonywany jest program od etykiety NIE. Rys. 3 pokazuje jak do 8255 można podłączyć diodę LED. Uwaga! Ponieważ stabilizator ZX Spectrum pracuje na granicy swoich możliwości więc wszystkie sterowane urządzenia powinny być zasilane z oddzielnego zasilacza. Do prób z diodami LED lub żarówką wystarczy nowa płaska bateria.

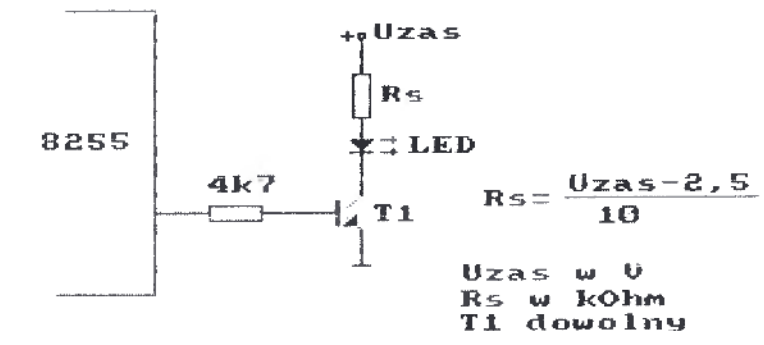
Rys. 4 pokazuje sposób podłączenia przekaźnika, np. popularnego MT 6. Dioda połączona równolegle z przekaźnikiem zabezpiecza tranzystor przed przepięciami występującymi podczas przełączania. Wysłanie OUT 31,BIN 00000001 (inaczej OUT 31,1) spowoduje zapalenie się diody LED lub włączenie przekaźnika. Wysłanie OUT 31,0 zgasi diodę LED (wyłączy przekaźnik).



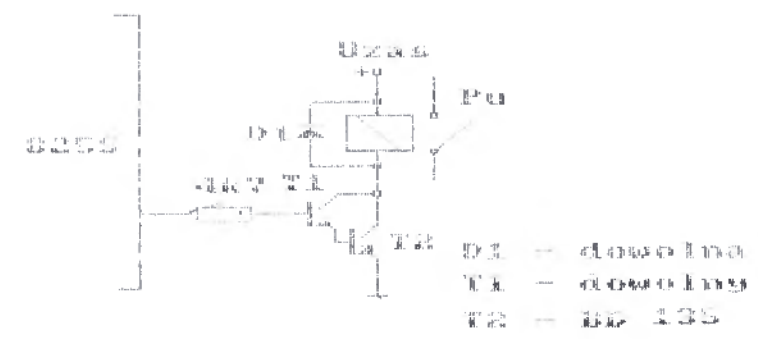
Rys.1 Schemat interfejsu.



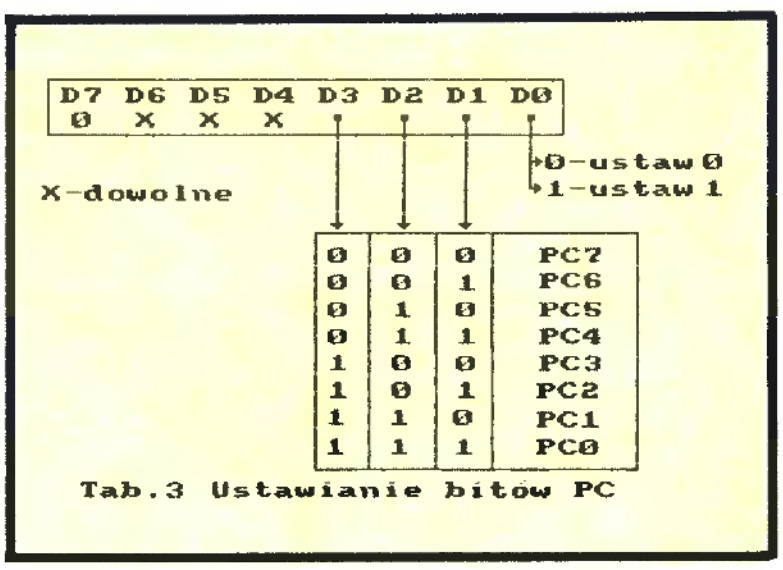
Rys.2 Podłączenie przełącznika.



Rys.3 Podłączenie LED



Rys.4 Podłączenie przekaźnika.



Tab.3 Ustawianie bitów PC

Przez opisany interfejs od ponad roku podłączam do mojego Spectrum różne urządzenia. Dla ciekawych podam, że np. dodając 2 oporniki i kilka przewodów możemy używać elektronicznej maszyny do pisania ERICA 3004 jako doskonałej drukarki rozetkowej, a prosty popularny timer 555 (ULY 7855) pozwala mierzyć rezystancję, pojemność kondensatorów, temperaturę, natężenie oświetlenia. Oczywiście z odpowiednim oprogramowaniem!

Grzegorz Bujanowski

Zainteresowanych montażem w/w układów prosimy o skontaktowanie się z autorem pod nr tel. 45-37-47 w Warszawie.

(red.)

Nie jest to wcale problem wydumany. Często zdarza się, że w opracowaniu chcemy umieścić fragment jakiegoś obrazu, zdjęcia lub rysunku. Nie każdy ma taki talent, aby skopiować obraz wielkiego Leonarda. Rozwiązaniem problemu jest maszyna o nazwie „skaner" — dla anglistów „scanner".

SKANER

czyli jak przenieść Giocondę do komputera

Jest to, mówiąc w największym skrócie, odwrotność drukarki (w sensie funkcjonalnym) — jego działanie polega na przenoszeniu gotowych obrazów z papieru do komputera, czyli przetworzeniu ich na zapis cyfrowy. Tak wprowadzony obraz może być obrabiany różnymi programami i powtórnie wyprowadzony na papier.

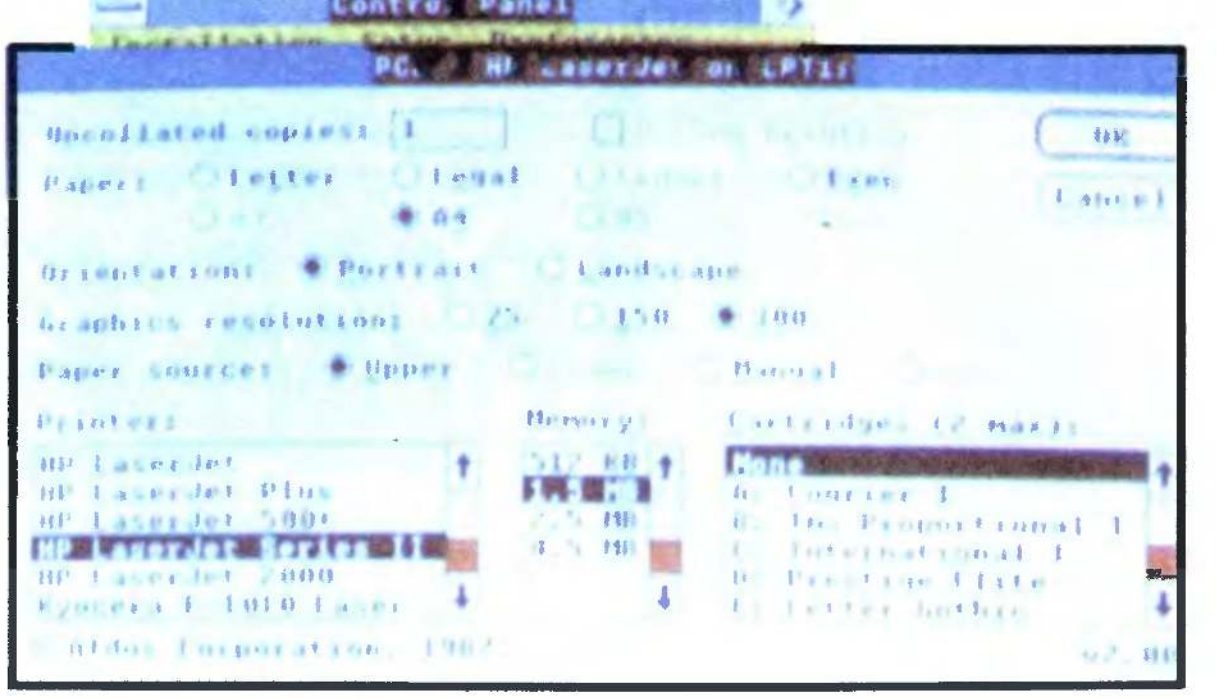
Z tej skrótowej charakterystyki widać od razu, dla kogo skaner będzie miał największe znaczenie — dla artystów zajmujących się grafiką komputerową, dla inżynierów wprowadzających gotowe rysunki techniczne w celu dalszej obróbki programami typu CAD (Computer Aided Design — projektowanie wspomagane komputerem) oraz dla wszystkich zajmujących się małą poligrafią (DTP — DeskTop Publishing).

Każda z tych grup zawodowych ma inne wymagania co do samych urządzeń skanujących oraz programów ich obsługi. Dla grafika najcenniejszy byłby skaner przekazujący obraz kolorowy, dla inżyniera — skaner o największej precyzji odczytu rysunku, dla poligrafa zaś skaner łączący te dwie cechy.

Z drugiej strony program obsługujący skaner powinien być maksymalnie uproszczony w wersji dla grafika i pracować przy minimalnej ingerencji użytkownika. Inżynier zaś powinien mieć możliwość szczegółowej kontroli tego, co dzieje się w programie. Poligraf zajmuje pozycję pośrednią, bliżej jednak potrzeb inżyniera.

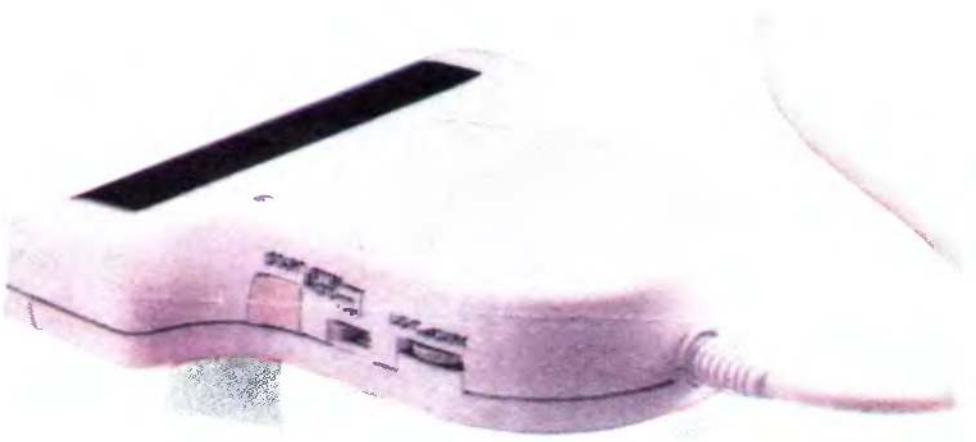
Nie będziemy się zajmować skanerami typowo inżynierskimi — są tak drogie i skomplikowane w użyciu, że ich popularność w Polsce jest bardzo niewielka. Rzadko spotykane są również skanery kolorowe, nie tylko w Polsce, lecz i na świecie. Ich cena jest nieproporcjonalnie wysoka w stosunku do możliwości. Jeżeli nie nastąpi przełom w technologii skanowania kolorowego, urządzenia te nie wyjdą poza specjalistyczne pracownie graficzno-edytorskie.

Zanim przejdziemy do bliższego opisu możliwości i obsługi skanerów, należałoby pokrótce przedstawić zasadę ich działania. Z mechanicznego punktu widzenia skaner przypomina kserokopiarkę. Posiada on głowicę skanującą z silnym źródłem światła (żarówką halogenową) i układem optycznym formującym strumień światła oraz elektro-optycznym, który przetwarza natężenie odbitej wiązki na strumień informacji o stopniu zaczernienia poszczególnych punktów skanowanego obrazu. W zależności od klasy skanera głowica może być poruszana przez precyzyjny układ mechaniczny (taki skaner nawet z wyglądu przypomina kserokopiarkę). W tańszych modelach głowica przesuwana jest nad skanowanymi dokumentami ręcznie przez operatora. Taki skaner (handy scanner) nie posiada części ruchomych i przypomina czytnik kodu kreskowego; jest on tani (100—200$), lecz nie jest to urządzenie wysokiej klasy.



Obraz uzyskany przez skaner musi być przesłany do komuptera, a następnie przetworzony do postaci umożliwiającej dalszą obróbkę. W prostszych modelach skaner podłączany jest bezpośrednio do łącza szeregowego (RS 232), skanery bardziej profesjonalne wymagają specjalnej karty umieszczanej wewnątrz komputera.

Konstrukcja mechaniczna skanera jest dość skomplikowana, często wielomodułowa. Aby nad tym sprzętem zapanować, program obsługujący musi być odpowiednio rozbudowany — musi sterować ruchem głowicy, nadzorować przesyłanie informacji i obsługiwać ewentualne błędy transmisji. Powinien także pozwalać na zmianę parametrów skanera w trakcie pracy, np. rozjaśnianie rysunków, zmianę kontrastu.



Przy takim natłoku zadań projektanci programu dostarczają często produkt skomplikowany w obsłudze, wymagający znajomości zasad skanowania, nieodporny na błędy użytkownika. Taki program stanowi zwykle zamkniętą całość i często uniemożliwia wymianę informacji (np. skanowych rysunków) z innymi programami, a na każdy błąd reaguje złowrogim „Error 1250, see manual pages 126, 178".

Na szczęście istnieje sposób, aby program obsługi skanera bez najmniejszych konfliktów włączyć w istniejące już na naszym komputerze oprogramowanie. Można będzie zachować standardowy sposób obsługi ekranu, stałe przypisanie klawiszy funkcyjnych etc. Nie będzie żadnych problemów w transmisji obrazów pomiędzy programem obsługi skanera a obecnymi w systemie programami graficznymi.

Wystarczy bowiem, by na naszym komputerze zainstalowany był MS Windows, a program obsługi skanera mógł być w tym środowisku uruchomiony i współpracował z nim. W tym momencie znikają wszystkie problemy związane np. z niekompatybilnością formatów plików graficznych. Rysunek wczytany przez skaner może być oprócz tego od razu poprawiony w jednocześnie aktywnym programie PaintBrush, następnie błyskawicznie wkomponowany w stronę tekstu np. pod PageMakerem i wydrukowany na drukarce laserowej.

Sposób pracy ze skanerem najlepiej pokazać opisując konkretny model. Niech będzie to Hewlett Packard Scan Jet, którym posługuję się od ponad roku. Jest to skaner tablicowy, może pracować z dokumentami o maksymalnym formacie LEGAL (nieco większy od A4) i być wyposażony w automatyczny podajnik dokumentów (stosowany głównie przy pracy z programami typu OCR), Optical Character Recognition — optyczne rozpoznawanie znaków i zamiana grafiki na kody ASCII.

Skaner sprzedawany jest z kartą interfejsu montowaną w komputerze. Karta ta zawiera pamięć ROM z procedurami obsługi skanera, RAM będącą buforem dla wprowadzanych obrazów oraz układy synchronizujące pracę karty z pracą komputera.

Montaż jest dość skomplikowany, gdyż nie wystarczy wsunięcie karty w wolny slot na płycie głównej. Potrzebne jest jeszcze ustawienie mikroprzełączników w ten sposób, by nie nastąpował konflikt karty skanera z innymi zainstalowanymi w komputerze kartami. Najczęstszym przypadkiem jest kolizja przestrzeni adresowanej karty skanera z obszarem wykorzystywanym przez kartę grafiki, jednak tylko w przypadku kart nietypowych.

Wstawienie karty i jej skonfigurowanie jest pierwszym krokiem do sukcesu. Następnym powinno być umieszczenie w pliku konfiguracyjnym CONFIG.SYS zlecenia udostępniającego kartę systemowi MS-DOS (DEVICE = SJDRIVER.SYS). Wtedy najczęściej okazuje się, że:

— plik konfiguracyjny przybrał takie rozmiary, że niektóre programy DOS-owskie nie dają się uruchomić,

— następują konflikty pomiędzy driverem skanera (SJDRIVER.SYS) a innymi zleceniami pliku CONFIG.SYS, przede wszystkim z programami obsługi pamięci stronicowanej (np. QEMM.SYS), programami typu CA-

CHE (np. C_DRIVE.SYS) lub z driverem niestandardowego dysku twardego,

— przestaje działać mysz (częste przy komputerach wyposażonych w więcej niż dwa złącza RS 232), pojawiają się zakłócenia we współpracy z twardym dyskiem.

W takich przypadkach (oprócz pierwszego) należy wyjąć kartę z komputera i manewrując mikroprzełącznikami doprowadzić do zgody. Udaje się to zawsze, nawet przy bardzo niestandardowych zestawach.

Możemy już przystąpić do zainstalowania oprogramowania obsługującego skaner. Dostarczany przez Hewlett Packard pakiet SCANGAL jest produktem najwyższej klasy i pozwala wykorzystać wszystkie możliwości skanera. Maksymalna rozdzielczość skanowania wynosi 600 dpi (*dots per inch* — punktów na cal), gdy tymczasem rozdzielczość większości drukarek laserowych nie przekracza 300 dpi. Pakiet SCANGAL umożliwia również skanowanie z różną rozdzielczością w pionie i poziomie. Jest to szczególnie cenne przy drukowaniu rysunków na drukarkach igłowych, które to w pionie posiadają zazwyczaj 72 dpi, w poziomie zaś zwykle 120, 144, 216 dpi. Rysunek powinien być skanowany z taką rozdzielczością, z jaką będzie potem drukowany.

Częstym błędem jest skanowanie każdego rysunku z maksymalną rozdzielczością (600 dpi) przy braku drukarki pracującej z tą rozdzielczością. Przede wszystkim jest to marnotrawstwo miejsca — rysunek A4 (8.5 x 11 cali = 93.5 cala kwadratowego) zajmuje 33660000 pixeli, co w najlepszym wypadku da plik prawie 4MB na dysku. Jakakolwiek praca z takim plikiem wymaga naprawdę szybkiego komputera.

A jeśli już dojdzie do drukowania takiego monstrum, to na standardowej drukarce laserowej wyjdzie on dwukrotnie powiększony lub też naturalnej wielkości, ale dużo gorszej jakości.

SCANGAL pozwala na skanowanie rysunków zarówno kreskowych (schemat, tekst drukowany) jak i rysunków o wielostopniowej skali barw. Skaner nie potrafi wprawdzie przenieść kolorów, ale potrafi każdemu pixelowi przypisać stopień szarości (w 16-stopniowej skali — 4 bity na pixel). Taki rysunek może być kolorowany jednym ze standardowych algorytmów.

Rysunki mogą być przyciemniane, rozjaśniane, wyostrzane itp. Można również żądać wygładzenia załamań linii, złagodzenia przejść między ciemnymi i jasnymi częściami. Może się to wydać dziwne, ale SCANGAL nie jest programem rysunkowym i nie umożliwia jakiejkolwiek ingerencji w treść rysunku, całą obróbkę cedując na profesjonalny program graficzny.

Przyjęło się kilka formatów zapisu rysunków, w zależności od popularnych programów edytorskich. Programy DTP i OCR stosują pliki .TIF (tagged image file format), PaintBrush —.PCX, system operacyjny GEM —.IMG, MS Paint — .MSP, wreszcie PostScript — .EPS.

SCANGAL umożliwia również przenoszenie całych obrazków w obrębie MS Windows poprzez specjalizowany program CLIPBOARD. Mając więc opisywany skaner oraz MS Windows, możemy nie tylko przenieść Monę Lisę do komputera, ale też bez specjalnego wysiłku domalować jej wąsy.

Z.K.

OPERA NA SZESNAŚCIE BITÓW, CZYLI JAK TO ZROBIŁ INTEL

Nie wszyscy zapewne wiedzą, że kolebką mikroprocesorów jest Intel.

W 1971 roku na zlecenie firmy Busicom pracownik Intela Ted Hoff stworzył niepozorny układ, którego podstawowym przeznaczeniem była praca w kalkulatorach biurowych. Hoff rozwiązał postawione zadanie w sposób niekonwencjonalny, obdarzając projektowany układ podstawowymi możliwościami programowania. Tak powstał czterobitowy pradziadek mikroprocesorów o nazwie 4004. Następnym był „duży" ośmiobitowy 8008. Oba nie wywołały jednak większego entuzjazmu użytkowników, gdyż nie były odpowiednio reklamowane. Dopiero 8080 stał się hitem. Wtedy ruszyła lawina.

Jak grzyby po deszczu zaczęły powstawać nowe firmy produkujące mikroprocesory. Zmieniały się technologie wytwarzania, a co za tym idzie — spadały ceny. Na przykład 8080, którego wartość na początku wynosiła 360 dolarów, dziś dostępny jest w cenie poniżej 2 dolarów. Najważniejszy był jednak skok możliwości technicznych. Powstawały coraz większe i szybsze pamięci, rosły również częstotliwości pracy procesorów. W pewnym momencie osiem bitów przestało wystarczać. Sięgnięto więc po następne osiem, i tak w 1978 roku powstał pierwszy procesor szesnastobitowy — 8088.

Zatrzymajmy się, by zajrzeć do środka tej czarnej kostki. Rzuci nam się w oczy podział na dwie części. Pierwsza z nich to EU (Execution Unit) — układ wykonawczy, czyli miejsce, w którym procesor wykonuje zapisane w pamięci rozkazy. Druga to BIU (Bus Interface Unit) — układ sterowania magistral, służący do komunikacji EU z otoczeniem. Oba połączone są ze sobą magistralą danych i nadzorującym je układem sterowania.

Co to takiego magistrala danych? Jest to swego rodzaju arteria przenosząca różne informacje wewnątrz i na zewnątrz procesora. Dzięki niej jednostka centralna, czyli procesor, może zażądać pewnej informacji z pamięci i otrzymać ją. Wyróżniamy dwa rodzaje magistral: danych i adresową. Pierwsza zajmuje się transferem informacji o programie i dla porgramu, druga informuje pamięć skąd i gdzie te dane mają być kierowane.

Układ wykonawczy EU składa się z dziewięciu rejestrów dostępnych dla użytkownika, rejestru buforowego, jednostki arytmetyczno-logicznej ALU (Arithmetic Logic Unit) oraz łączącej je ze sobą magistrali danych. Rejestr jest to miejsce, w którym procesor przechowuje dane niezbędne do wykonania rozkazu. Tu również często jest odkładany wynik operacji. Najważniejszym z nich jest akumulator, do którego odwołuje się najwięcej rozkazów. Drugim co do ważności jest rejestr flagowy, zawierający informację o stanie, w jakim znajduje się procesor, oraz o efektach wykonywanych zadań. Pozostałe rejestry zajmują się wspomaganiem pracy całości i obliczaniem adresu danych w pamięci.

ALU jest to miejsce, w którym procesor wykonuje wszystkie swoje obliczenia. Zadaniem ALU jest wykonywanie czterech podstawowych działań matematycznych (+, −, ∗, /) oraz czterech operacji logicznych (not, or, and, xor) na danych zawartych w rejestrach i w pamięci.

Osobliwością szesnastobitowej rodziny Intela jest BIU — element odpowiedzialny za pracę magistrali adresowej. 8088 może zaadresować 1MB pamięci, co oznacza, że w każdej chwili może odwołać się do jednej z 1048578 komórek. Jednak rejestry EU przy swoich szesnastu bitach mogą odwoływać się tylko do jednej z 65536 komórek. Jak w takim razie zaadresować resztę pamięci? Otóż w BIU znajduje się pięć specjalnych również szesnastobitowych rejestrów, zwanych rejestrami segmentowymi. Ich zawartość dodaje się w sposób przedstawiony na rys. 1 do zawartości odpowiedniego rejestru adresowego z EU i w ten sposób powstały wynik wysyła się do dwudziestobitowej magistrali adresowej.

```
s s s s s s s s s s s s s s s s 0 0 0 0
0 0 0 0 r r r r r r r r r r r r r r r r +
------------------------------------------
m m m m m m m m m m m m m m m m m m m m

Rys. 1 s - bit rejestru segmentowego (BIU)
       r - bit rejestru adresowego (EU)
       m - bit magistrali adresowej
```

Po tym zabiegu otrzymujemy pełen dwudziestobitowy adres. Metoda ta pozwala na bezkolizyjne współdziałanie w pamięci kilku programów o wielkości nie przekraczającej 64KB (65536 bajtów). Umożliwia ona również swobodne przesuwanie takich programów wewnątrz pamięci.

Ostatnim elementem BIU jest tak zwana kolejka rozkazów. Kiedy EU jest zajęty wykonywaniem swoich zadań i nie żąda od BIU dostępu do pamięci, układ sterowania magistral pobiera zawartość kolejnych komórek pamięci i umieszcza je w kolejce. Jeśli teraz EU zażąda następnej informacji z pamięci, BIU przekazuje ją wprost z kolejki, co znacznie przyśpiesza pracę procesora.

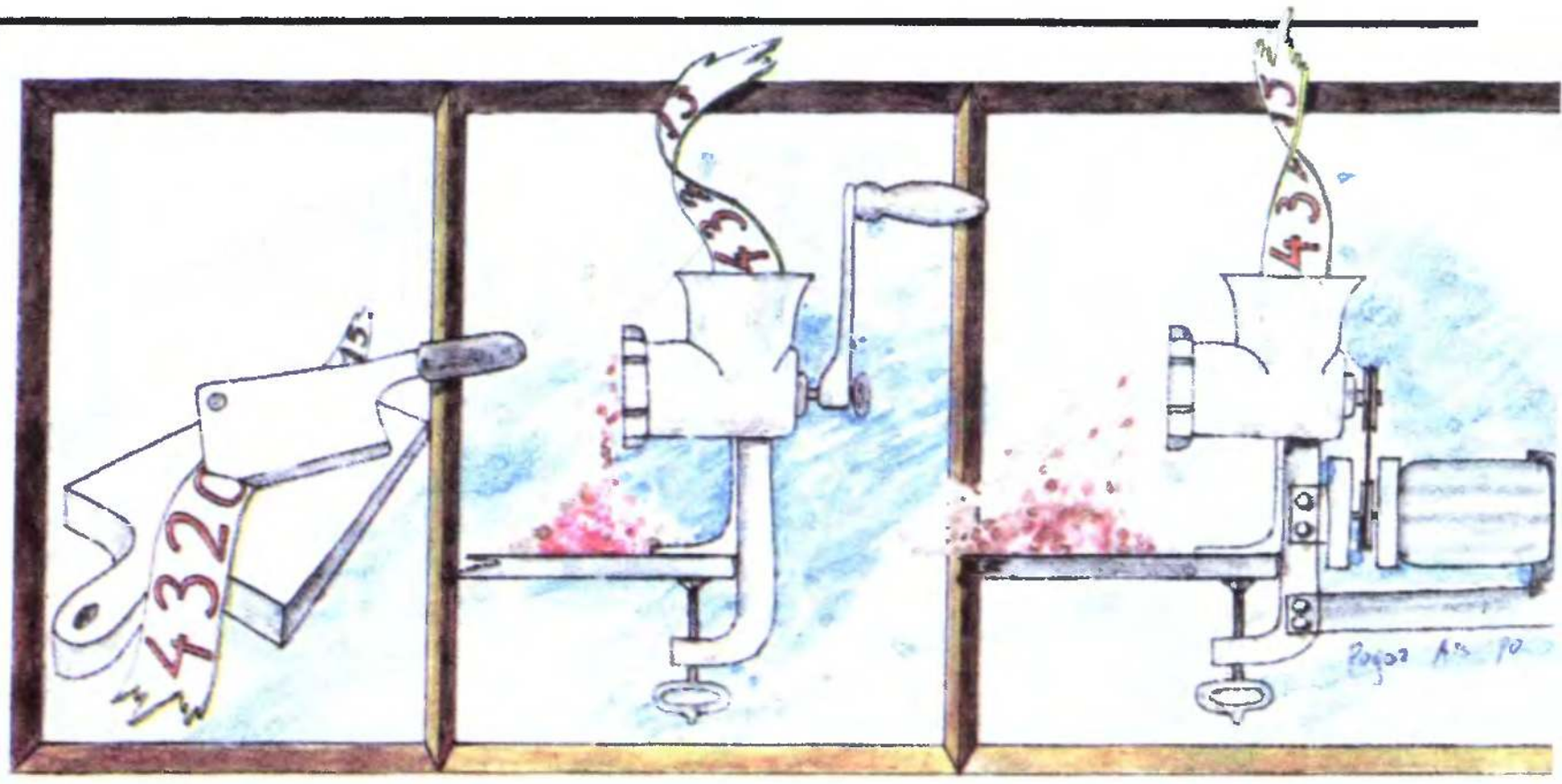
Całość pracy EU i BIU jest nadzorowana przez układ sterujący, który interpretuje rozkazy i na ich podstawie zarządza działaniem podległych mu elementów. Tu również docierają sygnały specjalne RESET i żądania przerwań. RESET powoduje natychmiastowe zatrzymanie procesora, skasowanie rejestrów i wprowadzenie go w stan początkowy. Przerwanie natomiast jest to wywołanie podprogramu pod wpływem jakiegoś zewnętrznego sygnału, na przykład spowodowanego naciśnięciem klawisza. Gdy procesor otrzyma taki sygnał, przechodzi do wykonywania innego programu, po którego zakończeniu wraca z powrotem. Bezkolizyjne wykonywanie programu głównego i programu przerwania wymaga oczywiście zachowania i odtworzenia zawartości rejestrów.

8088 posiada jeszcze oddzielną przestrzeń adresową, której komórki są rejestrami urządzeń wejścia-wyjścia, czyli drukarki, stacji dysków, joysticka, myszy i innych. Tędy właśnie procesor przesyła informację dla drukarki, żeby wydrukowała jakiś tekst, stąd otrzymuje również zapis programu z dyskietki, który umieszcza później w pamięci.

Czas najwyższy powrócić do naszej opowieści. Następnym procesem po 8088 był 8086. Pierwszy wyposażono w wąską ośmiobitową szynę danych, gdyż urządzenia towarzyszące produkowane w tym czasie przystosowane były do pracy z ośmiobitowymi procesorami. Tego rodzaju rozwiązanie miało zmniejszyć koszty w przypadku porażki nowego wyrobu. Tak się jednak nie stało i wkrótce powstał 3086 posiadający pełną szesnastobitową szynę danych. Później przyszedł czas na 80186 i 80188, w których oprócz dotychczasowych elementów umieszczono kilka dodatkowych, dotychczas tkwiących na zewnątrz — zegar, sterownik przerwań i kilka innych, a także nieco rozszerzono listę rozkazów. Następny procesor to 80286, w którym oprócz poszerzenia listy rozkazów dodano nowy tryb współpracy z pamięcią, umożliwiający zaadresowanie do 1 gigabajta pamięci. Ponadto procesor ten mógł pracować w systemach wielozadaniowych i wielodostępnych. Oznacza to, że potrafi on obsługiwać kilku użytkowników i kilka programów naraz.

To już właściwie wszystkie szesnastobitowe dzieci Intela. Można by jeszcze wspomnieć o koprocesorach rodziny 80i87, przeznaczonych do współpracy z omówionymi procesorami (*patrz artykuł obok — MB*). Po nich przyszła era trzydziestodwubitowców, takich jak 80386 i 80486, ale to już zupełnie inna opera.

Maciej Danecki



Co to jest koprocesor pisałem już rok temu (*Koprocesor arytmetyczny*, „Bajtek" 9/89), gdy nie istniał jeszcze klan IBM-a. Postanowiłem jednak wrócić do tego tematu. Powodów jest kilka, a najważniejszym jest ignorancja stwierdzona u wielu znanych mi użytkowników PC. Żeby nie powtarzać zbyt wielu informacji już znanych tym z Was, którzy czytali poprzedni artykuł, zajmiemy się głównie współpracą koprocesora z najbardziej rozpowszechnionym w Polsce kompilatorem — Turbo Pascalem. Nie znaczy to jednak, że zawarte w nim informacje nie przydadzą się korzystającym z innych kompilatorów.

KOPROCESOR

(przekładany Turbo Pascalem)

Koprocesor — co i po co?

Koprocesor to taki układ, którego zadaniem jest wyręczanie głównego procesora w wykonywaniu pewnych czynności. Najszerzej znane są koprocesory arytmetyczne, których zadaniem jest wykonywanie operacji arytmetycznych na liczbach zmiennoprzecinkowych, ale oprócz nich istnieją również koprocesory graficzne, pomagające w konstruowaniu obrazu, i inne. Koprocesor arytmetyczny w PC to osobny układ scalony, potrafiący wykonywać różne operacje na liczbach rzeczywistych — od najprostszego dodawania, przez mnożenie i dzielenie aż po obliczanie logarytmów, wartości funkcji wykładniczych i trygonometrycznych. Wszystkie (albo prawie wszystkie) te operacje wykonywane są w sposób sprzętowy, bez angażowania samego procesora, który w tym czasie odpoczywa czekając na wynik.

Interesujące nas (czyli pasujące do komputerów kompatybilnych z IBM PC) koprocesory produkują w tej chwili trzy firmy, ale skoncentrujemy się tylko na kościach produkowanych przez Intela. Zasadniczo są ich trzy modele — 8087, 80287 i 80387. Każdy z tych modeli występuje w kilku odmianach, przeznaczonych do pracy przy różnych częstotliwościach zegara taktującego, jednak różnice są czasem znacznie większe niż można by się było spodziewać. Dotyczy to szybkości; przykładowo 80387 w wersji 33 MHz był przed uruchomieniem produkcji dość mocno przeprojektowany w stosunku do układu 25 MHz, toteż wzrost szybkości jest większy niż można by się spodziewać na podstawie samej zmiany częstotliwości zegara. Między modelami różnice są dość znaczne, ale zachowana została między nimi kompatybilność „w górę" — stare programy na nowym koprocesorze można wykonywać bez żadnych problemów, choć na pewno bez pełnego wykorzystania możliwości nowych układów (dotyczy to zwłaszcza 80387). Tyle o krzemowym ciele, przechodzimy do programowej duszy.

Turbo Pascal — real i double

Podstawowym typem rzeczywistym w TP jest **real,** który powstał wiele lat temu, gdy koprocesor był tylko marzeniem niektórych projektantów. Typ został wymyślony, a operacje na nim po raz pierwszy zaimplementowane bodaj w firmie Microsoft, która postanowiła wycisnąć z procesora 8088 maksimum szybkości i dokładności w obliczeniach zmiennoprzecinkowych. Wszystkie kompilatory Turbo Pascala wcześniejsze niż 4.0 dysponują tylko tym typem rzeczywistym.

Tymczasem koprocesory Intela korzystają z trzech własnych, zupełnie innych formatów zmiennoprzecinkowych (patrz tabelka), zatwierdzonych jako standard przez amerykańską organizację IEEE. Poczynając od wersji TP 4.0 formaty te mają swoje odpowiedniki w postaci typów **single, double i extended.** O ile pierwszy z nich ze względu na kiepską precyzję nie zawsze nadaje się do użycia, dwa pozostałe ze swoimi kilkunastoma cyframi znaczącymi i dużym zakresem dopuszczalnych wartości mogą zadowolić nawet dość wybrednych liczbomanów. Operacje na liczbach zapisanych w tych formatach są wykonywane przez sam koprocesor, który również automatycznie dokonuje konwersji formatów. Typ **real** w pewnym sensie znalazł się na bocznym torze — przed załadowaniem go do rejestrów koprocesora zapamiętana w nim liczba musi zostać na drodze programowej rozszerzona do formatu **double,** a po wykonaniu działania znów obcięta do sześciu bajtów.

W czasach Turbo Pascala 4.0 cztery różne formaty liczb zmiennoprzecinkowych stanowiły bogactwo iluzoryczne. Jeżeli skompilowany program miał działać na każdym komputerze niezależnie od konfiguracji, można było korzystać tylko z typu **real**. Program używający pozostałych typów na komputerze bez koprocesora kończył swoje działanie wyświetlając komunikat **Numeric co-procesor required.** Sytuacja zmieniła się wraz z pojawieniem się nowej wersji kompilatora — 5.0.

Emulator

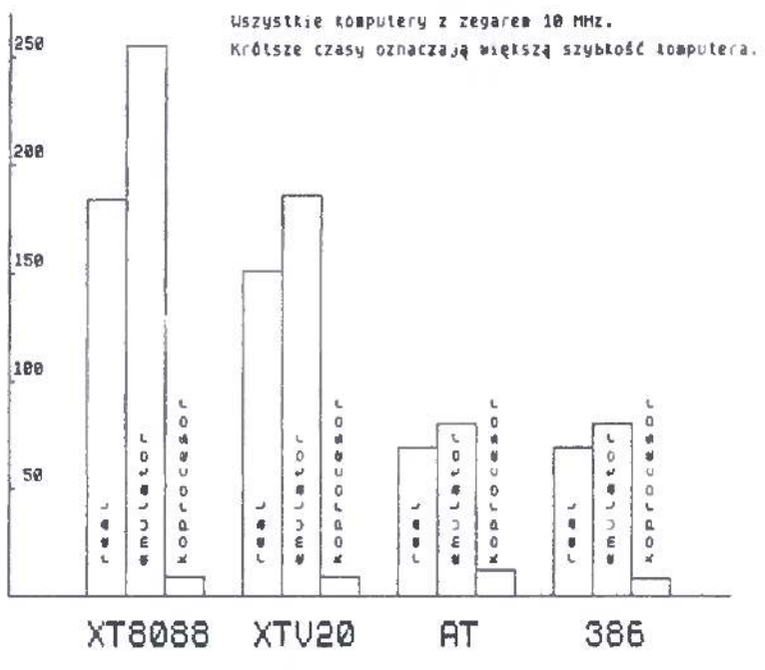
Zmiana polegała na rozszerzeniu biblioteki procedur systemowych o emulator koprocesora (o rozmiarze blisko 10 KB). Oczywiście emulator jest znacznie wolniejszy, ale — przynajmniej w teorii — zastępuje funkcjonalnie koprocesor. Dzięki niemu stało się możliwe uruchomienie programu korzystającego z formatów koprocesora przy jego nieobecności. W praktyce emulator w wersji 5.0 co najmniej w dwóch przypadkach okazywał się niezgodny z koprocesorem. Oba przypadki są zilustrowane wydrukami 1 i 2 — efekty działania programów zależą od konfiguracji.

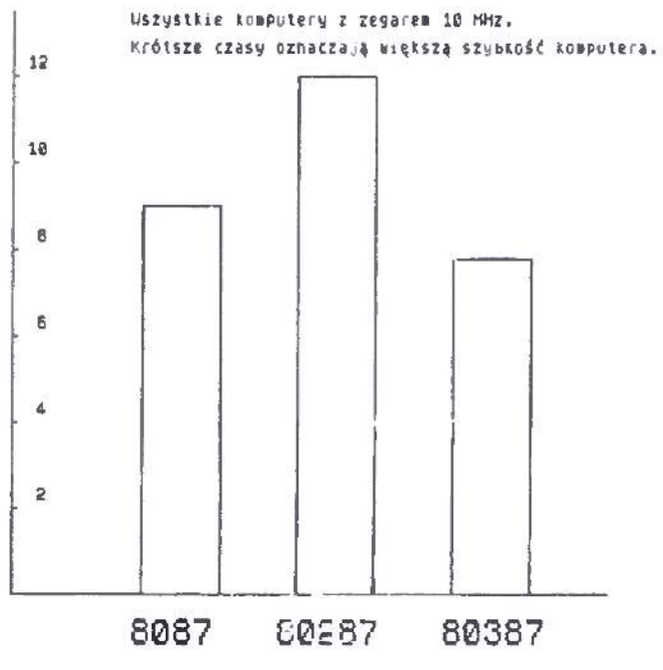
Pierwszy program w obecności koprocesora padnie dopiero, gdy zmienna **a** osiągnie wartość 11357, w przeciwnym razie błąd nadmiaru pojawi się znacznie wcześniej — emulator nie potrafi policzyć wartości funkcji wykładniczej, gdy jej argument jest większy niż 2839. Ten błąd pozostał również w emulatorze rozprowadzanym z TP 5.5.

Odwrotnie jest z drugim programem — wykonuje się on bezbłędnie w przypadku braku koprocesora, pada zaś w jego obecności. Związane jest to z budową kości, mającej osiem równoważnych rejestrów (tworzących wewnętrzny stos). Turbo Pascal 5.0 argumenty zmiennoprzecinkowe do podprogramów przekazuje w rejestrach koprocesora, toteż może ich być najwyżej osiem. Stos emulatora nie ma takiego ograniczenia. Dlatego w obecności koprocesora wywołanie funkcji z dziewięcioma argumentami powoduje błąd, który nie wystąpi w przypadku braku koprocesora. Wersja 5.5 kompilatora została zmodyfikowana i prezentowany program wykonuje się identycznie, niezależnie od konfiguracji.

Szybkość

Żeby móc w możliwie pewny sposób powiedzieć co jest szybsze, a co wolniejsze, przeprowadziłem serię testów. Polegały one na mierzeniu czasu potrzebnego do wykonania kilku różnych programów wykonujących wyłącznie obliczenia na komputerach wyposażonych w koprocesor i na kompute-





Ciekawie wygląda porównanie szybkości dwóch procesorów stosowanych w XT. Pierwszy z nich to oryginalna konstrukcja Intela — 8088/8086, drugi to wytwór japońskiego NEC-a — V20/V30. Przy zachowaniu pełnej kompatybilności programowej i sprzętowej, co oznacza, że w każdym XT można zmienić 8088 na V20 (8086 na V30), japończyk jest o kilkanaście procent szybszy — według wyników naszego testu o 18%, według innych o około 12%. Różnica bierze się z faktu intensywnego używania w teście emulatora operacji mnożenia i dzielenia liczb całkowitych. Na tym polu V20 jest dwa razy szybszy od konkurenta. W USA V20 kosztuje około 5 dolarów za sztukę i często bywa używany do produkcji XT zamiast 8088.

rach bez koprocesora. Oczywiście żeby porównania miały jakikolwiek sens, musiały dotyczyć obliczeń wykonywanych z tą samą dokładnością, czyli przy wykorzystaniu tego samego formatu. Wymusza to korzystanie z emulatora, ale — na wszelki wypadek — sprawdziłem również, jak szybko prowadzone są obliczenia na liczbach typu **real.**

Wyniki testów to kilka sporych tablic wypełnionych liczbami, których nie warto w całości prezentować. W związku z tym

omówimy tylko wnioski.

Po pierwsze — warto używać koprocesora. Wniosek jest mało oryginalny, ale ciekawe jest porównanie maksymalnego stopnia, w jakim zostają przyśpieszone różne modele komputerów. Najwyraźniejszy jest zysk w przypadku XT z procesorem 8088 — program wykonuje się w czasie nawet 28 razy krótszym. Zamiana procesora na odpowiadający mu funkcjonalnie V20 powoduje zmniejszenie różnicy między emulatorem a koprocesorem do mniej więcej 21 razy, co ciągle jest znacznie większą wartością niż obserwowane w pozostałych przypadkach. Najmniejszym zyskiem wykazuje się AT — koprocesor przyśpiesza je w najlepszym razie niecałe siedem razy. 386 (25 MHz) z dziesięciokrotną różnicą szybkości znajduje się gdzieś po drodze, raczej bliżej AT niż XT. (Przyczyny takich wyników wyjaśniłem rok temu). Oczywiście duże znaczenie ma jakość emulatora — ale na ten rozprowadzany z Turbo Pascalem nie ma powodu narzekać, jest napisany bardzo starannie i bardzo szybki.

Po drugie, tam gdzie dokładność oferowana przez typ **single** jest wystarczająca, nie warto korzystać z typów **double** ani **extended** — mniejsza liczba bajtów, które trzeba przenieść z rejestru koprocesora do pamięci, może spowodować przyśpieszenie działania programu o niemal 30%.

Po trzecie, podczas korzystania z koprocesora nie wolno używać typu **real** — czas potrzebny na przekształcenie formatu powoduje spowolnienie

Porównanie typów rzeczywistych (bez typu comp)

typ	rozmiar	zakres	cyfry znacz.
real	6	2.9E-39..1.7E38	11-12
single	4	1.5E-45..3.4E38	7-8
double	8	5.0E-324..1.7E308	15-16
extended	10	1.9E-4951..1.1E4932	19-20

obliczeń o kilkanaście (do dwudziestu) procent w porównaniu z typem **extended.**

Po czwarte, czas potrzebny do wykonania obliczeń na bazie typu **real** i biblioteki systemowej w porównaniu z obliczeniami wykonanymi z pomocą typu **extended** i emulatora jest krótszy o około 20%.

Po piąte i ostatnie — w obecności koprocesora obecność emulatora nie ma widocznego wpływu na szybkość obliczeń, mimo konieczności wykonywania przez program wyboru między emulatorem i koprocesorem.

Co z dokładnością?

Przy korzystaniu z omawianych typów, niezależnie od tego, czy korzystamy z emulatora czy koprocesora, wyniki pojedynczych działań są dokładne do przedostatniej cyfry znaczącej (patrz tabelka). Ostatnia cyfra zwykle trochę drepcze w miejscu, ale trudno od niej wymagać lepszego zachowania. W jednym przypadku jednak ta reguła zawodzi — dla liczb typu **single** mniejszych niż 3E-39. Wiąże się to z pewnym oszustwem — mantysa w tych liczbach nie jest znormalizowana, lecz wypełniona od lewej strony zerami. Dzięki temu, przy ośmiobitowej cesze liczby są mniejsze niż 3E-39, kosztem utraty cyfr znaczących. (Jeżeli ktoś nie rozumie o co chodzi, niech przeczyta artykuł: *Matematyczne podstawy zapisu liczb w postaci zmiennoprzecinkowej*, „Bajtek" 9/10 1988).

Jak liczyć szybko?

Oczywiście bardzo dużo zależy od zadania, jakie przed sobą stawiamy. Niewątpliwie najszybsza jest kombinacja koprocesora i typu **single,** jednak często nie zapewnia ona wystarczającej dokładności. Najlepszym rozwiązaniem jest wtedy użycie typu **double,** co przy starannym napisaniu programu (takim, by zmniejszyć do minimum liczbę przesłań danych między koprocesorem a pamięcią) wcale nie musi spowodować jego spowolnienia. Przykładowo, zastąpienie dwóch kolejnych instrukcji

```
b:=ln (a); c:=sin (b);
```

jedną, równoważną (o ile nie jest nam potrzebna wartość **b**):

```
c:=sin (ln(a));
```

spowoduje przyśpieszenie wykonania o 14% — wynik logarytmowania nie musi być przekazany do pamięci ani pobrany z niej na stos koprocesora.

Staranne napisanie programu i wybór właściwej reprezentacji danych może przyśpieszyć obliczenia nawet o kilkadziesiąt procent. Tam, gdzie potrzebny jest kilkakrotny wzrost szybkości operacji zmiennoprzecinkowych, trzeba, niestety, zmienić komputer. W przypadku PC oznacza to konieczność zakupu 386 lub nawet 486. Te ostatnie komputery — według testów opublikowanych w „Byte" z maja 90 — są w operacjach zmiennoprzecinkowych ponad dwukrotnie szybsze niż 386 z koprocesorem. Dobrze napisany program i 486 z zegarem 33 MHz, to w tej chwili najszybsze rozwiązanie w klasie IBM PC. A że najdroższe — cóż, nie od dzisiaj wiadomo, że lepiej być zdrowym i bogatym niż chorym i biednym.

Marcin Borkowski

wydruk 1:

```
{$E+,N+}
const
  a   :  extended = 2838.0;
begin
  repeat
    writeln(a:5:0,exp(a):30:6);
    a:=a+1
  until false
end.
```

wydruk 2:

```
{$E+,N+}

procedure x(a,b,c,d,e,f,g,h,i:extended);
begin
  writeln(a+b+c+d+e+f+g+h+i)
end;

begin
  x(1,1,1,1,1,1,1,1,1)
end.
```

NA KOLANACH...

Asemblerowiec, to programista, który chodzi na kolanach tam, gdzie inni jeżdżą Mercedesami. Chodzenie na kolanach pozwala jednak na wepchnięcie się w dziury niedostępne dla wyprostowanych grubasów, dlatego też mimo coraz lepszych, szybszych i wydajniejszych kompilatorów Pascala, C, Fortranu i innych języków, asembler nie traci na ważności.

Coraz mniej wprawdzie programów pisanych w całości w asemblerze, ale pojedyncze procedury, na przykład graficzne, zwykle pisane są właśnie w kodzie maszynowym.

Książka „Asembler 8086/88", o której mam zamiar napisać, powstała — przynajmniej z grubsza — właśnie w celu ułatwienia życia wszystkim tym, którzy trafia na potrzebę skorzystania z asemblera. Co prawda autor, pan Eugeniusz Wróbel, większy nacisk kładzie na pisanie w asemblerze całych programów niż pojedynczych procedur, ale ta różnica w filozofii nie wpływa na przydatność książki dla obu grup zadań.

Dwie podstawowe części książki, to rozdział poświęcony samemu asemblerowi, czyli językowi programowania, rozumianemu przez kompilator MASM (i TASM w trybie emulacji), i rozdział o rozkazach wykonywanych przez procesory rodziny 80 i 88/86. Oprócz tego, znajdziemy trochę informacji o architekturze procesorów, o uruchamianiu programów asemblerowych, oraz krótki i zwarty opis przerwań BIOS-u i usług DOS-u. W sumie — dość przejrzyście zorganizowane podręczne kompendium wiedzy dla programisty.

Z mojego punktu widzenia, czyli z punktu widzenia programisty piszącego w asemblerze tylko pojedyncze procedury, najważniejsza jest część książki poświęcona omówieniu zestawu rozkazów mikroprocesora. Do omówienia zostały one połączone w pewne grupy funkcjonalne — przykładowo razem omówione zostały wszystkie rozkazy przenoszenia danych między rejestrami i pamięcią, swoje podrozdziały mają rozkazy skoków, operacji logicznych, arytmetycznych itd. Mogło by to być niewygodne przy poszukiwaniu informacji na temat konkretnego rozkazu, ale że wszystkie mnemoniki zostały umieszczone w skorowidzu, do potrzebnego opisu można dotrzeć w ciągu kilkunastu sekund.

Opis każdego rozkazu zawiera informacje o efektach jego wykonania, dostępnych sposobach adresowania pamięci i rejestrów, zmienianych znacznikach i ilości taktów zegara potrzebnych do wykonania rozkazu. Taki zestaw informacji jest godny pochwały, jednak sposób ich prezentacji może wywołać głęboką frustrację. Do opisu wykonywanej operacji zastosowano bowiem notację ISP, wyjątkowo niewygodną do analizy. Powoduje to konieczność mozolnego przegryzania się (na kolanach!) przez kilka linijek zupełnie niestrukturalnego zapisu, składającego się ze strzałek, średników, gwiazdek, kwadracików, czterech rodzajów nawiasów i kilkunastu słów kluczowych. Jest to rozwiązanie nadające się na konkursy świąteczne w Bajtku, ale nie do normalnej pracy; odwracające uwagę od tego co naprawdę ważne, i ujmujące książce wiele z jej wartości. Po znalezieniu opisu potrzebnego rozkazu w ciągu piętnastu sekund traci się bowiem piętnaście minut na zanalizowanie zapisu, mówiącego co się stanie z zawartością rejestrów i pamięci.

Czy czegoś w książce brak? Kilku rzeczy. Część z nich — opis (przynajmniej zgrubny) koprocesora, dokładniejszy opis 80386, a zwłaszcza trybu 32 bitowego — wychodzi poza wynikający z tytułu zakres książki, ale chyba by jej nie zaszkodziły. Są jednak rzeczy, których brak drażni — na przykład tablica kodów rozkazów, służąca do ręcznej asemblacji/dizasemblacji — czasem trzeba rozgryźć dwa czy trzy rozkazy, i wtedy aż się prosi o zrobienie tego na piechotę. Wprawdzie przy opisach rozkazów są zamieszczone informacje o ich kodach, ale w formie równie wygodnej jak opis działań wykonywanych przez procesor — czyli praktycznie nie do użytku, zwłaszcza że informacje te są rozrzucone na kilkudziesięciu stronach. Zabrakło mi także choćby jednego słowa o istnieniu Turbo Debuggera — jest cały rozdział poświęcony programom wspomagającym uruchamianie, ale Turbo Debugger nie został nawet wymieniony, a jest to jeden z najlepszych tego typu programów.

Czy można tę książkę z czystym sumieniem polecić programistom? Raczej tak, choć mam na ten temat nieco mieszane odczucia. Z jednej strony — niewątpliwie jest w niej wszystko to, co może być potrzebne; z drugiej strony — część informacji jest przedstawiona w sposób uniemożliwiający jej sprawne wykorzystywanie (a wiadomo — choćby na podstawie programu Norton Guide — że można je przedstawić w sposób bardzo przejrzysty). Jest to niestety choroba większości wydawanych u nas książek mających służyć jako podręczne kompendia wiedzy. Na ich tle „Asembler 8086/88" nie wygląda wcale źle, dzięki dość trafnemu wyborowi zawartości. Co więcej — gdyby Wydawnictwa Naukowo-Techniczne zdecydowały się kiedyś na drugie, poprawione wydanie, mają do dyspozycji bardzo dobry materiał na wygodną i użyteczną książkę.

Marcin Borkowski

Eugeniusz Wróbel, „Asembler 8086/88", Wydawnictwa Naukowo-Techniczne, Warszawa 1990, nakład 8000, cena 39 000 zł.

wolno, wolniej, najwolniej...

Szybkość komputera, o której zwiększenie walczą konstruktorzy, jest bardzo potrzebna w wielu zastosowaniach, ale czasami może być zmorą.

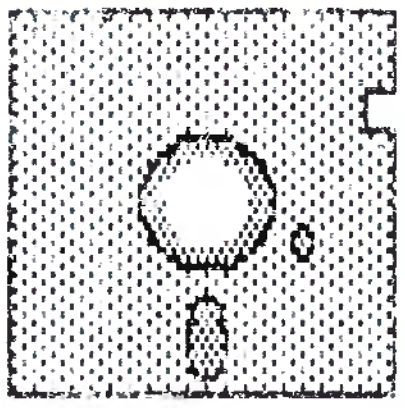
Dzieje się tak najczęściej, gdy usiłujemy używać szybkiego AT (lub 386) do grania w Diggera, ale nie tylko. Przelatujące z szybkością światła przez ekran informacje, których nie sposób przeczytać, zdarzają się również w przypadku poważnych programów użytkowych pisanych w czasach, gdy standardem było XT z zegarem 4.77 lub 8 MHz. Komputer oparty na procesorze 80386, kompatybilny ze staruszkiem XT, ale szybszy od niego 10 razy, to już swoista norma na tak zwanym zgniłym Zachodzie, a i u nas zdarza się coraz częściej. Co zrobić, żeby go spowolnić i w spokoju przeczytać informację o klawiszach funkcyjnych lub uciec przed goniącym potworem? Samo przełączenie zegara w wersji Turbo na Normal przestało wystarczać już kilka lat temu, toteż trzeba szukać bardziej zaawansowanych rozwiązań. Jedno z nich prezentuję poniżej. Jest to krótki (198 bajtów po skompilowaniu) program rezydentny slow.com, który wprawdzie nie zmniejsza szybkości, z jaką działa procesor, ale wymusza na nim bezproduktywne kręcenie się w kółko w pętli opóźniającej. Jak wiadomo, 18.2 raza na sekundę generowane jest w komputerach IBM PC przerwanie 1Ch. Pozwala to na zainstalowanie różnych programów wykonujących swoje zadanie w tle, bez przeszkadzania głównemu procesowi. Z tej możliwości właśnie korzystamy — niezależnie od tego, jaki program jest w danej chwili wykonywany, 18.2 raza na sekundę uruchamiana jest pętla opóźniająca, której rozmiar można dowolnie regulować. Pozwala to na spowolnienie komputera w takim stopniu, jaki jest nam potrzebny. Program doprowadza się do postaci slow.com w sposób opisywany już kilkakrotnie, więc nie będziemy go tym razem powtarzać. Po uruchomieniu i zainstalowaniu się w systemie, slow pozwala na zmianę szybkości komputera za pomocą klawiszy shift — naciśnięcie lewego powoduje przyśpieszenie komputera (oczywiście nie w nieskończoność), naciśnięcie prawego powoduje spowolnienie tempa pracy. Programu po zainstalowaniu nie można już usunąć z pamięci, ale jego wpływ na szybkość komputera może być na tyle niewielki, że nie będzie miał znaczenia przy wielu zastosowaniach. Uwaga: niektóre gry mają niemiły zwyczaj odcinania programów korzystających z przerwania 1Ch od możliwości uruchomienia. Program slow będzie wtedy bezradny.

Marcin Borkowski

```
ASSUME cs:code

code SEGMENT

                org     100h
Start:
                jmp     install

old1Co          dw      0               ; Adres starego programu
old1Cs          dw      0               ; przerwania 1Ch.
slc             dw      10h             ; Rozmiar pętli
                                        ; opóźniającej.

slow:
                push    cx
                push    ax
                push    ds
                xor     ax,ax
                mov     ds,ax
                test byte ptr ds:417h,1     ; Jeśli prawy shift
                jz noinc                ; wciśnięty, zwiększ
                inc     cs:[slc]        ; rozmiar pętli.
                jmp wait0
noinc:
                test byte ptr ds:417h,2     ; Jeśli lewy shift
                jz wait0                ; wciśnięty, zmniejsz
                dec     cs:[slc]        ; rozmiar pętli.
                jnz wait0               ; slc nie może być
                mov     cs:[slc],1      ; mniejsze niż 1.
wait0:
                mov     cx,cs:[slc]
wait1:                                  ; Pętla zewnętrzna,
                push    cx              ; powtarzana slc razy.
                mov     cx,100h
wait2:                                  ; Pętla wewnętrzna,
                loop wait2              ; ma 100h przebiegów.
                pop     cx
                loop wait1
                pop     ds              ; Odtwórz początkowe
                pop     ax              ; wartości rejestrów,
                pop     cx              ; i wykonaj skok
                jmp dword ptr cs:old1Co ; "w łańcuszku".
install:
                mov     dx,offset info
                mov     ah,09h
                int     21h             ; Wydrukuj opis.
                mov     ax,351Ch        ; Pobranie adresu
                int     21h             ; przerwania 1Ch
                mov     cs:old1Co,bx    ; i zapamiętanie go
                mov     cs:old1Cs,es    ; na przyszłość.
                mov     dx,offset slow  ; Ustawienie nowego
                push    cs              ; adresu przerwania 1Ch
                pop     ds              ; na początek procedury
                mov     ax,251Ch        ; slow.
                int     21h
                mov     dx,offset install ; Skończ, zostawiając
                int     27h             ; w pamięci wszystko
                                        ; przed adresem install.

info:           db      'Spowalniacz PC, (c) Bajtek 1991',13,10
                db      '  Prawy shift - wolniej,',13,10
                db      '  Lewy shift  - szybciej.',13,10,'$'

code ENDS

END Start
```

SPIS TREŚCI BAJTKA 1990

	Giełda	Sklep	Pewex	Zachód
	tys. zł		$	
SINCLAIR				
ZX 81	200	—	—	—
ZX Spectrum 48	800	990	—	—
ZX Spectrum +	900	1100	—	—
ZX Spectrum + 2	1200	—	—	—
Timex 2048	1000	1200	—	—
Stacja FDD 3000	1100	1300	—	—
Stacja FDD 3	800	—	—	—
drukarka GP–50	650	550	—	—
Masterface I	120	—	—	—
AY 3–8910	170	—	—	—
COMMODORE				
C 64	1900	1950	207	—
C 128	2500	2700	—	—
C 128 D	4000	4200	599	—
Amiga 500	5000	5800	795	—
Amiga 2000	12000	—	—	—
Amiga 3000	—	—	—	—
magnetofon	350	400	30	—
stacja 1541	1800	1900	—	—
stacja 1571	2200	—	—	—
stacja Oceanic	1900	—	—	—
drukarka LC 10C	2100	—	299	—
Final II	100	120	—	—
Final III	230	250	—	—
Action Replay	350	—	—	—
ATARI				
Atari 800 XL	1200	1250	—	—
Atari 65 XE	1500	1600	159	—
Atari 130 XE	1900	2200	239	—
Atari 520 STFM	—	4300	499	—
Atari 1040 STFM	—	—	899	—
magnetofon	300	350	51	—
stacja CA 2001	2600	—	249	—
monitor SM124	2200	—	—	—
monitor SM224	—	—	480	—
Turbo 2000	100	100	—	—
Centronics	200	220	44	—
AMSTRAD				
Amstrad 464	2000	2200	—	—
Amstrad 6128	2700	2900	—	—
Amstrad PCW 8256	—	—	—	—
Amstrad PCW 8512	3600	—	—	—
IBM				
IBM PC XT stand.	8000	9500	499	—
IBM PC AT stand.	11000	—	789	—
IBM PS/2	—	16000	—	—
HD 20 MB z kontrol.	3100	—	260	—
napęd 5.25''	600	700	—	—
monitor amber	—	—	219	—
klawiatura	440	—	—	—
INNE				
dyskietki 5.25''	4.5	4.5–8	0.5–3	—
dyskietki 3.5''	15	15–20	1–3	—
dyskietki 3''	29	35–45	—	—
kasety C–60	10	—	1	—
monitor	700	750	—	—
joystick	60–80	60–100	7–11	—



INDYWIDUALNY BANK DANYCH

Andrzej Motyl posiada komputer SHARP MZ-1P16. Pragnie nawiązać kontakt z użytkownikami tego typu komputera, oraz firmą która zajmuje się oprogramowaniem w tym systemie. Interesują mnie programy użytkowe.
Adres: 21-040 Świdnik k/Lublina, ul. Piękna 20 t 12753.

Maciej Szudło lat 11, posiada Atari 65XE, stację CA-2001 oraz magnetofon CA-12. Oprogramowanie: 200 gier i kilka programów użytkowych. Proponuje wymianę doświadczeń.
Adres: oś. Kolorowe 17a/73, 31-940 Kraków Nowa Huta.

Wiktor Koleśniak, uczeń kl. VII, posiada Commodore 16, firmowy magnetofon, około 50 gier i 30 programów muzycznych. Proponuje wymianę doświadczeń i programów.
Adres: 87-800 Włocławek, ul. Ostrowska 18/9.

Waldemar Szatanek, lat 14, posiada komputer Texas Instruments — Ti 99/4A oraz ok. 15 programów z instrukcją w języku francuskim. Nawiąże kontakt w celu wymiany doświadczeń i oprogramowania.
Adres: ul. Łąkowa 13, 05-081 Laski Wwskie.

Tomasz Paluch lat 15 posiada Commodore 64, Proponuje wymianę oprogramowania (na kasetę lub dysk).
Adres: ul. Gen. Andersa 4, 42-600 Tarnowskie Góry.

Paweł Mikołajczak lat 15 posiada Amstrad CPC 6128, pragnie nawiązać kontakt w celu wymiany doświadczeń i oprogramowania.
Adres: ul. Poznańska 5, 63-720 Koźmin woj. kaliskie.

Przemysław Jeziorski lat 13. Posiada Commodore 64 ze stacją dysków 1541 i drukarką MPS 803. Proponuje wymianę oprogramowania i doświadczeń.
Adres: 10-284 Olsztyn ul. Kolejowa 7 m 1.

Jacek Nowak lat 19 posiada Atari 520 ST z podwójną stacją dysków. Proponuje wymianę oprogramowania i doświadczeń.
Adres: ul. Telimeny 17/78, 30-835 Kraków.

Sebastian Zenderowski, posiada TIMEX 2048. Nawiąże kontakt w celu wymiany oprogramowania, doświadczeń i literatury.
Adres: Oś. Słoneczne 5 m 20, 11-010 Barczewo.

Piotr Konieczny posiada Atari 1040 STFM z monitorem monochromatycznym, Scaner firmy „Brother", drukarkę Star LC-10, stację dysków 5,25". Nawiąże kontakt w celu wymiany doświadczeń i oprogramowania.
Adres: ul. Chopina 21, 56-400 Oleśnica woj. wrocławskie.

Mariusz Rewkowski, lat 18 posiada Commodore C64C, magnetofon 1530 Batassete, cartridge — „Final II" i „Action Plus VGO". Dysponuje zestawem 400 programów. Interesuje się muzyką i grafiką komputerową.
Nawiąże kontakt w celu wymiany oprogramowania, doświadczeń i literatury (j. niemiecki).
Adres: Jarogniewa 30/5, 71-664 Szczecin.

Grzegorz Marchlewicz lat 14, posiada Atari 65 XE, magnetofon XC 12 (Turbo 2000) oraz stację dysków CA 2001. Nawiąże kontakt w celu wymiany doświadczeń, literatury i programów.
Adres: Al. Wilanowska 99/1, 02-765 Warszawa.

Jeżeli posiadasz komputer Commodore (od VC20 do Amigi) jeżeli chcesz poznać wielu użytkowników tego komputera wymienić literaturę, oprogramowanie i doświadczenia, przyłącz się do drugiego w Polsce korespondencyjnego klubu miłośników komputerów Commodore.
Adres: **„Komandor" Grzegorz Bańbuła**, ul. Brzóski 46/6, 41-800 Zabrze.

Leszek Kmieć lat 15, posiada Timexa oraz generator AY 3-8910 oraz 250 gier, 30 programów na AY. Pragnie nawiązać kontakt w celu wymiany doświadczeń oraz oprogramowania. Odpowie na każdą propozycję.
Adres: ul. Nawrockiego 28/1, 95-200 Pabianice.

Szymon Makowski lat 15, posiada Atari 520 STM i stację dysków SF 314. Pragnie nawiązać kontakt z użytkownikami tego komputera w celu wymiany oprogramowania i doświadczeń.
Adres: ul. Chełmżyńska 38, 04-247 Warszawa.

Drogi Bajtku!

Na listy czytelników odpowiadają autorzy „Bajtka"

Wybrałem Commodore'a ze względu na jakość gier. Ze smutkiem dowiedziałem się, że większości świetnych gier nie będę nigdy miał posiadając tylko magnetofon. Podobno istnieją tylko wersje dyskowe tych programów. Jak jest naprawdę? Arek Romanek, Giżycko

Faktycznie gry na dyskach są z reguły ciekawsze niż gry dostępne w wersji kasetowej. Pomocą może tu być moduł ACTION REPLAY, którym można wycinać dowolne kawałki gier i zgrywać na kasetę. Gry przygotowane w ten sposób można uzyskać na giełdach komputerowych. Np. długo krążyły (i jeszcze krążą) gry w wersjach kasetowych takie jak: LAST NINJA I i II, TEST DRIVE itp. Jedyną wadą tego systemu jest to, że można grać tylko w dany fragment gry. Po przejściu planszy program z reguły się zawiesza. Proponuję więc zdecydowanie się na większy wydatek i zakupienie stacji dysków, bo magnetofon to już przeżytek. (TNL)

Dlaczego mikrokomputery Commodore C64 sprzedawane obecnie w sklepach Pewex'u i Baltony wyciszają tzw. „sample"?
Klub Commodore

C64 sprzedawane obecnie w naszych sklepach (na giełdach też) mają dziwną właściwość, mianowicie zdigitalizowany dźwięk jest wręcz nieprzyzwoicie wyciszany. Ci, którzy zakupili takie mikrokomputery gdy zauważyli tę wadę próbowali wymienić lub naprawić w serwisach gwarancyjnych swój sprzęt.

W punktach tych grzecznie, aczkolwiek stanowczo odmawiano im, dedykując przysłowie: „widziały gały co brały". Paru bardziej pomysłowych próbowało nękać producentów skargami, niestety bezskutecznie.

Wyciszanie „samplingów" spowodowane jest zmianą układów w wersjach C 64C. Układ SID został zmodyfikowany w ten sposób, że dźwięki generowane przez komputer są wyraźniejsze, lecz odbywa się to kosztem „samplingu". Są na to dwie rady: pierwsza to zmienić komputer, a druga — podłączyć Commodore'a do wzmacniacza. Przy jego pomocy uzyskamy całkiem przyzwoitą digitalizację. (TNL)

Czy można podłączyć myszkę do ZX Spectrum 48K? Czy działa ona tak jak joystick, a jesli nie — to jakie programy ją obsługują?
Paweł Warden, Gdynia

Producenci Spectrum nie przewidzieli zastosowania myszy dla swojego produktu, lecz istnieją specjalne interface'y umożliwiające przyłączenie tego pożytecznego stworzenia.
Mysz jest bardzo popularnym sprzętem i dla wielu komputerów stanowi ich integralną część. Działanie myszy jest analogiczne do działania joystick-a, ale nie jest takie samo.

Jedynym, znanym mi systemem dla ZX Spectrum jest AMX MOUSE, zawierający specjalny program graficzny, który obsługiwany jest przez mysz. ART STUDIO — najlepszy program graficzny dla Spectrum też posiada w swoim bogatym menu opcję AMX.

Jedynym mankamentem łączenia „trumny" z myszką jest jej cena — czasem przekraczająca wartość samego komputera. (MBP)

Do czego służy filtr ochronny przymocowywany przed ekranem monitora?
Wojciech Kowalczyk, Krosno

Filtr ochronny do monitorów ekranowych został stworzony po to, aby chronić operatorów komputerów przed zgubnym wpływem promieniowania z kineskopu. Czarne włókna filtru pochłaniają szkodliwe promieniowanie i poprawiają wyrazistość monitora przez podwyższenie kontrastu. Jeszcze jedną zaletą takiego filtru jest eliminacja refleksów świetlnych, których sprawcami mogą być obce źródła światła.

Obecnie na świecie produkuje się wysokiej klasy filtry ze specjalnych tworzyw, w Polsce są popularne filtry z włókien — wyglądające jak pończocha rozciągnięta na ramce. Mimo swego wyglądu zadanie swe spełniają bardzo dobrze, i każdemu, kto siedzi godzinami wpatrując się w ekran polecam takie usprawnienie. (MBP)

Od dwóch lat jestem użytkownikiem Amstrada 464. Muszę przyznać, że poznałem już trochę jego możliwości, ale nie potrafię poradzić sobie z następującymi problemami:
1. Do czego służy rubryka „Co piszczy pod klawiaturą"?
2. Po wpisaniu programu „Mini Organy" z „Bajtka" 5—6/86 pojawia się komunikat RESUM in 1030. Jak usunąć błąd?
3. Po wspisaniu programu Doublescreen nie działa czworościan i ukazuje się komunikat o niedobrej tablicy w linii 180.
4. Po wpisaniu „Uczymy mówić CPC" pokazuje się Type Mismatch in 3060.
5. Co do programu „Strzałka" z „Bajtka" 2/89, to nie wiem, jak użyć komend on, off, aby nie pojawiał się syntax error.
6. Może wiecie, jak grać w grę „Jagd Auf Roter Oktober"?
Byłbym wdzięczny za pomoc.
Jacek Nawara, Konin

Odpowiedzi na te pytania nie są łatwe, gdyż pytasz o programy z dawnych numerów „Bajtka". Po konsultacji z dobrze zorientowanymi osobami mogę odpowiedzieć:
1. Rubryka „Co piszczy pod klawiaturą" przeznaczona była dla osób piszących w asemblerze. Rubryka ta jest teraz kontynuowana w cyklu „Procedury systemowe Amstrada".
Odpowiedź na pytania 2, 3 i 4 jest jednakowa: musiałeś pomylić się przy wpisywaniu! Wszystkie wymienione programy działają, więc sprawdź dokładnie!
5. Niedokładnie przeczytałeś opis programu. Komendy muszą być przeznaczone znakiem „:", czyli :ON i :OFF.
6 .Gra, którą wymieniasz, to niemiecka wersja „Hunt for Red October". Jest to symulacja atomowej łodzi podwodnej, oparta na powieści Toma Clancy'ego. Gra ta była opiasana w „Komputerze", niewykluczone, że zajmnie się nią też „Top Secret". (MSZ)

SV 119 – 64 900	SV 123 – 119 900	SV 128 – 319 900	SV 210 – 219 000
SV 120 – 74 900	SV 125 – 239 900	SV 130 – 339 000	SV 500 – 119 000
SV 122 – 89 900	SV 126 – 169 900	SV 201 – 249 000	SV 510 – 119 000
SV 124 – 109 900	SV 127 – 249 900	SV 202 – 229 000	

SV 201 + SV 210 – 439 000
SV 202 + SV 210 – 399 000

sprzedaż hurtowa
tel. 23-98-53

udziela informacji
tel. 662-35-16

CENY DETALICZNE AKTUALNE DO 31 STYCZNIA 1991 r.

JAK ZAMAWIAĆ?

NALEŻY PRZESŁAĆ PIENIĄDZE PRZEKAZEM TELEGRAFICZNYM (DUŻY ZIELONY BLANKIET) na adres:

„TAL" Sp. z o.o.
ul. Mikowa 45
02-411 WARSZAWA

UWAGA! W miejscu na KORESPONDENCJĘ prosimy PODAĆ PO RAZ DRUGI: adres domowy, kod i symbol zamówionej pozycji.

CZAS REALIZACJI ZAMÓWIENIA 3-6 dni.
W PRZYPADKU BRAKU ADRESU ZWROTNEGO, NIE REALIZUJEMY ZAMÓWIENIA.
DO KAŻDEGO JOYSTICKA DOŁĄCZAMY KUPON KONKURSOWY

PROMOCYJNA SPRZEDAŻ KOMPUTERA AMIGA 500 i COMMODORE 64 — wersja angielska!!!

DDD — Dostawa Do Domu!!!
szczegóły tel.: 662-35-16

365 dni GWARANCJI

Adres firmy: „TAL" Sp. z o.o., Mikowa 45,
02-411 Warszawa Włochy

tel. 23-98-53 sprzedaż hurtowa
fax: 659-12-35
tel. 662-35-16 udziela informacji

szukaj znaku

tam znajdziesz joysticki

SV 119
Junior

2 Fire
6 Blaszanych styków
Prosty mechanizm

SV 120
Junior–Stick

2 Fire
6 Blaszanych styków
Uchwyt pistoletowy

SV 122
Quickjoy II

2 Fire
6 Blaszanych styków
AutoFire
Drążek lotniczy

SV 124
Turbo

6 Mikrostyków
AutoFire
Drążek lotniczy

SV 123
Supercharger

2 Fire
6 Mikrostyków
Ergonomiczna budowa
Precyzyjny mechanizm

SV 126
Jet Fighter

2 Fire
6 Mikrostyków
AutoFire
ACS-Regulator
szybkości AUTO
Obsługa pod kciuk"
Drążek lotniczy

SV 125
Superboard

6 Fire
10 Mikrostyków
AutoFire
Cyfrowy wyświetlacz
czasu
Sygnał dźwiękowy
Przełącznik dla
leworęcznych
Drążek lotniczy

SV 130
IR Infrared

1 Fire
5 Mikrostyków
Podczerwień
Daleki zasięg
Odbiornik

SV 128
Megaboard

4 Fire
10 Mikrostyków
AutoFire
6 cyfrowy stoper
ATM — Anti Tilt Mechanism
Fire Pad

SV 140
Enterprice

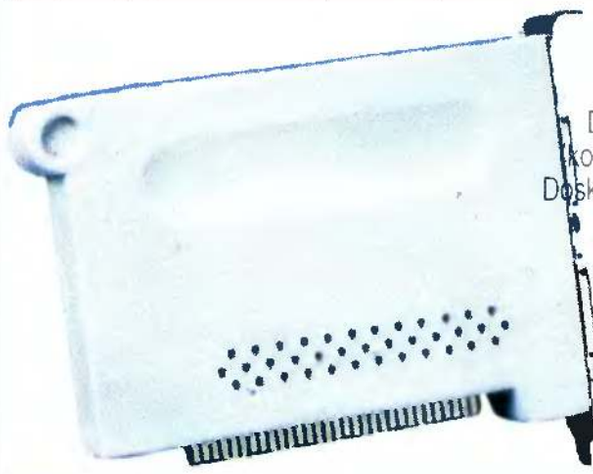
2 Fire
6 Mikrostyków
AutoFire
ACS — Regulator
szybkości AUTO
Drążek lotniczy
„kierownica"
Kabel 4 m

SV 201
Quickjoy M 5

Do IBM XT/AT
(kompatybilnych)
Współpracuje z Game-Card
lub I/O Card
2 Fire
2 AutoFire
6 Mikrostyków
Wybór AUTO
PSC — Regulator XY
Sygnalizacja Świetlna
Fire
ASC — Regulator szybkości
AUTO

SV 202
M 6 analog

Analogowy
DO IBM XT/AT
(kompatybilnych)
Współpracuje z Game-Card
lub I/O Card
2 Fire



SV 210
Game Card

Do IBM XT/AT
(kompatybilnych)
Doskonale pracuje
z M 5 i M 6

SV 127
Top Star

2 Fire
6 Mikrostyków
AutoFire
Przeźroczysta obudowa
SAS — Shock Absorbing
System
Platynowane części

SV 500
Van 3

Pudełka na dyskietki
80 sztuk 3½"
Zamknięcie na klucz

SV 510
Van 5

Pudełko na dyskietki
80 sztuk 5¼"
Zamknięcie na klucz

Quickjoy

TAL – najtaniej w Polsce!

JEDNOUKŁADOWE

O tym, że mikrokomputery bardzo dynamicznie wkraczają do życia codziennego, nie trzeba chyba nikogo przekonywać. Można je znaleźć w sprzęcie audio-video, kuchence mikrofalowej itp. Jednak najszerszy zakres zastosowań mają w rozlicznych dziedzinach przemysłu. Sterowanie procesami technologicznymi, maszynami, regulacje — to tylko drobny fragment możliwości ich zastosowania. Użycie sterownika mikrokomputerowego staje się wręcz niezbędne w miejscach niebezpiecznych dla zdrowia człowieka.

W znakomitej części zastosowań nieopłacalne jest wykorzystanie do ich budowy uniwersalnych i wydajnych mikroprocesorów, jakie spotykamy np. w popularnych mikrokomputerach domowych. Ich możliwości przetwarzania niekiedy wielokrotnie przewyższają potrzeby. Ponadto, aby na bazie popularnego mikroprocesora powstał funcjonalny sterownik, trzeba dołączyć do niego wiele dodatkowych układów. Są nimi pamięci ROM i RAM, układy wejścia-wyjścia, układy czasowe itp. Taka konstrukcja niejednokrotnie okazuje się zbyt duża i energochłonna.

Z tych właśnie przyczyn wiele firm produkuje mikrokomputery jednoukładowe. Układy te zawierają w jednym układzie scalonym procesor, pamięci ROM i RAM, układy wejścia-wyjścia. Bardzo często oprócz tych podstawowych bloków producenci umieszczają dodatkowo inne układy realizujące pomocnicze funkcje, jak: generator przebiegu zegarowego, układy czasowo-licznikowe.

Pierwsze mikrokomputery jednoukładowe pojawiły się pod koniec lat siedemdziesiątych, jednak były to urządzenia dość prymitywne i wolne. Jako przykład niech posłuży układ **MM 57140** firmy National Semiconductors. Posiada on 4-bitową jednostkę centralną, 55 bajtów pamięci RAM, 630 bajtów ROM. Maksymalna szybkość zegara wynosi tylko 280 kHz. Komunikacja z otoczeniem może być realizowana poprzez 24 dwukierunkowe linie wejścia-wyjścia. Dla programisty układ udostępniał swe 4 rejestry i 35 możliwych instrukcji.

Następne konstrukcje są już znacznie doskonalsze, najnowsze produkty wydajnością przetwarzania dorównują klasycznym mikroprocesorom **Z80, 8085** itp.

Jedną z najpopularniejszych jest cała rodzina układów firmy INTEL MCS 48. Wszystkie układy tej rodziny posiadają 8-bitową jednostkę centralną i są ze sobą zgodne programowo. Oznacza to, że program napisany na określony komputer z tej rodziny może być wykonywany na innym bez dodatkowych modyfikacji.

MIKROKOMPUTER 8048

Prezentację należy rozpocząć od najstarszego układu z tej rodziny — **8048.** Jest on wyposażony w 1 KB pamięci ROM i 64 bajty pamięci RAM. Ponadto zawiera dwa dwukierunkowe ośmiobitowe porty wejścia-wyjścia, jedno wejście przerwań, jeden licznik-timer. Na uwagę zasługują dwie specjalne linie wejściowe T0 i T1. Ich stan logiczny może wstępować w instrukcjach procesora jako warunek rozgałęzienia programu, analogicznie do bitów rejestru wskaźników.

Struktura mikrokomputera zawiera wbudowany generator zegarowy. Z zewnątrz należy dołączyć jedynie rezonator kwarcowy. Maksymalna częstotliwość wynosi zwykle 6 MHz.

Zarówno pamięć programu, jak i pamięć RAM można rozszerzyć przesłaniając je pamięcią zewnętrzną. Układ ma bowiem wyprowadzoną multipleksowaną szynę adresową i danych. Jeśli jednak tego nie uczynimy, to możemy wykorzystać szynę jako kolejny ośmiobitowy port wejścia-wyjścia.

Mikrokomputer **8048** posiada 12-bitowy licznik rozkazów, zatem maksymalną wielkością pamięci, którą może obsługiwać, jest 4 KB. Układ ogranicza pamięć programu i pamięć RAM — w granicznym przypadku może więc adresować 4 KB ROM i 4 KB RAM.

Programowanie sterowników z układem **8048** jest bardzo wygodne dzięki zestawowi 35 uniwersalnych instrukcji, ukierunkowanych specjalnie na sterowanie urządzeniami zewnętrznymi. Dodatkową zaletą jest możliwość dokonywania operacji logicznych bezpośrednio na portach wejścia-wyjścia.

Programista ma do dyspozycji dwa banki po osiem rejestrów, tzn. osiem podstawowych i tyle samo alternatywnych. Pewną ciekawostką może być to, że rejestry nie są integralną częścią procesora, lecz są umieszczone w pamięci RAM. Zawartość rejestrów można więc modyfikować na wiele sposobów, co podnosi uniwersalność i efektywność programowania. Rejestr można wskazać wymieniając jego nazwę w kodzie rozkazu, można również traktować go jak zwykłą komórkę pamięci. W pamięci jest również umieszczony 16-bajtowy stos.

Nie spotykaną w innych mikrokomputerach możliwością jest zdolność programowego odłączenia od zasilania jednostki centralnej, przy jednoczesnym pozostawieniu zasilania pamięci RAM. Pobór mocy przez układ spada wówczas do 10% poziomu nominalnego. Dla energooszczędnych wersji wykonanych w technologii CMOS wynosi on zaledwie 10 mW. Opuszczenie tego stanu może nastąpić po sygnale przerwania lub RESET. Możliwe jest zatem długotrwałe zasilanie bateryjne, czym nie pogardzą projektanci systemów alarmowych.

MODYFIKACJE MIKROKOMPUTERA 8048

Kolejne wersje układu zawierają zwiększoną ilość wewnętrznej pamięci ROM i RAM. Tak więc mikrokomputer **8049** posiada 2 KB ROM i 128 bajtów RAM. Układ **8050** zawiera 4 KB ROM i 256 bajtów RAM.

MIKROKOMPUTER 8051

Układem o znacznie szerszych możliwościach zastosowań jest **8051.** Niekiedy konstruktorzy nazywają go najlepszą konstrukcją firmy INTEL i jak się za chwilę okaże, jest w tym stwierdzeniu wiele prawdy.

Mikrokomputer **8051** zawiera: wewnętrzny oscylator do 12 MHz, cztery ośmiobitowe dwukierunkowe porty wejścia-wyjścia, dwa szesnastobitowe liczniki-timery sterowane przebiegami zewnętrznymi lub zegarem systemowym, interfejs RS-232 i dwa źródła zewnętrznych przerwań. Układ jest wyposażony w 4 KB ROM i 128 bajtów RAM. W odróżnieniu od układu 8048 licznik rozkazów jest 16-bitowy, co pozwala na podłączenie 64 KB zewnętrznej ROM i 64 KB RAM. Dzięki większej pamięci RAM układ posiada cztery banki rejestrów.

Modyfikacją **8051** jest układ **8052.** Mikrokomputer ten posiada w stosunku do **8051** dwukrotnie większą pamięć ROM i RAM, ale również zawiera dodatkowy licznik-timer z możliwością automatycznego przeładowywania.

Wszystkie wersje układów z rodziny MCS 48 mają swoje odpowiedniki z pamięcią EPROM zamiast ROM. Oznaczane są kodem **87XX,** gdzie XX to końcówka identyczna jak dla wersji z pamięcią ROM. Istnieją również układy bez wewnętrznej pamięci ROM. Mają one oznaczenie **803X.**

Osoby, które nie lubią lub też nie potrafią programować w języku asemblera, ucieszy zapewne fakt, że program dla mikrokomputera można przygotować w języku C. Specjalny kompilator na IBM PC potrafi wygenerować kod wynikowy dla układu **8051.** Ponadto niektórzy producenci umieszczają w pamięci ROM układu **8052** prosty interpreter języka BASIC.

Robert Magdziak