Nr ind. 366196

MOJE Atari

CENA 4500 ZŁ

1

MAGAZYN UŻYTKOWNIKÓW KOMPUTERÓW ATARI PAŹDZIERNIK 1990

MODEM XM-301P

Action (3)

ZACZYNAMY PROGRAMOWAC

CHARS DESIGNER

EDYTOR BASIC-a

OKNA

RAZ JESZCZE OMEGA CAD

Drogi Czytelniku!

Dwa lata temu w serii „Bajtkowych" wydań specjalnych ukazały się dwa numery poświęcone wyłącznie komputerom „Atari". Setki listów i telefonów do redakcji przekonały nas, że pomysł wydawania wyspecjalizowanych, monotematycznych pozycji ma sens. Tak jest na całym świecie, gdzie użytkownicy komputerów mają do dyspozycji często nawet kilka tytułów poświęconych jednej rodzinie sprzętu, czy nawet jednemu modelowi.

Pragniemy, aby w naszym kraju było podobnie. Właśnie dlatego Spółdzielnia „Bajtek" podjęła inicjatywę powołania do życia dwóch nowych tytułów: „Moje Atari" i „Top Secret". Wierzymy, że decyzja ta nie była pomyłką.

„Moje Atari" to jeden z naszych snów, który się spełnił. Część zespołu „Bajtka", która pracowała nad przygotowaniem poprzednich numerów specjalnych poświęconych tylko komputerom „Atari" już dwa lata temu zgłosiła możliwość redagowania pisma o takim charakterze. Nie były to czasy odpowiednie dla podobnych inicjatyw. W roku ubiegłym planowaliśmy wydanie dwóch numerów, ale w ramach byłej RSW więcej uwagi poświęcano wydawnictwom o charakterze propagandowym niż technicznym.

Dziś jest to przeszłość. Kiedy Sejm RP uchwalił ustawę o likwidacji RSW zrozumieliśmy, że los podarował nam szansę, o jakiej nawet nie śmieliśmy marzyć. Mogliśmy wziąć sprawy w swoje ręce. Tak też się stało. Znaleźli się ludzie dobrej woli, którzy zaufali grupie bardzo młodych zapaleńców i przeznaczyli dla naszej spółdzielni środki na wydawanie aż trzech pism. Jesteśmy im za to głęboko wdzięczni.

Wiemy, że nic nie przychodzi łatwo. Za każdym sukcesem kryje się ciężka, często mordercza wręcz praca. Kiedy nasze sprawy stały źle, zadawaliśmy sobie pytanie „Jeśli nie my — to kto? Jeśli nie dziś — to kiedy?". Odpowiedź na nie otrzymaliście, Drodzy Czytelnicy, w postaci pierwszego numeru naszego pisma.

Pragniemy, aby ten magazyn stał się źródłem rzetelnej informacji dla wszystkich właścicieli „Atari". W tym roku planujemy wydanie trzech numerów. W roku następnym sześciu.

Mamy nadzieję, że zechcecie wspierać nas swoimi pomysłami i tekstami. Piszcie do nas, postaramy się odpowiedzieć na wszystkie listy. Zorganizowaliśmy także prenumeratę czasopisma. Jesteśmy pełni obaw, ale i wiary w przyszłość.

Za dziesięć lat wszyscy będziemy żyli w XXI wieku. Już dziś komputery stały się w krajach wysoko rozwiniętych równie powszechne jak telefony. Jeśli Polska ma zamiar gonić coraz szybciej uciekający świat, MUSI dostosować się do obowiązujących tam standardów. Jednym z nich jest komputer.

Nie ma większego znaczenia, czy nosi on znaczek „Atari", „IBM" czy „Commodore", ważne jest jedynie to, czemu służy i czy człowiek, który się nim posługuje, wie jak to robić.

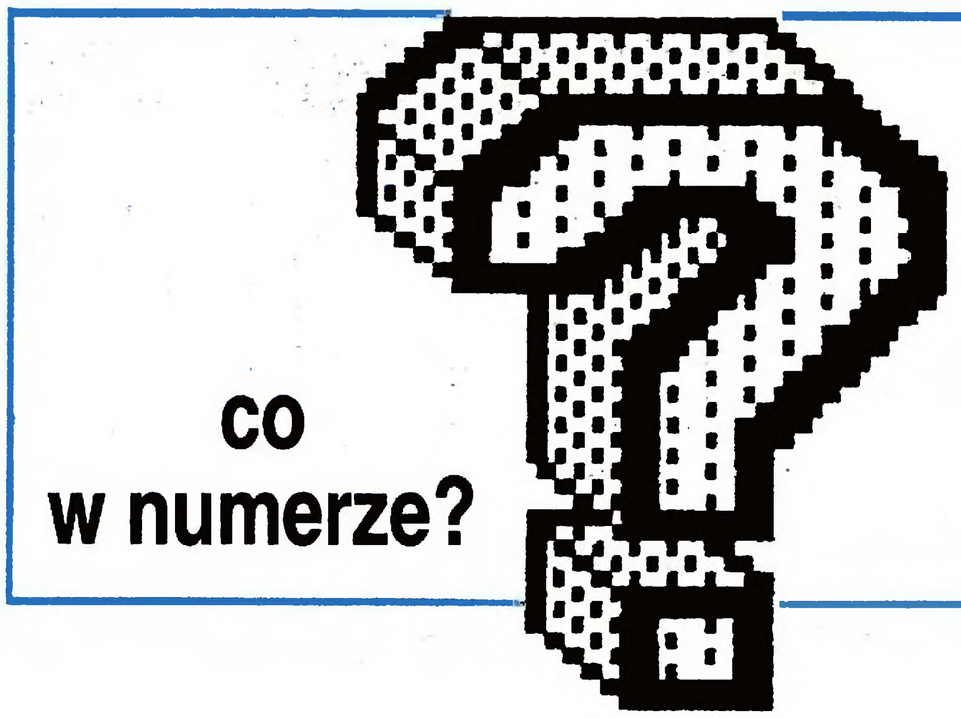
Nasz magazyn redagowany jest przez ludzi młodych. Jesteśmy uczniami szkół podstawowych i średnich, studentami, pracownikami naukowymi, dziennikarzami. Widzimy świat jakim mógłby być i pytamy — Dlaczego nie!

Redakcja

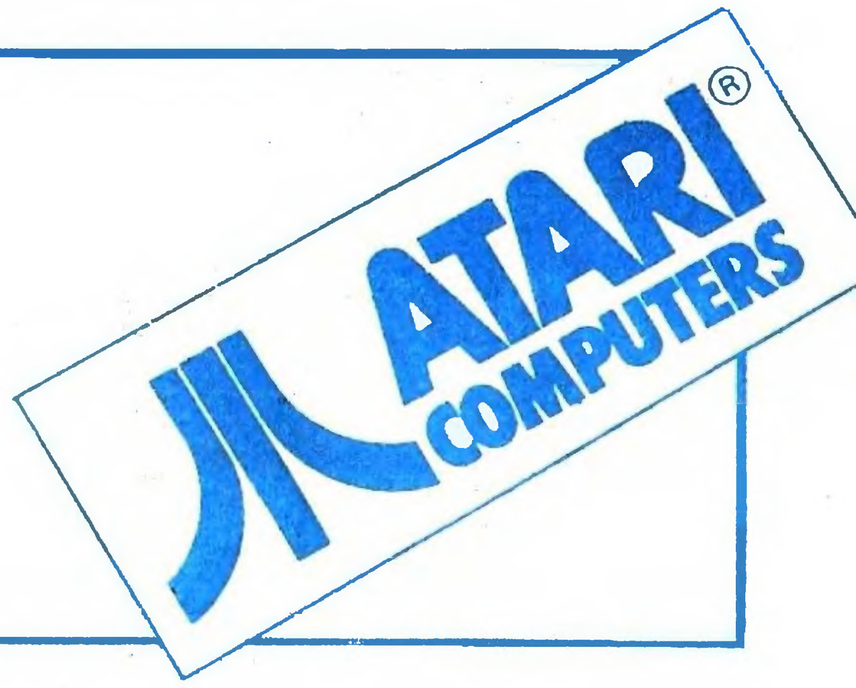
MOJE Atari

MAGAZYN UŻYTKOWNIKÓW KOMPUTERÓW ATARI

Redaktor naczelny:
Wojciech Zientara
Zespół redakcyjny:
Marek Czarkowski
Tomasz Wiśniewski
Grafik:
Wanda Roszkowska
Zdjęcia:
Leopold Dzikowski
Wydawca:
Spółdzielnia „Bajtek"
00-687 Warszawa
ul. Wspólna 61
tel. 21-12-05
Fotoskład:
Grażyna Kurzątkowska
Montaż offsetowy:
Grażyna Ostaszewska
Korekta:
Maria Krajewska
Teresa Rutkowska
Nr. zlecenia: 61510
Nakład: 105 tys.



co w numerze?



PROGRAMOWANIE

GRY

PROGRAMY UŻYTKOWE

SPRZĘT

SZKÓŁKA

INSTRUKCJE

DZIAŁ ST

INNE

PIĘĆ PYTAŃ

Dziś gościem „Bajtka" jest Joanna Baranowska — osoba odpowiedzialna w serwisie systemów mikrokomputerowych Atari za obieg sprzętu na linii klient-serwis. Osoba, która „z urzędu" skazana jest na uszczypliwe uwagi, na odpowiedzi na tysiące pytań nie zawsze mądrych, na prowadzenie konwersacji nawet wtedy, gdy nie ma na to ochoty.

— Czy rzeczywiście Twoja praca jest tak niewdzięczna jak mogłoby się wydawać po moim wstępie? Widziałem też klientów, którzy przychodzili do Ciebie z kwiatami?

— Tak, to prawda, zdarzają się czasami bardzo miłe momenty, ale każda praca polegająca na kontaktach z ludźmi wymaga dużo cierpliwości, wyrozumiałości i uprzejmości — kiedy ma się chandrę lub „zły dzień" przychodzi to bardzo trudno. Na szczęście większość klientów przychodzących do serwisu jest sympatyczna i uśmiechnięta. Chętnie z nimi rozmawiam — nie sprawia mi to trudności, bo z natury jestem „gadułą".

— Uważam, że w Polsce jest mnóstwo dziewczyn, które marzą o takiej pracy jak Twoja. Mam na myśli zarobki, kontakt z interesującymi ludźmi, styczność z nowoczesnym sprzętem jakim jest komputer Atari. Co Ty na to?

— Jeśli chodzi o zarobki nie muszę niczego dodawać. Lepiej więc ograniczę się do odpowiedzi na część pytania dotyczącą, jak to określasz, „interesujących ludzi". Przeważnie nie mam czasu na wykorzystanie takiej okazji i dłuższą rozmowę, muszę pracować tak, aby nasi klienci nie czekali zbyt długo w kolejce. Natomiast bardzo jestem zadowolona, że dzięki pracy w P.Z. „Karen" poznałam komputer Atari. Inaczej chyba nigdy by do tego nie doszło. Jestem humanistką i wcześniej czułam zdecydowaną niechęć do wszystkich „cudów techniki", a w szczególności urządzeń elektronicznych. Teraz nie wyobrażam sobie, aby współczesny człowiek nie miał pojęcia o tym, czym jest komputer. Oczywiście najlepiej znam Atarl, które „na bieżąco" wyręcza mnie w wykonywaniu „papierkowej roboty".

— Czy zdarzyło się w czasie Twojej trzyletniej pracy w serwisie Atari coś szczególnego, coś co będziesz pamiętać przez całe życie?

— Na pewno zawsze będę pamiętać moment, w którym mój szef po raz pierwszy pokazał mi komputer Atari i bez żadnego wstępu zaproponował prowadzenie przy jego pomocy dokumentacji serwisu. Ja po prostu bałam się dotknąć klawiatury. Dzisiaj z uśmiechem wspominam tamten dzień. Teraz to wszystko jest takie normalne...

— Jaka jest Twoja opinia na temat „małego Atari", bo choć niektórzy z klientów nie chcą w to uwierzyć, rzeczywiście potrafisz postawić trafną diagnozę w wielu przypadkach.

— „Małe Atari" zawładnęło polskim rynkiem komputerowym. Nie stało się tak bez przyczyny. Sprzęt jest bardzo dobry pod względem rozwiązań technicznych. Posiada bogate możliwości audiowizualne, jest łatwy w obsłudze i ma niezliczoną już chyba ilość oprogramowania. Jeżeli doda się do tego ofertę usług świadczonych przez naszą firmę takich jak rozszerzenie pamięci, montaż interpreterów języków oprogramowania, instalację polskich liter itp. to „małe Atari" zadowoli chyba najbardziej wybrednych użytkowników. Gdyby tak jeszcze każdy mógł zastosować w swoim zestawie stację dysków zamiast „wrażliwego" magnetofonu miałby sprzęt wręcz idealny.

— Co trzeba zrobić i jakich użyć argumentów, żeby nie czekać na naprawę dwa tygodnie?

— Nie ma na to sposobu. Taki termin naprawy wynika z kilku przyczyn. Obsługujemy cały kraj, pracownia elektroniczna jest niewielka, samego sprzętu „u ludzi" jest dużo ponad 100 000. Co najmniej 20 procent oddanych do naprawy komputerów jest sprawnych. Przetestowanie takiego sprzętu zabiera jednak cenny czas elektronikowi. Maksymalny dwutygodniowy termin naprawy stosujemy z konieczności, a nie z braku dobrych chęci. Oczywiście zdarzają się sytuacje, w których robimy wyjątki. Ostatnio był to ojciec dziecka z nogą w gipsie. To jednak dezorganizuje naszą pracę: uwzględniamy więc tylko naprawdę wyjątkowe sytuacje.

— Dziękując za rozmowę życzę jak najmniejszej ilości pracy.

rozmawiał
Sergiusz Piotrowski

EDYTOR BASIC-a

Przy przepisywaniu z „Bajtka" programów w Basicu nie sposób ustrzec się błędów.

Aby uniknąć żmudnego wyszukiwania popełnianych omyłek wszystkie programy w Basicu są publikowane wraz z kodami kontrolnymi. Zamieszczony obok program służy do kontroli tych kodów podczas przepisywania programu.

Wydruk „Edytora Basica" należy dokładnie przepisać i zapisać na kasecie lub dyskietce (najlepiej instrukcją LIST „C:" lub LIST „D:EDYTOR.LST"). Poprawność przepisania można sprawdzić samym „Edytorem" w opisany niżej sposób.

Przystępując do wpisywania dowolnego programu z naszego pisma trzeba najpierw wczytać i uruchomić „Edytor Basica". Następnie należy przepisywać kolejne wiersze programu. Po wpisaniu każdego wiersza i naciśnięciu klawisza RETURN wiersz ten pojawia się w dowolnej części ekranu wraz z obliczonym kodem kontrolnym. Jeżeli wyświetlony kod jest taki sam, jak wydrukowany przed numerem wiersza, można przystąpić do wpisywania następnego wiersza. Jeśli kody są różne, to ponowne naciśnięcie RETURN wyświetla wpisany wiersz w górnej części ekranu i umożliwia dokonanie w nim niezbędnych poprawek.

Samo naciśnięcie RETURN wywołuje zawsze ostatnio wpisany wiersz. W celu wywołania innego, wcześniej napisanego wiersza trzeba podać jego numer poprzedzony gwiazdką (np. 1000) i nacisnąć RETURN. Wpisanie samej liczby powoduje wymazanie z pamięci komputera wiersza programu o takim numerze.

Po poprawnym przepisaniu całego programu trzeba przerwać pracę „Edytora Basica" przez naciśnięcie klawisza BREAK lub RESET. Następnie w celu usunięcia „Edytora" i zbędnych zmiennych (z tablic nazw i wartości zmiennych) zapisujemy program na kasecie instrukcją LIST „C:",0,31999 lub na dyskietce instrukcją LIST „D:nazwa",0,31999. Teraz kasujemy zawartość pamięci komputera instrukcją NEW i odczytujemy program przy pomocy ENTER „C:" (z kasety) lub ENTER „D:nazwa" z dyskietki. Po wykonaniu tych czynności w pamięci komputera znajduje się tylko gotowy program bez „Edytora Basica" i można go już ostatecznie zapisać na odpowiedni nośnik.

Procedura ta jest może nieco kłopotliwa, lecz zabezpiecza w stu procentach przed popełnieniem omyłki przy przepisywaniu programu z „Bajtka". PAMIĘTAJ: ZAWSZE UŻYWAJ „EDYTORA BASICA".

```
NC  32000 REM EDYTOR BASICA
SE  32010 REM wersja 1.1 dla "Bajtka"
JB  32020 CLR :DIM LINIA$(120):CLOSE #2:CL
    OSE #3
MN  32030 OPEN #2,4,0,"E:":OPEN #3,5,0,"E:
    "
ZR  32040 ? CHR$(125):POSITION 11,1:? "■ED
    YTOR BASICA■"
YB  32050 TRAP 32040:POSITION 2,3:? "Wpisz
     linie programu"
KH  32060 POSITION 1,4:? " ":INPUT #2;LINI
    4$:IF LINIA$="" THEN POSITION 2,4:LIST
     B:GOTO 32060
XL  32070 IF LINIA$(1,1)="*" THEN B=VAL(LI
    NIA$(2,LEN(LINIA$))):POSITION 2,4:LIST
     B:GOTO 32060
TH  32080 POSITION 2,10:? "CONT"
BI  32090 B=VAL(LINIA$):POSITION 1,3:? " "
    ;
NY  32100 POKE 842,13:STOP
CN  32110 POKE 842,12
KI  32120 ? CHR$(125):POSITION 11,1:? "■ED
    YTOR BASICA■":POSITION 2,15:LIST B
BZ  32130 C=0:ODP=C
LL  32140 POSITION 2,16:INPUT #3;LINIA$:IF
     LINIA$="" THEN ? "LINIA ";B;" USUNIET
    A":GOTO 32050
GU  32150 FOR D=1 TO LEN(LINIA$):C=C+1:ODP
    =ODP+(C*ASC(LINIA$(D,D))):NEXT D
SR  32160 KOD=INT(ODP/676)
BX  32170 KOD=ODP-KOD*676
BM  32180 KODS=INT(KOD/26)
VF  32190 KODM=KOD-(KODS*26)+193
NF  32200 KODS=KODS+193
UW  32210 POSITION 0,16:? CHR$(KODS);CHR$(
    KODM)
HN  32220 POSITION 2,13:? "Jezeli kod sie
    nie zgadza, nacisnij   ■RETURN■ i popr
    aw linie.":GOTO 32050
```

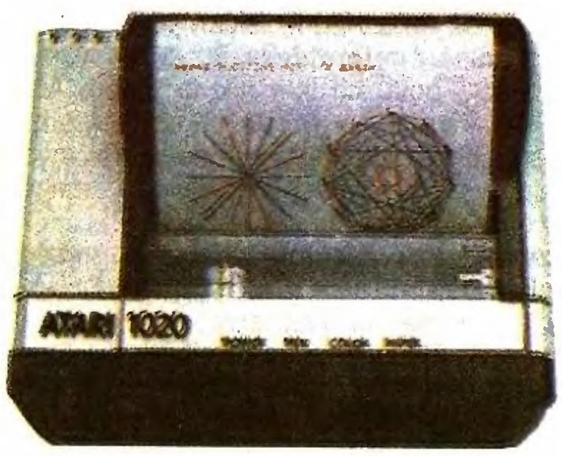
SPRZĘT rodzina ATARI (3)

W trzecim i ostatnim odcinku poświęconym rodzinie 8-bitowych Atari prezentujemy pozostałe peryferia przeznaczone dla tych komputerów: drukarki, interfejsy itd.

DRUKARKI

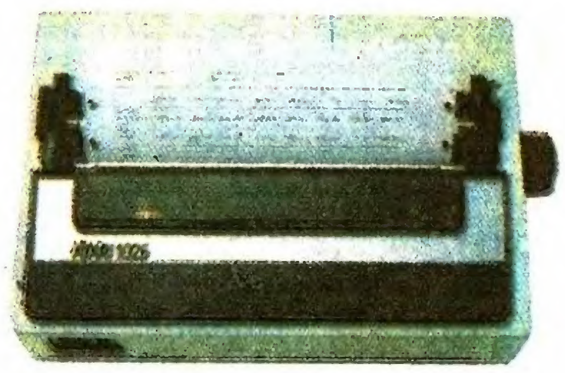
Aby drukarka mogła współpracować bezpośrednio z komputerem Atari, musi posiadać odpowiednie złącze. Takie wyposażenie mają tylko drukarki produkowane przez Atari i nieliczne modele innych firm (Seikosha i Epson). Pozostałe drukarki pracują poprzez interfejsy Centronics lub RS232.

ATARI 1020



Jest to drukarka graficzna, czyli tak zwany printerplotter. Drukuje w 40 kolumnach (10 znaków na cal), z prędkością 10 znaków na sekundę. Możliwe jest drukowanie tekstu zarówno w poziomie, jak i w pionie, a właściwe kreślenie odbywa się przy pomocy czterech specjalnych różnokolorowych pisaków. Bardzo łatwe tworzenie grafiki, oraz wbudowane zestawy znaków: standardowy i międzynarodowy. Papier tylko z wąskiej rolki. Bardzo tania, lecz w polskich warunkach kłopotliwa w eksploatacji (specjalny papier i specjalne pisaki).

ATARI 1025



Drukarka mozaikowa stanowiąca rozwinięcie wcześniejszego modelu Atari 825 (dla 400/800) i przystosowana wzorniczo do serii XL. Drukuje w 80 kolumnach z prędkością 40 znaków na sekundę. Znaki z matrycy 7 na 9 punktów mogą być drukowane w trzech szerokościach: 5 (enhanced), 10 (pica) i 16 (compressed) znaków na cal. Do wyboru są dwa różne odstępy między wierszami (6 lub 8 wierszy na cal). Cenną zaletą (w Polsce) jest korzystanie ze zwykłej taśmy do maszyn do pisania. Drukować można na papierze perforowanym, z rolki lub na pojedynczych arkuszach.

ATARI 1027

Drukarka znakowa (czcionkowa) drukująca w 80 kolumnach z prędkością 20 znaków na sekundę wyłącznie w trybie korespondencyjnym (NLQ). Dostępna tylko jedna szerokość pisma — 12 znaków na cal (elite). Ponieważ znaki drukowane są przy pomocy głowicy (podobnie jak w maszynach do pisania IBM), to niemożliwe jest uzyskanie grafiki, a polskie litery można otrzymać jedynie przez zmianę głowicy. Druk jest wykonywany tuszem znajdującym się w specjalnym pojemniku. Można używać papier perforowany, z rolki lub w arkuszach.

ATARI 1029

Kolejna drukarka mozaikowa, stanowiąca zaadaptowaną dla potrzeb serii XL wersję drukarki Seikosha GP500. Bardzo popularna w Polsce, gdyż jest sprzedawana przez Pewex. Drukuje w 80 kolumnach z prędkością 50 znaków na sekundę. Znaki z matrycy 5 na 7 punktów można drukować w dwóch szerokościach (5 lub 10 znaków na cal) i w dwóch odstępach (6 lub 9 wierszy na cal). Drukarka posiada dwa wbudowane zestawy znaków: standardowy i międzynarodowy. Ten ostatni w egzemplarzach sprzedawanych w Pewexie jest zastąpiony zestawem polskim. Ponadto możliwe jest uzyskanie grafiki o rozdzielności 60 punktów na cal (480 na wiersz). Druk odbywa się z taśmy w kasecie i na papierze perforowanym, z rolki lub w arkuszach. Obecnie jest to jednak sprzęt z epoki ZX-81.

ATARI XMM801 I XMM804

Nowe drukarki mozaikowe przystosowane zarówno do komputerów serii XE, jak i ST. Druk w 80 kolumnach z prędkością 80 znaków na sekundę. Pozwalają również na zastosowanie wszystkich trzech form papieru.

ATARI XDM121

Drukarka znakowa dla komputerów XE i ST. Druk przy pomocy głowicy: 80 kolumn, 12 znaków na sekundę. Pracuje wyłącznie w trybie korespondencyjnym. Można używać papier w dowolnej formie.

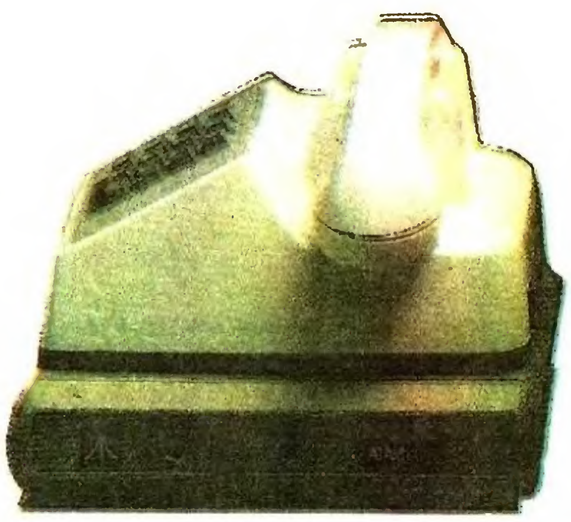
MODEMY

Modem jest urządzeniem służącym do przyłączenia komputera do sieci telefonicznej. Na Zachodzie jest to jedno z najbardziej popularnych urządzeń peryferyjnych. Niestety w Polsce poza brakiem banków danych, dodatkowe utrudnienia stwarza konserwatywna postawa Ministerstwa Komunikacji, Transportu i Łączności.

ATARI 830

Modem akustyczny dla komputerów Atari 400/800 pracujący z prędkością transmisji 300 bodów. Pracuje w systemie duplex lub half-duplex. Umożliwia transmisję plików ASCII. Do jego pracy konieczny jest interfejs Atari 850.

ATARI 835



Modem bezpośredni (direct connect) dla komputerów 400/800 do transmisji z prędkością 300 lub 1200 bodów. Pracuje w systemie duplex lub half-duplex i umożliwia przesyłanie plików ASCII.

ATARI 1030

Modem bezpośredni dla komputerów serii XL pracujący z prędkością 300 bodów. Transmisja w systemie duplex lub half-duplex. Możliwe jest przesyłanie zarówno plików ASCII, jak ATASCII. Posiada wbudowaną procedurę obsługi i dzięki temu nie wymaga żadnej pamięci zewnętrznej.

ATARI XM301

Modem bezpośredni dla komputerów serii XE (działa także z XL poza 1200XL) pracujący z prędkością 300 bodów. Transmisja w systemie duplex lub half-duplex według protokołu XMODEM. Przesyłanie plików ASCII i ATASCII. Zasilany jest z komputera i nie posiada własnego zasilacza. Procedura obsługi wczytywana jest z dyskietki. Jego odmiana oznaczona jako XM301P jest jedynym modemem dla komputerów XL/XE posiadającym homologację w Polsce.

SUPRA 300-AT

Modem bezpośredni dla komputerów AL/XE pracujący z prędkością 300 bodów. Transmisja w systemie duplex lub half-duplex według protokołu XMODEM. Przesyłanie plików ASCII.

HAYES SMARTMODEM 1200

Modem bezpośredni dla komputerów XL/XE/ST pracujący z prędkością 300 lub 1200 bodów. Transmisja w systemie duplex lub half-duplex z automatycznym rozpoznawaniem prędkości. Przesyłanie plików ASCII. Do jego pracy z komputerami XL i XE konieczny jest interfejs Atari 850.

INTERFEJSY

Interfejsy są to urządzenia przystosowujące do siebie różne inne urządzenia, a dokładniej przetwarzające sygnały wysyłane przez jedno z nich na postać zrozumiałą dla drugiego. W świetle tego, co zostało napisane poprzednio konieczność stosowania interfejsów nie powinna budzić wątpliwości.

ATARI 850

Zespół czterech interfejsów szeregowych RS232C i jednego równoległego Centronics. Umożliwia transmisję z prędkością od 75 do 9600 bodów w systemie duplex lub half-duplex z jednym lub dwoma bitami stopu. Poprzez jedno ze złącz RS232 pozwala na połączenie z dowolnym komputerem posiadającym takie złącze, np. Atari ST, IBM PC i in. Nie produkowany już od kilku lat.

P:R:CONNECTION

Zespół dwóch interfejsów szeregowych RS232C i jednego równoległego Centronics. Pozostałe dane jak Atari 850. Posiada wbudowaną procedurę obsługi oraz zasilanie z komputera (brak oddzielnego zasilacza). Dodatkowo na dołączonej dyskietce znajdują się programy transmisji XMODEM, AMODEM i TSCOPE.

MICROPRINT

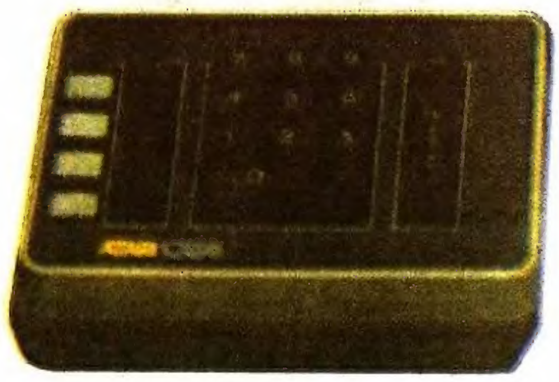
Interfejs równoległy Centronics do drukarek. Jedyny interfejs dostępny w Pewexie. Jego dokładny opis był zamieszczony w drugim wydaniu specjalnym „Tylko o Atari".

INNE

MICRONET

Sieć dla Atari XL/XE przewidziana dla szkół oraz klubów i laboratoriów komputerowych. Pozwala na przyłączenie ośmiu komputerów do jednego zestawu urządzeń peryferyjnych — stacja dysków, drukarka, modem itd.

ATARI CX85



Dodatkowa klawiatura numeryczna wyposażona w 10 klawiszy cyfrowych, 4 klawisze przesuwające kursor oraz klawisze "-," "." i "+/ENTER". Procedura obsługi klawiatury jest wczytywana z dyskietki.

ZOBIAN RAT

Mysz dla ośmiobitowych komputerów Atari. Na dołączonej do niej dyskietce RAT PACK znajduje się m.in. procedura obsługi, programy przykładowe i program graficzny (rysunkowy).

ERRATA

W opisie stacji dysków Atari XF551 w poprzednim odcinku „Rodziny Atari" znalazł się błąd. Stacja ta posiada dwie głowice i przy użyciu specjalnego DOS-u (np. SuperDOS) pozwala na zapis dwustronny bez odwracania dyskietki.

Wojciech Zientara

ZACZYNAMY PROGRAMOWAĆ

Po zapoznaniu się z klawiaturą i edytorem komputera możemy już przystąpić do nauki programowania. Tylko w jakim języku? Jeden ze znanych profesorów powiedział: „Najważniejszym językiem programowania nie jest Basic, Pascal czy Fortran, lecz ANGIELSKI. Będziemy więc programować w języku angielskim.

Przecież komputer nie zna angielskiego! Nic nie szkodzi. Po pierwsze wszystkie języki programowania oparte są na angielskim słownictwie. Po drugie — będziemy programować, więc język jest obojętny. To ostatnie stwierdzenie na pewno dziwi. Wynika to z rozpowszechnionego poglądu, że programowanie polega na pisaniu programu w jakimś języku komputerowym. Nic bardziej błędnego. Sztuka programowania polega bowiem na umiejętności ułożenia algorytmu, czyli sposobu rozwiązania zadania. W jakim języku ten algorytm zostanie zakodowany, to sprawa drugorzędna. Właśnie kodowanie jest zapisywaniem programu w konkretnym języku programowania.

Początkowo artykuł ten miał być zatytułowany „Wstęp do Basica". W porę jednak przypomniałem sobie, że języków programowania jest bez liku i nie można od początku ograniczać się do jednego z nich. Poza tym wiele mówi się — i nie bez racji — o wadach Basica i wyrabianiu przez niego złych nawyków. Jest tu jednak pewna przeszkoda. Basic jest wbudowany do komputera, natomiast inne języki trzeba wczytywać. Nie jest to żadnym problemem dla posiadaczy stacji dysków, ale użytkowanicy magnetofonów (czyli ponad 80% naszych Czytelników) mają na ten temat odmienne zdanie. Niezbędny jest jakiś kompromis. Będziemy układać algorytmy w oderwaniu od języka, zaś przykłady będą pisane w różnych językach, lecz przeważnie w Basicu.

ALGORYTM

Najtrudniejszym etapem programowania jest poprawne postawienie zadania. Należy najpierw dokładnie określić, co chcemy otrzymać, czyli mówiąc fachowo — jakie będą dane wyjściowe. Następnie ustalamy, jakie informacje (czyli dane wejściowe) są potrzebne do uzyskania pożądanego wyniku. Teraz dopiero można przystąpić do układania algorytmu.

Algorytm jest to dokładnie opisany sposób postępowania, który prowadzi do osiągnięcia założonego celu. Prostym przykładem algorytmu jest przepis na jakąś potrawę. Podaje on potrzebne składniki, kolejność czynności i rezultat końcowy. Tak więc książka kucharska jest po prostu zbiorem algorytmów. Algorytmy komputerowe opierają się na identycznych zasadach, choć wyrażonych w nieco innej formie.

Spróbujemy teraz postawić proste zadanie i ułożyć algorytm jego rozwiązania. Będzie to proste obliczenie, gdyż przykłady z matematyki najlepiej nadają się do celów pokazowych. Oto zadanie: obliczyć pierwszych szesnaście potęg całkowitych liczby 2. No i jest pierwszy kłopot. Co to znaczy „pierwszych". Czy od 1 do 16 czy od 0 do 15, a może od −8 do 8. Nie ma to wprawdzie żadnego znaczenia dla algorytmu, lecz uzyskane wyniki będą zupełnie inne. Ponadto nie wiadomo, co zrobić z obliczonymi potęgami. Chcę w ten sposób pokazać, jak ważne jest precyzyjne postawienie zadania. W skrajnym przypadku programista może ułożyć program, który będzie wykonywał coś zupełnie różnego od oczekiwań użytkownika (klienta). Powtórzmy więc: obliczyć i wypisać na ekranie kolejne potęgi całkowite liczby 2 od 0 do 15.

Następnym krokiem jest ustalenie danych wyjściowych. Będą to oczywiście obliczone potęgi, ale czy tylko? Kolumna cyfr może i ma jakieś walory artystyczne, lecz poza tym nic więcej nie mówi. Trzeba więc dodać jakąś informację. W tym przypadku najlepiej po prostu działanie. Zgodnie z wcześniejszym opisem określamy teraz niezbędne dane wejściowe. Tu nie ma problemu, gdyż zostały one podane w zadaniu. Są to: liczba potęgowana (2), dolna (0) i górna (15) granica potęgowania oraz krok potęgowania (1 — gdyż wykładnikami mają być kolejne liczby całkowite).

Czas wreszcie na algorytm. Piszemy:

1. oblicz 2 do potęgi 0
2. wypisz działanie i wynik
3. oblicz 2 do potęgi 1
4. wypisz działanie i wynik
5. oblicz 2 do potęgi 2
6. wypisz działanie i wynik
7. oblicz 2 do potęgi 3
8. wypisz działanie i wynik
9. .....

Bez sensu! Przecież to jest tyle pisania, że lepiej obliczyć tę potęgi ręcznie na kartce. Ale co zrobić ze skomplikowanymi działaniami? Jak to co? Uprościć! Oczywiście nie działanie, lecz zapis. Piszemy znów:

1. Wynik = 2^0
2. pisz WYNIK
3. WYNIK = 2^1
4. pisz WYNIK
5. WYNIK = 2^2
6. pisz WYNIK
7. WYNIK = 2^3
8. pisz WYNIK
9. .....

Trochę lepiej, ale jeszcze coś nie tak — pisania jest nadal dużo. A gdy trzeba obliczyć 1000 potęg, to co? Tym razem problem leży gdzie indziej. Zapis jest dobry, ale algorytm zły. Przecież te czynności się powtarzają, więc można wypisać raz i pokazać, że trzeba je wykonać kilka razy. Wprowadzamy nowe zmiany i piszemy:

1. WYKŁADNIK = 0
2. WYNIK = $2^{\text{wykładnik}}$
3. pisz WYNIK
4. zwiększ WYKŁADNIK o 1
5. czy WYKŁADNIK > 15?
6. jeśli nie, to przejdź do 2
7. jeśli tak, to koniec

Teraz nareszcie jest dobrze. Na pewno? Jeśli będziemy mieli bardziej skomplikowany algorytm, to nawet drobna zmiana zadania (np. obliczanie potęg liczby 3) spowoduje konieczność przejrzenia całego algorytmu. Aby tego uniknąć, użyte wartości umieścimy na samym początku i otrzymamy taki rezultat:

1. LICZBA = 2
2. WYKŁADNIK = 0
3. KONIEC = 15
4. KROK = 1
5. WYNIK = LICZBA $^{\text{wykładnik}}$
6. pisz WYNIK
7. zwiększ WYKŁADNIK o KROK
8. czy WYKŁADNIK > KONIEC?
9. jeśli nie, to przejdź do 5
10. jest tak, to koniec

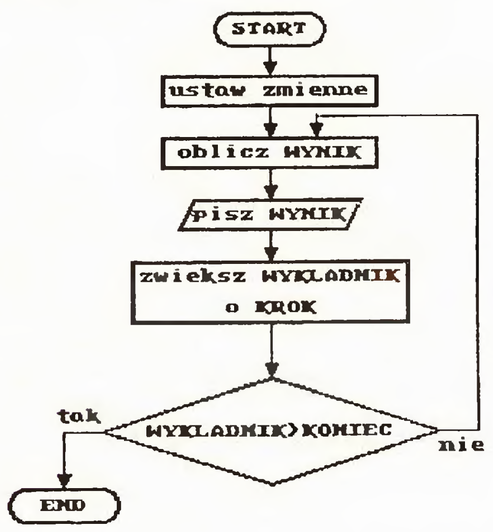
Mamy już poprawny algorytm, ale ma on jeszcze jedną wadę: jest mało przejrzysty. Dla większych algorytmów będzie to znacznie lepiej zauważalne (to znaczy będzie widać mniej). Wiadomo, że najłatwiej operuje się rysunkami. Przedstawimy więc algorytm w postaci graficznej — jako schemat blokowy.

SCHEMAT BLOKOWY

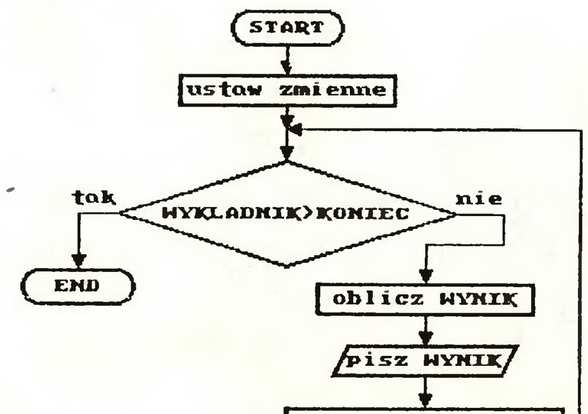
W celu graficznego przedstawienia naszego algorytmu musimy najpierw ustalić pewne symbole, które będziemy stosować w schematach.
Trzeba więc pogrupować czynności wykonywane przez komputer według jakiegoś systemu i każdej grupie przypisać symbol graficzny.

Najpowszechniej przyjęty jest podział czynności na cztery grupy: operacje wewnętrzne, operacje wejścia/wyjścia, decyzje i wywołania podprogramów (procedur). Ta ostatnia grupa jest używana przy bardziej rozbudowanych programach, w których pewne fragmenty powtarzają się i nie ma sensu ich wielokrotne programowanie. Decyzje są operacjami sprawdzenia warunków i podjęcia dalszego działania w zależności od rezultatu tego sprawdzenia (w naszym ostatnim algorytmie są to punkty 8—10). Operacje wejścia/wyjścia służą do transmisji do i z urządzeń zewnętrznych (np. "pisz WYNIK"). Wszystkie pozostałe operacje należą do grupy operacji wewnętrznych.

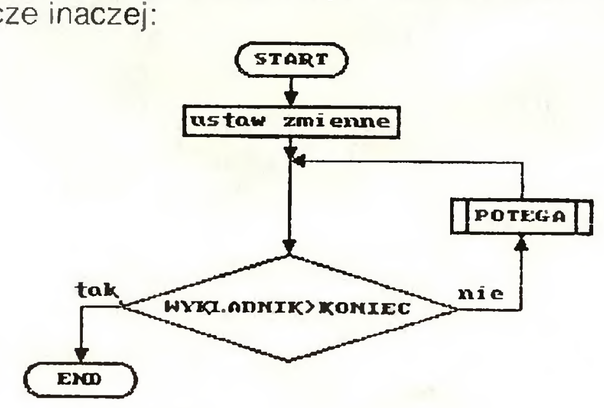
Poza symbolami operacji niezbędne są jeszcze symbole dodatkowe, które zwiększają czytelność schematu. Jeden z nich oznacza początek lub koniec algorytmu, zaś drugi jest łącznikiem, który pozwala łączyć fragmenty schematu leżące na różnych stronach lub w dużej odległości od siebie na tej samej stronie. Zestawienie wszystkich symboli znajduje się na końcu, a poniżej narysujemy nasz algorytm w postaci graficznej:



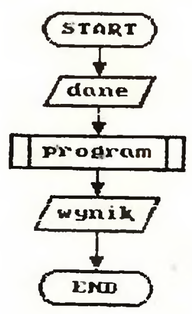
Teraz dopiero widać wyraźnie, co i jak ten algorytm wykonuje. Nie jest to jeszcze koniec, ponieważ nasz algorytm nie jest zupełnie poprawny. To znaczy, działa i nawet da prawidłowe rezultaty, ale w niektórych (a nawet w większości) języków programowania wystąpią pewne kłopoty z jego zakodowaniem. Dlaczego? Ponieważ nie jest on STRUKTURALNY. To straszne słowo oznacza po prostu, że nie można algorytmu podzielić na mniejsze części. Powiecie, że nie ma tu co dzielić. A właśnie, że jest — można zupełnie oddzielić obliczanie potęgi. Po tej zmianie algorytm wygląda następująco:



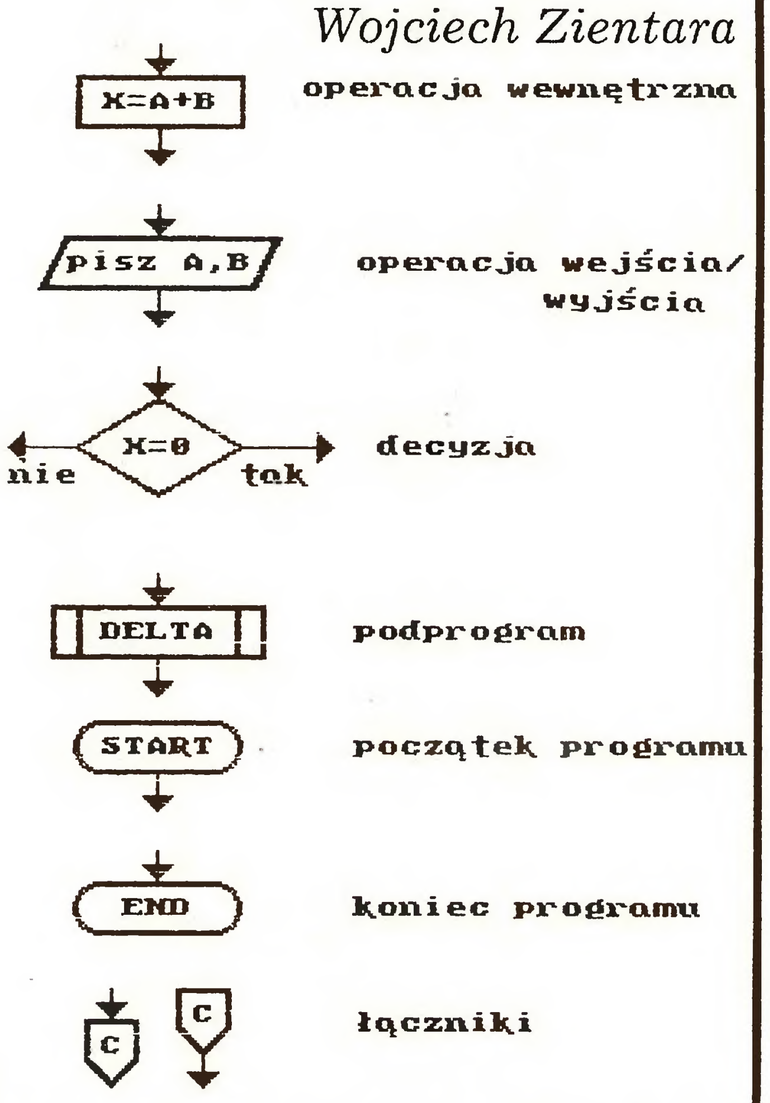
Co to za różnica? W pierwszym schemacie wszystkie czynności uszeregowane są kolejno, zaś w drugim można wydzielić z niego osobną STRUKTURĘ obliczania potęgi. Takie struktury nazywamy podprogramami lub procedurami. Sztuka poprawnego programowania polega przede wszystkim na umiejętnym budowaniu algorytmu z gotowych procedur. Można bowiem narysować nasz schemat jeszcze inaczej:



To wygląda już bajecznie prosto. Od tego schematu powinniśmy zacząć układanie algorytmu i następnym razem na pewno tak zrobimy. A na zakończenie schemat algorytmu, od którego należy rozpoczynać układanie każdego programu:



Wojciech Zientara



Symbole stosowane na schematach blokowych algorytmów

GRA — W LINIE —

Pomysł nie jest nowy: na ekranie telewizora rysowane są dwie linie, których ruchy sterowane są przez graczy, oczywiście za pośrednictwem joysticków.

Należy jak najdłużej unikać zderzenia „swojej" linii z polem już zamalowanym, czy to przez siebie, czy też przez przeciwnika, jak również z obramowaniem pola gry. Do tej pory nie spotkałem się jednak z realizacją tego programu dla Atari — być może programiści uważali ten problem za nazbyt prosty?

Program istotnie nie jest skomplikowany, a co najważniejsze niezbyt długi. Do uruchomienia gry wystarczy oczywiście program ładujący napisany w Basicu, jednak dla tych, których interesuje nie tylko kolejna możliwość sprawdzenia wytrzymałości własnego joysticka, zamieszczony jest również program źródłowy, napisany w asemblerze MAC/65.

Gra w linie jest mimo swej pozornej prostoty dosyć wciągająca, ale (na szczęście dla joysticków), aby w nią grać potrzeba jest aż dwóch chętnych. Gdy się znudzi, nie trzeba koniecznie wciskać RESET, grę przerywa również klawisz ESC, jednak tylko w czasie rysowania lini.

Po tych krótkich objaśnieniach można już przepisać zamieszczony program i — po zapisaniu na taśmie lub dyskietce rozpocząć grę. Program startuje od adresu 16384, po przerwaniu można go uruchomić ponownie przez USR(16384).

Życzę przyjemniej zabawy (do tego przede wszystkim powstał ten komputer).

Andrzej Biazik

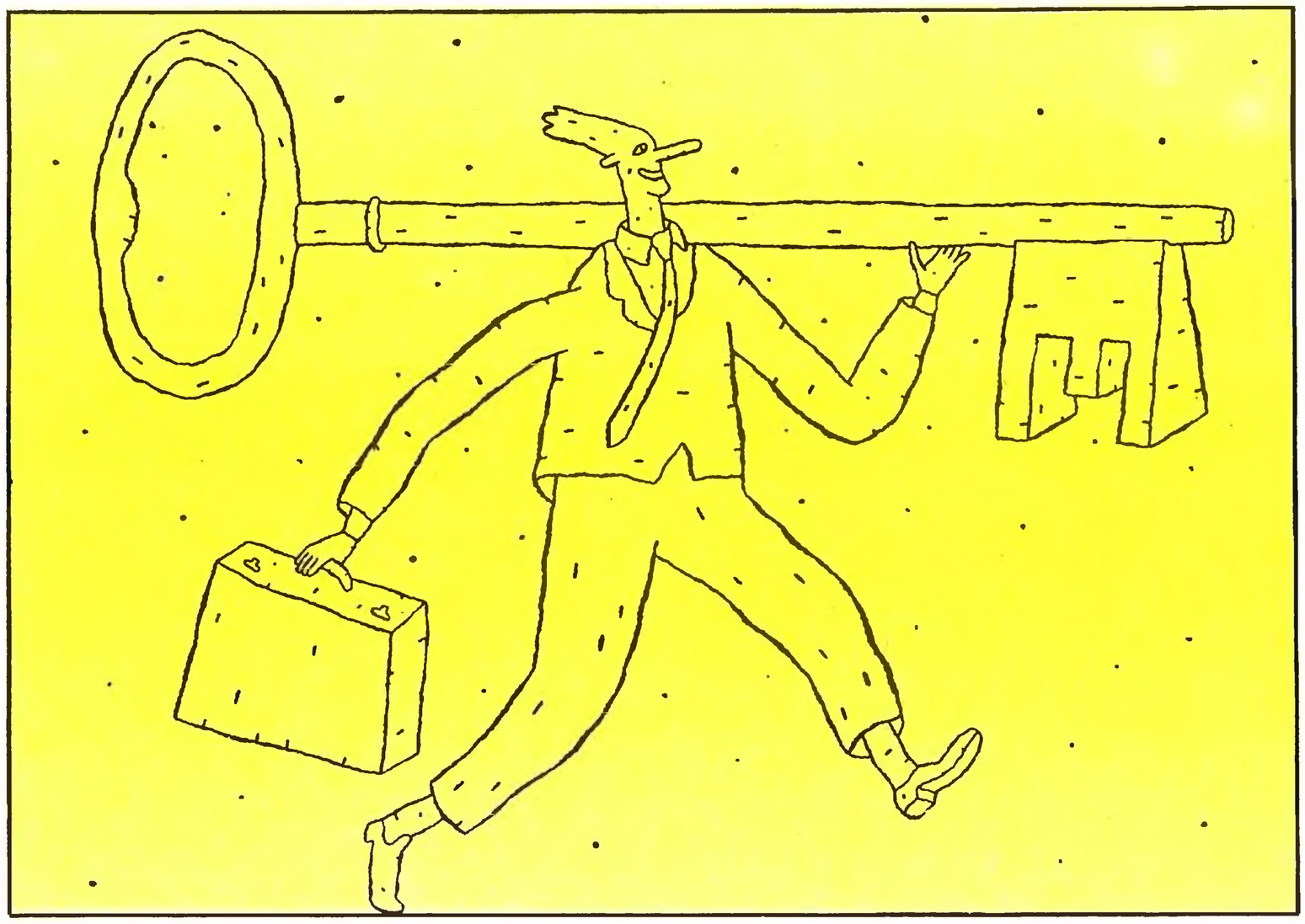
PROGRAM 1

```
SN   0 REM **********************
TN   1 REM *     L I N I E      *
KZ   2 REM * (c) 1988 A. Biazik *
SQ   3 REM **********************
NJ   4 REM
UL   10 A=16384:RESTORE 100
IK   20 FOR I=0 TO 68:S=0
XE   30 FOR J=0 TO 9:READ B:POKE A+J,B:S=S+
     B:NEXT J:READ SUMA:IF SUMA<>S THEN 50
ID   40 A=A+10:NEXT I:? "GOTOWE":END
EO   50 ? " BLAD DANYCH W LINII ";100+10*I:
     LIST 100+10*I:STOP
IE   100 DATA 104,169,26,141,48,2,169,64,14
     1,49,913
QE   110 DATA 2,169,7,133,87,169,0,133,88,1
     69,957
RX   120 DATA 145,133,89,76,158,64,112,112,
     112,77,1078
PP   130 DATA 0,145,13,13,13,13,13,13,13,13
     ,249
WM   140 DATA 13,13,13,13,13,13,13,13,13,13
     ,130
WO   150 DATA 13,13,13,13,13,13,13,13,13,13
     ,130
WQ   160 DATA 13,13,13,13,13,13,13,13,13,13
     ,130
WS   170 DATA 13,13,13,13,13,13,13,13,13,13
     ,130
WU   180 DATA 13,13,13,13,13,13,13,13,13,13
     ,130
WW   190 DATA 13,13,13,13,13,13,13,13,13,13
     ,130
WF   200 DATA 13,13,13,13,13,13,13,13,13,13
     ,130
YB   210 DATA 13,70,118,64,6,65,26,64,0,39,
     465
VE   220 DATA 50,33,35,58,0,17,0,0,0,39,232
YE   230 DATA 50,33,35,58,0,18,0,0,0,0,194
JD   240 DATA 0,0,0,16,0,0,0,0,0,0,16
CD   250 DATA 0,0,0,16,0,0,0,0,169,255,440
FB   260 DATA 141,252,2,169,0,141,1,210,32,
     32,980
NL   270 DATA 244,169,15,141,196,2,169,3,14
     1,197,1277
NJ   280 DATA 2,169,9,141,198,2,169,3,141,2
     51,1085
LJ   290 DATA 2,169,17,133,34,169,0,133,90,
     133,880
SR   300 DATA 91,133,92,133,84,133,86,169,1
     59,133,1213
IZ   310 DATA 85,32,175,249,169,79,133,84,3
     2,175,1213
NO   320 DATA 249,169,0,133,85,32,175,249,1
     69,0,1261
AC   330 DATA 133,84,32,175,249,169,1,141,2
     51,2,1237
VS   340 DATA 169,40,133,84,133,85,32,202,2
     41,169,1288
LS   350 DATA 2,141,251,2,169,40,133,84,169
     ,120,1111
FD   360 DATA 133,85,32,202,241,169,40,133,
     203,133,1371
QS   370 DATA 204,133,206,169,120,133,205,1
     69,8,133,1480
DR   380 DATA 207,169,4,133,208,169,0,133,0
     ,173,1196
TO   390 DATA 120,2,73,15,133,4,240,30,41,1
     2,670
CC   400 DATA 240,14,133,4,165,207,41,3,240
     ,18,1065
YY   410 DATA 165,4,133,207,208,12,165,207,
     41,12,1154
DF   420 DATA 240,6,165,4,41,3,133,207,173,
     121,1093
ZS   430 DATA 2,73,15,133,4,240,30,41,12,24
     0,790
FK   440 DATA 14,133,4,165,208,41,3,240,18,
     165,991
FL   450 DATA 4,133,208,208,12,165,208,41,1
     2,240,1231
GQ   460 DATA 6,165,4,41,3,133,208,165,207,
     41,973
OF   470 DATA 1,240,2,198,204,165,207,41,2,
     240,1300
PT   480 DATA 2,230,204,165,207,41,4,240,2,
     198,1293
EF   490 DATA 203,165,207,41,8,240,2,230,20
     3,165,1464
MF   500 DATA 208,41,1,240,2,198,206,165,20
     8,41,1310
DF   510 DATA 2,240,2,230,206,165,208,41,4,
     240,1338
SX   520 DATA 2,198,205,165,208,41,8,240,2,
     230,1299
ZN   530 DATA 205,165,203,197,205,208,21,16
     5,204,197,1770
KL   540 DATA 206,208,15,32,193,65,76,158,6
     4,173,1190
DS   550 DATA 132,2,45,133,2,208,248,96,165
     ,203,1234
TQ   560 DATA 133,85,165,204,133,84,32,128,
     241,240,1445
DC   570 DATA 3,76,94,66,165,205,133,85,165
     ,206,1198
VQ   580 DATA 133,84,32,128,241,240,3,76,71
     ,66,1074
HA   590 DATA 165,203,133,85,165,204,133,84
     ,169,1,1342
EU   600 DATA 141,251,2,32,202,241,165,205,
     133,85,1457
VF   610 DATA 165,206,133,84,169,2,141,251,
     2,32,1185
FZ   620 DATA 202,241,173,252,2,201,28,208,
     3,76,1386
GK   630 DATA 170,194,169,255,141,252,2,169
     ,163,141,1656
QW   640 DATA 1,210,165,0,73,255,141,0,210,
     162,1217
OH   650 DATA 10,164,0,234,234,234,234,234,
     234,234,1812
OU   660 DATA 234,200,208,245,202,16,240,16
     5,0,201,1711
TG   670 DATA 200,240,2,230,0,169,160,141,1
     ,210,1353
LX   680 DATA 76,33,65,32,131,249,169,17,13
     3,0,905
VA   690 DATA 173,143,64,201,25,240,32,238,
     143,64,1323
XS   700 DATA 32,193,65,76,158,64,32,131,24
     9,169,1169
ZL   710 DATA 18,133,0,173,153,64,201,25,24
     0,9,1016
NW   720 DATA 238,153,64,32,193,65,76,158,6
     4,169,1212
OI   730 DATA 16,141,143,64,141,153,64,169,
     156,141,1188
BR   740 DATA 112,64,169,66,141,113,64,165,
     0,141,1035
ZZ   750 DATA 171,66,32,193,65,169,118,141,
     112,64,1131
MW   760 DATA 169,64,141,113,64,76,158,64,0
     ,0,849
YD   770 DATA 55,57,39,50,33,44,0,39,50,33,
     400
CO   780 DATA 35,58,0,0,0,0,0,0,0,0,93
```

PROGRAM 2

```
0100 ;**********************
0110 ;*       LINIE        *
0120 ;* (c) 1988 A.Biazik *
0130 ;**********************
0140 ;
```

```
0150      *=  16384    ;adres startowy
0160 ;
0170 ; ustawienie adresow zmiennych
0180 ; i etykiet
0190 ;
0200 CLEAR = 62496
0210 KLICK = 63875
0220 DRAWTO = 63919
0230 PLOT =  61898
0240 LOC =   61824
0250 COLOR = 763
0260 JOY0 =  632
0270 JOY1 =  633
0280 TRIG0 = 644
0290 TRIG1 = 645
0300 DLAD =  560
0310 X0   =  203
0320 Y0   =  204
0330 X1   =  205
0340 Y1   =  206
0350 K0   =  207
0360 K1   =  208
0370 XN   =  85
0380 YN   =  84
0390 XS   =  91
0400 YS   =  90
0410 AUDFR = 53760
0420 AUDCN = 53761
0430 GRTR =  87
0440 EKRAN = 88
0450 CL0 =   708
0460 CMD =   34
0470 ILPET = 0
0480 PAM =   4
0490 KEY =   764
0500 RESET = 49834
0510 ATRACT = 77
0520 ;
0530 ; poczatek programu
0540 ;
0550      PLA
0560      LDA # <DLIST ;ustawienie
0570      STA DLAD     ;wartosci ekra-
0580      LDA # >DLIST ;nowych: adres
0590      STA DLAD+1   ;Display List,
0600      LDA #7       ;tryb grafiki
0610      STA GRTR     ;oraz poczatek
0620      LDA #0       ;pamieci ekranu
0630      STA EKRAN
0640      LDA #145
0650      STA EKRAN+1
0660      JMP POC
0670 ;
0680 ;zapis Display List (GR.7
0690 ;ze zmienionym oknem tekstowym)
0700 ;oraz pamieci obrazu okna
0710 ;
0720 DLIST .BYTE 112,112,112
0730      .BYTE 77,0,145
0740      .BYTE 13,13,13,13
0750      .BYTE 13,13,13,13,13
0760      .BYTE 13,13,13,13,13
0770      .BYTE 13,13,13,13,13
0780      .BYTE 13,13,13,13,13
0790      .BYTE 13,13,13,13,13
0800      .BYTE 13,13,13,13,13
0810      .BYTE 13,13,13,13,13
0820      .BYTE 13,13,13,13,13
0830      .BYTE 13,13,13,13,13
0840      .BYTE 13,13,13,13,13
0850      .BYTE 13,13,13,13,13
0860      .BYTE 13,13,13,13,13
0870      .BYTE 13,13,13,13,13
0880      .BYTE 13,13,13,13,13
0890      .BYTE 13,13,13,13,13
0900      .BYTE 70
0910 ZMN .WORD TE1
0920      .BYTE 6,65
0930      .WORD DLIST
0940 TE1 .SBYTE " GRACZ 1"
0950      .SBYTE "   GRACZ 2"
0960      .SBYTE "         "
0970 STO .SBYTE "0          "
0980 ST1 .SBYTE "0    "
0990 POC LDA #255       ;procedura
1000      STA KEY       ;inicjujaca:
1010      LDA #0        ;ustawianie
1020      STA AUDCN     ;wartosci
1030      STA ATRACT    ;poczatkowych
1040      JSR CLEAR     ;czyszczenie
1050      LDA #15       ;ekranu
1060      STA CL0
1070      LDA #3
1080      STA CL0+1     ;ustalenie
1090      LDA #9        ;kolorow
1100      STA CL0+2
1110      LDA #3
1120      STA COLOR
1130      LDA #17
1140      STA CMD       ;rysowanie
1150      LDA #0        ;ramki
1160      STA YS
1170      STA XS
1180      STA XS+1
1190      STA YN
1200      STA XN+1
1210      LDA #159
1220      STA XN
1230      JSR DRAWTO
1240      LDA #79
1250      STA YN
1260      JSR DRAWTO
1270      LDA #0
1280      STA XN
1290      JSR DRAWTO
1300      LDA #0
1310      STA YN
1320      JSR DRAWTO
1330      LDA #1
1340      STA COLOR
1350      LDA #40
1360      STA YN
1370      STA XN
1380      JSR PLOT      ;zapalenie
1390      LDA #2        ;punktow
1400      STA COLOR     ;startowych
1410      LDA #40
1420      STA YN
1430      LDA #120
1440      STA XN
1450      JSR PLOT
1460      LDA #40       ;nadanie
1470      STA X0        ;wartosci
1480      STA Y0        ;poczatkowych
1490      STA Y1        ;zmiennym
1500      LDA #120      ;opisujacym
1510      STA X1        ;polozenie
1520      LDA #8        ;i kierunek
1530      STA K0        ;ruchu
1540      LDA #4        ;punktow
1550      STA K1
1560      LDA #0        ;ilosc pow-
1570      STA ILPET     ;torzen petli
1580 DAL  LDA JOY0      ;opoznaijacej
1590      EOR #15
1600      STA PAM       ;sprawdzanie
1610      BEQ SK2       ;wartosci
1620      AND #12       ;joystickow
1630      BEQ SK1       ;oraz ewntualna
1640      STA PAM       ;korekcja
1650      LDA K0        ;aktualnych
1660      AND #3        ;kierunkow
1670      BEQ SK2
1680      LDA PAM
1690      STA K0
1700      BNE SK2
1710 SK1  LDA K0
1720      AND #12
1730      BEQ SK2
1740      LDA PAM
1750      AND #3
1760      STA K0
1770 SK2  LDA JOY1
1780      EOR #15
1790      STA PAM
1800      BEQ SK4
1810      AND #12
1820      BEQ SK3
1830      STA PAM
1840      LDA K1
1850      AND #3
1860      BEQ SK4
1870      LDA PAM
1880      STA K1
1890      BNE SK4
1900 SK3  LDA K1
1910      AND #12
1920      BEQ SK4
1930      LDA PAM
1940      AND #3
1950      STA K1
1960 SK4  LDA K0        ;zaleznie
1970      AND #1        ;od wartosci
1980      BEQ SK5       ;zmiennych
1990      DEC Y0        ;opisujacych
2000 SK5  LDA K0        ;kierunek
2010      AND #2        ;nastepuje
2020      BEQ SK6       ;modyfikacja
2030      INC Y0        ;pozycji
2040 SK6  LDA K0
2050      AND #4
2060      BEQ SK7
2070      DEC X0
2080 SK7  LDA K0
2090      AND #8
2100      BEQ SK8
2110      INC X0
2120 SK8  LDA K1
2130      AND #1
2140      BEQ SK9
2150      DEC Y1
2160 SK9  LDA K1
2170      AND #2
2180      BEQ SKA
2190      INC Y1
2200 SKA  LDA K1
2210      AND #4
2220      BEQ SKB
2230      DEC X1
2240 SKB  LDA K1
2250      AND #8
2260      BEQ SKC
2270      INC X1
2280 SKC  LDA X0
2290      CMP X1        ;sprawdzenie,
2300      BNE ET1       ;czy pozycje
2310      LDA Y0        ;obu punktow
2320      CMP Y1        ;(po modyfi-
2330      BNE ET1       ;kacji)
2340      JSR POW       ;sa jednakowe
2350      JMP POC
2360 POW  LDA TRIG0     ;petla oczekujaca
2370      AND TRIG1     ;na wcisniecie
2380      BNE POW       ;FIRE w ktoryms
2390      RTS           ;z joystickow
2400 ET1  LDA X0
2410      STA XN        ;sprawdzenie,
2420      LDA Y0        ;czy pierwsza
2430      STA YN        ;linia zderzyla
2440      JSR LOC       ;sie z polem juz
2450      BEQ ET2       ;zamalowanym
2460      JMP SP1
2470 ET2  LDA X1
2480      STA XN        ;jak wyzej - dla
2490      LDA Y1        ;drugiej linii
2500      STA YN
2510      JSR LOC
2520      BEQ ET3
2530      JMP SP0
2540 ET3  LDA X0        ;narysowanie
2550      STA XN        ;kolejnego punktu
2560      LDA Y0        ;pierwszej linii
2570      STA YN
2580      LDA #1
2590      STA COLOR
2600      JSR PLOT
2610      LDA X1        ;a takze drugiej
2620      STA XN
2630      LDA Y1
2640      STA YN
2650      LDA #2
2660      STA COLOR
2670      JSR PLOT
2680      LDA KEY       ;sprawdzenie
2690      CMP #28       ;klawisza <ESC>
2700      BNE HOP
2710      JMP RESET
2720 HOP  LDA #255
2730      STA KEY
2740      LDA #163
2750      STA AUDCN
2760      LDA ILPET
2770      EOR #255
2780      STA AUDFR
2790      LDX #10
2800 PEZ  LDY ILPET     ;petla
2810 PEW  NOP           ;opozniajaca
2820      NOP
2830      NOP
2840      NOP
2850      NOP
2860      NOP
2870      NOP
2880      NOP
2890      INY
2900      BNE PEW
2910      DEX
2920      BPL PEZ
2930      LDA ILPET     ;ewentualne
2940      CMP #200      ;zmniejszenie
2950      BEQ ET4       ;ilosci powtorzen
2960      INC ILPET     ;petli
2970 ET4  LDA #160
2980      STA AUDCN
2990      JMP DAL
3000 SP0  JSR KLICK     ;procedury
3010      LDA #17       ;obslugujace
3020      STA ILPET     ;zderzenie sie
3030      LDA ST0       ;ktorejs linii
3040      CMP #25       ;z juz zamalowa-
3050      BEQ KON       ;nym polem
3060      INC ST0
3070      JSR POW
3080      JMP POC
3090 SP1  JSR KLICK
3100      LDA #18
3110      STA ILPET
3120      LDA ST1
3130      CMP #25
3140      BEQ KON
3150      INC ST1
3160      JSR POW
3170      JMP POC
3180 KON  LDA #16       ;zakonczenie
3190      STA ST0       ;rozgrywki:
3200      STA ST1       ;informacja
3210      LDA # <TE2    ;o zwyciestwie
3220      STA ZMN       ;ktoregos
3230      LDA # >TE2    ;z graczy
3240      STA ZMN+1
3250      LDA ILPET
3260      STA WYG
3270      JSR POW
3280      LDA # <TE1
3290      STA ZMN
3300      LDA # >TE1
3310      STA ZMN+1
3320      JMP POC
3330 TE2  .SBYTE "  WYGRAL GRACZ "
3340 WYG  .SBYTE "               "
```



CHARS DESIGNER

CHARS DESIGNER to program użytkowy przeznaczony do współpracy z komputerami Atari XL/XE, napisany w języku Atari Basic.

Program umożliwia bardzo łatwe i ciekawe projektowanie znaków graficznych na matrycy 8 na 8 joystickiem i zastępowanie nimi znaków standardowych. Przedefiniować można znaki o kodach ATASCII 0 do 127, oprócz kodów 27 do 31 125 do 127. Znaki w trybie odwróconym (kody powyżej 127) przedefiniowywane są oczywiście jednocześnie ze swoim „pozytywem". Nowy, zdefiniowany przez użytkownika generator znaków można zapisać na nośniku magnetycznym (kaseta, dysk) i wgrywać specjalnym podprogramem „ChD's Generators Loader" dołączonym do dowolnego programu.

Dla ułatwienia pracy program posiada bogate menu obejmujące:

1) Wyświetlanie tabeli znaków i kodów standardowego zestawu.
2) Wyświetlanie tabeli znaków już przedefiniowanych.
3) Projektowanie nowego znaku.
4) Przywracanie znaku standardowego.
5) Projektowanie polskich liter.

Projektowanie polskich liter wyposażone jest w dwie opcje:

— projektowanie dowolnych liter polskich o dowolnych kodach (więcej pracy);
— zdefiniowanie pełnego zestawu polskich liter o kodach standardowych.

Zdefiniowany standard odbiega tylko nieznacznie od propozycji K. Leskiego (patrz „Bajtek" nr 2 i 3 z roku 1987), a jest chyba łatwiejszy do zapamiętania!

6) Zapis zdefiniowanego generatora znaków na taśmę lub dysk.

Szczegółowe informacje jak posługiwać się programem podawane są w trakcie pracy. Dodać należy, że po zakończeniu pracy, program informuje użytkownika o sposobie wywołania zdefiniowanych znaków, a następnie jest wymazywany z pamięci. Zdefiniowany przez siebie zestaw znaków można używać do chwili wyłączenia komputera z sieci (lub „zimnego startu"). W szczególności można go używać także po wczytaniu innego programu, o ile ten nie ingeruje w obszar pamięci z nowym zestawem znaków. Zdecydowanie lepszym rozwiązaniem jest skorzystanie z możliwości zapisu zdefiniowanego przez siebie generatora znaków na taśmie lub dysku.

W trybie GRAPHICS 1 lub 2 zdefiniowane znaki o kodach 0 do 26 i 96 do 127 uzyskujemy przez POKE 756,X+2, pozostałe normalnie: POKE 756,X. Oprócz wydruku programu zamieszczony jest wydruk podprogramu „ChD's Generators Loader".

Andrzej–Piotr Szyndrowski

LISTING 1

```
VJ   0 REM ***** CHARS DESIGNER 0.2 *****
XJ   1 REM Copyright by A.Szyndrowski' 88
WP   2 REM ******************************
RV   10 CLR :POKE 566,158:TRAP 20000
PC   15 GOSUB 100:REM  INICJALIZACJA
IM   20 GOSUB 300:REM  MENU GLOWNE
MK   25 GOSUB 10000
PJ   30 KEY=KEY-48
NF   35 IF KEY=1 THEN 1000
NZ   40 IF KEY=2 THEN 2000
PM   45 IF KEY=3 THEN 3000
QG   50 IF KEY=4 THEN 4000
RT   55 IF KEY=5 THEN 5000
SN   60 IF KEY=6 THEN 6000
UA   65 IF KEY=7 THEN 7000
TV   70 GOTO 25
SK   100 REM ******* INICJALIZACJA *******
TG   105 DIM MAT(8,8),RCH(128),S$(36),GEN$(
     1024),NAM$(15)
BL   110 S$(1,36)="a.c'e.l/n'o's'z.z'A.C'E.
     L/N'O'S'Z.Z'"
FE   115 FOR I=1 TO 128
ZY   120 IF I>27 AND I<33 OR I>125 THEN RCH
     (I)=2:GOTO 130
MS   125 RCH(I)=0
FX   130 NEXT I
DA   135 POKE 106,PEEK(106)-10:GRAPHICS 0
LA   140 ADRES=(PEEK(106)+2)
HV   145 IF ADRES<>INT(ADRES/4)*4 THEN ADRE
     S=ADRES+1:GOTO 145
RT   150 ADRES=ADRES*256
BR   155 ? CHR$(125)
CK   160 POKE 710,0:SETCOLOR 1,1,15:POKE 75
     2,1
HY   165 POSITION 2,0
LR   170 ? "  *** C H A R S  D E S I G N E R
     ***"
OP   175 POSITION 11,2
NL   180 ? "Polish Version 2.0"
QR   185 POSITION 2,23
TP   190 ? "  (c) Copyright by A.Szyndrowski
     '88 ";
LE   195 FOR I=0 TO 1023
MO   200 POKE ADRES+I,PEEK(57344+I)
GH   205 NEXT I
ZA   210 RETURN
HS   300 REM ******** MENU GLOWNE ********
BJ   305 ? CHR$(125)
JH   310 POSITION 5,1
TK   315 ? "  M E N U  P R O G R A M U :"
JP   320 POSITION 2,4
UZ   325 ? "1 - wyswietlenie tabeli znakow
     i kodow     standardowego zestawu,":?
VN   330 ? "2 - wyswietlenie zestawu znakow
      prze-      definiowanych,":?
GH   335 ? "3 - projektowanie nowego znaku,
     ":? :?
AV   340 ? "4 - przywrocenie znaku standard
     owego,":? :?
MD   345 ? "5 - projektowanie polskich lite
     r,":? :?
HT   350 ? "6 - zapis generatora znakow na
     tasme       lub dyskietke,":?
VH   355 ? "7 - zakonczenie programu."
ZL   360 RETURN
BQ   1000 REM **** ZESTAW STANDARDOWY ****
XB   1005 FOR I=0 TO 3
FY   1010 ? CHR$(125)
KE   1015 ? "      ZESTAW STANDARDOWY ZNAKOW
     "
XG   1020 POSITION 2,4
WI   1025 ? "znak":? "kod"
XA   1030 FOR J=0 TO 3
AU   1035 FOR K=0 TO 7
WG   1040 KOL=6+K*4:WRS=7+J*4
OB   1045 POSITION KOL,WRS
SI   1050 CHAR=I*32+J*8+K
VT   1055 IF RCH(CHAR+1)=2 THEN 1080
KK   1060 POSITION KOL,WRS:? CHR$(CHAR);CHR
     $(153);CHR$(160)
ZL   1065 POSITION KOL,WRS+1:? CHR$(21);CHR
     $(140);CHR$(160)
KN   1070 POSITION KOL,WRS+2:? CHR$(160);CH
     R$(160);CHR$(160)
JJ   1075 POSITION KOL,WRS+2:? CHAR
GJ   1080 NEXT K
GS   1085 NEXT J
GB   1090 POSITION 2,23
UO   1095 ? "Dalej ?.. (T/N)";
ZZ   1100 GOSUB 10000
KM   1105 IF KEY=84 OR KEY=116 THEN 1120
WP   1110 IF KEY=78 OR KEY=110 THEN POP :GO
     TO 20
NZ   1115 GOTO 1100
EX   1120 NEXT I
QF   1125 GOTO 20
NI   2000 REM ** ZESTAW PRZEDEFINIOWANY **
GQ   2005 ? CHR$(125)
SD   2010 ? "   ZESTAW ZNAKOW PRZEDEFINIOWA
     NYCH "
AO   2015 POSITION 2,8
ZM   2020 ? "  MENU:"
XL   2025 ? :? "1 - Wyswietlenie ekranu ze
     wszystkimi      znakami nowego zestawu.
     "
MU   2030 ? "      Znaki wyswietlone sa rzeda
     mi          wg wzrastajacych kodow."
PT   2035 ? "      Nowe znaki wyswietlone sa
     w trybie    odwroconym.":?
ZA   2040 ? "0 - Powrot do menu programu. O
     pcje te     nalezy wybrac teraz lub po
      wyswie-    tleniu nowych znakow."
BE   2045 GOSUB 10000
FP   2050 KEY=KEY-48
EQ   2055 IF KEY=2 THEN 20
SS   2060 IF KEY<>1 THEN 2045
HI   2065 ? CHR$(125)
JM   2070 POKE 756,ADRES/256
AN   2075 FOR I=0 TO 7
HA   2080 FOR J=0 TO 15
KT   2085 POSITION 5+2*J,6+2*I
UQ   2090 CHAR=I*16+J
TV   2095 IF RCH(CHAR+1)=0 THEN ? CHR$(CHAR
     )
WK   2100 IF RCH(CHAR+1)=1 THEN ? CHR$(CHAR
     +128)
FX   2105 NEXT J
EV   2110 NEXT I
AX   2115 GOSUB 10000
FI   2120 KEY=KEY-48
RV   2125 IF KEY<>2 THEN 2115
EE   2130 POKE 756,224
QJ   2135 GOTO 20
FL   3000 REM ** PROJEKT NOWEGO ZNAKU **
GR   3005 ? CHR$(125)
UX   3010 ? "     PROJEKTOWANIE NOWEGO ZNAK
     U "
AP   3015 POSITION 2,8
FR   3020 ? "Wcisnij znak do przedefiniowan
     ia!"
AZ   3025 GOSUB 10000
IU   3030 K2=KEY
ZP   3035 ? :? "Czy to ten znak?.. ";CHR$(K
     2+128);" (T/N)"
AL   3040 GOSUB 10000
FK   3045 IF KEY=78 OR KEY=110 THEN 3000
AP   3050 IF KEY<>84 AND KEY<>116 THEN 3040
ZR   3055 IF RCH(K2+1)=0 THEN 3085
UT   3060 ? :? "Znak ";CHR$(K2+128);" byl p
     rzedefiniowywany."
MU   3065 ? "Czy rezygnujesz z jego poprzed
     niego   wzoru? (T/N)"
AU   3070 GOSUB 10000
HD   3075 IF KEY=78 OR KEY=110 THEN 20
FF   3080 IF KEY<>84 AND KEY<>116 THEN 3070
TN   3085 ? CHR$(125):POSITION 2,3:? "  MENU
     U"
ZL   3090 ? :? "1 - Rozpoczecie projektowan
     ia znaku.       Do projektowania uzywa
     sie joy-"
IJ   3095 ? "      sticka, ktory steruje kurs
     orem.           Przycisk FIRE sluzy do prz
     edefi-"
BD   3100 ? "      niowywania pola, ktore wsk
     azuje           kursor."
FM   3105 ? :? "2 - Rezygnacja z wprowadzon
     ych przez       uzytkownika zmian i prz
     ejscie          do menu programu."
PY   3110 ? "      Opcje mozna wybrac teraz
                     lub w trakcie definiowania
     ."
XL   3115 ? :? "3 - Potwierdzenie dokonanyc
     h zmian         i przejscie do menu pro
     gramu."
IF   3120 ? "      Opcja dziala tylko w czasi
     e defi-         niowania znaku!"
BB   3125 GOSUB 10000
FM   3130 KEY=KEY-48
OX   3135 IF KEY=1 THEN 3150
DW   3140 IF KEY=2 THEN 20
SY   3145 GOTO 3125
GO   3150 ? CHR$(125)
GN   3155 POSITION 2,23
XR   3160 ? "  CHWILECZKE...  ";
VQ   3165 IF K2>=0 AND K2<32 THEN AD=(K2+64
     )*8
```

```
KL  3170 IF K2>=32 AND K2<96 THEN AD=(K2-3
    2)*8
WZ  3175 IF K2>=96 AND K2<128 THEN AD=K2*8
ZZ  3180 FOR I=0 TO 7
TN  3185 S=PEEK(AD+ADRES+I)
BP  3190 FOR J=1 TO 8
NI  3195 MAT(I+1,J)=S-INT(S/2)*2
LQ  3200 S=INT(S/2)
GA  3205 NEXT J
EY  3210 NEXT I
UX  3215 GRAPHICS 2+16
BO  3220 SETCOLOR 0,1,15:POKE 710,6
AD  3225 FOR I=0 TO 7
EH  3230 POSITION 7,2+I
YF  3235 FOR J=8 TO 1 STEP -1
GD  3240 IF MAT(I+1,J)=1 THEN ? #6;"O";
MP  3245 IF MAT(I+1,J)=0 THEN ? #6;"@";
FV  3250 NEXT J
GE  3255 NEXT I
EQ  3260 POSITION 6,2:? #6;"+"
CF  3265 IF STICK(0)<>7 THEN POSITION 6,2:
    ? #6;" ":GOTO 3260
VC  3270 POSITION 6,2:? #6;" ":PKW=2:PKK=7
HD  3275 POSITION PKK,PKW
RH  3280 ? #6;" "
OG  3285 IF STICK(0)=15 AND PEEK(764)=255
    AND STRIG(0)=1 THEN POSITION PKK,PKW:?
     #6;"+":GOTO 3275
PO  3290 IF STRIG(0)=0 THEN MAT(PKW-1,15-P
    KK)=ABS(MAT(PKW-1,15-PKK)-1):GOSUB 300
    00
LM  3295 IF PEEK(764)=30 THEN POKE 764,255
    :GRAPHICS 0:POKE 710,0:POKE 752,1:SETC
    OLOR 1,1,15:GOTO 20
YD  3300 IF PEEK(764)=26 THEN POKE 764,255
    :GOTO 3455
QC  3305 IF PEEK(764)<>255 THEN POKE 764,2
    55
QL  3310 IF STICK(0)=14 THEN GOSUB 3335:GO
    TO 3330
DX  3315 IF STICK(0)=7 THEN GOSUB 3365:GOT
    O 3330
XR  3320 IF STICK(0)=13 THEN GOSUB 3395:GO
    TO 3330
PC  3325 IF STICK(0)=11 THEN GOSUB 3425:GO
    TO 3330
VE  3330 GOTO 3275
GT  3335 POSITION PKK,PKW
QM  3340 IF MAT(PKW-1,15-PKK)=1 THEN ? #6;
    "O"
FD  3345 IF MAT(PKW-1,15-PKK)=0 THEN ? #6;
    "@"
DQ  3350 PKW=PKW-1
YS  3355 IF PKW<2 THEN PKW=9
AZ  3360 RETURN
HC  3365 POSITION PKK,PKW
QV  3370 IF MAT(PKW-1,15-PKK)=1 THEN ? #6;
    "O"
FM  3375 IF MAT(PKW-1,15-PKK)=0 THEN ? #6;
    "@"
TT  3380 PKK=PKK+1
LV  3385 IF PKK>14 THEN PKK=7
BI  3390 RETURN
HL  3395 POSITION PKK,PKW
QC  3400 IF MAT(PKW-1,15-PKK)=1 THEN ? #6;
    "O"
ET  3405 IF MAT(PKW-1,15-PKK)=0 THEN ? #6;
    "@"
CG  3410 PKW=PKW+1
WH  3415 IF PKW>9 THEN PKW=2
AP  3420 RETURN
GS  3425 POSITION PKK,PKW
QL  3430 IF MAT(PKW-1,15-PKK)=1 THEN ? #6;
    "O"
FC  3435 IF MAT(PKW-1,15-PKK)=0 THEN ? #6;
    "@"
UJ  3440 PKK=PKK-1
CO  3445 IF PKK<7 THEN PKK=14
AY  3450 RETURN
IQ  3455 GRAPHICS 0:POKE 752,1:POKE 710,0:
    ? CHR$(125)
ED  3460 SETCOLOR 1,1,15:POSITION 2,23
FZ  3465 ? "  PRZEDEFINIOWYWANIE ZNAKU...  "
    ;
BF  3470 FOR I=1 TO 8
YZ  3475 S=0
AP  3480 FOR J=0 TO 7
BU  3485 S=S+2^J*MAT(I,J+1)
GL  3490 NEXT J
ML  3495 POKE AD+ADRES+I-1,S
FB  3500 NEXT I
UQ  3505 RCH(K2+1)=1
PS  3510 GOTO 20
JK  4000 REM ***** ZNAK STANDARDOWY *****
GS  4005 ? CHR$(125)
XZ  4010 ? "  PRZYWROCENIE ZNAKU STANDARD
    OWEGO  "
YR  4015 POSITION 2,5
PB  4020 ? "Wcisnij znak, ktoremu chcesz p
    rzywro- cic standardowa postac!"
IA  4025 GOSUB 10000:K2=KEY
QO  4030 IF RCH(K2+1)=1 THEN 4040
WY  4035 ? :? "Znak ";CHR$(K2+128);" nie j
    est przedefiniowany!":GOTO 4090
UR  4040 ? :? "Czy to ten znak ";CHR$(K2+1
    28);" (T/N)?"
BG  4045 GOSUB 10000
WH  4050 IF KEY<>84 AND KEY<>116 THEN 4000
VM  4055 IF K2>=0 AND K2<32 THEN AD=(K2+64
    )*8
KH  4060 IF K2>=32 AND K2<96 THEN AD=(K2-3
    2)*8
WV  4065 IF K2>=96 AND K2<128 THEN AD=K2*8
ZV  4070 FOR I=0 TO 7
JM  4075 POKE AD+ADRES+I,PEEK(57344+AD+I)
FQ  4080 NEXT I
SV  4085 ? :? "Znak ";CHR$(K2+128);" przyw
    rocony!":RCH(K2+1)=0
LZ  4090 POSITION 2,15:? "  MENU:  "
TX  4095 ? :? "1 - powrot do menu programu
    ."
FM  4100 ? :? "2 - przywracanie nastepnego
     znaku,"
AW  4105 GOSUB 10000
FH  4110 KEY=KEY-48
DV  4115 IF KEY=1 THEN 20
KH  4120 IF KEY=2 THEN 4000
SE  4125 GOTO 4105
WM  5000 REM ** PROJEKT POLSKICH LITER **
GT  5005 ? CHR$(125)
JQ  5010 ? "  PROJEKTOWANIE POLSKICH LIT
    ER  "
AR  5015 POSITION 2,8
ZP  5020 ? "  MENU:  "
XT  5025 ? :? "1 - pelny zestaw polskich l
    iter o ko-      dach standardowych,"
DO  5030 ? :? "2 - dowolny zestaw polskich
     liter          o kodach wybieranych pr
    zez uzyt-       kownika,"
UE  5035 ? :? "3 - powrot do menu programu
    ."
AN  5040 GOSUB 10000
GJ  5045 KEY=KEY-48
LR  5050 IF KEY=1 THEN 5100
OO  5055 IF KEY=2 THEN 5300
EP  5060 IF KEY=3 THEN 20
RW  5065 GOTO 5040
GB  5100 ? CHR$(125)
SI  5105 ? "  STANDRDOWY ZESTAW POLSKICH
    LITER  "
IR  5110 ? :? "litery male:"
UM  5115 POSITION 24,4:? "n' -  14, CTRL+N
    ";
AO  5120 ? "a, -   1, CTRL+A      o' -  15
    , CTRL+O";
UW  5125 ? "c' -   3, CTRL+C      s' -  19
    , CTRL+S";
AH  5130 ? "e, -   5, CTRL+E      z. -  26
    , CTRL+Z";
ZB  5135 ? "l/ -  12, CTRL+L      z' -  24
    , CTRL+X"
NL  5140 ? "litery duze:"
JN  5145 POSITION 24,11:? "N' -  95,SHIFT+
    =";
NY  5150 ? "A, -  37,SHIFT+5      O' -  92
    ,SHIFT+-";
UY  5155 ? "C' -  38,SHIFT+6      S' -  94
    ,SHIFT+*";
VA  5160 ? "E, -  39,SHIFT+7      Z. -  93
    ,SHIFT+.";
WT  5165 ? "L/ -  64,SHIFT+8      Z' -  91
    ,SHIFT+,":?
WI  5170 ? "Jesli ktorys z proponowanych k
    odow    byl wczesniej przedefiniowany,
     zosta- nie zdefiniowany na nowo!!!"
DM  5175 ? :? "Czy zgadzasz sie na ten zes
    taw? (T/N)";
BB  5180 GOSUB 10000
YW  5185 IF KEY<>84 AND KEY<>116 THEN 5000
HC  5190 ? CHR$(125)
HB  5195 POSITION 2,23
VU  5200 ? "  PRZEDEFINIOWYWANIE ZNAKOW...
    ";
SB  5205 RESTORE 5265
HM  5210 FOR I=0 TO 17
YL  5215 READ CHAR
GK  5220 IF CHAR>=0 AND CHAR<32 THEN AD=(C
    HAR+64)*8
NY  5225 IF CHAR>=32 AND CHAR<96 THEN AD=(
    CHAR-32)*8
LZ  5230 RCH(CHAR+1)=1
AS  5235 FOR J=0 TO 7
QS  5240 READ B
OP  5245 POKE AD+ADRES+J,B
FX  5250 NEXT J
GG  5255 NEXT I
QD  5260 GOTO 20
UK  5265 DATA 1,0,0,60,6,62,102,62,12
HW  5266 DATA 3,12,24,60,96,96,96,60,0
GS  5267 DATA 5,0,0,60,102,126,96,62,12
JM  5268 DATA 12,0,56,24,60,24,24,60,0
WG  5269 DATA 14,12,24,124,102,102,102,102
    ,0
IU  5270 DATA 15,12,24,60,102,102,102,60,0
IB  5271 DATA 19,12,24,62,96,60,6,124,0
OH  5272 DATA 26,24,24,126,12,24,48,126,0
LV  5273 DATA 24,12,24,126,12,24,48,126,0
LJ  5274 DATA 37,0,24,60,102,102,126,102,1
    2
QU  5275 DATA 38,24,60,102,96,96,102,60,0
SD  5276 DATA 39,0,126,96,124,96,100,126,4
LW  5277 DATA 64,0,96,108,120,112,96,126,0
QF  5278 DATA 95,24,86,102,118,126,110,102
    ,0
FJ  5279 DATA 92,24,60,102,102,102,102,60,
    0
JE  5280 DATA 94,24,60,96,60,6,6,60,0
ZR  5281 DATA 93,24,126,12,24,48,96,126,0
NC  5282 DATA 91,24,126,22,12,24,48,126,0
HL  5300 FOR I=0 TO 17
GZ  5305 ? CHR$(125)
KF  5310 ? "  WYBOR POLSKICH LITER I ICH
    KODOW  "
YY  5315 POSITION 2,5
IF  5320 ? "LITERA: ";S$(2*I+1,2*I+2)
BA  5325 POSITION 2,8
ZY  5330 ? "  MENU:  "
VU  5335 ? :? "1 - rezygnujesz z tej liter
    y,"
ZL  5340 ? :? "2 - definiujesz te litere,"
UN  5345 ? :? "3 - powrot do menu programu
    ."
AW  5350 GOSUB 10000
GS  5355 KEY=KEY-48
WR  5360 IF KEY=3 THEN POP :GOTO 20
AK  5365 IF KEY=1 THEN 5485
UY  5370 IF KEY<>2 THEN 5350
YB  5375 RESTORE 5265+I
UE  5380 ? :? "Wcisnij znak do zastapienia
     litera!"
IZ  5385 GOSUB 10000:K2=KEY
FJ  5390 ? "Czy to ten znak... ";CHR$(K2+1
    28);" ? (T/N)"
CC  5395 GOSUB 10000
YW  5400 IF KEY<>84 AND KEY<>116 THEN I=I-
    1:GOTO 5485
VT  5405 IF RCH(K2+1)=0 THEN 5440
XP  5410 ? "Znak byl przedefiniowywany!"
PO  5415 ? "Czy rezygnujesz z jego poprzed
    niego   wzoru ?...  (T/N)"
AP  5420 GOSUB 10000
WL  5425 IF KEY=84 OR KEY=116 THEN 5440
DQ  5430 IF KEY=78 OR KEY=110 THEN I=I-1:G
    OTO 5485
SR  5435 GOTO 5420
XE  5440 READ CHAR
VS  5445 IF K2>=0 AND K2<32 THEN AD=(K2+64
    )*8
KN  5450 IF K2>=32 AND K2<96 THEN AD=(K2-3
    2)*8
XB  5455 IF K2>=96 AND K2<128 THEN AD=K2*8
AL  5460 FOR J=0 TO 7
RW  5465 READ B
OI  5470 POKE AD+ADRES+J,B
HB  5475 NEXT J
UU  5480 RCH(K2+1)=1
GT  5485 NEXT I
QQ  5490 GOTO 20
JQ  6000 REM ***** ZAPIS GENERATORA *****
GU  6005 ? CHR$(125)
VP  6010 ? "  ZAPIS GENERATORA ZNAKOW NA N
    OSNIK  "
SF  6015 TRAP 6900
YC  6020 POSITION 2,5
TF  6025 ? "Podaj nazwe zbioru!"
AU  6030 NAM$="                "
AY  6035 INPUT NAM$:IF NAM$(1,1)<>"C" AND
    NAM$(1,1)<>"D" THEN 6900
TI  6040 CLOSE #2:OPEN #2,8,0,NAM$
BK  6045 FOR I=0 TO 1023
WM  6050 PUT #2,PEEK(ADRES+I)
GD  6055 NEXT I
GQ  6060 POSITION 2,15
AW  6065 ? "  MENU:  "
UP  6070 ? :? "1 - informacje o sposobie u
    zywania         generatora z nosnika,"
UF  6075 ? :? "2 - powrot do menu programu
    ."
BA  6080 GOSUB 10000
GW  6085 KEY=KEY-48
EM  6090 IF KEY=2 THEN 20
WD  6095 IF KEY<>1 THEN 6080
GC  6100 ? CHR$(125)
CN  6105 ? "  INFORMACJE O GENERATORZE ZN
    AKOW  "
EG  6110 POSITION 2,5:? "Zapisany generato
    r znakow mozna wczy- tywac z nosnika p
    odprogramem o nazwie"
ON  6115 ? "    ** ChD's Generators Loader
    **."
US  6120 ? :? "Wywolanie podprogramu instr
    ukcja       GOSUB 31000."
WJ  6125 ? :? "Mozliwa jest renumeracja po
    dprogramu. Linie 0-2 mozna pominac."
ST  6130 ? :? "Dla stacji dyskow w instruk
    cji OPEN    zmiast C: wpisac odpowiedni
    a nazwe     zbioru!"
PF  6135 POSITION 2,22:? "Wcisnij START (p
    owrot do menu)!";
YK  6140 IF PEEK(53279)<>6 THEN 6140
QQ  6145 GOTO 20
IU  6900 ? :? CHR$(253);"Bledna nazwa zbio
    ru!":?
LW  6905 ? "Wcisnij START (powrot do menu)
    !"
EI  6910 IF PEEK(53279)<>6 THEN 6910
UI  6915 TRAP 20000:GOTO 20
HO  7000 REM *** ZAKONCZENIE PROGRAMU ***
GV  7005 ? CHR$(125)
CQ  7010 ? "  K O N I E C   P R O G R A
    M U  "
YU  7015 POSITION 2,5
NG  7020 ? "Zaprojektowane znaki uzyskuje
    sie in- strukcja POKE 756,X (po wczyta
    niu"
TL  7025 ? "generatora z nosnika) lub POKE
     756,";ADRES/256;"po wymazaniu tego pr
    ogramu z pamieci."
WS  7030 ? :? "Instrukcje te nalezy powtor
    zyc kazdo- razowo po instrukcji GRAPHI
    CS lub na- cisnieciu RESET."
RD  7035 ? :? :? "  MENU:  "
PR  7040 ? :? "1 - powrot do menu programu
    ,"
KP  7045 ? :? "2 - wymazanie programu z pa
    mieci.":?
AS  7050 GOSUB 10000
GO  7055 KEY=KEY-48
DR  7060 IF KEY=1 THEN 20
ST  7065 IF KEY<>2 THEN 7030
ZY  7070 ? "Potwierdz wymazanie programu..
    . (T/N)"
BS  7075 GOSUB 10000
LP  7080 IF KEY<>84 AND KEY<>116 THEN 20
IR  7085 GRAPHICS 0:CLR :POKE 566,146:NEW
EM  10000 REM ******** ODPOWIEDZ ********
QQ  10005 CLOSE #1
WA  10010 OPEN #1,4,0,"K:"
AJ  10015 GET #1,KEY
AU  10020 IF KEY>127 THEN KEY=KEY-128
YO  10025 IF KEY>26 AND KEY<32 OR KEY>124
    AND KEY<128 THEN 10000
DJ  10030 RETURN
UA  20000 REM ********** TRAP **********
TU  20005 TRAP 20000:GOTO 20
VZ  30000 REM ********** BEEP **********
RK  30005 SOUND 0,96,10,10
XI  30010 FOR X=1 TO 10
NM  30015 NEXT X
UN  30020 SOUND 0,0,0,0
EG  30025 RETURN
```

LISTING 2

```
LT  0 REM *** ChD's Generators Loader ***
WQ  1 REM Copyright by A.Szyndrowski'1988
EJ  2 REM ********************************
DW  31000 POKE 106,PEEK(106)-10
CF  31005 GRAPHICS 0:POKE 559,0
GJ  31010 X=PEEK(106)+2
FS  31015 IF X=INT(X/4)*4 THEN 31025
PB  31020 X=X+1:GOTO 31015
UY  31025 X=256*X
MX  31030 CLOSE #2:OPEN #2,4,0,"C:"
TM  31035 FOR I=0 TO 1023
UW  31040 GET #2,BIT
GB  31045 POKE X+I,BIT
GH  31050 NEXT I
SC  31055 CLOSE #2
YC  31060 X=X/256:GRAPHICS 0
XO  31065 POKE 756,X
ED  31070 RETURN
```

TELEWIZOR-MONITOR

Telewizory czarno-białe i kolorowe posiadające wewnętrzny zasilacz oddzielający galwanicznie masę wewnętrzną od sieci, nadają się do prac adaptacyjnych przystosowujących je jako monitory do komputerów domowych.

Opracowane, wykonane oraz sprawdzone w praktyce wejście monitorowe do telewizorów typu VELA-205 i JOWISZ-501 poprawiło w sposób widoczny jakość obrazu komputerowego. Schemat wejścia monitorowego (rys. 1) został zaprojektowany w sposób umożliwiający zastosowanie w różnych typach telewizorów np. HELIOS, VENUS, NEPTUN. Dodatkową zaletą tego wejścia jest automatyczne przełączanie źródeł sygnału antena-komputer. Podłączenie sygnału z komputera powoduje uaktywnienie się tonu monitorowego. Dla uzyskania obrazu kolorowego należy wyposażyć telewizory w dekodery PAL. Nie jest wymagane przestrajanie fonii. Omawiane wejście monitorowe współpracuje z komputerem ATARI 65XE.

OBSŁUGA

Komputer łączy się z telewizorem zwykłym monofonicznym kablem ekranowym stosowanym w sprzęcie elektroakustycznym. Podłączenie to może być zrealizowane na stałe nie powodując żadnych zmian w odbiorze telewizyjnym. Dopiero w momencie załączania komputera, po upływie ok. 2-3 sekund telewizor automatycznie przełączy się na odbiór monitorowy obrazu i fonii. Z chwilą wyłączenia komputera, telewizor automatycznie powraca na odbiór antenowy.

Wyjście monitorowe komputerów 8-bitowych ATARI pozwala na uzyskanie dodatkowych funkcji. W tym celu została opracowana prosta przystawka (rys. nr 2) stanowiąca łącznik między wyjściem z komputera a kablem i umożliwiająca wykonanie następujących funkcji:

1. Zdalne regulowanie poziomu dźwięku przy komputerze co stanowi pewną wygodę przy dużych odległościach komputera od telewizora.
2. Przełączanie obrazu czarno-białego na kolorowy i odwrotnie.
3. Podgląd pod audycję telewizyjną. Jest to szczególnie przydatne podczas wczytywania długich programów lub w przypadku oczekiwania na interesujący program telewizyjny.

OPIS DZIAŁANIA UKŁADU WEJŚCIOWEGO

Po dopasowaniu do oporności wyjściowej komputera sygnał wizyjny rozdzielany jest na dwa tory. Tor pierwszy stanowi wzmacniacz szerokopasmowy o sprzężeniu galwanicznym opartym na tranzystorach T1 i T2 oraz T3 dublujący tranzystor istniejący w większości krajowych telewizorów znajdujący się po eliminatorze fonii (patrz rys. 3 i 4). Wzmacniacz ten posiada regulator P1 poziomu napięciowego tranzystora T3 do poziomu identycznego jaki posiada tranzystor w telewizorze oraz regulator P2 określający wielkość wzmocnienia niezbędnego do wysterowania wzmacniacza wizji w telewizorze. Kondensator C o wartości od 0—100pF należy stosować w zależności od telewizora. Dla telewizorów kolorowych wartość ta kształtuje się na poziomie 75pF, a dla telewizorów czarno-białych wartość C = 0. Przy doborze tego kondensatora należy zwrócić uwagę na zakłócenia ekranu sygnałem chrominaucji.

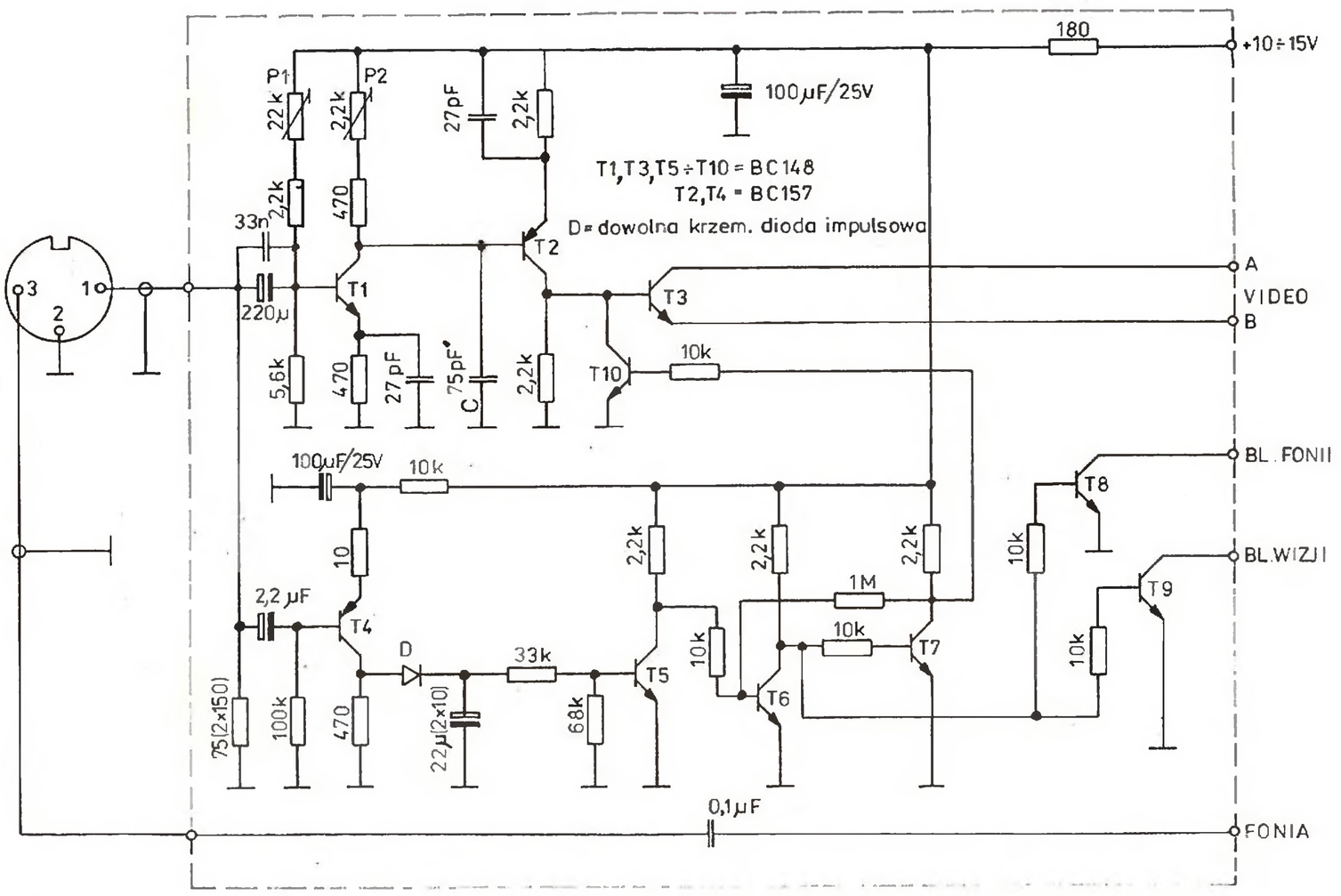
Tor drugi układu wejściowego sprzężony z wejściem kondensatorem 2,2μF pełni funkcję automatyki przełączania źródeł sygnału.

Tranzystor T4 stanowi separator impulsów synchronizacji, które po oddzieleniu diodą D ładują kondensator. Uzyskane w tej części obwodu opóźnienie czasowe załączania i wyłączania sygnału z komputera jest niezbędne dla poprawności pracy wejścia monitorowego w momencie przyciskania klawisza RESET, który powoduje wyłączenie na ok. 1 sekundę impulsów synchronizacji. Obraz w monitorze stawał się niestabilny i układ próbował powrócić na krótko do odbioru telewizyjnego co powodowało migotanie treści obrazu. Tranzystor T5 odciąża prądowo układ zwłoki czasowej. Następne tranzystory T6 i T7 stanowią klucz dwustabilny ustawiający w odpowiedniej pozycji tranzystory blokady fonii T8 i blokady wizji T9.

Tranzystor T10 blokuje tranzystor T3 w przypadku braku sygnału z komputera zapewniając czysty odbiór audycji telewizyjnych.

Część foniczna sprzężona jest poprzez kondensator ze wzmacniaczem m.cz. fonii odbiornika telewizyjnego. W zależności od rozwiązania toru fonicznego w telewizorze możliwe są następujące podłączenia:

1. Umożliwiające regulację głośności potencjometrom odbiornika telewizyjnego (rozwiązanie z VELI 205, rys. Nr 3).
2. Regulacja głośności realizowana jest poprzez przystawkę. Bez ingerencji w konfigurację płytki drukowanej telewizora nie jest możliwe takie podłączenie fonii komputera aby uzyskać efekt jak w pkt. 1. Ponieważ telewizor kolorowy wykorzystywany jest głównie do odbioru gier i znajduje się w znacznej odległości od komputera, znacznie wygodniejsze okazało się stosowanie przystawki z zamontowanym regulatorem fonii. Rozwiązanie to pokazane jest na rys. 4 na przykładzie telewizora JOWISZ 501. Przystawka ta jest tak użyteczna, że stosowana jest również przy innych telewizorach (w tym również przy VELI 205). Proszę zwrócić uwagę jak wygląda „kolor" na czarno-białym telewizorze np. w grze SEAWOLF.



Rys. 1.

WYKONANIE I PODŁĄCZENIE UKŁADU DO TELEWIZORA

Układ wejścia monitorowego należy zmontować na płytce drukowanej. Elementy użyte do montażu powinny być sprawdzone i dobrej ja-

kości, zwłaszcza te, które znajdują się w torze wizyjnym.

Podłączenie płytek w telewizorach pokazano na rys. 3 i 4. Zasilanie płytki +10 do + 15 V należy podłączyć do odpowiedniej ścieżki w telewizorze.

Płytkę łączymy odcinkami kabla ekranowego z gniazdem magnetofonowym, które mocujemy w miejscu najbardziej nam odpowiednim. Ze swej strony proponuję:

— dla telewizora VELA 205-miejsce z tyłu obudowy obok bezpiecznika sieciowego.
— dla telewizora JOWISZ 501-miejsce obok wyścia magnetofonowego.

Układ wymaga regulacji. Po włączeniu komputera i uzyskaniu obrazu na ekranie należy:

— Rezystorem P1 ustawić poziom jasności tła obrazu.
— Rezystorem P2 ustawić wielkości kontrastu zbliżonego do odbioru telewizyjnego.
— Powtórzyć operację regulacji od początku aż do uzyskania poprawnego odbioru obrazu komputerowego.
— Dobrać kondensator C.

Operacje regulacji przeprowadzać przy emisji „SELF TEST".

PRZYSTAWKA

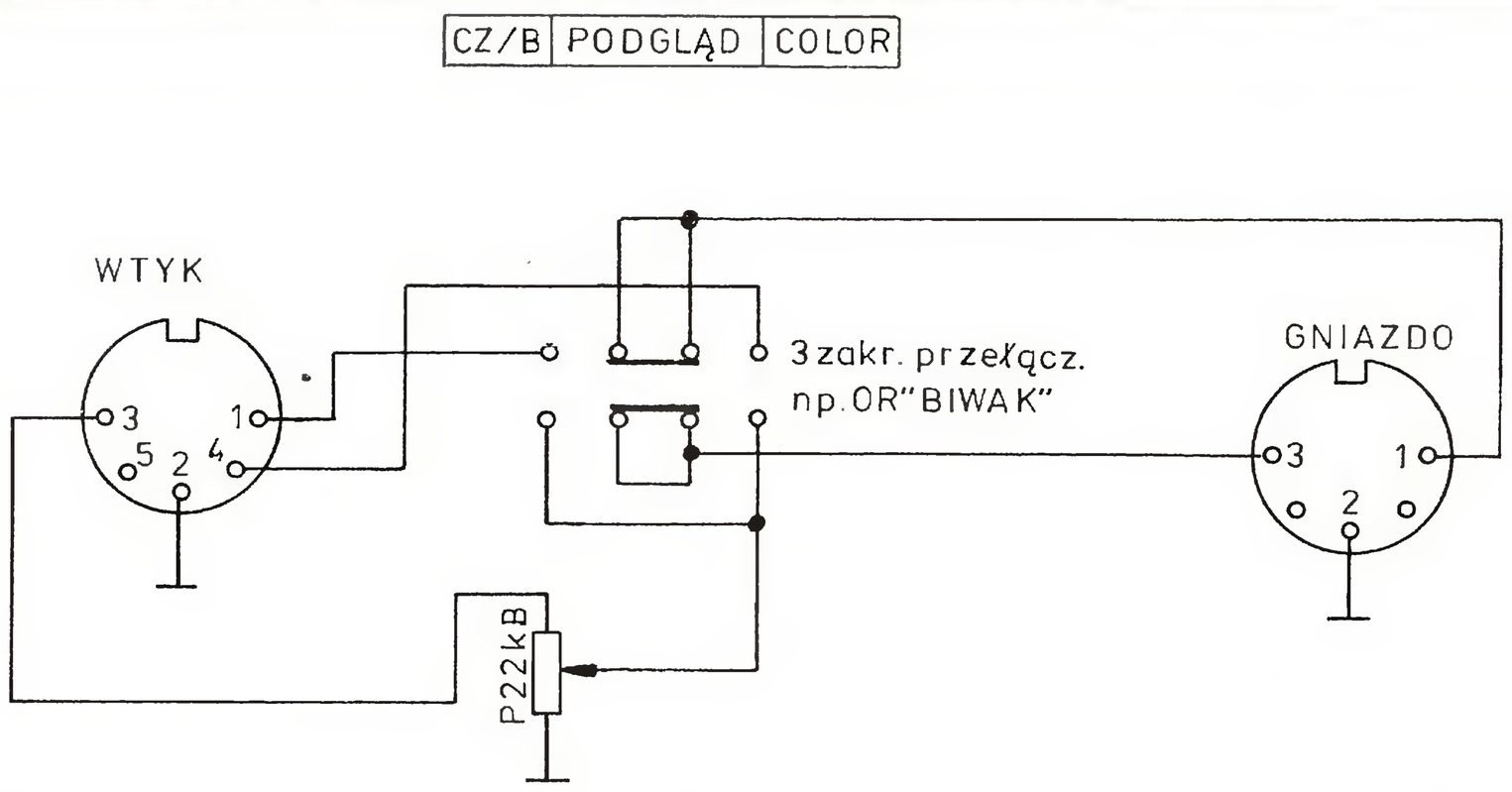
Najbardziej praktyczne rozwiązanie przystawki jest takie, które pozwala złączyć ją na stałe z komputerem. W dowolnym pudełku (koniecznie plastikowym) montujemy mechanicznie: potencjometr, przełącznik 3-pozycyjny, gniazdo magnetofonowe i wtyczkę magnetofonową (koniecznie 5-cio kołkową) w ten sposób, że po podłączeniu przystawki do komputera stanowi ona jego przedłużenie. Należy również uważać aby kształt i wielkości pudełka przystawki nie umożliwiał korzystania z innych gniazd komputera.

UWAGI OGÓLNE

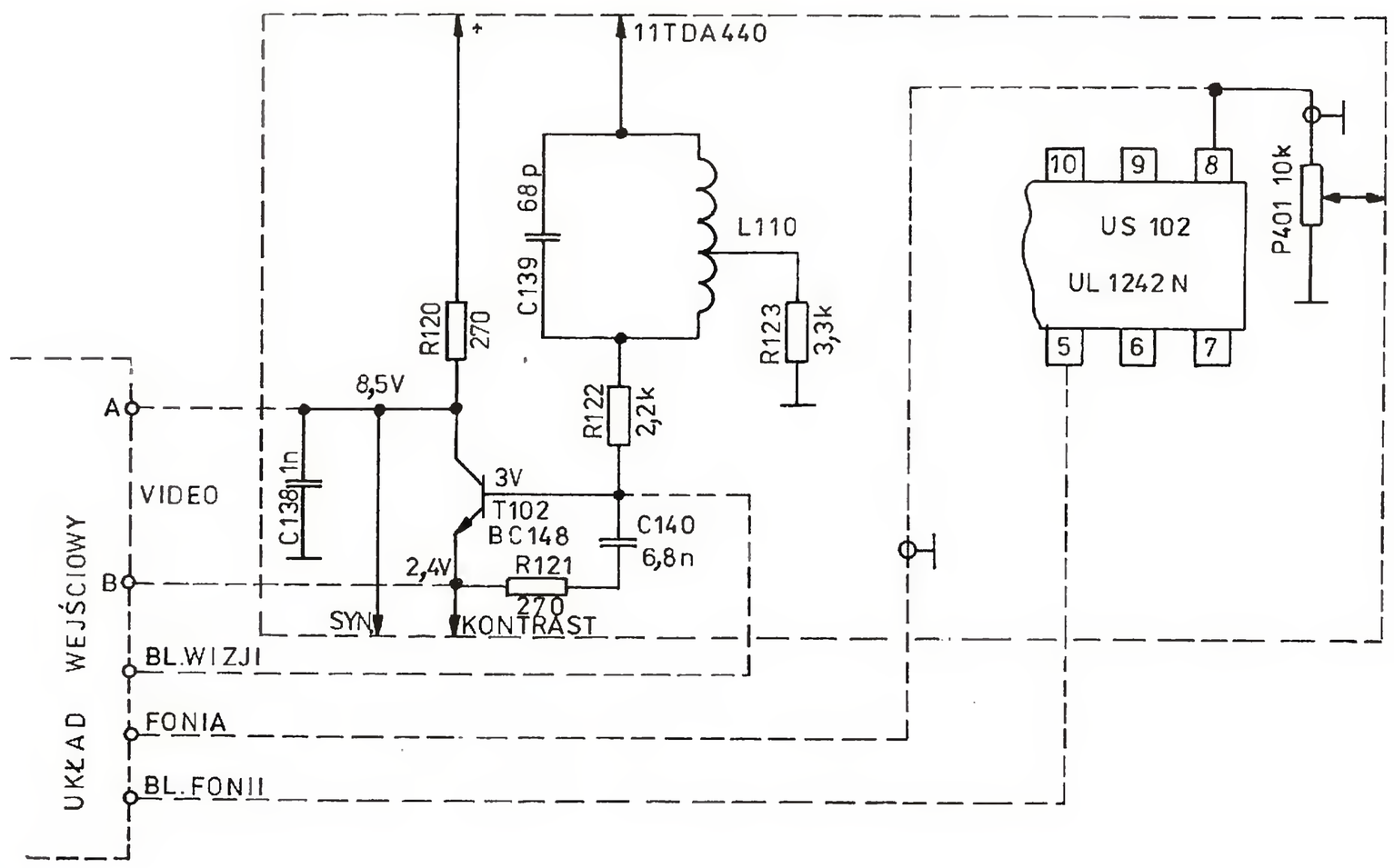
Telewizory produkcji krajowej wykazują szereg wad, które są bardzo widoczne przy pracy monitorowej. Kupując nowe telewizory musiałem przeprowadzić ich gruntowną regulację i tak:

— Telewizor VELA 205 posiadał całkowicie zniekształconą geometrię obrazu. Magnesy stosowane przez WZT były zbyt słabe aby można było nimi skorygować błędy kształtu. Pozytywne efekty uzyskałem dopiero po zastosowaniu magnesu z kwadratowym otworem, które są stosowane do korekcji szerokości obrazu. Po przeprowadzonej korekcji szerokości wymagane jest zabezpieczenie magnesów przed przesuwaniem przez pomalowanie gęstym lakierem.
— W obu telewizorach wystąpiła nieostrość konturów zwłaszcza na krawędziach ekranu. Jeżeli w JOWISZU problem sprowadza się do regulacji, to w VELI należy wykonać dzielnik napięcia w celu uzyskania optymalnego zogniskowania punktu na ekranie kineskopu. Dzielnik wykonujący z rezystorów o sumarycznej rezystancji 2,5 — 5 M podłączamy w punktach określonych na schemacie odbiornika jako „ostrość obrazu". Wielkość napięcia podana na siatkę (nóżka nr 7 kineskopu) powinna zapewniać pełną ostrość obrazu na całej powierzchni ekranu.

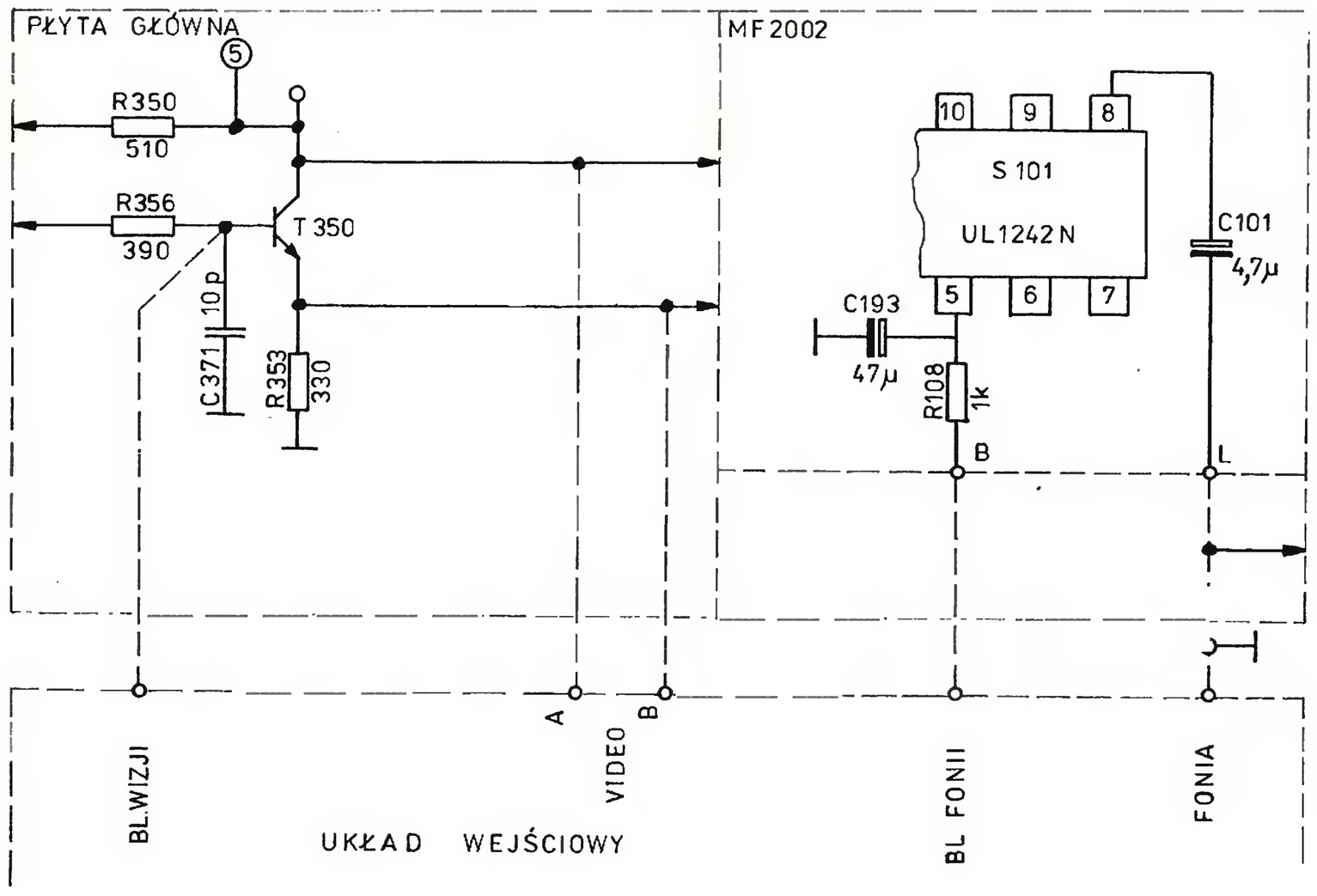
Jerzy Krzyżowski



Rys. 2.



Rys. 3.



Rys. 4.

OKNA

RAZ JESZCZE

Oglądając programy na komputerach Amstrad, Commodore czy sprzęcie profesjonalnym, np. IBM, Amiga, zachwycamy się ich wdziękiem i łatwością obsługi. A dzieje się tak ze względu na zastosowanie tzw. "okien" do wybierania opcji, wyświetlania napisów, itp.

Faktem jest, że podnosi to estetykę i prostotę użytkowania danego programu. Niestety posiadacze komputerów Atari mogli tylko podziwiać okna na innym sprzęcie lub w niewielu grach (np. "Spellbound"). Niniejszy program powstał, aby zniwelować ten niedostatek.

Program współpracuje nie tylko z Basicem, lecz również może być wykorzystany w programach napisanych w innych językach (asembler, Pascal, itp.). Dzieje się to dzięki zainstalowaniu urządzenia "W": w tablicy, zwanej HATABS zajmującej lokacje w pamięci od komórki 794 do 831 ($31A—$33F) i informującej system operacyjny o wszystkich urządzeniach zewnętrznych, jakie mogą być przez komputer obsłużone.

Program jest przeznaczony do pracy wyłącznie w grafice 0, a zmiana tego trybu na inny, np. 8 wiąże się ze zniszczeniem programu i nieuchronnym zawieszeniem systemu po wciśnięciu klawisza RESET. Jest to spowodowane tym, że program zajmuje lokacje tuż pod szczytowym adresem pamięci (MEMTOP), który ma wartość 39976. Z tego powodu zmiana trybu pracy ANTIC-a na tryb graficzny wywoła nieprawidłowe działanie programu lub komputera.

Po tych kilku uwagach wstępnych przejdźmy do rzeczy zasadniczej: jak obsługiwać nowe urządzenie? W Basicu okno otwiera się za pomocą instrukcji OPEN, zamyka się zaś za pomocą instrukcji CLOSE lub jej podobnej. Współrzędne i wymiary okna podaje się po nazwie urządzenia, rozdzielając je przecinkami. Dwa pierwsze parametry oznaczają pozycje na ekranie: pierwszy pozycję X, drugi zaś Y (analogicznie do instrukcji POSITION). Parametry: trzeci i czwarty odpowiadają za rozmiary okna: odpowiednio: szerokość i wysokość. Piąty parametr oznacza wzór ramki okna (dozwolone wartości od 0 do 7). Szósty z kolei parametr oznacza ilość linii do przesuwania w oknie po jego przepełnieniu. Kolejny i zarazem ostatni już parametr informuje o tym, czy przed instrukcją powodującą zamknięcie okna, program ma poczekać na naciśnięcie dowolnego klawisza (wartość parametru różna od zera), czy też nie.

Otwarcie okna w negatywie następuje po wpisaniu nieparzystej cyfry w polu trzeciego parametru instrukcji OPEN. Wszystkie parametry muszą mieć dodatnią wartość mniejszą od 256. Przykładowo, wykonanie OPEN #1,4,1,"W:10,5,7,12,1,2,255" spowoduje umieszczenie okna w negatywie, którego lewy, górny narożnik znajdzie się na pozycji 10,5, a wymiary wyniosą (łącznie z ramkami) 8,13. Należy pamiętać, aby suma danej pozycji i wymiaru nie przekroczyła wartości dopuszczalnej (38 dla pozycji X oraz 22 dla Y), gdyż spowoduje to błąd o numerze 141.

Dozwolone jest pominięcie parametru poprzez stawianie następujących po sobie przecinków, wówczas parametr, który powinien występować między nimi, przybiera wartość odpowiedniego parametru ostatnio otwartego okna. Przykładowo, wykonanie instrukcji OPEN #2,4,0,"W:,,,,,," (lub OPEN #2,4,0,"W:") spowoduje otworzenie okna o ostatnio zadeklarowanych parametrach.

Gdy już wiemy, jak je otwierać na ekranie, czas dowiedzieć się, jakie operacje możemy przeprowadzić na oknie? Dozwolone są dwie komendy: GET oraz PUT (lub ich pochodne, np. INPUT, PRINT). Po wykonaniu PUT w oknie ukazuje się wypisywany znak. Po komendzie GET w oknie pojawia się kursor, którym możemy poruszać się za pomocą klawisza CONTROL i strzałek góra-dół. Naciśnięcie klawisza RETURN spowoduje zwrócenie pozycji kursora, jako zadanego parametru. Można zrezygnować z wybranej opcji naciskając klawisz ESC (parametr przyjmie wartość kodu RETURN czyli 155).

UWAGA! Należy pamiętać o tym, że aktywne jest jedynie ostatnio otwarte okno. Bez względu na numer kanału IOCB (Input/Output/Control Block), przez który zostało wykonane zlecenie OPEN, kolejna operacja zostanie wykonana na ostatnio otworzonym oknie. Dotyczy to również komendy CLOSE. Czytelników, którzy pracują nie tylko z Basicem, informujemy, że otwarcie okna spowoduje odpowiednie ustawienie bloków kontroli We/Wy (patrz "Bajtek" 2/87).

Teraz, gdy wszyscy umiemy obsługiwać nowe urządzenie, kilka uwag na temat wydruku programu. Po bezbłędnym wpisaniu wydrukowanego obok programu uruchamiamy go zleceniem RUN. Spowoduje to nagranie kodu wynikowego na urządzenie, którego nazwę zawiera linia 80. W wypadku, gdy przy wpisywaniu przez pomyłkę poczyniliśmy usterkę, program inicjujący niezawodnie powiadomi nas o tym. Wylistuje on linię programu, w której znajduje się błąd, po jego usunięciu ponownie uruchamiamy poprawnie już wpisany program.

Po nagraniu kodu jesteśmy posiadaczami binarnej wersji programu (umożliwia to załadowanie go opcją "L" DOS-a (posiadacze stacji dysków) lub przez START (użytkownicy magnetofonów), który wzbogaca Twój komputer o kolejne urządzenie zewnętrzne. Osobom bliżej zainteresowanym działaniem "okien" proponujemy disasemblację programu i przeanalizowanie jego działania. Może on stanowić przykład własnej procedury obsługi urządzenia (handler) i zachęcić Czytelników do pisania programów dla kolejnych urządzeń zewnętrznych.

Jeszcze miła niespodzianka dla posiadaczy stacji dysków: polecamy napisać z poziomu DOS-a: A;RETURN D:,W:;RETURN. Kończąc, wszystkim użytkownikom nowego urządzenia "W:" życzymy przyjemnej pracy.

L. Pasternak
J. Pelc

```
QK   1 REM ********************************
GJ   2 REM *                              *
AP   3 REM *   Program OKNA na ATARI      *
GL   4 REM *                              *
QO   5 REM ********************************
XV   10 DIM D$(56),C$(10000),T$(6):TAB=0:FO
     R J=1 TO 6:READ A:T$(J,J)=CHR$(A):NEXT
      J
PG   15 DATA 104,162,16,76,86,228
UH   20 LN=1230:FOR Q=1 TO 44
FR   30 READ D$,SUM,LINE:IF LEN(D$)<>56 THE
     N 100
MG   40 S1=0:FOR W=1 TO 56 STEP 2:Z=0:FOR E
     =W TO W+1:BYTE=ASC(D$(E,E)):BYTE=BYTE-
     48-7*(BYTE>64)
XN   50 IF BYTE<0 OR BYTE>15 THEN 100
YE   60 Z=Z+BYTE*(1+15*(E=W)):NEXT E:TAB=TA
     B+1:C$(TAB,TAB)=CHR$(Z):S1=S1+Z:NEXT W
     :IF S1<>SUM THEN 100
LP   70 NEXT Q
MK   80 CLOSE #1:OPEN #1,8,0,"D:W.COM"
OO   81 REM UZYTKOWNICY MAGNETOFONOW LINIE
     80 WPISUJA W POSTACI: 80 CL.#1:O.#1,8,
     128,"C:"
JO   85 IOCB=848:AD=INT(ADR(C$)/256):AR=ADR
     (C$)-256*AD:MSB=INT(LN/256):LSB=LN-MSB
     *256:POKE IOCB+2,11
WG   90 POKE IOCB+4,AR:POKE IOCB+5,AD:POKE
     IOCB+8,LSB:POKE IOCB+9,MSB:I=USR(ADR(T
     $)):CLOSE #1:NEW
TG   100 ? "Blad w linii ";LINE:LIST LINE:?
      S1
PY   110 DATA FFFF1E97DD9BA609CAD00AA50C8D7
     797A50D8D7897A9768502A99785,3715,110
IE   115 REM UZYTKOWNICY MAGNETOFONOW LINIE
     110
EO   116 REM WPISUJA W POSTACI: 110 DATA 00
     0A18973C97A609CAD00AA50C8D7797A50D8D78
     97A9768502A99785,3044,110
RY   120 DATA 03A93C8D02D31860A2FDE8E8E8E02
     2B059BD1A03D0F4A9579D1A03A9,3621,120
DV   130 DATA C19D1B03A9979D1C03A9E98DF1968
     DE502A9968DF2968DE602A9008D,3729,130
UA   140 DATA D59B8D4402A902850960209E974C3
     C97A9008DD197B124C9309016C9,3115,140
DU   150 DATA 3AB012209F97B00E290F6DD1978DD
     197B004C8D0E4186048ADD1970A,3361,150
QJ   160 DATA 8DD297B01698A0020ED197B00E881
     0F8A8ADD197186DD2978DD19768,3778,160
XU   170 DATA 6010998B999C98D297189B189B4C1
     99B0000008DF096206A9BADED96,3224,170
OW   180 DATA D006ADDB9B8DED96ADD69B386DD89
     B8DEE96ADF096C99BD008A9008D,4341,180
ML   190 DATA EA96EEEB96ADEA96386DD69BCDEE9
     6B0EC8DEE96ADD79B386DD99B8D,4853,190
HY   200 DATA EF96ADEB96386DD79BCDEF969019A
     B8DEE96ADF096C99BD008A9008D,4341,180
ML   190 DATA EA96EEEB96ADEA96386DD69BCDEE9
     6B0EC8DEE96ADD79B386DD99B8D,4853,190
HY   200 DATA EF96ADEB96386DD79BCDEF969019A
     DDB9BF00BCEED96D00620909B20,4213,200
MJ   210 DATA 009B206198CEEB964C1698A8AEEE9
     620BA9BADF096C99BF012ADDA9B,4103,210
NC   220 DATA 29804DF09620A79BA00091CDEEEA9
     6206A9B4C199BAED69BE8ACD79B,3988,220
PH   230 DATA C820BA9BAED99BCAF01FCAACD89B8
     A489848186928A8B1CDAA68A88A,4070,230
HF   240 DATA 91CD8810EF68AA20D09BCA10E2ACD
     89B88ADDA9B29804C8A9B206A9B,3910,240
MJ   250 DATA AED69BE8ADD79B386DDD9BA820BA9
     B20039920909B8DEE96A514C514,3850,250
DH   260 DATA F0FC20039920009BADEE96C98ED00
     BADDD9BF0D0CEDD9B4CA098C98F,4301,260
TE   270 DATA D010AED99BCACAECDD9B90BCEEDD9
     B4CA098C90CD009206A9BADDD9B,4392,270
HJ   280 DATA 4C199BC91CD0A5206A9BA99B4C199
     BACD89B88B1CD498091CD8810F7,3742,280
```

```
YP  290 DATA 60A000B124C93AF003C8D0F7A2FFC
    88CD097207C97B01BE8E006B019,3915,290
UU  300 DATA C998F015C92CD00ECCD097F006ADD
    1979DD69B4C1E99A0A5608AF006,4016,300
WW  310 DATA ADD1979DD69BA52B0EDA9B4A6EDA9
    BADDA9B297FC90890034C1C9B20,3572,310
PP  320 DATA D69A206A9B204B9A20E99920B69AE
    ED59BA9008DDD9BA2008EEA968E,3830,320
HK  330 DATA EB96E88EED96206A9B4C199B206A9
    BADD59BF0E6ADDC9BF00320909B,4121,330
CX  340 DATA CED59B20C69AAED69BADD79B386DD
    99BA820BA9BAED89BE8E88EEE96,4661,340
RJ  350 DATA AED99BE8E8A000209C9A91CDC8CCE
    E96D0F5A5CD38E92885CDB002C6,4520,350
PV  360 DATA CECAD0E5ADD59BF09DCED59B20C69
    AEED59B4C7A99AED69BACD79B20,4815,360
MH  370 DATA BA9BADDA9B2980BDEE96ADDA9B297
    F8DEF960A0A0A186DEF96AA2016,3605,370
BQ  380 DATA 9AACD99B20189A4C169AA00198F02
    C48BD239BACD89B4DEE96208A9B,3541,380
NE  390 DATA BD249BACD89BC84DEE9691CDA000B
    D229B4DEE9691CD20D09B68A888,4094,390
TT  400 DATA 4C189AE8E8E860AED69BACD79B20B
    A9BAED99BE8E8ACD89BC8B1CD20,4671,400
VY  410 DATA 779A8810F820D09BCAD0EE60A691E
    4CC9004A690E4CB608EEF96206C,4211,410
NQ  420 DATA 9A90034C0F9BA6CBD002C6CCC6CBA
    20081CBA6CB8EE502A6CC8EE602,3909,420
OZ  430 DATA AEEF96608EEF96A200A1CBE6CBD00
    2E6CC4C8E9AADD59B0A0A0AAAA0,4066,430
PB  440 DATA 006020AC9AB9D69B9DF596E8C8C00
    890F46020AC9ABDF59699D69BE8,4372,440
JE  450 DATA C8C00890F460ADD69B186DD89BB01
    CC927B018ADD79B186DD99BB00F,3813,450
UF  460 DATA C917B00BADD89BF006ADD99BF0016
    06868A08D60A511F00160206A9B,3500,460
ZD  470 DATA 6868A0808411602O6A9B686868A
    09360A00160206A9BA0A5605152,2987,470
EK  480 DATA 457C007C5A5243515745410045A5
    843535253 7C007C535253C9D5CF,2536,480
RF  490 DATA 5900D9CB55CC49554FD900594BD54
    CD1D2C5FC00FCDAD2C35452547C,3822,490
HR  500 DATA 007C5452540000000000000000A
    003B9CB00AAB9F19699CB008A99,2318,500
YS  510 DATA F1968810EF60A9011865CD85CD900
    2E6CE6091CD8810FB60A2FF8EFC,4054,510
CH  520 DATA 02A511F0F6ECFC02F0F7ADFC028EF
    C024C83F90A084AC960B006E91F,3761,520
SW  530 DATA B0026960289002098060A55885CDA
    55985CE98F00720D09B884CC29B,3337,530
AM  540 DATA 8A4C809BA9284C809B000A0313080
    0010000E002E3023C971E970000,1953,540
```

```
        OPT %000101

*********************
*   HANDLER 'W'     *
*     88.02.15      *
*********************

*       PAGE ZERO

ZIOCB   EQU $20

ST      EQU $CB CC
HL      EQU $CD CE
MEMT    EQU $90
MEMTP   EQU 741

BEGN    EQU $96EA

CX      EQU BEGN+0
CY      EQU BEGN+1
HL      EQU BEGN+2
SC      EQU BEGN+3
A       EQU BEGN+4
B       EQU BEGN+5
C       EQU BEGN+6
PO      EQU BEGN+7
WNDO    EQU BEGN+11
HTBS    EQU $31A

        ORG BEGN+$34

MAIN EQU *

* RESET KEY

        LDX 9
        DEX
        BNE RES1
        LDA $0C
        STA RES2+1
        LDA $0D
        STA RES2+2
RES1    LDA <RES2
        STA $02
        LDA >RES2
        STA $03
        LDA #60
        STA 54018
        CLC
        RTS

* INSTALED HANDLER

MULT EQU *

        PHA
        LDA BYTE
        ASL @
        STA BYTE+1
        BCS RET
        TYA
        LDY #2
MUL1    ASL BYTE
        BCS RET
        DEY
        BPL MUL1
        TAY
        LDA BYTE
        CLC
        ADC BYTE+1
        STA BYTE
RET     PLA
        RTS

* HANDLER 'W'

TABL DTA A(OPEN-1)
     DTA A(CLOSE-1)
     DTA A(GET-1)
     DTA A(PUT-1)
     DTA A(OKAY-1)
     DTA A(OKAY-1)
     JMP OKAY

STTS DTA B(0)
BYTE DTA A(0)

* PUT CHARACTER

PUT  EQU *
     STA C
     JSR FLIP

     LDA SC
     BNE PUTC
     LDA SCRL
     STA SC

PUTC LDA POSX
     SEC
     ADC WIDT
     STA A

     LDA C
     CMP #$9B
     BNE PUT0

STRT LDX #$FD
INT1 INX
     INX
     INX
     CPX #$22
     BCS OVER
     LDA HTBS,X
     BNE INT1
     LDA #'W'
     STA HTBS,X
     LDA <TABL
     STA HTBS+1,X
     LDA >TABL
     STA HTBS+2,X
     LDA <BEGN-1
     STA PO
     STA MEMTP
     LDA >BEGN-1
     STA PO+1
     STA MEMTP+1
     LDA #0
     STA WNDN
     STA 580
     LDA #2
     STA 9
     RTS

RES2 JSR OVER
     JMP STRT

* CONVERSION

CONV LDA #0
     STA BYTE
CLHP LDA (ZIOCB+4),Y
     CMP #'0'
     BCC OVE1
     CMP #':'
     BCS OVE1
     JSR MULT
     BCS OVER
     AND #15
     ADC BYTE
     STA BYTE
     BCS OVER
     INY
     BNE CLHP     JMP
OVE1 CLC
OVER RTS

* * 10

IY   LDA #0
     STA CX
     INC CY

PUT0 LDA CX
     SEC
     ADC POSX
     CMP A
     BCS IY

PUT1 EQU *
     STA A
     LDA POSY
     SEC
     ADC HEIG
     STA B

PUT2 LDA CY
     SEC
     ADC POSY
     CMP B
     BCC PUT3
     LDA SCRL
     BEQ PUT4
     DEC SC
     BNE PUT4
     JSR GETK
     JSR BREK
PUT4 JSR WUP
     DEC CY
     JMP PUT2

PUT3 TAY
     LDX A
     JSR SPOS
     LDA C
     CMP #$9B
     BEQ EOL
     LDA TYPE
     AND #128
     EOR C
     JSR ATIN
     LDY #0
     STA (HL),Y
     INC CX
EOL  JSR FLIP
     JMP OKAY

WUP  EQU *
     LDX POSX
     INX
     LDY POSY
     INY
     JSR SPOS
     LDX HEIG
     DEX
     BEQ ONON
     DEX
WUP1 EQU *
     LDY WIDT

     TXA
     PHA

LAD  TYA
     PHA
     CLC
     ADC #40
     TAY
     LDA (HL),Y
     TAX
     PLA
     TAY
     TXA
     STA (HL),Y
     DEY
     BPL LAD

     PLA
     TAX
     JSR NEXTL

     DEX
     BPL WUP1
ONON LDY WIDT
     DEY
     LDA TYPE
     AND #128
     JMP STDR

* GET

GET  EQU *
     JSR FLIP
GETL LDX POSX
     INX
     LDA POSY
     SEC
     ADC CURS
     TAY
     JSR SPOS

     JSR INVR
     JSR GETK
     STA A
     STA CURS

OOPS LDX #0
     STX CX
     STX CY
     INX
     STX SC
     JSR FLIP
     JMP OKAY

* CLOSE WINDOW

CLOSE EQU *
     JSR FLIP
     LDA WNDN
     BEQ OOPS

     LDA WK
     BEQ *+5
     JSR GETK

     DEC WNDN
     JSR REPAR

     LDX POSX
     LDA POSY
     SEC
     ADC HEIG
     TAY
     JSR SPOS
     LDX WIDT
     INX
     INX
     STX A
     LDX HEIG
     INX
     INX
CLO1 LDY #0
CLO2 JSR POP
     STA (HL),Y
     INY
     CPY A
     BNE CLO2

     LDA HL
     SEC
     SBC #40
     STA HL
     BCS *+4
     DEC HL+1

     DEX
     BNE CLO1
     LDA 20
     CMP 20
     BEQ *-2
     JSR INVR
     JSR BREK
     LDA A
     CMP #14+128
     BNE NUP

* CRS UP

     LDA CURS
     BEQ GETL
     DEC CURS
     JMP GETL

NUP  EQU *
     CMP #15+128
     BNE NDOW

* CRS DOWN
     LDX HEIG
     DEX
     DEX
     CPX CURS
     BCC GETL
     INC CURS
     JMP GETL

NDOW EQU *
     CMP #12
     BNE NENT
     JSR FLIP
     LDA CURS
     JMP OKAY

NENT EQU *
     CMP #28
     BNE GETL
     JSR FLIP
     LDA #$9B
     JMP OKAY

INVR LDY WIDT
     DEY
INV1 LDA (HL),Y
     EOR #$80
     STA (HL),Y
     DEY
     BPL INV1
     RTS

* OPEN WINDOW

OPEN EQU *

     LDY #0
OPE1 LDA (ZIOCB+4),Y
     CMP #':'
     BEQ OPE2
     INY
     BNE OPE1
OPE2 LDX #255
     INY
     STY STTS
     JSR CONV
     BCS EOPE
     INX
     CPX #6
     BCS OPE3
     CMP #$9B
     BEQ OPE3
     CMP #','
     BNE EOPE
     CPY STTS
     BEQ EOPE-3
     LDA BYTE
     STA POSX,X
     JMP OPE2+2

EOPE LDY #165
     RTS

OPE3 TXA
     BEQ OPE4
     LDA BYTE
     STA POSX,X
OPE4 LDA ZIOCB+$B
     ASL TYPE
     LSR @
     ROR TYPE

     LDA TYPE
     AND #127
     CMP #8
     BCC *+5

     JMP WTYP

     JSR TEST
     JSR FLIP
     JSR SAVSC
     JSR DISPW
     JSR SPARA
     INC WNDN
     LDA #0
     LDA WNDN
     BEQ OOPS
     DEC WNDN
     JSR REPAR
     INC WNDN
     JMP OOPS

* DISPLAY WINDOWS

DISPW EQU *
     LDX POSX
     LDY POSY

     JSR SPOS

     LDA TYPE
     AND #128
     STA A
     LDA TYPE
     AND #127
     STA B
     ASL @
     ASL @
     ASL @
     CLC
     ADC B
     TAX

     JSR LINE-2
     LDY HEIG
     JSR LINE
     JMP LINE-2

     LDY #1  ;LINE-2
LINE EQU *
LIN1 TYA
     BEQ LEND
     PHA

     LDA WDEF+1,X
     LDY WIDT
     EOR A
     JSR STDR

     LDA WDEF+2,X
     LDY WIDT
     INY
     EOR A
     STA (HL),Y

     LDY #0
     LDA WDEF+0,X
     EOR A
     STA (HL),Y

     JSR NEXTL
     PLA
     TAY
     DEY
     JMP LIN1

LEND INX
     INX
     INX
     RTS

* SAVE SCREEN

SAVSC EQU *
     LDX POSX
     LDY POSY
     JSR SPOS
     LDX HEIG
     INX
     INX
SSC1 LDY WIDT
     INY
SSC2 LDA (HL),Y
     JSR PUSH
     DEY
     BPL SSC2
     JSR NEXTL
     DEX
     BNE SSC1
     RTS

* PUSH, POP

TSTS EQU *
     LDX MEMT+1
     CPX ST+1
     BCC TSTE
     LDX MEMT
     CPX ST
TSTE RTS
```

```
PUSH STX B
     JSR TSTS
     BCC *+5
     JMP STFUL
PUS1 LDX ST
     BNE PUS2
     DEC ST+1
PUS2 DEC ST
     LDX #0
     STA (ST,X)

PPRT LDX ST
     STX MEMTP
     LDX ST+1
     STX MEMTP+1
     LDX B
     RTS

POP  STX B
     LDX #0
     LDA (ST,X)
     INC ST
     BNE POP1
     INC ST+1
POP1 JMP PPRT

P    EQU *
     LDA WNDN
     ASL @
     ASL @
     ASL @
     TAX
     LDY #0
     RTS

SPARA JSR P
SPAR LDA POSX,Y
     STA WNDO,X
     INX
     INY
     CPY #8
     BCC SPAR
     RTS

REPAR EQU *
     JSR P
REPA LDA WNDO,X
     STA POSX,Y
     INX
     INY
     CPY #8
     BCC REPA
     RTS

* TO BIG?

TEST LDA POSX
     CLC
     ADC WIDT
     BCS TOBI
     CMP #39
     BCS TOBI
     LDA POSY
     CLC
     ADC HEIG
     BCS TOBI
     CMP #25
     BCS TOBI

     LDA WIDT
     BEQ TOBI
     LDA HEIG
     BEQ TOBI
     RTS

TOBI EQU *

     PLA
     PLA
     LDY #141
     RTS

SREK EQU *
     LDA 17
     BEQ *+3
     RTS

     JSR FLIP
     PLA
     PLA
     LDY #128
     STY 17
     RTS

STFUL EQU *
     JSR FLIP
     PLA
     PLA
     PLA
     PLA
STF1 LDY #147
     RTS

GRAY EQU *
     LDY #1
     RTS

WTYP JSR FLIP
     LDY #165
     RTS


WDEF EQU *
     DTA B($11)
     DTA B($12)
     DTA B($05)
     DTA B($7C)
     DTA B($20)
     DTA B($7C)
     DTA B($1A)
     DTA B($12)
     DTA B($03)

     DTA B($11)
     DTA B($17)
     DTA B($05)
     DTA B($01)
     DTA B($20)
     DTA B($04)
     DTA B($1A)
     DTA B($18)
     DTA B($03)

     DTA B($13)
     DTA B($12)
     DTA B($13)
     DTA B($7C)
     DTA B($20)
     DTA B($7C)
     DTA B($13)
     DTA B($12)
     DTA B($13)
     DTA B($7C)
     DTA B($C9)
     DTA B($D5)
     DTA B($CF)
     DTA B($59)
     DTA B($00)
     DTA B($D9)
     DTA B($CB)
     DTA B($55)
     DTA B($CC)

     DTA B($49)
     DTA B($55)
     DTA B($4F)
     DTA B($D9)
     DTA B($00)
     DTA B($59)
     DTA B($4B)
     DTA B($D5)
     DTA B($4C)

     DTA B($D1)
     DTA B($D2)
     DTA B($C5)
     DTA B($FC)
     DTA B($00)
     DTA B($FC)
     DTA B($DA)
     DTA B($D2)
     DTA B($C3)

     DTA B($14)
     DTA B($12)
     DTA B($14)
     DTA B($7C)
     DTA B($20)
     DTA B($7C)
     DTA B($14)
     DTA B($12)
     DTA B($14)

     DTA D'   '
     DTA D'   '
     DTA D'   '


FLIP EQU *
     LDY #3
FLI1 LDA ST,Y
     TAX
     LDA P0,Y
     STA ST,Y
     TXA
     STA P0,Y
     DEY
     BPL FLI1
     RTS

* Z80

IHL  LDA #1
AHL  CLC
     ADC HL
     STA HL
     BCC *+4
     INC HL+1
     RTS

STDR STA (HL),Y
     DEY
     BPL STDR
     RTS

* KEYBOARD

GETK LDX #255
     STX 764
GK   LDA 17
     BEQ GETK-1
     CPX 764
     BEQ GK
     LDA 764
     STX 764
     JMP 63875

* ATASCII TO INTERN

ATIN ASL @
     PHP
     LSR @
     CMP #96
     BCS ATI1
     SBC #31
     BCS ATI1
     ADC #96
ATI1 PLP
     BCC *+4
     ORA #128
     RTS

SPOS EQU *
     LDA 88
     STA HL
     LDA 88+1
     STA HL+1
SPO1 TYA
     BEQ SPO2
     JSR NEXTL
     DEY
     JMP SPO1
SPO2 TXA
     JMP AHL

NEXTL LDA #40
     JMP AHL

WNDN DTA B(0)
POSX DTA B(10)
POSY DTA B(3)
WIDT DTA B(19)
HEIG DTA B(8)
TYPE DTA B(0)
SCRL DTA B(1)
WK   DTA B(0)
CURS DTA B(0)

     ORG $2E0
     DTA A(STRT)
     DTA A(MAIN)

     END
```

OMEGA CAD

I oto pojawiła się na komputerowym rynku kolejna nowość. Ten kto myśli, że OMEGA CAD jest jeszcze jednym programem graficznym na ST jest w błędzie. Bo też nie jest to zwykły program, tylko nowoczesne rozwiązanie sprzętowe wykorzystujące doskonały procesor graficzny HD63484 firmy Hitachi.

Użycie tego właśnie układu scalonego pozwala podczas pracy z Atari ST uzyskać rozdzielczość 1024×512 punktów, z których 82×512 jest równocześnie dostępnych. OMEGA CAD wykorzystuje 256 barw i daje możliwość utworzenia z nich palety 256000 kolorów. To jeszcze nie wszystko. Omawiana karta graficzna dysponuje pamięcią video o pojemności 512 KB. Dzięki temu, że zastosowany procesor graficzny wykorzystuje standardowe funkcje graficzne, komunikacja urządzenia z komputerem jest bezproblemowa.

Karta graficzna OMEGA CAD zabudowana jest w metalowym pudełku o wymiarach 24×20×7 cm. Dołączona jest do niej instrukcja obsługi i trzy dyskietki z oprogramowaniem służącym do obsługi urządzenia (pakiet programów graficznych program demonstracyjny, pakiet programów użytkowych). Podłączenie do Atari ST odbywa się poprzez port ROM i służy do tego wyprowadzony z karty kabel ze specjalnej taśmy. Z tyłu obudowy znajduje się wyłącznik, jak również gniazdo podłączenia do monitora i ewentualnej sieci. Możliwość uzyskiwania bardzo wysokiej rozdzielczości determinuje potrzebę używania w zestawie supernowoczesnego monitora (np. MULTISCAN), którego synchronizacja automatycznie dopasowuje się do otrzymywanego sygnału na wejściu. Wynika to z bardzo szybkich zmian obrazu i wykorzystywania specjalnej techniki telewizyjnej (Interlace-Modus) w celu otrzymania określonego efektu, w tym przypadku wysokiej rozdzielczości.

Aby móc pracować z OMEGA CAD należy najpierw uruchomić kartę przy pomocy podprogramów do obsługi urządzenia tzw. akcesoriów. Po tym zabiegu można wypróbować dołączone programy. Pierwszym jest program graficzny o nazwie „Assist". Posiada on te wszystkie zalety, które użytkownicy Atari ST znają z innych programów tego typu. Ale ponadto dysponuje możliwością uzyskania palety 256000 kolorów, dzięki specjalnej opcji, która pozwala mieszać ze sobą 256 barw dostępnych jednocześnie na ST. Operacja ta odbywa się bardzo szybko, dzięki procesorowi graficznemu, który jest w stanie pokazać cztery miliony punkty obrazu na sekundę, przy czym wypełnianie pustych przestrzeni nie stanowi tu żadnego problemu i odbywa się błyskawicznie. OMEGA CAD dysponuje możliwością zapisu i odtwarzania uzyskanych efektów pracy oraz może korzystać przez odpowiednią konwersję z rysunków wykonanych przy pomocy DEGAS'a bądź NeoChrom'a. Możliwa jest także animacja. Wadą programu jest konieczność używania drugiego monitora — monochromatycznego, do obsługi menu Assist'a. Drugim będącym w zestawie programem graficznym jest trójwymiarowa AxioMega. Na następnej dyskietce znajduje się program demonstracyjny pokazujący możliwości zainstalowanej do ST karty graficznej. Najważniejszą dyskietką, dołączoną do OMEGA CAD jest ostatnia. Zawiera ona bibliotekę najważniejszych języków programowania takich jak: MEGAMAX C, LATTICE C, Omikron i GFA BASIC, CCD PASCAL, DR AR68-Assembler, MODULA II oraz prawie pięćdziesiąt rozkazów do dyspozycji użytkownika służących jednemu tylko celowi — uzyskaniu jak najlepszych efektów graficznych. Dzięki nim można wykreślić wszystko, użyć tekstu do opisu wykonanych rysunków, maksymalnie wykorzystać możliwości GEM'u.

Aktualnie na rynku znajdują się dwie wersje karty graficznej (z możliwością uzyskania dodatkowej palety kolorów i bez niej). Jeżeli ktoś nie jest zadowolony ze swojego ST i chciałby zwiększyć jego potencjał graficzny powinien dołączyć OMEGA CAD do posiadanego systemu. Należy tylko mieć nadzieję, że również całe istniejące oprogramowanie Atari zostanie dopasowane do możliwości opisanej karty graficznej.

Na podstawie „ST Computer"

Sergiusz Piotrowski

PIONOWE DRGANIE OBRAZU

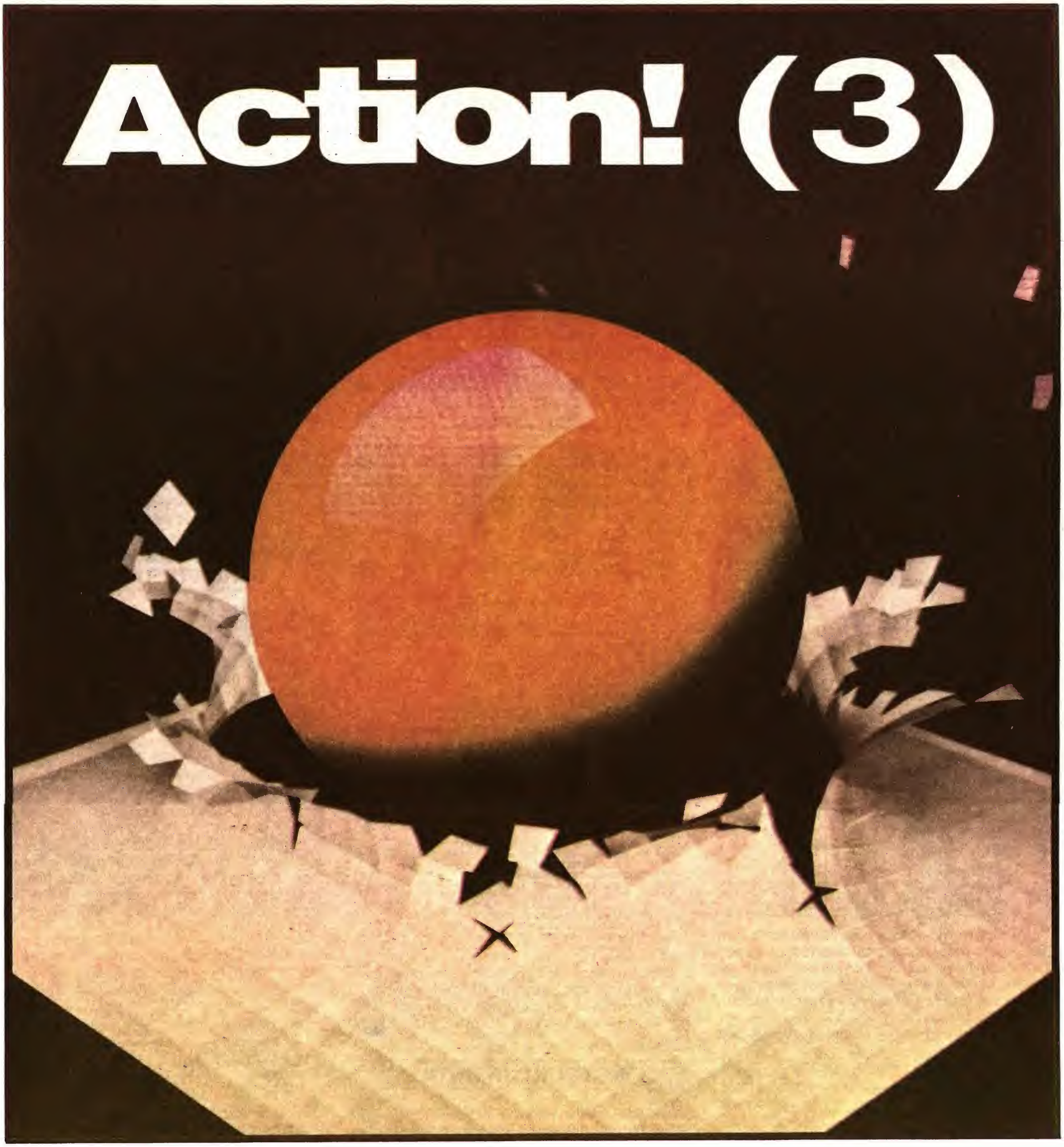
Przedstawiam krótką procedurę w języku maszynowym, która powoduje zmiany programu ANTIC-a.

Korzysta ona z przerwań zegara systemowego TIMER2 i wywołuje płynne ruchy obrazu w pionie, niezależnie od równolegle wykonywanego programu. Daje to możliwość uzyskania ciekawych efektów wizualnych we własnych programach. Procedura działa w trybie GRAPHICS 0, lecz można ją przystosować do innych trybów zmieniając wartości 9 i 32 w instrukcji DATA. Tempo ruchu obrazu reguluje trzynasta liczba w wierszu 30 — jest nią tu 1 (największe tempo). Zatrzymanie ruchu można uzyskać poprzez instrukcję POKE 538,0, a jego wznowienie przez POKE 538,1. Przed ponownym uruchomieniem procedury należy na wszelki wypadek wyzerować komórkę 1700. Procedura jest umieszczona na szóstej stronie pamięci (od adresu 1536) i nie jest odporna na RESET.

Adam Segert

```
RT   1 REM WAHANIE OBRAZU W PIONIE
QZ   2 REM ADAM SEGERT
XO   3 REM COPYRIGHT (C) BAJTEK
NJ   4 REM
DI   10 FOR A=1536 TO 1593:READ B:POKE A,B:
     NEXT A
BS   20 POKE 1700,0:X=USR(1536)
MN   30 DATA 104,169,11,141,40,2,169,6,141,
     41,2,169,1,141,26,2,173,164,6,201,1,24
     0,19,208,0,169,0,141,164,6,206,48,2
FK   40 DATA 173,48,2,201,9,240,2,208,15,169,1,141,
     164,6,238,48,2,173,48,2,201,32,240,224
     ,96
```

Action! (3)



KONTROLA PĘTLI

Action! ma trzy instrukcje strukturalne, które kontrolują podstawową pętlę DO—OD: FOR, WHILE i UNTIL. Sformułowanie „kontrolują pętlę DO—OD" oznacza, że ograniczają one ilość powtórzeń pętli bez końca, przez co uzyskujemy pętlę z wyjściem do dalszej części programu. Kontrolowane pętle są jedną z najbardziej użytecznych i najczęściej używanych konstrukcji programowych.

Teraz zajmiemy się szerzej każdą instrukcją kontroli pętli kolejno, a następnie omówimy specjalną cechę wszystkich instrukcji strukturalnych: zagnieżdżanie.

INSTRUKCJA FOR

Instrukcja FOR służy do powtarzania pętli określoną liczbę razy. Wymaga ona własnej specjalnej zmiennej, zwanej licznikiem lub — bardziej naukowo — zmienną sterującą. We wszystkich przykładach zmienna ta ma nazwę „licz", aby była łatwa do odróżnienia, lecz możesz ją nazwać dowolnie. Format instrukcji FOR jest następujący:

FOR ‹licznik›=‹wart_pocz› TO ‹wart_końc›
[STEP ‹krok›] ‹pętla DO—OD›

gdzie ‹licznik›	jest zmienną służącą do zliczania wykonań pętli
‹wart_pocz›	jest początkową wartością licznika
‹wart_końc›	jest końcową wartością licznika
‹krok›	jest liczbą, o którą komputer zwiększa licznik po każdym wykonaniu pętli (można to opuścić)
‹pętla DO—OD›	jest pętlą DO—OD bez końca

Zamiast objaśniania przy użyciu opisów i przenośni pokażemy kilka przykładów ilustrujących działanie instrukcji. Po każdym z nich następuje wydruk uzyskanego na ekranie wyniku.

Przykład 1:

```
PROC ahoj()

  BYTE licz   ;licznik dla petli FOR

  FOR licz=1 TO 5 ; nie ma STEP wiec
    DO            ; krok = 1
    PrintE("Ahoj przygodo!")
    OD
RETURN
```

Wynik 1:

```
Ahoj przygodo
Ahoj przygodo
Ahoj przygodo
Ahoj przygodo
Ahoj przygodo
```

Przykład 2:

```
PROC parzyste()

  BYTE licz   ;licznik dla petli FOR

  FOR licz=0 TO 16 STEP 2
    DO            ; krok = 2
    PrintB(licz)
    Print(" ")
    OD
RETURN
```

Wynik 2:

```
0 2 4 6 8 10 12 14 16
```

Spójrz ponownie na format instrukcji FOR. Zwróć uwagę, że nie mówiliśmy nic o użyciu zmiennych arytmetycznych jako ‹wart_pocz›, ‹wart_końc› lub ‹krok›. Jest to dozwolone i umożliwia uzyskanie pętli o zmiennej liczbie wykonań.

Jeżeli w środku pętli zmienisz wartości zmiennych użytych jako ‹wart_pocz›, ‹wart_końc› lub ‹krok›, to liczba wykonań pętli nie zmieni się, ponieważ po wejściu do pętli są one zapamiętywane jako wartości stałe. Jeśli są to zmienne, to przyjmowana jest ich wartość w chwili pierwszego wejścia do pętli.

Jeżeli wewnątrz pętli zmienisz wartość licznika, to możesz zmienić ilość pętli, ponieważ ‹licznik› jest w pętli zmienną. Jest tak, gdyż instrukcja FOR sama musi zmieniać wartość licznika po każdym wykonaniu pętli (FOR zwiększa ‹licznik› o wartość ‹krok›. Zmiany ‹wart_pocz›), ‹wart_końc› i ‹krok› wewnątrz pętli ilustruje następny przykład:

Przykład 3:

```
PROC zmianapetli()

   BYTE licz,
        poczatek=[1],
        koniec=[50]

   FOR licz=poczatek TO koniec
      DO
      poczatek=100 ;nie zmienia petli
      koniec=10    ;nie zmienia petli
      PrintBE(licz)
      licz==*2     ;ZMIENIA petle
      OD
RETURN
```

Wynik 3:

```
1
3
7
15
31
```

Zamieszczona niżej tabelka pokazuje, co dzieje się w pętli przy każdym jej wykonaniu. "wyk" pokazuje, które wykonanie pętli zostało zakończone, "zw licz" — wynik zwiększenia licznika przez FOR, "druk" — co jest wyświetlane na ekranie, a "licz==*2" pokazuje, jak instrukcja przypisania zmienia wartość licznika.

wyk	zw licz	druk	licz==*2
1	—	1	2
2	3	3	6
3	7	7	14
4	15	15	30
5	31	31	62

Po piątym wykonaniu pętli licznik jest równy 62. Jest to więcej niż ‹wart_końc› (50), więc pętla kończy się po jedynie 5 wykonaniach zamiast po 50 . Manipulowanie licznikiem wewnątrz pętli może dać ciekawe rezultaty i czasem jest bardzo użyteczne.

A oto przykład wspominanej wcześniej pętli czasowej:

```
BYTE licz

FOR licz=1 TO 250
DO
;*** tu jest instrukcja pusta
OD
```

Służy ona jedynie do opóźnienia wykonywania programu. Stosowana jest często w grach i innych programach, które wymagają dokładnego regulowania czasu pracy.

UWAGA: Jeżeli napiszesz pętlę FOR, w której licznik przekracza limit typu danej (255, gdy licznik jest typu BYTE i 65535, gdy CARD), to pętla nie będzie miała końca, ponieważ wtedy licznik nie może zostać zwiększony do wartości przekraczającej ‹wart_końc›.

INSTRUKCJA WHILE

Instrukcja WHILE (a także UNTIL) jest stosowana, gdy nie chcesz, aby pętla była wykonywana określoną wcześniej ilość razy. WHILE służy do wykonywania pętli tak długo, jak podane wyrażenie warunkowe jest prawdziwe. Ma ona format:

WHILE ‹wyr_war›
‹pętla DO—OD›

gdzie ‹wyr_war› jest wyrażeniem warunkowym kontrolującym pętlę
‹pętla DO—OD› jest pętlą DO—OD bez końca

Ponieważ wyrażenie warunkowe jest obliczane na początku pętli, to ‹pętla DO—OD› może wcale nie być wykonywana. Taki przypadek nie występuje w instrukcji UNTIL. Kolejne przykłady wykorzystują instrukcję WHILE.

Przykład 1:

```
PROC silnia()
;*** procedura podaje silnie, az do
;okreslonej wartosci (zmienna limit)

   CARD silnia=[1],  ;silnia 'num'
        numer=[1],   ;licznik
        limit=[6000] ;gorna granica

   Print("Silnie mniejsze niz ")
   PrintC(limit)
   PrintE(":")
   PutE()
   WHILE silnia*numer<limit
      DO        ;poczatek petli WHILE
      silnia==*numer
      PrintC(numer)
      Print(" silnia jest rowne ")
      PrintCE(silnia)
      numer==+1 ;zwiekszenie licznika
      OD        ;koniec petli WHILE
RETURN          ;koniec procedury
```

Wynik 1:

```
Silnie mniejsze niz 6000:

1 silnia jest rowne 1
2 silnia jest rowne 2
3 silnia jest rowne 6
4 silnia jest rowne 24
5 silnia jest rowne 120
6 silnia jest rowne 720
7 silnia jest rowne 5040
```

Przykład 2:

```
PROC numerkiwhile()
;*** procedura gry w numerki
; uzywajaca petli WHILE

   BYTE numer,          ;zgadywana liczba
        liczba=[200]    ;odpowiedz

   PrintE("Zapraszamy do gry w numerki.")
   PrintE("Pomyslalem liczbe od 0 do 100.")
   numer=Rand(101)
   WHILE liczba<>numer
      DO        ;poczatek petli WHILE
      Print("Jaka to liczba?  ")
      liczba=InputB()
      IF liczba<numer THEN
         PrintE("Za malo, jeszcze raz")
      ELSEIF liczba>numer THEN
         PrintE("Za duzo, jeszcze raz")
      ELSE
         PrintE("Gratulacje!!!")
         PrintE("Odgadles")
      FI        ;koniec sprawdzania liczby
      OD        ;koniec petli WHILE
RETURN          ;koniec procedury
```

Wynik 2:

```
Zapraszamy do gry w numerki.
Pomyslalem liczbe od 0 do 100.
Jaka to liczba?  50
Za malo, jeszcze raz
Jaka to liczba?  60
Za duzo, jeszcze raz
Jaka to liczba?  55
Za malo, jeszcze raz
Jaka to liczba?  57
Gratulacje!!!
Odgadles
```

Zwróć uwagę, jak duże możliwości daje użycie instrukcji warunkowych (np. IF) wewnątrz pętli. Pozwala to komputerowi wykonywać różne warianty instrukcji w różnych przejściach pętli.

INSTRUKCJA UNTIL

W poprzednim ustępie powiedzieliśmy, że pętla WHILE może nie zostać wykonana ani razu, ponieważ wyrażenie warunkowe jest sprawdzane naa początku pętli. Format instrukcji UNTIL jest taki, że pętla musi być wykonana przynajmniej jeden raz. Gdy zobaczysz ten format, zrozumiesz dlaczego tak jest:

DO
‹instrukcja›
UNTIL ‹wyr_war›
OD

Wygląda to jak zwykła pętla DO—OD, aż do miejsca tuż przed „OD". Umieszczone tu UNTIL kontroluje wyjście z pętli bez końca poprzez wyrażenie warunkowe. Jeżeli ‹wyr_war› jest prawdą, to wykonywanie programu jest kontynuowane od instrukcji następującej po OD, w przeciwnym razie wykonywane jest następne przejście pętli. Pamiętaj, że UNTIL musi być umieszczone bezpośrednio przed OD. Oto przykład:

```
PROC numerkiuntil()
;*** procedura gry w numerki
; uzywajaca petli UNTIL

   BYTE numer,          ;zgadywana liczba
        liczba          ;odpowiedz

   PrintE("Zapraszamy do gry w numerki.")
   PrintE("Pomyslalem liczbe od 0 do 100.")
   numer=Rand(101)
      DO        ;poczatek petli UNTIL
      Print("Jaka to liczba?  ")
      liczba=InputB()
      IF liczba<numer THEN
         PrintE("Za malo, jeszcze raz")
      ELSEIF liczba>numer THEN
         PrintE("Za duzo, jeszcze raz")
      ELSE
         PrintE("Gratulacje!!!")
         PrintE("Odgadles")
      FI        ;koniec sprawdzania liczby
      UNTIL liczba=numer   ;kontrola petli
      OD        ;koniec petli UNTIL
RETURN          ;koniec procedury
```

Wynik:

```
Zapraszamy do gry w numerki.
Pomyslalem liczbe od 0 do 100.
Jaka to liczba?  50
Za malo, jeszcze raz
Jaka to liczba?  60
Za duzo, jeszcze raz
Jaka to liczba?  55
Za malo, jeszcze raz
Jaka to liczba?  57
Gratulacje!!!
Odgadles
```

Jest to ten sam przykład, co w opisie WHILE, lecz teraz wykorzystuje pętlę UNTIL. Zwróć uwagę, że nie trzeba nadawać wartości początkowej zmiennej „liczba". Spowodowane jest to sprawdzaniem warunku „numer=liczba" dopiero po podaniu liczby przez użytkownika. Jest to jedna z zalet pętli UNTIL, a wynika ze sprawdzania warunku dopiero na końcu pętli. WHILE sprawdza warunek na początku pętli i wymaga, aby „liczba" miała już wartość.

ZAGNIEŻDŻANIE INSTRUKCJI STRUKTURALNYCH

Jak powiedzieliśmy wcześniej, instrukcje strukturalne składają się z innych instrukcji. Instrukcjami wewnątrz instrukcji strukturalnych mogą być zarówno instrukcje proste, jak i inne instrukcje strukturalne. Umieszczenie instrukcji strukturalnej wewnątrz innej instrukcji strukturalnej nazywa się zagnieżdżeniem.

W opisach instrukcji WHILE i UNTIL możesz zobaczyć przykłady zagnieżdżenia instrukcji IF w pętlach WHILE i UNTIL. Taki rodzaj zagnieżdżenia jest oczywisty i nie będzie szerzej omawiany. Zajmiemy się teraz wielokrotnym zagnieżdżeniem instrukcji strukturalnych tego samego typu (IF wewnątrz IF, FOR w FOR itd.).

Zagnieżdżenie instrukcji IF może spowodować kłopot z określeniem, które ELSE jest związane z jakimś IF. Kompilator unika pomyłek dzięki łączeniu w pary instrukcji IF i FI. FI jest łączone z ostatnią instrukcją IF, która jeszcze nie ma pary. Na przykład:

```
+ IF <wyr_A> THEN
|   + IF <wyr_B> THEN
|   |     <instrukcje>
|   | ELSEIF <wyr_C> THEN ;* do IF <wyr_B
|   |   + IF <wyr_D> THEN
|   |   |     <instrukcje>
|   |   | ELSE              ;* do IF <wyr_D>
|   |   |     <instrukcje>
|   |   + FI          ;* koniec IF <wyr_D>
|   + FI              ;* koniec IF <wyr_B>
| ELSEIF <wyr_E> THEN       ;* do IF <wyr_A>
|     <instrukcje>
| ELSE                ;* do IF <wyr_A>
|     <instrukcje>
+ FI                  ;* koniec IF <wyr_A>
```

Linie wskazują pary instrukcji IF—FI, a komentarze pokazują, do której instrukcji IF należą poszczególne FI i ELSEIF. Orientację ułatwiają dodatkowo wcięcia wydruku.

Następny przykładowy program zawiera zagnieżdżone pętle FOR.

```
PROC tabliczkamnozenia()
;*** procedura podaje tabliczke
;mnozenia w zakresie do 10*10

BYTE l1,  ;licznik zewnetrznej petli
     l2   ;licznik wewnetrznej petli

   FOR l1=1 TO 10 ;petla zewnetrzna
      DO
      IF l1<10 THEN
         Print(" ")
      FI
      PrintB(l1)
      FOR l2=1 TO 10 ;petla wewnetrzna
         DO
         IF l1*l2<10 THEN
            Print("  ")
         ELSEIF l1*l2<100 THEN
            Print("  ")
         ELSE
            Print(" ")
         FI
         PrintB(l1*l2)
         OD    ;koniec petli wewnetrznej
      PutE()
      OD        ;koniec petli zewnetrznej
RETURN          ;koniec procedury
```

Wynik:

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100

Jak możesz zobaczyć w powyższych przykładach, zagnieżdżenie może być bardzo pożyteczne, jeśli wiesz, co chcesz uzyskać.

PROCEDURY I FUNKCJE

Procedury i funkcje czynią program w **Action!** bardziej czytelnym i użytecznym. Prawie wszystko co robimy jest w taki czy inny sposób procedurą lub funkcją. Proszę zobaczyć:

PROCEDURY	**FUNKCJE**
Mycie naczyń	Odczyt z notesu
Gotowanie obiadu	Liczenie pieniędzy
Przejazd do pracy	

Co robią te procedury i funkcje? Każda z nich:
1. jest grupą powiązanych czynności wykonywanych dla zrealizowania określonego zadania;
2. ma ustaloną kolejność, w której te czynności są wykonywane.

W języku komputerowym wygląda to tak samo. Układasz grupę czynności, które wykonują pojedynczą, większą operację w procedurę lub funkcję i nadajesz jej nazwę. Gdy chcesz wykonać tą operację wystarczy użyć nazwy procedury lub funkcji. Jest to nazywane wywołaniem procedury lub funkcji. Oczywiście procedura lub funkcja musi być już zdefiniowana.

A jaka jest różnica między procedurą i funkcją? Obie są uporządkowanym zbiorem czynności wykonującym pewną operację, po co więc dwie nazwy dla takiej samej konstrukcji? Ponieważ nie są to takie same konstrukcje. Funkcja ma dodatkową własność: wykonuje pewną operację i zwraca (przekazuje do miejsca, z którego była wywołana) wartość.

W tabelce umieszczonej na początku tej części jako przykład funkcji został podany „Odczyt z notesu". Dlaczego? Ponieważ w celu odczytania z notesu trzeba wykonać kilka czynności, a na koniec jako wynik otrzymujemy numer telefornu. Ten numer jest zwracany i może być użyty w innej operacji (np. aby umówić się na randkę).

UWAGA: często stosuje się określenie „procedura" dla oznaczenia procedury lub funkcji. Jest to spowodowane niedoskonałością tłumaczenia z języka angielskiego, gdzie „procedure" i „function" oznaczają to samo co w języku polskim, lecz słowo „routine" oznaczające odrębną część programu (procedurę lub funkcję) nie ma specjalnego odpowiednika (czasem mówi się podprogram, ale to nie to samo).

PROCEDURY

Procedury łączą grupę instrukcji w blok z nazwą, który może być wywołany przez podanie nazwy. Aby wykorzystać procedury w **Action!** musisz nauczyć się dwóch rzeczy: deklarowania i wywoływania procedur. Trzy następne ustępy pokażą, jak to zrobić i podadzą kilka przykładów wykorzystania procedur w **Action!**

DEKLARACJA PROC

Słowo kluczowe „PROC" jest stosowane w **Action!** do wskazania początku deklaracji procedury. Konstrukcja procedury składa się z grupy instrukcji, które mają na początku nazwę i inne informacje, a na końcu instrukcję RETURN. A oto format tej konstrukcji:

```
PROC ‹nazwa› [=‹adres›] ([‹lista parametrów›])
        [‹deklaracje zmiennych›]
        [‹lista instrukcji›]
   RETURN
```

gdzie

PROC	jest słowem kluczowym oznaczającym deklarację procedury
‹nazwa›	jest nazwą procedury
‹adres›	określa adres początkowy procedury
‹lista parametrów›	jest wykazem parametrów wymaganych przez procedurę
‹deklaracje zmiennych›	jest to lista zmiennych deklarowanych lokalnie w tej procedurze (nie istniejących poza nią)
‹lista instrukcji›	są to instrukcje zawarte w procedurze
RETURN	oznacza koniec procedury

UWAGA! ‹lista parametrów›, ‹deklaracje zmiennych› i ‹lista instrukcji› mogą być opuszczone. Nigdy zapewne tego nie użyjesz, ale prawidłowa deklaracja procedury może wyglądać i tak:

```
PROC nic()      ;nawiasy są konieczne!
RETURN
```

Nie robi ona nic, lecz wpisanie takiej „pustej" procedury może być użyteczne przy pisaniu dużych programów. Służy to do zastępowania nie napisanych jeszcze procedur przy próbnym uruchamianiu niedokończonego programu.
RETURN i ‹lista parametrów› zostaną opisane dalej, a teraz przykład pokazujący pozostałe elementy deklaracji:

```
PROC numerkiuntil()
;*** procedura gry w numerki
; uzywajaca petli UNTIL

  BYTE numer,          ;zgadywana liczba
       liczba          ;odpowiedz

  PrintE("Zapraszamy do gry w numerki.")
  PrintE("Pomyslalem liczbe od 0 do 100.")
  numer=Rand(101)
     DO        ;poczatek petli UNTIL
     Print("Jaka to liczba?  ")
     liczba=InputB()
     IF liczba<numer THEN
        PrintE("Za malo, jeszcze raz")
     ELSEIF liczba>numer THEN
        PrintE("Za duzo, jeszcze raz")
     ELSE
        PrintE("Gratulacje!!!")
        PrintE("Odgadles")
     FI        ;koniec sprawdzania liczby
     UNTIL liczba=numer    ;kontrola petli
     OD        ;koniec petli UNTIL
RETURN         ;koniec procedury
```

Jest to ten sam program, który był przykładem przy opisie instrukcji UNTIL, lecz wtedy nie rozumiałeś jeszcze instrukcji PROC i znajdujących się po niej deklaracji zmiennych. Już na początku pisaliśmy, że program w **Action!** wymaga deklaracji procedury lub funkcji, aby mógł być skompilowany. Powyższy przykład deklaracji procedury, jest więc równocześnie prawidłowym programem w **Action!** i oczywiście może być skompilowany i uruchomiony. Wynik będzie taki sam jak podany w opisie UNTIL:

```
Zapraszamy do gry w numerki.
Pomyslalem liczbe od 0 do 100.
Jaka to liczba?  50
Za malo, jeszcze raz
Jaka to liczba?  60
Za duzo, jeszcze raz
Jaka to liczba?  55
Za malo, jeszcze raz
Jaka to liczba?  57
Gratulacje!!!
Odgadles
```

Ostatnią instrukcją w przykładzie jest RETURN. Zobaczymy więc, do czego ona służy.

INSTRUKCJA RETURN

RETURN nakazuje kompilatorowi opuszczenie procedury i powrót do miejsca jej wywołania. Jeśli Twój program wywołał procedurę, to jego wykonywanie będzie kontynuowane od instrukcji następującej po wywołaniu procedury. Jeżeli kompilowałeś pojedynczą procedurę (lub program złożony z jednej procedury), to sterowanie zostanie przekazane do monitora **Action!**
UWAGA: kompilator może nie wykryć braku RETURN na końcu procedury. W takim przypadku najczęściej następuje zawieszenie komputera. Dotyczy to także funkcji.

Można użyć więcej niż jedno RETURN w procedurze. Na przykład, jeśli procedura zawiera instrukcję IF z kilkoma wariantami ELSEIF, to możesz żądać opuszczenia procedury po jednym lub po kilku przypadkach ELSEIF. Oto przykład ilustrujący tą możliwość:

```
PROC testpolecenia()
;*** Ta procedura sprawdza, czy
;podane polecenie jest prawidlowe.
;Prawidlowe sa polecenia 0,1,2 i 3.
;Zle polecenie powoduje meldunek
;bledu i powrot do miejsca
;wywolania procedury.

  BYTE pol

  Print("Polecenie>> ")
  pol=InputB()
  IF pol>3 THEN
     PrintE("Bledne polecenie")
     RETURN
  ELSEIF pol=0 THEN
     <instrukcja 0>
  ELSEIF pol=1 THEN
     <instrukcja 1>
  ELSEIF pol=2 THEN
     <instrukcja 2>
  ELSEIF pol=3 THEN
     <instrukcja 3>
  FI
RETURN
```

Zwróć uwagę na RETURN po pierwszym warunku, który sprawdza prawidłowość polecenia. Możesz nie chcieć, aby były sprawdzane pozostałe przypadki, gdy polecenie jest nieprawidłowe, więc następuje tylko wyświetlenie meldunku o błędzie i opuszczenie procedury przez RETURN.

WYWOŁYWANIE PROCEDURY

Wielokrotnie widziałeś już wywołanie procedury, lecz prawdopodobnie nie wiedziałeś, że to właśnie to. We wcześniejszych przykładach używaliśmy już procedur bibliotecznych i właśnie stosowaliśmy ich wywołanie. Jego format jest bardzo prosty:

‹nazwa› ([‹lista parametrów›])

gdzie ‹nazwa›	jest nazwą procedury, którą chcesz wywołać
‹lista parametrów›	zawiera wartości, która chcesz przekazać do procedury jako parametry

Oto przykład:

```
PrintE("Witamy w krainie Bajtka")

silnia()

numerki()

BYTE z
CARD a
koniec(a,z)
```

Oczywiście musisz zadeklarować procedury „silnia", „numerki" i „koniec" przed użyciem ich tutaj. „PrintE" jest procedurą biblioteczną, więc nie jest deklarowana przez Ciebie, lecz jest już zadeklarowana w bibliotece **Action!**. Zwróć uwagę, że nawiasy są wymagane, nawet jeśli procedura nie ma parametrów. Gdy wywoływana procedura ma parametry, w wywołaniu nie może być więcej parametrów niż w deklaracji procedury (lecz może być mniej). Dokładnie opiszemy to później.

FUNKCJE

Podstawową różnicą między procedurą i funkcją jest — jak już wiesz — to, że funkcja zwraca wartość. Powoduje to odmienny sposób jej deklarowania i wywoływania. Ponieważ funkcja zwraca wartość, to musi być użyta w miejscu, w którym użycie wartości jest prawidłkowe (np. w wyrażeniu arytmetycznym).

DEKLARACJA FUNC

Deklaracja funkcji jest bardzo podobna do deklaracji procedury poza tym, że trzeba wskazać,

jaki jest typ wartości zwracanej przez funkcję (BYTE, CARD lub INT) oraz która to wartość. Format deklaracji jest więc następujący:

‹typ› FUNC ‹nazwa› [=‹adres›] ([‹lista parametrów›])
[‹deklaracje zmennych›]
[›lista instukcji›]
RETURN (‹wyrażenie arytm›)

gdzie

‹typ›	jest typem wartości zwracanej przez funkcję
FUNC	jest słowem kluczowym oznaczającym deklarację funkcji
‹nazwa›	jest nazwą funkcji
‹adres›	określa adres początkowy funkcji
‹lista parametrów›	jest wykazem parametrów wymaganych przez funkcję
‹deklaracje zmiennych›	jest to lista zmiennych deklarowanych lokalnie w tej funkcji (nie istniejących poza nią)
‹lista instrukcji›	są to instrukcje zawarte w funkcji
RETURN	oznacza koniec funkcji
‹wyrażenie arytm›	jest wartością, która zostanie zwrócona przez funkcję

Tak jak w deklaracji procedury ‹lista parametrów›, ‹deklaracje zmiennych› i ‹lista instrukcji› mogą być opuszczone. W przypadku procedury może to być użyteczne tylko w jednym przypadku. Natomiast w funkcji jest to stosowane znacznie częściej, jak w poniższym przykładzie:

```
CARD FUNC kwadrat(CARD x)
RETURN (x*x)
```

Ta funkcja pobiera liczbę typu CARD i zwraca jej kwadrat. Mówiliśmy, że zwracana wartość jest w formie wyrażenia arytmetycznego. W naszym przykładzie jest to "(×*×)".

W kolejnym przykładzie zwracana wartość jest po prostu nazwą zmiennej.

```
BYTE FUNC polecenie()
;*** Ta funkcja odczytuje numer
;polecenia i sprawdza, czy jest
;mniejszy niz 8. W przeciwnym
;razie ponownie pyta o polecenie.

   BYTE polecenie,  ;numer polecenia
        blad        ;=1, gdy blad

   DO
   Print("POLECENIE>  ")
   polecenie=InputB()
   IF polecenie<1 OR polecenie>7 THEN
      blad=1
      Print("Zle polecenie: ")
      PrintE("dozwolone tylko 1-7.")
   ELSE
      blad=0
   FI
   UNTIL blad=0
   OD
RETURN (polecenie)
```

UWAGA!‹wyrażenie arytm› w instrukcji RETURN musi być zawsze w nawiasach.

INSTRUKCJA RETURN

Zapewne zauważyłeś, że w formacie deklaracji funkcji RETURN jest użyte w inny sposób niż w deklaracji procedury. W funkcji następuje po nim (‹wyrażenie arytm›). Pozwala to funkcji zwracać wartość. Jeśli spróbujesz umieścić (‹wyrażenie arytm›) po RETURN w deklaracji procedury, otrzymasz błąd, gdyż procedura nie może zwracać wartości.

Pomimo pewnych różnic w użyciu RETURN, jest jedno ważne podobieństwo: zarówno w procedurze jek i w funkcji można kilkakrotnie użyć RETURN. Następny przykład pokazuje wielokrotne użycie RETURN w funkcji:

Funkcja „kwadrat" zadeklarowana poprzednio zwraca wartość kwadratu liczby typu CARD, lecz nie sprawdza, czy wystąpiło przepełnienie. Jeżeli podniesiesz do kwadratu 256, to otrzymasz 65536, a więc o jeden więcej niż zakres liczb CARD (65535). W rezulatacie funkcja zwróci wartość 65536−65535=1. Są dwa sposoby usunięcia tej wady:

1. Wymaganie, aby liczba podnoszona do kwadratu była typu BYTE, czyli czyni niemożliwym wprowadzenie liczby większej niż 255.

2. Sprawdzanie w samej funkcji, czyli wystąpi przepełnienie. Poniższy przykład ilustruje drugi sposób:

```
CARD FUNC kwadrat(CARD x)
;*** Ta funkcja sprawdza, czy 'x'
;nie spowoduje przepelnienia i
;jesli nie, to zwraca jego kwadrat.
;Jesli tak, to zwraca 0.

   IF x>255 THEN
      PrintE("Za duza liczba")
      RETURN (0)    ;zwraca zero
   FI
RETURN (x*x)    ;zwraca kwadrat x
```

Widzisz jakie to proste? Wielokrotne użycie RETURN jest szczególnie użyteczne przy sprawdzaniu wielu warunków, z których każdy wymaga zwrócenia innej wartości.

WYWOŁYWANIE FUNKCJI

Widziałeś już dwa przykłady wywołania funkcji. Możesz je znaleźć w przykładach do opisu WHILE i UNTIL. Jeśli spojrzysz do tych przykładów, zobaczysz następujące linie:

numer=Rand(101)
liczba=InputB()

Pierwsza jest przykładem funkcji wymagającej parametru, a druga — wywołania bez parametrów. Obie te funkcje są funkcjami bibliotecznymi. „Rand"zwraca wartość losową od zera do liczby podanej jako parametr zmniejszonej o jeden (0‹=numer‹=100).

„InputB" odczytuje wartość bajtu z urządzenia edytor. Oczywiście obie funkcje zwracają wartość. Ponieważ musi być ona użyta prawidłowo, to wywołania funkcji muszą być zastosowane w wyrażeniach arytmetycznych. W obu powyższych przykładach wyrażenia składają się tylko z wywołań funkcji i są użyte w instrukcjach przypisania.

Wywołanie funkcji może być użyte w dowolnym wyrażeniu arytmetycznym, z jednym wyjątkiem: **NIE WOLNO** użyć wywołania funkcji w wyrażeniu, które jest parametrem wywołania lub deklaracji procedury albo funkcji.

Przykład: ×=kwadrat(2*Rand(50)) ;Źle

A teraz przykłady prawidłowych wywołań funkcji:

```
×=5*Rand(201)
c=kwadrat(×)-100/×
IF wsk‹›Peek($8000)
znak=duzalitera(znak)
```

„Peek" i „Rand" są funkcjami bibliotecznymi, więc nie muszą być deklarowane przez Ciebie, lecz „kwadrat" i „dużalitera" są funkcjami użytkowanika, więc trzeba jej zadeklarować przed wywołaniem.

UWAGA: można, choć nie jest to zalecane, wywołać funkcję tak, jakby to była procedura. Zwracana wartość jest wtedy ignorowana.

DZIEDZINA ZMIENNYCH

Termin „dziedzina zmiennej" jest stosowany do określenia zasięgu działania i ważności zmiennej. W Basicu zmienne, niezależnie od miejsca, w którym zostały określone, są dostępne w całym programie. Takie zmienne nazywamy zmiennymi globalnymi. W **Action!** oprócz zmiennych globalnych występują zmienne lokalne. Są to zmienne, które, są „widoczne" tylko w ograniczonym obszarze, np. w procedurze. Zmienna lokalna określona w jakiejś procedurze nie istnieje poza nią. Unika się dzięki temu przypadkowej zmiany wartości zmiennej, gdyż w różnych procedurach mogą być użyte zmienne lokalne o takich samych nazwach i nie mające ze sobą żadnego związku. Najłatwiej będzie to zrozumieć na konkretnym przykładzie.

```
MODULE ;deklaracja zmiennych globalnych

CARD iloscgier=[0],
     iloscprob=[10],
     trafne=[0]

PROC wstep()
;**** Ta procedura wyswietla instrukcje
;do gry na ekranie.
   CARD licznik

   PrintE("Witaj w zgaduj-zgaduli.")
   PrintE("Pomysle liczbe od 0 do 100.")
   PrintE("Gdy zapytam Cie, jaka to liczba,")
   PrintE("musisz wpisac odpowiedz.")
   PutE()
   PrintE("Bede pamietal, ile gier rozegrales")
   PrintE("i powiem Ci, ile razy probowales")
   PrintE("odgadnac liczbe, ale najpierw")
   PrintE("musze znac dozwolona ilosc prob.")
   PutE()
   Print("  Wpisz tu ilosc prob --> ")
   iloscprob=InputC()
   FOR licznik=0 TO 2500  ;petla opozniajaca
      DO
      OD
   Put($7D)      ;czyszczenie ekranu
RETURN           ;koniec PROCedury wstep

PROC wynik()
;**** Ta procedura wyswietla aktualny stan gry.

   Print("Rozegrales juz ")
   PrintC(iloscgier)
   PrintE(" gier,")
   Print("i w ")
   PrintC(trafne)
   PrintE("grach odgadles liczbe")
   PrintE("zadajac mniej niz")
   PrintC(iloscprob)
   PrintE(" pytan.")
   PutE()
RETURN          ;koniec PROCedury wynik

PROC rozgrywka()

   CARD proba,   ;nume kolejnej proby
        licznik  ;licznik petli opozniajacej

   BYTE numer,   ;numer do odgadniecia
        liczba   ;odpowiedz gracza

   PrintE("Pomysle sobie liczbe...")
   FOR licznik=0 TO 4500   ;petla
      DO      ;symulujaca namyslanie sie
      OD      ;przez komputer
   PutE()
   PrintE("Juz, teraz Twoja kolei!")
   PutE()
   numer=Rand(101)   ;liczba do odgadniecia
   proba=0   ;ustawienie liczby prob na 0
   DO           ;poczatek petli UNTIL
      Print("Jaka jest Twoja odpowiedz?  ")
      liczba=InputB() ;zgadywanie liczby
      proba==+1  ;zwiekszenie numeru proby
      IF liczba<numer THEN      ;za malo
         PrintE("Za malo, jeszcze raz")
      ELSEIF liczba>numer THEN ;za duzo
         PrintE("Za duzo, jeszcze raz")
      ELSE      ;dobra odpowiedz
         PrintE("Moje gratulacje !!!")
         Print("Odgadles w ")
         PrintC(proba)
         PrintE(" probie.")
         IF proba<iloscprob THEN
            trafne==+1
         FI
      FI      ;koniec testowania liczby
   UNTIL liczba=numer   ;kontrola petli
   OD                   ;koniec petli
RETURN      ;koniec PROCedury rozgrywka

BYTE FUNC stop()
;**** Ta funkcja sprawdza, czy gracz chce
;grac dalej.

   BYTE znowu

   PrintE("Czy chcesz zagrac")
   Print("jeszcze raz? (T lub N)  ")
   znowu=GetD(1)  ; pobranie odpowiedzi z K:
   PutE()
   IF znowu='N OR znowu='n THEN
      RETURN(1)      ;nie chce
   FI
RETURN(0)            ;koniec FUNkCji stop

PROC glowna()

   Close(1)      ;na wszelki wypadek
   Open(1,"K:",4,0) ;otwarcie K: do odczytu
   wstep()      ;wyswietlenie instrukcji
   DO
      iloscgier==+1  ;kolejny numer gry
      rozgrywka()    ;zgadywanie
      wynik()       ;wyswietlenie wyniku
   UNTIL stop()     ;gdy nie chcesz grac
   OD
   PutE()
   PrintE("Zagraj jeszcze kiedys ponownie.")
   Close(1)     ;zamkniecie K:
RETURN          ;koniec PROCedury glowna
```

Poniższa tabela pokazuje, w jaki sposób program korzysta ze zmiennych. Zawiera ona nazwy zmiennych, ich zasięg i dotępność oraz procedury, które używają tych zmiennych. D oznacza, że zmienna jest dostępna w danej procedurze, a U — że została użyta.

ZMIENNA		PROC	PROC	PROC	FUNC	PROC
NAZWA	ZASIES	rozgrywka	wstep	wynik	stop	glowna
iloscgier	globalna	D	D	D U	D	D U
iloscprob	globalna	D U	D U	D U	D	D
trafne	globalna	D U	D	D U	D	D
proba	lokalna	D U				
numer	lokalna	D U				
liczab	lokalna	D U				
licznik	lokalna	D U				
znowu	lokalna				D U	
licznik	lokalna		D U			

Widać wyraźnie, że zmienne globalne są dostępne we wszystkich procedurach, podczas gdy zmienne lokalne są dostępne tylko w tych procedurach, w których zostały zadeklarowane. Zwróć uwagę, że są dwie zmienne o nazwie ´licznik´ — jedna w procedurze ´rozgrywka´ i druga w procedurze ´wstęp´. Pomimo takiej samej nazwy nie jest to ta sama zmienna.

Wojciech Zientara

Mam 12 lat. Komputer Atari 65XE posiadam od niedawna. Niniejszy program ułożony przeze mnie służy do nauki alfabetu Morse'a. Przesyłam go Wam i proszę o ocenę.

Tomek Czyżew
Warszawa

Oceny dokonam w dwóch etapach. Najpierw zajmę się formą, a następnie treścią nadesłanego programu, którego oryginalną wersję przedstawia listing 1.

Gdy spojrzymy na listing, to przede wszystkim rzuca się w oczy ciąg instrukcji IF. Służą one do przypisania każdemu znakowi alfabetu Morse'a odpowiedniej litery. Można to rozwiązać znacznie prościej i oszczędniej umieszczając te litery w instrukcjach DATA. Trzeba przy tym pamiętać o dopisaniu jakiegoś znaku na końcu każdego słowa. Jeśli bowiem liczba danych nie będzie parzysta, to już od drugiego wiersza DATA program przestanie działać. Ponadto konieczne jest dodanie danej po słowie „KONIEC" (wiersz 110), ponieważ jego brak spowoduje przerwanie programu z komunikatem „ERROR — 6 AT LINE 120".

Po odczytaniu danych następnym etapem jest wygenerowanie odpowiedniego sygnału Morse'a. Do tworzenia dźwięku zostały tu ułożone dwie procedury (wiersze 450–490 i 500–540). Pierwsze pytanie brzmi: dlaczego są one wywoływane przez GOTO, a nie przez GOSUB? Wspomniałem już poprzednio o dziwnym zamiłowaniu początkujących programistów do instrukcji GOTO. Jak oni będą pisać programy w Pascalu, Action!, C, Logo i innych językach? Nie ma tam przecież takiej instrukcji. Co prawda Atari Basic nie posiada żadnych instrukcji strukturalnych, ale nie znaczy to, że trzeba wstawiać GOTO wszędzie, gdzie tylko się da. Poza tym nie wymagam, aby wcale jej nie używać. Trzeba to jednak robić z umiarem i tylko tam, gdzie inaczej nie można albo znacznie skomplikuje to program.

Drugie pytanie jest następujące: po co dwie procedury, które robią prawie to samo, a różnią się tylko jedną liczbą. Trzeba wprowadzić dodatkową zmienną określającą czas trwania dźwięku i wtedy wystarczy jedna procedura. Ponadto, dzięki dodaniu jednego wiersza (220 w listingu 2) można tę samą procedurę wykorzystać do tworzenia między poszczególnymi literami przerwy, która musi być dłuższa niż przerwy między sygnałami w znaku.

Po naniesieniu opisanych poprawek otrzymujemy program pokazany na listingu 2. Wykonuje on dokładnie te same czynności, co oryginał, lecz jest znacznie krótszy.

Teraz należy zająć się spełnianą przez program funkcją. Według autora program ma uczyć alfabetu Morse'a. Znam się trochę na uczeniu, gdyż od wielu lat jestem instruktorem żeglarstwa. Na podstawie swoich doświadczeń sądzę, że program ten niczego nie nauczy. Zastanówmy się zatem, jakie cechy musi posiadać program edukacyjny.

Przede wszystkim, aby czegokolwiek nauczyć to trzeba to najpierw pokazać — w tym przypadku należy kolejno wyświetlać litery i nadawać odpowiadający im sygnał dźwiękowy. Dodatkowo można połączyć to z sygnałem akustycznym, gdyż wiele osób lepiej zapamiętuje to, co widzi niż to, co słyszy. Pokaz musi być następnie powtórzony w przypadkowej kolejności (losowe generowanie liter). Teraz powinno nastąpić sprawdzenie opanowanych umiejętności, czyli egzamin z rozpoznawania pojedynczych liter (również losowo wybieranych). Trzecim etapem będzie pokaz nadawania różnych wyrazów. To właśnie wykonuje ten program, lecz jeszcze niezupełnie prawidłowo (wyrazy powinny być oddzielnie, a nie w zdaniu). Na zakończenie trzeba sprawdzić umiejętność odczytania tych wyrazów.

Listing 3 pokazuje przykładowe rozwiązanie ostatniego etapu nauki. W oparciu o ten program łatwo można dopisać część sprawdzającą poszczególne litery oraz dwie fazy pokazu (wystarczy zamienić miejscami wyświetlanie litery lub słowa i tworzenie sygnału). Przykład ten także nie jest doskonały i można go jeszcze ulepszyć.

Chciałbym zwrócić uwagę na kilka spraw. Po pierwsze każdy program powinien być odporny na błędy użytkownika. Tu instrukcja TRAP w wierszu 20 zabezpiecza przed wpisaniem wartości nieliczbowej, zaś właściwy zakres wartości jest sprawdzany przez instrukcję IF (w tym samym wierszu). Drugim takim miejscem jest odpowiedź użytkownika na pytanie „JESZCZE RAZ?" (wiersze 160 i 170). Dwie znajdujące się tu instrukcje IF sprawdzają odpowiedź i powodują wykonanie branego wariantu tylko w przypadku naciśnięcia właściwego klawisza.

Duża liczba instrukcji DATA użytych w tym programie ułatwia dostęp do odpowiedniego sygnału alfabetu Morse'a. Jeśli chcemy to skrócić (4 bajty oszczędności na każdym wierszu programu), to można wpisać wszystkie znaki do jednej zmiennej tekstowej i wybierać z niej odpowiedni fragment. Wymaga to dodania instrukcji rozpoznającej koniec sygnału, ponieważ mają one różną długość. Na marginesie dodam, że przez ingerencję w obszar strony zerowej Basica można ominąć i ten kłopot i zapisać wszystkie znaki w kilku instrukcjach DATA. Jest to jednak nieco trudniejsze.

LISTING 1

```
10 DIM Q$(10),W$(1)
20 DATA .-,.-...,..-.,.-,-...,.,-,!
30 DATA --,---,.-.,...,.,.-,!
40 DATA .--,.,-..,.-..,..-,--.,!
50 DATA .,-.,-.-.,-.--,-.-,.-..,---,.-
-.,.,-...,.,.,!
60 DATA .--.,---,.--.,..-,.-..,.-,.-.,
-.,.,.---,!
70 DATA .--.,.--,-.,!
80 DATA ---,.--.,.-.,.-,-.-.,---,.--,.
-,.-..,!
90 DATA -,---,--,.,-.-,!
100 DATA -.-.,--..,-.--,--..,.,.--,!
110 DATA KONIEC
120 READ Q$
130 IF Q$=".-" THEN W$="A"
140 IF Q$="-..." THEN W$="B"
150 IF Q$="-.-." THEN W$="C"
160 IF Q$="-.." THEN W$="D"
170 IF Q$="." THEN W$="E"
180 IF Q$="..-." THEN W$="F"
190 IF Q$="--." THEN W$="G"
200 IF Q$="...." THEN W$="H"
210 IF Q$=".." THEN W$="I"
220 IF Q$=".---" THEN W$="J"
230 IF Q$="-.-" THEN W$="K"
240 IF Q$=".-.." THEN W$="L"
250 IF Q$="--" THEN W$="M"
260 IF Q$="-." THEN W$="N"
270 IF Q$="---" THEN W$="O"
280 IF Q$=".--." THEN W$="P"
290 IF Q$=".-." THEN W$="R"
300 IF Q$="..." THEN W$="S"
310 IF Q$="-" THEN W$="T"
320 IF Q$="..-" THEN W$="U"
330 IF Q$=".--" THEN W$="W"
340 IF Q$="-..-" THEN W$="X"
350 IF Q$="-.--" THEN W$="Y"
360 IF Q$="--.." THEN W$="Z"
370 IF Q$="!" THEN W$="!"
380 IF Q$="KONIEC" THEN END
390 FOR Q=1 TO LEN(Q$)
400 IF Q$(Q,Q)="." THEN 450
410 IF Q$(Q,Q)="-" THEN 500
420 NEXT Q
430 ? W$;
440 GOTO 120
449 REM *** KROTKI DZWIEK ***
450 SOUND 0,20,10,8
460 FOR E=0 TO 20:NEXT E
470 SOUND 0,0,0,0
480 FOR E=0 TO 5:NEXT E
490 GOTO 420
499 REM *** DLUGI DZWIEK ***
500 SOUND 0,20,10,8
510 FOR E=0 TO 120:NEXT E
520 SOUND 0,0,0,0
530 FOR E=0 TO 5:NEXT E
540 GOTO 420
```

Z powyższym zagadnieniem związany jest sposób odczytu informacji z DATA. Najprościej wykorzystać w tym celu instrukcję RESTORE. Jest ona bardzo użyteczna, a rzadko używana. Tu została zastosowana dwukrotnie: w wierszu 40 i w wierszu 60. W podobny sposób można zapisać w DATA znaczną liczbę rysunków, lecz temat ten wykracza poza ramy oceny i poruszę go innym razem.

Przedstawiony program ma jeszcze jedną zaletę — można go dowolnie rozbudować. W wierszu 900 znajduje się liczba określająca ilość wyrazów zawartych w programie. Można więc dopisywać nowe wyrazy pamiętając tylko, że kolejne numery wierszy muszą rosnąć o 10. Następnie w wierszu 900 wpisujemy nową liczbę wyrazów i to wszystko.

Na zakończenie chciałbym wyrazić nadzieję, że nie wszystkie programy nadsyłane przez Czytelników do oceny będą w Basicu. Przecież na Atari jest tyle lepszych języków.

Wojciech Zientara

LISTING 2

```
10 DIM Q$(10),W$(1)
20 DATA .-,A,.-...,L,..-.,F,.-,A,-...,B
,.,E,-,T,!,!
30 DATA --,M,---,O,.-.,R,...,S,.,E,.-,
A,!,!
40 DATA .--,W,.,E,-..,D,.-..,L,..-,U,-
-.,G,!,!
50 DATA .,E,-.,N,-.-.,C,-.--,Y,-.-,K,.
-..,L,---,O,.--.,P,.,E,-..,D,..,I,..,I
,!,!
60 DATA .--.,P,---,O,.--.,P,..-,U,.-..
,L,.-,A,.-.,R,-.,N,.,E,.---,J,!,!
70 DATA .--.,P,.--,W,-.,N,!,!
80 DATA ---,O,.--.,P,.-.,R,.-,A,-.-.,C
,---,O,.--,W,.-,A,.-..,L,!,!
90 DATA -,T,---,O,--,M,.,E,-.-,K,!,!
100 DATA -.-.,C,--..,Z,-.--,Y,--..,Z,.,
E,.--,W,!,!
110 DATA KONIEC,.
120 READ Q$,W$
130 IF Q$="!" THEN W$=" "
140 IF Q$="KONIEC" THEN END
150 FOR Q=1 TO LEN(Q$)
160 L=0
170 IF Q$(Q,Q)="." THEN L=20
180 IF Q$(Q,Q)="-" THEN L=100
190 GOSUB 220:NEXT Q
200 ? W$;
210 GOTO 120
219 REM *** DZWIEK ***
220 IF  NOT L THEN GOSUB 250:GOSUB 250
:GOTO 250
230 SOUND 0,20,10,8
240 FOR E=0 TO L:NEXT E
249 REM *** PAUZA ***
250 SOUND 0,0,0,0
260 FOR E=0 TO 10:NEXT E
270 RETURN
```

LISTING 3

```
10 DIM Q$(4),W$(15),A$(15):GRAPHICS 0:
? :?
20 TRAP 20:? "WYBIERZ STOPIEN TRUDNOSC
I (1-9)";:INPUT LVL:TRAP 40000:? :? :I
F LVL<1 OR LVL>9 THEN 20
30 READ MAX:LVL=(10-LVL)*2
40 WRD=INT(RND(0)*MAX):RESTORE 2000+WR
D*10:READ W$
50 FOR I=1 TO LEN(W$)
60 CHR=ASC(W$(I))-65:RESTORE 1000+CHR*
10:READ Q$
70 FOR Q=1 TO LEN(Q$):L=0:P=LVL
80 IF Q$(Q,Q)="." THEN L=20
90 IF Q$(Q,Q)="-" THEN L=100
100 GOSUB 200:NEXT Q
110 P=LVL*4:GOSUB 230:NEXT I
120 ? :? "PODAJ WYRAZ ";:INPUT A$:?
130 IF A$=W$ THEN ? "DOBRZE, BYL TO WY
RAZ ";
140 IF A$<>W$ THEN ? "ZLE, POPRAWNIE J
EST ";
150 ? W$:? :? "JESZCZE RAZ (T/N) ?"
160 OPEN #1,4,0,"K:":GET #1,K:CLOSE #1
:IF K=84 THEN 40
170 IF K<>78 THEN 160
180 END
199 REM *** DZWIEK ***
200 IF  NOT L THEN P=LVL*3:GOTO 230
210 SOUND 0,20,10,8
220 FOR E=0 TO L:NEXT E
229 REM *** PAUZA ***
230 SOUND 0,0,0,0
240 FOR E=0 TO P:NEXT E
250 RETURN
899 REM *** LICZBA WYRAZOW ***
900 DATA 10
999 REM *** DANE LITER ***
1000 DATA .-
1010 DATA -...
1020 DATA -.-.
1030 DATA -..
1040 DATA .
1050 DATA ..-.
1060 DATA --.
1070 DATA ....
1080 DATA ..
1090 DATA .---
1100 DATA -.-
1110 DATA .-..
1120 DATA --
1130 DATA -.
1140 DATA ---
1150 DATA .--.
1160 DATA --.-
1170 DATA .-.
1180 DATA ...
1190 DATA -
1200 DATA ..-
1210 DATA ...-
1220 DATA .--
1230 DATA -..-
1240 DATA -.--
1250 DATA --..
1999 REM *** LISTA WYRAZOW ***
2000 DATA ATARI
2010 DATA KOMPUTER
2020 DATA BAJTEK
2030 DATA MORSE
2040 DATA JACHT
2050 DATA PROGRAM
2060 DATA ENCYKLOPEDIA
2070 DATA OCENA
2080 DATA BASIC
2090 DATA ALFABET
```

PaperClip

Edytor tekstu PaperClip został napisany przez Daniela Moore'a i Stevena Ahlstroma — autorów znanej bazy danych SynFile+. Jest to najlepszy i jeden z najpopularniejszych na Zachodzie programów tego typu, zaś w Polsce jest prawie nieznany. Powodem takiego stanu jest brak instrukcji.

Przed opisem działania edytora przedstawię jego podstawowe cechy i parametry.

— pojemność pamięci około 25 KB, a w 130XE — 30 KB;
— duże i wyraźne litery — program używa trybu ANTIC 3;
— jednoczesne redagowanie dwóch tekstów;
— automatyczny zapis na dyskietce:
— dołączanie pliku adresowego z **SynFile+**;
— wbudowane proste operacje matematyczne;
— wbudowana funkcja macro;
— łatwe dopasowanie do każdej drukarki;
— dołączanie grafiki do tekstu;
— możliwość użycia polskich liter na drukarce;
— możliwość wydruku dwukolumnowego;
— rozkazy szybkie i łatwe do zapamiętania;
— wbudowana funkcja HELP.

Pliki zapisane przez **PaperClip** mogą być odczytane przez większość innych edytorów i odwrotnie. Dodatkowo na dyskietce znajduje się plik AWTOPC.OBJ. Jest to program, który — po wczytaniu przy pomocy dowolnego DOS-u — umożliwia przekształcenie pliku zapisanego przez **AtariWriter** na postać nadającą się bez dalszych zmian do druku przy pomocy **PaperClip**.

REDAGOWANIE DOKUMENTU

Redagowanie dokumentu odbywa się w oknie edycyjnym. Parametry określające wygląd tego okna mogą być ustalane przez użytkownika. W tym celu należy nacisnąć klawisz **OPTION**. W wierszu komunikatów pojawia się menu wariantów: „(E)dit (P)rint (M)acro (D)os (S)ave". Menu (oraz większość innych funkcji) można opuścić przy pomocy klawisza **ESC**. Wariant ustalania parametrów edytora wybieramy przez naciśnięcie klawisza **E**.

Teraz ukazują się kolejno pytania o poszczególne parametry oraz ich aktualne wartości (stany). Zmiany dokonuje się wpisując nową wartość (stara kasuje się automatycznie), a gdy odpowiedź ma brzmieć „Yes" (tak) lub „No" (nie), to wystarcza podanie pierwszej litery. Odpowiedni parametr zatwierdza się klawiszem **RETURN**.

Control for cursor — gdy „Yes", to ruch kursora wymaga wciśnięcia klawisza **CONTROL**; gdy „No", kursor przesuwany jest wyłącznie naciśnięciem klawisza ze strzałką, zaś normalne znaki tych klawiszy otrzymamy z **CONTROL** (standardowo „Yes").

Scroll whole screen — „No": przesuwanie wiersza, który nie mieści się na ekranie; „Yes" przesuwanie całego ekranu (standardowo „No").

Left margin — położenie lewego marginesu na ekranie (tylko), może przyjmować wartości od 0 do 2 (0).

Line Length — długość wiersza tekstu na ekranie, może przyjmować wartości od 15 do 132 (40). Parametr ten pozwala na ustalenie takiej szerokości tekstu na ekranie, jaki otrzymamy potem na drukarce. **UWAGA**: zmiana tego parametru powoduje skasowanie zawartości edytora.

Alarm Bell Y/N — sygnalizacja funkcji edytora przy pomocy dźwięku („Yes").

Window Size — wysokość górnego okna edycyjnego przy korzystaniu z dwóch okien, a dolnego przy przeglądaniu dokumentu. Może przyjmować wartości od 4 do 14, standardowo jest 12.

Auto Save Y/N — automatyczny zapis pliku na dyskietkę („No"). Po zmianie na „Yes" kolejne pytania dotyczą liczby znaków (How many characters) i numeru stacji dysków do zapisu (Auto save drive #). Wariant ten jest dokładniej opisany razem z operacjami dyskowymi.

Attract mode — tryb przyciągania uwagi — zmiana kolorów ekranu, gdy przez około 10 minut nie został naciśnięty żaden klawisz („No").

Screen colors — kolory ekranu. Klawisze ze strzałkami w górę i w dół zmieniają jasność tła ekranu, a klawisze < i > barwę. Nie można zmieniać koloru tekstu.

Ruch kursora

Do przemieszczania kursora po ekranie o jeden znak służą standardowe klawisze edytora Atari, czyli klawisze ze strzałkami razem z klawiszem **CONTROL**. Jeśli kursor znajduje się na początku wiersza, to przesunięcie go w lewo spowoduje przejście na koniec poprzedniego wiersza. Analogicznie kursor znajdujący się na końcu wiersza po przesunięciu w prawo przejdzie na początek następnego wiersza.

Ponadto przesunięcie kursora o jeden wyraz w prawo uzyskamy naciskając **SHIFT–CONTROL**-), a w lewo — **SHIFT–CONTROL**-). Kombinacja klawiszy **SHIFT–CONTROL**-> umieszcza kursor na końcu wiersza, w którym się aktualnie znajduje, zaś kombinacja **SHIFT–CONTROL**-< — na początku wiersza.

Druga grupa poleceń umożliwia przemieszczenie kursora w obrębie całego dokumentu. Kombinacja klawiszy **SHIFT–CONTROL**-↑ przesuwa kursor o jeden ekran w górę, a **SHIFT-CONTROL**–↓ o jeden ekran w dół. Bezwzględna wielkość przesunięcia zależy od aktualnej wielkości wykorzystywanego okna edycyjnego. Dwie następne kombinacje klawiszy są użyteczne głównie przy długości wiersza przekraczającej szerokość ekranu. **SHIFT-CONTROL-]** przesuwa okno edycyjne o jeden znak w prawo (więc tekst w oknie w lewo), odwrotne działanie ma **SHIFT-CONTROL-[**.

W celu ustawienia kursora na początku tekstu należy nacisnąć **SHIFT-CONTROL-H** (Head), zaś na końcu tekstu kursor jest umieszczany przez **SHIFT-CONTROL-E** (End).

Pozostało jeszcze przenoszenie kursora między oknami, gdy używane są dwa (niezależnie od sposobu wywołania drugiego okna). Służy do tego klawisz **SELECT**. Poza tym naciśnięcie go, gdy jest czynne tylko jedno okno spowoduje otwarcie drugiego okna.

Poszukiwanie tekstu

Jedną z najważniejszych funkcji każdego edytora jest przeszukiwanie tekstu i ewentualna wymiana jego fragmentów. Dla wyszukania w tekście ciągu znaków należy nacisnąć klawisze **SHIFT-CONTROL-F** (Find). Na pytanie „Find?" podajemy żądane znaki i naciskamy **RETURN**. Jeśli ciąg taki nie występuje w tekście, to pojawi się napis „Not Found". W przeciwnym przypadku kursor zostanie ustawiony na pierwszym znaku pierwszego napotkanego ciągu. Trzeba przy tym pamiętać, że dokument jest przeszukiwany zawsze w dół od aktualnej pozycji kursora.

Ponowne naciśnięcie podanych klawiszy powoduje odszukanie następnego miejsca występowania ciągu. Jeśli jednak w międzyczasie został naciśnięty jakikolwiek inny klawisz, to komputer żąda potwierdzenia poszukiwanego ciągu.

Wyszukiwanie i wymianę ciągu znaków uzyskuje się poprzez **SHIFT-CONTROL-S** (Substitute). Na pytanie „Global substitute?"odpowiadamy „No", jeśli chcemy wymienić jeden ciąg, a „Yes" w przeciwnym razie (teraz wymiana odbywa się od początku dokumentu, niezależnie od położenia kursora). Po odpowiedzi „Yes" pozytywna odpowiedź na kolejne pytanie „Include files also?" umożliwia dokonanie wymiany także w plikach dołączonych do dokumentu funkcją **Include**. Teraz na pytanie „Substitute?" podajemy nowy ciąg, zaś na „For?" stary. Zamiast wykonania operacji pojawia się jeszcze jedno pytanie: „More?". **PaperClip** pozwala bowiem na jednoczesną wymianę sześciu ciągów znaków i dopiero po negatywnej odpowiedzi na ostatnie pytanie przystępuje do jej realizacji.

Przemieszczanie i kopiowanie

Bez funkcji kopiowania i przenoszenia bloków tekstu nie może się odbyć żaden edytor. W **PaperClipie** są one połączone i wywoływane przez naciśnięcie **SHIFT-CONTROL-M** (Move). Kursor musi być przedtem umieszczony na początku bloku. Po poleceniu „Set Range" ustawiamy kursor na końcu bloku i naciskamy **RETURN**. W wierszu komunikatów ukazuje się menu:„(M)ove (C)opy (D)elete" (przeniesienie, skopiowanie, skasowanie). Wyboru dokonujemy odpowiednim klawiszem.

W przypadku dwóch pierwszych opcji komunikat „Destination" żąda wskazania miejsca umieszczenia przenoszonego bloku, co wykonujemy przesuwając kursor i naciskając **RETURN**. Po wybraniu kasowania programu upewnia się jeszcze: „Delete range Y/N?". Długość bloku dla tych operacji jest ograniczona i jej przekroczenie wywołuje komunikat „Move too large" i przerywa funkcję.

Do przenoszenia bloków tekstu można użyć jeszcze jednej funkcji. Kombinacja klawiszy **SHIFT--DELETE** usuwa wiersz, w którym znajduje się kursor i umieszcza go w buforze. Naciśnięcie klawiszy **SHIFT-CONTROL-P** (Paste) powoduje przepisanie zawartości bufora z powrotem do okna edycyjnego.

Kasowanie tekstu

Dwa sposoby usuwania zbędnych fragmentów tekstu zostały już przedstawione powyżej. Jest ich jednak znacznie więcej.

Do kasowania pojedynczych wyrazów służą klawisze **SHIFT-CONTROL-5** usuwające wyraz, w którym aktualnie znajduje się kursor. Duże fragmenty dokumentu usuwa się przez naciśnięcie **SHIFT-CONTROL-DELETE**. Następnie trzeba wybrać, czy tekst ma być usunięty od kursora do początku (Top), czy do końca (End) dokumentu.

Do skasowania okna (wraz z zawartością) służą klawisze **SHIFT-CONTROL-D** (Delete). Funkcja ta działa jedynie przy dwóch czynnych oknach, a dla okna edycyjnego wymaga dwukrotnego potwierdzenia.

Możliwe jest również skasowanie całej zawartości edytora poprzez klawisze **CONTROL-CLEAR**. PaperClip upewnia się pytaniem „Clear (Y)es (N)o (R)estart?". Ostatni wariant powoduje uruchomienie programu od początku.

Macro

Jednym z najciekawszych udogodnień udostępnianych przez **PaperClip** jest funkcja **Macro**. Pozwala ona na umieszczenie w dowolnych miejscach tekstu gotowych fragmentów przygotowanych wcześniej przez użytkownika. Jest to szczególnie wygodne przy częstym powtarzaniu pewnych fraz, akapitów itp.

Plik **Macro** przygotowuje się przy pomocy edytora, jak normalny dokument. Musi on spełniać tylko dwa warunki: mieć nie więcej niż 2000 znaków i nie zawierać znaku "=". W jednym pliku może być kilkanaście oddzielnych fragmentów **Macro**. Każdy z nich oznaczany jest na początku etykietą złożoną z cyfry, małej litery lub innego znaku i znaku "=" (dlatego nie może on występować w innych miejscach pliku). Etykiety mogą znajdować się w dowolnym miejscu tekstu, nawet w środku słowa. Odpowiednio przygotowany plik **Macro** zapisujemy na dyskietkę. Do programu dołączony jest przykładowy plik o nazwie MACRO, w którym etykietami są kolejno: 1, +, a, b, 2 i 3.

W celu wykorzystania pliku **Macro** wywołujemy klawiszem **OPTION** menu wariantów i naciskamy **M**. Następnie na pytanie "(M) Macro file name" wpisujemy nazwę pliku. Po odczytaniu z dyskietki jest on gotowy do użycia. Teraz każdorazowe naciśnięcie klawisze **START** i klawisza ze znakiem etykiety spowoduje umieszczenie w oknie edycyjnym odpowiedniego fragmentu pliku **Macro**.

Funkcje arytmetyczne

Kolejną nietypową funkcją **PaperClipa** jest możliwość wykonywania prostych działań arytmetycznych. Są one dostępne po wpisaniu znaku **CONTROL-Z** (na ekranie wygląda on jak "s" w ramce), znaku działania i liczby. Wszystkie działania są wykonywane z dokładnością dwóch miejsc po przecinku (standardowo) lub w zaokrągleniu do wartości całkowitych i korzystają z jednego bufora. Oto wykaz dostępnych działań:

s+ — dodanie liczby do bufora.
s− — odjęcie liczby od bufora.
s* — mnożenie bufora przez liczbę.
s/ — dzielenie bufora przez liczbę.
s? — drukowanie zawartości bufora.
s= — drukowanie i zerowanie zawartości bufora.
s# — zmiana formatu liczby.

Wszystkie operacje — poza ostatnią — powodują wydrukowanie liczby. Dla łatwiejszego zrozumienia tej rzadko spotykanej funkcji podam przykład:

na ekranie	wydruk
s=	0.00
s+2	2.00
s*3	3.00
s?	6.00
s#	
s?	6
s+4	4
s=	24
s?	0

Inne funkcje

Pozostałe funkcje redakcyjne zostaną opisane w skrócie, ponieważ ich działanie jest podobne, jak w innych edytorach tekstu.

SHIFT-CONTROL-I (Insert) przełącza tryby pracy edytora: wstawiania i zastępowania. Aktualny tryb jest wyświetlany w wierszu komunikatów (Replace/Insert)

SHIFT-CONTROL-CAPS przełącza literę pod kursorem z dużej na małą i odwrotnie.

SHIFT-CONTROL-1 powoduje podanie liczby wyrazów zawartych w dokumencie. Wynik nie jest dokładny, gdyż program liczy spacje, a nie wyrazy.

SHIFT-CONTROL-3 zamienia miejscami znak pod kursorem i znak w lewo od niego.

SHIFT-CONTROL-4 zamienia miejscami wyraz pod kursorem i wyraz w lewo od niego.

SHIFT-CONTROL-SPACE daje tzw. sztywną spację, czyli spację, na której nie można dokonać podziału przy przenoszeniu wyrazów podczas formatowania tekstu do druku.

SHIFT-CONTROL-T (tag) ustawia etykietę. Na pytanie "tag id:" należy podać literę oznaczającą tą etykietę.

SHIFT-CONTROL-G (Goto tag) powoduje skok do etykiety oznaczonej literą, która została podana w odpowiedzi na "tag id:" Jeżeli taka etykieta nie występuje w pliku, pojawia się komunikat "tag not set".

OPERACJE DYSKOWE

Podstawowymi operacjami dyskowymi są: zapis i odczyt pliku. Wywołuje się je odpowiednio przez kombinacje klawiszy **SHIFT-CONTROL-W** (Write) i **SHIFT-CONTROL-R** (Read), po czym należy podać nazwę pliku. Przy zapisie **PaperClip** pamięta nazwę pliku ostatnio odczytanego lub zapisanego (niezależnie dla każdego okna) i jeśli ma pozostać niezmieniona, to wystarczy zatwierdzić ją przez **RETURN**.

Odczytywany plik może być dopisany do znajdującego się w oknie edycyjnym lub powoduje jego skasowanie. Drugi przypadek zachodzi po pozytywnej odpowiedzi na pytanie "Clear current window?". Dopisywanie również należy potwierdzić na żądanie "Append to text?". Teraz kolejnym pytaniem jest: "Append at cursor position?" (dodać na pozycji kursora?). Trzeba przy tym pamiętać, że cały tekst znajdujący się za kursorem ulegnie zniszczeniu.

Trzecią standardową operacją jest odczyt katalogu dyskietki, który uzyskuje się poprzez **SHIFT--CONTROL-?.** Ponadto wymagane jest podanie numeru stacji dysków. Katalog jest odczytywany do drugiego okna edycyjnego. Jeżeli jest ono zajęte, pada pytanie "Clear 2nd window?". Negatywna odpowiedź przerywa operację.

PaperClip umożliwia także dołączanie plików z dyskietki do aktualnego tekstu. W tym celu należy nacisnąć **CONTROL-Z**, a następnie wpisać (bez spacji) **ID:nazwa** (Include). Dołączenie takie można wykonać w środku dokumentu, gdyż nie niszczy ono dalszej treści. Powoduje to jednak ograniczenie liczby kolejnych zagnieżdżeń funkcji **Include.** Po jej przekroczeniu wyświetlany jest komunikat „Too many nested includes" (zbyt dużo zagnieżdżonych dołączeń).

Pozostałe operacje dyskowe są dostępne z menu wariantów (**OPTION**) po wybraniu **(D)os**. Ukazuje się wtedy następne menu: "(E)rase (R)ename (I)nit (P)ro (U)npro". Kolejne jego pozycje oznaczają: usunięcie pliku, zmianę nazwy pliku, formatowanie dyskietki, zabezpieczenie i odbezpieczenie pliku. Dyskietka może być formatowana zarówno w pojedynczej (Single), jak i podwójnej (Double) gęstości. **PaperClip** automatycznie rozróżnia format dyskietki poza rozszerzonym (Enhanced) formatem DOS 2.5. Wymienione operacje są uruchamiane klawiszem **START**, a przerywane **ESC**.

Dodatkową funkcją edytora jest możliwość automatycznego zapisu redagowanego dokumentu. Można to uruchomić przy ustawianiu parametrów edytora lub przez naciśnięcie **SHIFT-CONTROL--TAB**. Po wpisaniu ustalonej liczby znaków redagowanie jest przerywane i plik zostaje zapisany jako PCTEMP z kolejnym numerem. Użytkownik jest ostrzegany na 10 znaków prze zapisem przez komunikat "10 key strokes till save".

Jeszcze jedna funkcja jest wybierana z menu wariantów: **Save**. Służy ona do zapisu kopii programu w pliku PC. SYS. Niestety w posiadanym przeze mnie egzemplarzu nie działa.

WYDRUK DOKUMENTU

Wydruk dokumentu można wywołać dwoma sposobami. Kombinacja klawiszy **SHIFT-CONTROL--ESC** powoduje drukowanie ze standardowymi parametrami, zaś **SHIFT-CONTROL-O** pozwala na wybór parametrów. Parametry wydruku są następujące (w nawiasach wartości standardowe):

Print device? — urządzenie do wydruku (P:). Można tu wpisać nazwę pliku dyskowego (D:nazwa), program pyta wtedy "Send control codes?". Po odpowiedzi "Yes" uzyskamy plik ze wszystkimi kodami sterującymi dla drukarki, zaś po "No" czysty plik ASCII.

Starting page number? — numer pierwszej drukowanej strony (1). Umożliwia pominięcie początku dokumentu.

Number of pages to print? — liczba drukowanych stron (All — wszystkie). Można wydrukować tylko kilka.

Number of copies to print? — liczba kopii wydruku (1).

Pause between pages Y/N — przerwy między stronami (No — nie). Przy druku na pojedynczych kartkach trzeba ustawić „Yes".

Przed drukiem dokument można obejrzeć naciskając klawisze **SHIFT-CONTROL-INVERSE**. Edytor pyta o numer pierwszej przeglądanej strony (Starting page number?), a następnie wyświetla ją w drugim oknie w takiej postaci, jaką uzyskamy na wydruku. Ponieważ, cała strona nie mieści się na ekranie, to można ją przesuwać kursorem. Klawisz **N** (Next) powoduje formatowanie następnej strony, **P** (Previous) poprzedniej, a **S** (Specify)-strony o podanym numerze. Przeglądanie kończy się przez skasowanie okna (**SHIFT-CONTROL-D**).

PaperClip posiada dodatkową możliwość pracy w trybie maszyny do pisania (TypeWriter mode). Do przełączania trybu pracy służą klawisze **SHIFT--CONTROL-2.** W trybie maszyny do pisania okno edycyjne ma wysokość jednego wiersza. Wpisany tekst jest drukowany natychmiast po każdym naciśnięciu **RETURN**.

Konfiguracja drukarki

Różnorodność produkowanych typów drukarek sprawia znaczne kłopoty użytkownikom i programistom. Albo program działa ze wszystkimi drukarkami, lecz jego możliwości są ubogie, albo ma bogate możliwości, lecz tylko dla jednego typu drukarki. Najczęściej stosowanym sposobem jest używanie oddzielnych plików ustalających konfigurację drukarki. Dyskietka **PaperClip** zawiera ponad 30 takich plików. Mają one nazwy z rozszerzeniem CNF. Jeśli Twoja drukarka ma inne kody sterujące, to zmianę konfiguracji można przeprowadzić wybierając **(P)rint** z menu wariantów. Ukaże się pytanie "Config file name?", po którym trzeba wpisać nazwę właściwego pliku i to wszystko.

Dla posiadaczy zupełnie nietypowych drukarek jest przewidziany dodatkowy program PRNT.COM, który należy wczytywać przy pomocy dowolnego DOS-u. Tworzy on plik konfigurujący według danych podanych przez użytkownika. Wystarczy przepisać z instrukcji drukarki odpowiednie kody.

Formatowanie wydruku

Uzyskanie odpowiedniego wyglądu wydruku wymaga umieszczenia w tekście poleceń formatujących. **PaperClip** dysponuje i tu szeroką gamą możliwości:

— Tabulacja — **CONTROL-A** — wysłanie do drukarki znaku tabulacji.

— druk wytłuszczony — **CONTROL-B** (Bold) — wymaga określenia, czy jest to początek (Start), czy koniec (End.).

— kursywa — **CONTROL-I** (Italics) — Start/ /End.

— podkreślenie — **CONTROL-U** (Underline) — Start/End.

— górne i dolne indeksy — **CONTROL-S** (Super/Subscript) — trzeba wybrać początek indeksów górnych (SuPer), początek indeksów dolnych (SuB) lub koniec indeksów (End).

— krój liter — **CONTROL-F** (Font) — do wyboru jest 10, 12 i 15 znaków na cal oraz druk poszerzony (Opt).

Normalnym ustawieniem programu jest standard Epson FX80.

— marginesy — **CONTROL-M** (Margins) — po wybraniu: górny (Top), dolny (Bottom), lewy (Left) lub prawy (Right) trzeba wpisać wartość (bez spacji).

— centrowanie tekstu — **CONTROL-C** (Center) — Start/End.

— początek akapitu — **CONTROL-P** (Paragraph).

— nowa strona — **CONTROL-T** (Top of form).

— przesunięcie w prawo — **CONTROL-R** (Right) — dotyczy tylko jednego wiersza i musi on być zakończony znakiem **RETURN**.

— numer strony — **CONTROL-N** (Number) — drukuje kolejny numer strony.

Kolejne polecenia są wpisywane przez naciśnięcie klawiszy **CONTROL-Z**, a następnie dopisanie bez spacji odpowiedniego znaku i ewentualnie wartości liczbowej:

— komentarz (.) — pomijany przy wydruku.

— odstęp 1/6" między wierszami (6) — standardowy.

— odstęp 1/8" między wierszami (8).

— definicja nagłówka strony (H — Header) — nagłówek może mieć długość trzech wierszy.

— definicja stopki (F — Footer) — także o długości do trzech wierszy.

— odstęp między wierszami druku (G) — normalnie 1.

— długość strony w wierszach (S — Size) — standardowo 66.

— wcięcie akapitu (P — Paragraph) — należy podać liczbę spacji dodawanych na początku akapitu i ewentualnie, po przecinku liczbę wierszy (np. sP4,2) — standardowo 5,1.

— lewy margines dla drugiej kolumny druku (L — Left).

— prawy margines dla drugiej kolumny druku (R — Right).

— ustalenie numeru strony (N — Number) — standardowo 1.

— justowanie do prawego marginesu (J — Just) — włączanie/wyłączanie (wyłączone).

— mikrospacje (X) — włącza i wyłącza dodatkowy odstęp między znakami o wielkości 1 punktu (wyłączone).

— zatrzymanie wydruku (W) — przerywa druk i wyświetla komunikat "Press START to continue".

— ustawienie tabulacji drukarki (T).

Dołączanie list z SynFile+

PaperClip pozwala na dodawanie do drukowanego tekstu pól z **SynFile**+. W tym celu należy najpierw wykonać przy pomocy opcji LISTS wyciąg z danych **SynFile'a** w postaci pliku dyskowego. W drukowanym dokumencie umieszczamy kolejno (bez spacji) znak **CONTROL-Z**, literę **M** i nazwę pliku do odczytu.

W odpowiednich miejscach dokumentu wpisujemy znaki **CONTROL-Z** i literę **m**. Podczas druku w tych miejscach będą drukowane kolejne pola danych odczytane w podanego wcześniej pliku.

Dołączanie grafiki

Możliwe jest także dodanie do drukowanego tekstu rysunków choć wymaga to pewnych przygotowań. Rysunek do druku należy wykonać jednym z niżej wymienionych programów i zapisać na dyskietce. Teraz z poziomu Basica uruchamiamy (poleceniem RUN"D:HIRESDMP.BAS") znajdujący się na dyskietce **PaperClip** program HIRESDMP.BAS.

Program ten najpierw pyta o typ drukarki (jeden z siedmiu). Naciskamy odpowiedni klawisz i odczytywany jest właściwy plik konfiguracji drukarki. Jeśli mamy inną drukarkę można utworzyć i zapisać na dyskietce własny plik.

Teraz wkładamy dyskietkę z rysunkiem i wybieramy format w jakim jest on zapisany. HIRESDMP rozpoznaje rysunki wykonane przy pomocy: Koala, Atari Light Pen, SynTrend (GRAPHICS 8), B/ /Graph, Fun With Art i Atari Paint. Odczytany rysunek można bezpośrednio wydrukować, ale nie to jest zasadniczym celem programu.

Po podaniu nazwy pliku rysunek jest zapisywany na dyskietce w postaci kodów dla drukarki. Można go teraz wydrukować nawet korzystając z funkcji COPY dowolnego DOS-u. **PeperClip** posiada natomiast specjalną funkcję kopiowania pliku dyskowego bezpośrednio na drukarkę podczas druku dokumentu.

Uzyskuje się to przez wpisanie znaku **CONTROL-Z**, litery **V** (Verbatim — dosłownie) i nazwy pliku do skopiowania. Jest on odczytywany z dyskietki i bez żadnych zmian przesyłany do drukarki.

Polskie litery

Autorzy programu przewidzieli nawet możliwość korzystania z innych zestawów znaków niż wbudowane w drukarce. Można więc w ten sposób uzyskać druk polskich liter. Trzeba w tym celu przygotować plik zawierający kody instalujące polskie litery w pamięci RAM drukarki. Wymaga to znajomości podstaw Basica i posiadania instrukcji drukarki.

Na początku redagowanego dokumentu należy umieścić znak **CONTROL-Z** z cyfrą **3**. Powoduje to zainicjowanie RAM drukarki, czyli przepisanie znaków z ROM do RAM. Następnie przy pomocy funkcji **Verbatim** kopiujemy plik z polskimi literami na drukarkę. W dowolnym miejscu dokumentu można teraz przełączać drukarkę ze znaków z ROM na znaki w RAM poprzez **CONTROL-Z** i **2** lub odwrotnie poprzez **CONTROL-Z** i **1**. Dzięki temu możliwe jest wykorzystywanie w tekście również znaków, których miejsce zajęły polskie litery.

Pozostaje mi tylko życzyć zadowolenia z tego wspaniałego programu, a myślę, że nikt nie będzie zawiedziony.

Marek Zachar

Adresy Wydawnictw

Czasopisma

Analog Computing Magazine
P.O.Box 625, Holmes, 19043 PA, USA
Antic Publishing (Antic, STart)
544 Second Street, San Francisco, 94107 CA, USA
Database Publications Ltd (Atari, User, Atari ST User)
Europa House, 68 Chester Road, Hazel Grove, Stockport SK7 5NY, G. Britain
U.K. Atari Computer Owners Club (Monitor)
P.O.Box 3, Rayleigh, Essex, G.Britain

Książki

Data Becker GmbH
Merowingerstrasse 30, 4000 Dusseldorf 1, BRD, tel. 0211/31-00-10
Markt & Technik Verlag
Hans-Pinsel-Strasse 2, 8013 Haar bei Munchen, BRD
Grosse Neugasse 28, 1040 Wien, Ostereich, tel. 0222/ 587-13-93
Sybex Verlag
Vogelsanger Weg 111, 4000 Dusseldorf 30, BRD, tel, 0211/618-02-20
2344 Sixth Street, Berkeley, CA 94710, USA, tel. (415) 848—8233
William Collins Sons & Co. Ltd.
8 Grafton Street, London W1X 3LA, G. Britain
Reston Publishing Co.
11480 Sunset Hills Road Reston, VA 22090, USA, tel, (800) 336-0338
Tronic Verlag
3444 Wehretal 1, BRD

Oprogramowanie

Activision
P.O.Box 6277, Mount View, CA 94039, USA
23 Pond Road, Hampstead, London NW3 2PN, G. Britain, 01-431 1101
Karlstrasse 26, 2000 Hamburg 76, BRD, tel. 040/220- -13-70
Ariolasoft
68 Long Acre, Covent Garden, London WC2E, 9JH, G.Britain, tel. 01-836 3411
Postfach 1350, 4830 Gutersloh 1, BRD
Artworx Software Co., Inc.
150 North Main Street, Fairport, NY 14450, USA, tel. (716) 425—2833
Batteries Included
30 Mural Street, Unit 9, Richmond Hill, Ontario L4B 1B9, Canada
Broderbund Software
1938 Fourth Street, San Rafael, CA 94901, USA, tel. (415) 456—6424
Datamost, Inc
8943 Fullbright Avenue, Chatsworth, CA 91311, USA, tel. (213) 709—1202
Datasoft, Inc.
9421 Winnetka Avenue, Chatsworth, CA 91311, USA, tel. (213) 701—5161
EPYX/Automated Simulations
1043 Kiel Court, Sunnyvale, CA 94089, USA, tel. (408) 745—0700
Electronic Arts
2755 Campus Drive, San Mateo, CA 94403 USA, tel. (415) 571—7171
Firebird
P.O.Box 49, Ramsey, NJ 07446, USA
First Star Software, Inc.
22 East 41st Street, New York, NY 10017, USA, tel. (212) 532—4666
Strategic Simulations Inc.
883 Stierlin Road, Mountain View, CA 94043, USA, tel. (800) 227—1617
Sublogic Software Corp.
713 Eaglebrook Road, Champaign, IL 61820, USA, tel. (217) 359—8482
Synapse Software
5221 Central Avenue, Richmond, CA 94804, USA, tel. (415) 527—7751
Thorn EMI
1370 Avenue of the Americas, New York, NY 10019, USA, tel, (212) 977—8990
Tronix, Inc.
701 W. Manchester Bivd., Inglewood, CA 90301, USA, tel. (213) 215—0529
US Gold
Units 2/3 Holford Way, Holford, Birmingham B6 7AX, G.Britain, tel. 021-356 3388
Xlent Software
P.O.Box 5228, Springfield, VA 22152, USA
Alumn Rock Road 514—516, Alumn Rock, Birmingham, G.Britain, tel. 021-327 6110
MMG Micro Software
P.O.Box 131, Marlboro, NJ 07746, USA, tel. (201) 431- -3472
MicroProse Software
10616 Beaver Dam Road, Hunt Valley, MD 21030, USA, tel. (301) 667—1151
Mirrorsoft Ltd.
Maxwell House, 74 Warship Street, London EC2A 2EN, G.Britain, tel. 01-377-4645
Optimized Systems Software
1221-B Kentwood Avenue, San Jose, CA 95129, USA
Origin Systems, Inc.
18100 Upper Bay Road, Houston, TX 77258, USA, tel. (713) 333-2539
Sierra On-Line
36575 Mudge Ranch Road, Coarsegold, CA 93614, USA, tel. (209) 683-6858

„ATARI MAGAZIN"

Jednym z wielu pism wydawanych w RFN i Austrii dla użytkowników Atari jest miesięcznik „Atari Magazin". Czasopismo to publikuje artykuły o wszystkim co jest związane z Atari. Zespół redakcyjny „Atari Magazin" pilnie śledzi wszystkie nowości techniczne w dziedzinie sprzętu i oprogramowania, uczestniczy we wszystkich ważniejszych wystawach komputerowych.

Czytelnik tego miesięcznika ma do dyspozycji kilka bardzo urozmaiconych tematycznie rubryk. Dużo miejsca przeznacza się tu dla listingów programów użytkowych i gier na Atari ST i Atari XL/XE. Dłuższe i ciekawsze programy są drukowane fragmentami w kilku kolejnych numerach. Znakomitą pomocą dla Czytelników potrafiących wymyśleć interesujący scenariusz jest cykl „Adventure-Editor", w którym przedstawiono szczegółowy opis i listing programu pozwalającego konstruować własne gry przygodowe na ST. Drugim bardzo aktualnym tematem są wirusy. Pisze się o nich prawie co miesiąc, ponieważ stanowią poważne zagrożenie dla społeczeństwa silnie związanego z komputerowym przetwarzaniem informacji. Obszerne artykuły ukazujące istotę tego zjawiska mają pomóc użytkownikom w uniknięciu strat spowodowanych zainfekowaniem dyskietki.

Wśród aktualności technicznych przewijają się doniesienia o transputerach — nowej generacji komputerów, których pierwsze modele zaprezentowano na targach CeBIT'88. Ale do rodziny Atari należą nie tylko transputery. „Atari Magazin" systematycznie opisuje szereg nowoczesnych urządzeń, które mogą się przydać każdemu użytkownikowi Atari. Są wśród nich drukarki laserowe, syntetyzatory dla muzyków, skanery i wiele innych.

Kilka stron w „Atari Magazin" jest zawsze zarezerwowanych na opisy najnowszych programów użytkowych i gier. Mnie szczególnie zainteresował pakiet oprogramowania „Mini Office II" na 8-bitowe komputery Atari. Zawiera on edytor tekstów, bazę danych, program kalkulacyjny. Czytelnicy mogą znaleźć tu również POKE, który umożliwi im pokonanie kolejnego etapu w ulubionej grze.

Sporo miejsca, jak w każdym zachodnim piśmie, zajmują ogłoszenia i reklamy. Zamieszczają je firmy, które chcą w ten sposób zachęcić do kupowania swoich wyrobów. Reklamy przynoszą korzyści również Czytelnikom „Atari Magazin". Prawie wszystko można kupić nie wychodząc z domu. Wystarczy tylko wypełnić i wysłać odpowiedni kupon z zamówieniem. Ogłaszają się także Czytelnicy pragnący sprzedać używany komputer, czy drukarkę lub wymienić doświadczenia.

„Atari Magazin" jest bardzo ładnie opracowany graficznie i zawiera dużo kolorowych ilustracji. Wszystkie publikowane w nim listingi można kupić również na dyskietce. Kolejne numery ukazują się zawsze bardzo regularnie, do 15 dnia miesiąca poprzedzającego datę wydania.

(j.j.)

„Atari Magazin", Verlag Ratz-Eberle, Melanchthonstr 75/1, 7518 Bretten, BRD, 124 strony, cena 7 DM.

PAMIĘCI DYSKOWE DLA KOMPUTERÓW

Atari 520ST jest obecnie na Zachodzie najpopularniejszym komputerem z rodziny Atari. Wydawnictwo Data Becker z Düsseldorfu i amerykańska firma Abacus, licząc się z dużym zainteresowaniem użytkowników, wydały całą serię książek o ST.

Jedną z ciekawszych pozycji w tej serii jest "ST Disk Drives: Inside and Out", której autorami są Uwe Braun, Stefan Dittrich i Axel Schramm. Książka ta pozwala od podszewki poznać budowę i funkcjonowanie pamięci dyskowych Atari ST.

Pierwszy rozdział zawiera definicje związane z tworzeniem baz danych, a więc pola danych, rekordu, zbioru sekwencyjnego i zbioru o dostępie bezpośrednim. Wykład teoretyczny jest zilustrowany przykładowymi programami napisanymi w Basicu, Pascalu, C i Fortranie.

Kolejny rozdział poświęcony jest strukturze dysku elastycznego. Dowiadujemy się z niego w jaki sposób są formatowane dyskietki, co to są ścieżki, sektory, tablica alokacji zbiorów i katalog zbiorów. Wiadomości te uzupełnione kilkoma ciekawymi procedurami, a wśród nich procedura pozwalająca sformatować dwie dodatkowe ścieżki na dyskietce.

W dalszej części omawiana jest stacja dysków elastycznych. Urządzenie to wykonane jest z ogromną precyzją i pozwala uzyskać niewyobrażalną wprost gęstość zapisu. Jeden bajt, czyli 8 bitów, zajmuje powierzchnię zaledwie 0.008 milimetra kwadratowego. Najważniejszym układem scalonym w 5,25" oraz 3,5" stacji dysków jest kontroler WD1772. Z tego względu autorzy bardzo dokładnie opisują wszystkie wyprowadzane z niego sygnały oraz wnikliwie analizują budowę rejestrów i sposób programowania. Przed napisaniem programu obsługującego dyskietkę należy poznać rozkazy sterujące pracą kontrolera. Jest ich w sumie 12 pogrupowanych w czterech zasadniczych typach. Realizują one funkcje pozycjonowania głowicy, zapisu i odczytu sektora lub ścieżki oraz obsługi przerwań. Zrozumienie działania stacji dysków byłoby trudne bez starannych czytelnych schematów blokowych ilustrujących czynności wykonywane podczas każdej operacji.

Użytkownicy Atari ST coraz częściej instalują do swojego komputera twardy dysk. Tak więc z pewnością znajdzie się wielu chętnych, którzy zapragną dowiedzieć się o nim czegoś więcej. Autorzy proponują im zaznajomienie się z komendami kontrolera twardego dysku SH204 oraz kilkoma interesującymi procedurami pozwalającymi poznać jego strukturę.

Dalszą część książki wypełniają listingi i opisy programów nadrzędziowych: programu ułatwiającego posługiwanie się RAMdyskiem, monitora dyskowego oraz wiele procedur maszynowych wzbogacających programy w Basicu.

(j.j.)

Uwe Braun, Stefan Dittrich, Axel Schramm, "ST Disk Drives: Inside and Out", Data Becker Verlag, 1987

INSTRUKCJE DO OPROGRAMOWANIA NA ATARI XL/XE

Nawet najlepszy program nie na wiele się przyda, jeśli nie dysponujemy instrukcją obsługi. Programy pisane na komputery osobiste zwykle zawierają bardzo rozbudowaną funkcję **HELP**, która w każdej chwili podpowiada użytkownikowi najtrafniejsze rozwiązanie. Gorzej jest w przypadku komputerów 8 bitowych. Pamięć 64 lub 128 KB pozwala bez ograniczenia możliwości samego programu jedynie na umieszczenie w **HELP** tylko najniezbędniejszych informacji.

W tej sytuacji bardzo cenną pomocą może okazać się książka „Instrukcje do edytorów tekstu i baz danych na mikrokomputer Atari". Opublikowano w niej opisy kilku popularnych programów użytkowych na Atari XL/XE: edytorów AtariWriter, AtariWriter Plus, StarTexter oraz baz danych The Home Filing Manager i SynFile+. W każdej instrukcji oprócz opisu komend znajduje się wiele cennych wskazówek pozwalających czytelnikowi w krótkim czasie mistrzowsko poznać tajniki redagowania tekstu i korzystania z bazy danych.

Równie cenną pomocą, tym razem dla tych, którzy interesują się samodzielnym pisaniem programów, są książki „Atari Basic XL" i „Atari Basic XE". Opisano w nich komendy, sposób posługiwania się interpreterem i uruchomiania obu tych programów. Nie są to podręczniki programowania, lecz materiał pomocniczy, który ułatwia dobranie najwłaściwszej instrukcji, informuje o jej parametrach, pozwala skorygować popełniony błąd. Basic XL i Basic XE mają znacznie większe możliwości niż Atari Basic, który jest standardowo instalowany w Atari XL/XE. Pozwalają one pewne problemy rozwiązywać w znacznie prostszy sposób. Mają one większą liczbę instrukcji, udoskonalony edytor, bardziej rozbudowane funkcje graficzne i znacznie szybszy interpreter. Cechy te czynią oba języki wygodnym narzędziem programowania dla komputera Atari.

Z wszystkich tych książek chętnie skorzystają użytkownicy traktujący Atari jako narzędzie pracy. Dzięki nim nie trzeba będzie poszukiwać instrukcji na giełdach.

(j.j.)

Ignacy Cukrowski, Marek Kuliński, Ludwik Piela, „Instrukcje do edytorów tekstu i baz danych na mikrokomputer Atari". Ośrodek Doskonalenia Kadr Rady Stołecznej NOT, Wyd. 1, Warszawa 1987,

Wiesław Migut, Mariusz Giergiel, „Atari Basic XE", Ośrodek Doskonalenia Kadr Rady Stołecznej NOT, Wyd. 1, Warszawa 1987

Wiesław Migut, Adam Stawowy, „Atari Basic XL", Ośrodek Doskonalenia Kadr Rady Stołecznej NOT, Wyd. 1, Warszawa 1987.

GAUNTLET XL/XE/ST

Tylko od Ciebie zależy czy podejmiesz wyzwanie — rzuconą rękawicę (ang. gauntlet). Musisz wiedzieć, że zadanie nie jest łatwe, gdyż czeka Cię (aż!) 512 poziomów fantastycznego świata, pełnego magii i przeciwności. Będziesz szukał skarbów w starych lochach i walczył ze strzegącymi ich potworami. Dla ułatwienia możesz zabrać ze sobą kolegę, z którym będziesz musiał ściśle i lojalnie współpracować.

Jeśli po takim wprowadzeniu jeszcze nie zrezygnowałeś, to wczytaj „Gauntlet" do komputera. Przywita Cię bardzo ładny tytułowy obrazek i nieco gorsza muzyka. W chwilę później pojawi się plansza wyboru opcji — gry jedno- lub dwuosobowej. Dalej będziesz (lub będziecie) mieli możliwość wcielenia się w jednego z czterech bohaterów: Thora, Thyrę, Merlina lub Questrona. Każdy z nich posiada inne przymioty. Jeśli wybrałeś grę z partnerem, zastanów się jakich dobrać bohaterów, by ich możliwości i siły uzupełniały się. Jest to ważne również dlatego, że obie postaci muszą zawsze znajdować się w polu widzenia.

Teraz kilka słów o każdym z bohaterów:

THOR — wojownik. Doskonale uzbrojony, bardzo słabo znający magię, posiadający mocną zbroję chroniącą od 20% ciosów, siła jego strzału jest ogromna — dwa razy większa od normalnej. Swoim toporem może niszczyć generatory wylęgające potwory.

THYRA — walkiria. Bardzo groźna. Jej zbroja jest dobrej jakości. Radzi sobie dobrze w walce, nie obawia się czarów. Tarcza chroni ją od 30% ciosów; strzela słabo. Mieczem może niszczyć generatory. Siła magiczna średnia.

MERLIN — czarownik. Nie nosi zbroi, jest słabo uzbrojony, ale wrogowie bardzo się go lękają. Jedynie on może zmierzyć się ze śmiercią... do czasu. Doskonale uzupełnia się z Thorem. Strzela dobrze, doskonale włada magią — może niszczyć prawie wszystkie potwory i generatory.

QUESTRON — elf. Walczy dobrze i obznajmiony jest z magią. Zbroja chroni go od 10% ciosów strzela słabo, nie jest w stanie zniszczyć generatorów, choć może sobie z nimi poradzić używając magii.

Wędrówka po labiryncie kończy się znalezieniem wyjścia do... następnego poziomu, które oznaczone jest dużą literą „E" w ramce. Zadanie z pozoru proste, ale jak już wspomniałem wcześniej, czekają Cię przeciwności oraz walka z wrogami. Spotkasz wiele groźnych potworów, które wylęgają się w generatorach porozrzucanych po całych lochach. Pamiętaj, aby jak najszybciej niszczyć generatory i później zabić potwory.

Najgroźniejszym przeciwnikiem jest śmierć (poznasz ją bez trudu) — może wyssać siły życiowe i zabrać do 200 pkt. na raz. Można ją pokonać wyłącznie przy pomocy magii.

W lochach możesz znaleźć przedmioty o szczególnych właściwościach.
Zbroja — wzmacnia odporność na ciosy.
Siła magiczna — wzmacnia efekt działania znalezionych rzeczy.
Strzały — zwiększą prędkość Twoich pocisków.
Tarcza — wzmacnia odporność na pociski wrogów.
Miecz — zwiększa zdolność walki wręcz.
Siła — zwiększa Twój „udźwig" 10 do 15 razy.
Żywność — wzmacnia siły życiowe (Health) o 100 jednostek i dodaje punkty. Uważaj jednak — niektóre jej porcje są zatrute.
Wino — można je niszczyć, ale spożycie daje taki sam efekt jak żywność.
Klucze — otwierają zamknięte drzwi i dodają 100 pkt. Możesz mieć ich ze sobą kilka.
Skarby — skrzynki z nimi nie mogą być zestrzelone, a za ich wzięcie dostajesz 100 pkt.
Amulet — zapewni Ci na krótko niewidzialność.
Transportery — czerwone kręgi, przenoszące gracza do najbliższego widocznego transportera.
Komnaty ze skarbami (Treasure Rooms) — pokazują się losowo i na ściśle określony czas. Zbiera się tam jak największą ilość skarbów i kosztowności. Nie można w nich przerwać gry przez wciśnięcie klawisza „SELECT".

Na początku otrzymujesz zapas „zdrowia" w wysokości 2000 jednostek. Ubywa go wraz z upływem czasu i w przypadku bezpośredniego kontaktu z wrogami — szczególnie ze śmiercią. Gracz I rzuca czary wciskając klawisz „Space Bar", gracz II klawisz od 0 do 9.

Jeśli gracze przez około minutę nie będą podejmowali walki, zamknięte drzwi otworzą się wypuszczając wszystkie potwory, lub ewentualnie wszystkie ściany zamienią się w przejście do następnego poziomu.

Gra jest starannie opracowana pod względem graficznym. Może nieco nużyć swymi 512 poziomami i nieco powolnym „poruszaniem" się bohaterów. Jeśli jednak przejdziesz wszystkie poziomy i będziesz czuł niedosyt, możesz dokupić program kolejnymi 512 poziomami. Program ten nosi tytuł „The Deeper Dungeon". Dajcie mi znać, jeśli przeszliście choćby pierwszą część „Gauntlet".

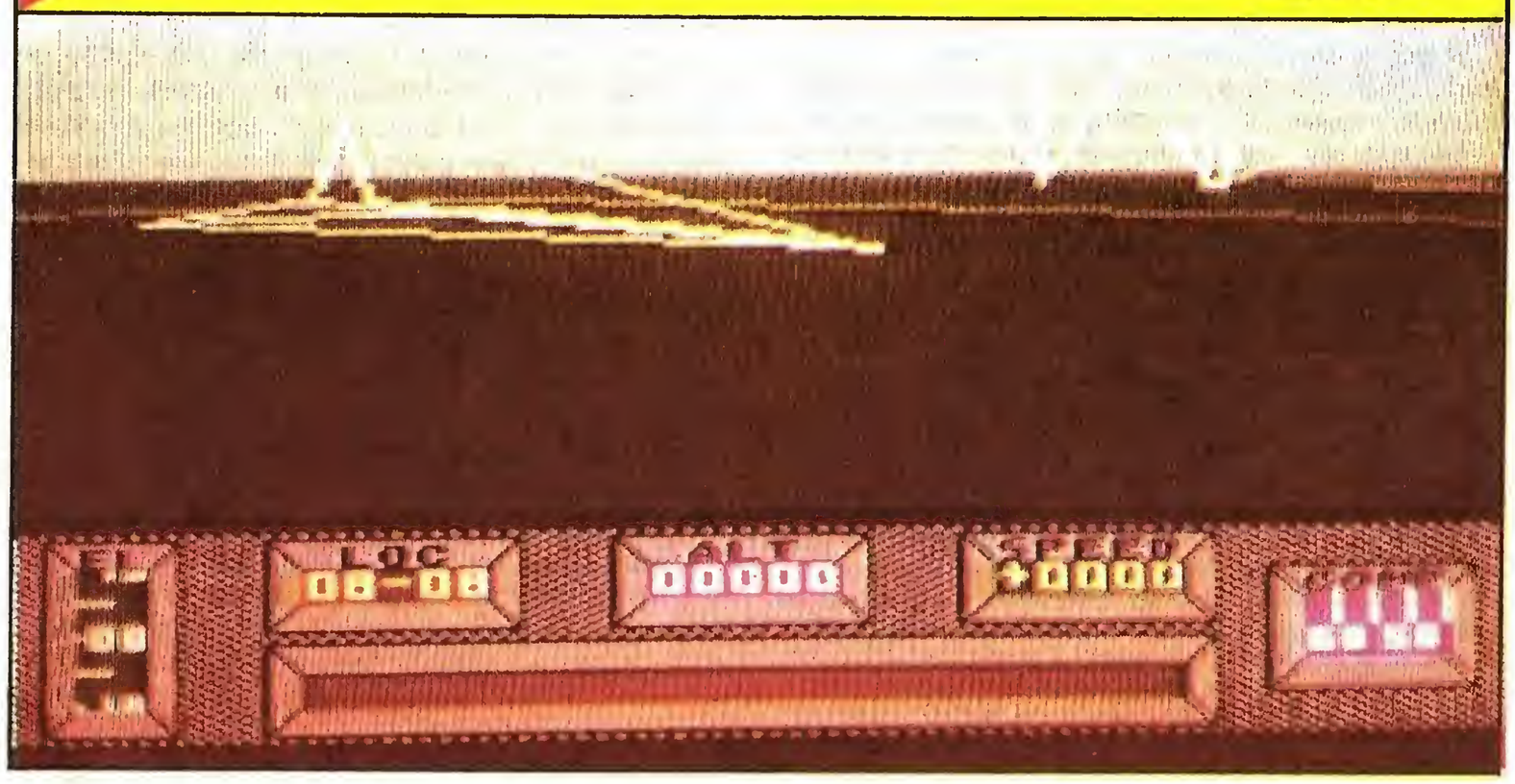
Ocena (w skali 1—10):

GRAFIKA	**8**
DŹWIĘK	**3**
SENS GRY	**5**

Producent: U.S. Gold
Autor: Bob Armour

(wist)

MERCENARY XL/XE



Stale rośnie liczba zwolenników myślenia przy komputerze, a nie tylko bezsensownego machania joystickiem. Dla nich właśnie przeznaczona jest gra „Mercenary". Program ten łączy w sobie cechy gry zręcznościowej, strategiczno-handlowej oraz symulacji komputerowej.

Podczas lotu do układu Gamma V w Twoim gwiazdolocie nastąpiła awaria układu sterowania. Po utracie przez automatycznego pilota kontroli nad statkiem przymusowo wylądowałeś na planecie Targ. Aby się z niej wydostać musisz zdobyć części do naprawy.

Zadanie nie jest łatwe, choć na początek dysponujesz skromną sumą 9000 kredytek (waluta międzygalaktyczna). Planeta Targ jest ogarnięta wojną domową między dwiema zamieszkującymi ją rasami: Palyarów i Mechanoidów. Jedna z nich opanowała powierzchnię planety, a druga jej wnętrze.

Zdobycie części do naprawy polega na umiejętnym handlowaniu z obiema walczącymi stronami oraz wynajmowaniu się w charakterze najemnika do walki. Oczywiście walka jest ostatecznością, gdyż uniemożliwia swobodny kontakt z przedstawicielami drugiej rasy.

Obecnie gra składa się z dwóch części: „Escape from Targ" („Ucieczka z Targ") i „Second City" („Drugie miasto"). Autor przewidział jednak możliwość rozbudowy programu przez wczytywanie, następnych części — ich liczba jest praktycznie nieograniczona. Tak więc każdy, kto ją ukończy będzie mógł zagrać ponownie w zupełnie innych warunkach.

Ekran monitora przedstawia widok pulpitu sterowniczego z najważniejszymi przyrządami i polem, w którym Twój podręczny komputer wyświetla rozmaite raporty. Powyżej widać to, co widać, czyli krajobraz miasta lub jego podziemi (zależnie od tego, gdzie przebywasz). Grafika jest dość prosta, lecz bardzo dobrze odwzorowuje wygląd miasta. Efekty dźwiękowe także są nieskomplikowane (szum silników, odgłosy uderzeń i wystrzałów), ale wystarczające i nie rozpraszają uwagi gracza.

W sumie gra jest bardzo pasjonująca, choć długotrwała. Nie trzeba jednak grać bez przerwy. Możliwe jest zapisanie aktualnej sytuacji na dyskietce i kontynuowanie gry następnego dnia. Dzięki temu można również rozważać różne warianty. W kłopotliwej sytuacji zapisujemy stan bieżący i gramy dalej. Jeżeli wybrane rozwiązanie jest błędne, to cofamy się poprzez odczyt zapisanej sytuacji i wypróbowujemy inny wariant. Daje to ogromną liczbę kombinacji.

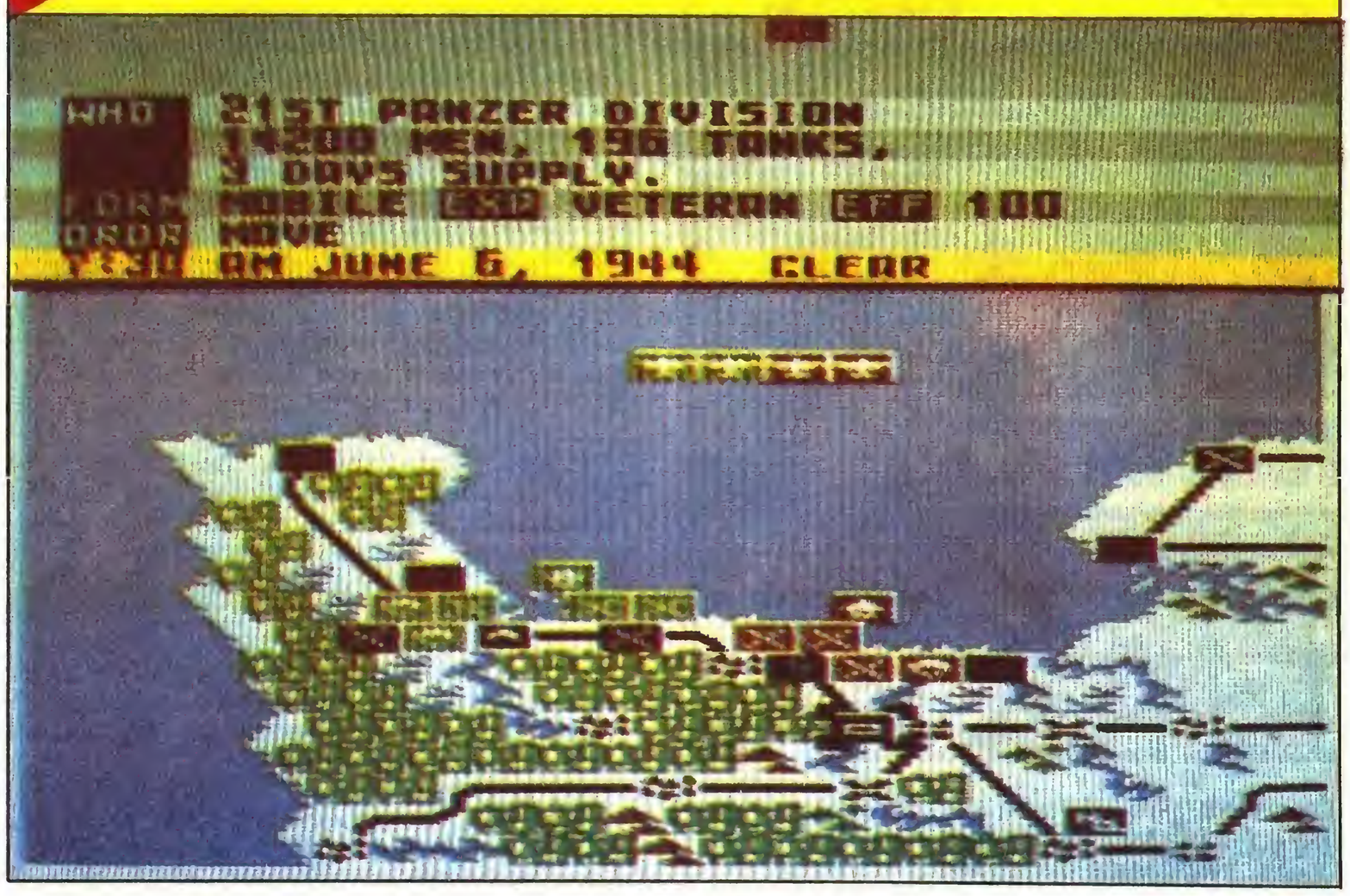
Ocena (w skali 1—10):

GRAFIKA	**6**
DŹWIĘK	**5**
REALNOŚĆ	**9**
SENS GRY	**8**

Autor: Paul Woakes
Producent: Novagen Software Ltd.

(ziew)

CRUSADE IN EUROPE XL/XE



„Krucjata w Europie” to popularne w Stanach Zjednoczonych określenie udziału wojsk amerykańskich w wyzwoleniu Francji spod okupacji niemieckiej podczas II wojny światowej. Jest to również tytuł gry, której autorem jest Sid Meier — twórca takich przebojów jak „Solo Flight”, „Kennedy Approach” i „Silent Service”. Taki autor jest już wystarczającą rekomendacją dla programu.

„Crusade in Europe” to gra strategiczna, przeznaczona dla jednego lub dwóch graczy, a jej tematem są walki o wyzwolenie Francji. Do wyboru jest pięć scenariuszy w kilku wariantach. Warianty te prezentują odmiany koncepcji strategicznych obu stron i zgodne są z rzeczywiście rozważanymi przez sztaby. Scenariusze prezentują kolejne fazy działań drugiego frontu w Europie: inwazja w Normandii (5 wariantów), marsz do Renu (2 warianty), operacja „Market-Garden” (2 warianty), kontrofensywa w Andenach (4 warianty) i bitwa o Francję (1 wariant). Ostatni scenariusz jest połączeniem trzech pierwszych.

Autor zadbał o zgodność z rzeczywistością. Podczas gry widoczne są tylko pobliskie jednostki przeciwnika oraz niektóre dalsze (wykryte przez wywiad lub rozpoznanie lotnicze). Dla każdej jednostki uwzględniona jest formacja, od której zależy ruchliwość oraz siła w ataku lub obronie. Na wartość bojową jednostki wpływa także doświadczenie (od zupełnie zielonych rezerwistów do elitarnych jednostek spadochronowych), morale, zmęczenie walką oraz poziom zaopatrzenia. Przy odpowiednio niskiej wartości bojowej jednostka może nawet zniknąć nagle, budząc zadziwienie gracza. Rozproszone i rozbite podczas walki oddziały, jeśli nie poniosły zbyt dużych strat, mogą się natomiast ponownie pojawić na polu walki.

Odczuwalny jest również wpływ pogody na efektywność lotnictwa, szybkość poruszania się wojsk oraz ich możliwości ataku i obrony. Nawet w nocy jest ciemno! Ponadto program uwzględnia działania lotnictwa strategicznego aliantów, które jest niewidoczne, lecz widać skutki nalotów. Zakłóca ten wspaniały obraz tylko niemożność wejścia dwóch jednostek na to samo pole planszy.

W odróżnieniu od innych gier tego typu „Crusade in Europe” nie jest podzielona na fazy, lecz toczy się ciągle. Gdy odejdziemy na kilka minut od komputera, możemy po powrocie zastać sytuację diametralnie inną. Może się nawet okazać, że nasze własne jednostki wykonują zupełnie inne rozkazy, niż otrzymały. Posiadają one pewną samodzielność i jeśli rozkaz jest niewykonalny lub nie ma go wcale, to podejmują decyzje zależne od sytuacji w najbliższym otoczeniu. Oczywiście możliwe jest zatrzymanie gry, jeśli chcemy zrobić przerwę. Przy zatrzymanej grze oddziały przyjmują jednak rozkazy i jest to dobry sposób na opanowanie zbyt szybko rozwijającej się sytuacji.

Pole gry jest oparte na siatce sześciokątnej, co pozwala na zachowanie proporcjonalnych odległości w różnych kierunkach. Orientacja w terenie jest bardzo ułatwiona przez zastosowanie żywych barw. Jednostki mogą być przedstawione jako symbole prostokątne lub graficzne (czołg, strzelec, samolot itp.). Bardzo śmiesznie wygląda dowóz zaopatrzenia dla walczących oddziałów (nocą) — po ekranie szaleją malutkie ciężarówki. Wygodne jest także sterowanie przy pomocy joysticka lub klawiatury — wybór zależy od preferencji użytkownika.

Muszę powiedzieć, że ta gra szybko zdobyła sobie moje uznanie i twierdzę, że lepszej nie było i długo jeszcze nie będzie. Radzę spróbować.

OCENA (w skali 1—10):
GRAFIKA 8
DŹWIĘK 6
REALNOŚĆ 9
SENS GRY 10
Producent: MicroProse Software
Autorzy: Sid Meier i Ed Bever.

(ziew)

LOCO XL/XE

Gier zręcznościowych jest tak dużo, że staje się kłopotliwe znalezienie nowego, oryginalnego tematu. Doskonale wybrnął z tego Russel Knight opracowując grę o jeżdżeniu lokomotywką.

Ukazująca się po uruchomieniu gry plansza podzielona jest na trzy części. W górnej znajduje się aktualny wynik i rekord, liczba lokomotyw oraz posiadany zapas węgla, który jest symbolizowany poziomą linią. W części dolnej widać plan torów, po których jedzie nasz pojazd. Na planie tym zaznaczone są także składy węgla (FUEL). W środkowym oknie znajduje się lokomotywa i tu można zaobserwować wszystkie obiekty. Okno to zawiera tylko mały fragment pokazanego niżej planu.

Zadaniem gracza jest przejechanie jak największej odległości. Sterowanie odbywa się przy pomocy joysticka. Tor jazdy wybiera się ruchami w górę i w dół. Prędkość lokomotywy jest stała, a ruch do przodu wywołuje strzał. Natomiast po naciśnięciu przycisku lokomotywa wypuszcza dym, który wznosi się dopóki przycisk jest wciśnięty. Wybierać trzeba taką drogę, aby przejeżdżając przez składy węgla uzupełniać zapas paliwa. Gdy go zabraknie, słychać „Bumm!!!” i lokomotywa rozlatuje się na kawałki.

Sama jazda nie stanowiłaby wystarczającej atrakcji, dodane są więc różne utrudnienia. Co kilka chwil nad lokomotywą przelatują samoloty (widać je wcześniej w dolnym oknie), które usiłują zbombardować nasz pojazd. Do zwalczania, zarówno samolotów, jak i zrzucanych przez nie bomb, służy dym! Trochę to dziwne, lecz widocznie taką autor miał fantazję. Od czasu do czasu nad lokomotywą pojawia się olbrzymi sterowiec. On również może i powinien — być zniszczony przez dym.

Równie uważnie trzeba obserwować dolne okno ekranu. Poza składami paliwa można tam zaobserwować poruszające się dziwne obiekty. Są to drezyny z dynamitem, które należy ominąć lub zniszczyć (joystick w prawo). Widać je także w środkowym oknie, lecz wtedy jest już przeważnie za późno na reakcję.

Gra toczy się przy dźwiękach pięknego standardu jazzowego „Chattanooga chu-chu” przerywanych gwizdami lokomotywy. Trzeba przyznać, że muzyka została zrobiona bardzo dobrze i dodaje programowi wiele uroku. Warto przed rozpoczęciem gry trochę tej melodii posłuchać.

Stwierdziliśmy podczas prób, że gra przeznaczona jest dla dwóch osób: kierowcy i pilota (a właściwie maszynisty i pomocnika). Jeden gracz trzyma joystick i obserwuje środkowe okno. Pomocnik informuje go natomiast o zauważonych w dolnym oknie składach paliwa, drezynach i samolotach. Dzięki takiej współpracy udało się przejechać spory kawałek drogi. Niestety później do gry usiadł dziewięcioletni chłopak, który samodzielnie zajechał dwa razy dalej. Być może miał rozdwojenie jaźni, bo mnie nie udawało się dokładnie obserwować dwóch okien jednocześnie.

OCENA (w skali 1—10):
GRAFIKA 6
DŹWIĘK 8
SENS GRY 6
Producent: Alligata Software
Autor: Russel Knight

(ziew)

CRUMBLE'S CRISIS XL/XE

Jako dozorca Intergalaktycznego ZOO nie spisałeś się wczoraj najlepiej. Nie dopełniłeś swoich obowiązków, zapominając zamknąć klatki z Fuzzami. Te, wykorzystując okazję, uciekły i skryły się w trudno dostępnych zakątkach krainy Multireverse. Dyrektor ZOO kategorycznie zażądał odnalezienia i odprowadzenia do klatek wszystkich Fuzzów. To zadanie należy oczywiście do Ciebie — jeśli go nie wykonasz, to w najlepszym razie grozi Ci „tylko" bezrobocie.

Do pomocy masz jedynie przenośny silnik odrzutowy. Aby z niego skorzystać wystarczy nacisnąć „fire". Przyda Ci się jeszcze pewna ręka i dobra koordynacja ruchów.

Zabawa składa się z pięciu poziomów, a na każdym z nich należy odnaleźć sześć Fuzzów. Samo zapakowanie ich do klatki nie jest kłopotliwe — nie walczą i nie stawiają oporu. Dotknięcie Fuzza wystarczy by znalazł się za kratkami. Aby nie zaprzątać głowy liczeniem już znalezionych stworów, na ekranie umieszczono sześć klatek, które zapełniają się sukcesywnie, kolejno odnalezionymi Fuzzami.

Należy grać bardzo ostrożnie, gdyż kontakty ze ścianą, jakimkolwiek z wielu złych duchów zamieszkujących Multireverse lub zbyt długie pozostanie w jednym pomieszczeniu kosztują Cię stratę cennej energii. Co prawda można uzupełniać jej zapas w specjalnych „punktach żywieniowych" (wyglądających na każdym poziomie inaczej), ale niestety autorzy programu nie pozwolili sobie na rozrzutność umieszczając ich bardzo niewiele. Przy okazji chciałbym zwrócić uwagę na nieruchomy rysunek zajmujący ok. 1/3 dolnej części ekranu, oraz na duże klatki umieszczone powyżej. Powoduje to, że fizyczne pole gry jest stosunkowo niewielkie i wymaga bardzo precyzyjnych ruchów joystickiem oraz silnie skoncentrowanej uwagi. Przy dłuższej grze oraz efektownie zrobionym przełączeniu obrazów (page flipping — polegającym na zwijaniu i rozwijaniu strony) jest to bardzo męczące.

Na plus można zapisać bardzo estetyczny rysunek pojawiający się po wgraniu programu i towarzyszącą mu muzykę, choć efekty dźwiękowe w samej grze są — powiedzmy — nieco ubogie.

Denerwujące jest również to, że nie wszystkie z sześciu Fuzzów są od razu dostępne na danym poziomie. Generują się pojedyńczo (losowo) dopiero po znalezieniu kolejnego, co stwarza konieczność zwiedzania całego labiryntu dla każdego Fuzza osobno.

Jest to jedna z tych gier, w których nie zabija się bezbronnych ufoludków, statków czy duchów, co — moim zdaniem — przemawia za nieco większym jej sensem niż znakomitej większości tych typu zabij-uciekaj. Choćby z tego powodu warto się nią pobawić.

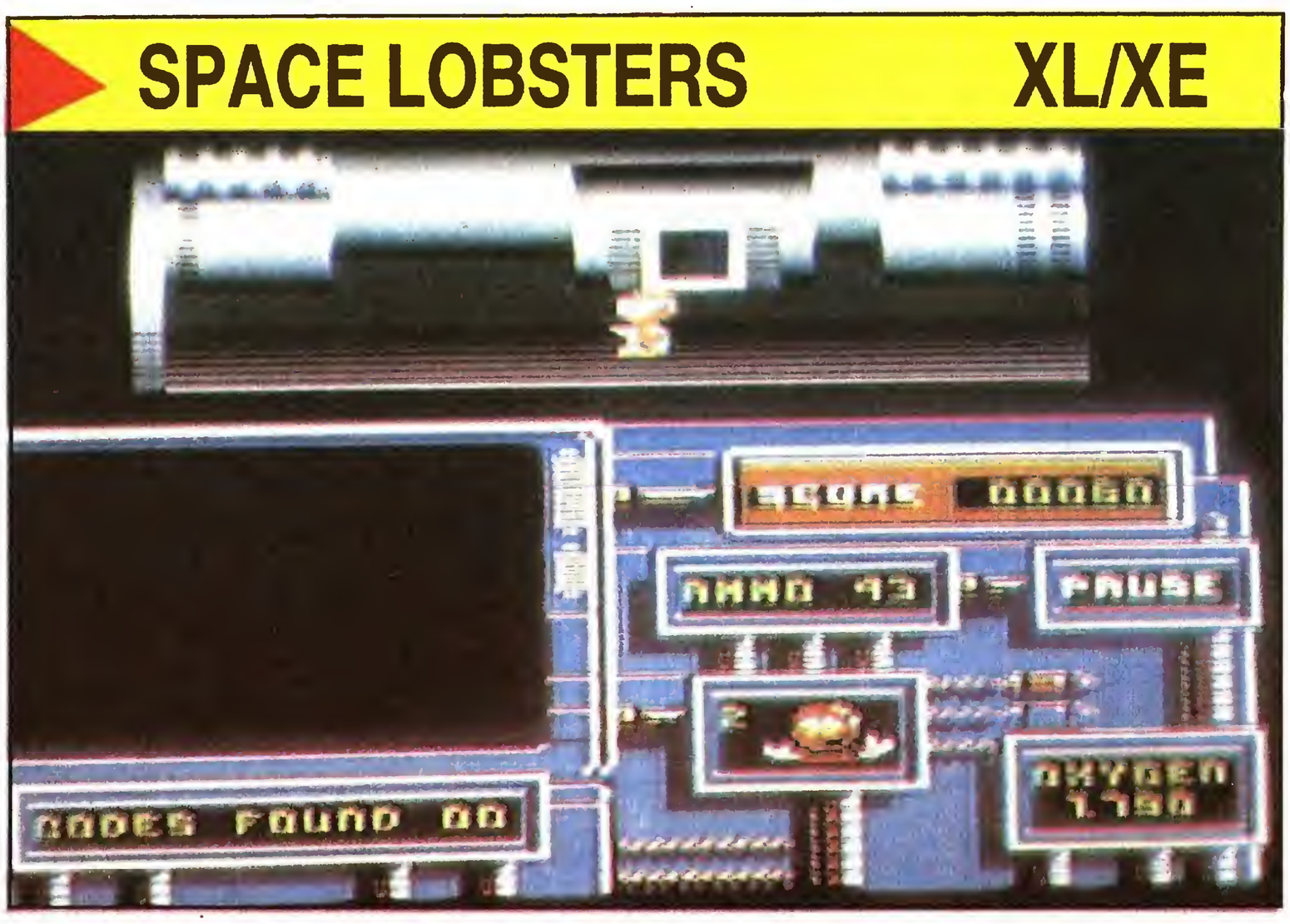
Ocena (w skali 1—10)

GRAFIKA	**8**
DŹWIĘK	**6**
SENS GRY	**8**

Producent: Red Rat Software
Autor: Ivan Mackintosh

(wist)

SPACE LOBSTERS XL/XE



Czy pamiętasz jak pełniłeś funkcję dozorcy ZOO w grze Crumble's Crisis? Masz teraz szansę objąć stanowisko dyrektora tej firmy — kapitana Crumble — wystarczy wczytać grę Space Lobsters.

Ostatnio postanowiłeś zdobyć kilka egzemplarzy, niezwykle rzadkich, długowłosych Fuzzów do swego zwierzyńca. Niestety, nie są one dostępne w twojej krainie (pamiętasz — Multireverse) i dlatego musisz odbyć długą podróż do swego przyjaciela, który mieszka w sąsiedniej galaktyce i zajmuje się hodowlą długowłosych Fuzzów.

Wsiadłeś w swój kosmiczny statek o dźwięcznej nazwie Colossus. Podróż miała się ku końcowi, gdy nagle popsuł się komputerowy system sterowania. Colossus dryfował w kosmosie, a ty zabrałeś się za poszukiwanie usterki. Odkryłeś, że aby szczęśliwie dolecieć do celu musisz znaleźć 10 kodów komputera pokładowego. Zadanie wydaje się łatwe, ale na takich kosmicznych rozbitków jak ty tylko czyhają przeróżne gnomy — takie kosmiczne stonki. Skorzystały natychmiast z okazji i opanowały cały twój statek. Kontakt z nimi jest oczywiście śmiertelny. Do obrony masz bardzo dobry pistolet laserowy typu LSP-15. Liczba pocisków jest ograniczona. Razem ze stworami do tego statku dostały się chmury radioaktywne, którym należy się kłaniać, gdyż są śmiercionośne. Jeśli do tego jeszcze dodać ciągle ubywający zapas tlenu, to okaże się, że gra jest wcale trudna.

Tak więc czeka cię ponad 150 ekranów doskonałej grafiki, w których poza wcześniej wspomnianymi przeciwnościami spotkasz specjalne kioski (małe kwadratowe okienka), w których będziesz mógł:

1. odkrywać kody (reveal code);
2. obejrzeć te które już znalazłeś (view codes)
3. dostać nowy zapas pocisków (buy ammo)
4. wyjść z kiosku (log off)

Aby wejść do kiosku należy stanąć równo pod okienkiem i wychylić joystick do góry. W okienku poniżej ukaże się zestaw opcji wspomnianych przed chwilą.

Do szybkiego przemieszczania się po statku służą teleportery — miejsca oznaczone dużą literą „T". Korzysta się z nich dokładnie tak samo jak z kiosków.

Ekran podzielony jest na dwie zasadnicze części. Górna — to pole gry — trzeba przyznać, że wygląda efektownie, ale ma tę zasadniczą wadę, że jest małe. Dzięki zastosowaniu tu grafiki 9 uzyskano doskonałe złudzenie obrazu trójwymiarowego. Dolna część jest nieruchoma, ale za to znacznie większa. Podawane są w niej ważne informacje: w lewym dużym oknie — opcje dostępne w kiosku, po prawej stronie — wynik (score), ilość broni, wskaźnik chwilowego przerwania gry (pause), ilość pozostałych żyć oraz ilość tlenu (oxygen). Grę można zatrzymać w dowolnym momencie przez naciśnięcie klawisza „OPTION"; powrót następuje po wykonaniu dowolnego ruchu joystickiem.

Grę wyposażono w ładny rysunek tytułowy (co w przypadku gier tej firmy jest już chyba regułą) oraz świetną muzyką, która mu towarzyszy. Animacja postaci stoi na niezłym poziomie. Efekty dźwiękowe są raczej takie sobie, ale myślę, że nie są potrzebne lepsze.

Amatorów tego typu gier nie brakuje. Powiem wam jeszcze dla pocieszenia, że w Wielkiej Brytanii cena tego programu wynosi: taśma 8, a dysk 10 funtów. Dokonajcie wstępnego przeliczenia i zapytajcie o cenę na najbliższej giełdzie komputerowej...

Ocena (w skali 1—10):

GRAFIKA	**9**
DŹWIĘK	**5**
SENS GRY	**5**

Producent: Red Rat Software
Autor: Ivan Mackintosh

(wist)

PMG GENERATOR

Przedstawiony program jest prosty w obsłudze i został przeznaczony dla osób słabo znających grafikę graczy i pocisków. Może spodoba się również czytelnikom bardziej zaawansowanym.

Generator składa się z trzech części. Edytor służy do zaprojektowania duszka, przy pomocy opisanych funkcji. Naciśnięcie innego, poza wymienionymi, klawisza spowoduje pojawienie się gwiazdki, symbolu bita. Zapisywacz umożliwi zapisanie danych duszka na kasecie. Użycie komendy listownia linii, spowoduje zapisanie samej linii w postaci DATA. Program, po wywołaniu, pokaże duszka na ekranie. Podprogram z ruchem umożliwia poruszanie duszka przy pomocy joysticka. Programy uzyskane przy pomocy tych opcji ładują się komendą ENTER „C:" Czwarta komenda zapisuje na taśmie dane rozkazem PUT. Definiator pozwala zmienić niektóre parametry ducha i ewentualnie skasować go. Na wykonanie niektórych opcji generatora trzeba chwilę poczekać.

Naciśnięcie RESET w dowolnym momencie spowoduje powrót do MENU głównego bez kasowania duszka. Program zajmuje ok. 10 KB.

Jakub Cebula

```
XY  0 REM PM/G GENERATOR
BQ  1 REM JAKUB CEBULA
XN  2 REM COPYRIGHT (C) BAJTEK
CF  10 ? CHR$(125):? "Momencik ...":POKE 5
    66,166:GOSUB 1830
NQ  20 REM PROC DEF
VJ  30 KEY=870:PLA=710:PMG=730:ODKOP=820
AG  40 L=144*256+1024+170
MV  50 REM DEKLARACJE
ZZ  60 DIM KE$(10):DIM DUCH(24,8),PODUCH(2
    4),MAG$(100),RKE$(100):COL=14:SZER=1
PR  70 REM BASIC ERROR
GI  80 FOR I=0 TO 24:FOR I1=0 TO 8:DUCH(I,
    I1)=0:NEXT I1:PODUCH(I)=0:NEXT I
LV  90 FOR I=L TO L+22:POKE I,0:NEXT I
MD  100 OPEN #6,12,0,"E:":POKE 764,12:LOCA
    TE 0,0,A
PG  110 GOTO 920
JX  120 REM EDYTOR PM/G
XO  130 GRAPHICS 0:POKE 559,0:? "
            EDYTOR"
SG  140 GOSUB ODKOP:GOSUB PMG
JX  150 POKE 752,1:POSITION 2,1:? "OKNO DE
    FINICJI";"          GRACZA"
UI  160 FOR I=16 TO 23:POSITION I,1:? "_";
    :POSITION I,23:? "¯";:NEXT I
MB  170 FOR I=2 TO 22:POSITION 0,I:? I-1:P
    OSITION 10,I:? PODUCH(I):POSITION 15,I
    :? "├":POSITION 24,I:? "┤":NEXT I
XY  180 RESTORE 190:FOR I=1 TO 8:READ MAG$
    :POSITION 25,I+2:? MAG$:NEXT I:MAG$=""
BS  190 DATA OPCJE:,ESC↑(GORA) - (-),ESC↓(DOL)
      - (=),ESC→(PRAWO)- (*),ESC←(LEWO) - (+),
    *(BIT)  - RET, (SPAC) - SPC
RA  200 DATA ESC(BSPAC)- DEL
AK  210 POSITION 25,13:? "ESC TO MENU"
OI  220 POSITION 25,15:? "!!! GRACZ !!!"
NQ  230 FOR I=16 TO 21:POSITION 25,I:? "!
                !":NEXT I:POSITION 25,22:?
    "!!!!!!!!!!!!!!!"
PB  240 REM OPCJE
KH  250 X=16:Y=2:POSITION X,Y:? "■"
FX  260 GOSUB KEY:X1=X:Y1=Y
XP  270 IF KE$="E" THEN 770
CJ  280 IF KE$="-" OR KE$="↑" THEN Y=Y-1:G
    OTO 330
IR  290 IF KE$="=" OR KE$="↓" THEN Y=Y+1:G
    OTO 330
FV  300 IF KE$="*" OR KE$=" " OR KL=155 OR
     KE$="→" THEN X=X+1:GOTO 330
EO  310 IF KE$="+" OR KE$="←" OR KE$="←" T
    HEN X=X-1:GOTO 330
II  320 X=X+1:PISZ=1
PP  330 IF X>23 THEN X=16
DO  340 IF KE$=" " OR KE$="←" THEN KAS=1
OH  350 IF X<16 THEN X=23
UJ  360 IF Y<2 THEN Y=22
ZQ  370 IF Y>22 THEN Y=2
AJ  380 IF KL=155 THEN PISZ=1
CU  390 PIS=0
YB  400 IF PISZ OR KL=155 THEN POSITION X1
    ,Y1:? "*"
NO  410 IF KL=32 OR KL=126 THEN POSITION X
    ,Y:? " "
IU  420 LOCATE X,Y,KON:IF KON<>32 THEN PIS
    =1
XA  430 IF PISZ THEN 450
NM  440 POSITION X1,Y1:? " "
BV  450 IF PISZ THEN BI=(X1-15)
EZ  460 IF KAS THEN BI=(X-15)
GA  470 IF  NOT KAS AND  NOT PISZ THEN 530
FA  480 IF DUCH(Y1,BI)=0 AND BI<>0 THEN DU
    CH(Y1,BI)=1
YD  490 IF KAS THEN KAS=0:DUCH(Y,BI)=0
VO  500 GOSUB 570
KD  510 POSITION 10,Y:? PODUCH(Y);"  "
ZI  520 GOSUB PLA
UV  530 PISZ=PIS:PIS=0:POSITION X,Y:? "■"
WO  540 BI=0:BIT=0:GOTO 260
CG  550 ? :? "NACISNIJ WYBRANA OPCJE"
SZ  560 REM PODUCH
MD  570 PODUCH1=0:FOR BI=1 TO 8:BIT=0
FZ  580 IF DUCH(Y,BI)=0 THEN 670
JC  590 IF BI=8 THEN BIT=1
IW  600 IF BI=7 THEN BIT=2
KF  610 IF BI=6 THEN BIT=4
NG  620 IF BI=5 THEN BIT=8
CT  630 IF BI=4 THEN BIT=16
AO  640 IF BI=3 THEN BIT=32
EN  650 IF BI=2 THEN BIT=64
XW  660 IF BI=1 THEN BIT=128
SQ  670 PODUCH1=PODUCH1+BIT
IR  680 NEXT BI
BG  690 PODUCH(Y)=PODUCH1
AV  700 PODUCH1=0:RETURN
LK  710 POKE L+Y,PODUCH(Y)
ZH  720 RETURN
BV  730 REM PMG
HY  740 POKE 54279,144:POKE 53277,3:POKE 5
    59,62
SZ  750 POKE 704,COL:POKE 53256,SZER:POKE
    53248,169
ZP  760 RETURN
QM  770 POSITION 0,0:? "          JESTES PEW
    NY ? (T/N)          "
GN  780 GOSUB KEY
BD  790 IF KE$="T" THEN POKE 53277,0:POKE
    559,34:GOTO 920
ZW  800 POSITION 0,0:? "                EDY
    TOR                 "
OW  810 GOTO 260
GL  820 REM ODKOP
LN  830 FOR I=1 TO 22:FOR I1=16 TO 23
XD  840 IF DUCH(I,I1-15)<>0 THEN POSITION
    I1,I:? "*"
GF  850 NEXT I1:NEXT I:IF DUCH(2,1)=1 THEN
     PISZ=1
ZQ  860 RETURN
EZ  870 REM KEY
ME  880 CLOSE #2:OPEN #2,4,0,"K"
TQ  890 GET #2,KL
UV  900 KE$=CHR$(KL)
ZH  910 RETURN
NM  920 POKE 53277,0:GRAPHICS 0:POKE 566,1
    66:REM MENU
XM  930 ? CHR$(125):POKE 710,0:POKE 752,1:
    DL=PEEK(560)+256*PEEK(561)
JP  940 POKE DL+6,7:POKE DL+7,7:? " PM/G G
    ENERATOR":? :? " (c)  Jakub Cebula SKA
    WINA  1988"
RF  950 L=144*256+1024+170
RC  960 ? :? "  START - EDYTOR"
NZ  970 ? :? "  SELECT- DEFINIATOR"
TA  980 ? :? "  OPTION- ZAPISYWACZ"
FQ  990 POKE DL+19,6:POKE DL+20,6:? :? :?
    :? " alium software"
OG  1000 IF PEEK(53279)=6 THEN 120
TO  1010 IF PEEK(53279)=3 THEN 1040
GQ  1020 IF PEEK(53279)=5 THEN 1590
MK  1030 GOTO 1000
AX  1040 GRAPHICS 0:POKE 752,1:? "
         ZAPISYWACZ  (ESC TO MENU)"
PK  1050 ? :? " L - listowanie lini data"
GC  1060 ? :? " P - listowanie podprogramu
    "
EW  1070 ? :? " A - listowanie programu z
    ruchem"
WB  1080 ? :? " B - zbior binarny"
SC  1090 ? :? " WYBIERZ OPCJE"
MI  1100 GOSUB KEY
KH  1110 IF KE$="L" THEN 1170
RB  1120 IF KE$="E" THEN 920
IX  1130 IF KE$="P" THEN 1310
ZK  1140 IF KE$="A" THEN RKE$="R":GOTO 131
    0
MB  1150 IF KE$="B" THEN 1550
NU  1160 GOTO 1100
FH  1170 ? CHR$(125):? "Podaj numer lini d
    ata"
GW  1180 POKE 764,255:TRAP 1190:INPUT LINI
    A:GOTO 1200
XW  1190 ? "↑↑":GOTO 1180
YQ  1200 ? :? "PRZYGOTUJ TASME NACISNIJ RE
    C I PLAY     A NASTEPNIE RETURN !!!"
ZZ  1210 GOSUB 1220:GOTO 1290
OK  1220 MAG$=STR$(LINIA)
FU  1230 MAG$(LEN(MAG$)+1)=" DATA "
FI  1240 FOR I=1 TO 22
EG  1250 MAG$(LEN(MAG$)+1)=STR$(PODUCH(I))
SP  1260 IF I<>22 THEN MAG$(LEN(MAG$)+1)="
    ,"
FO  1270 NEXT I
BB  1280 RETURN
KF  1290 GOSUB KEY:POKE 764,28:CLOSE #1:OP
    EN #1,8,0,"C:"
YE  1300 ? #1;MAG$:CLOSE #1:SOUND 0,0,0,0:
    SOUND 1,0,0,0:GOTO 920
CC  1310 ? CHR$(125):? "Podaj numer lini s
    tartowej (mniejszy  od 29900)"
FC  1320 POKE 764,255:TRAP 1330:INPUT LINI
    A:IF LINIA<29900 THEN 1340
UE  1330 ? "↑↑":GOTO 1320
EJ  1340 ? :? "PRZYGOTUJ TASME NACISNIJ RE
    C I PLAY     A NASTEPNIE RETURN !!!":G
    OSUB KEY
OV  1350 MAG$=STR$(LINIA)
QR  1360 MAG$(LEN(MAG$)+1)=" POKE 53277,3:
    POKE 559,62:POKE 54279,144:POKE 704,14
    :POKE 53248,100:POKE 53256,1"
WK  1370 POKE 764,28:CLOSE #1:OPEN #1,8,0,
    "C"
IW  1380 ? #1;MAG$
ZF  1390 LINIA=LINIA+1:MAG$=STR$(LINIA)
GE  1400 MAG$(LEN(MAG$)+1)="L=144*256+1024
    +100:FOR I=L TO L+21:READ X:POKE I,X:N
    EXT I:RESTORE "
GU  1410 MAG$(LEN(MAG$)+1)=STR$(LINIA+1)
IG  1420 ? #1;MAG$
MO  1430 LINIA=LINIA+1:GOSUB 1220
IM  1440 ? #1;MAG$
EG  1450 IF RKE$="R" THEN 1470
YF  1460 CLOSE #1:SOUND 0,0,0,0:SOUND 1,0,
    0,0:GOTO 920
SB  1470 LINIA=LINIA+1:MAG$=STR$(LINIA):MA
    G$(LEN(MAG$)+1)=" GOTO 30000":? #1,MAG
    $
GW  1480 CLOSE #1:SOUND 0,0,0,0:SOUND 1,0,
    0,0
ML  1490 RKE$=" "
GX  1500 POKE 764,28:LIST "C:",30000,30100
UG  1510 SOUND 0,0,0,0:SOUND 1,0,0,0
CO  1520 ? :? "UWAGA program w tej wersji
    nalezy      wczytywac do komputera popr
    zez         dwukrotna komende E.";
YZ  1530 ? CHR$(34);"C:";CHR$(34)
BJ  1540 ? :? "NACISNIJ DOWOLNY KLAWISZ":G
    OSUB KEY:GOTO 920
MU  1550 ? CHR$(125):? "USTAW TASME NACISN
    IJ REC I PLAY       NASTEPNIE  RETURN"
    :GOSUB KEY:POKE 764,28
QW  1560 OPEN #1,8,0,"C:":FOR I=2 TO 22:PU
    T #1,PODUCH(I):NEXT I:CLOSE #1
UY  1570 SOUND 0,0,0,0:SOUND 1,0,0,0
SI  1580 GOTO 920
SM  1590 GRAPHICS 0:GOSUB PMG:POKE 82,2:PO
    KE 83,39:REM      DEFINIATOR
CA  1600 ? CHR$(125):POKE 752,1:POSITION 2
    ,0:? "            DEFINIATOR  (ESC TO M
    ENU)"
BY  1610 POSITION 25,15:? "!!! GRACZ !!!"
QI  1620 FOR I=16 TO 21:POSITION 25,I:? "!
                !":NEXT I:POSITION 25,22:?
     "!!!!!!!!!!!!!!!"
LV  1630 POSITION 2,2:? "OPCJE:"
DN  1640 ? :? " C - zmiana koloru ducha":?
     :? " W - zmiana szerokosci"
SF  1650 ? :? " K - kasowanie ducha"
OK  1660 GOSUB KEY
HF  1670 IF KE$="C" THEN 1720
SD  1680 IF KE$="E" THEN 920
UZ  1690 IF KE$="W" THEN 1750
PK  1700 IF KE$="K" THEN 1770
SZ  1710 GOTO 1660
MD  1720 POSITION 8,16:? "Podaj nowy kolor
     (8-255)"
PH  1730 TRAP 1740:POKE 83,14:INPUT COL:PO
    KE 704,COL:POKE 83,39:GOTO 1600
BB  1740 ? "↑↑":GOTO 1730
KL  1750 TRAP 1600:POKE 83,23:POKE 82,0:?
    "podaj szer. (0,1,3)":INPUT SZER:IF SZ
    ER>3 OR SZER=2 THEN 1600
ZV  1760 POKE 53256,SZER:POKE 83,39:POKE 8
    2,2:GOTO 1600
ID  1770 ? :? "JESTES PEWNY ?N(T/N)":GOSUB
     KEY
BS  1780 IF KE$<>"T" THEN 1590
SH  1790 FOR I=0 TO 23:PODUCH(I)=0:FOR I1=
    1 TO 8:DUCH(I,I1)=0:NEXT I1:NEXT I:FOR
     I=L TO L+23:POKE I,0:NEXT I
OG  1800 GOTO 120
RM  1810 GOTO 1810
FJ  1820 END
ZI  1830 RESTORE 1850:FOR I=1536 TO 1590:R
    EAD A:POKE I,A:NEXT I
BC  1840 POKE 206,PEEK(88):POKE 207,PEEK(8
    9)+1:POKE 9,1:POKE 12,1:POKE 13,6:RETU
    RN
EC  1850 DATA 104,162,0,142,198,2,162,0,14
    2,197,2,160,26,185,8,6,145,206,200,192
    ,46,208,246,162,13
MB  1860 DATA 142,74,3,96,0,0,0,0,0,48,47,
    43,37,0,24,20,18,12,17,18,26,39,14,25,
    18,16,0,0,0,0
YT  30000 REM PROGRAM DOLACZANIA RUCHU
CI  30010 POKE 204,100:POKE 206,100:POKE 2
    07,148
GO  30020 RESTORE 30030:FOR I=0 TO 135:REA
    D X:POKE 1536+I,X:NEXT I:X=USR(1662):P
    OKE 54286,64
NY  30030 DATA 104,174,0,211,224,251,240,1
    04,224,247,240,89,138,41,1,201,0,240,3
    8,138,41,2,201,0,240,2,208,95
MK  30040 DATA 160,21,165,206,201,220,240,
    87,177,206,200,145,206,136,136,192,0,2
    08,245,169,0,145,206,230,206,24,76,83
KI  30050 DATA 6,160,1,165,206,201,10,240,
    58,177,206,136,145,206,200,200,192,22,
    208,245,169,0,145,206,198,206,24,138
ZX  30060 DATA 24,41,4,201,0,240,21,138,41
    ,8,201,0,240,3,76,123,6,230,204,24,165
    ,204,141,0,208,76,123,6
SE  30070 DATA 198,204,24,165,204,141,0,20
    8,76,123,6,76,98,228
IO  30080 DATA 104,160,1,162,6,169,7,76,92
    ,228
DL  30090 REM TUTAJ USTAW POWROT DO TWOJEG
    O PROGRAMU !!!
```

MODEM XM-301P

Dzięki uprzejmości P.Z. „Karen" prowadzącego serwis komputerów Atari otrzymaliśmy do testowania modem XM-301P. Modem ten jest sprowadzany z USA i w „KARENIE" przystosowywany do standardu polskiej sieci telefonicznej.

Atari **XM-301P** jest modemem galwanicznym, przyłączanym bezpośrednio do sieci telefonicznej (direct connect) i służącym do współpracy z komputerami Atari serii 800, XL i XE (poza 1200XL). Szybkość transmisji jest ustalona i wynosi 300 bodów (bitów na sekundę). Do modemu dołączone są dwie instrukcje (polska i angielska) oraz dyskietka z programem komunikacyjnym XE-Term.

BUDOWA I DZIAŁANIE

Cały modem zamknięty jest w obudowie z szarego tworzywa sztucznego o wymiarach: długość 135 mm, szerokość 75 mm i wysokość 40 mm. Do połączenia z komputerem służy standardowy wtyk złącza szeregowego Atari, zaś z siecią telefoniczną — wtyk telefoniczny Telos WT-4 (nawiasem mówiąc jest to jedyny polski element). **XM-301P** jest zasilany z komputera przez złącze szeregowe i nie posiada własnego zasilacza. Oba przewody połączeniowe są wyprowadzone z wnętrza bez pośrednictwa gniazd. Brak gniazd I/O zmusza do włączenia modemu jako ostatniego urządzenia w łańcuchu.

Wewnątrz obudowy umieszczona jest płytka drukowana, na której wyróżnia się serce modemu — mikroprocesor jednoukładowy 8048C. Ponadto na płytce znajdują się układy dopasowujące sygnał cyfrowy do standardu sieci telefonicznej oraz stabilizator napięcia. Elektronika jest więc mało skomplikowana, a całą pracę wykonuje w zasadzie mikroprocesor (właściwie mikrokomputer — ma on bowiem własną pamięć ROM i RAM).

Chociaż modem jest urządzeniem inteligentym, to nie może pracować samodzielnie. Niezbędny jest program, który steruje jego pracą. Program ten może być napisany w dowolnym języku (także w Atari Basic), a komunikacja z modemem jest prowadzona poprzez procedurę obsługi (handler) znajdującą się na dołączonej dyskietce.

Do praktycznego wykorzystania **XM-301P** można użyć programu XE-Term, który jest dostarczany wraz z modemem, lub dowolnego innego programu komunikacyjnego. Przy pracy z poziomu języka wyższego rzędu (np. Basic) dostępne są, następujące instrukcje: OPEN, CLOSE, GET, INPUT, PUT, PRINT i STATUS (w połączeniu z PEEK). Sterowanie pracą urządzenia odbywa się przy pomocy kodów sterujących (podobnie jak w przypadku drukarek). Występuje tu jednak zasadnicza różnica: ponieważ znak Escape (kod 27) może być przesyłany tak, jak wszystkie inne, to w celu odróżnienia polecenia od przesyłanej informacji należy zmienić zawartość rejestru CMCMD umieszczonego w pamięci RAM komputera (adres 7). Wymaga to użycia instrukcji POKE.

Jak wcześniej wspomniałem razem z modemem sprzedawana jest dyskietka. Tu ważna informacja: **XM-301P** NIE MOŻE współpracować z magnetofonem. Po pierwsze, wykorzystuje ten sam obszar pamięci co bufor magnetofonu (czyli „nie mamy armat"). Po drugie, seans łączności przy zapisie odbieranego pliku na magnetofon trwałby całe wieki. Po trzecie, tak naprawdę, to w cywilizowanych krajach nikt nie używa magnetofonu do komputera o pojemności pamięci przekraczającej 16 KB (Spectrum można tam obejrzeć w muzeum).

Systemowa dyskietka zawiera następujące pliki:

DOS.SYS
DUP.SYS
RAMDISK.COM
AUTORUN.SYS
HANDLER.OBJ
HANDLER.DOC

Pliki DOS.SYS, DUP.SYS i RAMDISK.COM są standardowymi plikami DOS 2.5 i nie będziemy ich omawiać. Zainteresowanych odsyłam do instrukcji DOS-u lub do opisu DOS 2.5 w pierwszym „Bajtku — Tylko o Atari".

AUTORUN.SYS jest właściwym programem XE-Term. Zapisanie go na dyskietce pod taką nazwą powoduje automatyczne wczytanie i uruchomienie programu po włączeniu komputera.

Plik HANDLER.OBJ zawiera — jak sama nazwa wskazuje — procedurę obsługi modemu. Po przepisaniu go na inną dyskietkę i zmianie nazwy na AUTORUN.SYS można wykorzystać go osobno. Właśnie dzięki temu możliwe jest napisanie własnego programu komunikacyjnego w dowolnym języku.

Dość dokładny opis działania tej procedury i oczywiście modemu zawarty jest w pliku HANDLER.DOC. Ma on niestety jedną wadę (poza tym, że jest po angielsku), mianowicie zajmuje ponad 57,5 KB, a więc nie można go wczytać w całości do żadnego edytora. Pozostaje jedynie skopiowanie na ekran lub drukarkę przy pomocy DOS-u, albo trzeba pogrzebać w dyskietce.

EKSPLOATACJA

Pierwszą rzeczą, która zaskakuje nabywcę, jest polska instrukcja, a dokładnie — jej okładka. Ponieważ na użytkowanie modemu trzeba posiadać zezwolenie (i to aż z samej Dyrekcji PPTT), to oczywiście niezbędne jest podanie do powyższego urzędu. I tu jest ta niespodzianka. Podanie zostało dołączone do instrukcji. Wystarczy odciąć ostatnią stronę okładki, wpisać imię, nazwisko, adres i numer telefonu (numer seryjny modemu już jest) oraz wysłać całość na adres, który także jest na podaniu. Całą procedurę załatwiania zezwolenia opisuje dokładnie Wojciech Zientara w „Bajtku" 3/89.

Porównując parametry techniczne modemu i polskiej sieci telefonicznej należałoby stwierdzić, że testowana była ta ostatnia, ale byłby to nadmiar złośliwości. Nie powinno się kopać leżącego, a nad Polską Pocztą wszyscy się już znęcają, przejdźmy więc do rzeczy.

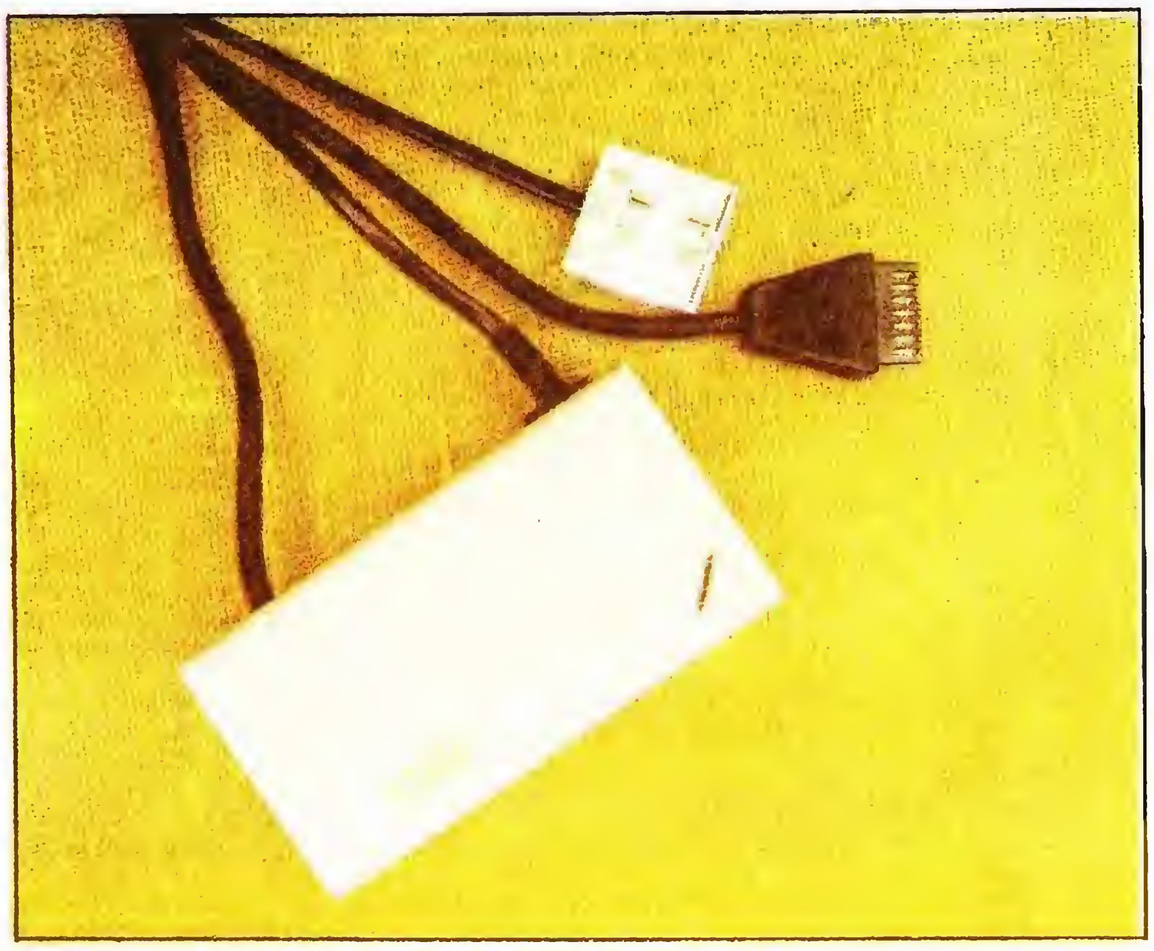
Praca **XM-301P** testowana była przy wykorzystaniu programów komunikacyjnych XE-Term, Pro*Term i HomeTerm, jak również z poziomu Basica w trybie bezpośrednim. Niestety działanie firmowego programu XE-Term okazało się zależne od dzielnicy i czegoś jeszcze (dokładnie nie udało się stwierdzić — może od pogody). Otóż często (zbyt często) przy rozpoczęciu komunikacji pojawia się napis „Carrier detected!" (nośna wykryta) przed wybraniem numeru abonenta! W takich przypadkach połączenie uzyskiwane było po wykonaniu kilku sztuczek z podnoszeniem słuchawki dołączonego równolegle telefonu i wybieraniem pierwszej cyfry numeru.

Najlepiej wybieranie numeru abonenta przeprowadzał program Pro*Term (Antic Publishing), zaś najgorzej robił to Home Term (Batteries Included). Nie, przepraszam najgorszy był Mini Office II, który uparcie twierdził, że modemu NIE MA! Także transmisja bezpośrednio z poziomu interpretera Atari Basic była poprawna. Jedyny kłopot sprawia w tym przypadku konieczność pamiętania o zmienianiu zawartości rejestru CMCMD. We wszystkich programach najwięcej zadowolenia sprawia automatyczne powtarzanie numeru, gdy nie zostanie uzyskane połączenie. Udało się mi połączyć w ten sposób z informacją o telefonach (cierpliwość komputera jest nieskończona).

Sama transmisja w obrębie Warszawy (dalej nie sprawdzaliśmy) przebiegała bez większych kłopotów. Rozmowa przez klawiaturę jest dość zabawna, choć trochę męczą się ręce. Także transmisja plików, zarówno tekstowych, jak i binarnych, odbywała się bez większych zakłóceń. Trwa jednak dosyć długo — 300 bodów to dwukrotnie wolniej niż z magnetofonu. Stosowany przy tym standardowy protokół XMODEM powtarza nadawanie bloku, jeśli został błędnie odebrany (do 10 razy). W praktyce takie powtarzanie bloku zdarzało się mniej więcej raz na 50 przesyłanych bloków.

PODSUMOWANIE

XM-301P jest niezawodnym urządzeniem i mam nadzieję, że okaże



się równie pożyteczny. Nie można jeszcze wykorzystywać go efektywnie z powodu braku w Polsce telefonicznych sieci komputerowych. Na razie najlepszym zastosowaniem jest przesyłanie plików tekstowych pomiędzy różnymi komputerami. Można w ten sposób przesłać tekst lub wyniki obliczeń (w postaci kodu ASCII) z posiadanego w domu Atari do znajdującego się w biurze IBM. Na razie nie zwalnia to jednak od podpisywania listy obecności lub odbijania karty zegarowej, a tego modem nie potrafi. Myślę jednak, że w miarę rozwoju komputeryzacji w naszym kraju upowszechnią się także sieci. Tymczasem poważną zachętą do nabycia tego urządzenia stanowi jego cena. Z przeliczenia ceny w Polsce na cenę w USA wychodzi, że dolar kosztuje 2450 zł (grudzień 1988).

Zalety:

— jest dostępny i ma homologację w Polsce (jedyny modem do komputerów 8-bitowych)
— brak oddzielnego zasilacza
— niewielkie wymiary
— niezawodna praca
— niska cena

Wady:

— mała prędkość transmisji
— kłopoty z wybieraniem przez XE-Term
— brak gniazda I/O

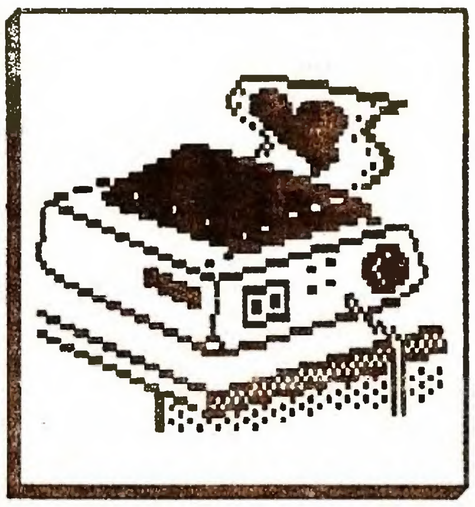
Marek Zachar

DANE TECHNICZNE

(w/g specyfikacji producenta)

Modulacja — kluczowanie częstotliwości (standard Bell 103)
Prędkość transmisji — 300 bodów
Przyłącze komputera — Atari SIO
Przyłącze telefoniczne — galwaniczne, w/g normy PPTT
Sygnalizacja nośnej — dioda LED
Sposób wybierania — dekadowe lub częstotliwościowe (DTMF)
Wybieranie — ręczne i automatyczne
Odpowiedź — ręczna i automatyczna
Kontrola akustyczna — głośnik monitora lub telewizora
Częstotliwość nadawania:
— modem wywołujący
znak 1270 ± 0,5%
spacja 1070 ± 0,5%
— modem odpowiadający
znak 2225 ± 0,5%
spacja 2025 ± 0,5%
— poziom transmisji
od −9 dBm do −16 dBm
Częstotliwość odbierania:
— modem wywołujący
znak 2225 ± 30 Hz
spacja 2025 ± 30 Hz
— modem odpowiadający
znak 1270 ± 20 Hz
spacja 1070 ± 20 Hz
— czułość odbioru
od −13 dBm do −46 dBm
— wykrycie nośnej
włączenie −44 dBm
wyłączenie −47 dBm
Błędy transmisji — mniej niż 1 na 100 000 bitów
Zasilanie — 5 V, 60 mA (z komputera)
Warunki pracy:
— temperatura 1—50°C
wilgotność 0—95% (bez kondensacji)
— wibracje 0,1 g (5—500 Hz)
— wysokość 0—4000 m

LISTY



Mam w planie zakup komputera Atari St. Chciałbym wam zadać kilka pytań, aby rozwiały się moje wątpliwości co do tego komputera:

1. Czy to prawda, że na ST z wbudowaną stacją dysków nie działa część programów?

2. Stacja SF314 daje możliwość zapisu i odczytu dyskietki z obu stron. Czy jednak w dyskietce 3,5 cala nie można wyciąć otworu zabezpieczającego zapis (tak jak to się robi z dyskietkami 5,25 cala) i używając stacji SF354 po prostu zmieniać strony dyskietki?

3. Czy wszystkie programy, które działają na monitorze monochromatycznym, nie będą mogły być używane na monitorze kolorowym i odwrotnie? Jeśli tak, to jakich programów jest więcej?

4. Czy w telewizorze kolorowym PAL można uzyskać (za pośrednictwem modulatora TV) rozdzielczość 640 na 200 punktów w czterech kolorach?

Tomasz Jurga
Warszawa

1. Nowe egzemplarze komputerów serii ST posiadają poprawiony system operacyjny. Dokonane w nim zmiany uniemożliwiają pracę programów, których autorzy nie stosowali się do zaleceń Atari Corp. Dlatego część programów nie działa z nową wersją systemu operacyjnego, natomiast część nowych programów może nie działać ze starą. Wbudowanie lub nie stacji dysków nie ma tu żadnego znaczenia. Posiadaną wersję systemu można rozpoznać wybierając wariant „Desktop Info" z menu „Desk". Jeżeli podany jest tam rok 1985, to mamy do czynienia ze starym systemem, późniejsza data oznacza nowy system.

2. Dyskietka 3,5 cala jest zbudowana zupełnie inaczej niż 5,25 cala. Przede wszystkim jest ona niesymetryczna i można ją włożyć do stacji tylko w jeden sposób. Z tego względu odczyt drugiej strony dyskietki jest możliwy tylko w stacji posiadającej dwie głowice (dla każdej strony oddzielna). Dyskietki o formacie dwustronnym nie mogą być więc odczytywane w całości przez stację SF354.

3. Istnieją programy, które są przeznaczone wyłącznie dla monitora monochromatycznego, inne zaś wyłącznie dla kolorowego. Jest także liczna grupa programów, które automatycznie rozpoznają rodzaj użytego monitora. Kolorowego monitora wymaga większość gier, natomiast programy użytkowe mogą pracować przeważnie z obydwoma rodzajami monitorów. Wydaje mi się, że najkorzystniejszym rozwiązaniem jest zakup monitora monochromatycznego, a dla programów wymagających koloru używanie telewizora PAL.

4. Telewizor kolorowy PAL pozwala na uzyskanie zarówno trybu niskiej, jak i średniej rozdzielczości, a więc wymienionego przez Pana. Jakość tego obrazu jest jednak nie najlepsza i praca w takim trybie jest dość męcząca (w grach nie przeszkadza to zupełnie).

Zepsuł mi się zasilacz do komputera 800XL. Jest on praktycznie nie do naprawy, bowiem w środku jest zalany żywicą. Na elektronice nie znam się zbyt dobrze, ale odkryłem, że przepaliło się uzwojenie pierwotne transformatora. Mam w związku z tym kilka pytań:

1. Czy Polska produkuje odpowiednik transformatora stosowanego w zasilaczu Atari?

2. Czy można wykonać zasilacz dający 5 V z części dostępnych w kraju i zasilać nim komputer?

3. Jaki prąd musi wytrzymać zasilacz?

4. Jak wyglądają połączenia we wtyczce?

5. Czy jakieś firmy sprzedają zasilacze do Atari?

6. Co by Pan zrobił w podobnej sytuacji?

7. Czy zepsucie zasilacza może być spowodowane przez komputer? Jak w takim przypadku sprawdzić, czy komputer jest dobry?

Marcin Bochenek
Kraków

Najlepszym wyjściem w takiej sytuacji jest zastosowanie nowego, oryginalnego zasilacza Atari. Niestety nawet w serwisie bywają z tym kłopoty i czasem zasilaczy brakuje. Miałem kiedyś podobny problem i wykonałem zasilacz samodzielnie. Można w tym celu wykorzystać dowolny schemat zasilacza stabilizowanego na napięcie 5 V i prąd 1,5 A (moc 7,5 VA). Oczywiście elementy mogą być polskiej produkcji. Połączenie styków we wtyczce zasilacza było podane w drugim „Bajtku — Tylko o Atari". Przypominam: +5 V — styki 1, 4 i 6, masa (0 V) — styki 3, 5 i 7 oraz ekran — styk 2. Wymiana samego transformatora jest kłopotliwa — polskie transformatory o takich samych parametrach mają znacznie większe wymiary. Przyczyną uszkodzenia zasilacza może być komputer, ale są to wyjątkowo rzadkie przypadki. Jeśli przepaleniu uległo uzwojenie pierwotne transformatora, to należy wykluczyć „winę" komputera. Przy okazji warto powiedzieć, że każdemu urządzeniu najbardziej szkodzi włączanie i wyłączanie. Należy więc wyłączać komputer jedynie wyłącznikiem, a zasilacz pozostawiać stale dołączony do sieci (przy pracy jałowej pobiera on minimalny prąd). Ponadto powinno się chronić sprzęt przed skokami napięcia w sieci zasilającej przy pomocy stabilizatora napięcia (stosowanego do kolorowych telewizorów).

Otrzymuję listy w sprawie programów żeglarskich opisanych w drugim „Bajtku — Tylko o Atari". W celu ich otrzymania należy na adres redakcji przesłać czystą dyskietkę. Przypominam, że niezbędne jest posiadanie stacji dysków i kolorowego telewizora lub monitora. Programy te NIE działają z magnetofonu.

PORT MIDI W ATARI ST

Cyfrowy interfejs dla instrumentów muzycznych, w który jest zaopatrzony każdy komputer Atari ST, z zewnątrz wygląda niepozornie — tylko dwa gniazda DIN na tylnej ściance komputera, lecz dzięki niemu powstała nowa dziedzina zastosowań dla komputerów osobistych.

Jest to łącze umożliwiające komunikację pomiędzy dowolnymi urządzeniami muzycznymi zaopatrzonymi w port MIDI (syntezatorami, sekwencerami, automatami perkusyjnymi itp.) oraz komputerami z zainstalowanym odpowiednim oprogramowaniem.

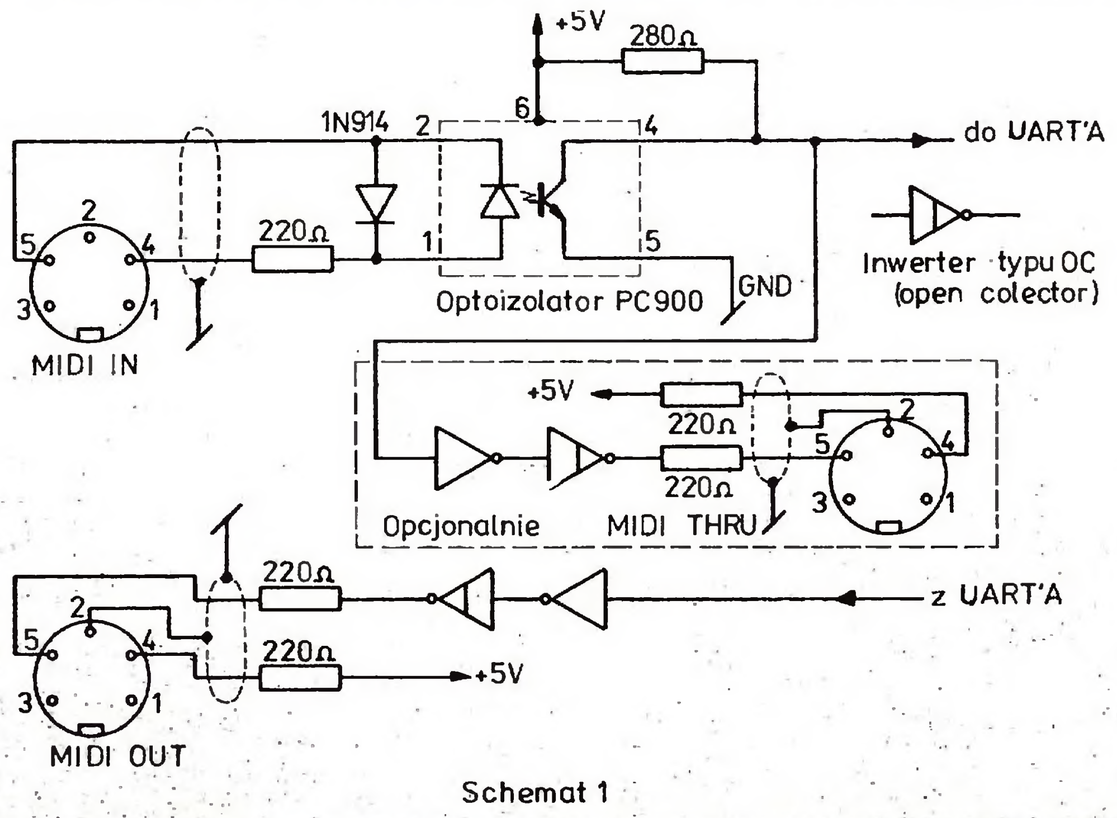
Przyjęcie tego standardu przez czołowych producentów elektronicznego sprzętu muzycznego miało miejsce w USA w 1982 roku i wywołało powstanie nowego, dynamicznie rozwijającego się już rynku specjalizowanego sprzętu MIDI (dostępne w sprzedaży są już m.in. gitara, saksofon, a nawet mikrofon wyposażone w MIDI). W tej chwili możliwości oferowane przez MIDI są już tak duże, że satysfakcjonują profesjonalnych muzyków — głównym ograniczeniem staje się wyobraźnia użytkownika). Zainstalowanie portu MIDI w komputerach Atari ST nie stanowiło żadnego problemu dla konstruktorów, a zaowocowało dużą popularnością tego komputera wśród muzyków — jest on używany jako narzędzie wybitnie usprawniające pracę kompozytorów i aranżerów. W chwili obecnej ocenia się, że ok. 8% sprzedanych komputerów ST, wykorzystywanych jest do celów muzycznych jako sterowniki MIDI.

Pomysł tego interfejsu był inspirowany interfejsem RS232C — zarówno RS, jak i MIDI, to interfejsy asynchronicznie szeregowe, zrealizowane w postaci „pętli prądowej". W odróżnieniu od RS 232 połączenie pomiędzy urządzeniami może być wykonane za pomocą standardowego przewodu stereofonicznego zaopatrzonego w pięcioszpilkowe wtyki DIN. Długość kabla połączeniowego nie powinna przekraczać 15 m ze względu na możliwość indukowania się zakłóceń. Według opisu standardu, w 5-szpilkowym gnieździe DIN poszczególne nóżki wykorzystano następująco:

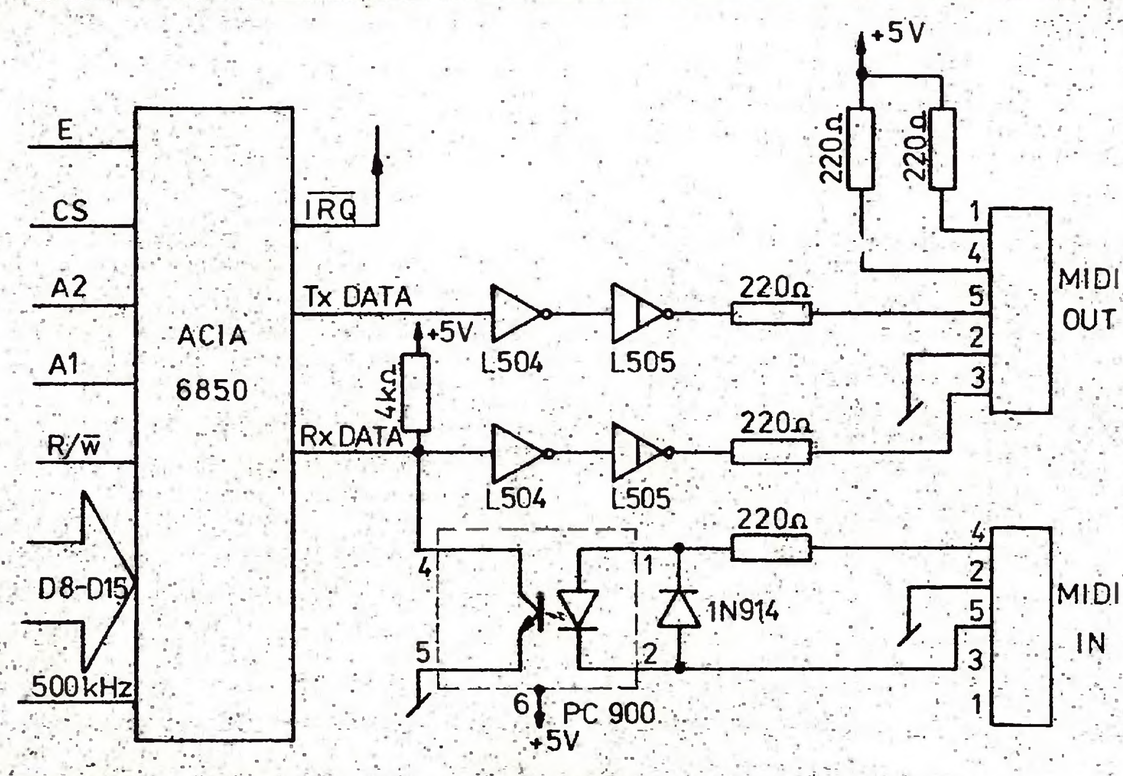
— nóżki 4 i 5 — przyłączenie pętli prądowej;
— nóżka 2 — masa poszczególnych urządzeń;
— nóżka 1 — sygnał start/stop dla sekwencerów i automatów perkusyjnych;
— nóżka 3 — sygnał taktujący (24 uderzenia na ćwierćnutę);

Informacja przesyłana jest pętlą prądową w ten sposób, że „1" logiczna to brak prądu, zaś „0" logiczne sygnalizowane jest obecnością prądu w pętli. Znamionowy prąd wystarczający do wysterowania odbiornika ustalono na 5mA. Każdy bajt informacji poprzedzany jest bitem startu (logiczne „0") i kończony bitem stopu (logiczna „1"). Szybkość transmisji określono równą 31.250 bitów/sekundę z tolerancją +/−1%. Duże podobieństwo w sposobie realizacji transmisji wynika z przyczyn ekonomicznych — dzięki temu do wykonania portu MIDI można użyć specjalizowanych układów typu UART (Universal Asynchronous Receiver — Transmitter) szeroko wykorzystywanych do obsługi portu RS 232C i powszechnie dostępnych. Zalecany przez standard sposób podłączenia portu MIDI jest pokazany na rysunku 1.

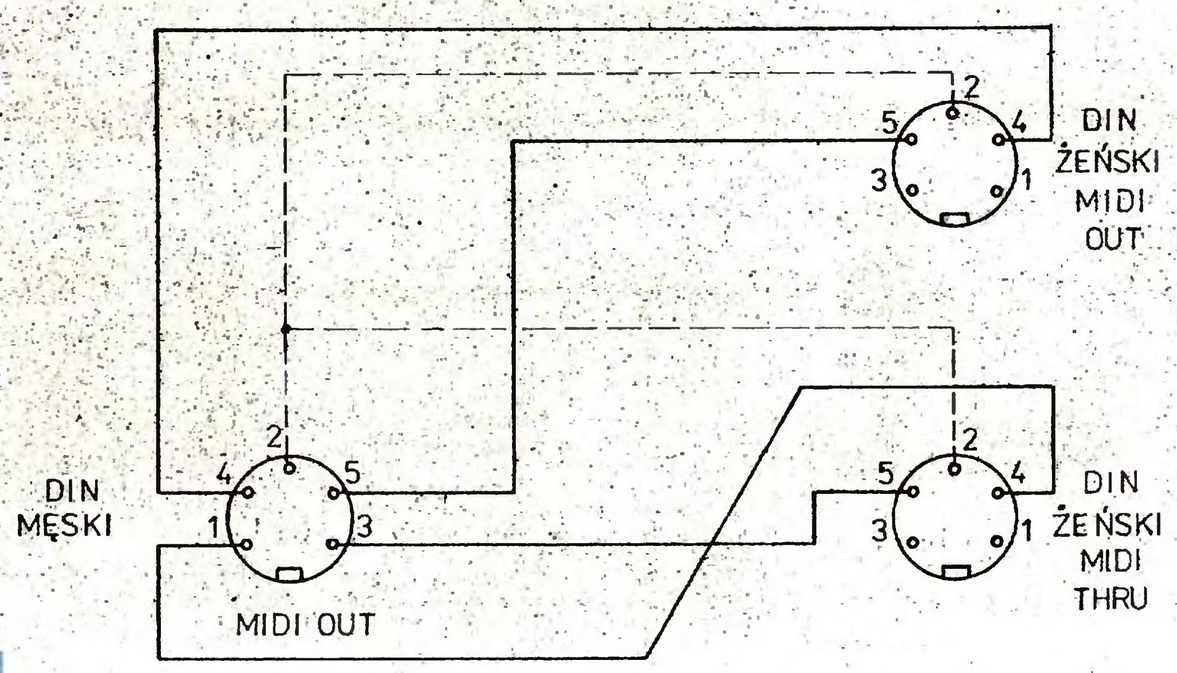
W charakterze elementu sprzęgającego zastosowano optoizolator (transoptor), który zapewnia galwaniczną izolację pomiędzy współpracującymi urządzeniami. Specyfikacja standardu określa nawet jego typ — jest to PC900 firmy Sharp. Według autorów standardu używanie innych typów, niż zalecany może spowodować problemy przy współpracy z różnymi typami syntezatorów. Wyjście MIDI THRU zastosowano w celu umożliwienia łączenia większej liczby urządzeń bez konieczności wykonywania zawodnych połączeń na kablach (MIDI



Schemat 1



Schemat 2



Schemat 3

umożliwia jednoczesną komunikację między szesnastoma urządzeniami połączonymi ze sobą). W przypadku dłuższych kabli i większej liczby współpracujących urządzeń MIDI THRU zapewnia regenerację słabszego sygnału.

Rysunek 2 pokazuje realizację interfejsu w Atari ST. Uważny czytelnik zauważył zapewne, że w odmienny sposób wykonano wyjście przejściowe MIDI THRU. Zostało ono (najprawdopodobniej ze względu na brak miejsca na tylnej ściance komputera) wkomponowane w wyjście MIDI OUT. W praktycznych zastosowaniach tą niedogodność można obejść wykonując złącze z rozdzielonymi gniazdami MIDI OUT i MIDI THRU według rysunku 3.

Po tej dygresji praktycznej powróćmy do opisu działania interfejsu. Jak widać to na rysunku 2, do obsługi portu MIDI w komputerze ST użyty został układ 6850 ACIA (Asynchronous Comunication Interface Adapter) z rodziny układów firmy Motorola zaprojektowanych do mikroprocesora 6800. Jest to układ tani i powszechnie dostępny. Jako element sprzęgający wykorzystano zgodnie z zaleceniami optoizolator PC900. Inwertery 74LS05 zapewniają wysterowanie prądowe wyjścia MIDI OUT i MIDI THRU, zaś inwertery 74LS04, odpowiednią fazę sygnału wyjściowego.

A oto pokrótce jak działa ACIA: Układ 6850 korzysta z sygnału zegarowego 500 kHz, który jest 16 razy większy od szybkości transmisji w MIDI (31.250 bitów/s) — umożliwia to podział każdego odcinka czasu odpowiadającego jednemu bitowi informacji na 16 krótkich odcinków. Dzięki temu układ ACIA ignoruje fałszywe bity startu powodowane przez krótkie impulsy zakłócające.

Zmiana poziomu sygnału z wysokiego na niski uruchamia licznik, który opóźnia próbkowanie sygnału o połowę czasu trwania pojedyńczego bitu. Wówczas zakłada się, że zniknął przejściowy impuls zakłócający i jeżeli poziom sygnału jest nadal niski, uważa się że bit startu jest ważny. Następnie układ powtarza cykl zegara zliczając impulsy przez okres pełnego bitu, aż do połowy odcinka czasowego pierwszego bitu danych i dokonuje próbkowania sygnału. Powyższa procedura jest powtarzana dla pozostałych bitów.

W ten sposób następuje weryfikacja sygnału odbieranego z gniazda MIDI IN. Po wydzieleniu bitu startu — bajt danych jest kolejno, bit po bicie ładowany do rejestru przesuwającego z wyjściami równoległymi. Cykl odbioru bajtu danych kończy się po załadowaniu do rejestru ostatniego ósmego bitu i jest potwierdzany odebraniem bitu stopu. Wówczas ACIA wysyła w kierunku mikroprocesora sygnał „żądanie przerwania" (Interrupt Request). Odebrany bajt danych jest dostępny w postaci równoległej w wyjściowym rejestrze układu ACIA i odczytywany przez mikroprocesor po przejściu do programu obsługi przerwania.

Nadawanie informacji do wyjścia MIDI OUT jest prostsze. Zewnętrzny zegar 500 kHz doprowadzony do układu 6850 po podzieleniu przez 16 w wewnętrznym dzielniku do częstotliwości 31.250 Hz jest wykorzystywany do konwersji bajtu przesyłanego do wyjścia MIDI OUT z postaci równoległej (w tej postaci bajt jest pobierany z szyny danych mikroprocesora) na postać szeregową. Wewnętrzny rejestr układu ACIA jest ładowany bajtem danych z szyny danych i w takt impulsów o częstotliwości 31.250 Hz jego zawartość jest przesuwana — na wyjściu pojawiają się kolejne bity informacji w postaci szeregowej, które po dodaniu bitów startu i stopu są wysyłane do wyjścia MIDI OUT.

Dostęp do portu MIDI w ST ma przyporządkowany poziom 6. Ambitny użytkownik oraz programista nie musi pisać swoich własnych podprogramów obsługi portu MIDI — w przypadku Atari ST dostępne są odpowiednie procedury BIOS-a i XBIOS-a:

- procedury BIOS-a:
  - bcinstat (sprawdzanie statusu wejścia);
  - bconin (pobranie znaku z wejścia);
  - bconout (wysłanie znaku na wyjście);
  - bcostat (sprawdzenie statusu wyjścia);
- procedury XBIOS-a:
  - midiws (wpisanie ciągu znaków na wyjście MIDI);
  - mfpint (inicjalizacja procedury obsługi przerwania);
  - iorec (przygotowanie bufora do odbioru ciągu znaków);
  - jdsint (zablokowanie przerwania).

Procedury te są wywoływane poprzez instrukcję TRAP # 13 (BIOS) oraz instrukcję TRAP # 14 (XBIOS).

I tutaj, a może już nieco wcześniej kończy się domena sprzętu, a zaczyna się królestwo oprogramowania, które dane przekazane przez port MIDI potrafi przetworzyć praktycznie w dowolny sposób, a efekt końcowy zadowoli nawet bardzo wymagającego muzyka. Ale to już temat na oddzielny artykuł.

Tadeusz Pękacz

ATARI 520 ST

Kilkakrotnie opisywaliśmy już w „Bajtku" komputery Atari ST. Tym razem zostawmy więc sprawy techniczne, a zajmijmy się stroną praktyczną.

Z licznej już rodziny ST wybraliśmy komputery 520STM i 520STFM oraz stacje dysków SF314 i Cumana. Oba komputery posiadają 512 KB pamięci RAM oraz modulator TV. Ponadto STFM ma wbudowaną dwustronną stację dysków, ma także nową wersję ROM. Obie dodatkowe stacje są także dwustronne. Poza tym teoretycznie nie ma żadnych innych różnic.

Jest to jednak tylko teoria. Po zmontowaniu systemu gołym okiem widoczna jest podstawowa różnica. Nieco większy komputer 520STFM zajmuje znacznie mniej miejsca, gdyż oprócz stacji ma w środku również zasilacze. Mamy więc jedno większe pudełko zamiast czterech mniejszych (zob. rys.). Ponadto w STFM mamy dwa przewody: do sieci i do monitora. Natomiast w zestawie STM są przewody: komputer-monitor, komputer-stacja, komputer-zasilacz, stacja-zasilacz oraz dwa przewody zasilacz-sieć (dla komputera i stacji) — razem sześć. Dodatkowo w obu przypadkach występuje przewód zasilający monitora. Jest to już gmatwanina kabli, w której łatwo się zaplątać. Zmniejsza ją nieco zastosowanie stacji Cumana, która posiada wbudowany zasilacz i mimo to ma mniejsze wymiary.

W użytkowaniu oba modele komputerów są podobne. Podobne, ale nie jednakowe — poza stacją różnią się jeszcze położeniem wyłącznika i przycisku RESET. Często zdarzało mi się „obmacywanie" tylnej ścianki w poszukiwaniu przycisku, który znajdował się akurat po przeciwnej stronie. Zdarza się także poszukiwanie stacji lub próba włożenia dyskietki do komputera 520STM. Na szczęście użytkownik korzystający z jednego modelu nie będzie miał tych kłopotów.

Niektóre programy wymagają umieszczenia joysticka w pierwszym gnieździe. Trzeba wtedy wyjąć z tego gniazda wtyczkę myszy. W modelu STM jest to czynność bardzo prosta, lecz w STFM gniazdo jest umieszczone we wnęce pod spodem komputera i dostęp do gniazda wymaga odwrócenia komputera. Ponadto wnęka jest wąska, co wcale nie ułatwia tej operacji.

Korzystanie ze wszystkich stacji (SF314, Cumana i wbudowana w STFM) jest także podobne. Daje się tu we znaki brak sygnalizacji zasilania w stacji SF314 — zdarzało się, że pozostała włączona całą noc. Cumana ma w tym celu podświetlony wyłącznik zasilania, lecz jest on umieszczony z tyłu stacji i słabo widoczny. Szybkość pracy wszystkich stacji jest bardzo podobna. Cumana ma jednak czasem trudności z odczytem dodatkowych (o numerach ponad 80) ścieżek na dyskietkach. Natomiast można jej użyć także do współpracy z Commodore Amiga, dla zwykłego użytkownika zaleta ta jest wszakże zupełnie zbędna. Ma za to wadę, która nie pozwala na zastosowanie drugiej stacji — brak gniazda do jej przyłączenia. Na obudowie jest nawet informujący o tym napis „Drive B".

Wyjaśnienia wymaga jeszcze sprawa różnych wersji systemu operacyjnego. Rzeczywiście część programów nie chce działać z jego nową wersją. Nie jest to dużym problemem, ponieważ zawsze można znaleźć podobny program, który działa. Ponadto większość starych programów pojawia się w wersji przystosowanej do nowego systemu. Nie należy się spodziewać, aby nowe programy miały także wersje dla starego systemu. Przy okazji stwierdziłem, że pewna grupa programów — szczególnie użytkowych — nie chce działać niezależnie od wersji komputera. W tym przypadku przyczyną jest zbyt mała pamięć operacyjna. Standardowym jej rozmiarem jest obecnie 1 MB. Przy ewentualnym zakupie komputera warto trochę poczekać, więcej zaoszczędzić i kupić od razu 1040ST.

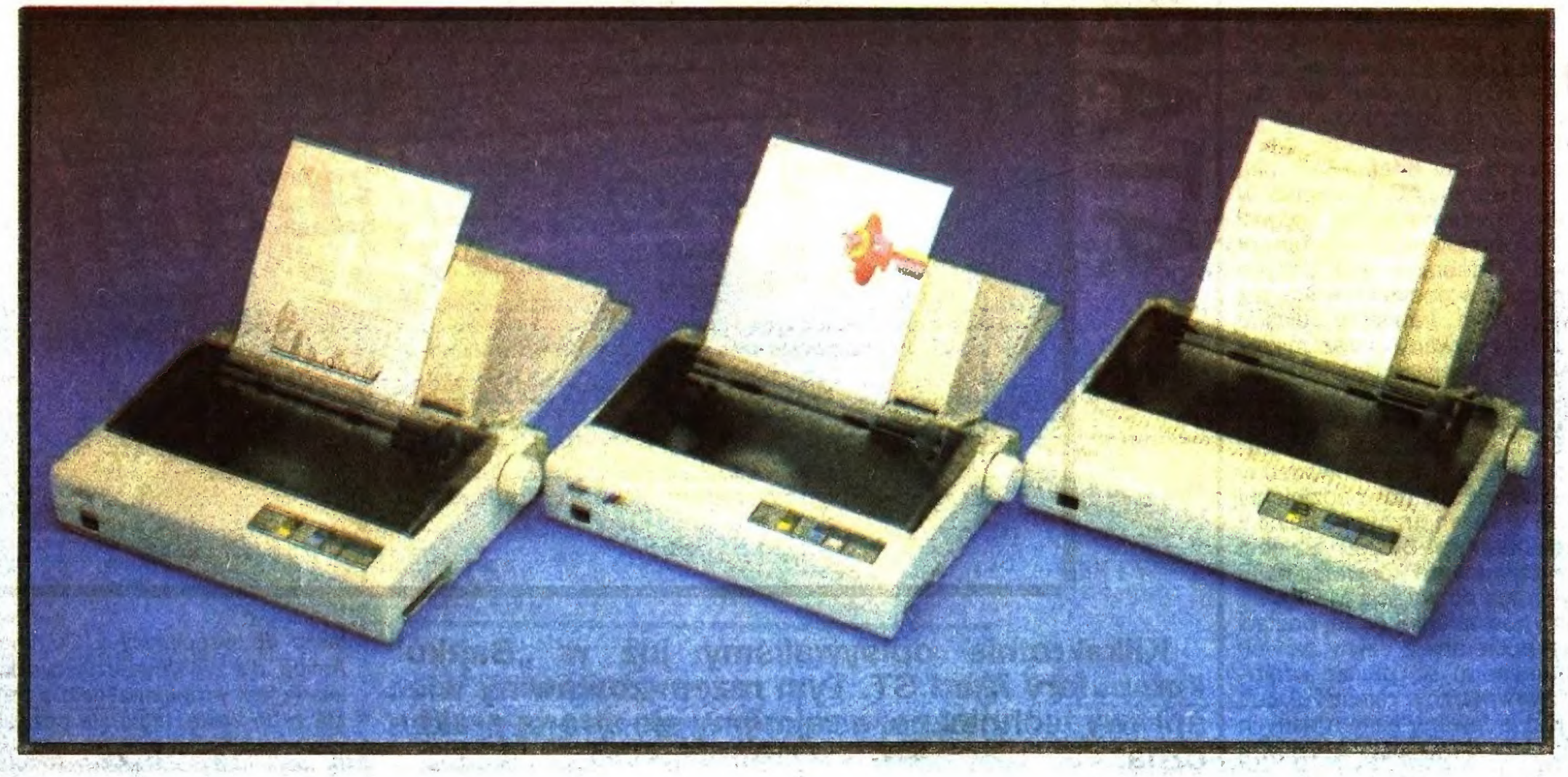
Marek Zachar

Opisany w artykule komputer Atari 520STFM i stację dysków Atari SF314 otrzymaliśmy z Logical Desing Works (Stany Zjednoczone), a komputer 520STM i stację dysków Cumana z Elektronics Export (W. Brytania).

New New New

Drukarka Star

to najlepszy przyjaciel Twojego Komputera!



STAR LC — 10
Najpopularniejsza drukarka świata! Ponad 2.000.000 użytkowników nie może się mylić! Jeśli chcesz mieć drukarkę solidną i niezawodną, wszechstronną i niedrogą, to decyzja jest prosta: LC–10!
Prędkość druku:
144zn/s (draft)
36zn/s (NLQ)

STAR LC — 10 colour
Ta sama japońska jakość i niezawodność jak LC–10, z dodatkową możliwością druku w 7 kolorach!
Wystarczy tylko działożyć barwną taśmę.

Prędkość druku:
144zn/s (draft)
36zn/s (NLQ)

STAR LC24 — 10
Drukarka roku 1989 w Europie Zachodniej! Doskonała jakość pisma dzięki nowoczesnej, 24-igłowej głowicy.
Niezastąpiona do korespondencji oraz precyzyjnej grafiki.

Prędkość druku:
170zn/s (draft)
57 zn/s (LQ)

Wszystkie trzy drukarki posiadają cztery kroje czcionek, wbudowany traktor oraz funkcje „Paper Park". Można w nich stosować papier z perforacją lub bez, jak również pojedyńcze kartki.

UWAGA: W celu zainstalowania polskich znaków, prosimy skontaktować się z naszym punktem serwisowym.

Zalecane ceny detaliczne	LC–10	2 700 000 zł, kaseta barwiąca (czarna) 40 000 zł
	LC–10 ctr	3 900 000 zł, kaseta barwiąca (kolorowa) 100 000 zł
	LC24–10	4 500 000 zł, kaseta barwiąca (czarna) 100 000 zł, karta z polskimi znakami 650 000 zł.

Drukarki Star można kupić:

W Warszawie w sklepie Agrocomputer, ul. Grażyny 1, tel. 48-63-94
W Kielcach w firmie Interbit, ul. Man. Lipcowego 4, tel. 441-99
oraz w zasadzie w każdej firmie komputerowej na terenie całego kraju.

Serwis i instalacja znaków polskich:

ABC Data Service, Warszawa, ul. Konopnickiej 6, tel. 28-92-81 w. 128
Interbit, Kielce, ul. Man. Lipcowego 4. tel. 441-99

Przedstawiciel na Polskę:
ABC Data Sp. z o.o.
Warszawa, ul. ZWM 9
tel. 643-53-36 tx. 817123

star
the ComputerPrinter

ABC Data