P. 1877/85

mikroKLAN 20

8 1985

informatyka

System LOGOS

Oprogramowanie pamięci kasetowych

Mikrokomputery w RWPG

Dalszy ciąg języka LOGO

Cena 100 zł

informatyka

8 1985

Nr 8
Miesięcznik Rok XX
Sierpień 1985

Organ Komitetu Informatyki MNSZWiT oraz Komitetu Naukowo-Technicznego NOT ds. Informatyki

KOLEGIUM REDAKCYJNE:

Dr inż. Wacław ISZKOWSKI, mgr Teresa JABŁOŃSKA (sekretarz redakcji), Władysław KLEPACZ (zastępca redaktora naczelnego), prof. dr hab. Leon ŁUKASZEWICZ (redaktor naczelny), mgr inż. Andrzej J. PIOTROWSKI, dr inż. Janusz ZALEWSKI

STALE WSPÓŁPRACUJĄ:

Mgr inż. Witold ABRAMOWICZ (Szwajcaria), mgr inż. Ryszard K. KOTT (Wielka Brytania), mgr inż. Teresa WILCZEK

PRZEWODNICZĄCY RADY PROGRAMOWEJ:

Prof. dr hab. Tadeusz PECHE

Materiałów nie zamówionych redakcja nie zwraca

Redakcja: 00-041 Warszawa, ul. Jasna 14/16, pok. 243 i 244, tel. 27-71-40 lub 26-82-61 w. 184

Zakł. Graf. „Tamka". Zam. 0718-1300/85. Obj. 4,0 ark. druk. Nakład 6300 egz. N-28

ISSN 0542-9951, INDEKS 36124

Cena egzemplarza zł 100,—
Prenumerata roczna zł 1200,—

WYDAWNICTWO
NACZELNA ORGANIZACJA TECHNICZNA
CZASOPISM I KSIĄŻEK TECHNICZNYCH
SIGMA
00-950 Warszawa
skrytka pocztowa 1004
ul. Biała 4

W NUMERZE: Strona

WACŁAW ISZKOWSKI
PAWEŁ GRZEGORZEWICZ

Instytut Informatyki
Politechnika Warszawska

POLITECHNIKA ŚLĄSKA
Biblioteka
GLIWICE

P. 1877/85

LOGOS — system generowania systemów operacyjnych

Umiejętność zaprojektowania a następnie zaimplementowania i uruchomienia prostych systemów operacyjnych jest obecnie tak konieczna dla inżynierów informatyki, jak umiejętność poprawnego programowania. Złożoność tych systemów oraz z konieczności (przynajmniej w części) zapis w asemblerze, utrudnia skonstruowanie i przetestowanie większego projektu w ramach zajęć dydaktycznych. Dodatkowym utrudnieniem jest to, że przy realizacji projektu komputer musi być całkowicie oddany pojedynczemu studentowi.

Dla usprawnienia zajęć projektowych z przedmiotu „Systemy operacyjne" skonstruowano w Instytucie Informatyki Politechniki Warszawskiej system LOGOS (LOGical Operating System), umożliwiający budowanie modularnych systemów operacyjnych dla minikomputerów serii SM 4. Systemy takie są generowane z rozłącznie kompilowanych modułów-segmentów asemblera MACRO-11, przy zastosowaniu oprogramowania systemowego RSX-11. Student korzystający z generatora LOGOS może złożyć i uruchomić system wykorzystując bogatą bibliotekę modułów standardowych oraz moduły przygotowane samodzielnie.

Celem projektu zajęć (z przedmiotu „Systemy operacyjne") przeprowadzanych przy zastosowaniu LOGOS jest poznanie:

- podstawowych zasad konstrukcji jądra wieloprocesowego systemu operacyjnego oraz zasad obsługi urządzeń zewnętrznych
- przykładowych postaci produktów translatorów języków programowania współbieżnego, a także metod realizacji maszyn wirtualnych dla tych języków
- metod testowania dynamicznego i usuwania błędów ze złożonych wieloprocesowych, asynchronicznie działających programów zapisanych w asemblerze.

Struktura systemów klasy LOGOS

System LOGOS definiuje określoną klasę systemów operacyjnych, jakie można otrzymać korzystając z funkcji przez niego dostarczonych. Cechą charakterystyczną tych systemów jest wieloprocesowość, otrzymana przez realizację procesorów wirtualnych w jądrze systemu wykorzystujących pojedynczy procesor fizyczny o architekturze stosowej. Otrzymane systemy są wielopoziomowymi maszynami wirtualnymi odwzorowującymi najnowsze koncepcje w tej dziedzinie. Elastyczność generowanych systemów jest realizowana poprzez umożliwienie samodzielnego definiowania:

- dowolnego typu mechanizmów synchronizacji i komunikacji pomiędzy procesami oraz różnych algorytmów szeregowania procesów gotowych do wykonania
- zleceń obsługi różnego rodzaju urządzeń zewnętrznych według rozmaitych metod (na przykład aktywne oczekiwanie, oczekiwanie na przerwanie czy według koncepcji języka MODULA-2 [1])
- procedur różnego rodzaju zleceń systemowych obejmujących gospodarkę pamięcią operacyjną i pomocniczą
- procesów systemowych realizujących funkcje prostego języka poleceń operatorskich i wspomagających uruchamianie oraz testowanie działania systemu i procesów użytkowych
- procesów użytkowych testujących zaprojektowany system lub wykorzystujących zaimplementowane w nim mechanizmy współpracy między procesami.

Każdy system klasy LOGOS może być złożony z pięciu poziomów maszyn wirtualnych rozszerzających funkcje poziomu zerowego, będącego sprzętem maszyny.

Kolejnymi poziomami systemu są:

1 — jądro systemu z operacjami na zbiorach procesów (dokładniej ich deskryptorów) dostarczając dowolną liczbę procesorów wirtualnych dla pozostałych poziomów oraz przechwytujące przerwanie programowe i wewnętrzne procesora

2 — poziom obsługi urządzeń zewnętrznych, realizujący polecenia operacji we-wy, oraz współpracę z urządzeniami wraz z obsługą generowanych przerwań przez procesy obsługi urządzeń

3 — poziom realizacji poleceń systemowych wykonywanych kosztem procesów użytkowych i procesów systemowych

4 — poziom procesów systemowych śledzących i sterujących działaniem systemu oraz jego współpracą z otoczeniem

5 — poziom procesów użytkowych.

Komunikacja między poziomami jest jednokierunkowa i odbywa się przez zlecenia (przy użyciu rozkazów przerwań programowych). Każdy z trzech pierwszych poziomów realizuje zlecenia odrębnie identyfikowane. Zestawienie poziomów systemu LOGOS wraz z opisem rodzajów realizowanych zleceń oraz stopniem zablokowania poziomów układu przerwań procesora przedstawiono w tabeli 1.

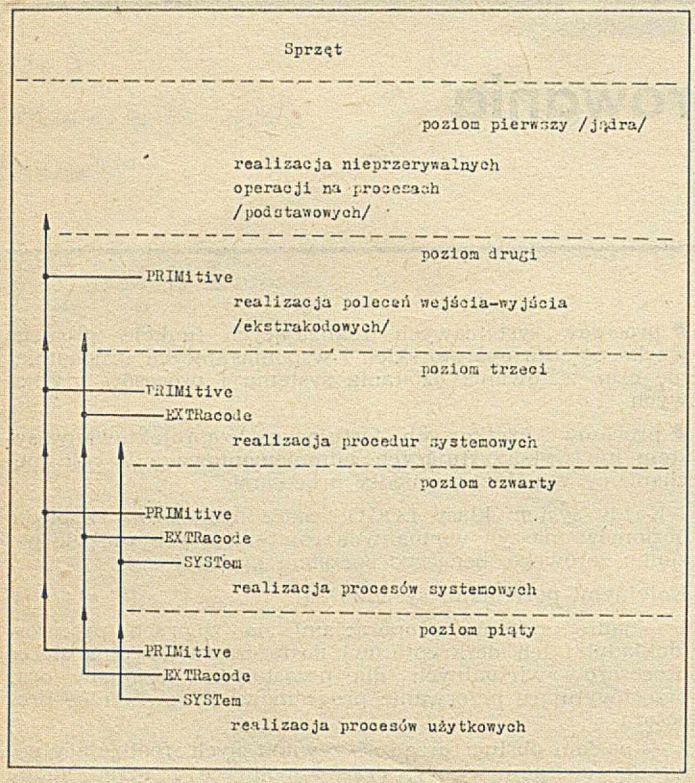
Zasady jednokierunkowej komunikacji w systemie przedstawia tabela 2. Każdy z rodzajów zleceń może mieć do

Tabela 1. Zestawienie poziomów systemu klasy LOGOS

Funkcja realizowana przez poziom	Sposób funkcjonowania poziomu	Nazwa zleceń realizowanych na tym poziomie	Rodzaj stosu	Blokowanie przerwań w czasie pracy poziomu
Sprzęt				
Poziom operacji na procesach	procedury zleceń	PRIMitive	systemowy stos jądra	zablokowane całkowicie
Poziom obsługi wejścia-wyjścia	procedury zleceń i procesy obsługi	EXTRa-code	stos zlecającego, a dla procesów obsługi stos własny	zablokowane częściowo
Poziom realizacji zleceń systemowych	procedury zleceń	SYSTem	stos procesu zlecającego	nie zablokowane
Poziom procesów systemowych	procesy	—	stos własny	nie zablokowane
Poziom procesów użytkowych	procesy	—	stos własny	nie zablokowane

trzech parametrów. Po wydaniu rozkazu przerwania programowego zawartego w treści wywołania zlecenia następuje zapamiętanie stanu procesu na jego stosie, rozpoznanie zlecenia, a następnie uruchomienie procedury realizującej to zlecenie. Po zakończeniu wykonania zlecenia odtworzony jest stan procesu.

Tabela 2. Zasady komunikacji w systemie LOGOS



Na poziomie pierwszym (jądra) realizowane są operacje podstawowe na deskryptorach procesów z wykorzystaniem stosu systemowego. Całkowicie zablokowany układ przerwań zapewnia nieprzerywalność wykonania tych operacji.

Tabela 3. Standardowe operacje na procesach w systemie klasy LOGOS

PRIM $USP, #adr–kolejki_zawieszania zawieszenie aktualnie działającego procesu w podanej kolejce
PRIM $ACT, #adr_kolejki_z_procesem_do_aktywowania aktywowanie pierwszego procesu z kolejki bez przełączania procesora fizycznego na ten proces
PRIM $WAP przełączenie procesora fizycznego na proces o najwyższym priorytecie (wybrany przez dyspozytora)
PRIM $AWAK, #adr–deskryptora–procesu zapoczątkowanie działania procesu
PRIM $TOP zatrzymanie procesu wydającego operację

Wykaz standardowo zdefiniowanych operacji podstawowych jest podany w tabeli 3. Poziom jądra zawiera również procedury elementarnych operacji na kolejkach procesów gotowych i zawieszonych, a także procedury dyspozytora, realizujące jedną ze strategii wyboru procesu gotowego dla procesora rzeczywistego. W bibliotece znajdują się moduły dyspozytorów szeregujące procesy według strategii:

- pierwszy przyszedł, pierwszy obsłużony (FIFO)
- algorytm okrężny (ang. round robin) z zastosowaniem dodatkowego procesu obsługi zegara
- priorytetowy.

Na poziomie obsługi urządzeń zewnętrznych są realizowane zlecenia operacji we-wy. Implementacja tych zleceń może polegać na synchronicznej współpracy z danym urządzeniem w trybie aktywnego oczekiwania, bądź też z wykorzystaniem dodatkowego procesu obsługi przerwań z urządzenia. Zakres dowolności metody implementacji operacji we-wy jest opisany w tabeli 4. W systemie LOGOS dostępne są standardowe zlecenia we-wy obsługi konsoli operatorskiej i drukarki w trybie aktywnego oczekiwania z zablokowanymi przerwaniami.

Tabela 4. Cechy implementacji zleceń systemu LOGOS

1. Dowolny sposób indentyfikacji urządzenia zewnętrznego
2. Parametry mogą być przekazywane przez stos, rejestry lub blok opisu transmisji
2. Zlecenie może być synchroniczne lub asynchroniczne
4. Zlecenie może być jedno lub wielodostępne
5. Koniec transmisji może być wykrywany w aktywnym oczekiwaniu (lub w oczekiwaniu) na przerwanie
6. Procedura realizacji transmisji może być wykonywana na koszt procesu zlecającego lub specjalnego procesu obsługi urządzenia
7. Przerwania z danego urządzenia są blokowane w trakcie wykonywania procedury realizacji transmisji
8. W modułach wielodostępnych można implementować złożone strategie obsługi procesów współpracujących z urządzeniem
9. Nie wszystkie podane wyżej cechy mogą być jednocześnie wykorzystane

Na pozostałych poziomach znajdują się procesy systemowe i użytkowe. Standardowo dołączony jest proces oczekiwania na przerwanie — wybierany przez dyspozytora gdy brak jest innych procesów gotowych do pracy.

```
;
; PROCES WAIT OCZEKIWANIA NA PRZERWANIE
;
        .PSECT  A%WAIT
WAITWE: WAIT
        BR      WAITWE
;
        DESCR%  WAITWE,PR0,L.WT,5,NIL%,WAIT
;
```

W bibliotece modułów systemu są dostępne standardowe moduły operatora inicjującego pozostałe procesy oraz procesu uruchomieniowego (debuggera) systemu.

Struktura modułowa generowanego systemu

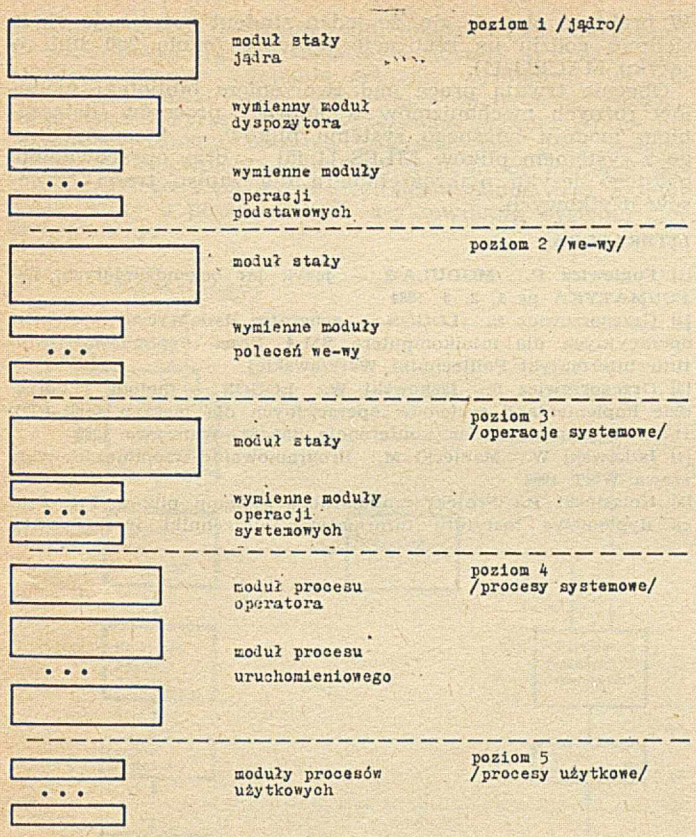
System LOGOS dostarcza szkielet projektowanego systemu operacyjnego jako zestaw modułów-segmentów w języku MACRO-11. Zadaniem użytkownika systemu LOGOS jest wybranie odpowiednich modułów z biblioteki oraz opracowanie własnych modułów (półskomplikowanych segmentów MACRO-11) uzupełniających wymagane funkcje systemu.

W zestawie podstawowych modułów standardowych występują moduły zawierające:

- procedury jądra systemu, wraz z deklaracją struktur danych wektora przerwań i dyspozytora procesów, oraz zestaw standardowych procedur operacji podstawowych uzupełniony procedurą identyfikacji zlecenia typu PRIMitive
- identyfikację zlecenia typu EXTRacode uzupełnioną zestawem trzech procedur prostej obsługi urządzeń zewnętrznych
- identyfikację zlecenia typu SYSTem
- proces systemowy operatora inicjującego pozostałe procesy
- proces uruchomieniowy (debugger)
- deklaracje dopełniające zbiór globalnych identyfikatorów systemu wykorzystywanych w jego konsolidacji.

Zestaw ten może być następnie uzupełniony o moduły własne użytkownika, rozszerzające funkcje każdego z poziomów lub zastępujące moduły standardowe, a także o moduły procesów użytkowych. Statyczny obraz struktury modularnej systemu klasy LOGOS przedstawia rysunek.

Dla ułatwienia użytkownikowi zapisu algorytmu jego modułu w języku MACRO-11, z uwzględnieniem wymagań powiązań międzymodułowych wynikających ze struktury systemu klasy LOGOS, jest opracowana biblioteka makrodefinicji. W bibliotece tej są zawarte makrodefinicje opisujące podstawowe struktury danych systemu, takie jak: deskryptor procesu, kolejka itp. oraz makroinstrukcje opisujące wywołania zleceń, inicjowania operacji we-wy oraz włączające procedury realizacji nowych zleceń czy proce-



Modularna struktura generowanych systemów

sów w strukturę systemu. Zastosowanie makrodefinicji zmniejsza znacznie długość tekstu powodując tym samym zwiększenie czytelności i zmniejszenie ryzyka popełnienia błędu. Zestaw przykładowych makrodefinicji prezentuje tabela 5.

Jako przykład modułu systemu pokazano w dodatku A moduł obsługi drukarki z realizacją zlecenia wyprowadzania zawartości bufora przy zastosowaniu procesu współpracującego z urządzeniem przez oczekiwanie na przerwanie.

Tabela 5. Wykaz wybranych makrodefinicji standardowych

EXTR ident [, par1 [, par2 [, par3]]] PRIM ident [, par1 [, par2 [, par3]]] SYST ident [, par1 [, par2 [, par3]]] makrorozkazy wywo'ania zleceń z umieszczeniem parametrów na stosie
DESCR$ adr_pocz, poziom_blok przerw, prior_wewnętrzny, wielkość_stosu, adr_danych, ident_procesu deklaracja deskryptora procesu DESOF$ i definicje identyfikatorów jego pól
TST$UZ adr_rej_stanu, etykieta_błędu DOIO$ adr_wek_przerw, adr_rej_stanu, etykieta DOIO$M względny_adr_wek_przerw, względny_adr_rej_stanu, względna_etykieta INTER$ adr_rej_stanu, etykieta makrorozkazy: aktywnego oczekiwania, oczekiwania na przerwanie, oczekiwania na przerwanie przez proces jednorodny, a także deklaracja wektora przerwań
SAVER$ LOADR$ makrorozkazy zapamiętania i odtwarzania rejestrów ze stosu
● QUEUE ● SEMA deklaracje kolejki i semafora
CONST$ NIL$ PRIOR$ QLEN$ PARAM$ deklaracje stałych systemu, wartości NIL$, poziomów blokowania przerwań, długości nagłówka kolejki oraz pól parametrów na stosie
TABLE$ rodzaj_tablicy,liczba_pozycji INEX$ ident zlecenia,nr zlecenia,adr_procedury INPRM$ ident zlecenia,nr zlecenia,adr_procedury INSYS$ ident zlecenia,nr zlecenia,adr_procedury makrodefinicje realizujące powiązania w strukturze statycznej systemu: deklaracja tablicy oraz deklaracje elementów tej tablicy
QCLEAR adr_kolejki makrorozkaz opróżniania kolejki

Dodatek A. Przykładowy moduł systemu LOGOS

```
        .TITLE  LPBASE
        .IDENT  /05/
;-------------------------------------------------------
;  MODUL OBSLUGI DRUKARKI MOZAIKOWEJ
;    SYNCHRONICZNIE Z BUFOROWANIEM W PROCESIE
;    ZLECAJACYM
;
;    AUTOR: WACLAW ISZKOWSKI
;   16-DEC-83
;-------------------------------------------------------
;    MAKRODEFINICJE GLOBALNE
;
        .MCALL  EXTLBR
        EXTLBR
;-------------------------------------------------------
;   STALE MODULU
;
LP=12
LPV=200         ; LP11   ADRES WEKTORA PRZERWAN
LPS=177514      ;        REJESTR STANU
LPB=177516      ;        REJESTR BUFORU
;-------------------------------------------------------
;   DANE LOKALNE MODULU
;
        .PSECT  DOLP%D
ADRBUF: .WORD   NIL%            ; ADRES BUFORA
KELNER: .QUEUE                  ; PROCES OBSLUGI LP11
KLIENT: .QUEUE                  ; PROCES ZLECAJACY
        INTER%  LPS,LPLAB       ; ODBIOR PRZERWANIA
KOM:    .ASCII  /LP-ERROR/<12>  ; KOMUNIKAT O BLEDZIE
        .EVEN
;-------------------------------------------------------
;   PROCEDURA OBSLUGI ZLECENIA
;
        .PSECT  DOLP%E
LPWE:   MOV     P.P1(R0),ADRBUF ; ADRES BUFORA
        PRIM    #ACT,$KELNER    ; PRZELANCZENIE NA PROCES
        PRIM    #USP,$KLIENT    ;    OBSLUGI LP11
        RTS     PC              ;
;
;   DOLACZENIE ZLECENIA
;
        INEX%   PRINT,2,LPWE
;-------------------------------------------------------
;   PROCEDURA PROCESU OBSLUGI LP11
;
        .PSECT  DOLP%I
LPPROC: QCLEAR  KLIENT          ; INICJALIZACJA
        QCLEAR  KELNER          ;
        CLR     LPS             ;
LP.NAT: PRIM    #USP,$KELNER    ; OCZEKIWANIE NA PRACE
LP.REP: MOV     ADRBUF,R3       ; ADRES BUFORA
LPP.1:  MOVB    (R3)+,R2        ; KOLEJNY ZNAK
        BEQ     LP.NXT          ; KONIEC BUFORA
        MOV     R2,LPB          ; WYPROWADZENIE ZNAKU
        DOIO%   LPV,LPS,LPLAB   ; OCZEKIWANIE NA PRZERWANIE
        TST     LPS             ; CZY BLAD
        BMI     LP.ERR          ; TAK
        CMPB    #LF,R2          ; CZY OSTATNI ZNAK
        BNE     LPP.1           ; NIE
        PRIM    #ACT,$KLIENT    ; AKTYWACJA ZLECAJACEGO
        BR      LP.NXT          ;
;
LP.ERR: EXTR    OUT%,$KOM       ; WYPROWADZENIE KOMUNIKATU
        CLR     LPS             ; ZEROWANIE
        MOV     #LF,LPB         ; KONIEC LINII
        TST%UZ  LPS             ;
        BR      LP.REP          ; POWTORZENIE TRANSMISJI
;
;  DEKLARACJA DESKRYPTORA PROCESU
;
        DESCR%  LPPROC,PR4,L.BR3,5,ADRBUF,DLP0
        .END


 Wywolanie ekstrakodu implementowanego w tym module
jest postaci:

        ...
BUF:    .ASCII / ... /<12>
        ...
        EXTR    PRINT,$BUF
        ...
```

Generowanie gotowego systemu

Po przygotowaniu modułów systemu, korzystając z konsolidatora TKB-11 systemu RSX-11, następuje złożenie w jedną całość gotowego do działania systemu. Dla ułatwienia tej procedury przygotowano plik dyrektyw systemu RSX-11, w którym przez formę pytań i odpowiedzi uzyskano klasyczną postać programu generatora systemu. Program ten dołącza do obowiązkowo istniejących w systemie modułów — moduły wybrane przez projektanta z biblioteki gotowych półskomplikowanych segmentów lub z własnych plików zawierających segmenty samodzielnie przygotowane. Po konsolidacji gotowy system jest wprowadzany bezpośrednio dyrektywą BOO (z systemu RSX) do pamięci operacyjnej stając się samodzielnie działającym programem. Dla usprawnienia działań studenta-projektanta systemu oraz prowadzącego zajęcia w całej dość złożonej procedurze implementacji systemu klasy LOGOS wprowadzono szereg plików dyrektyw, organizujących gospodarkę dość dużą liczbą plików i ułatwiający pokonywanie następujących kolejnych etapów przygotowania systemu:

- edycji czyli wprowadzania i poprawiania nowych wersji modułów
- asemblacji modułów w półskomplikowane segmenty z wykorzystaniem biblioteki makrodefinicji i podprogramów
- konsolidacji półskompilowanych segmentów wybranych z bibliotek segmentów standardowych i własnych
- gospodarowanie zawartościami katalogów plików poszczególnych studentów.

Efekty eksploatacji systemu

Opisany system jest aktualnie wykorzystywany do zajęć w ramach projektu do przedmiotu „Systemy Operacyjne". Jego obecna postać jest rozszerzoną wersją systemu poprzednio wykorzystywanego w ramach zajęć projektowych [3]. Dotychczasowe doświadczenia wskazują na dużą efektywność pracy studentów, którzy w początkowym okresie mogą wspólnie wykorzystywać pięciodostępny system, przygotowując fragmenty swoich projektów. Ostatni etap pracy — uruchomienie nowego systemu musi być wykonywane pojedynczo — przy czym jest to okres intensywnego korzystania z programu wspomagającego — bez konieczności wykorzystania pulpitu operatorskiego komputera. W praktyce okazało się, że jeden student potrzebuje około siedmiu godzin na realizację projektu (około 200 linii w języku MACRO-11).

Obecnie trwają prace nad tworzeniem biblioteki modułów różnych mechanizmów współpracy procesów, dołączeniem modułu własnego systemu plików — kompatybilnego z systemem plików FILES-11 [5] — oraz opracowaniem prostego języka makroasemblera dla zapisu treści procesów użytkowych.

LITERATURA

[1] Fuglewicz P.: MODULA-2 — język lat osiemdziesiątych. INFORMATYKA nr 1, 2, 3 1984

[2] Grzegorzewicz P.: LOGOS — generator dydaktyczny systemów operacyjnych dla minikomputera SM-4. Praca dyplomowa Instytutu Informatyki Politechniki Warszawskiej

[3] Grzegorzewicz P., Iszkowski W.: LOGOS — metoda i narzędzie implementacji systemów operacyjnych dla maszyn serii PDP 11/SM 3. Komunikat na konferencję SM 85, Warszawa 1985

[4] Iszkowski W., Maniecki M.: Programowanie współbieżne. Warszawa WNT 1982

[5] Kolasiński P.: Projekt modularnego systemu plików F11. Praca dyplomowa Instytutu Informatyki Politechniki Warszawskiej.

CEZARY STĘPIEŃ
JAROSŁAW SZYRKOWIEC

Instytut Informatyki
Politechnika Warszawska

Wielopoziomowy system oprogramowania pamięci kasetowych PK-I

Rozwój pamięci na dyskach elastycznych nie zmniejsza w naszym kraju zainteresowania pamięciami kasetowymi [1, 2]. Również na świecie w niektórych zastosowaniach, pamięci kasetowe skutecznie konkurują z innymi rodzajami pamięci masowych. Są one szczególnie użyteczne w systemach cyfrowych do wielu zastosowań ze względu na dużą niezawodność działania, mały pobór mocy, odporność na wpływy czynników zewnętrznych i prostotę obsługi. Dowodem tego jest fakt wypuszczenia przez firmę HEWLETT-PACKARD nowego modelu pamięci kasetowej 82161A [3, 4].

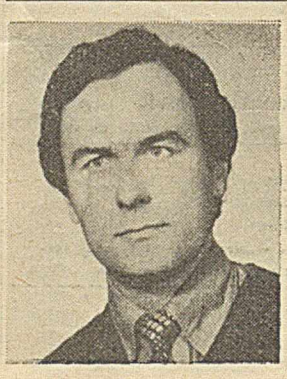
W Polsce istnieje wielu użytkowników systemów z mikroprocesorem 8080, którzy wykorzystują pamięci kasetowe PK-1 produkcji zakładów MERAMAT. Z myślą o nich w Instytucie Informatyki Politechniki Warszawskiej opracowano kompleksowy, wielopoziomowy system oprogramowania pamięci kasetowych PK-1, napisany w asemblerze 8080, zapewniający możliwość prostej, wygodnej i niezawodnej współpracy systemu mikroprocesorowego z pamięciami kasetowymi. Istnieje możliwość przystosowania systemu do współpracy z pamięcią kasetową PK-3.

Schemat blokowy systemu II-SOPK przedstawiono na rys. 1. Główną część systemu stanowią procedury współpracy z pamięciami kasetowymi, pogrupowane w moduły na poziomie niskim, średnim i wysokim.

Opis ogólny systemu

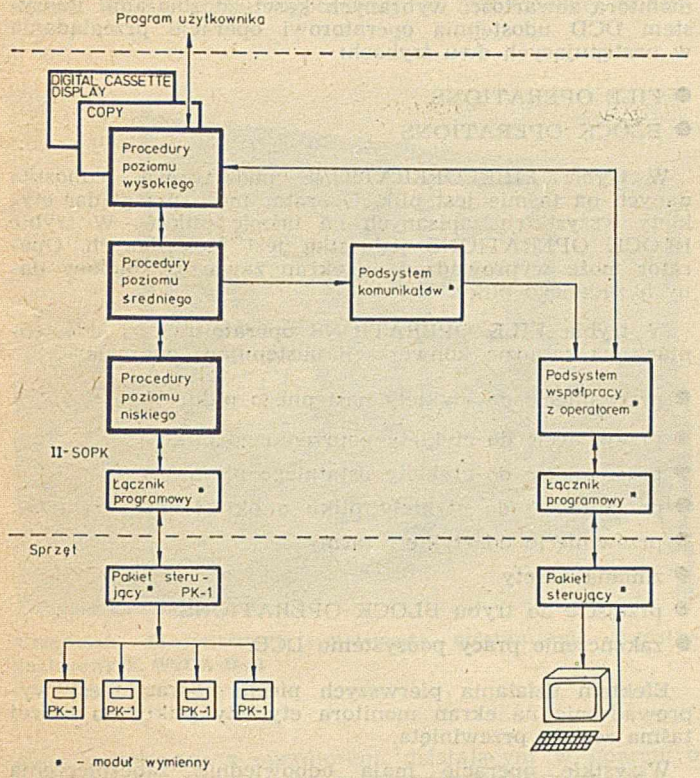
Procedury poziomu niskiego zapewniają wykonanie operacji podstawowych, takich jak: zapis bloku, zapis znacznika taśmy (ang. tape mark), odczyt bloku, kasowanie odcinka taśmy itp., wraz z obowiązkową kontrolą poprawności przebiegu tych operacji. Procedury te na podstawie informacji odczytywanych w trakcie wykonywania operacji z pakietu sterującego, tworzą informację zwrotną, na

Dr inż. CEZARY STĘPIEŃ ukończył studia w 1974 r. na Wydziale Elektroniki Politechniki Warszawskiej o kierunku — Informatyka. W 1975 r. rozpoczął pracę w Instytucie Informatyki PW. Od 1983 r. pracuje na stanowisku adiunkta. Zajmuje się projektowaniem urządzeń mikroprocesorowych oraz podstawami teoretycznymi m.c.



Mgr inż. JAROSŁAW SZYRKOWIEC ukończył studia na Wydziale Elektroniki Politechniki Warszawskiej o kierunku — Informatyka. Pracuje obecnie w OETO Politechniki Białostockiej jako asystent. Współpracując z Instytutem Informatyki PW zajmuje się oprogramowaniem systemów mikroprocesorowych.

podstawie której po zakończeniu działania procedury można zorientować się, czy zakończyła się ona w sposób standardowy, czy też wystąpiła sytuacja wyjątkowa. W celu zapewnienia możliwości instalowania systemu w różnych środowiskach sprzętowych wyróżniono wymienny moduł, spełniający funkcję łącznika programowego. Jego zadanie polega na zapewnieniu właściwej współpracy procedur poziomu niskiego z pakietem sterującym pamięciami kasetowymi, który u poszczególnych użytkowników systemu może być różny.



Rys. 1. Schemat blokowy systemu II-SOPK

Procedury poziomu średniego przeznaczone są do zapisu lub odczytu bloków danych na poziomie plików. Obejmują one takie operacje, jak: otwarcie pliku dla zapisu lub odczytu, zapis bloku danych do pliku, odczyt bloku, zamknięcie pliku po zakończeniu operacji zapisu lub odczytu. Procedury te charakteryzują się wysokim stopniem automatyzacji, dotyczącej w szczególności złożonych procesów otwierania i zamykania plików. Daje to z jednej strony proste zasady współpracy programisty z pamięcią kasetową, a z drugiej zapewnia odpowiednio dużą niezawodność przechowywania danych. Procedury te komunikują się z pamięcią kasetową wyłącznie poprzez wywołanie procedur niskiego poziomu, które z kolei — przekazując informacje zwrotne — umożliwiają procedurom poziomu średniego dokładną kontrolę przebiegu operacji oraz przekazywanie wybranych informacji operatorowi w oparciu o specjalny podsystem komunikatów. Podsystem ten jest wymienny, umożliwiając przystosowanie systemu oprogramowania pamięci kasetowych do wymagań użytkownika. Przykładowo, możliwe jest wykorzystanie rozbudowanego podsystemu komunikatów, podającego i zachowującego informacje tekstowe w ustalonych dla każdej pamięci kasetowej polach monitora ekranowego tak długo, jak jest to konieczne. Inny podsystem komunikatów, charakteryzujący się niewielką zajętością pamięci stałej, może wyświetlać komunikaty wyłącznie w postaci dwucyfrowego symbolu. Istnieje także możliwość wykorzystywania systemu II-SOPK bez podsystemu komunikatów.

Poziom najwyższy tworzy zestaw operacji na plikach danych. Procedury tego poziomu pozwalają zapisać lub odczytać z pamięci operacyjnej cały plik danych, a nie pojedynczy blok z pliku, jak to występuje w procedurach poziomu średniego. Wykonano także następujące dwa dodatkowe podsystemy, ułatwiające posługiwanie się pamięciami kasetowymi:

- podsystem przeglądania zawartości pamięci kasetowej (DIGITAL CASSETTE DISPLAY),
- podsystem kopiowania plików (COPY).

Wykorzystując procedury poziomu średniego użytkownik może łatwo tworzyć inne podsystemy (np. podsystem sortowania plików), lub modyfikować istniejące. Podsystemy wysokiego poziomu komunikują się z operatorem w trybie konwersacyjnym.

Oprogramowanie niskiego poziomu

Bezpośrednią obsługę pamięci kasetowej zapewnia zestaw następujących trzynastu procedur:

LREZ — rezerwacja jednostki pamięci
LREL — zwolnienie jednostki pamięci
LTEST — testowanie stanu jednostki
LREW — przewinięcie kasety do początku
LKAS — kasowanie odcinka taśmy
LCZT — odczytanie bloku danych
LPSZ — zapisanie bloku danych
LPTM — zapisanie znacznika taśmy
LCOF — cofnięcie o 1 blok
LCTM — cofnięcie do znacznika taśmy
LPRP — przewinięcie w przód o n bloków
LPRW — przewinięcie wstecz o n bloków
LPRB — przewinięcie do boku o numerze n.

Po włożeniu kasety jako pierwsza powinna być wywołana procedura LREZ, a jako ostatnia przed wyjęciem — procedura LREL.

Procedury komunikują się między sobą poprzez wydzielone w pamięci RAM pole sterujące, co w połączeniu z ich rozbudowanymi systemami kontroli nie dopuszcza do niewłaściwego ich użycia. Przykładowo, żadna z procedur wykorzystujących ruch wstecz nie może stworzyć sytuacji, w której głowica zapisu — odczytu znajdzie się przed znacznikiem początku taśmy (BOT).

Przyjęto zasadę, że w przypadku niewłaściwego użycia lub nieprawidłowego zakończenia procedury głowica pozostaje w tym samym położeniu co przed jego wywołaniem. Informacja zwrotna dotycząca przebiegu i zakończenia procedury jest przekazywana w ustalonym bajcie pamięci RAM. Znaczenie poszczególnych bitów jest następujące:

b_0 — wykryto BOT lub EOT
b_1 — zakaz zapisu albo zbyt długa przerwa międzyblokowa
b_2 — operacja nie wykonana albo sygnalizacja strony kasety dla procedury (LTEST)
b_3 — brak gotowości
b_4 — kaseta nie załadowana
b_5 — zbyt długi blok danych
b_6 — wykrycie znacznika taśmy
b_7 — błąd transmisji.

Procedury niskiego poziomu charakteryzują się wysokim stopniem automatyzacji współpracy z pamięcią kasetową. Ilustruje to przykład działania procedury zapisu bloku LPSZ. Procedura ta przy zapisie powoduje jednocześnie wykonanie odczytu kontrolnego mającego na celu porównanie informacji odczytywanej z informację źródłową. W wypadku stwierdzenia niezgodności, co w większości spowodowane jest uszkodzeniem nośnika magnetycznego, procedura kasuje odpowiedni odcinek taśmy i dokonuje zapisu bezpośrednio za tym odcinkiem. Gdy w tym wypadku zapis nie powiedzie się operacja ta jest powtarzana wielokrotnie, do chwili znalezienia nieuszkodzonego odcinka taśmy lub osiągnięcia maksymalnej długości przerwy międzyblokowej.

Takie rozwiązanie pozwala wykorzystywać nie tylko specjalne kasety do zapisu cyfrowego (ang. digital cassette), lecz także zwykłe chromowe kasety muzyczne, na których trzeba jedynie samodzielnie wykonać otwory BOT i EOT.

Przy projektowaniu procedur niskiego poziomu założono skrócenie czasu dostępu do dowolnego bloku na taśmie. Zrealizowano to w procedurach LPRP, LPRW, LPRB przez odszukiwanie żądanego bloku ruchem szybkim. Pozwoliło to skrócić czas dostępu w najbardziej niekorzystnym wypadku z 15 min (przy ruchu wolnym) do około 1 minuty (przy ruchu szybkim).

Oprogramowanie średniego poziomu

Procedury poziomu średniego zaprojektowano z uwzględnieniem wygody programisty, zakładając, że może on wykorzystywać te procedury przy rozbudowie podsystemów oprogramowania wysokiego poziomu lub pisaniu własnych

programów użytkowych. Oprogramowanie średniego poziomu tworzy siedem procedur:

MRES — rezerwacja logiczna jednostki pamięci

MRLS — zwolnienie jednostki pamięci

MOPR — otwarcie pliku do odczytu

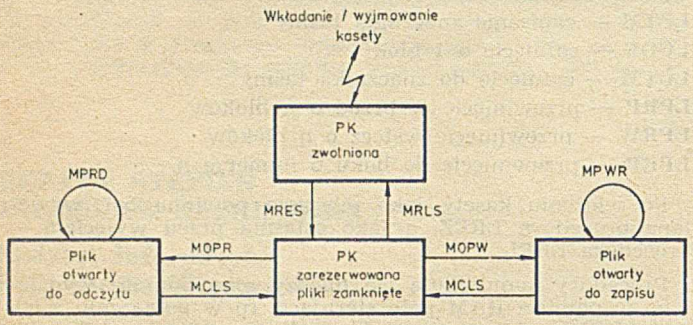
MOPW — otwarcie pliku do zapisu

MPRD — odczyt bloku z pliku (otwartego do odczytu)

MPWR — zapis bloku do pliku (otwartego do zapisu)

MCLS — zamknięcie pliku.

Kolejność użycia tych procedur nie może być przypadkowa lecz powinna wynikać z grafu stanów jednostki pamięci kasetowej (rys. 2).



Rys. 2. Zmiany stanów jednostki pamięci kasetowej pod wpływem procedur średniego poziomu

Procedury komunikują się między sobą za pomocą pól sterujących, odrębnych dla każdej jednostki pamięci kasetowej. Z każdą jednostką, która nie jest zwolniona, związany jest odrębny proces. W określonej chwili może wystąpić do czterech procesów, co jest istotne np. w czasie kopiowania zapisanego pliku do drugiej pamięci kasetowej. Podczas realizacji procesu sprawdzana jest legalność użycia wywoływanej procedury w danym stanie jednostki. Wszelkie nieprawidłowości są sygnalizowane przez podsystem komunikatów. Podsystem ten przekazuje także komunikaty o aktualnym stanie realizacji procesu oraz ewentualnych awariach technicznych. Po zakończeniu działania, procedury średniego poziomu pozostawiają w akumulatorze ślad, w postaci numeru komunikatu, informujący o sposobie zakończenia danej operacji.

Wszystkie procedury wymagają ustawienia odpowiednich parametrów, takich jak numer jednostki pamięci, nazwa pliku do otwarcia itp. Dwa z tych parametrów zostaną poniżej omówione.

Przed wywołaniem procedury MRES w rejestrze HL ustawia się adres, pod który zostanie przekazane sterowanie programu w wypadku zaistnienia technicznej awarii jednostki pamięci lub uszkodzenia nośnika. Adres ten obowiązuje przez cały czas realizacji procesu, aż do wywołania procedury MRLS. Drugi z omawianych parametrów dotyczy procedur MOPR i MOPW. Przed wywołaniem obu procedur ustawia się w rejestrze DE adres, pod który przekazane zostanie sterowanie programu w wypadku wykrycia końca pliku podczas operacji czytania lub końca taśmy podczas operacji pisania. Adresy te są ważne aż do chwili wywołania procedury MCLS. Oba parametry są opcjonalne. Podanie zer powoduje, że w opisywanych sytuacjach program wykonywany jest sekwencyjnie. W tym ostatnim wypadku wykrycie faktu zaistnienia sytuacji specjalnej i określenie jej rodzaju, programista powinien wykonać samodzielnie w oparciu o pozostawioną w akumulatorze informację zwrotną. Ustawienie ww. adresów przypomina ustawianie warunków ON ERROR, ON ENDFILE w języku PL/I, co jak wiadomo bardzo ułatwia pisanie programów.

Poziom wysoki tworzą dwie procedury: LOADFILE oraz DUMPFILE dostępne z programu użytkowego. Dla operatora są dodatkowo dostępne dwa podsystemy: DIGITAL CASSETTE DISPLAY oraz COPY. Procedury LOADFILE i DUMPFILE pozwalają odpowiednio wczytać cały plik z taśmy do pamięci operacyjnej lub zapisać plik z tej pamięci na taśmę kasety. Dzielenie pliku na bloki (DUMPFILE) lub łączenie pliku z bloków w PAO (LOADFILE) jest na tym poziomie dokonywane automatycznie i nie jest widoczne dla programisty. Procedury te są bezparametrowe, a wszystkie potrzebne dane uzyskują one w wyniku dialogu z operatorem (por. rys. 1). Operację ładowania i składowania pliku przeprowadzają obie procedury w oparciu o wywołania procedur poziomu średniego.

Podsystem DIGITAL CASSETTE DISPLAY (DCD) stworzono celem zapewnienia wygody użytkowania zestawu kaset. Umożliwia on operatorowi przeglądanie na ekranie monitora zawartości wybranych kaset ze zbiorami. Podsystem DCD udostępnia operatorowi operacje przeglądania w następujących dwu trybach:

- FILE OPERATIONS
- BLOCK OPERATIONS

W trybie FILE OPERATIONS podstawową jednostką danych na taśmie jest plik. Operator może przeglądać etykiety wszystkich zapisanych na taśmie plików. W trybie BLOCK OPERATIONS jednostką jest blok danych. Operator może wyprowadzać na ekran zawartość bloków danych żądanego pliku.

W trybie FILE OPERATIONS operatorowi są udostępniane na zasadzie konwersacji następujące operacje:

- przewinięcie do etykiety następnego pliku
- przewinięcie do etykiety poprzedniego pliku
- przewinięcie do etykiety ostatniego pliku
- przewinięcie do etykiety pliku o określonej nazwie
- przewinięcie do etykiety taśmy
- zmiana kasety
- przejście do trybu BLOCK OPERATIONS
- zakończenie pracy podsystemu DCD.

Efektem działania pierwszych pięciu operacji jest wyprowadzenie na ekran monitora etykiety pliku, do której taśma została przewinięta.

Wszystkie operacje mają odpowiednie zabezpieczenia i sygnalizacje dotyczące sytuacji nietypowych (np. przewinięcie do etykiety następnego pliku, w wypadku gdy głowica była ustawiona na etykiecie ostatniego pliku na taśmie). Operacje na plikach wykorzystują przewijanie ruchem szybkim, dzięki czemu podsystem DCD wykazuje znaczną sprawność działania.

W trybie BLOCK OPERATIONS operator może się posługiwać następującymi operacjami:

- odczytanie następnego bloku
- odczytanie poprzedniego bloku
- odczytanie bloku o numerze wskazanym przez operatora (w ramach jednego pliku)
- przewinięcie o n bloków w przód
- przewinięcie o n bloków wstecz
- powrót do trybu FILE OPERATIONS.

Efektem działania wszystkich operacji, z wyjątkiem ostatniej, jest wyprowadzenie zawartości bloku na ekran. Operacje blokowe działają zawsze w ramach pliku, a podsystem DCD zabezpiecza przed przekroczeniem jego granic.

Podsystem COPY pozwala kopiować pliki z jednej taśmy na drugą. Możliwe jest wierne kopiowanie plików oraz tworzenie pliku z połączenia wielu całych plików lub ich fragmentów. W odpowiednich fazach kopiowania udostępniany jest operatorowi podsystem DCD, za pomocą którego wskazuje on plik lub blok w pliku, który ma być skopiowany.

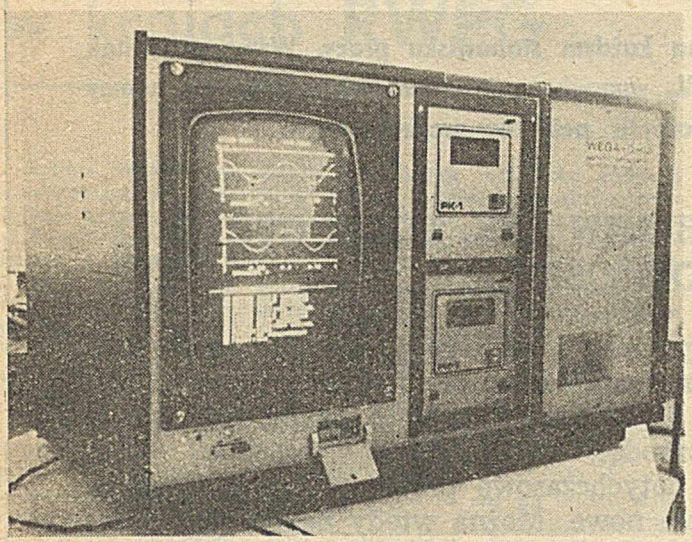
Podsystemy DCD i COPY są wygodnymi narzędziami umożliwiającymi utrzymanie zestawu kaset we właściwym porządku, przez wykonywanie duplikatów kaset, usuwanie zbiorów nieaktualnych itp.

* * *

System II-SOPK został zainstalowany w mikroprocesorowym urządzeniu do rejestracji i wstępnego przetwarzania sygnałów analogowych WEGA-D-02 zbudowanym przez

Instytut Informatyki PW do zastosowań terenowych (fot.) [5]. W części cyfrowej urządzenia wykorzystano opracowany również w Instytucie modułowy system mikroprocesorowy MSM [6]. Wejściowe sygnały analogowe są zamieniane na ciąg próbek cyfrowych, które są poddawane wstępnemu przetwarzaniu i transformacie Fouriera. Wyniki transformaty są zapisywane w pamięci kasetowej i mogą być później poddawane dalszemu procesowi przetwarzania w omawianym urządzeniu lub w dużym komputerze.

System II-SOPK w wersji zaimplementowanej w urządzeniu WEGA został wyposażony w rozbudowany podsy-



Urządzenie do rejestracji i wstępnego przetwarzania sygnałów analogowych WEGA-D-02

stem współpracy z operatorem, działającym w oparciu o zasadę wyświetlania menu na ekranie monitora.

Zajętość pamięci przez system II-SOPK ilustruje tabela.

Zajętość pamięci dla systemu II-SOPK

	Procedury			DCD COPY
	poziom niski	poziom średni	poziom wysoki	System komunikatów*) System współpracy z operatorem*)
Pamięć programu	3,5 K	6 K	0,3 K	4 K
Pamięć danych	0,3 K	0,2 K	—	0,25 K

*) w wersji rozbudowanej — jak w urządzeniu WEGA

LITERATURA

[1] Pluta A.: Programowa obsługa pamięci kasetowej. INFORMATYKA nr 2, 1984, str. 13—16

[2] Jackiewicz B.: Zastosowanie mikroprocesorów w aparaturze pomiarowo-kontrolnej do sprawdzania układów scalonych. INFORMATYKA nr 4, 1983, str. 3—7

[3] Buskirk W. A., Gilson C. W., Shelley D. J.: Compact Digital Cassette Drive for Low-Cost Mass Storage. Hewlett-Packard Journ., vol. 34, May 1983, pp. 17—24

[4] Tatarkiewicz J.: HP-IL. INFORMATYKA nr 3, 1984, str. 27—28

[5] Dotryw S., Panas S., Rudowski M., Rzeszut J., Stępień A., Stępień C., Szyrkowiec J., Zabrodzki J.: Mikroprocesorowe urządzenie do rejestracji i wstępnego przetwarzania sygnałów analogowych. Materiały z IV Krajowej Konferencji Naukowo-Technicznej SEP „Zastosowanie mikroprocesorów w automatyce i pomiarach", Warszawa, wrzesień 1984

[6] Woźniak A., Pawłowski M., Rzeszut J., Skorpuski A., Sosnowski J., Stochlak J.: Modułowy system mikroprocesorowy — MSM. INFORMATYKA nr 2, 1983, str. 17—21.

Zakład Elektronicznej Techniki Obliczeniowej

ZETO

ZIELONA GÓRA

SERWIS TECHNICZNY
ŚWIADCZY USŁUGI
W ZAKRESIE
REGENERACJI
TAŚM BARWIĄCYCH
DO DRUKAREK WIERSZOWYCH
WSZYSTKICH TYPÓW

GWARANTUJEMY
WYSOKĄ JAKOŚĆ USŁUGI

TERMIN REALIZACJI ZAMÓWIEŃ
DO 1 MIESIĄCA

PRZY LICZBIE TAŚM
POWYŻEJ 150 SZTUK
ZAPEWNIAMY WŁASNY TRANSPORT

65-021 ZIELONA GÓRA
ul. Dąbrowskiego 25
Telefony: 719-22, 604-79
Telex: 0432525

EO/127/K/85

CSK—Computer Studio Kajkowscy

81-505 GDYNIA ORŁOWO, ul. Balladyny 3B, tel. 29-00-18

Komputer osobisty może być przydatny niemal na każdym stanowisku pracy. Wymaga jednak odpowiedniego oprogramowania użytkowego. W ramach tego oprogramowania oferujemy zainteresowanym dostawę uniwersalnych pakietów programowych:

BANK DANYCH CSK, TABPLAN CSK, TEKST CSK, TRANSCOM CSK

To doskonałe narzędzia pracy dla każdego. Aby z nich korzystać, nie trzeba być informatykiem! Zupełnie samodzielnie można tworzyć złożone systemy zarządzania przedsiębiorstwem, każdym przedsiębiorstwem; nawet najbardziej specyficzne uwarunkowania nie są przeszkodą.
To jednak jeszcze nie wszystko... Kiedy dotychczasowe problemy łatwo i szybko zostały rozwiązane — pojawiają się zupełnie nowe. Można wtedy bez kłopotów samemu udoskonalić dotychczasowy system!

BANK DANYCH CSK, TABPLAN CSK, TEKST CSK, TRANSCOM CSK

składają się w zakładowe systemy płacowe, osobowe, finansowo-księgowe lub magazynowe. Korzystając z nich, z łatwością można prowadzić planowanie, kalkulacje i sprawozdawczość. Można też sporządzać kosztorysy i oferty, a nawet prowadzić „automatyczną" korespondencję czy redagować dowolne teksty. Można wreszcie skorzystać z już zgromadzonych zasobów na komputerze ODRA (pod nadzorem systemu GEORGE-3), wykorzystując komputer osobisty jako inteligentny terminal — stację lub emulator TTY.

Nowość:

Oferujemy system operacyjny kompatybilny z CP/M 2.2. dla mikrokomputerów ROBOTRON 5120/5130 oraz systemy finansowo-księgowe FK dla dowolnych mikrokomputerów

---

Szczegółowych informacji udziela:

CSK—Computer Studio Kajkowscy

81-505 GDYNIA ORŁOWO, ul. Balladyny 3B, tel. 29-00-18

EO/129/K/85

PETÉR BROCZKÓ

Budapeszt
Węgry

Produkcja mikrokomputerów w 1984 r. w krajach RWPG

Tabela stanowiąca istotę niniejszego artykułu zawiera szczegółowe charakterystyki mikro- i minikomputerów produkcji krajów RWPG wg stanu 1984 r. Wskutek swej obszerności jest ona stosunkowo trudna do analizy porównawczej. Poniższy opisowy komentarz wg poszczególnych krajów oraz krótkie podsumowanie sytuacji powinny pomóc Czytelnikom w wyrobieniu całościowego poglądu na ten temat.

BUŁGARIA

Do końca roku 1983 produkowano tylko kilka typów mikrokomputerów 16-bitowych, z których wszystkie były odpowiednikami produktów firmy DEC. W 1984 roku kontynuowano tę linię rozwoju, a opierając się na radzieckim mikroprocesorze powstał IZOT 1039. Nie to jest jednak największą nowością, lecz zaskakująco duży asortyment produktów zgodnych z IBM PC. W Instytucie Cybernetyki Technicznej i Robotyki Bułgarskiej Akademii Nauk opracowano IMKO-4 oraz rodzinę MIC 16A. Tę ostatnią zrealizowano w trzech wersjach: przenośnej, wyposażonej w dyski elastyczne oraz w dysk typu Winchester. Trzeba podkreślić, że przenośny MIC 16A — razem z kubańskim LTEL/2M — jest pierwszym 16-bitowym mikrokomputerem tego typu w krajach socjalistycznych.

Sofijski zakład Zjednoczenia IZOT wprowadził na rynek dwa modele mikrokomputerów. Standardową wersją jest IZOT 1036 a rozszerzoną IZOT 1037 — pierwszy i do niedawna jedyny w krajach RWPG mikrokomputer będący odpowiednikiem IBM PC XT. Ostatnim mikrokomputerem zgodnym z IBM PC jest PRAVEC 16, będący konstrukcją zakładów w Pravec.

Trzeci kierunek rozwoju bułgarskich mikrokomputerów 16-bitowych reprezentuje IZOT 1030, którego mikroprocesor jest radzieckim odpowiednikiem INTELA 8086.

Produkcja mikroprocesora CM 601, odpowiadającego funkcjonalnie mikroprocesorowi MOTOROLA 6800, do początku 1984 roku wyznaczała elektroniczną bazę bułgarskich mikroprocesorów 8-bitowych. W ten sposób Bułgaria odeszła od kierunku wybranego przez inne kraje socjalistyczne, gdyż poza nią mikrokomputery na bazie MOTOROLI 6800 produkuje się w małych seriach tylko na Węgrzech. Zmiany w bazie elektronicznej intensywnie rozwijającego się i zasadniczo zorientowanego na eksport przemysłu mikrokomputerowego Bułgarii były już nieuniknione.

W styczniu 1984 roku zaanonsowano IZOT 1031. Jego mikroprocesorem jest obecnie najbardziej rozpowszechniający się w krajach socjalistycznych mikroprocesor 8-bitowy Z80, a dokładniej jego funkcjonalny odpowiednik produkowany w NRD. Mikrokomputer ten, pracujący pod nadzorem systemu operacyjnego CP/M, wchodzi w skład trójczłonowej rodziny, do której należą poza nim wspomniane już IZOT 1036 i IZOT 1037.

W Bułgarii pojawiła się w 1984 roku rodzina mikrokomputerów PRAVEC. Nie jest to rodzina w ścisłym znaczeniu, bo główną cechą, która łączy poszczególne jej modele jest tylko nazwa, a dokładniej miejsce produkcji, gdzie zlokalizowano fabrykę komputerów (miasto urodzenia Todora Żiwkowa).

Najmniejszym produktem Praveckiej Fabryki Komputerów jest PRAVEC 8D, pierwszy bułgarski komputer domowy. Stanowi on funkcjonalny odpowiednik angielskiego mikrokomputera ORIC-2 i jest dotychczas jedynym komputerem tej kategorii w krajach RWPG, zgodnym z konstrukcją zachodnią. PRAVEC 8B i PRAVEC 8M stanowią zmodernizowane wersję PRAVEC-82, które zachowały nadal zgodność z APPLE II. Spośród komputerów zakładów PRAVEC największą moc obliczeniową ma wspomniany już PRAVEC 16.

W roku 1984 ukazały się tylko dwa wyroby reprezentujące tradycyjną linię MOTOROLA 6800. Są to IZOT 1029 i IZOT 1035, stanowiące funkcjonalne odpowiedniki modeli IZOT 1001 i IZOT 1003, które były już produkowane w 1980 roku. Zmieniono jedynie obudowy na bardziej nowoczesne.

CZECHOSŁOWACJA

W roku 1984, w porównaniu z okresem wcześniejszym, nie zanotowano znaczących zmian asortymentu czy charakterystyki produkowanych mikrokomputerów. Warto jednak zauważyć, że koncentrowano się na wykorzystaniu w produkcji rezultatów rozwoju lat ubiegłych. Postawiono sobie cel stworzenia możliwości zakupu mikrokomputera przez szeroki krąg nabywców.

W roku 1984 zademonstrowano pierwszy czechosłowacki mikrokomputer 16-bitowy SMEP PP 04. Jest to maszyna dająca postawić się na biurku, stanowiąca kontynuację tradycyjnej w Czechosłowacji linii DEC, ze wspaniałymi możliwościami prezentacji graficznej i manipulowania kolorami.

Spośród mikrokomputerów 8-bitowych najlepsze perspektywy rysują się przed rodziną SMEP, głównie ze względu na znaczne zwiększenie możliwości produkcyjnych. SMEP PP 02 i SMEP PP 03 są mikrokomputerami profesjonalnymi. W przypadku tego ostatniego interesujący jest fakt, że tworzy on bazę sprzętową dla małego mikrokomputera biurowego TEKST 01.

Najmniejszy mikrokomputer z rodziny SMEP został opracowany głównie z przeznaczeniem dla szkół, ale można go także stosować jako komputer domowy. Do tej samej kategorii należy IQ 151, którego 500 egzemplarzy przekazano do szkół średnich w 1984 r., a w roku bieżącym planuje się trzykrotne zwiększenie tych dostaw.

KUBA

Kontynuowana jest tradycyjna linia mikrokomputerów CID, w ramach której szczególnie godnym uwagi jest mikrokomputer CID-1408. Oparto go na starszym modelu MIKROCID-02, który już w 1980 r. został zaakceptowany w kategorii SM 50/40-1 z symbolem SM 1625. CID-1408, oparty na mikroprocesorze INTEL 8080, zawiera do 64 KB pamięci operacyjnej i pracuje pod nadzorem systemu operacyjnego CP/M.

W 1984 r. pojawił się minikomputer CID 30-20, będący odpowiednikiem PDP 11/40 i należący również do rodziny SM-4.

Znaczące osiągnięcia kubańskiej mikroelektroniki to system teleprzetwarzania TELEMIC-01, programator pamięci EPROM oraz tester pamięci operacyjnej CID-7220.

W roku 1984 zaprezentowano dwa mikrokomputery odpowiadające IBM PC. Są to LTEL/2 M i LTEL/PC. LTEL/PC stanowi wersję przenośną. Trwają prace nad

Mikro- i minikomputery produkcji krajów RWPG wg stanu 1984 r.

Kraj	Model	Producent	Typ mikroprocesora[1])	Liczba bitów	Pojemność pamięci w Kbajtach[2])	LT[3])	System operacyjny	Język programowania[4])	Orientacyjna cena[5])
1	2	3	4	5	6	7	8	9	10
BUŁGARIA	IMKO-3	Fabryka Komputerów, Pravec	SM 600 (R6502)	8	64/8	1	APPE-II KOMP.	B, P	5000 BGL
	IMKO-16	jw.	INTEL 8088	16	256-512/16	1	MS-DOS, CP/M-86	A, B, F, P	×
	IZOT 1029	IZOT, Sofia	SM 601 (M 6800)	8	32	1		IZOT 1003[6]	
	IZOT 1030	jw.	K 1810VM86	16	192—1024	1	MS-DOS, CP/M-86	INTELEC-III[6])	×
	IZOT 1031	jw.	U 880 (Z80)	8	64	1	UMCO (CP/M)	A, B	×
	IZOT 1035	jw.	SM 601 (M 6800)	8	32	1	rejestrowanie obecności	IZOT 1001[6])	×
	IZOT 1036	jw.	INTEL 8088	16	64-256/40	1	MS-DOS	IBM PC[6])	×
	IZOT 1037	jw.	INTEL 8088	16	64-640¢64	1	MS-DOS	IBM PC XT[6])	×
	IZOT 1039	jw.	K 1801VM2	16	64-512	1	RT-11[6])	PDP[6])	×
	MIC 16A	Instytut Robotyki, Sofia	INTEL 8088	16	64-256/40	1	MS-DOS	IBM PC[6])	×
	PRAVEC 8B	Fabryka Komputerów, Pravec	SM 600 (R6502)	8	64	1	PRAVEC (CP/M)	APPLE-II[6]	×
	PRAVEC 8D	jw.	U 880 (Z80)	8	16	1	ORIC-2[6])	A, B, F, P, PILOT	×
	PRAVEC 8 M	jw.	SM 600 (R6502)	8	64	1	PRAVEC (CP/M)	APPLE-II[6]	×
	PRAVEC 16	jw.	INTEL 8088	16	64-256/40	1	MS-DOS	IBM PC[6])	×
	SM 1504	IZOT, Sofia	×	16	×	19	VAX 11/780[6]	ALTALANOS CELU	×
CZECHOSŁOWACJA	IQ 151	ZPA, Novy Bor	MHB 8080 (8080)	8	32-64/8	1	MONITOR	A, B	20 000 CSK
	MARS/SM 53-10	ZPA, Čakovice	jw.	8	×	19	FOLY. IR.	×	×
	MVS 810	TESLA, Kolin	jw.	8	4-48/4-16	1	INTELEC-MEOF.	×	×
	PMD 85	TESLA, Pieštany	jw.	8	48/12	1	HP 85	A, B	11 000 CSK
	SMEP SP 01	VUVT, Žilina	jw.	8	32/8	1	MONITOR	A, B	29 000 CSK
	SMEP PP 02	jw.	jw.	8	40-64/8	1	MONITOR	A, B, C	60 000 CSK
	SMEP PP 03	jw.	jw.	8	64/8	1	MIKROS (CP/M)	A, B, C, F	130 000 CSK
	SMEP PP 04	ZVT, Banska Bystrica	jw.	16	256	19	FOBOS, DOS-RV	B, F	×
	SM 1505 (52/12)	ZVT, Namestovo	×	16	×	19	VAX 11/780[6])	ALTALANOS CELU	×
	TEKST 01/SM 6915	ARITMA, Praha	MHB 8080 (8080)	8	64/8	1	MIKROS (CP/M)	A, B, C, F	150 000 CSK
NRD	System 16-bitowy	ROBOTRON	U8000 Z8000	16	32/2	1	księgowość	A, B	×
	A 5510	jw.	U880 Z80	8	64	1	CP/M	graficzny	×
	HC 900	MIKROELEKTRONIK, Mühlhausen	U880 Z80	8	32/8	1	MONITOR	A, B	1500—2000 DDM
	R 1715	ROBOTRON	U880 Z80	8	64/8	1	CP/M	A, B, P, PLM f	×
	Z 9001	jw.	U880 Z80	8	16/12	1	MONITOR	A, B	1500—2000 DDM
POLSKA	AC 825	AMEPROD	U880 (Z80)	8	48	1	CP/M 2.2	A, B	×
	COMPAN	MERA-ELZAB	MCY 7880 (8080)	8	64	1	CP/M 2.2, ISIS	A, B, F, P, FORTH	×
	CS 80	COMPUTEX	U880 (Z80)	8	64	1	CP/M 1.4	A, B	×
	IMP 85	IMPOL II	INTEL 8085A	8	64	1	CP/M 1.4	A, B, C, F	×
	MERITUM	MERA-ELZAB	U880 (Z80)	8	18/12	1	TRS 80-1[6])	A, B	30 000 HUF
	MK 4501/2	MERA-KFAP	INTEL 8085	8	64	1	CP/M 1.4 (IMP-85)[6])	A, B, F	×
	MSA 80	UNITRA SZCZYTNO	MCY 7880 (8080)	8	64-512/12	×	×	B, ASM-80, PLM-80	×
	NEPTUN 184	Prac. Elektroniki Medycznej	ROCKWELL 6502	8	32	1	SAJAT	A, B	×
	RTDS 8	MERA ELZAB	INTEL 8085	8	16/8	1	CP/M[6])	A, B, FORTH	×
	ZX 81 POLSKI	AMEPROD	U880 (Z80)	8	1-64/8	1	MONITOR	A, B	180 000 PLZ
RUMUNIA	M216	ICE FELIX	INTEL 8086, 8080	16	128-104/32	1	SFDX (ISIS), CP/M	A, B, F, P	×
	DAF 2015	FEPER	INTEL 8080	8	×	1	TERMINAL	×	×
	PRAE 1000	x, Cluj	jw.	8	×	1	MONITOR	×	×
	PRIM XX	IPA	jw.	8	×	×	sterowanie procesami	×	×

1	2	3	4	5	6	7	8	9	10
WĘGRY	AGRINFO-100	LIGNIFER	U880 Z80	8	64	1	AGRINFO CP/M[6])	A, B, F	500 000 HUF
	AIRCOMP-64	BOSCOOP-PERSONAL GT	U880 Z80	8	64/16	1	PGT-DOS (CP/M)[6])	A, B	250 000 HUF
	AX-II	ALK. IFJ. EGY.	ROCKWELL 6502	8	48-256x)	5	APPLE-II CP/M[6])	A, B, F, P	105 000 HUF
	IPT 002	TRITON GMK	Z80	8	16-64/8	1	MONITOR	A, B	19 900 HUF
	PRIMO	MICROKEY KFTT	U880 (Z80)	8	16-48/16	1	MONITOR	A, B	15-24 000 HUF—
	PROCOM 16	SZKI	×	16	1024-4096	16	RSX-11	B, F	×
	PROFESSOR	COMPROJECT GMK	M 68000	16	256-16384/16-32	19	UNIX, OASIS, CP/M-68[6])	A, B, P, C	3 000 000 HUF
	PROPER 16W/1833	SZKI	INTEL 8088	16	832/48	1	PROPOS-16	B, F, P	900 000 HUF
	TRANSMIC 16	MÜSZERTECHNIKA GMK	M 68000	16	256-1024	8	CP/M-68[6])	A, B, F, P, C-NYELY	800 000 HUF
	TV-COMPUTER	VIDEOTON	U880 (Z80)	8	32-64/8	1	MONITOR	A, B	16 000 HUF
	VT-16	VIDEOTON	INTEL 8088, Z80	16	256	1	CP/M, CP/M-86	A, B, C, F, P	599 000 HUF
	VT-32	VIDEOTON	M 68000	16	512-2048	19	SOS (UNIX)	A, B, F, P	×
	ALFA-2X	DATAKOORD	Z80	8	16/12	1	rejestrowanie danych	A	45 000 HUF
	BURO-X	DATAKOORD	Z80	8	64/12	1	VP/M (CP/M)	przetwarzanie tekstów	175 000 HUF
	COMP-X	DATAKOORD	Z80	8	64/12	1	VP/M (CP/M)	A, B, P	175 000 HUF
	CTX 80	COMPUTEXT	Z80	8	64/8	1	MSYS (CP/M)	A, B, F, P	350 000 HUF
	MINOMOD	MEDICOR	U880 (Z80)	8	32-64/32	1	MSYS (CP/M)[6])	A, B	410 000 HUF
	MOBI-X	DATAKOORD	INTEL 8085	8	2	1	rejestrowanie danych	A	19 000 HUF
	MULTI CENTER	MÜSZERTECHNIKA	Z80	8	256	8	MP/M	A, B, F, P	0,8—1,6 mln HUF
	NEZ 215	MTA KFKI	×	8	64/8	1	×	×	×
	PROLOCON D2	VILATI	INTEL 8085	8	64/24	1	sterowanie procesami	TRANSIT-85+SAJAT	80 000 HUF
	SAMDS	MMG	INTEL 8085	8	64	1	FDOS (CP/M)[6])	A, B, P, PLM	450 000 HUF
	TM 16	MÜSZERTECHNIKA GMK	M 68000	16	256-1024x)	8	TRSDOS, MSYS, CP/M-68[6]	A, B, F, P	1 400 000 HUF
	TRDS	KONTAKTA	INTEL 8085	8	64	1	TRDS, IFDS, CP/M	A, B	500 000 HUF
	VM 02	MTA SZTAKI	U 800 (Z8000)	16	128-256	1	przetwarzamie obrazów	×	×
	VOLAN	VOLAN ELEKTRONIKA	Z80	8	64	1	rejestrowanie danych	×	×
ZSRR	AGAT	×	K588VS2+K588VU2	8	×	1	CP/M	A, B, F, P	3 000 SUR
	DVK-2 M	×	×	8	64	19	×	×	×
	ELEKTRONIKA 60-1	VUM, Kisów	×	16	256-1024	19	PDP 11/23[6])	B	×
	ELEKTRONIKA 79	VUM, Kijów	×	16	×	19	×	TOBB	×
	ELEKTRONIKA ×	VUM, Kijów	×	16	4096	19	PDP 11/23 PLUS[6])	TOBB	×
	ELETRONIKA 100-25	VUM, Kijów	×	16	256	19	PDP[6])	TOBB	×
	ELEKTRONIKA BK 0010	x, Moskwa	KR580IK80A	8	16-32	1	MONITOR	A, B	800—2000 SUR
	ELEKTRONIKA DZ 28	SVETLANA, Leningrad	KR580IK80A	8	16-32	1	MONITOR	A, B	800—2000 SUR
	ISKRA 250	×	K1810NV86	16	64-256/40	1	MS-DOS	IBM PC[6])	×
	ISKRA 555	×	K589IK02	8	32/4	1	księgowość	×	×
	ISKRA 2106	×	KR580IK80A	8	4-16	1	księgowość	×	×
	NEVA 501	×, Leningrad	KR580IK80A	8	4-32/4	1	księgowość	×	×
	SM 1210	×	×	×	×	×	DOS RV	×	×
	SM 1410	×	×	16	×	19	PDP 11[6])	×	×
	SM 1420	×	×	16	4096	19	RAFOS, FOBOS	TOBB	×
	SM 1600	×	×	16	64	1	DIAMS, DOS RV	TOBB	×
	V7	×	MCY 7848 8048	8	16/8	1	MONITOR	B	×
	VEF MIKRO 1021	VEF, Riga	KR580IK80A	8	16-64	1	MONITOR	A, B	800—2000 SUR

[1]) w nawiasach odpowiednik produkcji zachodniej

[2]) dotyczy pamięci operacyjnej (min-max); druga część zapisu (po kresce ukośnej) określa pojemność pamięci ROM

[3]) liczba terminali

[4]) użyto następujących skrótów: A = ASSEMBLER, B = BASIC, C = COBOL, F = FORTRAN, P = PASCAL

[5]) w walutach krajowych — symbole walut wg normy ISO: BGL = lewa, CSK = korony, DDM = marki, HUF = forinty, SUR = ruble

[6]) kompatybilny

x) brak danych

rozwojem oprogramowania, w tym nad systemami operacyjnymi odpowiadającymi CP/M i MS-DOS.

Główną bolączkę kubańskiej informatyki stanowi brak urządzeń peryferyjnych, a szczególnie drukarek i dysków. Są one importowane głównie z krajów socjalistycznych.

NRD

Pierwszy 16-bitowy mikrokomputer zaprezentowano wiosną 1984 r. Oparto go na funkcjonalnym odpowiedniku Z8000, produkowanym w NRD pod nazwą U8000.

Wśród 8-bitowych mikrokomputerów profesjonalnych najbardziej popularny był automat księgujący ROBOTRON 1715, którego system operacyjny jest odpowiednikiem CP/M. Oczekuje się także, że w krajach socjalistycznych sukces odniesie mikrokomputerowy system graficzny A 5510.

W roku 1984 ukazały się również dwa modele mikrokomputerów domowych. Model Z9001 jest produktem kombinatu ROBOTRON, natomiast HC 900 dostarczają zakłady Mühlhausen, które dotychczas produkowały tylko kalkulatory elektroniczne.

POLSKA

Trudności ekonomiczne kraju spowodowały, że przyspieszenie rozwoju i produkcji mikrokomputerów wymagało włączenia do realizacji tych zadań również małych przedsiębiorstw. Osobliwością jest tu fakt, że małe przedsiębiorstwo może być utworzone w kooperacji z firmami zagranicznymi, a partner zachodni istotny jest w pierwszym rzędzie z punktu widzenia zaopatrzenia w elementy mikroelektroniczne.

W grupie mikrokomputerów 8-bitowych można wymienić model RTDS-8, pokazany na wszystkich międzynarodowych targach w krajach RWPG, gdzie można było oglądać również model NEPTUN 184. W grudniu 1983 r. pojawił się również model ELWRO 500.

Mikrokomputer AC 825 jest wyrobem małego przedsiębiorstwa z kapitałem zagranicznym AMEPROD. Jest to jedyny w kraju model, wyposażony w pamięć dyskową typu WINCHESTER. AMEPROD zajmuje się również montażem ZX 81. Innym reprezentantem w kategorii komputerów domowych jest MERITUM, pokazany na ubiegłorocznych Wiosennych Targach Lipskich.

RUMUNIA

Wiosną ub. r. zaprezentowano 16-bitowy mikrokomputer M 216 oparty na mikroprocesorze INTEL 8086 i pracujący pod nadzorem systemu operacyjnego SFDX, stanowiącego adaptację systemu ISIS. Model ten produkowany jest w największej rumuńskiej fabryce komputerów ICE-FELIX, co rokuje rozpoczęcie produkcji wielkoseryjnej.

Spośród mikrokomputerów 8-bitowych nie pojawił się żaden nowy model o większym znaczeniu, a jedynie trzy modele ukierunkowane na specyficzne zastosowania i w związku z tym produkowane w niewielkich ilościach.

WĘGRY

Kontynuowano produkcję oraz rozwój konstrukcji mikrokomputera PROPER 16, będącego pierwszym w krajach socjalistycznych odpowiednikiem IBM PC. Prace badawczo-rozwojowe doprowadziły do skonstruowania modelu PROPER 16W, wyposażonego w dysk typu WINCHESTER. Do nowości należy zaliczyć też VT 16, firmy VIDEOTON, prezentujący inny odpowiednik IBM PC.

W roku 1984 ukazały się cztery, pierwsze na Węgrzech, a jednocześnie w krajach socjalistycznych, modele mikrokomputerów opartych na procesorze MOTOROLA 68000. Trzy z nich opracowały małe firmy, a jeden spośród nich — TRANSMIC 16 — jest dotąd pierwszym i zarazem jedynym węgierskim 16-bitowym mikrokomputerem przenośnym.

W końcu 1984 r. ukazały się pierwsze mikrokomputery oparte na mikroprocesorze Z8000. Jako pierwszy pojawił się model VM 02.

Wspólną cechą nowych modeli mikrokomputerów 8-bitowych jest to, że najczęściej oparte są na procesorze Z80 lub jego funkcjonalnym odpowiedniku U880„ produkowanym w NRD.

Wśród konstrukcji rozwojowych należy wymienić MULTI CENTER, umożliwiający tworzenie tanich sieci lokalnych (do 8 terminali).

Rodzina mikrokomputerów X przeznaczona jest głównie do zbierania danych w miejscu ich powstawania. Rozwinięto ją opierając się na przenośnym urządzeniu do rejestracji danych z własnym zasilaniem MOBI-X.

W kategorii komputerów domowych, na szczególną uwagę zasługuje pojawienie się mikrokomputera PRIMO, którego do końca ubiegłego roku wyprodukowano 3000 egz. oraz nowego modelu TV COMPUTER firmy VIDEOTON, którego produkcja rozpocznie się w 1985 roku.

Rok 1984 był na Węgrzech rokiem pojawienia się szerokiego asortymentu mikrokomputerów 16-bitowych. Nowe modele mikrokomputerów 8-bitowych należały do kategorii mikrokomputerów domowych albo były ukierunkowane na spełnianie specyfizycznych wymagań pewnych grup użytkowników.

ZWIĄZEK RADZIECKI

W połowie ubiegłego roku ukazał się oczekiwany, najbardziej znaczący model radzieckiego mikrokomputera 16-bitowego ISKRA 250, będący odpowiednikiem IBM PC.

W kategorii mikrokomputerów 8-bitowych rok 1984 zapisał się powstaniem nowej klasy tych urządzeń — mikrokomputerów domowych. Wśród nich wyróżnia się AGAT — odpowiednik APPLE II, który ma być produkowany i rozprowadzany na dużą skalę. Już dziś produkuje się go całkowicie z radzieckich elementów. Oprócz tej inicjatywy jednego z zakładów moskiewskich, również w innych miastach, np. w Leningradzie i Rydze, produkowane są mikrokomputery domowe oparte na radzieckim odpowiedniku procesora INTEL 8080.

* * *

Możemy stwierdzić, że w 1984 r. każdy spośród europejskich krajów socjalistycznych produkował mikrokomputery 16-bitowe. Warto podkreślić, że liczba modeli mikrokomputerów, zgodnych z IBM PC wzrosła z jednego (w 1983 r.) do trzynastu (w 1984 r.). Baza elektroniczna produkowanych w obozie socjalistycznym mikrokomputerów 16-bitowych pochodzi najczęściej ze Związku Radzieckiego lub z NRD, choć jest także uzupełniana importem z krajów kapitalistycznych. Głównym prototypem systemów operacyjnych jest UNIX.

Zastosowanie profesjonalnych 16-bitowych mikrokomputerów o większej mocy obliczeniowej ogranicza mała podaż pamięci dyskowych typu WINCHESTER, których wytwarzanie należy możliwie szybko rozpocząć.

W grupie mikrokomputerów 8-bitowych głównym zadaniem w 1983 r., a także na początku 1984 r., było przygotowanie wielkoseryjnej produkcji sprzętu profesjonalnego o profilu maszyn serii CM 50/40, działającego pod nadzorem systemu operacyjnego CP/M. O ile chodzi o bazę elementową, to nowocześniejszy mikroprocesor Z80 stosowany jest dotąd tylko w niektórych modelach.

Rok 1984 można również uznać za rok pojawienia się mikrokomputerów domowych, bowiem każdy europejski kraj socjalistyczny produkuje już mikrokomputer należący do tej kategorii.

Tłum.

ROMAN GRABOWICZ

ERRATA

Do artykułu dr. Wacława Iszkowskiego „Superkomputer CRAY", jaki zamieściliśmy w nr 5, 1985 INFORMATYKI, wkradł się przykry, systematyczny błąd. W treści całego artykułu, przy parametrach określających czasy działania układów lub wykonywania operacji, podano zamiast nanosekund (ns) — mikrosekundy (μs), co oczywiście całkowicie zdeklasowało słynny superkomputer.

Przepraszamy wszystkich Czytelników za tego rodzaju nieporozumienie.

Redakcja

Od dłuższego czasu staramy się przekonać Czytelników, że współczesne zastosowania komputerów osobistych odbiegają bardzo daleko od pierwotnej funkcji komputera (ang. compute — liczyć). Chociaż przy przetwarzaniu danych komputery faktycznie wykonują dużo obliczeń (stąd testy szybkości i dokładności dla komputerów) jest to jednak czynność zupełnie niedostrzegalna z poziomu użytkownika i równie mało interesująca jak procesy chemiczne w brzuchu konia pociągowego.

W poniższym tekście Jakub Tatarkiewicz opisuje kolejne „nieliczące" zastosowanie komputera, mimo że przeznaczone tym razem dla osób... intensywnie posługujących się matematyką. „Osobista" wersja programu MACSYMA rozprowadzana jest przez firmę MICROSOFT pod nazwą muMAT w cenie ok. 250 dolarów. Oferowana jest wersja dla komputerów 8-bitowych, pracująca pod kontrolą systemu operacyjnego CP/M, jak również dla 16-bitowych IBM-PC, pracujących pod systemem PC-DOS.

Spodziewając się dużego zainteresowania programem, skonsultowaliśmy z autorem tekstu możliwość uzyskiwania przez Czytelników muMATa. Otóż właściciel (zapewne nie jedyny w kraju) bacząc na poniesione koszty nie zamierza go rozdawać. Nie mniej nadesłane pod adresem redakcji propozycje wymiany na inne programy zbliżonej klasy będą wnikliwie rozpatrzone.

Obliczenia symboliczne

Wszyscy przyzwyczailiśmy się już do użycia komputera jako nieco bardziej skomplikowanego liczydła: wszelkie problemy numeryczne (typu wyliczenia wartości całki, rozwiązania układu równań czy znalezienia wartości własnych macierzy) potrafimy rozwiązywać szybko i stosunkowo dokładnie. Stosunkowo, bo wiele problemów, napotykanych przez naukowców i studentów nie da się policzyć d o k ł a d n i e. Typowym przykładem są wszelkie równania, w których występuje liczba **pi** (3.1415... gdzie kropki oznaczają nieskończone rozwinięcie). Oczywiście w praktyce interesuje nas tylko skończona dokładność rozwiązań, bo pomiary na podstawie których opisujemy otaczający nas świat również mają skończoną dokładność. Czasami jednak istotna jest nie tylko przybliżona wartość całki, ale też jej postać analityczna. Żartobliwym przykładem może być egzamin, w którym złośliwy asystent użył funkcji, dającej rozwiązania zależne od **pi**. Nic nam wtedy nie pomoże najlepszy komputer — trzeba mozolnie rozwiązywać problem, myląc się zwykle po drodze wiele razy. W praktyce jedynie analityczna postać funkcji pozwala na podjęcie prób uogólnienia mechanizmów badanego zjawiska.

Od niedawna sytuacja uległa jednak zasadniczej zmianie. Dzięki pracy grupy informatyków z MIT, rozpoczętej w roku 1969, po dziesięciu latach (ponad 100 osobolat pracy) pojawił się program MACSYMA. Potrafił on manipulować s y m b o l i c z n y m i wyrażeniami algebraicznymi, abstrahując od ich wartości numerycznej. Na przykład, MACSYMA potrafi wyznaczyć wyrażenie analityczne będące wynikiem całkowania skomplikowanej funkcji. Niestety, program w obecnej wersji składa się z ponad 300 tysięcy linii kodu w języku LISP i może być uruchomiony tylko na dużych systemach komputerowych (np. VAX firmy DEC). Co gorzej, program jest bardzo drogi: licencja na używanie programu kosztuje ok. 2000 dol. plus liczne opłaty za pomoc techniczną, konsultacje itp.

Dopiero stosunkowo niedawno pojawiły się uproszczone wersje tego programu na komputery osobiste. Pojawiły się one też już w Polsce! Nieliczni jeszcze szczęśliwcy, którzy je posiadają, mogą za przyciśnięciem paru klawiszy uzyskać wyniki jakie dotąd były okupione wielogodzinnymi rachunkami i w dodatku często obarczone błędami.

A oto przykładowe użycie programu muMAT na komputerze IBM PC:

```
INT(X^2+X+1,X);
@: X + X^2/2 + X^3/3
```

Zadaliśmy stosunkowo łatwą funkcję i poprosiliśmy komputer o jej scałkowanie. Po kilku sekundach otrzymaliśmy powyższy wynik. Możemy go sprawdzić zlecając wykonanie odwrotnej operacji:

```
? DIF(@,X,1);
@: 1 + X + X^2
```

Nie przypadkowo jako ilustrację działania programu wybrałem całkowanie symboliczne. Wszyscy, którzy zetknęli się z analizą matematyczną, wiedzą, że różniczkowanie czy rozwiązywanie równań algebraicznych podlega prostym regułom. Nauka metod rozwiązywania tego typu problemów polega w gruncie rzeczy na przyswojeniu sobie tych reguł. Z całkowaniem problem jest nieco bardziej skomplikowany. Wiemy, że różniczkowanie wyniku powinno dać funkcję podcałkową. Jest to jednak jedyna reguła i nie daje ona żadnych wskazówek co do metody postępowania. Okazało się jednak, że dla całek nieoznaczonych, które dadzą się rozwiązać w postaci skończonej (a nie w postaci nieskończonego szeregu), można podać ogólną postać algebraiczną rozwiązania (dla „materializacji" tego przewidywania potrzebne były doświadczenia wielu pokoleń matematyków!). Dalej program poprzez różniczkowanie znajduje odpowiednie współczynniki i podaje wynik jako funkcję. W wielu przypadkach możemy też podać j a k a jest postać szeregu nieskończonego. Jako ciekawostkę należy podać fakt, że eksploatując program stwierdzono, iż większość tablic całek nieoznaczonych zawiera mniej więcej 10 procent błędnych odpowiedzi, a są i takie tablice gdzie co c z w a r t a całka jest źle policzona! A oto przykład całkowania, które daje w wyniku szereg nieskończony (całka z rozkładu Gaussa):

```
? INT(#E^(-X^2),X);
@: #PI^(1/2) ERF (X)/2
```

Jako wynik otrzymujemy dobrze znaną funkcję błędu. Proszę zwrócić uwagę, że program podał stałą jako **pi** a nie jako liczbę! A oto kilka innych przykładów całkowania:

```
? INT(X^3/(1+X),X);
@: 11/6 + X - X^2/2 + X^3/3 - LN (1 + X)

?INT((LN(1. /(1+X))^2),X);
@: 2 + 2 X + X LN (1 + X)^2 - 2 X LN (1 + X) + LN (1 + X)^2 - 2
LN (1 + X)
```

Prawda, że te wyniki robią wrażenie?!
Niestety, ze względu na niewielką pojemność pamięci komputerów osobistych (w tym przypadku trzeba i tak użyć 256 KB pamięci), nie wszystkie całki dadzą się policzyć:

```
? INT(TAN(LN(1/(1+X))),X);
@: -INT (SIN LN (1 + X)/COS LN (1 + X), X)

? INT(SIN(LN(1+X)),X);
@: INT (SIN LN (1 + X), X)

? INT(LN SIN(1+X),X);
@: INT (LN SIN (1 + X), X)
```

Nie wiem, czy takie całki przekraczają możliwości IBM PC, czy też może po prostu mają jedynie rozwiązania w postaci szeregów nieskończonych — program nie daje żadnej wskazówki, a szkoda...

Program muMAT potrafi wiele — manipulowanie macierzami, rozwiązywanie równań różniczkowych (oczywiście poprzez całkowanie symboliczne!), rozwiązywanie równań

algebraicznych — to typowe przykłady problemów nie przysparzających kłopotów programowi muMAT.

Reklamy programów symbolicznego manipulowania zmiennymi podają przykład, który zapiera dech w piersiach. Otóż w XIX wieku francuski astronom Charles Delaunay przez 10 lat wyznaczał analityczny wzór na ruch Księżyca wokół Ziemi. Problem nie jest prosty (ruch trzech ciał w polu grawitacyjnym) i dlatego następne dziesięć lat trwało sprawdzanie wyników obliczeń. Prace zostały ukończone w 1867 roku. Od tego czasu tablice, oparte na wynikach obliczeń Delaunaya, były używane przez pokolenia astronomów. Dopiero w roku 1970, po 20 godzinach pracy program MACSYMA znalazł błąd! Całe szczęście, że dotyczył on tylko czynnika przed jednym z wyrażeń (cały wzór zajmuje wiele stron) zamiast 23/16 powinno być 33/16 i ten niewielki błąd nie powoduje większych różnic w końcowych wynikach. A więc proporcja: 20 lat i 20 godzin! Czym się to wszystko skończy? Może już niedługo będziemy świadkami masowego poprawiania starych prac naukowych?! W każdym razie fizyka teoretyczna i wiele innych specjalności, szczególnie w matematyce stosowanej, gdzie dotąd królowały kartka papieru i ołówek, zyskały potężne narzędzie pracy. Miejmy nadzieję, że i w Polsce komputery profesjonalne będą dysponowały programami symbolicznego obliczania wyrażeń.

JAKUB TATARKIEWICZ

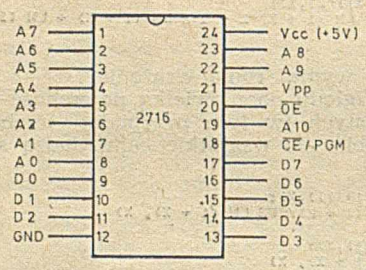
Opublikowany w mikroKLANIE 12 schemat programatora pamięci 2716 wywołał falę listów i telefonów z prośbami o dane katalogowe zarówno samego układu EPROM, jak i algorytmu programowania. Przykro odmawiać potrzebującym, ale redaktor mikroKLANU nie jest w stanie odwalić roboty, którą powinny wykonywać branżowe ośrodki informacji naukowej i technicznej. Zresztą jeśli je o to poprosić, to ją wykonują (np. BOINTE, Warszawa, ul. Marynarska 10), tyle że za kwoty mocno nadwyrężające prywatną kieszeń, a na dodatek raczej niespiesznie.

Ponieważ układy EPROM są dość powszechnie stosowane w systemach mikroprocesorowych, a układ 2716 należy do tych najpopularniejszych, sądzimy, że zamieszczony opis rozwiąże problemy wielu Czytelników — tym bardziej, że został wzbogacony o informacje, których nie ma w katalogu.

W przyszłości nadal będziemy zamieszczać opisy tych układów, o które najczęściej dopytują się Czytelnicy (lub układów, które naszym zdaniem należy lansować), ale prosimy pamiętać, że mikroKLAN nie jest drukowanym w odcinkach katalogiem.

Pamięć EPROM 2716

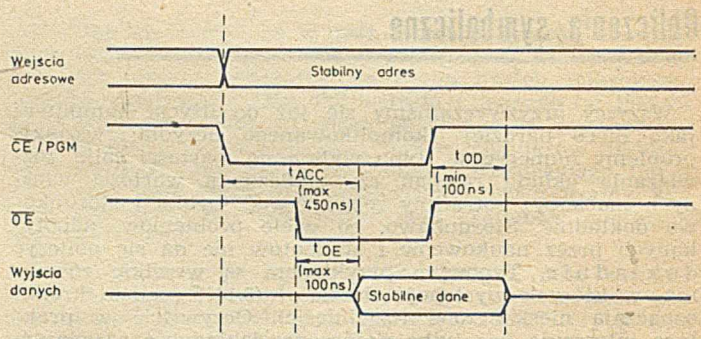
Topografia wyprowadzeń układu 2716 przedstawiona została na rysunku 1. Układ ma organizację 2 K × 8 bitów i jest następcą układów 2704 i 2708 firmy INTEL. Od poprzedników odróżnia go nie tylko większa pojemność, ale też pojedyncze napięcie zasilania (+5V) i uproszczony algorytm programowania. Sposób programowania jest identyczny jak dla późniejszych układów EPROM 2732, 2764, 27128 i wreszcie 27256.



Rys. 1. Topografia wyprowadzeń układu 2716 firmy INTEL

Realizacja cyklu odczytu zawartości określonej komórki pamięci wymaga wymuszenia niskiego poziomu na wejściu CE/-PGM. Podanie poziomu wysokiego na to wejście przełącza układ w tryb „uśpienia" (ang. power down). W tym trybie pobór mocy redukowany jest o ok. 75%, co poważnie odciąża zasilacz, ale też powoduje silne impulsy prądowe przy przechodzeniu układu w stan aktywny — może być to źródłem zakłóceń dla innych układów.

Dla realizacji odczytu niezbędne jest również wymuszenie poziomu niskiego na wejściu OE/ (ang. Output Enable). Wejście to aktywizuje wyjściowe bufory danych. Jak wynika z wykresu czasowego (rys. 2), dane mogą być odczytywane po ok. 450 ns (wersja standardowa) do ustalenia się informacji na wejściach adresowych. Wykorzystywanie trybu uśpienia w czasie, gdy mikroprocesor nie realizuje odczytu z układu, nie wnosi więc dodatkowych opóźnień czasowych. Wejścia CE/-PGM i OE/ mogą być sterowane tym samym sygnałem. W praktyce na wejście CE/-PGM podawany jest sygnał z dekodera adresów, natomiast wejście OE/ sterowane jest strobem odczytu z pa-



Rys. 2. Przebiegi czasowe przy realizacji cyklu odczytu pamięci

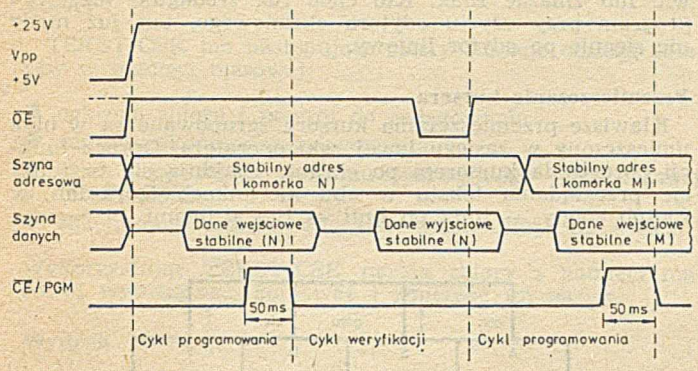
mięci. Ponieważ dla adresów zajmowanych przez pamięć EPROM nie są wykonywane cykle zapisu, nie powoduje to niepotrzebnego pobudzania układu. „Fałszywe" pobudzenie może jednak wystąpić przy zbieżności adresu generowanego przy operacji we-wy z adresem zajmowanym przez EPROM.

Należy zwrócić uwagę, że prawidłowa praca układu wymaga napięcia +5V na wejściu Vpp (nóżka 21). Podanie na to wejście stanu logicznego „1" zazwyczaj okazuje się niewystarczające. Stanowi to pewną niedogodność przy projektowaniu konstrukcji, w których w miejsce układu 2716 można by ewentualnie stosować 2732 (na nóżkę 21 podawany jest wtedy A11).

Przy realizacji odczytu, na nóżce 21 może występować napięcie wyższe od +5V (do +25V) jednak na wejściu CE/-PGM musi być wtedy (w sposób ciągły) wymuszony poziom niski.

Programowanie układu wymaga podania na wejście Vpp napięcia +25V. Każda z komórek układu może być programowana niezależnie od pozostałych (komórka zawiera 8 bitów). W tym celu należy na szynie adresowej wymusić adres żądanej komórki, a na szynie danych informację, która ma być wpisana. Aby dana została zapamiętana na wejściu CE/-PGM należy przez ok. 50 ms wymusić poziom wysoki. Weryfikację zapisu można zrealizować wymuszając poziom niski na wejściu OE/ (rys. 3). W tym czasie

na wejściu Vpp może być utrzymane napięcie +25V, pod warunkiem jednak, że na CE/-PGM będzie stan logiczny „0".

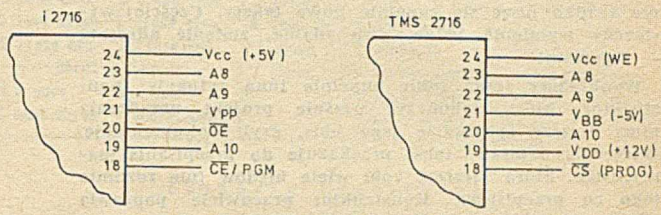


Rys. 3. Przebiegi czasowe przy programowaniu i weryfikacji zawartości pamięci

Układ 2716 po wymazaniu zawartości promieniowaniem ultrafioletowym (lub fabrycznie nowy) ma wszystkie bity ustawione w stan logiczny „1". Programowanie polega na wymuszeniu przejścia żądanych bitów w stan logiczny „0". Jeżeli w trakcie programowania niektóre bity pozostawiono w stanie logicznym „1", to można je w dowolnym momencie „doprogramować" postępując analogicznie jak przy pierwszym programowaniu, z tym, że oczywiście informacja wejściowa musi mieć „0" na odpowiednim bicie. Stanu logicznego bitów, które zostały zaprogramowane jako „0" nie można już zmienić w inny sposób, niż wymazanie zawartości całego układu promieniowaniem ultrafioletowym.

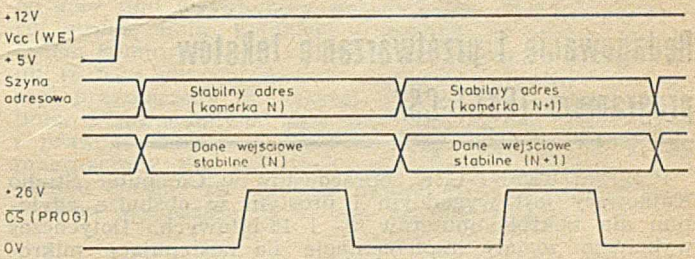
Programowanie i wymazywanie zawartości układów EPROM może być wykonywane wielokrotnie, jednak po każdej takiej operacji pozostaje ślad w postaci pewnej ilości rozproszonego ładunku. Zbyt duża ilość ładunku rozproszonego zmniejsza skuteczność programowania i w efekcie końcowym powoduje „zużycie" układu. Standardowa procedura programowania przewiduje, że impuls na wejściu CE/-PGM trwa ok. 50 ms. Powoduje to, że czas programowania całego układu zbliża się do dwóch minut. Przy układach o większej pojemności czas ten wydłuża się proporcjonalnie sięgając ok. pół godziny dla 27256. Skłoniło to producenta do poszukiwania algorytmów o krótszym czasie programowania. Algorytm taki opisany zostanie w jednym z następnych wydań mikroKLANU.

Układ EPROM 2716 produkowany jest obecnie przez wiele różnych firm i z reguły w oznaczeniach zawarte są cyfry 2716. Warto jednak zwrócić uwagę, że firma TEXAS INSTRUMENTS użyła oznaczenia TMS 2716 dla pamięci EPROM, która zasadniczo różni się od układu firmy INTEL. Podstawową różnicą jest konieczność stosowania trzech napięć zasilających dla układu TMS 2716 (rys. 4).



Rys. 4. Różnice w topografii wyprowadzeń układów i2716 (firma INTEL) oraz TMS 2716 (firma TEXAS INSTRUMENTS)

Całkowicie odmienny jest też algorytm programowania. Poszczególne komórki muszą być programowane w sposób sekwencyjny impulsami o czasie trwania od 0,1 do 1 ms. Po przejściu wszystkich komórek procedura jest powtarzana aż do momentu, gdy sumaryczny czas trwania impulsów programujących będzie dla każdej komórki dłuższy od 100 ms (rys. 5). Algorytm programowania układu TMS 2716 jest więc bardzo zbliżony do algorytmu wykorzystywanego przy programowaniu układów 2708 firmy INTEL. Zbliżona topografia wyprowadzeń układów 2708 i TMS 2716 stwarza możliwość wykorzystania tego drugiego w konstrukcjach, gdzie planowane było wykorzystywanie pamięci EPROM 2708. Należy w tym miejscu podkreślić, że włożenie standardowego układu 2716 w miejsce 2708 prowadzi do jego uszkodzenia.



Rys. 5. Przebiegi czasowe przy programowaniu układu TMS 2716

Firma TEXAS INSTRUMENTS produkuje też układ pamięci EPROM zbliżony do INTELowskiego 2716. Oznaczany jest on jednak TMS 2516. W mikroKLANIE 12 podano różnicę w topografii wyprowadzeń układów 2716 i TMS 2516.

AJP

Po sukcesie komputera CPC 464 firma AMSTRAD wprowadziła nowy model CPC 664. Zamieniono w nim magnetofon kasetowy na napęd dyskietek 3". Wkrótce jednak rozeszły się wieści o nowszym modelu z rozszerzoną pamięcią RAM. Zmiana ta poważnie zwiększyła wartość użytkową komputera: bardziej popularne programy, pracujące pod nadzorem systemu CP/M, wymagają nieco większego obszaru pamięci operacyjnej niż mogą to zapewnić modele 464 i 664. Powoduje to konieczność podziału programu na segmenty sprowadzane — stosownie do sytuacji — z dysku. W rezultacie program pracuje znacznie wolniej.

Aby nie hamować sprzedaży modelu 664 ogłoszono, że nowa wersja będzie początkowo oferowana tylko na rynek amerykański. Mimo tych zapewnień, w sierpniu na rynek brytyjski trafił CPC 6128 i to w cenie modelu 664. Stało się więc jasne, że model CPC 664 zostanie wycofany z produkcji. Wywołało to wiele protestów ze strony nabywców (producenci oprogramowania przerwali prace nad adaptacjami). Narzekali też sprzedawcy, którym w magazynach zostały modele 664. Jednak duży popyt na CPC 6128 szybko wyrównał im straty.

Pod koniec września 1985 r. AMSTRAD znowu wszystkich zaskoczył: wprowadzono kolejny nowy model — PCW JOYCE 8256. O ile poprzednie wersje uwzględniały fakt, że większość komputerów domowych kupuje się dla zabawy, to z charakterystyki PCW 8256 wynikało, że jest on przeznaczony dla tych, którzy traktują sprawę poważnie. Zielony monitor (choć z większą rozdzielczością) studzi niewątpliwie zapędy fanatyków gier. Dysk o większej pojemności, pamięć RAM 256 KB i drukarka w ramach konfiguracji podstawowej to sprzęt godny zastosowań profesjonalnych. Swoją propozycję AMSTRAD opiera na przeświadczeniu, że odbiorcy wyrośli już z gier i szukają komputera-pomocnika. Przetwarzanie tekstów i baza danych to aplikacja przydatna dla każdego.

Wygląda na to, że AMSTRAD zapoczątkuje rewolucję podobną do tej, jaką kiedyś zrobił ZX 80, tyle że o innym ciężarze gatunkowym. Zabawki wezmą się do roboty.

Redagowanie tekstów nie jest wyłącznie domeną redaktorów. Większa część tzw. prac biurowych sprowadza się do przetwarzania tekstów. Nie chodzi tu tylko o ułatwienie poprawiania błędów, ale często też o wykorzystywanie całych fragmentów napisanych już przy innej okazji. Biurowa praktyka wskazuje, że niezmiernie rzadko pisze się zupełnie nowe teksty. Częściej wystarczy wymienić jedno, dwa zdania, zmienić adresata — i gotowe!

Wyobraźmy teraz sobie zupełnie inną sytuację: konstruktor, który zakończył właśnie projekt urządzenia musi jeszcze sporządzić jego opis, czyli dokumentację. Odręcznie napisany tekst przekazuje do przepisania maszynistce, która niestety robi wiele błędów (nie rozumie tego co przepisuje). Konstruktor pracowicie poprawia błędy maszynopisu i przekazuje tekst do ponownego przepisania. Gdy tekst wraca, okazuje się, że choć stare błędy zostały poprawione to pojawiły się nowe. „Zabawa" może powtarzać się wielokrotnie. Gdyby maszynistka wprowadziła tekst do komputera, to bez większych kłopotów konstruktor poprawiłby ewentualne błędy (nie ma potrzeby powtórnego przepisywania) i materiał byłby gotów do rozpowszechniania.

Niedawno opisywaliśmy procesor tekstów dla ZX SPECTRUM. W poniższym tekście prezentujemy program nadający się już do zastosowań profesjonalnych — na profesjonalnych komputerach. Podstawową wadą programu jest brak możliwości stosowania typowo polskich znaków (np. ą, ę, ł itp.). Taka jest niestety cena przenoszenia programu między różnymi komputerami. Oczekujemy jednak, że w następnej generacji programu CSK udoskonali go przynajmniej dla kilku typów mikrokomputerów lub może nawet przygotuje specjalny moduł adaptacyjny, pozwalający użytkownikowi przystosować program do specyfiki posiadanego komputera i drukarki. Czekamy niecierpliwie!

Redagowanie i przetwarzanie tekstów programem TEKST-CSK

Program TEKST-CSK, opracowany w Computer Studio Kajkowscy jest wygodnym i prostym w obsłudze edytorem dla mikrokomputerów 8- i 16-bitowych. Dotychczas wykonane zostały implementacje na następujące mikrokomputery: APPLE II, LIDIA, ELWRO-513/523, ROBOTRON-5120/5130, IMP-85, MK-4, RTDS-8 i MC oraz IBM PC, Olivetti M24, LIDIA 2, EMIX, IMP-86 i inne kompatybilne z APPLE II lub IBM PC. Oprogramowanie może działać na sprzęcie spełniającym następujące wymagania:

— mikroprocesor Z80, 8080, 8085, 8086 lub 8088
— minimum 48 KB pamięci RAM
— pamięć masowa na dyskach elastycznych
— alfanumeryczny monitor ekranowy o szerokości 64 lub 80 znaków w linii
— drukarka.

TEKST-CSK jest programem bardzo przydatnym do prac biurowych takich, jak: prowadzenie korespondencji, przygotowywanie pism wewnątrz zakładowych, tworzenie i przechowywanie sprawozdań oraz innych dokumentów, redagowanie ulotek reklamowych itp. Jako przykład zastosowania TEKST-CSK można podać niniejszy artykuł zredagowany i wydrukowany [1]) (w maszynopisie) przez autora przy użyciu opisywanego programu na mikrokomputerze IBM PC.

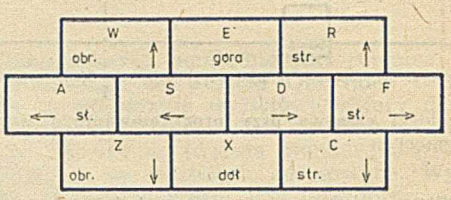
TEKST-CSK służy również do redagowania programów źródłowych w dowolnym języku programowania. W przypadku, gdy mikrokomputer stanowi inteligentną końcówkę dużego komputera, ekranowy program redakcyjny staje się nieodzowny.

[1]) Niestety INFORMATYKA nie dysponuje tzw. składem komputerowym i artykuł został przygotowany do druku tradycyjnymi metodami znanymi z czasów Gutenberga (przypis AJP)

Pojęcie „ekranowy" oznacza, że czynność redagowania (dopisywanie, wymazywanie, poprawianie) odbywa się w dowolnym miejscu ekranu, na którym wyświetlony jest wybrany fragment tekstu. Zatem ekran jest jak kartka papieru, na której można w każdym miejscu dopisać, poprawić lub zmazać znak. Kto choć raz zredaguje tekst lub program przy użyciu edytora ekranowego, ten już nigdy nie sięgnie po edytor liniowy.

Przemieszczanie kursora

Klawisze przemieszczania kursora zgrupowane są w blok umieszczony w zasięgu lewej ręki operatora. Oprócz funkcji poruszania kursorem po ekranie znajdują się tu funkcje przeglądania tekstu w obu kierunkach (góra-dół), w sposób ciągły — linia po linii — lub stronami.



Program TEKST-CSK ma 10 rejestrów, w których można zapamiętać dowolne adresy odpowiadające różnym pozycjom kursora. Można je wykorzystać w dowolnym momencie. W każdej chwili można także wykonać skok do jednego z pięciu ważnych adresów: na początek lub na koniec zbioru, na początek lub na koniec bloku i do poprzedniej pozycji kursora (powrót).

Redagowanie

Pozycja kursora jest istotna z tego względu, że wszelkie czynności korekcyjne odbywają się właśnie w tym miejscu. Naciskając odpowiedni klawisz można:

— zmazać: znak z lewej lub z prawej strony, słowo, część linii od kursora do końca, część linii od początku do kursora, całą linię
— dopisać: znak, słowo, linię
— poprawić dowolny znak w tekście.

Przewidziano również funkcję automatycznego wyszukiwania i wymiany dowolnej frazy na inną.

Formatowanie

Ekran dla TEKST-CSK ma szerokość 80 lub 64 znaków. Teksty redagowane przy użyciu programu mogą mieć dowolną szerokość. Deklaracja szerokości tekstu oraz jego rozmieszczenie na monitorze odbywa się poprzez ustawienie lewego i prawego marginesu. Uwzględnianie marginesów polega na przeniesieniu wyrazu nie mieszczącego się przed prawym marginesem i przesunięciu pozostałych wyrazów tak, by pole pomiędzy marginesami było równomiernie wykorzystane. Do formatowania według marginesów należy zaliczyć także funkcję centrowania, która ustawia tekst w środku pola między marginesami (istotne przy tytułach).

Użytkownik programu TEKST-CSK ma do dyspozycji różne typy tabulatorów. Tabulator stały (o długości skoku — 8) dostępny jest podczas redagowania zarówno tekstów jak i programów źródłowych. Tabulatory zmienne (ustawiane przez użytkownika) można stosować jedynie przy redagowaniu tekstów. Rozróżniane są dwa rodzaje tabulatorów zmiennych: dla słów oraz dla liczb.

Bloki

Blok jest to część zbioru wyznaczona dwoma adresami (pozycja kursora) — początkowym i końcowym. Adres początkowy musi oczywiście poprzedzać adres końcowy. Zaznaczony wcześniej blok można wymazać, przenieść w in-

ne miejsce zbioru, przekopiować w miejsce ustawienia kursora, zapisać w oddzielnym zbiorze pamięci masowej lub odczytać ze zbioru na dysku.

Zbiory

TEKST-CSK ma następujące cztery funkcje obsługi zbiorów w pamięci masowej:

- ochrona zbioru i powrót do redagowania
- zapis i zakończenie redagowania
- zapis i powrót do systemu operacyjnego (koniec pracy z TEKST-CSK)
- zakończenie redagowania bez zapisu zbioru na dysku.

Wykorzystując TEKST-CSK można zbiory z pamięci masowej wymazywać, kopiować i zmieniać im nazwy.

Wydruk

Końcowym efektem redagowania jest sporządzenie trwałej kopii tekstu na drukarce. W trakcie wydruku realizowane są dyrektywy (linie tekstu, które zaczynają się od kropki). Formatowanie wydruku zależne jest od typu przyłączonej do mikrokomputera drukarki i polega na dzieleniu tekstu na strony z uwzględnieniem: długości stron, marginesów, numeracji, komentarzy nad i pod tekstem, szerokości i wysokości liter, odstępu między liniami, koloru taśmy, zestawu znaków drukarskich itp.

Język przetwarzania tekstowego

Program TEKST-CSK umożliwia również wykonywanie bardziej skomplikowanych czynności przetwarzania tekstowego. W przypadku, gdy zachodzi potrzeba wielokrotnego wydruku tego samego zbioru, podobnych tekstów, łączenia zbiorów w trakcie drukowania lub skorzystania ze zgromadzonych danych — można zastosować język przetwarzania tekstowego. W ten sposób zautomatyzowano edycję pism o podobnej treści i formie, a różniących się tylko fragmentami.

* * *

W artykule opisano tylko najważniejsze z listy ok. 150 funkcji redakcyjnych. Ich wywołanie odbywa się poprzez naciśnięcie jednego lub dwóch klawiszy. Funkcje podzielone są na sześć grup według przeznaczenia. Dla każdej grupy na ekranie wyświetlane jest tzw. menu, czyli lista funkcji z opisem. Po nabraniu wprawy w operowaniu programem, można zrezygnować z wyświetlania komentarzy pomocniczych. Na rysunku 2 przedstawiono kopię obrazu monitora w trakcie działania TEKST-CSK. W pierwszej od góry linii wyświetlany jest aktualny stan (nazwa zbioru, strona, linia, kolumna) i komunikaty dla operatora. Osiem kolejnych linii, to obszar dla jednego z sześciu menu (z tego obszaru można zrezygnować, rozszerzając obszar przeznaczony dla tekstu). Pod menu narysowany jest linial z zaznaczonymi pozycjami tabulatorów oraz lewego i prawego marignesu. Pod linialem znajduje się obszar przeznaczony dla redagowanego tekstu. Na samym dole ekranu opisane są klawisze funkcji specjalnych dla komputerów wyposażonych w tego rodzaju funkcje.

```
A:APEL.ART  STR. 1 LIN. 1 KOL 01          DOPIS. WL
<<<  GLOWNE MENU  >>>
------ Przemieszczenia ------¦- Kasowanie -¦---- Pomocnicze ---¦--- Inne menu --
^S lewy znak  ^D prawy znak  ¦^G prawy zn.¦^I tab    ^B format ¦^J dla objasnien
^A lewe slowo ^F prawe slowo ¦^L lewy zn.¦^V dopisywanie      ¦^K dla blokow
^E linia nad  ^X linia pod   ¦^T slowo   ¦^L szukanie         ¦^U dla skokow
------- Obrot ekranu --------¦^Y linii   ¦^R koniec akapitu   ¦^P dla wydruku
^Z w gore     ^W w dol       ¦           ¦^N posta linia      ¦^O dla ekranu
^C nast.obraz ^R poprz.obraz ¦           ¦^U zatrzymanie      ¦
L----¦----¦----¦----¦----¦----¦----¦----¦----¦----¦-------R

                Szanowny Czytelniku

            Autor artykulu prosi o kierowanie

            swoich uwag i pytan na adres :

                    Antoni Urban

                ul. Balladyny 3b

                81-524 Gdynia - Orlowo
10BJAS. 2AKAPIT 3L.MARG 4P.MARG 5PODKR. 6CIEN.  7POC.BL 8KON.BL 9POC.ZB 10KON.ZB
```

ANTONI URBAN
CSK-Gdynia

Realizując proste systemy mikroprocesorowe przeznaczone do celów sterowania stajemy często przed koniecznością dołączania wielu układów, które wykorzystywane są w ograniczonym zakresie. Przykładowo, potrzeby w zakresie pamięci typu RAM sprowadzają się zazwyczaj do kilkudziesięciu bajtów. Układy z rodziny MCS 85 projektowane były często właśnie z myślą o tego typu zastosowaniach.

Połączenie kilku funkcji w jednym układzie (np. pamięć RAM, port we-wy, zegar) nie tylko upraszcza konstrukcję, ale też przyczynia się do zwiększenia niezawodności. Mniejszy pobór prądu zasilającego może uprościć konstrukcję zasilacza, a zaoszczędzone miejsce — spowodować zmniejszenie gabarytów urządzenia.

W prezentowanym rozwiązaniu Jerzy Orkiszewski proponuje umieszczenie układu w przestrzeni adresowej we-wy. Zastępując linie I-OR/ i I-OW/ przez odpowiednio MEMR/ i MEMW/, i ewentualnie rozbudowując dekoder, możemy umieścić układ w przestrzeni adresowej pamięci, co pozwoli wykorzystywać w programach znacznie szerszy zestaw rozkazów. Jednak nawet taka zmiana nie pozwoli wyeliminować stosunkowo poważnej wady rozwiązania: zawarta w układzie 8155 pamięć RAM nie może być wykorzystana jako stos. Jeżeli więc chcemy korzystać z przerwań lub podprogramów musimy niestety dołączyć dodatkową pamięć RAM.

Układy z multipleksowaną szyną danych w systemie 8080

Magistrale systemowe mikroprocesorów podzielić można na dwie grupy. Pierwsza obejmie rozdzielone szyny adresów i danych czyli takie, w których dane i adresy przesyłane są po osobnych liniach. Przedstawicielami tej grupy są popularne mikroprocesory, takie jak: 8080, Z80, 6800 czy 16-bitowy 68000. W drugiej grupie dla szyny adresowej i danych wykorzystywane są te same linie. Szyna danych służy tu do przesyłania części adresu a następnie słowa danych. Odpowiednie impulsy strobujące pozwalają rozróżnić typ aktualnie podawanej informacji. Tego typu szyny określane są mianem „multipleksowanych". Multipleksowane szyny występują m.in. w 8035, układach z serii 6805, 8086 i większości mikroprocesorów zrealizowanych w technologii CMOS.

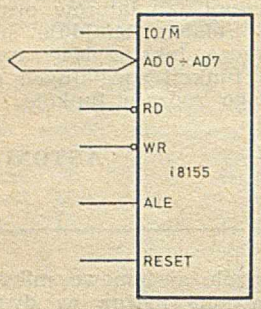
Każdej z tych grup towarzyszy rodzina układów peryferyjnych, zazwyczaj przystosowanych do multipleksowanej szyny danych i adresów. Często są to układy zrealizowane jako

CMOS i wykazują znane zalety tej technologii, Szczególnie interesujące są układy peryferyjne CMOS z multipleksowaną szyną danych, oferowane przez firmę MOTOROLA. Są one zaopatrzone w specjalny interfejs nazwany MOTEL (od nazw MOTorola i IntEL). Pozwala to na prostą realizację współpracy z szyną mikroprocesora typu 8085 jak i 6805. To, który z mikroprocesorów steruje układem rozpoznawane jest przez interfejs automatycznie, na podstawie zależności czasowych na szynie.

Dołączenie „zwykłego" układu peryferyjnego do mikroprocesora z multipleksowaną szyną danych nie nastręcza zwykle większych problemów. Wystarczy rozdzielić obie szyny za pomocą rejestru zatrzaskowego. Znacznie więcej problemów sprawia sytuacja odwrotna, np. połączenie mikroprocesora typu 8080 z układem multipleksowanym z rodziny 8085.

Opisany dalej przykład rozwiązania tego problemu dotyczy połączenia procesora 8080 z układem 8155. Podana metoda pozwala łączyć w zasadzie dowolny, „zwykły" procesor z każdym układem we-wy lub pamięci z multipleksowaną szyną danych i adresów.

Układ 8155 jest elementem z rodziny MCS-85. Kostka zawiera 256 bajtów pamięci RAM, dwa 8-bitowe i jeden 6-bitowy port we-wy oraz licznik. Zestaw linii łączących układ z systemem przedstawia rysunek 1.



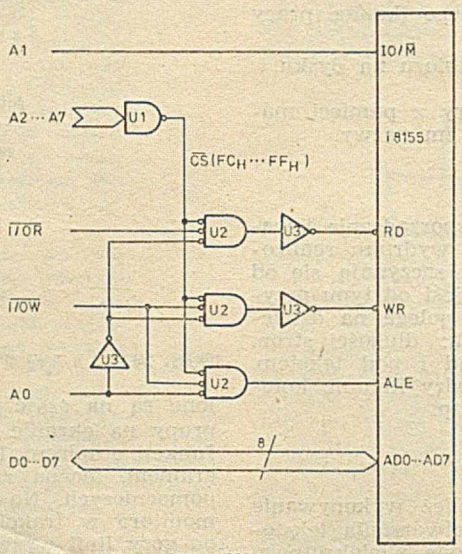
Rys. 1. Linie interfejsu systemowego dla układu 8155

Adres przesyłany w pierwszej części cyklu dostępu zatrzaskiwany jest sygnałem ALE. W drugiej części cyklu, dana przesyłana jest do lub z układu. O kierunku przesłania decydują stany logiczne na wejściach RD/ i WR/. O tym, czy adres zatrzaśnięty sygnałem ALE dotyczył wewnętrznej pamięci RAM czy też wybierał któryś z rejestrów we-wy albo licznika decyduje stan podany na linię IO-M/.

Schemat dołączenia układu 8155 do szyny mikroprocesora 8080 przedstawia rysunek 2. Mikroprocesor „widzi" układ jako zestaw czterech komórek przestrzeni adresowej we-wy. Dwie spośród tych komórek to zatrzaski adresowe. Wpis do tych rejestrów odbywa się poprzez wygenerowanie przez układy logiki adaptacyjnej impulsu ALE. Aby sygnał taki powstał muszą być spełnione następujące warunki:

a) procesor adresuje układ 8155, czyli aktywny jest dekoder, do którego dołączone są linie A2...A7 oraz aktywny jest sygnał IOW/

b) na linii A0 wymuszony jest stan logiczny 0.



U1: TTL 7430 CMOS 4068
U2: TTL 7427 CMOS 4025
U3: TTL 7404 CMOS 4069

Rys. 2. Przykładowy sposób podłączenia układu 8155 do szyny systemowej mikroprocesora 8080

Stan linii A1 decyduje czy wysłany bajt danych (będący adresem wewnętrznym dla układu 8155) stanowi odwołanie do jego części pamięciowej czy też we-wy.

Pozostałe dwie komórki to pseudorejestry danych. Wpis bądź odczyt danych z tych komórek odbywa się poprzez wygenerowanie odpowiednich impulsów na wejściach WR/ lub RD/ układu 8155. Sposób generowania tych sygnałów jest podobny jak dla ALE, z tym, że na linię A0 musi zostać podany poziom wysoki. W przykładzie przedstawionym na rysunku 2, poszczególnym komórkom przyporządkowane są następujące adresy:

— zatrzask adresowy dla pamięci FC_H (A7...A2=1, A1=0, A0=0)

— pseudorejestr danych pamięci FD_H (A7...A2=1, A1=0, A0=1)

— zatrzask adresowy dla we-wy FE_H (A7...A2=1, A1=1, A0=0)

— pseudorejestr danych we-wy FF_H (A7...A2=1, A1=1, A0=1)

Współpraca z układem odbywa się według następującego schematu:

a) wysłanie adresu do zatrzasku adresowego

b) odczyt lub zapis danej z (lub) do odpowiedniego pseudorejestru.

Przykładowa procedura odczytu danej z wewnętrznej pamięci układu 8155 (spod adresu 38H) ma następującą postać:

```
MVI  A, 38H  ; adres w akumulatorze
OUT  FCH     ; zapis do zatrzasku
               adresowego
IN   FDH     ; dana do akumulatora
```

Zapisując daną do pamięci możemy wykorzystać program:

```
MVI  A,38H      ; adres w akumulatorze
OUT  FCH        ; zapis do zatrzasku
                  adresowego
MVI  A, DANA    ; dana do akumulatora
OUT  FDH        ; zapis do pseudorejestru
```

Opisana metoda polega na dołączeniu prostego układu kombinacyjnego i nieznacznej rozbudowie programu. W zamian otrzymujemy możliwość wykorzystania całej gamy nowych układów, zazwyczaj wyjątkowo uniwersalnych, co szczególnie w małych systemach jest nie bez znaczenia.

JERZY ORKISZEWSKI
Warszawa

- Wygląda na to, że obietnice firmy ATARI, dotyczące nowej rodziny mikrokomputerów ST, zostały dotrzymane. Model 260 ST „Little Saint" zawiera wbudowany napęd dysków 3,5" i razem z myszą (ale bez monitora) ma kosztować 500 funtów. Na wystawie promocyjnej pokazywano też wersję z 1 MB RAM (!), zapowiadając, że będzie też i wersja z 4 MB RAM. Dla serii ST oferowanych jest już obecnie 80 pakietów programowych. Równocześnie drastycznie obniżono ceny na starsze modele. 800XL z pamięcią 64 KB, drążkiem sterowniczym i modułem pamięci z grami kosztuje tylko 70 funtów. 130XE (128 KB RAM) kosztuje 140, a razem z napędem dysków — 265 funtów. Za 350 funtów można nabyć komplet z drukarką i oprogramowaniem redaktora tekstu.
- Firma SINCLAIR uchroniła się przed bankructwem i wykupieniem przez Roberta Maxwella. Projektowana jest nowa wersja ZX SPECTRUM+ z nowym rozwiązaniem ROM zawierającym ulepszony BASIC, z 128 KB RAM, generatorem dźwięków AY 33910 i o większych możliwościach graficznych. Cenę QL obniżono do 200 funtów.

Ogłoszenia • Ogłoszenia

Oprogramowanie do COMMODORE C-64 piszemy na zlecenie instytucji, osób prywatnych. 27-400 Ostrowiec, skrytka 40
EO/1103/K/85

W poprzednim wydaniu mikroKLANU opublikowaliśmy schemat ideowy niewielkiego systemu zbudowanego na bazie mikroprocesora MOTOROLA 6800. Ten popularny na świecie mikroprocesor jest w naszym kraju stosunkowo mało znany. W tej sytuacji pierwsze kroki z systemem mogą być stosunkowo trudne. Ponieważ mikrokomputer tak naprawdę nabiera życia dopiero z chwilą, gdy jest do niego jakieś oprogramowanie, publikujemy teraz program, który na początku przygód z 6800 może okazać się bardzo przydatny.

Program MONITOR dla mikroprocesora MOTOROLA 6800

Program MONITOR [1]) zajmuje w pamięci system obszar o długości 1KB (od $FC00_H$ do $FFFF_H$). Umożliwia to pobranie przez system wektorów restartu i przerwań bezpośrednio z programu. Zmienne systemowe, tablice adresów i stan rejestrów procesora pamiętane są w obszarze roboczym pamięci RAM (minimum 128 bajtów). Powinien być on umieszczony od adresu $E000_H$. Procedury we-wy obsługują układ **ACIA** — MC6850 umieszczony pod adresami $8008/8009_H$. Program nie wykorzystuje przerwań sprzętowych; są one w pełni dostępne dla użytkownika. Jedyne ograniczenia dotyczą użycia przerwania programowego **SWI**, które wykorzystywane jest przy pracy krokowej. Istnieje jednak możliwość zmiany adresu obsługi tego przerwania, gdyż jest on każdorazowo pobierany z pamięci RAM.

Opis działania programu

Po inicjacji, procesor rozpoczyna działanie od wpisania do licznika rozkazów zawartości komórek o adresach $FFFE/FFFF_H$. Znajduje się w nich adres tzw. „zimnego" startu programu — $FC20_H$. Na początku program przygotowuje w pamięci RAM „notatnik". Przepisywana jest do niego tabelka kodów modyfikowanej części programu, a pozostała część jest zerowana. Następnie programowany jest układ 6850. Słowo programujące (19_H przechowywane pod adresem $FC3C_H$) powoduje: wewnętrzny podział TxC i RxC przez 16; długość transmitowanego słowa 8 bitów + bit stopu + bit parzystości; wyjście RTS/ ustawiane w stan 0; przerwania z nadajnika i odbiornika są zablokowane. Słowo programujące po „zimnym" starcie pamiętane jest w pamięci RAM pod adresem $E019_H$. W tzw. „gorącym" wejściu programu (adres $FC40_H$) wpisywane jest ono do rejestru sterującego ACIA. Wskaźnik stosu ustawiany jest na adres $E078_H$.

Następnie program wysyła do terminala zgłoszenie gotowości (znak: *) i przechodzi do oczekiwania na dyrektywę. Po jej odebraniu przegląda tablicę adresów i kodów dyrektyw. Tablica ta (zajmująca obszar: $FFD1_H$-$FFF7_H$) zorganizowana jest następująco: bajt kodu ASCII dyrektywy + dwa bajty adresu procedury obsługi. Gdy po przejrzeniu całej tablicy dyrektywa nie zostanie rozpoznana, program wraca do pętli oczekiwania.

System dopuszcza wprowadzenie ośmiu różnych adresów pułapek. Dojście wykonywanego programu do adresu pułapki powoduje wykonanie podprogramu obsługi przerwania programowego. Podprogram ten drukuje zawartość rejestrów procesora w chwili napotkania pułapki. Po przerwaniu znajdują się one na stosie. Wydruk ma przykładową postać:

```
E9 FC DB E200 E103 E078
```

Kolejno wypisana jest: zawartość rejestru warunków **CC**, akumulatora **ACCA**, akumulatora **ACCB**, rejestru indeksowego **X**, licznika programu **PC** i wskaźnika stosu **SP**. Wartość SP = = $E078_H$ oznacza, że adresy pamięci RAM, w których pamiętana jest zawartość rejestrów są następujące: CC — $E079_H$; ACCA — $E07A_H$; ACCA — $E07B_H$; X — $E07C/E07D_H$; PC — $E07E/E07F_H$.

Przed wykonaniem dyrektywy C, N lub T zawartość stosu (a więc także rejestrów procesora w programie użytkownika) może być zmieniona dyrektywą M.

Każda próba zapisu do pamięci ROM lub pamięci nie istniejącej fizycznie sygnalizowana jest wydrukiem „?" i zakończeniem dyrektywy.

Dyrektywy programu MONITOR

Na każdą dyrektywę składa się jej kod (jedna litera i ewentualnie jeden lub dwa argumenty).

B — drukowanie adresów wszystkich zdefiniowanych pułapek programowych. W czasie wykonywania programu kod znajdujący się pod takim adresem zastępowany jest kodem instrukcji SWI.

D — skasowanie wszystkich pułapek.

C — wznowienie wykonywania programu po zatrzymaniu w pułapce. Kod instrukcji SWI zostaje zastąpiony oryginalnym kodem programu.

G xxxx — wykonanie programu od adresu xxxx.

M xxxx — operacja na zawartości komórki pamięci o adresie xxxx. Po przyjęciu dyrektywy, system drukuje zawartość podanej komórki w postaci szesnastkowej. Operator może teraz zażądać wydruku zawartości komórki o adresie xxxx+1, wprowadzając znak **LF**. Dla dokonania zmiany zawartości pamięci należy wpisać dwa znaki szesnastkowe i LF. Aby odczytać zawartość komórki o adresie xxxx—1, należy wprowadzić zamiast LF znak ↑ (ASCII:$5E_H$). Operacje na pamięci kończy wprowadzenie znaku różnego od: **LF**, ↑ i 0...9 i A...F. Może to być np. spacja lub kropka.

```
*M EC00 CF
EC01 9E 86
EC02 1D ^
EC01 86
EC02 1D 78
EC03 F7 .
```

Wydruk 1. Przykład operacji na zawartości pamięci

N — wykonanie jednego rozkazu po zatrzymaniu programu na pułapce. Drukowana jest aktualna zawartość wszystkich rejestrów procesora.

P xxxx yyyy — wydruk zawartości pamięci od adresu xxxx do adresu yyyy, w formacie S1-S9.

R — wydruk zawartości rejestrów.

T xxxx — podobnie jak dyrektywa C, z tym, że wykonaniu podanej szesnastkowo liczby rozkazów towarzyszy wydruk aktualnej zawartości rejestrów.

U xxxx — skasowanie pułapki pod adresem xxxx.

V xxxx — ustawienie pułapki pod adresem xxxx.

```
*D
*B
*V E000
*B E000
*V E050
*V E060
*B E000 E050 E060
*U E050
*B E000 E060
*D
*B
```

Wydruk 2. Ustawianie i kasowanie pułapek

```
*V E407
*B E407
*
*
*G
*G E400
C8 81 01 F000 E407 E078
*N
C0 81 02 F000 E408 E078
*N
C8 82 02 F000 E409 E078
*T 000A
C8 82 02 F001 E40A E078
C8 82 02 F001 E407 E078
C0 82 03 F001 E408 E078
C8 83 03 F001 E409 E078
C8 83 03 F002 E40A E078
C8 83 03 F002 E407 E078
C0 83 04 F002 E408 E078
C8 84 04 F002 E409 E078
C8 84 04 F003 E40A E078
C8 84 04 F003 E407 E078
*
```

Wydruk 3. Przykład użycia pułapki przy uruchamianiu programu

[1]) Program stanowi modyfikację firmowego monitora MINIBUG3E.

S xxxx yyyy — podobnie jak dyrektywa P, z tym, że wewnętrzny dzielnik ACIA ustawiany jest na podział częstotliwości TxC i RxC przez 64, wyjście RTS/ w stan „1", a przed wysłaniem żądanego bloku danych wysyłany jest ciąg 256 bajtów zerowych. Dyrektywa może być użyta do wyprowadzania zawartości pamięci na magnetofon lub perforator, przy czym szybkość transmisji wynosi 300 bodów, a sygnał RTS/ = „1" wskazuje, że dane nie są przeznaczone dla terminala.

```
*
*
*P FC00 FC0F
S113FC00FEE0006E00FEE0066E00FEE0346E00537F89
*
*
*
*
```

Wydruk 4. Przykładowa postać wydruku rekordu

L — ładowanie do pamięci blok zapisany komendą S. Wewnętrzny dzielnik i poziom sygnału RTS/ — jak wyżej.

Obsługa przerwań

Po przyjęciu przerwania adres odpowiedniej procedury obsługi pobierany jest sprzętowo z odpowiednich komórek w obszarze $FFF8_H$-$FFFD_H$. Aby jednak umożliwić obsługę przerwań, zgodnie z życzeniem użytkownika, procedury te oddają sterowanie pod adres pobrany z bufora w pamięci RAM. Jest on umieszczany w następujący sposób:

- adres obsługi IRQ/ : $E000/F001_H$
- adres obsługi NMI/ : $E006/E007_H$
- adres obsługi SWI/ : $E036/E037_H$

Adres dla SWI/ jest wstępnie wstawiany w trakcie inicjacji systemu tak, aby zawierał adres procedury obsługi w programie MONITOR.

Wydruk programu

Przedstawiony wydruk kodu maszynowego programu został wyprowadzony na drukarkę dyrektywą P. Wykonany został według formatu S1-S9 wprowadzonego przez firmę MOROTOLA. Blok danych dzielony jest na rekordy. Każdy rekord rozpoczyna się znakami początku S1. Następne dwa znaki określają liczbę bajtów w rekordzie (bez nich samych). Następne cztery znaki tworzą adres początku rekordu. Końcowe dwa znaki, to bajt sumy kontrolnej. Następne znaki (po dwa) oznaczają bajty danych. Koniec bloku oznaczany jest znakami S9. Cały blok jest zapisany w postaci ciągu znaków ASCII:0...9 i A...F oraz S. Przy wyprowadzaniu i wprowadzaniu dokonywana jest odpowiednia konwersja z (lub) na kod szesnastkowy. Przykładowy rekord przedstawiony jest na wydruku 4.

```
*
*
*P FC00 FFFF
S113FC00FEE0006E00FEE0066E00FEE0346E00537F
S113FC103904E078FED9FF0830037EFF467EFF4D8D
S113FC20BEE03FCEFC1FA60036098CFC1126F7C6D9
S113FC302A4F365A26FCB603D7B0088619B7E0197E
S113FC40BEE032B6E019B780087FE00CCEFD32DDCD
S113FC50FD18BDFD17BD7316BDFDB0CEFFD1E100EB
S113FC60270A0808088CFFF826FA20D4EE016E0059
S113FC708D5880302BCA81092F0A81112BC2B1161D
S113FC802EBE800739BDFD81FFE0028B4CFFE004EC
S113FC9039B6E01984FE8A42B780087CE00C8D7F77
S113FCA0B15326FA8D798139279A813124F07FE0B8
S113FCB00B8D348002B7E00E8D1F8D2B7AE00E2758
S113FCC00BA700A100260B0B20F020535C27CF864C
S113FCD007BD35863FBD312091BD0CB7E00F8D0750
S113FCE0B7E010FEE00F398D874B4B4B4B16BDFC40
S113FCF0701B16FBE00DF7E00D3944444444840FB7
S113FD008B30B13923028B0737F6B008575724F943
S113FD10B7800933398DF108A600B10426F739B676
S113FD208008724FAB68009B47F7DE00C27D939FE
S113FD3020E6130A0D1400000000042A04BD42FE7C
S113FD40E00F8D3A098DD8810A2718815E270EBDF0
S113FD50FC728D95A700A10027EB7EFCCF860A8D4F
S113FD60A709850BFFE00FCEFD34BDACCEE00F8DE2
S113FD700B20CCA60008368D8132208B28DF58DF3C0
S113FD80BC8D60B62020B1CEFD338D8CFEE0320880
S113FD908DEC8DEABDE88DE48DE2CEE03220D8D8D0
S113FDA0428609BC8628CB8611200B8D368609BD21
S113FDB0378605BD337EFC408D29FEE032A707B6D9
S113FDC0E00FA7067CE015CE00012009CE00018CCF
S113FDD0BD117FE015FFE00AFEE032EE06BDFF86DE
S113FDE07EFF337EFCD9868236CEE01AFFE01133E3
S113FDF037A602EE002703C4FB8CC4C47DE014279D
S113FE0003C4FD8CC4FE542402A70054240DA60090
S113FE1036863FA70032FEE011A702FEE01154240B
S113FE200EB6E00FA700B6E010A7013284FB3654EB
S113FE30240EA600B1E00F2607A601B1E0102703A7
S113FE405424046F006F01542404BDFD7C8C080B05
S113FE50088CE0322696F7E0143239BDFC85865AC8
S113FE60010101B7800BC6FF4FBDFD085A26F920DD
S113FE7003BDFCB5FEE002FFE011B6E005B0E01230
S113FE80F6E004F2E0112604B1102502860F8B04AB
S113FE90B7E00E8003B7E00DCEFED0BDFD185FCEF7
S113FEA0E00E8D27CEE0118D228D20FEE0118D1BFA
S113FEB07AE00D26F9FFE0115317BDFD7609BCE089
S113FEC00426B7CEFC0FBDFD182042EB007EFD7367
S113FED00D0A000000005331048FE0328D25306D5F
S113FEE00626026A056A06EE05BCE0082722A6007B
S113FEF0813F2614306C0626026C05BDFDE6FEE04B
S113FF00366E0086017EFDE8BDF9BDFD877EFC40DE
S113FF10B6E013A7007DE01527OD7FE015BDFDE6D3
S113FF207FE0087FE0093BBDFD87FEE00A09FFE0B2
S113FF300A27DAB6E013B7E038BD6E4D2A0A323458
S113FF408A10067EE038B602FEE00BE600C16E27CD
S113FF5044C17E2750C1AD273CC1BD2748C13B27C2
S113FF603BC139273CC13F2713C1BD260DFEE00B54
S113FF70A6010B08204094C26FC8D158D08863F25
S113FF80A700BEE0323BFFE00BA600B7E013390B43
S113FF904A4D26FB39A60130EE0320DE30EE0C205C
S113FFA0DB30EE07BCEE0120D3164444444ACEFFEC
S113FFB0C18DDEA6002609B602C40DC10C26014CA3
S113FFC039010182010101020300020203000202SD
S113FFD00342FDA443FDC444FDA747FDB84CFC9176
S113FFE04DFD3D4EFDCC50FE7152FF0A53FE5B5455
S113FFF0FD0055FD9F56FDABFC00FC0AFC05FC2022S9
```

Wydruk 5. Kod maszynowy programu MONITOR

Procedury komunikacyjne MONITORA

Program komunikuje się z terminalem za pomocą następujących procedur:

INCHAR — procedura wprowadzająca znak z terminala. Wywołanie: **JSR INCHAR.** Funkcja: procedura czeka na gotowość sekcji odbiornika układu ACIA i ewentualnie wprowadza znak do ACCA. Bit D_7 jest zerowany i badana jest zawartość komórki $E00C_H$ (ECHO). Wartość zero pod tym adresem powoduje wywołanie procedury OUTCHAR. Wyjście: znak z terminala w ACCA. Adres procedury: $FD1F_H$.

OUTCHAR — procedura wyprowadza znak do terminala. Wejście: znak do wysłania w ACCA. Wywołanie: **JSR OUTCHAR.** Funkcja: procedura sprawdza gotowość sekcji nadajnika ACIA i ewentualnie wysyła znak do terminala. Adres procedury: $FD08_H$.

Obie procedury nie niszczą zawartości ACCB i X.

Struktura programu

Znajdujące się w programie tablice stałych są umiejscowione jak następuje: $FC0F$-$FC1F_H$, $FD32$-$FD3C_H$, $FED0$-$FED8_H$ i $FFC1$-$FFFF_H$.

Program MONITOR napisany jest w sposób niezwykle zwarty. Wykorzystane są wszystkie 1024 komórki pamięci. Stanowi to pewne utrudnienie przy wszelkich próbach rozszerzenia programu. Repertuar dyrektyw można jednak powiększyć umieszczając nową, bogatszą tablicę kodów i adresów poza obszarem programu. Należy wtedy zmienić adresy w procedurze przeszukiwania; nowy adres początkowy tabeli powinien być umieszczony pod adresem $FC5C/D_H$, a nowy adres końca tabeli (+ 1 komórka) w miejscu $FC66/7_H$.

Po kilku miejscach programu zastosowany jest pewien chwyt pozwalający na ścieśnienie kodu. Oto przykład:

```
FE01 C4 FD      ANDB # FD
FE03 8C C4FE    CPX # C4FE
FE06 54         LSRB
FE07 24 02      BCC DALEJ
```

* * *

Instrukcja ANDB # FD to iloczyn logiczny zawartości ACCB ze stałą FD_H. Program wykonany od adresu $FE01_H$ wykona operację iloczynu ze stałą FD_H, rozkaz porównania rejestru X ze stałą nie zmieni ACCB i wykonane zostaną rozkazy przesunięcia i skoku warunkowego. Jeżeli jednak z pewnego miejsca programu zostanie wykonany skok do adresu $FE04_H$ to pierwszy bajt a r g u m e n t u rozkazu CPX potraktowany zostanie jako kod rozkazu ANDB z argumentem FE_H. Wykonując zatem program od adresu $FE04_H$ rozkaz LSRB nastąpi po wykonaniu iloczynu ze stałą FE_H.

Powyższy zabieg pozwala na zaoszczędzenie jednej komórki pamięci programu.

JERZY ORKISZEWSKI
Warszawa

Wkładkę mikroKLAN prowadzi:

mgr inż. Andrzej J. Piotrowski
tel. domowy 48-22-85

Zanim wykręcisz ten numer weź pod uwagę, że jesteś jednym z kilkudziesięciu tysięcy zainteresowanych mikrokomputerami.

STANISŁAW WALIGÓRSKI
Instytut Informatyki
Uniwersytet Warszawski

LOGO (2)

Słowa i listy

W pierwszej części artykułu omówiono komendy i procedury. Poniżej przedstawiono operacje na słowach i listach.

SŁOWA

Słowo w sensie LOGO jest to ciąg dowolnych znaków, różnych od odstępu i nawiasów kwadratowych [i], lub ciąg dowolnych znaków, różnych od apostrofów ', ujęty w apostrofy. W tym drugim przypadku apostrofy są tylko „opakowaniem" słowa i do niego nie należą. Przykładowo, wszystkie poniższe ciągi są słowami:

```
CO??
'Z ODSTĘPAMI [ALE W APOSTROFACH]'
HOME
+
STARA. WARTOŚĆ. X
X
:X
"X
+5513
—27.32
1.37E5
152N3
```

Słowem może więc być liczba, stała napisowa, nazwa procedury języka lub procedury definiowanej, operator arytmetyczny, nazwa zmiennej lub parametru procedury. Wszystko zależy od tego, w jaki sposób słowo zostało użyte — interpretacja i sposób potraktowania słowa zależy od kontekstu, w jakim ono się znajdzie.

W razie wątpliwości, czy coś jest słowem, można użyć procedury WORD?, która przyjmuje wartość TRUE, jeśli wartość jej argumentu jest słowem i FALSE w przeciwnym wypadku (na przykład, gdy jest listą, a nie słowem). Przykładowo, wywołanie

```
PRINT WORD? "COTO??
```

spowoduje pojawienie się na ekranie napisu TRUE. Komenda PRINT jest potrzebna do wydrukowania wyniku, gdyż w przypadku procedur udostępniających jakąś wartość, trzeba zawsze określić, co należy z nią zrobić (inaczej byłby sygnalizowany błąd). Cudzysłów " przed argumentem znaczy, że chodzi właśnie o słowo COTO??, a nie o procedurę o tej nazwie.

Wiedząc, że coś jest słowem, można badać przy użyciu procedur jednoargumentowych, przyjmujących wartości TRUE lub FALSE, czy jest liczbą (procedura NUMBER?), czy jest zmienną, mającą wartość (procedura THING?).

Liczby i działania na liczbach

Liczby w LOGO mogą być całkowite albo rzeczywiste, w zapisie stało- lub zmiennopozycyjnym. Charakterystyczne dla zapisu stałopozycyjnego liczb rzeczywistych jest użycie kropki dziesiętnej, zaś dla zapisu zmiennopozycyjnego — litery **E** lub **N** oraz, ewentualnie kropki dziesiętnej. Litera **E** oznacza, że liczba ją poprzedzająca ma być pomnożona przez 10 podniesione do potęgi podanej po tej literze (jest to zawsze liczba całkowita dodatnia). Litera **N** oznacza, że wykładnik ma być ujemny. Tak na przykład, 45E2 oznacza 4500, —2.714E6 oznacza —2714000, a 3125N3 jest tym samym co 3.125. Znaki liczby nie mogą być oddzielone od siebie odstępami ani przecinkami.

Znaki czterech działań arytmetycznych są takie same, jak w innych językach programowania: +, —, *, /, odpowiednio dla dodawania, odejmowania, mnożenia i dzielenia. Znaki + i — mogą być również używane jako operatory jednoargumentowe — umieszczenie ich bezpośrednio przed liczbą lub zmienną o wartości liczbowej oznacza pomnożenie jej odpowiednio przez +1 lub —1. Taki operator jednoargumentowy nie może być oddzielony od swego argumentu odstępem. Byłoby błędem napisać — 2 albo — :X zamiast —2 albo —:X. Natomiast dla uniknięcia nieporozumienia, czy chodzi o operator jedno- czy dwuargumentowy, należy znaki dodawania i odejmowania oddzielać od swoich argumentów odstępami (dla jednolitości przyjmuje się tę zasadę dla wszystkich czterech operatorów). Przykłady zostały podane wyżej.

Jeśli w komendach LOGO występuje jedno z tych czterech działań, to obowiązuje następująca kolejność czynności:

- wyznacza się wartości wszystkich parametrów
- wykonuje się wszystkie mnożenia i dzielenia; kolejność ich wykonania jest od lewej do prawej, tak że na przykład 6 / 3 * 4 jest równe 2 * 4, a nie 6 / 12
- wykonuje się wszystkie dodawania i odejmowania, także od lewej do prawej
- jeśli występują nawiasy, to mogą one określić kolejność wykonywania działań w zwykły sposób.

Poza tymi czterema działaniami w LOGO używa się następujących operacji:

QUOTIENT — część całkowita ilorazu dwu liczb
REMAINDER — reszta z dzielenia dwu liczb
ROUND — zaokrąglenie liczby do najbliższej całkowitej
INTEGER — część całkowita liczby
SQRT — pierwiastek kwadratowy
SIN — sinus
COS — cosinus
ATAN — arctg
RANDOM — jeśli wartość argumentu jest liczbą całkowitą dodatnią, to wartością procedury jest liczba wylosowana, w rozkładzie równomiernym prawdopodobieństwa, z przedziału pomiędzy 0 a tą wartością argumentu minus 1 (na przykład, wywołanie RANDOM 3 powoduje wylosowanie liczby spośród trzech: 0, 1, 2).

Jeśli którekolwiek z dwu argumentów operacji QUOTIENT albo REMAINDER nie są całkowite, to najpierw stosuje się do nich automatycznie ROUND, a potem bierze się część całkowitą ilorazu lub resztę dla liczb całkowitych.

Inne interpretacje słów i operacje na słowach

Nazwy zmiennych lub procedur definiowanych mogą być dowolnymi słowami LOGO, ale dobry zwyczaj programowania nakazuje, aby budować je z ograniczonego zestawu znaków, zawierającego litery, cyfry i ewentualnie wybrane znaki przestankowe, jak pytajnik lub kropka. Każda nazwa powinna zgodnie z tym zwyczajem zaczynać się od litery.

Nazwa nie poprzedzona dwukropkiem (:) ani cudzysłowem (") jest z reguły traktowana jako nazwa procedury i jeśli nie da się jej prawidłowo zinterpretować w tej roli, to jest sygnalizowany błąd. Jeśli chodzi o wartość zmiennej, należy przed jej nazwą umieścić dwukropek. Poprzedzenie słowa cudzysłowem " oznacza, że należy je wziąć dosłownie jako stałą napisową.

Istnieje jeden wyjątek od tych reguł. Słowo znajdujące się wewnątrz pary nawiasów kwadratowych (czyli, jak zobaczymy poniżej, będące elementem listy) jest zawsze brane dosłownie jako stała napisowa i, na przykład, poprzedzenie go w tej sytuacji dwukropkiem byłoby bezskuteczne w tym sensie, że nie spowodowałoby pobrania wartości zmiennej o tej nazwie. Tak samo, słowo w tej sytuacji nie może być zinterpretowane jako wywołanie procedury.

Procedura WORD służy do składania słów. Przykładowo, komenda .

```
MAKE "X3 WORD "IKS "TRZY
```

spowoduje nadanie zmiennej X3 wartości IKSTRZY. Procedura WORD jest zasadniczo dwuargumentowa, ale jest także, jak to się określa w LOGO, „zachłanna". Jeśli ujmiemy w nawiasy okrągłe operator WORD i następujący po nim dowolny ciąg słów, oddzielonych odstępami od siebie, od WORD i od nawiasu zamykającego, to wartością takiego wyrażenia będzie słowo złożone ze wszystkich tych słów (lub ich wartości, jeśli są to zmienne), np.

```
MAKE "X2 (WORD "IKS "— "DWA )

MAKE "X1 (WORD "IKS " ' ' "RA "Z )
```

Jednoargumentowa procedura THING wyznacza wartość zmiennej, której nazwa jest wartością argumentu tej procedury.

Zatem wywołanie:

```
MAKE "I 1
PRINT THING WORD "X I
```

spowoduje wypisanie na ekranie wartości zmiennej X1, czyli

```
IKS RAZ .
```

Natomiast wywołanie

```
PRINT THING WORD "X "2
```

spowoduje wypisanie

```
IKS—DWA
```

W LOGO nie ma tablic i zmiennych wskaźnikowych, ale możliwość stosowania takiej techniki pozwala się bez nich obejść. Można dokonywać superpozycji procedury THING, np. wywołanie:

```
MAKE "ZMIENNA "X3
PRINT THING "ZMIENNA
PRINT THING THING "ZMIENNA
```

powoduje wydrukowanie kolejno

```
X3
IKSTRZY
```

Dwie ostatnie komendy można by uprościć, pisząc

```
PRINT :ZMIENNA
```

bo w tym przypadku użycie procedury THING jest tym samym, co dwukropek przed nazwą zmiennej oraz

```
PRINT THING :ZMIENNA
```

Tej ostatniej komendy już bez użycia procedury THING zapisać się nie da. Komenda

```
MAKE "IKSTRZY "ZMIENNA
```

stwarza dalsze możliwości wędrówek po tak utworzonej strukturze.

LISTY

Lista jest to ciąg słów lub list, ujęty w nawiasy kwadratowe, albo para nawiasów kwadratowych []. W tym ostatnim przypadku mówimy, że lista jest pusta. Jeżeli w liście słowa następują po sobie, to muszą być oddzielone odstępami, np.

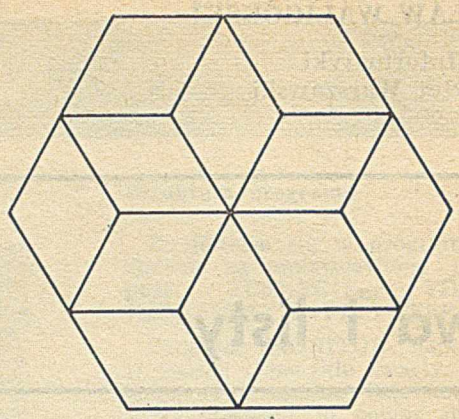
```
[— TA LISTA MA 8 RÓŻNYCH ELEMENTÓW !]
[— [TA LISTA] [MA [ICH [[2 RAZY] MNIEJ]]] !]
```

Przykładem listy jest drugi argument komendy REPEAT. Może tu też istnieć lista, której elementem jest lista, np.

```
REPEAT 6 [RIGHT 60 REPEAT 6 [FORWARD 40 RIGHT 60]]
```

Jeśli żółw był przedtem w położeniu HOME i miał pióro opuszczone, to po wykonaniu tego zdania powstanie obraz jak na rys. 1.

Wartości, obliczane przez pewne procedury, mają postać list. Przykładowo, wartością procedury bezargumentowej TURTLESTATE jest lista złożona z czterech elementów:



Rys. 1. Obraz powstały po wykonaniu komendy REPEAT

- TRUE — jeśli pióro jest opuszczone lub FALSE — gdy jest podniesione
- TRUE — gdy żółw jest widoczny lub FALSE — gdy jest ukryty
- numer koloru tła
- numer koloru kreski rysowanej przez pióro.

Jeśli wartością argumentu procedury TEXT jest nazwa procedury, to wartością TEXT jest lista list, opisujących kolejne wiersze tekstu tej procedury.

Można przyjąć, że rysunki rysowane przez żółwia składają się z pewnej liczby łamanych. Po każdej takiej łamanej żółw przechodzi z piórem opuszczonym, rysując ją, a przy przejściu od jednej łamanej do następnej podnosi pióro. Wtedy opis rysunku można przedstawić jako listę opisów łamanych.

Każdemu odcinkowi łamanej można przyporządkować parę liczb: długość odcinka oraz kąt, o jaki żółw ma obrócić się w prawo, gdy ten odcinek przejdzie. Jeśli pominiemy dla uproszczenia takie szczegóły opisu, jak kolor linii, to opis łamanej będzie listą list dwuelementowych.

Jeżeli w takim opisie łamanej jakaś para powtarza się na kolejnych miejscach listy, to możemy przyjąć, że piszemy ją tylko raz, a przed nią, jako element poprzedzający, piszemy liczbę, określającą ilość powtórzeń. Rozszerzając tę zasadę na powtarzające się sekwencje elementów opisów łamanej (szczegóły opisu metody pozostawiamy Czytelnikowi), możemy opisać łamaną z rys. 1 za pomocą listy

```
[6 [[0 60] 6 [40 60]]]
```

Opis całego rysunku składającego się z tej jednej łamanej otrzymamy ujmując to wyrażenie jeszcze raz w nawiasy kwadratowe.

Jest to opis rysunku z dokładnością do początkowego położenia żółwia. Aby był on jednoznaczny, trzeba podać współrzędne początkowe i początkowy kąt żółwia.

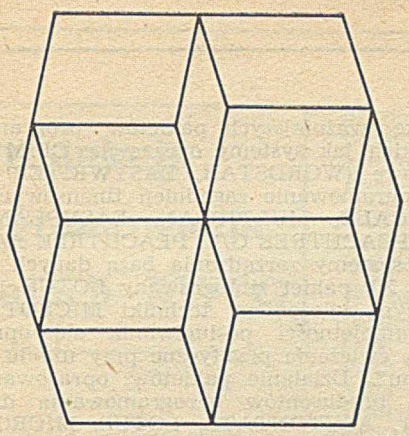
Innym sposobem opisania łamanej jest podanie współrzędnych jej kolejnych wierzchołków. Będzie to znowu lista list. Opis całego rysunku będzie listą opisów kolejnych łamanych, z jakich on się składa.

Opis zawierający współrzędne wszystkich punktów ma tę istotną zaletę, że taki rysunek można łatwo przekształcać, na przykład, przez zmianę skali. Jeżeli ma to być obraz jakiegoś przedmiotu w przestrzeni trójwymiarowej, to proste przekształcenie współrzędnych pozwoli pokazać ten przedmiot z innego kierunku lub punktu widzenia, a odpowiedni algorytm pozwoli dodać obrazowi więcej realizmu przez eliminowanie linii zasłoniętych.

Zakładając przykładowo, że na rys. 1 jest przedstawiona bryła, której wszystkie elementy są widoczne i nie ma linii zasłoniętych, możemy go opisać jako listę opisów trzech łamanych w przestrzeni trójwymiarowej:

```
[[[1 1 0] [2 1 0] [2 2 0] [1 2 0] [1 1 0] [1 1 1] [1 2 1] [0 2 1]
 [0 2 0] [1 2 0] [1 0 2] [0 0 2] [0 2 1] [0 2 2] [0 1 2] [0 1 1]
 [1 1 1] [1 1 2] [1 0 1] [1 1 1] [0 1 2] [1 1 2]] [[1 0 1] [1 0 2]
 [2 0 2] [2 0 1] [1 2 1]] [[0 1 1] [2 1 1] [2 1 0] [2 0 0] [2 0 1]
 [2 1 1]]].
```

Przekształcając odpowiednio układ współrzędnych, możemy na tę bryłę spojrzeć z innego kierunku (rys. 2). Zaprogramowanie tego wymagałoby zarówno znajomości LOGO, jak geometrii, ale zadanie nie jest zbyt trudne, zwłaszcza jeśli przyjmiemy, że wszystkie ścianki są przezroczyste i nie trzeba eliminować linii zakrytych.



Rys. 2. Przekształcenie bryły z rys. 1

Działania na listach

Następujące operacje jednoelementowe można stosować do list i do słów (obok nazwy procedury podano jej wartość):

FIRST	pierwszy element,
LAST	ostatni element,
BUTFIRST	pozostałość po odrzuceniu pierwszego elementu
BUTLAST	pozostałość po odrzuceniu ostatniego elementu.

Procedura SENTENCE służy do łączenia list w listę, której elementami są elementy list składowych. Przykładowo, wartością

```
SENTENCE [HEJ HEJ] [LA LA LA]
```

jest lista [HEJ HEJ LA LA LA]. Jeśli natomiast wartością któregoś argumentu jest słowo, a nie lista, to samo słowo jest brane jako element nowej listy, np. procedura

```
SENTENCE "DNIA [28 06 85]
```

ma wartość [DNIA 28 06 85]. Procedura SENTENCE jest zasadniczo dwuargumentowa, ale jest „zachłanna" podobnie jak WORD. Jeśli ujmie się w nawiasy okrągłe SENTENCE i następujący potem dowolny ciąg argumentów, przedzielonych odstępami od siebie i od nawiasu zamykającego ten ciąg, to wartością takiego wyrażenia będzie lista powstała przez połączenie wartości wszystkich tych argumentów. Przykładowo, wartością wywołania

```
(SENTENCE [BYĆ [ALBO [NIE BYĆ]]] "OTO [JEST PYTANIE])
```

łączącego listę dwuelementową, słowo i listę dwuelementową, jest lista pięcioelementowa

```
[BYĆ [ALBO [NIE BYĆ]] OTO JEST PYTANIE]
```

Lista zapisana w postaci ciągu, ujętego w nawiasy kwadratowe, jest zawsze brana dosłownie i jej elementy nie podlegają żadnym obliczeniom ani badaniom wartości. Dlatego we wszystkich przytoczonych przykładach ani przed listą, ani wewnątrz niej nie trzeba było stosować znaku dosłowności ("). Jeżeli trzeba utworzyć listę z wartości wyrażeń, zmiennych lub procedur, które powinny być przedtem obliczone, to nie należy ujmować ich bezpośrednio w nawiasy kwadratowe, ale posłużyć się procedurą SENTENCE albo LIST.

Procedura LIST służy również do łączenia wartości swoich argumentów w listę, ale w odróżnieniu od SENTENCE nie zdejmuje zewnętrznych nawiasów kwadratowych z tych wartości argumentów, które są listami. Tak więc dla ostatniego przykładu procedura LIST, użyta zamiast SENTENCE, udostępniłaby wartość

```
[[BYĆ [ALBO [NIE BYĆ]]] OTO [JEST PYTANIE]]
```

W razie wątpliwości, czy coś jest listą, można użyć jednoargumentowej procedury LIST?, której wartością może być TRUE lub FALSE.

Operacje wejścia-wyjścia obejmują również działania na listach. Procedura PRINT używana w wielu przykładach jest procedurą jednoargumentową, ale jest „zachłanna". Ujęcie słowa PRINT i następującego po nim ciągu w nawiasy okrągłe powoduje wydrukowanie całego tego ciągu. Nawias zamykający powinien być oddzielony odstępem od ostatniego elementu ciągu, np.:

```
(PRINT [TO BĘDZIE] "WYDRUKOWANE [BEZ NAWIASÓW])
```

Jeżeli argument PRINT ma wartość, która jest listą, to procedura działa podobnie jak SENTENCE, tzn. zdejmuje z niej zewnętrzne nawiasy kwadratowe. Zwykle robi się to prościej niż w powyższym przykładzie, ujmując po prostu zdanie do wydrukowania w nawiasy kwadratowe, np. wywołanie

```
PRINT [WITAJ. JAK MASZ NA IMIĘ?]
```

spowoduje wydrukowanie tego tekstu bez obejmujących go nawiasów. Po zakończeniu wykonania PRINT przechodzi się zawsze do następnego wiersza. Zgodnie z tą zasadą komenda

```
PRINT "
```

spowoduje wydrukowanie pustego wiersza, czyli tylko przejście do następnego.

Jeśli trzeba uniknąć przejścia do następnego wiersza po drukowaniu, należy użyć komendy PRINT1. Jednoargumentowa „zachłanna" procedura PRINT1 ma dokładnie takie same właściwości jak PRINT, z tym jednak wyjątkiem, że po jej wykonaniu nie przechodzi się do następnego wiersza. W ten sposób komenda

```
PRINT1 "
```

nie spowoduje żadnego widocznego skutku.

Do wczytywania danych służy procedura REQUEST. Wartością tej bezargumentowej procedury jest lista, utworzona ze słów wchodzących w skład wczytanego wiersza. Kontynuując przykład z drukowaniem przywitania można napisać

```
MAKE "IMIE REQUEST
PRINT SENTENCE :IMIE [— TO LADNE IMIE]
```

Wszystkie te operacje języka są stosunkowo proste, ale można za ich pomocą definiować nowe procedury o znacznie większych możliwościach. Poniżej podano przykłady definicji takich procedur, działających na listach. Pierwsza procedura bada, czy coś jest elementem listy.

```
TO MEMBER? :ELEMENT :LISTA
IF :LISTA = [] THEN OUTPUT "FALSE
IF :ELEMENT = FIRST :LISTA THEN OUTPUT "TRUE
OUTPUT MEMBER? :ELEMENT BUTFIRST :LISTA
END
```

Zastosowano tu komendę OUTPUT, która powoduje przypisanie nazwie procedury wartości argumentu oraz zakończenie wykonywania procedury.

Do zakończenia wykonywania procedury bez nadawania wartości służy komenda STOP. Istnieje jeszcze trzeci sposób kończenia wykonywania procedury, który jest połączony z przejściem na najwyższy poziom rekurencji, tzn. na poziom komend wprowadzanych z klawiatury — TOPLEVEL. Jest on użyteczny w sytuacjach awaryjnych, kiedy trzeba radykalnie przerwać tok obliczeń, np.

```
TO ERROR :TEKST
PRINT :TEKST
TOPLEVEL
END
```

Poniżej podano procedurę, która udostępnia N-ty element listy, jeśli ten istnieje (w niektórych wersjach LOGO jest to operacja języka).

```
TO ITEM :N :LISTA
IF :N < 1 THEN ERROR [ITEM: ZLY NUMER ELEMENTU]
IF :LISTA = [] THEN ERROR [ITEM: LISTA ZA KROTKA]
IF :N = 1 THEN OUTPUT FIRST :LISTA
OUTPUT ITEM :N — 1 BUTFIRST :LISTA
END
```

Przedstawione przykłady pokazują zasadę budowania procedur działających na listach, podobnie jak poprzednie przykłady pokazywały, jak budować procedury graficzne. Stosując te proste zasady, Czytelnik może opracować własne przykłady procedur umożliwiających rozwiązywanie różnych zagadnień. Do bardziej elementarnych można zaliczyć procedurę tworzącą rysunek według jego opisu w postaci listowej, do bardzo skomplikowanych — interpreter LOGO zbudowany metodą wciągania (ang. bootstrap).

Postęp w szkoleniu informatycznym

Na tych samych łamach przed laty (INFORMATYKA nr 2, 1978) opisywałem zastosowanie techniki video do szkolenia z zakresu informatyki, opierając się na przykładzie firmy ASI (ADVANCED SYSTEMS, Inc.). Szybki rozwój tego przedsiębiorstwa, ekspansja terytorialna i wyniki finansowe potwierdziły słuszność inwestowania w nowoczesne narzędzia szkoleniowe, wykorzystujące obraz, dźwięk i znakomicie zredagowany tekst (podręczniki, zadania, testy). Od chwili powstania w 1969 r. ASI przygotowała ponad 3000 kursów magnetowidowych dotyczących problematyki komputeryzacji, zarządzania, produkcji, marketingu, sprzedaży itp., przy czym część z nich przetłumaczono na język hiszpański, francuski, niemiecki, włoski i japoński. Do siedmiotysięcznego grona klientów (w 35 krajach na wszystkich kontynentach) należy ponad 90% słynnej grupy pięciuset największych przedsiębiorstw na świecie.

Większość materiałów szkoleniowych z informatyki ukierunkowana jest na użytkowników sprzętu IBM, co oczywiście nie wymaga specjalnego uzasadnienia. Chociaż bieżąco aktualizowane taśmy kursów magnetowidowych pozostają głównym wyrobem ASI, to oprócz tego wprowadza ona na rynek również nowoczesne narzędzia szkoleniowe, oparte na niemal już powszechnej dostępności terminali dużych komputerów, a przede wszystkim — komputerów osobistych. Komputer taki stanowi indywidualne stanowiska treningowe, dostosowane do poziomu i wymagań konkretnego użytkownika. Umożliwia on także prowadzenie bardzo efektownego szkolenia dla licznego, geograficznie rozproszonego audytorium.

Powstało szereg technik, będących urzeczywistnieniem hasła „nauczanie nowej technologii poprzez jej stosowanie". Różnią się one stopniem wykorzystania oraz rodzajem stosowanego komputera. Duży komputer z ekranowymi terminalami konwersacyjnymi może być wykorzystany jako:

- narzędzie dydaktyczne uzupełniające technikę video i drukowany tekst, opierając się na technice nauczania wspomaganego komputerem (Computer Assisted Instruction). Narzędzie to jest stosowane do wstępnej identyfikacji zasobu wiedzy, ustalania indywidualnego toku szkolenia, ćwiczeń i wdrażania nabytych umiejętności oraz kontroli przyswajanego materiału;
- podstawowe narzędzie dydaktyczne realizowane w technice CBT (Computer Based Training), umożliwiające szkolenie użytkowników rozproszonych geograficznie, z których każdy może wybrać indywidualny tok nauczania.

Podstawą powodzenia tych technik jest ich konwersacyjny charakter oraz przejrzysta logika oprogramowania kursów. Dodatkowa technika CPF (Curriculum Placement Facility) umożliwia precyzyjne określenie specyficznych potrzeb każdego uczestnika szkolenia, drogą sprawdzenia jego dotychczasowych umiejętności i wydrukowania na tej podstawie odpowiedniego zalecenia z kompletną listą elementów kursu lub też kompletnych kursów prowadzących do uzyskania żądanych umiejętności zawodowych. Technika CPF, stosowana łącznie z techniką CAI, znakomicie ułatwia kierowanie procesami szkolenia.

Duży komputer wykorzystywany jest również w metodzie PCIS (IBM Personal Computer Instructional System), ale jego rola ogranicza się do przechowywania treści kursów napisanych zgodnie ze standardem pakietów szkoleniowych IBM. Kursy te mogą być zarówno kopiowane na dysk elastyczny komputera osobistego, jak i przesyłane w przeciwną stronę. Rolę warsztatu szkoleniowego przejmuje komputer osobisty IBM PC, co nie tylko znacznie skraca czas odpowiedzi, ale również niemal dwukrotnie przyspiesza i zmniejsza koszt opracowywania nowych kursów. Gdy nie ma dostępu do dużego komputera lub też jego użycie jest niecelowe, stosuje się autonomiczne techniki treningowe, dostosowane do określonego mikrokomputera. Do tej klasy należą kursy obsługi klawiatury oraz nauki języka programowania, dostarczane na dyskach elastycznych lub kasetach przez różnych producentów komputerów osobistych. Technika MICROTUTOR opracowana wspólnie przez firmy ASI i ATI (American Training International) umożliwia szybką naukę ważniejszych pakietów programowych dla IBM PC, takich jak systemy operacyjne CP/M i DOS, systemy tekstowe (WORDSTAR, EASYWRITER i BENCHMARK), oprogramowanie zagadnień finansowych i księgowości (VISICALC, MULTIPLAN, EASYPLANNER, BPI GEN. ACC., PEACHTREE G/L, PEACHTREE A/R, PEACHTREE A/P), systemy zarządzania bazą danych (DBASE II, EASYFILER X), pakiet zintegrowany LOTUS 1-2-3.

Podstawowym założeniem techniki MICROTUTOR jest wdrażanie umiejętności posługiwania się oprogramowaniem poprzez ćwiczenia praktyczne przy użyciu tzw. „dzielonego ekranu". Działanie pakietów, opracowanych przez ważniejszych producentów oprogramowania dla IBM PC (MICROSOFT, ASHTONTATE, LOTUS, MICRO-PRO) jest symulowane w górnej połówce ekranu, natomiast dolna — zawiera wyjaśnienia, komentarze i instrukcje niezbędne do wykonania ćwiczeń wyświetlonych powyżej. Oprócz umiejętności operowania pakietem użytkownik MICROTUTORA poznaje również możliwości praktycznego wykorzystania pakietu.

W wielu tematach nawet najbardziej rozwinięte kursy konwersacyjne muszą być poparte demonstracją wizualną. Połączeniem interaktywnego trybu szkolenia z komunikatywnością video i atrakcyjnością gry komputerowej jest technika IVI (Interactive Video Instruction). Oprócz mikrokomputera IBM PC, sterującego przebiegiem kursu, w technice tej wykorzystuje się laserowy odtwarzacz płyty video z nagranym wykładem oraz kolorowy monitor z ekranem wrażliwym na dotyk, zapewniający szybką konwersację z uczestnikiem szkolenia.

Laserowy odczyt płyty video umożliwia niemal natychmiastowy dostęp do dowolnej partii materiału. Po załadowaniu dysku elastycznego z oprogramowaniem kursu oraz płyty video, na ekranie zostaje wyświetlone menu oraz niezbędne informacje wstępne. Od tego momentu uczestnik może wybrać własną indywidualną drogę szkolenia lub będzie prowadzony zgodnie z logiką specjalnego oprogramowania.

Nauka rozpoczyna się testem wstępnym, ustalającym rzeczywisty poziom i profil umiejętności kursanta. Pozwala to uniknąć zbędnych powtórzeń oraz wyznaczyć podstawy oceny wzrostu posiadanych już umiejętności w trakcie szkolenia. Zgodnie z ustalonym punktem startu zostaje wybrana i odtworzona z płyty video odpowiednia lekcja, po której następują ćwiczenia rozszerzające i utrwalające materiał. W przypadku wystąpienia trudności w opanowaniu materiału następuje powrót do określonej części wykładu, poprzez zwykłe dotknięcie ekranu monitora. Po zakończeniu lekcji badane jest opanowanie zawartych w niej tematów. W podobny sposób przeprowadzana jest kontrola opanowania określonych części materiału po kilku lekcjach (tzw. Mastery Test). W dowolnym momencie kursu wyniki testów mogą być wyświetlone na ekranie również w formie graficznej, w postaci histogramów.

Wyjątkowa efektywność metody IVI łączy się z zastosowaniem specjalnego języka wysokiego poziomu AUTHORITY, w którym wykonuje się oprogramowanie. W logice tego języka uwzględniono psychologiczne aspekty uczenia się. Percepcja materiału jest stymulowana dzięki włączeniu następujących technik:

- **zapytania** (inquiry), umożliwiającej uczestnikowi — poprzez dotyk — przerwanie wykładu w celu natychmiastowego powtórzenia jego części, zatrzymanie obrazu, przeskok do tekstowego streszczenia prezentacji lub też do pełnego tekstu
- **rozpoznawania** (recognition), służącej do konwersacji i demonstrowania opanowania prezentowanych treści
- **działania** (action), służącej do symulacji rzeczywistych procesów i związanych z nimi problemów ćwiczeniowych, a następnie do oceny wykazanych umiejętności.

Technika IVI została uznana za jedną z najbardziej efektywnych metod nauczania, a skwantyfikowanie oceny wyników szkolenia i jej wyświetlenie w postaci graficznej zwiększa efektywność kierowania szkoleniem. Zlecone przez rząd USA badania wykazały, że IVI pozwala skrócić czas szkolenia o 40-60% w porównaniu do metod tradycyjnych. Jest to dostatecznie wymowna przesłanka szybkiego rozwoju tej techniki, mimo stosunkowo wysokiej ceny niezbędnych urządzeń (ok. 10 000 dolarów nie licząc mikrokomputera IBM PC).

ANDRZEJ ELEK
Centrum Informatyki Gospodarki Morskiej, Gdańsk

Produkcja oprogramowania w krajach RWPG

Efektywność wykorzystania ETO w większości krajów socjalistycznych zależy obecnie przede wszystkim od dostępności oprogramowania i jego jakości. Mimo dużej liczby opracowanych pakietów programów użytkowych nadal jest ich za mało w stosunku do potrzeb. Wielu użytkowników zmuszonych jest do samodzielnego opracowywania środków programowych, co jest na ogół bardzo kosztowne i ekonomicznie nieefektywne.

Cykl życia środków technicznych ETO jest obecnie znacznie krótszy od cyklu życia oprogramowania. Kraje socjalistyczne współpracujące w ramach Komisji Międzyrządowej ds. Współpracy w dziedzinie Techniki Obliczeniowej (KM ETO) powinny więc dążyć do takiej organizacji wytwarzania oprogramowania użytkowego, która pozwoliłaby uniknąć nieuzasadnionych kosztów i szybciej zaspokoić potrzeby użytkowników informatyki. W ostatnich latach w poszczególnych krajach powstały duże zasoby oprogramowania, lecz nie dokonano dotychczas całościowej analizy ich przydatności.

Według danych posiadanych przez Centrum Koordynacyjne KM ETO, kraje socjalistyczne posiadają ponad 12 tys. produktów programowych dla różnych zastosowań, w tym Bułgaria — ok. 1 tys., Węgry — ok. 1,3 tys., NRD — ok. 4,5 tys., Związek Radziecki — ponad 6 tys. 238 pakietów programowych opracowano wspólnie w ramach prac KM ETO (jest to liczba pakietów, które przeszły odbiory międzynarodowe przed 1 stycznia 1983 r.). 127 pakietów włączono do Wspólnego Banku Programów Użytkowych KM ETO.

Według roboczej klasyfikacji stosowanej przez Centrum Koordynacyjne KM ETO, oprogramowanie użytkowe dzieli się na następujące grupy:

- produkty programowe ogólnego przeznaczenia (w tym systemy zarządzania bazami danych (SZBD), systemy wyszukiwania informacji, programy przygotowania i wprowadzania danych)
- narzędzia technologii programowania
- programy rozszerzające możliwości systemów operacyjnych
- programy ukierunkowane na metody obliczeniowe (metody optymalizacyjne, statystyczne, sieciowe, różne metody matematyczne, metody modelowania i symulacji)
- oprogramowanie przemysłowych systemów informatycznych (techniczne przygotowanie produkcji, sterowanie produkcją podstawową, ewidencja księgowa i zarządzanie działalnością finansową, zarządzanie kadrami i in.)
- oprogramowanie systemów informatycznych dla zastosowań nieprzemysłowych
- oprogramowanie systemów automatyzacji projektowania w budownictwie, energetyce, przemyśle maszynowym i w innych dziedzinach.

Na początku 1983 r. liczba wielokrotnych zastosowań pakietów programowych opracowanych w ramach KM ETO wynosiła średnio ok. 50, przy ogólnej liczbie zastosowań objętych ewidencją ok. 7000. Około 5% pakietów programowych miało ponad 200 zastosowań. Dane o zastosowaniach różnych grup oprogramowania przytoczono w tabeli.

Najszerzej stosowane są środki programowe ogólnego przeznaczenia i pakiety ukierunkowane na metody obliczeń, ponieważ są one przydatne w różnych branżach i gałęziach gospodarki narodowej. Dużą powtarzalność zastosowań obserwuje się również w przypadku oprogramowania przemysłowych systemów informatycznych.

Struktura zbiorów (banków, bibliotek) programów użytkowych w poszczególnych krajach w zasadzie niewiele się różni od struktury wspólnych zasobów oprogramowania KM ETO. Podstawową część (25 — 55%) stanowi oprogramowanie systemów informatycznych przemysłowych i nieprzemysłowych, ponad 10% — pakiety ukierunkowane na metody obliczeniowe, ok. 10% — środki programowe ogólnego przeznaczenia oraz ok. 3,5% — narzędzia technologii programowania.

Analiza istniejącego oprogramowania w poszczególnych krajach wykazuje, że wszędzie opracowywane są niezależnie podobne produkty programowe. Według szacunkowych danych, ponad 70% prac w dziedzinie tworzenia oprogramowania jest dublowanych. Jest oczywiste, że powoduje to niepotrzebny wzrost łącznych nakładów na produkcję oprogramowania. Poprawa tego stanu jest możliwa, jeśli zarówno w poszczególnych krajach jak i w ramach współpracy międzynarodowej zostanie udoskonalony system informowania wytwórców i użytkowników produktów programowych o istniejącym już lub wykonywanym oprogramowaniu, jeśli poprawi się jego jakość i udoskonali serwis. W ramach KM ETO istnieje potrzeba udoskonalenia systemu planowania wspólnych prac nad oprogramowaniem, w tym potrzeba specjalizacji poszczególnych krajów w wytwarzaniu określonych grup produktów programowych. Planowanie nowych opracowań powinno uwzględniać zainteresowanie poszczególnych krajów zakupami określonych produktów programowych. Celowe jest również rozwijanie współpracy międzynarodowej przy realizacji takich opracowań, a nawet tworzenie dwu- i wielostronnych organizacji międzynarodowych, w oparciu o doświadczenia bułgarsko-radzieckiego Instytutu Naukowo-Badawczego i Projektowego INTERPROGRAMMA.

Rozwiązanie problemów wytwarzania oprogramowania użytkowego jest niemożliwe bez udostępnienia projektantom i programistom nowoczesnych systemów operacyjnych, ponieważ zawierają one wiele istotnych narzędzi technologii programowania, takich jak np. narzędzia do testowania programów. Niestety, w krajach RWPG nie zorganizowano dotąd na właściwym poziomie dostaw i serwisu systemów operacyjnych.

W rozwoju zastosowań ETO w latach siedemdziesiątych i osiemdziesiątych widać wyraźnie, że w stosunku do nakładów na środki techniczne, nakłady na oprogramowanie znacznie wzrosły i mają tendencje do dalszego wzrostu. Jest to powodowane, po pierwsze — wzrostem liczby programów potrzebnych dla nowoczesnych systemów informatycznych, a po drugie — niedostatecznym wzrostem wydajności pracy programistów. Wydajność ta zależy głównie od poziomu kwalifikacji programistów oraz stwarzanych im warunków pracy.

Liczba specjalistów zajmujących się w krajach socjalistycznych wytwarzaniem oprogramowania użytkowego jest chyba zbyt duża, gdyż według

Zastosowanie powtarzalnych pakietów programowych

Grupa oprogramowania	Liczba pakietów objętych ewidencją zastosowań	Liczba zastosowań ogółem	Średnia liczba zastosowań 1 pakietu
Oprogramowanie ogólnego przeznaczenia	22	1 904	86
Narzędzia technologii programowania	1	14	14
Programy rozszerzające możliwości systemów operacyjnych	9	54	6
Programy ukierunkowane na metody obliczeniowe	36	2 402	66
Oprogramowanie przemysłowych systemów informatycznych	42	2 083	49
Oprogramowanie nieprzemysłowych systemów informatycznych	16	103	6
Oprogramowanie systemów automatyzacji projektowania	12	41	3
Razem	**138**	**6 601**	**48**

ocen ekspertów wynosi ok. 150 tys. osób. Większość oprogramowania (ok. 80%) we wszystkich tych krajach powstaje u użytkowników ETO, gdzie praca programistów jest zwykle słabo zorganizowana, a jej wydajność niska.

Producenci komputerów i organizacje wyspecjalizowane w produkcji oprogramowania opracowują i dostarczają w dużych ilościach oprogramowanie przeznaczone tylko do niektórych celów. Zajmują się tym zwykle duże zespoły wysoko wykwalifikowanych programistów, wskutek czego koszty oprogramowania są tu niższe niż u użytkowników. Wynika z tego celowość koncentracji programistów w dużych wyspecjalizowanych jednostkach, którym należy dostarczyć najbardziej nowoczesne środki techniczne i narzędzia technologii programowania.

Wytwarzanie oprogramowania metodami konwencjonalnymi nie zapewnia niezbędnej jakości produktu programowego, wskutek czego nakłady na konserwację i serwis mogą znacznie przewyższać koszt jego opracowania. Konieczne jest więc nawiązanie do światowych trendów, zgodnie z którymi tworzenie środków programowych przestaje być sztuką lecz staje się działalnością przemysłową. Przechodzenie na metody przemysłowe przy wytwarzaniu i serwisie oprogramowania wymaga rozwiązania wielu zagadnień organizacyjnych, metodologicznych i technologicznych oraz zapewnienia odpowiednich środków technicznych dla zajmujących się tym jednostek.

Jednym z najważniejszych problemów jest stworzenie przemysłowej technologii wytwarzania produktów programowych. Pojęcie technologii przemysłowej obejmuje cały proces wytwarzania, od momentu powstania zapotrzebowania na konkretne oprogramowanie, do momentu całkowitego zaprzestania jego stosowania, tzn. wszystkie stadia cyklu życia oprogramowania. Zalicza się tu analizę i projektowanie oprogramowania, jego realizację (kodowanie, testowanie, dokumentowanie) oraz stosowanie (eksploatację, serwis, modyfikację). Technologia programowania powinna spełniać następujące wymagania:

- wysoką wydajność pracy programistów we wszystkich etapach opracowania produktu programowego
- efektywną organizację planowania i zarządzania pracą zespołów programistów, a zwłaszcza stworzenie możliwości równoległego wykonywania prac przy wytwarzaniu wielkich systemów oprogramowania
- łatwość weryfikacji i testowania programów, ujawnianie i eliminacja błędów we wczesnych etapach prac
- łatwość wprowadzania zmian i poprawek
- łatwość przenoszenia produktów programowych do środowisk sprzętowych użytkowników.

Zagadnieniami technologii programowania w ramach KM ETO zajmuje się Rada ds. Zastosowań Środków Techniki Obliczeniowej. Podstawowe kierunki działalności w tym zakresie są następujące:

- podstawy naukowe technologii (inżyniera oprogramowania)
- metodologia i narzędzia dla wytwarzania produktów programowych dla różnych dziedzin zastosowań
- organizacja i zarządzanie produkcją oprogramowania przy wykorzystaniu komputerów.

Podstawowym, dotychczasowym wynikiem współpracy międzynarodowej w zakresie technologii programowania jest zbliżenie poglądów specjalistów w krajach uczestniczących w pracach KM ETO. Opracowano i przyjęto dokument pn. „Ogólna koncepcja technologii programowania" oraz opracowano i uzgodniono: „Uogólniony model cyklu życia produktów programowych", „Słownik terminologiczny w zakresie technologii", „Klasyfikator problemów technologii", a także materiał pn. „Ewidencja, przekazywanie i wykorzystywanie technologii programowania". Opracowano również wiele programów i pakietów narzędziowych, w tym: „Warsztat pracy programisty (TKP-1)" (ZSRR, odbiór międzynarodowy w 1981 r.), „Spektr", „Kontroler", „Bibliotekarz" (ZSRR, odbiór międzynarodowy w 1982 r.), „Środki technologiczne dla programowania w trybie interaktywnym (ANSWER)", „Wdrożenie metodologii Jacksona" oraz „Narzędzia dla technologii mozaikowej (MOZ-ART)" (WRL, odbiór międzynarodowy w 1983 r.). Pod koniec pięciolatki kraje uczestniczące w pracach KM ETO będą miały zapewniony dostęp do programów narzędziowych, opracowanych na podstawie jednolitej koncepcji technologii programowania.

W następnej kolejności należy zająć się problemami handlu oprogramowaniem oraz przygotowaniem wspólnych materiałów normatywnych i metodycznych.

Oprac.:
SWIETŁANA ASMOŁOWA

w oparciu o artykuł M. E. Rakowskiego i T. W. Żukowej „Organizacja rozwoju oprogramowania" (Wyczislit. Technika Socjalisticzeskich Stran, nr 15, 1984 r.)

WARUNKI PRENUMERATY NA 1986 R.

Prenumeratorzy zbiorowi — jednostki gospodarki uspołecznionej, instytucje i organizacje społeczne zamawiają prenumeratę dokonując wpłat na blankiecie „polecenie przelewu" rozszerzonym dla potrzeb Wydawnictwa o część dotyczącą zamówienia. Blankiety te będą dostarczane przez Zakład Kolportażu.

Prenumeratorzy indywidualni — osoby fizyczne zamawiają prenumeratę dokonując wpłaty w UPT lub NBP na blankiecie Wydawnictwa lub blankiecie NBP. Na odwrocie wszystkich odcinków blankietu należy wpisać tytuł czasopisma, okres prenumeraty, liczbę zamawianych egzemplarzy oraz wartość wpłaty.
Wpłacać należy na konto NBP III O/M Warszawa 1036-7490-139-11.

Prenumerata ulgowa — przysługuje wyłącznie osobom fizycznym — członkom SNT, studentom i uczniom szkół zawodowych. Warunkiem prenumeraty ulgowej jest poświadczenie blankietu wpłaty (przed jej dokonaniem) na wszystkich odcinkach pieczęcią Koła SNT, wyższej uczelni lub szkoły.

Sposób zamawiania prenumeraty taki sam jak dla prenumeraty indywidualnej.

Prenumerata ze zleceniem wysyłki za granicę — zamawia się tak jak prenumeratę indywidualną. Dodatkowo należy podać na blankiecie wpłaty nazwisko i dokładny adres odbiorcy. Cena prenumeraty ze zleceniem wysyłki za granicę jest dwukrotnie wyższa.

Przedpłaty na prenumeratę przyjmowane są w terminach:
— do 10 listopada na I kwartał, I półrocze i cały rok następny,
— do 28 lutego na II, III, IV kwartał i II półrocze,
— do 31 maja na III, IV kwartał i II półrocze,
— do 31 sierpnia na IV kwartał.

U w a g a!

Wpłaty na dwumiesięczniki przyjmowane są na okresy półroczne lub roczne.

Informacji o prenumeracie udziela — Zakład Kolportażu Wydawnictwa NOT-SIGMA, ul. Bartycka 20, 00-716 Warszawa, lub skr. poczt. 1004, 00-950 Warszawa, tel. 40-00-21 w. 249, 293, 297, 299 oraz 40-35-89 i 40-30-86.

Egzemplarze archiwalne czasopism — można nabyć za gotówkę w Klubie Prasy Technicznej w Warszawie ul. Mazowiecka 12, tel. 27-43-65 oraz w Dziale Handlowym Wydawnictwa ul. Bartycka 20 skr. poczt. 1004, 00-950 Warszawa, na rachunek dla instytucji lub za zaliczeniem pocztowym dla osób fizycznych.

Cena miesięcznika INFORMATYKA została ustalona na 120 zł za numer (35 zł — cena ulgowa).

Cena prenumeraty wg cennika					
kwartalna		półroczna		roczna	
normalna	ulgowa	normalna	ulgowa	normalna	ulgowa
360	105	720	210	1440	420

Zwiększenie pojemności kostek pamięciowych

W roku 1984 sprzedano na świecie za blisko 2,5 mld dolarów kostek dynamicznej pamięci operacyjnej (DRAM — Dynamic Randomm Access Memory) o pojemności 64 K bitów. Poniższa tabela pokazuje jednak, że był to szczytowy rok dla kostek o tej pojemności. Na początku 1985 r. przewidywano, że cena za bit w kostkach o pojemności 64 K bitów i 256 K bitów wyrówna się w ciągu 3—12 miesięcy. Jeśli istniejące tendencje utrzymają się, to produkcja kostek o pojemności 256 K bitów zwiększy się w 1985 roku 12-krotnie, w stosunku do roku ubiegłego (z 25 do 300 mln szt.). Rokiem szczytowym dla tego rodzaju kostek powinien być rok 1989, z przewidywaną produkcją 2—3 mld jednostek. Firmy, które chcą utrzymać się wśród czołowych wytwórców kostek 256 K bitowych muszą w końcu 1985 roku produkować 3—5 mln szt. miesięcznie. Większe szanse mają na to firmy japońskie, jak to pokazały już w 1984 roku. FUJITSU LTD. już obecnie sprzedaje 2,5 mln szt. miesięcznie, a NEC CORP. i HITACHI LTD. prawdopodobnie prawie tyle samo. Pozostałe firmy, tzn. OKI, MITSUBISHI ELECTRIC i TOSHIBA wciąż powiększają produkcję. Mimo to firmy amerykańskie nie czują się zagrożone i przestawiają swe najnowsze linie produkcyjne, które w ubiegłym roku wytworzyły miliony kostek o pojemności 64 K bitów, na nowe projekty kostek o pojemności 256 K bitów. Produkcją tą zainteresowane są również firmy zachodnioeuropejskie, jak SIEMENS AG w RFN i THOMSON-CSF we Francji.

Jednocześnie opracowuje się nowe technologie, które stosowane będą w następnych generacjach kostek, a także w kostkach megabitowych. Technologie te, dotyczące głównie litografii, pozwolą na dalsze zmniejszanie powierzchni matrycy i utrzymanie tempa obniżania cen, które wynosi 70% rocznie. Prace takie prowadzi m.in. TEXAS INSTRUMENTS w Houston, który jest obecnie czołowym dostawcą kostek o pojemności 64 K bitów, a w pierwszym kwartale 1985 r. rozpoczyna sprzedaż kostek 256 K wytwarzanych w technologii n-MOS. Firma ta od roku zwiększa produkcję swych wytwórni w Japonii.

Wprowadzane są też różne innowacje dla skrócenia czasu dostępu. Obejmują one zmniejszenie oporności połączeń dyfuzyjnych, bardzo szybkie układy adresacji bitów oraz wprowadzanie specjalnych kondensatorów zabezpieczających przed zakłóceniami. Obecnie powierzchnia matrycy kostki o pojemności 256 K bitów wynosi 40 mm² i według dotychczasowych tendencji zmniejszy się ponad dwukrotnie do roku 1989. Ocenia się, że przejście z pojemności 64 K bitów na 256 K bitów sprawiło konstruktorom i wytwórcom mniej problemów aniżeli poprzednio przejście z 16 K bitów na 64 K bitów. W połowie roku 1985 średni uzysk dobrych kostek o pojemności 256 K bitów osiągnie 37%, przy czym lepsi wytwórcy uzyskiwać będą 45%, a pod koniec tego roku parametr ten osiągnie średnio 40%. Latem 1985 r. cena kostki o pojemności 256 K bitów spadnie prawdopodobnie poniżej 10 dolarów (na początku 1985 r. rynkowa cena za te kostki kształtowała się ok. 20 dolarów, mimo oficjalnych cen na poziomie 30 dol.).

Pojemność kostki	Wartość sprzedaży w mln dolarów		
	1983 r.	1984 r.	1985 r.
16 K	270	100	30
64 K	1220	2440	1845
256 K	50	575	1675

Coraz więcej firm amerykańskich wprowadza do sprzedaży te kostki. MOTOROLA zaczęła dostawy kostek wytwarzanych w technologii n-MOS, o pojemności 256 K × 1 bit na początku stycznia 1985, z zapowiedzią uruchomienia produkcji kostek w technologii CMOS w tym samym roku. TEXAS INSTRUMENTS wprowadza kostki TMS 4256 o strukturze stronicowej i TMS 4257 o strukturze porcjowej (ang. nibble-mode). Również w pierwszym kwartale tego roku INNMOS CORP. oferowała próbki szybkich kostek CMOS o symbolu IMS 2800, o czasie dostępu od 60 do 100 ns, które wykorzystują nowy, przyspieszający ten proces, układ dekodowania kolumn, licencję którego zakupiła już japońska firma NMB SEMICON-DUCTOR LTD.

Również firma MOSTEK CORP. wprowadza 2 kostki: MK 4556 (256 K × 1 bit) i MK 4856 (32 K × 8 bitów). Pierwsza z nich ma czas dostępu do wiersza 59 ns. Firma ta przygotowuje też tańsze, nieco wolniejsze kostki w technologii n-MOS.

Kostki firmy MICRON TECHNOLOGY INC. mają mieć organizację 64 K × 4 bity i zawierać układy kodowania z korekcją błędów (ECC-error-correction-coding). Dostawy tych kostek mają się rozpocząć w drugim kwartale 1985 r. Podobne kostki planuje dostarczać w tym roku firma TRISTAR SEMICONDUCTOR INC. w Korei Południowej.

Również w tym roku MICRON rozpocznie dostawy próbek „grupowych" modułów pamięciowych, które zawierają np. dziesięć matryc o pojemności po 64 K bitów zawartych na jednym płatku krzemu. Próbki modeli o pojemności pół M bitów miały być dostarczone w lutym 1985 r., a modele o pojemności 2 M bitów — pod koniec roku.

W styczniu 1985 r. firma NATIONAL SEMICONDUCTOR CORP. planowała rozpocząć sprzedaż kostek NMC 41257 (256 K × 1 bit) z nowej linii produkcyjnej płytek 6-calowych w Salt City, a kostki o organizacji 64 K × 4 bity mają pojawić się w ciągu tego roku.

Jako nowa firma na tym rynku pojawiła się AT&T TECHNOLOGIES, która od początku roku 1985 zaopatruje hurtowników w kostki o pojemności 256 K bitów, wytwarzane w technologii n-MOS. Natomiast INTEL CORP., która już od pół roku sprzedaje kostkę 51C256 (256 K × 1 bit w technologii HCMOC) planuje wprowadzić obecnie wersję 64 K × 4 bity. Według dotychczasowych danych, sprzedawana kostka INTELA skutecznie współzawodniczy ze statycznymi kostkami o pojemności 64 K bitów, zwłaszcza w przenośnych systemach zasilanych z baterii.

W dalszych miesiącach br. planowane jest wprowadzenie kostki o pojemności 256 K bitów w technologii CMOS przez firmę ADVANCED MICRO DEVICES INC.

Opracował JAN RYŻKO
na podstawie czasopisma
ELECTRONICS WEEK nr 3, 1985

Instytut Automatyki Systemów Energetycznych

ul. Wystawowa 1, 51-618 Wrocław tel. 48-42-21 w. 240

sprzedaje

urządzenia typu UDMD-1

umożliwiające współpracę drukarki DZM 180 KSRE z monitorem ekranowym MERA 7911, w zestawie teletransmisyjnym złożonym z adaptera UPD-305-8/5 oraz modemów typu 200

EO/1155/K/85

P K P

CENTRALNE BIURO KONSTRUKCYJNO-TECHNOLOGICZNE

Z N T K w Poznaniu

OŚRODEK OBLICZENIOWY W OSTROWIE WIELKOPOLSKIM

oferuje

wszystkim ośrodkom obliczeniowym w kraju
posiadającym zestawy do lokalnej transmisji danych
typu: Jsg 7802 + MERA 7911

UDOSTĘPNIENIE W POSTACI LICENCJI KNOW-HOW

własnej, oryginalnej, prostej i skutecznej

METODY WYTWARZANIA LOKALNIE WIELODOSTĘPNYCH KONWERSACYJNYCH SYSTEMÓW UŻYTKOWYCH DLA KOMPUTERÓW ODRA 1305 i KOMPATYBILNYCH

Metoda ta umożliwia przygotowanie szerokiego zakresu zastosowań przez każdy średnio doświadczony zespół projektowo-programowy, posługujący się biegle językiem COBOL. W języku tym są programowane wszystkie funkcje użytkowe systemu w postaci odpowiednich podprogramów. Zestaw skompilowanych i odrębnie przetestowanych podprogramów jest w drodze konsolidacji przyłączany do gotowych modułów systemowych, realizujących zasadnicze funkcje organizacyjne związane z przesyłaniem informacji oraz rejestrowaniem prac wykonywanych na poszczególnych urządzeniach końcowych, a także z ochroną przed niepowołanym dostępem.

Korzystanie z naszej metody nie wymaga znajomości i instalowania pakietów MENAGER KOMUNIKACYJNY oraz DRIVER. Szerszy opis metody zostanie zamieszczony w kolejnym numerze „INFORMATORA DLA UŻYTKOWNIKÓW ELWRO".

Udostępnienie metody rozpoczynamy w 1986 roku w kolejności zawieranych umów licencyjnych

Zainteresowanych prosimy o bezpośredni kontakt z

CENTRALNYM BIUREM KONSTRUKCYJNO-TECHNOLOGICZNYM
ZAKŁADÓW NAPRAWCZYCH TABORU KOLEJOWEGO
W POZNANIU
OŚRODEK ZMECHANIZOWANYCH OBLICZEŃ

ul. Wrocławska 93, 63-400 Ostrów Wielkopolski, tel. 602-44, teleks 0462457

EO/1216/K/85

Nagroda czasopisma ELEKTRONICS WEEK

Od jedenastu lat wydawcy ELECTRONICS WEEK (poprzednio ELECTRONICS) przyznają nagrody za wybitne osiągnięcia w dziedzinie elektroniki i informatyki. Jeśli przyjrzymy się liście przyznanych dotychczas nagród, stwierdzimy, że obejmuje ona najistotniejszy dorobek w wymienionych dziedzinach, który pozostawił trwały wpływ na rozwój społeczeństwa w ostatnich latach. Rozpoczyna ją w 1974 r. Gordon E. Moore, prezes firmy INTEL CORP., która jako pierwsza wyprodukowała mikroprocesor. Następnie idą czterej naukowcy z firm IBM oraz PHILIPS za scalone układy logiczne, R. C. Dobkin z NATIONAL SEMICONDUCTOR CORP. za układy liniowe, C. H. House (HEWLETT-PACKARD) i B. Moore (BIOMATION CORP.) za usprawnienia w przyrządach pomiarowych, prezes STANDARD MICROSYSTEMS CORP. za technologię MOS, A. H. Bobeck z BELL LABORATORIES za pamięci na domenach magnetycznych, trzej naukowcy z PERKIN-ELMER CORP. za opracowanie układu ustawiającego maski projekcyjne, C. Maed (California Institute of Technology) i L. Conway (XEROX CORP.) za układy o bardzo dużym stopniu scalenia (VLSJ) oraz K. Thompson i D. M. Ritchie z BELL LABORATORIES za system operacyjny UNIX i język C. Listę zamyka Robert M. Metcalfe za opracowanie i udział w realizacji sieci lokalnych (zwłaszcza ETHERNET).

Nagroda tegoroczna została przyznana Alanowi F. Shugartowi za jego kierowniczą rolę w opracowaniu stacji dysków elastycznych. Działając na terenie różnych firm, Shugart przyczynił się do powstania taniej pamięci dyskowej, a tym samym — rozwoju komputerów osobistych. Miał on szczególną umiejętność doboru właściwych ludzi. Trzecia część jego współpracowników, którzy niegdyś przeszli z nim z IBM do MEMOREX, to dziś prezesi niezależnych firm.

Tegoroczny laureat poświęcił ponad 30 lat opracowywaniu magnetycznych urządzeń pamięciowych. Brał udział w opracowaniu pierwszej pamięci dyskowej RAMAC. Później pojawił się problem urządzenia wprowadzającego programy do systemu IBM 370. Pamięci półprzewodnikowe były już stosunkowo tanie, ale traciły informację po wyłączeniu zasilania. Zamierzano zastosować taśmę magnetyczną, ale kierujący wówczas pracami, D. Nelson, wybrał dyski elastyczne. Jednakże Shugart, nie mogąc w pełni realizować swych pomysłów, w 1969 roku przeniósł się do firmy MEMOREX, która stała się czołowym dostawcą pamięci dyskowych zgodnych ze standardami IBM. Osiągnięto w nich większe gęstości zapisu, a dysk elastyczny użyto do sterowania jednostką dyskową.

Jednakże po czterech latach wystąpiły trudności finansowe, w wyniku których firma nie mogła już inwestować w rozwój dysków elastycznych. Był to okres powstawania małych komputerów przeznaczonych dla indywidualnego użytkownika. Komputerom takim potrzebne były małe i tanie pamięci dyskowe. Shugart założył własne przedsiębiorstwo, jednak jego plany przekraczały rzeczywiste możliwości. Chciał tworzyć pełne systemy mikrokomputerowe, a nie tylko same pamięci dyskowe. W rezultacie znalazł się on bez pracy, dorabiając konsultacjami i prowadząc w pewnym okresie nawet bar. Oczywiście szukał sposobności by wrócić do przemysłu.

W roku 1979 wraz z F. Connorem zaprojektował nowy system pamięci dyskowej typu Winchester o 15-krotnie większej pojemności, w porównaniu do ówczesnych dysków elastycznych, chociaż system ten kosztował tylko trzykrotnie drożej. Poparcia finansowego udzielił N. Dion z firmy DYSAN CORP., w wyniku czego powstała firma SEAGATE, która w maju 1980 r. rozpoczęła dostawę systemów dyskowych Winchester o średnicy 5¼ cala. W ciągu trzech lat obroty wzrosły do 100 mln dolarów. Jednakże konkurencja i w tym przypadku okazała się silniejsza. W 1982 r. firma przeniosła większość swych zakładów produkcyjnych na Daleki Wschód, pozostawiając np. niezakończoną inwestycję zakładu w Watsonville. Shugart musi teraz wiele czasu poświęcić przedsiębiorstwu, zwłaszcza po odejściu Connera we wrześniu 1984 r.

Opracował JAN RYŻKO

na podstawie czasopisma ELECTRONICS WEEK nr 3, 1985

Zamykanie głów na klucz

Oferty ogłoszeniowe dla informatyków sięgają czasem wręcz zawrotnych kwot, nawet 100 tys. funtów szt. rocznie. Na łamach brytyjskiego tygodnika informatycznego COMPUTING ukazało się tak atrakcyjne zaproszenie dla wieloletnich (!) praktyków w dziedzinie CAD/CAF, którzy byliby w stanie dokonać na terenie Zjednoczonego Królestwa promocji pewnej firmy zachodnioniemieckiej wyspecjalizowanej w oprogramowaniu narzędziowym dla komputerowego projektowania inżynierskiego. Normalnie jednak programistom oferuje się znacznie skromniejsze warunki, 6—11 tysięcy funtów, a analitykom nieco więcej. Jest to niewątpliwie oznaką pewnego niedoboru kwalifikowanej kadry informatycznej. Na tym tle interesujące informacje przynosi ubiegłoroczna wypowiedź posła liberalnego z okręgu Yeovil, Paddyego Ashdowna [1]). Wyjawił on, że w ślad za uzgodnioną w lipcu 1984 r. nową wersją ograniczeń COCOM, rząd brytyjski może wydać dosyć osobliwe ograniczenia dla niektórych grup informatyków.

Tak więc po prawie 200 latach, kiedy to tkaczom nie wolno było wyjeżdżać poza Anglię — a zwłaszcza do USA — podobne prawo grozi obecnie programistom brytyjskim. Oczywiście teraz nie grozi już kara obcięcia czegoś tam, ale w myśl przygotowywanego projektu każdy programista wyjeżdżający z Wielkiej Brytanii musiałby uzyskiwać specjalną zgodę Ministerstwa Przemysłu i Handlu. Podobno analogicznych sankcji żąda również amerykański Departament Obrony.

Projekt nowych ograniczeń COCOM wywołał poważny sprzeciw w kręgach Brytyjskiego Towarzystwa Komputerowego (BCC) oraz wśród grupy posłów obawiających się utrudniania brytyjskiego eksportu, pod pozorem uniedostępniania krajom spoza COCOM nowoczesnych osiągnięć technicznych. Gdyby chęci pewnych kół potraktować na serio, to każdy brytyjski informatyk — zgodnie z procedurą COCOM — musiałby uzyskiwać indywidualną zgodę na wyjazd, zarówno prywatny jak i służbowy, także z drugiej strony Atlantyku.

Tej swoistej podejrzliwości strategicznej doświadczyli również poniektórzy nie-Brytyjczycy. Nawet specjaliści, ubiegający się o dłuższy pobyt w Wielkiej Brytanii, mimo ewidentnego deficytu kwalifikowanych informatyków na tym rynku — nie są dopuszczani do bardziej ambitnych prac.

Informacje te wywołały protesty wielu informatyków, uważających takie próby za zagrożenia wolności osobistej, a w dalszej konsekwencji — za ograniczenie swobodnej wymiany myśli naukowej.

A.B.E.

[1]) John Murphy: MP claims COCOM may restict staff. COMPUTING 1984.11.22, p. 5.

Słownik pojęć i terminów z dziedziny grafiki komputerowej (uzupełnienie)

Słownik opublikowany w numerach 2, 3 i 4 INFORMATYKI nie jest (i nie może być) opracowaniem zamkniętym, całkowicie zakończonym. Grafika komputerowa przeżywa bujny rozwój, również terminologiczny. Z miesiąca na miesiąc pojawiają się nowe pojęcia, terminy i określenia. W obecnej, czwartej części „Słownika" zamieszczamy pojęcia i terminy, które weszły w użycie w ostatnim okresie i nie znalazły się w trzech pierwszych częściach a, naszym zdaniem, powinny się w tym „Słowniku" znaleźć:

CZUJNIK, DETEKTOR (pick device) — GRAFICZNE URZĄDZENIE WEJŚCIOWE służące do wskazywania (wyróżniania) ELEMENTÓW OBRAZU na POWIERZCHNI OBRAZOWANIA, zwykle na ekranie MONITORA GRAFICZNEGO lub RYSOWNICY. CZUJNIKIEM jest PIÓRO ŚWIETLNE, PIÓRO NAPIĘCIOWE, ULTRADŹWIĘKOWE itp.

GREL → KOMÓRKA OBRAZU (płaskiego)

JASKRAWOŚĆ, LUMINACJA (brightness) — miara gęstości powierzchniowej natężenia światła decydująca o nasileniu subiektywnego wrażenia jasności. Jest równa stosunkowi natężenia światła ΔI elementu ΔS powierzchni świecącej w danym kierunku, do pola jego rzutu na płaszczyznę prostopadłą do tego kierunku

$$B = \frac{\Delta I}{\Delta S \cos \alpha}$$

Jednostki: stilb, apostilb, nit, lambert.

KOMENDA BEZWZGLĘDNA (absolute command, absolute instruction) — KOMENDA GRAFICZNA wprowadzająca urządzenie graficzne w stan, w którym DANE GRAFICZNE są interpretowane jako dane określone bezwzględnie.

KOMENDA WZGLĘDNA (relative command, relative instruction) — KOMENDA GRAFICZNA wprowadzająca urządzenie graficzne w stan, w którym DANE GRAFICZNE są interpretowane jako dane określone względnie.

KOZIOŁKOWANIE (tumbling) — dynamiczne przedstawienie obrotów ELEMENTÓW OBRAZU (bądź SEGMENTOW) wokół osi ciągle zmieniającej orientację.

LOKALIZATOR (locator) — urządzenie wejściowe KONSOLI GRAFICZNEJ służące do lokalizacji (określania WSPÓŁRZĘDNYCH położenia), np. ELEMENTU OBRAZU na POWIERZCHNI OBRAZOWANIA. Może nim być DRĄŻEK, MYSZ, MANIPULATOR TARCZOWY itp.

LISTA OBRAZOWA (display file) — ROZKAZY GRAFICZNE i WSPÓŁRZĘDNE zawarte w BUFORZE OBRAZU bezpośrednio opisujące OBRAZ FIZYCZNY (program syntezy OBRAZU FIZYCZNEGO wyrażany w ROZKAZACH GRAFICZNYCH urządzenia obrazującego).

MARKER (marker) — znak o specyficznym wyglądzie, służący do oznaczania specjalnych miejsc (pozycji) na POWIERZCHNI OBRAZOWANIA (zwykle MONITORA GRAFICZNEGO).

MIKROFILMOWANIE KOMPUTEROWE (computer output microfilming) — technika przekształcania i rejestracji DANYCH GRAFICZNYCH i tekstowych z komputera wprost na mikrofilm lub mikrofiszę.

MONITOR ALFANUMERYCZNY (alphascope, character display) — MONITOR EKRANOWY przeznaczony do prezentacji informacji alfanumerycznej bądź półgraficznej (graficznej w obszarze pola znaku alfanumerycznego).

MONITOR GRAFICZNY (graphic display, image display) — MONITOR EKRANOWY przeznaczony do prezentacji informacji graficznej (przedstawiania DANYCH GRAFICZNYCH).

ODZEW, ECHO (echo) — w komputerowych systemach graficznych bezpośrednie potwierdzenie dla operatora KONSOLI GRAFICZNEJ, że jego akcja — polegająca na użyciu urządzenia wejściowego (np. lokalizacja położenia) — została przez KONSOLĘ przyjęta. ODZEW ma zwykle postać specjalnego napisu bądź znaku graficznego na ekranie monitora.

PIERWOTNIK GRAFICZNY → ELEMENT OBRAZU

PIERWOTNIK WEJŚCIOWY (input primitive) — jednostka (element) danych, otrzymywanych z urządzenia wejściowego, jak klawiatura. WYBIERAK, LOKALIZATOR, CZUJNIK lub TAKSOWNIK.

SEGMENT (segment) — w grafice komputerowej zbiór ELEMENTÓW OBRAZU (PIERWOTNIKÓW GRAFICZNYCH), którymi można operować jako całością.

ŚLADOWNIK (stroke device) — urządzenie wejściowe KONSOLI GRAFICZNEJ, które generuje współrzędne swojego położenia (wyznaczające drogę przebytą przez urządzenie, np. po RYSOWNICY).

TAKSOWNIK (valuator) — urządzenie wejściowe KONSOLI GRAFICZNEJ służące do zadawania wielkości skalarnych. Może nim być MANIPULATOR TARCZOWY, potencjometr itp.

TRANSFORMACJA WIDOKOWA (viewing transformation, window/viewport transformation) — odwzorowanie zawartości obszaru OKNA w zawartość obszaru WZIERNIKA.

TREL (voxel) — komórka sześcienna cyfrowego obrazu trójwymiarowego.

WEJŚCIE GRAFICZNE (graphic input) — zdolność do wprowadzania DANYCH GRAFICZNYCH do komputera za pomocą GRAFICZNYCH URZĄDZEŃ WEJŚCIOWYCH.

WIELKOŚĆ EKRANU (screen area) — wymiary ekranu monitora wyrażone w jednostkach fizycznych (cm, cal).

WIELKOŚĆ PRZYROSTU (increment size) — odległość między dwoma sąsiednimi PUNKTAMI ADRESOWALNYMI na POWIERZCHNI OBRAZOWANIA. W wypadku monitorów i drukarek graficznych używany jest termin JEDNOSTKA RASTRU (raster unit), a w przypadku KREŚLAKA — KROK KREŚLAKA (plotter step size).

WSPÓŁRZĘDNE BEZWZGLĘDNE (absolute coordinates) — WSPÓŁRZĘDNE określające położenie ADRESOWANEGO PUNKTU w stosunku do POCZĄTKU określonego układu współrzędnych.

WSPÓŁRZĘDNE MODELU, WSPÓŁRZĘDNE GLOBALNE (world coordinates) — WSPÓŁRZĘDNE wprowadzane dla wygody opisu PROBLEMOWEGO MODELU realizowanego zadania.

WSPÓŁRZĘDNE PRZYROSTOWE (increment coordinates) — WSPÓŁRZĘDNE WZGLĘDNE wykorzystywane w ten sposób, że poprzednio adresowany punkt jest punktem odniesienia.

WSPÓŁRZĘDNE URZĄDZENIA (device coordinates) — WSPÓŁRZĘDNE określone w układzie współrzędnych określonego urządzenia.

WSPÓŁRZĘDNE UŻYTKOWNIKA (user coordinates) — WSPÓŁRZĘDNE określone przez użytkownika i wyrażone w sprzętowo niezależnym układzie współrzędnych.

WSPÓŁRZĘDNE WZGLĘDNE (relative coordinates) — WSPÓŁRZĘDNE określające położenie adresowanego punktu w stosunku do innego punktu adresowanego.

WYBIERAK (choice device) — urządzenie wejściowe KONSOLI GRAFICZNEJ służące do wybierania jednej możliwości ze zbioru danych. WYBIERAKIEM jest np. KLAWIATURA FUNKCYJNA.

WYJŚCIE GRAFICZNE (graphic output) — zdolność do wyprowadzania DANYCH GRAFICZNYCH z komputera na GRAFICZNE URZĄDZENIE WYJŚCIOWE.

ZNACZNIK (tracking symbol) — specjalny znak graficzny przeznaczony (w odróżnieniu od KURSORA) do współpracy z urządzeniami wejściowymi KONSOLI GRAFICZNEJ, umieszczony na POWIERZCHNI OBRAZOWANIA (ekranie monitora) w pozycji odpowiadającej parze współrzędnych podawanych za pomocą LOKALIZATORA.

ZNAKOCIĄG (string) — ciąg znaków zwykle alfanumerycznych (lub specjalnych) przesyłanych z urządzenia wejściowego do KONSOLI GRAFICZNEJ bądź bezpośrednio do komputera.

WOJCIECH MOKRZYCKI

Terminologia

Szanowna Redakcjo,

Zawsze z zainteresowaniem czytając terminologiczne artykuły p. red. dr. J. Zalewskiego zostałem jakoś sprowokowany artykułem z nr 12/84 INFORMATYKI („Czy nasze poglądy są kompatybilne") do przedstawienia kilku uwag:

1. Obok tezy *„Jestem jednak przeciwny wprowadzaniu wyrazów obcych wszędzie tam, gdzie mamy własne terminy, określające w miarę dokładnie to, o co chodzi"* proponuję dodać: *„Jestem jednak przeciwny wprowadzaniu wyrazów polskich wszędzie tam, gdzie nie mamy własnych terminów określających w miarę dokładnie to o co chodzi".*

2. Obok tezy *„Często jednak wyrazów obcego pochodzenia, które są akceptowane, używa się niezgodnie ze znaczeniem. Nie znając dokładnie znaczenia obcego wyrazu, ktoś może użyć go w niewłaściwym kontekście"* proponuję dodać — abstrahując od tego czy zamiast *często* nie powinno być *czasami* — *„Często jednak wyrazów polskiego pochodzenia, które są wprowadzane, używa się niezgodnie z przeznaczeniem. Nie znając dokładnie przeznaczenia polskiego wyrazu, ktoś może użyć go w niewłaściwym kontekście".*

I jeszcze jedno: cieszy mnie, że obok *sprzęgu* p. Zalewski dopuszcza (w końcu!) *interface*, aczkolwiek nadal zwalcza *interfejs*. Tylko jak ten biedny *interface* odmieniać w polskich tekstach pisanych?

Dr inż. M. T. JANKOWSKI

Systemy Zarządzania Bazą Danych

W dniach od 30 listopada do 5 grudnia 1986 r. w zamku Reinhardsbrunn (NRD) odbędzie się kolejne, dziewiąte międzynarodowe seminarium na temat systemów zarządzania bazą danych (SZBD). Tak jak poprzednio, współorganizatorami seminarium są specjalizujące się w tej dziedzinie oprogramowania instytucje badawczo-rozwojowe z Bułgarii (Interprogramma), Czechosłowacji (VUSEI-AR), NRD (Leitzentrum für Anwendungsforschung), Polski (Centrum Projektowania i Zastosowań Informatyki) oraz Węgier (SZAMALK).

Komitet Programowy zaprasza wszystkich zainteresowanych do uczestnictwa w seminarium oraz zwraca się z apelem o nadsyłanie referatów na następujące tematy:

- stan obecny oraz przewidywany rozwój w dziedzinie SZBD
- projektowanie bazy danych
- efektywność SZBD
- SZBD rozłożone przestrzennie
- SZBD w zastosowaniu do automatyzacji biur
- SZBD w zastosowaniu do wspomaganego komputerem projektowania (CAD) oraz wytwarzania (CAM)
- systemy oparte o zasoby wiedzy
- metodologia i normalizacja
- inne zastosowania SZBD.

Referaty o objętości do 20 stron maszynopisu (z podwójnym odstępem) winny być napisane w języku angielskim i przesłane w terminie do 15 marca 1986 r.: jeden egzemplarz do sekretariatu Komitetu Programowego, drugi — polskiemu przedstawicielowi Komitetu Programowego. Autorzy zostaną powiadomieni w terminie do 15 maja 1986 r. o przyjęciu lub odrzuceniu referatu. Wersja ostateczna przyjętego referatu, uwzględniająca proponowane poprawki i uzupełnienia, powinna mieć objętość do 12 stron maszynopisu (z pojedynczym odstępem) oraz postać przystosowaną do powielenia. Ostateczny termin przekazania materiałów do sekretariatu Komitetu Programowego upływa 30 czerwca 1986 r.

Dodatkowe informacje na temat referatów oraz seminarium otrzymać można od polskich członków Komitetu Programowego: **Władysława Boguckiego (tel. 41-21-81) lub Stanisława Mrozika (tel. 25-43-72), CPiZI, 02-555 Warszawa, Al. Niepodległości 190.**

Adres sekretariatu seminarium: **VEB Leitzentrum für Anwendungsforschung, DDR, 1156 Berlin, Jacques-Duclos-Str. 50/52, 9th ISDBMS (tel. 5-57-10, teleks 112849).**

EO/1215/K/85

Komitet ds. Systemu CAMAC przy Zarządzie Głównym Stowarzyszenia Elektryków Polskich organizuje w Warszawie w dniach 22—24 kwietnia 1986 roku, Międzynarodowe Sympozjum pn.

CAMAC — 86

Tematyka sympozjum:

- systemy modularne CAMAC, Multibus II, VME bus i inne,
- lokalne sieci komputerowe,
- projektowanie wspomagane komputerowo (systemy CAD/CAE),
- komputery osobiste i stanowiska robocze (ang. workstations).

Imprezą towarzyszącą będzie wystawa sprzętu komputerowego.

Szczegółowych informacji na temat sympozjum udziela:

doc. Roman Trechciński
Instytut Problemów Jądrowych
05-400 Otwock-Świerk
tel. 79-89-28, 79-88-33, teleks 813244.

Organizacja wystawy: **mgr Mirosław Herman, Interatominstrument, ul. Chocimska 28, 00-791 Warszawa, tel. 49-81-48, teleks 814859.**

Sekcja Automatyki i Pomiarów Oddziału Warszawskiego Elektroniki i Telekomunikacji SEP organizuje w Warszawie w dniu 25 września 1986 r. V Krajową Konferencję Naukowo-Techniczną pn.

Zastosowanie mikroprocesorów w automatyce i pomiarach

Konferencja poświęcona będzie:

- zastosowaniom mikroprocesorów w układach sterowania i aparaturze pomiarowo-kontrolnej,
- zastosowaniom układów kalkulatorowych w automatyce i pomiarach,
- podzespołom, blokom funkcjonalnym, urządzeniom pomocniczym oraz oprogramowaniu sterowników mikroprocesorowych,
- problemom projektowania, uruchamiania i diagnostyki mikroprocesorowych urządzeń automatyki i pomiarów.

Przewodniczącym Rady Naukowej Konferencji jest **prof. dr hab. inż. Krzysztof Badźmirowski.**

Wstępne zgłoszenie uczestnictwa w konferencji (z podaniem tytułu, kilkuwierszowego streszczenia oraz określeniem objętości proponowanych referatów i komunikatów) prosimy nadsyłać pod adresem sekretarza organizacyjnego:

mgr Lech Szyngwelski
00-241 Warszawa
ul. Długa 44/50

Przemysłowy Instytut Elektroniki
(tel. 31-52-21 wew. 371)

Zgłoszenia referatów i komunikatów będą przyjmowane do 15 grudnia 1985 r., a zostaną rozpatrzone w styczniu 1986 r. Zakwalifikowane wystąpienia, których teksty zostaną nadesłane do 15 marca 1986 r. będą opublikowane w materiałach konferencyjnych, doręczonych wszystkim uczestnikom. Program Konferencji zostanie ustalony i rozesłany uczestnikom w czerwcu 1986 r.

Iszkowski W., Grzegorzewicz P.: LOGOS — system generowania systemów operacyjnych

INFORMATYKA 1985, nr 8, s. 1

System LOGOS został opracowany w Instytucie Informatyki Politechniki Warszawskiej jako pomoc dydaktyczna do projektowania przez studentów prostych systemów operacyjnych.

Stępień C., Szyrkowiec J.: Wielopoziomowy system oprogramowania pamięci kasetowych PK-1

INFORMATYKA 1985, nr 8, s. 4

Charakterystyka systemu oprogramowania o nazwie II-SOPK, opracowanego w Instytucie Informatyki Politechniki Warszawskiej. System ułatwia współpracę mikrokomputerów opartych o mikroprocesor typu 8080 z pamięciami kasetowymi.

Broczkó P.: Produkcja mikrokomputerów w krajach RWPG

INFORMATYKA 1985, nr 8, s. 9

Przegląd mikrokomputerów produkowanych w 1984 r. w krajach RWPG. Omówiono rozwój w poszczególnych krajach oraz zestawiono w formie tabeli podstawowe parametry poszczególnych modeli.

Waligórski S.: LOGO (2). Słowa i listy

INFORMATYKA 1985, nr 8, s. 21

Druga część charakterystyki języka LOGO, zawierająca szczegółowe omówienie operacji na słowach i listach.

Iszkowski W., Grzegorzewicz P.: LOGOS — the system for operating systems generation

INFORMATYKA 1985, No. 8, p. 1

The LOGOS system has been elaborated in the Informatics Institute of the Warsaw Technical University as an educational instrument for students which design simple operating systems.

Stępień C., Szyrkowiec J.: The multilevel software system for the PK-1 cassette tape store

INFORMATYKA 1985, No. 8, p. 4

Characteristics of the II-SOPK software system, elaborated in the Informatics Institute of the Warsaw Technical University. The system assists the PK-1 cassette tape store cooperation with microcomputers, which are based on 8080 microprocessor.

Broczkó P.: Microcomputer production in the COMECON countries

INFORMATYKA 1985, No. 8, p. 9

Survey of microcomputers manufactured in 1984 in the COMECON countries. Development in separate countries as well as the table with basic parameters of individual microcomputer models are presented.

Waligórski S.: LOGO (2). Words and lists

INFORMATYKA 1985, No. 8, p. 21

Second part of the LOGO language characteristics, which includes detailed presentation of operations on words and lists.

Ишковски В., Гжегожевич П.: LOGOS — система генерирования операционных систем

INFORMATYKA 1985, № 8, с. 1

Система ЛОГОС разработана Институтом вычислительной техники Варшавского политехнического института в качестве учебного пособия при проектировании студентами простых операционных систем.

Стемпень Ц., Ширковец Я.: Многоуровневая система программного обеспечения кассетных ЗУ РК-1

INFORMATYKA 1985, № 8, с. 4

Характеристика системы программного обеспечения под названием II-SOPK, разработанной Институтом вычислительной техники Варшавского политехнического института. Система облегчает взаимодействие микро-ЭВМ на базе микропроцессора типа 8080 с кассетными ЗУ.

Брочко П.: Производство микро-ЭВМ в странах СЭВ

INFORMATYKA 1985, № 8, с. 9

Обзор микро-ЭВМ, выпускаемых в 1984 г. в странах СЭВ. Обсуждено развитие микро-ЭВМ в отдельных странах и составлено в форме таблицы основные параметры отдельных моделей.

Валигурски С.: LOGO (2). Слова и списки

INFORMATYKA 1985, № 8, с. 21

Вторая часть характеристики языка ЛОГО, содержащая подробное описание операций со словами и списками.

Iszkowski W., Grzegorzewicz P.: LOGOS — ein System zur Generierung von Betriebssystemen

INFORMATYKA 1985, Nr. 8, S. 1

Das LOGOS-System wurde im Informatik Institut der Warschauer Technischen Universität als didaktisches Hilfsmittel für Studenten zur Projektierung einfacher Betriebssysteme erarbeitet.

Stępień C., Szyrkowiec J.: Multiniveau Programmiersystem für PK-1 Kassettenspeicher

INFORMATYKA 1985, Nr. 8, S. 4

Eine Charakteristik des II-SOPK Programmiersystems, das im Informatik Institut der Warschauer Technischen Universität erarbeitet wurde. Das System unterstützt Mitwirkung des PK-1 Kassettenspeichers mit den auf 8030 Mikroprozessor basierten Mikrorechnern.

Broczkó P.: Mikrorechnerproduktion in den RGW-Ländern

INFORMATYKA 1985, Nr. 8, S. 9

Ein Übersicht der in den RGW-Ländern im Jahre 1984 produzierten Mikrorechner. Es wurden die Entwicklung in einzelnen Ländern und eine umfassende Tabelle mit Parametern der einzelnen Mikrorechnermodelle vorgestellt.

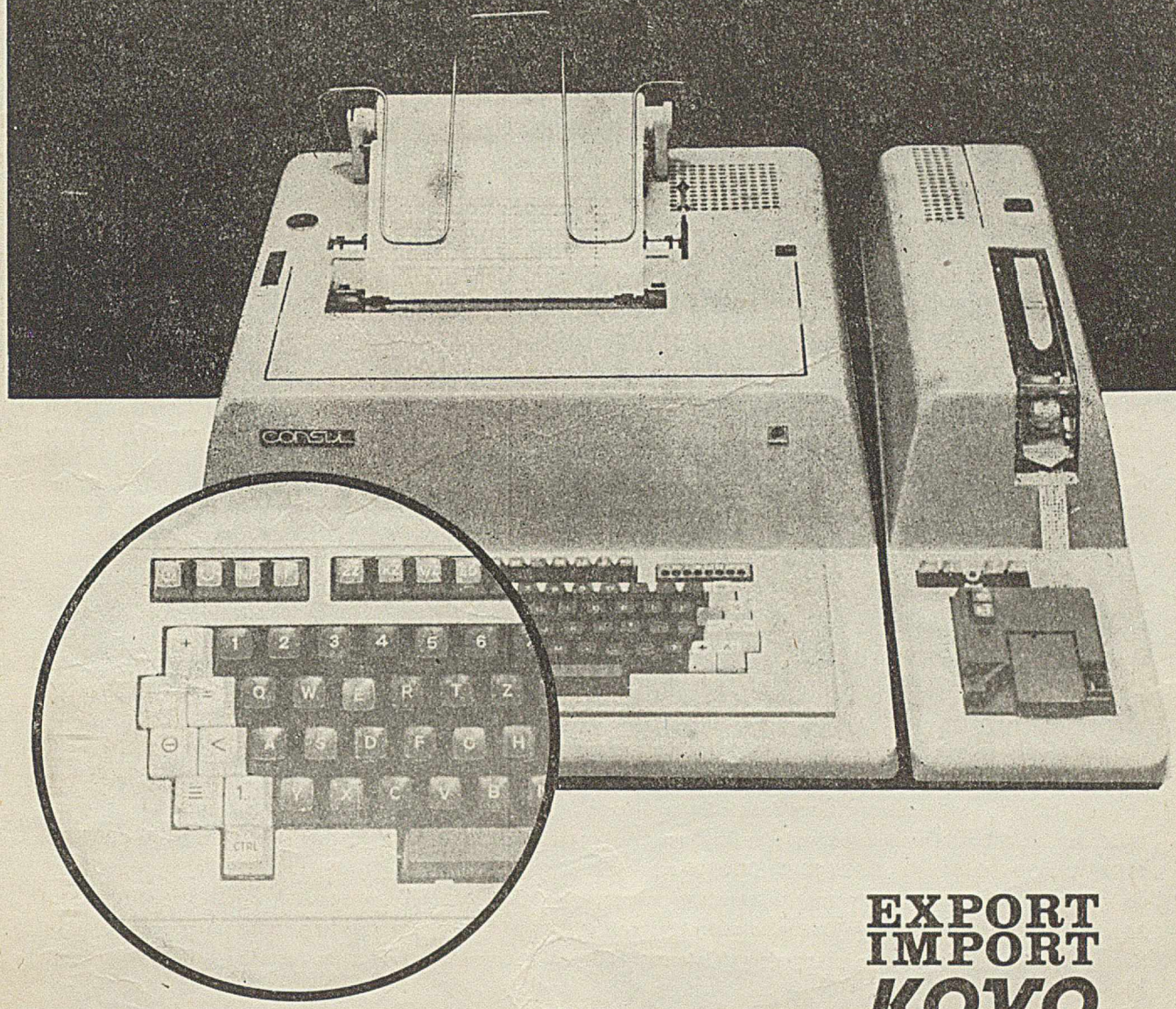
Waligórski S.: LOGO (2). Wörter und Listen

INFORMATYKA 1985, Nr. 8, S. 21

Zweiter Teil einer Charakteristik von LOGO-Sprache, die eine detailierte Besprechung der Wörter- und Listenoperationen umfasst.

CONSUL 321.2

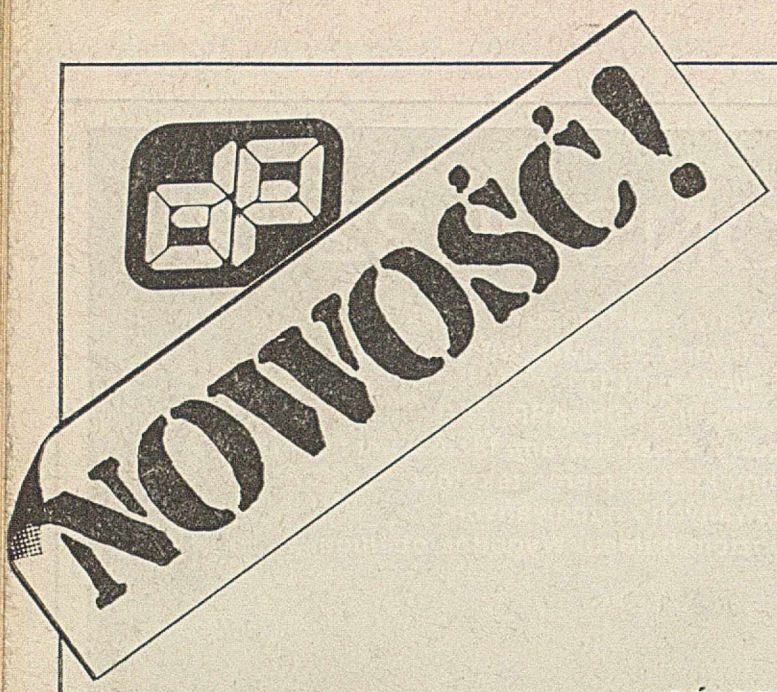
Nowoczesny, elektroniczny dalekopis przystosowany do pracy w kodzie TTA 2 lub ITA według CCITT.
Wykonanie RO, KSR, ASR, ESR.
Mozaikowy. Pisze literami łacińskimi i cyrylicą.
Odmienny kształt pisma tekstów przyjmowanych i nadawanych.
Niski poziom hałasu. Wygodna obsługa.



EXPORT
IMPORT
KOVO
PRAHA
CZECHOSLOVAKIA

Dostarcza:
PZO KOVO,
Praha 7,
ul. Jankovcova 2,
CSRS

EO/929/K/85



Jesteśmy znani na polskim rynku
już od ponad pięciu lat.
Nasze nowości to:

DPM 200

PÓŁPRZEWODNIKOWA PAMIĘĆ ZEWNĘTRZNA DLA KOMPUTERÓW ODRA I ICL

SEMIDISK DPM 200 to superszybka, półprzewodnikowa pamięć masowa o pojemności od 1,5 MB do 6 MB, zastępująca tradycyjne dyski magnetyczne. Jest niezbędnym uzupełnieniem konfiguracji przy pracy pod systemem GEORGE-3. Współpracuje z komputerami ODRA 1305, ODRA 1325 i ICL 1900. Symuluje bęben magnetyczny, jest kompatybilna z PB-304-1-256Ks, ICL-1963-512Ks i ICL-1964-2Ms.
SEMIDISK DPM 200 otwiera nową generację elektronicznych pamięci zewnętrznych, w pełni eliminując zawodne podzespoły mechaniczne. Gwarantuje wysoką niezawodność całego systemu, skracając kilkaset razy czas dostępu. Efekty ekonomiczne zapewnione. Wkrótce będziemy oferować podobną pamięć dla minikomputerów.

DPM 132

PÓŁPRZEWODNIKOWA PAMIĘĆ OPERACYJNA DLA KOMPUTERÓW ODRA 1325

dpm 132 to nowoczesna, półprzewonikowa pamięć operacyjna dla komputerów ODRA 1325 o pojemności 64 Ks lub 128 Ks. Zwiększa szybkość i niezawodność jednostki centralnej, zachowując pełną kompatybilność z dotychczasowym systemem operacyjnym. Po zainstalowaniu umożliwia pracę pod systemami E6RM i E6BM z dyskami w systemie UDAS.
Dzięki naszej pamięci DPM 132 otrzymujecie Państwo zmodernizowany komputer o potężnej mocy obliczeniowej, umożliwiający realizację obszerniejszych zadań o wiele szybciej niż dotychczas.

DANPOL

TWOJA SZANSA NA POSTĘP

Oferujemy nowoczesne rozwiązania, rozwinięte technologie, precyzję wykonania, wysoką jakość i wzorową obsługę klientów. Specjalizujemy się w elektronice, mechanice precyzyjnej i inżynierii materiałowej. Listy referencyjne naszych wyrobów obejmują już setki użytkowników. Nasze doświadczenie – to gwarancja sukcesu.
W dziedzinie elektroniki produkujemy między innymi pamięci operacyjne dla EC-1032 i EC-1035 (do 8 MB), ODRY 1305, ICL 1903 i ICL 1904 (do 512 Ks), ODRY 1325, ICL 2900 oraz dla minikomputerów PSPD-90, SM-4 i PDP-11. Zapewniamy fachową instalację, pełną obsługę gwarancyjną, serwis pogwarancyjny, dokumentację i szkolenie. Udzielamy gwaracji na okres 24 miesięcy. Uwzględniamy indywidualne potrzeby i życzenia użytkowników. Zawsze służymy pomocą i radą.
Nasi specjaliści chętnie udzielą wyczerpujących informacji i doradzą optymalne rozwiązanie. Jesteśmy zawsze do Państwa dyspozycji.

ul. Kopernika 71
81-456 Gdynia
tel.: 22-30-53
tlx: 054560 dan

ul. Szpitalna 1
00-020 Warszawa
tel.: 26-66-37
tlx: 816725 danw

DANPOL

EO/1108/K/85