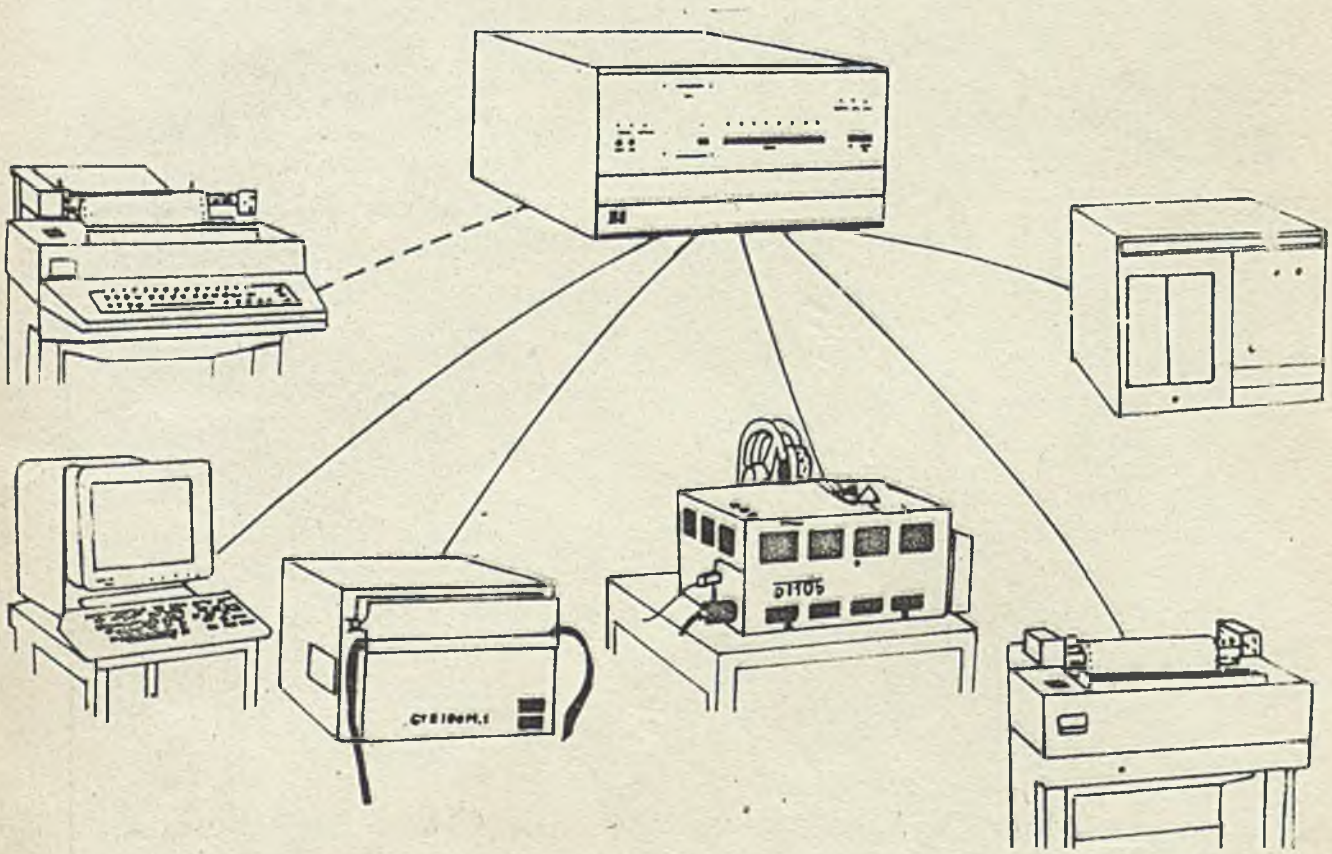
techniki komputerowe

5-6
'84

IMM

BIULETYN INFORMACYJNY



INSTYTUT MASZYN MATEMATYCZNYCH
BRANŻOWY OŚRODEK INTE

Rys. na okładce: Zestaw urządzeń systemu MSWP /zob. opracowanie
mgr inż. T.Sinkiewicza/

TECHNIKI KOMPUTEROWE

Rok XXII Nr 5-6 1984

Spis treści

str.

D W U M I E S I Ę C Z N I K

Wydaje:

I N S T Y T U T M A S Z Y N M A T E M A T Y C Z N Y C H

Branżowy Ośrodek Informacji Naukowej Technicznej i Ekonom.

Adres redakcji: ul. Krzywickiego 34, 02-078 Warszawa,
tel. 28-37-29 lub 21-84-41 w. 244

TECHNIKI KOMPUTEROWE 5-6/84

dr inż. Tadeusz SINKIEWICZ
Instytut Maszyn Matematycznych

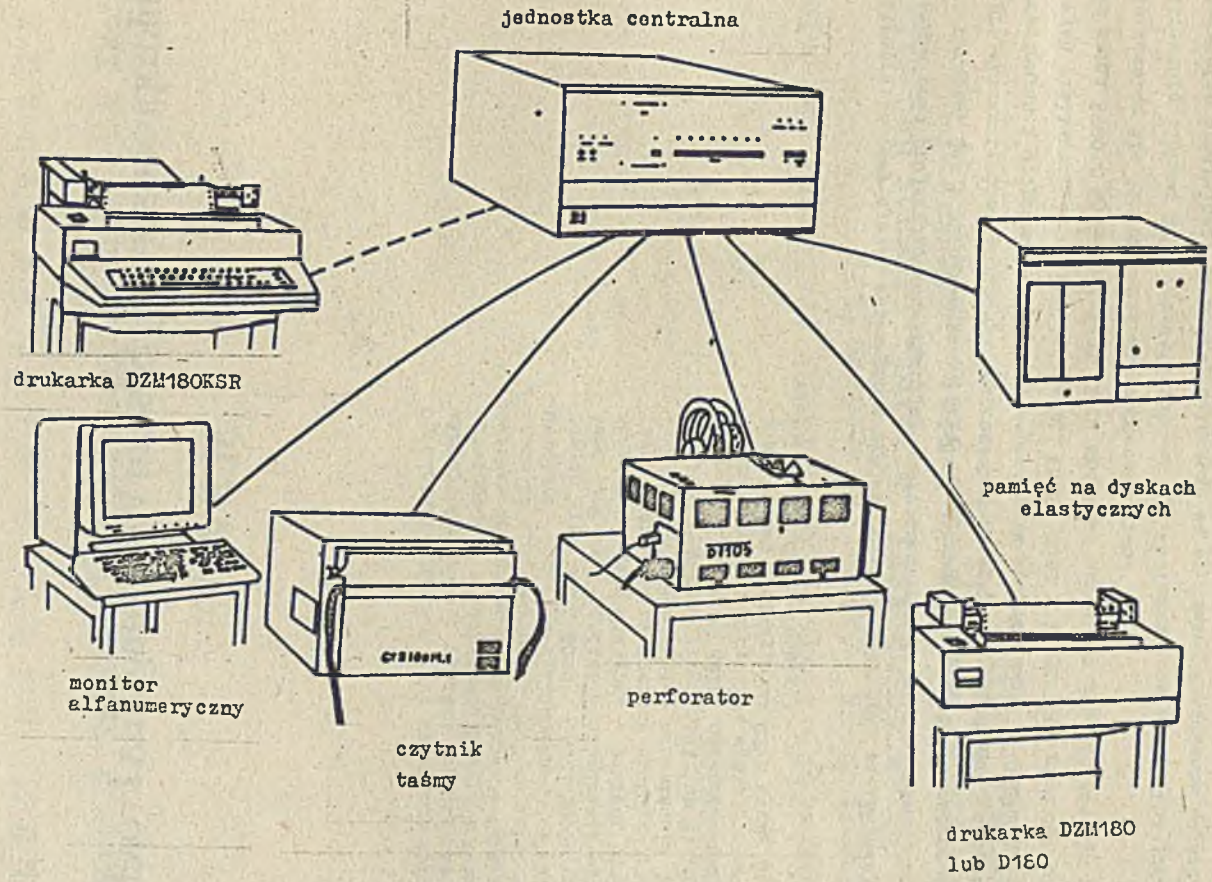
Mikroprocesorowy System Wspomagania Projektowania MSWP

Opracowanie zawiera opis struktury sprzętu i oprogramowania mikroprocesorowego systemu wspomagania projektowania - MSWP opracowanego w Instytucie Maszyn Matematycznych. System ten zawiera sprzętowe i programowe środki wspomagania przeznaczone do opracowywania, uruchamiania, badania i testowania urządzeń cyfrowych realizowanych na układach mikroprocesorowych typu INTEL 8080 i INTEL 3000. Omówiona została struktura blokowa i możliwe konfiguracje systemu oraz oprogramowanie systemowe dostępne w pamięci stałej, na taśmach perforowanych oraz na dyskach elastycznych. System stanowi rozszerzony funkcjonalny odpowiednik takich systemów, jak Intellec MDS-800 firmy Intel lub Microprocessor Lab 8002 firmy Tektronix. W skład oprogramowania systemowego wchodzą między innymi dwa dyskowe systemy operacyjne kompatybilne z systemami operacyjnymi CP/M firmy Digital Research i ISIS II firmy Intel.

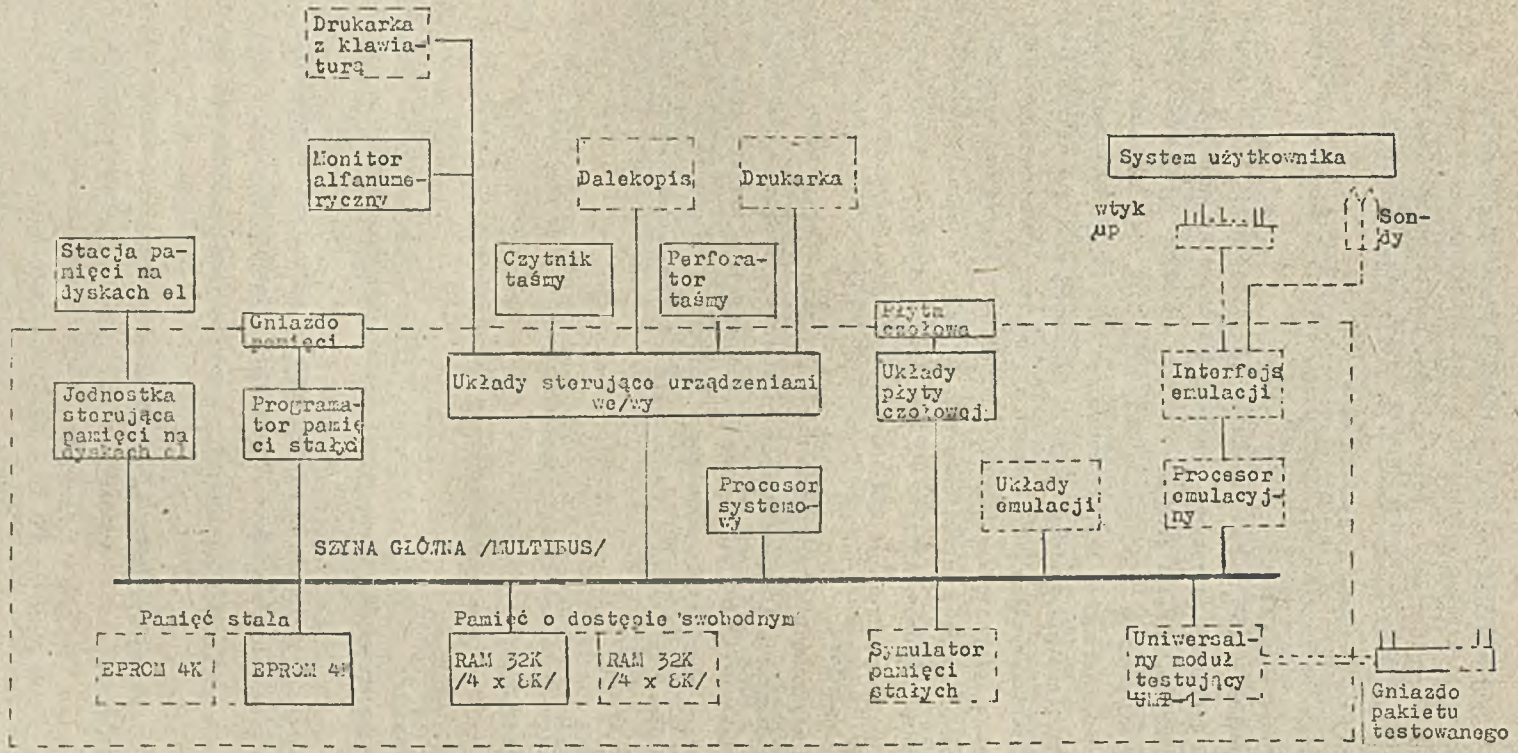
Mikroprocesorowy system wspomagania projektowania - MSWP jest wielofunkcjonalnym narzędziem wspomagania prac projektowo-konstrukcyjnych przeznaczonym do opracowywania, uruchamiania i badania, zarówno sprzętu, jak i oprogramowania urządzeń mikroprocesorowych. Mikroprocesorowy system wspomagania projektowania jest modułowym dwuprocesorowym systemem wspomagania dysponującym bogatym zestawem sprzętowych i programowych środków wspomagania projektowania i uruchamiania urządzeń mikroprocesorowych realizowanych na układach serii INTEL 8080 i INTEL 3000 oraz ich odpowiedników, np. układów serii K580 i K589 produkcji ZSRR. Modułowa struktura systemu pozwala rozszerzyć jego zastosowanie również dla innych serii mikroprocesorowych.

Strukturę blokową MSWP przedstawiono na rys.1b. Podstawowe bloki funkcjonalne MSWP umieszczone są w jednostce centralnej, do której dołączone są systemowe urządzenia wejścia/wyjścia, pamięć masowa na dyskach elastycznych i uruchamiany system użytkownika. Na płycie czołowej jednostki centralnej umieszczone są podstawowe elementy sterowania i sygnalizacji, przyciski przerwań i wymienne gniazdo programatora programowalnych pamięci stałych i układów FPLA.

Użytkownik systemu, w zależności od potrzeb i posiadanych środków, może kompletować MSWP w różnych konfiguracjach. Pozwala na to struktura bloków funkcjonalnych jednostki centralnej, magistrala systemowa typu MULTIBUS oraz rozwiązanie układów wejścia/wyjścia zapewniające możliwość stosowania różnych urządzeń zewnętrznych.



Rys. 1a. Zestaw urządzeń systemu



Rys.1b. Struktura blokowa MSWP

Moduły sprzętowe wchodzące w skład zestawu podstawowego MSWP pokazano na rys.1 linią ciągłą, natomiast moduły dodatkowe stanowiące rozszerzenie systemu - pokazane są linią przerywaną.

Zestaw podstawowy zawiera jednostkę centralną wyposażoną w blok procesora systemowego /INTEL 8080/, jednostki sterujące systemowymi urządzeniami wejścia/wyjścia, pamięć operacyjną typu RAM o pojemności 32 Kbajty, pamięć stałą typu PROM o pojemności 4 Kbajty, układy sterowania i sygnalizacji płyty czołowej, jednotkę sterującą pamięcią na dyskach elastycznych.

Do jednostki centralnej dołączony jest monitor alfanumeryczny, czytnik i perforator taśmy oraz stacja pamięci na dyskach elastycznych o pojemności ok. 500 Kbajtów.

MSWP w pełnej konfiguracji wyposażany jest obecnie w pamięć RAM o pojemności 60 KB, pamięć ROM o pojemności 4 KB, pamięć masową na dyskach elastycznych o pojemności ok. 1 MB, symulator pamięci stałych o pojemności 512 słów 64 bitowych, wkładki programatora pamięci EPROM typu INTEL 1702A, 2704, 2708, 2716, 2732 oraz PROM typu Tungsram TM 601, 621, 602, 622, 604, 624, emulator układowy do uruchamiania sprzętu i oprogramowania urządzeń z mikroprocesorem typu INTEL 8080, uniwersalny moduł testujący UMT-1 i lampę kasującą zapis w pamięciach EPROM za pomocą promieni ultrafioletowych.

Jako standardową konsolę operatorską systemu MSWP stosuje się monitor alfanumeryczny MERA 7953 z klawiaturą MERA 7946. Zamiennie może być zastosowana również drukarka z klawiaturą typu DZM 180 KSR, inny monitor alfanumeryczny lub inna drukarka z interfejsem V24. Do pracy z taśmami perforowanymi przewidziano stosowanie czytnika taśmy perforowanej typu CT 2100 oraz perforatora taśmy typu DT105S. Dodatkowym urządzeniem zewnętrznym może być drukarka DZM 180 /z interfejsem równoległym/, szczególnie przydatna do wyprowadzania listingów.

Uwzględniając zapotrzebowanie użytkowników opracowano nowe opcjonalne moduły MSWP: programator układów FPLA typu 82S100 firmy Signetics i odpowiedników radzieckich, monitor alfanumeryczny na bazie standardowego odbiornika telewizyjnego, układ sterujący pozwalający tworzyć system wielodostępny oraz nowe rozwiązanie pamięci operacyjnej pozwalające uzyskać pojemność do 256 Kbajtów.

Moduły te będą sukcesywnie wdrażane do produkcji, co wpłynie na podniesienie walorów użytkowych i rozszerzy zakres zastosowań systemu.

Dostępne obecnie oprogramowanie systemowe MSWP pozwala

- opracowywać oprogramowanie użytkowe dla urządzeń mikroprocesorowych, a w szczególności:
  - realizować edycję programów źródłowych,
  - translować programy z postaci źródłowej na postać wynikową /binarną/,
  - uruchamiać i sprawdzać programy wynikowe,
  - symulować i poprawiać mikroprogramy,
  - tworzyć języki ukierunkowane problemowo oparte na języku asemblera z wykorzystaniem makroinstrukcji,
  - emulować listy rozkazów mikroprocesorów innych niż INTEL 8080 za pomocą makroinstrukcji,
  - tworzyć i rozwijać własne oprogramowanie systemowe /operacyjne i testujące/.

Oprogramowanie systemowe MSWP w zależności od nośnika, na którym jest przechowywane dzieli się na trzy grupy:

1/ programy przechowywane w pamięci stałej systemu i wchodzące w skład oprogramowania firmowego,
2/ programy na taśmach perforowanych,
3/ programy na dyskach elastycznych.

Do programów grupy 1 należą następujące programy:

- MONITOR - podstawowy system operacyjny zawierający między innymi program inicjacji systemu, programy obsługi urządzeń wejścia/wyjścia oraz 11 dyrektyw do uruchamiania programów binarnych,

- DYSK - program sterujący pracą pamięci na dyskach elastycznych.

W grupie 2 znajdują się programy:

- EDYTOR 1 wer. 1.6 /REEDYT/ - edytor tekstów z buforem tekstu wielkości ograniczonej rozmiarem pamięci RAM,
- EDYTOR 2 wer. 1.0 - edytor tekstów zapisanych na taśmie perforowanej, wykorzystujący bufor tekstu wielkości równej liczbie znaków poprawianego wiersza tekstu,
- MAKROASEMBLER REM wer. 1.4 - kompilator programów pisanych w języku makroasemblera typu INTEL 8080,
- MAKROASEMBLER-EDYTOR REMAK wer. 1.0 - program będący połączeniem programu EDYTOR 1 wer. 1.6 i programu REM wer. 1.4 ze wspólnym buforem zawierającym translowany program źródłowy,
- MINI BASIC wer. 1.0 - interpreter języka BASIC działający w trybie konwersacyjnym z krokowym wykonywaniem kolejno wprowadzanych instrukcji; zawiera arytmetykę stałoprzecinkową,
- BASIC wer. 1.0 - interpreter języka BASIC działający w trybie konwersacyjnym z buforem 6 KB przechowującym instrukcje i dane programu; zawiera arytmetykę zmiennoprzecinkową,
- EMULATOR 8080 wer. 3.0 - oprogramowanie modułu emulatora układowego dla systemów z mikroprocesorem typu INTEL 8080,
- SYMROM wer. 2.0 - oprogramowanie symulatora pamięci ROM o pojemności 512 x 64 bitów dla systemów z mikroprocesorem typu INTEL 8080,
- TEST RAM - zestaw testów pamięci RAM,
- TEST CT/DT - zestaw testów czytnika CT 2100 i perforatora DT 105 S,
- MONITOR TESTERA UMT-1 wer. 1.1 - program realizujący wczytane z taśmy perforowanej testy pakietów na testerze UMT-1,
- JĘZYK TESTERA UMT-1 wer. 1.1 - translator języka testera UMT-1 na kod sterujący.

Grupa 3 obejmuje dwa dyskowe systemy operacyjne i programy działające pod nadzorem tych systemów na dyskach elastycznych.

Dyskowy system operacyjny OS-I wer. 1.4 /odpowiednik systemu CP/M firmy Digital Research/

- podstawowy zestaw programów systemu MSWP OS-I wer. 1.4:
  - kontekstowy edytor tekstów /odpowiednik ED/,
  - translator asemblera typu INTEL 8080 /odpowiednik ASM/,
  - program obsługi urządzeń /odpowiednik PIP/,
  - debugger /odpowiednik DDT/,
  - program zmieniający format zbioru /odpowiednik LOAD/,
  - program statusowo-statystyczny /odpowiednik STAT/,
  - program wyprowadzający zawartość zbioru dyskowego /odpowiednik DUMP/,
  - program interpretujący makrodyrektywy systemowe /odpowiednik SUBMIT/,
- dodatkowe programy systemu MSWP OS-I wer. 1.4:
  - translator makroassemblera typu INTEL 8080 /odpowiednik MAC/,
  - relokowalny translator makroasemblera MAKREL,
  - symboliczny debugger /odpowiednik SID/,
  - program formatujący tekst wg reguł edytorskich /odpowiednik TEX/,

- BASIC wer. 2.1 - szybki interpreter języka BASIC zgodny ze standardem ANSI z szerokim zestawem funkcji, operacji na zbiorach i ciągach znaków,
- BASIC wer. 2.2 - półkompilator języka BASIC przeznaczony do obliczeń numerycznych naukowych i technicznych; pozwala opracowywać znacznie większe programy niż w wypadku czystego interpretera,
- BASIC wer. 2.3 - półkompilator języka BASIC przeznaczony do zastosowań administracyjnych wyposażony w arytmetykę dziesiętną i bogaty zestaw funkcji operujących na zmiennych znakowych; pozwala realizować duże programy,
- MONITOR TESTERA UMT-1 wer. 2.1 - program sterujący wykonywaniem testów pakietów ze zmontowanymi elementami na testerze UMT-1,
- JĘZYK TESTERA UMT-1 wer. 2.1 - program tłumaczący test pakietu napisany w języku symbolicznym na kod sterujący pracą testera UMT-1,
- TEST FD - test pamięci na dyskach elastycznych /może również służyć jako test dysketki/,
- ODZYSK - program odzyskujący z dokładnością 1 kB uszkodzone zbiory,
- RETRI - program odzyskujący z dokładnością 1 rekordu uszkodzone zbiory,
- UZDA - program uzdatniający uszkodzoną dyskietkę,
- SPRAWDZ - program sprawdzający kopię taśmową zbioru dyskowego
- dyskowe wersje programów testujących jednostkę centralną, pamięć i urządzenia wejścia/wyjścia systemu,

Dyskowy system operacyjny OS-II wer. 2.0 /odpowiednik systemu ISIS-II firmy Intel/

- podstawowy zestaw programów systemu OS-II wer. 2.0:
  - program formatowania dysketek i kopiowania zbiorów /odpowiednik FORMAT/,
  - program interpretujący makrodyrektywy systemowe /odpowiednik SUBMIT/,
  - program wyświetlający zawartość katalogu /odpowiednik DIR/,
  - program kopiujący zbiory /odpowiednik COPY/,
  - program usuwający zbiory /odpowiednik DELETE/,
  - program zmieniający atrybuty /odpowiednik ATTRIB/,
  - programy zmieniające formaty zbiorów /odpowiedniki BINOBJ, HEXOBJ, OBJHEX/,
  - program zarządzający bibliotekami /odpowiednik LIB/,
  - program przekształcający relokowalne zbiory wynikowe na zbiory wynikowe o adresach absolutnych /odpowiednik LOCATE/,
  - program łączący moduły wynikowe /odpowiednik LINK/,
- dodatkowe programy systemu OS-II wer. 2.0:
  - edytor tekstów /odpowiednik EDIT/,
  - translator makroasemblera relokowalnego typu INTEL 8080 /odpowiednik ASM-80/,
  - kompilator języka PL/M /odpowiednik PL/M-80/,
  - kompilator języka FORTRAN /odpowiednik FORT 80/,
  - biblioteka arytmetyki zmiennoprzecinkowej,
  - program konwersji zbiorów z OS-II do OS-I.

Projektant systemu mikroprocesorowego może korzystać z MSWP, zarówno przy niezależnym opracowywaniu sprzętu i oprogramowania, jak i przy łącznym uruchamianiu sprzętu wraz z oprogramowaniem.

Przy opracowywaniu programów użytkowych projektant może posługiwać się środkami wspomagania MSWP począwszy od etapu kodowania programu w języku źródłowym poprzez etap uruchamiania programu.

Programy dla mikroprocesorów o stałej liście rozkazów /typu INTEL 8080/ mogą być całkowicie opracowywane za pomocą podanych wyżej programowych środków wspomagania tworzących oprogramowanie

systemowe MSWP. Projektant korzysta z potrzebnego programu po wprowadzeniu go do pamięci systemowej z taśmy perforowanej lub dysketki.

Wprowadzanie, poprawianie i wyprowadzanie programów w postaci źródłowej wykonuje się za pomocą najwygodniejszej z kilku dostępnych wersji edytora. Użytkownik MSWP ma do wyboru kilka programów assemblujących oraz translatory języków wyższego rzędu. Postać wynikowa programu assemblera może być uzyskana w rezultacie działania wybranej wersji programu translującego.

Do uruchamiania programów wynikowych można wykorzystywać dyrektywy MONITORA, a gdy system wyposażony jest w pamięć dyskową - znacznie efektywniejszy program uruchomieniowy /DDT/ oraz jego rozszerzenie - program symbolicznego debuggera /DES/. Oba te programy pracują pod nadzorem systemu operacyjnego OS-I.

Uruchamianie sprzętu realizowanego na układach mikroprocesorowych ze stałą listą rozkazów, łącznie z oprogramowaniem przebiega najbardziej efektywnie przy wykorzystaniu emulatora układowego sterowanego programem EMULATOR. Dyrektywy EMULATORA pozwalają uruchamiać i testować programy użytkownika za pomocą procesora emulacyjnego /identycznego z procesorem użytkownika/ i układów emulacji układowej początkowo nawet bez dołączonych układów użytkownika. W zależności od typu mikroprocesora użytkownika stosowana jest odpowiednia wersja programu EMULATOR łącznie z odpowiednim emulatorem układowym.

Programy użytkowe po ich uruchomieniu mogą być wpisane do pamięci stałych systemu użytkownika za pomocą odpowiednich wkładek programatora, a następnie badane dalej łącznie ze sprzętem użytkownika.

Przy uruchamianiu systemów z mikroprocesorami mikroprogramowanymi serii INTEL 3000 i jej odpowiedników w pierwszej kolejności sprawdzana jest poprawność przygotowanych mikroprogramów za pomocą symulatora pamięci stałych /ROM/. Użytkownik korzysta w tym wypadku z programu obsługi symulatora SYMROM, za pomocą którego może wczytywać i wyprowadzić mikroprogramy w standardzie języka CROMIS oraz zmieniać zawartość symulowanej pamięci ROM.

Dołączenie do MSWP modułu testera UMT-1 czyni z niego programowany tester płytek drukowanych i pakietów z układami cyfrowymi o dużej użyteczności dla małoseryjnych producentów sprzętu cyfrowego.

MSWP może być wykonywany w zróżnicowanych konfiguracjach przeznaczonych nie tylko do celów wspomagania projektowania i uruchamiania urządzeń mikroprocesorowych. Wśród wielu innych jako przykłady mogą być podane następujące zastosowania systemu:

- sterowanie i przetwarzanie informacji w systemach kontrolno-pomiarowych,
- zbieranie i przetwarzanie danych w systemach handlowych, bibliotecznych, medycznych, itp.,
- tworzenie i wykorzystanie systemów konwersacyjnych, np. testy psychologiczne, systemy edukacyjne w szkolnictwie,
- tworzenie laboratoriów dydaktycznych, np. do nauki projektowania sprzętu i oprogramowania urządzeń mikroprocesorowych, programowania w językach wyższego rzędu, itp.

Do tych zastosowań szczególnie przydatna wydaje się wdrażana obecnie do produkcji - wielodostępna wersja systemu MSWP-M, która łącznie z wielodostępnym systemem operacyjnym OS-I/M umożliwia tworzenie konfiguracji wieloterminalowych /4+8/ pozwalających realizować pracę wieloprogramową przy każdym terminalu.

TECHNIKI KOMPUTEROWE 5-6/84

mgr inż. Lech NAUMOWSKI
Instytut Maszyn Matematycznych

Charakterystyka techniczna emulatora układowego EM8080 i zakres jego zastosowań

Wstęp

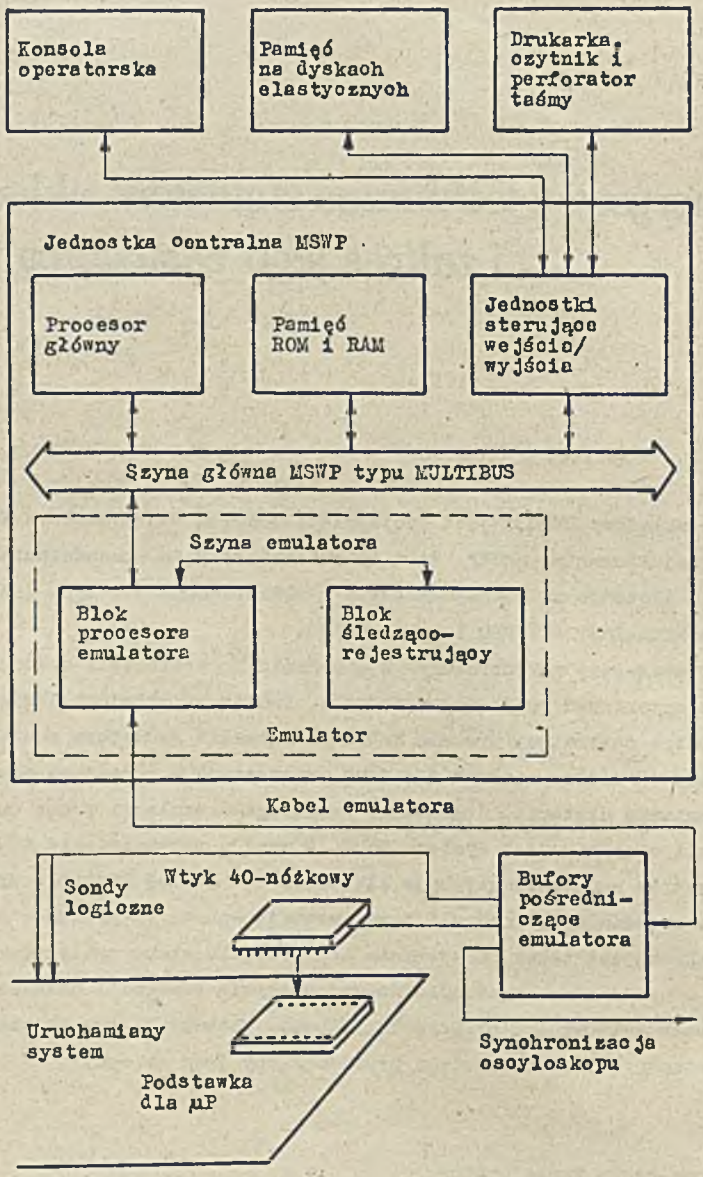
Emulator układowy EM8080 jest opcjonalnym modułem funkcjonalnym Mikroprocesorowego Systemu Wspomagania Projektowania - MSWP. Jest on przeznaczony do uruchamiania i testowania sprzętu oraz oprogramowania systemów mikrokomputerowych z mikroprocesorami typu Intel 8080 i ich odpowiednikami takimi, jak mikroukłady MCY7880 lub KP580UK80.

Emulator zastępuje mikroprocesor w uruchamianym systemie i wykonuje wszystkie funkcje mikroprocesora bez ograniczeń prędkości działania. Pakiety z układami logicznymi emulatora są umieszczone w jednostce centralnej systemu MSWP. Do łączenia emulatora z uruchamianym systemem służy kabel z wtykiem 40-nóżkowym. Wtyk ten jest umieszczany w podstawce przeznaczonej dla mikroprocesora w uruchamianym systemie. Sterowanie przebiegiem emulacji przez operatora odbywa się za pośrednictwem konsoli operatorskiej systemu MSWP. W wyniku przeniesienia mikroprocesora z uruchamianego systemu do wnętrza emulatora uzyskuje się możliwość pełnej kontroli wykonywanego w czasie rzeczywistym oprogramowania oraz kontroli i rejestracji wybranych procesów zachodzących w uruchamianym systemie. Możliwe jest także obrazowanie i zmienianie stanu pamięci, rejestrów mikroprocesora i innych układów uruchamianego systemu oraz wypożyczanie pamięci i urządzeń wejścia/wyjścia systemu MSWP dla potrzeb testowanego urządzenia. Schemat blokowy MSWP z dołączonym emulatorem EM8080 i uruchamianym systemem mikroprocesorowym przedstawiony jest na rys.1.

Podstawowe cechy funkcjonalne

Podczas uruchamiania sprzętu i oprogramowania dołączonego systemu mikroprocesorowego emulator EM8080 umożliwia wykonanie następujących czynności:

- sprawdzanie poprawności działania zegara w uruchamianym systemie,
- zastąpienie całości lub części pamięci (do 44 Kbajtów w blokach po 4 K) i urządzeń wejścia/wyjścia uruchamianego systemu zasobami MSWP oraz blokadę nieistniejących pamięci i urządzeń zewnętrznych,
- ładowanie programów i danych do pamięci uruchamianego systemu z dysku lub z taśmy perforowanej,
- zapamiętywanie uruchomionych i poprawionych programów na dyskach lub na taśmie perforowanej,



Rys. 1. System MSWP, emulator i uruchamiany system mikrokomputerowy

- przenoszenie zawartości pamięci,
- odczyt i listowanie zawartości wybranych obszarów lub komórek pamięci i portów wejściowych,
- wpisywanie żądanych danych do wybranych obszarów lub komórek pamięci i portów wyjściowych,
- rozpoczęcie wykonywania uruchamianego programu od zadanego miejsca i z zadanymi wartościami rejestrów mikroprocesora,
- zatrzymywanie wykonywania uruchamianego programu w następujących sytuacjach:
  - po wykonaniu jednego cyklu rozkazowego (praca krokowa),
  - po wykonaniu jednej z dwóch wyszczególnionych operacji odczytu lub zapisu pamięci i portów we/wy (punkty zatrzymania),
  - po odwołaniu się programu do nieistniejącej pamięci lub portu we/wy,
  - po przekroczeniu limitu czasu pozostawania procesora w stanie nieaktywnym,
  - po pojawieniu się wybranego sygnału w uruchamianym sprzęcie,
  - bezwarunkowe zatrzymanie na żądanie operatora,

- rejestrację i wyświetlenie zawartości rejestrów mikroprocesora w momencie zatrzymania emulacji,
- rejestrację stanów wejść i wyjść sterujących mikroprocesora w chwili zatrzymania emulacji,
- pomiar czasu trwania emulacji (licznik cykli zegarowych),
- rejestracja stanu linii adresowych, danych, słowa stanu mikroprocesora oraz wybranych sygnałów w uruchamianym urządzeniu podczas ostatnich 32 cykli maszynowych mikroprocesora przed zatrzymaniem emulacji (pamięć śladu),
- wytwarzanie impulsów do synchronizacji oscyloskopu użytego do badań sprzętu w czasie emulacji.

Emulator nie ogranicza szybkości działania uruchamianego urządzenia, w którym zegar może mieć częstotliwość do 2 MHz właściwą dla użytego w emulatorze procesora MCY7880 lub KP580UK80. Jedynie podczas pracy z wykorzystaniem zasobów MSWP, czas dostępu do pamięci i urządzeń zewnętrznych MSWP emulujących sprzęt uruchamianego systemu może opóźnić wykonywanie programów w najgorszym razie o około 60 %.

Konstrukcja

Emulator układowy EM8080 składa się z czterech pakietów z drukiem dwustronnym o wymiarach 160 x 277 mm, zawierających łącznie około 200 układów scalonych, oraz z kabla emulatora o długości około 1,5 m z buforami emulatora, zakończonego wtykiem 40-nóżkowym, wstawianym w miejsce mikroprocesora w uruchamianym systemie. Pakiety z układami logicznymi emulatora umieszcza się w wyróżnionych miejscach kasety jednostki centralnej MSWP, których łączówki wyposażone są w połączenia tworzące szynę wewnętrzną emulatora. Do łączówek tych jest również doprowadzona szyna główna systemu MSWP. Kabel emulatora dołączony do pakietów tworzących blok procesora emulatora przechodzi przez otwór w tylnej ścianie obudowy JC MSWP i umożliwia ustawienie dołączonego uruchamianego systemu w wygodnym miejscu w pobliżu jednostki centralnej MSWP.

Struktura logiczna emulatora

Emulator EM8080 składa się z trzech odrębnych bloków funkcjonalnych spełniających zasadniczo różne zadania. Pierwszy blok zawiera mikroprocesor MCY7880 lub KP580UK80 i układy sterowania umożliwiające rozpoczęcie i zakończenie wykonywania programu w uruchamianym systemie w kontrolowany sposób, drugi blok jest blokiem śledząco-rejestrującym, zawierającym pamięć programu sterującego oraz zespół układów kontrolujących i rejestrujących przebieg emulacji, a trzeci blok tworzy kabel z buforami i wtykiem 40-nóżkowym umożliwiającym dołączenie emulatora do uruchamianego systemu.

Bloki procesora emulatora i śledząco-rejestrujący składają się z następujących zespołów:

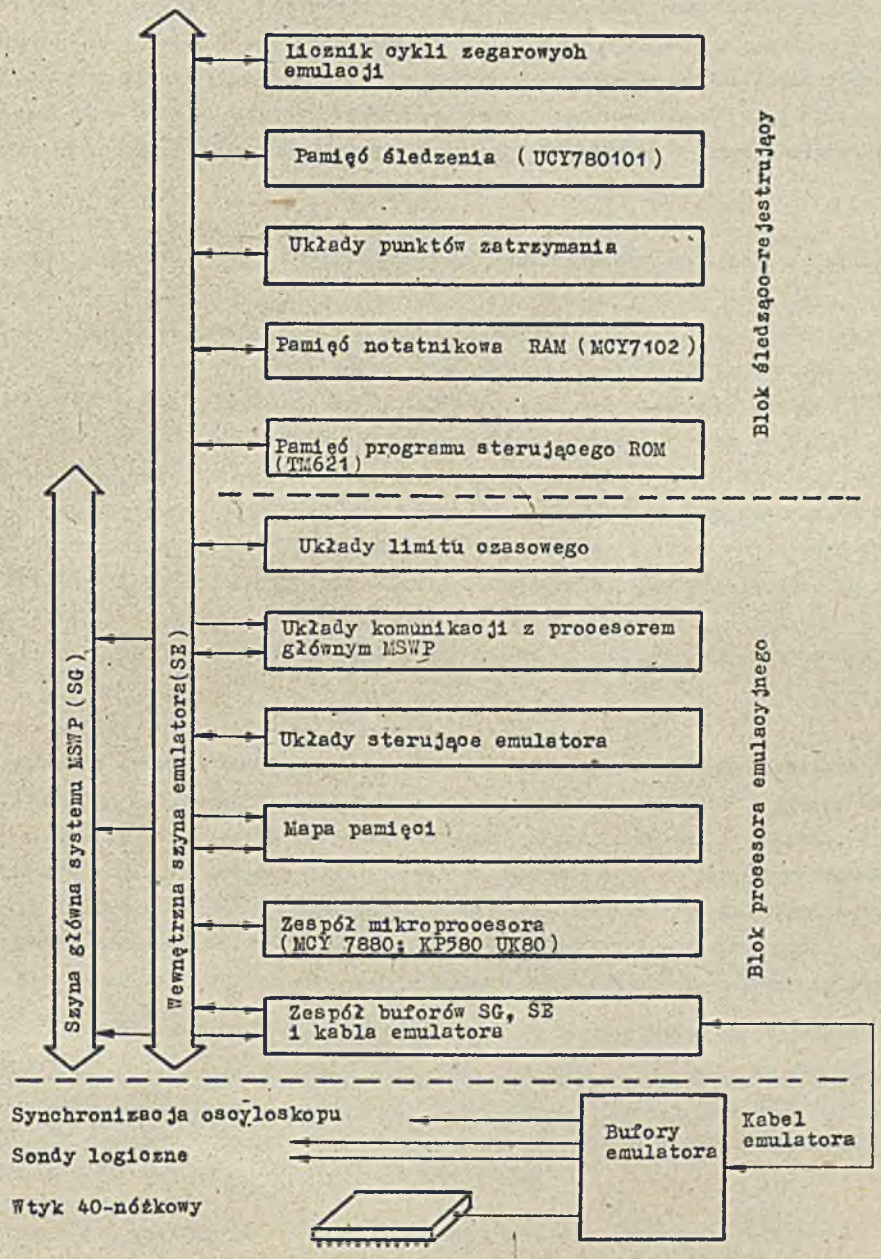
- zespół mikroprocesora zawierający CPU, zegar wewnętrzny i odbiorniki sygnałów zegarowych z testowanego systemu z układem kontroli, oraz logikę rozkodowującą sygnały sterujące mikroprocesora dla potrzeb emulatora.
- zespół buforów łączących mikroprocesor z szyną wewnętrzną emulatora, z szyną główną jednostki centralnej MSWP i za pośrednictwem kabla emulatora z uruchamianym systemem,
- układy mapy pamięci umożliwiające zastąpienie pamięci i portów we/wy uruchamianego systemu przez zasoby JC MSWP i blokadę nieistniejących pamięci i urządzeń zewnętrznych,
- układy sterowania emulatora umożliwiające w sposób sprzętowo-programowy załadowanie rejestrów mikroprocesora i przejście emulatora w stan emulacji oraz zakończenie emulacji z odczytaniem stanu rejestrów mikroprocesora,
- układy komunikacji z procesorem głównym JC MSWP odbierające rozkazy dla emulatora, realizujące wymianę informacji między emulatorem i JC MSWP oraz umożliwiające dostęp do zasobów MSWP emulujących pamięć i porty we/wy uruchamianego systemu,
- zespół pamięci stałej ROM i notatnikowej RAM dla programu sterującego emulatora,
- układy punktów zatrzymania umożliwiające zatrzymanie emulacji po odwołaniu się uruchamianego programu do jednego z dwóch wyróżnionych adresów pamięci lub portów we/wy oraz generujące impulsy synchronizacji oscyloskopu,
- pamięć śledzenia pamiętająca adresy, dane, słowo sterujące mikroprocesora i stany dwóch wybranych sygnałów z testowanego systemu w czasie ostatnich 32 cykli maszynowych emulacji,
- licznik cykli zegarowych emulacji,
- układy limitu czasowego.

Wymienione układy emulatora łączy szyna wewnętrzna emulatora zawierająca przewody adresów, danych, i sygnałów sterujących.

Schemat blokowy emulatora, na którym znajdują się opisywane zespoły pokazany jest na rys.2.

Zasada działania

Emulator działa pod kontrolą procesora głównego jednostki centralnej MSWP wykonującego nadrzędny program sterujący nazwany EMULATOR 8080. Procesor główny komunikuje się z operatorem za pomocą konsoli operatorskiej systemu MSWP. Odbiera on polecenia operatora i przetwarza je na ciągi prostych operacji wykonywanych przez sprzęt emulatora. Program EMULATOR 8080 zajmuje wraz z buforem danych i polem kontrolnym, za pomocą którego przekazywane są informacje do i z emulatora, 8k bajtów pamięci o dostępie swobodnym systemu MSWP. Pozostała pamięć, z wyjątkiem obszarów zajętych przez systemy operacyjne MSWP, umożliwia emulację dowolnego fragmentu pamięci uruchamianego urządzenia. Emulator działa w dwóch podstawowych stanach: w stanie sterowania i emulacji. W stanie sterowania emulator odbiera i wykonuje elementarne rozkazy wysyłane przez procesor główny umożliwiające



Rys. 2. Schemat blokowy emulatora EM8080

zaprogramowanie układów śledząco-rejestrujących, zgodnie z informacją zawartą w polu kontrolnym, odczyt lub zmianę zawartości pamięci oraz odczyt i zapis portów we/wy, zgodnie z parametrami podanymi w polu kontrolnym, załadowanie rejestrów mikroprocesora wartościami umieszczonymi w polu kontrolnym i rozpoczęcie wykonywania uruchamianego programu, zapamiętanie zawartości rejestrów procesora ustalonych w chwili przerwania emulacji oraz przesłanie do pola kontrolnego programu EMULATOR 8080 informacji o przebiegu emulacji zebranej przez układy śledząco-rejestrujące. W stanie emulacji emulator wykonuje program uruchamianego systemu. Przebieg emulacji jest rejestrowany, a wykonywanie uruchamianego programu zostaje zatrzymane w momencie spełnienia żądanych warunków.

Współpraca z jednostką centralną systemu MSWP

Emulator EM8080 jest dołączony do szyny głównej systemu MSWP na prawach, zarówno modułu biernego, jak i aktywnego. Emulator jest modułem biernym, gdy znajduje się w stanie sterowania i procesor główny systemu MSWP wykonujący nadrzędny program sterujący EMULATOR 8080 odczytuje rejestr stanu emulatora albo zapisuje jeden z czterech rejestrów rozkazowych emulatora dołączonych do szyny głównej jako porty we/wy. Jako moduł aktywny przejmujący kontrolę nad szyną główną systemu MSWP, emulator występuje w czasie wykonywania rozkazów otrzymanych od procesora głównego przy odwoływaniu się do pola kontrolnego oraz w czasie wykonywania uruchamianego programu w stanie emulacji, gdy zachodzi konieczność odwoływania się do pamięci i portów we/wy wypożyczonych z zasobów JC MSWP.

Oprogramowanie

Program obsługi emulatora EM8080 nazwany EMULATOR 8080 dostępny jest w dwóch wersjach: taśmowej i dyskowej. Wersja taśmowa programu EMULATOR 8080 wczytywana jest do pamięci systemu MSWP z taśmy perforowanej pod kontrolą systemu operacyjnego dla taśmy perforowanej - MONITOR. Wersja dyskowa programu obsługi emulatora wczytywana jest do pamięci MSWP z dysku elastycznego pod kontrolą dyskowego systemu operacyjnego OS-I. Obie wersje programu EMULATOR 8080 różnią się jedynie nośnikiem, na którym są przechowywane i nośnikiem, na którym mogą być przechowywane uruchamiane programy. Program obsługi emulatora zgłasza się wydrukiem:

EMULATOR 8080 WER.3.0.(2.0.)

na konsoli operatorskiej. Następnie na początku nowej linii pojawia się znak * oznaczający gotowość przyjęcia dyrektywy od operatora. Uruchamianie oprogramowania i sprzętu dokonywane jest za pomocą następujących dyrektyw:

- CLOCK
  Określenie źródła zegara dla procesora emulatora. Może to być zegar wewnętrzny lub z uruchamianego systemu.
- XFORM MEMORY
  XFORM IO
  Dyrektywa rozplanowania pamięci i portów we/wy operująca 16 blokami pamięci o pojemności 4 Kbajtów i 16 blokami po 16 adresów portów. Dyrektywa umożliwia zadeklarowanie, które bloki pamięci i adresów portów istnieją fizycznie w uruchamianym systemie, które będą wypożyczone od MSWP i które mają być blokowane,

- TIMEOUT

  Dyrektywa umożliwiająca dołączenie lub odłączenie układu limitu czasowego.

- PROBE

  Dyrektywa sterująca funkcją przerywania emulacji od sygnału z sondy logicznej.

- LOAD

  Ładowanie programów i danych z taśmy perforowanej albo dysku do pamięci uruchamianego urządzenia.

- SAVE

  Zapamiętywanie na taśmie papierowej albo na dysku zawartości pamięci uruchamianego systemu.

- MOVE MEMORY

  FILL MEMORY

  Przenoszenie zawartości pamięci i wypełnianie obszaru pamięci stałą.

- DISPLAY MEMORY

  DISPLAY INPUT

  DISPLAY REGISTER

  DISPLAY CYCLES

  DISPLAY CC

  Wyświetlanie zawartości odpowiednio: pamięci, portów wejściowych, rejestrów mikroprocesora, pamięci śladu, licznika cykli zegarowych emulacji i licznika cykli zapisanych w pamięci śladu.

- CHANGE MEMORY

  CHANGE OUTPUT

  CHANGE REGISTER

  Wpisywanie żądanych wartości odpowiednio do: pamięci, portów wyjściowych i rejestrów mikroprocesora.

- STEP

  Dyrektywa pracy krokowej umożliwiająca zadeklarowanie adresu początkowego, liczby kroków i żądania wyświetlenia informacji.

- GO

  Dyrektywa wykonywania uruchamianego programu w sposób ciągły od zadanego adresu początkowego do napotkania jednego z dwóch punktów zatrzymania. Dyrektywa ta umożliwia także zadeklarowanie liczby przejść przez zaprogramowane punkty zatrzymania, zadeklarowanie wyświetlania oraz generacji impulsów synchronizacji oscyloskopu użytego do badań sprzętu podczas emulacji.

Program EMULATOR 8080 umożliwia także przerwanie każdej podjętej akcji, w tym także wykonywania uruchamianego programu za pomocą przycisku przerwania 3 znajdującego się na płycie pulpitu jednostki centralnej MSWP. Program EMULATOR 8080 wyświetla ponadto komunikaty o błędnym działaniu zegara w uruchamianym systemie oraz sygnalizuje przyczynę znalezienia się procesora emulatora w stanie nieaktywnym podczas emulacji.

Zakres zastosowań

Wśród sprzętowych środków wspomagania emulatory układowe wyróżniają się uniwersalnością i najbogatszym zestawem możliwości uruchamiania systemów mikroprocesorowych. Emulatory układowe szczególnie nadają się do uruchamiania sprzętu modeli i prototypów systemów mikroprocesorowych oraz do łączenia sprzętu z oprogramowaniem przygotowanym za pomocą programowych środków wspomagania projektowania. Emulatory znajdują także szerokie zastosowanie przy testowaniu prototypów systemów mikroprocesorowych w rzeczywistych warunkach pracy. Z tych względów emulatory są niezastąpionym

narzędziem uruchomieniowym dla konstruktorów systemów mikroprocesorowych. Emulatory układowe są też stosowane w zakładach produkujących systemy mikrokomputerowe do ich uruchamiania i testowania. Mogą być bardzo pomocne przy lokalizacji uszkodzeń i naprawach mikrokomputerów.

Nowe generacje emulatorów w systemie MSWP

Obecnie realizowany jest program budowy rodziny emulatorów w systemie MSWP zawierający następujące urządzenia:

- emulator dla mikroprocesora 16-bitowego typu Intel 8086 i jego odpowiedników,
- zestaw emulatorów dla mikroprocesorów 8-bitowych typu Intel 8080, 8085 i Z80 i ich odpowiedników wykorzystujących blok śledząco-rejestrujący emulatora dla mikroprocesora Intel 8086,
- zestaw emulatorów dla mikrokomputerów jednoelementowych typu Intel 8048, 8049 i 8041 oraz ich odpowiedników.

Projektowane emulatory będą wyposażone w pamięci śladu o pojemności zwiększonej do 1 K cykli maszynowych, z możliwością warunkowego zapamiętywania wybranych sekwencji rozkazowych. Wprowadzone zostaną nowe warunki przerwania emulacji po przepełnieniu liczników cykli i pamięci śladu oraz po wykonaniu wybranych rozkazów. Zwiększona zostanie liczba sond logicznych wyprowadzonych z pamięci śladu, a w dalszej kolejności planuje się wyposażenie emulatorów w analizatory stanów logicznych. Oprogramowanie sterujące nowej rodziny emulatorów umożliwi symboliczne zadawanie parametrów w dyrektywach uruchomieniowych oraz tworzenie ciągów dyrektyw uruchomieniowych wykonywanych automatycznie przy długotrwałym testowaniu sprzętu. Planowane jest także umożliwienie jednoczesnej pracy dwóch i więcej emulatorów w systemie MSWP przy uruchamianiu systemów wielomikroprocesorowych.

mgr inż. Krzysztof DZIK
Instytut Maszyn Matematycznych

Charakterystyka techniczna i zastosowanie symulatora pamięci stałych SYM-1

Wprowadzenie

Symulator pamięci stałych SYM-1 jest modułem opcjonalnym Mikroprocesorowego Systemu Wspomagania Projektowania MSWP przeznaczonym do symulacji pewnych typów pamięci stałych. Symulator SYM-1 może być zastosowany szczególnie efektywnie do uruchamiania systemów mikroprogramowanych.

Symulator SYM-1 ma budowę modułową; pozwala symulować bloki pamięci stałej o maksymalnej pojemności 512 słów o długości słowa maksymalnie 64 bity. Moduł symulatora SYM-1 składa się z czterech jednakowych segmentów, przy czym każdy segment obejmuje:

- pakiet SYM-11
- kasetę buforów BP
- zespół kabli 4 x 16 lub zespół kabli 2 x 24.

Symulator współpracuje z systemem MSWP za pośrednictwem szyny głównej (typu MULTIBUS), natomiast z badanym układem łączy się za pomocą zespołu kabli zakończonych wtykami wkładanymi w podstawki symulowanych pamięci stałych.

Podstawowe dane techniczne

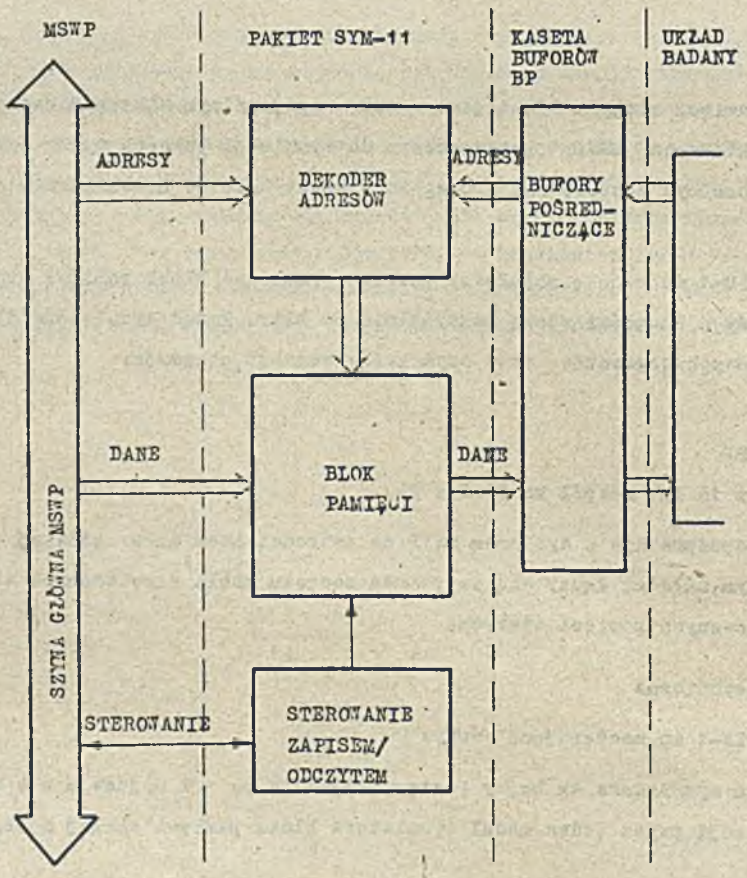
Symulator SYM-1 ma następujące cechy:

- pojemność modułu symulatora 4k bajty (cztery segmenty po 1 k bajtów = 4 k bajty),
- możliwość symulacji przez jeden moduł symulatora bloku pamięci stałej o organizacji 512 słów x 64 bity,
- czas dostępu do pamięci symulatora od strony układu badanego 150 ns,
- możliwość symulacji bloków pamięci 512 słów x 16 bitów lub 512 słów x 32 bity lub 512 słów x 48 bitów lub 512 słów x 64 bity w zależności od liczby użytych segmentów (1, 2, 3 lub 4),
- elastyczność konstrukcyjna umożliwiająca powiększenie długości słowa do maksymalnie 128 bitów przez dołączenie dodatkowych segmentów (liczba modułów jest ograniczona miejscami w kasecie jednostki centralnej systemu MSWP),

- możliwość symulacji pamięci bipolarnych firmy Intel typu 3602/3622, 3604/3624 lub ich odpowiedników mających stosowne wyprowadzenia,
- korzystanie z symulatora upraszcza program sterujący SYM ROM pracujący pod systemem operacyjnym OS-1 (program SYM ROM ma dwie wersje - taśmową VER. 2.0 i na dysku elastycznym VER.2.1).

Struktura logiczna

W skład modułu symulatora wchodzą cztery jednakowe segmenty, z których każdy zawiera pakiet SYM-11, kasetę buforów BP oraz jeden z zespołów kabli 4 x 16 lub 2 x 24. Na rys. 1 przedstawiono schemat blokowy segmentu symulatora.



Rys. 1. Schemat blokowy segmentu symulatora

Pakiet SYM-11 jest wykonany w postaci płytki drukowanej o standardzie MSWP czyli o wymiarach 277,6 x 159,5 mm zaopatrzonej w dwa złącza do połączenia z systemem MSWP i dwa złącza znajdujące się po przeciwległej stronie, przeznaczone do połączenia z kasetą buforów BP.

Jak wynika z rys. 1 pakiet SYM-11 składa się z bloku pamięci, układu wybierania i dekodera adresu oraz z układu sterowania zapisem-odczytem.

Kaseta buforów BP zawiera układy pośredniczące w transmisji sygnałów między pakietem SYM-11 i badanym układem. Układy te zmontowane na płytce są zamknięte w kasecie metalowej. Od strony pakietu SYM-11 kaseta wyposażona jest w kabel zakończony złączami do nasadzenia na pakiet, a od strony układu badanego w złącze szufladowe służące do dołączenia odpowiedniego zespołu kabli. Kaseta pośrednicząca zawiera układy wzmacniające sygnały adresowe i sygnały danych.

Zespół kabli służy do połączenia kasety buforów BP z badanym układem. Długość kabla nie przekracza 30 cm. Jest on zakończony z jednej strony złączem szufladowym, a z drugiej strony wtykami do wkładania w podstawki pamięci w badanym układzie. Zespół kabli 4 x 16 przeznaczony dla pamięci typu Intel 3602/3622 jest zakończony czterema wtykami 16-stykowymi, natomiast zespół kabli 2 x 24 przeznaczony dla pamięci typu Intel 3604/3624 ma dwa wtyki 24-stykowe.

Zasada działania

Podstawowy element segmentu symulatora pakiet SYM-1 zawierający blok pamięci współpracuje z systemem MSWP, pamięć ta jest traktowana jako fragment pamięci systemowej. Mogą być na niej zatem wykonywane takie same operacje jak na pamięci o dostępie swobodnym (RAM). Pamięć zawarta na pakiecie SYM-11 jest widziana od strony systemu MSWP jako blok pamięci o organizacji 1024 x 8, natomiast od strony układu badanego jako blok o organizacji 512 x 16. Na pamięci symulatora można zatem działać, jak na pamięci systemowej, używając wszystkich dostępnych środków i programów stojących do dyspozycji w systemie MSWP, a zwłaszcza programu SYM ROM.

Ze strony badanego układu możliwy jest tylko odczyt, bowiem pamięć zawarta na pakiecie SYM-11 spełnia rolę pamięci stałej dla tego układu. Jednoczesna współpraca z systemem MSWP i badanym układem jest niemożliwa. Priorytet ma system MSWP, który po odwołaniu się do symulatora blokuje go dla badanego układu.

Adres modułu symulatora pamięci stałej jest wybierany czterema najstarszymi bitami adresowymi i ustala się go za pomocą pola zwor e, f, g, h. Grupy 256-bajtowe w ramach modułu są wybierane środkowymi adresami A8, A9, A10 i A11 i ustalane za pomocą pola zwor a, b, c, d.

Adres bajtu w grupie 256-bajtowej jest wybierany ośmioma najmłodszymi bitami adresowymi A0-A7.

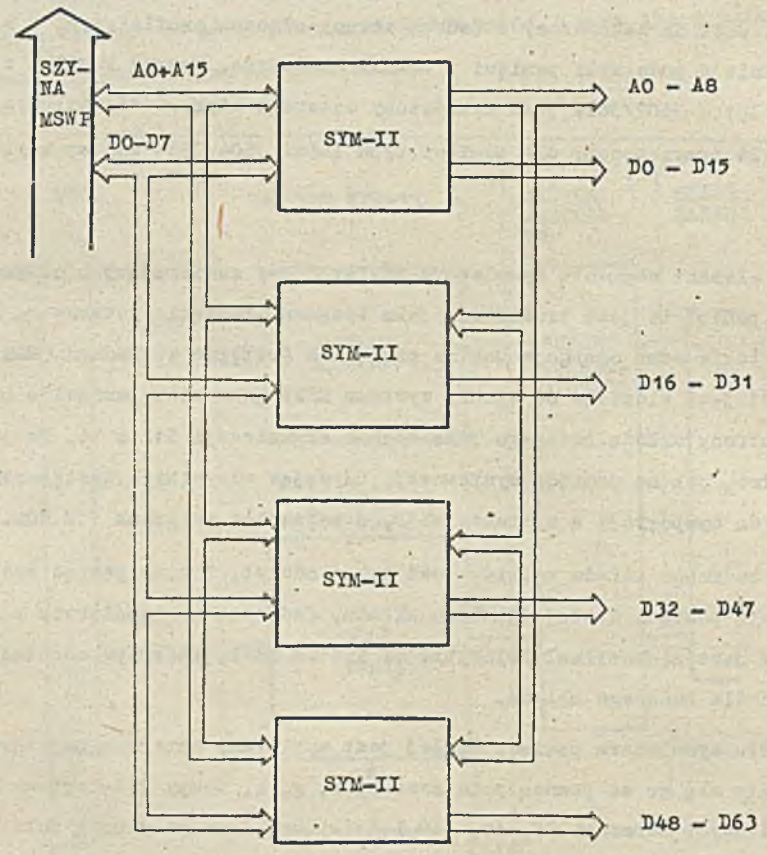
Współpraca z IC MSWP

Symulator współpracuje z systemem MSWP za pomocą szyny głównej, przy czym wykorzystywane są następujące sygnały:

- adresowe - A0 ÷ A15,
- danych - D0 ÷ D7,
- sterowania zapisem - MEM W, MEM R odczytem z pamięci
- blokowania pamięci - INHRAM
- potwierdzenia - XACK

Znaczenie używanych sygnałów jest zrozumiałe, wyjaśnienia wymaga jedynie sygnał XACK, który potwierdza wykonanie rozkazu. Ze względu na krótki czas dostępu do pamięci symulatora jest generowany po przyjęciu adresu i sygnału MEM W lub MEM R. Sygnał INH RAM jest wysyłany z pakietu SYM-11 po jego wybraniu i jest wykorzystywany w systemie MSWP do blokowania fragmentów pamięci o tych samych adresach.

Na rys. 2 pokazano połączenie segmentów symulatora w jego moduł i dołączenie go do szyny głównej MSWP. Od strony układu badanego jest utworzone słowo o długości 64 bity.



Rys.2. Połączenie segmentów symulatora w moduł

Oprogramowanie symulatora SYM-1

Program SYM ROM wchodzą w skład oprogramowania systemowego MSWP pracującego pod OS-1 daje możliwość wykonywania wszelkich niezbędnych operacji w symulatorze, tj.:

- zapisywania informacji,
- wyświetlania zawartości,
- listowania zawartości,
- modyfikacji zawartości,
- wyprowadzania zawartości na taśmę papierową.

Symulator stanowiąc pamięć od strony systemu MSWP może być traktowany jako fragment pamięci systemowej z wszystkimi tego konsekwencjami i może być zwłaszcza jako pamięć systemowa testowany.

Program SYM ROM działa konwersacyjnie umożliwiając zapis, odczyt i zmianę zawartości symulowanej pamięci oraz generację danych przeznaczonych do uruchamianego mikroprogramu. Użytkownik może operować na całej pamięci lub na jej fragmentach. Program ten zajmuje obszar pamięci 1,5 k bajtów. Bufor o pojemności 4 k bajty, zawierający dane wejściowe, zajmuje obszar od adresu 5000 H. Program SYM ROM VER.2.1 zapisany jest na dysketce, jako zbiór o nazwie katalogowej SYM ROM D.COM.

Inicjacja obu programów następuje w sposób typowy w systemie MSWP dla programów taśmowych bądź dyskowych.

Program SYM ROM ma następujące dyrektywy: A, M, P, S, Z i W.

- Dyrektywa A umożliwia zapisywanie w symulowanej pamięci mikroprogramów czyli w symulatorze mikroprogramu użytkownika.
- Dyrektywa M pozwala na wyświetlanie określonego przez użytkownika obszaru pamięci stałej w kodzie heksadecymalnym lub binarnym.
- Dyrektywa S ustala długość słowa mikroinstrukcji.
- Dyrektywa P oznacza powrót do programu MONITOR.
- Dyrektywa Z daje możliwość zmiany zawartości bajtu. Użytkownik ma możliwość wyboru kodu, w jakim będzie wyświetlana zmieniana zawartość bajtu.
- Dyrektywa W powoduje wyprowadzenie na taśmę perforowaną albo na dysketkę zawartości symulowanej pamięci mikroprogramów.

W czasie pracy programu SYMROM wydawane są komunikaty świadczące o różnego rodzaju błędach.

Zakres zastosowań

Symulator SYM-1 znajduje zastosowanie w uruchamianiu systemów wykorzystujących pamięci stałe, zwłaszcza typu Intel 3602/3622 lub 3604/3624. Jest wygodnym narzędziem do tworzenia mikroprogramów o strukturze podanej w opisie. Pewna niezależność programu SYMROM od sprzętu symulatora pozwala na tworzenie dowolnych mikroprogramów w postaci zbiorów wyprowadzanych na taśmę papierową lub dysketkę.

Symulator SYM-1 nadaje się zwłaszcza do uruchamiania systemów opartych na mikroprocesorach segmentowych.

mgr inż. Andrzej BACHAŃSKI
mgr inż. Irena KAMIENIECKA-WILD
Instytut Maszyn Matematycznych

Programatory pamięci EPROM typu INTEL 1702A oraz INTEL 2704, 2708, 2716, 2732

Wstęp

Programatory pamięci EPROM typu INTEL 1702A i 2704/32 należą do grupy wymiennych programatorów pracujących w mikroprocesorowym systemie wspomagania projektowania MSWP i stanowią jego opcjonalne moduły. Zapewniają one obsługę pamięci EPROM typu INTEL 1702A, 2704, 2708, 2716, 2732.

Inne typy programatorów zapewniają obsługę układów FPLA firmy Signetics typu 82S100/101, oraz pamięci produkcji węgierskiej typu TM 601, 602, 604, 621, 622 i 624 stanowiących po zaprogramowaniu odpowiedniki bipolarnych pamięci PROM typu INTEL 3601, 3602, 3604, 3621, 3622 i 3624.

Pamięci EPROM typu INTEL 1702A, 2704, 2708, 2716, 2732

Wymienione typy pamięci są reprogramowalnymi pamięciami o następujących konfiguracjach:

1702A	-	256 słów 8-bitowych
2704	-	512 słów 8-bitowych
2708	-	1024 słów 8-bitowych
2716	-	2048 słów 8-bitowych
2732	-	4096 słów 8-bitowych

Pamięci te zachowując tę samą liczbę pinów, różnią się między sobą konstrukcją wewnętrzną, liczbą wejść adresowych oraz poziomami napięć wymaganych do zapisu i odczytu informacji. Dla dokonania zapisu pamięć typu 1702A wymaga wysterowania 8 wejść adresowych oraz napięć zasilających +5V, +76V i -9V. Pamięci 2704 i 2708 wymagają wysterowania odpowiednio 9 i 10 wejść adresowych oraz doprowadzenia napięć zasilających +5V, +12V, +25,5V i -5V. Pamięci 2716 i 2732 wymagają wysterowania odpowiednio 12 i 12 wejść adresowych oraz doprowadzenia napięć zasilających +5V i +25,5V. W pamięciach typu 1702A 2716 i 2732 można zapisywać pojedyncze komórki pamięci, części obszarów pamięci oraz całe obszary pamięci, natomiast w pamięciach typu 2704 i 2708 możliwe jest zapisywanie wyłącznie całego obszaru pamięci.

Konstrukcja programatorów

Moduły konstrukcyjne programatorów 1702A i 2704/32 mają postać odpowiednio dwóch lub trzech standardowych pakietów umieszczanych w kasecie jednostki centralnej MSWP. Programator 2704/32 obsługuje cztery typy pamięci EPROM. Dopasowanie programatora do odpowiedniego typu pamięci realizowane jest za pomocą przełącznika klawiszowego znajdującego się na jednym z pakietów programatora. Obydwa programatory współpracują z wymiennym pulpitem programatora znajdującym się na płycie czołowej jednostki centralnej MSWP. Na pulpicie tym znajduje się podstawka 24-nóżkowa przystosowana do obsługiwanych typów pamięci EPROM oraz wskaźniki informujące o typie obsługiwanej pamięci EPROM. Obydwa typy programatorów korzystają ze specjalnych napięć stabilizowanych, dostarczanych przez układy stabilizacyjne, znajdujące się na wydzielonych pakietach programatorów.

Współpraca programatorów z MSWP

Programatory współpracują z MSWP za pośrednictwem szyny głównej MSWP typu MULTIBUS pod nadzorem programu sterującego.

Współpraca programatorów z MSWP (rys.1 i 2) odbywa się w sposób opisany niżej.

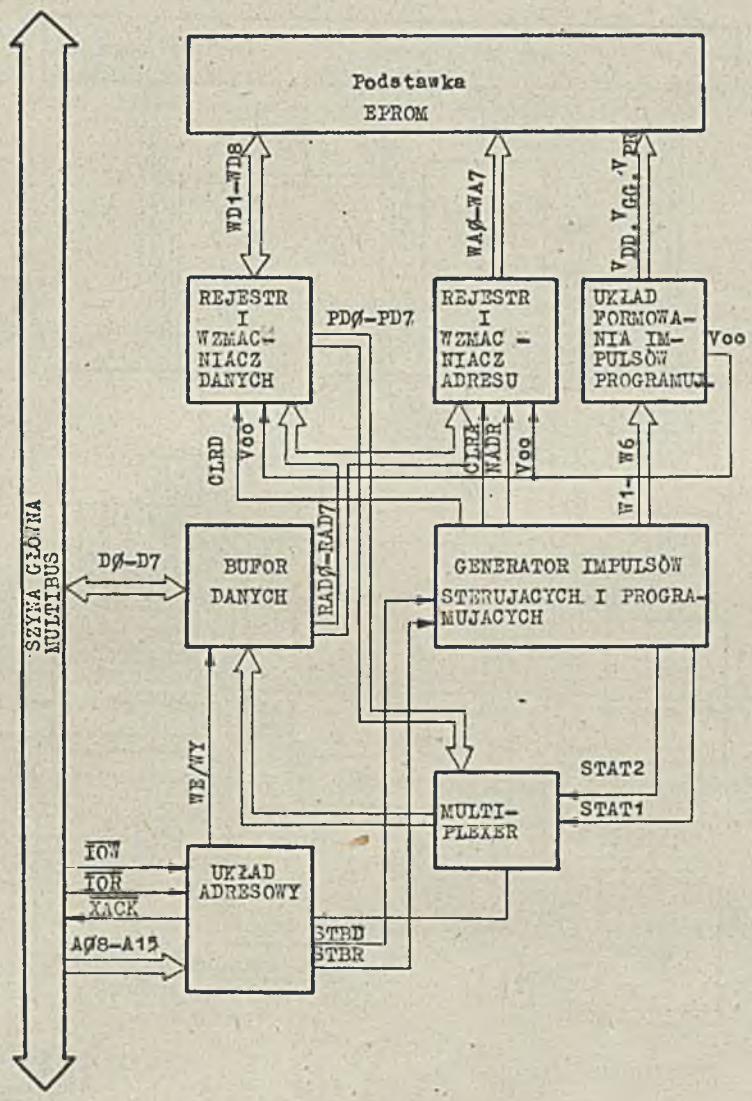
- Przez szyny adresowe A08 ÷ A15 i szyny sterujące I/O i I/OR realizowane jest przesyłanie do programatora rozkazów przez niego wykonywanych.
- Przez szynę danych D0 ÷ D7 zrealizowano w kierunku "zapis" :
  - przesyłanie do programatora 8-bitowego słowa danych przeznaczonych do zapisania w pamięci EPROM,
  - przesyłanie do programatora 8-bitowego adresu zapisywanej komórki pamięci EPROM - w wypadku programatora pamięci EPROM typu INTEL 1702A, lub kolejno młodszego i starszego bajtu adresowego zapisywanej komórki - w wypadku programatora pamięci EPROM typu INTEL 2704/32,
  - przesyłanie do programatora sekwencji sygnałów ustawiających w programatorze napięcia zasilające i programujące odpowiednie dla danego typu pamięci EPROM.
- Za pośrednictwem szyny danych D0 ÷ D7 zrealizowano też w kierunku "odczyt" :
  - przesyłanie z programatora danych odczytanych z pamięci EPROM,
  - przesyłanie z programatora informacji o stanie programatora,
  - przesyłanie z programatora pamięci EPROM typu INTEL 2704/32 informacji o ustawieniu programatora na żądany typ pamięci EPROM.

Bloki funkcjonalne programatora

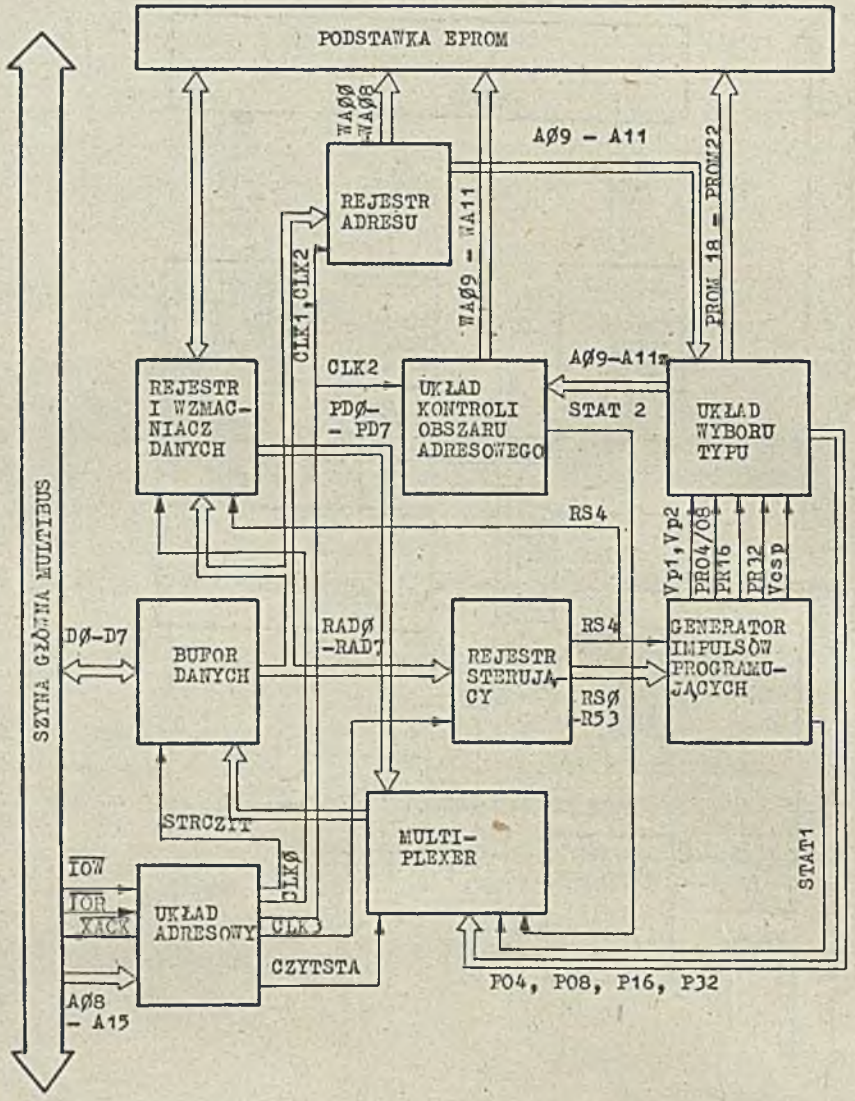
Programatory pamięci EPROM typu INTEL 1702A i 2704/32 zawierają odpowiednio 7 i 9 bloków funkcjonalnych (rys.1 i 2).

Programator pamięci EPROM typu INTEL 2704/32 zapewniający obsługę czterech typów pamięci EPROM rozbudowany jest o układ kontroli obszaru adresowego, układ wyboru typu pamięci oraz rejestr sterujący.

Programator pamięci EPROM typu INTEL 1702A ma specjalny układ formowania impulsów programujących.



Rys. 1. Schemat blokowy programatora pamięci EPROM typu INTEL 1702



Rys. 2. Schemat blokowy programatora pamięci EPROM typu INTEL 2704/32

Układ adresowy

Układ ten odbiera rozkazy przychodzące z systemu za pośrednictwem szyn adresowych A08 ÷ A15 i szyn sterujących I/OW i I/OR. Generuje sygnały sterujące w programatorze pamięci EPROM typu INTEL 1702A blokiem multipleksera, buforem danych i generatorem impulsów sterujących i programujących.

Układ adresowy w programatorze pamięci EPROM typu INTEL 270/32 steruje blokiem multipleksera, blokiem danych, rejestrem i wzmacniaczem danych, rejestrem adresu oraz rejestrem sterującym.

Multiplekser

Sterowany jest sygnałami z układu adresowego i przesyła do bufora danych informacje o stanie programatora oraz dane odczytane z pamięci EPROM. W programatorze pamięci EPROM typu INTEL 2704/32 przesyła on dodatkowo do bufora danych informacje o typie pamięci EPROM, na obsługę której programator został ustawiony.

Bufor danych

Zapewnia dwukierunkowe przesyłanie danych między programatorem i szyną główną MSWP.

Rejestr i wzmacniacz danych

Sterowany jest sygnałem generatora impulsów sterujących i programujących w programatorze pamięci EPROM typu INTEL 1702A i sygnałami z układu adresowego i rejestru sterującego w programatorze pamięci EPROM typu INTEL 2704/32.

Do rejestru i wzmacniacza danych wpisywane są dane przychodzące z bufora danych, które w fazie programowania podawane są bezpośrednio na odpowiednie wejścia pamięci EPROM. W czasie odczytu dane odczytane z pamięci EPROM zostają przez wzmacniacze danych przesłane do multipleksera.

Rejestr adresu

Sterowany jest sygnałami przychodzącymi z generatora impulsów sterujących i programujących w programatorze pamięci EPROM typu INTEL 1702A i sygnałami z układu adresowego w programatorze pamięci EPROM typu INTEL 2704/32. W programatorze pamięci EPROM typu INTEL 1702A do rejestru adresu dołączone są dodatkowe wzmacniacze adresu, co wynika z wymagań napięciowych obsługiwanej przez ten programator pamięci.

Do rejestru adresowego wpisywany jest z bufora danych adres zapisywanej lub odczytywanej komórki pamięci EPROM.

Generator impulsów sterujących i programujących

Występuje on w programatorze pamięci EPROM typu INTEL 1702A. Przetwarza sygnały sterujące podawane z układu adresowego na sygnały sterujące rejestrami danych i adresu. Generator wytwarza ponadto ciągi impulsów sterujących układem formowania impulsów programujących.

Układ formowania impulsów programujących

Stanowi jeden z bloków programatora 1702A i wytwarza sekwencje impulsów programujących poziomy napięć wymaganych dla obsługiwanego typu pamięci EPROM.

Rejestr sterujący

Rejestr ten jest sterowany z układu adresowego programatora pamięci EPROM typu INTEL 2704/32. Wpisywane są do niego dane z bufora danych, określające rodzaj pracy programatora, oraz sterujące generatorem impulsów programujących.

Generator impulsów programujących

Generator ten wytwarza sekwencje impulsów programujących odpowiednie dla obsługiwanych typów pamięci EPROM. Sterowany jest zawartością rejestru sterującego pamięci EPROM typu INTEL 2704/32.

Układ wyboru typu

Układ ten umożliwia ustawienie programatora pamięci EPROM typu INTEL 2704/32 na obsługę wybranego typu pamięci EPROM i jest sterowany przełącznikiem klawiszowym znajdującym się na jednym z pakietów programatora.

Przez odpowiednie ustawienie przełączników następuje przełączenie napięć zasilających i sygnałów doprowadzanych z generatora impulsów programujących i rejestru adresu na odpowiednie wyprowadzenia /piny/ pamięci EPROM.

Układ kontroli obszaru adresowego

Zadaniem tego układu w programatorze pamięci EPROM typu INTEL 2704/32 jest testowanie ostatniego bitu adresowego doprowadzanego z układu wyboru typu. W sytuacji, gdy adres przekracza obszar adresowy danej pamięci EPROM w układzie tym generowany jest sygnał przekroczenia obszaru adresowego.

Oprogramowanie programatorów pamięci EPROM typu INTEL 1702A i 2704/32

Celem oprogramowania programatorów pamięci EPROM jest dostarczenie wygodnego narzędzia do zapisu i odczytu pamięci EPROM typu INTEL 1702A oraz 2704, 2708, 2716 i 2732.

Oprogramowanie to zapewnia też możliwość wprowadzania i obróbki danych przeznaczonych do zapisu do pamięci mikroukładu EPROM, a także wyprowadzenia na urządzeniu listującym zawartość mikromodułu pamięci EPROM dla celów kontrolnych lub dokumentacyjnych.

Programy pisane są w języku assemblera INTEL 8080. Działają zależnie od wersji programu pod kontrolą programu MONITOR VER. 1.8 lub systemu operacyjnego OS-I VER.1.4. Programy są w dwóch wersjach, zależnych od rodzaju nośnika, na którym program się znajduje:
- gdy nośnikiem jest taśma papierowa przewidziana jest wersja 1.1 lub 1.0 programu,
- gdy nośnikiem jest dysk elastyczny przewidziana jest wersja 2.1 lub 2.0 programu.

Ze względu na brak obsługi dysków elastycznych w programie MONITOR VER.1.8 dla programów wersji 2.1 przyjęto system operacyjny OS-I VER.1.4 jako system nadrzędny. Konsekwencją tego są różne sposoby inicjowania programów w zależności od użytej wersji, a także różne postacie modułów ładowalnych programów.

Programy obsługi programatora 1702A i program obsługi programatora 2704/32 różnią się organizacją i możliwościami. Program obsługi programatora 1702A ma pięć dyrektyw realizujących następujące funkcje:
- zaprogramowanie mikroukładu pamięci EPROM 1702A danymi z dowolnego miejsca pamięci RAM,
- odczytanie zawartości mikroukładu pamięci EPROM 1702A i sprawdzenie, czy jest on zapisany,
- przepisanie do zadeklarowanego miejsca w pamięci RAM danych z mikroukładu pamięci EPROM 1702A,
- wyprowadzenie zawartości mikroukładu pamięci EPROM 1702A na urządzenie listujące,
- powrót do programu nadrzędnego MONITOR VER.1.8.

Program ten nie zawiera dyrektyw związanych z wprowadzeniem i obróbką danych do zaprogramowania.

Zakłada się, że dane do zaprogramowania powinny być wprowadzane do pamięci i ewentualnie poddane zmianom za pomocą standardowych dyrektyw programu MONITOR VER.1.8. Dane te mogą znajdować się w dowolnym miejscu pamięci RAM z wyłączeniem obszaru zajętego przez program obsługi programatora oraz przez pole robocze programu MONITOR VER.1.8. Wymagana jest przy tym, w przypadku pamięci EPROM typu INTEL 1702A, zgodność młodszych bajtów adresów danych w pamięci RAM z adresami w mikroukładzie pamięci EPROM.

Program obsługi programatora 2704/32, w porównaniu z programem obsługi programatora 1702A ma dodatkowe możliwości. Są to: możliwość wprowadzania do bufora programu w pamięci RAM danych do zaprogramowania oraz możliwość ich oglądania i ewentualnej zmiany w buforze. Bufor ten zajmuje stały obszar w pamięci RAM o pojemności 4 Kbajtów. Do mikroukładu pamięci EPROM mogą być zapisane tylko dane z tego bufora. Jest przy tym zachowana pełna odpowiedniość adresów w buforze i w mikromodule pamięci EPROM. Adresacja buforu nie pokrywa się z adresacją pamięci RAM. Zakłada się, że dane do zaprogramowania wprowadzane do bufora za pomocą dyrektywy programu R, znajdują się na takim samym nośniku, na którym znajduje się program obsługi programatora. Tak więc dla wersji 1.1 programu obsługi programatora 2704/32 - dyrektywa R dotyczy wprowadzania danych z taśmy papierowej, podczas gdy dla wersji 2.1 programu obsługi programatora - dyrektywa R dotyczy wprowadzania danych do bufora z dysku. Program ten wraz z buforem zajmuje ok. 5,5 Kbajtów pamięci RAM.

Dyrektywy programów obsługi programatorów

Dyrektywy wprowadza się po wywołaniu programu i po jego zgłoszeniu się, w którym podawana jest nazwa programu, jego wersja oraz typ pamięci EPROM, której program dotyczy. Np.

IMM/MSWP PROGRAMATOR 2704/32 VER.2.1
2716:

Dyrektywy można wprowadzać także wtedy, gdy program obsługi programatora zgłosi się podaniem typu mikroukładu pamięci EPROM i dwukropkiem, np.

2732:

co oznacza jego gotowość do pracy. Tego typu zgłoszenie następuje po zakończeniu działania uprzednio podanej dyrektywy.

Wszystkie dyrektywy programów obsługi programatorów mają mnemoniki jednoliterowe i po podaniu parametrów dyrektywy powinny być zakończone znakiem kropki ".", z wyjątkiem dyrektywy D, w której dopuszcza się także użycie znaku "*". Jest to omówione szczegółowo w opisie dyrektyw.

Przy opisie dyrektyw przyjęto jako zasadę, że wielkości nie podkreślone są wypisywane przez program obsługi programatora, podkreślane - przez użytkownika.

Dyrektywy programu obsługi programatora 1702A

◉ Dyrektywa P

Dyrektywa ta służy do zaprogramowania mikromodułu pamięci EPROM typu Intel 1702A danymi z zadeklarowanego w dyrektywie obszaru pamięci RAM. Postać dyrektywy jest następująca:

1702A: P
<adres początku obszaru> <adres końca obszaru> <.>

Pobrane z pamięci RAM dane zostają przesłane do mikromodułu pamięci EPROM pod adresy określone przez młodsze bajty adresu pamięci RAM, np. dana pobrana spod adresu 0101H zostanię zapisana pod adresem 01H w mikromodule pamięci EPROM.

W wypadku błędnego zapisu pojawia się komunikat WRITE ERROR. Najczęściej przyczyną błędnego zapisu jest uprzednio już zapisany lub uszkodzony mikromoduł pamięci EPROM.

◉ Dyrektywa D

Dyrektywa ta służy do wypisania na urządzeniu listującym zawartości mikromodułu pamięci EPROM typu Intel 1702A. Obszar pamięci podlegający listowaniu deklarowany jest podczas podawania dyrektywy: Postać dyrektywy jest następująca:

1702A: D
<adres początku obszaru> <adres końca obszaru> {*, .}

Znak "." kończący dyrektywę powoduje wydruk na standardowym dla danego programu zarządzającego urządzeniu listującym, np. dla programu MONITOR VER.1.8 standardowym urządzeniem listującym jest monitor ekranowy, a dla systemu operacyjnego OS-I VER.1.4 - drukarka. Podanie znaku "*" powoduje wydruk na niestandardowym urządzeniu listującym.

◉ Dyrektywa M

Dyrektywa ta służy do przepisania zawartości mikromodułu pamięci EPROM typu Intel 1702A do zadeklarowanego w dyrektywie obszaru pamięci RAM. Postać dyrektywy jest następująca:

1702A: M
<adres początku obszaru> <adres końca obszaru> <.>

Dyrektywa ta przydatna jest głównie w sytuacji, gdy kilka mikromudułów pamięci EPROM typu Intel 1702A ma być zaprogramowanych wg mikromodułu wzorcowego. Przepisana za pomocą tej dyrektywy zawartość mikromodułu wzorcowego może stanowić dane do zaprogramowania innych mikromodułów pamięci EPROM typu Intel 1702A.

◉ Dyrektywa T

Dyrektywa ta służy do sprawdzania czy mikromoduł pamięci EPROM typu Intel 1702A jest zapisany. Jeżeli jest zapisany, pojawi się komunikat: CHIP IS NOT ERASED. Po wydrukowaniu komunikatu lub jeżeli pamięć EPROM jest nie zapisana, program zgłasza się podaniem typu pamięci EPROM i oczekuje na podanie kolejnej dyrektywy. Postać dyrektywy:

1702A: T<.>

◉ Dyrektywa E

Działanie dyrektywy polega na oddaniu sterowania programowi zarządzającemu. Dla programu w wersji 1.1 jest to program MONITOR VER.1.8, a dla programu wersji 2.1 - system OS-I VER.1.4.

Dyrektywy programu obsługi programatora 2704/32

◉ Dyrektywa P

Dyrektywa P służy do programowania mikromodułów pamięci EPROM typu Intel 2704, 2708, 2716 lub 2732 danymi z zadeklarowanego obszaru bufora programu. Istnieje przy tym odpowiedniość adresów bufora i mikromodułu pamięci EPROM. Postać dyrektywy:

<typ pamięci EPROM>: P
<adres początku obszaru> <adres końca obszaru> <.>

Po zaprogramowaniu istnieje kontrola zapisu do mikroukładu pamięci EPROM. W wypadku błędu na konsoli systemowej wyświetlana jest informacja o błędzie. Podawany jest adres błędnie zapisanej komórki, wartość która powinna być w tej komórce i wartość, która została odczytana z tej komórki.

◉ Dyrektywa D

Dyrektywa ta służy do wydruku na urządzeniu listującym zawartości określonego w dyrektywie obszaru pamięci mikromodułu pamięci EPROM. Postać dyrektywy jest następująca:

TYP PAMIECI : D
<adres początku> <adres końca> <.> lub <*>

Działanie dyrektywy polega na przepisaniu zadeklarowanego obszaru mikromodułu pamięci EPROM pod analogiczne adresy do bufora w pamięci RAM, a następnie wydrukowanie na urządzeniu listującym zawartości zadeklarowanego obszaru bufora.

Gdy dyrektywa zakończona jest znakiem ".", urządzeniem listującym jest urządzenie standardowe dla danego systemu operacyjnego. Zakończenie zaś dyrektywy znakiem "*" powoduje wydruk na niestandardowe urządzenie listujące.

Dla programu MONITOR standardowym urządzeniem listującym jest monitor ekranowy, a niestandardowym drukarka. Dla systemu operacyjnego OS-I jest odwrotnie.

◉ Dyrektywa T

Dyrektywa ta służy do sprawdzanie czy dany mikromoduł pamięci EPROM jest zapisany. Zapisanie pięciu lub więcej komórek mikroukładu pamięci EPROM powoduje wyświetlenie na konsoli systemowej komunikatu:

CHIP IS NOT ERASED

Gdy wszystkie komórki pamięci EPROM mikroukładu są niezapisane, komunikat ma postać:

CHIP IS ERASED

Gdy zapisanych jest 5 lub mniej komórek mikroukładu pamięci EPROM - komunikat zawiera adresy zapisanych komórek, ich zawartość oraz informację:

APART FROM ABOVE CELLS CHIP IS ERASED

◉ Dyrektywa R

Dyrektywa ta służy do wprowadzania danych przewidzianych do zaprogramowania - do bufora. Dane te mogą być wprowadzane bądź z taśmy perforowanej dla wersji programu 1.0, bądź z dysku elastycznego dla wersji programu 2.0. Dane do zaprogramowania mają postać rekordów, z których każdy ma między innymi własny adres ładowania do bufora.

Prawidłowe wykonanie dyrektywy R w programie VER.2.0 uwarunkowane jest zadeklarowaniem podczas ściągania programu obsługi programatora 2704/32 z dysku elastycznego zbioru, w którym znajdują się dane do zaprogramowania. Postać dyrektywy jest następująca:

< TYP PAMIĘCI EPROM > : R

◉ Dyrektywa S

Dyrektywa ta służy do wyświetlania na konsoli i ewentualnej zmiany zawartości zadeklarowanej komórki bufora. Postać dyrektywy:

< TYP PAMIĘCI EPROM > : S

< adres komórki pamięci bufora > < zawartość komórki pamięci > [; / nowa wartość]

Wykonanie dyrektywy powoduje wypisanie zawartości zadeklarowanej komórki bufora i oczekiwanie na podanie nowej zawartości, znaku "." lub znaku ",". Znak "." kończy działanie dyrektywy, znak "," powoduje wyświetlenie zawartości komórki pamięci o adresie o 1 większym od ostatnio wyświetlonego.

◉ Dyrektywa E

Dyrektywa E powoduje:

- dla wersji 1.0 programu powrót do programu MONITOR VER.1.7,
- dla wersji 2.0 programu powrót do systemu operacyjnego OS-I VER.1.4.

Uwagi końcowe

W Zakładzie Systemów Mikroprocesorowych IMM opracowywane są programatory innych typów pamięci wymienione we wstępie. W związku z tym przewiduje się ujednolicenie programów obsługi wszystkich programatorów. Pod względem funkcjonalnym programy te będą zbliżone do programu obsługi programatora pamięci EPROM typu INTEL 2704/32, tzn. będą dawały użytkownikowi podobne możliwości w zakresie obróbki danych, jak mówione powyżej.

mgr inż. Magdalena CZAIŃSKA
mgr inż. Krzysztof DZIK
mgr inż. Małgorzata SADOWSKA-ROSIŃSKA
Instytut Maszyn Matematycznych

Charakterystyka techniczna testera UMT-1 i zakres jego zastosowań

Wprowadzenie

Tester UMT-1 jest modułem opcjonalnym Mikroprocesorowego Systemu Wspomagania Projektowania - MSWP. Tester ten jest zainstalowany w jednostce centralnej MSWP. Jest on przeznaczony do testowania płytek obwodów drukowanych, pakietów, zespołów i układów cyfrowych, z sygnałami wejściowymi i wyjściowymi o poziomach TTL. Tester UMT-1 wysyła zaprogramowane stany logiczne odpowiadające poziomom logicznym TTL do punktów testowych definiowanych jako wejścia testowanego modułu i po upływie określonego czasu odczytuje stany logiczne występujące na punktach testowych stanowiących wyjścia badanego modułu. Punktami testowymi mogą być, zarówno styki złącz, jak i dodatkowe punkty testowe, np. kontakty układów scalonych, punkty lutownicze i inne występujące w badanym układzie. Dodatkowe punkty testowe są dostępne dla testera, ponieważ wyposażony jest w dodatkowe wtyki w podstawkach układów scalonych, klipsy obejmujące układy scalone i sondy umożliwiające ich zawieszenie w dowolnym punkcie badanego układu.

Podstawowe dane techniczne testera UMT-1

- Możliwość jednoczesnego testowania maksymalnie 320 punktów testowych badanego układu definiowanych jako jego wejścia lub wyjścia,
- odpowiedniość poziomów logicznych sygnałów testujących z poziomami układów scalonych TTL /"0" logiczne 0,3 V, "1" logiczne 3 V/,
- wydajność prądowa jednego styku testera większa od 16 mA /10 jednostek TTL/,
- obciążalność badanego punktu testowego przez styk testera mniejsza od 1,6 m A /1 jednostka TTL/,
- możliwość programowania czasu opóźnienia /strobu/ upływającego od momentu wysłania zaprogramowanych stanów logicznych do badanego układu - do momentu odczytu stanów logicznych badanego układu w zakresie od 100 ns do 1 s,
- możliwość wykonania w jednym przebiegu maksymalnie 200 kroków testu, przy czym krok testu jest zdefiniowany jako jednokrotne wysłanie zaprogramowanych stanów logicznych do badanego układu i po czasie strobu odczytanie stanów logicznych badanego układu,
- wymienność uchwytu złącz testowych układów dająca możliwość łatwego sprawdzania pakietów o różnych standardach konstrukcyjnych,

- możliwość łatwego zestawiania testera stosownie do liczby obsługiwanych punktów testowych /modularność konstrukcyjna/.

Struktura logiczna testera UMT-1

Tester UMT-1 zbudowany jest z układu sterowania współpracującego z jednostką centralną MSWP i wysyłającego sygnały sterujące do układów bezpośrednio podających i odbierających stany logiczne z badanego układu i z układów bezpośrednio obsługujących punkty testowe /zwykle styki złącz zamontowanych na pokrywie jednostki centralnej MSWP/. Ponadto w skład testera wchodzi układ doprowadzający napięcie zasilania do badanego układu.

Wszystkie wymienione wyżej układy są wykonane w postaci 12 pakietów o standardzie MSWP ustawianych w odpowiednie miejsce kasety jednostki centralnej MSWP.

Schemat blokowy testera UMT-1 oraz jego sposób dołączania do szyny głównej systemu MSWP jest przedstawiony na rysunku 1.

Na rysunku 1 przedstawiono strukturę logiczną testera UMT-1, która obejmuje układ sterowania i układy obsługujące punkty testowe.

Układ sterowania zawiera:

- dekoder starszych bitów adresowych wybierający odpowiednie układy obsługujące poszczególne grupy punktów testowych,
- sterowanie wytwarzające sygnały sterujące dla układów obsługujących punkty testowe,
- licznik służący do odliczania czasu odczytania stanów logicznych punktów testowych /strob/.

Układ obsługujący punkty testowe zawiera:

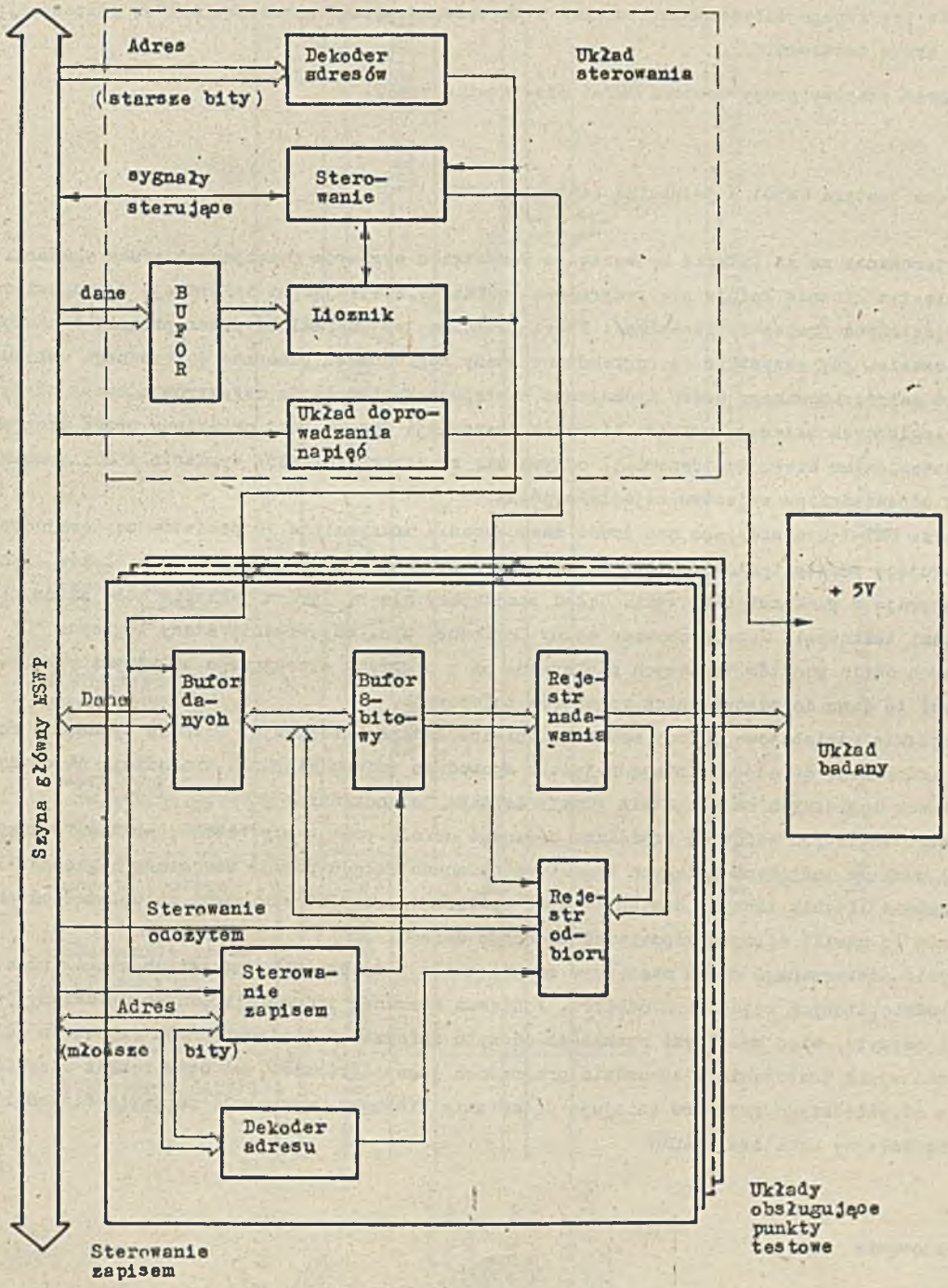
- dekoder młodszych bitów adresowych wybierający odpowiednie 8-bitowe grupy punktów testowych w układzie obsługującym,
- 8-bitowy bufor kompletujący zaprogramowane stany logiczne,
- rejestry nadawania i odbioru służące do bezpośredniego wysyłania i odbioru sygnałów testera do i od badanego układu,
- bufor danych współpracujący z szyną główną MSWP.

Zasada działania

Działanie testera jest opisane na podstawie sekwencji występującej przy wykonaniu kroku testu. Test badanego układu składa się zwykle z wielu kroków wykonywanych identycznie przez tester.

Realizacja każdego kroku testu przebiega w sposób następujący:

- załadowanie zaprogramowanej wartości czasu opóźnienia /strobu/ do licznika,
- załadowanie zaprogramowanych stanów logicznych w postaci ciągu zerojedynkowego do ośmiobitowych buforów. Każdy bit bufora odpowiada określonemu punktowi testowemu. Zerowa wartość bitu bufora spowoduje zadanie stanu logicznego "0" /około 0,3 V / na przyporządkowany temu bitowi punkt testowy, natomiast wartość 1 bitu bufora będzie prowadzić do zadania stanu logicznego "1" /około 3 V /,
- start licznika z jednoczesnym podawaniem stanów logicznych na wszystkie punkty testowe badanego układu,
- odczytanie stanów logicznych występujących w punktach testowych badanego układu do rejestrów odbioru w momencie zakończenia odliczania licznika,



Rys. 1. Schemat blokowy testera UMT-1

- odczytanie przez program ośmiobitowych rejestrów odbioru i zbadanie poprawności odpowiedzi badanego układu na zadane stany logiczne,
- umożliwienie poprawnego załadowania licznika i buforów następną wartością, a więc rozpoczęcie następnego kroku testowania.

Harmonogram czasowy pracy testera UMT-1 przedstawia rys.2.

Współpraca testera UMT-1 z jednostką centralną MSWP

Układ sterowania ma za zadanie wytworzenie wszystkich sygnałów sterujących pracą testera. Znajdujący się tam licznik ładuje się programowo rozkazami wysyłającymi informację do odpowiednich rejestrów wyjściowych /rejestry licznika/. Start licznika jest określony przez program i następuje w tym samym czasie, gdy wszystkie zaprogramowane stany logiczne są podawane do badanego układu. Po odliczeniu zaprogramowanego czasu opóźnienia następuje zapisanie do rejestrów odbioru odczytanych stanów logicznych badanego układu. Kierunek transmisji danych jest określony przez program. Sterowanie ustawieniem kierunku transmisji odbywa się za pomocą rozkazów wysłania i odbierania informacji z odpowiedniego rejestru wejścia/wyjścia.

W testerze UMT-1 przewidziano możliwość zastosowania maksymalnie 10 pakietów zawierających układy obsługujące punkty testowe. Każdy taki układ zbudowany jest z 4 jednakowych bloków, z których każdy steruje 8 punktami testowymi. Układ znajdujący się na jednym pakiecie może zatem sterować 32 punktami testowymi. Zaprogramowane stany logiczne, tzn. odpowiednie stany logiczne "0" lub "1" dla każdych ośmiu punktów testowych zapisywane są z programu sterującego kolejnymi rozkazami przesyłającymi te dane do odpowiednich rejestrów buforowych.

Gdy wszystkie ośmiobitowe bufory zostaną zapisane żądaną informacją, wówczas wykonanie rozkazu wysłania informacji do odpowiedniego rejestru spowoduje zmianę kierunku transmisji na odczyt i podanie stanów logicznych na wszystkie punkty testowe jednocześnie.

Do punktów testowych będących wejściami badanego układu podaje się "zero-jedynkowe" stany logiczne, a do punktów testowych będących wyjściami badanego układu podaje się stany logiczne "1".

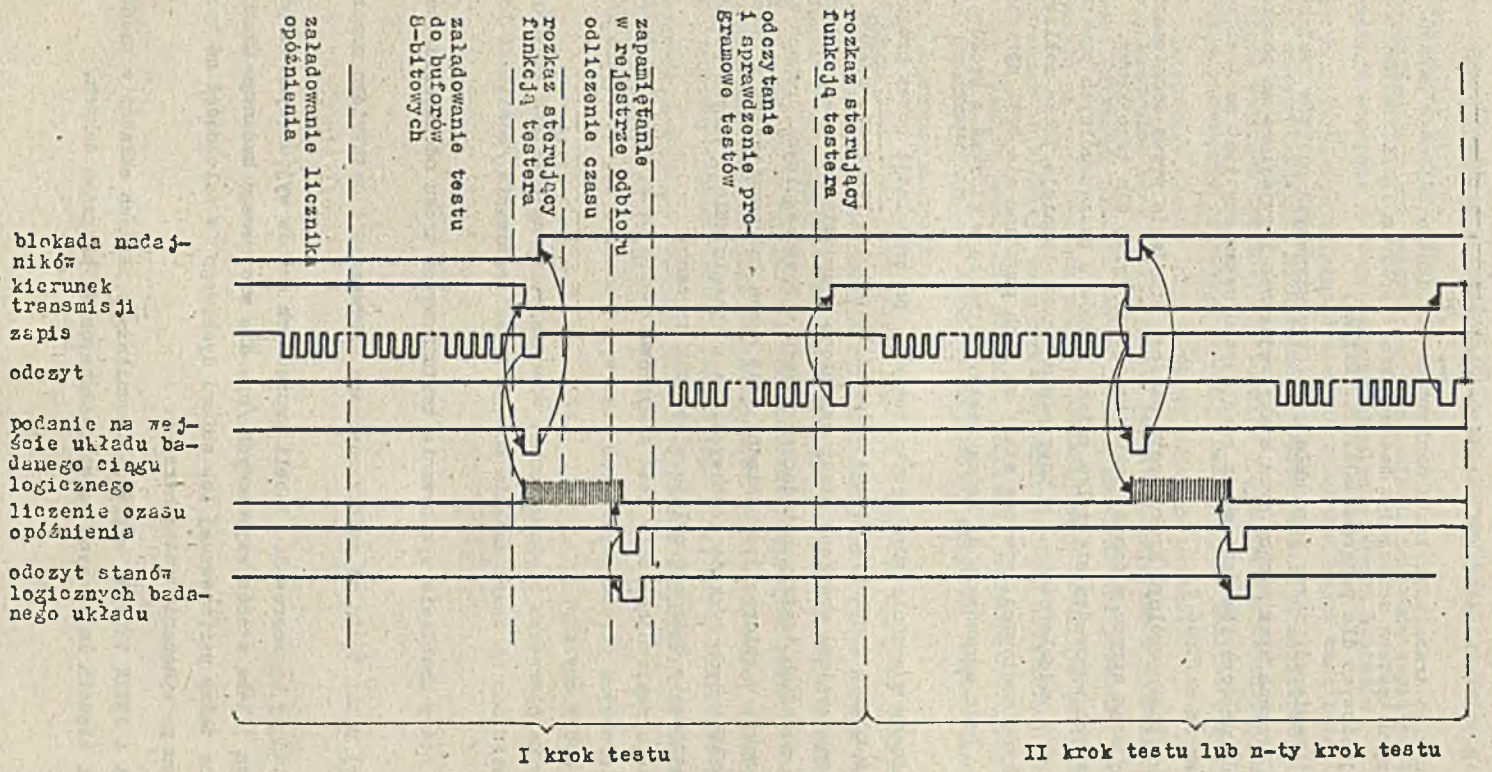
Jednocześnie licznik zaczyna odmieniać czas opóźnienia, czyli czas, jaki ma upłynąć od chwili zadania stanów do chwili odczytu odpowiedzi badanego układu.

Po upływie odmierzonego czasu następuje zapamiętanie jednocześnie wszystkich stanów punktów testowych w ośmiobitowych rejestrach odbioru. Ponieważ kierunek transmisji danych ustawiony jest dla operacji odczytu, więc kolejnymi rozkazami odczytu informacji z odpowiednich rejestrów odbioru można odczytać wynik testowania i sprawdzić programowo jego poprawność, po czym rozkaz odczytania informacji z odpowiedniego rejestru inicjuje ustawienie kierunku transmisji na zapis danych i może rozpocząć się kolejny krok testowania.

Oprogramowanie

Oprogramowanie testera UMT-1 składa się z dwu programów: MONITOR TESTERA i JEZYK TESTERA. Programy te wykonane są w dwu wersjach:

- w wersji dyskowej, wówczas pracują pod systemem operacyjnym OS-I, a dane wejściowe i wyjściowe zapisane są jako zbiory systemu OS-I,
- w wersji na taśmach perforowanych, wówczas współpracują z programem MONITOR, a dane wejściowe i wyjściowe zapisane są również na taśmie perforowanej.



Rys. 2. Harmonogram czasowy testera UMT-1

Program MONITOR TESTERA steruje pracą TESTERA UMT-1, wysyła zadane stany wejściowe do punktów testowych badanego układu według zadanego testu oraz bada po zadanym czasie strobu poprawność stanów wyjściowych występujących w punktach testowych badanego układu.

Program JEZYK TESTERA służy do translacji testu napisanego dla danego badanego układu. Test pisany jest w symbolicznym języku testera opisującym badany układ: program JEZYK TESTERA transluje ten test na tablicę danych wejściowych dla programu MONITOR TESTERA.

Test badanego układu musi zawierać: nazwę tego układu, liczbę używanych punktów testowych oraz w kolejnych numerowanych krokach czas strobu oraz stany wejściowe i wyjściowe na poszczególnych punktach testowych. Przed rozpoczęciem testowania czas strobu ustawiany jest na 1 mikrosekundę, a wszystkie stany wyjściowe na wartość 1.

Nie podanie czasu strobu w poszczególnym kroku testu oznacza przyjęcie przez domniemanie czasu strobu równego 1 us; podany czas strobu w danym kroku obowiązuje tylko dla tego kroku.

W kolejnych krokach testowych wprowadza się tylko zmiany stanów w poszczególnych punktach testowych, w porównaniu do stanów zadeklarowanych w poprzednim kroku. Istnieje także możliwość maskowania poszczególnych punktów testowych, wówczas stany na tych punktach nie będą sprawdzane. Zamaskowane punkty testowe nie będą sprawdzane do końca testu, chyba że w którymś z kroków zostaną odmaskowane.

Test badanego układu napisany w symbolicznym języku testera, po translacji przez program JEZYK TESTERA, stanowi zbiór w kodzie wynikowym będący zbiorem danych dla programu MONITOR TESTERA. Jest to dwuwymiarowa tablica zawierająca wysyłane stany wejściowe i spodziewane stany wyjściowe we wszystkich zadeklarowanych punktach testowych badanego układu w poszczególnych krokach oraz wielkość strobu dla każdego kroku; w zbiorze tym zapisana jest nazwa badanego układu i ewentualnie maski dla maskowanych i odmaskowywanych punktów testowych. Po wczytaniu zbioru danych program MONITOR TESTERA pozwala na testowanie badanego układu w siedmiu trybach:

- tryby testowania standardowego realizujące cały test z wyprowadzeniem na konsolę lub także drukarkę wszystkich błędów, pierwszych pięciu błędów lub bez wyprowadzania błędów tylko komunikatu czy układ badany jest dobry czy zły,
- tryb zatrzymania testowania na pierwszym błędnym kroku, wówczas stany na punktach testowych badanego układu zostaną niezmienione do czasu zadania nowego trybu testowania /możliwość lokalizacji uszkodzenia/,
- tryb testowania krokowego, który umożliwia wykonywanie testu krok po kroku od zadeklarowanego numeru kroku,
- tryb testowania w zamkniętej pętli testu od zadeklarowanego pierwszego i ostatniego numeru kroku w pętli,
- tryb autogeneracji umożliwiający wygenerowanie tabeli wzorcowych stanów wyjściowych na podstawie wykonania testu z zadaną tabelą stanów wejściowych /np. dla wzorcowego badanego układu/. Istnieje możliwość uzyskania taśmy perforowanej lub zbioru dyskowego /w zależności od wersji programu/ z poprawnym testem po wykonaniu autogeneracji.

Programy MONITOR TESTERA i JEZYK TESTERA wypisują komunikaty o błędach składni w zadawaniu trybów oraz błędach składni i błędach logicznych we wprowadzanych do programu danych.

Zakres zastosowań testera UMT-1 i jego możliwości

Tester UMT-1 znajduje głównie zastosowanie w uruchamianiu układów, pakietów i modułów cyfrowych zbudowanych z układów scalonych TTL.

Uzyskane bardzo dobre wyniki przy uruchamianiu pakietów serii prototypowej MSWP na testerze UMT-1 pozwalają znacznie skrócić czas uruchamiania nowych systemów.

Prostota i łatwa wymienialność elementów mechanicznych umożliwia testowanie pakietów o różnych standardach konstrukcyjnych. Różnorodność wyposażenia daje możliwość odczytu stanów logicznych nie tylko ze złącz badanych układów, ale również z punktów testowych znajdujących się wewnątrz badanego układu. Ocena parametrów czasowych badanych układów jest możliwa przez dołączenie oscyloskopu, sondy lub analizatora stanów logicznych do badanego układu.

Punkty kontrolne są w bardzo prosty sposób deklarowane w symbolicznym języku testera - daje to możliwość szybkiego pisania testów również przez średni personel techniczny. Dyrektywy programowe pozwalają na wybór sposobu pracy odpowiadającego aktualnym potrzebom osoby obsługującej. pętle programowe umożliwiają dołączenie wspomnianych już dodatkowych urządzeń, jak sondy, oscyloskopy i inne. Pisanie testów dla układów, pakietów i modułów cyfrowych znacznie ułatwiają szerokie możliwości programowe systemu MSWP. Ma to istotny wpływ na zwiększenie efektywności pisania, uruchamiania i poprawiania testów.

Obecnie opracowywana nowa wersja testera z analizą sygnatur rozszerza znacznie jego zakres zastosowań, dając możliwość lokalizacji uszkodzeń, co ułatwia serwis urządzeń i modułów cyfrowych. Środki programowe i sprzętowe testera pozwalają na tworzenie powtarzalnych i stabilnych sygnatur. Sygnatury te mogą być odczytywane przez analizator sygnatur będący dodatkowym wyposażeniem systemu MSWP. Przygotowany test badanego układu stanowi jego pobudzenie, a zdejmowane w wybranych punktach testowych sygnatury odzwierciedlają jakość jego pracy dając możliwość szybkiej lokalizacji uszkodzeń. Zdejmowanie sygnatur może być przeprowadzane w zespołach serwisowych i nie wymaga wykwalifikowanego personelu obsługującego.

TECHNIKI KOMPUTEROWE 5-6/84

dr inż. Jolanta BRZOSTEK-PAWŁOWSKA
mgr inż. Wojciech KUBERA
Instytut Maszyn Matematycznych

Proces tworzenia oprogramowania użytkowego wspomagany programami narzędziowymi systemu MSWP

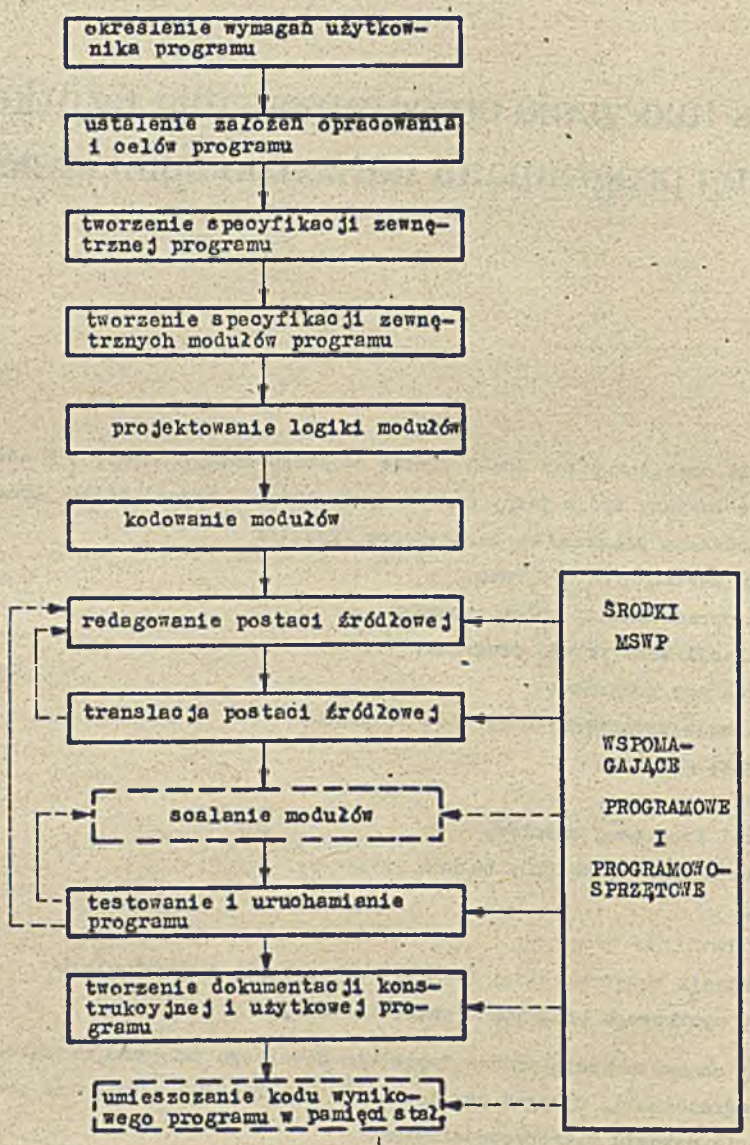
Wprowadzenie

Proces tworzenia oprogramowania można rozbić na kilka różnych etapów. W dobrze przemyślanym opracowaniu etapy te powinny być w jasny sposób wyodrębnione. Typowe etapy opracowania, rozumianego jako budowanie i produkcja programu są nastcpujące /rys.1/:

- określenie wymagań użytkownika programu,
- ustalenie założeń opracowania i celów programu,
- tworzenie specyfikacji zewnętrznej programu,
- projektowanie struktury programu,
- tworzenie specyfikacji zewnętrznych modułów programu,
- projektowanie logiki modułów,
- kodowanie modułów,
- redagowanie postaci źródłowej modułów,
- translacja postaci źródłowej modułów na kod wynikowy,
- scalanie modułów,
- testowanie i uruchamianie programu,
- tworzenie dokumentacji konstrukcyjnej i użytkowej programu i ewentualnie
- umieszczanie kodu wynikowego programu w pamięci stałej.

Powyższe etapy charakteryzują proces tworzenia dowolnego programu niezależnie od jego zastosowania, języka programowania, wymagań użytkownika, itp. Charakteryzują też proces tworzenia użytkowego oprogramowania sprzętu mikroprocesorowego.

Celem tego artykułu jest zaznajomienie czytelnika z możliwościami wspomagania realizacji niektórych wymienionych etapów tworzenia użytkowego oprogramowania sprzętu mikroprocesorowego przez dostępne w MSWP systemowe środki narzędziowe. Środki sprzętowo-programowe ułatwiają tworzenie nie tylko oprogramowania sprzętu mikroprocesorowego, ale również tworzenie tego sprzętu /zwłaszcza uruchamianie/ oraz integrację sprzętu z jego oprogramowaniem. Ze względu na szeroką gamę programowych środków wspomagających, dostępnych w MSWP, szerzej zostaną omówione tylko wybrane środki programowe, o innych zostaną podane krótkie uwagi.



Rys. 1. Etapy tworzenia oprogramowania użytkowego

Rys.1 pokazuje, które etapy tworzenia użytkowego oprogramowania mikroprocesorowego mogą być wspomagane systemowymi środkami MSWP. Są to etapy:

- redagowania /edycji/ postaci źródłowej programu,
- translacji postaci źródłowej programu,
- scalania modułów programu,
- testowania i uruchamiania programu,
- umieszczania kodu wynikowego programu w pamięci stałej, i częściowo
- tworzenie dokumentacji konstrukcyjnej i użytkowej programu.

Narzędziowe środki wspomagające dany etap procesu tworzenia użytkowego oprogramowania różnią się możliwościami funkcjonalnymi oraz nadrzędnym środowiskiem programowym, w jakim mogą działać. Ono też ma wpływ na zakres możliwości funkcjonalnych środków narzędziowych. Również zasoby sprzętowe danego zestawu MSWP ograniczają te możliwości funkcjonalne lub też uniemożliwiają stosowanie niektórych środków wspomagających.

W dalszej części opracowania podamy zestawienie wymagań sprzętowych wybranych, dostępnych w MSWP, programowych środków wspomagających oraz zestawienie porównawcze możliwości funkcjonalnych programów narzędziowych z danej klasy zastosowań.

Klasyfikacja narzędziowych środków wspomagających

Środki programowe i programowo-sprzętowe

Podstawowym kryterium klasyfikacji narzędziowych środków jest sposób realizacji tych środków, który determinuje dwie klasy, tj. środki programowe i środki programowo-sprzętowe.

Środki programowe są to programy o specjalistycznym przeznaczeniu, działające w określonym nadrzędnym środowisku programowym, tzn. w środowisku określonego systemu operacyjnego. Specjalistyczne przeznaczenie oznacza ukierunkowanie działania programu narzędziowego na wspomaganie realizacji danego etapu tworzenia oprogramowania. Podstawowymi narzędziowymi środkami programowymi są edytory, translatory, programy łączące i ładujące oraz programy uruchomieniowe działające w środowiskach określonych systemów operacyjnych oraz pomocnicze systemowe programy użytkowe, w które wyposażony jest każdy system operacyjny.

Środki programowo-sprzętowe są to specjalizowane układy wspomagające tworzenie i uruchamianie sprzętu mikroprocesorowego, wyposażone w odpowiednie oprogramowanie sterujące ich działaniem i realizujące komunikację z użytkownikiem. Wspomagające środki sprzętowo-programowe mogą być skonstruowane jako wymienne moduły układowe w nadrzędnym uruchomieniowym systemie mikrokomputerowym lub też mogą stanowić niezależne, małe, specjalizowane systemy mikroprocesorowe. W MSWP środki programowo-sprzętowe zostały zrealizowane jako wymienne pakiety jednostki centralnej MSWP. Środkami tymi są:

- emulator układowy EM-8080 sprzętu mikroprocesorowego opartego na mikroprocesorze typu INTEL 8080 wraz z programem sterującym EMU80 i programem obsługi dyrektyw EM8080,
- symulator pamięci stałych SYM-1 wraz z programem SYMROM obsługi dyrektyw,
- programatory pamięci stałych typu INTEL 1702A, 2704/32, TUNGSRAM TM 601/624 wraz z odpowiednimi programami obsługi dyrektyw,
- uniwersalny tester UMT-1 pakietów z układami TTL wraz z translatorem wejściowego języka zapisu sekwencji testujących JĘZYK TESTERA i programem obsługi testera MONITOR TESTERA.

Klasyfikacja środków programowych

Nadrzędne oprogramowanie zarządzające

Programy narzędziowe wspomagające tworzenie oprogramowania użytkowego działają w środowiskach nadrzędnych programów zarządzających. Podstawowym oprogramowaniem zarządzającym w mikrokomputerowych systemach wspomagających są systemy taśmy papierowej zwane monitorami. Monitor jest zestawem podprogramów inicjujących system, obsługujących urządzenia wejścia/wyjścia, realizujących polecenia /dyrektywy/ zadawane z konsoli systemowej przez użytkownika i pomocniczych, często potrzebnych i wykorzystywanych w programach użytkowników systemu. Urządzeniami wejścia/wyjścia, których obsługę realizuje monitor są:

- monitor ekranowy z klawiaturą,
- drukarka,
- czytnik taśmy papierowej,
- dziurkarka taśmy papierowej,

i ewentualnie

- magnetofon kasetowy.

W MSWP podstawowym systemem zarządzającym jest MONITOR VER.1.8 umieszczony w 2 KB pamięci stałej. Możliwości funkcjonalne monitorów są ograniczone możliwościami funkcjonalnymi sprzętu /urządzeń wejścia/wyjścia taśmy papierowej lub taśmy magnetofonowej/ oraz narzucanymi, z definicji, małymi rozmiarami pamięci stałej, jaką mogą zajmować, aby nie uszczuplać w znacznym stopniu przestrzeni adresowej przeznaczowej na pamięć operacyjną systemu minikomputerowego. Z kolei możliwości funkcjonalne monitorów oraz zestawu sprzętu, który obsługują, ograniczają zakres możliwości programów narzędziowych, takich jak edytory, translatory,ładowanych z taśmy papierowej do PaO i uruchamianych mechanizmami monitora, działających w środowisku monitora.

Rozszerzenie zestawu sprzętu systemu mikrokomputerowego o pamięć na dyskach elastycznych stwarza możliwość implementacji nadrzędnego oprogramowania wykorzystującego zalety zewnętrznej pamięci o stosunkowo dużej pojemności z bezpośrednim dostępem do niej. Tym nadrzędnym oprogramowaniem, współdziałającym z pamięcią na dyskach elastycznych, są dyskowe systemy operacyjne. Działając w systemie mikrokomputerowym z zainstalowanym monitorem, wykorzystują jego funkcje obsługi podstawowego zestawu urządzeń wejścia/wyjścia i ze swojej strony realizują nowe funkcje związane z obsługą pamięci na dyskach elastycznych i obsługą komunikacji z użytkownikiem. Nośnikiem dyskowych systemów operacyjnych są dyski elastyczne. Z dysku elastycznego jest ładowana do pamięci operacyjnej podstawowa część /jądro/ systemu operacyjnego w czasie inicjacji systemu mikrokomputerowego. Jądro systemu, rezydujące w PaO, odpowiedzialne jest za komunikację z użytkownikiem, zarządzanie pamięcią dyskową i zarządzanie urządzeniami wejścia/wyjścia /z wykorzystaniem funkcji monitora/. Realizować ono też może pewien zestaw funkcji użytkowych. Pozostałe funkcje użytkowe realizowane są przez systemowe programy użytkowe, umieszczone na dyskach elastycznych w postaci zbiorów i na czas wykonywania ładowane do pamięci operacyjnej przez jądro systemu.

Wśród systemowych programów użytkowych wyróżnić można grupę programów bezpośrednio wspomagających realizację etapów tworzenia oprogramowania: edycję, translację, scalanie, uruchamianie , czyli programów narzędziowych oraz grupę programów użytkowych pomocniczych o zastosowaniu ogólniejszym, jak np. programy transmisji danych, zmiany atrybutów zbiorów, wyprowadzanie informacji o wolnej przestrzeni na dyskietkach. Dyskowe systemy operacyjne umieszczają dane i programy przede wszystkim w pamięci na dyskach elastycznych w postaci zbiorów, skąd, gdy to potrzebne, ładowane są do PaO, ale nie wykluczają możliwości przesyłania informacji do/z urządzeń wejścia/wyjścia.

Wykorzystanie pamięci na dyskach elastycznych czyni pracę użytkownika efektywną - umożliwia bowiem szybkie przeładowanie do PaO potrzebnych, częstokroć dużych programów i danych.

W MSWP dostępne są dwa dyskowe systemy operacyjne: OS-I VER.1.4 i VER.2.2 /kompatybilny z CP/M VER.1.4 i VER.2.2/ i OS-II VER.2.0 i VER.3.4 /kompatybilny z ISIS-II VER.2.2 i VER.3.4/. Każdy z tych systemów dysponuje zestawem programów narzędziowych omawianych w dalszej części opracowania.

Programy narzędziowe

Programy narzędziowe można podzielić na grupy wg kryterium ich zastosowania, a więc funkcji, jakie pełnią w procesie tworzenia oprogramowania użytkowego. Programy te można zgrupować następująco:

- edytory,
- translatory,
- programy łączące i ładujące,
- programy uruchomieniowe.

Programy narzędziowe umożliwiają tworzenie oprogramowania przeznaczonego dla sprzętu opartego na tym samym typie mikroprocesora, w środowisku którego działają lub też opartego na innym typie mikroprocesora. Ta cecha tworzy podział oprogramowania narzędziowego na:

- oprogramowanie rezydentne,
- oprogramowanie skrośne.

Podział ten ma zastosowanie do translatorów i programów uruchomieniowych.

Głównym kryterium klasyfikacji edytorów jest istnienie lub brak możliwości wizualizacji wskaźnika tekstu.

Pozycja wskaźnika tekstu określa miejsce, w którym są wykonywane operacje na tekście. Kryterium to dzieli edytory na:

- edytory kontekstowe,
- edytory ekranowe.

W edytorach kontekstowych o pozycji wskaźnika tekstu można dowiedzieć się w sposób pośredni /przez kontekst/, wykonując dodatkowe operacje, np. wyświetlanie linii tekstu począwszy od pozycji wskaźnika.

W edytorach ekranowych wskaźnik tekstu jest obrazowany za pomocą kursora monitora ekranowego. Sterując z klawiatury kursorem ustawia się na żądaną pozycję wskaźnik tekstu. W każdej z tych grup edytory mogą się różnić możliwościami funkcjonalnymi. Te możliwości edytorów kontekstowych dostępnych w MSWP zostaną scharakteryzowane dalej.

W grupie translatorów wyróżnić można:

- translatory języków wyższego rzędu: kompilatory, interpretery, zestawy interpreter plus kompilator kodu pośredniego generowanego przez interpreter,
- translatory języków asemblerowych strukturalnych: kompilatory,
- translatory języków asemblerowych: asemblery i makroasemblery.

Translatory języków asemblerowych mogą być wyposażone w mechanizmy interpretacji makroinstrukcji lub mogą tych mechanizmów być pozbawione. Wynika stąd podział na:

- asemblery,
- makroasemblery.

Makroasemblery generują kod wynikowy programu. Kod ten, w zależności od możliwości programu tłumaczącego, może być tworzony w przestrzeni adresów rzeczywistych /absolutnych/ lub adresów względnych /względem adresu bazowego/. Adresacja względna kodu wynikowego umożliwia jego późniejszą relokację na żądane adresy rzeczywiste. To czyni podział na:

- makroasemblery generujące kod wynikowy absolutny,
- makroasemblery generujące kod wynikowy relokowalny.

Programy wykonujące operacje łączenia relokowanych kodów wynikowych modułów programowych i umieszczania ich w przestrzeni rzeczywistych adresów mogą być zrealizowane dwojako. Może to być jeden program łączęco-ładujący lub też dwa programy, tj. łączący i ładujący, działające sekwencyjnie, tzn. program ładujący /lokujący/ przetwarza wynik scalenia modułów wykonanego przez program łączący.

Programy uruchomieniowe - debuggery, umożliwiające testowanie programów oraz wykrycie i lokalizację błędów w nich, są programami o różnych możliwościach funkcjonalnych. W podstawowy zestaw środków uruchomieniowych wyposażone są też monitory /również MONITOR VER.1.8/. Podstawową cechą decydującą o efektywności i pewnym komforcie współpracy z debuggerem jest istnienie w nim aparatu interpretacji wyrażeń symbolicznych, tzn. takich, których elementami są symbole zdefiniowane w programie /symboliczne adresy i zmienne/. Ta możliwość debuggerów może być wykorzystana tylko wtedy, gdy dostępna jest tablica symboli, generowana przez większość translatorów, stanowiąca obok kodu wynikowego programu daną wejściową debuggera.Dostępność w debuggerach aparatu interpretacji wyrażeń symbolicznych lub jego brak klasyfikuje debuggery na:

- proste
- i symboliczne.

W tabl.1 przedstawione są programowe środki wspomagające, dostępne w MSWP, wg omówionej klasyfikacji.

Charakterystyka wybranych programów narzędziowych dostępnych w MSWP

Edytory kontekstowe

Edytory, inaczej zwane programami redagującymi, umożliwiają stworzenie pożądanej postaci dowolnego tekstu /ciągu znaków ASCII/, np. postaci źródłowej programu i umieszczenie go na nośniku - taśmie papierowej lub dysketce. Edytory są programami konwersacyjnymi. Przyjmują, w formie dyrektyw, polecenia wykonania pewnych operacji na tekście i operacje te realizują lub też informują użytkownika o przyczynach ich nie zrealizowania. Przetwarzany tekst lub jego fragmenty, umieszczony jest przez edytor w buforze - obszarze pamięci operacyjnej, będącym polem roboczym edytora. W celu umożliwienia wykonania pewnych operacji na tekście zdefiniowany jest w edytorach kontekstowych umowny wskaźnik tekstu, który może znajdować się w następujących pozycjach:

- przed znakiem,
- za znakiem,
- między znakami.

Wskaźnik tekstu przesuwany jest na żądaną pozycję odpowiednimi dyrektywami. O jego położeniu można dowiedzieć się pośrednio, zlecając edytorowi wykonanie pewnych operacji /np. wyświetl tekst od aktualnej pozycji wskaźnika/. Pozycja wskaźnika tekstu określa miejsce, w którym użytkownik chce wykonać operację na tekście.

Tab.1. Klasyfikacja programowych środków wspomagających dostępnych w MSWP

Nadrzędne oprogramowanie zarządzające		Typ oprogramowania	EDYTORY		TRANSLATORY							PROGRAMY ŁĄCZĄCE I ŁADUJĄCE			DEBUGERY	
			kontekstowe	ekranowe	INTERPRETERY	ZESTAWY KOMPILATOR-INTERPRETER	KOMPILATORY języków asembl. strukt.	KOMPILATORY języków wyższego rzędu	ASEMBLERY	MAKROASEMBLERY generujące kod absol.	MAKROASEMBLERY generujące kod relok.	programy łączącoładujące	program łączący	program ładujący	proste	symboliczne
Monitory	MONITOR VER. 1.8.	rezydentne	EDYTOR 1	-	MINIBASIC	-	-	-	-	REM	-	-	-	-	środki MONITORA VER.1.8	-
		skrośne			-	-	-	-	-	-	-				- -	-
Systemy operacyjne	OS-I VER.1.4 i VER. 2.2	rezydentne	ED	-	INTBAS80	BASNTR+ BASNTC BASRUN3+ BASC3	-	KOMBAS80	ASM	MAK	MAKREL	MLINK	-	-	DDT	DES
		skrośne							-	-	-					
	OS-II VER.2.0 i VER.3.2	rezydentne	EDIT	EDEK	-	-	PLM80	FORTS0 /tylko dla OS-II VER.3.4/	-	-	ASM80	-	LINK	LOCATE	środki MONITORA VER.1.8	-
		skrośne			-	-	-	-	-	-	ASM48 ASM86				-	-

Zasadniczymi operacjami przetwarzającymi tekst mogą być następujące:

- wprowadzenie tekstu do bufora z konsoli systemowej lub nośnika,
- wyprowadzenie tekstu na nośnik lub konsolę systemową,
- usuwanie fragmentów tekstu,
- zamiana określonego fragmentu tekstu na inny,
- zmiana kolejności fragmentów tekstu,
- szukanie określonego fragmentu tekstu.

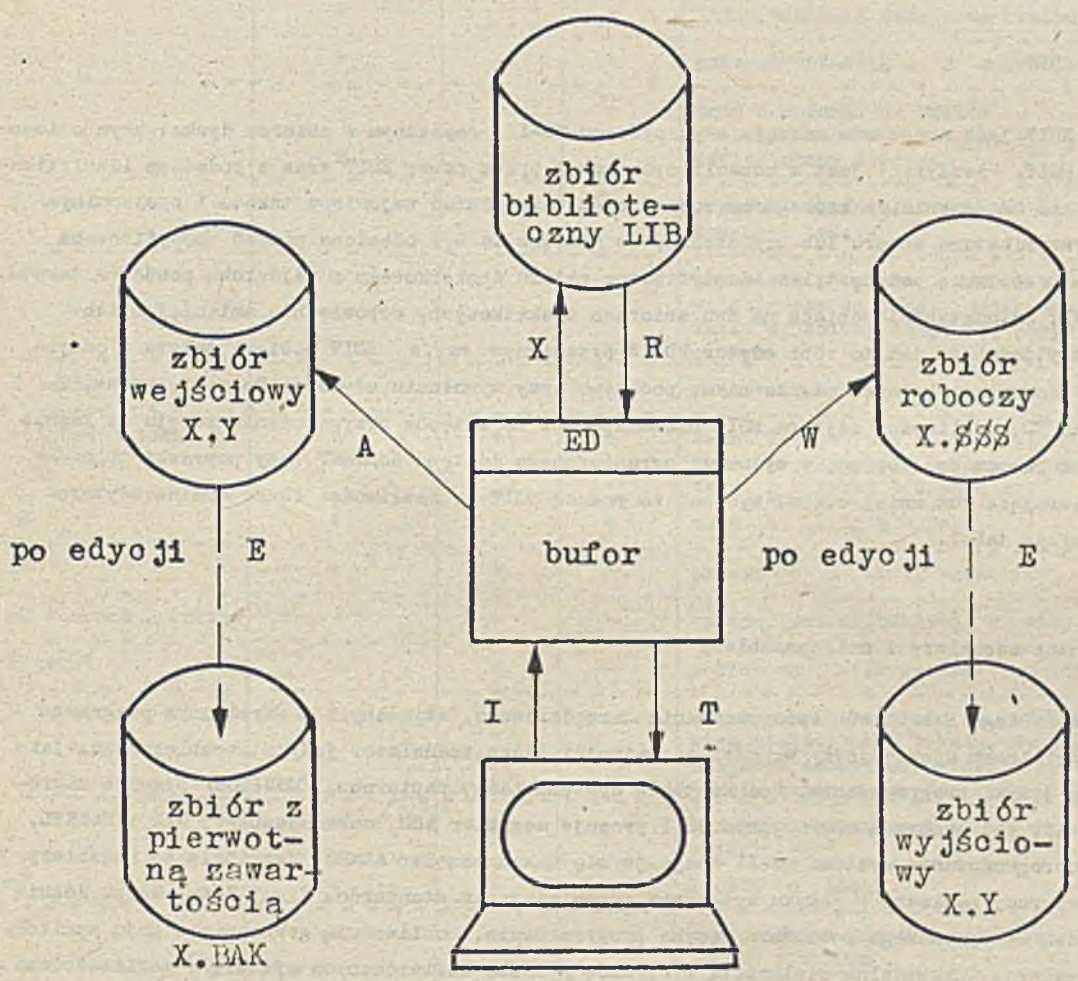
Pomocniczymi operacjami umożliwiającymi lub ułatwiającymi wykonanie operacji zasadniczych, są operacje związane ze zmianą położenia wskaźnika tekstu, mogą też być nimi operacje numeracji linii tekstu w buforze, konwersji małych liter na duże, obliczania rozmiaru wolnego obszaru bufora i in.

Edytor EDYTOR 1

Edytor EDYTOR 1 jest programem działającym pod kontrolą programu MONITOR VER.1.8. Za jego pomocą można tworzyć nowy tekstu wprowadzając go z konsoli, aby następnie, po uzyskanej poprawnej jego postaci, wyprowadzić go na taśmę papierową; można też modyfikować tekst wprowadzony do bufora z taśmy papierowej i po zrobieniu odpowiednich poprawek wyprowadzić go na taśmę. Przegląd możliwości funkcjonalnych tego edytora obrazuje tab.2, w której podano zbiór dyrektyw akceptowanych przez niego. EDYTOR 1 jest zapisany na taśmie papierowej, z której jest ładowany do pamięci operacyjnej MSWP i startowany odpowiednimi dyrektywymi MONITORA.

Edytor ED

Edytor ED jest programem narzędziowym systemu OS-I, zapisanym w zbiorze dyskietkowym o identyfikatorze ED.COM. Wywoływany jest z konsoli systemowej przez nazwę ED wraz z podaniem identyfikatora zbioru, zawierającego tekst przeznaczony do modyfikacji lub zbioru pustego, do którego zostanie przesłany tekst stworzony w czasie sesji edycyjnej. Po zakończeniu edycji dostępne są dwa zbiory /z wyjątkiem przypadku kończenia edycji dyrektywą Q/: jeden zawierający tekst,będący wynikiem edycji, drugi zawierający tekst wejściowy, modyfikowany w czasie edycji lub będący zbiorem pustym, jeżeli tekst powstał dopiero w czasie edycji. Identyfikatory tych zbiorów różnią się typami /nazwy pozostają nie zmienione/ - zbiór zawierający postać wejściową tekstu otrzymuje po edycji zawsze typ BAK, zbiór z aktualną postacią tekstu otrzymuje typ taki, jaki miał przed edycją zbiór z tekstem wejściowym. Ta automatyczna zmiana identyfikatorów, wykonywana przez edytor, ułatwia pracę użytkownikowi umożliwiając mu wykonywanie ponownych edycji kolejnych postaci tekstu przy nie zmienionym identyfikatorze zbioru przechowującego ostatnią postać tekstu. Nośnikiem obu postaci tekstu /pierwotnej i zmodyfikowanej/ w tym edytorze jest dyskietka lub dwie dyskietki /każda z dwu postaci tekstu na innej dyskietce/. Istnieje możliwość założenia w czasie edycji, zbioru bibliotecznego z przesłanym do niego przez użytkownika fragmentem tekstu, aby w czasie tej samej edycji /zbiór biblioteczny jest przez edytor na końcu sesji likwidowany, jeżeli nie zrobi tego użytkownik/ móc ten fragment wkomponować w dowolne miejsce tekstu. Istnieje też możliwość wprowadzania w tekst zawartości zbiorów bibliotecznych, stworzonych przez użytkownika przed przystąpieniem do sesji edycyjnej. Zbiory takie nie są likwidowane przez edytor.



I,T,A,E,R,W,X - dyrektywy edytora ED

Rys. 2. Proces edycji realizowany za pomocą edytora ED

Proces edycji realizowany za pomocą edytora ED przedstawia rys.2. Możliwości funkcjonalne tego edytora reprezentuje zestaw jego dyrektyw podany w tab.2. Jest to najsilniejszy z trzech omawianych edytorów. Warto zwrócić uwagę na takie dyrektywy, jak N, R, X, V, n, n::m oraz dyrektywy O, H, Q, E w różny sposób kończące edycję.

Edytor EDIT

Edytor EDIT jest programem narzędziowym systemu OS-II, zapisanym w zbiorze dysketkowym o identyfikatorze EDIT. Wywoływany jest z konsoli systemowej przez nazwę EDIT wraz z podaniem identyfikatora zbioru lub urządzenia, z którego ma zostać odczytana postać wejściowa tekstu i opcjonalnym podaniem identyfikatora zbioru lub urządzenia, do którego ma być odesłana postać zmodyfikowana tekstu. Jeżeli zostanie podany tylko identyfikator zbioru dysketkowego z wejściową postacią tekstu, to edytor EDIT automatycznie działa na dwu zbiorach dysketkowych, odpowiednio zmieniając ich identyfikatory,podobnie jak to robi edytor ED. W przeciwnym razie EDIT pobiera tekstu i odsyła tekst zmodyfikowany wg wskazań użytkownika, podanych przy wywołaniu edytora. Te rozszerzone, w stosunku do ED, możliwości edytora EDIT pozwalają np. na wstępne przygotowanie tekstu na taśmie papierowej za pomocą dalekopisu, w sytuacji ograniczonego dostępu do MSWP, aby poprawki tego tekstu, zabierające już mniej czasu, wykonać za pomocą EDIT-u. Możliwości funkcjonalne edytora EDIT demonstruje tab.2.

Rezydentne asemblery i makroasemblery

W skład każdego z zestawów oprogramowania narzędziowego, związanych z określonym programem nadzorczym /systemem operacyjnym/ wchodzi co najmniej jeden translator języka asemblerowego, jako podstawowego języka programowania. Pod kontrolą systemu taśmy papierowej /MONITOR/ pracuje makroasembler REMAK, pod systemem operacyjnym OS-I pracuje asembler ASM, makroasemblery MAK i MAKREL, w zestawie oprogramowania systemu OS-II znajduje się makroasembler ASM80. Wszystkie te asemblery akceptują programy napisane w języku źródłowym odpowiadającym standardom firmy Intel Corp. Różnią się wielkością akceptowanego podzbioru języka programowania, możliwością generowania kodu wynikowego relokowalnego, maksymalną wielkością programów /liczba definiowanych symboli/, możliwościami operatorskimi, zestawem opcjonalnych środków pomocniczych. Szczegółowe charakterystyki tych programów podane są w tab.3.

Makroasembler REM

Makroasembler REM pracuje pod nadzorem programu sterującego MONITOR. Akceptuje język źródłowy zdefiniowany przez firmę Intel Corp. w 1974 r.

Program źródłowy wprowadza się z taśmy papierowej lub z bufora w pamięci operacyjnej /jeżeli może się w nim w całości zmieścić/. Istnieje wersja REM-a sprzężona z edytorem EDYTOR1 i współpracująca z nim przez wspólny bufor, w którym program źródłowy może być utworzony i ewentualnie poprawiany za pomocą edytora, a następnie pobierany do translacji przez asembler. Ten sprzężony wariant jest bardzo wygodny w użyciu, umożliwia bowiem stworzenie poprawnego formalnie programu źródłowego i przetranslowanie go bez posługiwania się taśmą papierową, ma jednak ograniczone zastosowanie - nadaje się tylko dla stosunkowo niewielkich programów.

Tabl.2 Zestawy dyrektyw edytorów: EDYTOR 1, ED, EDIT

Dyrektywy	EDYTOR 1	ED	EDIT	Opis działania dyrektyw
nA	+	+	+	wprowadzić n linii tekstu wejściowego
B	+	+	+	ustaw wskaźnik tekstu na krańcu bufora
nC	+	+	+	przesuń wskaźnik o n znaków
nD	+	+	+	usuń z bufora n znaków
E	+	+	+	koniec pracy edytora
F tekst	+	+	+	przesuń wskaźnik za ostatni znak zadanego tekstu
H	-	+	-	przepisanie bufora i reszty tekstu wejściowego do tekstu wyjściowego. Tekst wyjściowy staje się nowym wejściowym.
I tekst	+	+	+	wstaw zadany tekst do bufora
J tekst 1 tekst 2 tekst 3	-	+	-	tekst 2 wstawiony za znalezionym tekst 1. Zawartość bufora między tekst 2 i tekst 3 usunięta
nK	+	+	+	usuń n linii z bufora
nL	+	+	+	przesuń o n linii wskaźnik
nM łańcuch dyrektyw	-	+	-	wykonaj makrodyrektywę n razy
N tekst	-	+	-	perforowanie 128 dziurek prowadzących
N tekst	-	+	-	ustaw wskaźnik tekstu za ostatnim znakiem znalezionego tekstu. Szukanie obejmuje zawartość bufora i resztę tekstu wejściowego /na dysketce/
O	-	+	-	zerowanie zawartości bufora i tekstu wyjściowego
nP	+	-	-	listuj n linii
nP	-	+	-	wyprowadź na konsolę systemową n stron tekstu z bufora
Q	-	+	+	koniec pracy edytora. Zeruj bufor i tekst wyjściowy
R nazwa zbioru R	-	+	-	wprowadź tekst z biblioteki do bufora w miejsce wskazane przez wskaźnik bufora
S tekst 1 tekst 2	+	+	+	zamień tekst 1 tekstem 2
nT	+	+	+	wyprowadź na konsolę systemową n linii tekstu z bufora
U	-	+	-	zamieniaj małe litery na duże
nW	+	+	+	wyprowadź do tekstu wyjściowego n początkowych linii z bufora i skasuj je w buforze
V ØV	-	+	+	włącz numerację linii, lub gdy ØV podaj wielkość wolnego obszaru bufora
nX	+	-	-	wyperforuj n linii z bufora /począwszy od pozycji określonej przez wskaźnik tekstu/

Tab.2 c.d.

Dyrektywy	EDYTOR 1	ED	EDIT	Opis działania dyrektywy
nX nX nazwa zbioru	-	+	-	przenieś n linii tekstu do zbioru bibliotecznego
Z	+	-	+	ustaw wskaźnik tekstu za ostatnim znakiem w buforze
n	-	+	-	przesuń wskaźnik o n linii i wyświetl linię, przed którą stoi
m:	-	+	-	ustaw wskaźnik na początku linii o numerze m
m_1::m_2 dyrektywa	-	+	-	wykonaj operację określoną dyrektywą na liniach od numeru m_1 do m_2

n - liczby całkowite dodatnie /bez znaku/ lub ujemne

m, m_1, m_2 - liczby całkowite dodatnie bez znaku.

Produktami REM-a są: wydruk /listing/ wyprowadzany na urządzenie listujące, słownik symboli wypisywany jako część listingu i/lub wyprowadzany na taśmę papierową oraz absolutny kod wynikowy w standardowym, intelowskim formacie szesnastkowym /HEX/ wyprowadzany na taśmę papierową.

REM interakcyjnie współpracuje z użytkownikiem, dając do jego dyspozycji kilka opcji operatorskich. Daje to maksimum wygody użytkowania przy ograniczonych możliwościach systemu taśmy papierowej.

Asembler ASM

Asembler ASM jest to prosty, szybko działający asembler działający pod systemem operacyjnym OS-I.

Program źródłowy musi być zapisany w zbiorze dyskietkowym /typu ASM/. Produktami asemblera są: wydruk /zbiór typu .PRN/ i kod wynikowy w standardowym formacie szesnastkowym /zbiór typu .HEX/. Wydruk można wprowadzać do zbioru dyskietkowego, na konsolę /z ewentualnym kopiowaniem na drukarkę/ lub do zbioru pustego /nie wyprowadzać w ogóle/. Istnieje możliwość blokowania tworzenia kodu wynikowego /asemblacja diagnostyczna/.

Asembler ASM pomimo ograniczonych możliwości może być wygodny zwłaszcza w małych konfiguracjach MSWP.

Makroasembler MAK

Makroasembler MAK pracuje pod systemem operacyjnym OS-I. Akceptuje on język źródłowy zgodny ze standardem firmy Intel Corp. z 1977 r. oprócz specyficznych środków związanych z mechanizmami relokacji i programowania modularnego.

Tab.3. Dane porównawcze makroasemblerów rezydentnych dostępnych w MSWP

Parametr	REM	ASM	MAK	MAKREL	ASMSO	Uwagi
1	2	3	4	5	6	7
System operacyjny	MONITOR	OS-I	OS-I	OS-I	OS-II	
Nośnik programu źródłowego	taśma papier. bufor PAO	zbiór dysketkowy	zbiór dysketkowy	zbiór dysketkowy	zbiór dysketkowy, urządz. we/wy	
Produkty asemblera						
listing	jest	jest	jest	jest	jest	
kod wynikowy	hex-absol.	hex, absol.	hex-absol.	relokowalny kod wynikowy	relokowalny kod wynikowy	
tablica symboli	jest	nie ma	jest	jest	jest	
tablica odwołań /cross-reference/	nie ma	nie ma	nie ma	jest	jest	
Identyfikatory zmiennych /nazwy/: długość maksymalna /w znakach/	5	16	16	6		
Stałe:						
dziesiętne - D	tak	tak	tak	tak	tak	
binarne - B	tak	tak	tak	tak	tak	
szesnastkowe - H	tak	tak	tak	tak	tak	
oktalne - Q	tak	tak	tak	tak	tak	
znakowe	tak	tak	tak	tak	tak	
Wyrażenia /zestaw operatorów/	+	+	+	+	+	plus jedno- i dwuargumentowy
	-	-	-	-	-	minus jedno- i dwuargumentowy
	*	*	*	*	*	mnożenie
	/	/	/	/	/	dzielenie całkowite
	MOD	MOD	MOD	MOD	MOD	reszta z dzielenia
	NOT	NOT	NOT	NOT	NOT	operacja logiczna NIE
	AND	AND	AND	AND	AND	operacja logiczna I
	OR	OR	OR	OR	OR	operacja logiczna LUB
	XOR	XOR	XOR	XOR	XOR	operacja logiczna ALBO
	SHL	SHL	SHL	SHL	SHL	przesunięcie w lewo
	SHR	SHR	SHR	SHR	SHR	przesunięcie w prawo
			HIGH	HIGH	HIGH	bardziej znaczący bajt
			LOW	LOW	LOW	mniej znaczący bajt
			EQ,=	EQ	EQ,=	relacja równości
			LT,<	LT	LT,<	relacja mniejszości

Tab.3. c.d.

1	2	3	4	5	6	7
			LE, = GT, >· GE, =	LE GT GE TYPE	LE, = GT, >· GE, =	relacja mniejsze lub równe relacja większości relacja większe lub równe określenie typu argumentu
Zestaw dyrektyw	ORG	ORG	ORG	ORG	ORG	ustawienie licznika rozmieszczenia
	END	END	END	END	END	koniec programu
	EQU	EQU	EQU	EQU	EQU	definicja stałej
	SET	SET	SET	SET,DEFL, ASET	SET	nadanie wartości zmiennej czasu translacji
	IF	IF	IF	IF,IFT,COND	IF	początek translacji warunkowej /wartość warunku różna od zera/
				IFE, IFF		początek translacji warunkowej /wartość warunku równa zeru/
				IF1		początek translacji warunkowej /translacja, gdy pierwszy przebieg/
				IF2		początek translacji warunkowej /translacja, gdy drugi przebieg/
				IFDEF, IFNDEF		początek translacji warunkowej /warunek ustalony przez typ symbolu/
				IFB, IFNB, IFIDN,IFDIF		początek translacji warunkowej /gdy odpowiedni format warunku/
	ENDIF	ENDIF	ENDIF	ENDIF,ENDC	ENDIF	koniec translacji warunkowej
	DB	DB	DB	DB	DB	deklaruj bajt
	DW	DW	DW	DW	DW	deklaruj słowo
	DS	DS	DS	DS	DS	deklaruj obszar
				DC		deklaruj łańcuch znaków
	TITLE	-	TITLE	TITLE	TITLE	nadaj tytuł programowi
				SUBTTL, STITLE		nadaj podtytuł programowi
	MACRO		MACRO	MACRO	MACRO	początek makrodefinicji
	ENDM	-	ENDM	ENDM	ENDM	koniec makrodefinicji
	-	-	EXITM	EXITM	EXITM	zatrzymanie procesu rozwinięcia makrodefinicji
	-	-	ELSE	ELSE	ELSE	alternatywa w translacji warunkowej

c.d. Tab.3

1	2	3	4	5	6	7
	-	-	PAGE	PAGE,EJECT,	EJECT	nowa strona wydruku
	-x/	-	-	EJECT	EOT	koniec nośnika x/ w REMAK realizowane automatycznie: dyrektywa END jest obowiązkowa
				ASEG	ASEG	deklaracja segmentu absolutnego
				CSEG	CSEG	relokowalny segment rozkazowy
				DSEG	DSEG	relokowalny segment danych
				COMMON		deklaracja wspólnego obszaru
				PUBLIC,GLOBAL ENTRY	PUBLIC	deklaracja zmiennych globalnych
				EXT, EXTRN, EXTERNAL	EXTRN	deklaracja zmiennych zewnętrznych
				BYTE EXT, BYTE EXTRN, BYTE EXTERNAL		deklaracja zmiennych zewnętrznych 1-bajtowych
				. COMMENT		początek długiego komentarza
				. RADIX		ustawienie podstawy liczb
				. REQUEST		określenie zbiorów, w których są zdefiniowane symbole zewnętrzne
				. PHASE		początek fragmentu programu wykonywanego w innym obszarze
				. DEPHASE		koniec fragmentu programu wykonywanego w innym obszarze
				. PRINTX		wyprowadzenie podczas asemblacji zadanego tekstu na konsolę
				. Z80		asemblacja instrukcji mikroprocesora Z80
				. 8080		asemblacja instrukcji mikroprocesora I 8080
					STKLN	deklaracja wielkości stosu
				NAME	NAME	definicja nazwy modułu
Możliwość dołączenia biblioteki tekstów źródłowych	nie ma	nie ma	dyrektywa MACLIB	dyrektywy INCLUDE,SINCLUDE, MACLIB	dyrektywa INCLUDE	w MAK można dołączać tylko makrodefinicje
Makrodefinicje wbudowane w asembler	brak	brak	REPT IRP IRPC	REPT IRP IRPC	REPT IRP IRPC	iteracja /powtórzenie tekstu/ iteracja z podstawieniem tekstów iteracja z podstawieniem znaków
Biblioteki źródłowe	brak	brak	są x/	brak	brak	x/zob. pkt "Macroasembler MAK"

Program źródłowy musi być umieszczony w zbiorze dyskietkowym. Produktami asemblera są absolutny kod wynikowy w formacie szesnastkowym, wydruk /listing/ i słownik symboli. Istnieje możliwość elastycznego sterowania zawartością wydruku i miejscem lokowania produktów asemblera /zbiór dyskietkowy, ekran konsoli, drukarka/.

Charakterystyczną cechą makroasemblera MAK jest możliwość korzystania z bibliotek makrodefinicji. Upraszcza to proces kodowania i zwiększa niezawodność programów. Biblioteki te mogą być tworzone przez użytkownika i dowolnie rozszerzane.

Wraz z makroasemblerem MAK dostarczone są biblioteki przeznaczone do trzech podstawowych celów:

- realizacja elementarnych konstrukcji programowych, tj. badania warunków i operatorów programowania strukturalnego,
- współpracy z systemem operacyjnym OS-I,
- asemblacji skrośnej.

Biblioteka COMPARE.LIB zawiera makrooperacje porównywania liczb ośmiobitowych traktowanych jako całkowite bez znaku. Definiuje następujące relacje: LSS /mniejsze/, LEQ /mniejsze lub równe/, EQL /równe/, NEQ /różne/, GEQ /większe lub równe/, GTR /większe/. Wszystkie makrosy tej biblioteki mają identyczną strukturę argumentów, np. makrorozkaz LSS X, Y, ETYKIETA powoduje przejście do adresu ETYKIETA, jeżeli X < Y, a do następnego rozkazu, jeżeli X ≥ Y. Pierwszy argument /X/ może być adresem, wyrażeniem adresowym lub być pusty. W tym ostatnim przypadku drugi argument porównywany jest z zawartością akumulatora. Drugi argument może być wyrażeniem adresowym lub konstantą.

Biblioteka NCOMPARE.LIB realizuje takie same operacje porównywania, jak COMPARE.LIB z tym, że w zależności od sposobu zadania argumentów aktualnych skok pod wyspecyfikowany adres zachodzi albo wtedy, gdy realacja jest prawdziwa, albo wtedy, gdy jest fałszywa.

Biblioteka WHEN.LIB realizuje zagnieżdżone konstrukcje wykonania warunkowego /IF...THEN/ w programowaniu strukturalnym. Konstrukcje te tworzone są z dwóch makrooperacji WHEN oraz ENDW i mają postać:

```
WHEN warunek
   .
   .
   .
   WHEN warunek
   .
   .
   .
   ENDW
   .
   .
   .
ENDW
```

Warunek jest listą trzech parametrów: identyfikator, nazwa relacji, identyfikator, np. WHEN X, LSS, Y oznacza, że odcinek programu, aż do odpowiedniego makrorozkazu ENDW będzie wykonany tylko wtedy, gdy X < Y.

Biblioteka DOWHILE.LIB służy do realizacji pętli programowych /dowolnie zagnieżdżonych/. W jej skład wchodzą dwa makrorozkazy: DOWHILE i ENDDO. Można z nich tworzyć konstrukcje typu:

```
DOWHILE warunek
    .
    .
    .
    DOWHILE warunek
        .
        .
        .
    ENDDO
        .
        .
        .
ENDDO
```

Warunek ma identyczną postać, jak w przypadku biblioteki WHEN.LIB. Znaczenie tej konstrukcji można opisać następująco:

Jeżeli warunek jest prawdziwy, wykonywana jest sekwencja rozkazów, aż do odpowiedniego makrorozkazu ENDDO, po czym następuje ponowne sprawdzenie warunku i, jeżeli jest nadal prawdziwy, wspomniana sekwencja rozkazów jest wykonywana powtórnie, itd. Jeżeli warunek nie jest spełniony, wykonywany jest rozkaz następny po makro-wywołaniu ENDDO.

Biblioteka SELECT.LIB realizuje konstrukcję wielokrotnego rozgałęzienia /operator CASE w programowaniu strukturalnym/. Składa się ona z trzech makrorozkazów SELECT, SELNEXT, ENDSEL. Za ich pomocą można tworzyć konstrukcje typu:

```
SELECT identyfikator
    .
    .
    .
        sekwencja-Ø
    .
    .
    .
SELNEXT
    .
    .
    .
        sekwencja-1
    .
    .
    .
SELNEXT
    .
    .
    .
        sekwencja-n
ENDSEL
```

Jeżeli wartość związana z identyfikatorem równa jest k /k=0,1,...,n/, to wykonywana jest k-ta sekwencja rozkazów zawarta między wywołaniami SELECT i SELNEXT, SELNEXT i SELNEXT lub SELNEXT i ENDSEL. Pozostałe sekwencje są pomijane.

Biblioteka SIMPIO.LIB realizuje proste operacje komunikacji z konsolą systemową.

Biblioteka SEQIO.LIB realizuje sekwencyjny dostęp do zbiorów OS-I. Użycie makrorozkazów z tej biblioteki pozwala jednorodnie traktować w programie zbiory dyskietkowe i urządzenia zewnętrzne

tak, jak znakowe urządzenia wejścia/wyjścia, bez konieczności programowania buforowania informacji i bezpośredniego korzystania z funkcji systemowych OS-I. W skład biblioteki wchodzą następujące - makrorozkazy:

- FILE - jego argumenty określają nazwę zewnętrzną i wewnętrzną /w danym programie/ zbioru, sposób dostępu i buforowania; otwiera zbiór
- GET - czyta znak ze zbioru lub urządzenia wejścia/wyjścia do akumulatora
- PUT - wypisuje znak z akumulatora do zbioru lub na urządzenie wejścia/wyjścia
- FINIS - opróżnia bufory i zamyka zbiory
- ERASE - likwiduje zbiór
- DIRECT - sprawdza, czy zbiór znajduje się na dysketce
- RENAME - zmienia nazwę zbioru.

Biblioteka I8085.LIB umożliwia asemblację skrośną programów napisanych dla mikroprocesora Z80.

Makroasembler MAKREL

Makroasembler MAKREL pracuje pod systemem operacyjnym OS-I. Akceptuje język źródłowy firmy Intel dla mikroprocesora I8080 oraz język źródłowy firmy Zilog dla mikroprocesora Z80. Ukierunkowany jest na programowanie modularne.

Program źródłowy musi być umieszczony w zbiorze dysketkowym. Produktami makroasemblera MAKREL jest wydruk /listing/ ze słownikiem symboli i opcjonalnym opisem odwołań do nich oraz relokowalny kod wynikowy. Miejscami przesłania produktów makroasemblera mogą być zbiory dysketkowe lub urządzenia zewnętrzne. Istnieje możliwość sterowania pracą makroasemblera dotyczącego zawartości i nazw produktów wyjściowych oraz sposobu pracy /akceptacja języka I8080 lub Z80/.

Moduł wynikowy zawierający relokowalny kod ze słownikiem symboli globalnych i zewnętrznych może być łączony /za pomocą programu łącząco-ładującego/ z innymi modułami uprzednio zasemblowanymi. Złączone moduły mogą być umieszczone na rzeczywistych adresach w pamięci operacyjnej i wykonane lub też mogą być zapisane w zbiorze dysketkowym w formie wykonywalnej spełniającej wymogi systemu OS-I. Możliwe jest sterowanie rozmieszczeniem na rzeczywistych adresach wydzielonych segmentów /segmenty relokowalnego kodu, relokowalnych danych, wspólnych obszarów COMMON/ modułów wynikowych. W zestawie elementów języka akceptowanego przez MAKREL są elementy umożliwiające deklarację typów segmentów programu źródłowego oraz symboli zewnętrznych i globalnych, a więc umożliwiające programowanie modularne.

Makroasembler ASM80

Makroasembler pracuje pod systemem operacyjnym OS-II. Akceptuje pełen język źródłowy według standardu firmy Intel z roku 1977. Ukierunkowany jest na programowanie modularne.

Produktami makroasemblera ASM80 są: wydruk /listing/, słownik symboli, tablica odwołań do symboli /miejsca definicji symbolu i wszystkich odwołań do niego/ oraz relokowalny kod wynikowy.

Moduł wynikowy zawierający kod relokowalny może być łączony /linkowany/ z innymi modułami przygotowanymi wcześniej. Umożliwia to programowanie modularne. ASM80 jest wyposażony w środki do definiowania symboli globalnych, dostępnych dla innych modułów /dyrektywa PUBLICS/ i zewnętrznych, to jest takich, które są zdefiniowanie w innych modułach /dyrektywa EXTRN/.

Istnieje możliwość elastycznego sterowania sposobem relokacji. Między innymi można podzielić moduł na tzw. segmenty, czyli części modułu, które mogą być relokowane niezależnie, ze względu na rodzaj pamięci, w których mają być umieszczone /stałe czy zapisywalne/.

Z operatorskiego punktu widzenia ASM80 ma również liczne udogodnienia, np. możliwość sterowania zawartością zbiorów wynikowych, wyboru sposobu pracy asemblera, itp.

W przeciwieństwie do makroasemblera MAK makrodefinicje nie są pamiętane w słosniku symboli, ale w roboczych zbiorach dyskietkowych. Oszczędza to wiele miejsca w tablicy symboli i umożliwia translację bardzo dużych modułów.

Informacja diagnostyczna wydawana przez ASM80 jest pełna i bardzo dokładna.

Programy uruchomieniowe

Tradycyjne, "klasyczne" metody uruchamiania programów polegają na uzupełnianiu programu operacjami wyprowadzania dodatkowych informacji kontrolnych. Otrzymuje się wtedy dużą ilość informacji, którą bardzo trudno jest analizować. W dodatku jest trudno, a nawet nie można przewidzieć, jakie informacje będą potrzebne. W rezultacie proces uruchamiania jest kłopotliwy, czasochłonny i nieekonomiczny.

W specjalnych systemach wspomagania projektowania, takich jak MSWP, przeznaczonych do produkcji oprogramowania, a nie jego eksploatacji, takie metody, o jakich wspomniano wyżej są w ogóle nie do przyjęcia. Przede wszystkim dlatego, że rozbudowując uruchamiany program o środki kontrolno-diagnostyczne traci się całkowicie kontrolę nad takimi parametrami programu, jak wymagana /rzeczywiście/ wielkość pamięci, czy czas wykonania, co jest niezwykle istotne w urządzeniach mikroprocesorowych. Dlatego też stosuje się metody dynamiczne - interakcyjne. Służą do tego specjalne środki programowe, tzw. debuggery, pod których kontrolą można testować i korygować programy użytkownika. Wspomniane środki muszą umożliwiać:

- załadowanie do pamięci operatorskiej jednego lub kilku modułów programowych,
- zadanie warunków początkowych dla pracy programu: stanu miejsca pamięci i rejestrów procesora,
- wykonanie dowolnej sekwencji rozkazów testowanego modułu, tj. pojedynczego rozkazu, podprogramu, kilku rozkazów, a następnie sprawdzenia efektu jej wykonania,
- wydawanie w możliwie komunikatywnej formie informacji o stanie procesora /rejestrów i wskaźników/ i pamięci,
- wprowadzanie na bieżąco poprawek do uruchamianego programu /zwykle są to uzupełnienia/ i kontrolę wprowadzanych zmian,
- zachowanie aktualnego stanu programu po wprowadzeniu poprawek.

Pożądane są również bardziej wyrafinowane środki umożliwiające np. lokalizację błędów, wspomaganie optymalizacji programu, itp.

W systemie MSWP dostępne są następujące środki wspomagające uruchamianie programów:

- program MONITOR wykorzystywany do uruchamiania programów pracujących pod MONITOR-em, a także programów wynikowych pracujących pod OS-II,
- programy DDT i DES wspomagające uruchamianie programów pracujących pod OS-I.

Zestawy dyrektyw akceptowanych przez MONITOR /dyrektywy uruchomieniowe/ DDT i DES prezentuje tab.4.

Uruchamianie programów użytkowych za pomocą MONITOR-a

Program MONITOR daje do dyspozycji użytkownika pewien minimalny zestaw środków /dyrektyw MONITOR-a/ ułatwiających uruchamianie jego programów.

Program użytkowy w postaci wynikowej zapisany na taśmie papierowej w formacie szesnastkowym wprowadzany jest do pamięci operacyjnej pod właściwe adresy /dyrektywa R/.

Istnieje możliwość wyświetlania zawartości miejsc pamięci /szesnastkowo/ i ustawienia ich zawartości /dyrektywy S i D/. Stan wskaźników i rejestrów procesora również może być wyświetlony i zmieniany /dyrektywa X/.

Program użytkowy może być uruchomiony dyrektywą G od adresu zadanego lub wskazanego przez licznik rozkazów z jednoczesnym zadaniem jednego lub dwóch punktów zatrzymań /ang. break-point/. Po natrafieniu na punkt zatrzymania sterowanie przechodzi do programu MONITOR, co umożliwia sprawdzenie zawartości rejestrów i miejsc roboczych w pamięci operacyjnej. Deklaracje punktów zatrzymań są odwoływane automatycznie w momencie realizacji zatrzymania.

Modyfikacja i uzupełnienie uruchamianego programu może się odbywać jedynie na poziomie binarnym za pomocą dyrektywy S zmieniającej zawartość miejsc pamięci.

Zmodyfikowany program użytkownika może być wyprowadzony na taśmę papierową w formacie szesnastkowym za pomocą dyrektywy W, a poprawność wyprowadzonej kopii może być sprawdzona dyrektywą T.

Środki uruchomieniowe programu MONITOR są również wykorzystywane do testowania i poprawiania programów przygotowywanych za pomocą systemu operacyjnego OS-II. Dyrektywa DEBUG tego systemu operacyjnego wprowadza program użytkownika do pamięci operacyjnej i przekazuje sterowanie do programu MONITOR. Dalsze postępowanie jest takie, jak w przypadku programów zapisanych na taśmie papierowej.

Program uruchomieniowy /debugger/ DDT

Program uruchomieniowy DDT służy do interakcyjnego testowania i poprawiania programów przeznaczonych do pracy pod kontrolą systemu operacyjnego OS-I.

Za pomocą programu DDT można uruchamiać programy użytkowe zapisane w zbiorach dyskietkowych w postaci szesnastkowej bądź w postaci binarnej. Najprostszym sposobem wprowadzania testowanego programu jest umieszczenie jako parametru wywołania debuggera nazwy zbioru zawierającego program użytkowy. Wyspecyfikowany program jest wtedy automatycznie ładowany do systemowego obszaru roboczego w pamięci operacyjnej. Jeżeli zachodzi potrzeba powtórnego wprowadzenia programu lub, jeśli program użytkowy składa się z kilku modułów zapisanych w różnych zbiorach, to załadowania uruchamianego modułu można dokonać przez wyspecyfikowanie nazwy odpowiedniego zbioru /dyrektywa I/ i wczytanie informacji za pomocą dyrektywy R.

Dyrektywa I, ustawiająca systemowe pola sterujące, może być też użyta do symulowania warunków początkowych dla programu użytkowego, który ma być inicjowany z parametrem będącym nazwą zbioru.

Stan rejestrów i wskaźników procesora może być wyświetlony i zmieniony za pomocą dyrektywy X. Tak jak i w przypadku innych dyrektyw, informacja wyświetlana przez debugger, jak i wprowadzana przez użytkownika przedstawiona jest w zapisie szesnastkowym.

Stan pamięci /poszczególnych komórek/ może być wyświetlany i zmieniany dyrektywą S.

Wyświetlenie zawartości obszaru pamięci umożliwia dyrektywa D, która oprócz zawartości w zapisie szesnastkowym podaje zawartość obszaru interpretowaną jako ciąg znaków w kodzie ASCII.

Do modyfikacji i sprawdzania stanu uruchamianego programu służy para dyrektyw asemblacji /A/ i desasemblacji /L/. Umożliwiają one wyświetlanie tekstu programu w oznaczeniach mnemonicznych z adresami szesnastkowymi, a także translację programu wprowadzanego w tej samej formie i umieszczanie postaci binarnej w pamięci operacyjnej.

Możliwe jest również przemieszczanie obszarów pamięci operacyjnej /kopiowanie zawartości/ za pomocą dyrektyw M, a także zapełnianie obszaru pamięci zawartością zadaną /dyrektywa F/.

W programie uruchomieniowym DDT istnieją trzy możliwości uruchamiania testowanego programu.

Dyrektywa G umożliwia uruchomienie programu od adresu zadanego lub wskazanego przez licznik rozkazów z jednoczesnym zadaniem jednego lub dwóch zatrzymań kontrolnych /break-point/. Program uruchamiany dyrektywa G wykonuje się w rzeczywistym tempie /sterowanie jest rzeczywiście przekazywane programowi testowanemu/.

Dyrektywa U pozwala wykonać zadaną liczbę instrukcji, a dyrektywa T powoduje wykonywanie programu krok po kroku z jednoczesnym wyświetlaniem stanu rejestrów po wykonaniu każdej instrukcji. Program uruchamiany dyrektywami U i T wykonywany jest interpretacyjnie - kilkaset razy wolniej niż w warunkach rzeczywistych. Wynika stąd, że dyrektyw tych nie można stosować do testowania fragmentów programów, które są uwarunkowane czasowo.

Symboliczny debugger DES

Program uruchomieniowy DES, podobnie jak DDT, przeznaczony jest do testowania programów dostosowanych do pracy pod kontrolą systemu operacyjnego OS-I. Stanowi on rozwinięcie debuggera DDT. Oznacza to, że akceptuje on wszystkie dyrektywy DDT. Ma on jednak kilka dodatkowych możliwości i bogatsze sposoby określenia parametrów, nowe dyrektywy, rozszerzenia funkcjonalne.

Podstawową cechą debuggera DES jest możliwość operowania na symbolach. Wraz z programem uruchamianym może być wprowadzony do pamięci operacyjnej słownik symboli. Słownik ten jest generowany przez makroasemblery MAK i MAKREL, ale może być również przygotowany niezależnie za pomocą edytora. DES umożliwia zadawanie parametrów dyrektyw w postaci wyrażeń zbudowanych z nazw i stałych. Stałe liczbowe mogą być zadawane w formie szesnastkowej lub dziesiętnej. Można również zadawać stałe znakowe. Wyrażenia buduje się za pomocą operacji arytmetycznych sumy i różnicy oraz operatorów działających na symbolach: wzięcia adresu związanego z symbolem /operacja "kropka"/ lub wzięcia zawartości bajtu, lub słowa spod adresu związanego z symbolem /operacje "=" i "@"/. Elementem wyrażenia może być również zawartość wierzchołka stosu.

Symbole umieszczone w słowniku są również wykorzystywane przez debugger przy wyprowadzaniu informacji dla użytkownika. Aktualną zawartość słownika symboli można wyświetlić za pomocą dyrektywy H.

Następną cechą szczególną programu uruchomieniowego DES jest możliwość definiowania stałych punktów kontrolnych z licznikami /pass-point/. Można jednocześnie zdefiniować osiem takich punktów. Jeżeli w trakcie wykonywania testowanego programu licznik rozkazów stanie się równy adresowi któregoś z aktywnych stałych punktów kontrolnych, to na konsoli zostanie wyświetlona informacja o stanie procesora. Jeżeli związany z tym punktem kontrolnym licznik ma wartość większą od jedności, to zostaje zmniejszony o jeden i wykonywanie testowanego programu jest kontynuowane. Jeżeli licznik związany z danym punktem kontrolnym jest równy jedności, testowany program zatrzymuje się.

Użycie dyrektywy uruchamiającej program poprzedzonej znakiem "-" blokuje wyświetlanie informacji o stanie procesora przy stanie licznika różnym od jedności.

Debugger DES umożliwia także wykonywanie podprogramu w testowanym programie. Służy do tego celu dyrektywa C. Powoduje ona wykonanie testowanego programu począwszy od adresu zadanego lub wskazanego przez licznik rozkazów, aż do napotkania instrukcji powrotu z podprogramu.

Ostatnim rozszerzeniem funkcjonalnym w stosunku do debuggera DDT jest możliwość zastosowania podprogramów uruchomieniowych. Jeżeli testowany program użytkowy wykonywany jest krokowo w trybie interpretacyjnym, to po zinterpretowaniu każdego rozkazu testowanego programu można wykonać zewnętrzny podprogram. Podprogram uruchomieniowy pozwala na przykład na zatrzymanie testowanego programu w momencie zmiany zawartości wybranego miejsca pamięci, na zebranie informacji statystycznej o testowym programie itp.

Opisany mechanizm pozwala nie tylko na używanie podprogramów uruchomieniowych tworzonych ad hoc przez użytkownika. Możliwe jest również jego wykorzystanie do istotnego rozszerzenia możliwości debuggera DES.

Wraz z DES-em dostarczane są jego uzupełnienia funkcjonalne zapisywane w zbiorach typu .UTL. Wspomniane programy uzupełnień funkcjonalnych mogą być załadowane do pamięci operacyjnej łącznie z testowanym programem przy wywołaniu debugera DES. Po załadowaniu program rozszerzeń funkcjonalnych pyta o parametry /mogą to być na przykład adresy kontrolowanych obszarów pamięci/, a następnie przekazuje sterowanie debuggerowi. Jeżeli program rozszerzenia funkcjonalnego zbiera informacje o charakterze statystycznym, to można /wywołując podprogram o ustalonym adresie startowym/ uzyskać jej wydruk na ekranie konsoli.

Wraz z debuggerem DES dostarczane są dwa programy rozszerzeń funkcjonalnych zapisane w zbiorach TRACE.UTL i HIST.UTL.

Pierwszy z nich umożliwia wyświetlenie "historii" dojścia do punktu zatrzymania w programie. Wyświetlane jest 256 wartości licznika rozkazów poprzedzające moment zatrzymania programu. Pozwala to na ustalenie np. przyczyny wywołania podprogramu sygnalizacji błędów /w programie użytkowym/.

Drugi z nich - HIST.UTL - zbiera informację statystyczną o częstości wykonywania rozkazów z zadanych obszarów. Po zakończeniu zbierania informacji wyświetla histogram ze względnymi częstościami wykonywania rozkazów. Pozwala to na wykrycie "wąskich gardeł" w uruchamianym programie i ukierunkowuje proces jego optymalizacji pod względem szybkości wykonywania.

Proces tworzenia oprogramowania użytkowego wspomagany różnymi zestawami oprogramowania narzędziowego dostępnymi w MSWP

Przez zestaw oprogramowania narzędziowego rozumiany jest zbiór programów, z których każdy ukierunkowany jest na wspomaganie realizacji określonego etapu procesu tworzenia oprogramowania. Zbiór ten jest kompletny w sensie zapewnienia minimum środków wspomagających każdy z etapów: edycji, translacji i uruchamiania. Programy z danego zestawu działają pod określonym systemem operacyjnym. Włączane do realizacji procesu tworzenia oprogramowania, kolejno, zgodnie z logiką tego procesu, muszą akceptować rodzaj i format danych wyjściowych, generowanych przez określony program narzędziowy w poprzednim etapie, a stanowiących dane wejściowe w bieżącym etapie. Wraz z każdym systemem operacyjnym zainstalowanym w MSWP dostarczany jest co najmniej jeden minimalny zestaw podstawowych programów narzędziowych. W każdym zestawie dostarczany jest przynajmniej jeden translator - jest nim /makro/asembler, ze względu na uzasadnioną popularność języków asemblerowych w zastosowaniach mikroprocesorowych. Co prawda, języki asemblerowe nie zapewniają przenoszalności postaci źródłowej programów na sprzęt o innym typie mikroprocesora, co mogą zapewnić języki wyższego rzędu /o ile tylko na tym innym sprzęcie jest zainstalowany kompilator tego języka, lub też jeżeli dostępny jest

Tab.4. Zestawienie dyrektyw programów uruchomieniowych

Funkcja	MONITOR	DDT	DES	Uwagi
Załadowanie programu uruchamianego	-	parametr wywołania	parametr wywołania	
	-	I nazwa	I nazwa	zadaje nazwę zbioru z modułem programowym
	R	R	R	wczytuje program
	Rb	Rb	Rb	wczytuje program z przesunięciem adresu
Sprawdzanie i zmiana stanu procesora /rejestrów i wskaźników/	X	X	X	wyświetla stan procesora
	Xr	Xr	Xr	ustawia rejestry i wskaźniki
Sprawdzenie i zmiana stanu pamięci operacyjnej	Ds, f	Ds, f	Ds, f	wyświetla szesnastkowo zawartość miejsc pamięci /bajtów/ zadanych adresem początkowym i końcowym obszaru /DDT i DES wyświetlają także znakowo/
	-	Ds	Ds	wyświetla zawartość obszaru pamięci od adresu początkowego /12 wierszy po 16 bajtów/
	-	D	D	wyświetla /12 wierszy/ zawartość pamięci od ostatnio wyświetlonego adresu
	-	-	D, f	wyświetl zawartość pamięci od ostatnio wyświetlanego adresu do adresu zadanego
	-	-	DWs, f DWs DW	analogicznie jak wyżej, z tym, że wyświetlana jest zawartość par bajtów /słów/, a nie pojedynczych bajtów
	-	-	DW, f	
	Ss	Ss	Ss	wyświetl jedno miejsce pamięci /bajt/ i ewentualnie zmień jego zawartość
	-	-	SWs	wyświetl słowo /dwa bajty/ z pamięci i ewentualnie zmień jego zawartość
	Ms,f,d	Ms,f,d	Ms,f,d	kopiuj zawartość obszaru pamięci zawartego pomiędzy adresami s i f do obszaru zadanego adresem d
	Fs,f,c	Fs,f,c	Fs,f,c	Zapełnij obszar pamięci zadany adresami s i f zawartością c
Sprawdzenie stanu programu i jego modyfikacja	-	Ls,f	Ls,f	listuj w postaci mnemonicznej program zapisany w pamięci od adresu s do adresu f
	-	Ls	Ls	listuj w postaci mnemonicznej 12 rozkazów programu począwszy od adresu s
	-	L	L	listuj w postaci mnemonicznej 12 rozkazów programu począwszy od ostatnio listowanego adresu
	-	-	- Ls,f	analogicznie, jak wyżej, lecz bez wyświetlania adresów symbolicznych
	-	-	- Ls	
	-	-	- L	

c.d. Tab.4

Funkcja	MONITOR	DDT	DES	Uwagi
Sprawdzenie stanu programu i jego modyfikacja	-	As	As	dokonaj asemblacji /translacji/ programu wprowadzanego z konsoli w postaci mnemoniczno-symbolicznej i wpisz go do pamięci począwszy od adresu s
	-	-	A	asemblacja programu z wpisaniem do pamięci począwszy od adresu następnego za ostatnio asemblowanym
	-	-	- A	odłącz moduł asemblacji /desasemblacji /zwalnia ożęść pamięci zajętej przez debugger/
Deklarowanie adresów punktów zatrzymań kontrolnych	-	-	Pp	ustaw pass-point na adresie p
	-	-	Pp, c	ustaw pass-point na adresie p z licznikiem przejść równym c
	-	-	P	wyświetl aktywne pass-pointy
	-	-	-Pp	kasuj pass-pointy na adresie p
	-	-	-P	kasuj wszystkie aktywne pass-pointy
Uruchomienie testowanego programu	G	G	G	uruchom program od adresu określonego przez licznik rozkazów
	Gs	Gs	Gs	uruchom program od zadanego adresu s
	Gs, b	Gs, b	Gs, b	uruchom program od adresu s, ustaw break-pointy na adresie b
	Gs,b,c	Gs,b,c	Gs,b,c	uruchom program od adresu s, ustaw break-pointy na adresach b i c
	G, b	G, b	G, b	uruchom program od adresu wskazanego przez licznik rozkazów, ustaw break-pointy na adresie b
	G,b,c	G,b,c	G,b,c	uruchom program od adresu wskazanego przez licznik rozkazów, ustaw dwa break-pointy na adresach b i c
	-	-	-G	jak wyżej, lecz przejście przez adresy pass-pointów z licznikami różnymi od jedności nie jest sygnalizowane wyświetlaniem stanu procesora
	-	-	-Gs	
	-	-	-Gs,b	
	-	-	-Gs,b,c	
	-	-	-G,b	
	-	T	T	wykonaj jeden rozkaz programu interpretacyjnie z wyświetleniem stanu procesora
	-	-	Tn	wykonaj interpretacyjnie n rozkazów z wyświetleniem stanu procesora po każdym rozkazie
	-	-	Tn, c	wykonaj interpretacyjnie n rozkazów, po interpretacji każdego rozkazu wywołaj podprogram o adresie c

c.d. Tab.4

Funkcja	MONITOR	DDT	DES	Uwagi
	-	-	Tc	wykonaj interpretacyjnie jeden rozkaz, a następnie wywołaj program o adresie c
	-	-	-T	jak wyżej, bez wyświetlania symboli
	-	-	-Tn	
	-	-	-Tn,c	
	-	-	-T,c	
	-	-	TW	jak wyżej w analogicznych wariantach dyrektywy T, z tą różnicą, że podprogramy wykonywane są w czasie rzeczywistym, a nie w trybie interpretacyjnym
	-	-	TWn	
	-	-	TWn,c	
	-	-	TW,c	
	-	-	-TW	
	-	-	-TWn	
	-	-	-TWn,c	
	-	-	-TW,c	
	-	U	U	wykonaj jeden rozkaz interpretacyjnie
	-	Un	Un	wykonaj n rozkazów interpretacyjnie bez wyświetlania stanu procesora po każdym rozkazie
	-	-	Un, c	wykonaj n rozkazów interpretacyjnie, po każdym rozkazie wywołaj podprogram o adresie c
	-	-	U, c	wykonaj jeden rozkaz interpretacyjnie, następnie wywołaj podprogram o adresie C
	-	-	-U	jak wyżej, z tą różnicą, że nie jest wyświetlany stan procesora przy przejściu przez adresy passpointów z licznikami różnymi od jedności
	-	-	-Un	
	-	-	-Un,c	
	-	-	-U,c	
	-	-	UW	analogicznie, jak w dyrektywie U, z tą różnicą, że podprogramy nie są interpretowane, lecz wykonywane w czasie rzeczywistym
	-	-	UWn	
	-	-	UWn,c	
	-	-	UW,c	
	-	-	-UW	
	-	-	-UWn	
	-	-	-UWn,c	
	-	-	-UW,c	
	-	-	Cs	przechowaj rejestry, a następnie ustaw BC i DE na zero, wywołaj podprogram o adresie s, po powrocie odtwórz stan rejestrów
	-	-	Cs,b	jak w Cs, z tym, że przed wywołaniem podprogramu do BC wpisywana jest wartość wyrażenia b
	-	-	Cs,b,d	jak wyżej, przed wywołaniem podprogramu do rejestrów BC i DE wpisywane są wartości wyrażeń b i d
Dyrektywy pomocnicze	H a b		Ha, b	oblicza sumę i różnicę w zapisie szesnastkowym
	-	-	Ha	wyświetla wartość wyrażenia, w zapisie szesnastkowym, dziesiętnym i znakowo oraz symbol ze słownika, związany z tą wartością
	-	-	H	wyświetla zawartość słownika symboli

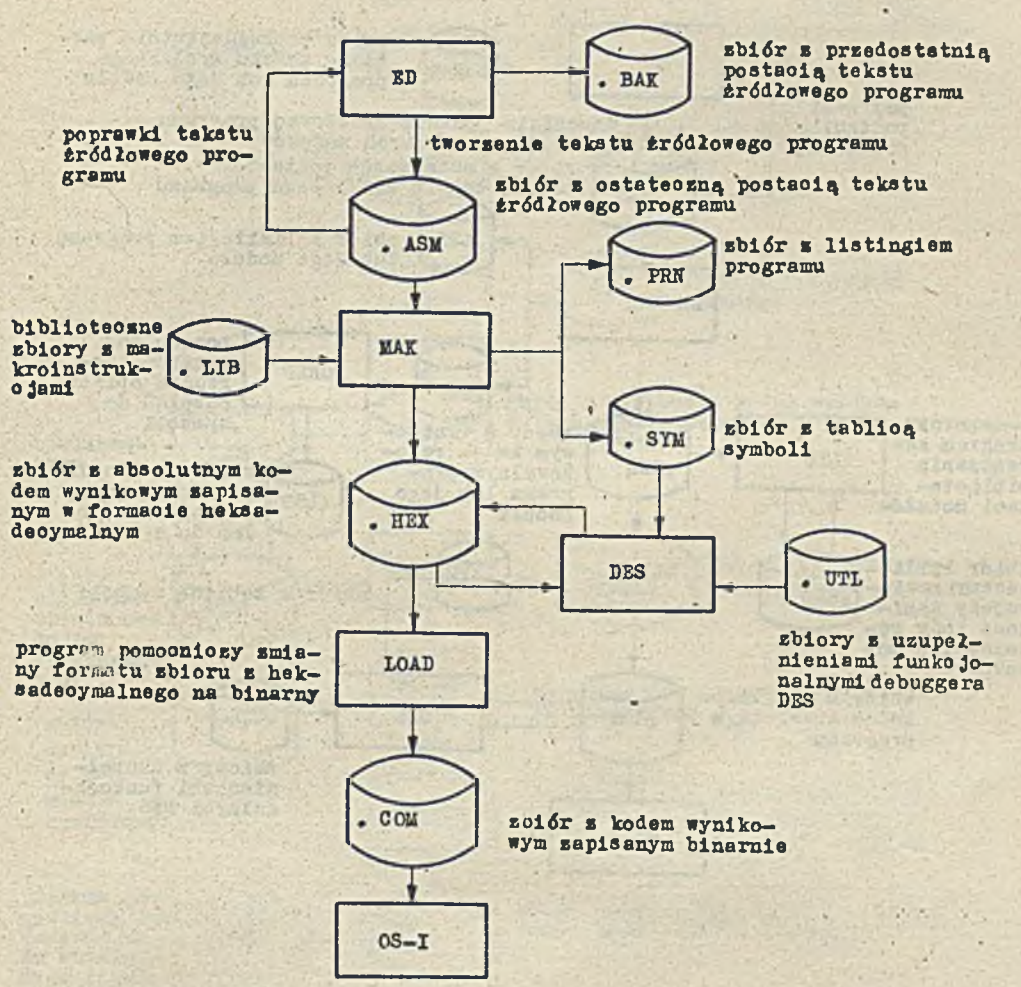
gdziekolwiek kompilator skrośny na ten tym mikroprocesora/, ale z kolei nie istnieją trudności z przenoszalnością na inny sprzęt o tym samym typie mikroprocesora kodu wynikowego programu napisanego w języku asemblerowym. Trudności te pojawiają się dla kodu wynikowego otrzymanego z kompilacji. Zawarte są w nim podprogramy biblioteczne obsługi urządzeń wejścia/wyjścia wykorzystywane przez kompilator. Chcąc zapewnić przenoszalność kodu wynikowego programu napisanego w języku wyższego rzędu, użytkownik musi zastąpić podprogramy biblioteczne obsługi urządzeń wejścia/wyjścia dostarczane wraz z kompilatorem i charakterystyczne dla sprzętu, w którym kompilator działa, podprogramami napisanymi przez siebie, dostosowanymi do wymagań innego sprzętu /ale opartego na tym samym typie mikroprocesora/.

Każdy z dalej przedstawionych zestawów ma różną moc funkcjonalną. Uboższy zestaw zapewnia, co prawda, mniejsze możliwości funkcjonalne, ale też jest prostszy w obsłudze, łatwiejszy do audzenia, no i tańszy. Największe możliwości dają zestawy, ukierunkowane na wspomaganie programowania modularnego, w skład których wchodzą różne translatory. Za ich pomocą można realizować duże przedsięwzięcia programowe wykonywane przez kilka osób, z których każda może z dostępnych języków, akceptowanych przez translatory zestawu, wybrać najbardziej jej i/lub problemowi odpowiadający i w tym języku opracowywać przypisany jej moduł programowy.

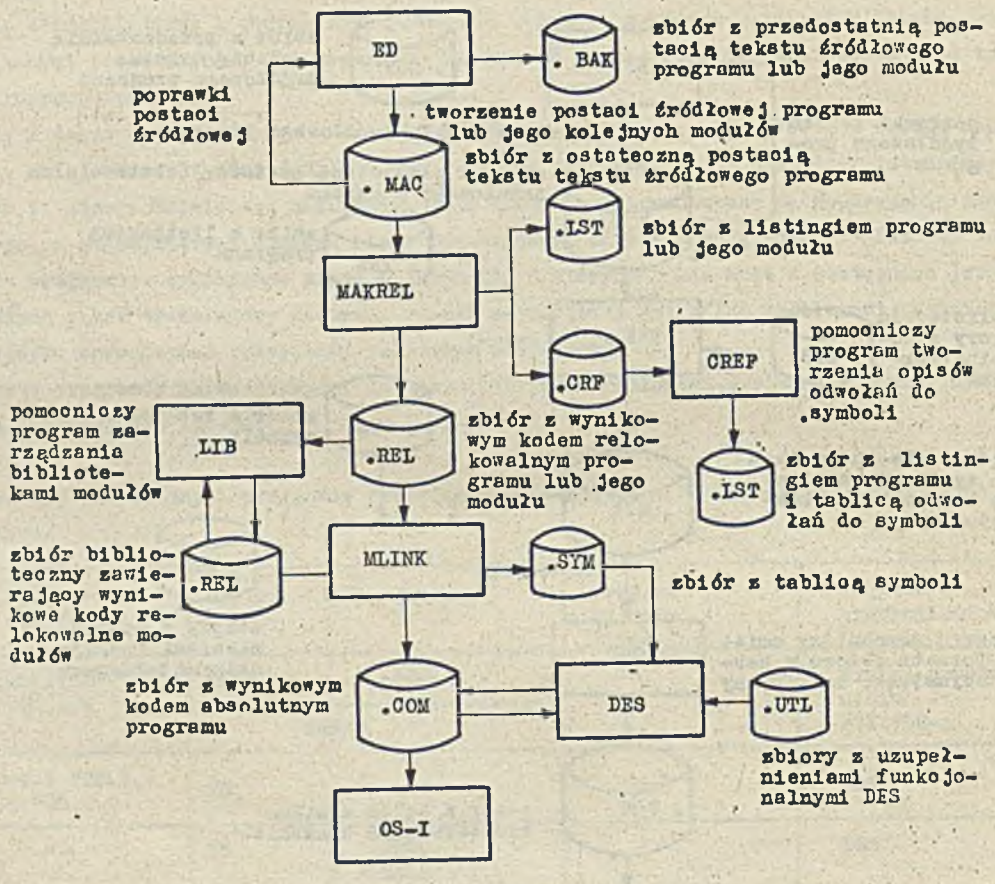
Zestawy programów narzędziowych, działających pod określonymi systemami operacyjnymi, dostępne w MSWP są następujące /tab.5/.

Tab.5. Zestawy rezydentnych programów narzędziowych dostępne w MSWP /nawiasy prostokątne oznaczają opcje/

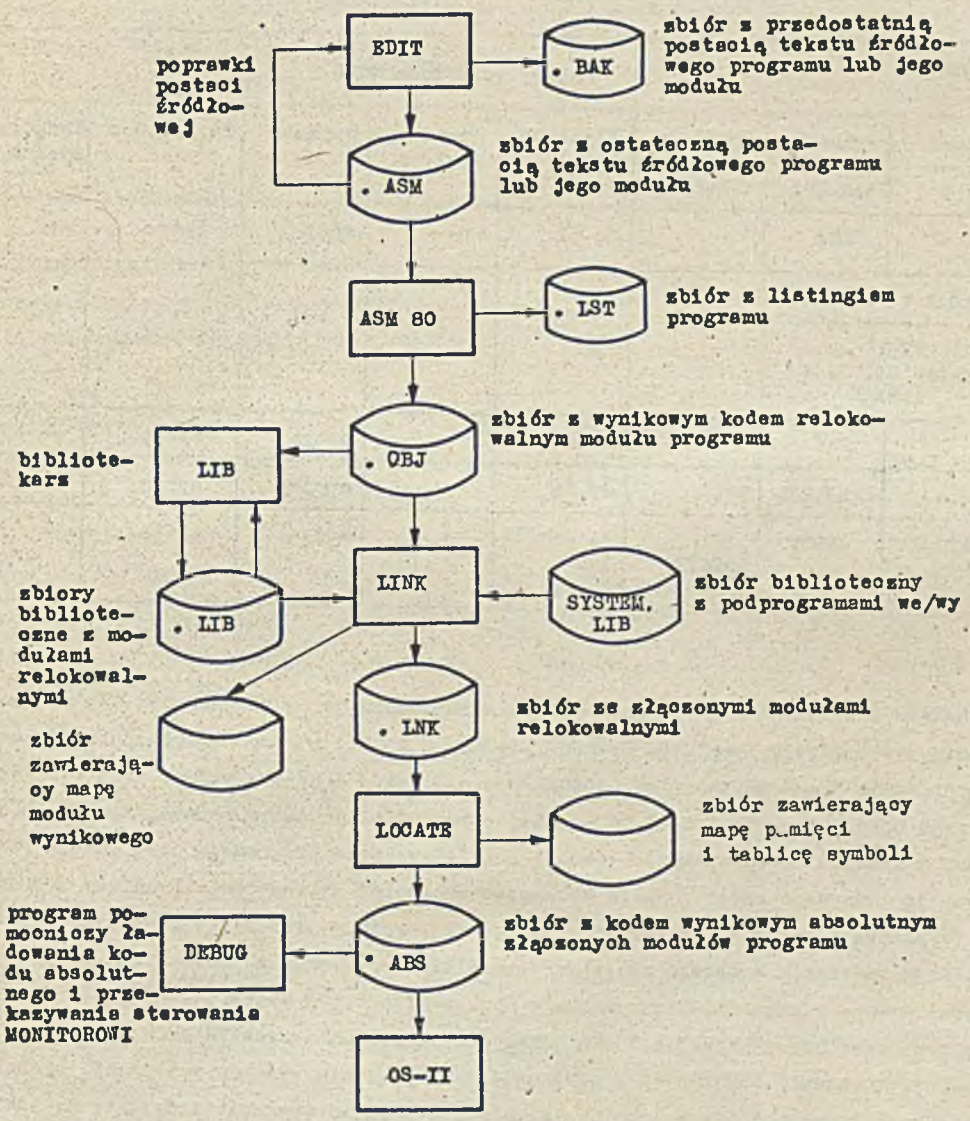
Lp. zestawu	System operacyjny	Edytor	Translator	Program łączący i ładujący	Program uruchomieniowy
1	MONITOR VER.1.8	EDYTOR 1	REM	-	środki uruchomieniowe MONITOR-a
		REMAK			
2	OS-I VER.1.4 i VER.2.2	ED	ASM	-	DDT
3	-"-	ED	MAK + biblioteki makroinstrukcji	-	DES
4	-"-	ED	MAKREL + CREF	MLINK	DES
5	OS-II VER.20	EDIT [EDEK]	ASM80 lub/i PLM80 +biblioteki SYSTEM i PLM80	LINK LOCATE	DEBUG + środki uruchomieniowe MONITOR-a
6	OS-II VER.3.4	EDIT [EDEK]	[ASM80] [PLM80] FORT80 + biblioteki, m.in. arytmetyki zmiennoprzecinkowe	LINK LOCATE	DEBUG + środki uruchomieniowe MONITOR-a



Rys. 3. Proces tworzenia programu jednomodułowego wspomagany programami narzędziowymi systemu OS-I



Rys. 4. Proces tworzenia programu wielomodułowego wspomagany programami narzędziowymi systemu OS-I.



Rys. 5. Proces tworzenia programu wielomodułowego wspomagany programami narzędziowymi systemu OS-II

W zestawach tych nie są wymienione programy pomocnicze, jak np. bibliotekarze. Każdy z tych zestawów dając różne możliwości funkcjonalne określa konkretne wymagania sprzętowe. Niezbędne zestawy sprzętu dla niektórych zestawów programów narzędziowych przedstawia tab.6.

Tab.6. Wymagania sprzętowe niektórych zestawów programów narzędziowych

Zasoby programowe MSWP				Zasoby sprzętowe MSWP					
System operacyjny	Program redagujący	Program tłumaczący	Programy łączenia i ładow.	Wielkość PaO /min./	Pamięć dyskowa	Czytnik	Perforator	Konsola system	Urz. listujące
OS-I	ED	ASM	-	16 Kb	+	opcja	opcja	+	opcja
MONITOR	EDYTOR 1	REM	-	32 Kb	-	+	+	+	opcja
	REMAK /zestaw EDYTOR 1 i REM/								
OS-I	ED	MAK	-	20 Kb	+	opcja	opcja	+	opcja
OS-I	ED	MAKREL	MLINK	29 Kb	+	opcja	opcja	+	opcja
OS-II	EDIT	ASM80	LINK LOCATE	34 Kb lub 50 Kb	+	opcja	opcja	+	opcja

Wybór zestawu z zestawów dostępnych, w celu zastosowania go do wspomagania tworzenia danego przedsięwzięcia programowego, powinien być podyktowany z jednej strony względami praktycznymi /możliwość opanowania przez użytkownika obsługi operatorskiej danego zestawu, przewidziany czas wykonania danego oprogramowania i in./, z drugiej strony efektywnością wspomagania przez dany zestaw procesu tworzenia oprogramowania. Jeżeli przedsięwzięcie programowe jest dużych rozmiarów i przewidziany jest do jego zrealizowania kilkuosobowy zespół wykonawców, to należy wybrać zestaw ułatwiający tworzenie programów modularnych, a więc zawierający translator generujący kod wynikowy relokowalny. Przy tworzeniu prostego programu niewielkich rozmiarów z powodzeniem można wykorzystywać mniej wyrafinowane w swym działaniu zestawy /np. zestaw 2 i 3 z tab.5/.

Procesy przedstawione na rys.3 ÷ 5 nie uwzględniają np. umieszczania kodu wynikowego programu w pamięci stałej, tworzenia dokumentacji użytkowej, umieszczania wyników realizacji poszczególnych etapów procesu na różnych nośnikach, nie uwidoczniają też wielu operacji dodatkowych, jakie użytkownik może wykonać lub też wykonuje współpracując z systemowym oprogramowaniem MSWP. Przykładem takich operacji może być tworzenie zapasowych kopii nośników, sprawdzanie wielkości określonych zbiorów /przez wyświetlanie lub porównanie ze wzorcem/ i in. Rysunki przedstawiają pewne warianty /typowe/ realizacji procesu tworzenia oprogramowania wspomagane różnymi zestawami programów narzędziowych MSWP.

Warto wspomnieć, w jaki sposób oprogramowanie narzędziowe MSWP ułatwia tworzenie opisanych dokumentacji użytkowych programów. Proces produkcji i aktualizacji dokumentacji, uciążliwy i czasochłonny, zwłaszcza przy znacznych rozmiarach dokumentacji, może być częściowo zautomatyzowany za pomocą środków programowych.

Tymi środkami w systemie są programy: redagujący ED i formatujący tekst TXT, oba pracujące pod systemem OS-I. Program TXT może wykonywać następujące operacje na tekście:

- justowanie linii, tj. wypełnianie linii wejściowych tekstem, aby wyrównać marginesy lewy i prawy,
- przenoszenie linii tekstu wejściowego do wyjściowego bez zmian,
- wypełnianie linii wyjściowych tekstem bez wyrównania marginesu prawego,
- centrowanie tekstu w linii wyjściowej,
- wcinanie linii wyjściowej,
- tworzenie stron,
- numeracja stron,
- dodawanie stron pustych,
- rozmieszczanie nagłówka stron,
- pomijanie linii tekstu wejściowego,
- tworzenie marginesów: lewego, prawego, górnego, dolnego,
- ustalanie odstępów między liniami wyjściowymi,
- i inne.

Operacje te zadawane są dyrektywami umieszczonymi we właściwych /z punktu widzenia sensowności operacji/ miejscach tekstu podlegającego formatowaniu, w czasie edycji tego tekstu za pomocą programu ED. Tak przygotowany tekst jest zapisany jako zbiór dyskowy i jest tekstem wejściowym programu TXT. Wywołując program TXT należy podać identyfikator zbioru wejściowego oraz opcjonalnie proponowany identyfikator zbioru wyjściowego /nazwa zbioru, miejsce przesłania/, będącego sformatowanym tekstem wejściowym - produktem wynikowym działania programu TXT na tekście wejściowym. Wszelkie zmiany w tekście wejściowym /np.aktualizacja dokumentacji/ czynione są za pomocą programu edytora ED.

Procesy produkcji i aktualizacji dokumentacji za pomocą programów ED i TXT są szybsze niż realizowane tradycyjnie w postaci maszynopisów.

Zakończenie

Materiał przeglądowy zawarty w tym opracowaniu został podany wybiórczo, ale nie przypadkowo. To, że główny nacisk jest położony na przedstawienie możliwości MSWP wspomagania tworzenia oprogramowania w języku asemblerowym, zostało podyktowane doświadczeniem autorów wskazującym na to, że głównym zainteresowaniem użytkowników systemu MSWP cieszą się /makro/ asemblery. To samo doświadczenie wskazuje też, że nie są przez użytkowników wykorzystywane w pełni wszystkie możliwości funkcjonalne /makro/asemblerów i innych programów z zestawów narzędziowych. Stąd zrodził się pomysł tego opracowania, aby możliwie w pełni zademonstrować oprogramowanie narzędziowe ukierunkowane na języki asemblerowe i sposoby jego wykorzystywania.

Sprawą innego opracowania będzie rozreklamować i przybliżyć użytkownikowi translatory języków wyższego rzędu, dostępne w MSWP.

mgr inż. Małgorzata SADOWSKA-ROSIŃSKA
mgr inż. Wojciech KUBERA
Instytut Maszyn Matematycznych

Programy mikroprocesorowego systemu wspomagania projektowania zapisane w pamięci stałej

W pamięci stałej systemu MSWP zapisane są następujące programy:

- BOOTSTRAP - program inicjujący pracę systemu MSWP,
- DYSK - program inicjujący pracę i obsługę dysków elastycznych,
- MONITOR - prosty program uruchomieniowy realizujący 14 dyrektyw, które służą do wprowadzania, wyprowadzania i uruchamiania programów zapisanych na taśmie perforowanej. Program ten zawiera również obsługę urządzeń zewnętrznych MSWP, tj. czytnika, perforatora, konsoli, drukarki.

PROGRAM BOOTSTRAP

Naciśnięcie przycisku "PROGRAM STARTU" na pulpicie MSWP powoduje włączenie pamięci stałej o pojemności 256 bajtów /na najmłodsze adresy/, na której zapisany jest program BOOTSTRAP. Jest to pamięć wyłącznie czytana, natomiast zapisywanie danych odbywa się do pamięci operacyjnej MSWP pod tymi samymi adresami. Naciśnięcie przycisku "ZEROWANIE" na pulpicie MSWP powoduje start programu BOOTSTRAP, który poza inicjacją systemu MSWP sprawdza, czy w systemie zainstalowany jest program obsługi dysków - DYSK. Jeśli tak, to sterowanie zostaje przekazane do programu DYSK, jeśli nie - do programu MONITOR.

Zwolnienie przycisku "PROGRAM STARTU" powoduje zgłoszenie się na konsoli odpowiedniego programu: MONITOR-a, systemu operacyjnego OS-I lub OS-II w zależności od zainstalowanej dyskietki systemowej.

Program BOOTSTRAP spełnia następujące funkcje:

- programuje interefejs szeregowy typu USART 8251 Intela,
- programuje kontroler przerwań typu 8214 typu Intel,
- oblicza maksymalny rozmiar zainstalowanej w systemie pamięci operacyjnej /sprawdza co 256 bajtów/ i wpisuje najwyższy adres do komórek pamięci o adresach 0004 - młodszy bajt, 0005 - starszy bajt adresu,
- wpisuje do komórek pamięci o adresach 0000-2 skok do obsługi przerwania 0,
- ustala stos programu MONITOR i zapisuje jego adres do komórek pamięci o adresach 0006-7,
- wpisuje do komórki pamięci o adresie 0003 status urządzeń zewnętrznych przypisując: konsola = monitor ekranowy lub drukarka z klawiaturą,

perforator = perforator taśmy papierowej,
czytnik = czytnik taśmy papierowej,
urządzenie listujące = konsola,

- wpisuje do 32 bajtów o najwyższych adresach pamięci operacyjnej dane robocze dla programu MONITOR,
- sprawdza czy zainstalowany jest program DYSK, jeśli tak, to przekazuje do niego sterowanie, jeśli nie, to sterowanie przekazywane jest do programu MONITOR.

PROGRAM OBSŁUGI DYSKÓW ELASTYCZNYCH - DYSK

System MSWP wyposażony jest w stosunkowo prosty pod względem realizacji sprzętowej kontroler pamięci na dyskach elastycznych, który wymaga rozbudowanej obsługi programowej. Obsługę tę realizuje rezydujący w pamięci stałej MSWP program DYSK.

Program DYSK wykonuje trzy podstawowe funkcje traktowane jako niezależne podprogramy:

- odczyt bloku danych /sektora/ z dysku do pamięci operacyjnej,
- zapis bloku danych /sektora/ z pamięci operacyjnej na dysk,
- początkowe ładowanie dyskowego systemu operacyjnego.

Podprogramy odczytu korzystają z 16 miejsc roboczych, które mają na stałe przypisane /dla danej wersji programu DYSK/ miejsca w pamięci operacyjnej. Program wywołujący musi przed wywołaniem podprogramu zapisu lub odczytu wpisać w odpowiednie miejsca robocze adres dyskowy /numer stacji dyskowej, strona dyskietki, ścieżka, sektor/ i adres budora w pamięci operacyjnej. Podprogramy zapisu i odczytu dokładnie kontrolują przebieg operacji dyskowych i zapewniają automatyczne kilkakrotne /standardowo 5 razy/ powtarzanie operacji po błędzie i dokładną sygnalizację błędów.

Podprogram ładowania dyskowego systemu operacyjnego wywoływany jest w trakcie inicjacji systemu przez program BOOTSTRAP. Podprogram ten sprawdza, czy w lewej kieszeni /A/ stacji pamięci na dyskach elastycznych załadowany jest dysk i czy jest to dysk systemowy. Jeżeli tak, to inicjuje ładowanie systemu operacyjnego do pamięci operacyjnej systemu wczytując odpowiednie moduły programowe z dysku, a następnie przekazuje im sterowanie. Jeżeli dysk systemowy nie jest umieszczony w kieszeni stacji pamięci na dyskach elastycznych lub nie daje się odczytać, sterowanie przekazywane jest programowi MONITOR.

Oprócz wyżej opisanych podstawowych podprogramów na program DYSK składają się liczne podprogramy pomocnicze wykorzystywane przez specjalne oprogramowanie takie, jak program formatowania dysków, testy pamięci na dyskach elastycznych, itp.

PROGRAM MONITOR

Program MONITOR w rozpowszechnianej obecnie wersji 1.8 zajmuje 2k pamięci stałej zainstalowanej pod najwyższymi adresami. Program ten realizuje 14 dyrektyw, które służą do wprowadzania, wyprowadzania i uruchamiania programów użytkownika. Dyrektywy wprowadzania i wyprowadzania programów użytkownika umożliwiają:

- ładowanie do pamięci operacyjnej danych zapisanych na ośmiościeżkowej taśmie perforowanej z bitem parzystości, w heksadecymalnym standardzie asemblerów Intela; możliwe jest ładowanie do pamięci z przesunięciem adresów /dyrektywa R/,
- wyprowadzanie zawartości pamięci na ośmiościeżkową taśmę perforowaną z bitem parzystości /w heksadecymalnym kodzie asemblerów Intela/ lub urządzenie listujące /dyrektywy W, D/,

Tabela 1. Zestaw dyrektyw programu MONITOR

Funkcje	Dyrektywa	Uwagi
Wprowadzanie programu	R	wczytuje program z taśmy perforowanej
	Rb	wczytuje program z taśmy perforowanej z przesunięciem
Sprawdzanie i zmiana stanu procesora /rejestrów i wskaźników/	X	wyświetla rejestry i wskaźniki procesora
	Xr	ustawia rejestry i wskaźniki procesora
Sprawdzanie i zmiana zawartości pamięci operacyjnej	Ds,f	wyświetla w kodzie heksadecymalnym zawartość pamięci /bajty/ obszaru zadanego adresami s /początkowy/ i f /końcowy/
	Ss	wyświetla jeden bajt pamięci o adresie s i ewentualnie zmienia jego zawartość; umożliwia także takie działanie na następnych i poprzednich bajtach
	Ms,f,d	kopiuje zawartość obszaru pamięci zadanego adresami s i f do obszaru pamięci zadanego adresem początkowym d
	Fs,f,c	zapełnia obszar pamięci zadany adresami s i f zawartością c
	Ps,f,d	porównuje zawartość obszaru pamięci zawartego pomiędzy adresami s i f z zawartością obszaru pamięci zadanego adresem początkowym d
Uruchomienie programu	G	uruchamia program od adresu określonego przez licznik rozkazów
	Gs	uruchamia program od zadanego adresu s
	Gs,b	uruchamia program od zadanego adresu s, ustawia break-point na adresie b
	Gs,b,c	uruchamia program od zadanego adresu s, ustawia break-pointy na adresach b i c
	G,b	uruchamia program od adresu określonego przez licznik rozkazów, ustawia break-point na adresie b
	G,b,c	uruchamia program od adresu określonego przez licznik rozkazów, ustawia break-pointy na adresach b i c
Wyprowadza program	Ws,f	wyprowadza zawartość obszaru pamięci zadanego adresami s, f na taśmę perforowaną
	Eb	wyprowadza rekord końcowy z zadanym adresem startu programu b na taśmę perforowaną
	N	wyprowadza 127 znaków pustych na perforator
	Ts,f	porównują dane na taśmie perforowanej z zawartością obszaru pamięci zadanego adresami s i f
Deklarowanie urządzenia listującego	L	wyświetla, co jest przypisane jako urządzenie listujące
	Lx	przypisuje konsolę lub drukarkę /zadane parametrem x/ jako urządzenie listujące
Dyrektywa pomocnicza	Ha,b	oblicza sumę i różnicę dwu zadanych liczb heksadecymalnych a i b

- wyprowadzanie na taśmę perforowaną 127 znaków pustych /dyrektywa N/,
- porównywanie zgodności zawartości pamięci z danymi na taśmie perforowanej /dyrektywa T/,
- porównywanie zawartości dwóch obszarów pamięci /dyrektywa P/,
- deklarowanie jako urządzenia listującego konsoli lub drukarki /dyrektywa L/.

Użycie dyrektyw uruchamiania programu /listę dyrektyw podano w tabeli 1/ umożliwia użytkownikowi:

- sprawdzenie i ewentualną zmianę zawartości komórek pamięci operacyjnej /dyrektywa S/,
- sprawdzanie i ewentualną zmianę zawartości rejestrów mikroprocesora /dyrektywa X/,
- start programu od zadeklarowanego adresu lub kontynuację wykonywania programu według licznika rozkazów, umieszczenie dwóch punktów przerwań programowych /dyrektywa G/,
- wstawianie stałej wartości do zadeklarowanego obszaru pamięci /dyrektywa F/,
- obliczanie sumy i różnicy dwu czterocyfrowych liczb heksadecymalnych /dyrektywa H/,
- przepisywanie obszarów pamięci /dyrektywa M/.

Program MONITOR ma zamaskowane wszystkie przerwania oprócz przerwania 0. Przerwanie 0 pozwala użytkownikowi w dowolnym momencie przerwać wykonywanie programu; zapamiętane wówczas zostaną wszystkie rejestry mikroprocesora, na konsoli zostanie wypisany adres instrukcji, która miała być wykonana, po czym zgłosi się program MONITOR. Powrót z przerwania następuje według licznika rozkazów /dyrektywa G/.

Program MONITOR współpracuje ze standardowymi urządzeniami zewnętrznymi we/wy, którymi są:

- jako konsola systemu zamiennie:
  drukarka z klawiaturą /DZM 180 KSR/,
  monitor ekranowy z klawiaturą /VIDEOTON 340 lub MERA 7952, 7953/,
  teletype,
- perforator taśmy perforowanej /DT 105S lub DT103/,
- czytnik taśmy perforowanej /CT 2100/,
- drukarka /DZM 180/.

Program MONITOR zawiera obsługę tych urządzeń czytaj/pisz znak, czytaj status konsoli w postaci podprogramów, które mogą być wykorzystywane przez użytkownika, podobnie jak podprogramy MONITORA, np. obsługa przerwań.

mgr inż. Wojciech KUBERA
Instytut Maszyn Matematycznych

Dyskowe oprogramowanie systemowe MSWP

Wstęp

W niniejszej pracy omówione zostanie oprogramowanie Mikroprocesorowego Systemu Wspomagania Projektowania /MSWP/ rezydujące na dyskach elastycznych. W tej grupie programów wyróżnić można:

- systemy operacyjne zarządzające pracą MSWP,
- systemy programowania w językach niskiego i wysokiego poziomu,
- oprogramowanie sterujące pracą modułów wyspecjalizowanych.

Dyskowe systemy operacyjne

Rodzina dyskowych systemów operacyjnych OS-I

Systemy operacyjne OS-I przeznaczone są do sterowania pracą mikrokomputerów wyposażonych w pamięci masowe na magnetycznych dyskach elastycznych. Zapewniają one tworzenie na dysku elastycznym zbiorów zawierających programy lub dane i odwoływanie się do nich przez operatora lub z programu użytkowego wyłącznie za pomocą nadanej im nazwy. System przyjmuje na siebie całość zadań związanych z administrowaniem pamięcią masową: wyszukiwaniem informacji na dysku, gospodarowaniem pamięcią na dysku /przydzielanie nowych obszarów zbiorom, zwalnianie niepotrzebnych/ itd.

Systemy operacyjne OS-I są funkcjonalnie zgodne z systemem CP/M, który w wyniku szerokiego rozpowszechniania stał się prawdziwym standardem światowym w dziedzinie oprogramowania mikrokomputerów ośmiobitowych.

System OS-I składa się z części rezydentnej stanowiącej zestaw podprogramów dostępnych dla oprogramowania użytkowego, realizujących elementarne operacje obsługi wejść-wyjść i obsługi zbiorów dyskowych, z modułu realizującego polecenia /dyrektywy/ użytkownika oraz z zestawu niezbędnych użytkowych programów ogólnego przeznaczenia.

W zestawie dyrektyw dostępnych dla operatora znajdują się dyrektywy wyprowadzania katalogu zbiorów, usunięcia zbioru, wyprowadzenia zawartości zbioru tekstowego na konsolę systemową, zapisu zawartości pamięci operacyjnej do zbioru dyskowego oraz dyrektywa ładowania i inicjacji wykonania programu zapisanego w zbiorze dyskowym.

Zestaw podstawowy programów użytkowych zawiera między innymi: program wydawania informacji o zbiorach dyskowych i urządzeniach zewnętrznych oraz zmiany ich stanów, program wyprowadzania zawartości zbioru w postaci szesnastkowej, program interpretacji makropoleceń operatorskich zapisanych w zbiorze dyskowym, program realizujący przesyłanie danych między urządzeniami i zbiorami oraz najprostszy zestaw oprogramowania narzędziowego składający się z kontekstowego redaktora /edytora/ najprostszego asemblera, programu uruchomieniowego /debuggera/ i programu tworzącego zbiór zawierający moduł programowy w postaci binarnej - gotowy do wykonania.

System OS-I oferowany jest w dwóch wersjach: OS-I VER. 1.4 i OS-I VER.2.2. Pierwsza z nich może obsługiwać do czterech logicznych mechanizmów dyskowych o pojedynczej gęstości zapisu sformatowanych standardowo. Wersja 2.2 umożliwia dołączenie do mikrokomputera do 16 logicznych mechanizmów dyskowych o różnej pojemności i dowolnym sposobie formatowania, a więc nie tylko dysków elastycznych, ale i innych pamięci masowych z bezpośrednim dostępem, np. dysków twardych, czy dysków typu Winchester. Wersja 2.2 ma również bogatszy i bardziej elastyczny sposób zarządzania zbiorami, m.in. zapewnia obsługę zbiorów w trybie bezpośredniego dostępu, nadawanie zbiorom atrybutów /np. ochrony przed zapisem, przynależności do określonego użytkownika/ i bogatszy funkcjonalnie zestaw podprogramów obsługi zbiorów.

Rodzina dyskowych systemów operacyjnych OS-II

Systemy operacyjne OS-II spełniają podobne funkcje, jak systemy operacyjne OS-I. Są jednak ukierunkowane na zastosowania w produkcji oprogramowania dla innych urządzeń lub systemów mikroprocesorowych niż te, na których zainstalowany jest sam system OS-II. System OS-II zapewnia środki dla programowania modularnego. Wzorowany /i funkcjonalnie z nimi zgodny/ na systemach operacyjnych klasy ISIS-II, opracowanych przez firmę Intel Corporation dla systemów Intellec MDS jest typowym systemem operacyjnym dla systemów uruchomieniowych.

Zestaw podstawowy OS-II oprócz rezydentnego jądra, będącego zestawem podprogramów obsługi zbiorów /urządzenia zewnętrzne są w tym systemie traktowane równorzędnie ze zbiorami dyskowymi i obsługiwane przez te same podprogramy/ i modułu interpretującego polecenia /dyrektywy/ użytkownika, zawiera programy realizujące elementarne zlecenia operatorskie /wyprowadzenia katalogu zbiorów, przemianowania zbioru, usunięcia zbioru, kopiowania zbioru, nadania zbiorowi atrybutów, inicjacji nośnika/, program łączęco-redagujący /linker/, program tworzenia modułu absolutnego /alokator/, program prowadzenia bibliotek modułów wynikowych oraz bibliotekę systemową.

System OS-II oferowany jest w dwóch wersjach: OS-II VER.2.2 i OS-II VER. 3.4. Ta ostatnia różni się od pierwszej /poza drobnymi udogodnieniami operatorskimi/ bardziej rozwiniętą strukturą standardowego kodu wynikowego, co umożliwia spełnienie pewnych specyficznych wymagań stawianych przez programy pisane w językach wysokiego poziomu.

Oprogramowanie działające pod kontrolą systemu OS-I

Poza zestawem podstawowym systemu operacyjnego OS-I w skład oprogramowania MSWP wchodzą programy użytkowe, systemy programowania w językach asemblerowych, systemy programowania w językach BASIC i FORTH oraz oprogramowanie przetwarzania tekstów działające pod kontrolą tego systemu operacyjnego.

Systemy programowania w języku asemblera

W obecnej chwili oferowane są systemy programowania w języku asemblera mikroprocesorów Intel 8080 i Z-80. Do programów tej klasy należą:

- MAK - makroasembler akceptujący najnowszy standard języka makroasemblera dla mikroprocesora Intel 8080; ma rozwinięty aparat makroinstrukcji, generuje absolutny kod wynikowy /format hex/.
  Cechą charakterystyczną MAKa jest możliwość korzystania z bibliotek makroinstrukcji. Wraz z makroasemblerem dostarczane są biblioteki realizujące operatory programowania strukturalnego i dostęp do rezydentnych podprogramów systemu operacyjnego. Możliwe jest dostarczanie bibliotek makrodefinicji umożliwiających wykorzystanie MAK-a jako asemblera skrośnego dla innych typów mikroprocesorów.
- MAKREL - makroasembler dla mikroprocesorów Intel 8080 i Z-80. Generuje kod relokowalny, umożliwia programowanie modularne.
  Charakteryzuje się bogatym zestawem dyrektyw sterujących /pseudoinstrukcji/ i rozwiniętym aparatem makroprzetwarzania,
- LINK - program łączącо-ładujący dla modułów wynikowych generowanych przez makroasembler MAKREL,
- LIB - program do tworzenia i administrowania bibliotekami modułów wynikowych generowanych przez makroasembler MAKREL,
- DES - program uruchomieniowy /debugger/ wspomagający proces testowania programów, lokalizacji i usuwania błędów. Działa w sposób interakcyjny, operując na symbolicznych adresach i nazwach zmiennych. Odpowiednie słowniki symboli uzyskuje się z programów LINK lub MAK. Realizuje bogaty zestaw dyrektyw uruchomieniowych. Pozwala na zadawanie parametrów w postaci szesnastkowej, dziesiętnej, znakowej i jako wyrażeń arytmetycznych z argumentami symbolicznymi. Umożliwia dołączanie rozszerzeń funkcjonalnych, to jest dodatkowych specjalistycznych programów wspomagających proces uruchamiania. Dostarczane są rozszerzenia umożliwiające śledzenie programu wstecz /backtracing/ i rozszerzenie umożliwiające badanie intensywności wykonywania poszczególnych fragmentów programu celem optymalizacji czasu wykonywania. Oprócz powszechnie stosowanych punktów wstrzymania /break-point/ umożliwia zdefiniowanie stałych punktów kontrolnych z licznikami przejść /pass-point/.
- DISAM - disasembler dla programów napisanych w języku wewnętrznym mikroprocesora Intel 8080. Na podstawie binarnej postaci programu tworzy równoważny zapis programu w języku asemblera. Automatycznie odróżnia obszary danych i rozkazów, generuje wiele różnych typów symboli i etykiet w zależności od ich roli w analizowanym programie /na przykład wyróżnia adresy podprogramów, adresy stałych i zmiennnych, zmienne bajtowe od słowowych, itp./. Użytkownik ma możliwość wprowadzania dodatkowych informacji polepszających czytelność uzyskiwanego programu w języku asemblera. Nie nakłada się żadnych ograniczeń na wielkość analizowanego programu /do 64 K bajtów/.

Systemy programowania w językach wysokiego poziomu

Z języków wysokiego poziomu w systemie operacyjnym OS-I zrealizowane zostały do tej pory języki BASIC i FORTH.

Istnieje kilka implementacji języka BASIC dostosowanych do różnych wymagań użytkowników, a mianowicie:

- INTERPRETER BASIC VER. 2.1 - interpretator pełnego języka BASIC /zgodnie ze standardem ANSI/. Bogaty zestaw funkcji wbudowanych w interpretator, możliwość wykonywania działań na liczbach całkowitych, zmiennoprzecinkowych o pojedynczej i podwójnej precyzji oraz na łańcuchach znaków, bogaty zestaw operacji ze zbiorami na dysku elastycznym. Interpretator realizuje rozbudowany zestaw dyrektyw sterujących jego pracą i wspomagających proces testowania i poprawiania programu,
- KOMPILATOR BASIC VER. 2.1 - kompilator akceptujący ten sam język wejściowy, co i INTERPRETER BASIC VER. 2.1. Zaletą stosowania kompilatora jest możliwość uzyskania programu działającego znacznie szybciej niż przy stosowaniu interpretatora,
- BASIC VER. 2.2 - system programowania typu: kompilacja na język pośredni - interpretacja kodu pośredniego, akceptujący wariant języka BASIC dostosowany do obliczeń naukowych i inżynierskich. Zaletą takiego sposobu implementacji w porównaniu z bezpośrednią interpretacją jest możliwość wykonywania znacznie większych programów przy tej samej dostępnej pamięci operacyjnej;
- BASIC VER. 2.3 - system programowania typu kompilacja na język pośredni - interpretacja kodu pośredniego, akceptujący wariant języka BASIC ukierunkowany na zastosowania w rachunkowości i zarządzaniu. Cechą charakterystyczną tego wariantu języka jest stosowanie arytmetyki dziesiętnej /eliminacja błędów konwersji/, rozbudowane środki operowania na łańcuchach znaków i zbiorach dyskowych, bogate możliwości redagowania /formatowania/ wyników.

Język FORTH zrealizowany został w dwóch wariantach: standardowym i rozszerzonym. Obydwa systemy programowania realizowane są metodą interpretacji.

Inne programy działające pod kontrolą OS-I

Do opracowywania wszelkiego rodzaju dokumentacji opisowej służy formator tekstów TXT. Program ten pozwala przetworzyć tekst przygotowany uprzednio w postaci zbioru dyskowego /na przykład za pomocą redaktora/ w tekst odpowiadający wymaganiom edytorskim. Umożliwia podział tekstu na numerowane, opatrzone nagłówkami strony, wyrównanie marginesów, uzyskiwanie specjalnego sposobu rozmieszczenia tekstu na stronie, itp. Sterowanie formatem uzyskuje się przez wpisanie w tekst źródłowy specjalnych dyrektyw interpretowanych przez program TXT.

Ze względu na możliwość awarii sprzętu lub zużycie nośnika niewykluczona jest sytuacja, w której część zbiorów zapisanych na dysku staje się niedostępna przez normalne mechanizmy systemu operacyjnego np. zniszczenie katalogu zbiorów. Zrekonstruowanie utraconych danych może być bardzo żmudne i czasochłonne. W takich sytuacjach można ułatwić sobie pracę za pomocą opracowanych w IMM programów ODZYSK i RETRI. Programy te umożliwiają przeglądanie zawartości wskazanych obszarów dysku /ma to sens tylko dla zbiorów znakowych, np. programów źródłowych/, wyszukanie potrzebnych danych i przepisanie ich do zbioru dyskowego na nieuszkodzonym nośniku.

Inny program - UZDA pozwala wykorzystywać częściowo uszkodzone nośniki do pracy w systemie OS-I. Program ten testuje nośnik i łączy wszystkie wadliwe obszary w zbiór, blokując systemowi operacyjnemu dostęp do tych obszarów nośnika. Pozwala to zmniejszyć prawdopodobieństwo przydzielenia uszkodzonego obszaru nośnika zbiorowi użytkownika.

Ponieważ taśma papierowa bywa jeszcze dość często stosowana jako nośnik danych, opracowany został program sprawdzenia poprawności kopii taśmowej zbioru dyskowego SPRAWDZ, który również jest oferowany w zestawie programów systemu OS-I.

Oprogramowanie działające pod kontrolą OS-II

Poza zestawem podstawowym, pod kontrolą systemu operacyjnego OS-II działają następujące programy:

- EDIT - kontekstowy redaktor tekstów przeznaczony do przygotowywania programów źródłowych,
- ASM-80 - makroasembler dla mikroprocesorów Intel 8080 i Intel 8085. Standardowy makroasembler generujący przemieszczalny kod wynikowy z bogatymi możliwościami sterowania procesem translacji i rozbudowaną informacją kontrolno-diagnostyczną,
- PLM-80 - kompilator języka wysokiego poziomu PL/M-80 specjalnie opracowanego dla potrzeb oprogramowania systemów i urządzeń mikroprocesorowych,
- FORT-80 - kompilator standardowego podzbioru języka FORTRAN-77 dla urządzeń z mikroprocesorem Intel 8080. Główne różnice w stosunku do pełnej wersji języka sprowadzają się do braku typów zespolonego i podwójnej precyzji. Jednocześnie język akceptowany przez kompilator zawiera rozszerzenia ukierunkowane na programowanie urządzeń mikroprocesorowych,
- FPA - biblioteka podprogramów arytmetyki zmiennego przecinka dla mikroprocesora Intel 8080,
- CHANGE - program umożliwiający przekazywanie zbiorów dyskowych z systemu OS-II do systemu OS-I.

Specjalistyczne oprogramowanie Mikroprocesorowego Systemu Wspomagania Projektowania

W skład Mikroprocesorowego Systemu Wspomagania Projektowania wchodzi kilka wyspecjalizowanych modułów sprzętowych. Każdy z tych modułów sterowany jest przez specjalny program. Chociaż wspomniane oprogramowanie sterujące ma charakter specjalistyczny, podamy tu krótką informację o nim. W skład omawianego oprogramowania wchodzą:

- EMULATOR 8080 - program sterujący emulatorem układowym mikroprocesora Intel 8080,
- SYMROM - program sterujący modułem symulatora bipolarnych pamięci stałych,
- MONITOR TESTERA - program sterujący uniwersalnym modułem testującym UMT-1 /tester do sprawdzania pakietów logicznych i obwodów drukowanych/,
- JĘZYK TESTERA - program tłumaczący zapis testu w języku opisu testu na tablicę sterującą testera UMT-1,
- PROGRAMATOR 1702 - program sterujący pracą programatora pamięci stałych typu Intel 1702A,
- PROGRAMATOR 2704/32 - program sterujący pracą programatora pamięci stałych /EPROM/ typów: Intel 2704, 2708, 2716 i 2732,
- PROGRAMATOR TM 621/24 - program sterujący pracą programatora pamięci stałych typu TM621, TM622 i TM624 firmy Tungsram.

Oprogramowanie MSWP w opracowaniu

W IMM prowadzone są nadal prace nad rozszerzeniem dyskowego oprogramowania systemowego MSWP. Prace te idą w następujących kierunkach:

- w dziedzinie systemów operacyjnych - zwiększenie efektywności wykorzystania MSWP /praca wieloprogramowa i wielodostępna/ i zwiększenie szybkości działania systemu /systemy z buforowaniem operacji dyskowych/,
- w dziedzinie systemów programowania w językach asemblerowych - opracowanie asemblerów skrośnych dla nowych typów mikroprocesorów /planuje się takie programy dla Intel 8086 i Intel 8048/

i polepszenie własności użytkowych istniejących asemblerów,

- w dziedzinie systemów programowania w językach wysokiego poziomu - opracowanie systemów programowania w językach PASCAL i C,
- w dziedzinie oprogramowania pomocniczego - polepszenie własności użytkowych redaktorów tekstu /przejście na ekranowy tryb redagowania/ i wzbogacenie zestawu oprogramowania pomocniczego.

TECHNIKI KOMPUTEROWE 5-6/84

mgr inż. Małgorzata SADOWSKA-ROSIŃSKA
Instytut Maszyn Matematycznych

Oprogramowanie MSWP na taśmach perforowanych

Przez oprogramowanie na taśmach perforowanych rozumie się te programy pracujące w systemie MSWP, dla których dane wejściowe i wyjściowe mogą być zapisane tylko na ośmiościeżkowych taśmach perforowanych z bitem parzystości. Dla samych zaś programów nośnikiem może być, zarówno taśma perforowana, jak i dysk elastyczny (dyskietka); programy zapisane są jako zbiory systemu operacyjnego OS-I.

W skład oprogramowania na taśmach perforowanych wchodzą następujące programy:

- dwa programy do redagowania tekstów - EDYTOR 1 i EDYTOR 2,
- makroasembler - REM,
- zestaw EDYTORA 2 i makroasemblera REM - REMAK,
- dwa interpretatory języka BASIC - MINIBASIC i INTERPRETER BASIC,
- test pamięci RAM - TEST RAM, dla którego nośnikiem jest jedynie taśma perforowana

i programy, które wykonane są także w wersji na dysku elastycznym pracującym pod systemem operacyjnym OS-I (współpracujące z danymi zapisanymi jako zbiory OS-I):

- program emulatora dla układów wykonanych na mikroprocesorze typu Intel 8080 - EMULATOR 8080,
- program symulatora pamięci stałych typu - SYMROM,
- test czytnika i perforatora - TEST CZYTNIKA I PERFORATORA,
- oprogramowanie testera MONITOR TESTERA i JEZYK TESTERA,
- oprogramowanie programatorów pamięci stałej typu Intel 1702, 2704, 2716, 2732 - PROGRAMATOR 1702, PROGRAMATOR 2704/32.

Poniżej opisane zostaną programy, które pracują wyłącznie w wersji na taśmach perforowanych. Natomiast programy mające wersje współpracujące z danymi na dysku elastycznym zostały opisane w innych opracowaniach.

Program EDYTOR 1

Programy edytora tekstowego EDYTOR 1 i makroasemblera REM wykonane są w dwóch wersjach. Jedna z nich umożliwia współpracę programu EDYTOR 1 z programem makroasemblera REM na wspólnym buforze danych - zestaw tych programów nosi nazwę REMAK. Druga wersja tych programów wymaga

oddzielnej pracy obu programów ze względu na to, że wymagają przy pracy tego samego obszaru pamięci RAM.

Edytor tekstowy EDYTOR 1 jest programem przeznaczonym do redagowania tekstów stanowiących zbiory w kodzie ASCII. W szczególności może on być wykorzystywany do przygotowania i poprawiania treści programów źródłowych pisanych w języku makroasemblera dla mikroprocesora Intel 8080. Program ten zajmuje około 3 K bajtów pamięci RAM i wymaga bufora tekstu co najmniej 12,5 K bajtów pamięci RAM.

W jednej komórce pamięci bufora tekstu jest przechowywany jeden znak tekstu, wielkość bufora jest wyznaczana przez pamięć RAM dostępną w systemie. Gdy w pamięci operacyjnej o pojemności 32 K bajtów rezyduje program makroasemblera wykorzystywana jest na bufor pamięć RAM wystarczająca do pamiętania ok. 20 000 znaków. Zwiększenie lub zmniejszenie pojemności pamięci operacyjnej w systemie, powoduje automatycznie zmianę wielkości bufora tekstu. Ciągi dyrektyw programu EDYTOR 1 przed wykonaniem są rejestrowane w pamięci RAM, tej samej, w której przechowywany jest bufor tekstu, lecz poczynając od drugiego końca dostępnej pamięci roboczej. Przy przekraczaniu dopuszczalnej wielkości bufora wprowadzanie znaków do bufora jest blokowane, wówczas należy odpowiednimi dyrektywami usunąć część tekstu z bufora. Użytkownik edytora tekstowego EDYTOR 1 ma do dyspozycji 16 dyrektyw, za pomocą których w sposób konwersacyjny wyznacza kolejne działanie tego programu. Każda dyrektywa oznaczana jest za pomocą pojedynczej litery. Dyrektywy są bezargumentowe lub z argumentem w postaci liczb dziesiętnych lub ciągu znaków. Użytkownik ma możliwość wprowadzania pojedynczych dyrektyw lub też ciągu dyrektyw, może poprawiać i kasować dyrektywy przy ich wprowadzaniu. Wykonywanie pojedynczych dyrektyw może być zwielokrotniane przez odpowiedni ich zapis.

Dyrektywy programu EDYTOR 1 (zestaw dyrektyw podano w tabeli 1) można podzielić na trzy klasy:

- dyrektywy, których podstawowym zadaniem jest zmiana położenia wskaźnika bufora (wskaźnik ten wyznacza określoną miejsce w ciągu znaków zawartych w buforze), bez wprowadzania zmian w buforze dyrektywy B, C, L, Z ,
- dyrektywy, które nie wpływają na wskaźnik bufora (dyrektywy N, P, T, X),
- dyrektywy, które zmieniają położenie wskaźnika w buforze i zawartość bufora (dyrektywy A, D, F, I, K, S, W).

Dla programu EDYTOR 1 współpracującego z programem makroasemblera REM dodatkowa dyrektywa Q służy do wywołania makroasemblera.

Program makroasemblera REM

Makroasembler REM jest programem przeznaczonym do translacji programów pisanych w języku makroasemblera, zgodnie ze standardami przyjętymi przez firmę Intel dla mikroprocesora 8080. Makroasembler REM umożliwia między innymi stosowanie mechanizmów makrogeneracyjnych oraz definiowanie odcinków programu translowanych warunkowo. Zestawienie pseudoinstrukcji w języku makroasemblera dla programu REM przedstawiono w tabeli II, zaś w tabeli III zestawienie stosowanych operatorów. Makroasembler REM zajmuje 9 Kbajtów pamięci RAM, a wraz ze współpracującym z nim programem EDYTOR 1 12 Kbajtów pamięci RAM o najniższych adresach.

Zmienne robocze makroasemblera, tablica symboli wraz z definicjami makroinstrukcji oraz tekst programu źródłowego podlegający translacji w całości lub też fragmencie w zależności od rozmiarów programu źródłowego są przechowywane w pamięci RAM. Dla zmiennych roboczych przewidziano 705 komórek pamięci RAM. W tablicy symboli są przechowywane nazwy symboli i odpowiadające im wartości

(adresy) oraz makrodefinicje. Pozostała część pamięci RAM przeznaczona jest na bufor dla programu źródłowego (w jednej komórce jeden znak) podlegającego translacji.

Makroasembler REM jest wieloprzebiegowy, co oznacza, że program źródłowy podlegający translacji musi być co najmniej dwa razy analizowany i przetwarzany w procesie translacji. Makroasembler bezpośrednio pobiera program źródłowy z bufora makroasemblera; jeśli w całości program źródłowy mieści się w buforze, nie ma potrzeby wprowadzania go ponownie.

W pierwszym przebiegu tworzona jest tablica symboli oraz ciągi znaków odpowiadające makrodefinicjom; nie wyprowadzane są żadne informacje. Drugi przebieg makroasemblera może generować wydruk programu źródłowego i kod wynikowy programu. Użytkownik może sterować przebiegiem pracy makroasemblera REM za pomocą wprowadzenia z konsoli numeru działania programu lub dyrektywy. Drugi przebieg translacji może być dowolnie powtarzany.

Numery działań wprowadzane z konsoli przez użytkownika mogą być poprzedzane parametrami literowymi, które spowodują:

- zatrzymanie działania programu REM przy napotkaniu błędu przy wyprowadzaniu wydruku z możliwością kontynuowania działania programu (parametr B),
- wyprowadzanie wydruku lub wierszy z błędami na drukarkę (standardowo wyprowadzane są na konsolę - parametr P),
- perforowanie tablicy symboli przy perforacji kodu wynikowego - parametr S.

Zestaw numerów działań i dyrektyw programu REM zawiera tablica IV.

Program REMAK

Program REMAK składający się ze współpracujących ze sobą programów: EDYTOR 1 i makroasembler REM pozwala na używanie obu tych programów oddzielnie. Przy wystartowaniu każdego z tych programów dyrektywą z MONITOR-a, program zgłasza się swą nazwą i wówczas bufor tekstu lub bufor programu źródłowego jest pusty. Przy przechodzeniu z programu EDYTOR 1 do programu REM dyrektywą Q (lub odwrotnie) dane umieszczane w buforze są dostępne dla każdego z tych programów, co pozwala na łatwe poprawienie translowanych programów.

Programy INTERPRETER BASIC i MINIBASIC

Programy INTERPRETER BASIC, MINIBASIC są interpreterami języka BASIC, realizują kilkanaście typowych dla tego języka instrukcji i podstawowe dyrektywy: wprowadzania i wyprowadzania z pamięci programu na taśmę perforowaną, start programu, wydruk programu, usuwanie programu z bufora. Można używać podstawowych operatorów arytmetycznych (+, -, *, /) oraz relacje $>$, $<$, $=$, $\neq$, $\geqslant$, $\leqslant$. Programy zawierają dwie specjalne funkcje umożliwiające wprowadzanie i wyprowadzanie danych na porty we/wy.

Program INTERPRETER BASIC zajmuje w pamięci RAM ~6 Kbajtów, wymaga bufora o pojemności 6 Kbajtów. Program składa się z dwu części: interpretera języka BASIC i integralnie z nim związanego pakietu zmiennego przecinka. Stałe zadeklarowane w programie lub liczby wprowadzane z konsoli są przetwarzane przez pakiet procedur zmiennego przecinka; liczby są przechowywane i mogą być wprowadzane z siedmiocyfrową dokładnością. Najmniejsza i największa liczba wynosi odpowiednio $\pm$ 2.71051E-20 i $\pm$ 9.22337E18. Dozwolone są tylko jednowymiarowe tablice o całkowitej liczbie elementów, określone za pomocą indeksów od 0 do 1. Nazwa tablicy musi być oznaczonajedną literą.

Zmienna nie będąca tablicą może być literą z jedną cyfrą.

Przy wprowadzaniu nielegalnej instrukcji sygnalizowany jest błąd, w razie wystąpienia błędu - podczas wykonywania programu na konsoli wypisywana jest informacja o błędzie z podaniem numeru błędu (nr 1 ÷ 14) i numeru linii. Podczas wykonywania programu i wprowadzania nowych linii do pamięci sprawdzana jest zajętość pamięci i sygnalizowane jest jej przepełnienie.

Program MINIBASIC składa się z części podstawowej, która może być zapisana w pamięci PROM i zajmuje ok. 3,25 Kbajtów oraz częśći roboczej dla pamiętania nazw używanych zmiennych i ich wartości, która zajmuje ok. 0,5 Kbajtów. Pozostała część pamięci operacyjnej wykorzystywana jest na bufor dla programu użytkownika. Liczby używane w programie muszą być liczbami całkowitymi z zakresu od 32767 do 32761, mogą być zapisane w kodzie heksadecymalnym (z literą H po liczbie) lub w kodzie dziesiętnym. Liczby wypisywane są przez program zawsze w kodzie dziesiętnym. Można używać 26 zmiennych 16-bitowych oznaczanych literami od A do Z, które mogą przyjmować wartości całkowite, można także tworzyć jednowymiarową tablicę. W programie MINIBASIC-a można używać podstawowych operatorów logicznych (NOT, AND, OR, XOR) i sześciu funkcji.

Instrukcja SEND i funkcja GET pozwalają na korzystanie z podprogramów (wprowadzonych do pamięci niezależnie od interpretera) z przekazywaniem danych w akumulatorze.

Funkcja PEEK i instrukcja POKE pozwalają na zapisywanie i odczytywanie danych do i z komórek pamięci. Funkcje ABS i RND obliczają wartość absolutną wyrażenia, generują liczby losowe od 1 do wartości wyrażenia. Funkcja SIZE podaje liczbę wolnych komórek pamięci RAM możliwych do wykorzystywania przez program użytkownika.

Linie programu wprowadzane bez numeru są wykonywane bezpośrednio z zapamiętywaniem wartości zmiennych. Ta cecha jest szczególnie przydatna, gdyż zmienne mogą być sprawdzone, zmieniane i przesyłane z urządzeń zewnętrznych. Interpreter MINIBASIC sygnalizuje trzy rodzaje błędów:

- zła składnia,
- niemożliwość wykonania operacji,
- przekroczenie zakresu pamięci.

Program EDYTOR 2

Najprostszy edytor tekstów - program EDYTOR 2, zajmujący - 1,5 Kbajtów pamięci RAM, służy głównie do poprawianie tekstu zapisanego na taśmie perforowanej w kodzie ASCII. W buforze EDYTOR-a 2 umieszczona jest co najwyżej jedna linia tekstu wejściowego.

Możliwe jest reperforowanie tekstu wejściowego, pomijanie części tekstu wejściowego i wyprowadzanie na taśmę perforowaną tekstu wprowadzanego z konsoli; można także otrzymywać wydruk poprawianego tekstu.

Program EDYTOR 2 wykonuje 10 następujących dyrektyw:

- M - powrót do początku programu EDYTOR 2,
- S - powrót do początku programu MONITOR,
- L - włącz listowanie tekstu źródłowego,
- NL - wyłącz listowanie,
- P - włącz perforację,
- B - perforuj 64 znaki puste,
- NP - wyłącz perforację,
- RT - czytaj i kopiuj zadaną liczbę linii lub do napotkanego stringu,
- TT - pomiń zadaną liczbę linii,
- WM - wyprowadź tekst wprowadzony z konsoli.

Test pamięci RAM - TEST RAM

Test pamięci RAM - program TEST RAM, służy do testowania pamięci operacyjnej MSWP. Testowany może być dowolny obszar pamięci RAM z wyjątkiem obszaru pamięci zajętego przez sam program (~1,75 Kbajtów pamięci o najmłodszych adresach) oraz obszaru pamięci stałej (4 Kbajtów pamięci o najwyższych adresach).

Program TEST RAM testuje zadany przez użytkownika obszar pamięci operacyjnej. Składa się on z 9 testów oddzielnie wywoływanych z programu. Są to testy:

- zerowanie całego zadanego obszaru pamięci ze sprawdzeniem,
- zerowanie kolejnych komórek pamięci z bezpośrednim sprawdzeniem,
- wpisywanie do komórek pamięci młodszych bajtów i ich adresu ze sprawdzeniem,
- wpisywanie do komórek zadanego obszaru pamięci jedynki na kolejne pozycje bitów,
- zadany obszar pamięci wypełniany jest "szachownicą zero-jedynkową" (tzn. na przemian zero i jeden) i jej negację ze sprawdzeniem bezpośrednio,
- wpisywanie wartości FFH do całego zadanego obszaru pamięci i sprawdzenie,
- kolejne jedynkowanie (wartością FFH) komórek pamięci w zadanym obszarze i bezpośrednie sprawdzenie,
- wpisywanie do całego zadanego obszaru pamięci zera, a następnie kolejnych kombinacji do wartości FFH i sprawdzanie po wpisaniu każdej wartości,
- zadany obszar pamięci wypełniony jest na przemian wartościami AAH i 55H i na polecenie operatora sprawdzane są te wartości lub po sprawdzeniu wpisywana ich negacja.

Testowanie pamięci operacyjnej poszczególnymi testami może być wykonywane wielokrotnie. W przypadku wykrycia błędu wypisywany jest adres komórki pamięci, w której wystąpił błąd, wartość prawidłowa i błędna oraz numery bitów z błędnymi wartościami.

Tab.1. Zestawienie alfabetyczne dyrektyw edytora testowego EDYTOR 1

Nazwa dyrektywy	Funkcja dyrektywy
A	Wprowadzenie określonej liczby linii tekstu z czytnika taśmy papierowej
B	Umieszczenie wskaźnika bufora na krańcu tekstu
C	Przesunięcie wskaźnika bufora o określoną liczbę znaków
D	Usunięcie określonej liczby znaków
E	Wydziurkowanie zawartości bufora tekstu i reperforacja taśmy
F	Wyszukanie ciągu znaków
I	Wystawienie tekstu
K	Usunięcie określonej liczby linii
L	Przesunięcie wskaźnika bufora o określoną liczbę linii
N	Wydziurkowanie pustych znaków
P	Wydrukowanie określonej liczby linii
S	Zamiana ciągu znaków
T	Wyprowadzenie określonej liczby linii na konsolę
W	Wydziurkowanie określonej liczby linii z początku bufora tekstu wraz ze skasowaniem ich
X	Wydziurkowanie określonej liczby linii
Z	Umieszczenie wskaźnika bufora na końcu tekstu

Tab. 2. Zestawienie alfanumeryczne pseudoinstrukcji /dyrektywy/ w języku makroasemblera dla programu REM

Oznaczenie pseudoinstrukcji	Funkcja
DB	Definiowanie 8-bitowych danych - bajtu lub ciągu bajtów
DS	Rezerwacja obszaru pamięci
DW	Definiowanie 16-bitowych danych - pary komórek lub ciągu par komórek pamięci
END	Zaznaczenie końca translowanego programu
ENDIF	Zaznaczenie końca odcinka programu translowanego warunkowo
ENDM	Zaznaczenie końca definicji makroinstrukcji
EQU	Definiowanie wartości symbolu bez możliwości zmiany wartości
IF	Zaznaczenie początku odcinka programu translowanego warunkowo
MACRO	Określenie początku definicji makroinstrukcji wraz z nazwą i ewentualnie parametrami formalnymi
ORG	Ustalenie adresu komórki pamięci, od której zaczyna się umieszczenie kodu wynikowego
SET	Definiowanie wartości symbolu z możliwością zmiany wartości
TITLE	Nadawanie tytułu stronie wydruku

Tab. 3 . Zestawienie operatorów stosowanych w wyrażeniach w języku makroasemblera dla programu REM

Oznaczenia operatora	Znaczenie operatora
Operatory arytmetyczne	
+	dodawanie
-	odejmowanie
x	mnożenie
/	dzielenie
MOD	modulo
Operatory logiczne	
NOT	negacja logiczna
AND	iloczyn logiczny
OR	suma logiczna
XOR	różnica symetryczna
Operatory przesunięcia	
SHL	przesunięcie w lewo
SHR	przesunięcie w prawo

Tab.4. Zestawienie numerów działań i dyrektyw programu REM wprowadzonych przez użytkownika w odpowiedzi na "P="

	Nr działania programu lub dyrektywa	Dopuszczalne dodatkowo parametry literowe	Funkcje programu REMAKAS-80	Uwagi
Działania	1	-	Pierwszy przebieg translacji: przygotowanie tablicy symboli i rejestracja makrodefinicji	Konieczny przy rozpoczynaniu translacji lub przed wyprowadzeniem tablicy symboli dyrektywą T
	2	B i P	Wyprowadzenie tylko wydruku	
	3	S	Perforowanie kodu wynikowego i ewentualnie tablicy symboli	Wprowadzenie w dowolnej kolejności po działaniu nr 1
	4	B, P i S	Wyprowadzenie wydruku i perforowanie kodu wynikowego /ewentualnie tablicy symboli/	
	5	B i P	Wyprowadzenie wierszy wydruku zawierających błędy	
Dyrektywa	T	P	Wyprowadzenie tablicy symboli na konsolę lub drukarkę	

x/ Parametr P jest skuteczny w działaniu, jeśli uprzednio w monitorze systemu zadeklarowano drukarkę jako urządzenie listujące

sprawozdania

Sprawozdanie

z II Konferencji Technicznej Zrzeszenia Przedsiębiorstw MERA

II Konferencja Techniczna Przedsiębiorstw Zrzeszenia Mera odbyła się w Poświętnem koło Płońska, w dniach 15-17 października br. Tematem konferencji były sprawy związane z bazą elementową i podzespołową w przedsiębiorstwach Zrzeszenia zajmujących się produkcją automatyki i aparatury pomiarowej oraz trudności technologiczne spowodowane słabym poziomem rozwoju technologii brakami sprzętowymi i elementowymi.

W obradach udział wzięło 31 osób z 15 przedsiębiorstw; wygłoszono 11 referatów, zaś referat dr inż. E. Gardiasa ze Zrzeszenia Mera "Przegląd podzespołów mikroelektronicznych opracowywanych i produkowanych w krajach RWPG dla potrzeb środków techniki obliczeniowej, automatyki i aparatury pomiarowej" organizatorzy obiecali przedstawić przy okazji Rady DT i tą drogą udostępnić członkom konferencji. Tematyka referatów była generalnie ukierunkowana na omówienie zagadnień nowych opracowań i wdrożeń w aspekcie potrzeb nowoczesnej bazy elementowej i podzespołowej oraz nowoczesnej technologii. Wykaz tematów w załączniku nr 1. Ponadto pracownicy Biura Zrzeszenia przedstawili informację o pracach w dziedzinie analizy technologii stosowanych w zakładach Zrzeszenia oraz o pracach prognostycznych w zakresie automatyki

i aparatury pomiarowej. Ośrodek Badawczo-Rozwojowy Elektronicznych Układów Specjalizowanych "MERA-OBREUS" z Torunia zaprezentował swoje wyroby, zarówno produkowane seryjnie, jak i opracowywane oraz planowane do opracowania i wdrożenia do produkcji do roku 1990. W trzech referatach przedstawili:

- diody referencyjne skompensowane temperaturowo typu BZY566A do BZY584A /12 typów/, przeznaczone do pracy w wysokostabilnych układach zasilających o dużej niezależności napięcia wyjściowego od napięcia zasilania, rezystancji obciążenia i temperatury;

- mikroukłady hybrydowe H0050, H0081, H0020, z których pierwszy stosowany jest jako odbiornik interfejsu ODRA1305, drugi - jako nadajnik interfejsu ODRA 1305 zaś trzeci jest przeznaczony do pracy w stopniu wejściowym odbiornika interfejsu JS EMC;

- zespoły rezystorów cienkowarstwowych ZRC898-1 i ZRC898-3, które znajdują zastosowanie w cyfrowych i analogowych urządzeniach automatyki i pomiarów;

- układ scalony typu DPACA-10, który jest 10-bitowym przetwornikiem cyfrowo-analogowym, przetwarzającym binarne sygnały TTL na napięcie lub prąd, przeznaczonym do pracy w systemach sprzężeń komputerów lub innych urządzeń cyfrowych z siłownikami, zainstalowanymi w obiektach;

- hybrydowy układ scalony typu HPCA-12, który jest 12-bitowym przetwornikiem analogowo cyfrowym, również /jak DPACA-10/ przeznaczonym do urządzeń automatyki w systemach sprzężenia komputerów z obiektami;

- hybrydowe układy scalone typu HPDC i typu 12-12, będące przetwornikami mocy prądu stałego pozwalającymi uzyskać przetworzenie napięcia stałego, przy czym układ HPDC - z zachowaniem izolacji galwanicznej wejścia i wyjścia; przeznaczone są do systemów sprzężenia komputerów z obiektami, zaś układ 12-12 ponadto do zasilania układów transmisji sygnałów cyfrowych;

- poza tym przedstawiono następujące typy układów hybrydowych:
  - mnożąco-dzielący typ AM-1,
  - nieliniowy układ funkcji półtrapezowych typu AN-1,
  - źródło napięcia typu AU-1,
  - wzmacniacz uniwersalny typu AW-1;

- zintegrowaną głowicę do drukarki termograficznej robioną na zamówienie MERA-BŁONIE;

- przybory mikromechaniczne z krzemu monokrystalicznego: czujniki ciśnienia, przyśpieszeniomierze, termometry itp., które są produkowane dlatego, że krzem monokrystaliczny ma bardzo interesujące własności, technologia mikromechaniczna i mikroelektroniczna są jednolite do momentu głębokiej obróbki chemicznej w obróbce mikromechanicznej, jonowej lub litografii elektronowej w obróbce mikroelektronicznej. Za pomocą obróbki mikromechanicznej można uzyskać detale nieosiągalne w innych technologiach.

Charakterystykę asortymentową potrzeb przedsiębiorstw MERA kompleksowo ujął przedstawiciel IKSAIP Zbigniew Mroziński. Ponadto potrzeby przedstawione w tym referacie są skorelowane z potrzebami w zakresie bazy elementowej i podzespołowej przedsiębiorstw produkujących sprzęt komputerowy. Dlatego ten referat omówimy bardziej szczegółowo.

POTRZEBY IKSAIP W ZAKRESIE BAZY ELEMENTOWEJ I ROZWOJU TECHNOLOGII do 1995 r.

Instytut Komputerowych Systemów Automatyki i Pomiarów zajmuje się obecnie następującymi dziedzinami:

- automatyzacją prac w obiektach przemysłowych takich, jak elektrownie, cukrownie, papiernie, kopalnie, przetwórnie itp.,
- automatyzacją prac w kolejnictwie,
- pomiarami własności fizykochemicznych,
- pomiarami składu cieczy i gazów.

Urządzenia konstruowane przez Instytut w ciągu najbliższego dziesięciolecia będą w stosunku do sprzętu produkowanego obecnie odznaczać się następującymi parametrami:

- co najmniej o rząd większą dokładnością pomiarów,
- o około 30% mniejszym poborem energii,
- o rząd większą niezawodnością,
- krótszym czasem pomiarów i czasem reakcji,
- większą odpornością na zakłócenia,
- znacznie większymi możliwościami funkcjonalnymi.

Ponadto od sprzętu pomiarowego będzie wymagana automatyczna analiza i obróbka otrzymanych wyników. Z tego też względu sprzęt pomiarowy będzie wspomagany mikrokomputerowo.

Potrzeby IKSAIP można podzielić na pięć grup:

○ Elementy półprzewodnikowe analogowe

W tej grupie elementów wymienić należy wzmacniacze operacyjne i analogowe, przetworniki C/A i A/C, źródła napięć wzorcowych, tranzystory itp.

Wzmacniacze operacyjne powinny charakteryzować się bardzo małym prądem wejściowym ok. 1 pA lub jeszcze mniej, powinny mieć duże wzmocnienie ok. 40 K, pożądane jest też aby były mało wrażliwe na zmiany temperatury otoczenia oraz na zmiany napięcia zasilania. Do obróbki sygnałów impulsowych niezbędny będzie szerokopasmowy wzmacniacz analogowy o poziomie 100-300 MHz i wzmocnieniu do 1 K, jako przykład może służyć wzmacniacz AD500 firmy Signetics.

Przetworniki A/C i C/A będą potrzebne głównie 8- i 16-bitowe, łącznie z drabinkami rezystancyjnymi. Charakterystyczne wielkości wejściowe i wyjściowe to odpowiednio 0,1 V i 4 lub 20 mA, oporność wejściowa przetworników jaknajwiększa - ok. 10^{16} omów. Czas przetwarzania nie powinien przekraczać 5 µs, a ich liniowość nie powinna być gorsza niż $\pm$ 0,01 %. Ponadto interesujące są przetworniki CMOS z wyjściem do sterowania 3 1/2 cyfrowego pola odczytowego.

○ Elementy półprzewodnikowe cyfrowe

Wobec generalnego przejścia w sprzęcie produkowanym i opracowywanym u nas, na końcową obróbkę mikrokomputerową, okazują się niezbędne:

- kompletne zestawy mikrokomputerów 8 bitowych, np. serie Intel 8080, 8085,
- sterowniki urządzeń zewnętrznych do tych zestawów,
- specjalizowane układy mikrokomputerów jednopłytkowych, 8-bitowych, np. odpowiednik 8035,
- mikrokomputery z wyjściami analogowymi,
- kompletne zestawy mikrokomputerów 16-bitowych, odpowiedniki Intel 8086, 8087 i 8088,
- kompletne zestawy mikrokomputerów segmentowych - odpowiedniki 2900,
- wszelkiego typu pamięci półprzewodnikowe - RAM, PROM, EPROM dynamicznie, od małych pojemności do dużych ok. 256 K,
- ponadto będziemy również sięgać po minikomputer 32-bitowy.

Sądzimy, że niektóre problemy uda się rozwiązać za pomocą opracowywanych przez CEMI na zlecenie IKSAIP układów matrycowych.

○ Elementy optoelektroniczne

Oferowany przez CEMI i POLAM zestaw elementów optoelektronicznych nie wystarczy dla rozwoju naszego sprzętu. Brakuje dużych alfanumerycznych pól odczytowych do zastosowań tablicowych. Wykonywane one powinny być jako diody LED lub

ciekłokrystaliczne. Nie ma rynku pól kombinacyjnych do zastosowań semantycznych - np. dla oznaczenia kierunku przepływu na pulpitach operatorskich lub innych tablicach. Brakuje ekranów półprzewodnikowych z możliwością wyświetlania tekstu, a także zestawu elementów do łączności światłowodowej, która jest potrzebna w obiektach z dużym poziomem zakłóceń. Do zestawu takiego powinny należeć nadajniki, odbiorniki, sprzęgacze, linie transmisyjne, wzmacniacze itp. Duże doświadczenia pod tym względem mają NRD i ZSRR.

Potrzebne też będą, i to już niedługo, lampy kineskopowe kolorowe nadające się do zastosowań w monitorach ekranowych. Powinny się one odznaczać dużą gęstością linii - do 900, oraz małymi zniekształceniami.

O Elementy bierne

Oferowane od tej pory rezystory stałe i zmienne już w chwili obecnej częstokroć mają parametry nieodpowiednie. Brak jest na rynku rezystorów wielkiej mocy. Przypuszczamy, że w przyszłości niezbędne będą rezystory stałe o tolerancji poniżej 0,1% w zakresie od 5 omów do 10 Momów i mocy od 1/8 W do 1W. Ich współczynniki TWR winny być na poziomie 10^{-6}.

Niezbędne będą trymery montażowe podobne do typu CT32 lecz mniejsze i o lepszej jakości. Konieczne będą wysokostabilne potencjometry precyzyjne jedno- i wieloobrotowe.

Do zasilaczy będą niezbędne bezindukcyjne elektrolity w zakresie od 6800 uF do 47000 uF na napięcia 10, 16, 25 i 40 V. Odpowiednikiem mogą być kondensatory Philipsa.

Konieczne będą też kondensatory polistyrenowe o dużej izolacji - ok. 5.10^{11}. Brakuje dokładnych kondensatorów przeciwzakłóceniowych.

○ Galanteria elektroniczna

W tej grupie należy wymienić przede wszystkim wszelkiego typu podstawki pod układy scalone LSI i VLSI - zwłaszcza pod elementy mikroprocesorowe i układy pamięciowe. Brakuje złącz pośrednich do płytek drukowanych o małej liczbie styków. Pod tym względem należy rozpatrzyć asortyment firmy Canon. Brakuje również łączówek do pasm, jak również jedno- i wielostykowych złącz w ekranie o dużej oporności izolacji ok. 10^{13}. Niezbędne będą miniaturowe przełączniki obrotowe oraz segmentowe przechylne, przystosowane do przenoszenia bardzo małych prądów i niskich napięć, odznaczające się dużą opornością izolacji. Dla urządzeń montowanych na obiektach niezbędne będą złącza o napięciu izolacji 2 kV.

Potrzeby w zakresie rozwoju technologii

Biorąc pod uwagę rozwój bazy elementowej oraz rozwój konstrukcji niezbędnej staje się prowadzenie rozwojowych prac technologicznych w niżej omówionych kierunkach.

○ Prace montażowe

Przewiduje się, że w ciągu najbliższych 10 lat utrwali się tendencja do stosowania płaskiego montażu powierzchniowego. Podyktowane to będzie obniżką pracochłonności, zmniej-

szeniem powierzchni, wzrostem niezawodności sprzętu. jak również wzrostem odporności na narażenie mechaniczne. Należy więc rozwijać prace w kierunku opracowania technologii takiego montażu, opracowywać materiały do lutowania, topniki, narzędzia, zarówno do produkcji seryjnej, jak i małoseryjnej.

○ Płytki montażowe

Szybkie wykonanie wielowarstwowe płytki drukowanej jest obecnie bardzo trudne. Obecne parametry tych płytek będą w przyszłości niewystarczające. Konieczne będzie zmniejszenie szerokości ścieżek, zmniejszenie odstępu między ścieżkami. Dlatego bardzo ważną rzeczą jest przekazanie zakładom montażowym oraz instytutom takim, jak IKSAIP możliwości wykonania płytek wielowarstwowych dla celów prototypowych.

○ Procesy wspomagania konstrukcji i procesy pomiarowo-kontrolne

Stosowane do tej pory metody opracowywania konstrukcji sprzętu są archaiczne, żmudne i długotrwałe. Konstruktor nie dysponuje sprzętem, który ułatwiłby mu czynności przy opracowywaniu konstrukcji. Nie może on również symulować /np. za pomocą komputera/ urządzenia, aby sprawdzić główne założenia konstrukcyjne. Należy więc rozwijać prace zmierzające do przekazania konstruktorowi takich narzędzi.

Odrębną sprawą są możliwości pomiarowe. Potrzeby konstruktorów nie są zaspakajane. Należy również podkreślić,

że stan handlowej aparatury pomiarowej jest wprost katastrofalny /woltomierze, oscyloskopy, amperomierz itp./, szczególnie aparatury o dużej dokładności, przeznaczonej do pomiarów i legalizacji.

Rozwój techniki mikroprocesorowej sprzyja temu, aby układy pomiarowe dla tej techniki były wykonywane centralnie. Opracowywane przez poszczególne zakłady narzędzia pomiarowe niepotrzebnie rozpraszają siły i środki które należałoby lepiej spożytkować. Postulujemy więc, aby sprawą tą zajęto się centralnie - np. przez producentów bazy mikroprocesorowej. Przypuszczamy, że narzędzia kontrolno-pomiarowe też będą oparte na technice mikrokomputerowej, a więc zapewnienie odpowiedniego oprogramowania systemowego i użytkowego będzie sprawą ważną.

○ Podsumowanie

Braki na rynku elementów i podzespołów elektronicznych są i będą bolączką nie tylko IKSAIPU lecz również WZE ELWRO. Dlatego podawane wyżej potrzeby należy liczyć ilościowo w skali produkcji co najmniej seryjnej.

Innym aspektem jest profesjonalny a nie powszechnego użytku charakter środków automatyki i aparatury pomiarowej. Taka też musi być baza elementowa do nich. Dlatego interesuje nas baza do zastosowań profesjonalnych, z rozszerzonymi parametrami mechaniczno-klimatycznymi oraz z lepszą niezawodnością i jakością. Tymczasem jesteśmy zmuszeni do kupowania i używania elementów dla powszechnego użytku.

To zmusza nas do rozbudowywania ponad potrzeby stanowisk kontrolno-pomiarowych jak również do wstępnego starzenia elementów.

Ponadto nasi odbiorcy żądają często aby nasze urządzenia spełniały wymagania transportowe w zakresie temperatur od - 65°C do + 65°C. Również potrzeby eksportu do krajów tropikalnych stawiają nowe wymagania naszemu sprzętowi. Te wymagania dyktują odpowiednie wymagania w zakresie bazy elementowej, zarówno tropikalizowanej, jak i przenoszącej tak szeroki zakres temperatur. Ale elementów takich dotychczas nasz przemysł nie produkuje.

W dyskusji, która wynikła po wysłuchaniu referatów, zwłaszcza po informacji wygłoszonej przez przedstawiciela Biura Zrzeszenia na temat analizy technologii stosowanych w przedsiębiorstwach MERA podkreślono, że warto się zastanowić nad możliwością częściowego montażu powierzchniowego, w miarę jak elementy będą przystosowywane do takiego montażu. Na temat wykonywania wielowarstwowych płytek drukowanych, zebrani mogli w przedstawionej informacji usłyszeć o ofercie strony czechosłowackiej na sprzedaż technologii "multiwire" wraz z urządzeniami. W tej technologii podstawą jest płytka ze szkła epoksydowego, laminowana folią miedzianą i pokryta warstwą klejową. Wcześniej na płytce są wytrawione obwody zasilania, pola uziemień oraz końcówki stykowe. Na płytce takiej układa się przewody o średnicy 0,125 mm w podwójnej izolacji, co umożliwia krzyżowanie ich oraz

układanie wielowarstwowe. Po ułożeniu schematu, przewody zostają unieruchomione i utwardzone na prasie, w odpowiedniej temperaturze. W punktach wymagających połączenia między przewodami poszczególnych warstw lub zamocowania i połączenia podzespołów wiercone są otwory i prowadzi się metalizację, podobnie jak przy produkcji płytek drukowanych. Podobnie też przeprowadza się montaż elementów i ich lutowanie. Przewody w tej metodzie układa się po jednej lub po obu stronach płytki.

Referat dr E. Gardias "Przegląd podzespołów mikroelektronicznych..." będzie zamieszczony w jednym z najbliższych numerów Biuletynu MERA /licząc od listopada 1984 r./ toteż nie będę go tu omawiał.

Komisja wnioskowa powołana spośród zebranych, opracowała wnioski i postulaty, które w dalszym ciągu zostaną przytoczone.

Wnioski

● Stwierdza się, że dostępna dla przedsiębiorstw MERA baza podzespołowa jest stanowczo niewystarczająca pod względem asortymentowym, jakościowym, a co gorsza - brakuje podzespołów w wykonaniu profesjonalnym.

● Przedsiębiorstwa są zdecydowane zrezygnować z dostaw elementów i podzespołów z importu z II obszaru płatniczego - świadomie rezygnując z wyższych parametrów technicznych opracowywanych urządzeń, na rzecz importu z krajów RWPG. Z dotychczasowych doświadczeń wynika jednak, że jakość ele-

mentów i terminowość dostaw pozostawia wiele do życzenia; mocno podkreślono też brak rzeczowych informacji na temat nowych wyrobów podzespołowych opracowywanych i wdrażanych w krajach RWPG.

● Stosowane obecnie w przedsiębiorstwach MERA technologie montażu są pracochłonne, o niskim stopniu zautomatyzowania. Uczestnicy spotkania podkreślili niedostatki w wyposażeniu zakładów w podstawowe linie technologiczne, w szczególności w linie do produkcji obwodów drukowanych; możliwość zastosowania bardziej efektywnych technologii typu montaż powierzchniowy jest uwarunkowana odpowiednią bazą elementową i oprzyrządowaniem technologicznym.

● Należy znacznie większą rangę nadać problemom testowania elementów, podzespołów i wyrobów oraz podjąć zdecydowane działania zmierzające do wyposażenia fabryk w odpowiednią aparaturę testującą niezbędną na różnych etapach cyklu produkcyjnego.

W świetle powyższych wniosków zebrani wysunęli liczne postulaty.

● Należy podjąć intensywne działania prowadzące do rozwoju własnej bazy produkcyjnej w zakresie podzespołów elektronicznych, w pierwszej kolejności powinna być doinwestowana OBR MERA OBREUS, zwłaszcza w zakresie uruchomienia produkcji układów matrycowych.

● Niezależnie od rozwoju MERA OBREUS, należy dążyć w ramach Zrzeszenia do zorganizowania bazy produkcyjnej także innych podzespołów oraz urządzeń technologicznych.

IKSAIP interesują multipleksery analogowe 16 kanałowe. Wzorcami w rodzinie takich multiplekserów mogą być wyroby firmy National Semiconductor: LF 351, LF 13508N, LF 13201N, LF13202N. Zbliżonym do nich jest ośmiokanałowy multiplekser TMX 18PC z WRL.

Innym elementem, który mógłby być produkowany w kraju jest detektor pojemnościowy: można by zastosować go w opracowywanej przez IKSAIP klawiaturze pojemnościowej.

W sprzęcie pomiarowym niezbędne jest stosowanie scalonego źródła napięcia wzorcowego. Zadowalający byłby analog układu AD 584 VH z firmy Analog Devices.

Spośród dyskretnych elementów półprzewodnikowych niezbędne będą do opracowań IKSAIP następujące:

- tranzystory polowe z kanałem typu P,
- szybkie diody mocy o parametrach 5-30 A i 50V - np. 1N5833 firmy Motorola,
- tranzystory mocy wykonane w technologii SIP MOS - np. BUZ firmy Siemens,
- układy Darlingtona o mocy 0,3 W przy 100 V,
- triaki sterowane fotoelektrycznie na napięcia do 600 V i prądy do 10 A,
- tyrystory w układzie mostkowym,
- diody Zenera o prądzie 1 mA jako źródła napięcia odniesienia.

● Istnieje konieczność podjęcia w ramach zrzeszenia właściwych działań zmierzających do:

- zobowiązania CEMI do dostarczenia zakładom MERA podzespołów w wykonaniu profesjonalnym,
- dotrzymania przez CEMI zobowiązań wobec przedsiębiorstw MERA w zakresie nowych uruchomień podzespołów, szczególnie układów z serii MCY,
- dopracowania formy współpracy z CEMI w zakresie jakości i terminowości dostaw, obwarowanych konsekwencjami finansowymi dla CEMI z tytułu niedotrzymywania przyjętych zobowiązań;

● W zakresie złącz i obwodów drukowanych podjęcie natychmiastowych działań w celu zaspokojenia potrzeb przedsiębiorstw co najmniej do roku 1990;

● Niezależnie od wymienionych doraźnych działań na styku z jednostkami UNITRA, stwierdza się konieczność podjęcia zdecydowanych kroków w ramach Zrzeszenia MERA na rzecz uniezależnienia się w maksymalnym stopniu od dostaw i współpracy kooperacyjnej z przedsiębiorstwami UNITRA;

● Postuluje się istotne skrócenie terminu dostaw próbek i wzorców elementów z krajów RWPG do celów prac modelowo-badawczych (przed dostawami próbek powinna być udostępniana pełna informacja techniczna o powyższych podzespołach);

● Proponuje się rozważenie możliwości wykorzystania do tego celu agend PHZ METRONEX działających w krajach RWPG.

mgr inż. Ignacy STREMBICKI
Instytut Maszyn Matematycznych

nowości techniczne

Nowe mikrokomputery IBM we Francji

Firma IBM France Diffusion oferuje najnowszą wersję nowego komputera osobistego PC/AT. Maksymalna pojemność pamięci operacyjnej może tu osiągnąć 3 Mbajty, a pamięci zewnętrznej 40 Mbajtów, przy czym może on być używany w wersji autonomicznej lub z wieloma końcówkami. Stosowany jest też mikroprocesor 32-bitowy Intel 80286, lecz zapewniona jest kompatybilność z większością oprogramowania opracowanego dla dotychczasowych modeli. Obecnie można zamawiać dwie konfiguracje: model 1 o pojemności pamięci operacyjnej 256 kbajtów, z czytnikiem dyskietek o pojemności jednostkowej 1,2 Mbajta za 45467 franków i model 2 z pamięcią operacyjną 512 kbajtów zawierający oprócz czytnika dyskietek stację dyskową o pojemności jednostkowej 20 Mbajtów za 62332 franki. Jednocześnie oferuje się nowe systemy operacyjne DOS 3.0 i DOS 3.1, które są nowymi wersjami systemu DOS 2.11 oraz Xenix firmy Microsoft będący ulepszoną wersją Unix-a. Ten ostatni pozwala na pracę na 3 końcówkach w systemie wielozadaniowym. Wszystkie trzy systemy będą dostępne w końcu pierwszego kwar-

tału 1985 r. Cena dwóch pierwszych wynosi 719 franków, a Xenix-a - 4418 franków.

Micro-Systemes nr 47/84

Nowe komputery firmy ACT we Francji

Szkocka firma ACT, znana z komputera Apricot, oferuje na na rynek francuski nową wersję tego komputera o nazwie Apricot F1, który w minimalnej konfiguracji /128 Kbajtów pamięci, ekran jednobarwny/ kosztuje poniżej 12 tys. franków, a w wersji średniej z "myszką", kolorowym monitorem i dobrym oprogramowaniem może konkurować z Mcintoshem firmy Apple. Interesującym rozwiązaniem jest połączenie między jednostką centralną a klawiaturą za pomocą światła podczerwonego.

Wersja przenośna Apricot ma ekran na ciekłych kryształach /25 wierszy po 80 znaków/ oraz system rozpoznawania mowy /rozróżnia 20 tys. słów/. Używa się tu systemu operacyjnego MS-DOS 2.1. Firma oferuje też dwa rodzaje sieci lokalnych - jeden do niniejszych zastosowań z 7 punktami wymiany i większy z 32 stanowiskami do jednoczesnej pracy

Micro Systemes nr 45/84

Minikomputer przenośny firmy Panasonic

Model RL-H 7000W może realizować większość programów napisanych na minikomputer IBM. Wymiary jego wynoszą 470x335x210 mm, awaga nie przekracza 15 kg. Zawiera on ekran o przekątnej 23 cm, drukarkę, dwa czytniki dyskietek o średnicy 5 1/4 cala, wejścia-wyjścia i podzespół koloru dla zewnętrznego zobrazowania na monitorze RVB. Niezależna od systemu klawiatura dołączona jest do jednostki centralnej za pomocą kabla telefonicznego. Przy transporcie służy ona jako osłona zabezpieczająca ekran i czytniki.

Jako mikroprocesor zastosowany jest tu Intel 8088, a dodatkowo może być użyty 8087. Pamięć operacyjna ma pojemność 256 kbajtów. Klawiatura zawiera 83 przyciski, blok cyfrowy 10 przycisków funkcyjnych. Wyświetlanie przy systemie alfanumerycznym zawiera 25 wierszy o 40 lub 80 kolumnach, przy systemie graficznym 640 x 200 punktów. Pojemność jednostkowa dyskietek wynosi 360 kbajtów. Model RL-H 7000/100 zawiera jedną stację dyskietek i jedną zwykłych dysków. Cena tego modelu wynosi 45 tys. franków, a RL-H 7000 W - 28 tys. franków. Jako system operacyjny używany jest MS-DOS 2.11, dostępne jest pełne oprogramowanie działające pod tą wersją systemu.

Micro-Systemes nr 47/84

Nowy komputer firmy Morrow Inc.

Na wystawie związanej z National Computer Conference, która odbyła się w lipcu 1984 r. w Las Vegas, firma Morrow Inc. z San Leandro zademonstrowała nowy komputer Tricep, oparty na UNIX-ie /System V/, na którym może pracować 4-8 użytkowników. Zastosowanie 3-portowej architektury DMA /Direct Memory Access - bezpośredni dostęp do pamięci/ umożliwia porozumiewanie się sterownika urządzeń wejścia-wyjścia i sterowników stacji dyskowych bezpośrednio z pamięcią operacyjną. Jednostka centralna zbudowana jest na mikroprocesorze 68000 o częstotliwości zegara 10 MHz, a procesory podległe na 80188, przy czym pracują one w systemie MS-DOS i mają po 128 do 512 Kbajtów dwuportowej pamięci operacyjnej każdy. Ponadto system obejmuje stację dysków 16-Mbajtowych i stację dysków elastycznych, przy czym pamięć operacyjna może być powiększona do 2 Mbajtów, a dyskowa do 4 dysków 34-Mbajtowych. Połączenia wewnętrzne realizowane są na szynie S-100. Najprostszy system kosztuje 9000 dol.,

a przy większych zamówieniach cena spada do 5500 dol.

EDN nr 16/84
Byte nr 8/84

Komputer osobisty IBM do projektowania układów o dużej skali integracji

Firma Micro Linear Corp. opracowała oprogramowanie na IBM PC, które wykorzystując graficzne i symulacyjne własności tego komputera pozwala na projektowanie liniowych i cyfrowych układów o dużej skali integracji. Oprogramowanie to obejmuje bibliotekę, którą można dalej rozszerzać, a która przyspiesza znacznie projektowanie układów liniowych. Oprogramowanie to kosztuje 7900 dol. i może być stosowane w sieci Cybernet, stosowanej do symulowania układów zawierających ponad 140 tranzystorów.

EDN nr 16/84.

Systemy dla wielu użytkowników

Visual Technology Inc. oferuje system oparty na mikroprocesorze 80286 pracującym z systemem operacyjnym XENIX. Posiada on 512 Kbajtów pamięci operacyjnej, stację dysków 19 Mbajtowych i stację taśmy. Przewidziany jest do prac dla 6-16 użytkowników. Cena kształtuje się w zakresie od 10 do 15 tys.dol.

Brytyjska firma Polebrock Computer Systems zakończyła opracowywanie systemu z mikroprocesorem 68000 opartego na UNIX-ie, który z pamięcią operacyjną 256 Kbajtów i stacją dysków 10 Mbajtowych będzie kosztował 2750 dol.

Byte nr 8/84

Sinclair QL na rynku francuskim

Na jesiennej wystawie SICOB w Paryżu zademonstrowano nowy minikomputer firmy Sinclair oznaczony jako QL. We Francji sprzedażą tego komputera zajmuje się firma Direco. Jest to system nieco bardziej rozbudowany w stosunku do innych komputerów osobistych, a jednocześnie cena jego /6 tys. franków/ nie jest zbyt wysoka, uwzględniając możliwości i oprogramowanie. To ostatnie obejmuje 4 grupy programów: Abacus - do analizy obliczeń, Archiwe - do zarządzania bazami danych, Easel - do grafiki i Quill - do przetwarzania tekstów. Udogodnienia sprzętowe obejmują m.in. dwie mikrostacje dyskowe o pojemności jednostkowej 100 Kbajtów w konfiguracji podstawowej, wkładki pamięci stałej, sieć lokalną, układy sterowania i sprzęgające. Od października 1984 r. sprzedawana jest wersja angielska QL, czas dostawy wynosi kilka tygodni. Na początku roku 1985 pojawi się wersja francuska, dla której obecnie opracowywana jest klawiatura i oprogramowanie, wówczas dostawy będą znacznie szybsze.

Micro-Systemes nr 47/84

Lekki minikomputer przenośny

Data General oferuje swój mikrokomputer Data General/One, który jest lżejszy o 4 kg, kompatybilny z PC IBM i działa pod systemami operacyjnymi MS-DOS, CP/M86 i Venix /wersja UNIX-a firmy ITT/. Wyświetlenia tekstów i obrazów dokonuje się na ekranie ciekłokrystalicznym o parametrach takich, jak dla ekranów konwencjonalnych. Jako mikroprocesor stosowany jest tu układ 80C88 będący wersją C-MOS układu 8088. Pamięć operacyjna ma pojemność 128 Kbajtów z możliwością powiększenia do 512 Kbajtów, pamięci stałych 64 Kbajty. Klawiatura typu maszyny do pisania z 79 przyciskami i 10 przyciskami funkcyjnymi.

Jako pamięć zewnętrzna mogą być dwie stacje dysków elastycznych 3 1/2 cala o pojemności jednostkowej 720 Kbajtów lub dysków zwykłych 5 1/4 cala o pojemności jednostkowej 720 Kbajtów lub dysków zwykłych 5 1/4 cala. Jako języki stosowane są tu: Pascal, Fortran, C i Basic. Dodatkowo można zamawiać baterie z układem ładowania, modem, drukarkę, ochronę do transportu. Oprogramowanie obejmuje programy działające pod wymienionymi systemami operacyjnymi, w tym także znane, jak Multiplan, Lotus 1-2-3, Symphony, Friday, Supercalc. Cena 29300 franków.

Micro-Systemes, nr 47/84

Minikomputer wizyjny firmy Sony

Minikomputer o nazwie SMC-70 GP "Genlocker" umożliwia synchronizację z dowolnym zewnętrznym źródłem wizyjnych /kamera, generator efektów, magnetoskop, czytnik dysków wizyjnych/. Można nanosić na obraz wizyjny tekst lub rysunki w kolorach wywytwarzane przez minikomputer i rejestrować całość bezpośrednio na kasetę wizyjną. Zastosowano tu miniprocesor Z 80A i pamięć wewnętrzną 64 Kbajty plus 38 Kbajtową pamięć ekranu wizyjnego. Sprzęt obejmuje klawiaturę 72-przyciskową i podwójną stację dysków elastycznych o średnicy 3,5 cala i pojemności jednostkowej 280 Kbajtów. Głównymi funkcjami SMC 706P jest tworzenie tekstów, rysunków i grafiki w 16 kolorach, przez klawiaturę i tablicę graficzną, gromadzenie danych na mikrodyskietkach, wzbogacenie tekstów i obrazów. Mikrokomputer wyposażony jest w różne sprzęgi wejścia-wyjścia, pozwalające na dołączanie dodatkowych urządzeń zewnętrznych, jak moduł inkrustacji i kodowania PAL SMI 7074, stacja dyskowa 7050, tablica graficzna, ołówek optyczny i moduł digityzacji obrazu w 16 kolorach.

Oprogramowanie jest łatwe w użyciu: u dołu ekranu pojawiają się napisy pomocnicze. Wyboru dokonuje się za pomocą naciśnięcia jednego z pięciu klawiszy funkcyjnych. Program "Video Titler" pozwala na wytwarzanie znaków w 16 kolorach i 6 wykrojach czcionek lub tworzenie własnego alfabetu, a "Graphics Editor" umożliwia tworzenie rysunków również w 16 kolorach z 8-krotnym rozjaśnieniem. Jako język wykorzystywaną jest firmowa wersja BASIC-u, a system operacyjny CP/M zapewnia dostęp do urozmaiconej biblioteki oprogramowania.

Cena waha się od 40 do 60 tys. franków w zależności od żądanej konfiguracji. Stosowany jest w studiach filmowych, w salach konferencyjnych, wydawnictwach, itp.

Micro Systems nr 45/84

Sprzedaż superkomputerów amerykańskich do Japonii

Mimo wysiłków firm japońskich, aby przodować w dziedzinie superkomputerów, użytkownicy tego kraju dokonują nadal zakupów dużych maszyn w USA. W 1980 r. dwa superkomputery pierwszej generacji sprzedane były przez Cray Research Inc. uniwersytetom japońskim. Od tej pory nie słychać było o tego typu transakcjach. Jednakże obecnie wiele japońskich instytucji badawczych osiągnęło taki poziom, że potrzebują one dużej mocy obliczeniowej do prac w dziedzinie złożonych układów półprzewodnikowych, przetwarzania chemicznego, widzących robotów i innych inteligentnych systemów cyfrowych. W rezultacie znów firma Cray uzyskała licencję eksportową na swój największy superkomputer X-MP o wartości 12 mln dolarów, który został w sierpniu 1984 roku zainstalowany w Nippon Telegraph end Telephone Public Corp. Druga taka sama maszyna ma być dostarczona innemu przedsiębiorstwu japońskiemu na początku 1985 r.

Inna firma amerykańska Denelcor Inc. z Aurory w stanie Colorado eksportuje swój system o równoległym przetwarzaniu HEP w cenie 2 mln. dol, do dużej japońskiej firmy chemicznej Showa Denko K.K., która ma go zastosować do modelowania molekularnego. Aby uatrakcyjnić ten system firma wyposaża go w nowy system operacyjny na UNIX-ie. Również inne firmy amerykańskie, jak Control Data Corp. i ETA Systems Inc. mają nadzieję na udział w rynku japońskim.

Mimo tego wzrostu sprzedaży zakupy superkomputerów w Japonii pozostają niewysokie w porównaniu z resztą świata. W 1983 r. sprzedano 26 superkomputerów za 250 mln dolarów, z czego 15 w USA, a 9 w Europie. Obroty w tym roku powinny wzrosnąć o 20-40% w zależności od realizacji zawartych umów.

Electronics Week nr 24/84

Dane katalogowe na bieżąco

Znana z gier telewizyjnych i urządzeń do przetwarzania tekstu firma Videotekst przewiduje prowadzenie aktualnej listy elementów elektronicznych. Zakodowane zostały istniejące katalogi DATA, a w najbliższej przyszłości będą zakodowane katalogi czołowych wytwórców elementów. Użytkownik będzie mógł je uzyskać za pośrednictwem sieci usługowej i wyświetlać na komputerze osobistym IBM lub na specjalnych końcówkach Videotekst. Nowe informacje będą wprowadzane do katalogów w 24 godziny po ich uzyskaniu, co jest okresem bardzo krótkim w porównaniu z przygotowaniem i rozprowadzaniem materiałów drukowanych.

EDN nr 16/84

Nowa technika dyskowa

Firma 3M wprowadza nowe rozwiązanie w dziedzinie technologii dysków magnetycznych. Polega ona na nałożeniu na twardy dysk z plastiku elastycznej warstwy magnetycznej. Zbudowano już prototypowe dyski o średnicy 5 1/4 cala i pojemności 5 Mbajtów na jednej stronie. Firma przewiduje produkcję wymienionych dysków o pojemności 37 Mbajtów opartych na tym rozwiązaniu, przy czym nie byłyby tu stosowane ani zapis pionowy, ani ośrodki cienkowarstwowe. Częstotliwość występowania błędów w dyskach prototypowych była tego samego rzędu, co w zwykłych dyskach, a odporność na wstrząsy dwukrotnie większa. Przy stosowaniu takich dysków wymagane są niewielkie zmiany w standardowych stacjach typu Winchester.

EDN nr 16/84

Kostki pamięciowe IBM o pojemności 256 Kbitów

Ośrodek opracowania i wytwarzania elementów półprzewodnikowych IBM w Essex Junction rozpoczął wytwarzanie na dużą skalę kostek pamięci operacyjnej o pojemności 256 Kbitów w technologii n-MOS. Pozwala to przechowywać ponad 4 miliony znaków informacji na pojedynczym pakiecie o wymiarach 7x9 cali, co jest rekordem w tej dziedzinie. Kostka, o rozstępie ścieżek 1,5 um, wytwarzana jest konwencjonalną techniką optycznej litografii. IBM produkuje te kostki tylko dla własnych potrzeb i nie ujawnia wielkości produkcji. W USA tylko ITT wytwarza takie kostki od przeszło roku w swej wytwórni elementów w Alletown /nowe wytwórnie w Kansas City i Orlando rozpoczną produkcję na początku 1985 r./. Firma ta oprócz produkcji na własny użytek również sprzedaje te kostki, ale nie podaje liczby odbiorców i rozmiarów produkcji. Zakłady Motorola Inc. w Schammberg

i Mostek Corp. w Carollton wysyłają próbki kostek 256 k swym klientom, a Texas Instruments przygotowuje się do [illegible]oszenia sprzedaży.

Jeśli chodzi o firmy japońskie to NEC Corp. rozpoczęła ograniczoną sprzedaż kostek 256 k w maju 1983 r., a duże ilości /partie po 10 tys. szt./ w końcu 1983 r. W lipsu 1984 r. wysyłka osiągnęła milion sztuk miesięcznie i oczekiwane jest podwojenie tempa sprzedaży do końca roku. Prawie takie samo tempo osiąga Hitachi Ltd., a Fujitsu Ltd. niewiele pozostaje w tyle. Natomiast każde z przedsiębiorstw Toshiba Corp. i Mitsubishi Electric Corp. wysyłają po ok. 50 tys. kostek miesięcznie.

Electronics Week nr 24/84

Próby łączenia różnych zastosowań w standardowe pakiety programowe

Wydaje się, że przedsiębiorstwa zajmujące się oprogramowaniem wkraczają w nową fazę rozwoju. Koncentrują się one na tworzeniu programów łatwiejszych w użyciu i łączących różne zastosowania, takie jak przetwarzanie tekstów i gospodarka finansowa w jednym pakiecie. Stąd wiele nowych programów określanych jako "scalone", aczkolwiek nie ma ścisłej definicji tego określenia.

Istnieje wiele stopni scalania i liczne sposoby ich osiągania. Celem ich jest umożliwienie użytkownikom wykonywania wielu zadań jednocześnie. Użytkownik może np. poszukiwać danych w elektronicznym systemie zbiorów, wstawiać je do prognoz budżetowych, przygotowywać wyciągi z wyników, które będzie następnie wstawiał do listu, pisanego za pomocą programu przetwarzania tekstów. Dawniej użytkownicy musieli kopiować potrzebne dane z jednego programu, wkładać do komputera dysk zawierający

następny program i ponownie wprowadzać te dane, czasami nawet z klawiatury. Scalanie, przynajmniej teoretycznie, ułatwia przesyłanie danych, a ponadto powinno dać użytkownikowi wspólną listę rozkazów dla różnych programów.

Rozróżniamy tu dwa główne podejścia. Pierwsze łączy wiele zadań w pojedynczy program. Drugie zapewnia sposób łączenia programów dostarczanych przez różnych klientów. Przykładem pierwszego podejścia jest program firmy Lotus Development o nazwie "1-2-3", znany też jako program wielofunkcyjny. Zawiera on tzw. arkusz elektroniczny /spreadsheed/, w którym można manipulować wierszami i kolumnami liczb, program kreślący i zarządzanie bazą danych zawarte w systemie zbiorów. Ponieważ napisany jest on jako zunifikowany pakiet, poszczególne części pracują razem wyjątkowo dobrze. Przeszkodą jest to, że wykorzystanie ograniczone jest do tego, co zawiera pakiet. Nie miał on dotychczas np. przetwarzania tekstów, nie można więc było łatwo jego wyników włączać do pism itp. Ponadto właściwości programu wielofunkcyjnego mogą być ograniczone w porównaniu z programami wyspecjalizowanymi. Gdy inne firmy opracowały programy konkurencyjne, Lotus włącza przetwarzanie tekstów do swego programu.

Rozwiązanie z oddzielnymi programami wykorzystuje podział ekranu monitora na obszary /okienka/, w których pojawiają się te programy. Pozwala to na przesyłanie między nimi danych, przy czym często posługujemy się tu urządzeniem zwanym "myszką", pozwalającym na kierowanie wskaźnikiem po ekranie.

Kiedy w maszynach Star firmy Xerox i Lisa firmy Apple wprowadzono okienka i myszkę zapoczątkowało to konkurencję między firmami Microsoft Corp. i Visicorp. Pierwsza z nich opracowała system Window, który można stosować do istniejących programów i oferuje go jako rozszerzenie swych systemów opera-

cyjnych MS-DOS. Samo wykorzystanie okienek wiele użytkownikowi nie pomaga. Różne programy mają wciąż różne listy rozkazów i mogą nie wykorzystywać możliwości posługiwania się myszką. Microsoft ma nadzieję, że opracowujący program będą je modyfikowali, aby wykorzystać te możliwości.

Podobne rozwiązanie przedstawiła firmy Quartedeck Office Systems z Santa Monica, której program DesQ zapewnia wspólną listę rozkazów dla użytkowników. Kiedy użytkownik wprowadza te rozkazy lub pokazuje je myszką, program DesQ tłumaczy każdy rozkaz na inny, zrozumiały przez poszczególne programy. Tłumaczenie zapewnione jest dla wielu popularnych programów, ale jeśli jest to program nieznany dotychczas DesQ, użytkownik musi sam dokonać translacji.

Visicorp natomiast sprzedaje programy aplikacyjne takie, jak przetwarzanie tekstów i arkusz elektroniczny, dopasowane do pracy z systemem Visi-On. Pozwala to na łatwiejsze scalanie, ale wymaga, aby użytkownik zrezygnował ze swoich programów. IBM ma sprzedawać system Visi-On, natomiast na temat systemu Window nie wypowiedziała się jeszcze. System ten ma poparcie wielu innych firm sprzętowych i programowych.

Trudno jest w pełni ocenić do oprogramowanie scalone, ponieważ wiele firm ogłasza swe wyroby zanim jeszcze one powstaną. Trudności w ich realizacji /mała szybkość i zawodność/ częściowo spowodowane są ograniczonymi możliwościami współczesnych komputerów osobistych.

International Herald Tribune
z 2.XII.83 r.

Opracował:
mgr inż. Jan RYŻKO

Informacja o cenach i warunkach prenumeraty na 1985 r.
- dla czasopism Instytutu Maszyn Matematycznych

● Cena prenumeraty rocznej

Techniki Komputerowe - Biuletyn Informacyjny	1560.-	dwum.
Przegląd Dokumentacyjny - Nauki i Techniki Komputerowe	1260.-	dwum.
Informacja Ekspresowa - Nauki i Techniki Komputerowe	2400.-	mies.
Prace naukowo-badawcze Instytutu Maszyn Matematycznych	660.-	3x w roku

● Warunki prenumeraty

1/ dla osób prawnych - instytucji i zakładów pracy:
- instytucje i zakłady pracy zlokalizowane w miastach wojewódzkich i pozostałych miastach, w których znajdują się siedziby oddziałów RSW "Prasa-Książka-Ruch" zamawiają prenumeratę w tych oddziałach;
- instytucje i zakłady pracy zlokalizowane w miejscowościach, gdzie nie ma oddziałów RSW "Prasa-Książka-Ruch" i na terenach wiejskich opłacają prenumeratę w urzędach pocztowych i u doręczycieli;

2/ dla osób fizycznych - prenumeratorów indywidualnych:
- osoby fizyczne zamieszkałe na wsi i w miejscowościach, gdzie nie ma oddziałów RSW "Prasa-Książka-Ruch" opłacają prenumeratę w urzędach pocztowych i u doręczycieli;
- osoby fizyczne zamieszkałe w miastach - siedzibach oddziałów RSW "Prasa-Książka-Ruch" opłacają prenumeratę wyłącznie w urzędach pocztowych nadawczo-oddawczych właściwych dla miejsca zamieszkania prenumeratora. Wpłaty dokonują używając "blankietu wpłaty" na rachunek bankowy miejscowego oddziału RSW "Prasa-Książka-Ruch";

3/ Prenumeratę ze zleceniem wysyłki za granicę przyjmuje RSW "Prasa-Książka-Ruch", Centrala Kolportażu Prasy i Wydawnictw, ul. Towarowa 28, 00-958 Warszawa, konto NBP XV Oddział w Warszawie nr 1153-201045-139-11. Prenumerata ze zleceniem wysyłki za granicę pocztą zwykłą jest droższa od prenumeraty krajowej o 50% dla zleceniodawców indywidualnych i o 100% dla zlecających instytucji i zakładów pracy.

● Terminy przyjmowania prenumeraty na kraj i za granicę:

- do dnia 10 listopada na I kwartał, I półrocze roku następnego oraz na cały rok następny.
- do dnia 1-każdego miesiąca poprzedzającego okres prenumeraty roku bieżącego.

Zamówienia na prenumeratę "Prac naukowo-badawczych Instytutu Maszyn Matematycznych przyjmuje Dział Sprzedaży Wysyłkowej Ośrodka Rozpowszechniania Wydawnictw Naukowych PAN, Warszawa, Pałac Kultury i Nauki, tel. tel.20-02-11 w.2516. Egzemplarze pojedyncze Prac są do nabycia w księgarni ORWN PAN, Warszawa, Pałac Kultury i Nauki, tel.20-02-11 w.2105.

Cena: 260.-

informacja ekspresowa

NAUKI
I TECHNIKI
KOMPUTEROWE

Instytut Maszyn Matematycznych zawiadamia, że od 1984 r., po dwuletniej przerwie, wznawia wydawanie miesięcznika "Informacja ekspresowa - Nauki i Techniki Komputerowe". W czasopiśmie zamieszczamy opisy bibliograficzne /wraz z krótkimi notatkami objaśniającymi/ dokumentów źródłowych, które znajdują się w bibliotece IMM - najlepiej zaopatrzonej w branży komputerowej.

Dokumentujemy ok. 600 pozycji książkowych rocznie /krajowych, i zagranicznych/ oraz 184 tytuły czasopism /około 2000 zeszytów/ w językach: polskim, angielskim, rosyjskim, niemieckim, czeskim; katalogi i in.

Informacja ekspresowa NiTK informuje o najnowszych publikacjach z zakresu branży komputerowej i dziedzin pokrewnych oraz nauk związanych z branżą /monografie, słowniki, podręczniki, materiały szkoleniowe, artykuły w czasopismach, przyczynki, krótkie notatki o najnowszych zdobyczach techniki komputerowej na świecie itp./ jest więc podstawowym i niezbędnym narzędziem pracy każdego pracownika naukowego, studenta, inżyniera - praktyka, projektanta i in.

Nasi Czytelnicy mogą zamawiać mikrofilmy i kserokopie dokumentów, których opisy znajdują się w Informacji ekspresowej.