Barbara Maj
Jacek Mączyński

SYSTEM RSX-11 — PRELIMINARIA

25/1984

WARSZAWA 1984

ISSN 0208 - 5658

Praca wpłynęła do Redakcji dnia 24 maja 1984 r.

Na prawach rękopisu

Instytut Podstawowych Problemów Techniki PAN

Nakład 200 egz. Ark.wyd. 3, Ark. druk. 4
Oddano do drukarni w czercu 1984 r.
Nr zamówienia 434/84.

Warszawska Drukarnia Naukowa, Warszawa,
ul. Śniadeckich 8

Barbara Maj
Jacek Mączyński
Ośrodek Informatyki
Pracownia Obliczeń Numerycznych
IPPT PAN

SYSTEM RSX-11 - PRELIMINARIA

Spis treści

P R Z E D M O W A

Użytkowanie systemu komputerowego SM-4 dla obliczeń naukowych stwarza potrzebę udostępnienia informacji o sposobie posługiwania się tym systemem. W szczególności użyteczne są tu systemy oprogramowania OC-PB /OS-RW/ oraz RSX-11. W aktualnej wersji opracowania zebrano te elementy systemu, które umożliwiają efektywną pracę pod systemem.

Barbara Maj
Jacek Mączyński

0. System RSX-11 preliminaria

Praca użytkownika każdego systemu operacyjnego składa się najczęściej z następujących etapów.

1. Inicjalizacja pracy pod systemem.
2. Zakładanie i przetwarzanie plików /np. zbiorów liczbowych, programów/.
3. Translacja i konsolidacja programów.
4. Uruchamianie programów.
5. Zakończenie pracy pod systemem.

W systemie RSX wykonanie każdego z wymienionych etapów umożliwia odpowiedni zestaw zleceń i programów. Zlecenie HE[LLO] inicjalizuje pracę pod systemem. Zakładania i przetwarzania zbiorów dokonują programy: EDI, PIP, FLX, DSC.
Program FOR /oraz dla innych języków odpowiednio PAS, COB, MAC, BAS/ umożliwia przeprowadzenie translacji tzn. tłumaczenie z języka wysokiego poziomu /FORTRAN/ na język wewnętrzny maszyny. Następnie program TKB /FTB/ dokonuje konsolidacji. Uruchamianie programów /por. Rozdz.7/ następuje poprzez zlecenie RUN.
Potrzebna w etapach 2,3,4 specyfikacja plików ma pełną postać:

dev:[n,m]nazwa.[typ;wersja]

dev - nazwa urządzenia na którym jest lub ma być dany plik.
Para n,m określa kod użytkownika /por. opis zlecenia HEL/,
nazwa - nadana przez użytkownika nazwa pliku. Nazwa zaczyna się od litery i składa się z co najwyżej dziewięciu liter i cyfr.
typ - trzyliterowy skrót określający zawartość zbioru. Typy zbiorów zostaną omówione poniżej w Rozdz. 5 i 6.
wersja - liczba określająca numer wersji.
Nazwa urządzenia dev ma postać: DVn, gdzie
DV - dwuliterowy skrót nazwy urządzenia tzn.:
DKn - dysk,
DXn - dysk elastyczny /floppy-disk/,
LPn - drukarka wierszowa,
TTn - terminal ekranowy,
CTn - kaseta taśmy magnetycznej,

MT - przewijak taśmy magnetycznej,
PP - perforator taśmy papierowej,
PR - czytnik taśmy papierowej,
GR - grafoskop,
gdzie n - numer dostępnego urządzenia /n=0,1,.../.
Oprócz urządzeń rzeczywistych użytkownik może korzystać z tzw. pseudourządzeń, którym system przyporządkowuje urządzenia rzeczywiste.
CL - urządzenie drukujące /najczęściej konsola operatorska/,
TI - urządzenie wejściowe /najczęściej terminal użytkownika/,
LB - urządzenie domyślne dla systemu /dysk systemowy/,
SY - urządzenie domyślne użytkownika.
Zgodnie ze specyfikacją pliku j.w. wszystkie nazwy urządzeń i pseudourządzeń są zakończone dwukropkiem.
Nie przy każdym odwołaniu się do pliku musimy podawać pełną jego specyfikację. W systemie RSX przyjęte są pewne zasady domyślności.
Jeśli nie podana została nazwa urządzenia to przyjmuje się, że zbiór znajduje się na urządzeniu domyślnym SY:.
Jeśli nie podany jest kod to przyjmuje się, że plik jest własnością użytkownika, który wywołał program korzystający z danego pliku.
Domyślny typ zbioru zależy od programu przetwarzającego dany zbiór, domyślne typy zbiorów zostaną podane w dalszej części opracowania /Tab. 5 i 6/.
Jeśli użytkownik nie podał wersji zbioru to przyjmuje się najwyższy istniejący numer wersji zbiorów o danej nazwie.
Zbiory przechowywane na różnych wolumenach /nośnikach informacji/ często różnią się formatami. Rozróżniamy trzy formaty wolumenów:
RT-11, DOS-11, Files-11
Format Files-11 jest formatem domyślnym.

Dysponujemy następującymi /podstawowymi/ programami przetwarzającymi pliki:
FLX - program umożliwiający przepisywanie plików z jednego

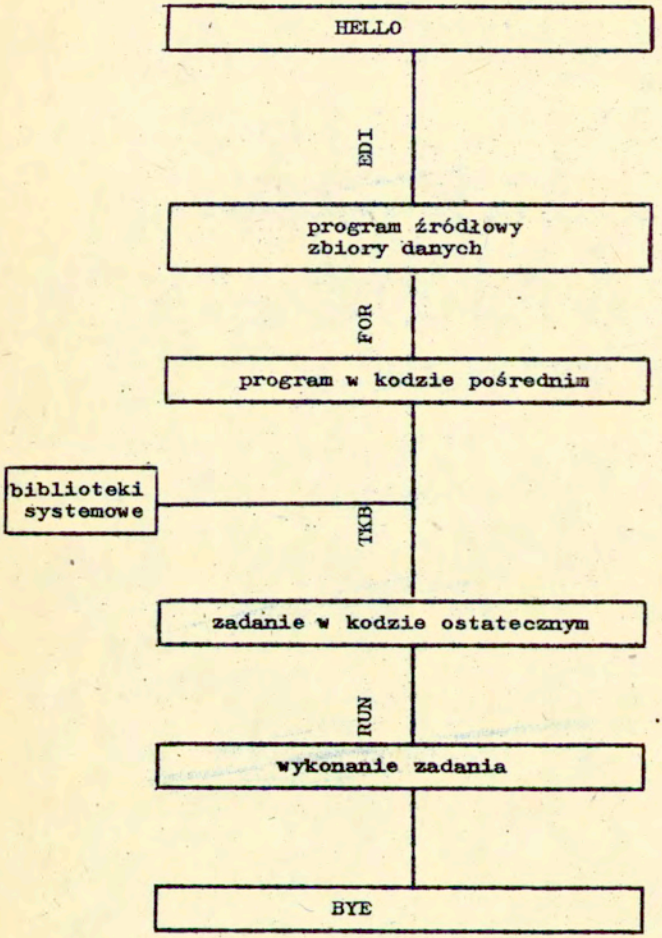
wolumenu na inny. Program FLX umożliwia zmianę formatów zbiorów.

PIP - program umożliwia przetwarzanie jedynie zbiorów typu Files-11.

DSC - program przepisuje zawartość dysków i taśm na dyski i taśmy, dodatkowo przeprowadza kompresję wolnych obszarów.

PRI - umożliwia drukowanie plików przez spooler. Podstawowy program przy pomocy którego możemy wyprowadzać teksty na drukarkę.

Zakończenie współpracy z systemem odbywa się przy pomocy zlecenia BYE.



Rys. 1

Schemat przebiegu uruchamiania zadania /program źródłowy napisany w języku FORTRAN/ pod systemem RSX.

1. Opis klawiatury terminala

Zestaw klawiszy funkcyjnych:
Znaki sterujące wprowadza się naciskając jednocześnie klawisz CTRL /YC/ z odpowiednim klawiszem znaku. Zapis ¬A oznacza, że klawisz A jest naciskany jednocześnie z klawiszem CTRL.
TABELA 0

Klawiatura terminala

znak	funkcja
¬C	Wywołanie MCR - nadrzędnego programu systemu.
¬K	Przesunięcie tekstu na ekranie terminala o 4 linie.
¬L	Przesunięcie tekstu na ekranie terminala o 8 linii.
¬O	Zakończenie wyświetlania tekstu na terminalu.
¬I	Tabulacja.
¬R	Wyświetlanie linii bieżącej bez usuniętych znaków.
¬S	Zawieszenie wyświetlania informacji na terminalu.
¬Q	Wznowienie wyświetlania informacji.
¬U	Usunięcie bieżącej linii tekstu.
¬Z	Zakończenie pracy z zadaniami systemowymi.
¬[	Zakończenie linii wejściowej bez przesunięcia kursora.

Klawisz SHIFT /BP/ naciśnięty łącznie z klawiszem "-" powoduje usunięcie ostatnio napisanego znaku /lub kilku znaków/.
Klawisz TAB /GT lub ¬I/ umożliwia ustawienie tabulatora co 8 znaków.

2. Zlecenie HELLO /HEL/, zgłoszenie wejścia do systemu

Operator systemu RSX-11 przydziela każdemu użytkownikowi /który chce rozpocząć po raz pierwszy pracę pod systemem/ kod i hasło. Kod składa się z pary liczb, z których pierwsza określa numer grupy użytkowników, a druga numer użytkownika w grupie. Hasło to łańcuch co najwyżej 6 znaków alfanumerycznych. Przed każdym rozpoczęciem pracy pod systemem RSX użytkownik musi podać /systemowi/ swój kod /ew. nazwisko/ i hasło. Po sprawdzeniu /przez system/ poprawności kodu i hasła użytkownik może przystąpić do właściwej pracy tzn. pisania programów, zakładania zbiorów itd.. Zgłoszenie do systemu odbywa się przy pomocy zlecenia HEL.

Jednoliniowy format zlecenia HEL

>HEL "UIC" lub nazwisko/hasło

"UIC" - kod użytkownika /USER IDENTIFICATION CODE/ postaci:

nnn,mmm

gdzie: nnn, mmm liczby oktalne.

Uwaga: W przykładach napisy wyprowadzane przez komputer będą podkreślane, a napisy wyprowadzane przez użytkownika, które nie są słowami kluczowymi będą nadkreślane. Trzy kropki opisują możliwość powtórzenia konstrukcji. Nawias graniasty oznacza opcjonalność elementu, jeśli nie stanowi wyraźnie części napisu.

Wieloliniowy format komendy HEL:

>HEL
ACCOUNT OR NAME "UIC" lub nazwisko
PASSWORD hasło

Hasło nie jest wyświetlane na terminalu.

Przykłady:

>HEL 210,7 /ANDY

>HEL
ACCOUNT OR NAME 210,7
PASSWORD ANDY

>HEL KOWALIK/ANDY

```
>HEL KOWALIK
    PASSWORD ANDY

>HEL
  ACCOUNT OR NAME KOWALIK
  PASSWORD ANDY
```

Powyższe przykłady ilustrują w jaki sposób użytkownik o nazwisku KOWALIK i posiadający hasło ANDY oraz "UIC" = 210,7 może zgłosić się do systemu.

3. Zlecenie BYE, zakończenie pracy pod systemem RSX

Zlecenie BYE sygnalizuje, że użytkownik zakończył współpracę z systemem RSX. W wyniku wywołania tego zlecenia, system zawiesza wykonanie wszystkich zadań aktywnych użytkownika i zamyka otwarte zbiory, odłącza przydzielone urządzenia. Postać zlecenia BYE:

> BYE

4. Program EDI /edytor/

Zadaniem programu EDI jest utworzenie nowego zbioru od początku lub przez modyfikację zbioru już istniejącego. Na ogół program EDI pracuje w trybie blokowym, pobiera ze zbioru co najwyżej 38 linii tekstu /więcej, jeżeli w pamięci zarezerwowany jest większy obszar dla bufora/. W obrębie bloku można się poruszać stosując dyrektywy B,T,BO,NEXT /por.tab. 1,2/ oraz uzyskać dostęp do następnego bloku dyrektywą RENEW, czy też READ. EDI dopuszcza też operowanie stronami /strona to tekst między znakami przesuwu papieru FF, a także między znakiem FF i znakiem zakończenia zbioru EOF/ przez zastosowanie dyrektywy PAGE. Istnieje również tryb dostępu po jednej linii, bez możliwości cofania się, włączany dyrektywą BLOK OFF /wyłączenie trybu blokowego/ i usuwany przez BLOCK ON.

Postać wywołania programu EDI:

≥EDI [dev:]NAZWA [.typ] [;wersja]

dev - nazwa urządzenia,

NAZWA - nazwa zbioru,

typ, wersja - typ i numer wersji zbioru.

PRZYKŁAD:

≥EDI NEWTON.FTN;7

Spacja, średnik kropka są znakami początku odpowiednich elementów napisu.

Jeśli numer wersji nie jest podany, to edytor modyfikuje zbiór o najwyższym numerze wersji lub tworzy nowy zbiór, jeśli zbiór o podanej nazwie nie występuje.

Jeżeli edytor tworzy nowy zbiór od początku to na terminalu ukazuje się następująca sygnalizacja:

CREATING NEW FILE

INPUT

Następnie użytkownik może rozpocząć wprowadzanie tekstu.

Jeśli odwołano się do istniejącego zbioru to drukowany jest tekst:

00036 LINES READ IN

PAGEØ

Następnie można wprowadzać dyrektywy dla edytora. Można przechodzić ze stanu edycji w stan wprowadzania korzystając z dyrektywy INSERT. Jeżeli w nowoutworzonym zbiorze należy dokonać poprawek, to jako pierwszy symbol nowej linii należy wprowadzić karetkę. Redaktor tekstu wyprowadza wtedy na terminalu znak * , który oznacza, że program EDI znajduje się w stanie edycji. W opisach dyrektyw programu EDI linię, do której program EDI ma dostęp w danej chwili, będziemy nazywać linią bieżącą.

TABELA 1

Podstawowe dyrektywy programu EDI

Dyrektywa	Format	Opis
ADD	ADD łańcuch	Dołączenie łańcucha do linii bieżącej.
ADD&PRINT	AP łańcuch	Dołączenie łańcucha do linii bieżącej i wyświetlenie linii.
BOTTOM	BO	Przesunięcie wskaźnika linii do końca linii lub bufora.
CHANGE	[n]C/ła1/ła2	Zamiana łańcucha ła1 na ła2 n razy w linii bieżącej. Zamiast znaku "/" może wystąpić dowolny znak, różny od litery i cyfry, który nie występuje w łańcuchach ła1, ła2.
CRTL/Z		Dokonuje zamknięcia zbioru i zakończenia edycji.
DELETE	D[n] lub D[-n]	Skasowanie linii bieżącej i n-1 następnych lub jeśli n<0 linii bieżącej i n-1 poprzednich /w reżimie blokowym/.
DELETE & PRINT	DP[n] lub DP[-n]	Dyrektywa działa analogicznie jak DELETE, dodatkowo wyświetlane są kasowane linie.
ESC	klawisz ESC	Powoduje cofnięcie wskaźnika linii o jedno miejsce i wyświetlenie

Dyrektywa	Format	Opis
		linii bieżącej /działa tylko w reżimie blokowym/.
EXIT	EX	Zamknięcie zbioru i zakończenie edycji.
INSERT	I tekst⌡ lub I⌡tekst	Umożliwia przejście ze stanu edycji w stan wprowadzania tekstu. Przy formacie wieloliniowym, aby zakończyć wprowadzanie należy dwukrotnie nacisnąć klawisz BK.
LOCATE	[n]L łań	Lokalizuje n-te pojawienie się łańcucha łań.
NEXT	N[n] lub N[-n]	Przesunięcie wskaźnika linii o n pozycji w przód lub, gdy n < 0 w tył /działa tylko w reżimie blokowym/.
NEXT PRINT	NP[n] lub NP[-n]	Działa analogicznie jak dyrektywa NEXT, dodatkowo wyświetlana jest bieżąca linia.
RENEW	REN [n]	Przejście do następnego lub n-tego bloku zbioru.
RETYPE	R łańcuch	Zamienia linię bieżącą na łańcuch lub jeśli łańcuch nie występuje usuwa linię bieżącą.
TOP	T	Przesuwa wskaźnik linii na początek bloku /w reżimie blokowym/ lub zbioru /w reżimie liniowym/.
TOP OF FILE	TOF	Powrót do początku zbioru przy zachowaniu poprzednio zredagowanych stron.

TABELA 2

Dyrektywy ustawiające wskaźnik linii

Dyrektywa	Format	Opis
BLOCK ON	B ON B OF	Przełączniki reżimu dostępu do zbioru B ON - dostęp blokowy, B OF dostęp liniowy.
OPEN SECONDARY	OP spec.	Otworzenie innego wyspecyfikowanego zbioru.
OUTPUT ON	OU ON	Kontynuacja lub przerwanie
OUTPUT OFF	OU OFF	transferu do zbioru wyjściowego.
SELECT PRIMARY	SP	Ustawienie pierwszego zbioru jako zbioru wejściowego.
SELECT SECONDARY	SS	Ustawienie drugiego zbioru jako wejściowego.
TAB	TAB ON TAB OFF	Włączenie tabulatora, wyłączenie tabulatora.
BEGIN	B	Bieżącą linią staje się linia poprzedzająca pierwszą linię zbioru /reżim liniowy/ lub bufora /reżim blokowy/, w reżimie liniowym tworzona jest nowa wersja zbioru.
BOTTOM	BO	Bieżącą linią staje się ostatnia linia zbioru /reżim liniowy/ lub bufora /reżim blokowy/.
FIND	[n]F łańcuch	Znaleźć w bieżącym bloku lub zbiorze począwszy od linii bieżącej n-te pojawienie się łańcucha, który zaczyna się od pierwszej kolumny. Wskaźnik linii przesuwa się na tę linię.
NEXT NEXT & PRINT		Klucze opisane w tabeli 1.
PAGE	PAG n	Wczytanie n-tej strony do bufora.

Dyrektywa	Format	Opis
RETURN	CR	Przejście do następnej linii, staje się ona linią bieżącą.
SEARCH & CHANGE	SC/ła1/ła2	Lokalizuje łańcuch ła1 i zamienia na ła2, zamiast znaku "/" można użyć innego znaku, który nie występuje w łańcuchach ła1 i ła2.

TABELA 3

Dyrektywy modyfikacji tekstu

Dyrektywa	Format	Opis
ADD		Opis wymienionych dyrektyw znajduje się w tabeli 1.
ADD&PRINT		
CHANGE		
DELETE		
DELETE&PRINT		
ERASE	ERASE[n]	W reżimie blokowym kasuje blok bieżący i n-1 następnych w reżimie liniowym linię bieżącą i n-1 następnych.
LINE CHANGE	[n]LC ła1/ła2	Zamienia łańcuch ła1 na ła2 w linii bieżącej i n-1 następnych.
OVERLAY	O[VERLAY]n	Kasuje n linii, przejście do reżimu wejściowego. Linie tekstu napisane po tym rozkazie zostaną umieszczone w miejscu skasowanych linii.
PASTE	PASTE/ła1/ła2	Przeszukuje pozostałe linie bloku lub zbioru /zależnie od reżimu/ i zamienia wszędzie łańcuch ła1 na ła2.

Dyrektywa	Format	Opis
UNSAVE	UNS [spec.zb]	Umieszcza wszystkie linie wyspecyfikowanego zbioru za linię bieżącą. Jeśli brak specyfikacji zbioru to przyjmuje się, że zbiór ma nazwę SAVE.TMP.

TABELA 4

Dyrektywy WE/WY i zamykania zbiorów

Dyrektywa	Format	Opis
FILE	F spc.zb	Dokonuje transferu linii ze zbioru wyjściowego do wyspecyfikowanego zbioru wejściowego, aż do napotkania znaku końca strony /reżim blokowy/ lub końca zbioru /reżim liniowy/.
READ	REA n	Wczytuje n kolejnych stron tekstu.
SAVE	SA n [spec. zb.]	Umieszczenie linii bieżącej i n-1 następnych w wyspecyfikowanym zbiorze. Jeśli brak specyfikacji to linie przechowywane są w zbiorze o nazwie SAVE.TMP. Dyrektywy SAVE i UNSAVE używane są głównie w celu przepisywania powtarzających się fragmentów programu.
WRITE	W	Przepisuje zawartość bufora do zbioru wyjściowego i oczyszcza bufor.
CLOSE	CL	Dokonuje transferu pozostałych linii bufora do zbioru wejściowego i zamknięcia zbiorów. Nie

Dyrektywa	Format	Opis
		następuje wyjście z programu EDI.
CLOSE &	CD	Dyrektywa działa analogicznie jak
PRINT		CLOSE dodatkowo kasowany jest zbiór wejściowy.
CTRL/Z		Działa jak dyrektywa CLOSE.
EXIT	EX SPEC	Zamyka zbiór, kończy edycję i zamienia nazwę zbioru wyjściowego, /jeśli występuje SPEC - nowa nazwa zbioru/.
EXIT &		Działa podobnie jak dyrektywa EXIT dodatkowo kasowany jest
DELETE	ED	zbiór wejściowy.

Jeśli użytkownik korzystający z edytora nie podaje specyfikacji zbioru wyjściowego, to zbiór ten otrzymuje nazwę zbioru wejściowego. Numer wersji zostaje zwiększony o jeden.

PRZYKŁADY:

1. Mamy tekst:

REAL A , SUMA
N = N+1

użycie ciągu dyrektyw:

* C/A/K, ⌐ ⌐
* D ⌐

powoduje zmianę tekstu na następujący:

REAL K , SUMA

2. Jeśli bieżąca linia tekstu ma postać:

N = N+1

to użycie dyrektywy:

* 2C/N/M1

powoduje zamianę linii na następującą:

M1=M1+1

3. Mamy następujący fragment programu:

```
      INTEGER I,J,K

      CALL ASSIGN(1,'ZBIOR')
```

jeśli strzałka EDI znajduje się na początku powyższego fragmentu to w wyniku dyrektyw:

```
*AP, N/
*I/
      REAL A , SUMA
```

otrzymujemy tekst:

```
      INTEGER I,J,K,N
      REAL A,SUMA
      CALL ASSIGN ( 1,'ZBIOR')
```

5. Translacja programu w języku FORTRAN

Zlecenie wykonania tłumaczenia przez translator FORTRANu ma postać:

> FOR OBJECT,LIST /klucze=FILE1[.FTN][;n] /klucze gdzie:

OBJECT - nadana przez użytkownika nazwa przetłumaczonego programu,

LIST - nazwa urządzenia lub zbioru w którym przechowany zostanie listing /tabulogram/ translacji,

FILE1 - nazwa zbioru wejściowego,

n - numer wersji zbioru, jeśli parametr n nie występuje to przeprowadzana jest translacja zbioru o podanej nazwie i najwyższym numerze generacji.

/klucze - wykaz kluczy o postaci /"klucz" bez innych separatorów. Podstawowe klucze zlecenia FOR mają postać:

/LI:0 lub /NOLI Drukowanie diagnostyki błędów.

/LI:1 lub /LI:SRC Drukowanie diagnostyki błędów i tekstu programu źródłowego.

/LI:2 lub /LI:MAP Drukowanie mapy pamięci.

/LI:4 lub /LI:COD Drukowany jest tabulogram wygenerowanych kodów i diagnostyka błędów.

Jeśli użytkownik chce uzyskać informację o błędach, tekst programu źródłowego i mapę pamięci, to powinien użyć klucza /LI:3 jest on kombinacją kluczy 1 i 2 .

/-SP Zawieszenie drukowania tabulogramu na urządzeniu fizycznym. Klucz ten należy zawsze stosować jeśli tabulogram nie jest konieczny,

/ID na terminalu użytkownika wyświetlana jest nazwa i numer wersji translatora,

/EX znaki znajdujące się od 72 do 80 kolumny będą czytane przez translator,

/DE będą tłumaczone wiersze programu w których w pierwszej kolumnie występuje litera D. Jeśli klucz /DE nie występuje, to translator traktuje te linie jak komentarz. Wiersze z literą D mogą zawierać pomocnicze wydruki ułatwiające wstępne uruchomienie programu.

PRZYKŁAD:

> FOR XXX,LIST/-SP=TASK.FTN

XXX - nazwa programu w języku wewnętrznym,

LIST - nazwa zbioru w którym przechowywany jest tabulogram wraz z diagnostyką błędów.

TASK.FTN - nazwa zadania źródłowego.

W wyniku translacji w zbiorze o nazwie XXX przechowany zostanie przetłumaczony program /nie nadaje się on do uruchomienia bez konsolidacji/. W zbiorze o nazwie LIST przechowany będzie tabulogram i informacja o błędach. Informacja ta nie będzie drukowana na drukarce /klucz -SP/.

TABELA 5

Domyślne typy zbiorów

zbiór	typ domyślny
tekst przetłumaczony	OBJ /ang. OBJECT/
tabulogram	LST /ang. LISTING/
program źródłowy	FTN /ang.FORTRAN/
zbiór zleceń	CMD /ang. COMMAND/

PRZYKŁADY:

> FOR PULL=PULL.FTN

> FOR PULL,LIST/LI:0=PULL.FTN

> FOR PULL,LIST/-SP=PULL.FTN

Wywołanie translatora języka makroasembler ma analogiczną postać jak translatora FORTRANu. W miejsce nazwy FTN należy napisać nazwę MAC /translator makroasemblera/.

Program napisany w języku makroasembler ma domyślny typ MAC.

6. Konsolidacja programu

Programu w języku pośrednim /po translacji/ nie można jeszcze uruchamiać, należy dokonać konsolidacji tzn. dołączyć dodatkowe moduły, z których korzysta program, zarezerwować niezbędny obszar pamięci konieczny do przechowywania zmiennych itd.. Wymienione wyżej czynności wykonuje konsolidator - program TKB /ang. TASK BUILDER/.

Jednoliniowy format zlecenia uruchamiającego konsolidator:

>TKB TASK,MAPA=OBJECT1/klucz1,...,OBJECTn/kluczn

TASK - nadana przez użytkownika nazwa zadania /postać nadająca się do uruchomienia/,

MAPA - specyfikacja pliku na którym znajduje się mapa konsolidacji,

OBJECT1,...,OBJECTn - nazwy modułów wejściowych,

klucz1,...,kluczn - klucze określające zawartość modułów wejściowych.

Podstawowe klucze programu TKB mają postać:

/LB klucz określa, że zbiór wejściowy jest biblioteką,

/MP klucz określa, że w zbiorze wejściowym znajduje się opis struktury nakładkowej.

Wieloliniowy format wywołania konsolidatora:

```
>TKB
TKB>TASK,MAPA=
TKB>OBJECT1/klucz1
 .
 .
TKB>OBJECTn/kluczn
TKB>/
```

ENTER OPTION

tu następują wprowadzane przez użytkownika opcje TKB

TKB> //

Dwa znaki "slash" uruchamiają konsolidację.

Podstawowe opcje programu TKB:

UNITS deklaruje liczbę urządzeń WE/WY w programie. Domyślnie przyjmuje się UNITS=6.

ACTFIL określa liczbę zbiorów, które mogą być otwarte jednocze-

śnie w programie użytkownika. Domyślnie ACTFIL=4.

ASG opcja ASG ma postać:

$dev1:un1_1:\ldots:un1_k,\ldots\ldots\ldots,devn:unn_1:\ldots:unn_t$

$dev1,\ldots,devn$ - nazwy urządzeń i pseudourządzeń,

$un1_1,\ldots,un1_k,\ldots,unn_t$ - logiczne numery urządzeń i pseudourządzeń,

Numer logiczny jest liczbą całkowitą i dodatnią.

PRZYKŁAD:

```
>TKB
TKB> PROG=PROG1
TKB> /
ENTER OPTION
TKB> ASG=TI:5
TKB> //
```

W wyniku działania programu TKB następuje konsolidacja programu o nazwie PROG1, program w ostatecznej postaci przechowany zostanie w zbiorze o nazwie PROG.Terminal użytkownika TI: wewnątrz programu powiązany będzie z numerem logicznym 5.

TABELA 6

Domyślne nazwy zbiorów w programie TKB

plik	typ domyślny
tekst ostateczny	TSK /ang. TASK/
mapa konsolidacji	MAP /ang. MAP/
tekst pośredni	OBJ /ang.OBJECT/
biblioteka	OLB /ang.LIBRARY/
nakładki	ODL /ang.OVERLAY/
zbiór zleceń	CMD /ang. COMMAND/
tablica symboli	STB /ang.TASK BUILDER SYMBOLS/

PRZYKŁADY:

1. >TKB

```
TKB> PROG=PROG
TKB> LB:[1,1]GLIB/LB
TKB> /
     ENTER OPTION
TKB> ASG=GRØ:1                GRØ - grafoskop
TKB> ASG=KLØ:1                KLØ - klawiatura grafo-
                                    skopu
TKB> COMMON=DFILE:RW:6
TKB>//
```

Konsolidacja programu, który korzysta z systemu MGRAF, znajdującego się w odpowiedniej bibliotece /program korzysta z grafoskopu/. Pojedyncza kreska ukośna /slash/ oddziela wiersze zlecenia od wierszy opcji.

2. Typowe zlecenie konsolidacji ma postać:

```
>TKB PROG,PROG=PROG,LB:[1,1]FOROTS/LB
```

Konsolidacja programu, który korzysta z biLlioteki FORTRANu FOROTS i gromadzi /pod różnymi typami TSK i MAP/ pod wspólną nazwą program wynikowy i mapę konsolidacji.

3.
```
>TKB
TKB> PROG=PROG
TKB>/
     ENTER OPTION
TKB> UNITS=7
TKB> ASG=TI:2:3
TKB> ASG=DK2:4
TKB> //
```

Konsolidacja programu o nazwie PROG, program nie korzysta z bibliotek. Wewnątrz programu terminal użytkownika będzie miał numer logiczny 2 i 3, a dysk DK2: numer logiczny 4.

7. Zlecenie RUN

Zlecenie RUN inicjalizuje i uruchamia zadanie użytkownika.
Format wywołania zlecenia:

>RUN TASK

TASK - nazwa zadania.

PRZYKŁAD:

>RUN PROG

Uruchomienie zadania o nazwie PROG.
Niektóre wersje systemu RSX wymagają, aby przed wywołaniem zlecenia RUN zadanie zostało zainstalowane w systemie.
Instalacji zadania dokonuje zlecenie INS.
Postać wywołania zlecenia:

>INS TASK

TASK - nazwa zadania.

8. Program PIP /Peripheral Interchange Program/

Program PIP działa jedynie na zbiorach Files-11 i umożliwia:

- kopiowanie i usuwanie zborów /klucze DE,SD,PU/,
- zmianę nazwy zbiorów /klucze NV, RE/,
- tworzenie sekwencji połączonych /konkatenacja/ zbiorów /klucz AP/,
- wgląd do katalogów i ich drukowanie /klucz LI/.

Format wywołania programu PIP:

≥ PIP $\overline{OUF1}$/sw1,...,$\overline{OUFn}$/swn=$\overline{INF1}$/sw11,...,$\overline{INFm}$/swm1

$\overline{OUF1}$,...,$\overline{OUFn}$ - specyfikacja zbiorów wyjściowych,

$\overline{INF1}$,...,$\overline{INFm}$ - specyfikacja zbiorów wejściowych,

sw1,...,swn,sw11,...,swm1 - klucze programu PIP,

Specyfikacja zbiorów ma postać:

dev:[n,m]nazwa.typ;wersja

dev - nazwa urządzenia na którym przechowywany jest zbiór,

n,m - kod właściciela zbioru,

nazwa - nazwa zbioru,

typ - typ zbioru,

/Dokładniejsze informacje o specyfikacji zbioru znajdują się w Rozdz. 0/.

wersja - numer wersji zbioru. Jeśli numer wersji =-1,to użytkownik będzie miał dostęp do najstarszej wersji zbioru o danej nazwie. Jeśli numer wersji=0,to użytkownik będzie korzystał z najnowszej wersji zbioru.

Program PIP posiada kilkanaście kluczy i podkluczy. Podklucz może występować łącznie z odpowiadającym jemu kluczem. Specyfikacja klucza składa się ze znaku "slash" i dwu lub trzyliterowej nazwy klucza, po specyfikacji klucza często występują nazwy podkluczy oddzielone od siebie znakiem "slash", podklucze mogą mieć argumenty, oddzielone od nazwy podklucza znakiem ":"/dwukropek/.

TABELA 6

Podstawowe klucze i podklucze programu PIP

Klucz	Podklucz	Opis
/BL:n		Określa ilość ciągłych bloków n zarezerwowanych na zbiór wyjściowy, n może być wartością oktalną lub dziesiętną /zakończoną kropką/.
/CO		Żądanie, aby zbiór wyjściowy był ciągły.
/FO		Określa, że właścicielem zbioru wyjściowego jest właściciel zbioru wejściowego. Jeśli klucz ten nie występuje, to zbiór wyjściowy ma "UIC" pod którym pracuje PIP.
/AP		Dodaje zbiór do końca zbioru istniejącego.
/BS:n		Określa rozmiar /ilość bloków/ zarezerwowanych na zbiór wyjściowy przechowywany na taśmie magnetycznej.
/CD		Zbiór wyjściowy ma tą samą datę utworzenia, co zbiór wejściowy.
/DE		Kasowanie zbioru /bezwarunkowe/.
/LI		Drukowanie katalogu zbiorów użytkownika.
/RE		Zmiana nazwy zbioru.
/SD		Kasowanie zbioru pod kontrolą konwersacyjną.
/TB		Drukowanie tabulogramu całkowitego, ilości bloków zarezerwowanych na katalog, ilości zbiorów, ilości bloków przydzielonych,
/PR		zmiana maski dostępu:
	/GR:RWED	dla właściciela grupowego,

Klucz	Podklucz	Opis
	/OW	dla użytkownika,
	/SY	dla systemu,
	/WO	ogólnie.
/UF		Tworzy UFD /katalog zbiorów/ na taśmie, na którą zbiór jest przepisywany.
/NV		Zwiększa o jeden numer wersji zbioru, klucz ten nie jest potrzebny, jeśli zbiór wejściowy i wyjściowy mają ten sam "UIC".
/PU		Kasuje zbiory, zostawia ostatnią wersję.

PRZYKŁADY:

>PIP TI:=TEST.MAC;1

Na terminalu wydrukowany zostanie tekst zbioru o nazwie TEST.MAC i numerze wersji 1.

>PIP TESTFILE.DAT;1=TEST.DAT;5/RE

Zbiór o nazwie TESTFILE.DAT;1 przyjmuje nazwę TEST.DAT;5.

Jeśli w specyfikacji zbioru zamiast numeru wersji wystąpi:

- * to oznacza wszystkie numery wersji zbioru,
- -1 określa najstarszą wersję zbioru,
- 0 określa najnowszą wersję zbioru.

>PIP DK2:/LI

Drukowanie katalogu zbiorów /użytkownika wywołującego program PIP/ znajdujących się na dysku DK2:.

>PIP ZBIOR.FTN;1/DE

Bezwarunkowe kasowanie zbioru ZBIOR.FTN;1 znajdującego się na urządzeniu SY:.

>PIP TEST.DAT;* /DE

Kasowanie wszystkich zbiorów /użytkownika wywołującego PIP/ znajdujących się na SY: i posiadających nazwę

TEST.DAT.

>PIP ZBIOR1.FTN;1,ZBIOR2.FTN;2/AP

Dopisanie zbioru ZBIOR2.FTN;2 na końcu zbioru ZBIOR1.FTN;1.

>PIP TI:=TEST.DAT;5

Wyświetlenie zbioru TEST.DAT;5 na terminalu użytkownika.

>PIP [200,20]*.*;*/FO

Zlecenie powyższe powoduje, że właścicielem wszystkich zbiorów użytkownika wywołującego program PIP będzie użytkownik posiadający kod identyfikacji 200,20 .

Znak * może występować także w miejscu określającym typ zbioru/ oznacza wtedy wszystkie zbiory o podanej nazwie i numerze wersji i dowolnym typie/, a także w miejscu określającym nazwę zbioru /określa zbiory użytkownika o dowolnej nazwie/.

>PIP *.*;*/DE

Zlecenie powyższe powoduje, że zostaną skasowane wszystkie zbiory użytkownika wywołującego program PIP.

9. Program FLX

Program FLX /File Transfer Program/ umożliwia przepisywanie plików z jednego wolumenu na inny, jeżeli wolumeny mają różne formaty, to program FLX zmienia dodatkowo format plików /por. System RSX - preliminaria/.

Podstawowe funkcje programu FLX:

- tabulogram zbiorów RT-11 i DOS-11 /klucz DI/,
- kasowanie zbiorów RT-11 i DOS-11 /klucz DE/
- zmiana formatów zbiorów /klucze DO,RS,RT,FA,FB,IM/.

Wywołanie programu FLX ma postać:

> FLX dev/sw=$\overline{INF1}$/sw1,...,$\overline{INFn}$/swn

dev - nazwa urządzenia wyjściowego,

$\overline{INF1}$,...,$\overline{INFn}$ - specyfikacja plików wejściowych,

sw,sw1,...,swn - klucze wg TAB.8.

W programie FLX zbiór wyjściowy przyjmuje nazwę zbioru wejściowego.

Klucze określające format zbioru:

/DO-format DOS-11, /RS - format RT-11, /RS - format Files-11.

TABELA 8

Podstawowe klucze transferu zbiorów

Klucz	Opis
/FB:n	Określa, że zbiór ma format binarny. Zbiór wyjściowy DOS-11 lub RT-11 będzie formatowany binarnie. Suma kontrolna i binarne nagłówki są dodawane na końcu zbioru wyjściowego. n - określa długość rekordu w słowach.
/FA:n	Zbiór wyjściowy DOS 11 lub RT 11 będzie formatowany w kodzie ASCII. Dane umieszczone są w rekordach zakończonych karetką /CR - LF/ lub pionową tabulacją /VT/. Przy transferze z DOS 11 i RT 11 na Files 11, znaki /CR - LF/ są usuwane z końca rekordu. Przy transferze z Files 11 na DOS 11 lub RT 11, pary /CR-LF/ są uzupełniane na końcu każdego rekordu,

Klucz	Opis
	n - określa długość rekordu, rekordy wyjściowe są w razie potrzeby uzupełniane zerami.
/RM:n	Rekordy zbioru wyjściowego będą stałej długości n słów /o długości 16 bitów/. Jeśli n nie występuje, to rekord ma długość 512 /10/ bajtów.
	Klucze kontrolne
/BL:n	Określa liczbę bloków ciągłego pola przydzielonego dla zbioru wyjściowego. Klucz ten jest używany łącznie z kluczem /CO. Jeśli klucz /BL nie występuje to obszar zbioru wyjściowego jest taki jak zbioru wejściowego.
/BS:n	Określa w bajtach rozmiar zbioru wyjściowego dla kaset.
/CO	Określa, że zbiór wyjściowy jest ciągły, klucz ten jest używany dla dysków i DECtype. Jeśli zbiorem wejściowym jest taśma papierowa lub kaseta to klucz /CO powinien występować łącznie z /BL .
/DE	Kasuje zbiory DOS 11, używany jest łącznie z kluczem RT do kasowania zbiorów RT 11.
/DI /LI	Klucze umożliwiają drukowanie katalogów kasety lub tomów DOS na wyspecyfikowanym zbiorze wyjściowym.
/UI	Wskazuje, że zbiór wyjściowy ma mieć ten sam UIC, co wejściowy. Klucz ten jest ważny tylko dla zbiorów Files 11 i DOS 11 /por. Rozdz. 0/.
/VE	Klucz powoduje, że każdy rekord zapisany na kasecie zostanie odczytany i zweryfikowany. Klucz ten jest ważny jedynie dla kaset.
/ZE	Inicjalizuje kasety typu DOS-11.

PRZYKŁADY:

1. >FLX PP:/DO=DK2:TASK.FTN/RS
   Zbiór TASK.FTN przechowywany na dysku DK2: w formacie Files 11 zostanie wyprowadzony na perforator w formacie DOS 11.

2. >FLX DK2:/RS/BL:1/CO=PR:TASK.FTN/DO
   Zbiór zostanie wczytany z czytnika taśmy na dysk DK2:, zbiór zajmuje jeden ciągły blok pamięci i będzie miał nazwę TASK.FTN.

3. >FLX DK0:/DO=DK1:SYS1.MAC/RS
   Przepisywanie zbioru SYS1.MAC na dysk DKØ:, zmiana formatu z Files 11 na DOS 11.

4. >FLX DK1:/ZE
   Inicjalizacja dysku DK1: typu DOS.

5. >FLX DK2:ZBIOR.DAT/DE
   Kasowanie zbioru ZBIOR.DAT.

10. Program PRI

Program PRI umożliwia drukowanie plików, jest to podstawowy program przy pomocy którego, możemy drukować zawartość plików na drukarce.

Format wywołania:

>PRI [nnk:=] plik/klucze

nnk: - specyfikacja urządzenia, /domyślnie LP:/,

plik - specyfikacja pliku /por. Rozdz. 0/,

klucze - następujące klucze:

/CO:n - klucz określający ilość kopii wydruku, jeśli klucz CO nie występuje to przyjmuje się n=1 /jedna kopia/.

/DEL - klucz określa, że plik po wydrukowaniu powinien być usunięty.

PRZYKŁAD:

>PRI TEKST.LST/CO:2

Na drukarce zostanie dwukrotnie wydrukowany zbiór TEKST.LST.

11. Opis programu DSC

Program DSC umożliwia kopiowanie obrazu dysku lub taśmy magnetycznej na dysku lub taśmie magnetycznej.

Postać wywołania programu DSC:

> DSC urzwy=urzwe

urzwy - nazwa urządzenia wyjściowego /dysku lub taśmy/,

urzwe - nazwa urządzenia wejściowego /dysku lub taśmy/.

PRZYKŁADY:

1. >DSC DK2:=DK3:

Kopiowanie obrazu dysku DK3: na dysku DK2:

2. >DSC MT1:=DK2:

Kopiowanie obrazu dysku DK2: na taśmie magnetycznej MT1:.

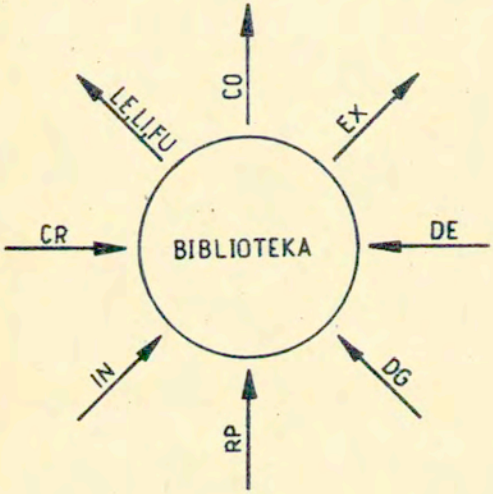
3. >DSC DK2:=MT1:

Kopiowanie obrazu taśmy magnetycznej MT1: na dysk DK2:.

12. Program obsługi bibliotek LBR

Program obsługi bibliotek umożliwia:

1. Tworzenie bibliotek /klucz CR/
2. Umieszczanie modułów programowych w bibliotece /klucz IN/
3. Uzyskiwanie informacji o zawartości biblioteki /klucze LE, LI,FU/
4. Uzyskanie dostępu do modułów zawartych w bibliotece i utworzenie z nich nowych plików /klucz EX/
5. Modyfikację modułów biblioteki /klucz RP/
6. Usunięcie punktów wejścia /klucz DG/
7. Usunięcie modułów z biblioteki /klucz DE/
8. Kompresję biblioteki /klucz CO/



rys. 2 Schematyczne przedstawienie programu LBR.

Biblioteka użytkownika określona jest przez następujące wartości: typ, maksymalną liczbę modułów, wchodzących do biblioteki, maksymalną liczbę punktów wejścia. x/
Typ biblioteki określa jakiego typu moduły w bibliotece występują. W bibliotece mogą występować jedynie moduły tego samego typu. Biblioteki zakładane pod systemem RSX mogą mieć typ:

.OBJ - zawierają moduły programowe w kodzie wynikowym,
.MAC - zawierają makroinstrukcje,
.UNI - zawierają moduły innych nie wymienionych powyżej typów /np. LST, FOR/

Modułom znajdującym się wewnątrz bibliotek typu .OBJ nadaje się nazwy wewnątrz programu źródłowego /nazwy te nie pochodzą od nazw plików/.
Modułom zawierającym makroinstrukcje nadaje się nazwy dyrektywą .MACRO, nazwy modułów występujących w bibliotekach typu .UNI są identyczne jak nazwy plików, na których moduły te się znajdują.
Nazwy punktów wejścia /symboli globalnych/ i nazwy modułów są umieszczane w specjalnych tablicach punktów wejścia /EPT/ i nazw modułów /MNT/. Jeśli do biblioteki wprowadzany jest nowy moduł, to do tablic MNT i EPT są automatycznie wprowadzane nazwy modułów i odpowiadające im nazwy punktów wejścia.

A. Tworzenie bibliotek

Wywołanie programu LBR:

LBR BIB/CR[:[size]:[ep]:[mn]:[lt]:[li]]=INF1,...INFn

BIB - specyfikacja biblioteki,
size - rozmiar biblioteki w blokach po 256 słów, domyślnie przyjmuje się 100 bloków,
ep - liczba punktów wejścia do biblioteki, wartość domyślna 512 dla bibliotek typu .OBJ i 0 dla pozostałych,
mn - liczba modułów w bibliotece, domyślnie 256,
lt - typ biblioteki:

x/ Uwaga: dotyczy np. wersji FORTRANu IV PLUS

.OBJ - dla modułów w kodzie pośrednim,
.MAC - dla bibliotek zawierających makroinstrukcje,
.UNI - dla bibliotek uniwersalnych,

domyślnie przyjmuje się typ .OBJ.

li - typy modułów wchodzących do biblioteki uniwersalnej, dla bibliotek pozostałych typów parametr ten nie jest podawany,

CR - klucz oznaczający tworzenie nowej biblioteki.

Parametry liczbowe występujące w powyższym opisie są wartościami dziesiętnymi /jeśli są zakończone kropką/ lub wartościami oktalnymi /w pozostałych przypadkach/.

INF1,...,INFn - nazwy modułów wejściowych.

B. Umieszczanie modułów programowych w bibliotece

Wywołanie programu LBR:

>LBR BIB/IN=INF1,...,INFn

BIB - specyfikacja biblioteki,

INF1,...,INFn - specyfikacja plików zawierających moduły, które będą umieszczone w bibliotece.

IN - klucz oznaczający, że zawartość plików zostanie umieszczona w bibliotece.

C. Uzyskanie informacji o zawartości biblioteki:

Wywołanie programu LBR:

>LBR BIB,LIST/klucz

BIB - specyfikacja biblioteki,

LIST - specyfikacja zbioru w którym przechowywane będą informacje o bibliotece,

klucz - nazwa jednego z następujących kluczy:
LI - drukowanie nazw modułów,
LE - drukowanie nazw modułów i punktów wejścia,
FU - drukowanie pełnej informacji o modułach.

D. Przeczytanie modułu z biblioteki i stworzenie nowego pliku.

Wywołanie programu LBR:

>LBR OUF=BIB/EX:MOD1,MOD2,...,MODn

OUF - specyfikacja pliku w którym umieszczone zostaną moduły,
BIB - specyfikacja biblioteki,
MOD1,...,MODn - nazwy modułów,
EX - klucz oznaczający, że moduły będą czytane z biblioteki i zapisywane w pliku OUF.

E. Modyfikacja modułów biblioteki

Wywołanie programu LBR:
> LBR BIB/RP=INF1,...,INFn
BIB - specyfikacja biblioteki,
INF1,...,INFn - specyfikacje plików zawierających moduły, które mają być zamienione,
RP - klucz oznaczający modyfikację.
W wyniku powyższego wywołania programu LBR moduły biblioteki zostaną zamienione przez moduły wejściowe o tej samej nazwie.

F. Usuwanie punktów wejścia z tablicy punktów wejścia

Biblioteka użytkownika zawiera dwie tablice: tablicę punktów wejścia /EPT/ i tablicę nazw modułów /MNT/. Czasami korzystne jest usunięcie nazw punktów wejścia z tablicy np. w przypadku wprowadzania modułów o tych samych punktach wejścia.
Wywołanie programu LBR:
>LBR BIB/DG:EP1[:EP2...:EPn]
BIB - nazwa biblioteki,
EP1,...,EPn - nazwy punktów wejścia usuwanych z biblioteki,
DG - klucz oznaczający usuwanie punktów wejścia.

C. Usunięcie modułów z biblioteki

Wywołanie programu LBR:
>LBR BIB/DE:MOD1 [:MOD2:...:MODn]
BIB - nazwa biblioteki
MOD1,...,MODn - nazwy usuwanych modułów.
W wyniku powyższego wywołania programu LBR moduły MOD1,..., MODn zostaną usunięte z biblioteki /wraz z odpowiadającymi

punktami wejścia/. Moduły stają się niedostępne dla użytkownika, ale nadal fizycznie istnieją w bibliotece.

H. Kompresja biblioteki

Wywołanie programu LBR:

> LBR NEWBIB/CC[:size1]:[ep]:[mn]]=OLDBIB

NEWBIB - specyfikacja nowej biblioteki, uzyskanej w wyniku kompresji biblioteki istniejącej.

size - rozmiar nowej biblioteki w blokach po 256 słów, wartością domyślną jest rozmiar starej biblioteki.

ep - maksymalna liczba punktów wejścia do biblioteki, wartością domyślną jest wartość parametru ep dla starej biblioteki,

mn - maksymalna liczba modułów nowej biblioteki, wartością domyślną jest wartość parametru mn dla starej biblioteki.

OLDBIB - specyfikacja starej biblioteki.

W wyniku powyższego wywołania programu LBR tworzona jest nowa biblioteka, która jest skompresowaną starą biblioteką tzn. niedostępne dla użytkownika moduły /por. Usuwanie modułów z biblioteki/ zostaną fizycznie usunięte z biblioteki. Zwolniona przestrzeń zostaje umieszczona na końcu nowoutworzonej biblioteki.

PRZYKŁADY:

> LBR BIB/CR=PROG1

Utworzenie biblioteki o nazwie BIB, biblioteka będzie składała się ze 100 bloków po 256 słów, będzie miała typ .OBJ może zawierać do 256 modułów. W pliku PROG1 znajdują się moduły, które zostaną umieszczone w bibliotece.

> LBR BIB/IN=PROG2

Do istniejącej biblioteki BIB zostaną wprowadzone moduły znajdujące się na pliku PROG2.

> LBR BIB/FU

Na terminalu ukaże się pełna informacja o modułach zawartych

w bibliotece BIB.

>LBR NEW=BIB/EX:MOD1

Moduł MOD1 występujący w bibliotece BIB zostanie zapisany na pliku NEW.

>LBR BIB/RP=FIL1

Moduły znajdujące się na pliku FIL1 zastąpią moduły biblioteki BIB posiadające te same nazwy.

>LBR BIB/DE:MOD1

Skasowanie modułu MOD1 znajdującego się w bibliotece BIB.

>LBR BIB2/CO=BIB

Utworzenie nowej biblioteki BIB2, biblioteka ta jest skompryмowaną biblioteką BIB. Atrybuty bibliotek BIB2 i BIB są takie same.

13. Pośrednie pliki zleceń

Pośrednie pliki zleceń są plikami tekstowymi zawierającymi listę zleceń interpretowalnych przez pojedyncze zadania systemowe /np. TKB, MAC, FOR/. Jeśli użytkownik przewiduje, że dane zadanie systemowe będzie wykonywał kilkakrotnie z niezmienianą listą zleceń, to nie musi przy każdym wywołaniu programu pisać pełnej ich listy, a jedynie nazwę /poprzednio utworzonego/pliku zawierającego te zlecenia. Nazwa pliku musi być poprzedzona znakiem @ /.

PRZYKŁAD:

Plik /przeznaczony dla programu TKB/ o nazwie PTKB.CMD zawiera tekst:

```
TASK,TI:=PROG,LB:[1,1]FOROTS/LB
```

Następujące wywołania programu TKB są równoważne:

```
TKB TASK,TI:=PROG,LB:[1,1]FOROTS/LB
TKB @PTKB.CMD
```

Gdy wykonanie zleceń zawartych w pośrednim pliku zleceń zostaje zakończone to na terminalu ukazuje się informacja:

```
@EOF
```

Pośrednie pliki zleceń można podzielić na dwie grupy: pośrednie pliki zleceń dla zadań i pośrednie pliki zleceń dla MCR.

Pośrednie pliki zleceń dla zadań zawierają jedynie linie tekstu normalnie związane z danym zadaniem /por. PRZYKŁAD/, pośrednie pliki zleceń dla MCR mogą zawierać dodatkowo specjalne dyrektywy umożliwiające sterowanie kolejnością wykonywania zleceń. Dyrektywy pośredniego pliku zleceń rozpoczynają się znakiem "." i umożliwiają:

1. Definiowanie symboli i przyporządkowanie im wartości logicznych, liczbowych lub znakowych /.ASK,.ASKS,.ASKN .DEC,.INC,.ENABLE,.DISABLE,.SET,.SETS,.SETN,.TEST/
2. Zatrzymanie wykonywania zleceń /.PAUSE,.WAIT,.DELAY, .STOP/
3. Wykonywanie testowań logicznych /.IF,.IFACT,.IFNACT, .IFDF,.IFNDF,.IFT,.IFF/

4. Wykonywanie operacji logicznych /.AND, .OR/
5. Dostęp do zbiorów danych i plików zleceń /.CLOSE,.CLOSEA, .OPEN,.DATA,.CHAIN/
6. Zmianę kolejności wykonywania zleceń /.EXIT,.GOTOSUB, .GOTO/
7. Określenie struktury blokowej /.BEGIN,.END/
8. Równoległe wykonywanie zadań /.XQT/.

Dyrektywy .BEGIN i .END określają strukturę blokową /z zanurzeniami/pliku zleceń. Struktura blokowa umożliwia wprowadzenie symboli i etykiet lokalnych /tzn. określonych jedynie w bloku, w którym zostały zdefiniowane/. Jeśli użytkownik definiuje symbol, to nazwa /i typ symbolu/ zostanie umieszczona w specjalnej tablicy symboli. Jeśli użytkownik usiłuje nadać symbolowi wartość niezgodną z uprzednio zdefiniowanym typem to sygnalizowany jest błąd. Nazwy symboli lokalnych /tzn. definiowanych wewnątrz bloku .BEGIN - .END/ zostają usunięte z tablicy symboli po zakończeniu wykonywania dyrektyw bloku w którym zostały zdefiniowane.

Pośrednie pliki zleceń MCR mogą zawierać także komentarze. Komentarz zaczynający się od nowej linii musi być poprzedzony znakiem ";", jeśli komentarz występuje w tej samej linii w której dyrektywa to musi rozpoczynać się znakiem "!". Pośrednie pliki zleceń umożliwiają definiowanie symboli. Symbole można podzielić na trzy grupy: symbole logiczne, liczbowe i znakowe. Na nazwę symbolu może składać się od jednego do sześciu znaków alfanumerycznych i znaku "$". Symbol w plikach pośrednich odgrywa rolę zmiennej. Etykiety muszą występować na początku linii. Nazwa etykiety składa się z /od 1 do 6/ znaków alfa numerycznych dodatkowo może wystąpić znak "$".

Łańcuchy znaków są ujmowane w znaki/"/. Długość łańcucha nie może przekroczyć 80 znaków. Na łańcuchach i symbolach powiązanych z łańcuchami można dokonywać operacji konkatenacji. /Operator "+" jest operatorem konkatenacji/. Z łańcuchów powiązanych z symbolami znakowymi można wyodrębnić podłańcuchy. Zapis $\overline{S}[n,m]$, gdzie $\overline{S}$ to nazwa symbolu określa podłańcuch

zawierający od n-tego do m-tego znaku łańcucha $\overline{S}$.
PRZYKŁAD:
Niech symbol $\overline{S2}$ będzie powiązany z łańcuchem "ABCDEF", wtedy zapis $\overline{S2}$:[1;3] określa podłańcuch "ABC".
Oprócz nazw pliki pośrednie są charakteryzowane przez klucze.
Klucz:
/MC - określa, że pośredni plik zleceń jest plikiem utworzonym dla MCR.
/DE - określa, że plik powinien być zniszczony po wykonaniu zawartych w nim zleceń.
/TR - powoduje, ze na terminalu wyświetlana będzie lista wykonywanych zleceń pośredniego pliku.
Zaprzeczeniem kluczy /MC,/DE,/TR są klucze /NOMC,/NODE, /NOTR lub klucze /-MC,/-DE,/-TR.

A. Dyrektywy definiowania i nadawania symbolom wartości logicznych, liczbowych i znakowych.
Dyrektywa .ASK
Format wywołania:
.ASK ssssss string
string - łańcuch znaków,
ssssss - nazwa symbolu logicznego /o długości od 1 do 6 znaków/.
W wyniku wywołania dyrektywy na terminalu ukazuje się pytanie /string/, procesor czeka na odpowiedź. W zależności od uzyskanej odpowiedzi /YES lub NO/ symbolowi logicznemu ssssss zostanie nadana wartość TRUE lub FALSE. Jeśli symbol ssssss wystąpił pierwszy raz w liście instrukcji to procesor dokonuje zapisu tego symbolu w liście symboli. Jeśli symbol był zdefiniowany wcześniej jako łańcuch lub liczba, to sygnalizowany jest błąd.

Dyrektywa .ASKS
Format wywołania:
.ASKS ssssss string lub
.ASKS [low:high] ssssss string
ssssss - nazwa symbolu o wartościach znakowych. Nazwa ma

długość od jednego do sześciu znaków.
low:high - dolne i górne ograniczenie długości odpowiedzi,
string - łańcuch znaków o długości co najwyżej 70 znaków /tekst wyświetlany na terminalu/.
W wyniku wykonania dyrektywy na terminalu ukazuje się tekst /string/. Nazwa symbolu powiązana jest z tekstem napisanym przez użytkownika. /Por. .ASK/.

Dyrektywa .DEC
Format wywołania:
.DEC ssssss
ssssss - nazwa symbolu liczbowego,
Dyrektywa. DEC zmniejsza wartość symbolu ssssss o jeden.

Dyrektywa .INC
Format wywołania:
.INC ssssss
Dyrektywa zwiększa wartość symbolu liczbowego o jeden.
Dyrektywa .ASKN
Format wywołania :
.ASKN ssssss string lub
.ASKN [low:high] ssssss string lub
.ASKN [:def] ssssss string lub
.ASKN [low:high:def] ssssss string
gdzie:
string - łańcuch znaków o maksymalnej długości 70 znaków. Jest to tekst wyświetlany przez procesor pliku pośredniego na terminalu.
ssssss - nazwa symbolu o wartościach liczbowych. Nazwa może mieć długość od jednego do sześciu znaków.
low:high - ograniczenia wartości liczbowej jaka może być przyjmowana przez symbol.
def - wartość domyślna symbolu.
W wyniku wywołania dyrektywy .ASKN na terminalu wyświetlane jest pytanie /string/ o wartość symbolu ssssss, jeśli odpowiedzią jest linia pusta to symbol przyjmuje wartość domyślną określoną przez parametr def, w przeciwnym przypadku otrzymuje wartość nadaną przez użytkownika. Wartość ta może być

wartością oktalną, jeśli rozpoczyna się od znaku "%" lub wartością dziesiętną, jeśli kończy się kropką. Jeśli oba wymienione znaki /"%", "."/ nie występują to przyjmuje się typ dziesiętny symbolu, jeśli parametry low, high, def są typu dziesiętnego /zakończone kropką/ lub typ oktalny w pozostałych przypadkach.

Dyrektywy .SETT i .SETF

Format wywołania dyrektyw:

.SETT ssssss

.SETF ssssss

ssssss - nazwa symbolu logicznego.

Dyrektywa .SETT nadaje symbolowi logicznemu wartość TRUE, a dyrektywa .SETF wartość FALSE.

Dyrektywa .SETN

Format wywołania dyrektywy:

.SETN ssssss numexp

ssssss - symbol o wartościach liczbowych,

numexp - wyrażenie arytmetyczne.

Dyrektywa nadaje symbolowi ssssss obliczoną wartość wyrażenia, jeśli wszystkie składniki wyrażenia są liczbami oktalnymi to wynik jest zapamiętany oktalnie, w przeciwnym przypadku jest zapamiętany dziesiętnie.

. Dyrektywa .SETS

Format wywołania dyrektywy:

.SETS ssssss strexp

ssssss - nazwa symbolu o wartościach znakowych,

strexp - wyrażenie o wartościach znakowych, strexp może składać się z łańcucha znaków zawartego w apostrofy, z łańcuchów i symboli o wartościach znakowych połączonych znakiem "+" /"+" - znak konkatenacji/.

Dyrektywa .SETS przyporządkowuje symbolowi ssssss wartość wyrażenia strexp.

Dyrektywa .TEST

Format wywołania dyrektywy :

.TEST strsym

strsym - nazwa symbolu o wartościach znakowych.

Dyrektywa .TEST oblicza ilość znaków łańcucha powiązanego z symbolem strsym, otrzymany wynik przechowuje symbol globalny STRLEN. Następnie określony zostaje kod znakowy łańcucha. W zależności od wyniku symbolom globalnym ALPHAN i RAD50 nadawane są odpowiednie wartości logiczne.

B. Dyrektywy zatrzymania wykonania zleceń

Dyrektywa .PAUSE
W wyniku wywołania dyrektywy następuje zatrzymanie wykonywania zleceń i na terminalu ukazuje się informacja:
AT.-- PAUSING TO CONTINUE TYPE " UNS tttttt"

Po napisaniu zlecenia UNS tttttt, gdzie tttttt nazwa pliku pośredniego z którego wywołana została dyrektywa .PAUSE, na terminalu ukazuje się informacja:

AT.--CONTINUING
i zostaje wznowione wykonywanie dyrektyw pośredniego pliku.

Dyrektywa .DELAY
Format wywołania:
.DELAY nnu
nn - liczba jednostek czasu podana oktalnie lub dziesiętnie /jeśli została zakończona kropką/.
u - litera określająca jednostkę czasu:
H - godziny, M-minuty, S-sekundy, T-tiki /tik odpowiada 1/50 sekundy/.
W wyniku wywołania dyrektywy .DELAY zawieszone zostanie wykonywanie dyrektyw pośredniego pliku zleceń na określony przez parametr nnu czas, a na terminalu ukazuje się informacja:
AT. DELAYING
Jeśli czas określony przez parametr nnu minął to na terminalu ukazuje się tekst:

AT. CONTINUING

Dyrektywa .WAIT
Format wywołania :
.WAIT NAME

NAME - nazwa aktywnego zadania.
Dyrektywa .WAIT powoduje zawieszenie wykonywania zleceń pliku pośredniego, aż do zakończenia wykonywania zadania o nazwie NAME. Jeśli nazwa zadania nie została podana to wykonywanie zleceń jest zawieszone, aż zostanie zakończone wykonywanie ostatnio uruchomionego dyrektywą RUN zadania użytkownika.

Dyrektywa .STOP
Dyrektywa .STOP kończy wykonywanie zleceń pliku pośredniego.

C. Dyrektywy wykonujące testowania logiczne

Dyrektywa .IF
Format wywołania dyrektywy:
.IF symbol relop expr dyrektywa
symbol - nazwa symbolu o wartościach liczbowych lub znakowych,
expr - wyrażenie o wartościach liczbowych lub znakowych,
relop - operator arytmetyczny /GE lub > = , GT lub >,
LE lub < = , LT lub < , NE, EQ lub = /.
Jeśli wyrażenie symbol relop expr jest prawdziwe to wykonywana będzie dyrektywa.

Dyrektywy .IFACT i .IFNACT
Format wywołania dyrektyw :
.IFACT TASK dyr
.IFNACT TASK dyr
TASK - nazwa zadania.
Dyrektywa .IFACT /IFNACT/ sprawdza czy zadanie o nazwie TASK jest aktywne /nieaktywne/, jeśli wynik testu jest pozytywny to wykonywana jest dyrektywa dyr. W przeciwnym wypadku dyrektywa dyr nie będzie wykonywana.
Dyrektywy .IFDF i .IFNDF
Format wywołania dyrektyw:
.IFDF ssssss dyr
.IFNDF ssssss dyr
ssssss - nazwa symbolu,
dyr - nazwa dyrektywy.
Dyrektywa .IFDF /.IFNDF/ sprawdza czy symbol ssssss został /nie został/ zdefiniowany, jeśli wynik testu jest pozytywny

to wykonywana będzie dyrektywa dyr.

Dyrektywy .IFINS i .IFNINS
Format wywołania dyrektyw :
.IFINS TASK dyr
.IFNINS TASK dyr
TASK - nazwa zadania,
dyr - nazwa dyrektywy .
Dyrektywa . IFINS /.IFNINS/ sprawdza czy zadanie o nazwie TASK jest /nie jest/ zainstalowane. Jeśli wynik testu jest pozytywny to wykonywana będzie dyrektywa dyr.
W przeciwnym wypadku dyrektywa dyr nie jest wykonywana.
Dyrektywy .IFT i .IFF
Format wywołania dyrektyw :
.IFT ssssss dyr
.IFF ssssss dyr
ssssss - nazwa symbolu,
dyr - nazwa dyrektywy.
Dyrektywa .IFT /.IFF/ sprawdza wartość logiczną symbolu ssssss. Jeśli symbol ssssss ma wartość logiczną TRUE /FALSE/ to wykonana zostanie dyrektywa dyr.

D. Operacje logiczne

Dyrektywy .AND i .OR umożliwiają łączenie dyrektyw wykonujących testowania logiczne. Dyrektywy połączone przez .AND i .OR można zamykać w okrągłe nawiasy. W jednej linii tekstu może wystąpić więcej niż jedna dyrektywa .AND lub .OR.

E. Dyrektywy dostępu do zbiorów danych i plików zleceń.

Dyrektywa .CLOSE
Format wywołania:

.CLOSE n
n - numer zbioru /nadany dyrektywą .OPEN lub .OPENA/.
Dyrektywa .CLOSE zamyka zbiór o numerze n, zbiór ten został otwarty dyrektywą .OPEN lub .OPENA .

Dyrektywa .OPEN
Format wywołania :

.OPEN n NAME
n - numer zbioru /od 1 do 3/,
NAME - nazwa zbioru.
Dyrektywa .OPEN tworzy i otwiera zbiór, nadaje utworzonemu zbiorowi nazwę NAME i numer n. Jeśli podana przez użytkownika nazwa NAME jest nazwą zbioru, który powstał wcześniej to tworzona jest nowa wersja zbioru.

Dyrektywa .OPENA
Format wywołania :
.OPENA n NAME
n - numer zbioru,
NAME - nazwa zbioru.
Dyrektywa otwiera i nadaje istniejącemu zbiorowi o nazwie NAME numer n.

Dyrektywa .DATA
Format wywołania :
.DATA n string
n - numer zbioru /nadany dyrektywą .OPEN/, jeśli parametr n nie występuje to przyjmuje się n = 0.
string - tekst, który zostanie umieszczony w zbiorze o numerze n.
Dyrektywa .DATA umieszcza w zbiorze otwartym dyrektywą .OPEN łańcuch znaków /string/. Jeśli string jest umieszczoną w apostrofach nazwą symbolu, to w zbiorze umieszczony zostanie łańcuch znaków związany z daną nazwą /por. .ENABLE SUBSTITUTION/.

Dyrektywa .CHAIN
Format wywołania :
.CHAIN NAME/klucze
NAME - nazwa pośredniego pliku zleceń,
klucze - klucze występujące przy specyfikacji pośrednich plików zleceń.
Dyrektywa .CHAIN zamyka bieżący plik pośredni, maskuje wszystkie symbole lokalne. Następnie wykonywane są dyrektywy pliku o nazwie NAME.

F. Dyrektywy definiowania symboli

Dyrektywy .ENABLE i .DISABLE
Format wywołania dyrektyw:
.ENABLE tekst n
.DISABLE tekst n
n - numer zbioru danych /por. .ENABLE DATA/,
tekst - jeden z następujących tekstów:
GLOBAL /por.p.H/
SUBSTITUTION
DATA
ESCAPE

Dyrektywa .ENABLE GLOBAL informuje, że wszystkie symbole występujące wewnątrz pliku pośredniego /lub, jeśli została określona struktura blokowa wewnątrz bloku/ są symbolami globalnymi, symbole te powinny zaczynać się znakiem "◇".
Dyrektywa .ENABLE SUBSTITUTION powoduje, że wszystkie symbole znakowe zapisane w apostrofach zostaną zastąpione przez łańcuchy im odpowiadające.
Dyrektywa .ENABLE DATA powoduje przesłanie wszystkich linii znajdujących się między dyrektywami .ENABLE DATA i.DISABLE DATA do poprzednio otwartego dyrektywą .OPEN zbioru posiadającego numer n.
Dyrektywa .ENABLE ESCAPE powoduje, że na dyrektywy .ASK , .ASKN , .ASKS można odpowiadać naciskając jedynie klawisz ESC. Po naciśnięciu klawisza ESC zmiennej globalnej .ESCAPE zostaje nadana wartość TRUE.
Dyrektywa .DISABLE zawiesza działanie dyrektywy .ENABLE.

G. Dyrektywy zmiany kolejności wykonywania zleceń

Dyrektywa .EXIT
W wyniku wywołania dyrektywy .EXIT zakończone zostanie wykonanie dyrektyw bieżącego pliku zleceń i nastąpi powrót do poprzedniego pliku. Jeśli dyrektywa została wywołana wewnątrz bloku (.BEGIN,.END) to wykonywane będą dyrektywy występujące po dyrektywie .END.

Dyrektywa .GOSUB
Format wywołania:
.GOSUB label
label - nazwa etykiety, etykieta oznacza pierwszą linię procedury.
Dyrektywa .GOSUB powoduje zapamiętanie bieżącej pozycji pośredniego pliku zleceń, wykonanie dyrektyw znajdujących się między etykietą label i dyrektywą .RETURN. Następnie wykonywane będą dyrektywy znajdujące się za dyrektywą .GOSUB label.

Dyrektywa .GOTO
Format wywołania:
.GOTO label
label - nazwa etykiety.
Dyrektywa powoduje przejście do wykonywania dyrektywy znajdującej się w linii oznaczonej etykietą label. Przy wykonywaniu dyrektywy .GOTO wszystkie dyrektywy znajdujące się między dyrektywą .GOTO, a etykietą label nie będą wykonywane.

Dyrektywa .ONERR
Format wywołania:
.ONERR label
label - nazwa etykiety.
Jeśli przy wykonywaniu poprzedniego zlecenia nastąpił błąd to wykonywana będzie dyrektywa opatrzona etykietą label.

H. Dyrektywy określające strukturę blokową

Dyrektywa .BEGIN
Dyrektywa .BEGIN określa początek bloku dyrektyw.

Dyrektywa .END
Dyrektywa .END określa koniec bloku dyrektyw.

Wszystkie symbole określone wewnątrz bloku są symbolami lokalnymi /o ile nie wystąpiła dyrektywa .ENABLE GLOBAL/ tzn. są określone jedynie wewnątrz bloku.
Dyrektywy .BEGIN i .END muszą być jedynymi dyrektywami występującymi w danej linii.

I. Dyrektywy umożliwiające równoległe wykonywanie zadań

Dyrektywa .XQT

Format wywołania dyrektywy:

.XQT TASK command

TASK - nazwa zadania,

command - zlecenie dla zadania o nazwie TASK.

Zwykle procesor pośrednich plików zleceń przesyła zlecenie do programu MCR i czeka, aż wykonywanie zlecenia zostanie zakończone.

Dyrektywa .XQT umożliwia równoległe wykonywanie zadania TASK i występujących po dyrektywie .XQT zleceń pośredniego pliku.

Znak "/" występujący jako pierwszy niepusty znak w linii jest znakiem logicznego końca pliku /działa identycznie jak dyrektywa .STOP/. Może wystąpić w dowolnym miejscu pliku.

PRZYKŁADY:

1. .SETS SEND " THIS DATA "
   .OPEN TEMP
   .DATA "SEND"
   .CLOSE
   powyższy ciąg dyrektyw powoduje zapisanie tekstu
   THIS DATA jako treści zbioru TEMP.

2. .SETN N1 2
   .SETN N2 7
   .IF N1 = N2 PIP/LI
   Jeśli spełniony jest warunek N1 = N2 to wydrukowany zostanie katalog zbiorów użytkownika.

3. .SETS A "ABCDEF"

   Powyższa dyrektywa wiąże symbol A z łańcuchem "ABCDEF".

4. .XQT MAC TEST;TEST=TEST
   .XQT TKB BLD,BLD=BLD
   .WAIT MAC
   .WAIT TKB

   W wyniku wywołania powyższego ciągu dyrektyw zadania

MAC i TKB będą wykonywane równolegle.

5. .SETS STR2 "ZZZZ"
   .SETS X STR2+"BCD"
   W wyniku wykonania powyższych dyrektyw symbol STR2 będzie powiązany z łańcuchem ZZZZBCD.

6. .ASK CONT PRZERWAC PRACE
   .IFF CONT .GOTO 100
   / /por.p. I/
   .100: /etykieta/
   Jeśli użytkownik powiązał symbol CONT z wartością FALSE to wykonywanie dyrektyw pośredniego pliku zleceń będzie kontynuowane, w przeciwnym przypadku nastąpi zakończenie wykonywania dyrektyw.

7. .IFT A .AND .IFF B RUN HELP
   Jeśli symbol A został powiązany z wartością TRUE, a symbol B z wartością FALSE to zostanie uruchomiony program HELP.

14. Współpraca ze zbiorami i urządzeniami WE/WY w programach napisanych w j.a. FORTRAN

Wszystkie zbiory i urządzenia wykorzystywane w programach napisanych w j.a. FORTRAN są identyfikowane przy pomocy ich numerów logicznych. Powiązanie urządzeń lub zbiorów z ich numerami może odbywać się trzema sposobami:

a. wewnątrz programu /procedura ASSIGN/,
b. przy pomocy opcji programu TKB /konsolidator/,
c. umownie tzn. wg pewnego standardu.

A. Procedura ASSIGN

Procedura ASSIGN nadaje urządzeniu WE/WY lub zbiorowi numer logiczny. Powiązanie z numerem logicznym musi dokonać się przed pierwszą operacją WE/WY wykonywaną na tym zbiorze /urządzeniu/.

Wywołanie procedury ASSIGN ma następującą postać:

```
CALL ASSIGN ( n,NAME,i)
```

gdzie:

n - numer logiczny zbioru lub urządzenia, stała lub zmienna o dodatnich wartościach całkowitych,

NAME - nazwa zbioru lub urządzenia /w apostrofach/ lub nazwa tablicy /bez apostrofów/ w której nazwa zbioru lub urządzenia jest przechowywana. Jeśli parametr NAME nie występuje to przyjmuje się nazwę domyślną/por.Tab.9/

i - liczba znaków nazwy. Jeśli parametr ten nie występuje to znak pusty lub zero przyjmowany jest jako znak końca nazwy. Parametr ten jest istotny jedynie w przypadku gdy nazwa zbioru lub urządzenia przechowywana jest w tablicy.

Procedura ASSIGN nadaje numery logiczne zbiorom, których nazwy już występują w katalogu zbiorów użytkownika i zbiorom, które mają być utworzone w wyniku działania programu/, a więc w momencie wywołania procedury ASSIGN ich nazwy w katalogu jeszcze nie ma/.

PRZYKŁAD:

Poniższy segment programowy umożliwia uniwersalne wykorzystanie urządzeń i plików do operacji WE/WY.

```
      DIMENSION IDATE(5), IFILE(17)
      DATA NREC,LREC /5Ø, 13/
      CALL DATE(IDATE)
1ØØ   CALL ASSIGN(1,'TI:')
      WRITE(1,1ØØØ)
1ØØØ  FORMAT('O'/' *NAZWA ZBIORU? ')
      READ(1,1Ø1Ø) I, IFILE
1Ø1Ø  FORMAT(Q, 17A2)
      CALL ASSIGN(2,IFILE,I)
1Ø<Ø  FORMAT('*URZADZENIE REJESTRACJI! ')
      READ(1,1Ø1Ø) I, IFILE
      CALL ASSIGN(3,IFILE,I)
```

Instrukcja CALL ASSIGN (1,'TI:') /7. linia programu/ przyporządkowuje terminalowi użytkownika numer logiczny 1. Kolejny ciąg instrukcji:

```
      READ(1,1Ø1Ø) I, IFILE
1Ø1Ø  FORMAT(Q,17A2)
      CALL ASSIGN(2,IFILE,I)
```

umożliwia przeczytanie z terminala nazwy zbioru lub urządzenia i umieszczenie jej w tablicy, a następnie przyporządkowanie zbiorowi lub urządzeniu o wczytanej nazwie numeru logicznego 2.
Podstawową zaletą tej metody jest możliwość zmiany nazwy zbioru lub urządzenia bez konieczności ponownej edycji programu lub konsolidacji.

B. Opcje programu TKB

Aby nadać urządzeniom fizycznym numery logiczne na etapie konsolidacji /pracy programu TKB/ należy zastosować wieloliniowy format TKB z opcją ASG /por. opis konsolidacji/.

C. Umowne nazwy i numery logiczne zbiorów i urządzeń

TABELA 9

Umowne nazwy i numery logiczne zbiorów i urządzeń

Numer logiczny	Urządzenie	Nazwa umowna	Opis odpowiada następującemu wywołaniu procedury ASSIGN
1	SY:	FOR001.DAT	CALL ASSIGN(1,'SY:') lub CALL ASSIGN(1,'FOR001.DAT')
2	SY:	FOR002.DAT	CALL ASSIGN(2,'SY:') lub CALL ASSIGN(2,'FOR002.DAT')
3	SY:	FOR003.DAT	CALL ASSIGN(3,'SY:' lub CALL ASSIGN(3,'FOR003.DAT'
4	SY:	FOR004.DAT	CALL ASSIGN(4,'SY:') lub CALL ASSIGN(4,'FOR004.DAT')
5	TI:	FOR005.DAT	CALL ASSIGN(5,'TI:')
6	CL:	FOR006.DAT	CALL ASSIGN(6,'CL:')

Formatowe instrukcje WE/WY, które nie korzystają bezpośrednio z numerów logicznych

TABELA 10

Instrukcja	Forma równoważna	Opis
READ f,list	READ(1,f) list	f - numer formatu,
PRINT f,list	WRITE(6,f) list	list - lista parametrów
ACCEPT f,list	READ(5,f) list	WE/WY
TYPE f,list	WRITE(5,f)list	

Jeżeli użytkownik używa swoich zbiorów jedynie do przechowywania wyników z dostępem sekwencyjnym /zapisywanie lub odczytywanie kolejnych elementów zbioru/, to nie muszą występować

żadne dodatkowe deklaracje struktury zbioru i określania trybu dostępu. Ubocznym efektem każdej instrukcji WE/WY jest przesunięcie "strzałki" zbioru o liczbę rekordów określoną przez listę WE/WY. Jeżeli w jednym programie chcemy zapisać elementy do zbioru, a następnie odczytać dane, to po zakończeniu zapisywania można zamknąć zbiór procedura CLOSE, a następnie nadać mu ponownie numer logiczny procedurą ASSIGN /o ile nie korzystamy z umownych nazw zbiorów/. W wyniku tej operacji strzałka zbioru będzie ustawiona na początku zbioru.

PRZYKŁAD:

```
          DIMENSION OMEGA ( 6,1Ø,5 )
          CALL ASSIGN ( 5,'SY:')
    5Ø22  FORMAT( 5(2X , 4PE14.3))
          DO 919 J=1,6
          DO 919 I=1,1Ø
    919   READ ( 5,5Ø22)(OMEGA ( J,IR,IZ), IR=1,5)
          CALL  CLOSE(5)
          CALL ASSIGN ( 5 , 'SY:')
          DO 92Ø J=1,6
          DO 92Ø IZ=1,1Ø
    92Ø   WRITE (5,5Ø22)(OMEGA ( J,IR,IZ) , IR=1,5)
```

Opis fragmentu programu:
Przechowywane w tablicy OMEGA częściowe wyniki są zapamiętywane w zbiorze o numerze logicznym 5. Następnie zbiór o numerze logicznym 5 jest zamykany i następuje ponowne jego przyłączenie /procedurą ASSIGN/. Dzięki tej operacji strzałka zbioru jest ustawiona na początku zbioru i można ponownie odczytać przechowywane wyniki. Aby uniknąć ewentualnych pomyłek wskazane jest, aby odczytać dane dokładnie w ten sam sposób w jaki były zapisane.
Jeżeli użytkownik korzysta ze zbiorów w jakiś szczególny sposób np. musi odwoływać się do konkretnego rekordu zbioru, chce przechowywać dane w postaci binarnej, lub też z danego zbioru korzysta kilka programów jednocześnie, to konieczne jest dodatkowe określenie struktury, rozmiaru i trybu dostępu do zbioru. Specyfikacji zbioru można dokonać przy pomocy

instrukcji DEFINE FILE i procedury FDBSET.

D. Opis instrukcji DEFINE FILE

Instrukcja DEFINE FILE określa, że zbiór o danym numerze logicznym, będzie lub jest zbiorem bezpośredniego dostępu. Przy pomocy tej instrukcji określa się także strukturę i rozmiar zbioru /tzn. ilość i długość rekordów/. W instrukcji DEFINE FILE specyfikowana jest tzw. strzałka przechowująca numer rekordu do którego następnie, wewnątrz programu możemy się odwoływać. Strzałkę można inicjalizować, a także zmienia się ona w wyniku instrukcji WE/WY /jako efekt uboczny/. Postać wywołania instrukcji /o specyficznej strukturze bez określenia CALL/:

DEFINE FILE nl (m,n,U,v)

gdzie

nl - numer logiczny zbioru, stała lub zmienna całkowita i dodatnia.

m - stała lub zmienna całkowita i dodatnia określająca liczebność rekordów zbioru.

n - stała lub zmienna całkowita określająca długość /w 16-bitowych słowach/ pojedynczych rekordów składających się na zbiór.

U - określa, że zbiór jest zbiorem bezformatowym /binarnym/.

v - nazwa zmiennej całkowitej tzw. strzałki zbioru, inaczej zmiennej wiążącej.

E. Bezformatowe instrukcje WE/WY

Postać wywołania bezformatowych instrukcji READ i WRITE:

READ (u'r ,ERR=s) lista

WRITE (u'r ,ERR=s) lista

gdzie

u - numer logiczny zbioru,

r - stała lub zmienna /strzałka zbioru/ określająca numer jednostki zapisu /kolejny numer rekordu/,

s - etykieta.

Zapis ERR=s określa, że jeśli nastąpi błąd przy wykonywaniu

operacji WE/WY to sterowanie będzie przekazane w miejsce wskazane przez etykietę s.
Jeśli przy odczytywaniu /READ/ długość w bajtach jednostki zapisu jest dłuższa niż wymaga tego lista parametrów to pozostała część rekordu jest pomijana. Jeśli ilość danych przechowywanych w zbiorze jest za mała /za długa lista parametrow/ to sygnalizowany jest błąd.
Jeśli przy zapisywaniu danych /WRITE/ długość listy danych przekracza długość jednostki zapisu to sygnalizowany jest błąd. Jeśli lista danych jest za krótka, to pozostałe wolne miejsca rekordu uzupełniane są zerami, po zakończeniu operacji WE/WY zmienna wiąząca zwiększa swoją wartość.
PRZYKŁAD:

```
DIMENSION ARRAY(15)
DEFINE FILE 2(10,5,U,I)
I=1
.
.
.
WRITE(2'I) ARRAY
```

Tablica ARRAY jest zapisywana do zbioru o numerze logicznym 2.

F. Procedura FDBSET

Procedura FDBSET umożliwia określenie operacji WE/WY, które mają być wykonywane na zbiorze.
Wywołanie FDBSET ma postać:
CALL FDBSET (n,mode,share,nbuf,initsz,extend)
gdzie
n - numer logiczny zbioru, stała lub zmienna całkowita i dodatnia,
mode - łańcuch liter w apostrofach, określający tryb dostępu do zbioru, mode może przyjmować następujące wartości:
READ ONLY - możliwe jest jedynie odczytywanie danych z istniejącego już zbioru,
NEW - w programie tworzony będzie nowy zbiór,

OLD - przyłączenie istniejącego już zbioru,

APPEND - możliwe jest dopisywanie danych na końcu istniejącego zbioru sekwencyjnego,

UNKNOW - parametr ten jest używany jeśli niewiadomo jaki ma być tryb dostępu do zbioru,

MODIFY - modyfikacja zbioru już istniejącego, stara wersja jest niszczona,

LSUP - umożliwia modyfikację starego zbioru, wersja stara i nowa są zachowane.

Jeśli parametr mode nie występuje to tryb dostępu do zbioru zależy od pierwszej operacji WE/WY wykonywanej na zbiorze. Jeśli pierwszą operacją będzie READ to przyjmuje się mode=OLD, a jeśli WRITE to mode=NEW.

share - jeśli parametr share występuje to share=SHARE. Parametr ten określa, że zbiór może być używany przez więcej niż jeden program jednocześnie.

nbuf - parametr określający liczbę buforów wewnętrznych używanych przy operacjach WE/WY. W systemie RSX-11 wielobuforowanie nie występuje, toteż nbuf=1.

initsz - parametr określa wielkość obszaru, która powinna być zarezerwowana dla nowoutworzonego zbioru.

extend - parametr określa o jaką liczbę bloków zbiór ma zostać rozszerzony.

PRZYKŁAD:

Poniższy program zakłada zbiór danych o nazwie ZBIOR. ZBIOR jest zbiorem binarnym, zmienna NUMER jest strzałką rekordów. Ze zbioru będzie mogło korzystać kilka programów jednocześnie.

```
      INTEGER Nr,J,K
      REAL A
      READ (5,95) I
```

```
95      FORMAT(I3)
        CALL ASSIGN(1,'ZBIOR')
        DEFINE FILE 1 (1ØØ,2,U,NR)
        CALL FDBSET(1,1,'NEW','SHARE',1,1,Ø)
        NR=1
        WRITE(1'NR) I
        DO 1Ø J=1,I
        READ(5,96) K,A
96      FORMAT(I3,F5.2)
        WRITE(1'NR) K
        WRITE(1'NR) A
1Ø      CONTINUE
        NR=2
        DO 2Ø J=1,I
        READ(1'NR) K
        READ(1'NR) A
        WRITE(5,97) K, A
97      FORMAT(1H, I3, 1H, F5.2)
```

Powrót strzałki rekordów do początku uprzednio otwartego zbioru o numerze logicznym u można również spowodować stosując instrukcję:
REWIND u.

15. Spis literatury

1. Operacjonnaja sistema realnogo wremieni OS-RW. Translator s jazyka FORTRAN čast'1 , 2.
2. Manual TXT - 1.0
   Podstawowe możliwości systemu RSX 11 - M,
   Warszawa 1982.
3. PDP - 11 FORTRAN Language Reference Manual,
   Order No. DEC - 11 - LFR - C - D Digital Equipment
   Corporation, Maynard, Mass. 1975.
4. RSX - 11 Utilities Mannual , /Order No. AA - H268A-Tc/
   Digital Equipment Corporation, Maynard , Mass. 1979.
5. J.Wiśniewska
   Wybrane informacje o systemie operacyjnym RSX - 11M,
   Wydawnictwa Uniwersytet Warszawski 1983.

16. Spis tabel