

II 232510
Polska Akademia Nauk
ZAKŁAD APARATÓW MATEMATYCZNYCH

prace
zakładu aparatów matematycznych p a n



Praca C 1

MASZYNA XYZ
PODRĘCZNIK PROGRAMOWANIA W JĘZYKU MASZYNY
Systemy SAB, SAS i SO

Prace

ZAKŁADU APARATÓW MATEMATYCZNYCH
Polskiej Akademii Nauk

Praca C 1

MASZYNA XYZ
PODREĆZNIK PROGRAMOWANIA W JĘZYKU MASZYNY
Systemy SAB, SAS i SO

Warszawa 1961

Copyright © 1961 - by Zakład Aparatów Matematycznych, Warszawa
Wszelkie prawa zastrzeżone



11232.510

Biblioteka Narodowa
Warszawa



30001016505115

KOMITET REDAKCYJNY

Jerzy DĄDA, Jerzy PIETT /z-ca redaktora/, Maria LESZEŻANKA
/sekretarz redakcji/, Leon ŁUKASZEWICZ /redaktor/, Antoni MAZUR-
KIEWICZ, Tomasz PIETRZYKOWSKI, Zygmunt SAWICKI.
Adres redakcji: Warszawa, ul. Koszykowa 79 tel. 21-84-41
wew. 131

1961 eo 3388/
S /2

Podręcznik niniejszy ma służyć jako pomoc dla programisty, układającego programy w języku SAS. Zasadniczym celem opracowania jest podanie pełnej listy rozkazów maszyny XYZ-1 oraz instrukcji programowania w systemie SAS.

System SAS jest wynikiem prac prowadzonych w latach 1959-1960 przez zespół pracowników Działu Programowania Zakładu Aparatów Matematycznych PAN pod bezpośrednim kierownictwem Leona Łukaszewicza.

Zasadnicze prace związane z opracowaniem programów translatora wykonali: Stefan Sawicki, Jerzy Swianiewicz i Mieczysław Wiśniewski.

Podręcznik opracował Jerzy Swianiewicz.

M A S Z Y N A X Y Z
PODREČNIK PROGRAMOWANIA
W JĘZYKU MASZYNY
Systemy SAB, SAS 1 SO

S P I S T R E Ś C I

C z ę s ć p i e r w s z a

ROZDZIAŁ I - Opis maszyny XYZ-1	
1. PAMIĘĆ, SŁOWO, LICZBA, ROZKAZ	7
.1 Budowa liczby	9
.2 Budowa rozkazu	10
2. ARYTMOMETR I URZĄDZENIE STERUJĄCE	11
.1 Rejestry arytmometru	11
.2 Rejestry urządzenia sterującego	15
3. ARYTMETYKA MASZYNY	16
.1 Dodawanie, odejmowanie, odejmowanie wartości bezwzględnych	16
.2 Mnożenie	16
.3 Dzielenie	17
.4 Przesunięcia	18
.5 Zaokrąglanie	19
.6 Koniunkcja	19
.7 Właściwości zera	19
4. BĘBEN MAGNETYCZNY	20
5. URZĄDZENIA WEJŚCIA I WYJŚCIA	21
.1 Wejście	21
.2 Wyjście	23
ROZDZIAŁ II - Lista rozkazów maszyny XYZ-1	
OZNACZENIA	25
Rozkazy dotyczące arytmometru - Grupa A	26
Rozkazy sterujące - Grupa S	32
Rozkazy wejścia i wyjścia - Grupa W	35
Rozkazy bębnowe - Grupa B	40
Rozkazy specjalne - Grupa SP	42

SPIS TREŚCI

2

Prace ZAM

C z e s t c d r u g a

ZEWNĘTRZNE SYSTEMY PROGRAMOWANIA	47
ROZDZIAŁ I - System adresów bezwzględnych SAB	
1. OGÓLNE ZASADY PISANIA PROGRAMÓW W SAB	48
2. ROZKAZY	48
3. LICZBY	49
.1 Liczby całkowite krótkie	49
.2 Liczby całkowite długie	49
.3 Liczby ułamkowe długie	50
.4 Liczby ułamkowe krótkie	51
4. DYREKTYWY	51
.1 Rozdział N	52
.2 Start n	53
.3 Funkcja b	53
5. NUMERY	54
6. PODZIAŁ MIEJSC PAMIĘCI WEWNĘTRZNEJ	55
ROZDZIAŁ II - System adresów symbolicznych SAS	
1. BLOKI	58
2. SYMBOLE	58
.1 Symbole zmiennych liczbowych	58
.2 Numery symboliczne	59
.3 Paragrafy	59
3. ROZMIESZCZANIE INFORMACJI W PAMIĘCI WEWNĘTRZNEJ	60-
4. ADRESY ROZKAZÓW	60
.1 Adresy bezwzględne	60
.2 Adresy względne	61
.3 Adresy symboliczne	61
5. LICZBY	62
6. SZABLONY	62
7. NUMERY BEZWZGLĘDNE	63
8. NUMERY SYMBOLICZNE	63
9. PARAGRAFY	64
10. DYREKTYWY	64
.1 Rozdział N	66
.2 Start	66
.3 Funkcja b	66
.4 Wymiar	66
.5 Locum	67
.6 Tablica	68
.7 Tekst	68
11. KOMENTARZ	69

ROZDZIAŁ III - Podprogramy bębnowe. Podprogram komunikacji

1. PODPROGRAMY BĘBNOWE	70
.1 Funkcje bębnowe	71
.2 Podprogramy wejścia i wyjścia	76
2. PROGRAM KOMUNIKACJI	82
.1 Program współpracy z bębnem	82
.2 Program czytania taśmy	84

Uzupełnienia

1. Tablica znaków dalekopisowych dla wejścia i wyjścia na taśmie dziurkowanej	87
2. Tablica dwuliterowych skrótów części operacyjnych rozkazów	88
3. Tablica stałych uniwersalnych	89
4. Instrukcja wprowadzania i uruchamiania programów	90
5. Sygnalizacja błędów	100

C z e s t p i e r w s z a

ROZDZIAŁ I
O P I S M A S Z Y N Y X Y Z - 1

Maszyna XYZ-1 składa się z czterech podstawowych części:

1. PAMIĘĆ
2. ARYTMOMETR
3. URZĄDZENIA STERUJĄCE
4. URZĄDZENIA WEJŚCIA I WYJŚCIA

1. PAMIĘĆ, SŁOWO, LICZBA, ROZKAZ

Pamięć

Pamięć składa się z komórek zwanych miejscami pamięci. Każde miejsce ma swój numer, który jest adresem tego miejsca pamięci. Komórki pamięci mogą zawierać liczby lub rozkazy zapisane w systemie binarnym /w postaci ciągu zer i jedynek/.

Pamięć maszyny XYZ-1 składa się z

pamięci wewnętrznej, mającej bezpośrednie połaczenie z pozostałymi częściami maszyny, i

pamięci pomocniczej, służącej do przechowywania informacji, które nie mogą się pomieścić w pamięci wewnętrznej. Pamięć pomocnicza posiada bezpośrednie połaczenie jedynie z pamięcią wewnętrzną.

W maszynie XYZ-1 rolę pamięci pomocniczej spełnia bębenny magnetyczny /pamięć bębnowa/.

Pamięć wewnętrzna maszyny składa się z

- 512 długich miejsc pamięci; w każdym z nich może być zapamiętany ciąg 36 cyfr binarnych /bitów/

lub

- 1024 krótkich miejsc pamięci; w każdym z nich może być zapamiętany ciąg 18 cyfr binarnych /bitów/.

Każde długie miejsce pamięci jest sumą dwóch krótkich miejsc pamięci.

Długie miejsca pamięci mają numery parzyste: 0, 2, 4, 6,.. 1020, 1022. Krótkie miejsca pamięci ponumerowane są kolejno od 0 do 1023. Długie miejsce pamięci o numerze $2n$ składa się z dwóch krótkich miejsc pamięci o numerach $2n$ i $2n+1$.

Pamięć wewnętrzna podzielona jest na 32 części zwane rurami, z których każda zawiera 32 miejsca krótkie lub 16 miejsc długich. Rury ponumerowane są od 0 do 31, przy czym

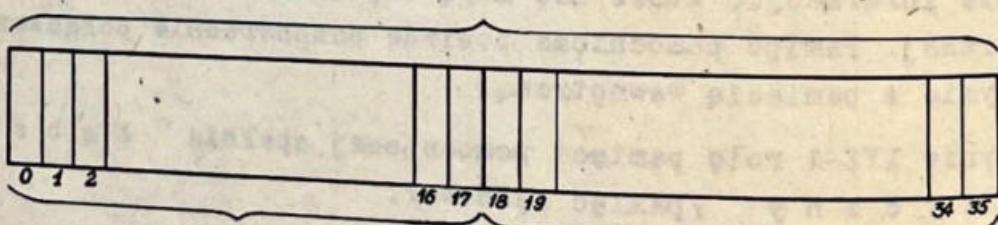
rura 0 obejmuje krótkie miejsca pamięci	0 - 31
rura 1 obejmuje krótkie miejsca pamięci	31 - 63
.....	
rura 31 obejmuje krótkie miejsca pamięci	992 - 1023.

Ciągi 36 bitów zapamiętane w długich miejscach pamięci nazywają się słowami długimi maszyny.

Ciągi 18 bitów zapamiętane w krótkich miejscach pamięci nazywają się słowami krótkimi maszyny.

Słowo długie zawarte w komórce pamięci o numerze $2n$ składa się ze słów krótkich zawartych w komórkach pamięci $2n$ i $2n+1$.

SŁOWO DŁUGIE



NIEPARZYSTE SŁOWO KRÓTKIE

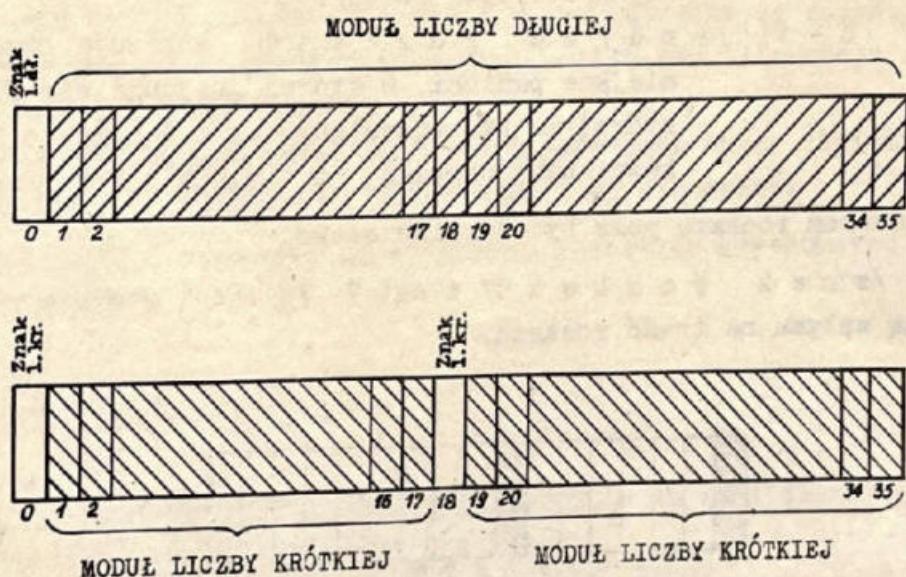
PARZYSTE SŁOWO KRÓTKIE

Rys. 1. Słowo długie

Ciągi bitów zawarte w komórkach pamięci mogą przedstawiać liczby lub rozkazy.

1.1. Budowa liczby

Słowo /długie lub krótkie/ traktowane jako liczba składa się ze znaku liczby i modułu.



Rys. 2. Budowa liczby

Znak liczby zapisany jest w zerowym bicie słowa /jedynka oznacza minus, a zero plus/. Pozostałe bity są cyframi binarnymi modułu liczby; najwyższą pozycją modułu jest bit 1, a najniższą bit 35 /w przypadku liczby długiej/ lub bit 17 /w przypadku liczby krótkiej/..

Jeżeli w jednym słowie długim umieszczone są dwie liczby krótkie, to znak liczby krótkiej, umieszczonej w miejscu pamięci o adresie pełrzystym, jest 18-tym bitem odpowiedniego słowa długiego, a sama liczba pokrywa się z niższymi pozycjami danego słowa długiego.

1.2. Budowa rozkazu

Rozkaz zajmuje jedno słowo krótkie.

Znaczenie poszczególnych pozycji binarnych rozkazu jest następujące:

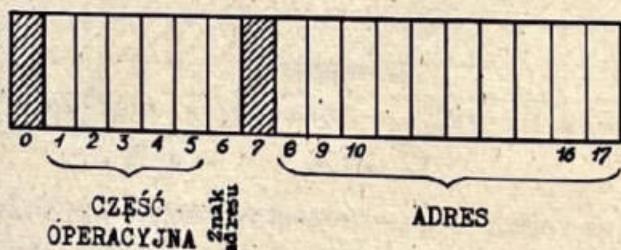
BITY 1 - 5 - część operacyjna określa operację, która ma być wykonana.

BIT 6 - znak adresu wskazuje czy adres rozkazu dotyczy słowa długiego /jedynka/ czy krótkiego /zero/.

BITY 8 - 17 - adres rozkazu wskazuje numer miejsca pamięci, w której znajduje się albo do której ma być przesyłany argument rozkazu, bądź określa parametr rozkazu.

Argumentem rozkazu może być dowolne słowo.

Bit 0 /znak rozkazu/ i bit 7 /bit rezerwowy/ nie mają wpływu na treść rozkazu.



Rys. 3. Rozkaz

W pamięci maszyny rozkazy niczym nie różnią się od liczb. O tym, jak jest traktowane poszczególne słowo, decyduje fakt, dokąd to słowo zostaje pobrane.

Jeżeli słowo krótkie zostaje pobrane do urządzenia sterującego, to jest ono traktowane jako rozkaz. Jeżeli dowolne słowo jest pobierane do arytmometru, to jest ono traktowane jak liczba. W komunikacji między pamięcią wewnętrzną a urządzeniami wejścia i wyjścia oraz bębnem słowa traktowane są jak zwykłe układy bitów.

Przy pobieraniu informacji z pamięci maszyny zawartość odpowiedniej

komórki pamięci pozostaje nie zmieniona. Natomiast przy przesyłaniu informacji do pamięci poprzednia zawartość odpowiedniej komórki zostaje skasowana i na jej miejsce wstawiona nowa. Jest to ogólna właściwość pamięci, zarówno wewnętrznej jak i bębnowej.

2. ARYTMOMETR I URZĄDZENIE STERUJĄCE

Arytmometr i urządzenie sterujące stanowią jądro maszyny cyfrowej. Wykonuje się w nich całość obliczeń i interpretowane są rozkazy zawarte w programie. Praca maszyny polega na pobieraniu informacji z pamięci wewnętrznej i przetwarzaniu ich.

W związku z tym arytmometr i urządzenie sterujące muszą zawierać rejestr y, do których te informacje są pobierane.

Poniżej podany jest opis podstawowych rejestrów arytmometru i urządzenia sterującego.

2.1. Rejestry arytmometru

Operacje arytmetyczne i logiczne wykonywane są na liczbach w części maszyny zwanej arytmometrem.
Zasadniczymi elementami arytmometru są: rejestr y (akumulator i mnożnik) oraz układy wykonujące działania.

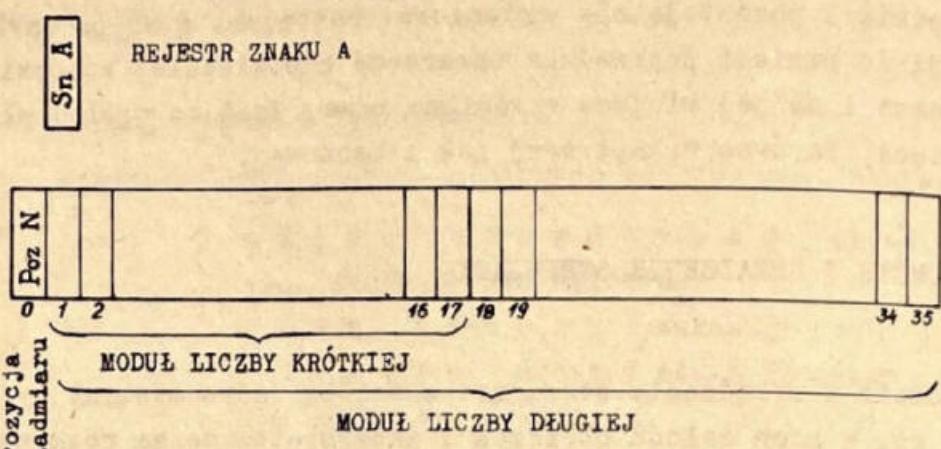
2.1.1. Akumulator, symbol A

Akumulator jest to główny register arytmometru. Posiada on właściwości addytywne, tzn. można do jego zawartości dodać lub odejmować od niej liczby z pamięci. W przypadku mnożenia umieszczana jest tu bardziej znacząca część iloczynu, a w przypadku dzielenia - reszta /z dzielenia/.

Pod nazwą akumulatora rozumie się w zasadzie dwa rejestr y:

rejestr akumulatora i

rejestr znaku akumulatora, symbol SnA



Rys. 4. Akumulator

R e j e s t r a s k u m u l a t o r a - jest to rejestr 36 bitowy. Przy przesyłaniu liczby do akumulatora poprzednia zawartość akumulatora zostaje skasowana, po czym znak liczby zostaje przekazany do rejestru znaku akumulatora, a moduł wchodzi na pozycje 1 - 35 rejestru akumulatora. Pozycja zerowa - zwana pozycją nadmiaru akumulatora - pozostaje zerem. W pozycji nadmiaru może się pojawić jedynka w trakcie wykonywania pewnych działań w akumulatorze, gdy liczba będąca wynikiem któregoś z działań przekroczyła zakres maszyny.

Przy przesyłaniu liczby z akumulatora do pamięci cyfra z pozycji nadmiaru nie jest przesyłana.

Przy przesyłaniu do akumulatora liczby krótkiej moduł tej liczby zajmuje pozycje 1 - 17 rejestru akumulatora, pozostałe pozycje są wyzerowane. Ogólnie można powiedzieć, że liczba krótka w arytmometrze jest traktowana tak samo jak liczba dłuża, której bity 18 - 35 są zerami.

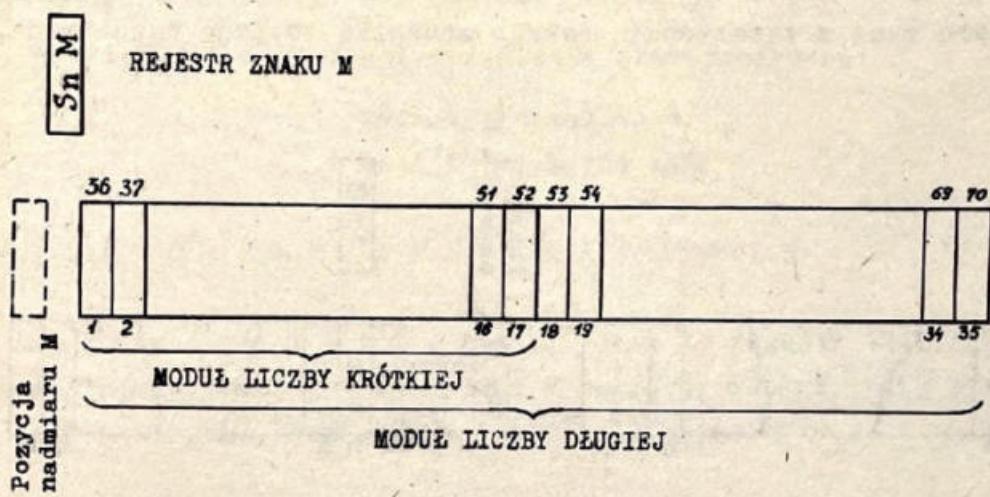
2.1.2. Mnożnik, symbol M

Mnożnik jest to drugi podstawowy rejestr arytmometru. Umieszcza się w nim jeden z czynników w przypadku wykonywania operacji mnożenia. Po wykonaniu mnożenia w mnożniku znajdują się niższe pozycje iloczynu. W przypadku dzielenia przed wykonaniem operacji w mnożni-

ku znajdują się niższe pozycje dzielnej /wyższe są w akumulatorze/, a po wykonaniu dzielenia zostaje tu umieszczony iloraz.

Podobnie jak w przypadku akumulatora pod nazwą mnożnika rozumie się w zasadzie dwa rejestryst:

rejestr mnożnika
rejestr znaku mnożnika, symbol SnM .



Rys. 5. Mnożnik

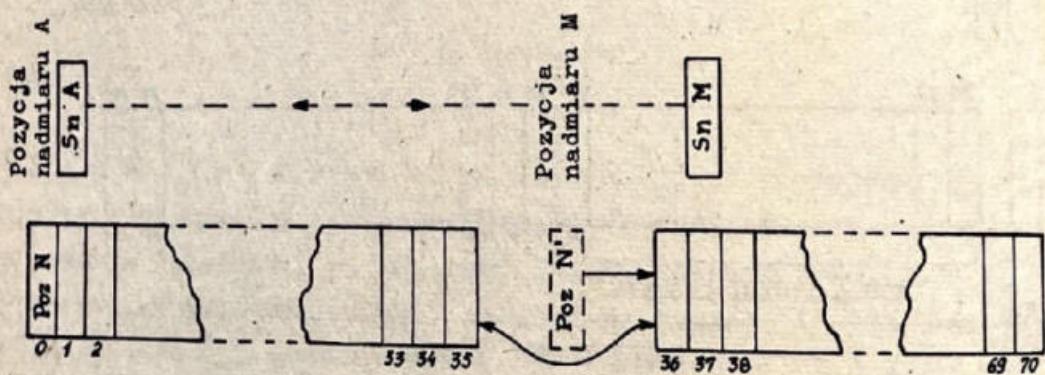
Rejestr mnożnika jest to rejestr 35-bitowy. Przy przesyłaniu liczby do mnożnika poprzednia jego zawartość zostaje wyzerowana, po czym znak liczby zostaje przesłany do SnM, a moduł wchodzi na pozycje 1 - 35 rejestrów mnożnika. Przy przesyłaniu do mnożnika liczby krótkiej moduł tej liczby zajmuje pozycje 1 - 17 rejestrów M, a pozostałe pozycje zostają wyzerowane.

Jak wynika z powiedzianego wyżej, rejestr mnożnika często jest traktowany jako przedłużenie akumulatora. Tak jest, gdy w rejestrach tych umieszczony jest iloczyn dwóch liczb lub dzielna /przed dzieleniem/. Również w przypadku operacji przesuwania zawartości A i M rejestr te traktowane są jako jedna całość. W powyższych przypadkach bitы mnożnika otrzymują numery od 36 do 70.

Rejestr mnożnika posiada również pozycję nadmiaru. Może się tam pojawić jedynka w jedynym przypadku, gdy podczas wykonywania operacji dzielenia powstaje przekroczenie zakresu maszyny.



W odróżnieniu od pozycji N akumulatora, pozycja nadmiaru mnożnika nie jest przedłużeniem rejestru mnożnika. Jest ona omijana przy zapisie liczby podwójnej długości w rejestrach A i M oraz przy operacjach przesuwania. Jedynie przy przesuwaniu A i M w prawo bit z pozycji nadmiaru mnożnika wchodzi na pozycje rejestru mnożnika i sumuje się logicznie z bitem, który został przesunięty z ostatniej pozycji akumulatora. Jednocześnie wpisuje się do pozycji nadmiaru mnożnika 0. Przy przesyłaniu liczb z pamięci do mnożnika zawartość pozycji nadmiaru wraz z zawartością całego mnożnika zostaje wyzerowana.



Rys. 6. Akumulator i mnożnik jako jedna całość

2.1.3. Wskaźnik nadmiaru, symbol N

Wskaźnik nadmiaru jest to rejestr jedno-bitowy. Jego zawartość może więc być jedynką lub zerem.

Jedynka w rejestrze N wskazuje na to, że w trakcie wykonywania poprzednich obliczeń wynik conajmniej jednej operacji przekroczył zakres maszyny.

Zero w rejestrze N wskazuje, że przekroczenia zakresu maszyny nie było. Przekroczenie zakresu maszyny możliwe jest przy wykonywaniu arytmometrze operacji dodawania, odejmowania, dzielenia i przesuwania w lewo. Właściwość zerowania rejestrów N posiada wyłącznie rozkaz "Skocz przy Nadmiarze" /patrz niżej opis rozkazów maszyny XYZ/.

2.2. Rejestry urządzenia sterującego

Funkcją urządzenia sterującego jest pobieranie z pamięci wewnętrznej maszyny rozkazów i zgodnie z ich treścią wysyłanie odpowiednich impulsów sterujących do pozostałych części maszyny.

Urządzenie sterujące steruje więc pracą całej maszyny zgodnie z informacjami zawartymi w rozkazach programu.

Podstawowymi rejestrami urządzenia sterującego są:

rejestr rozkazów /R/

licznik rozkazów /LR/

2.2.1. Rejestr rozkazów, symbol R

Rejestr rozkazów jest to rejestr 17-bitowy, w którym zostaje umieszczony aktualnie wykonywany rozkaz. Znak rozkazu nie ma wpływu na jego treść i nie wchodzi do rejestrów R.

2.2.2. Licznik rozkazów, symbol LR

Licznik rozkazów jest to rejestr 10-bitowy. Jego zawartość wskazuje numer komórki pamięci, z której pobiera się rozkaz do wykonania. Po wykonaniu dowolnego rozkazu zawartość rejestrów LR zmienia się tak, aby wskazać położenie w pamięci wewnętrznej rozkazu, który ma być wykonany jako następny. Najczęściej zmiana ta polega na automatycznym dodaniu do zawartości rejestrów LR jedynki lub - w przypadku rozkazów sterujących - na przesyłaniu do rejestrów LR części adresowej rozkazu poprzedniego. Istnieją poza tym pewne specjalne rozkazy, które powodują inne zmiany zawartości rejestrów LR.

W każdym jednak przypadku o typie zmiany zawartości LR decyduje ostatni rozkaz wykonywany.

3. ARYTMETYKA MASZYNY

Maszyna XYZ jest maszyną stałoprzecinkową. Działania wykonują się w arytmometrze na liczbach 35-bitowych. O umiejscowieniu przecinka w dowolnej liczbie decyduje programista, przy czym decyzja ta wpływa jedynie na sposób programowania obliczeń. W arytmometrze natomiast wszystkie liczby traktowane są jednakowo jako układy bitów /bez informacji o umiejscowieniu przecinka/.

Będziemy mówili o skali binarnej liczby rozumiejąc przez to umiejscowienie przecinka w danej liczbie. Tak więc liczba w skali binarnej n jest to liczba, w której pozycja jedności wypada w n -tym bicie /przecinek po n -tym bicie/.

Arytmometr maszyny XYZ wykonuje następujące działania arytmetyczne:

3.1. Dodawanie, odejmowanie, odejmowanie bezwzględnych wartości

Działania te wykonują się w akumulatorze. Do zawartości akumulatora można dodać /od zawartości akumulatora można odjąć/ dowolną liczbę z pamięci wewnętrznej maszyny, oraz od bezwzględnej wartości, zawartości akumulatora można odjąć bezwzględną wartość liczby z dowolnej komórki pamięci wewnętrznej. Wynik każdego z tych działań zostaje w akumulatorze. Działania powyższe wykonywane są z pełną dokładnością, o ile nie wystąpi przekroczenie zakresu maszyny. Przekroczenie zakresu maszyny /które jest sygnalizowane za pośrednictwem wskaźnika nadmiaru/ może wystąpić przy operacji dodawania lub odejmowania, natomiast nie może wystąpić przy odejmowaniu bezwzględnych wartości. Powyższe operacje mają sens, jeżeli liczby, na których są wykonywane, są w jednakości skal i.

3.2. Mnożenie

Mnożenie polega na tym, że liczba zawarta w mnożniku zostaje pomnożona przez liczbę z pamięci wewnętrznej, a wynik /podwójnej długości/ zostaje umieszczony w rejestrach A i M, które traktowane są w tym

przypadku jako jedna całość. Znak A i znak M stają się równe znakowi iloczynu. Moduł iloczynu powstaje przez formalne wymnożenie ciągu bitów stanowiących moduły czynników i umieszczony w rejestrach A i M w ten sposób, że najniższa pozycja iloczynu trafia do najniższej pozycji rejestru M.

A zatem:

Jeżeli pomnożymy liczbę w skali n przez liczbę w skali m, to iloczyn otrzymamy w skali $n+m$ /pozycja jedności będzie na $n+m$ -tym miejscu, licząc od pierwszego bitu akumulatora/.

W szczególnym przypadku:

Jeżeli pomnożymy dwie liczby w skali 0, to w akumulatorze otrzymamy starszą część iloczynu również w skali 0.

Jeżeli pomnożymy dwie liczby w skali 35 /liczby całkowite/, to w iloczynie pozycja jedności wypadnie na 35 pozycji mnoźnika. Zatem, jeżeli iloczyn nie przekroczy zakresu maszyny, będzie on umieszczony w rejestrze M jako liczba w skali 35.

Przy wykonywaniu operacji mnożenia nie może wystąpić sygnalizacja nadmiaru.

3.3. Dzielenie

Dzielenie polega na tym, że liczba /podwójnej długości/ zawarta w akumulatorze i mnoźniku /znakiem tej liczby jest znak A/ zostaje podzielona przez liczbę z pamięci wewnętrznej. Iloraz zostaje umieszczony w mnoźniku, a reszta w akumulatorze. Znak reszty równa się znakowi dzielnej /tzn. akumulatora przed dzieleniem/.

Dzielenie wykonuje się prawidłowo w przypadku, gdy moduł zawartości akumulatora jest mniejszy od modułu dzielnika /traktując obie te liczby jako liczby w jednakowej skali/. W przeciwnym razie wynik dzielenia jest nieprawidłowy i sygnalizowane jest przekroczenie zakresu maszyny poprzez wskaźnik nadmiaru.*

* Sygnalizacja nadmiaru może nie wystąpić, jeżeli w dzielnej występowała jedynka na pozycji nadmiaru akumulatora.

18

Jeżeli dzielna posiada skalę n /pozycja jedności w n-tym bicie licząc od początku akumulatora/, a dzielnik skalę m , to iloraz będzie miał skalę $n-m$, a reszta będzie w skali $n-35$.*)
W szczególnym przypadku:

Jeżeli dzielimy liczby w skali 0, to wynik otrzymujemy w mnożniku w skali 0.

Jeżeli chcemy podzielić dwie liczby całkowite w skali 35, to dzielną należy umieścić w mnożniku, a w akumulatorze liczbę zero ze znakiem dzielnej /w ten sposób dzielna będzie miała skalę 70/. Iloraz otrzymamy w mnożniku jako liczbę całkowitą w skali 35.

Specjalne właściwości posiada dzielenie przez zero. W wyniku dzielenia przez zero w mnożniku zostaje umieszczona zanegowana zawartość akumulatora, zaś poprzednia zawartość mnożnika zostaje przesunięta do akumulatora. Znak mnożnika jest zgodny ze znakiem ilorazu, a znak akumulatora pozostaje nie zmieniony. W przypadku dzielenia przez zero na ogół /patrz odnośnik na str. 17/ jest sygnalizowany nadmiar oraz powtarza się jedynka w pozycji nadmiaru mnożnika.

3.4. Przesunięcia

W arytmometrze mogą być wykonywane operacje przesuwania zawartości rejestrów A i M, które w tym przypadku traktowane są jako jedna całość. Arytmetycznie operacje te odpowiadają mnożeniu liczby podwójnej długości w A i M przez całkowitą potęgę dwójki. W przypadku przesuwania w prawo /mnożenie przez 2^{-n} / znak mnożnika zrównuje się ze znakiem akumulatora, który zostaje nie zmieniony. Przy przesuwaniu w lewo /mnożenie przez 2^n / znak akumulatora zostaje przyrównany do znaku mnożnika, który zostaje nie zmieniony.

* Skala -n /ujemna/ rozumiana jest w ten sposób, że pierwszy bit liczby traktowany jest jako $n+1$ -szy za przecinkiem.

** Negowanie liczby polega na zastąpieniu jedynek zerami i na odwrot.

3.5. Zaokrąglanie

Operacja zaokrąglania powoduje dodanie jedynki na 35-tej pozycji do akumulatora, jeżeli w 1-szej pozycji mnoźnika była jedynka, oraz wstawienie jedynki na 35-tej pozycji mnoźnika.

3.6. Koniunkcja

Operacja koniunkcji na dwóch bitach daje wynik 1, jeżeli oba te bity są jedynkami, wynik 0 w przeciwnym przypadku.

$$\begin{array}{l} 0 \wedge 0 = 0 \\ 1 \wedge 0 = 0 \\ 0 \wedge 1 = 0 \\ 1 \wedge 1 = 1 \end{array}$$

W maszynie XYZ operacja koniunkcji polega na wykonaniu koniunkcji na odpowiadających sobie bitach mnoźnika i słowa z pamięci wewnętrznej. Wynik umieszczony zostaje w akumulatorze, zawartość mnoźnika nie ulega zmianie. Operacji koniunkcji podlegają również znaki.

3.7. Właściwości zera

W maszynie XYZ liczba zero może występować w dwóch postaciach: ze znakiem plus lub ze znakiem minus.

Poniżej podane wzory informują, jaki znak otrzymuje liczba 0 w wyniku działań dodawania i odejmowania.

$$\begin{array}{ll} (+a) + (-a) = -0 & \\ (-a) + (+a) = +0 & \\ (+a) - (+a) = -0 & a \geq 0 \\ (-a) - (-a) = +0 & \end{array}$$

Ponadto w przypadku działań na samych zerach zachodzą związki następujące:

$$\begin{array}{ll} (+0) + (+0) = +0 & \\ (-0) + (-0) = -0 & \\ (+0) - (-0) = +0 & \\ (-0) - (+0) = -0 & \end{array}$$

Przy odejmowaniu wartości bezwzględnych otrzymujemy wyniki takie, jak przy odejmowaniu liczb dodatnich /posiadających znak plus/.

Można wypowiedzieć ogólną regułę:

Znak wyniku jest zawsze równy znakowi działania /+ lub -/, jaki wystąpi pomiędzy składnikami, gdy w powyższych wzorach zniesiemy nawiasy.

4. BĘBEN MAGNETYCZNY

Pamięć pomocniczą maszyny XYZ stanowi bęben magnetyczny. Jego pojemność wynosi 8192 słów długich. Komórki pamięci bęбnowej, z których każda może zawierać jedno słowo długie, ponumerowane są kolejno od 0 do 8191.

Numer komórki, w której znajduje się pewne słowo na bębnie, nazywa się adresem bębnowym tego słowa. Pamięć bębnowa składa się z 64 ścieżek ponumerowanych od 0 do 63. Adresy bębnowe początków poszczególnych ścieżek są następujące:

$$0, 128, 256, \dots, n \cdot 128, \dots, 63 \cdot 128$$

gdzie n - numer ścieżki.

Można przesyłać słowa długie z bębna do pamięci wewnętrznej i na odwrót. Przesyłanie słów z pamięci wewnętrznej na bęben nazywa się pisaniem na bęben, zaś przesyłanie w przeciwnym kierunku nazywa się czytaniem z bębna.

Pamięć bębnowa zaopatrzona jest w rejestr bębnowy, symbol RB; zawartość jego wskazuje adres bębnowy skąd będzie czytane lub dokąd będzie zapisywane słowo długie.

Tak więc, jeśli chcemy zapisać coś na bębnie lub przeczytać jakieś słowo z bębna, należy najpierw do rejestru bębnowego przesłać odpowiedni adres bębnowy, a następnie dać rozkaz czytania względnie pisania.

Zawartość rejestru bębnowego po przeczytaniu jednego słowa z bębna lub po zapisaniu jednego słowa na bębnie automatycznie zwiększa się o 1. Jeżeli zatem chcemy przeczytać z bębna grupę słów umieszczonych w kolejnych komórkach pamięci, wystarczy raz załadować rejestr bębnowy adresem początkowym, a następnie wykonać odpowiednią ilość rozkazów czytania z bębna.

Przesyłanie adresu do rejestru bębnowego dokonuje się przy pomocy rozkazu, PRZYGOTUJ CZYTANIE /jeżeli mamy czytać z bębna/ lub rozkazu PRZYGOTUJ PISANIE /jeżeli mamy pisać na bębnie/. Nie można poprzedzać rozkazu CZYTAJ z bębna rozkazem PRZYGOTUJ PISANIE oraz rozkazu PISZ na bęben rozkazem PRZYGOTUJ CZYTANIE.

Wszelkie przesyłania z bębna do pamięci wewnętrznej podlegają tzw. kontroli parity. Kontrola ta polega na tym, że przy zapisywaniu słów długich na bębnie każde słowo zostaje zaopatrzone w dodatkowy bit w ten sposób, żeby suma bitów każdego słowa na bębnie była parzysta. Przy przesyłaniu z bębna do pamięci wewnętrznej specjalny układ kontrolny kontroluje parzystość sumy bitów każdego słowa przesyłanego. W razie wykrycia nieparzystości maszyna staje, zapalając odpowiedni wskaźnik na stoliku operatora.

5. URZĄDZENIA WEJŚCIA I WYJŚCIA

Urządzenia wejścia i wyjścia pośredniczą we wprowadzeniu i wyprowadzeniu informacji do /z/ pamięci wewnętrznej maszyny.

Maszyna XYZ zaopatrzona jest w dwa rodzaje urządzeń wejściowych i trzy rodzaje urządzeń wyjściowych.

5.1. Wejście

5.1.1. Taśma dziurkowana

Zasadniczym rodzajem wejścia do maszyny XYZ jest wejście na taśmie dziurkowanej. Informacje zapisane na taśmie są przekazywane do pamięci wewnętrznej maszyny za pośrednictwem czytnika firmy FERRANTI.

Urządzenie to pozwala na czytanie taśmy z szybkością 300 r z a d - k ó w na sekundę. W jednym rzędzie taśmy znajduje się układ do 5-ciu dziurek. Taki układ dziurek przedstawia jeden z dopuszczalnych znaków dalekopisowych, których spis podany jest w uzupełnieniach na str. 87.

Znaki dalekopisowe podzielone są na dwie grupy: cyfry i LITERY. O tym, czy dany rząddek na taśmie przedstawia literę czy cyfrę, decyduje fakt, czy wśród poprzedzających go rządków występował ostatnio znak CYFRY czy znak LITERY.

Znak CYFRY powoduje, że wszystkie rządki występujące po nim aż do znaku LITERY przedstawiają znaki cyfrowe. Podobnie znak LITERY powoduje, że wszystkie rządki aż do znaku CYFRY przedstawiają znaki literowe.

Specjalny rozkaz czytania rządka taśmy powoduje przepisanie rządka taśmy do 6-ciu najmniej znaczących pozycji słowa krótkiego, wpisując na 6-tą od końca pozycję 0, jeżeli rządtek przedstawia cyfrę, i 1, jeżeli rządtek przedstawia literę. Rządki przedstawiające znaki: CYFRY, LITERY, BŁĄD nie są wczytywane do pamięci /są pomijane/.

Wiersz taśmy - jest to ciąg rządków, zawartych między dwoma kolejnymi znakami PK /Powrót Karetki/ z ostatnim znakiem PK włącznie.

Urządzenie czytające taśmę jest tak skonstruowane, że wiersz taśmy musi być czytany z pełną szybkością 300 rządków na sekundę. Przerwanie czytania taśmy może nastąpić dopiero po przeczytaniu rządka PK kończącego wiersz taśmy.

Do przygotowywania taśmy dziurkowanej używa się specjalnych urządzeń dalekopisowych firmy CREED, przy pomocy których można dziurkować taśmę, jednocześnie otrzymując tabulogram.

5.1.2. Wejście na kartach

Wejście na kartach dziurkowanych oparte jest na reproducerze firmy BULL. Używa się do niego kart systemu 80-kolumnowego. Słowa zapisywane /dziurkowane/ są binarnie w 12 wierszach karty; w każdym wierszu zapisuje się jedno słowo 36-bitowe w kolumnach 1 - 36, licząc od lewej. Razem więc na karcie możemy zapisać 12 słów długich i 24 słowa krótkie. Kolumna 37 może zawierać otwory zwane znakami końca pliku.

Specjalny rozkaz czytania karty powoduje przesłanie informacji, zawartych w jednym wierszu karty, do komórki pamięci wewnętrznej. Jeżeli w czytanym wierszu karty występuje znak końca pliku, maszyna po przeczy-

taniu tego wiersza przeskoczy 4 rozkazy /LR = LR + 5/.

Rozkaz czytania uruchamia drogę czytania reproducera, wskutek czego kolejna karta zaczyna się przesuwać pod s z c z o t k a m i czytającymi. Pod szczotkami pojawiają się kolejno wiersze 9, 8, 7, ..., 0, 11, 12, przy czym karta przesuwa się z ustaloną szybkością bez zatrzymania.

W związku z tym w odpowiednich okresach czasu między pojawieniem się kolejnych wierszy karty pod szczotkami reproducera muszą się pojawiać w sterowaniu rozkazy CZYTANIA z kart. Okresy te /między kolejnymi wierszami karty/ wynoszą około 12 ms. Po odczytaniu 12-tego wiersza karty, automatycznie zaczyna się przesuwać pod szczotkami czytającymi następną kartą.

Jeżeli wiersz 12-tej karty nie zostanie odczytany drogą czytania, reproducer zatrzyma się. Szybkość czytania kart przy pomocy reproducera wynosi 2 karty na sekundę.

5.2. Wyjście

Maszyna XYZ posiada 3 rodzaje urządzeń wyjścia:

- wyjście na taśmie perforowanej,
- wyjście poprzez drukarkę dalekopisową R.F.T.,
- wyjście na kartach dziurkowanych.

5.2.1. Wyjście na taśmie

Wyjście za pośrednictwem taśmy perforowanej jest zasadniczym rodzajem wyjścia w maszynie XYZ. Urządzeniem wyjściowym jest tutaj REPERFORATOR CREEDA. Dziurkuje on taśmę z maksymalną szybkością 25 rządów na sekundę. Tak wydziurkowana taśma wyjściowa przechodzi do urządzenia drukującego, które działa z szybkością 5,6 znaków na sekundę. Drukowanie informacji zawartych na taśmie jest niezależne od maszyny i może być wykonywane w innym miejscu i czasie.

Specjalne podprogramy standartowe służą do przeliczenia liczb względnie rozkazów z systemu binarnego na ciągi znaków dalekopisowych odpowiadające ich zapisowi dziesiętnemu.

5.2.2. Wyjście na drukarkę R.F.T.

Specjalny rozkaz maszyny umożliwia drukowanie danych wyjściowych bezpośrednio z maszyny. Używa się do tego celu międzynarodowego kodu telegraficznego.

5.2.3. Wyjście na kartach

Wyjście na kartach dziurkowanych dokonuje się poprzez reproducer. Specjalny rozkaz maszyny powoduje wydziurkowanie słowa z pamięci wewnętrznej maszyny w jednym wierszu karty. W celu wydziurkowania jednej karty potrzeba 12 takich rozkazów, które - podobnie jak w przypadku czytania - winny być wykonane w odpowiednim czasie jeden po drugim, aby nadążyć za kartą poruszającą się na tzw. drodze pisania w reproducerze. Czas pomiędzy dwoma rozkazami dziurkującymi kolejne wiersze karty nie powinien przekraczać 10 ms.

Droga pisania reproducera zostaje zatrzymana, jeżeli w odpowiednim czasie nie przyjdzie rozkaz powodujący dziurkowanie 12-tego wiersza karty. Jest to jedyny sposób powodujący zaprzestanie podawania kart do urządzenia dziurkującego.

ROZDZIAŁ II

L I S T A R O Z K A Z Ó W M A S Z Y N Y X Y Z - 1

OZNACZENIA

- A, M, R, LR, KL, RB, N - symbole rejestrów maszyny lub ich wartości
- AM - register podwójnej długości, złożony z pozycji 0 - 35 rejestru A, pozycji 1 - 35 rejestru M oraz S_nA
- n - numer miejsca pamięci wewnętrznej
- b - numer miejsca pamięci bębnowej
- t - rządtek taśmy
- w - wiersz karty
- p - parametr rozkazu /podawany w części adresowej/
- (x), (n), (b), itp - zawartość rejestru lub odpowiedniej komórki pamięci
- p_nX - n-ta pozycja rejestru X
- 1_n - słowo utworzone z samych zer i jedynki na n-tej pozycji
- Znak równości "=" - we wzorach opisujących funkcje rozkazów oznacza operacje podstawiania: zawartość rejestru występującego z lewej strony znaku równości ma być przyrównana do wartości prawej strony
- Kropka " ." - przed numerem miejsca pamięci w części adresowej rozkazu oznacza, że adres dotyczy słowa długiego
- Numer operacji - liczba odpowiadająca binarnemu kodowi części operacyjnej rozkazu .

Maszyna XYZ jest maszyną jednoadresową. A więc w trakcie pracy maszyny rozkazy wykonują się w zasadzie w kolejności, w jakiej są one umieszczone w pamięci wewnętrznej.

W opisie funkcji rozkazu znaczone to jest przy pomocy wzoru

$$\text{LR} = \text{LR} + 1$$

/po wykonaniu rozkazu, weź rozkaz umieszczony w kolejnym miejscu pamięci/. Wyjątek stanowią tu rozkazy sterujące, rozkazy wejścia w pewnych szczególnych okolicznościach, oraz rozkazy występujące po rozkazie PL /patrz opis rozkazu PL/.

Rozkazy maszyny zostały podzielone na grupy w zależności od ich treści*:

Grupa A	-	ROZKAZY dotyczące ARYTMOMETRU
"	S -	" STERUJĄCE
"	W -	" WEJŚCIA I WYJŚCIA
"	B -	" BEBNOWE
"	SP -	" SPECJALNE

ROZKAZY DOTYCZĄCE ARYTMOMETRU

Grupa A

1. Dodaj do akumulatora

Zapis:

DO n

Funkcja: do wykonywania dany liczbowy do akumulatora podprogramem CZA. Funkcja jest umieszczana w kolejnych miejscach pamięci rege-

A = A + (n) ; LR = LR + 1

Ewentualna sygnalizacja nadmiaru.

Numer operacji: 6

Przykłady:

DO.256 : Dodaj liczbę długą z pam. 256

DO 1017 : " " krótką z pam. 1017

Treść:

Do zawartości akumulatora dodaj zawartość n-tego miejsca pamięci wewnętrznej i sumę umieść w akumulatorze.

* patrz Uzupełnienia - Tabela 2, str. 88.

2. Odejmij od akumulatora

Zapis:

OD	n
----	---

Funkcja:

$$A = A - (n) ; \quad LR = LR + 1$$

Ewentualna sygnalizacja nadmiaru.

Numer operacji: 7

Przykłady:

OD.350 : Dodaj liczbę długą z pamięci 350

OD 121 : Odejmij liczbę krótką z pamięci 121

Treść:

Od zawartości akumulatora odejmij zawartość n-tego miejsca pamięci wewnętrznej i różnicę umieść w akumulatorze.

3. Odejmij Bezwzględnie

Zapis:

OB	n
----	---

Funkcja:

$$A = |A| - |(n)| ; \quad LR = LR + 1$$

Nadmiar nie może wystąpić.

Numer operacji: 8

Przykłady:

OB 1002 : Odejmij bezwzgl. liczbę krótką z 1002

OB.35 : Odejmij bezwzgl. liczbę długą z 354

Treść:

Od wartości bezwzględnej zawartości akumulatora odejmij wartość bezwzględnej zawartości n-tego miejsca pamięci wewnętrznej i różnicę umieść w akumulatorze.

4. Mnóż

Zapis:

MN	n
----	---

Funkcja:

$$AM = M \cdot (n) ; \quad LR = LR + 1$$

Nadmiar nie może wystąpić.

Numer operacji: 9

Przykłady:

MN 263 : Pomnóż M przez liczbę krótką z pam. 263

MN.400 : Pomnóż *M przez liczbę długą z pam. 400

Treść:

Wyzeruj akumulator. Pomnóż zawartość mnożnika przez zawartość n-tego miejsca pamięci wewnętrznej. Z 70-ciu bitów pełnego iloczynu, 35 lewych umieść w pozycjach 1 - 35 akumulatora, a 35 prawych w pozycjach 1 - 35 rejestru M. Znaki A i M uczyń równe znakowi iloczynu.

5. Dziel

Zapis:

DZ	n
----	---

Funkcja:

$$M = AM : (n) ; \quad A = Reszta ; \quad LR = LR + 1$$

Nadmiar jest sygnalizowany, gdy zawartość A przed dzieleniem jest $> (n)$, z wyjątkiem przypadku, gdy w pozycji nadmiaru A znajduje się jedynka

Numer operacji: 10

Przykłady:

DZ.108 ; Dziel przez liczbę długą z pamięci 108

DZ 15 : Dziel przez liczbę krótką z pamięci 15

Treść:

Podziel łączną zawartość rejestrów A i M, traktując ją jako jedną

liczbę o znaku równym S_{nA} i module złożonym z pozycji 1 - 35 rejestru A i pozycji 1 - 35 rejestru M przez zawartość n-tego miejsca pamięci wewnętrznej. 35 bitów ilorazu umieść w M, a 35 bitów reszty w A. Znak M = znakowi ilorazu. Znak A równa się znakowi dzielnej / nie ulega zmianie/.

W przypadku, gdy jest sygnalizowany nadmiar, pojawia się jedynka w pozycji nadmiaru rejestru M.

6. Umieść w Akumulatorze

Zapis:

UA	n
----	---

Funkcja:

$$A = (n) ; \quad LR = LR + 1$$

Nadmiar nie może wystąpić

Numer operacji: 11

Przykłady:

UA 1000 : Umieść w A liczbę krótką z pamięci 1000

UA.302 : Umieść w A liczbę długą z pamięci 302

Treść:

Wyzeruj akumulator, a następnie zawartość n-tego miejsca pamięci wewnętrznej prześlij do akumulatora.

7. Umieść w Mnożniku

Zapis:

UM	n
----	---

Funkcja:

$$M = (n) ; \quad LR = LR + 1$$

Nadmiar nie może się pojawić

Numer operacji: 12

Przykłady:

UM.12 : Umieść w M liczbę długą z pamięci 12

UM 101 : Umieść w M liczbę krótką z pamięci 101

Treść:

Wyzeruj mnożnik, a następnie zawartość n-tego miejsca pamięci wewnętrznej prześlij do mnożnika.

8. Pamiętaj Akumulator

Zapis:

PA	n
----	---

Funkcja:

$$(n) = A \quad ; \quad LR = LR + 1$$

Nadmiar nie może się pojawić.

Numer operacji: 13

Przykłady:

PA 133 : Pamiętaj A w pamięci krótkiej 133

PA.8 : Pamiętaj A w pamięci długiej 8

Treść:

Wyzeruj n-te miejsce pamięci wewnętrznej, a następnie zawartość akumulatora prześlij do n-tego miejsca pamięci wewnętrznej.

9. Pamiętaj Mnożnik

Zapis:

PM	n
----	---

Funkcja:

$$(n) = M \quad ; \quad LR = LR + 1$$

Nadmiar nie może się pojawić.

Numer operacji: 14

Przykłady:

PM 1023 : Pamiętaj M w pamięci krótkiej 1023

PM.262 : Pamiętaj M w pamięci długiej 262

Treść:

Wyzeruj n-te miejsce pamięci wewnętrznej, a następnie zawartość mnoźnika prześlij do n-tego miejsca pamięci wewnętrznej.

10. Zaokrąglaj

Zapis:

OK

Funkcja:

$$A = A + SnA \cdot 1_{35} \cdot p_1 M ; \quad p_{35} M = 1 ; \quad LR = LR + 1$$

Ewentualna sygnalizacja nadmiaru.

Numer operacji: 15

Przykład:

OK 0 : Część adresowa nie ma wpływu na treść tego rozkazu.

Treść:

W pozycji 35 rejestru M zapisz 1. Jeżeli w pierwszej pozycji mnożnika jest jedynka, dodaj do modułu akumulatora jedynkę na 35 pozycji.
Znak akumulatora nie ulega zmianie.

11. Przesuń w lewo

Zapis:

LW	p
----	---

Funkcja:

$$AM = AM \cdot 2^P ; \quad SnA = SnM ; \quad LR = LR + 1$$

Ewentualna sygnalizacja nadmiaru

Numer operacji: 16

Przykłady:

LW 10

LW 53

Treść:

Łączną zawartość rejestrów A i M przesuń o p miejsc w lewo.

Znak A uczyń równy znakowi M.

12. Przesuń w prawo

Zapis:

PW	p
----	---

Funkcja:

$$AM = AM \cdot 2^{-P} ; \quad SnM = SnA ; \quad LR = LR + 1$$

Nadmiar nie może się pojawić

Numer operacji: 17

Przykłady:

PW 17

PW 35

Treść:

Lączna zawartość rejestrów A i M przesuń o p miejsc w prawo
Znak M uczyń równy znakowi A.

13. Koniunkcja

Zapis:

KO	n
----	---

Funkcja:

$$A = M \cap (n) ; \quad SnA = SnM \cap Sn n ; \quad LR = LR + 1$$

Nadmiar nie może się pojawić.

Numer operacji: 31

Przykłady:

KO 400

KO 993

Treść:

Dokonaj koniunkcji słowa zawartego w mnożniku ze słowem zawartym w n-tym miejscu pamięci. Wyzeruj akumulator i wynik koniunkcji umieść w akumulatorze. Znak akumulatora jest wynikiem koniunkcji znaku M i znaku słowa z n-tej pamięci. Zawartość mnożnika nie ulega zmianie.

ROZKAZY STERUJĄCE

G r u p a S

1. Stop i Skocz

Zapis:

SS	n
----	---

Funkcja:

Numer operacji: 0

STOP ; LR = n

Przykłady:

SS 992

SS 0

Treść:

Zatrzymaj maszynę. Po naciśnięciu guzika START przejdź do wykonywania rozkazu z n-tej komórki pamięci wewnętrznej.

2. Wykonaj Rozkaz

Zapis:

WR	n
----	---

Funkcja:

R = (n)

Numer operacji: 1

Przykłady:

WR 15

WR 1010

Treść:

Wykonaj rozkaz zawarty w n-tej komórce pamięci wewnętrznej. Po wykonaniu tego rozkazu przejdź do wykonywania rozkazu następnego za rozkazem WR.

3. Skocz

Zapis:

SK	n
----	---

Funkcja:

LR = n

Numer operacji: 2

Przykłady:

SK 300

SK 45

Treść:

Przejdź do wykonywania rozkazu zawartego w n-tej komórce pamięci wewnętrznej.

4. Skocz przy Plusie

Zapis:

SP	n
----	---

Funkcja:

Jeżeli $(SnA) = 0$ to $LR = n$
 jeśli $(SnA) = 1$ to $LR = LR + 1$

Numer operacji: 3

Przykłady:

SP 18
 SP 127

Treść:

Jeżeli SnA zawiera 1, przejdź do wykonywania rozkazu następnego.
 W przeciwnym przypadku przejdź do wykonywania rozkazu zawartego w n-tym miejscu pamięci wewnętrznej.

5. Skocz przy Zerze

Zapis:

SZ	n
----	---

Funkcja:

Jeżeli $A = \pm 0$ to $LR = n$
 jeśli $A \neq 0$ to $LR = LR + 1$

Numer operacji: 4

Przykłady:

SZ 15
 SZ 83

Treść:

Jeżeli pozycje 1-35 akumulatora są zerami, przejdź do wykonywania rozkazu umieszczonego w n-tym miejscu pamięci wewnętrznej. W przeciwnym przypadku przejdź do wykonywania rozkazu umieszczonego w kolejnym miejscu pamięci. Zawartość pozycji nadmiaru jak i znak akumulatora nie sąbrane pod uwagę przy wykonywaniu rozkazu SZ.

6. Skocz przy Nadmiarze

Zapis:

SN	n
----	---

Funkcja:

jeżeli $N = 1$ to $LR = n$ i $N = 0$
 jeżeli $N = 0$ to $LR = LR + 1$

Numer operacji: 5

Przykłady:

SN 100
 SN 415

Treść:

Jeżeli zawartość rejestru N jest jedynką, umieść zero w rejestrze N i przejdź do wykonywania rozkazu umieszczonego w n-tym miejscu pamięci wewnętrznej. W przeciwnym przypadku przejdź do wykonywania rozkazu umieszczonego w kolejnym miejscu pamięci.

ROZKAZY WEJŚCIA I WYJŚCIA

Grupa W

1. Czytaj z Taśmy

Zapis:

CT	n
----	---

Funkcja:

jeżeli $t \neq PK$ to $(n) = t + 112 \cdot (z)$; $LR = LR + 1$
 jeżeli $t = PK$ to $(n) = t + 112 \cdot (z)$; $LR = LR + 9$

Numer operacji: 18

Przykłady:

CT 15
 CT 348

Treść:

Przepisz najbliższy rządek taśmy, nie będący znakiem CYFRY, LITERY lub BŁĄD, do pięciu najmniej znaczących pozycji n-tego krótkiego miejsca pamięci wewnętrznej, wstawiając na szóstą od końca pozycję zawartość wskaźnika Z; n-te miejsce pamięci zostało uprzednio wyzerowane. Znaki CYFRY i LITERY nie są wczytywane do pamięci, ale powodują

znak CYFRY - wpisanie zera do wskaźnika Z

znak LITERY - wpisanie jedynki do wskaźnika Z.

Po wczytaniu znaku POWRÓT KARETKI /PK/ maszyna przeskakuje przez 8 kolejnych rozkazów i przechodzi do wykonania dziewiątego.

Po wczytaniu każdego innego znaku maszyna przechodzi do wykonania rozkazu umieszczonego w kolejnym miejscu pamięci.

Znak BŁĄD nie jest wczytywany do maszyny.

2. Pisz na Taśmie

Zapis:

PT	n
----	---

Funkcja:

$$t = (n) ; \quad LR = LR + 1$$

Numer operacji: 19

Przykłady:

PT 35

PT 1020

Treść:

Zawartość 5 najmniej znaczących pozycji n-tego słowa krótkiego z pamięci wewnętrznej wydziurkuj na taśmie. Przesuń taśmę w reperforatorze o 1 rządek.

3. Czytaj z Karty

Zapis:

CK	n
----	---

Funkcja:

(n) = (w) ; LR = LR + 1 jeżeli wiersz nie posiada znaku KP /koniec pliku/
 (n) = (w) ; LR = LR + 5 jeżeli wiersz posiada znak KP.

Numer operacji: 20

Przykłady:

CK.300 : wiersz karty przepisz do długiego słowa 300

CK 110 : prawą część wiersza przepisz do krótkiego
słowa 110

Treść:

Włącz mechanizm podający karty na drodze czytania reproducera i prześlij zawartość pierwszego wiersza, jaki pojawi się pod szczotkami czytającymi, do n-tego miejsca pamięci wewnętrznej. Przy przesyłaniu wiersza karty do słowa długiego odpowiedniość między kolumnami na karcie i pozycjami słowa jest następująca:

Kolumna 1 - pozycja 35
" 2 - " 34
.....
" 36 - " 0

a przy przesyłaniu do słowa krótkiego:

Kolumna 19 - pozycja 17
" 20 - " 16
.....
36 - " 0

Jeżeli czytany wiersz posiada znak końca pliku, maszyna przeskakuje 4 kolejne rozkazy i przechodzi do wykonania piątego.

Jeżeli znaku końca pliku nie ma, maszyna przechodzi do wykonania rozkazu umieszczonego w kolejnym miejscu pamięci.

4. Pisz na Kartę

Zapis:

PK	n
----	---

Funkcja:

$W = (n)$; $LR = LR + 1$; $KP = 0$

Numer operacji: 21

Przykłady:

PK.232 : Wydziurkuj zawartość 232-go długiego miejsca pamięci

PK 1011 : Wydziurkuj zawartość 1011-go krótkiego miejsca pamięci.

Treść:

Włącz mechanizm podający karty na drodze dziurkowania reproducera. Wydziurkuj zawartość n-tego miejsca pamięci wewnętrznej w najbliższym wierszu karty oraz wydziurkuj w tym samym wierszu koniec pliku, jeżeli był zapalony wskaźnik KP. Zgaś wskaźnik KP.

Odpowiedniość między pozycjami słowa i kolumnami karty: jak w przypadku czytania /patrz opis rozkazu CK/.

5. Pisz Koniec Pliku

Zapis:

KP

Funkcja:

$KP = 1$;

$LR = LR + 1$

Numer operacji: 23

Przykład: KP

Treść:

Zapal wskaźnik KP. Przy wykonywaniu najbliższego rozkazu PK zostanie wydziurkowany Koniec Pliku /patrz opis rozkazu PK/.

6. Drukuj na Dalekopisie

Zapis:

DD

p.

Funkcja:

DRUKUJ p ; LR = LR + 1

Numer operacji: 22

Przykłady:

- DD 37 : wydrukuj znak odpowiadający 37 w kodzie R.F.T.
- DD 58 : wydrukuj znak odpowiadający 58 w kodzie R.F.T.
- DD 10 : Nic nie rób.

Treść:

Jeżeli $P \geq 32$, wydrukuj poprzez dalekopis R.F.T. znak odpowiadający liczbie p w Międzynarodowym Kodzie Dalekopisów. W przeciwnym przypadku rozkaz DD jest rozkazem nieefektywnym. Przez p rozumie my tutaj liczbę zawartą w 6 ostatnich bitach części adresowej rozkazu. Stan pozostałych bitów części adresowej nie ma wpływu na treść rozkazu DD.

7. Czytaj Klucze

Zapis:

KL	n
----	---

Funkcja:

(n) = KL ; LR = LR + 1

Numer operacji: 30

Przykłady:

KL 256

KL 421

Treść:

Prześlij zawartość rejestru klucz y do n-tego miejsca pamięci wewnętrznej. /Klucze po lewej stronie stanowią pozycje bardziej znaczące/. Jeżeli część adresowa rozkazu KL wskazuje pamięć krótką, to informacje pobierane są z lewej części rejestru klucz y.

ROZKAZY BĘBNOWE

Grupa B

1. Przygotuj Pisanie na bęben

Zapis:

PP	n
----	---

Funkcja:

$$RB = (n) ; \quad LR = LR + 1$$

Numer operacji: 26

Przykład: PP 348

Treść:

Prześlij słowo krótkie zawarte w n-tym miejscu pamięci do rejestru bębnowego. Przygotuj bęben do zapisywania informacji przesyłanych z pamięci wewnętrznej.

2. Pisz na Bęben

Zapis:

PB	n
----	---

Funkcja:

$$(b) = (n) ; \quad RB = RB + 1 ; \quad LR = LR + 1 , \quad \text{gdzie } b = (RB)$$

Numer operacji: 27

Przykłady:

PB 418

PB 715

Treść:

Zawartość n-tego miejsca pamięci wewnętrznej prześlij do b-tego miejsca pamięci bębnowej, gdzie b jest zawartością rejestru RB. Zawartość rejestru RB zwiększą o 1. Przy przesyłaniu na bęben poprzednia zawartość odpowiedniej komórki pamięci bębnowej zostaje zniszczona.

Przy przesyłaniu słowa krótkiego na bęben słowo to zostaje umieszczone na pozycji 0 - 17 odpowiedniego słowa na bębnie.

Reszta słowa uzupełniona jest zerami.

3. Przygotuj Czytanie z bębna

Zapis:

PC	n
----	---

Funkcja:

$$RB = (n), \quad LR = LR + 1$$

Numer operacji: 29

Przykład: PC 417

Treść:

Prześlij słowo krótkie zawarte w n-tym miejscu pamięci wewnętrznej do rejestru bębnowego. Przygotuj bęben do czytania informacji i przesyłania ich do pamięci wewnętrznej.

4. Czytaj z Bębna.

Zapis:

CB	n
----	---

Funkcja:

$$(n) = (b); RB = RB + 1; LR = LR + 1; \text{ gdzie } b = RB$$

Numer operacji: 28

Przykłady:

CB 256

CB 653

Treść:

Zawartość b-tego miejsca pamięci bębowej, gdzie b jest zawartością RB, prześlij do n-tego miejsca pamięci wewnętrznej. Jeżeli część adresowa rozkazu CB wskazuje słowo krótkie w pamięci wewnętrznej, to zostają tam przesłane bity 0 - 17 odpowiedniego słowa na bębnie.

ROZKAZY SPECJALNE

Grupa SP

1. Nic nie rób

Zapis:

NI

Funkcja:

LR = LR + 1

Numer operacji: 25

Przykład:

OK : Część adresowa nie ma wpływu na treść rozkazu

Treść:

Przejdz do wykonywania rozkazu umieszczonego w kolejnym miejscu pamięci.

Część adresowa rozkazu OK nie ma wpływu na jego działanie.

2. Przygotuj Licznik

Zapis:

PL

Funkcja:

LR = LR + 1 ; Zmiana reżimu pracy maszyny.

Numer operacji: 24.

Część adresowa rozkazu nie ma wpływu na treść rozkazu.

Treść:

Rozkaz PL powoduje zmianę reżimu pracy maszyny: poczynając od rozkazu następnego za rozkazem PL, po wykonaniu każdego rozkazu, po którym normalnie zawartość LR zwiększała się o 1, - zawartość LR zwiększa się o 9 /wykonuje się co 9-ty rozkaz/.

Działanie skoków warunkowych przy spełnionym warunku oraz skoku bezwarunkowego nie ulega zmianie.

Po rozkazie CK, gdy w' czytanym wierszu występuje koniec pliku, zawartość LR zwiększa się o 13.

Po rozkazie CT, gdy czytanym rzędkiem jest rządtek PK, zawartość LR zwiększa się o 17 i jednocześnie maszyna wraca do normalnego reżimu pracy.

Powrót do normalnego reżimu pracy maszyny następuje w dwóch przypadkach.

- a/ po wykonaniu rozkazu SZ przy zerowej zawartości akumulatora,
- b/ po wykonaniu rozkazu CT, gdy czytanym rzędkiem taśmy był rządtek PK.

C z e s c d r u g a

ZEWNĘTRZNE SYSTEMY PROGRAMOWANIA

Opracowano opisy dwóch zewnętrznych systemów programowania dla maszyny XYZ-1

SYSTEM ADRESÓW BEZWZGLĘDNYCH, SAB

SYSTEM ADRESÓW SYMBOLICZNYCH, SAS

Systemy takie pozwalają programistie pisanie programów zgodnie z zasadami i symboliką przyjętą w danym systemie zewnętrznym. Tak napisany program na ogół różni się znacznie od programu napisanego językiem maszyny /czyli w systemie binarnym/. Program napisany w języku systemu zewnętrznego musi więc być przetłumaczony na język maszyny. Ponadto program ten musi być wprowadzony i umieszczony w odpowiednich miejscach pamięci maszyny.

Te dwie czynności tzn.

1. tłumaczenie z języka zewnętrznego na język maszyny
2. wprowadzanie programu do pamięci

wykonuje specjalny dla każdego systemu zewnętrznego program, zwany translatorem danego systemu.

Po napisaniu programu w jednym z systemów SAB lub SAS programista ogranicza się do wydziurkowania tego programu na taśmie dziurkowanej. Wydziurkowana taśma zostaje dołączona do urządzenia czytającego i z chwilą, gdy odpowiedni translator znajduje się w pamięci maszyny, reszta czynności, łącznie z przejściem do wykonania napisanego programu, wykonuje się automatycznie po uruchomieniu maszyny.

Program napisany w jednym z powyższych systemów składa się z rozdziałów. Rozdziały są to części programu, mieszczące się w pamięci wewnętrznej. Translator tłumaczy kolejne rozdziały programu na język maszyny i odsyła je na bęben. Przy końcu translacji wszystkie rozdziały programu umieszczone są na bębnie. Następnie zostaje sprowadzony do pamięci wewnętrznej rozdział 0 /pierwszy z rozdziałów wprowadzonych do maszyny/ i zaczyna się wykonywać program. Przejście od wykonywania jednego rozdziału do innego jest już funkcją samego programu.

ROZDZIAŁ I

S Y S T E M A D R E S Ó W B E Z W Z G L È D N Y C H S A B

1. OGÓLNE ZASADY PISANIA PROGRAMÓW W SAB

Ogólne zasady pisania programów w SAB można ująć w następujących punktach:

- a. W jednym wierszu można napisać jeden rozkaz, liczbę lub dyrektywę.
- b. Z lewej strony rozkazów lub liczb pisze się numer pamięci szybkiej, w której mają być umieszczone.
- c. Wielkość odstępów /tzn. ilość spacji/ między kolejnymi znakami oraz położenie pierwszego znaku w wierszu są nieistotne. System pisania jest zatem n i e p o z y c y j n y: decyduje jedynie kolejność znaków w wierszu, a nie odstępy między nimi.

Poniżej podane będą szczegółowe zasady pisania rozkazów, liczb i dyrektyw oraz ich znaczenie w języku SAB.

2. ROZKAZY

Przykłady:

SK	420
DO.	1006
.UA.	800
.SZ	20

Znaczenie: Szczegółowe znaczenie poszczególnych rozkazów podane zostało w opisie rozkazów maszyny /Część 1, Rozdz. II/.

Forma:

Część operacyjna w postaci dwuliterowego skrótu.
 Część adresowa w postaci liczby dziesiętnej - podaje adres rzeczywisty w pamięci szybkiej. Jeżeli pierwszym znakiem adresu jest kropka, oznacza to, że adres dotyczy słowa dłużego. Kropka przed częścią operacyjną oznacza, że rozkaz posiada znak minus /w pozycji zerowej słowa jest jedynka/.

3. LICZBY

Pierwszym znakiem każdej liczby jest znak plus (+) lub minus (-).

Zapis w systemie dziesiętnym.

Kropka oddziela część całkowitą od części ułamkowej liczby.

3.1. Liczby całkowite krótkie

Przykłady:

$$\begin{array}{r} +17 \\ -854 \\ +234556 \end{array}$$

Znaczenie: Są to liczby zapisane w maszynie po przetłumaczeniu w słowach krótkich tak, że pozycja jedności wypada w bicie 17-tym /przecinek po 17-tym bicie/.

Forma:

Zapis dziesiętny. Istotny jest brak kropki.

Uwaga:

Największą możliwą liczbą całkowitą krótką jest liczba:

$$2^{17} - 1 = 131\ 071$$

3.2. Liczby całkowite długie

Przykłady:

$$\begin{array}{r} +71D \\ -18D \\ -278496789D \end{array}$$

Znaczenie:

Są to liczby zapisane w maszynie po przetłumaczeniu w słowach długich tak, że pozycja jedności wypada w bicie 35-tym /skala 35/.

Forma:

Zapis dziesiętny. Brak kropki - istotny.

Litera D za ostatnią cyfrą.

Uwaga:

Największą możliwą liczbą całkowitą długą jest liczba

$$2^{35} - 1 = 34359738367$$

3.3. Liczby ułamkowe długie**Przykłady:**

+7435 . 847	:	skala binarna	14
+7435 . 847,18	:	" "	18
-008 . 29	:	" "	10
-8 . 29,11	:	" "	11

Znaczenie:

Liczba ułamkowa jest to liczba składająca się z części całkowitej i ułamkowej. Zostaje zapisana w słowie długim. Skalę binarną ustala się w maszynie na podstawie położenia kropki dziesiętnej wg tabelki 1 lub zgodnie ze specjalną informacją podaną w postaci liczby całkowitej umieszczonej za przecinkiem na końcu liczby.

Tabela 1												
Kropka dziesiętna po cyfrze:	0	1	2	3	4	5	6	7	8	9	10	11
Skala binarna	0	4	7	10	14	17	20	24	27	30	34	35

Forma:

Kropka oddziela część całkowitą od ułamkowej. Ewentualna informacja dotycząca skali binarnej podana jest na końcu za przecinkiem. Mówiąc o niej, po którym miejscu binarnym ma być przecinek. Ilość pozycji dziesiętnych za kropką nie może przekraczać 10-ciu.

Uwaga:

Część całkowita nie może być większa niż

$$2^{35} - 1 = 34\ 359\ 738\ 367$$

3.4. Liczby ułamkowe krótkie

Przykłady:

+ 358 . 256 K	:	Skala binarna	10
+ 358 . 256,15 K	:	" "	15
- 07 . 4 K	:	" "	7
- 7 . 4,3 K	:	" "	3

Znaczenie:

Jak w 3.3, z tym że liczba zostaje zapisana w słowie krótkim.

Forma:

Zapis jak w 3.3, z tym że za ostatnią cyfrą liczby występuje litera K.

Uwaga: Część całkowita nie może przekraczać liczby

$$2^{17} - 1 = 131\ 071$$

4. DYREKTYWY

Przykłady:

ROZ	0
ROZ	12
STA	350
FUN	7200

Znaczenie:

Są to instrukcje dla programu translatora SAB.

W pewnym sensie kierują jego działaniem. Występują w programie zapisanym w języku SAB, ale nie występują wewnątrz maszyny, w programie przetłumaczonym.

Forma:

Część operacyjna podana jest w postaci skrótu trójliterowego.

Część adresowa podaje parametr dyrektywy. Dokładne znaczenie parametrów podane będzie poniżej w szczegółowym opisie dyrektyw.

4.1. Rozdział N

ROZ	N
-----	---

Przykłady:

ROZ 17

ROZ 6

Znaczenie:

W wielu przypadkach program jest tak duży, że nie może być w całości umieszczony w pamięci szybkiej maszyny. Wobec tego program dzieli się na części zwane rozdziałami, z których każdy mieści się w pamięci szybkiej. Zwykle rozdział programu stanowi pewną logicznie wyodrębnioną całość.

Każdemu rozdziałowi programista nadaje numer.

Rozdział, który pierwszy jest wprowadzany do maszyny musi mieć numer 0. Numeracja pozostałych rozdziałów - dowolna. Po wprowadzeniu do maszyny rozdziały zostają umieszczone na bębnie skąd mogą być pobierane w razie potrzeby. Jednocześnie na bębnie tworzy się skorowidz rozdziałów. W skorowidzu tym każdemu rozdziałowi odpowiada jedno słowo długie. W N-tym słowie długim znajdują się dane dotyczące N-tego rozdziału.

Na dane te składają się:

- a. Adres bębnowy początku rozdziału /część parzysta słowa długiego/.
- b. Ilość słów długich rozdziału /nieparzysta część słowa długiego/.

Dyrektywa ROZ N sygnalizuje początek rozdziału N.

Forma:

Parametr N wskazujący numer rozdziału może przyjmować wartości całkowite 0, 1, 2, ..., 19.

4.2. Start n

STA	n
-----	---

Przykłady:

STA 0
STA 350

Znaczenie:

Sygnalizuje koniec translacji i przejście do wykonania programu. Wejściem do programu jest rozkaz o numerze n w rozdziale 0. Powoduje sprowadzenie z bębna do pamięci szybkiej rozdziału 0 i skok do n-tego miejsca pamięci szybkiej.

Forma:

Parametr n może być dowolnym adresem w pamięci szybkiej.

4.3. Funkcja b

FUN	b
-----	---

Przykłady:

FUN 6800
FUN 5435

Znaczenie:

Powoduje sprowadzenie z bębna podprogramu, którego metryka jest umieszczona pod adresem bębnowym b, umieszczenie go w kolejnych miejscach formowanego rozdziału i przeadresowanie jego rozkazów zgodnie z położeniem w pamięci szybkiej.

Budowa podprogramu na bieżnie jest następująca:

- Podprogram jest zapisany na bieżnie w kodzie maszyny /w systemie binarnym/.
- Rozkazy zgrupowane są na początku podprogramu. Liczby na końcu.
- Rozkazy o adresach bezwzględnych są dodatnie.
- Adresy rozkazów ujemnych wskazują położenie argumentu rozkazu względem położenia pierwszego rozkazu podprogramu.

Podprogram na bieżnie poprzedzony jest metryką. Metryka podprogramu jest to słowo długie, w którym podane są:

- a/ ilość rozkazów podprogramu /parzysta część słowa/
- b/ ilość słów długich podprogramu /nieparzysta część słowa/.

W przypadku dyrektywy FUN translator wykonuje następujące czynności:

1. Pobranie z bieżna metryki podprogramu /parametr b podaje adres bieżnowy tej metryki/.
2. Na podstawie informacji zawartych w metryce sprowadza do pamięci szybkiej podprogram, przy czym rozkazy ujemne tego podprogramu zostają przeadresowane w ten sposób, że adresy tych rozkazów zostają powiększone o numer miejsca pamięci, w jakim zostanie umieszczony pierwszy rozkaz tego podprogramu wewnątrz rozdziału.
3. Przeadresowany podprogram zostaje przesłany na bęben do kolejnych miejsc formowanego rozdziału.

Forma:

b może być dowolnym adresem bieżnowym.

5. NUMERY

Przykłady:

Numery	Rozkazy lub liczby
--------	-----------------------

280/ SK 420

1 DO .716

2 +281.345

4 -27.342K

5 +1600

+2D

400/ FUN 2400

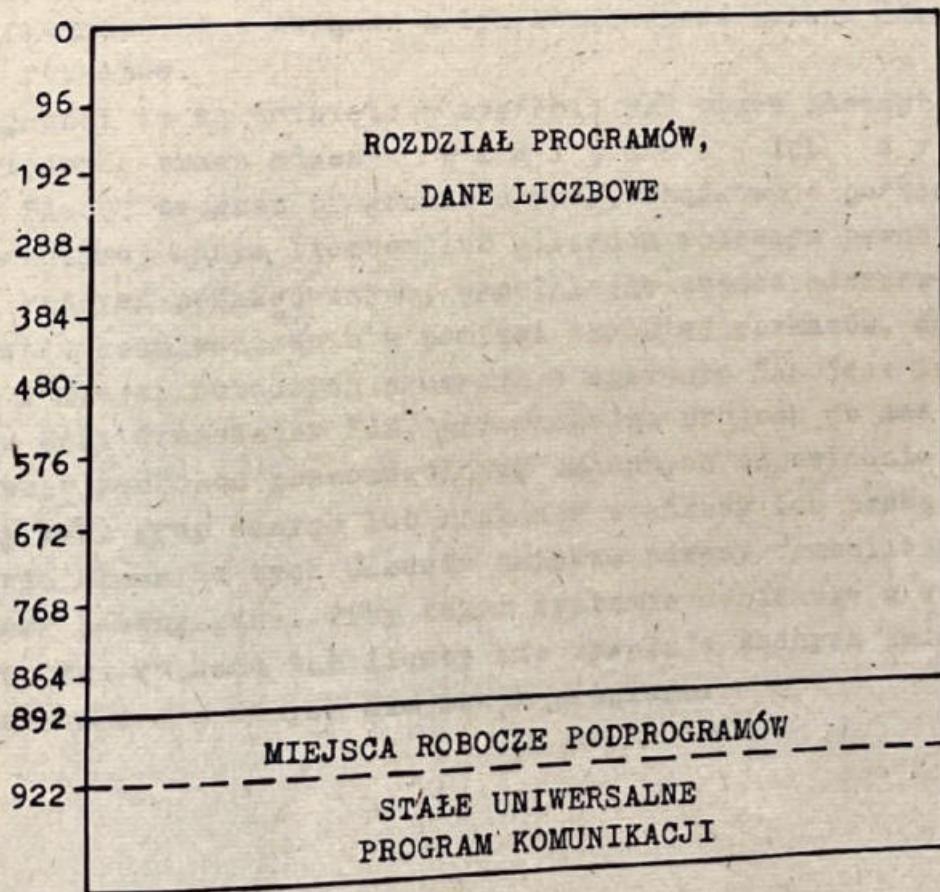
Znaczenie:

Numery wskazują adresy miejsca pamięci szybkiej, w których mają być umieszczone odnośne /napisane na prawo/ rozkazy lub liczby. Numer przy dyrektywie FUN wskazuje miejsce pamięci, w którym ma być umieszczony pierwszy rozkaz podprogramu.

Forma:

Numer pisze się z lewej strony odpowiedniego rozkazu czy liczby w tym samym wierszu. Numer musi być zakończony kreską ułamkową. Jeżeli rozkazy i liczby umieszczamy w kolejnych miejscach pamięci, wówczas tylko pierwszy rozkaz /liczba/ musi być zaopatrzony w numer. Następne rozkazy czy liczby mogą nie posiadać numerów lub dla wygody programisty mogą być zaopatrzone w dowolne numery bez kreski ułamkowej na końcu. Takie numery są pomijane przez translator SAB.

W przypadku braku numeru /lub numeru bez kreski ułamkowej na końcu/ liczby długie automatycznie są umieszczane w kolejnych długich miejscach pamięci. Ewentualne luki między liczbami krótkimi /lub rozkazami/ a liczbami długimi, są wypełniane rozkazami NIC NIE RÓB.

6. PODZIAŁ MIEJSC PAMIĘCI WEWNĘTRZNEJ

Miejsca pamięci od 0 do 891 przeznaczone są na

- rozkazy,
- dane liczbowe i
- miejsca robocze rozdziału programu.

Tą częścią pamięci wewnętrznej dysponuje programista pisząc program.
Miejsca pamięci 892-1023 zawierają:

1. miejsca robocze podprogramów,
2. program komunikacji,
3. stałe uniwersalne i pseudorozkazy.

Na miejsca robocze podprogramów przeznaczone są miejsca pamięci 892-921.

Lista stałych uniwersalnych podana jest na str. 89 w Uzupełnieniach
Wśród stałych uniwersalnych oprócz pewnych często używanych liczb i
pseudorozkazów występują pewne specjalne pseudorozkazy programu ko-
munikacji.

ROZDZIAŁ II

S Y S T E M A D R E S Ó W S Y M B O L I C Z N Y C H S A S

System SAS usuwa cały szereg niedogodności przy pisaniu programu, jakie posiadał system SAB.

Zasadniczą wadą systemu SAB jest konieczność używania adresów bezwzględnych. Programista musi znać dokładne rozmieszczenie w pamięci szybkich wszystkich danych liczbowych i rozkazów programu i posługuje się w programie - zamiast nazwami liczb lub części programu - numerami pamięci, w których są one umieszczone. W takim systemie niezwykle pracochłonnym jest korygowanie błędów w programie. Jeżeli np. błąd polega na opuszczeniu jednego rozkazu w początkowej części programu, to poprawienie tego błędu wywołuje zmianę rozmieszczenia większości rozkazów programu, a w związku z tym konieczność zmiany adresów dużej ilości rozkazów.

Trudności te są omijane w systemie SAS przez zastąpienie adresów bezwzględnych przez adresy względne lub symboliczne. Pisząc program programista przyporządkowuje poszczególnym partiom programu, danym liczbom lub miejscom roboczym pewne symbole, których później używa, wypełniając części adresowe rozkazów.

Kwestia rozmieszczenia w pamięci szybkiej rozkazów, danych liczbowych i miejsc roboczych programu w systemie SAS jest funkcją translatora SAS. Translator SAS, wprowadzając program do maszyny, przyporządkowuje symbolom poszczególnych zmiennych odpowiednie adresy /w przypadku grup danych lub rozkazów - adresy ich początków/ i na podstawie słownika tych adresów zmienia adresy symboliczne rozkazów na adresy bezwzględne. Przy takim systemie dopisanie w dowolnym miejscu programu rozkazu lub liczby nie wywołuje żadnych zmian - lub bardzo niewielkie - w innych miejscach programu.

1. BLOKI

Partie programu lub grupy danych liczbowych nazywać będziemy często blokami rozkazów lub liczb. Blokiem będziemy nazywali grupę liczb lub rozkazów umieszczonych w kolejnych miejscach pamięci maszyny. Każdemu blokowi przyporządkowany zostaje symbol stanowiący jego nazwę.

Będziemy rozróżniały bloki liczbowe /złożone z liczb/ i bloki rozkazowe /złożone z rozkazów/.

Adres początku bloku jest to adres miejsca pamięci, w którym umieszczone jest pierwsze słowo bloku.

Kolejne słowa bloku są ponumerowane poczynając od zera.

Numery elementów bloku nazywa się ich indeksami.

W systemie SAS przyjęta jest zasada, że kolejne słowa bloku są umieszczane w kolejnych miejscach pamięci. Zatem słowo bloku o indeksie 0, tzn. początek bloku ma adres najmniejszy. Przyjmując powyższą terminologię można powiedzieć, że w systemie SAS programista przyporządkowuje symbole blokom /liczbowym lub rozkazowym/ występującym w programie.

2. SYMBOLE

Symbole, jakie nadaje programista poszczególnym blokom występującym w programie, mogą się składać z co najwyżej 4-ch znaków dalekopisowych.

Obowiązują przy tym podane niżej zasady.

2.1. Symbole zmiennych liczbowych

Przykłady:

A1, BETA, PI, R02,
ROB1, ROB2, OMEG, C17

Znaczenie:

Pod pojęciem zmiennych l i c z b o w y c h rozumiemy: bloki danych liczbowych, pojedyncze liczby^{*)}, zmienne robocze, dla których translator SAS rezerwuje miejsca pamięci wewnętrznej maszyny. Wprowadzenie odpowiednich liczb na te miejsca jest już zadaniem programu.

Forma:

Symbole zmiennych liczbowych zaczynają się od liter y.

2.2. Numery symboliczne**Przykłady:**

11, 2, 2A, 1WYI, 3WEJ

170, 1SUM, 3CZY

Znaczenie:

Numery symboliczne są to symbole bloków rozkazowych, stanowiących partie programu lub poszczególnych miejsc w programie.

Forma:

Symbole numerów symbolicznych zaczynają się od cyfry.

2.3. Paragrafy**Przykłady:**

*TRY, *WIE, *10, *BOR

Znaczenie:

W systemie SAS rozdział składa się z paragrafów. Paragrafy charakteryzują się tym, że mają niezależną symbolikę: ten sam symbol w dwóch różnych paragrafach oznacza co innego. Paragrafy są zwykle pewne zamknięte części rozdziału pełniące funkcje podprogramów.

Forma:

Symbole paragrafów zaczynają się od gwiazdki /*.

^{*)} Pojedyncze liczby można traktować jako bloki jednoelementowe.

3. ROZMIESZCZANIE INFORMACJI W PAMIĘCI WEWNĘTRZNEJ

Rozmieszczenia rozkazów i danych liczbowych rozdziału w pamięci wewnętrznej oraz zamiany adresów symbolicznych rozkazów na adresy bezwzględne dokonuje translator SAS.

Zasada rozmieszczania informacji w pamięci wewnętrznej jest następująca:

- a. Rozkazy oraz liczby programu umieszcza się w kolejnych miejscach pamięci poczynając od zerowego lub tak, jak wskazują numery bezwzględne programu /patrz 6./.
- b. Miejsca na zmienne liczbowe rezerwowane są w kolejności występowania ich symboli w programie, poczynając od 892-go miejsca pamięci wewnętrznej wstecz.

W przypadku, gdy okaże się, że rozkazy rozdziału oraz zmienne liczbowe nie mieszczą się razem w pamięci wewnętrznej, translator daje odpowiedni sygnał błędu.

4. ADRESY ROZKAZÓW

4.1. Adresy bezwzględne

Przykłady:

DO 1005	:	Dodaj do A liczbę krótką z pamięci 1005
UA .256	:	Umieść w A liczbę dużą z pamięci 256
.SK 127	:	
.OD 242	:	Rozkazy ujemne

Znaczenie:

Adresy bezwzględne wskazują numery miejsca pamięci szybkiej. Adresy te nie ulegają zmianie przy translacji.

Forma:

Adresy bezwzględne pisze się tak jak w SAB.

4.2. Adresy względne

Przykłady:

UM	+2
UM	.+1
.UA	+0
SK	-1
.DO	.-2

Znaczenie:

Wskazuję położenie argumentu rozkazu względem położenia samego rozkazu. Znak adresu wskazuje czy argument rozkazu jest umieszczony przed /minus (-)/ czy po /plus (+)/ rozkazie zawierającym adres. Wartość bezwzględna adresu wskazuje "odległość" argumentu rozkazu od rozkazu, mierzoną w słowach krótkich.

Forma:

W zapisie adresy względne charakteryzują się tym, że zaczynają się od znaku (+) lub (-).

Wartość bezwzględna adresu może być dowolną liczbą całkowitą mniejszą od 1024.

4.3. Adresy symboliczne

Przykłady:

DO	.SA3(1)
UA	MR21
SK	13(1)
DO	.21A)
SK	*SIN
SZ	*WIE(4)
SK	1(-1)

Znaczenie:

Adresy symboliczne określają argument rozkazu przez podanie symbolu bloku /rozkazowego lub liczbowego/ oraz indeksu. Indeks w tym przypadku wskazuje względne położenie argumentu rozkazu względem początku danego bloku. Znak indeksu wskazuje czy argument położony jest w pamięci przed /minus/ czy za /plus/ początkiem bloku. Wartość bezwzględna indeksu wskazuje "odległość" argumentu od po-

czętku bloku mierzoną, w przypadku bloków rozkazowych /numerów, paragrafów/, w słowach krótkich.

W przypadku bloków zmiennych liczbowych odległość ta mierzona jest w słowach krótkich, gdy adres dotyczy słowa krótkiego /brak kropki/ i w słowach długich, gdy adres dotyczy słowa długiego /kropka przed symbolem bloku/.

Forma:

I n d e k s pisze się za symbolem bloku w n a w i a s a c h.

Znaku plus /+/ przy indeksie można nie pisać. Indeks 0 łącznie z nawiasami może być pominięty.

Numer symboliczny w adresie symbolicznym, jeżeli nie ma indeksu, musi być zakończony nawiasem zamkającym.

5. LICZBY

Przykłady:

+285.643	Liczba ułamkowa - długa
+94.785K	" " - krótka
+1960	" całkowita - krótka
-16D	" " - długa

Znaczenie i forma zapisu liczb w języku SAS są takie same, jak w języku SAB.

6. SZABLONY

Przykłady:

=03777000177	Szablon długi
=776037	Szablon krótki

Znaczenie:

S z a b l o n y są to układy bitów, zapisane w postaci słów /długości lub krótkich/, służące najczęściej do "wycinania" części słowa umieszczonego w rejestrze M przy pomocy rozkazu koniunkcji.

Forma:

W SAS szablony zapisuje się w systemie ósemkowym:

12 cyfr ósemkowych w przypadku szablonu długiego

6 " " " " " krótkiego.

Przed pierwszą cyfrą szablonu stawia się znak tożsamości $/=$.

7. NUMERY BEZWZGŁĘDNE**Przykłady:**

Numer	Rozkaz
100/UA .ALG(7)	
DZ	IK
180/SK	21
OB	+3
500/LOC.BETA	

Znaczenie:

Numery bezwzględne są odpowiednikami numerów w języku SAB. Wskazują one adres bezwzględny w pamięci wewnętrznej, gdzie ma być umieszczony rozkaz lub liczba napisane po prawej stronie numeru. Dalsze rozkazy i liczby są umieszczane w kolejnych miejscach pamięci aż do pojawienia się następnego numeru bezwzględnego.

Forma:

Numery bezwzględne pisze się z lewej strony odpowiednich rozkazów programu. Numer zakończony jest kreską ułamkową (/) tak jak w SAB.

8. NUMERY SYMBOLICZNE**Przykłady:**

Numer	Rozkaz
-------	--------

Znaczenie:

Numer y s y m b o l i c z n e są to symbole poszczególnych partií programu /bloków rozkazowych/. W programie pisze się je z lewej strony rozkazu stanowiącego poczatek odpowiedniego bloku rozkazów.

Forma:

Numer symboliczny musi być oddzielony od rozkazu nawiasem zamykającym.

Jeżeli za numerem symbolicznym wystąpią dwa nawiasy zamykające, to rozkaz stanowiący początek odpowiedniego bloku umieszczony będzie w parzystym miejscu pamięci. W ewentualnie wolne miejsce, jakie powstanie przed tym rozkazem, wstawiony będzie rozkaz NIC NIE RÓB. Numerem może być dowolny układ 4-ch znaków dalekopisowych, z których pierwszy jest cyfrą.

Uwaga:

Dopuszczalne jest pisanie kilku numerów symbolicznych przy jednym rozkazie. Wtedy odpowiedni blok rozkazowy otrzymuje kilka nazw, których znaczenie jest takie samo.

Przy jednym rozkazie można również napisać numer bezwzględny, a z nim jeden lub kilka numerów symbolicznych. W ten sposób odpowiedni blok rozkazowy będzie się zaczynał w miejscu pamięci wskazanym przez numer bezwzględny i będzie posiadał nazwy określone przez numery symboliczne.

Przykład:

```
1AR)7KOS)UA 512
          DO 1000
          PA 512
342/3KOL)OD 1022
```

9. PARAGRAFY**Przykłady:**

```
*TRY)LOC.A
      LOC.XY
*IPA)DO 12
      PA 5
```

Znaczenie:

Paragraf jest to część rozdziału, która charakteryzuje się tym, że ma własną /niezależną od reszty rozdziału/ symbolikę. Te same symbole w różnych paragrafach odnoszą się do różnych liczb lub grup rozkazów. Paragrafy stanowią zazwyczaj podprogramy programu głównego, który jest umieszczony na początku rozdziału.

Początek paragrafu oznacza programista - podobnie jak w przypadku numerów symbolicznych - pisząc z lewej strony pierwszego rozkazu paragrafu jego symbol oddzielony od rozkazu nawiasem zamykającym.

Pierwszy rozkaz paragrafu jest automatycznie umieszczony w parzystym miejscu pamięci wewnętrznej.

Forma:

Symbol paragrafu jest symbolem co najwyżej 4-znakowym, z których pierwszy musi być gwiazdką /*/. Zazwyczaj jest to symbol 4-znakowy, w którym 3 ostatnie znaki są trójliterowym skrótem nazwy paragrafu /nazwa pochodzi od funkcji, którą wykonuje/.

10. DYREKTYWY

Przykłady:

ROZ 6
LOC.GAMA
TAB OM13
TEK

Znaczenie:

Podobnie jak w przypadku języka SAB dyrektywy są to instrukcje dla programu translatora SAS. W języku SAS dyrektywy jest znacznie więcej niż w SAB. Wynika to stąd, że translator SAS ma znacznie bardziej złożone funkcje niż translator SAB.

Forma:

Trójliterowy skrót nazwy + parametr.
Szczegółowy opis formy wystąpi przy omawianiu poszczególnych dyrektyw.

10.1. Rozdział N

ROZ

N

Przykład:

ROZ 3

Znaczenie i forma jak w SAB.10.2. Start

STA

Znaczenie:

Dyrektyna STA w języku SAS ma takie samo znaczenie jak dyrektywa STA O w języku SAB.

Forma:

Różni się od formy w SAB tym, że dyrektywa ta występuje tu bez parametru.

10.3. Funkcja b

FUN

b

Przykład:

FUN 6688

Znaczenie i forma jak w SAB.10.4. Wymiar

WYM zmieniona (n)

Przykłady:

WYM A (5)

WYM.BETA (18)

Znaczenie:

Podaje wymiar bloku zmiennych liczbowych; liczba elementów bloku jest n.

Powoduje zarezerwowanie dla bloku n miejsce pamięci szybkiej /długiach lub krótkich w zależności od rodzaju zmiennej/.

Forma:

Zmienną może określać dowolny symbol. zmiennej liczbowej. Jeżeli zmienna oznacza wektor liczb długich, to jej symbol w dyrektywie WYM musi być poprzedzony kropką. n wskazuje liczbę składowych zmiennej-wektora i może być liczbą naturalną <1024.

10.5 Locum

LOC	zmienna
-----	---------

Przykłady:

LOC.ALEF

LOC KL

Znaczenie:

Wskazuje położenie zmiennej w programie. Miejsce dla zmiennej zostaje zarezerwowane między rozkazami programu w miejscu, gdzie występuje dyrektywa LOC. Jeżeli potrzeba zarezerwować dla zmiennej długie słowo, to przed symbolem zmiennej musi być postawiona kropka.

Forma:

Zmienna może określać dowolny symbol zmiennej liczbowej.

Uwaga 1:

Dyrektyna LOC znajduje główne zastosowanie przy pisaniu podprogramów stanowiących oddzielne paragrafy. Dyrektywa LOC używa się w celu zarezerwowania i "nazwania" w podprogramie miejsc, do których Program Główny przesyła argumenty podprogramu. Gdy argumentem podprogramu jest wektor, zamiast całego wektora przesyła się do miejsca zarezerwowanego przez odpowiedni LOC adres początku tego wektora. W takim przypadku, mimo że LOC dotyczy zmiennej odnoszącej się do słów długich, nie należy pisać przed nazwą zmiennej w LOC kropki.

Uwaga 2:

Zmienna, dla której rezerwuje miejsce dyrektywa LOC, nie może występować w rozkazach poprzedzających dyrektywę LOC.

10.6. Tablica

TAB	zmienna (n)
-----	-------------

Przykład:

```
+TAB.EPSI (15) : Powoduje zarezerwowanie 15 długich
+0.0007      : miejsc pamięci na blok EPSI i wprowa-
+0.0014      : dzenie do tego bloku liczb napisanych poniżej /do gwiazdki/
+0.0021      :
=777000777000
*
```

Znaczenie:

Powoduje zarezerwowanie w programie n miejsc pamięci dla bloku zmiennej liczbowej, której symbol podany jest w adresie dyrektywy. Początek bloku będzie w miejscu programu, gdzie występuje dyrektywa TAB.

Liczby lub szablony napisane za dyrektywą TAB aż do wiersza z gwiazdką zostają wprowadzone do kolejnych miejsc bloku. Jeżeli nie wyczerpują one pojemności bloku, to reszta miejsc zostaje uzupełniona zerami.

Jeżeli przed symbolem zmiennej w adresie dyrektywy TAB występuje kropka, to zostaje zarezerwowanych n miejsc długich; w przeciwnym przypadku n miejsc krótkich.

Forma:

Nazwą zmiennej może być dowolny symbol zmiennej liczbowej. Liczby względnie szablony, które stanowią "tablicę" /są elementami bloku, dla którego rezerwuje miejsce dyrektywa TAB/, pisze się pod dyrektywą TAB po jednej w wierszu; za ostatnią liczbą wzgl. szablonem w następnym wierszu musi być umieszczona gwiazdka.

10.7. Tekst

TEK

Przykład:

TEK

ROZWIĄZANIA UKŁADU SORI

Znaczenie:

Sygnalizuje znaki tekstu, które mają być przepisane do programu bez zmian /w jednym słowie krótkim - jeden znak dalekopisowy/. Znaki tekstu w programie przetłumaczonym są poprzedzone przez dwa krótkie miejsca pamięci, które zawierają kolejno: ilość znaków tekstu i pseudorozkaz DO 1024.

Forma:

Tekst pisze się tak, aby znajdował się w następnym wierszu po rozkazie TEK i cały mieścił się w tym wierszu.

Początek tekstu liczy się od pierwszego znaku, który nie jest spacja /SP/, i kończy się na ostatnim takim znaku.

11. KOMENTARZ**Przykład:**

UA + O : ŁĄCZNIK DO PROGRAMU SPACJA
SK * SPA

Znaczenie:

K o m e n t a r z e m nazywamy dowolny tekst objaśniający program. Tekst komentarza jest pomijany przez translator.

Forma:

Każdy komentarz zaczyna się od dwukropka /:/ . Wszystkie znaki występujące w dowolnym wierszu programu za dwukropkiem należą do komentarza. Koncem komentarza jest koniec wiersza /a ścisłe znak PK - powrót karetki/.

ROZDZIAŁ III

PODPROGRAMY BĘBNOWE PROGRAM KOMUNIKACJI

1. PODPROGRAMY BĘBNOWE

Typowe obliczenia często bywają opracowywane w postaci podprogramów bibliotecznych przechowywanych na kartach lub taśmie wydzielarkowej w systemie binarnym /ew. oktalnym/. System SAS umożliwia łatwe korzystanie z takich podprogramów, pod warunkiem że zostaną one uprzednio nagrane na bęben.

Podprogram na bębnie /podprogram bębnowy/ musi być przy tym zbudowany zgodnie z zasadami podanymi przy opisie dyrektywy FUN /Rozdział I, § 3.3/.

W celu włączenia podprogramu bębnowego do rozdziału programu pisze się w tym rozdziale dyrektywę

FUN b

gdzie b jest adresem bębnowym metryki tego podprogramu.

Zwykle podprogramom bębnowym nadaje się nazwy paragrafów /powstałe przez dopisanie gwiazdki przed trójliterowym skrótem nazwy podprogramu/. Nazwę taką pisze się z lewej strony odpowiedniej dyrektywy FUN. W ten sposób podprogram staje się samodzielnym paragrafem. Pozwala to odwoływać się do tego samego podprogramu w różnych paragrafach jednego rozdziału.

Dyrektyny FUN dla podprogramów bębnowych, użytych w danym rozdziale, najwygodniej jest pisać na końcu tego rozdziału.

Niektóre często używane podprogramy zostały nagrane na bęben "na stałe", tzn. umieszczone w stałej części pamięci bębnowej przeznaczonej dla translatora SAS.

Są to:

- a/ Podprogramy obliczające wartości najczęściej używanych funkcji np.: $\sin x$, \sqrt{x} itp. - tzw. funkcje bębnowe.
- b/ Podprogramy Wejścia-Wyjścia służące do wprowadzania oraz wyprowadzania danych liczbowych przez program.

Omówimy te dwie grupy kolejno.

1.1. Funkcje bębnowe

W stałej części pamięci bębnowej maszyny XYZ umieszczone są podprogramy następujących funkcji:

Tablica 4

Lp.	Funkcja	Symbol w SAS
1	$\sin x$	SIN
2	$\cos x$	COS
3	$\operatorname{tg} x$	TG
4	$\operatorname{arc sin} x$	ASN
5	$\operatorname{arc cos} x$	ACS
6	$\operatorname{arc tg} x$	ATG
7	$\operatorname{arcus}(x,y)$	ARC
8	\sqrt{x}	PWK
9	$\sqrt[3]{x}$	PWS
10	$\ln x$	LN
11	e^x	EXP
12	x^y	POT

1.1.1. Układ podprogramów funkcji na bębnie

1.1.1.1. Grupy podprogramów

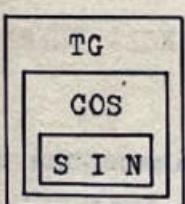
Podprogramy funkcji bębnowych podzielone są na pewne grupy, których struktura wyjaśniona będzie poniżej przy pomocy tablicy przedstawiającej schematy tych grup. Podział podprogramów funkcji na grupy spowodowany był tym, że przyjęte algorytmy niektórych funkcji zawierają algo-

rytmy innych funkcji. Względy oszczędności pamięci kazały dołączyć podprogramy takich funkcji tak, aby podprogramy pierwszych zawierały w sobie podprogramy drugich.

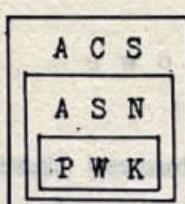
GRUPY PODPROGRAMÓW FUNKCJI

Tablica 5

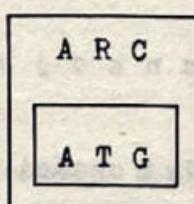
Grupa 1



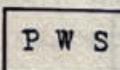
Grupa 2



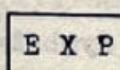
Grupa 3



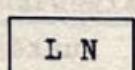
Grupa 4



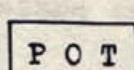
Grupa 5



Grupa 6



Grupa 7



Schematy grup podprogramów przedstawione w tablicy należy rozumieć w sposób następujący:

- a/ Podprogram funkcji, której symbol wypisany jest w danym prostokącie, zawiera w sobie podprogramy funkcji, których symbole wypisane są w prostokątach zawartych w tym prostokącie.
- b/ Każdej nazwie funkcji odpowiada pewna sekwencja rozkazów i liczb. Podprogram funkcji, której symbol wypisany jest w danym prostokącie, jest sumą sekwencji i oznaczonych symbolami funkcji wypisanymi we wszystkich prostokątach zawartych w tym prostokącie.

A zatem:

Podprogram obliczający wartości funkcji cosinus /Tablica 5, grupa 1/ składa się z sekwencji rozkazów oznaczonych odpowiednio przez COS i SIN.

Podprogram obliczający wartości funkcji arc cos x składa się z sekwencji rozkazów oznaczonych przez ACS, ASN i PWK.

Podprogramy funkcji sin x, \sqrt{x} , arc tg x, $\sqrt[3]{x}$, e^x , $\ln x$, x^y są podprogramami samodzielnymi.

1.1.1.2. Rozmieszczenie podprogramów na bębnie

Podprogramy zapamiętane są na bębnie z zachowaniem podziału na grupy. Sekwencje umieszczone są w kolejności zgodnie z budową grup /jak wskazuje Tablica 6/. Jeżeli w pewnej sekwencji występują liczby, to są one zgrupowane na końcu.

Każda sekwencja poprzedzona jest metryką.

Metryka jest to słowo długie, w którym podane są:

- a/ ilość rozkazów sekwencji /parzysta część słowa długiego/,
- b/ ilość słów długich sekwencji /nieparzysta część słowa długiego/.

UKŁAD PODPROGRAMÓW NA BĘBNIE

Tablica 6

Adres bębnowy metryki	Układ na bębnie	Objaśnienia
b_1	_____	← Metryka sekwencji TG Sekwencja TG
b_2	_____	← Metryka sekwencji COS Sekwencja COS
b_3	_____	← Metryka sekwencji SIN Sekwencja SIN
b_4	_____	← Metryka sekwencji ACS Sekwencja ACS
b_5	_____	← Metryka sekwencji ASN Sekwencja ASN itd. kolejno grupami

Przykłady:

1. Sprowadzenie do pamięci wewnętrznej podprogramu tg x.
Z układu grupy 1 /patrz Tablica 5/ wynika, że do pamięci wewnętrznej należy sprowadzić sekwencje rozkazów oznaczone przez TG, COS i SIN. Zatem translator kolejno wg metryki o numerach b_1 , b_2 i b_3 sprowadza odpowiednie fragmenty tego podprogramu. Jasne jest, że można wówczas korzystać z podprogramu cos x lub sin x.
2. Sprowadzenie do pamięci wewnętrznej podprogramu sin x.
Z budowy grupy 1 wynika, że do pamięci wewnętrznej trzeba sprowadzić fragment oznaczony przez SIN, tzn. translator sprowadza z bębna sekwencję rozkazów według metryki o numerze b_3 .

1.1.2. Sposób korzystania z podprogramów funkcji

W celu obliczenia wartości którejkolwiek z funkcji wymienionych w Tablicy 1 należy:

- a. Przesłać argumenty do podprogramu.
 - W przypadku funkcji jednoargumentowej - argument, jako słowo długie, przesyła się na pozycję 4-tą danego podprogramu.
 - W przypadku funkcji dwuargumentowej - pierwszy argument /jako słowo długie/ przesyła się na pozycję 4-tą i 5-tą a drugi - na pozycje 6-tą i 7-mą podprogramu.

Uwaga: Przez pozycje podprogramu rozumiemy kolejne słowa krótkie tego podprogramu ponumerowane od zera.

- b. Informację o skali binarnej argumentów /i wyniku/ umieścić w miejscach 1010 i 1011 pamięci wewnętrznej w postaci rozkazów

nr pam.	rozkaz
1010	PW p
1011	LW p

gdzie p jest skalą binarną.

c. Odwołać się do podprogramu odpowiedniej funkcji przy pomocy łącznika:

UA +0
SK *XXX

gdzie w miejsca XXX w drugim rozkazie łącznika należy wstawić odpowiedni trójliterowy Symbol Funkcji /patrz Tablica 1/.

Przed powrotem do programu wywołującego podprogram zostawia obliczoną wartość funkcji w akumulatorze. Skala w wyniku /wartości funkcji/ równa jest Skali Argumentów.

Włączenia podprogramu do rozdziału programu dokonuje dyrektywa FUN /patrz rozdz. 1, pkt. 10.3/

Przykład:

Rozdział odwołujący się do podprogramów EXP i ARC

ROZ 3		
1)) PW 10	}	Skala = 10
LW 10		
UA. 1)		
PA. 1010		
UA. X	}	Podanie argumentu dla podprogramu EXP
PA. *EXP(4)		
UA +0	}	Łącznik do EXP
SK *EXP		
UA. X	}	Podanie argumentów dla ARC
PA. *ARC(4)		
UA. Y		
PA. *ARC(6)		
UA +0	}	Łącznik do ARC
SK *ARC		
*EXP)FUN 6023		
ARC)FUN 6080	}	Włączenie odpowiednich podprogramów do rozdziału)
*ATG)FUN 6084		

*) Adresy metryk w dyrektywach FUN są fikcyjne.

1.2. Podprogramy wejścia i wyjścia

Wprowadzenia do maszyny programu wydzielkowanego na taśmie dokonuje odpowiedni program wprowadzający /translator/. Natomiast wprowadzanie danych liczbowych i wyprowadzanie wyników jest funkcją samego programu.

Wprowadzanie danych przez program nazwiemy **czytaniem**, a wyprowadzanie wyników **drukowaniem**.

Do wczytywania i drukowania informacji przez program służą specjalne podprogramy bębnowe, objęte ogólną nazwą podprogramów wejścia i wyjścia. Podprogramy te będą omówione poniżej.

PODPROGRAMY WEJŚCIA I WYJŚCIA

Tablica 7

Lp.	Podprogram /Nazwa podprogramu/	Symbol w SAS
1.	CZYTANIE	CZY
2.	DRUKUJ	DRU
3.	TEKST	TEK

1.2.1. Wprowadzanie danych liczbowych

Do wczytywania danych liczbowych do maszyny służy podprogram CZYTANIE. Funkcją jego jest wczytanie do żądanych miejsc pamięci wewnętrznej bloku liczb, wydzielkowanych na taśmie aktualnie podłączonej do urządzenia czytającego. Liczby wydzielkowane są w systemie dziesiętnym i zostają przez podprogram CZY przeliczone na system binarny.

1.2.1.1. Przygotowanie danych liczbowych do czytania. Przykład arkusza z danymi liczbowymi

Blok liczb ułamkowych

$X_1 = 23.281$	$X_2 : -18.442$	$Y : 804.17$
28.485	-33.747	98.20
-342.251	8.32	-15.344
H = +0.125		

Blok liczb całkowitych

N :	38	M :	17	P :	-45
	620		-840		2

Forma:

Dane liczbowe pisze się na arkuszach według następujących zasad:

- a/ Liczby wypisane na arkuszu podzielone są na bloki. Jeden blok sprowadza się do maszyny przez jedno odwołanie się do podprogramu Czytaj.
- b/ Każdy blok zaczyna się od nowego wiersza.
- c/ Blok składający się z dwóch lub więcej liczb musi być zakończony wierszem, w którym pierwszym znaczącym znakiem jest gwiazdka (*). Po bloku jednoliczbowym gwiazdki nie pisze się.
- d/ Bloki są jednorodne tzn. składają się z liczb jednego typu, np. same liczby ułamkowe lub same liczby całkowite krótkie itp.
- e/ Wśród liczb mogą występować komentarze; komentarzem może być również cały wiersz. Komentarz zaczyna się od litery i kończy dwukropkiem lub znakiem "=" /Wewnątrz komentarza dwukropki ani znak "=" wystąpić nie może/.
- f/ Liczby pisze się wierszami - ilość liczb w wierszu dowolna. Koniec liczby wskazuje spacja, stąd między cyframi liczby - jak i między znakiem liczby - z samą liczbą spacja wystąpić nie może.
- g/ Część całkowita oddzielona jest od części ułamkowej przy pomocą kropki. Liczba ułamkowa charakteryzuje się tym, że zawiera kropkę. Liczba bez kropki - całkowita. Znak plus /+/ przed liczbą dodatnią może lecz nie musi być pisany.
- h/ W liczbach ułamkowych ilość cyfr po kropce nie może przekraczać 10-ciu.

Liczby zapisane na arkuszu zostają wraz z komentarzami wydziurkowane na taśmie, którą następnie wkłada się do urządzenia czytającego.

1.2.1.2. Korzystanie z podprogramu CZYTAJ

W celu wczytania bloku danych liczbowych z taśmy do maszyny należy odwołać się do podprogramu CZY przy pomocy następującego łącznika:

Lącznik do podprogramu CZY

UA +0

SK *CZY

PA ZMIENNA : ADRES WSKAZUJĄCY : dokąd wprowadzać
m : ilość liczb w bloku

Adres 3-ciego rozkazu powyższego łącznika wskazuje początek bloku, do którego mają być wczytane dane liczbowe z taśmy.

Jeżeli adres tego rozkazu dotyczy słowa długiego, wczytane liczby będą traktowane jako długie; w przeciwnym przypadku - jako krótkie.

Na 4-tym miejscu łącznika podana jest ilość liczb wczytywanych. Informacja dotycząca skali binarnej liczb wczytywanych musi być podana w miejscach pamięci 1010, 1011 w postaci:

1010/PW α

α - skala binarna

1011/LW α

Jeżeli ilość liczb w bloku nie będzie zgadzała się z liczbą m podaną w łączniku, to zostanie zasygnalizowany błąd. /Dotyczy to bloków złożonych z conajmniej 2-oh liczb. W przypadku wczytywania jednej liczby sygnalizacji błędu nie ma/.

Sygnalizacja błędu występuje również w przypadku, gdy liczba wczytywana nie mieści się w zakresie maszyny, lub gdy ilość cyfr po kropce jest większa od 10.

1.2.2. Drukowanie wyników. Wydawnictwo

1.2.2.1. Podprogram DRUKUJ /DRU/

Funkcją podprogramu DRUKUJ jest przeliczenie z systemu binarnego na dziesiętny i wydziurkowanie na taśmie grupy liczb wskazanej przez łącznik.

W celu wydziurkowania na taśmie grupy liczb należy odwołać się do podprogramu DRUKUJ poprzez następujący łącznik

```
JA +0
SK *DRU
UA ZMIENNA /indeks liczbowy/
UA SYMBOL INDEKSU
    c      : ilość cyfr przed kropką
    u      : - " - po kropce
+0 lub +1 : ułamkowe /+0/ czy całkowite /+1/
    m      : ilość liczb
```

Adresy rozkazów 3-go i 4-go łącznika określają adres w pamięci wewnętrznej liczby, która ma być wydrukowana jako pierwsza. Adres ten jest to miejsce określone przez zmienią podaną w adresie rozkazu 3-go oraz indeks, który powstaje ze zsumowania wartości symbolu podanego w adresie 4-go rozkazu łącznika i indeksu liczbowego z adresu 3-go rozkazu.

Następne liczby wypisuje się z kolejnych miejsc pamięci wewnętrznej.

W następnych wierszach łącznika podaje się kolejno następujące informacje:

- c - ilość cyfr przed kropką,
- u - ilość cyfr po kropce,
- informacja dotycząca rodzaju liczb: ułamkowe /+0/ całkowite /+1/,
- m - ilość liczb.

Obowiązują następujące ograniczenia dotyczące parametrów podprogramu DRUKUJ:

- 1/ u ≤ 10
- 2/ c+u ≤ 21

Informacja o skali binarnej liczb drukowanych brana jest z miejsc pamięci 1010 i 1011, gdzie powinno być:

1010/PW α	α - skala binarna
1011/LW α	

Rodzaj adresu 3-go rozkazu łącznika mówi, ozy mają być wyprowadzone liczby długie czy krótkie.

Przykład 1.

Łącznik zaczynający się od rozkazów:

```
UA 0
SK *DRU
UA .BETA(5)
UA K
=====
```

powoduje wyprowadzenie liczby długiej o adresie symbolicznym
BETA ($K + 5$).

Gdy adres początku grupy liczb drukowanych określony jest
przez zmienną z indeksem liczbowym, to na 4-tym miejscu łączni-
ka pisze się rozkaz: UA 1017.

Przykład 2.

Łącznik:

```
UA +0
SK *DRU
UA .BETA(5)
UA 1017
+5
+8
+0
+1
```

powoduje wydrukowanie liczby długiej ułamkowej z miejsca pamię-
ci BETA(5)

W przypadku $m > 1$ liczby drukowane są po trzy w wierszu.

Forma liczb wydrukowanych za pośrednictwem podprogramu DRUKUJ:
LICZBY CAŁKOWITE:

15	724	-21426
----	-----	--------

LICZBY UŁAMKOWE:

+17.3460	-315.7833	+84.0000
----------	-----------	----------

Znak plus przed liczbami całkowitymi nie jest drukowany. Zera na początku są zastępowane spacjami. Znak liczby jest umieszczony tuż przed pierwszą cyfrą wydrukowaną. Za kropką występuje zawsze tyle cyfr, ile zażądano. Kropka oddziela część całkowitą od ułamkowej.

1.2.2.2. Podprogram TEKST /TEK/

Funkcją tego podprogramu jest wydziurkowanie ciągu dowolnych znaków dalekopisowych podanych w łączniku do tego podprogramu. Ciąg ten w języku zewnętrznym poprzedzony jest dyrektywą TEK i nie może przekroczać jednego wiersza.

Łącznik do podprogramu TEKST:

```
UA +0
SK *TEK
TEK
TEKST DO WYDZIURKOWANIA
```

Uwaga:

Spacje na początku i na końcu tekstu są usuwane.

1.2.2.3. Wydawnictwo

Przykład arkusza wydawniczego:
Rozwiązańie układu równań różniczkowych.

X	Y ₁	Y ₂
+0.000	+2.734856	-1.200000
+0.002	+2.989330	-2.000374
+0.004	+3.721488	-1.385420
+0.006	+6.142300	-0.500123
+0.008	+10.297000	+0.101291
+0.010	+12.300839	+1.094835
		itp.

2. PROGRAM KOMUNIKACJI

Ogólną nazwą programu komunikacji objęte są dwa niezależne programy:

PROGRAM WSPÓŁPRACY Z BĘBNEM	PWB
PROGRAM CZYTANIA Z TAŚMY	PCT

Oba te programy są programami stałomiejscowymi i wraz ze stały-mi uniwersalnymi zajmują 102 ostatnie miejsca pamięci wewnętrznej /922-1023/. Sprowadzane są one z bębna jeszcze przed rozpoczęciem pracy translatora i pozostają we wskazanych wyżej miejscach pamięci przez cały czas pracy maszyny.

2.1. Program współpracy z bębnem

Program Współpracy z Bębnem wykonuje 3 następujące czynności:

- a/ Sprowadzanie bloku słów z bębna do pamięci wewnętrznej.
- b/ Przesyłanie bloku słów z pamięci wewnętrznej na bęben.
- c/ Sprowadzenie do pamięci wewnętrznej nowego rozdziału i skok do jego wykonania.

W celu sprowadzenia do pamięci wewnętrznej z bębna bloku m słów należy odwołać się do Programu Współpracy z Bębnem za pomocą następującego łącznika:

```

PC x : x - adres miejsca pamięci wewnętrznej,
           w którym umieszczony jest adres bę-
           nowy /skąd sprowadzać/
UA +0
SK 953
CB Adres: adres w pamięci wewnętrznej
           m : liczba słów

```

Adres 4-go rozkazu tego łącznika podaje początek bloku w pamięci wewnętrznej, dokąd ma być sprowadzony blok m słów z bębna. Jeżeli adres ten dotyczy słowa długiego, to zostanie sprowadzonych m słów długich; w przeciwnym przypadku - m słów krótkich /słowa krótkie pobierane są ze starszych części słów

długich na bębnie i umieszczane w kolejnych słowach krótkich w pamięci wewnętrznej/.

W celu przesłania z pamięci wewnętrznej na bęben bloku m słów należy odwołać się do Programu Współpracy z Bębnem przy pomocy następującego łącznika:

PP x : x - adres miejsca pamięci wewnętrznej,
w którym umieszczony jest adres bę-
bnowy /dokąd przesyłać/

UA +0

SK 953

PB Adres: adres w pamięci szybkiej /skąd prze-
syłać/

m : ilość słów bloku.

Jeżeli adres 4-go rozkazu łącznika dotyczy słowa krótkiego, to zostanie przesłanych na bęben m słów krótkich, z których każde będzie umieszczone w osobnym miejscu pamięci na bębnie.

W celu sprowadzenia do pamięci wewnętrznej N-tego Rozdziału i przejścia do jego wykonania należy odwołać się do programu Współpracy z Bębnem przy pomocy następującego łącznika:

1N)) PC +6

CB.+5

PC +5

UA +0

SK 950

CB. n : n - adres, od którego będzie umiesz-
czony rozdział.

X + N : X + N - adres charakterystyki n-tego
rozdziału w Skorowidzu Roz-
działów na bębnie,

SS 0

X - adres charakterystyki rozdziału 0 w Skorowidzu Rozdzia-
łów na bębnie.

Numer przy pierwszym rozkazie został napisany w celu wska-
zania, że rozkaz ten musi być umieszczony w parzystym miejscu
pamięci.

Rozdział umieszczony jest zwykle poczynając od zero-wego miejsca pamięci wewnętrznej.

Zatem na ogólny $n = 0$.

Po sprowadzeniu rozdziału do pamięci wewnętrznej Program Współpracy z Bębnem wykonuje skok do zerowego miejsca pamięci. Jeżeli chcemy, aby po wprowadzeniu rozdziału wykonany był skok do dowolnego innego miejsca pamięci, należy odpowiedni rozkaz skokowy umieścić w mnożniku, a w powyższym łączniku zastąpić rozkaz: SK 950 na rozkaz: SK 951.

2.2. Program czytania taśmy

Jest to pętla czytająca taśmę. Funkcją tego programu jest wprowadzenie do pamięci wewnętrznej, w miejsca wskazane przez łącznik, ciągu znaków wydziurkowanych na aktualnie podłączonej do urządzenia czytającego taśmie dalekopisowej, kończąc wprowadzanie po wczytaniu rzędu taśmy z kombinacją dziurek, odpowiadającą znakowi PK /Powrót Karetki/. Ciąg rzędów zawartych pomiędzy dwoma znakami PK, z ostatnim znakiem PK włącznie, nazywamy wierszem taśmy.

W celu wprowadzenia do kolejnych miejsc pamięci wewnętrznej poczynając od adresu n znaków wiersza taśmy należy odwołać się do programu Czytania Taśmy /PCT/ przy pomocy następującego łącznika:

UA +0

SK 978

CT n : n - adres, gdzie ma być umieszczony pierwszy znak wiersza.

Kolejne znaki wiersza taśmy zapisywane są w kolejnych słowach krótkich pamięci wewnętrznej w postaci układów 6-ciu bitów na najniższych pozycjach słowa krótkiego.

U z u p e l n i e n i a

Tablica 1

ZNAKI DALEKOPISOWE DLA WEJŚCIA I WYJŚCIA NA TAŚMIE DZIURKOWANEJ

Cyfry	Litery	Taśma	Wartość binarna	Wartość dziesiętna
		5 4 3 2 1		
C y f r y		.	0	0
1	A	• •	1	1
2	B	• • •	10	2
*	C	• • • •	11	3
4	D	• • • • •	100	4
(E	• • • • • •	101	5
)	F	• • • • • • •	110	6
7	G	• • • • • • • •	111	7
8	H	• • • • • • • • •	1000	8
≡	I	• • • • • • • • • •	1001	9
=	J	• • • • • • • • • • •	1010	10
-	K	• • • • • • • • • • • •	1011	11
v	L	• • • • • • • • • • • • •	1100	12
LINIA	M	• • • • • • • • • • • • • •	1101	13
SP (spacja)	N	• • • • • • • • • • • • • • •	1110	14
,	O	• • • • • • • • • • • • • • • •	1111	15
0	P	• • • • • • • • • • • • • • • • •	10000	16
>	Q	• • • • • • • • • • • • • • • • • •	10001	17
:	R	• • • • • • • • • • • • • • • • • • •	10010	18
3	S	• • • • • • • • • • • • • • • • • • •	10011	19
→	T	• • • • • • • • • • • • • • • • • • • •	10100	20
5	U	• • • • • • • • • • • • • • • • • • • •	10101	21
6	V	• • • • • • • • • • • • • • • • • • • •	10110	22
/	W	• • • • • • • • • • • • • • • • • • • •	10111	23
x	X	• • • • • • • • • • • • • • • • • • • •	11000	24
9	Y	• • • • • • • • • • • • • • • • • • • •	11001	25
+	Z	• • • • • • • • • • • • • • • • • • • •	11010	26
L i t e r y		• • • • • • • • • • • • • • • • • • • •	11011	27
•	•	• • • • • • • • • • • • • • • • • • •	11100	28
n	?	• • • • • • • • • • • • • • • • • • • •	11101	29
P.K.	£	• • • • • • • • • • • • • • • • • • • •	11110	30
*/BLAD/	*/BLAD/	• • • • • • • • • • • • • • • • • • • •	11111	31

DWULITEROWE SKRÓTY CZĘŚCI OPERACYJNYCH ROZKAZÓW

Tablica 2

Nr operacji	Skrót dwuliterowy	Nazwa operacji
0	SS	Stop i Skocz
1	WR	Wykonaj Rozkaz
2	SK	Skocz
3	SP	Skocz przy Plusie
4	SZ	Skocz przy Zerze
5	SN	Skocz przy Nadmiarze
6	DO	Dodaj do akumulatora
7	OD	Odejmij od akumulatora
8	OB	Odejmij Bezwzględnie
9	MN	Mnóż
10	DZ	Dziel
11	UA	Umieść w Akumulatorze
12	UM	Umieść w Mnożniku
13	PA	Pamiętaj Akumulator
14	PM	Pamiętaj Mnożnik
15	OK	Zaokrąglij
16	LW	Przesuń w Lewo
17	PW	Przesuń w Prawo
18	CT	Czytaj z Taśmy
19	PT	Pisz na Taśmie
20	CK	Czytaj z Karty
21	PK	Pisz na Kartę
22	DD	Drukuj Dalekopis
23	KP	Pisz Koniec Pliku
24	PL	Przygotuj Licznik
25	NI	Nic nie rób
26	PP	Przygotuj Pisanie na bębnie
27	PB	Pisz na Bębnie
28	CB	Czytaj z Bębna
29	PC	Przygotuj Czytanie z bębna
30	KL	Czytaj Klucze
31	KO	Koniunkcja

Uwaga: Numer operacji odpowiada numerowi operacji rozkazu w kodzie binarnym.

STAŁE UNIWERSALNE

Tablica 3

Numer miejsca pamięci	Stała lub pseudorozkaz	Uwagi
975	SK 0	
976	OB.2044	
987	-0	
988	.KO.2047	
989	KO.2047	
990	PW 0	
991	LW 0	
992	+2047	jedynki w części adresowej + bit 7
993	SS.0	jedynka w pozycji 6 słowa krótkiego
997	KO.0	jedynki w części operacyjnej + bit 6
998	$\frac{\pi}{4}$	
999		
1000	+2	
1001	+2	
1006	+1	
1007	+1	
1010	PW α	α - skala binarna liczb
1011	LW α	
1015	PA 0	
1016	+1	DŁUGA JEDYNKA
1017	+0	
1019	OB.2047	stała wykorzystywana w ŁĄCZNIKU
1020	UA 0	
1023	OB.2046	stała wykorzystywana w ŁĄCZNIKU

INSTRUKCJA

W P R O W A D Z A N I A I U R U C H A M I A N I A
P R O G R A M Ó W

1. PRZYGOTOWANIE TAŚMY Z PROGRAMEM

Napisany program musi być wydziurkowany na taśmie papierowej. Dokonuje się tego za pośrednictwem urządzeń dalekopisowych i dziurkujących taśmy. W trakcie dziurkowania taśmy powstaje jednocześnie tabulogram, który służy jako dokumentacja programu. Tabulogram pozwala również na kontrolę poprawności wydziurkowania taśmy.

2. PODZIAŁ PAMIĘCI BĘBNOWEJ

Miejsca pamięci bębnowej od 3552 zwykle zajęte są przez: Translator SAS, Programy Organizacyjne, Podprogramy Bębnowe i Magazyn Pamięci Wewnętrznej.

Magazyn pamięci wewnętrznej zajmuje na bębnie miejsca 3584-4095. Przechowywana tam jest zawartość pamięci wewnętrznej w trakcie działania pewnych programów organizacyjnych jak: Post Mortem, Wejście Oktalne, Wyjście Oktalne itp. Z tych miejsc mogą korzystać również programy eksploatacyjne.

Miejsca pamięci bębnowej, 3552-3584 zawierają skorowidz rozdziałów.

Programista dysponuje miejscami na bębnie od numeru 0 do $3552 - \Sigma n_i$, gdzie n_i - ilość słów długich i-tego rozdziału programu. Kolejne rozdziały programu umieszczane są bowiem na bębnie jeden za drugim poczynając od adresu bębnowego 3852 wstecz. Pozostałe miejsca przeznaczone są na dane liczbowe programu.

3. WSTĘPNE PRZESŁANIE

Pierwszą czynnością jaką trzeba wykonać przed rozpoczęciem wprowadzania programu jest wstępne przesłanie. Polega ono na sprowadzeniu z bębna do pamięci wewnętrznej programu komunikacji.

Dokonuje się tego przy pomocy specjalnej pętli samoładowającej wydziurkowanej na karcie. Kartę tę wprowadza się za pośrednictwem reproducera, tak jak każdą pętlę samoładowającą. Podczas dokonywania przesłania wstępnego /ze względu na korzystanie z reproducera/ klucze na stoliku operatora muszą być podniesione. Po sprowadzeniu programu komunikacji maszyna stanie na adresie 928. Należy wtedy opuścić klucze i ustawić na nich informacje, dotyczące zamierzonej pracy maszyny /znaczenie poszczególnych kluczy opisane będzie poniżej/.

Po naciśnięciu klucza START maszyna wykona następujące czynności:

1. Przesłanie zawartości pamięci wewnętrznej do magazynu pamięci wewnętrznej na bębnie.
2. Sprowadzenie z bębna programu sterującego I i skok do niego.

Program sterujący czyta klucze i interpretuje ich treść.

4. ZNACZENIE POSZCZEGÓLNYCH KLUCZY

Rejestr kluczy na stoliku operatora podzielony został na dwie części, odpowiadające dwóm słowom krótkim. W prawej części /klucze 0-17/ ustawia się informacje dla programu sterującego I, dotyczące rodzaju pracy maszyny.

Lewą część kluczy /klucze 18-35/ wykorzystuje się w przypadku, gdy jest wprowadzany program w jednym z języków SAS, SAB lub SO /system oktalny/. Wtedy w lewej części kluozy podaje się pewne informacje dla odpowiedniego translatora, mówiące o sposobie wprowadzania lub żądania wyprowadzania rozdziałów po zaadresowaniu.

W celu zbadania zawartości tych kluczy translator odwołuje się do programu sterującego II, który na podstawie informacji zawartych na kluczach steruje dalszą pracą maszyny.

A. ZNACZENIE KLUCZY W CZĘŚCI PRAWEJ

WPROWADZANIE PROGRAMÓW, klucze 0, 1, 2

Klucze 0, 1, 2 - dają informacje, w jakim systemie napisany jest program /wzgl. rozdział programu/, który ma być wprowadzany.

Klucz 0 - wejście w SAS

Taśma z programem jest wczytywana do maszyny. Dokonuje się translacja, adresowanie i umieszczanie programu na bębnie. Po przeczytaniu każdego rozdziału Program Sterujący II bada zawartość lewej części kluczy i steruje dalszą pracą maszyny. Przed rozpoczęciem wczytywania kolejnego rozdziału zostaje sprowadzony Program Sterujący I i maszyna staje; można teraz zmienić ustawienie kluczy /każdy rozdział może być napisany w innym systemie/. Jeżeli podniesiony jest klucz 35, wówczas maszyna nie staje po każdym rozdziale.

Klucz 1 - wejście w SAB

Taśma z programem w SAB jest wczytywana do maszyny. Po przeczytaniu rozdziału Program Sterujący II bada zawartość lewej części kluczy i steruje dalszą pracą maszyny. Przed rozpoczęciem wczytywania kolejnego rozdziału zostaje sprowadzony Program Sterujący I i maszyna staje lub skacze do Programu Sterującego I /jeżeli podniesiony klucz 35/.

Klucz 2 - wejście S0 /System Oktalny/

Program wydziurkowany na taśmie w systemie oktalnym wprowadzony zostaje do maszyny. Każde słowo długie programu zapisane jest w dwunastu rządkach taśmy /cyfry 0,1,2,3,4,5,6,7/.

Co najwyżej po 16 słowach długich musi być wydziurkowany rządek PK. Po każdym rozdziale musi być wydziurkowana litera R z dwuoyfrowym numerem rozdziału następnego. Ostatni rozdział musi być zakończony literą S, która pełni rolę dyrektywy START. Pierwszy rozdział wprowadzany automatycznie otrzymuje numer 0.

Po każdym rozdziale - podobnie jak w przypadkach wejścia w SAS i SAB - sterowanie obejmuje Program Sterujący II a następnie Program Sterujący I.

POST MORTEM, klucze 3-8

Klucze 3-8 dotyczą różnych rodzajów Post Mortem. W przypadku, gdy jest podniesiony jeden z tych kluczy, Program Sterujący I sprowadza z bębna odpowiedni program Post Mortem i staje na wejściu do niego. Wtedy należy ustawić na kluczach adres /ewnętrzny lub bębnowy/ odkąd chcemy wyprowadzać, ilość słów oraz - w przypadku Post Mortem Liczby - skalę binarną liczb.

Po ustawieniu kluczy nacisnąć START.

Po wyprowadzeniu zadanej ilości słów zostaje sprowadzona z bębna zawartość magazynu pamięci wewnętrznej, po czym maszyna staje na 928, umożliwiając przez to dalsze sterowanie maszyną z kluczy.

Ilość słów podaje się w postaci ilości bloków, odpowiadających pojemności jednej rury. Tak więc w przypadku Post Mortem Rozkazy lub Liczby Krótkie podaje się ilość bloków po 32 słowa krótkie, a w przypadku Post Mortem Liczby Długie podaje się ilość bloków po 16 słów długich.

Klucz 3 - Post Mortem Rozkazy Pam. Wewn.

Na kluczach należy ustawić:

1. W części prawej /klucze 0-17/ - Adres w pamięci wewnętrznej.
2. W części lewej /klucze 18-35/ - Ilość bloków po 32 rozkazy.

Klucz 4 - Post Mortem Rozkazy Bęben

Na kluczach należy ustawić:

1. W części prawej /klucze 0-17/ - Adres bębnowy.
2. W części lewej /klucze 18-35/ - Ilość bloków po 32 rozkazy.

Na tabulogramie wypisany zostaje najpierw adres bębnowy podany na kluczach, za nim rozkazy ponumerowane kolejno poczynając od zera.

Klucz 5 - Post Mortem Liczby Długie Pam. Wewn.

Na kluczach należy ustawić:

1. W części prawej - Adres w pamięci wewnętrznej.
2. W części lewej -
 - a. Klucze 18-27 - Ilość bloków po 16 liczb długich.
 - b. Klucze 29-34 - Skala binarna liczb.

Na tabulogramie wypisanych zostaje 11 cyfr przed przecinkiem i 11 cyfr po przecinku. Z lewej strony każdej liczby - jej adres w pamięci wewnętrznej.

Klucz 6 - Post Mortem Liczby Długie Bęben

Na kluczach należy ustawić:

1. W części prawej - Adres bębnowy
2. W części lewej - jak przy klucozu 5.

Z lewej strony każdej liczby wypisany zostaje jej adres bębnowy. Forma zapisu liczby - jak przy klucozu 5.

Klucz 7 - Post Mortem Liczby Krótkie Pam. Wewn.

Na kluczach należy ustawić:

1. W części prawej - Adres w pamięci wewnętrznej.
2. W części lewej -
 - a. Klucze 18-27 - Ilość bloków po 32 liczby krótkie.
 - b. Klucze 29-34 - Skala binarna liczb.

Z lewej strony każdej liczby wypisany zostaje numer komórki pamięci wewnętrznej. Forma zapisu liczb - jak przy klucozu 5.

Klucz 8 - Post Mortem Liczby Krótkie, Bęben

Na kluczach należy ustawić:

1. W części prawej - Adres Bębnowy.
2. W części lewej - jak przy klucozu 7.

Forma zapisu liczb na formularzu - jak przy klucozu 7.

WEJŚCIE OKTALNE, klucze 9, 10

Klucze 9, 10 - służą do sterowania wprowadzaniem informacji wydziurkowanych na taśmie w systemie oktalnym pod zadany adres w pamięci.

W przypadku gdy jest podniesiony jeden z tych klucozy, Program Sterujący I sprowadza z bębna odpowiedni program "Wejście Oktalne" i staje na wejściu do niego. Wtedy należy ustawić na kluczach adres /wewnętrzny lub bębnowy/ dokąd chcemy wprowadzać informacje i nacisnąć klawisz START.

Informacje na taśmie muszą być wydziurkowane w systemie oktalnym tak, że każde słowo długie zapisane jest w 12 rzędach taśmy przy pomocy cyfr 0,1,2,3,4,5,6,7.

Po co najwyżej 16 słowach długich musi być wydziurkowany rządek PK. Rządek PK musi być wydziurkowany również za ostatnim słowem. Po ostatnim słowie dziurkuje się literę K, która sygnalizuje k o - n i e c wprowadzania i powoduje restaurację zawartości pamięci wewnętrznej oraz stop na 928.

Uwaga:

Przy dziurkowaniu taśmy w systemie oktalnym można zamiast cyfr używać znaków dalekopisowych, które mają układ 3 dziurek na najniższych po - zycjach taki sam, jak odpowiednia cyfra. Nie można jedynie używać zna - ków CYFRY, LITERY i BŁĄD, gdyż te nie są wczytywane do maszyny.

Kluoz 9 - Wejście Oktalne do Pamięci Wewnętrznej

Na kluczach w części prawej /klucze 0-17/ należy ustawić a d - r e s, od którego poczynając mają być umieszczane informacje w pa - mięci wewnętrznej. Poza miejscami, do których wprowadzane są informa - cje, zawartość pamięci wewnętrznej pozostaje niezmieniona.

Po przeeczytaniu litery K na końcu, maszyna staje na 928.

Klucz 10 - Wejście Oktalne na Bęben

Na kluczach w części prawej /klucze 0-17/ należy ustawić a d - r e s b e b n o w y, od którego mają być umieszczane informacje na bębnie.

Zamiast ustawać adres na kluczach, można na taśmie - przed infor - macjami, które mają być wprowadzane - wydziurkować literę B, a bez - pośrednio za nią dziesiętny adres bębnowy /nie pisać spacji!/ za - kończony znakiem PK. W przypadku taśmy z literą B na początku ustawienie kluczy nie odgrywa roli.

Zawartość pamięci wewnętrznej po zakończeniu wprowadzania zostaje sprowadzona do stanu przed rozpoczęciem wprowadzania. Maszyna staje na 928.

WYJSCIE OKTALNE, klucze 11, 12

Klucze 11, 12 - służą do sterowania wyprowadzaniem informacji z maszyny w systemie oktalnym, poczynając od podanego adresu pamięci /wewnętrznej lub bębnowej/.

W przypadku, gdy jest podniesiony jeden z tych klucozy, Program Sterujący I sprowadza z bębna odpowiedni program "Wyjście Oktalne" i staje na wejściu do niego. Wtedy należy ustawić na kluozach adres /wewnętrzny lub bębowy/ odkąd chcemy wyprowadzać oraz ilość słów długich, jaka ma być wyprowadzona.

Po ustawieniu klucozy nacisnąć START.

Po wyprowadzeniu żądanej ilości słów długich dokonuje się regeneracja zawartości pamięci wewnętrznej i maszyna staje na 928.

Sposób dziurkowania taśmy przez podprogram Wyjście Oktalne odpowiada opisanemu powyżej /opis wejścia oktalnego, kluce 9, 10/. Za ostatnim słowem wydziurkowana zostaje litera K. Tak otrzymana taśma może być wprowadzana do maszyny za pośrednictwem Wejścia Oktalnego /kluce 9, 10/.

Kluoz 11 - Wyjście Oktalne z Pamięci Wewnętrznej

Na kluzech należy ustawić:

1. W części prawej /kluce 0-17/ - Adres wewnętrzny.
2. W części lewej /kluce 18-35/ - Ilość słów długich.

Klucz 12 - Wyjście Oktalne z Bębna

Na kluzech należy ustawić:

1. W części prawej /kluce 0-17/ - Adres bębowy.
2. W części lewej /kluce 18-35/ - Ilość słów długich.

B. ZNACZENIE KLUCZY W CZĘŚCI LEWEJ /klucze 18-35/**Klucz 18 - Wyjście SAS**

Po zakończeniu translacji każdego rozdziału, rozdział ten zostaje wyprowadzony w systemie SAS.

Klucz 19 - Wyprowadzenie Słownika Symboli

Po zakończeniu translacji każdego rozdziału i ewentualnym wyprowadzeniu go w systemie SAS /jeżeli był podniesiony klucz 18/ zostaje wyprowadzony Słownik Symboli użytych w tym rozdziale.

Na tabulogramie Słownika Symboli po lewej stronie wypisane są nazwy poszczególnych symboli, a po prawej - odpowiadające im adresy w pamięci wewnętrznej.

Klucz 20 - Wyjście SAB

Po zakończeniu translacji każdego rozdziału i zinterpretowaniu kluczy 18 i 19 rozdział ten zostaje wyprowadzony w systemie SAB. Z lewej strony każdego rozkazu wypisany jest adres tego rozkazu w pamięci wewnętrznej; na końcu rozdziału wypisana jest dyrektywa ROZ α , gdzie α numer rozdziału następnego, lub gdy jest to rozdział ostatni - dyrektywa STA.

Klucz 21 - Wyjście SO

Po zakończeniu translacji każdego rozdziału i zinterpretowaniu kluczy 18, 19, i 20 rozdział ten zostaje wyprowadzony na taśmie dziurkowanej w Systemie Oktalnym. Sposób dziurkowania taśmy dostosowany jest do wymagań podanych przy opisie wejścia w Systemie Oktalnym /klucz 2/, tak że taśma otrzymana w powyższy sposób może być traktowana jako program napisany w Systemie Oktalnym.

Klucz 34 - Stop przed Startem

W przypadku gdy podniesiony jest klucz 34, działanie dyrektywy STA polega na sprowadzeniu do pamięci wewnętrznej rozdziału 0 i wykonaniu rozkazu SS 0.

Klucz 35 - Brak Stopów między Rozdziałami

Normalnie po zakończeniu translacji każdego rozdziału maszyna staje, umożliwiając w ten sposób dokonanie zmian w ustawieniu klawiszy lub założenie taśmy z następnym rozdziałem. W przypadku gdy podniesiony jest klucz 35, rozdziały wchodzą do maszyny w sposób ciągły bez zatrzymań.

SYGNALIZACJA BŁĘDÓW

Translacja Rozdziału programu napisanego w języku SAS składa się z dwóch etapów:

1. Wprowadzenia informacji wydziurkowanych na taśmie do maszyny. W tym czasie działa program zwany PROBINEM.
2. Adresacja, wykonywana przez program zwany ADRESATOREM.

Błędy programu wykrywane są zarówno przez Probin jak i Adresator. Po wykryciu każdego błędu maszyna zatrzymuje się. Rozpoznanie błędu dokonuje się na podstawie stanu rejestrów LR i R. Stan tych rejestrów odczytuje się z lampek na stoliku operatora.

A. Błędy sygnalizowane w czasie działania Probinu:

- Błąd 1. Brak nawiasu po numerze symbolicznym lub kreski ułamkowej (/) po numerze bezwzględnym.
- Błąd 2. Symbol złożony z więcej niż 4 znaków w adresie symbolicznym rozkazu.
- Błąd 3. Nawiąz po symbolu zmiennej w adresie symbolicznym rozkazu.
- Błąd 4. Nawiąz po symbolu paragrafu w adresie rozkazu.
- Błąd 5. Brak znaku (+,-) przed liczbą lub znaku (\equiv) przed szablonem.
- Błąd 6. Liczba nie mieści się w podanej przez programistę skali.
- Błąd 7. Więcej elementów w tablicy, niż podaje dyrektywa TAB.

W przypadku wystąpienia jednego z błędów 1-7 sygnalizacja na stoliku jest następująca:

- a/ rejestr LR - 947
- b/ rejestr R - numer wiersza, w którym wystąpił błąd.

Aby kontynuować wprowadzanie programu, należy ustawić w rej. R adres 8 i nacisnąć START.

C 1

B. Błędy sygnalizowane przez Adresator

Błąd 8. Aktualnie translowany rozdział nie mieści się w pamięci wewnętrznej maszyny.

Sygnalizacja na stoliku:

- a/ rejestr LR - 240
- b/ rejestr R - 240.

Błąd 9. W aktualnym rozdziale użyto więcej niż 200 symboli.

Błąd 10. W adresie rozkazu wystąpił symbol paragrafu nie zadeklarowany w danym rozdziale.

Sygnalizacja na stoliku dla błędów 9 i 10.

- a/ stan rej. LR - 468 lub 360
- b/ stan rej. R - 235.

W przypadku błędu 10 stan LR zawsze 360.

Błąd 11. W adresie rozkazu wystąpił numer symboliczny nie zadeklarowany w odpowiednim paragrafie /użycie nie istniejącego numeru symbolicznego/.

Sygnalizacja na stoliku:

- a/ stan rej. LR - 443
- b/ stan rej. R - adres rzeczywisty miejsca pamięci wewnętrznej, w którym ma być umieszczony błędny rozkaz.

Aby kontynuować adresację w przypadku błędów 8-11, należy w rejestrze R ustawić adres 345 i nacisnąć START.

W przypadku błędów 8 i 9 dalsza adresacja danego rozdziału jest zwykle bezcelowa. W celu uzyskania pewnych dodatkowych informacji można "ustawić żądania" na kluozach 18-33 i nacisnąć START /nie zmieniając stanu rejestru R/. Efekt działania poszczególnych kluczy będzie zgodny z podanym w Instrukcji str. 90.

Uwaga:

Do błędów nie wykrywanych należą:

1. Numer bez nawiasu w adresie symbolicznym rozkazu.
2. Nieprawidłowa część operacyjna.
3. Brak gwiazdki (*) po tablicy.
4. Więcej niż 100 numerów symbolicznych.



Druk „ZNAK”

6. III G1