

Biblioteka Śląska w Katowicach

Id: 0030000687646



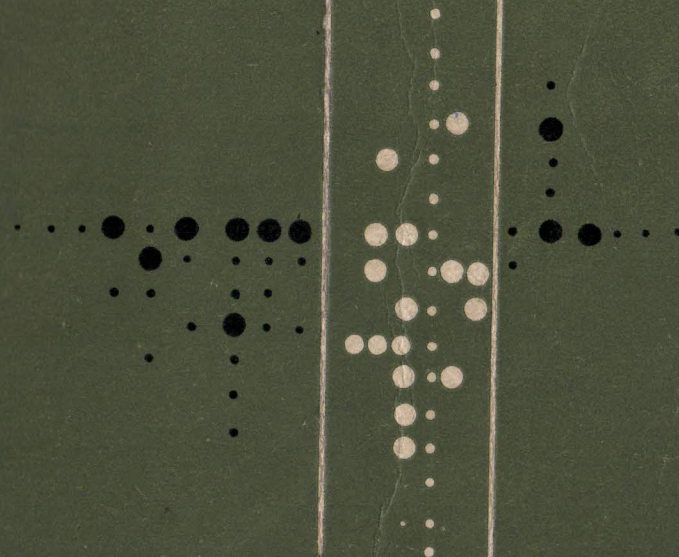
I 75820

75820I

FIAŁKOWSKI

Maszyna CYfrowa

ZAM2



NT

KONRAD FIAŁKOWSKI

MASZYNA CYFROWA ZAM-2

BUDOWA, PROGRAMOWANIE, ZASTOSOWANIA

NT

WYDAWNICTWA NAUKOWO-TECHNICZNE
WARSZAWA

4933/63
Opiniodawca:
Instytut Maszyn Matematycznych PAN
Mgr inż. Jerzy Fiett
Mgr Antoni Mazurkiewicz

681.14

Książka zawiera opis budowy, programowania i zastosowań maszyny cyfrowej polskiej produkcji ZAM-2.

Książka przeznaczona jest dla techników, inżynierów, ekonomistów, działaczy gospodarczych zainteresowanych problematyką maszyn matematycznych.

WSZELKIE PRAWA ZASTRZEŻONE

Printed in Poland

D.K.
Księgarnia
12-00
ul. Wawrzynowa 11.
12.4.63. 6-21.

RT14
75820
I

Okladkę projektował Z. Jaskierski
Redaktor techniczny A. Woźniakowska
Korektor techn. T. Arciszewska-Zenker

WNT Warszawa 1963. Wydanie I. Nakład 4190.
Ark. wyd. 2,8. Ark. druk. 2,99/A. Format B6.
Pap. ilustr. kl. III 70×100 80 g. Rękopis oddano do
składania 11.12.62. Podpisano do druku 26.1.63.
Druk ukończono w lutym 1963. Symbol 77053/Et.
Cena zł 6.—

Bielskie Zakłady Graficzne, Bielsko-Biała,
ul. Grunwaldzka 6 — zam. 2941/62 — D-018



PRZEDMOWA

W związku ze wzrostem zainteresowania elektronicznymi maszynami cyfrowymi, zasadami ich pracy i możliwością zastosowań Dział Informacji Naukowej i Wydawnictw Instytutu Maszyn Matematycznych PAN zwrócił się do mnie z propozycją przygotowania krótkiego opracowania, które zawierałoby w skrócie informacje o budowie, programowaniu i zastosowaniach maszyn cyfrowych. Ponieważ uważam, że każdą popularyzację najłatwiej jest przeprowadzać w oparciu o konkretny przykład, postanowiłem pisać o maszynie ZAM-2, której organizacja daje dobre wyobrażenie o organizacji nowoczesnych maszyn cyfrowych. Poza tym, z polskich maszyn cyfrowych jedynie maszyna ZAM-2 jest wyposażona w Autokod SAKO, którego stworzenie jest osiągnięciem na skalę europejską. Zarys opisu Autokodu i korzyści, jakie z jego stosowania odnosi programista, starałem się scharakteryzować w drugiej części tego opracowania.

Opracowanie zawiera szereg niedopowiedzeń i problemów ledwo naszkicowanych. Wiąże to się zarówno z objętością jak i przeznaczeniem opracowania, które pisałem z myślą o ludziach posiadających wiadomości z matematyki i fizyki w zakresie szkoły średniej, któ-

rzy dotychczas nie mieli okazji zetknąć się z problematyką automatycznych maszyn cyfrowych. Opracowanie to, mam nadzieję, będzie również pomocą dla operatorów, początkujących programistów i studentów cyfroniki.

Chciałbym wyrazić moją wdzięczność prof. dr Leonowi Łukaszewiczowi, dr Tomaszowi Pietrzykowskiemu, mgr inż. Alfredowi Chwieralskiemu, mgr inż. Jerzemu Fietтови, mgr Antoniemu Mazurkiewiczowi, mgr Jerzemu Swianiewiczowi oraz mgr inż. Stanisławowi Waligórskiemu za cenne uwagi i udostępnienie mi materiałów do tej pracy.

Konrad Fiałkowski

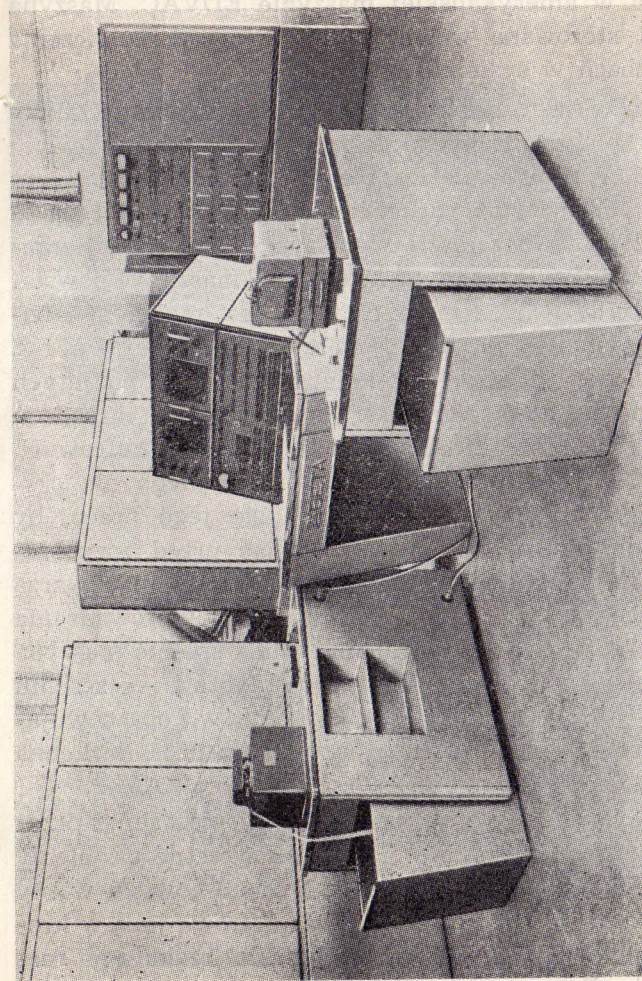
SPIS TREŚCI

Wstęp	7
Budowa i organizacja maszyny ZAM-2	15
Ogólny opis maszyny ZAM-2	15
Postać informacji w maszynie	23
Arytmometr	29
Najprostsze przykłady stosowania rozkazów maszyny ZAM-2	31
Sterowanie i cykl pracy, stolik operatora	33
Pamięć wewnętrzna maszyny ZAM-2	37
Pomocnicza pamięć bębnowa	41
Niektóre dane techniczne maszyny ZAM-2	44
Programowanie maszyny ZAM-2	47
Przygotowanie zadania dla maszyny	47
System Adresów Symbolicznych SAS	48
System Automatycznego Kodowania — SAKO	50
Konkretne problemy rozwiązywane na maszynie ZAM-2	68
Bibliografia	71

jomość położenia sputnika w momencie wystrzelenia rakiety, a więc znajomość toru sputnika. Dokładność wyznaczenia tego toru musi być przy tym tak duża, że wpływ przypadkowych zakłóceń, działających w czasie większym niż jedno okrążenie i nieustannie zmieniających tor sputnika, nie może być pominięty. Tak więc, mając dane charakteryzujące tor, uzyskane z obserwacji w n -tym okrążeniu, należy wyznaczyć tor teoretyczny dla $n + 1$ okrążenia, przy czym wiadomo, że odstępstwa rzeczywistego toru będą w tym wypadku mniejsze niż dopuszczalny błąd. Ale, wobec tego, ten teoretyczny tor musi zostać wyznaczony w czasie krótszym od jednego obiegu sputnika wokół Ziemi, a tego dokonać może tylko maszyna matematyczna.

Inny typ problemów obliczeniowych, w których maszyna ma ogromną przewagę nad rachmistrzem, to na przykład wyznaczenie liczby „ π ” z dokładnością dwustu miejsc dziesiętnych po przecinku, czy wyciąganie pierwiastka kwadratowego z dokładnością do stu dziesiętnych miejsc znaczących. Rachmistrz w trakcie tego typu obliczeń robi nieuniknione błędy, przez co wynik staje się bezwartościowy.

Właśnie szybkość obliczeń, bezbłądność uzyskanych wyników i zdolność jednoczesnego pamiętania dużej ilości danych uwarunkowała niezwykle szybki rozwój maszyn cyfrowych. Początkiem historii elektronicznych maszyn cyfrowych jest rok 1946, gdy na uniwersytecie w Pensylwanii uruchomiono ENIAC'a. Jednakże koncepcja nowoczesnych maszyn cyfrowych pochodzi od Johna von Neumanna i po raz pierwszy została zreali-

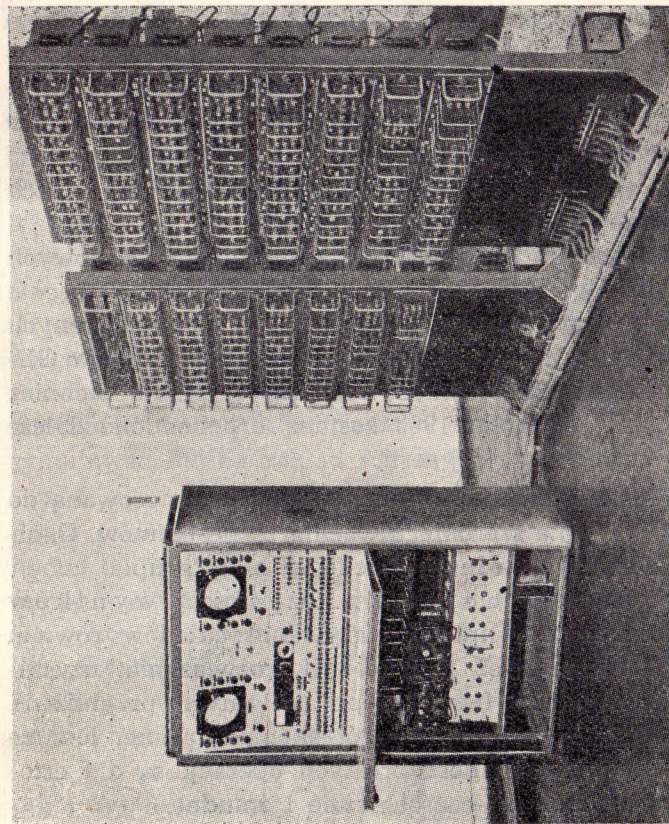


Rys. 1. Widok ogólny maszyny ZAM-2

zowana w amerykańskiej maszynie EDVAC. Maszyna ta jest sterowana wewnętrznie i prowadzi obliczenia na liczbach w systemie binarnym.

Po EDVAC'u, a potem EDSAC'u nastąpił burzliwy rozwój elektronicznych maszyn cyfrowych na całym świecie. W Polsce pierwsza elektroniczna maszyna cyfrowa XYZ została uruchomiona jesienią 1958 roku w Zakładzie Aparatów Matematycznych Polskiej Akademii Nauk. Projekt wstępny tej maszyny powstał w roku 1957. Sama maszyna XYZ jest modelem laboratoryjnym elektronicznej maszyny cyfrowej ZAM-2. Model laboratoryjny — to kolejny etap prac podjętych w celu wyprodukowania serii jakichkolwiek urządzeń technicznych, w tym wypadku maszyn cyfrowych ZAM-2. W zasadzie modelu laboratoryjnego nie eksploatuje się, a jedynie analizuje się jego pracę, by wszystkie zauważone usterki usunąć przed rozpoczęciem produkcji seryjnej. Model laboratoryjny stwarza również możliwości dalszych badań i wprowadzania ulepszeń. Takie właśnie zadanie miała spełnić maszyna XYZ według zamierzeń jej projektantów — zespołu matematyków i inżynierów cyfroników, opracowujących pod bezpośrednim kierownictwem prof. dr Leona Łukaszewicza projekt maszyny cyfrowej ZAM-2.

Mimo to jednak XYZ weszła do eksploatacji i przez trzy lata rozwiązywano na niej konkretne problemy obliczeniowe naszej gospodarki, była to bowiem wtedy największa elektroniczna maszyna cyfrowa jaką w kraju dysponowaliśmy, a setki problemów czekało na rozwiązanie. Przez te trzy lata XYZ pracowała w sposób



Rys. 2. Fragment maszyny cyfrowej XYZ — laboratoryjnego modelu serii maszyn ZAM-2

ciąży jedynie z przerwą na konserwację i remonty. Model laboratoryjny był eksploatowany, tak jak normalnie pracuje prototyp.

XYZ była zresztą nie tylko laboratoryjnym modelem technicznym. Na niej właśnie wypróbowano opracowane w ZAM systemy programowania: System Adresów Symbolicznych SAS oraz później autokod SAKO (System Automatycznego KOdowania). Zostały one przygotowane dla serii maszyn ZAM-2. Piszę o nich w dalszej części tego opracowania.

W tej chwili XYZ pracuje już tylko w wyjątkowych wypadkach. Natomiast z serii ZAM-2 pracują obecnie cztery maszyny: w Instytucie Maszyn Matematycznych PAN, w Biurze Projektów Syntezy Chemicznej w Gliwicach, na Politechnice Łódzkiej i na Politechnice Gdańskiej. W roku 1963 zostaną uruchomione dalsze cztery maszyny z tej serii.

ZAM-2 jest maszyną uniwersalną przystosowaną do rozwiązywania szerokiego wachlarza problemów. Ogólnie zadania te można podzielić na kilka grup:

1. Rozwiązywanie zadań z dużą ilością wyników pośrednich, np. rozwiązywanie układów równań liniowych, czy też jakiegokolwiek rozwiązywania, uzyskiwane metodami iteracyjnymi (kolejnych przybliżeń). Obliczenia te nie są na ogół skomplikowane, lecz ze względu na dużą ilość prostych operacji są dla człowieka nadzwyczaj czasochłonne i żmudne.

2. Rozwiązywanie zadań z dużą ilością zmiennych. Przy obliczaniu tych zadań bez zastosowania maszyn rachmistrz zmuszony jest ze względu na zbyt

wielką ilość obliczeń pominąć wpływ niektórych zmiennych, uzyskując tym samym mniej dokładny wynik. Maszyna, umożliwiając liczenie z dużą szybkością, pozwala na uwzględnienie wszystkich zmiennych i przez to wynik staje się dokładniejszy.

3. Symulowanie czyli badanie uproszczonego matematycznego modelu, np. przedsiębiorstwa, czy zakładu produkcyjnego. Symulowanie pozwala na badanie właściwości projektowanych urządzeń jeszcze w trakcie ich opracowania. Przykładowo, zamiast budować raketę kosmiczną, tworzymy jej matematyczny model w maszynie i badamy jego zachowanie w różnorodnych warunkach lotu. Koszt tego przedsięwzięcia jest nieporównywalny z budową prawdziwej rakiety, pomijając już fakt, że w ten sposób możemy przebadać dziesiątki wariantów urządzenia. Podobnie symulować można zakład przemysłowy.

4. Redukowanie danych. W czasie obserwacji pracy jakichkolwiek urządzeń otrzymujemy wiele danych, takich jak temperatura, ciśnienie, naprężenia, napięcia elektryczne. Prowadząc obserwacje przez pewien czas w różnorodnych warunkach pracy urządzenia, uzyskujemy miliony danych. Maszyna umożliwia uzyskanie średnich, wybranie maksymalnych czy minimalnych wartości parametrów itp.

5. Zastosowania administracyjne, takie jak sporządzanie listy płac czy prowadzenie magazynu. Zastosowania administracyjne nie są typowe dla maszyny ZAM-2, jednak po pewnym przystosowaniu maszyna może być do tych celów użyta.

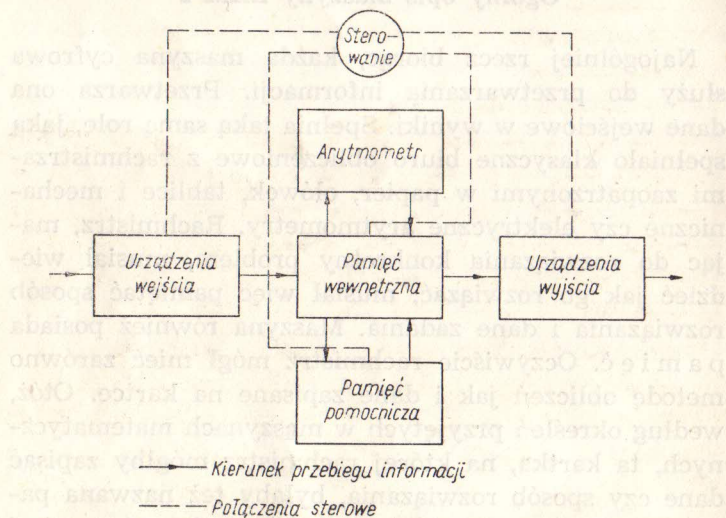
BUDOWA I ORGANIZACJA MASZYNY ZAM-2

Ogólny opis maszyny ZAM-2

Najogólniej rzecz biorąc, każda maszyna cyfrowa służy do przetwarzania informacji. Przetwarza ona dane wejściowe w wyniki. Spełnia taką samą rolę, jaką spełniało klasyczne biuro obliczeniowe z rachmistrzami zaopatrzonymi w papier, ołówek, tablice i mechaniczne czy elektryczne arytmometry. Rachmistrz, mając do rozwiązania konkretny problem, musiał wiedzieć jak go rozwiązać, musiał więc pamiętać sposób rozwiązania i dane zadania. Maszyna również posiada pamięć. Oczywiście rachmistrz mógł mieć zarówno metodę obliczeń jak i dane zapisane na kartce. Otóż, według określeń przyjętych w maszynach matematycznych, ta kartka, na której rachmistrz mógłby zapisać dane czy sposób rozwiązania, byłaby też nazwana pamięcią, tylko, w odróżnieniu od prawdziwej pamięci rachmistrza, nazwana byłaby pamięcią zewnętrzną lub pomocniczą. Podobnie w maszynie wyróżniamy pamięć wewnętrzną i pamięć pomocniczą. O ile z zawartości pamięci wewnętrznej maszyna może w każdej chwili korzystać, o tyle, by skorzystać z zawartości pamięci pomocniczej, maszyna musi tę za-

wartość wprowadzić do pamięci wewnętrznej, zupełnie tak samo jak rachmistrz, który, chcąc skorzystać z danych zapisanych na kartce, musi najpierw te dane przeczytać czyli po prostu zapamiętać „w głowie” tej swojej „pamięci wewnętrznej”.

Dalej, gdy rachmistrz wie już, w jaki sposób należy zadanie rozwiązać, musi rozpocząć wykonywanie ko-



Rys. 3. Schemat przetwarzania informacji przez maszynę cyfrową

lejnych czynności, a więc w pewnym sensie kierować swoją pracą. W maszynie rolę tę spełnia sterowanie. Wszelkie obliczenia rachmistrz wykonuje na papierze lub arytmometrze. Podobnie maszyna posiada

arytmometr. Wreszcie rachmistrz otrzymuje kartkę z danymi zadania, a oddać musi formularz z wypisanymi wynikami. W maszynie czynności przyjmowania danych i wydawnictwa spełniają urządzenia wejścia-wyjścia.

Tak więc maszyna ZAM-2, podobnie jak każda automatyczna maszyna cyfrowa, posiada:

1. Pamięć
2. Arytmometr
3. Sterowanie
4. Urządzenia wejścia-wyjścia.

Pamięć najłatwiej sobie wyobrazić jako zbiór komórek, z których każda ma swój numer, tzn. adres. Każda z takich komórek może pomieścić pewną ilość informacji zwaną słowem. Pamięć wewnętrzna maszyny ZAM-2 składa się z 1024 komórek o adresach od 0 do 1023.

Podobnie jak rachmistrz, który pamięta dane początkowe i metodę rozwiązania, tak samo przygotowana do obliczeń maszyna zawiera w swej pamięci dane i metodę uzyskania rozwiązania zwaną programem. Tak więc program jest to uporządkowany zbiór czynności, które maszyna ma wykonać, by z danych początkowych zadania otrzymać rozwiązanie. Dla pomnożenia dwu liczb przez siebie program byłby następujący:

1. Weź pierwszą liczbę.
2. Pomnóż przez drugą.
3. Podaj wynik.

W maszynie każda z takich czynności programu nazywa się rozkazem. Programista, układając program dla maszyny, pisze po prostu ciąg rozkazów. Rozkazy te wykonywane po kolei przez maszynę doprowadzają ją od danych początkowych do wyników.

Pojedynczy rozkaz mieści się w jednej komórce pamięci. Cała pamięć wewnętrzna maszyny ZAM-2 może pomieścić najwyżej 1024 rozkazy. Oprócz rozkazów, w pamięci maszyny znajdują się jeszcze liczby: dane początkowe, pośrednie wyniki obliczeń itp. Zasadniczym zadaniem sterowania jest właściwa interpretacja kolejnych rozkazów programu i ustawianie arytmometru, urządzeń wejścia-wyjścia i innych zespołów maszyny tak, by możliwe było wykonanie każdego rozkazu. Sterowanie zatem koordynuje pracę maszyny.

Zasadniczym przeznaczeniem arytmometru jest wykonanie w nim działań arytmetycznych na liczbach. Pracuje on szybko w porównaniu do innych elementów maszyny. Czas dodania lub odjęcia dwu liczb wynosi około 0,0001 sekundy, czas zaś mnożenia lub dzielenia 0,003 sekundy. Oczywiście wykonanie całego rozkazu trwa dłużej, bowiem sterowanie musi najpierw pobrać z pamięci odpowiedni rozkaz, zinterpretować go, potem pobrać z pamięci liczbę (druga jest już w arytmometrze umieszczona tam poprzednim rozkazem) i dopiero wykonać działanie.

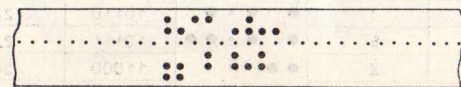
Urządzenia wejścia-wyjścia z kilku względów zasługują na szczegółowe omówienie. Nasza cywilizacja na obecnym etapie rozwoju magazynuje informacje głównie w postaci zapisu alfabetycznego.

Maszyna natomiast przetwarza informację dopiero wówczas, gdy ma ona postać sekwencji impulsów elektrycznych. Zadaniem więc urządzeń wejścia-wyjścia jest przetworzenie informacji zapisanej alfabetycznie w postać elektryczną i odwrotnie.

W maszynie ZAM-2, która jako urządzenia wejścia-wyjścia posiada czytniki i drukarki papierowej taśmy perforowanej oraz dalekopisy, etapem pośrednim przetworzenia informacji z alfabetycznej postaci w elektryczną jest perforowana taśma papierowa.

Dalekopis, przypominający maszynę do pisania, pozwala na wydziurkowanie taśmy, której treść jest odpowiednikiem treści, wydrukowanej równocześnie na papierze. Taśma jest pięciokanałowa, można więc na niej zapisać $2^5 = 32$ różnych znaków. Znaki dalekopisowe i odpowiadające im kombinacje dziurek na taśmie w tzw. kodzie Ferrantiego podaje tablica 1.

Jak łatwo zauważyć, tej samej kombinacji dziurek odpowiada jedna cyfra i jedna litera. Odróżnianie ich możliwe jest przy użyciu znaku „cyfra” lub „litera”. Mianowicie, wszystkie znaki po wydrukowanym znaku



Rys. 4. Taśma papierowa z wydziurkowaną informacją ZAM - 2

„litera” maszyna interpretować będzie jako litery aż do użycia znaku „cyfra”. Po użyciu znaku „cyfra” — jako cyfry, aż do kolejnego znaku „litera”. Sposób przedstawienia informacji na taśmie ilustruje rys. 4.

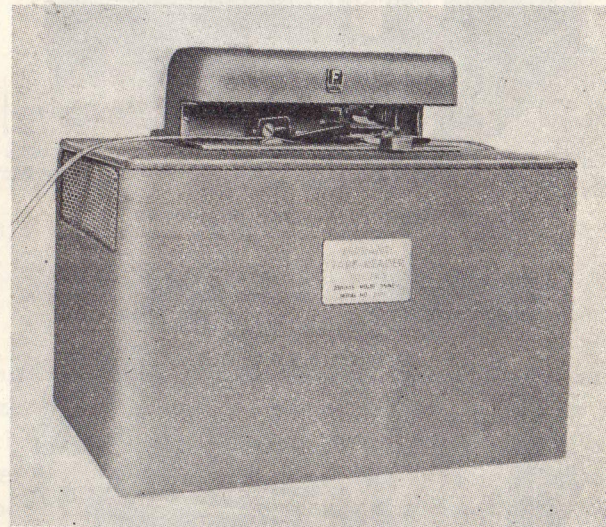
Tablica 1

Znaki dalekopisowe i odpowiadające im kombinacje
dziurek na taśmie papierowej

Cyfry	Litere	Taśma					Kod binarny	Wartość dziesiętna
		5	4	3	2	1		
FS (C y f r y)		.					00000	0
1	A	.	.	.			00001	1
2	B	.	.				00010	2
*	C	.	.	.			00011	3
4	D	.	.				00100	4
(	E	.	.	.			00101	5
)	F	.	.	.			00110	6
7	G	.	.	.	.		00111	7
8	H	.	.				01000	8
≡	I	.	.	.			01001	9
=	J	.	.	.	.		01010	10
-	K	.	.	.	.		01011	11
∨	L	.	.	.	.		01100	12
LF (linia)	M	.	.	.	.	.	01101	13
SP (spacja) ¹⁾	N	.	.	.	.	.	01110	14
,	O	.	.	.	.	.	01111	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
LS (L i t e r y)		.	.	.	.	.	11011	27
.	.	.	.	.	.	.	11100	28
n	?	.	.	.	.	.	11101	29
CR (powrót karetki)	Ł	.	.	.	.	.	11110	30
(ER-błąd)	(ER-błąd)	.	.	.	.	.	11111	31

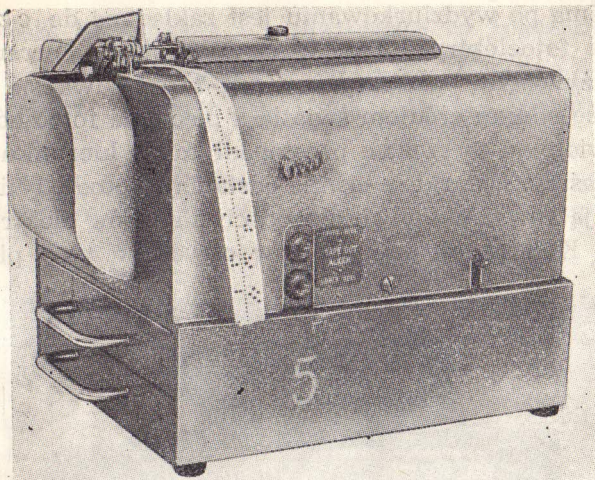
¹⁾ tzn. odstęp

Taśma po wydziurkowaniu jest zakładana do czytnika fotoelektrycznego. W czytniku typu Ferrantiego, używanym w maszynie ZAM-2 przesuwana się ona między silnym źródłem światła a pięcioma fotodiodami ustawionymi w rzędzie prostym do kierunku ruchu taśmy. Dziurki w taśmie odsłaniają fotodiody i powodują powstanie w ich obwodach impulsów elektrycznych. Kombinacje impulsów w obwodach tych pięciu

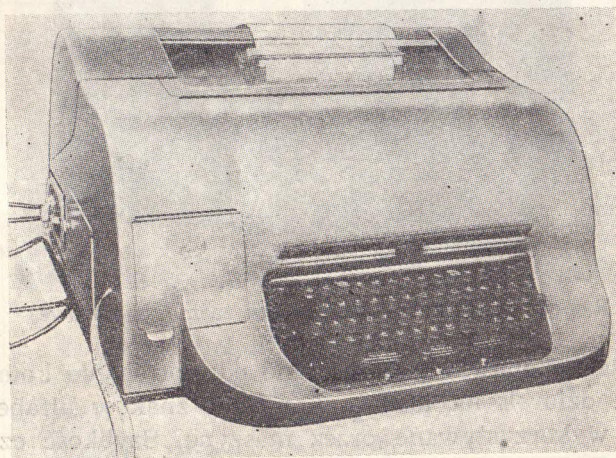


Rys. 5. Czytnik fotoelektryczny taśmy papierowej

fotodiod stanowią jednoznaczne odwzorowanie kombinacji dziurek na taśmie, a więc i znaków alfabetu, i są wykorzystywane przez maszynę. Szybkość czytnika ZAM-2 wynosi około 300 znaków na sekundę.



Rys. 6. Drukarka taśmy papierowej

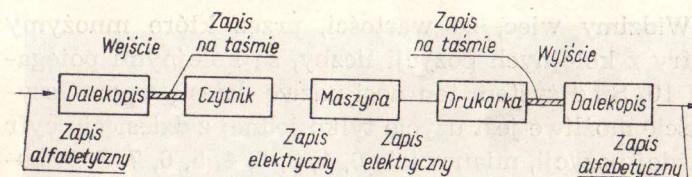


Rys. 7. Dalekopis

Wyniki uzyskane przez maszynę otrzymywane są w postaci takiej samej taśmy wydziurkowanej na drukarce taśmy papierowej, sterowanej przez maszynę. Pracuje ona z szybkością 30 znaków na sekundę.

Uzyskaną taśmę można odczytać na dalekopisie i otrzymuje się informację wydrukowaną znakami alfabetu.

Szybkość dalekopisu zastosowanego w ZAM-2 wynosi 7 znaków na sekundę.



Rys. 8. Schemat współpracy urządzeń wejścia-wyjścia w maszynie ZAM-2

Postać informacji w maszynie

Zarówno rozkazy jak i liczby zapisane są w pamięci maszyny w kodzie dwuwartościowym, tzn. jako ciągi zer i jedynek. Przykładem kodu dwuwartościowego, gdzie występować mogą tylko dwa znaki, jest omówiony poprzednio zapis na taśmie papierowej. Jeśli przyjmujemy, że brak dziurki oznacza zero, a dziurka — jedynkę, każdy znak alfabetu możemy zapisać symbolami kodu dwuwartościowego. Tak więc, na przykład litera R, której na taśmie odpowiada dziurka, brak dziurki, brak dziurki, dziurka, brak dziurki — będzie zakodowana jako 10010 (patrz tablica 1).

Najogólniej mówiąc, każdy zapis jest jakimś sposobem zakodowania informacji. Kodem na przykład jest używany przez nas dziesiętny zapis liczby. Każda bowiem liczba, np. 383, oznacza, że należy wziąć trzy razy po sto, do tego dodać osiem razy po dziesięć i jeszcze dodać trzy razy po jeden, a otrzymamy żadaną wartość. W cyfrowym zapisie dziesiętnym wygląda to następująco:

$$383 = 3 \times 100 + 8 \times 10 + 3 \times 1 = 3 \times 10^2 + 8 \times 10^1 + 3 \times 10^0.$$

Widzimy więc, że wartości, przez które mnożymy cyfry z kolejnych pozycji liczby, są kolejnymi potęgami 10. Stąd system ten nosi nazwę dziesiętnego. Oczywiście możliwe jest użycie tylko jednej z dziesięciu cyfr każdej pozycji, mianowicie 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, bowiem dziesięć jest już równoważne jednności na kolejnej pozycji w lewo od rozpatrywanej. Jeżeli na jakiegokolwiek pozycji występuje zero, oznacza to, że należy wartość tej pozycji dodać do poprzednich zero razy, np.

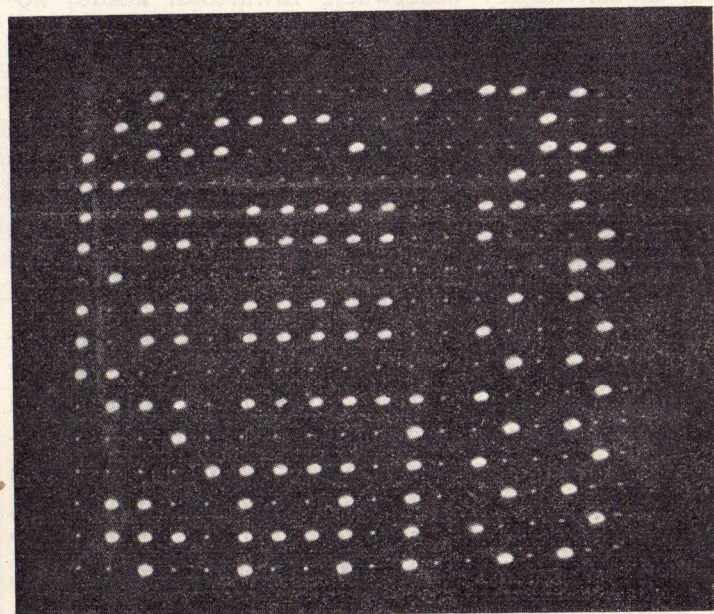
$$30109 = 3 \times 10^4 + 0 \times 10^3 + 1 \times 10^2 + 0 \times 10^1 + 9 \times 10^0.$$

W analogiczny sposób możemy przypisać kodowi binarnemu wartości dla poszczególnych pozycji. Będą to kolejne potęgi dwójki. Oczywiście, na każdej pozycji możliwe są tylko dwie cyfry 0 i 1, ponieważ dwa jest już jedynka na kolejnej pozycji w lewo. Tak więc przykładowo zakodowany znak R 10010 ma w systemie binarnym wartość

$$10010 = 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0,$$

co, jak łatwo obliczyć, odpowiada w systemie dziesiętnym wartości 18. Wartości dziesiętne liczb, które w sy-

stemie binarnym odpowiadają znakom dalekopisowym, podaje ostatnia kolumna tabeli 1. Maszyna, do której przez urządzenia wejścia wprowadzona zostaje liczba, zanim zapisze tę liczbę w pamięci, musi najpierw przetłumaczyć ją z systemu dziesiętnego na system binarny. W maszynie ZAM-2 czynność tę wykonuje zapisany na stałe w pamięci maszyny PROgram BINaryzujący — PROBIN. Tak więc programista, wydziurkowawszy na taśmie liczbę 1049 i wprowadziwszy ją przez czytnik



Rys. 9. Zawartość pamięci wewnętrznej widziana w ekranie synchroskoku (jasne punkty odpowiadają stanowi „1”)

do maszyny, po przetłumaczeniu jej przez PROBIN otrzyma w pamięci maszyny liczbę

000000010000011001.

W maszynie ZAM-2, co zasługuje na szczególne podkreślenie i jest dużym ułatwieniem dla programisty, możliwa jest bezpośrednia obserwacja wprowadzonej liczby w pamięci, o ile programista wie pod jakim adresem, tzn. w jakiej komórce została zapamiętana. W stoliku operatora znajduje się bowiem ekran synchroskopu, umożliwiający obserwację zawartości każdej komórki pamięci wewnętrznej maszyny.

Oprócz programów takich jak PROBIN, na stałe umieszczonych w pamięci maszyny, które maszyna stale „pamięta”, programista rozwiązujący jakieś zadanie wprowadza do maszyny napisany przez siebie program, w którym informuje maszynę o sposobie rozwiązania zadania. Oprócz rozkazów programu, programista wprowadza do maszyny liczby, stanowiące dane lub parametry zadania. Liczby, zgodnie z tym co powiedzieliśmy poprzednio, tłumaczy na system binarny PROBIN i dopiero potem są one zapamiętywane. Podobnie rozkazy umieszczane są w pamięci dopiero po zakodowaniu w kodzie binarnym. Obserwując w ekranie synchroskopu zawartość dowolnej komórki pamięci nie można stwierdzić, czy zapisany jest w niej rozkaz czy liczba, ponieważ widać tylko ciąg zer i jedynek. O tym, czy zostanie on potraktowany jako rozkaz czy jako liczba, decyduje sterowanie.

Słowo jest to jednostka ilości informacji. Rozkaz w maszynie ZAM-2 zawiera informację równą słowu

(krótkiemu). Tak więc każdy rozkaz zajmuje jedną komórkę pamięci wewnętrznej i ma swój adres. Można bowiem powiedzieć, że w komórce 678 (adres) znajduje się DODAJ (rozkaz).

Jedno słowo, a więc i jeden rozkaz, posiada 18 pozycji binarnych — jest osiemnasto-bitowy. Np. rozkaz: „DODAJ zawartość komórki 1...” zapisany w pamięci ma postać

011000000000000001,

podczas gdy programista na taśmie zapisze go jako DO 1. Dwie litery, stanowiące skrót nazwy operacji, są częścią operacyjną rozkazu, natomiast liczba określająca adres komórki, której rozkaz dotyczy, w tym wypadku 1 — jest adresem rozkazu. Bardziej szczegółowy opis budowy rozkazu maszyny ZAM-2 i znaczeń poszczególnych rozkazów można znaleźć w pozycji [2] bibliografii podanej na końcu opracowania. W sumie maszyna ZAM-2 posiada 32 rozkazy. Operacje przez nie dokonywane podaje tablica 2.

Tablica 2
Znaczenie skrótów części operacyjnych rozkazów

- | | |
|----|--|
| 0. | SS — Stop i skocz. |
| 1. | WR — Wykonaj rozkaz wskazany adresem, po czym wróć do wykonywanego programu. |
| 2. | PG — Przygotuj przepisywanie z wejścia albo bębna do pamięci operacyjnej lub z pamięci na wyjście lub bęben. |
| 3. | SK — Skok bezwarunkowy. |
| 4. | SZ — Skocz, jeśli w akumulatorze jest 0. |
| 5. | SP — Skocz, jeśli w akumulatorze jest liczba dodatnia. |

Tabl. 2 (cd.)

6. SN — Skocz, jeśli został zasygnalizowany nadmiar.
7. SW — Skocz, jeśli ilość jedynek w przeczytanym rzędku taśmy była parzysta.
8. SB — Przeskocz dwa rozkazy, jeśli zawartość B rejestru i wskazanego adresem miejsca pamięci są jednakowe.
9. BT — Binarna taśma.
10. UA — Umieścić w akumulatorze liczbę pobraną z pamięci wskazanej adresem.
11. UM — Umieścić liczbę pobraną z pamięci w rejestrze mnożnika.
12. UB — Umieścić liczbę pobraną z pamięci w B-rejestrze.
13. DB — Dodaj liczbę pobraną z pamięci do zawartości B-rejestru.
14. BB — Odejmij liczbę pobraną z pamięci od zawartości B-rejestru.
15. RB — Umieścić w rejestrze bębnowym liczbę z pamięci wskazanej adresem.
16. PP — Przepisz 1 słowo z wejścia lub bębna do pamięci lub z pamięci na bęben albo na wyjście.
17. PA — Prześlij zawartość akumulatora do miejsca pamięci wskazanego adresem.
18. PM — Prześlij zawartość rejestru mnożnika do pamięci.
19. PB — Prześlij zawartość B-rejestru do pamięci.
20. OK — Zaokrąglaj zawartość akumulatora i prześlij wynik do pamięci.
21. LW — Przesuń w lewo łączną zawartość akumulatora i rejestru mnożnika.
22. PW — Przesuń w prawo łączną zawartość akumulatora i rejestru mnożnika.
23. LC — Przesuń cyklicznie w lewo zawartość akumulatora.

Tabl. 2 (cd.)

24. DO — Dodaj słowo pobrane z pamięci do zawartości akumulatora.
25. OD — Odejmuj słowo pobrane z pamięci od zawartości akumulatora.
26. OB — Odejmuj wartość bezwzględną słowa pobranego z pamięci od wartości bezwzględnej zawartości akumulatora.
27. MN — Pomnóż słowo z pamięci przez zawartość rejestru mnożnika. 70-bitowy wynik umieść w akumulatorze i rejestrze mnożnika traktowanych jako całość.
28. DZ — Łączną zawartość akumulatora i rejestru mnożnika podziel przez liczbę pobraną z pamięci, iloraz umieść w rejestrze mnożnika, resztę w akumulatorze.
29. KO — Oblicz koniunkcję zawartości rejestru mnożnika i słowa pobranego z pamięci i zapisz ją w akumulatorze.
30. AL — Oblicz alternatywę słowa pobranego z pamięci i zawartości akumulatora.
31. NI — Nic nie rób.

Arytmometr

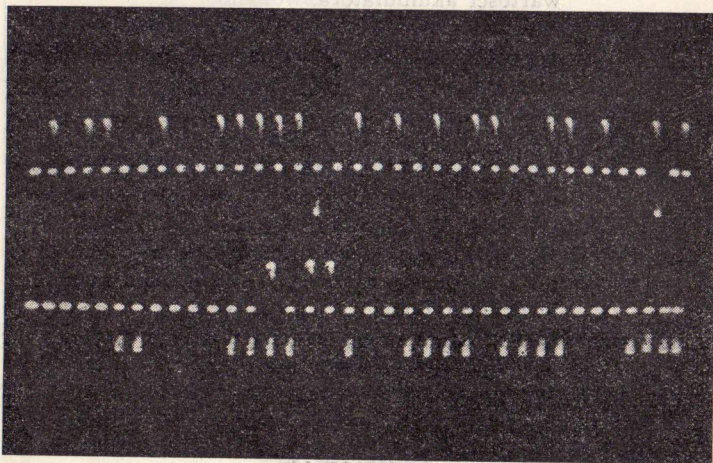
Arytmometr i sterowanie stanowią zasadniczą część maszyny cyfrowej. W arytrometrze wykonywane są operacje arytmetyczne i logiczne.

W celu przetworzenia informacji należy ją najpierw pobrać z pamięci, gdzie została zmagazynowana po wczytaniu z taśmy do maszyny programu i danych. Informacje pobiera się do rejestrów.

Rejestry przechowują informację, która ma być aktualnie przetwarzana.

Zasadniczymi rejestrami arytmometru maszyny ZAM-2 są akumulator i mnożnik. Przetwarzanie informacji umieszczonej w tych rejestrach dokonywane jest w sumatorze.

Zarówno rejestr akumulatora jak i rejestr mnożnika posiada 36 pozycji (35 pozycji modułu i pozycja zna-



Rys. 10. Zawartość rejestrów: akumulatora, mnożnika, pomocniczego i modyfikacji widziana w ekranie synchroskopy (jasne punkty odpowiadają stanowi „1”)

ku, akumulator posiada dodatkowo pozycję nadmiaru). 35 pozycji binarnych odpowiada dziesięcio-cyfrowej liczbie w systemie dziesiętnym i to jest w zasadzie dokładność, z jaką prowadzi się obliczenia w maszynie ZAM-2. Aby zapamiętać w pamięci maszyny taką liczbę, dwie kolejne komórki pamięci — parzystą i nieparzystą — łączą się w jedną. Informacja zawarta

w takiej „połączonej” komórce nazywa się słowem długim. Jeżeli liczba, powstała w wyniku działań, przekroczy zakres maszyny, tzn. jej rozwinięcie w systemie binarnym zawiera więcej niż 35 pozycji, jest to sygnalizowane przez zmianę stanu jednopozycyjnego rejestru nadmiaru. Zawartość 36 pozycji liczby jest jeszcze zapisywana w pozycji nadmiaru akumulatora, natomiast zawartość 37 i wyższych, najbardziej znaczących pozycji liczby jest bezpowrotnie tracona przez maszynę.

Oprócz jednopozycyjnego rejestru nadmiaru, arytmometr zawiera jednopozycyjny rejestr znaku akumulatora i jednopozycyjny rejestr znaku mnożnika. Zawartość tych rejestrów określa znak liczby zawartej odpowiednio w akumulatorze lub w mnożniku.

Akumulator i mnożnik współpracując ze sobą umożliwiają maszynie realizację mnożenia i dzielenia.

Maszyna ZAM-2 liczy z szybkością ok. 1000 dodawań lub odejmowań w sekundzie albo 300 mnożeń lub dzieleni w sekundzie.

Najprostsze przykłady stosowania rozkazów maszyny ZAM-2

Do zawartości akumulatora można dodawać i odejmować zawartości dowolnych miejsc pamięci wewnętrznej. Załóżmy, że w 50 i 51 komórce pamięci znajdują się dwie liczby, które należy dodać i zapamiętać w komórce 52. Program tego zadania jest następujący (części operacyjne rozkazów — patrz tab. 2):

- UA 50 Umieść w akumulatorze zawartość komórki 50
- DO 51 Dodaj do zawartości akumulatora zawartość komórki 51
- PA 52 Zapamiętaj sumę znajdującą się w akumulatorze w 52.

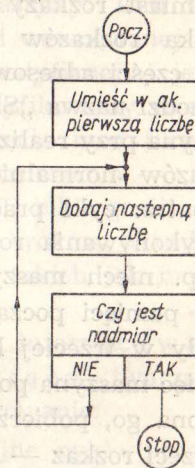
Realizując ten program maszyna pobierze liczbę z komórki 50 i umieści ją w rejestrze, w akumulatorze. Istotną sprawą jest, że pobierając informacje z pamięci czy z rejestru, maszyna nie zmienia zawartości tej komórki czy rejestru. Przykładowo — w komórce 50 zostanie niezmiernona liczba, mimo że ta sama liczba została już umieszczona w akumulatorze. Przy umieszczaniu informacji natomiast poprzednia zawartość komórki czy rejestru ulega zniszczeniu. Tak więc, w naszym przykładzie, poprzednia zawartość akumulatora została zniszczona.

Teraz, gdy pierwsza liczba jest już umieszczona w akumulatorze, maszyna, realizując następny rozkaz, dodaje do niej w sumatorze drugą liczbę i wynik dodawania umieszcza w akumulatorze. Następny rozkaz powoduje zapamiętanie sumy w komórce 52.

Rejestr wskaźnika nadmiaru jak również rejestry znaków są wykorzystywane przez maszynę przy wykonywaniu rozkazów warunkowych. Maszyna może podejmować sama proste decyzje i wybierać jedną z dwu możliwych dalszych dróg postępowania w zależności od uzyskanych poprzednio rezultatów. Właściwości te zapewniają maszynie rozkazy warunkowe.

Powiedzmy, że chcemy dodawać w maszynie ciąg liczb, tak długo aż ich suma przekroczy zakres maszyny. Z góry nie wiemy ile liczb należy dodać, by przekroczyć zakres maszyny. Piszemy więc program dodawania kolejnych liczb, z tym że przed każdym kolejnym dodaniem maszyna sprawdza czy zakres nie został już przekroczony, badając wskaźnik nadmiaru. Graficzny obraz takiego programu przedstawiony jest na rys. 11.

Rozkazami warunkowymi oprócz „SKOCZ przy Nadmiarze”, są „SKOCZ przy Zerze”, „SKOCZ przy Plusie”, „SKOCZ przy Wskaźniku Parzystości” (patrz tablica 2).



Rys. 11. Schemat działań programu z zastosowaniem rozkazów warunkowych

Sterowanie i cykl pracy maszyny. Stolik operatora

Zasadniczymi zadaniami sterowania są: interpretacja rozkazów zawartych w programie oraz kierowanie i synchronizacja procesów zachodzących w maszynie podczas wykonywania tych rozkazów.

Rejestr sterowania, w maszynie ZAM-2 zwany licznikiem rozkazów, zawiera adres komórki pamięci, z której maszyna pobiera rozkaz. Przy normalnych

rozkazach zawartość licznika rozkazów przy każdym kolejno wykonywanym rozkazie zwiększa się o 1. Natomiast rozkazy sterujące zwiększają zawartość licznika rozkazów o liczbę podaną przez programistę w części adresowej rozkazu sterującego. Stąd też pochodzi nazwa „Skocz” części operacyjnej, bowiem maszyna przy realizacji tych rozkazów przerywa ciąg rozkazów normalnie wykonywanych jeden po drugim (patrz: cykl pracy maszyny str. 35) i przechodzi do wykonywania rozkazów w innym odcinku programu. Np. niech maszyna wykonuje program umieszczony w pamięci począwszy od pierwszej komórki, podczas gdy w trzeciej komórce jest rozkaz SKOCZ 15. Tak więc maszyna pobierze z pamięci pierwszy rozkaz, wykona go, pobierze drugi rozkaz, wykona go, pobierze trzeci rozkaz — właśnie Skocz. Maszyna wykonuje go, lecz efektem tego nie jest żadna operacja na liczbach w arytmometrze, lecz jedynie to, że następnym rozkazem pobranym przez maszynę będzie rozkaz z komórki 15, a nie z czwartej, jak zostałby normalnie pobrany, gdyby Skocz zastąpić innym rozkazem. Grupę rozkazów „SKOCZ” nazywamy rozkazami sterującymi, ponieważ sterują one w pewnym sensie pracą maszyny. Oprócz nich istnieją jeszcze w maszynie ZAM-2 specjalne rozkazy, które powodują inne zmiany zawartości licznika rozkazów.

Innym rejestrem sterowania jest rejestr rozkazów, w którym w czasie pracy maszyny umieszczony jest aktualnie wykonywany rozkaz.

Istotny również jest rejestr pośredniczący, który pośredniczy w przekazywaniu informacji między poszczególnymi częściami maszyny. Tam też podczas przesyłania rozkazu z pamięci wewnętrznej do rejestru rozkazów badany jest bit modyfikacji i jeśli bit ten jest jedynką, rozkaz jest modyfikowany w rejestrze modyfikacji. Objaśnienie pojęcia modyfikacji rozkazów przekracza zakres tego opracowania — zainteresowanych odsyłam do pozycji [3] bibliografii.

Warto jeszcze wspomnieć o liczniku kroków, określającym czas trwania operacji długich, takich jak mnożenie, dzielenie czy przesuwanie.

Przekazywaniem i przekształcaniem informacji w całej maszynie kieruje sieć sterująca, tworząca wraz z wymienionymi rejestrami sterowanie.

Maszyna pracuje, wykonując kolejne cykle pracy. Każdy cykl składa się z dwu czynności: pobrania rozkazu i wykonania rozkazu. W pierwszej fazie pierwszej czynności rozkaz pobrany zostaje z pamięci i umieszczony w rejestrze pośredniczącym. Następuje to bezpośrednio po zakończeniu poprzedniego cyklu pracy lub po naciśnięciu przycisku „start”. Adres pobranego rozkazu określa aktualny stan licznika rozkazów. W rejestrze pośredniczącym badany jest bit modyfikacji i, jeśli jest on jedynką, rozkaz zostaje zmodyfikowany. Następnie rozkaz zostaje przesłany z rejestru pośredniczącego do rejestru rozkazów. Na tym kończy się pierwsza czynność cyklu pracy maszyny. W drugiej części cyklu rozkaz zostaje wykonany, przy czym część operacyjna rozkazu zawartego w rejestrze rozkazów

określa rodzaj operacji, część adresowa natomiast określa zazwyczaj adres komórki pamięci wewnętrznej, zawierającej liczbę, do której odnosi się rozkaz. Po wykonaniu rozkazu (o ile nie był to rozkaz sterujący) zawartość licznika rozkazów zostaje zwiększona o 1 i rozpoczyna się następny cykl pracy maszyny.

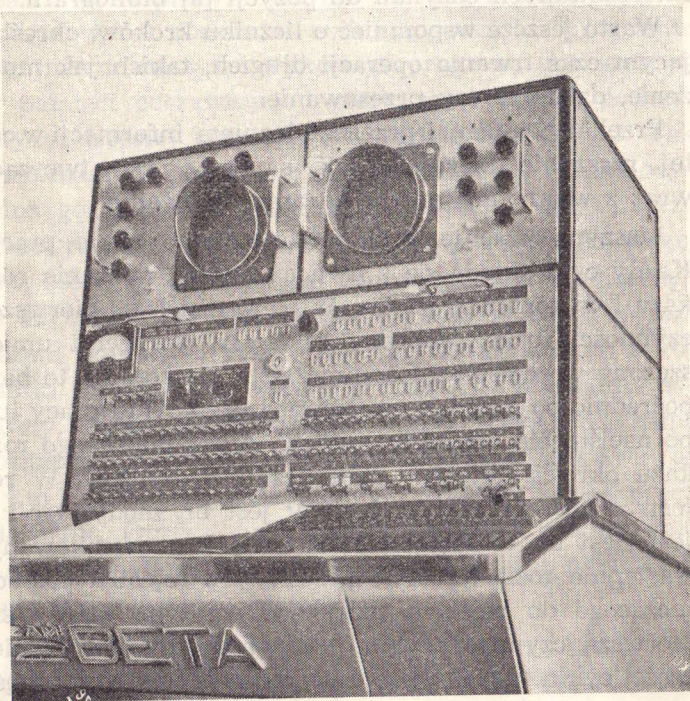
W wypadku rozkazów sterujących (typu: skocz, stop) oraz przy spełnieniu warunków dla rozkazów warun-

kowych zawartość części adresowej rozkazu zostaje przesłana do licznika rozkazów i następny rozkaz zostanie pobrany z komórki pamięci podanej w części adresowej (patrz str. 34). Do bezpośredniej wymiany informacji między programistą a maszyną służy stolik operatora. Przekazywanie informacji maszynie dokonuje się na stoliku operatora za pomocą kluczy. Umożliwiają one w razie potrzeby zewnętrzne sterowanie pracą maszyny. Informacja z maszyny przekazywana jest natomiast przy pomocy obrazów synchroskopowych (patrz rys. 9 i 10) oraz widocznych na rys. 12 neonów.

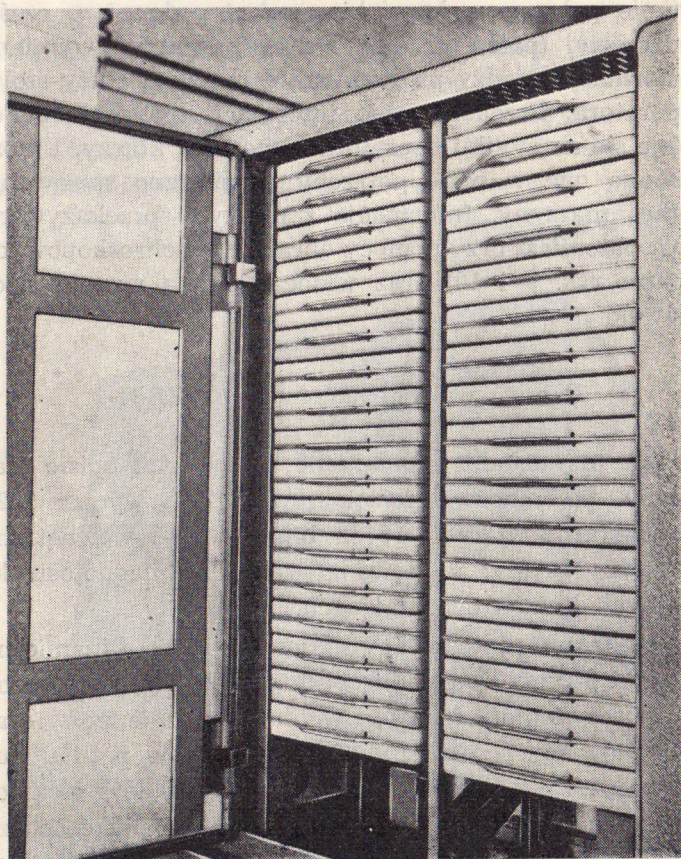
Pamięć wewnętrzna maszyny ZAM-2

Jak już zostało wspomniane w ogólnym opisie maszyny, pojemność pamięci wewnętrznej wynosi 1024 słowa (krótkie) czyli 18432 bity. Pamięć wewnętrzna maszyny ZAM-2 zbudowana jest na 32 magnetostrycyjnych liniach opóźniających.

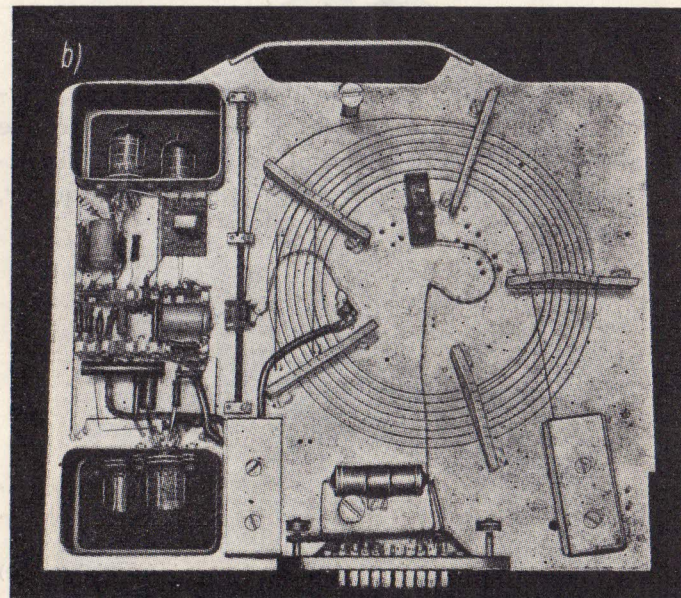
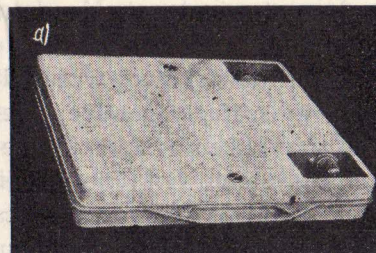
Pamięć magnetostrycyjna jest pamięcią dynamiczną i w odróżnieniu od pamięci statycznych, geometryczne umiejscowienie informacji w linii jest zmienne w czasie. Mówiąc prościej, informacja krąży w pętli utworzonej z linii magnetostrycyjnej i układów elektronicznych na jej wejściu i wyjściu, które połączone ze sobą zamykają pętle. Pojemność informacyjna jednej pętli wynosi 16 słów (krótkich) czyli 288 bitów. Czas opóźnienia linii, a więc czas jaki upływa od momentu



Rys. 12. Stolik operatora w maszynie ZAM-2



Rys. 13. Szafa pamięci wewnętrznej maszyny ZAM-2



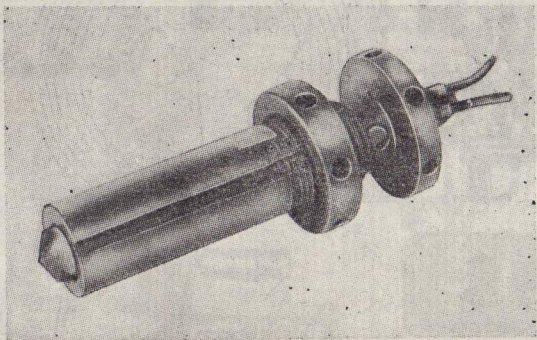
Rys. 14. Kaseta magnetostrykcyjnej linii opóźniającej: a) widok zewnętrzny; b) wnętrze kasety. Widać niklową spiralę, w której rozchodzą się fale ultradźwiękowe

wejścia impulsu w linię do pojawienia się jego na wyjściu, wynosi 711 mikrosekund.

Ponieważ rozchodzenie się fali dźwiękowej w ośrodku zależy od temperatury, w szafach pamięci wewnętrznej musi panować w przybliżeniu stała temperatura. Dzięki odpowiedniemu systemowi klimatyzacji, zastosowanemu w maszynie ZAM-2, temperatura ta nie zależy od temperatury powietrza w pomieszczeniu, w którym znajduje się maszyna.

Pomocnicza pamięć bębnowa

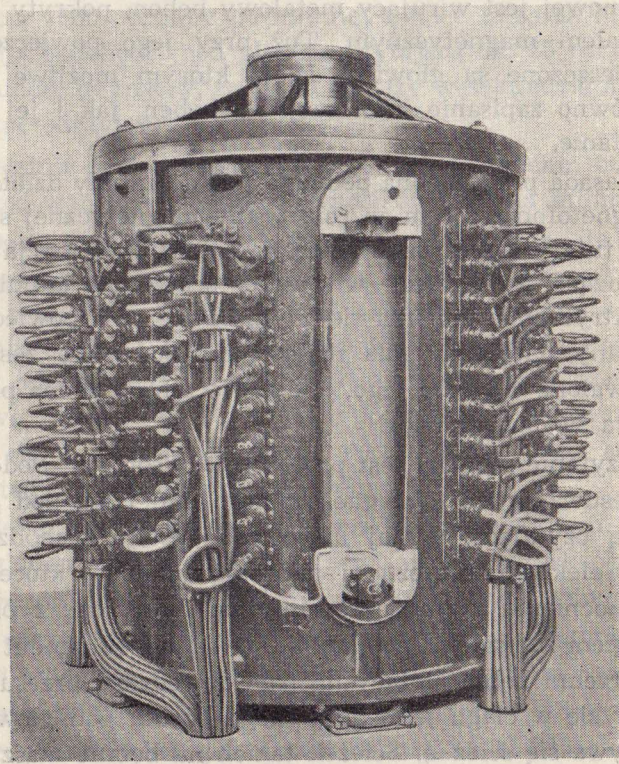
Istnieją programy, których objętość przekracza tysiąc rozkazów i w związku z tym nie mieszczą się one w pamięci wewnętrznej maszyny ZAM-2. Aby umożliwić automatyczne wykonywanie takich programów,



Rys. 15. Głowica bębna pamięci magnetycznej (powiększenie)

maszyna ZAM-2 została zaopatrzona w pomocniczą pamięć bębnową. Maszyna korzysta z pamięci bębnowej w taki sam sposób, jak człowiek z no-

tatek. Magazynuje w niej całą nieprzetwarzaną na bieżąco informację i pobiera ją w miarę potrzeby do pamięci wewnętrznej, oraz, jeśli przewiduje to program, umieszcza w pamięci bębnowej przetworzoną już informację. Czytanie z bębna oraz pisanie na bęben umożliwiają rozkazy z grupy rozkazów urządzeń



Rys. 16. Bęben pamięci magnetycznej maszyny ZAM-2

zewewnętrznych, a mianowicie niektóre warianty rozkazów PRZYGOTUJ i PRZEPISZ (bowiem są to rozkazy złożone) oraz rozkaz USTAW BEBEN (patrz tabl. 2).

Pojemność pamięci bębnowej jest trzydziestodwukrotnie większa niż pamięci wewnętrznej i wynosi 32768 słów krótkich. Zasadniczym elementem pamięci bębnowej jest wirujący metalowy bęben, pokryty materiałem magnetycznym. Tuż przy jego powierzchni umieszczone są głowice, dzięki którym możliwe jest zarówno zapisanie informacji na bęben, jak i jej odczytanie.

Zasada pracy bębna podobna jest do zasady działania magnetofonu, z tym że rolę taśmy magnetycznej spełnia tutaj magnetyczne pokrycie bębna. Informacja dochodząca do głowicy w postaci sekwencji impulsów elektrycznych zmienia się w jej cewce w odpowiednie zmiany natężenia pola magnetycznego, które zostają utrwalone w miarę tego, jak powierzchnia bębna przesuwa się pod głowicą.

Czytanie z bębna jest procesem odwrotnym. Podczas przesuwania się namagnesowanego materiału pod głowicą czytającą, w jej uzwojeniu zostają wzbudzone siły elektromotoryczne i powstają impulsy, które po wzmocnieniu i zregenerowaniu są identyczne z przebiegiem zapisanym poprzednio na bęben. Fragment powierzchni bębna, leżący na jego obwodzie i przesuwany się w ciągu jednego pełnego obrotu pod głowicę, nazywa się ścieżką. Ścieżek takich na bębnie maszyny ZAM-2 jest 128. Pojemność informacyjna każdej ścież-

ki wynosi 256 słów (krótkich). Podobnie jak w pamięci wewnętrznej, każde miejsce na bębnie ma swój adres, pod który można zapisać słowo lub je stamtąd odczytać.

W porównaniu do pamięci wewnętrznej pamięć bębnowa jest stosunkowo wolna. Bęben wiruje z szybkością 1500 obrotów na minutę, a więc ta sama informacja podchodzi pod głowicę czytającą ok. 25 razy na sekundę, to znaczy, że tylko 25 razy w sekundzie może zostać odczytana.

Gęstość zapisu magnetycznego na powierzchni bębna zastosowanego w ZAM-2 wynosi 6 bitów na milimetr.

Osobnym zagadnieniem jest sterowanie bębna. W maszynie ZAM-2 sterowanie bębna zbudowane jest na tranzystorach i wybiera żądany adres w ciągu 2 mikrosekund (dwóch milionowych sekundy). Potem, gdy adres został już wybrany, maszyna czeka aż właściwe miejsce bębna podejdzie pod głowicę, bęben ustawi się w takim położeniu jak wymaga tego adres i wtedy czyta lub pisze na bęben.

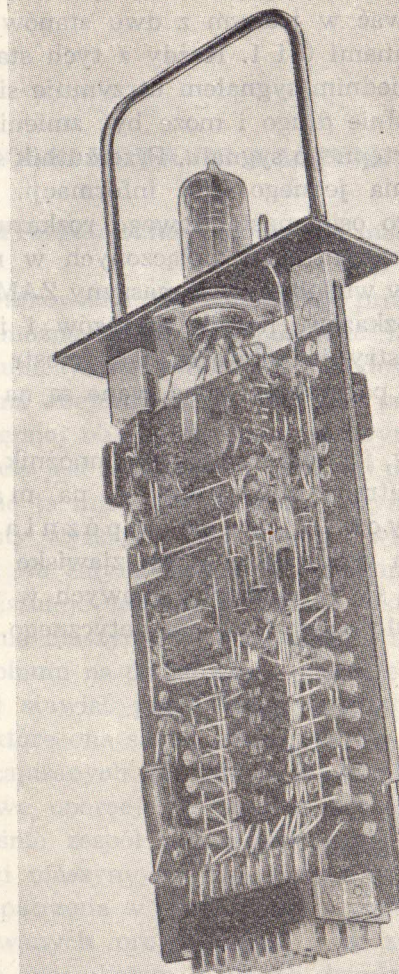
Bęben jest urządzeniem niezwykle precyzyjnym pod względem mechanicznym. „Bicie” w jego łożyskach nie może przekraczać 2 mikronów. Nic zresztą dziwnego, że wymagana jest tak wysoka precyzja wykonania, jeśli odległość między głowicą a powierzchnią magnetyczną bębna wynosi zaledwie 20 mikronów. Przy tych warunkach bęben jest przewidziany na 15 000 godzin pracy bez przerwy.

Niektóre dane techniczne maszyny ZAM-2

Maszyna ZAM-2 jest maszyną szeregową, tzn. informacja przetwarzana jest w niej bit po bicie. Zbudowana została w dynamicznej technice lampowej. Zasilana jest z sieci 3 faz. 380/220 V, 50 Hz. Pobór mocy maszyny wynosi ok. 12,5 KW.

Układy liczące oraz układy pamięci maszyny ZAM-2 umieszczone są w dwu szafach o wymiarach ok. $2300 \times 2000 \times 400$ mm. Stolik operatora, pamięć bębnowa, wejście, wyjście oraz zasilanie wykonane są w postaci odrębnych urządzeń. W szafy i pozostałe urządzenia wbudowany jest system chłodzenia i w związku z tym klimatyzacja pomieszczeń nie jest wymagana. Układy maszyny ZAM-2 składają się ze znormalizowanych podstawowych zespołów, zmontowanych na wymiennych pakietach. Mała ilość typów pakietów w poważnym stopniu upraszcza konserwację maszyny. Maszyna jest wyposażona w pewną ilość zapasowych pakietów i w razie awarii uszkodzone pakiety mogą być natychmiast wymieniane.

Podstawowym elementem maszyny ZAM-2 jest lampowy przerzutnik dynamiczny, zaprojektowany i wykonany w pracowniach Instytutu Maszyn Matematycznych w oparciu o dokumentację Moskiewskiego Instytutu Mechaniki Precyzyjnej i Techniki Obliczeniowej. Oprócz szeregu innych elementów przerzutnik ten posiada kondensator pamiętający, którego ładunek określa stan przerzutnika.



Rys. 17. Typowy pakiet maszyny ZAM-2

Najistotniejszą cechą przerzutnika jest fakt, że może się on znajdować w jednym z dwu stanów umownie nazwanych stanami 0 i 1. Każdy z tych stanów, wywołany odpowiednim sygnałem utrzymuje się w przerzutniku dowolnie długo i może być zmieniony tylko nadejściem następnego sygnału. Przerzutnik służy więc do zapamiętania jednego bitu informacji. Do zapamiętania całego osiemnastobitowego rozkazu potrzeba osiemnaście przerzutników połączonych w rejestr. Część rejestrów wielobitowych maszyny ZAM-2, takich jak rejestr rozkazów, licznik rozkazów i inne, oraz wszystkie rejestry jednobitowe, jak rejestr nadmiaru czy wskaźnika parzystości, zbudowane są na przerzutnikach.

Inne rejestry, jak akumulator czy mnożnik, oraz cała pamięć wewnętrzna zbudowane są na magnetystrykcyjnych liniach opóźniających. W liniach tych wykorzystane jest zjawisko powstania i rozchodzenia się fal ultradźwiękowych w niklu pod wpływem przyłożonego pola magnetycznego.

PROGRAMOWANIE MASZyny ZAM-2

Przygotowanie zadania dla maszyny

Zaprogramowanie jakiegokolwiek problemu obliczeniowego można zasadniczo podzielić na dwa etapy: analizę numeryczną problemu i kodowanie. Analiza numeryczna obejmuje sformułowanie problemu w postaci dostępnej obliczeniom maszynowym. Im bardziej prymitywna jest maszyna i system jej programowania, tym postać ta musi być prostsza i wymaga ze strony programisty wyższych kwalifikacji i większego nakładu pracy. Na całym świecie w dziedzinie maszyn cyfrowych istnieje tendencja takiego wstępnego zaprogramowania maszyny, by programista nie musiał rozwiązywać problemu na podstawowe operacje arytmetyczne, lecz mógł stawiać przed maszyną problemy bardziej złożone, które ona sama, posługując się zespołem programów zapisanych na stałe w jej pamięci, rozłoży na podstawowe operacje, a potem operacje te wykona. Taki właśnie zespół programów zapisanych na stałe w pamięci maszyny nazywa się *autokodem*. Maszyna zaopatrzona w autokod nie wymaga tak wysoko kwalifikowanych programistów i w związku z tym umożliwia specjalistom z innych dziedzin samodzielne

prorowadzenie obliczeń, umożliwia wykorzystanie maszyny dla rozwiązywania interesujących ich problemów. Maszyna ZAM-2 wyposażona jest w autokod SAKO, o którym piszę w dalszej części opracowania.

Kodowanie jest drugim etapem programowania problemu i podobnie jak w etapie poprzednim ilość pracy związana z kodowaniem zależy ściśle od maszyny. W wypadku pracy bez autokodu, programista musi znać całą specyfikę danej maszyny, wszystkie szczególności ją charakteryzujące. Wszelkie działania musi on opisać rozkazami maszyny, przy czym, jeśli operuje adresami bezwzględными, musi dodatkowo pamiętać, która komórka zawiera jaki rozkaz. Przy tej ilości szczegółów, z którymi programista ma do czynienia, pomyłki są prawie nieuniknione i każdy nowy program wymaga starannej kontroli.

System Adresów Symbolicznych SAS

Maszyna ZAM-2 może zostać zaprogramowana w Systemie Adresów Symbolicznych SAS lub w Systemie Automatycznego Kodowania SAKO, przy czym programowanie w SAKO nie wyklucza możliwości pisania fragmentów programu w SASie. O przejściu z języka SAKO na język SAS i odwrotnie, decyduje według swego uznania programista.

System Adresów Symbolicznych SAS pozwala włączyć najbardziej elementarne pracochłonne czynności drugiego etapu programowania do zadań wykonywanych przez samą maszynę. Uwalnia więc od nich pro-

gramistę, skracając tym samym czas programowania i obarcza nimi maszynę, która wykonuje je szybko, a co najważniejsze — bezbłędnie.

Tak więc programista wykonuje tylko niektóre czynności drugiego etapu. Jego czynności ograniczają się do jednoznacznego przyporządkowania symboli rozkazom i danym liczbowym, natomiast nie interesuje go zupełnie, w jakim miejscu pamięci dane te zostaną zapisane. Stąd wynika, że pisząc program w SASie programista wykonuje tylko część pracy wykonywanej przez niego przy programowaniu w języku maszyny, resztę zaś tej pracy wykonuje przed przystąpieniem do właściwych obliczeń sama maszyna. Posiada ona na stałe nagrany w pamięci bieżącej program tłumaczący — TRANSLATOR SAS. Translator ten, zanim maszyna zacznie odczytywać taśmę z programem napisanym w SASie, sprowadza programista do pamięci wewnętrznej za pomocą przyciśnięcia określonych kluczy na stoliku operatora. Po wczytaniu taśmy, ale przed rozpoczęciem obliczeń, translator opracowuje podany mu program w ten sposób, że po zakończeniu działalności translatora program umieszczony w pamięci maszyny operuje adresami bezwzględnymi, a więc wygląda tak, jakby został napisany bez użycia SASu. Jest to oczywiste, bowiem ostateczna postać programu jest zawsze zapisana w adresach bezwzględnych i translator wykonał tylko pracę, którą bez niego wykonałby programista. Efekt pracy jest w końcu ten sam. Pracując jednak w SASie osiągamy następujące korzyści:

— prawdopodobieństwo błędu przy programowaniu w systemie SAS jest mniejsze, a w związku z tym czas potrzebny do uruchomienia programu odpowiednio krótszy,

— w razie jakichkolwiek zmian w programie zmianom ulegają jedynie niektóre adresy symboliczne, większość natomiast programu pozostaje nienaruszona. (Zmiany przyporządkowania symboli i adresów pamięci, w stosunku do wcześniejszej wersji programu, dokona sama maszyna przyporządkowując symbolom inne komórki pamięci. Bez SASu musiałby to robić programista).

Jednakże pracując w SASie programista musi operować rozkazami maszyny i posiadać elementarne wiadomości o jej organizacji. Programowanie SAKO nie stawia przed programistą nawet tych wymagań.

System Automatycznego KODowania SAKO

System Automatycznego KODowania SAKO stworzony został na podstawie opracowań teoretycznych Instytutu Maszyn Matematycznych PAN. Projektodawcami jego są prof. dr Leon Łukaszewicz i mgr Antoni Mazurkiewicz oraz zespół Zakładu Programowania IMM PAN.

Wszystkie maszyny serii ZAM-2 wyposażone są w autokod SAKO zapisany trwale w ich pamięciach bębnowych.

Celem, dla którego stworzony został SAKO, jest znaczne zmniejszenie wysiłku i czasu, potrzebnych

programiście dla przygotowania programu. Chodziło o to, by umożliwić naukowcowi, inżynierowi czy ekonomiście użycie maszyny ZAM-2 jako narzędzia do wykonywanych przez niego obliczeń, nie zmuszając go równocześnie do poznawania podstaw organizacji maszyny i żmudnej nauki programowania w Systemie Adresów Symbolicznych. Opanowanie zaś zasad programowania w języku SAKO trwa zaledwie trzy dni.

Najogólniej mówiąc, SAKO pełni funkcję tłumacza między programistą a maszyną. Programista formułuje problem w postaci podobnej do ogólnie przyjętej w matematyce, wydziurkowuje go na taśmie papierowej i wprowadza do maszyny. Maszyna przed przystąpieniem do obliczeń tłumaczy program z języka SAKO na program *wynikowy*, bezpośrednio dostępny dla obliczeń maszyny. Dokonuje tego program kodujący zapisany na stałe w pamięci bębnowej maszyny. Dopiero po uzyskaniu programu wynikowego maszyna przystępuje do właściwych obliczeń. Tworzenie programu wynikowego jest wieloetapowe i pierwszym etapem jest przetłumaczenie programu z języka SAKO na język SAS, który sam jest fragmentem autokodu SAKO.

Pisanie programów w SAKO nie wyklucza możliwości użycia w specjalnych wypadkach SASu. W tym celu w programie SAKO wystarczy napisać deklarację JEZYK SAS i część programu po tej deklaracji pisać w SASie, aż do deklaracji JEZYK SAKO.

Tak więc korzyści związane z programowaniem w języku SAKO są następujące:

— krótki, trzydniowy czas nauki zasad programowania,

— maksymalne skrócenie drugiego etapu programowania (kodowania),

— duża przejrzystość zapisanego programu,

— małe prawdopodobieństwo popełnienia błędu,

— ułatwienie kontroli, bowiem najczęściej popełniane omyłki podczas pisania programu w języku SAKO maszyna wykrywa sama i sygnalizuje programiście o błędzie i jego rodzaju.

Ogólnie mówiąc, SAKO stanowi dla użytkownika klucz do samodzielnego i szerokiego wykorzystania maszyny w interesujących go dziedzinach i w związku z tym podnosi wartość użytkową maszyn serii ZAM-2.

Programy, napisane w języku SAKO, składają się z ciągu zdań. Rozróżniamy kilkadziesiąt różnych typów zdań w języku SAKO, z których każde ma ściśle określone znaczenie. Również forma każdego z tych zdań jest ściśle ustalona, a posługiwanie się nimi ujęte jest w szereg reguł.

Jak już wspomniałem, praca maszyny zaprogramowanej w języku SAKO składa się z dwu etapów:

— wprowadzenia programu w języku SAKO i przetworzenia go w program wynikowy,

— wykonania programu.

Część zdań SAKO — deklaracje, odnoszą się do pierwszego etapu. Podają one informacje dotyczące budowy programu oraz ustalają znaczenie używanych symboli.

Do drugiego etapu odnoszą się rozkazy. W porównaniu do rozkazów używanych w SASie, są to rozkazy kompleksowe o szerszym znaczeniu.

Możliwy jest również trzeci typ zdań — komentarze. Mają one jednak znaczenie wyłącznie dla programisty, który stosuje je jako objaśnienia dla poszczególnych partii programu. Maszyna, natomiast, komentarzy w ogóle nie bierze pod uwagę.

Cały program napisany w języku SAKO dzieli się na rozdziały, to jest takie odcinki programu, które wraz z przyporządkowanymi im danymi mieszczą się jednocześnie w pamięci wewnętrznej maszyny.

Język SAKO używa konwencjonalnych symboli działań. I tak:

$A + B$ oznacza do A dodaj B

$A - B$ „ od A odejmij B

$A \times B$ „ pomnóż A przez B

A / B „ podziel A przez B

$A * B$ „ podnieś A do potęgi B.

Zmienna, to znaczy symbol przyjmujący w trakcie obliczeń różne wartości, nie musi być oznaczana pojedynczą literą. Jej symbol może być również wieloliterowy, np. EPSILON, SUMA KWADRATÓW.

W języku SAKO możemy używać kilkunastu funkcji, bez uprzedniego specjalnego określania ich znaczenia. Na przykład, chcąc obliczyć pierwiastek kwadratowy zmiennej EPSILON, piszemy

PWK (EPSILON) .

Są to funkcje języka SAKO, zawarte już w samym systemie. Pełną ich listę podaje Tablica 3. Oprócz nich

Skrót	Nazwa	Znaczenie
SIN (X)	Sinus	
COS (X)	Cosinus	
TNG (X)	Tangens	
ASN (X)	Arcus sinus	
ACS (X)	Arcus cosinus	
ATG (X)	Arcus tangens	
ARC (X, Y)	Arcus	Oznacza kąt, leżący w ćwiartce katowej wyznaczonej przez znak X i znak Y, którego tangens jest równy Y/X
PWK (X)	Pierwiastek kwadratowy	
PWS (X)	Pierwiastek sześcienny	
LN (X)	Logarytm naturalny	
EXP (X)	Eksponent	
MAX (X, Y, ..., Z)	Maksimum	Oznacza największą z liczb X, Y, ..., Z
MIN (X, Y, ..., Z)	Minimum	Oznacza najmniejszą z liczb X, Y, ..., Z
MOD (X, Y)	Moduł	Oznacza resztę powstałą z dzielenia X przez Y (tylko dla całkowitych A i B)
SGN (X, Y)	Signum	Oznacza liczbę $ X \cdot \text{sgn } Y$
ABS (X)	Wartość bezwzględna (absolutna)	Oznacza wartość absolutną X
ENT (X)	Część całkowita	Oznacza część całkowitą liczby X

w języku SAKO można tworzyć dowolne funkcje, tak zwane funkcje definiowane. Funkcje definiowane należy jednak najpierw określić i wprowadzić rozkazem PODPROGRAM.

Znając symbole działań i funkcje języka możemy już napisać proste wyrażenia arytmetyczne w języku SAKO. Takim wyrażeniem jest

$$\text{LOG}(\text{SIN}(X + Y) / \text{PWK}(\text{GAMMA}))$$

$$\text{SIGMA}(B / \text{LN}(\text{DZETA} + U - \text{PWS}(\text{KSI})))$$

Chcąc, by maszyna obliczyła wyrażenie arytmetyczne, stosujemy formułę arytmetyczną, np.

$$A = \text{LOG}(\text{SIN}(X + Y) / \text{PWK}(\text{GAMMA}))$$

Rozkaz ten powoduje obliczenie wartości wyrażenia arytmetycznego i nadanie zmiennej A, stojącej po lewej stronie znaku równości poprzednio obliczonej wartości wyrażenia arytmetycznego. Tak więc formuła arytmetyczna nadaje lub zmienia dotychczasowe wartości zmiennych.

W języku SAKO programista dysponuje ponadto formułą operacyjną, która stanowi uogólnienie formuły arytmetycznej. Formuła arytmetyczna bowiem nadaje wartości pojedynczej zmiennej (w naszym przykładzie zmiennej A) przez wykonanie określonych działań na zmiennych wyrażenia arytmetycznego, podczas gdy formuła operacyjna nadaje wartości układowi zmiennych poprzez wykonanie działań podprogramu na układzie jego argumentów.

Oprócz formuł arytmetycznej i operacyjnej istnieje jeszcze szereg innych rozkazów arytmetycznych języka

SAKO, których omówienie przekracza zakres tego opracowania. Podane są one w tablicy 4. Język SAKO umożliwia sterowanie pracą maszyny przez użycie rozkazów sterujących. Rozkaz SKOCZ DO α powoduje przejście do wykonywania rozkazu o numerze α . Natomiast SKOCZ WEDŁUG I: $\alpha, \beta, \gamma, \delta$, jest uogólnieniem poprzedniego rozkazu i powoduje, zależnie od wartości zmiennej I, przejście do rozkazu o odpowiedniej kolejności na liście (jeśli $I = 0$ — do alfa, $I = 1$ — do beta itd.). Wszelkie wielokrotne przejścia tego samego odcinka programu czyli tzw. pętle realizowane są przy użyciu rozkazu POWTÓRZ OD α : $I = J(K)L$. Rozkaz ten powoduje powtórzenie odcinka programu między rozkazem α a nim samym tyle razy, aż zmienna I mająca przy pierwszym powtórzeniu wartość J, zmieniając się przy każdym powtórzeniu o K, osiągnie wartość L. Rozkazy warunkowe powodują przejście do rozkazu α , jeśli warunek został spełniony, w przeciwnym razie — do rozkazu β . Takimi rozkazami są: GDY $A > B$: α , INACZEJ β , GDY $A = B$: α , INACZEJ β , GDY BYL NADMIAR: α , INACZEJ β , GDY KLUCZ I: α , INACZEJ β (gdych klucz I na stoliku operatora został włączony).

Dane w języku SAKO przygotowuje się zazwyczaj w postaci osobnej taśmy perforowanej, którą po wprowadzeniu programu do maszyny zakłada się do czytnika. Maszyna w miarę wykonywania programu wczytuje kolejne dane. Oczywiście można wczytać wszystkie dane od razu i zmagazynować je w pamięci wewnętrznej lub bębnowej; decyzja w tej sprawie cał-

kowicie zależy od programisty. Pod tym względem maszyna ZAM-2 programowana w systemie SAKO stwarza wiele możliwości i każdą z nich programista może wykorzystać zależnie od metody pracy, jaką wybierze.

Wczytywanie taśmy przez maszynę powoduje rozkaz CZYTAJ.

Tablica 4

Lista wyrażeń w języku SAKO

1. DEKLARACJE
 1. ROZDZIAŁ: nazwa
 2. PODPROGRAM: (lista wyników) = nazwa (lista argumentów)
 3. BLOK: (n, m, ..., k): lista
 4. STRUKTURA (I, J, ..., K): lista
 5. TABLICA (n, m, ..., k): nazwa
 6. TABLICA OKTAŁNA (n, m, ..., k): nazwa
 7. CAŁKOWITE: lista
 8. SKALA DZIESIĘTNA PARAMETRÓW n
 9. SKALA BINARNA PARAMETRÓW n
 10. JEZYK SAS
 11. JEZYK SAKO
 12. KONIEC
2. ROZKAZY STERUJĄCE
 1. SKOCZ DO α
 2. SKOCZ WEDŁUG I: $\alpha, \beta, \gamma, \delta, \dots, \omega$
 3. POWTÓRZ OD α : $I = J(K)L$
 4. IDZ DO ROZDZIAŁU: nazwa
 5. WROC
 6. GDY $A > B$: α , INACZEJ β
 7. GDY $A = B$: α , INACZEJ β
 8. GDY BYL NADMIAR: α , INACZEJ β
 9. GDY KLUCZ I: α , INACZEJ β
 10. STOP α

3. ROZKAZY ARYTMETYCZNE I BULOWSKIE

1. $A = B$
2. $A \equiv B$
3. (lista) = nazwa podprogramu (lista)
4. PODSTAW: nazwa podprogramu (lista)
5. USTAW SKALE DZIESIETNIE : I
6. USTAW SKALE BINARNIE : I
7. ZWIEKSZ SKALE DZIESIETNIE O I : lista
8. ZWIEKSZ SKALE BINARNIE O I : lista

4. ROZKAZY WEJSCIA I WYJSCIA

1. CZYTAJ : lista
2. CZYTAJ OKTALNIA : lista
3. CZYTAJ WIERSZ : nazwa bloku
4. DRUKUJ (I.J) : lista zmiennych
5. DRUKUJ WIERSZ : nazwa bloku
6. DRUKUJ SLOWO : lista zmiennych
7. TEKST :
8. TEKST WIERSZY n :
9. SPACJA n
10. LINIA n

5. ROZKAZY WSPOLPRACY Z BEBNEM

1. CZYTAJ Z BEBNA OD I : lista
2. PISZ NA BEBEN OD I : lista

Dla wyraźniejszego zilustrowania korzyści jakie są osiągane przy programowaniu w systemie SAKO, podaję program obliczający okres drgań wahadła w zależności od jego długości i kąta wychylenia. Program ten napisany jest raz w języku maszyny i powtórnie w języku SAKO. Sam problem jest następujący:

Jak wiadomo okres drgań wahadła fizycznego wyraża się wzorem

$$T = 2\pi \sqrt{\frac{I}{mgr}} \left(1 + \frac{1}{4} \sin^2 \frac{1}{2} \alpha + \dots \right)$$

gdzie

T — okres drgań wahadła,

m — masa wahadła,

I — moment bezwładności wahadła względem punktu zawieszenia,

r — odległość punktu zawieszenia od środka ciężkości wahadła,

g — przyspieszenie grawitacyjne,

α — amplituda kątowa drgań wahadła.

Problem polega na wyznaczeniu okresu drgań w zależności od r i α

$$T = T(r, \alpha)$$

Do obliczeń posłużymy się przybliżeniem wzoru ze względu na α , mianowicie przyjmiemy, że

$$T = 2\pi \sqrt{\frac{I}{mgr}} \left(1 + \frac{\alpha^2}{16} \right)$$

Dla ustalonych wartości parametrów m , g , I obliczymy T dla zmiennej r w zakresie $1 \leq r \leq 10$, liczone co 1 i dla zmiennej α w zakresie $\frac{\pi}{16} \leq \alpha \leq \frac{\pi}{2}$, liczone co $\frac{\pi}{16}$.

Tak więc otrzymamy 80 wartości T, mianowicie

$$T\left(1, \frac{\pi}{16}\right), T\left(1, \frac{\pi}{8}\right), \dots, T\left(1, \frac{\pi}{2}\right)$$

$$T\left(2, \frac{\pi}{16}\right), T\left(2, \frac{\pi}{8}\right), \dots, T\left(2, \frac{\pi}{2}\right)$$

.....

$$T\left(10, \frac{\pi}{16}\right), T\left(10, \frac{\pi}{8}\right), \dots, T\left(10, \frac{\pi}{2}\right)$$

Zakładamy ponadto, że ani jedna z danych oraz żaden z wyników pośrednich nie przekraczają co do modułu wartości 10 000, co odpowiada w maszynie skali dziesiętnej 4 (pojęcie skali, patrz poz. [1] bibliografii). Zarówno program w języku maszyny jak i w SAKO napisany jest w skali 4.

Poniżej podaję program dla ZAM-2 napisany w języku maszyny. Program zawiera pewne uproszczenia. Pojęcie liczb długich (patrz poz. [2] bibliografii) nie zostało w tym przykładzie wprowadzone. Zakłada się jednakże, że zarówno rozkazy jak i liczby umieszczone są wyłącznie w miejscach długich (36 bitowych), przy czym miejsca te są ponumerowane od 0 do 512. Pominięta również została możliwość automatycznej modyfikacji rozkazów.

Program napisany jest przy założeniu, że dane parametry i stałe zostały wprowadzone do maszyny i zapisane w skali 4. Rozmieszczenie ich w pamięci wewnętrznej jest następujące:

NR kom. P.W.	Zawartość	Objaśnienia
0	$\frac{\pi}{16}$	stała
1	liczbowa wartość I	I < mgr (Jeśli warunek nie jest spełniony program byłby bardziej skomplikowany niż podany w przykładzie)
2	liczbowa wartość m	
3	liczbowa wartość g	
4		Miejsce robocze przeznaczone na każdorazową wartość r
5		Miejsce robocze przeznaczone na każdorazową wartość α
6	1	Jedynka w skali 4
7		
8		miejsca robocze
9		
10	liczba A	Liczba A musi spełniać
11	liczba B	warunek $A > \frac{I}{mgr_{\min}}$ } stałe potrzebne dokładność z jaką chcemy obliczyć $\sqrt{\frac{I}{mgr}}$ } do obliczeń pierwiastka kwadrat.
12	11	Liczba 11 w skali 4
13	PA 150	Pseudorozkaz
14	$1 \cdot 2^{-35}$	Jedynka w skali maszyny, czyli bit na 35 pozycji słowa
15	0	Zero

NR kom. P.W.	Zawartość		Objaśnienia
	Program umieszczony jest w maszynie począwszy od miejsca 16 i pierwszym rozkazem pobranym będzie rozkaz z komórki 16.		
16	UA	13	Pseudorozkaz PA 150 zostaje zapamiętany w komórce 63 Pierwsza wartość r , $r=1$ zostaje zapamiętana w komórce 4 $m \cdot g$
17	PA	63	
18	UA	6	
19	PA	4	
20	UM	2	
21	MN	3	Rozkaz ustawia iloczyn w skali 4 i przesyła go do mnożnika mgr Wartość mgr zapamiętana w komórce 7 I mgr Wartość I/mgr zapamiętana w komórce 7
22	PW	31	
23	MN	4	
24	LW	4	
25	PA	7	
26	UA	1	$y_0 = A$ $\frac{a}{y_i}$ $\frac{a}{y_i} - y_i$
27	DZ	7	
28	PW	4	
29	PM	7	
30	UA	10*	
31	PA	8	
32	UA	7	
33	DZ	8	
34	UA	15	
35	LW	31	
36	OD	8	

*) Obliczanie pierwiastka kwadratowego iteracyjną metodą Newtona-Raphsona $y_{i+1} = y_i + 1/2 \left(\frac{a}{y_i} - y_i \right)$. Pierwszą wartością na y , $y_0 = A$. Iteracja jest powtarzana tak długo aż $|y_{i+1} - y_i| < B$

NR kom. P.W.	Zawartość		Objaśnienia
37	PW	1	$1/2 \left(\frac{a}{y_i} - y_i \right)$
38	PA	9	$1/2 \left(\frac{a}{y_i} - y_i \right)$ Zapamiętana w komórce 9
39	OB	11	$ 1/2 \left(\frac{a}{y_i} - y_i \right) - B$ Rozkaz warunkowy.
40	SP	42	Czy $ 1/2 \left(\frac{a}{y_i} - y_i \right) - B > 0$?
41	SK	46	Jeśli tak prowadź dalej iteracje (skok do 42 komórki)
42	UA	8	Jeśli nie, iteracja skończona.
43	DO	9	
44	PA	8	
45	SK	32	
46	UM	0	
47	LW	5	$\pi/16 \cdot 2^5 = 2\pi$
48	MN	8	$2\pi \sqrt{\frac{I}{mgr}}$
49	LW	4	
50	PA	9	$2\pi \sqrt{\frac{I}{mgr}}$ zapamiętany w komórce 9
51	UA	0	
52	PA	5	$\pi/16$ — pierwsza wartość α zapamiętana w komórce 5
53	LW	3	
54	PA	7	$\pi/16 \cdot 2^3 = \pi/2$ — zapamiętana w komórce 7
55	UM	5	
56	MN	5	$\alpha \cdot \alpha = \alpha^2$
57	LW	4	

NR kom. P.W.	Zawartość		Objaśnienia
58	PW	4	$\frac{\alpha^2 \cdot 2^{-4}}{16} = \frac{\alpha^2}{16}$
59	DO	6	$\frac{\alpha^2}{16} + 1$
60	PW	35	$\left(\frac{\alpha^2}{16} + 1\right)$ przesunięta do mnożnika
61	MN	9	
62	LW	4	
63	PA	150	$2\pi \sqrt{\frac{I}{mgr} \left(\frac{\alpha^2}{16} + 1\right)}$
64	UA	63	$T\left(\frac{\pi}{16}, 1\right)$ zapamiętana w komórce 150
65	DO	14	Zamiast PA 150 w komórce 63 jest
66	PA	63	PA 151. Tak więc $T\left(\frac{\pi}{8}, 1\right)$ zostanie zapamiętana w 151,
67	UA	5	$\left(T \frac{3\pi}{16}, 1\right)$ w 152 itd.
68	DO	0	$\alpha_i + \pi/16$ — tworzenie kolejnej wartości α
69	OD	7	$\alpha_i - \pi/2$
70	SP	74 **	Rozkaz warunkowy. Czy $\alpha_i - \pi/2 > 0$? Jeśli tak, to ostatnią wartością liczoną było $\pi/2$, a więc zostały obliczone $T(r_1, \pi/16)$, $T(r_1, \frac{2\pi}{16}), \dots, T(r_1, \frac{\pi}{2})$; zakres α wy-

**) Dla uproszczenia przyjmujemy, że warunek dla rozkazu SP jest spełniony tylko dla liczb dodatnich większych od zera.

NR kom. P.W.	Zawartość		Objaśnienia
71	DO	7	czerpany i należy zacząć liczyć dla następnego r. Jeśli nie, należy liczyć dla następnego α $(\alpha_i - \pi/2) + \pi/2$
72	PA	5	
73	SK	55	
74	UA	4	Obliczana kolejna wartość r $r_i + 1 = r_{i+1}$ $r_{i+1} - 11$
75	DO	6	
76	OD	12	
77	SZ	81	Rozkaz warunkowy: Czy $(r_{i+1} - 11) = 0$. Jeśli tak, to r_i było równe 10, zakres zmienności r został wyczerpany i należy przerwać liczenie. Jeśli nie, należy liczyć dalej dla kolejnego r_i
78	DO	12	
79	PA	4	
80	SK	20	Maszyna przerywa pracę. W razie naciśnięcia klucza „start” rozpocznie pracę od pobrania zawartości komórki 16, a więc zacznie wykonywać program od początku.
81	SS	16	

Ten sam program napisany w SAKO ma następującą postać:

USTAW SKALE DZIESIETNIE : 4

1) CZYTAJ : I, M, G, PI

** 2) $T = 2 \times PI \times PWK (I/M \times G \times R) \times (1 + A \cdot 2/16)$

LINIA

DRUKUJ (4·4) : T

PI POL = PI/2

PI SZESNASTE = PI/16

POWTORZ OD 2 : A = PI SZESNASTE (PI SZESNASTE) PI POL

POWTORZ OD 2 : R = 1.0(1.0)10.0

STOP 1

KONIEC

Należy przy tym dodać, że program napisany w SAKO, wykonując to wszystko co poprzedni program, wprowadza oprócz tego parametry do maszyny oraz drukuje wyniki.

Chcąc uzyskać czytanie i drukowanie programem w języku maszyny, ilość rozkazów należałoby co najmniej podwoić.

Pozostawiając czytelnikowi samodzielne zapoznanie się z kolejnymi czynnościami programu napisanego w języku maszyny (pomocą mogą być przy tym szczegółowe objaśnienia z prawej strony programu), omówię krótko działanie programu napisanego w SAKO.

Pierwszy wiersz zawiera rozkaz odnoszący się do skali. Skala jest problemem nie omawianym w tym opracowaniu. Wykonanie rozkazu CZYTAJ spowoduje odczytanie czterech liczb z taśmy papierowej założonej uprzednio do czytnika i nazwanie tych liczb odpowiednio I, M, G, PI. Są to parametry występujące we wzorze. Następny wiersz jest formułą arytmetyczną. Wartość wyrażenia po prawej stronie znaku równości zostanie obliczona i nazwana T. Nietrudno zauważyć, że formy zapisu

$$T = 2 \times PI \times PWK (I/M \times G \times R) \times (1 + A \cdot 2/16)$$

$$T = 2\pi \sqrt{\frac{I}{mgr} \cdot \left(1 + \frac{A^2}{16}\right)}$$

pierwsza w SAKO i druga tradycyjna, przyjęta w matematyce, są zbliżone. Rozkaz: DRUKUJ powoduje wydrukowanie wartości T, z tym że będzie ona zawierała 4 miejsca dziesiętne przed przecinkiem i po przecinku.

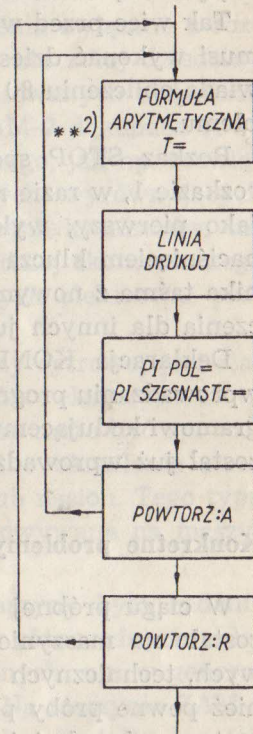
Poprzedzający DRUKUJ rozkaz LINIA powoduje jedynie rozpoczęcie drukowania z nowego wiersza. Następne dwie formuły służą do obliczenia wartości PI POL,

tn. $\frac{\pi}{2}$ i PI SZESNASTE, tzn. $\frac{\pi}{16}$

potrzebnych do wykonania rozkazu POWTORZ. Rozkaz POWTORZ powoduje powtórne wykonanie części programu zawartej między rozkazem 2) (CZYTAJ), a samym POWTORZ. Powtórzeń takich będzie tyle, aż wartość, początkowo równa PI SZESNASTE, zwiększając się za każdym powtórzeniem o PI SZESNASTE, stanie się równa wartości PI POL. A więc powtórzeń takich będzie 8. Odpowiada to obliczeniu i wydrukowaniu kolejno

$$T\left(\frac{\pi}{16}, 1\right), T\left(\frac{2\pi}{16}, 1\right), \dots, T\left(\frac{\pi}{2}, 1\right).$$

Po obliczeniu tych 8 wartości zostanie wykonany drugi z kolei



Rys. 18. Fragment sieci działań omawianego programu ilustrujący działanie rozkazów POWTORZ

rozkaz POWTORZ, odnoszący się do R. Powtórzeń tych będzie 10. Łatwo zauważyć, że każdemu powtórzeniu odnoszącemu się do R odpowiada 8 powtórzeń odnoszących się do A. Ilustruje to rysunek 18.

Tak więc przed wykonaniem rozkazu STOP maszyna musi wykonać dziesięciokrotnie 8 powtórzeń, co odpowiada obliczeniu 80 wyników. Jest to rozwiązanie problemu.

Rozkaz STOP spowoduje zatrzymanie maszyny na rozkazie 1, w razie naciśnięcia klucza START maszyna, jako pierwszy, wykona rozkaz CZYTAJ. Jeśli przed naciśnięciem klucza START włożona zostanie do czytelnika taśma z nowymi danymi, maszyna powtórzy obliczenia dla innych już wartości, I lub M.

Deklaracja KONIEC ma znaczenie wyłącznie przy wprowadzaniu programu do maszyny i sygnalizuje programowi kodującemu, że cały program w języku SAKO został już wprowadzony do maszyny.

Konkretne problemy rozwiązane na maszynie ZAM-2

W ciągu próbnej pracy w IMM PAN rozwiązanych zostało na maszynie ZAM-2 wiele problemów naukowych, technicznych i ekonomicznych. Poczyniono również pewne próby przystosowania maszyny ZAM-2 do zastosowań administracyjnych. Zastosowania ekonomiczne głównie sprowadzają się do rozwiązań zadań z programowania liniowego. Maszyna ZAM-2 wykonuje tego typu obliczenia głównie dla Ministerstwa Handlu Wewnętrznego i dla potrzeb budownictwa.

Klasycznym i typowym przykładem zadania z dziedziny programowania liniowego jest obliczenie najekonomiczniejszego przewozu piasku z n źródeł na m budów.

Zastosowania naukowe to obliczenia związane z rozpraszaniem nukleonów. Doświadczenia z tej dziedziny prowadzi się w najbardziej zaawansowanych badaniach jądra atomowego. Na maszynie ZAM-2 dokonano również wielu obliczeń typu geodezyjnego, jak np. rozwiązywanie układów równań liniowych wyrównujących sieć geodezyjną metodą najmniejszych kwadratów. Inny problem, to wyznaczenie najbardziej ekonomicznego rozkładu obciążeń dla poszczególnych elektrowni pracujących w jednym systemie mocy.

Typowym zastosowaniem maszyny cyfrowej jest statystyczne opracowywanie wyników eksperymentów. Przedmiotem badań w tym wypadku może być np. rozrzut parametrów tranzystorów, albo określenie standardowych odchyłek próbek zbóż lub nasion. Tego typu problemy były wielokrotnie opracowywane na maszynie ZAM-2.

Technicznym zastosowaniem maszyny było badanie charakteru drgań występujących w różnorodnych układach elektrycznych lub mechanicznych, np. w generatorach mocy, lub w kratownicach. Na maszynie ZAM-2 wyznaczane były również optymalne kształty łopatek turbin.

Z obliczeń prowadzonych na ZAM-2 dla potrzeb przemysłu warto wspomnieć o badaniach procesów cieplnych zachodzących w transformatorach mocy. Sy-

stematyczna analiza tego zagadnienia bez użycia elektronowej maszyny cyfrowej stanowi nadzwyczaj żmudny i pracochłonny problem. Na ZAM-2 obliczono również najkorzystniejsze zestawy soczewek optycznych w obiekttywach. Dla potrzeb przemysłu chemicznego określono parametry mieszanin gazowych w różnych warunkach ciśnienia i temperatury.

Ostatnio przystąpiono do prac nad przystosowaniem maszyny ZAM-2 do zastosowań administracyjnych. Ponieważ zastosowania te charakteryzują się mniejszą różnorodnością obliczeń, natomiast większą ilością danych wejściowych i wyników, maszyna ZAM-2 przystosowana do zastosowań administracyjnych jest wyposażona w dwa bębny pamięci magnetycznej, dwa czytniki oraz dwie drukarki. Pierwszym ukończonym już problemem z tej dziedziny jest prowadzenie ewidencji materiałowej magazynu przez maszynę ZAM-2. Również maszynę ZAM-2 zastosowano do prób opracowania wyciągów ewidencji ludności dla celów Stołecznej Rady Narodowej.

Wszystkie te zastosowania podane są ilustracyjnie i nie wyczerpują możliwości maszyny, która jako maszyna uniwersalna może być stosowana do obliczania szerokiego wachlarza problemów.

WYDAWNICTWO NAUKOWO-TECHNICZNE
Warszawa, Mokotowska 24

BIBLIOGRAFIA

1. Leon Łukasiewicz, Antoni Mazurkiewicz, SYSTEM AUTOMATYCZNEGO KODOWANIA SAKO, Prace Zakładu Aparatów Matematycznych PAN, Warszawa 1961.
2. Konrad Fiałkowski, Jerzy Świaniewicz, MASZYNA ZAM-2. OPIS MASZYNY, KOMPENDIUM PROGRAMOWANIA W JEZYKU SAS, Prace Zakładu Aparatów Matematycznych PAN, Warszawa 1962.
3. D. D. McCracken, PROGRAMOWANIE MASZYN CYFROWYCH, PWN, Warszawa 1962.
4. R. K. Richards, ARITHMETIC OPERATIONS IN DIGITAL COMPUTERS, D. Van Nostrand Company, New York 1955.
5. A. Kitow, ELEKTRONICZNE MASZYNY CYFROWE, Wydawnictwo MON, Warszawa 1959.
6. Sbornik pieriewodow pod redakcją A. P. Jerszowa AWTO-MATIZACIJA PROGRAMIROWANIA, gosudarstwennoje izdatielstwo fiziko-matematicheskoy literatury, Moskwa 1961.
7. A. B. Empacher, MASZYNY LICZĄ SAME?, Wiedza Powszechna, Warszawa 1960.

Konopinski T., Łachinski J.
ELEKTRONICZNE STEROWNIKI MOCY

str. 382, rys. 102, tabl. 24, 25.

Do nabycia w księgarniach „Dziennik Książkowy”

WYDAWNICTWA NAUKOWO - TECHNICZNE
Warszawa, Mazowiecka 2/4

polecają:

Hunter L. P.

ELEKTRONIKA PÓLPRZEWODNIKOWA

str. 732, rys. 504, tabl. 40, zł 98.—

Jonscher A. K.

**PODSTAWY DZIAŁANIA PRZYRZĄDÓW
PÓLPRZEWODNIKOWYCH**

Tłum. z ang. (USA) W. ROSIŃSKI.

str. 220, rys. 58, tabl. 4, zł 25.—

Baranowski J., Jankowski T.

TRANZYSTOROWE UKŁADY IMPULSOWE

str. 265, rys. 168, tabl. 1, zł 27.—

Konopiński T., Luciński J.

ELEKTRONICZNE STEROWNIKI MOCY

str. 252, rys. 192, tabl. 24, zł 25.—

Do nabycia w księgarniach „Domu Książki”