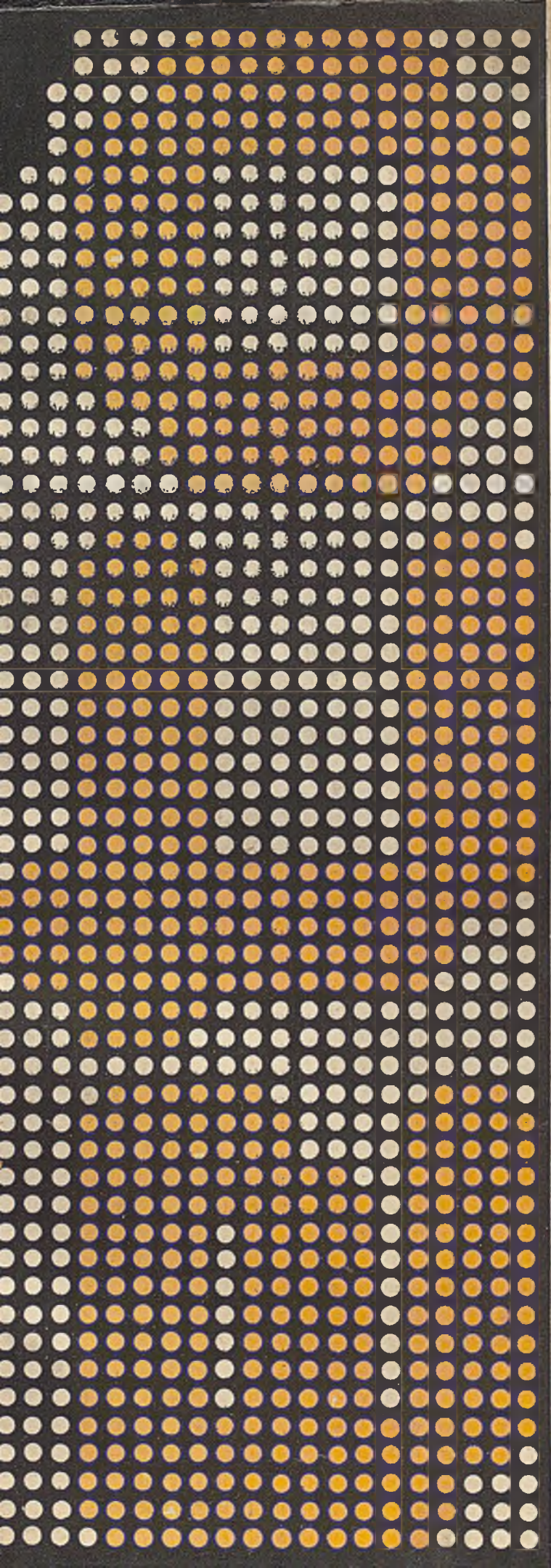


elektroniczna technika obliczeniowa

NOWOŚCI NR 1 1975

ZJEDNOCZENIE
PRZEMYSŁU
AUTOMATYKI
I APARATURY
POMIAROWEJ „MERA"

INSTYTUT MASZYN
MATEMATYCZNYCH
BRANŻOWY
OŚRODEK INTE

Z A W I A D O M I E N I E

Branżowy Ośrodek INTE Instytutu Maszyn Matematycznych zawiadamia, że nakładem IMM ukazały się następujące wydawnictwa przeznaczone dla użytkowników maszyny ZAM 41:

1/ "Kompendium oprogramowania" - z tzw. małej serii oprogramowania - zawierające wyczerpujące informacje dotyczące maszyny ZAM 41, niezbędne w pracy programistów oraz operatorów a także przydatne dla projektantów systemów; cena 1 egzemplarza wynosi 45 zł;

2/ "Maszyna ZAM 41" - t.1. System operacyjny 141 - z tzw. dużej serii oprogramowania; cena 1 egzemplarza wynosi 100 zł.
Tom zawiera opisy rozkazów maszyny użytkowej, opis eksploatacji maszyny, opis języka operacyjnego maszyny oraz opis systemu magazynowania i aktualizacji SMAD.

Ewentualne zamówienia podpisane przez dyrektora i głównego księgowego z podanym numerem konta bankowego prosimy kierować bezpośrednio na adres BOINTE IMM, ul.Krzywickiego 34, 02-078 Warszawa.

IMM z. 70/75 n. 2000

ELEKTRONICZNA TECHNIKA OBLICZENIOWA

N O W O Ś C I

Rok XIV Nr 1 1975

S p i s t r e ś c i

Wydaje

INSTYTUT MASZYN MATEMATYCZNYCH

B r a n ż o w y O ś r o d e k I n f o r m a c j i

N a u k o w e j T e c h n i c z n e j i E k o n o m i c z n e j

Adres redakcji: 09-078 Warszawa, ul. Krzywickiego 34
tel. 28-37-29 lub 21-84-41 w. 431

Druk IMM zam. 51/75 nakł. 1800 egz. pap. offset. kl. III g. 90. B-34.
INDEKS-35804/35709

ETO NOWOŚCI Nr1 /1975

mgr inż. Władysław GAJEWSKI
mgr inż. Edward LENARCZYK
mgr inż. Andrzej RADZIMIŃSKI
Instytut Łączności

681.322-181.4

KOMPUTER SYSTEM 10

1. Wstęp

Firma Singer Business Machines wyprodukowała wiele urządzeń komputerowych. Jednym z nich jest komputer o nazwie firmowej "System Ten" (System 10). Stanowi on ciekawe i niespotykane dotychczas w naszym kraju rozwiązanie konstrukcyjne, które zasługuje naszym zdaniem na przedstawienie czytelnikowi. Architekturę i oprogramowanie komputera "System 10" omówiono szczegółowo w dwóch kolejnych artykułach zawartych w dalszej części bieżącego numeru "ETO NOWOŚCI". Natomiast ten artykuł ma za zadanie ogólne przedstawienie komputera System 10 i przykładów jego zastosowań.

2. Producent

Singer jest wielonarodową firmą, której obrót przekroczył w 1971 r. kwotę 2 miliardów dolarów. Głównymi dziedzinami produkcyjnymi tej firmy są: aparatura nawigacyjna dla lotnictwa i żeglugi, symulatory lotów i kierowania pojazdami, aparatura kontrolno-pomiarowa dla radiokomunikacji, urządzenia do automatycznego nauczania, maszyny do szycia domowe i przemysłowe, meble, budownictwo mieszkaniowe, sprzęt informatyczny i maszyny biurowe, aparatura automatyki przemysłowej i in.

Jak widać jest to firma wielobranżowa. Komputer System 10 jest wynikiem poszukiwań sprzętu do automatyzacji systemu zarządzania własną siecią sprzedaży obejmującą kilka tysięcy punktów na świecie.

3. Ogólna charakterystyka komputera System 10

Jest to w zasadzie maszyna specjalizowana do organizacji banków danych na dyskach magnetycznych i przetwarzania danych typu buchalteryjnego, jakkolwiek możliwe jest również zastosowanie jej do innych celów.

Podstawową zaletą komputera System 10 jest jego modularność. Jednostka centralna ma pamięć operacyjną o pojemności od 10 do 110 tys. znaków. Zwiększenie pojemności pamięci może nastąpić przez kolejne dodawanie pakietów pamięci o pojemności 10 tysięcy znaków.

Podział pamięci na sekcje umożliwia realizowanie do 20 autonomicznych procesów przetwarzaniowych bazujących na wspólnych lub niezależnych zbiorach na dyskach lub taśmach magnetycznych.

Rozbudowane układy wejścia/wyjścia pozwalają dołączyć do jednostki centralnej, za pośrednictwem do 20 kanałów, 200 wolnych urządzeń zewnętrznych. Przez kanał szybkich urządzeń zewnętrznych można dołączyć do procesora do 10 jednostek dyskowych oraz do 4 jednostek taśmy magnetycznej.

Realizację wieloprogramowości w komputerze System 10 zapewnia sterowanie układowe (hardware control). Czynności sterujące w maszynie wykonywane są przez układy elektroniczne. Maszyna nie potrzebuje więc programu nadrzędnego typu executive, jak np. w komputerach serii ODRA 1300. Przyczynia się to do zwiększenia szybkości działania maszyny. Ponadto pamięć operacyjna oddana jest niemalże w całości do dyspozycji użytkowników.

Komputer System 10 jest maszyną dziesiętną kodowaną binarnie. Jest to maszyna trzeciej generacji, do jej wykonania użyto standardowych elementów scalonych firmy Texas Instruments o średniej i małej skali integracji. Maszyna pracuje w standardowym kodzie amerykańskim USASCII, a karty perforowane mają kod Holleritha.

Komunikowanie się operatora z programami bieżąco wykonywanymi w maszynie następuje za pośrednictwem konsoli operatorskiej lub monitora ekranowego. Ze względu na niezależną pracę programów w poszczególnych sekcjach pamięci, operator ma dostęp przez jedno urządzenie tylko do programu wykonywanego w danej sekcji.

Możliwe jest również wykorzystanie jednego monitora ekranowego w kilku sekcjach przez przydzielanie go za pośrednictwem przełącznika układowego kolejno z jednej sekcji do drugiej.

Dalekopisy komunikacyjne, monitory ekranowe i inne urządzenia zdalnego dostępu mogą być podłączane do komputera z dowolnej odległości za pomocą modemów i linii telefonicznych.

Interesującą, a zarazem dosyć reprezentatywną charakterystykę ogólną komputera System 10 podaje sam producent. Stwierdza on bowiem, że rozwój sprzętu informatyki na świecie pozostawił pewien typ przedsiębiorstw w trudnej sytuacji. Są to te przedsiębiorstwa, których obrót jest za duży na posiadanie tylko wyposażenia mechanicznego w zakresie przetwarzania danych, a zarazem za mały na wkroczenie do "towarzystwa" użytkowników elektronicznych komputerów na dużą skalę. Takie przedsiębiorstwa w wielu przypadkach boją się podejmowania prac nad budową systemów informatycznych głównie z następujących powodów:

- konieczne są zmiany organizacyjne, a zwłaszcza stworzenie nowego działu, tj. informatyki,
- specjaliści (programiści, analitycy i operatorzy) otrzymują pobory, które zakłócają normalną strukturę płac danego przedsiębiorstwa,
- niezbędne są znaczne zakupy sprzętu komputerowego stanowiącego nowość dotychczas nieznaną w przedsiębiorstwie oraz konieczne są duże nakłady na opracowanie systemów.

Odbiorcy ci potrzebują wobec tego komputera, który nie jest naprawdę komputerem. Potrzebują urządzenia, które wykonuje prace przedsiębiorstwa szybciej i sprawniej, informuje o wyczerpaniu się zapasów lub o zaległościach płatniczych klientów itd., potrzebują maszyny cyfrowej, która nie wymaga całej ekipy programistów, analityków i operatorów do obsługi, urządzenia, które może być ulokowane tam, gdzie praca tego wymaga i które może rosnąć razem z rozwojem przedsiębiorstwa, bez potrzeby zamiany na większą maszynę.

System 10 jest właściwym rozwiązaniem dla tego typu przedsiębiorstw. Może on wykonywać do 20 różnych zadań jednocześnie przez przydzielenie

jednej sekcji pamięci dla każdego zadania, nie wymaga stałej obecności operatora i jego programowanie jest proste w porównaniu z innymi komputerami. Wszystkie urządzenia razem lub osobno mają dostęp do zbiorów informacyjnych na dyskach lub taśmach magnetycznych. Poza tym System 10 jest w cenie, która mieści się w budżecie małego przedsiębiorstwa. Mimo niskiej ceny, System 10 dostarcza użytkownikowi ułatwień takich, jakie dają duże wieloprogramowe komputery III generacji z zsynchronizowanym przetwarzaniem.

Znając istniejącą w Stanach Zjednoczonych i w Europie sytuację w zakresie instalacji komputerów można stwierdzić, że większość użytkowników posiada komputer po raz pierwszy i chce osiągnąć szybki system niezależnej pracy. Każde przedsiębiorstwo załatwia wiele spraw jednocześnie, każda z nich stanowi potencjalne zadanie dla komputera, ale żadne z nich nie uzasadnia kupna komputera.

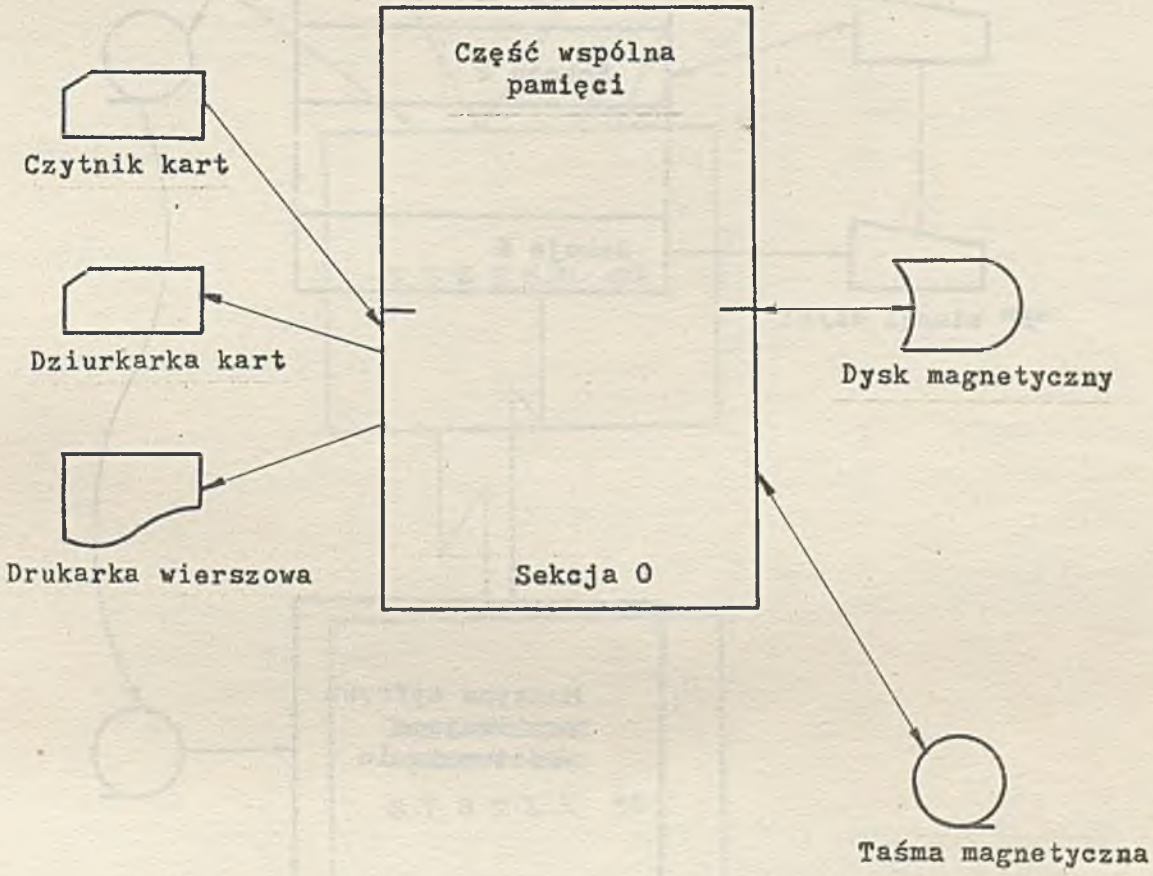
Specyficzna struktura Systemu 10 umożliwia wykonanie jednocześnie wszystkich zadań przy tej samej jednostce centralnej. W tej sytuacji każda funkcja ma jakby całą instalację dla siebie. Innymi słowy System 10 pracuje wieloprogramowo pozostając jednocześnie "małym" komputerem.

Powyższa charakterystyka, zawarta w materiałach producenta, wydaje się właściwym przedstawieniem ogólnym maszyny cyfrowej System 10. Należy zwrócić uwagę, że sam producent klasyfikuje swoją maszynę jako narzędzie pośrednie między urządzeniami wysokiego stopnia automatyzacji a maszyną cyfrową III generacji.

4. Przykłady zastosowań komputera System 10

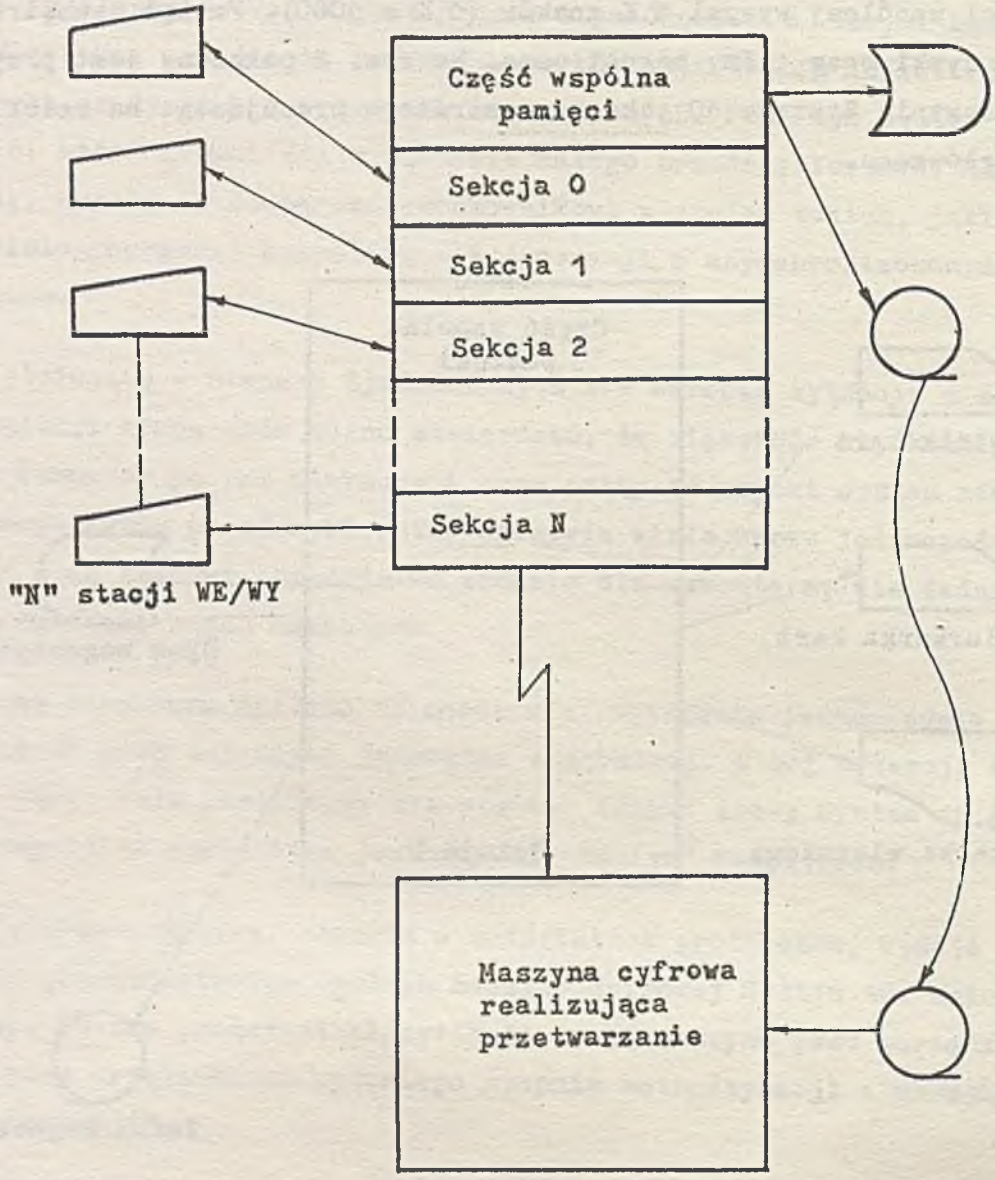
Najprostszy przykład zastosowania Systemu 10 w przetwarzaniu danych pokazany jest na rys. 1. Klasyczne urządzenia peryferyjne, jak np. czytnik kart, perforator kart i drukarka wierszowa dołączone są do jednostki centralnej, zawierającej co najmniej jedną sekcję pamięci oraz część wspólną pamięci. Część wspólna pamięci używana jest przez poszczególne sekcje pamięci do wzajemnego komunikowania się (nie ma innych możliwości wymienienia informacji w pamięci operacyjnej przez programy poszczególnych sekcji). Ponadto część wspólna używana jest przez

programy biblioteczne systemu współpracy z dyskami. Minimalny rozmiar pamięci wspólnej wynosi 1 K znaków (1 K = 1000). Pamięć zewnętrzną stanowią dyski oraz taśmy magnetyczne. Na rys. 2 pokazany jest przykład zastosowania Systemu 10 jako koncentratora pracującego na rzecz komputera głównego.



Rys. 1. Przykład konfiguracji do przetwarzania danych

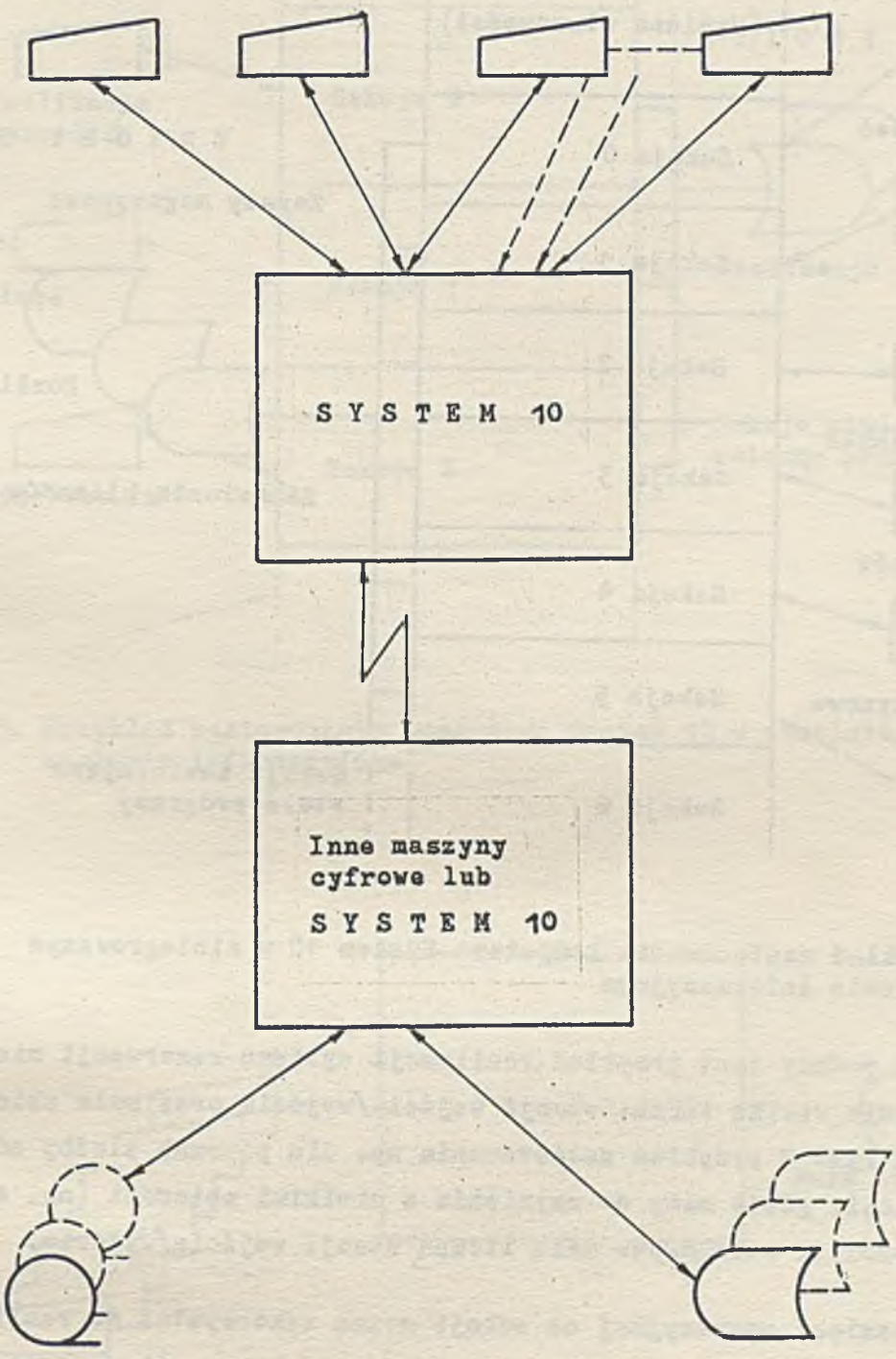
Dane wprowadzane z niezależnie pracujących stacji wejścia/wyjścia gromadzone są na dyskach lub taśmach magnetycznych i przekazywane następnie w systemie integralnym (on-line) lub autonomicznym (off-line) do komputera głównego w celu ich przetworzenia. Komputerem głównym może być komputer System 10 lub dowolna inna maszyna cyfrowa. Zastosowanie - o podobnym charakterze, z tą różnicą, że możliwe jest prowadzenie pracy dialogowej, pokazane na rys. 3, gdzie System 10 pełni funkcję komputera buforowego.



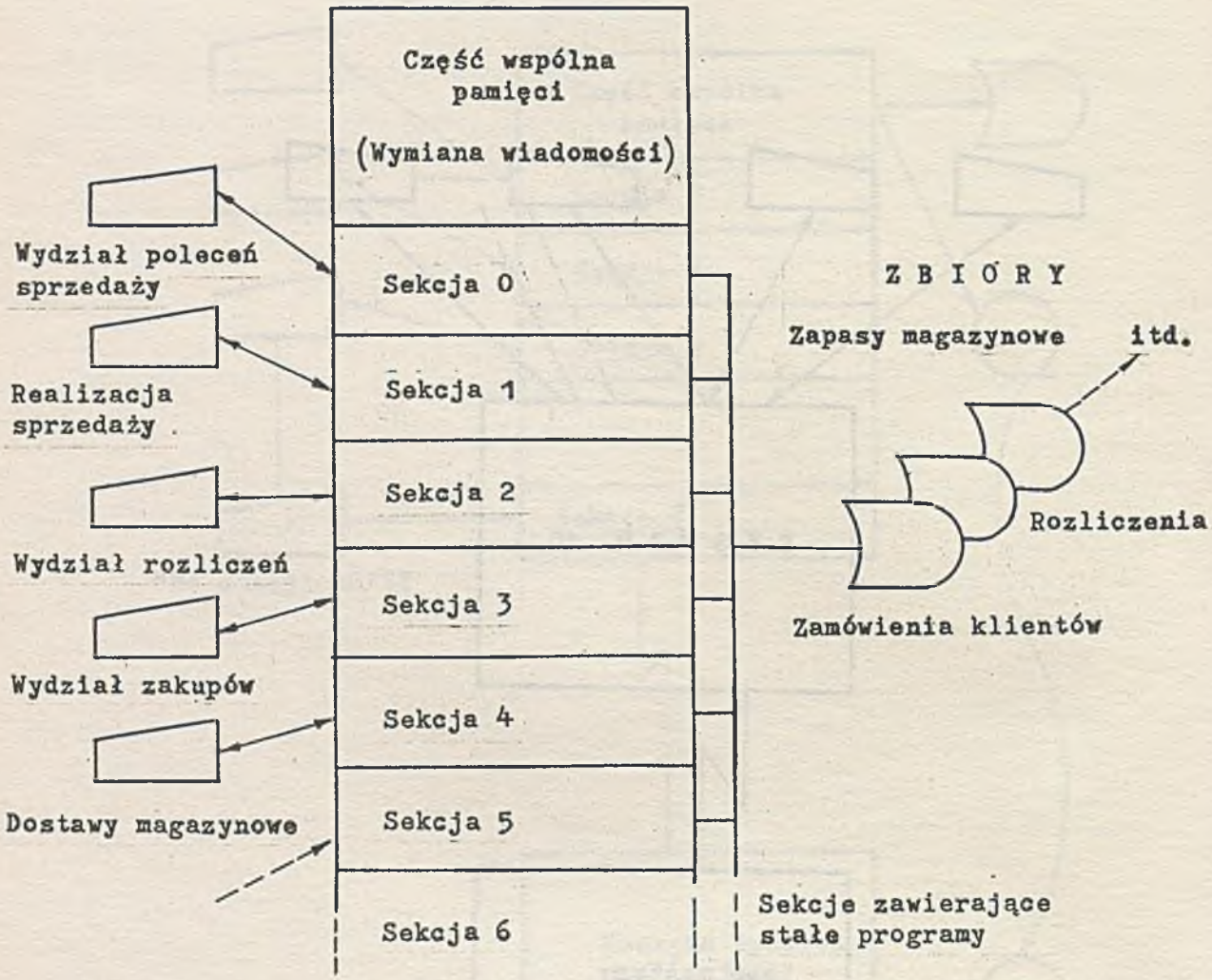
Rys. 2. Zastosowanie komputera System 10 jako koncentratora

Na rys. 4 pokazany jest przykład realizacji opartego na Systemie 10 - zintegrowanego systemu informacyjnego do zarządzania przedsiębiorstwem handlowym. Liczne programy pracujące na rzecz poszczególnych wydziałów przedsiębiorstwa operują na wspólnych zbiorach dyskowych.

Na rys. 5 ten sam przykład zrealizowany jest systemem niezintegrowanym. Różnice między przykładem na rys. 4 i 5 polegają na umieszczeniu w poszczególnych sekcjach pamięci stałych lub zmiennych programów operacyjnych banku danych zorganizowanego na dyskach magnetycznych.



Rys. 3. Zastosowanie Systemu 10 jako komputera buforowego

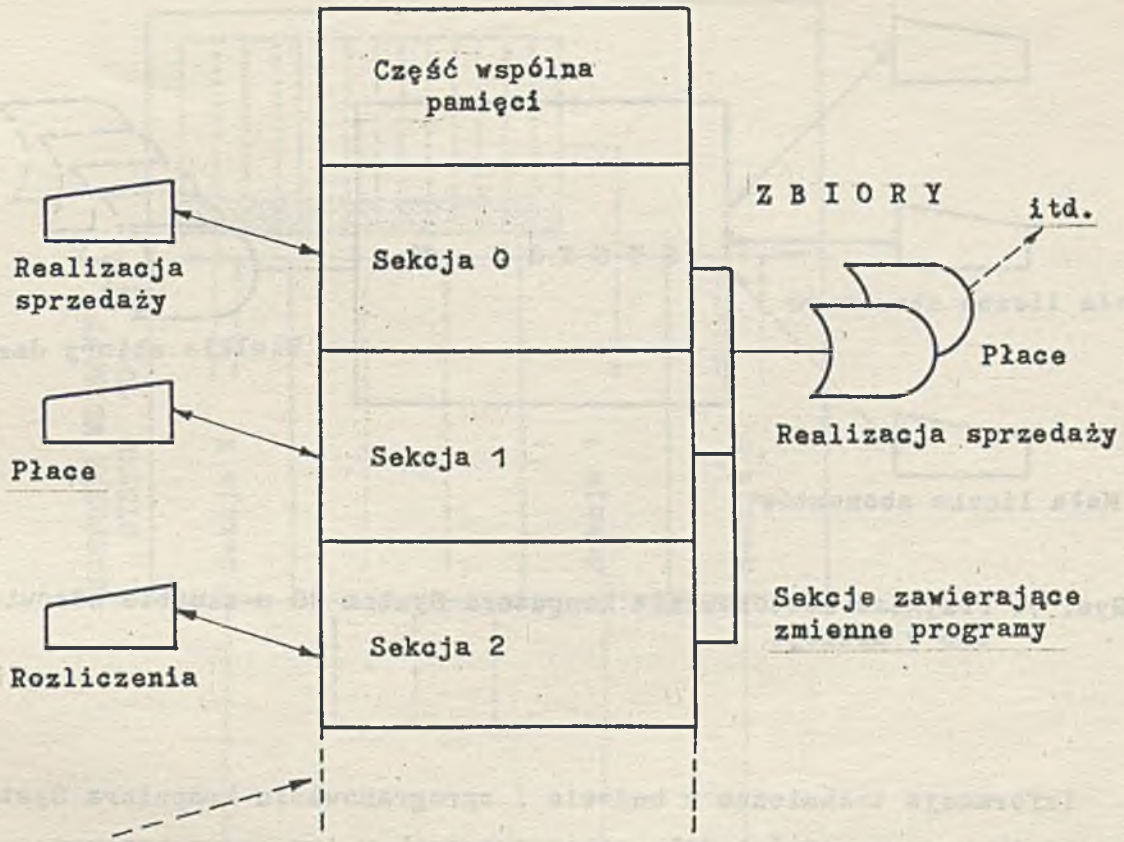


Rys. 4. Przykład zastosowania komputera System 10 w zintegrowanym systemie informacyjnym

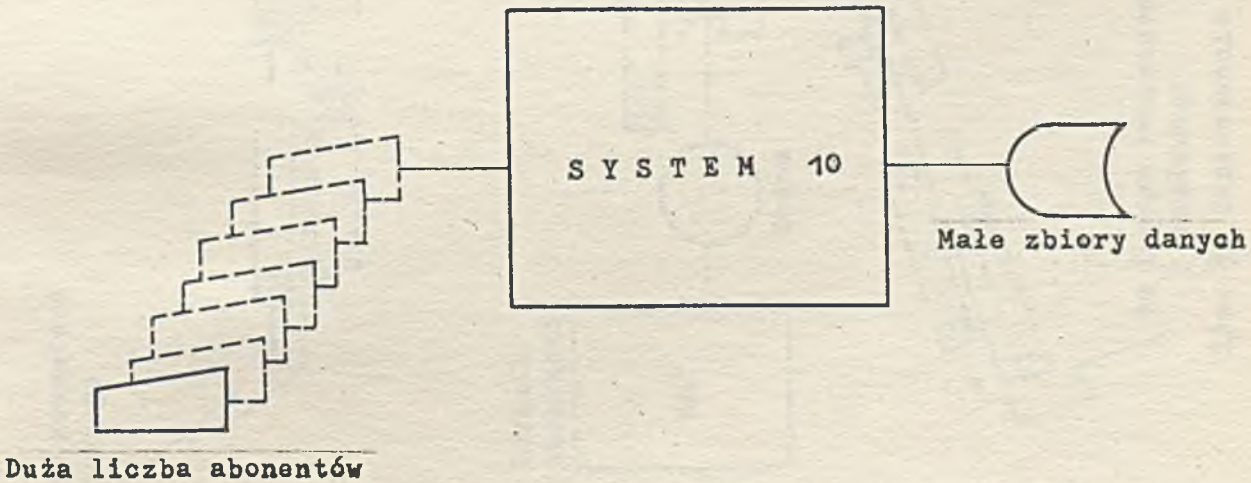
Na rys. 6 podany jest przykład realizacji systemu rezerwacji miejsc, gdzie występuje wielka liczba stacji wejścia/wyjścia oraz małe zbiory danych, a na rys. 7 przykład zastosowania np. dla potrzeb służby zdrowia lub milicji, gdzie mamy do czynienia z wielkimi zbiorami (np. ewidencja ludności) i stosunkowo małą liczbą stacji wejścia/wyjścia.

Podział pamięci operacyjnej na sekcje można wykorzystać do realizacji komutacji pakietów wiadomości w zastosowaniach komunikacyjnych.

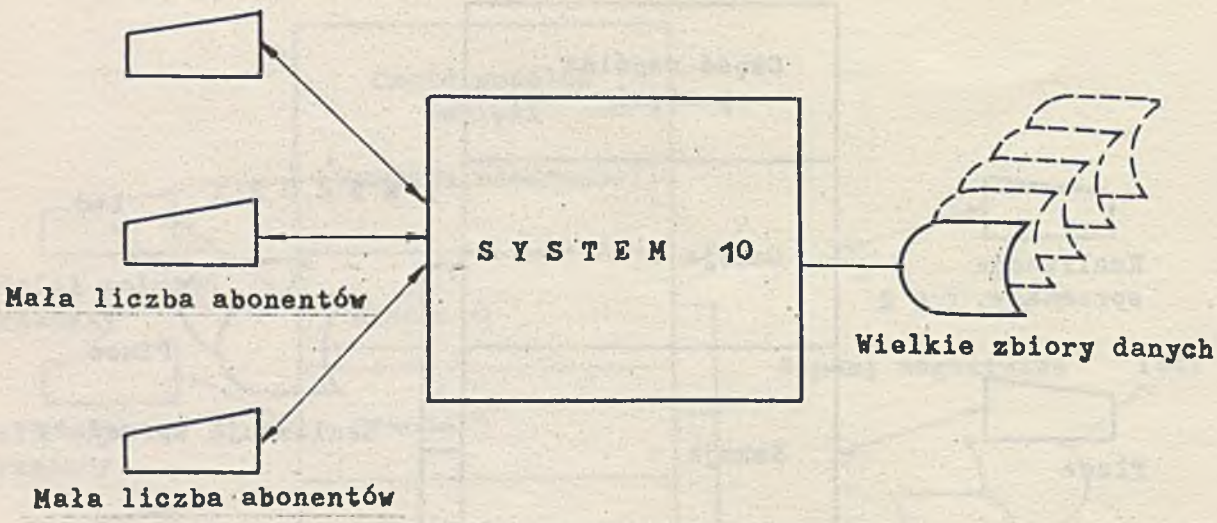
Na rys. 8 pokazany jest schemat takiego zastosowania. Bloki wiadomości otrzymane z innego komputera lub stacji zdalnego dostępu dzielone są na pakiety o standardowym formacie i wysyłane według przeznaczenia.



Rys. 5. Przykład zastosowania komputera System 10 w niezintegrowanym systemie informacyjnym

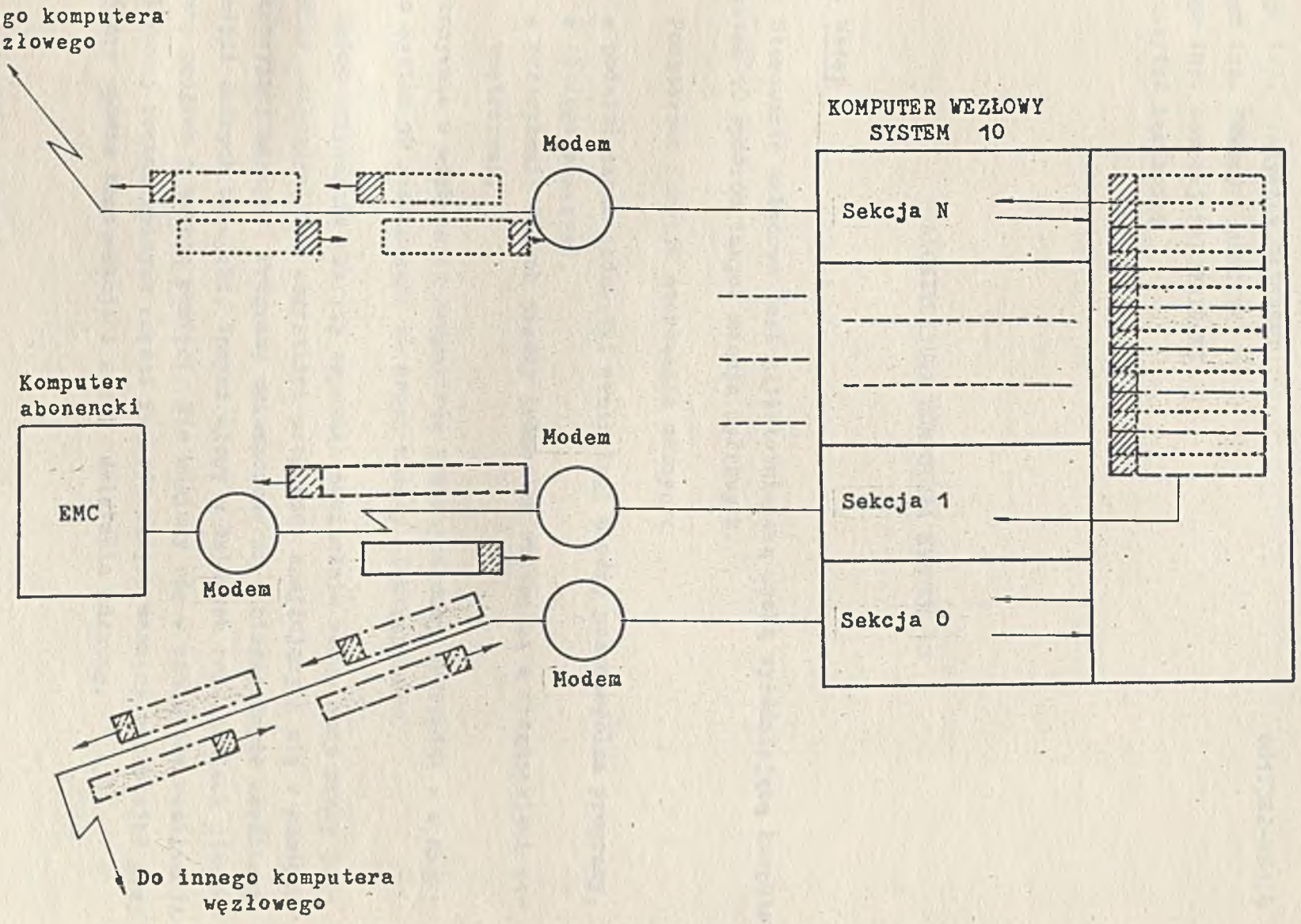


Rys. 6. Przykład zastosowania komputera System 10 w systemie rezerwacji miejsc

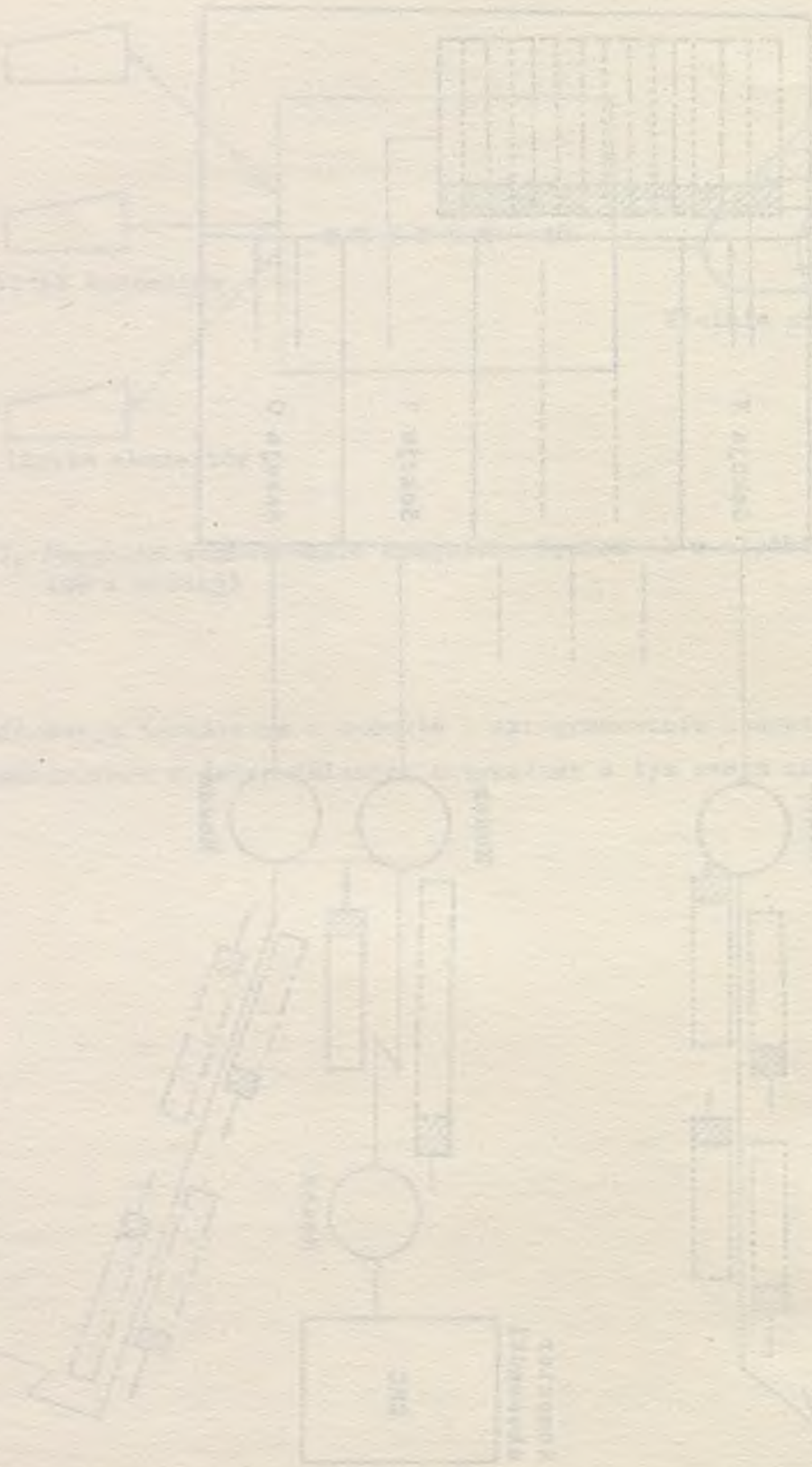


Rys. 7. Przykład zastosowania komputera System 10 w służbie zdrowia lub w milicji

Informacje techniczne o budowie i oprogramowaniu komputera System 10 są zamieszczone w dwóch dalszych artykułach w tym samym numerze.



Rys. 8. Zastosowanie komputera System 10 do komutacji pakietów wiadomości



mgr inż. Władysław GAJEWSKI
mgr inż. Edward LENARCZYK
mgr inż. Andrzej RADZIMIŃSKI
Instytut Łączności

681.322-181.4

ARCHITEKTURA KOMPUTERA SYSTEM 10

1. Wstęp

Sterowanie układowe jest najistotniejszą cechą wyróżniającą komputer System 10 spośród innych maszyn cyfrowych.

Podstawowe funkcje sterowania maszyny:

- podział czasu jednostki centralnej między poszczególne programy,
- obsługa przerwań,
- przesyłanie danych między jednostką centralną a urządzeniami zewnętrznymi,

wykonywane w większości komputerów przez programy nadrzędne - w maszynie System 10 realizowane są przez układy elektroniczne.

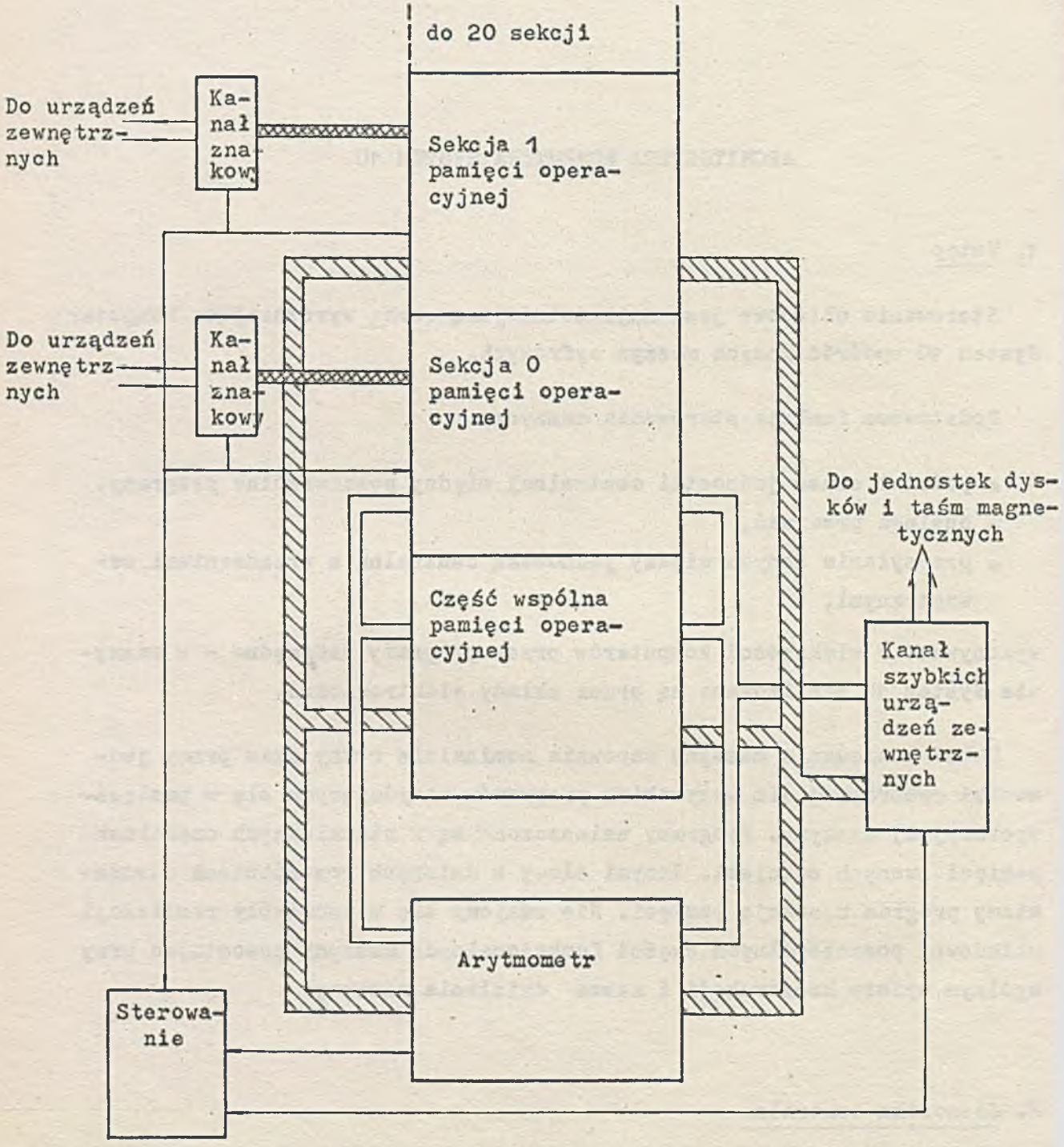
Układ sterowania maszyny zapewnia nominalnie równy czas pracy jednostki centralnej dla wszystkich programów znajdujących się w pamięci operacyjnej maszyny. Programy umieszczone są w niezależnych częściach pamięci zwanych sekcjami. Innymi słowy w dalszych rozważaniach utożsamiamy program z sekcją pamięci. Nie wdajemy się w szczegóły realizacji układowej poszczególnych części funkcjonalnych maszyny pozostając przy ogólnym opisie konstrukcji i zasad działania maszyny.

2. Jednostka centralna

Jednostka centralna komputera System 10 zrealizowana została w dwóch wersjach: Model 20 i Model 21. Różnice między nimi są niewielkie

i sprowadzają się w zasadzie do bardziej elastycznego podziału pamięci operacyjnej w Modelu 21, który jest konstrukcją nowszą i bardziej uniwersalną.

Schemat organizacyjny jednostki centralnej przedstawia rys. 1.



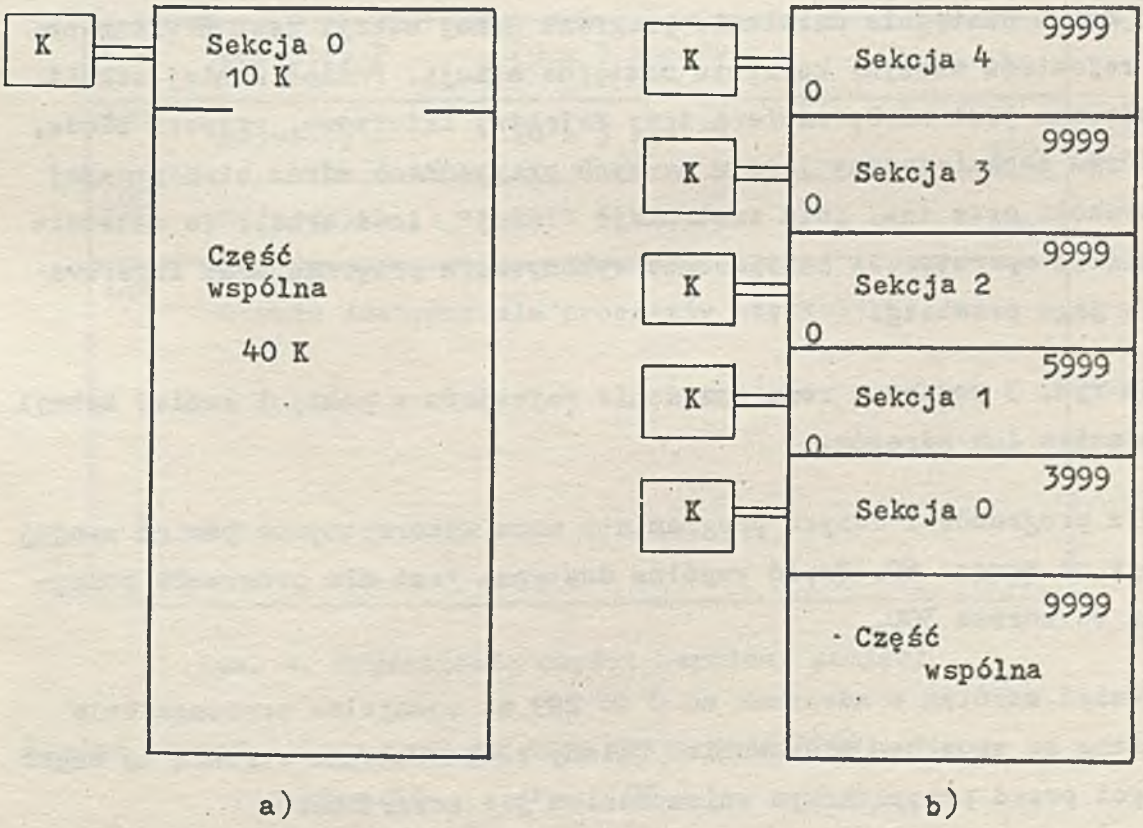
Rys. 1. Schemat jednostki centralnej komputera System 10

W skład jednostki centralnej wchodzą następujące części:

- pamięć operacyjna,
- arytmometr,
- układ sterowania oraz
- kanały wolnych i szybkich urządzeń zewnętrznych.

Kanały urządzeń wejścia/wyjścia oraz zasady sterowania omówione zostaną w dalszej części artykułu, tu natomiast zajmiemy się opisem pamięci operacyjnej, której organizacja jest niestandardowa.

Pamięć operacyjna komputera System 10 jest pamięcią rdzeniową o czasie cyklu 3,3 μs. Szerokość słowa pamięci wynosi 1 znak, to znaczy w jednym cyklu zapisanych i odczytanych jest jednocześnie 6 bitów. Firma dostarcza moduły pamięci o pojemności 10 K znaków (1K = 1000). Maksymalna pojemność pamięci wynosi 110 K a więc 11 modułów. Modularna struktura pamięci nie ma nic wspólnego z logicznym czy też programowym jej podziałem zaznaczonym na rys. 2.



Rys. 2. Organizacja pamięci operacyjnej

a) jeden program wykorzystujący całą pamięć, b) pięć programów korzystających z danych i podprogramów umieszczonych w części wspólnej

Pod względem logicznym pamięć dzieli się na sekcje (partitions) i część wspólną (common). Liczba sekcji jest zmienna, z tym, że musi istnieć przynajmniej jedna sekcja a maksymalnie może ich być 20. Pojemność pamięci każdej sekcji może wynosić od 1K do 10K znaków z możliwością zmiany co 1K. Sekcje są ponumerowane od 0 do 19. Część wspólna pamięci musi mieć pojemność minimum 1K znaków. Maksymalna teoretyczna pojemność części wspólnej 80K znaków. Pojemność pamięci wspólnej może być zmieniana również co 1K. Zmiany konfiguracji pamięci: dodanie fizycznego modułu, przydzielenie pamięci dla nowej sekcji, zmiana pojemności sekcji i części wspólnej wykonywane są w sposób bardzo prosty przez operatora, co daje możliwość różnorodnego wykorzystania systemu w zależności od bieżących potrzeb. Na rys. 2 podane są dwa przykłady organizacji pamięci o łącznej pojemności 50K znaków.

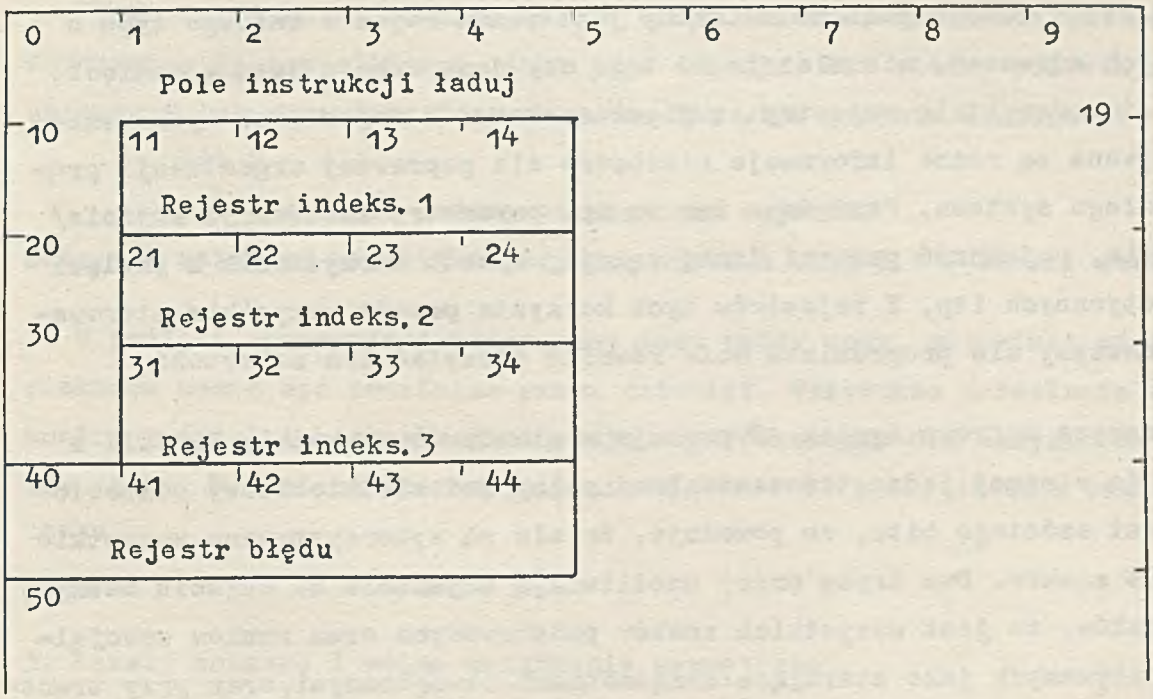
Każda sekcja pamięci jest niezależna, tzn. ma własne urządzenia zewnętrzne: dane i programy umieszczane w jej pamięci są niedostępne dla innych sekcji. Każda sekcja pamięci wykorzystuje układy sterowania i arytmometr jednostki centralnej przez określony czas (nominalnie 37,5 ms) a następnie działanie programu danej sekcji jest zawieszane, a z rejestrów maszyny korzysta następna sekcja. Pamięć każdej sekcji adresowana jest od 0, zawiera trzy rejestry indeksowe, rejestr błędu, w którym zapamiętywany jest w pewnych przypadkach adres niepoprawnej instrukcji oraz tzw. pole instrukcji "ładuj" (load area). To ostatnie umożliwia operatorowi inicjowanie wykonywania programu oraz ingerowanie w jego przebieg.

Na rys. 3 pokazano rozmieszczenie rejestrów w pamięci każdej sekcji z podaniem ich adresów.

Dla programów i danych programista może wykorzystywać pamięć swojej sekcji od adresu 50. Część wspólna dostępna jest dla programów poczynając od adresu 300.

Pamięć wspólna o adresach od 0 do 299 ma specjalne przeznaczenie związane ze sposobem sterowania. Układy elektroniczne chronią tę część pamięci przed przypadkowym zniszczeniem jej zawartości.

Organizację części wspólnej pamięci operacyjnej przedstawia rys. 4.



Rys. 3. Organizacja sekcji pamięci

0	Rejestry P	20 x 5 znaków	99
100	Rejestry B	20 x 5 znaków	199
200	Rejestry A	20 x 5 znaków	299
300	Obszar dostępny dla programów użytkowych		9999

Rys. 4. Organizacja części wspólnej pamięci

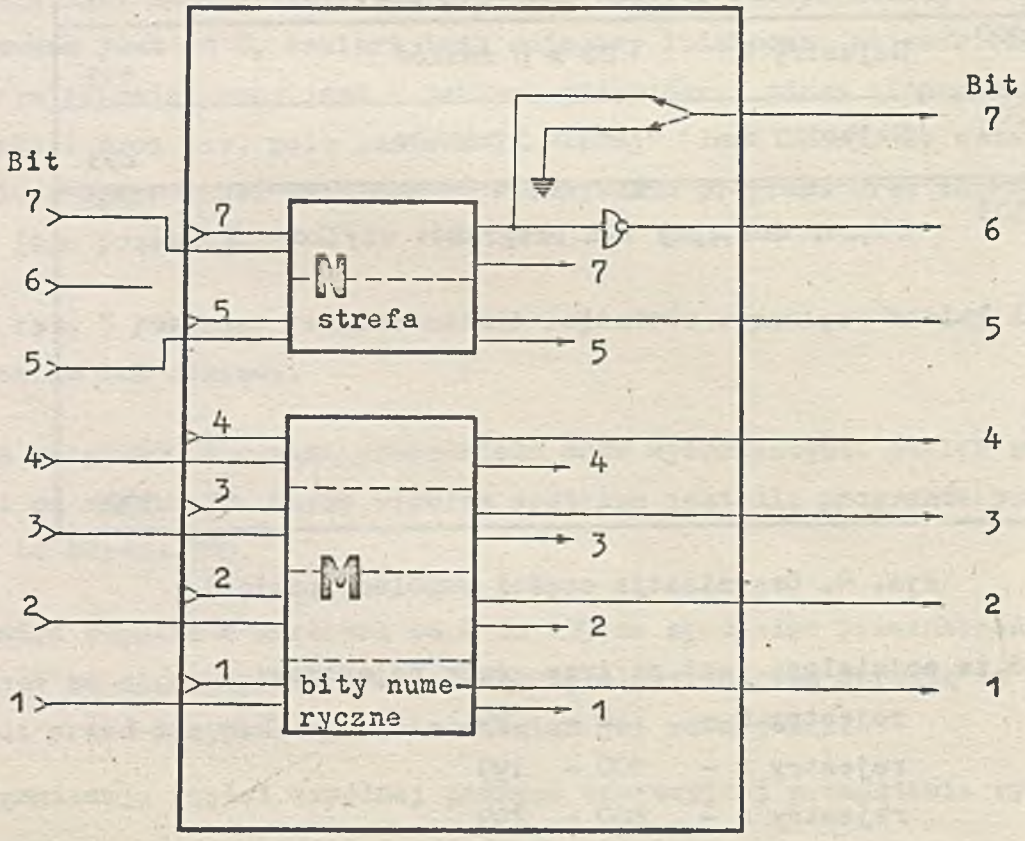
Pamięć ta podzielona jest na trzy grupy rejestrów:

rejestry P - 0 - 99
rejestry B - 100 - 199
rejestry A - 200 - 299

Z każdą sekcją pamięci związany jest jeden rejestr każdego typu o stałych adresach, niezależnie od tego czy dana sekcja jest w pamięci, czy nie. Wszystkie rejestry są pięcioznakowe. W rejestrach tych przechowywane są różne informacje niezbędne dla poprawnej organizacji pracy całego systemu. Pamiętane tam są np. parametry instrukcji wejścia/wyjścia, pojemność pamięci danej sekcji, sposób korzystania z pamięci magnetycznych itp. Z rejestrów tych korzysta przede wszystkim sterowanie maszyny ale programista może również odczytać ich zawartość.

Maszyna cyfrowa System 10 pracuje w standardowym kodzie USASCII z tym, że w samej jednostce centralnej pełny kod siedmiobitowy pozbawiony jest szóstego bitu, co powoduje, że nie są wykorzystywane wszystkie ze 128 znaków. Dwa tryby pracy umożliwiają uzyskanie na wyjściu maszyny 96 znaków, to jest wszystkich znaków podstawowych oraz znaków specjalnych używanych jako sterujące urządzeniami zewnętrznymi, oraz przy transmisji danych.

Rys. 5 pokazuje zasadę konwersji kodu zewnętrznego na wewnętrzny oraz przekształcenia odwrotnego.



Rys. 5. Konwersja kodów na wejściu i wyjściu procesora

Pięć znaków mniej znaczących nie ulega zmianie, natomiast bit szósty odtwarzany jest z siódmego, jako jego negacja. Bit siódmy pozostaje niezmieniony lub ustawiany jest na zero. Tak więc jedynie kombinacji

bit 7 = 1

bit 6 = 1 nie możemy otrzymać na wyjściu.

Bity 1-4 nazywamy częścią numeryczną znaku a bity 5 i 7 bitami strefy.

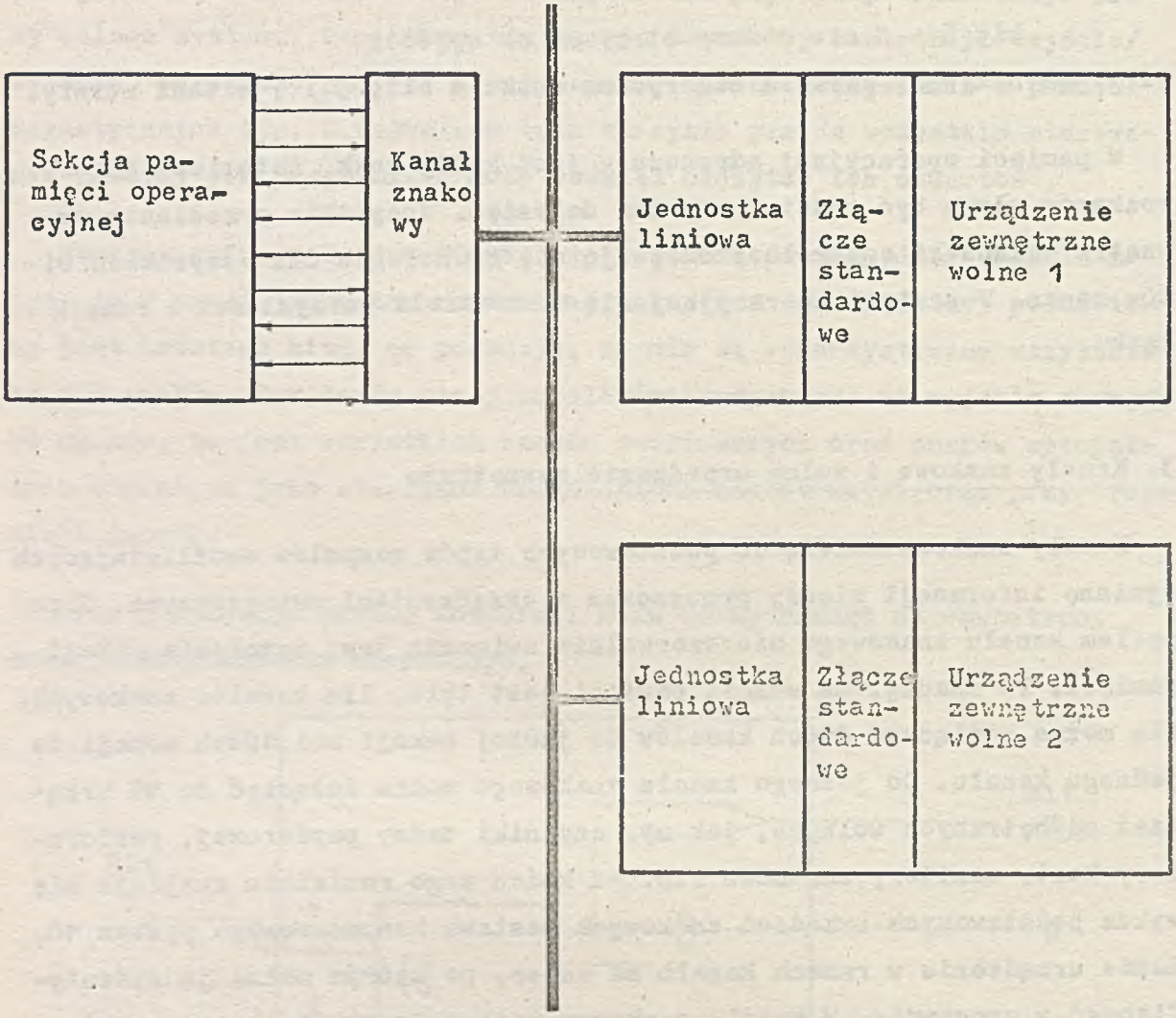
W pamięci operacyjnej adresowany jest każdy znak, natomiast adresy rozkazów muszą być podzielne przez dziesięć. Wszystkie przesłania wewnątrz jednostki centralnej odbywają się równolegle dla wszystkich bitów znaku. W pamięci operacyjnej nie ma kontroli parzystości w ramach znaku.

3. Kanały znakowe i wolne urządzenia zewnętrzne

Kanały znakowe należą do podstawowych typów zespołów umożliwiających wymianę informacji między procesorem a urządzeniami zewnętrznymi. Z pojęciem kanału znakowego nierozerwalnie związane jest istnienie sekcji pamięci. To znaczy, że sekcji pamięci jest tyle, ile kanałów znakowych. Nie można podłączyć dwóch kanałów do jednej sekcji ani dwóch sekcji do jednego kanału. Do jednego kanału znakowego można dołączyć do 10 urządzeń zewnętrznych wolnych, jak np. czytniki taśmy papierowej, perforatory kart, monitory ekranowe itp. Na końcu tego rozdziału znajduje się wykaz podstawowych urządzeń znakowych zestawu komputerowego System 10. Każde urządzenie w ramach kanału ma numer, po którym można je zidentyfikować w programie. W kanale znakowym może w danej chwili pracować tylko jedno urządzenie zewnętrzne. W całym systemie może pracować jednocześnie tyle urządzeń znakowych, ile jest kanałów. Transmisja w kanale odbywa się systemem znakowym (byte mode transmission) na zasadzie przerwań. W każdym kanale musi być jedno urządzenie konwersacyjne lub wejściowe z przyporządkowanym numerem zerowym. Jest to urządzenie wyróżnione układowo i tylko za jego pośrednictwem operator może komunikować się z maszyną.

Urządzenia zewnętrzne połączone są z kanałem za pomocą tylko dwóch przewodów. Taki sposób podłączenia umożliwia instalację urządzeń zewnętrznych w znacznej odległości od jednostki centralnej. W Systemie 10 urządzenia znakowe można podłączyć za pomocą kabla na odległość do ok.

600 m. Łatwiej jest również zrealizować połączenie na większe odległości za pomocą modemów i linii telefonicznej. Całość połączenia procesora z wolnymi urządzeniami zewnętrznymi przedstawia rys. 6.



Rys. 6. Schemat połączenia jednostki centralnej z wolnymi urządzeniami zewnętrznymi

Kanał znakowy zamienia równoległą wiązkę informacji przychodzącą z jednostki centralnej na szeregowy ciąg sygnałów wysyłanych w linię. Po stronie urządzenia zewnętrznego odwrotną rolę pełni jednostka liniowa (line unit) grupująca przychodzące szeregowo sygnały i wysyłająca je równolegle (wszystkie bity znaku oraz bity sterujące) do urządzenia zewnętrznego. Rola złącza standardowego (interface logic) sprowadza się do kontroli parzystości oraz buforowania informacji w celu dopasowania szybkości transmisji do szybkości zespołów urządzenia zewnętrznego. Jeżeli mamy do czynienia z urządzeniami

wejściowymi role jednostki liniowej i kanału zmieniają się. Jednostka liniowa przekształca równoległą informację np. z czytnika kart na szeregowy ciąg sygnałów, które są odebrane przez kanał i zamieniane na odpowiednią wiązkę równoległą. Kanał transmituje znak do pamięci operacyjnej przerywając na okres jednego cyklu wykonywanie bieżącego programu.

Do jednego kanału może być dołączonych nawet 10 urządzeń tego samego typu, z tym jednak, jak już zaznaczono wcześniej, że jedno z nich będzie układowo wyróżnione.

Tab. 1. Zestawienie podstawowych urządzeń zewnętrznych Systemu 10

Typ i nazwa	Opis, dane techniczne
Model 70 - Konsola operatorska (work-station)	Konwersacyjne urządzenie końcowe ze standardową klawiaturą alfanumeryczną oraz sterującą. Długość wiersza 175 znaków. Repertuar znaków 64. Maksymalna szybkość pisania 25 znaków/s
Model 80 - Monitor ekranowy (VDU)	Końcówka konwersacyjna ze standardową klawiaturą alfanumeryczną i sterującą. Pojemność ekranu 20 linii po 80 znaków w każdej linii. Repertuar znaków 53. Szybkość wyświetlania 1500 znaków/s
Model 35 - Perforator kart	Urządzenie wyjściowe perforujące karty 80-kolumnowe z szybkością maksymalną 100 kart/min
Model 30 - Czytnik kart	Urządzenie wejściowe czytające karty 80-kolumnowe w kodzie Holleritha z szybkością 300 kart/min
Model 60 - Czytnik taśmy papierowej	Urządzenie wejściowe czytające taśmę papierową 5, 6, 7 lub 8-ścieżkową z szybkością 275 znaków/s, możliwość automatycznego cofania taśmy
Model 65 - Perforator taśmy papierowej	Urządzenie wyjściowe perforujące taśmę 5, 6, 7 i 8-ścieżkową z szybkością maksymalną 150 znaków/s
Model 50 - Drukarka wierszowa	Urządzenie wyjściowe, długość linii 132 znaki. Repertuar znaków 64. Maksymalna szybkość drukowania 450 linii/min
Model 52 - Drukarka wierszowa	Urządzenie wyjściowe. Długość linii 132 znaki. Repertuar znaków 64. Szybkość drukowania 110 linii/min

Tab. 1 /c.d.

Typ i nazwa	Opis, dane techniczne
Model 900 - Końcowa stacja wprowadzania danych	Urządzenie wejściowe, wolnostojące "punkt sprzedaży". Programowana końcówka z klawiaturą numeryczną i 20 kluczami sterującymi. Ma 13 znaków, monitor ekranowy wyświetlający wszystkie wprowadzane liczby, sumy częściowe i ogólne. Wyposażony jest ponadto w minidrukarkę numeryczną o szybkości 35 znaków/s

4. Kanał szybkich urządzeń zewnętrznych

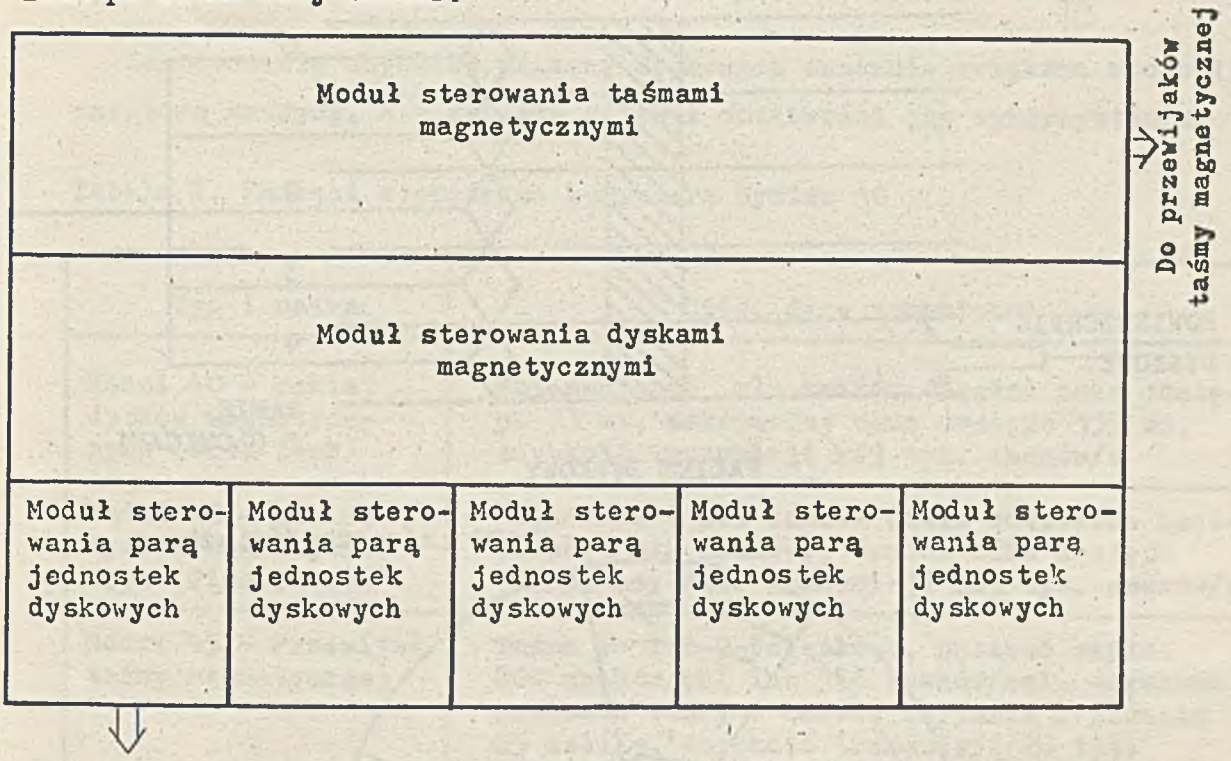
W odróżnieniu od kanałów znakowych, kanał szybkich urządzeń zewnętrznych jest tylko jeden dla całego zestawu maszyny cyfrowej System 10. Do kanału tego podłączyć można maksymalnie dziesięć jednostek dyskowych i cztery przewijaki taśmy magnetycznej. Jednostki dyskowe numerowane są od 0 do 9 i muszą mieć numery kolejne. Jeśli mamy trzy jednostki dyskowe to muszą one mieć numery 0, 1 i 2. Przewijaki taśmy magnetycznej mają numery od 1 do 4,przy czym kolejność nie jest tu istotna i mając dwa przewijaki można dać im numery np. 2 i 4. Zmiana numeru przewijaka jest bardzo łatwa (naciśnięcie klawisza) i należy tylko uważać, aby nie nadać dwóm przewijakom tego samego numeru.

Kanał urządzeń szybkich jest zbudowany modularnie tak, że odpowiednią konfigurację można dobrać w zależności od potrzeb. I tak np. jeśli nie przewidujemy zastosowania taśmy magnetycznej - nie instalujemy modułu sterowania taśmami magnetycznymi, jeśli mamy w zestawie tylko dwie jednostki dysków - instalujemy jedynie moduł sterowania dyskami magnetycznymi i jeden moduł sterowania parą jednostek dyskowych.

Rys. 7 przedstawia konfigurację kanału szybkich urządzeń zewnętrznych.

W kanale szybkim transmisja odbywa się na innej zasadzie niż w kanale znakowym. Jednostka centralna zaangażowana jest w tę transmisję na cały okres jej trwania, to jest przez czas przesyłania jednego bloku - w przypadku taśmy magnetycznej i jednego sektora w przypadku dysku. Współpraca jednostki dyskowej z procesorem odbywa się systemem grupo-

wym (burst mode transmission). Jednostka centralna wysyła lub przyjmuje znak po znaku cały sektor.

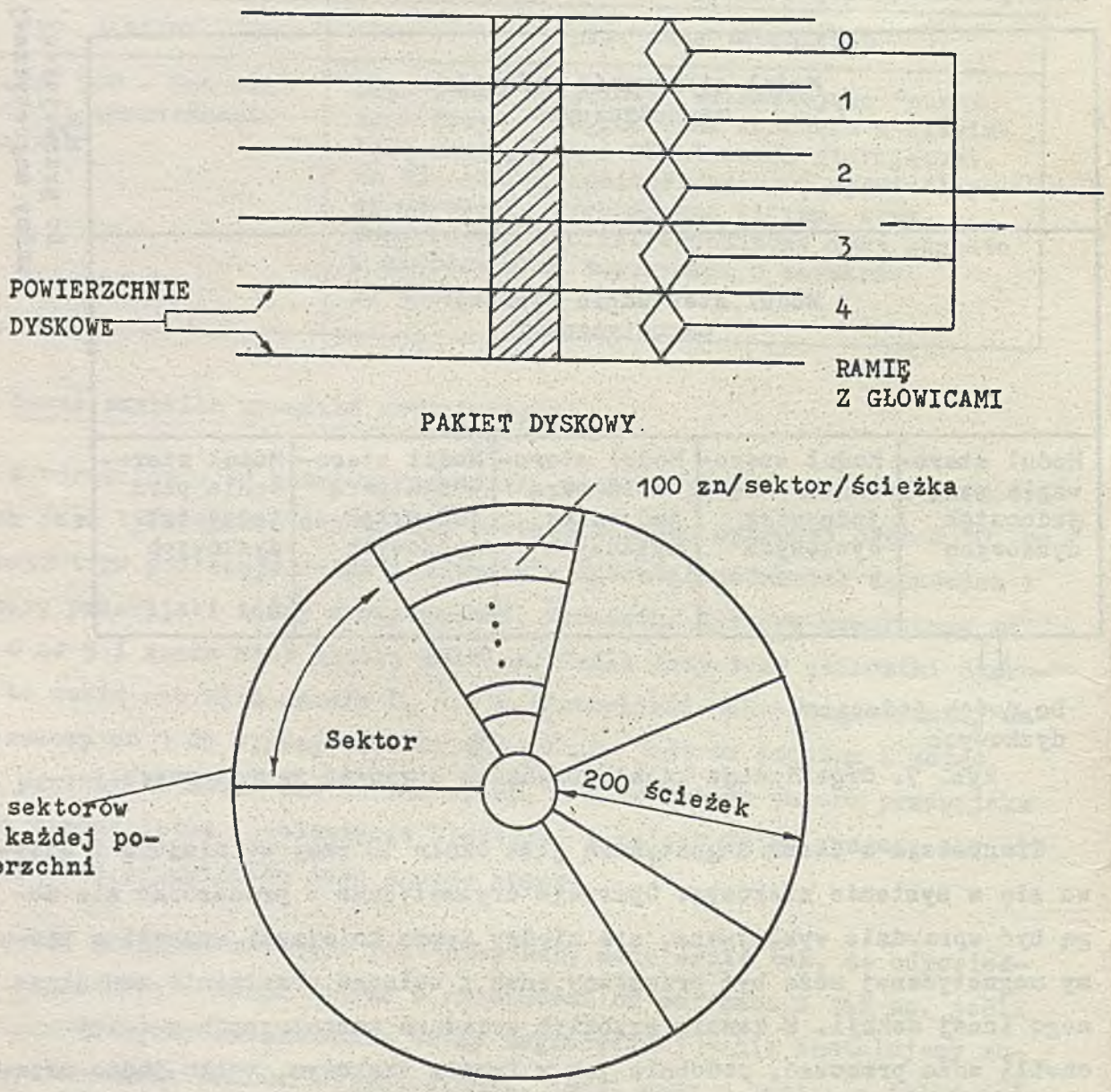


Rys. 7. Organizacja kanału szybkich urządzeń zewnętrznych

Transmisja z taśmą magnetyczną jest około 10 razy wolniejsza i odbywa się w systemie znakowym. Operacje arytmetyczne w procesorze nie mogą być wprawdzie wykonywane, ale między dwoma kolejnymi znakami z taśmy magnetycznej może być przesłany znak z wolnego urządzenia zewnętrznego innej sekcji. W kanale szybkich urządzeń zewnętrznych w danej chwili może pracować, podobnie jak w kanale znakowym, tylko jedno urządzenie. Istnieje możliwość zapisu i odczytu taśmy magnetycznej 7- i 9-ścieżkowej. Nie ma ograniczenia długości bloku na taśmie, chociaż zaleca się nie przekraczać 2048 znaków. Cały zestaw znaków sterujących pozwala dosyć wygodnie operować taśmą magnetyczną.

Podstawowe dane dotyczące gęstości zapisu i szybkości transmisji podano w tabeli nr 2.

Rys. 8 przedstawia organizację dysku magnetycznego dla jednostki dyskowej Model 40.



Rys. 8. Schemat organizacji dysku magnetycznego dla jednostki Model 40

Czytanie i pisanie odbywa się sektorami po 100 znaków. Przy właściwym zorganizowaniu zbioru na dyskach pamięć ta staje się bardzo szybka i może stanowić przedłużenie pamięci operacyjnej, gdyż większość podanego czasu dostępu stanowi ruch głowic wzdłuż promienia. Zapisana na dysku informacja podlega kontroli parzystości i poprawności zapisu przez ponowne jej odczytanie bezpośrednio po zapisie i porównanie z informa-

cją pierwotną. W przypadku niepoprawnego zapisu fakt ten jest sygnalizowany układowo i programista może wykorzystać tę informację.

Zastosowanie szybkich pamięci dyskowych znacznie zwiększa moc obliczeniową maszyny, a w związku z tym i możliwości jej wykorzystania.

Tabela 2. Pamięci zewnętrzne komputera System 10

Typ i nazwa	Opis, dane techniczne
Model 40 - pakiet dysków magnetycznych (disc pack)	Pojemność 10 mln znaków. Średni czas dostępu 73 ms, maksymalny czas dostępu 135 ms, szybkość transmisji 229 tys. znaków/s
Model 42 - pakiet dysków magnetycznych (disc pack)	Pojemność 8 mln znaków (dwie oddzielne części po 4 mln znaków), średni czas dostępu 73 ms, szybkość transmisji 229 tys. znaków/s
Model 45 - Przewijak taśmy magnetycznej	Taśma 7- lub 9-ścieżkowa, gęstość zapisu 800 znaków/cal lub 556 znaków/cal, szybkość przesuwu taśmy w czasie czytania i pisania 25 cali/s, szybkość transmisji 20 tys. znaków/s (dla 800 znaków)cal i 13900 znaków/s dla 556 cali/s, szybkość przewijania 150 cali/s. Możliwa jest układowa konwersja kodów z lub na ISO (ASCII) i EBCDIC

5. Układowe sterowanie maszyny

Jedną z największych osobliwości maszyny System 10 jest jej układ sterowania. Komputer nie ma żadnego programu nadrzędnego typu executive, jak np. elektroniczne maszyny cyfrowe z serii ODRA 1300. Sterowanie jest wyłączną domeną układów elektronicznych. Jest to tzw. sterowanie układowe (hardware control). Pod tym pojęciem rozumiane jest wykonywanie czynności sterujących w maszynie cyfrowej przez układy elektroniczne. Cały cykl rozkazowy realizowany jest na drodze układowej. Wszystkie instrukcje języka wewnętrznego wykonywane są bez udziału ekstrakodów.

Obsługa przerwań i realizacja wieloprogramowości w maszynie odbywa się przez układ sterujący.

Jak wiadomo komputer System 10 dopuszcza podział pamięci operacyjnej najwyżej na 20 sekcji oraz obszar pamięci wspólnej. W poszczególnych sekcjach znajdują się niezależne programy. Każda sekcja ma własny rejestr P, w którym zostaje zapamiętany adres następnej do wykonania instrukcji programu danej sekcji, w momencie przerwania. Realizacja programów odbywa się według systemu round-robin. Oznacza to, że sterowanie maszyny przydzielane jest dla poszczególnych sekcji pamięci operacyjnej średnio na jednakowy okres czasu - 37,5 ms. Po upływie tego czasu pierwsza instrukcja skoku w programie bieżąco wykonywanym powoduje przełączenie sekcji (partition switching) tzn. zapamiętanie stanu licznika rozkazów w rejestrze P oraz przekazanie sterowania do sekcji o numerze bezpośrednio większym.

Na podstawie rejestru P następnej sekcji zostaje odtworzona zawartość licznika rozkazów, a wykonywanie programu tej sekcji jest wznawiane od miejsca, w którym zostało przerwane poprzednio. Na okres kolejnych 37,5 ms sterowanie maszyny pozostaje przy następnym programie, a pierwsza instrukcja skoku, po upływie tego czasu dokonuje kolejnego przełączenia sekcji. Po wykonaniu obliczeń programu w sekcji 19 sterowanie przekazywane jest do sekcji 0.

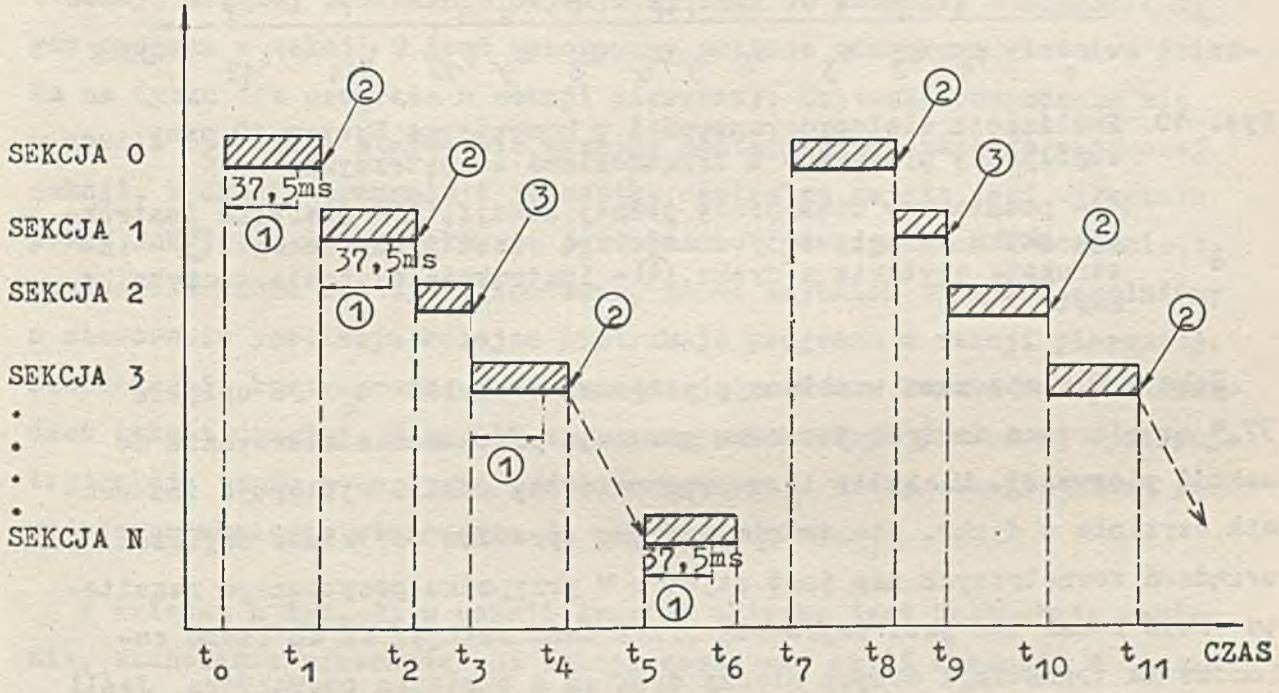
Istnieje również możliwość wcześniejszego przełączenia sekcji przez użycie w programie tzw. instrukcji skoku z przełączeniem (branch and switch). Po jej napotkaniu sterowanie maszyny przechodzi do realizacji instrukcji w sekcji o numerze bezpośrednio większym.

Dopuszcza się przypadek, w którym sekcja o numerze bezpośrednio większym nie występuje w danej konfiguracji pamięci operacyjnej. Układ sterowania zawsze zwiększa numer sekcji i sprawdza czy dana sekcja znajduje się w zestawie.

Przykład pracy wieloprogramowej maszyny System 10 podaje rys. 9.

Omawiany przykład zakłada istnienie w danej konfiguracji pamięci operacyjnej N sekcji ($N \leqslant 19$). W chwili t_0 rozpoczyna się realizacja programu w sekcji 0. Po upływie czasu $t_1 - t_0 \geqslant 37{,}5$ ms instrukcja skoku w chwili t_1 powoduje zawieszenie wykonywanego do tej pory programu i przekazanie sterowania do sekcji pierwszej. W chwili t_2 następuje zapa-

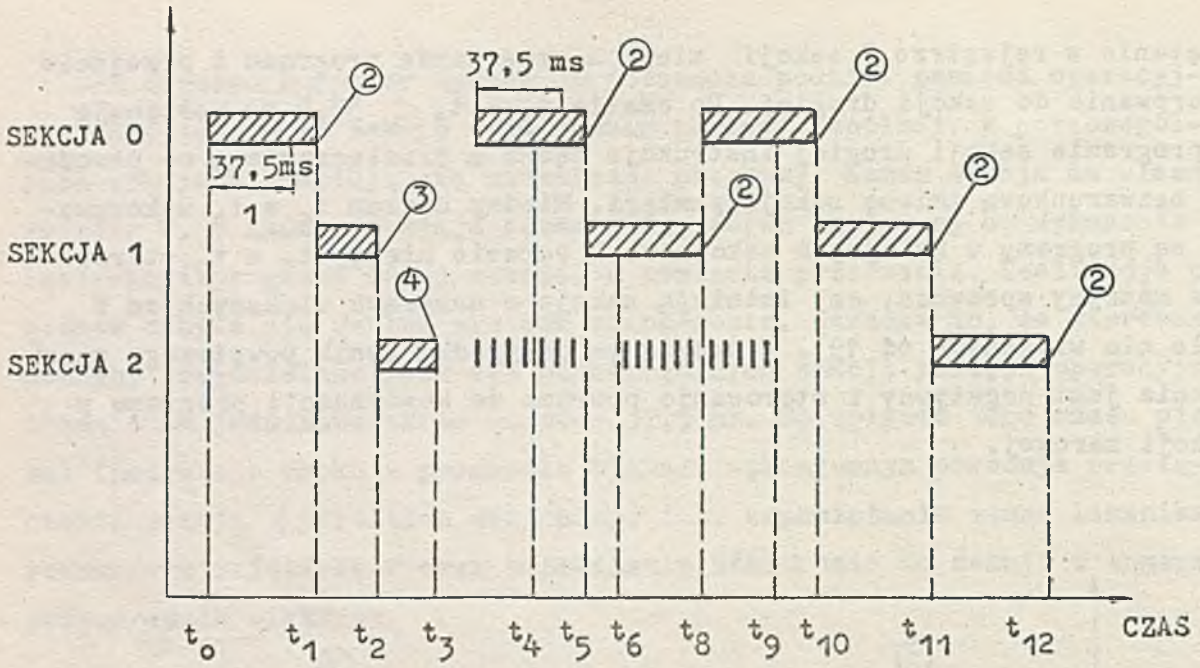
miętanie w rejestrze P sekcji miejsca przerwania programu i przejście sterowania do sekcji drugiej. Po czasie $t_3 - t_2 < 37{,}5$ ms występuje w programie sekcji drugiej instrukcja skoku z przełączaniem, co powoduje bezwarunkową zmianę sekcji pamięci. Między czasem t_5 a t_4 wykonywane są programy w kolejnych sekcjach. W okresie między t_6 a t_7 sterowanie maszyny sprawdza, czy istnieją sekcje o numerach większych od N (ale nie większych od 19). W omawianym przypadku wynik powyższego sprawdzania jest negatywny i sterowanie powraca do kontynuacji programu w sekcji zerowej.



Rys. 9. Realizacja wieloprogramowości w komputerze System 10

①- podstawowy czas pracy jednej sekcji, ②- pierwsza instrukcja skoku po upływie podstawowego czasu pracy sekcji, ③- instrukcja skoku z przełączaniem (branch and switch)

Powyższy przykład pomija współpracę poszczególnych programów z urządzeniami zewnętrznymi. Ilustracją przypadku wieloprogramowej pracy maszyny System 10, przy uwzględnieniu wymiany informacji między różnymi typami urządzeń we/wy a jednostką centralną, jest rys. 10.



Rys. 10. Realizacja wieloprogramowości w komputerze System 10 przy współpracy programów z urządzeniami zewnętrznymi

①- podstawowy czas pracy jednej sekcji, ② - pierwsza instrukcja skoku po upływie podstawowego czasu pracy sekcji, ③ - instrukcja czytania z dysku, ④ - instrukcja czytania z czytnika kart

Sekcja 0 rozpoczyna realizację programu w chwili t_0. Po upływie 37,5 ms pierwsza instrukcja skoku powoduje przekazanie sterowania do sekcji pierwszej. W chwili t_2 w programie tej sekcji występuje instrukcja czytania z dysku. Sterowanie maszyny sprawdza czy kanał szybkich urządzeń zewnętrznych nie jest zajęty. W przypadku pozytywnego rezultatu testu kanał ten jest zajmowany przez daną sekcję, aż do czasu zakończenia transmisji danych między dyskiem a pamięcią operacyjną. Jeśli kanał urządzeń szybkich jest zajęty w momencie, kiedy występuje w programie instrukcja odnosząca się do dysków lub taśmy magnetycznej, sterowanie maszyny przekazywane jest do następnej sekcji a test zajętości kanału zostaje powtórzony przy następnym dojściu do programu.

Omawiany przykład zakłada, że w czasie t_2 kanał urządzeń szybkich jest dostępny. Następnie sprawdza się czy głowice magnetyczne znajdują się na odpowiedniej ścieżce. Jeśli warunek ten nie jest spełniony - sterowanie maszyny przekazane jest do następnej sekcji (drugiej), a w czasie $t_4 - t_2$ następuje przesunięcie ramienia dysku na odpowiednią ścieżkę.

Gdyby głowice wskazywały żądaną w bieżącej instrukcji ścieżkę - sterowanie pozostałoby w danej sekcji i po przesunięciu dysku na właściwy sektor nastąpiłaby transmisja 100 znaków do pamięci operacyjnej. W chwili t_2 sterowanie przechodzi do sekcji drugiej. Rozpoczyna się realizacja programu tej sekcji, która zostaje przerwana w chwili napotkania instrukcji czytania, odnoszącej się do czytnika kart (chwila t_3).

Praca urządzeń zewnętrznych wolnych w komputerze System 10 jest autonomiczna. Sterowanie maszyny przekazane jest do sekcji zerowej, gdzie kontynuowane są obliczenia programu. Jego realizacja jest wstrzymywana na okres jednego cyklu pamięci (hesitation) w momentach, kiedy kanał w sekcji drugiej przekazuje pojedynczy znak do pamięci. W momencie t_4 gdy program w sekcji 0 jest wykonywany, zostaje odszukana właściwa ścieżka na dysku dla programu w sekcji pierwszej. Czytanie rozpoczyna się w chwili t_5, kiedy sterowanie maszyny zostaje przekazane do pierwszej sekcji. W czasie transmisji jednostka centralna zajęta jest wyłącznie przez sekcję, w której aktywna jest instrukcja czytania. W momencie t_6 czytanie sektora zostaje zakończone, kanał szybkich urządzeń zwolniony a sterowanie realizuje kolejne instrukcje programu w sekcji pierwszej. Jednocześnie kontynuowana jest transmisja znaków w kanale wolnych urządzeń sekcji drugiej. W chwili t_7, w programie sekcji pierwszej występuje instrukcja skoku po upływie podstawowego czasu pracy sekcji, która powoduje przekazanie sterowania do sekcji drugiej.

W związku z tym, że w sekcji drugiej aktywna jest instrukcja czytania, sterowanie przechodzi do sekcji następnej czyli zerowej. W czasie t_9 zakończone zostaje przesyłanie znaków z czytnika kart w sekcji drugiej, co oznacza, że przy kolejnym sekwencyjnym dojściu sterowania maszyny do niej (w chwili t_{11}) nastąpi dalsze wykonywanie jej programu.

mgr inż. Władysław GAJEWSKI
mgr inż. Edward LENARCZYK
mgr inż. Andrzej RADZIMIŃSKI
Instytut Łączności

681.322-181.4.06

OPROGRAMOWANIE KOMPUTERA SYSTEM 10

1. Wstęp

Artykuł stanowi omówienie wybranych zagadnień z zakresu oprogramowania komputera System 10, jest niejako przewodnikiem po oprogramowaniu Systemu 10 i określa w ogólnym zarysie możliwości programowe tego komputera.

2. Język wewnętrzny Systemu 10

Maszyna cyfrowa System 10 jest maszyną znakową i prawie wszystkie operacje w maszynie wykonywane są na sześciobitowych znakach. W budowie znaku wyróżniamy czterobitową część numeryczną oraz dwa bity strefy. Dane numeryczne i alfanumeryczne przechowywane są w pamięci w jednakowej postaci a operacje arytmetyczne wykonywane są na znakach bez konwersji liczby na postać binarną. I tak na przykład znak, którego wszystkie bity są zerami, tj. 000000 przedstawia zawsze spację, a znak 010000 przedstawia zawsze cyfrę zero. Również rozkazy mają tę samą postać i składają się z 10 kolejnych znaków. W pamięci operacyjnej adresowany jest każdy znak, jednak elementy elektroniczne maszyny zaprojektowano tak, że rozkazy muszą zaczynać się w komórkach o adresach podzielnych przez 10.

Rys. 1 przedstawia istotne elementy rozkazu w postaci wewnętrznej. Jak widać, maszyna System 10 jest dwuadresowa. Obydwa argumenty instrukcji pobierane są z pamięci operacyjnej. Maszyna nie ma adresowanych

programowo akumulatorów ani innych rejestrów z wyjątkiem trzech rejestrów indeksowych.

Znak	0	1	2	3	4	5	6	7	8	9
bit 6			F			AC	IA		I	B	BC
bit 5	PA					1	PB				1
część numeryczna bity 1-4	LA		A				LB			B	

Rys. 1. Przedstawienie instrukcji w języku wewnętrznym

Niektóre pola w instrukcji mają stałe znaczenie, a inne zmieniają swe znaczenie w zależności od rozkazu. Stałą interpretację mają pola: A, AC i BC, F, IA i IB, PA i PB.

F - Czterobitowy kod operacyjny: teoretyczna możliwość 16 instrukcji, ale jedna kombinacja nie jest wykorzystywana.

AC, BC - Jednobitowe wskaźniki mówiące o tym, czy argument (odpowiednio A lub B) znajduje się w pamięci sekcji (wartość 0) czy w pamięci wspólnej (wartość 1). W przypadku, gdy w polu A lub B nie ma adresu argumentu wartość tego bitu nie ma znaczenia.

IA, IB - Dwubitowe pola wskazujące numer rejestru indeksowego odnoszącego się do argumentu A lub B. Kombinacja 00 oznacza, że dany argument jest nieindeksowany.

PA, PB - Bit adresowania pośredniego PA = 0 oznacza, że w polu o adresie A znajduje się czteroznakowy adres właściwego argumentu. Analogicznie dla argumentu B.

A - W operacjach arytmetycznych i w instrukcjach przesyłania pól zawiera adres jednego z argumentów; w instrukcjach wejścia/wyjścia zawiera adres pierwszego znaku pola, z (lub do) którego odbywa się transmisja.

B - W operacjach arytmetycznych i w instrukcjach przesyłania pól - adres jednego z argumentów; w instrukcjach wejścia/wyjścia - liczba transmitowanych znaków.

LA, LB - W instrukcjach arytmetycznych - długość argumentu odpowiednio A lub B, w instrukcjach przesyłania pól zawartość ich stanowi dwucyfrową długość obydwu pól (maksymalna długość przesyłanych pól - 100 znaków: LA = 0 i LB = 0); w instrukcjach wejścia/wyjścia pola te służą do wskazania urządzenia i sposobu transmisji.

Znajomość języka wewnętrznego, jakkolwiek niekonieczna dla programisty, może ułatwić mu kontrolę i testowanie programu. Jednocześnie znajomość rozmieszczenia poszczególnych elementów rozkazu pisanego w języku Assembler w postaci języka wewnętrznego ułatwia rozumienie i zapamiętanie pewnych reguł obowiązujących w Assemblerze.

3. Podstawowy język programowania Assembler 1

Assembler 1 jest najprostszym językiem symbolicznym dla komputera System 10. Repertuarem jego instrukcji jest 14 rozkazów podzielonych na cztery zasadnicze grupy. W języku Assembler kod operacyjny F zastąpiony jest kodem mnemonicznym podanym w nawiasach w poniższym zestawieniu instrukcji.

- Instrukcje arytmetyczne
  - Dodawanie (A)
  - Odejmowanie (S)
  - Mnożenie (M)
  - Dzielenie (D)

- Instrukcje przesyłania, wydawnictwa i porównywania pól
  - Przesyłanie znaków (MC)
  - Przesyłanie bitów numerycznych znaków (MN)
  - Przesyłanie adresów (MA)
  - Wymiana zawartości pól (X)
  - Tworzenie wartości numerycznej (FN)
  - Wydawnictwo (E)
  - Porównanie (C)

- Instrukcje wejścia/wyjścia
  Wprowadzenie z urządzenia do pamięci operacyjnej (R)
  Wyprowadzenie z pamięci operacyjnej na urządzenie zewnętrzne (W)

- Instrukcje skoku (BC)
  Skoki bezwarunkowe
  Skoki przy określonym stanie rejestru warunkowego
  Skoki ze śladem
  Skoki specjalne

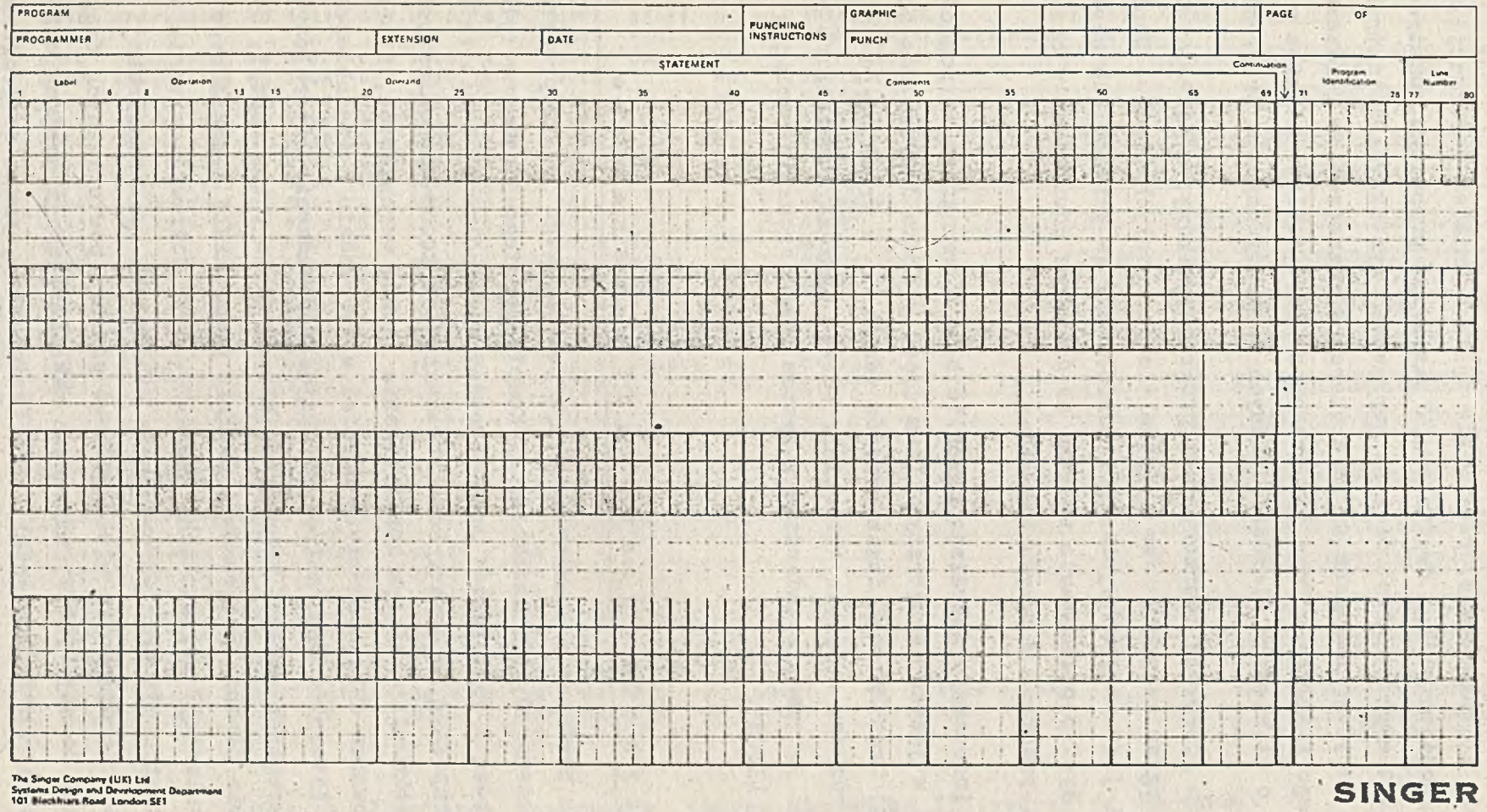
Do pisania programu w języku Assembler używane są specjalne arkusze programowe (rys. 2), które wypełniamy wg podanych nagłówków. W kolumnach 1-6 może być umieszczona etykieta, w kolumnach 8-13 umieszczony jest kod mnemoniczny rozkazu a od kolumny 15 argument instrukcji.

Oprócz rozkazów programu w skład języka Assembler 1 wchodzą jeszcze dyrektywy sterujące i deklaracje pól. W Assemblerze można deklarować pola numeryczne, alfanumeryczne oraz pola zawierające adresy dowolnych obszarów pamięci. Za pomocą odpowiedniej dyrektywy sterującej programista ma możliwość deklarowania pól, jak również umieszczania instrukcji programu w pamięci swojej sekcji lub w obszarze pamięci wspólnej. Dane, pola robocze oraz podprogramy umieszczone w pamięci wspólnej przez program znajdujący się w dowolnej sekcji mogą być wykorzystywane również przez programy z innych sekcji.

Ze względu na postać instrukcji długość podstawowych pól numerycznych nie powinna przekraczać 10 cyfr dziesiętnych. Przy argumentach dłuższych programowanie staje się uciążliwe. Z tych samych względów długość podstawowych pól alfanumerycznych nie powinna przekraczać 100 znaków.

Argumentami w instrukcjach mogą być adresy bezwzględne lub symboliczne obszarów znajdujących się w danej sekcji lub w pamięci wspólnej. Instrukcje skoku, jako adresy przeniesienia sterowania, mogą wskazywać rozkazy odległe od bieżącego adresu o określoną liczbę znaków.

Istnieją dwie wersje kompilatora języka Assembler 1 dla maszyny cyfrowej System 10. Są to: wersja kartowa i wersja dyskowa. Obie wersje



Rys. 2. Arkusz programowy języka Assembler

są podobnie zorganizowane, różnią się tylko nośnikiem informacji. Proces kompilacji przebiega dwufazowo. W pierwszym przebiegu konstruowana jest tablica etykiet i kontrolowana poprawność syntaktyczna instrukcji. Błędy tej fazy wyprowadzane są na drukarce z zaznaczeniem gwiazdką niezrozumiałego fragmentu. Poza tym drukowana jest tablica etykiet z komentarzem o etykietach niezdefiniowanych i nieużywanych w programie.

W drugim przebiegu tworzony jest program wynikowy, a także drukowany cały program ze wskazaniem ewentualnie pozostałych błędów. Wynikiem kompilacji jest program wynikowy wyperforowany na kartach lub zapisany na dysku. Rys. 3 przedstawia wydruki na drukarce powstające w czasie kompilacji programu w języku Assembler.

Podane informacje pozwalają określić błąd i usunąć go za pomocą instrukcji w języku wewnętrznym. Lewa część wydruku dotyczy postaci wewnętrznej oraz bezwzględnych adresów rozkazów, a prawa część podaje brzmienie rozkazu w języku Assembler. Wersja dyskowa kompilatora pozwala dołączyć programy standardowe i system gospodarowania pamięcią dyskową (tzw. LIOCS).

4. Język programowania Assembler 2

Istnieje bardziej rozbudowana wersja podstawowego języka programowania dla Systemu 10, tj. Assembler 2. Kompilator tego języka ma tylko wersję dyskową i jest lepiej przystosowany do współpracy z biblioteką programów zapisaną na dysku. Poza rozszerzeniem kodów mnemonicznych rozkazów skoku ułatwiającym konstruowanie skoków warunkowych, Assembler 2 wprowadza dwie zasadnicze nowości w porównaniu z Assemblerem 1: stosowanie literałów oraz makroinstrukcję. Wprowadzenie literałów umożliwia umieszczanie stałych argumentów bezpośrednio w instrukcji, co z kolei ułatwia pisanie programów i zwalnia programistę z dokładnego pamiętania deklaracji pól stałych, nagłówków itp. Możliwość pisania własnych makroinstrukcji oraz wprowadzenie biblioteki makroinstrukcji systemu pozwala na ominięcie wielokrotnego pisania powtarzalnych fragmentów programu. Do najczęściej używanych makroinstrukcji systemu należą makroinstrukcje MPY (mnożenie) i DVD (dzielenie), przez co programowanie tych działań w komputerze System 10 jest znacznie prostsze niż w

```
SYSTEM TEN ASSEMBLER II                                                                                PAGE 0001

SEQ. LOCN  INSTR/DATA OP  A/R   M I  B/S   M I   LINE                                     IMAGE                    C

     0000               0300                     0001          ORG    300
     0300  8PTP080400 07 0400   8 0 0400   8 0   0002  START   S      TOTAL,TOTAL          ZEROISE TOTAL
                                                 0003  *THIS PROGRAM DEMONSTRATES ASSEMBLER CODING
     0310  0040230005 01 0412   0 0 0005   3 0   0004  EWA     WCM    =C'M-04I'(WS)
     0320  0040810004 00 0408   0 0 0004   1 0   0005          RM     ABC(WS)
     0330  S0SV000000 11 0360   3 0 0000   0 0   0006          BC     EXIT(3)
     0340  4P40800400 04 0408   4 0 0400   8 0   0007          A      ABC,TOTAL
     0350  U0S0000000 11 0310   5 0 0000   0 0   0008          B      EWA
     0360  0040730001 01 0417   0 0 0001   3 0   0009  EXIT    WCM    =C'M'(WS)
     0370  004P010008 01 0400   0 0 0008   1 0   0010          WM     TOTAL(WS)
     0380  0040830005 01 0418   0 0 0005   3 0   0011          WCM    =C'M/04K'(WS)
     0390  U0SP000000 11 0300   5 0 0000   0 0   0012          B      START
     0400               0000                     0013  WS      EQU    0
     0400  00000000     0001        0008         0014  TOTAL   DM     C'00000000'
     0408               0001        0004         0015  ABC     DM     C4
     0412               0412                     0016           ORG
     0412  M-04I        0001        0005         0017  @00001  DM     C'M-04I'
     0417  M            0001        0001         0018  @00002  DM     C'M'
     0418  M/04K        0001        0005         0019  @00003  DM     C'M/04K'
     0423               0300                     0020          END    START

SYSTEM TEN ASSEMBLER II                                                                                PAGE 0002

TYP  I  LNTH  ADDRESS LINE  SYMBOL                                                        **REFERENCES**

        0005  0412P   0017  @00001  U0004 W  -A
        0001  0417P   0018  @00002  U0009 W  -A
        0005  0418P   0019  @00003  U0011 W  -A
        0004  0408P   0015  ABC     S0005 R  -A   U0007 A  -A
        0010  0310P   0004  EWA     E0008 BC -A
        0010  0360P   0009  EXIT    E0006 BC -A
        0010  0300P   0002  START   E0012 BC -A   U0020 END
        0008  0400P   0014  TOTAL   S0002 S  -B   U0002 S  -A   S0007 A  -B   U0010 W  -A
        0001  0000    0013  WS      U0004 W  -AM  U0005 R  -AM  U0009 W  -AM  U0010 W  -AM  U0011 W  -AM
```

Rys. 3. Przykład programu w języku Assembler II (wydruk na drukarkę)

języku Assembler 1. Wszystkie programy napisane w Assemblerze 1 mogą być tłumaczone przez kompilator Assemblera 2.

5. Oprogramowanie dysków

Disc Management Facility (DMF) jest systemem zarządzającym pracą dysków. Pozwala on programiście tworzyć, zmieniać i kasować w trybie konwersacyjnym własne obszary na dysku, w których można operować instrukcjami na poziomie logicznym w czasie wykonywania programu.

5.1. Organizacja danych na dysku

Jednostkami logicznymi danych są: volume, pool, zbiór i rekord. Podajemy nazwy oryginalne występujące w literaturze źródłowej, przyjmując jako odpowiednik angielskiego terminu file - zbiór oraz record - rekord.

- Volume

Każdy pakiet dyskowy nazywany jest w DMF volumem. W czasie tworzenia systemu zapisywane są etykiety volumu w sektorze 1 każdego pakietu.

- Pool

Pool jest to stworzony przez programistę obszar na dysku zawierający się w stałych granicach i mający ustaloną liczbę sektorów. W ramach poolu można tworzyć dowolną liczbę zbiorów. Pool spełnia następujące warunki:

- ma unikalną nazwę w ramach volumu,
- granice poolu nie zachodzą na sektory innego poolu,
- wszystkie sektory wewnątrz poolu należą do tego poolu (żaden z nich nie może być użyty do innych celów),
- granice poolu muszą zawierać się w ramach jednego volumu,
- volum może zawierać dowolną liczbę pooli,
- każdy pool ma unikalną etykietę zajmującą 1 sektor i utworzoną przez system w czasie tworzenia poolu,
- każdy pool ma listę wolnych sektorów, tzn. sektorów nieprzydzielonych do żadnego ze zbiorów w tym poolu.

Są dwa typy pooli w DMF:

- poole z łącznikami (linked pool), które zawierają zbiory danych lub programy użytkownika; sektory poolu są połączone sekwencyjnie;
- poole robocze (user's scratch pool), które używane są przez pewne programy standardowe, takie jak DISC SORT I, Assembler 2 czy kompilator RPG.

- Zbiór

Ciąg logicznie powiązanych rekordów tworzy zbiór. Zbiór jest częścią poolu.

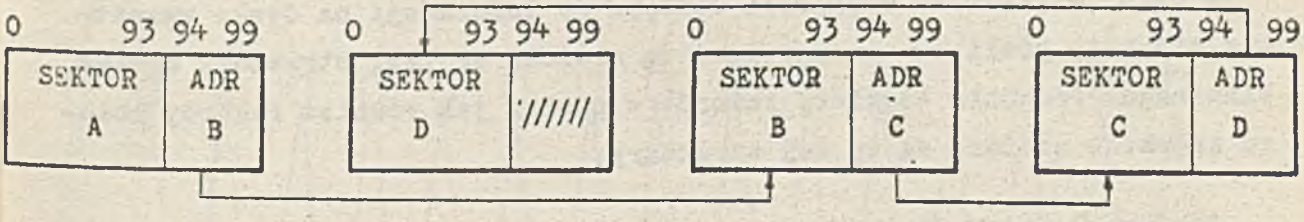
Każdy zbiór spełnia następujące warunki:

- ma unikalną nazwę w ramach poolu,
- ma unikalną etykietę zajmującą 1 sektor i utworzoną w momencie tworzenia zbioru.

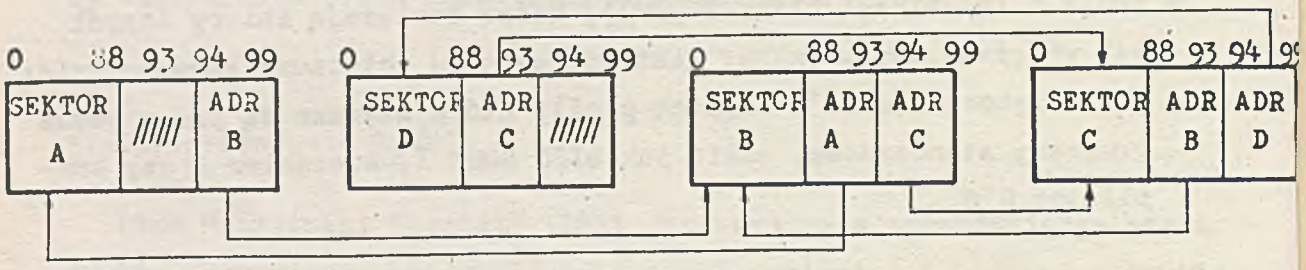
Są trzy typy zbiorów:

- połączone sekwencyjnie (linked sequential),
- połączone podwójnie (doubly-linked),
- indeksowe.

Każdy sektor wewnątrz zbioru połączonego sekwencyjnie zawiera fizyczny adres logicznie następnego sektora w zbiorze (rys. 4). Logiczna sekwencja nie pokrywa się najczęściej z sekwencją fizyczną sektorów w wyniku zmian rozmiarów zbiorów w poolu. Ostatni rekord w zbiorze zawiera w pozycji adres - sześć kresek ukośnych, co jest dla systemu znakiem końca zbioru. Połączenie sektorów w zbiorach połączonych podwójnie przedstawia rys. 5.



Rys. 4. Połączenie sektorów w zbiorach połączonych sekwencyjnie (linked sequential file)



Rys. 5. Połączenie sektorów w zbiorach połączonych podwójnie (doubly-linked file)

Każdy sektor zawiera adres sektora poprzedniego oraz następnego. Każdy sektor oprócz fizycznego adresu następnego w sekwencji logicznej sektora zawiera adres sektora poprzedniego. Niektóre programy standardowe, jak np. Text Editor współpracują z tego typu zbiorami.

Zbiory indeksowe przeznaczone są do pracy ze zbiorami danych w dostępie przypadkowym. Każdy sektor w zbiorze indeksów zawiera fizyczny adres sektora danych. Sektory zbioru indeksów połączone są ze sobą wg struktury drzewa binarnego.

- Rekord

Rekord jest ciągiem pól opisujących ten sam podmiot. W DMF 1 rekord jest synonimem sektora. Tak więc maksymalny rozmiar rekordu wynosi 94 znaki. Pozostałe sześć znaków zarezerwowano na adres następnego rekordu. Ograniczenia tego nie ma w DMF 2. Wówczas rekord zawierający więcej niż 94 znaki zajmuje kolejne sektory dysku. Każdy przypadek z DMF 2 można więc sprowadzić do DMF 1.

- Struktura drzewa binarnego

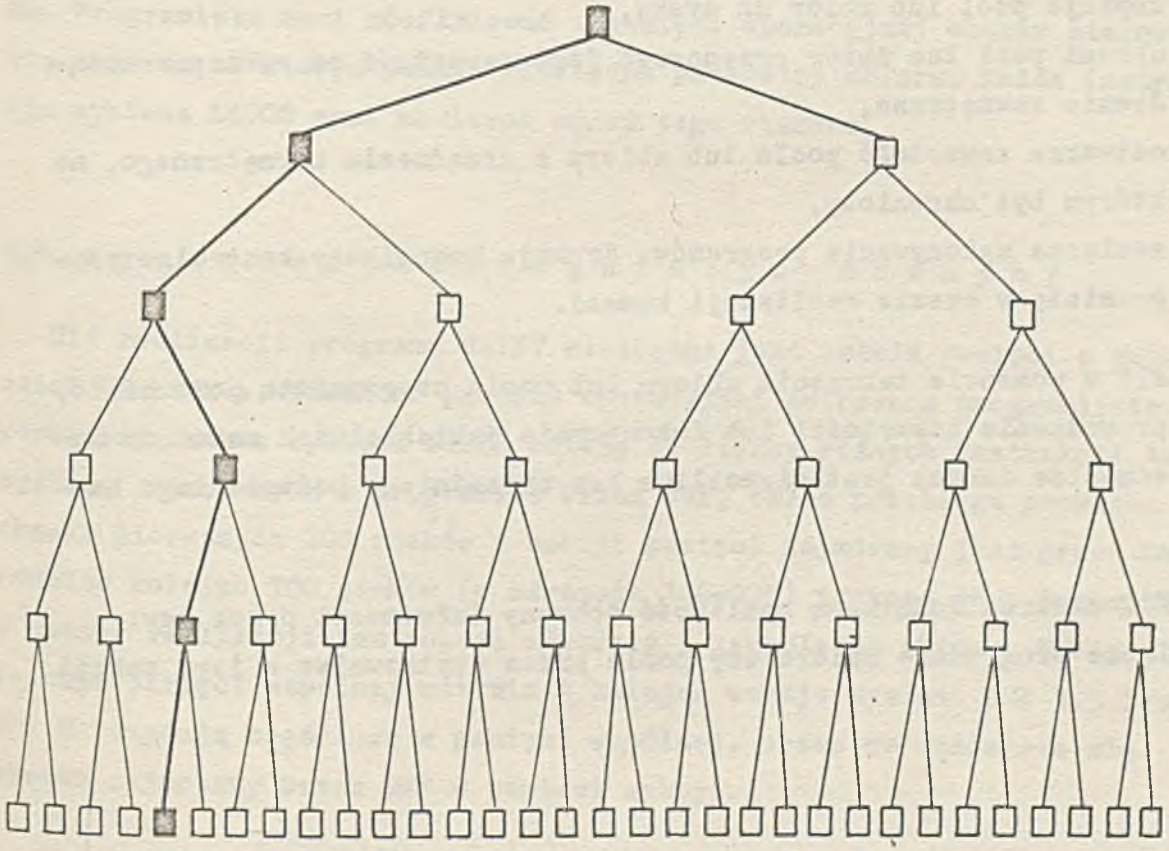
W celu zwiększenia szybkości dostępu do informacji na dysku wszystkie etykiety pooli w ramach volumu są ułożone wg tzw. struktury drzewa binarnego. Podobnie etykiety zbiorów w poolu, jak również rekordy zbioru indeksów ułożone są wg tej struktury.

Jak wspomniano wyżej, etykiety tworzone są przez DMF i włączane są w odpowiednie miejsce drzewa. Każda etykieta zawiera wskaźnik prowadzący do bezpośrednio mniejszego i bezpośrednio większego elementu drzewa. Jeśli dana etykieta znajduje się na końcu gałęzi - wskaźnik prowadzący

do następnych elementów ustawiony jest na zero (sześć kresek ukośnych).

W czasie poszukiwania danego poolu czy zbioru jego nazwa porównywana jest z nazwą bazy drzewa binarnego (pierwszego elementu drzewa). W wyniku porównania eliminujemy połowę elementów drzewa i przechodzimy do następnej etykiety. Proces ten powtarzamy aż do momentu znalezienia wymaganej etykiety lub stwierdzenia, że dany element nie występuje w wykazie. Rys. 6 ilustruje poszukiwanie rekordu w drzewie binarnym zawierającym 63 elementy. W najgorszym przypadku należy wykonywać sześć porównań, aby dojść do każdego rekordu. Zwiększając drzewo do 127 elementów dodajemy tylko jedno dodatkowe porównanie pozwalające osiągnąć każdy rekord. Dla miliona sektorów wymagane jest maksimum 20 porównań prowadzących do poszukiwanego elementu.

Etykieta volumu zawiera adres pierwszego poolu na dysku. Podobnie etykieta każdego poolu zawiera adres pierwszego zbioru w poolu. Etykieta zbioru wskazuje pierwszy rekord w zbiorze.



Rys. 6. Przykład struktury drzewa binarnego dla 63 elementów

5.2. Operacje zarządzania systemem

Program MAINT będący częścią DMF umożliwia operowanie informacją na dysku na poziomie pooli i zbiorów. Współpraca z programem MAINT odbywa się w trybie konwersacyjnym, tzn. po napisaniu komendy (command) na konsoli operatorskiej lub na monitorze ekranowym następuje wykonanie polecenia i oczekiwanie na podanie następnej komendy.

Program MAINT realizuje następujące funkcje:

- tworzy poole,
- zakłada, modyfikuje, zmienia nazwy zbiorów połączonych sekwencyjnie,
- zakłada zbiór indeksów w celu stworzenia dostępu przypadkowego do zbioru danych,
- kasuje zbiory w poolu lub poole w systemie,
- wyprowadza zawartości zbiorów i pooli na wskazane urządzenie zewnętrzne,
- zmienia zawartość zbioru pozostawiając jego nazwę,
- kopiuje pool lub zbiór na dysku,
- chroni pool lub zbiór przenosząc jego zawartość na wskazane urządzenie zewnętrzne,
- odtwarza zawartość poolu lub zbioru z urządzenia zewnętrznego, na którym był chroniony,
- zawiesza wykonywanie programów, drukuje komunikaty kontrolne programisty w czasie realizacji komend.

Jeśli w momencie tworzenia zbioru lub poolu programista poda hasło, to wyprowadzenie zawartości lub dokonywanie jakichkolwiek zmian na danej jednostce danych jest niemożliwe bez uprzedniego podania tego hasła.

DMF 2 stwarza dodatkową możliwość ochrony informacji przez tzw. zamknięcie programowe zbioru czy poolu przez użytkownika z jego sekcji pamięci.

5.3. Instrukcje logiczne

Operacje na poziomie logicznym tzn. na rekordach, umożliwia w czasie wykonywania programu system LIOCS (Logical Input/Output Control System). Programista może używać instrukcji systemu LIOCS w celu:

- otwarcia lub zamknięcia zbioru na dysku,
- zapisania rekordu do zbioru o dostępie sekwencyjnym lub przypadkowym,
- odczytania rekordu ze zbioru o dostępie sekwencyjnym lub przypadkowym,
- modyfikacji rekordu,
- wprowadzenia rekordu w określone miejsce zbioru,
- kasowania dowolnego rekordu w zbiorze,
- znajdowania początku lub końca zbioru,
- zapisu znacznika końca zbioru.

Poszczególne czynności realizowane są przez odpowiednie podprogramy systemu, które dołączane są w czasie kompilacji do programu użytkownika. Programista musi zdefiniować w pamięci operacyjnej obszar sterowania zbiorem, w którym podaje niezbędne parametry zbioru. Każda instrukcja systemu LIOCS musi zawierać adres tego obszaru.

5.4. Wymagana konfiguracja maszyny

Dla realizacji programu MAINT niezbędna jest sekcja pamięci o pojemności 9K znaków. Wielkość pamięci operacyjnej potrzebna programiście korzystającemu z systemu LIOCS zależy od liczby różnych instrukcji tego systemu stosowanych w programie. Przez cały okres przebiegu programu obszar pierwszych 300 znaków w sekcji pamięci zajmowany jest przez DMF. Ponadto kolejne 700 znaków (o adresach 300-999) używane jest przez DMF w czasie realizacji instrukcji otwarcia i zamknięcia zbioru. DMF 1 nie zajmuje pamięci wspólnej natomiast kolejne wersje systemu DMF 1.5 i DMF 2 rezydują częściowo w pamięci wspólnej, przez co zmniejsza się obszar zajmowany przez DMF w pamięci sekcji.

Wymaganą konfigurację urządzeń zewnętrznych przedstawia poniższa tabela.

Tabela 1. Konfiguracja sprzętu dla oprogramowania dysków

	Jednostka centralna	Jednostka (i) dysków	Konsola operatorska lub monitor ekranowy	Czytnik kart	Perforator (y) kart	Drukarka (i)	Przewijak (i) taśm magnetycznych	Czytnik (i) taśmy papierowej	Perforator (y) taśmy papierowej
Generacja systemu	X	X	X	X	-	-	O	-	-
System MAINT	X	X	X	O	O	O	O	-	-
System LIOCS	X	X	X	O	O	O	O	O	O

X = obowiązkowe

O = opcyjne

- = nigdy nie używane

6. Zakończenie

Artykuł nie wyczerpuje wszystkich problemów dotyczących oprogramowania komputera System 10. Sygnalizuje on jedynie niektóre punkty niezwykle szerokiego tematu, którym jest oprogramowanie maszyny cyfrowej. Wspomnieć w tym miejscu wypada o wielu programach standardowych ułatwiających testowanie, uruchamianie i dokonywanie poprawek w programach pisanych w języku Assembler 1 i Assembler 2, takich jak "Tester 20", "Tester 21" oraz "Text Editor", a także o istniejącym w wersji dyskowej kompilatorze języka RPG (Report Program Generator). Jest to już język wyższego rzędu, w którego kompilatory wyposażone są również inne typy maszyn cyfrowych zainstalowanych w Polsce.

Po cechach języka wewnętrznego, oprogramowaniu podstawowym, wyposażeniu w kompilatory i bibliotekę programów standardowych można ocenić przydatność komputera do określonych celów.

Wymienione w poprzednim artykule cechy sprzętu oraz pokrótce omówione tutaj oprogramowanie, pozwalają określić maszynę System 10 jako maszynę uniwersalną, jakkolwiek najczęściej jest ona stosowana w systemach koncentracji i gromadzenia danych z dużą ilością aktualizacji i niezależnym ich wykorzystaniem przez kilka programów.

Wyposażenie komputera w synchroniczne i asynchroniczne adaptery komunikacyjne umożliwia przetwarzanie danych w zdalnym dostępie za pośrednictwem linii transmisyjnych.

Wydaje się, że ze względu na pewne cechy, jak choćby brak systemu operacyjnego, a co za tym idzie układowe sterowanie wieloprogramową pracą maszyny lub dwuprzewodowe połączenie jednostki centralnej z urządzeniami zewnętrznymi, które wyróżniają System 10 spośród innych maszyn cyfrowych popularnych w naszym kraju, warto było chociaż w tak skróconej formie poznać ten komputer.

ETO NOWOŚCI Nr 1 /1975

mgr inż. Zbigniew KĘDZIOR
Instytut Maszyn Matematycznych

681.327.12:744(0.84.2):
744.32

URZĄDZENIA ODCZYTUJĄCE INFORMACJĘ GRAFICZNĄ ZAPISANĄ NA NOŚNIKACH NATURALNYCH

1. Charakterystyka urządzeń

Urządzeniami odczytującymi informację graficzną nazywamy takie urządzenia, które przyporządkowują liczby określonym punktom, występującym na naturalnych nośnikach informacji oraz wyprowadzają te liczby w postaci akceptowanej przez elektroniczną maszynę cyfrową. Urządzenia tej klasy są również nazywane koderami informacji graficznej, koordynatometrami lub przetwornikami obrazów.

Odczyt informacji może obejmować punkty, linie (czyli skończone zbiory punktów) jak również całe obrazy, z których charakterystyczne punkty lub linie są wydzielane w drugim etapie obróbki cyfrowej. Tę ostatnią grupę nazywamy urządzeniami rozpoznającymi informację graficzną.

Urządzenia odczytujące punkty i linie są wyposażone w dające się prowadzić zespoły nastawcze lub element przypominający kształtem pióro. Naprowadzanie zespołu nastawczego na odczytywane punkty może odbywać się ręcznie, półautomatycznie lub automatycznie.

Zespół nastawczy urządzeń przystosowanych do pracy ręcznej ma okienko, najczęściej z krzyżykiem nitek. Urządzenia półautomatyczne i automatyczne są wyposażone w monitory, które pozwalają wygodnie kontrolować położenie urządzenia nastawczego.

Urządzenia rozpoznające odczytują i analizują obraz metodami zapożyczonymi z techniki rozpoznawania znaków.

Urządzenia odczytujące informację graficzną pracują przede wszystkim w reżymie pośredniej współpracy z komputerem ("off line"). Wyprowadzane dane są rejestrowane na nośnikach takich, jak taśma magnetyczna, taśma papierowa lub karty dziurkowane. W nielicznych przypadkach ten rodzaj informacji rejestruje się na papierze (drukarki, monitory dalekopisowe). Ze względu na małą prędkość odczytu w porównaniu z możliwościami EMC reżym bezpośredniej współpracy - "on line" jest stosowany w nielicznych przypadkach.

Głównymi parametrami charakteryzującymi urządzenia są: wymiary przetwarzanych obrazów, rozróżnialność odczytu (parametr ten niekiedy jest nazywany działką elementarną), bezwzględna dokładność odczytu oraz maksymalna prędkość odczytu.

2. Podział urządzeń komercjalnych

Współcześnie produkowane urządzenia można podzielić na trzy następujące grupy:

- urządzenia małoskalowe,
- urządzenia z płaskim stołem,
- plotery wyposażone w głowicę odczytującą.

Nazwa urządzeń pierwszej grupy wynika z rozmiaru przetwarzanych rysunków. Przeważnie stosowane są w nich najprostsze rozwiązania, w związku z czym ich parametry znacznie się różnią w porównaniu z urządzeniami z pozostałych grup. Do grupy tej zalicza się urządzenia Graftran firmy Bolt, Beranek and Newman. Urządzenie to wykorzystuje potencjometryczną metodę odczytu. Na odczytywane punkty obrazu naprowadza się element w kształcie pióra, który jest zamocowany na ramieniu teleskopowym. Ramię to daje się obracać pod kątem 110°. Oznacza to, że odczytywane punkty są rejestrowane we współrzędnych biegunowych. Innymi przykładami urządzeń małoskalowych mogą być kodery takie, jak DI-1400 firmy Graph-Data oraz Model 520 Graphic Tablet Ecricon firmy Shintron Company Inc. Parametry ich są podane w tabeli 1 (kolumny 1 i 3).

W urządzeniach drugiej grupy wykorzystuje się rozwiązania zapewniające dużą dokładność i prędkość odczytu. Stosuje się tu metody takie, jak

fotoelektryczna, elektroniczna z elektrycznym układem nadążnym lub ultradźwiękowa. Do grupy tej zalicza się między innymi następujące urządzenia:

- GCD 1 firmy Gerber (powierzchnia odczytu 1,07 x 1,52 m, dokładność odczytu 0,254 mm),
- Datagrid Digitizer firmy Bendix (maksymalna powierzchnia odczytu 1,07 x 1,52 m, dokładność odczytu 0,127 mm),
- Gradicon Grafic Coordinate Digitizer firmy Instronics (maksymalna powierzchnia odczytu 1,22 x 1,52 m, dokładność odczytu 0,1 mm),
- 3990 firmy Autorol (maksymalna powierzchnia odczytu 1,52 x 2,03 m, dokładność odczytu 0,1 mm),
- Graf/Pen GP-2 firmy Science Accessories Corporation (maksymalna powierzchnia 1,83 x 1,83 m).

Trzecią grupę stanowią graficzne urządzenia wejściowo-wyjściowe. Są to właściwie pisaki (plotery) z płaskim stołem, wyposażone dodatkowo w głowicę odczytującą. Przyjęcie takiego rozwiązania podyktowane jest potrzebami niektórych użytkowników. Należy zaznaczyć, że w pewnych zastosowaniach rozwiązanie to jest bardzo ekonomiczne. W urządzeniach tej grupy zespół nastawczy może być sterowany ręcznie lub automatycznie. Przykładem może być tu urządzenie KART 2, którego parametry przedstawiono w punkcie 3 lub urządzenie OLF 1 Automating Digitizing System firmy Gerber (maksymalna powierzchnia odczytu 1,53 x 6,04 m, dokładność odczytu 0,076 mm). Ten ostatni koordynatometr jest sterowany przez komputer; odczyt rysunków może być wykonywany ręcznie, półautomatycznie lub automatycznie. Ponadto w skład wyposażenia wchodzi monitor telewizyjny pozwalający operatorowi na bezpośrednie śledzenie odczytu.

W tabeli nr 1 przedstawiono wybrane urządzenia odczytujące, produkcji firm amerykańskich. Przykłady pozwalają zorientować się w zakresie parametrów, wyposażenia oraz w cenach, które w 1972 r. wahały się w granicach od 3.200 do 200.000 dolarów USA.

Tabela 1. Zestawienie urządzeń do wprowadzania informacji graficznej

Producent	Graph-Data	Shintron Company IMC	Autotrol
Nazwa lub numer modelu urządzenia	Digitizing Datatrace DJ-1400	Model 520 Graphic Data Tablet Ecricon	3990
Zespoły i urządzenia wprowadzające informację	Pióro trasujące	Pióro	Okienko Klawiatura
Wymiary powierzchni odczytywanej (mm)	228 x 915	280 x 280	1015 x 1520 1220 x 1520 st 1520 x 2030
Maksymalna prędkość przesuwu głowicy (mm/s)	25,4 (do 600 pkt na godzinę)	500 par współrzędn./s	2540
Rozróżnialność (mm)	0,229	0,345	0,0254
Dokładność odczytu (mm)	0,254	1,7	0,101
Nośniki, urządzenia lub sposób wyprowadzania informacji	Taśma pap. Sprzęgacz telefoniczny. Monitor dalekop. (opcja)	Monitor cyfrowy Bezp. do EMC	Taśma magnet. Taśma pap. Karty. Drukarka. Bezp. do EMC
Sposób odczytu, niektóre szczegóły techniczne	Pióro odczytujące jest osadzone w karetce dwukółkowej zamocowanej na ramce	Metoda odczytu nie opisana w publikacjach	Mechanizm kinetyczny + metoda fotoelektryczna
Rok produkcji	Metoda odczytu nie opisana w publikacjach	1973	1971
Cena w dolarach amerykańskich	5.000	-	8.500

Objaśnienie: st - wykonanie standardowe

zapisanej na nośnikach naturalnych produkcji firm amerykańskich:

Gerber	Instronics	Bendix	Science Accessories Corporation	Gerber
GCD 1	Gradicon Graphic Coordinate Digitizer	Data Grid Digitizer	Graf/Pen GP-2 Digitizing System	OLF 1
Okienko Klawiatura	Okienko Klawiatura	Okienko Klawiatura	Pióro	Głowica elektronicznego układu nadążnego
1070 x 1520	618 x 915 915 x 1220 1220 x 1520 915 x 1520	76 x 915 915 x 1220 1066 x 1520	355 x 355 st 1829 x 1829	1530 x 244 1530 x 6040
-	1010	2620	200 par współrzędn./s	4580
0,0254	0,0254	0,0254	0,178 dla stołu 355 x 355	0,254
0,254	0,101	0,127	-	0,0762
Karty Opcje Taśma magn. Taśma pap. Bezp. do EMC	Taśma magn. Taśma pap. Karty	Taśma magn. Taśma pap. Karty Dalekopis Bezp. do EMC	Taśma magn. Taśma pap. Karty Bezp. do EMC	Taśma magn. Taśma pap.
Mechanizm kinetyczny + metoda fotoelektryczna	Mechanizm kinetyczny pod powierzchnią stołu. Metoda: elektroniczny układ nadążny	Metoda elektroniczna. Płyta stołu z tworzywa sztucznego, w który są wtopione druty	Metoda wykorzystująca ultradźwięki	Mechanizm kinetyczny. Metoda: elektron. układ nadążny (odczyt ręczny i automatyczny)
1964		1968	1971	1971
13.900	15.490	14.440	3.215 bez urządzeń wyjść. i opcji	200.000

3. Opis metod odczytu informacji graficznej

- Podstawowe zespoły koderów

W pracy omówiono tylko te metody odczytu, które są stosowane w koderach informacji graficznej obecnie dostępnych na rynku. W urządzeniach tego rodzaju można wyróżnić trzy podstawowe zespoły: zespół nastawczy, zespół układów elektronicznych i elektrycznych oraz stół.

Zespół nastawczy jest tym zespołem, który naprowadza dajnik na odczytywany punkt. Może on być przesuwany ręcznie, półautomatycznie lub - w nielicznych przypadkach - automatycznie. Sposób półautomatyczny polega na tym, że operator steruje położeniem dajnika za pomocą silników. W sposobie automatycznym dajnik jest przesuwany wzdłuż odczytywanej linii przez silniki, które z kolei są sterowane przez dajnik.(sprzężenie zwrotne).

Układy elektroniczne przetwarzają informację odczytaną na słowa binarne, akceptowane przez EMC. Jak już wspomniano, słowa te mogą być rejestrowane na odpowiednich nośnikach lub wprowadzane bezpośrednio do komputera.

W większości klasycznych koordynatometrów stół stanowi ich część integralną. Wynika to z przyjętych metod odczytu, o których będzie mowa w następnych podpunktach.

Należy nadmienić, że w tym artykule nie zostaną omówione metody rozpoznawania informacji graficznej. Jak już wspomniano, są one zapożyczone z techniki optycznego rozpoznawania znaków. Metody te opisano m.in. w publikacjach [2], [3].

- Metoda potencjometryczna

Odpowiednie elementy mechaniczne kinetycznego zespołu nastawczego są sprzężone z potencjometrami stanowiącymi element sterujący wzmacniacza napięcia stałego. Przetworniki analogowo-cyfrowe zamieniają napięcie stałe na liczby binarne określające współrzędne odczytywanych punktów. Metoda ta daje małą dokładność odczytu wskutek dryftu występującego w wymienionych wzmacniaczach. Obecnie jest już rzadko stosowana. Do

nielicznych przykładów wykorzystania tej metody należy urządzenie Graftran firmy Bolt, Beranek and Newman.

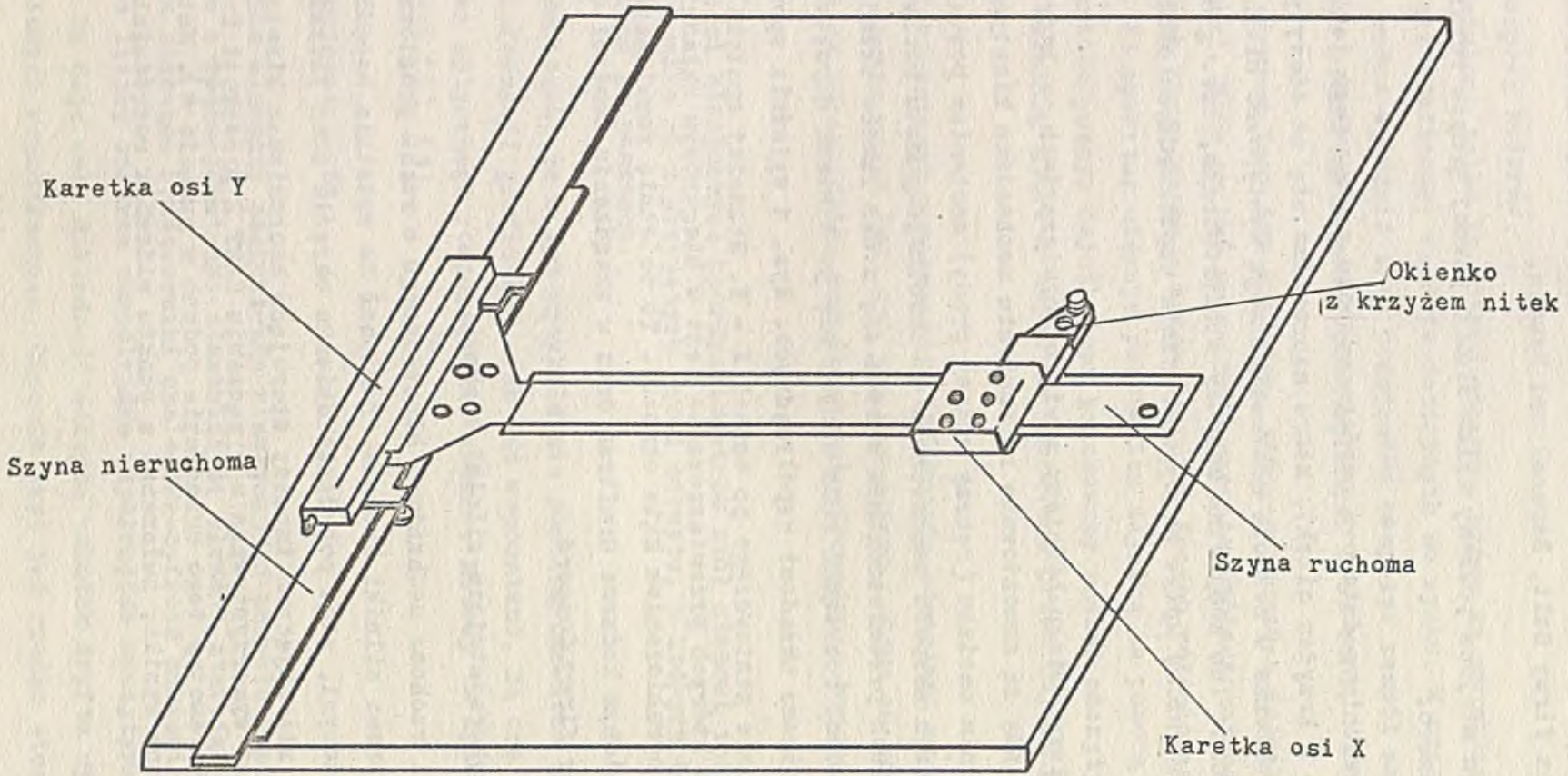
W tym miejscu powiemy kilka słów o pewnej grupie rozwiązań zespołu nastawczego.

Przeważnie podstawową częścią zespołu nastawczego jest okienko (często z krzyżem nitek), które naprowadza się na odczytywany punkt. Oprócz okienka w zespole nastawczym mogą występować bierne elementy elektryczne lub kompletne zespoły elektroniczne, tzw. głowice, które mają zasadniczy wpływ na wygenerowanie współrzędnych odczytywanego punktu.

W dalszym ciągu będziemy mówić o rozwiązaniach, w których wymienione elementy są zamontowane na ramieniu mechanizmu kinetycznego. Przy odpowiednim nacisku (ręczny rodzaj pracy) mechanizm pozwala na przemieszczanie zespołu nastawczego do dowolnego punktu odczytywanego rysunku. Ruch przemieszczenia składa się z dwu ruchów składowych. W większości rozwiązań ruch odbywa się po liniach odpowiadających prostokątnemu układowi współrzędnych. Rys. 1 wyjaśnia sposób przesuwania zespołu nastawczego po osiach X - Y. Rzadziej spotyka się rozwiązania, w których przemieszczanie jest w biegunowym układzie współrzędnych (z ograniczeniem kąta obrotu). To ostatnie rozwiązanie występuje w wymienionym koderze Graftran oraz w urządzeniu Control Graphic Data Digitizer firmy Concord.

- Metoda wykorzystująca silniki krokowe

Części ruchome mechanizmu kinetycznego o ruchu prostokątnym są poruszane przez silniki krokowe. Ponieważ te ostatnie napędza się impulsami prądowymi, stąd położenie okienka określa stan liczników (rewersyjnych) zliczających impulsy sterujące mechanizmem kinetycznym. Metoda ta jest wykorzystywana w urządzeniu KART 2 produkcji krajowej (ważniejsze parametry tego urządzenia podano w punkcie 4). Metoda ta daje umiarkowane wyniki, zwłaszcza z punktu widzenia rozróżnialności odczytu.



Rys. 1. Przykład rozwiązania zespołu nastawczego z mechanizmem kinetycznym

• Metoda fotoelektryczna

Metoda ta jest stosowana w urządzeniach wykorzystujących mechanizm kinetyczny z przesuwaniem okienka w prostokątnym układzie współrzędnych. Zespół nastawczy stanowi tu karetkę, którą operator może przesuwać ręką po jednym ramieniu (np. X) mechanizmu kinetycznego. W karetce oprócz okienka występuje lampka, maska i fotoogniwo. Natomiast wzdłuż ramienia jest umieszczona przezroczysta skala z działkami naniesionymi w stałych odstępach. Światło lampki przechodzące przez skalę i szczelinę maski jest odbierane przez fotoogniwo. Przy przesuwaniu karetki wzdłuż ramienia szczelina maski jest przesłaniana przez nieprzezroczyste działki skali. Wskutek tego następuje zmiana napięcia(wytwarzanego przez fotoogniwo), które jest przetwarzane na impulsy zliczane przez odpowiedni licznik. Stan licznika określa wartości odpowiedniej współrzędnej. Generowanie liczby określającej wartości drugiej współrzędnej odbywa się w analogiczny sposób, z tą tylko różnicą, że lampka, maska i fotoogniwo są zainstalowane na karetce ramienia X (które przesuwa się po szynie Y, z przezroczystą skalą). Pewną odmianą wyżej opisanego rozwiązania jest zastosowanie skali okrągłej przymocowanej do kółka zębatego, przesuwanego po zębatce ramienia mechanizmu kinetycznego. Metoda fotoelektryczna zapewnia stosunkowo dużą dokładność odczytu; jest ona wykorzystywana w urządzeniu typu 3990 firmy Autotrol oraz GCD 1 firmy Gerber.

• Metoda elektroniczna

W płycie stołu, wykonanej ze specjalnego tworzywa są wtopione dwa zespoły równoległych przewodów elektrycznych, z których każdy jest wzajemnie prostopadły. W metodzie tej zespół nastawczy nie wymaga mechanizmu kinetycznego. Ma on formę pudełka, w którym występuje cewka zasilana prądem przemiennym. Pole elektromagnetyczne cewki przecina przewody stołu powodując indukowanie w nich napięcia. Za pomocą odpowiednich układów elektronicznych wyznacza się jednoznacznie współrzędne punktu, na które jest ustawione okienko zespołu nastawczego.

Metoda ta odznacza się wieloma zaletami: pozwala na znaczną prędkość odczytu i daje dużą dokładność pomiaru (odczytu). Powyższa metoda jest zastosowana m.in. w urządzeniu Data Grid Digitizer firmy Bendix oraz Graftcon 2020 firmy Bolt, Beranek and Newman.

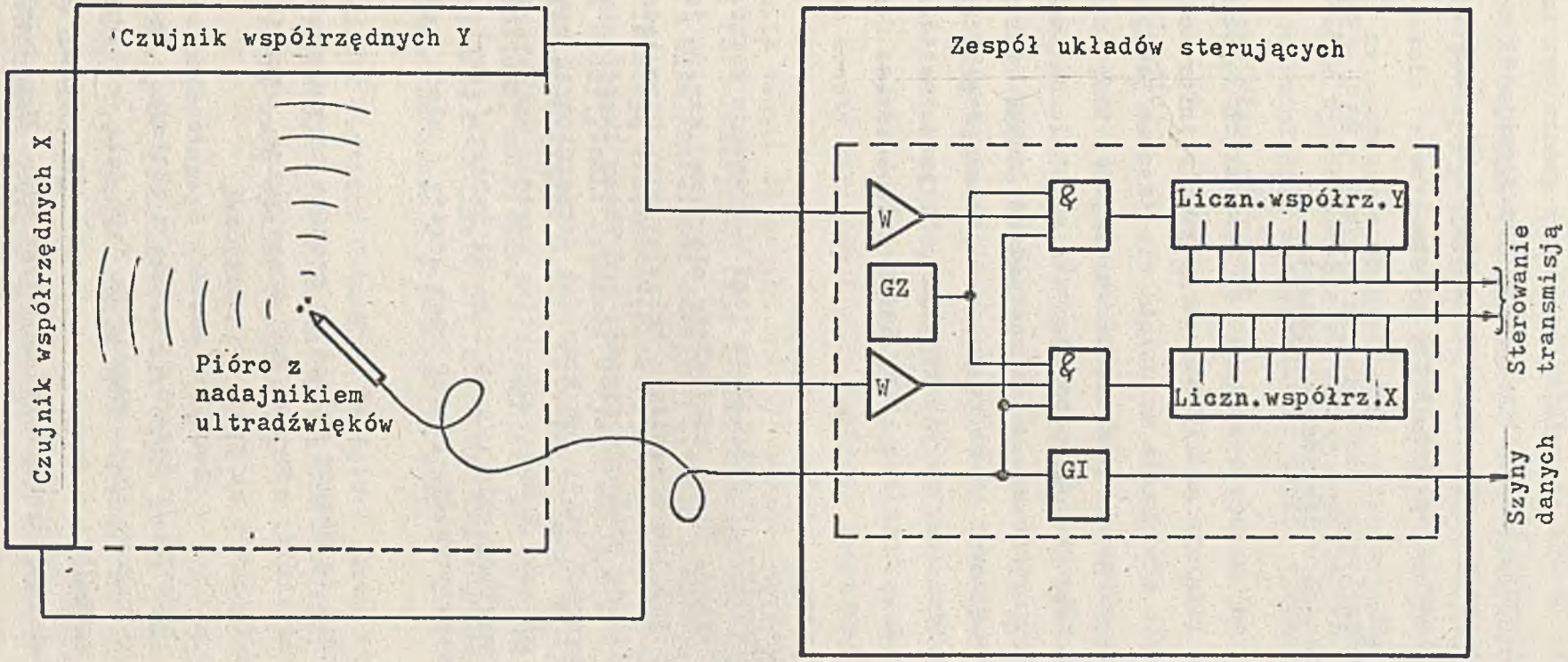
W dalszym ciągu powiemy kilka słów o innym rozwiązaniu zespołu nastawczego. W metodach wykorzystujących indukcję, zespoły nastawcze nie muszą być związane z jakimkolwiek mechanizmem, dlatego buduje się je w formie pudełek, pudełek z rękojeścią itp. kształtów. Są one tak skonstruowane, że operator może je swobodnie przesuwać po odczytywanym rysunku rozłożonym na odpowiednim stole. Zespół nastawczy tego rodzaju może być połączony z zespołami elektronicznymi urządzenia za pomocą giętkiego przewodu. Istnieją rozwiązania, w których nie ma połączenia przewodowego. Jak już wspomniano, w zespołach nastawczych tego rodzaju najczęściej występują cewki, które są elementem emitującym energię elektromagnetyczną. Oprócz okienka i cewki w zespołach tych mogą występować przełączniki, za pomocą których operator steruje pracą urządzenia.

- Metoda z elektrycznym układem nadążnym

W metodzie tej również wykorzystuje się zjawisko indukcji elektromagnetycznej. Jednak nie ma tu siatki drutów, natomiast pod powierzchnią stołu znajduje się mechanizm kinetyczny utrzymujący karetkę z cewkami, które odbierają sygnały z głowicy zespołu nastawczego przesuwanego ręcznie przez operatora po odczytywanym rysunku. Wymienione sygnały powodują wysterowanie wzmacniaczy układu nadążnego, który automatycznie przesuwa ustalony punkt karetki dokładnie pod miejsce ustawienia okienka zespołu nastawczego. Zmiana położenia karetki jest odpowiednio rejestrowana, stąd też pochodzi informacja o położeniu okienka nastawczego.

- Metoda wykorzystująca ultradźwięki

Na krawędziach dwu prostopadłych do siebie boków odczytywanego obrazu ustawia się czujniki (mikrofony) odbierające ultradźwięki. Ostrze zespołu nastawczego o kształcie pióra jest naprowadzane przez operatora na odczytywane punkty obrazu. Wewnątrz "pióra" znajduje się generator (nadajnik) częstotliwości naddźwiękowej. Fala naddźwiękowa zostaje wypromieniowana w chwili dociśnięcia ostrza do odczytywanego obrazu. Natomiast w zespole układów elektronicznych znajdują się liczniki zliczające impulsy z generatora o stosunkowo dużej częstotliwości powtarzania (rys. 2). W momencie włączenia generatora ultradźwięków



Rys. 2. Objaśnienie ultradźwiękowej metody odczytu informacji graficznej
GZ - generator impulsów zegarowych, GI - generator impulsów wyzwalających

następuje otwarcie bramki kierującej impulsy z generatora impulsów do licznika. W chwili odebrania ultradźwięków przez czujniki następuje zatrzymanie pracy liczników. Ze stanu liczników określa się współrzędne punktu, na które jest ustawione ostrze pióra.

Powierzchnia odczytywana może osiągać wymiary 1,83 x 1,83 m. Natomiast brak jest danych o dokładności odczytu.

Poważną zaletą tej metody w stosunku do wszystkich innych znanych metod jest to, że istnieje możliwość odczytywania informacji trójwymiarowej (w tym celu urządzenie uzupełnia się jeszcze jednym zespołem czujników i odpowiednimi układami elektronicznymi). Drugą ważną zaletą jest to, że do odczytu nie potrzeba specjalnego stołu ani mechanizmu kinetycznego. W ten sposób np. można wprowadzać do EMC lub rejestrować na nośnikach akceptowanych przez komputer informacje graficzne (np. krzywe) bezpośrednio z lamp oscyloskopowych. Powyższa metoda jest zastosowana w urządzeniu Graf/Pen firmy Science Accessories Corporation.

- Metoda z elektronicznym układem nadążnym

Na ramieniu mechanizmu kinetycznego jest zamocowana kamera telewizyjna, np. typu widikon. Obraz znajdujący się przed kamerą jest wybierany w taki sam sposób, jak w systemie telewizyjnym. Odpowiednie układy elektroniczne analizują odebrane sygnały pod kątem określenia wartości nachylenia odczytywanych linii lub krawędzi zaczernionych części obrazu w stosunku do osi ekranu. Z kolei odczytana wartość nachylenia jest wykorzystywana do wysterowania napędu karetki kamery (głowicy). Opisane sprzężenie zwrotne powoduje ciągły ruch kamery (układu nadążnego) wzdłuż linii.

Obsługa urządzenia tą metodą polega na ręcznym naprowadzeniu okienka karetki w pobliże linii, która ma być odczytywana, ustawieniu kierunku ruchu i przełączeniu na pracę automatyczną.

Metoda pozwala budować urządzenia wygodne w obsłudze. Powierzchnia odczytu może być stosunkowo duża. Metoda ta daje dużą dokładność i prędkość odczytu. Natomiast z danych urządzenia OLF 1 firmy Gerber wynika, że w tym rozwiązaniu rozróżnialność jest mała. Urządzenia tego typu są bardzo kosztowne.

• Stoły

W większości rozwiązań stoły stanowią integralną część urządzeń odczytujących informację graficzną. Do nielicznych należą rozwiązania bez własnego stołu (niektóre typy z grupy małoskalowych oraz urządzenia pracujące na zasadzie wykorzystywania ultradźwięków nie muszą mieć stołu). W pewnych typach stosuje się stoły podświetlane (szyby matowe), co umożliwia dokładniejsze odczytywanie rysunków wykonanych na kalce technicznej oraz klisz filmowych.

• Ocena metod odczytu

Rozwiązania wykorzystujące mechanizm kinetyczny mają stosunkowo proste układy elektroniczne. Natomiast rozwiązania bez mechanizmów odznaczają się rozbudowanymi urządzeniami elektronicznymi. Stąd w wielu przypadkach przyjęcie takiej lub innej metody należy uważać za kompromis między możliwościami technologicznymi w dziedzinie mechaniki a stanem techniki w zakresie elektroniki danego wytwórcy. Dużą rolę odgrywają tu również koszty nakładów i opatentowanie rozwiązań.

Na podstawie danych firmowych należy uważać, że najlepsze wyniki dają metody: fotoelektryczna oraz z elektrycznym układem nadążnym. Osiąga się tu dużą dokładność odczytu (0,1 mm) i wysoką rozróżnialność (0,025 mm). Istnieją urządzenia(firm Autotrol i Instronics), które pozwalają na odczytywanie rysunków o wymiarach 1,2 x 1,5 m. W czytniku typu 3990, wykorzystującym metodę fotoelektryczną, zespół nastawczy może być przesuwany po rysunku z prędkością 2,5 m/s, natomiast w czytniku Gradicon Grafic Coordinate Digitizer wykorzystującym metodę z elektrycznym układem nadążnym z prędkością 1 m/s.

Na przykładzie czytnika Data Grid Digitizer firmy Bendix można stwierdzić, że metoda elektroniczna (stół z wtopionymi przewodami) daje prawie takie same wyniki (tabela 1).

Metoda z elektronicznym układem nadążnym, w porównaniu z trzema wyżej wymienionymi, ma pewne zalety ale także pewne wady. Pozwala na pracę automatyczną, tzn. jest o wiele mniej męcząca dla operatora. Odczyt automatyczny w urządzeniu OLF 1 firmy Gerber zachodzi z prędkością 4,58 m/s, dokładność odczytu wynosi 0,08 mm, a wymiary odczytywanych

rysunków mogą mieć 1,5 x 6,04 m. Natomiast rozróżnialność jest o wiele gorsza w porównaniu z koderami wykorzystującymi inne metody i w urządzeniu OLF 1 wynosi 0,25 mm. Ze względu na znaczny stopień złożoności cena urządzenia jest wysoka.

Brak jest informacji jaką dokładność zapewnia metoda wykorzystująca ultradźwięki. Natomiast producent urządzenia Graf/Pen GP 2 podaje rozróżnialność na 0,178 mm przy wymiarach rysunku 355 x 355 mm. Nie pozwala to zaliczyć metody ultradźwiękowej do klasy metod takich, jak fotoelektryczna, z elektrycznym układem nadążnym lub elektroniczna. Z drugiej strony ma ona wiele własności, których nie mają żadne inne metody. Przede wszystkim pozwala budować bardzo tanie urządzenie, stąd ma szanse szerokiego rozpowszechnienia. Firma SAC, produkująca Graf/Pen GP2 podaje wiele nowych dziedzin zastosowań (np. kreślenie wspomagane przez komputer), których nie miały dotychczasowe rozwiązania koordynatometrów.

Metoda potencjometryczna odznacza się małą dokładnością pomiaru. W nowoprodukowanych urządzeniach nie jest już stosowana.

4. Prace prowadzone w Polsce

W Instytucie Organizacji i Kierowania Polskiej Akademii Nauk prowadzone są prace badawcze w zakresie urządzeń rozpoznających informację graficzną [2], [4] i [5]. Zbudowany tam został system automatycznego przetwarzania informacji obrazowej CPO-1 wyposażony w przetwornik o rastrze 2^{14} elementowym i o 4 stopniach jasności. System ten jest wykorzystywany praktycznie w badaniach biologicznych i diagnostyce lekarskiej. Obecnie w Instytucie buduje się przetwornik o większej dokładności odczytu (tj. o większym rastrze i większej liczbie stopni jasności).

Budową urządzeń klasycznych zajmują się: Wojskowa Akademia Techniczna, Instytut Geodezji i Kartografii oraz Instytut Maszyn Matematycznych. W WAT wykonano urządzenie o nazwie cyfrowy wtórnik wykresów, w którym zespół nastawczy przesuwa się ręcznie. Parametry techniczne tego urządzenia są następujące:

- wymiary powierzchni odczytywanej - 1200 x 800 mm
- rozróżnialność - 0,1 mm
- wyprowadzanie informacji - taśma papierowa lub wydruk na dalekopisie

W 1970 r. w Instytucie Geodezji i Kartografii wykonano urządzenie do wprowadzania i wyprowadzania informacji graficznej o nazwie KART 2 [6]. Zespół nastawczy jest tu naprowadzany na odczytywane punkty za pomocą silników, które są włączane przez operatora. Parametry urządzenia (w reżymie odczytu):

- wymiary powierzchni odczytywanej - 800 x 800 mm
- rozróżnialność - 0,05 mm
- prędkość przesuwania głowicy - 10 lub 2 mm/s
- wyprowadzanie informacji - taśma papierowa lub wydruki na dalekopisie

5. Przykłady zastosowań

Omawiane w artykule urządzenia służą do odczytu informacji graficznej naniesionej na nośnikach takich, jak:

- wykresy
- rysunki techniczne,
- mapy,
- fotografie i obrazy występujące na ekranach rzutników.

Należy dodać, że opisana metoda wykorzystująca ultradźwięki (pozwalająca odczytywać współrzędne punktów przestrzeni trójwymiarowej) umożliwia wykonanie opisu różnego rodzaju części maszyn i detali bezpośrednio z modeli fizycznych.

Przykładami zastosowań mogą być:

- przetwarzanie danych do automatycznej produkcji elementów scalonych oraz płytek obwodów drukowanych,
- projektowanie dróg i linii kolejowych,
- rejestrowanie punktów map,
- sporządzanie szablonów w przemyśle odzieżowym i obuwniczym,
- odczytywanie fotografii w eksperymentach fizyki wysokiej energii.

Inne przykłady zastosowań podano w następnym punkcie.

6. Rozwiązania perspektywiczne

W ostatnich latach coraz więcej pisze się o urządzeniach wykorzystujących ultradźwięki. Koordynatometry tego rodzaju nie odznaczają się dużą dokładnością odczytu w porównaniu z niektórymi innymi urządzeniami, jednak mają bardzo prostą budowę, a przez to i niską cenę, co w znacznej mierze wpływa na ich rozpowszechnienie. Jak wynika z danych firmowych, urządzenie Graf-Pen (SAC-USA) wykorzystuje się m.in. w następujących systemach:

- do przetwarzania szkiców schematów elektrycznych na wykończone rysunki techniczne (Computer-Aided-Drafting),
- do sterowania numerycznego, w którym dane wejściowe są zdejmowane bezpośrednio z detali lub z rysunków warsztatowych,
- w systemach konwersacyjnych,
- do składania szpalt w czasopismach,
- w systemach redukcji informacji graficznej, takiej jak wykresy EKG i EEG oraz charakterystyki przenoszenia układów elektronicznych,
- w systemach sterowania zapasami,
- w systemach rozpoznawania znaków ręcznych,
- w radiolokacyjnych systemach rozpoznawania obiektów.

Biorąc pod uwagę tak szeroki zakres zastosowań, wydaje się, że urządzenia wykorzystujące ultradźwięki będą dość popularne w najbliższych latach.

Do perspektywicznych urządzeń wyprowadzania informacji graficznej z nośników naturalnych należy zaliczyć urządzenia, a właściwie systemy rozpoznawania informacji obrazowej. Od wielu lat w licznych krajach prowadzone są badania w tym kierunku. Jednak ze względu na złożoność problemu nie można mówić jeszcze o rozpowszechnieniu odpowiednich urządzeń. Obecnie znane są tylko pojedyncze egzemplarze, do których można zaliczyć wyżej wspominany system CPO-1.

Literatura

[1] Auerbach Computer Technological Report. Raporty nr 305.0000.010 - 305.5879.720. Tom E 2

[2] Metody automatycznego przetwarzania informacji o złożonej strukturze ze szczególnym uwzględnieniem informacji obrazowej. Prace Instytutu Cybernetyki Stosowanej PAN, zeszyt nr 6, Warszawa 1972

[3] WILSON R.A.: Optical Page Reading Devices. Reinhold Publishing Corporation 1966

[4] KULIKOWSKI J.L.: Kierunki rozwojowe i metody automatycznej klasyfikacji. Zbiór referatów z konferencji "Metody bezpośredniego wprowadzania i wyprowadzania informacji tekstowej i obrazowej w systemach informacyjnych". Jabłonna 17-19.10.1973

[5] KAPUSTO A.: Algorytm przetwarzania - szkieletyzacji obrazów binarnych. Zbiór referatów jak w pozycji [4]

[6] PODGÓRSKI R.: KART 2 - Urządzenie do zautomatyzowanego przetwarzania informacji. ETO Nowości, 1972, nr 3

ETO NOWOŚCI Nr 1 /1975

mgr Jacek WITASZEK
Instytut Maszyn Matematycznych

681.322.06;800.92

ANALIZA SKŁADNIOWA

1. Wstęp

Czytelnikowi cyklu "Jak pracuje translator" termin "analiza składniowa" nie powinien być obcy. W ramach tego cyklu ukazały się już bowiem dwa artykuły mgr T. Krawczyka traktujące o metodach analizy składniowej.

Analiza składniowa jest drugą po analizie leksykalnej fazą pracy każdego translatora; celem tego artykułu jest dalsze wprowadzanie czytelnika w tę problematykę.

2. Język i gramatyka

Czytelnik niniejszego artykułu ma na pewno jakieś wyobrażenie o tym, czym jest język. Gdyby jednak po głębszym zastanowieniu stwierdził, że ma pewne trudności w sprecyzowaniu tego pojęcia, to w encyklopedii lub słowniku pod hasłem "Język" znajdzie definicję w rodzaju: język to "zespół środków, za pomocą których ludzie porozumiewają się ze sobą, posługując się głosem lub pismem" (Mała Encyklopedia Powszechna PWN). W definicji tej główny akcent położono jednak na funkcje języka. Rzecz w tym, aby sformułować definicję, która ujmowałaby właściwości samego języka i to wyrażone w terminach matematyki. Ponadto, ponieważ w tym artykule będziemy się zajmowali językami sztucznymi, a dokładniej językami programowania, więc nie będziemy dbali o to, aby podane definicje i stwierdzenia odnosiły się również w pełni do języków naturalnych. Niemniej intuicje związane z tymi ostatnimi będą punktem wyjścia dla naszych rozważań.

Możemy się wstępnie zgodzić, że językiem nazywamy pewien zbiór sygnałów, którym odpowiadają określone znaczenia. Sygnały te, czyli elementy języka nazwiemy zdaniami. W dalszym ciągu ograniczymy nasze rozważania do specjalnego rodzaju sygnałów a mianowicie napisów. Przez napis rozumieć będziemy skończony ciąg nierozkładalnych (oczywiście z pewnego punktu widzenia) już elementów, które nazwiemy symbolami. W językach naturalnych symbolami takimi są np. słowa, sylaby, głoski itp.

Oczywiście zbiór symboli, z których utworzone są zdania danego języka jest jedną z jego charakterystycznych cech. W dalszym ciągu symbole te nazywać będziemy symbolami języka. Zakładamy, że zbiór symboli języka jest zawsze skończony. Aby zdefiniować pewien określony język musimy, zgodnie z tym co powiedziano wyżej, określić zbiór zdań tego języka oraz podać znaczenie każdego z nich. Ponieważ w tym artykule w zasadzie nie będziemy się zajmowali sprawą przypisywania znaczenia zdaniom, definicję języka sprowadzimy do zdefiniowania zbioru jego zdań. Oczywiście definiowanie polegające na wyliczeniu wszystkich zdań języka nie ma większej wartości, nie nadaje się bowiem do definiowania języków złożonych z dużej lub nieskończonej liczby zdań. Wady tej nie ma metoda definiowania zbioru zdań polegająca na wyliczeniu symboli języka oraz podaniu reguł konstruowania z nich zdań. Tego rodzaju definicja przyjmuje zazwyczaj formę tzw. gramatyki. Gramatykę określają cztery następujące jej elementy:

- skończony zbiór symboli języka, który oznaczamy przez V_T i który jest również nazywany zbiorem symboli terminalnych gramatyki,
- skończony zbiór symboli pomocniczych, tzw. symboli nieterminalnych gramatyki, który oznaczamy przez V_N; symbole terminalne jak i nieterminalne nazywamy symbolami gramatyki; zbiór symboli gramatyki oznaczamy przez V; zakłada się, że żaden z symboli gramatyki nie może jednocześnie być symbolem terminalnym i nieterminalnym,
- skończony zbiór reguł konstruowania zdań języka; reguły te noszą nazwę produkcji; do definiowania języków programowania używa się najczęściej takich gramatyk, w których każda produkcja składa się z dwóch następujących części:

-symbolu nieterminalnego,
-napisu utworzonego z symboli gramatyki[1]; (zbiór produkcji zazwyczaj oznaczamy przez P),

- symbol startowy gramatyki. Symbol ten zazwyczaj oznaczamy przez S i jest nim jeden z symboli nieterminalnych danej gramatyki.

Rolę poszczególnych elementów gramatyki wyjaśnimy w dalszym ciągu tej pracy. Gramatykę G, dla której zbiór symboli terminalnych, zbiór symboli nieterminalnych, zbiór produkcji i symbol startowy są odpowiednio V_T, V_N, P oraz S zapisywać będziemy w formie czwórki:$<V_T, V_N, P, S>$. Produkcje gramatyki zapisywać będziemy w następujący sposób:
symbol nieterminalny $\longrightarrow$ napis utworzony z symboli gramatyki

Ze względu na tę postać zapisu symbol nieterminalny stanowiący pierwszą część produkcji będziemy nazywać jej lewą stroną, natomiast napis stanowiący drugą część produkcji - jej prawą stroną. Symbole nieterminalne będziemy na razie zapisywać w formie ciągów literowo-cyfrowych zamkniętych w ostre nawiasy: <oraz> . Ciągi te dobiera się w ten sposób, aby dodatkowo podawały zrozumiałe dla czytającego informacje o reprezentowanych przez nie symbolach, np.: <blok>, <wyrażenie>, <rzeczownik> itp.

Symbole terminalne będziemy zapisywać w takiej postaci w jakiej występują one w języku. Zazwyczaj są to bądź pojedyncze znaki pisarskie, bądź ich sekwencje, np.: A, a, :=, KOT, begin, 'GREATER THAN' itp. Należy tutaj pamiętać, że każdy symbol stanowi jedną całość; w związku z tym np. symbolu KOT nie należy traktować jako napisu złożonego z symboli K, O, oraz T. Ta forma zapisu, zwana też notacją BNF, została zaproponowana przez Backusa i Naura i wykorzystana po raz pierwszy do opisu języka ALGOL 60[2].

---

[1] Gramatyki, w których produkcje mają taką strukturę, noszą nazwę gramatyk bezkontekstowych. Oprócz takich gramatyk znane są również i inne, jak np. gramatyki regularne, gramatyki kontekstowe, gramatyki dwupoziomowe.

[2] Korzystamy tutaj z uproszczonej notacji BNF - nie uwzględniliśmy bowiem możliwości łączenia kilku produkcji o tych samych lewych stronach.

Pokażemy teraz w jaki sposób na podstawie gramatyki można skonstruować zdania języka, który ta gramatyka definiuje. Dodamy tutaj, że język definiowany przez gramatykę G przyjęto oznaczać przez L(G).

Proces konstruowania zdań języka nazywa się procesem wywodzenia. Algorytm, według którego ten proces przebiega jest następujący:

1) Tworzymy napis złożony jedynie z symbolu startowego S

2) a) W utworzonym ostatnio napisie (na początku jest to napis złożony z symbolu startowego) wybieramy jeden symbol nieterminalny,
   b) w zbiorze produkcji P wybieramy dowolną produkcję, której lewa strona jest równa wybranemu w a) symbolowi nieterminalnemu,
   c) tworzymy nowy napis zastępując w napisie dotychczas rozważanym symbol wybrany w a) prawą stroną produkcji wybranej w b).

Czynności opisane w p. 2 powtarzamy tak długo aż napis otrzymany w p. 2c składa się wyłącznie z symboli terminalnych. Język L(G), tj. język definiowany przez gramatykę G, składa się dokładnie z takich napisów, które dadzą się na podstawie tej gramatyki skonstruować według podanego wyżej algorytmu.

Opisaną w p. 2c czynność polegającą na zastąpieniu pewnego symbolu prawą stroną produkcji, w której ten symbol stanowi lewą stronę, nazywać będziemy rozwijaniem tego symbolu, natomiast czynność odwrotną nazywać będziemy zwijaniem lub redukcją.

Fragment procesu wywodzenia złożony z sekwencji czynności opisanych pod 2a, 2b i 2c nazywać będziemy krokiem tego procesu.

Zauważmy, że konstruując zdanie według powyższego algorytmu otrzymujemy dodatkowo pewien ciąg napisów. Pierwszym elementem tego ciągu jest napis złożony z samego tylko symbolu startowego, ostatnim - konstruowane zdanie. Ciąg ten nazywać będziemy wywodem danego zdania a poszczególne napisy formami zdaniowymi. Ścisłą definicję obu tych pojęć podamy niżej, teraz natomiast podamy przykłady dwóch języków oraz ich gramatyk. Już tutaj należy sobie jednak zdać sprawę, że zarówno wywód jak i poszczególne formy zdaniowe zależą od gramatyki, której produkcja

i symbole były wykorzystywane w procesie wywodzenia. Z tego też względu mówiąc o wywodzie lub formie zdaniowej należy mówić o wywodzie lub formie zdaniowej w określonej gramatyce. Niemniej tam, gdzie z góry wiadomo jaką gramatykę mamy na myśli, będziemy mówili po prostu o wywodzie lub formie zdaniowej.

Przykład 1

Jednym z typów zdania prostego w języku polskim jest zdanie złożone z podmiotu, orzeczenia i dopełnienia, przy czym podmiot i dopełnienie są rzeczownikami odpowiednio w mianowniku i bierniku, zaś orzeczenie jest czasownikiem.

Weźmy pod uwagę język - nazwiemy go L1 - złożony tylko z takich zdań, z tym, że dla uproszczenia założymy, że występują w nim tylko dwa rzeczowniki KOT i KOGUT oraz dwa czasowniki - GONI i DRAPIE. Zauważmy, że dla obu tych rzeczowników biernik można otrzymać z mianownika dodając do tego ostatniego końcówkę A, mamy bowiem KOT - KOTA, KOGUT - KOGUTA. Język L1 składa się zatem ze zdań: KOT GONI KOGUTA, KOGUT DRAPIE KOTA, itd.

Aby podać gramatykę języka L1 musimy oczywiście określić wszystkie cztery jej elementy. Rozpoczniemy od określenia zbioru symboli terminalnych V_T. Do zbioru tego należą oczywiście słowa KOT, KOGUT, GONI, DRAPIE, należy do niego również głoska A a ponadto dołączymy do niego kropkę oznaczającą koniec zdania oraz odstęp, który będziemy oznaczali przez ␣.

Zbiór symboli nieterminalnych V_N składa się z następujących symboli:

<zdanie>, <podmiot>, <orzeczenie>, <dopełnienie>, <rzeczownik> oraz <czasownik>.

Zbiór produkcji składa się z następujących produkcji:

<zdanie> ⟶ <podmiot> ␣ <orzeczenie> ␣ <dopełnienie> (1)

<podmiot> ⟶ <rzeczownik> (2)

<rzeczownik> ⟶ KOT (3)

<rzeczownik> ⟶ KOGUT (4)

<orzeczenie> ⟶ <czasownik> (5)

<czasownik> ⟶ GONI (6)

<czasownik> ⟶ DRAPIE (7)

<dopełnienie> ⟶ <rzeczownik> A (8)

Liczby w nawiasach zapisane na prawo od każdej produkcji są numerami, którymi będziemy się posługiwać w podanym niżej przykładzie wywodu.

Symbolem startowym jest symbol <zdanie> .

Określoną w powyższy sposób gramatykę oznaczymy przez G1. Na rys. 1 podany został wywód zdania: "KOT GONI KOGUTA" w gramatyce G1. W kolejnych formach zdaniowych tego wywodu zaznaczono rozwijany symbol pisząc pod nim numer produkcji, według której został on rozwinięty.

<zdanie>
1
<podmiot> ␣ <orzeczenie> ␣ <dopełnienie> .
5
<podmiot> ␣ <czasownik> ␣ <dopełnienie> .
2
<rzeczownik> ␣ <czasownik> ␣ <dopełnienie> .
KOT ␣ <czasownik> ␣ <dopełnienie> .
8
KOT ␣ <czasownik> ␣ <rzeczownik> A.
6
KOT ␣ GONI ␣ <rzeczownik> A.
4
KOT ␣ GONI ␣ KOGUTA.

Rys. 1. Przykład wywodu zdania w gramatyce G1

Jak nietrudno zauważyć gramatyka G1 jest rzeczywiście gramatyką języka L1 złożonego z ośmiu zdań. Skonstruowanie tej gramatyki nie nastręczało nam większych trudności. W ogólnym jednak przypadku skonstruowanie gramatyki dla danego języka nie jest sprawą tak łatwą. Należy też zauważyć, że dla danego języka można znaleźć więcej gramatyk. Np. gdybyśmy w naszej gramatyce G1 usunęli symbol nieterminalny <czasownik> oraz produkcje od 5 do 7, wprowadzając jednocześnie produkcje: <orzeczenie> ⟶ GONI i <orzeczenie> ⟶ DRAPIE otrzymalibyśmy inną gramatykę tego samego języka L1.

Przykład 2

Rozpatrzmy teraz język, którego zdaniami są proste wyrażenia arytmetyczne złożone z liczb, nawiasów oraz znaków operacji. Dla uproszczenia

przyjmiemy, że jako znaki operacji występują jedynie + i * (znak mnożenia), a jako liczba tylko 1. Język ten oznaczymy przez L2. Jego zdaniami są np. takie wyrażenia jak 1, (1), 1*(1+1*1) itp. Język L2 definiuje gramatyka G2 = $\langle V_T, V_N, P, S \rangle$, dla której

$$V_T = \{ 1, +, *, (,) \},$$

$V_N = \{$ <wyrażenie> , <składnik> , <czynnik> $\}$

P = $\{$ <wyrażenie> → <wyrażenie> + <składnik> , (1)

<wyrażenie> → <składnik> , (2)

<składnik> → <składnik> * <czynnik> , (3)

<składnik> → <czynnik> , (4)

<czynnik> → 1, (5)

<czynnik> → (<wyrażenie>) $\}$ (6)

S = <wyrażenie>

Na rys. 2 przedstawiono wywód zdania 1+1 w podanej powyżej gramatyce G2. Tak jak na rys. 1 tak i tutaj pod symbolem, który został rozwinięty w danym kroku procesu wywodzenia zapisano numer produkcji, którą zastosowano przy jego rozwijaniu.

<wyrażenie>
1
<wyrażenie> + <składnik>
4
<wyrażenie> + <czynnik>
2
<składnik> + <czynnik>
4
<czynnik> + <czynnik>
5
<czynnik> + 1

1 + 1

Rys. 2. Przykład wywodu w gramatyce G2

Czytelnik bez trudu może zauważyć, że przedstawiony na rys. 2 wywód nie jest jedynym wywodem zdania 1+1. Przykład innego wywodu zdania 1+1 został przedstawiony na rys. 3. Ta niejednoznaczność wywodu tego samego zdania w tej samej gramatyce jest konsekwencją faktu, że w p. 2a algorytmu konstruowania zdań, możemy rozwijany symbol wybrać w dowolny sposób. Aby uniknąć tej niejednoznaczności można się umówić, że w p. 2a każdego kroku procesu wywodu będzie wybierany pierwszy od prawej sym-

bol nieterminalny danej formy zdaniowej. Otrzymany w ten sposób wywód nazywamy wywodem prawostronnym, a każdą występującą w nim formę zdaniową - prawostronną formą zdaniową. Fragment prawostronnej formy zdaniowej, który powstał w wyniku rozwinięcia symbolu nieterminalnego wybranego w p. 2a, w tym samym kroku procesu wywodzenia, w którym została utworzona ta forma zdaniowa - nazywamy jej rdzeniem. Wywód wyrażenia 1+1 przedstawiony na rys. 3 jest właśnie wywodem prawostronnym. Na rysunku tym podkreślono w kolejnych formach zdaniowych ich rdzenie.

<wyrażenie>
<wyrażenie> + <składnik>
<wyrażenie> + <czynnik>
<wyrażenie> + 1
<czynnik> + 1
1 + 1

Rys. 3. Przykład prawostronnego wywodu zdania w gramatyce G2

Stosowane dotychczas oznaczenia symboli terminalnych i nieterminalnych, aczkolwiek niezwykle przydatne w przypadku praktycznych zastosowań, nie są wygodne w przypadku jedynie teoretycznych rozważań, wymagają bowiem zbyt wiele pisania. Ze względu na oszczędność zapisu wprowadzimy nieco inne oznaczenia stosowane również i przez innych autorów.

Umówimy się mianowicie oznaczać symbole nieterminalne dużymi literami z początku alfabetu, a więc np. A, B, C, symbole terminalne małymi literami z początku alfabetu, a więc a, b, c a także cyframi lub pojedynczymi znakami pisarskimi np. 1, *, 0, :, itp. Oczywiście teraz postać symboli nieterminalnych nie będzie miała dodatkowych walorów informacyjnych, a symbole terminalne nie będą mogły się składać z większej liczby znaków. Ponieważ jednak będziemy się posługiwali przykładami prostych języków i gramatyk, ta oszczędna notacja zupełnie nam wystarczy.

Aby oswoić czytelnika z tą nową formą zapisu symboli podamy jeszcze raz gramatykę definiującą język L2. Gramatykę tę oznaczymy przez G2', a to w celu podkreślenia faktu, że różni się ona od G2 jedynie innymi oznaczeniami symboli nieterminalnych.

$G2' = \langle V_T, V_N, P, A \rangle$

$V_T = \{1, +, *, (,)\}$

$V_N = \{A, B, C\}$

$P = \{$

$A \rightarrow A+B,$ (1)

$A \rightarrow B,$ (2)

$B \rightarrow B * C,$ (3)

$B \rightarrow C,$ (4)

$C \rightarrow 1$ (5)

$C \rightarrow (A)\ \}$ (6)

Poza omówioną powyżej zmianą notacji wprowadzimy jeszcze i dalsze oznaczenia. I tak symbole gramatyki, tj. elementy $V = V_T \cup V_N$, będziemy oznaczać dużymi literami z końca alfabetu, a więc np. Z, X, Y, itd.

Dowolne napisy utworzone z symboli terminalnych danej gramatyki (niekoniecznie muszą to być zdania języka definiowanego przez tę gramatykę) oznaczać będziemy małymi literami z końca alfabetu, a więc z_j, x, y, w_3, itd.

Ponieważ w lingwistyce matematycznej zbiór wszystkich napisów utworzonych z elementów pewnego zbioru U oznacza się przez U^* (szczególnie do U^* należy również napis pusty) więc zbiór wszystkich napisów utworzonych z symboli zbioru V_T oznaczymy przez V_T^*. Zgodnie z tym oznaczeniem, konwencję odnoszącą się do napisów utworzonych z symboli terminalnych możemy wypowiedzieć następująco: elementy V_T^* oznaczać będziemy małymi literami z końca alfabetu, a więc np. z_j, x, y, w_3, itd.

Elementy V^* (gdzie V oznacza zbiór symboli gramatyki, tj. $V = V_N \cup V_T$) będziemy oznaczali małymi literami z początku alfabetu greckiego, a więc α, β, γ, itd.

Wróćmy teraz jeszcze raz do podanego w tym punkcie algorytmu konstruowania zdań, a szczególnie do p. 2 tego algorytmu i weźmy pod uwagę dowolną gramatykę $G = \langle V_T, V_N, P, S \rangle$ oraz zbiór $V = V_N \cup V_T$. Jeżeli α i β są dwoma napisami z V^* i β powstał z α przez zastosowanie czynności opisanych w p. 2 wspomnianego wyżej algorytmu, to fakt ten będziemy zapisywać jako $\alpha \underset{G}{\Rightarrow} \beta$ lub krócej, gdy wiadomo o jakiej gramatyce mowa $\alpha \Rightarrow \beta$.

Jeżeli natomiast α i β są dwoma napisami należącymi do V^*, dla których istnieje taki ciąg należących do V^* napisów $\alpha_1, \alpha_2, \ldots \alpha_k$ gdzie $k \geqslant 1$, że $\alpha_1 = \alpha$, $\alpha_k = \beta$

oraz $\alpha_i \underset{G}{\Longrightarrow} \alpha_{i+1}$ dla $i = 1, 2, \ldots, k-1$,

to wówczas będziemy mówili, że α wywodzi β w gramatyce G β lub β wywodzi się w gramatyce G z α i zapiszemy ten fakt jako $\alpha \overset{*}{\underset{G}{\Longrightarrow}} \beta$ lub krócej $\alpha \overset{*}{\Longrightarrow} \beta$

Zwróćmy tutaj uwagę na to, że ponieważ k może być równe 1 więc dla każdego napisu α należącego do V^* mamy $\alpha \overset{*}{\Longrightarrow} \alpha$.

Wprowadzone tu oznaczenia wykorzystamy do podania definicji formy zdaniowej oraz języka.

Otóż o napisie $\alpha \in V^*$ mówimy, że jest formą zdaniową w gramatyce G jeżeli $S \overset{*}{\underset{G}{\Longrightarrow}} \alpha$. Natomiast językiem definiowanym przez gramatykę G nazywamy zbiór wszystkich takich $w \in V_T$, dla których zachodzi $S \overset{*}{\underset{G}{\Longrightarrow}} w$.

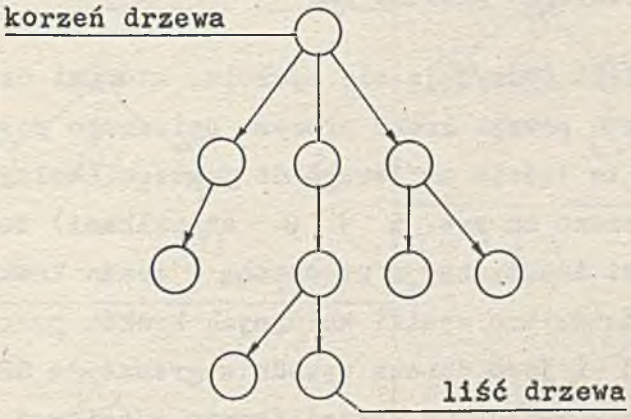
Tak więc w wyniku rozważań przeprowadzonych w tym punkcie wiemy co nazywamy językiem, a co gramatyką i jaki jest ich wzajemny stosunek. W następnym punkcie zajmiemy się pewnymi konsekwencjami tej zależności.

3. Drzewo wywodu i analiza składniowa

Jak wspomnieliśmy na początku poprzedniego punktu, z każdym zdaniem związane jest pewne znaczenie. Opierając się na tzw. regułach semantycznych znaczenie zdań wyprowadza się na podstawie znaczenia poszczególnych symboli oraz informacji o tym, według jakich produkcji były rozwijane poszczególne symbole nieterminalne w procesie wywodzenia tego zdania. Czytelnika, którego zainteresuje ta sprawa, odsyłamy do dalszych artykułów tego cyklu. Niemniej jednak nie powinien budzić wątpliwości fakt, że np. spośród dwóch zdań języka L2, w których występują symbole KOT, KOGUT oraz GONI, zdanie, w którym symbol KOT wywodzi się z podmiotu a symbol KOGUT wywodzi się z dopełnienia - ma zupełnie inne znaczenie niż zdanie, w którym symbol KOGUT wywodzi się z podmiotu, a symbol KOT z dopełnienia.

Wspomniane wyżej informacje o tym jak były rozwijane poszczególne zmienne w procesie wywodzenia zapisuje się zazwyczaj w formie tzw. drzewa wywodu. Ogólnie drzewem nazywamy twór złożony z pewnej liczby węzłów (zapisywanych w formie kółek) połączonych skierowanymi gałęziami (zapisywanymi w formie strzałek). Do każdego węzła drzewa - z wyjątkiem jednego, do którego nie wchodzi żadna - musi wchodzić zawsze jedna gałąź. Natomiast wychodzić może z każdego węzła dowolna liczba gałęzi; w szczególnym przypadku może z węzła nie wychodzić żadna gałąź.

Jedyny węzeł, do którego nie wchodzi żadna gałąź nazywamy korzeniem drzewa, zaś węzły z których nie wychodzi żadna gałąź nazywamy jego liśćmi. Przykład drzewa przedstawiono na rys. 4.



Rys. 4. Przykład drzewa

Jeżeli $G = \langle V_T, V_N, P, S \rangle$ jest pewną gramatyką i z jest zdaniem języka L(G) to drzewo nazywamy drzewem wywodu zdania z w gramatyce G, jeżeli każdy węzeł tego drzewa jest oznaczony pewnym symbolem tej gramatyki (symbole, którymi oznaczono poszczególne węzły będziemy zapisywali wewnątrz kółek przedstawiających te węzły) i jeżeli drzewo to można skonstruować posługując się opisanym poniżej algorytmem. Algorytm ten jest faktycznie rozszerzeniem podanego wcześniej algorytmu konstruowania zdania i w związku z tym podane poniżej czynności należy traktować jako uzupełnienie czynności podanych w tym ostatnim. W ten sposób konstruowanie drzewa wywodu będzie się odbywało równocześnie z konstruowaniem samego zdania.

Po wykonaniu czynności opisanej w p. 1 algorytmu konstruowania zdania tworzymy węzeł, który oznaczamy symbolem startowym gramatyki. Węzeł

ten stanowi korzeń konstruowanego drzewa wywodu. Następnie w każdym kroku procesu wywodzenia, po wykonaniu czynności opisanych w punktach 2a, 2b i 2c algorytmu konstruowania zdania - wykonujemy jeszcze następujące czynności:

2d) w dotychczas utworzonym drzewie znajdujemy liść odpowiadający symbolowi nieterminalnemu wybranemu w punkcie 2a,

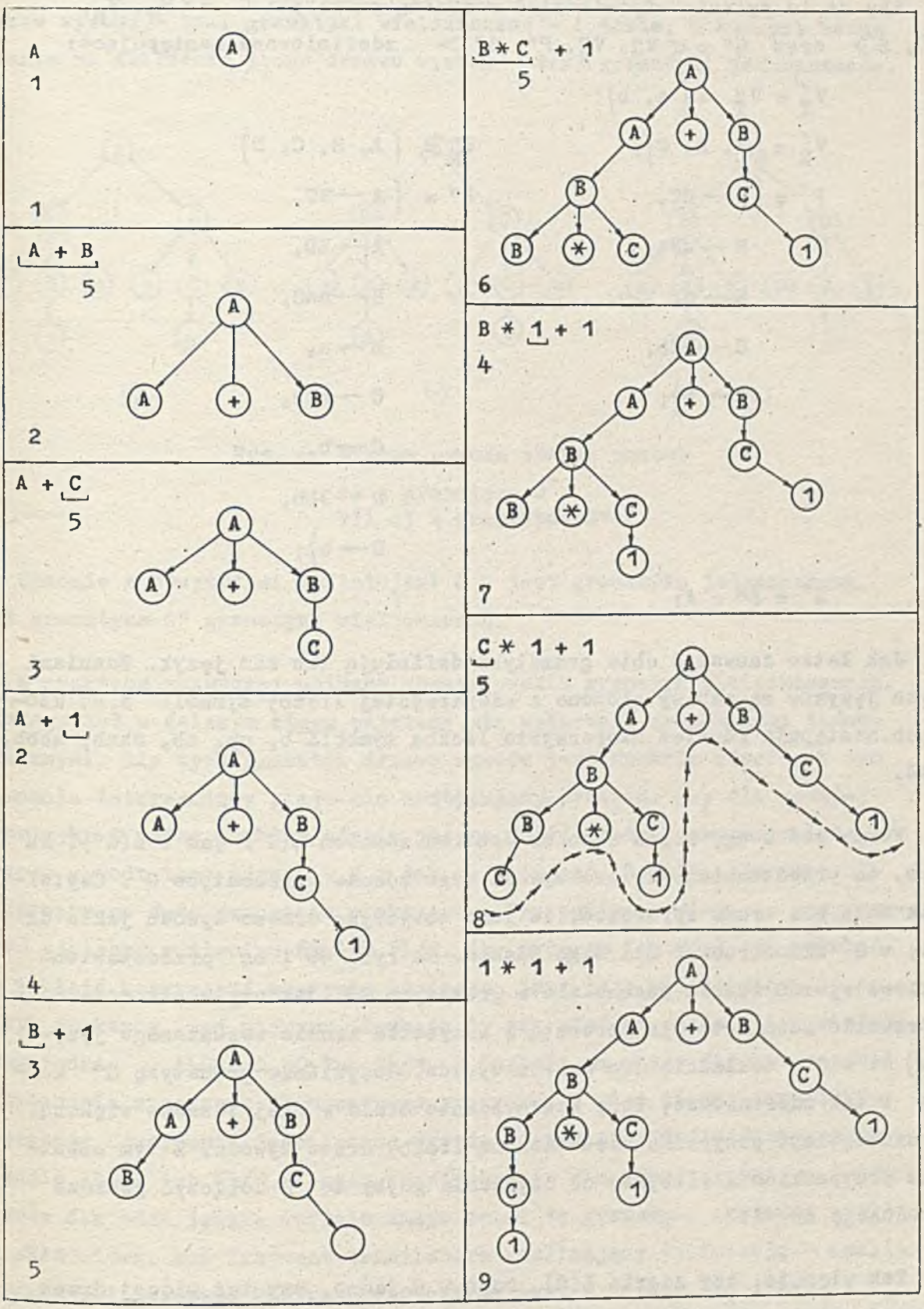
2e) tworzymy tyle nowych węzłów z ilu symboli składa się prawa strona wybranej w punkcie 2b produkcji i oznaczamy je poczynając od lewego, kolejnymi symbolami prawej strony tej produkcji,

2f) z węzła określonego w punkcie 2d prowadzimy gałęzie do wszystkich węzłów utworzonych w punkcie 2e.

Zauważmy tutaj, że jeżeli odczytuje się symbole, którymi oznaczono liście drzewa utworzonego w pewnym kroku procesu opisanego powyższym algorytmem, przebiegając te liście od lewego do prawego (kolejność przebiegania liści zaznaczono na rys. 5 i 8 strzałkami) to odczytany w ten sposób napis jest identyczny z utworzoną w tymże kroku formą zdaniową. Na rys. 5 przedstawiono wyniki kolejnych kroków procesu konstruowania zdania 1* 1 + 1 i jego drzewa wywodu w gramatyce G2'. Podobnie, jak w poprzednich przykładach, w każdej formie zdaniowej pod symbolem wybranym w p. 2a zapisano numer produkcji wybranej w p. 2b. Ponieważ podany na rys. 5 wywód jest wywodem prawostronnym, dodatkowo w każdej formie zdaniowej podkreślono jej rdzeń.

Ze sposobu konstruowania drzewa wywodu wynika, że nie zależy ono od tego, który z możliwych symboli nieterminalnych rozwinięto w każdym z kolejnych kroków wywodu. Może się o tym czytelnik przekonać konstruując np. drzewa wywodu zdania 1* 1 + 1 na podstawie innych wywodów niż wywód prawostronny. W istotny natomiast sposób drzewo wywodu zależy od gramatyki, której symbole i produkcje były wykorzystywane w procesie jego konstruowania. Z tego względu mówiąc o drzewie wywodu, mówimy zawsze o drzewie wywodu w określonej gramatyce.

Na podstawie tego co powiedziano dotychczas można wysnuć wniosek, że każde zdanie danego języka L(G) ma w G co najmniej jedno drzewo wywodu. Zachodzi jednak pytanie, czy dokładnie jedno, czy też więcej?



Rys. 5. Kolejne etapy konstruowania zdania 1 * 1 + 1 i jego drzewa wywodu

Aby na to pytanie odpowiedzieć rozważmy gramatyki $G' = \langle V'_T, V'_N, P', S' \rangle$ oraz $G'' = \langle V''_T, V''_N, P'', S'' \rangle$ zdefiniowane następująco:

$V'_T = V''_T = \{a, b\}$;

$V'_N = \{A, B, C\}$;

$P' = \{A \rightarrow BC,\ B \rightarrow aBa,\ B \rightarrow a,\ C \rightarrow bCb,\ C \rightarrow b\}$;

$V''_N = \{A, B, C, D\}$

$P'' = \{A \rightarrow BC,\ A \rightarrow BD,\ B \rightarrow aaB,\ B \rightarrow a,\ C \rightarrow bCb,\ C \rightarrow b,\ D \rightarrow Dbb,\ D \rightarrow b\}$;

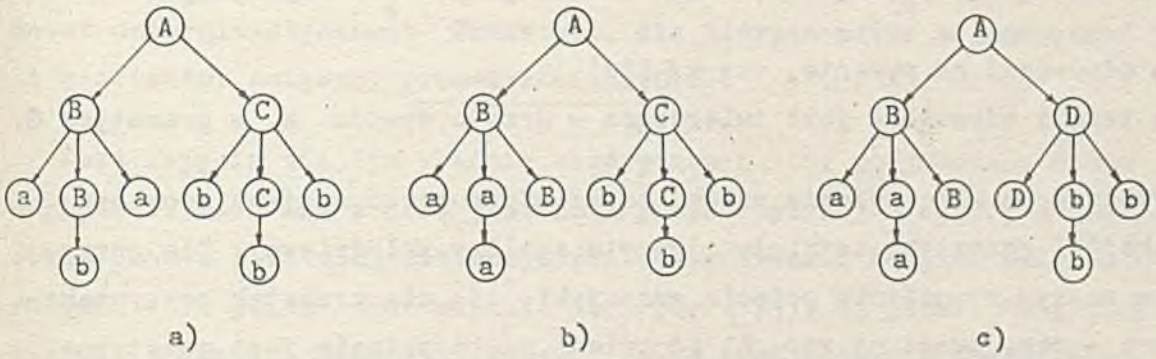
$S' = S'' = A$;

Jak łatwo zauważyć obie gramatyki definiują ten sam język. Zdaniami tych języków są napisy złożone z nieparzystej liczby symboli a, po których następuje również nieparzysta liczba symboli b, np. ab, aaab, abbb, itd.

Weźmy pod uwagę napis aaabbb. Jest on zdaniem $L(G')$ jak i $L(G'')$. Na rys. 6a przedstawiono drzewo wywodu tego zdania w gramatyce G'. Czytelnik może bez trudu sprawdzić, że jest to jedyne drzewo wywodu jakie da się w G' skonstruować dla tego zdania. Na rys. 6b i 6c przedstawiono drzewa wywodu zdania aaabbb ale w gramatyce G". Jak czytelnik może sprawdzić zdanie to (jak zresztą i wszystkie zdania rozważanego języka) ma w G" dokładnie dwa drzewa wywodu. Oczywiście gramatykę G" można i tak zdefiniować, żeby każde zdanie miało w niej jeszcze większą, w szczególnym przypadku nieskończoną liczbę drzew wywodu. Z tym ostatnim przypadkiem mielibyśmy do czynienia gdyby do P dołączyć jeszcze produkcję $A \rightarrow A$.

Tak więc to, czy zdania $L(G)$ mają w G jedno, czy też więcej drzew wywodu zależy wyłącznie od samej gramatyki G. Z tego też względu gramatyki podzielono na takie, w których to samo zdanie może mieć wiele

drzew wywodu - tzw. gramatyki wieloznaczne - i takie, w których każde zdanie ma dokładnie jedno drzewo wywodu - tzw. gramatyki jednoznaczne.



Rys. 6. Drzewa wywodu zdania aaabbb

a) w gramatyce G'
b)i c) w gramatyce G"

Zgodnie z powyższymi definicjami G' jest gramatyką jednoznaczną, zaś gramatyka G" gramatyką wieloznaczną.

W praktyce zazwyczaj unikamy konstruowania gramatyk wieloznacznych. Dlatego też w dalszym ciągu zajmiemy się wyłącznie gramatykami jednoznacznymi. Dla tych gramatyk drzewo wywodu jest funkcją zdania. W tym momencie interesujący staje się następujący problem: czy dla zadanej gramatyki G mając dowolne zdanie języka L(G), można zawsze znaleźć drzewo wywodu tego zdania w G i jak to osiągnąć. Rozwiązanie tego zagadnienia ma duże znaczenie praktyczne. Przypuśćmy bowiem, że mamy program napisany w Algol-u 60 lub PL/1. Aby program ten mógł się wykonać na jakiejś konkretnej maszynie konieczne jest jego przetłumaczenie na język rozkazów tej maszyny. Funkcję tę jak wiadomo spełnia translator odpowiednio ALGOL-u 60 lub PL/1. Z funkcją tą wiąże się konieczność rozpoznania znaczenia tłumaczonego programu, a więc między innymi konieczność utworzenia jego drzewa wywodu w gramatyce definiującej odpowiednio ALGOL lub PL/1. Proces konstruowania w zadanej gramatyce drzew wywodu dla zdań języka definiowanego przez tę gramatykę nazywamy analizą składniową, zaś fragment translatora realizujący tę funkcję - analizatorem składniowym. Ponieważ w praktyce zawsze należy się liczyć z faktem, że programista pisząc program może popełnić błędy i omyłki, więc w ramach analizy składniowej sprawdza się również, czy napis bę-

dący przedmiotem tego procesu jest w ogóle programem, tj. zdaniem określonego języka programowania. Tak więc przez analizę składniową rozumieć będziemy proces, w wyniku wykonania którego dla zadanej gramatyki $G = \langle V_T, V_N, P, S \rangle$, i dla dowolnego $z \in V_T$ otrzymujemy:

- odpowiedź na pytanie, czy $z \in L(G)$,
- jeżeli odpowiedź jest twierdząca - drzewo wywodu z w gramatyce G.

Postawione więc pytanie możemy teraz sformułować następująco: czy dla każdej gramatyki istnieje algorytm analizy składniowej. Dla gramatyk w naszym rozumieniu pojęcia gramatyki, tj. dla gramatyk bezkontekstowych - por. uwaga na str. 83 odpowiedź na to pytanie jest pozytywna. Dowodu odpowiedniego twierdzenia nie będziemy tutaj przeprowadzali odsyłając czytelnika do odpowiedniej literatury [1] lub [2]. Ta pozytywna odpowiedź nie przesądza oczywiście sprawy, czy algorytmy takie istotnie są znane i czy można je praktycznie wykorzystać w translatorze.

Należy jednak tutaj stwierdzić, że rzeczywiście znaleziono wiele algorytmów analizy składniowej oraz opracowano metody konstruowania ich na podstawie zadanej gramatyki (np. [3] i [4]).

Pośród tych algorytmów szczególną rolę odgrywają algorytmy, które wypełniają swoje zadanie przeglądając analizowany napis tylko raz np. od lewej do prawej.

Analizator syntaktyczny działający na podstawie takiego algorytmu jest bowiem efektywny zarówno pod względem czasu (jednorazowe przeglądanie zdania) jak i pod względem wykorzystanej pamięci (nie trzeba bowiem analizowanego zdania umieszczać w pamięci maszyny). Niestety okazuje się, że nie dla każdej gramatyki da się skonstruować algorytm o tej własności. Gramatyki, dla których taki algorytm istnieje nazywamy gramatykami deterministycznymi[1]. Wiele tego typu algorytmów opisano w pracy [4]. Tutaj natomiast omówimy tzw. algorytm LR(0) podany przez D. Knutha [5].

---

[1] Nazwa ta pochodzi stąd, że dla takiej gramatyki można skonstruować analizator w formie deterministycznego automatu ze stosem por. np. [2].

4. Algorytm LR (0)

Na samym wstępie musimy stwierdzić, że omówiony w tym punkcie algorytm LR(0) nie da się zastosować w przypadku wszystkich gramatyk i to nawet deterministycznych. Gramatyki, dla których można skonstruować taki analizator nazywamy gramatykami LR(0).

Fakt zajęcia się tym właśnie algorytmem został podyktowany dwoma względami. Pierwszy z nich jest natury dydaktycznej - algorytm ten posiada proste i intuicyjnie oczywiste uzasadnienie. Drugi z nich ma charakter teoretyczny: otóż okazuje się (por. [5]), że jeżeli dany język posiada gramatykę deterministyczną, to po ewentualnym wprowadzeniu do tego języka niewielkich zmian można dla niego również znaleźć gramatykę LR(0). Zmiany, o których mowa, sprowadzają się do wprowadzenia specjalnego symbolu służącego do oznaczania końca zdań. To uzupełnienie nie jest potrzebne jeżeli w danym języku jeden z jego symboli pełni już tę funkcję, jak np. L1, gdzie koniec zdania jest zaznaczany kropką.

Na początku rozważań, które doprowadzą nas do wspomnianego algorytmu LR(0), zastanówmy się, jak w ogóle może przebiegać proces analizy syntaktycznej. Otóż na razie wiemy jak konstruować drzewo wywodu zdania w ramach procesu konstruowania samego tego zdania. Zauważmy jednak, że konstrukcji drzewa dokonujemy jedynie na podstawie informacji: który liść drzewa w każdym kroku procesu konstruowania drzewa się "rozrósł" i którą produkcję zastosowano do tego celu.

Jeżeli zaś ograniczyć się do wywodów prawostronnych, to wówczas wystarcza tylko ta druga informacja. Tak więc w przypadku wywodu prawostronnego dla skonstruowania drzewa wywodu wystarczy nam jedynie znajomość ciągu produkcji zastosowanych w kolejnych krokach procesu wywodzenia. Możemy bowiem wtedy w podanym poprzednio algorytmie, w każdym kroku pominąć punkty 2a, 2b i 2c związane wyłącznie z konstruowaniem kolejnych form zdaniowych wywodu, w p. 2d wybrać pierwszy z prawej liść oznaczony symbolem nieterminalnym, a w p. 2e zastosować kolejną produkcję wspomnianego wyżej ciągu. Tak więc problem skonstruowania drzewa wywodu danego zdania sprowadza się do znalezienia owego ciągu produkcji.

Jak nietrudno zauważyć, to ostatnie zagadnienie moglibyśmy rozwiązać gdyby była znana metoda znajdowania rdzenia prawostronnej formy zdanio-

wej oraz produkcji, według której został on rozwinięty. Biorąc najpierw pod uwagę analizowane zdanie znaleźlibyśmy jego rdzeń oraz produkcję, którą zastosowano przy jego tworzeniu. Produkcja ta byłaby oczywiście ostatnią produkcją poszukiwanego przez nas ciągu. Jednocześnie zastępując w naszym zdaniu rdzeń lewą stroną znalezionej produkcji (tj. redukując rdzeń) otrzymalibyśmy przedostatnią formę zdaniową prawostronnego wywodu tego zdania. Postępując z tą formą zdaniową w identyczny sposób znaleźlibyśmy przedostatnią produkcję potrzebnego nam ciągu produkcji itd.

Jak widać, tym sposobem bylibyśmy w stanie, biorąc za punkt wyjścia jedynie analizowane zdanie, najpierw znaleźć ciąg produkcji użytych przy jego wywodzeniu, a następnie na tej podstawie skonstruować drzewo wywodu. Opisana wyżej procedura postępowania jest charakterystyczna dla wielu algorytmów analizy składniowej, które od faktu, że w każdym kroku następuje redukcja rdzenia nazywa się algorytmami typu redukcyjnego. Inna nazwa to algorytmy analizujące od "dołu do góry" lub z angielska "bottom up". Różnią się natomiast te algorytmy metodą znajdowania rdzenia i produkcji użytej do jego utworzenia [4].

Prezentowany algorytm LR(0) jest również algorytmem typu redukcyjnego. W algorytmie tym,w celu znalezienia rdzenia w prawostronnej formie zdaniowej oraz wspomnianej wyżej produkcji,przegląda się tę formę zdaniową symbol po symbolu od lewej do prawej. W trakcie przeglądania, posługując się utworzonymi na podstawie gramatyki tablicami, stwierdza się, czy badany symbol jest ostatnim symbolem rdzenia czy też nie. W pierwszym przypadku dalsze przeglądanie danej formy zdaniowej zostaje zakończone, a odpowiednia tablica podaje produkcję, która została zastosowana w celu utworzenia rdzenia i według której należy teraz ten rdzeń zredukować. W drugim natomiast przypadku przystępuje się do zbadania następnego symbolu.

Jak z tego wynika, niezbędnym warunkiem wykorzystania algorytmu LR(0) jest skonstruowanie tablic stanowiących podstawę jego działania. Zanim podamy pełny algorytm konstruowania tych tablic przeprowadzimy pewne wstępne rozważania posługując się przykładem prostej gramatyki G3 zdefiniowanej następująco:

$V_T = \{a, b\}$

$V_N = \{A, B, C\}$

$P = \{A \longrightarrow BC,$ (1)

$B \longrightarrow b,$ (2)

$B \longrightarrow aB,$ (3)

$C \longrightarrow bb\}$ (4)

$S = A.$

Zauważmy tutaj, że to, czy symbol występujący na danej pozycji formy zdaniowej jest rdzeniem, czy też nie, zależy od tego, w wyniku zastosowania jakich produkcji symbol ten tam się znalazł. I tak np. napisy Bbb oraz abbb są prawostronnymi formami zdaniowymi w gramatyce G3. Ich prawostronne wywody są odpowiednio:

A, BC, Bbb oraz A, BC, Bbb, aBbb, abbb - pod rozwijanym symbolem nieterminalnym wypisano numer użytej produkcji.
1 4 1 4 3 2

W obu tych formach zdaniowych na drugiej pozycji występuje symbol b ale w pierwszym wypadku powstał on z rozwinięcia C według produkcji $C \longrightarrow bb$, a zatem jest pierwszym z dwóch symboli rdzenia. Natomiast w drugim wypadku b powstał w wyniku rozwinięcia B według produkcji $B \longrightarrow b$, a zatem jest ostatnim (i jedynym) symbolem rdzenia, który należy oczywiście zredukować zgodnie z $B \longrightarrow b$.

Konstruowanie tablic będzie polegało na podaniu wszystkich możliwych produkcji, które mogą znaleźć zastosowanie przy tworzeniu symbolu stojącego na danej pozycji i zależnie od tego związanie z tym symbolem jednej z następujących informacji: "koniec rdzenia, który należy zredukować według określonej produkcji" lub "jeszcze nie koniec rdzenia - przystąpić do zbadania następnego symbolu".

Wróćmy znów do naszej gramatyki G3 i zastanówmy się jaki może być pierwszy symbol prawostronnej formy zdaniowej w tej gramatyce i jakie należy związać z nim informacje.

Otóż pierwszy symbol w prawostronnej formie zdaniowej gramatyki G3 może powstać bądź w wyniku rozwinięcia symbolu startowego A według produkcji $A \longrightarrow BC$ i jest nim wówczas symbol B, który oczywiście nie jest ostatnim symbolem rdzenia, bądź w wyniku dalszego rozwinięcia tegoż B.

W tym drugim przypadku mamy dwie możliwości: B zostało rozwinięte według produkcji B→aB i wówczas pierwszym symbolem jest a, które również nie jest końcem rdzenia lub B zostało rozwinięte według produkcji B→b i wówczas pierwszym symbolem prawostronnej formy zdaniowej jest b, które jest jednocześnie ostatnim symbolem rdzenia. Rdzeń ten należy oczywiście zredukować według produkcji B→b.

W wyniku tych rozważań możemy powiedzieć, że na pierwszej pozycji mogą pojawić się symbole: B, a oraz b, przy czym w dwóch pierwszych przypadkach nie jest to jeszcze koniec rdzenia, zaś w ostatnim mamy do czynienia z końcem rdzenia, który należy zredukować według produkcji B→b. Ponadto możemy również powiedzieć, że pojawienie się na pierwszej pozycji badanego napisu jakiegokolwiek innego symbolu niż trzy podane wyżej (np. C) oznacza, że napis ten na pewno nie jest prawostronną formą zdaniową w gramatyce G3.

Zbadajmy teraz jakie istnieją możliwości w przypadku drugiej pozycji. Otóż jak łatwo zauważyć możliwości te zależą od tego, jaki symbol pojawił się na pierwszej pozycji. Ponieważ tylko pojawienie się B lub a implikuje dalsze badanie więc wystarczy rozpatrzeć tylko te dwa przypadki. Przypuśćmy, że pierwszym symbolem prawostronnej formy zdaniowej było B. Ponieważ zgodnie z tym co powiedziano wyżej mogło się ono tam pojawić wyłącznie w wyniku rozwinięcia symbolu startowego według produkcji A→BC, więc drugim symbolem takiej formy zdaniowej może być bądź C - i C jest wtedy oczywiście ostatnim symbolem rdzenia, który należy zredukować według produkcji A→BC, bądź pierwszy symbol rozwinięcia C. Ponieważ C można rozwinąć jedynie według produkcji C→bb, więc symbolem tym może być b stanowiące pierwszy symbol prawej strony tej produkcji. To b oczywiście nie jest jeszcze końcem rdzenia. Tak więc, jeżeli na pierwszej pozycji analizowanego napisu pojawiło się B, to na drugiej pozycji może się pojawić C lub b. W pierwszym wypadku mamy do czynienia z końcem rdzenia, który należy zredukować według produkcji A→BC, w drugim zaś wypadku należy przejść do zbadania następnego, trzeciego symbolu badanego napisu. Tym trzecim symbolem może tutaj być tylko b stanowiące drugi i ostatni symbol rdzenia, oczywiście rdzeń ten trzeba tutaj zredukować według produkcji C→bb.

Jeżeli jednak pierwszym symbolem analizowanego napisu było a, to zgodnie z rozważaniami odnoszącymi się do pierwszego symbolu, mogło się ono na tej pozycji pojawić wyłącznie w wyniku rozwinięcia B według produkcji B→aB. Dlatego też drugim symbolem może teraz być bądź B stanowiące drugi (i ostatni) symbol prawej strony tej produkcji, bądź pierwszy symbol rozwinięcia B. Tym symbolem może być a, jeśli B rozwinięto według produkcji B→aB lub b, jeżeli według produkcji B→b.

A zatem, jeżeli pierwszym symbolem formy zdaniowej gramatyki G3 było a, to drugim symbolem może być B, a oraz b z tym, że w pierwszym i ostatnim wypadku mamy do czynienia z ostatnim symbolem rdzenia, który należy zredukować odpowiednio według produkcji B→aB lub według produkcji B→b. Natomiast w drugim przypadku należy przejść do zbadania następnego symbolu. Zauważmy tutaj, że jeżeli a pojawi się również na drugiej pozycji, to dla trzeciej pozycji mamy znowu identyczne możliwości B, a oraz b, z którymi związane są identyczne informacje dotyczące położenia rdzenia. Podobnie będzie z czwartą i dalszymi pozycjami.

Ogólnie możemy więc powiedzieć, że jeżeli w prawostronnej formie zdaniowej gramatyki G3 na k początkowych pozycjach występuje symbol a to na k+1 pozycji może wystąpić bądź symbol B, bądź a, bądź wreszcie b. Pojawienie się B oznacza, że znajdujemy się na końcu rdzenia, który należy zredukować według produkcji B→aB. Pojawienie się a oznacza, że dla znalezienia końca rdzenia należy przystąpić do zbadania następnego symbolu. Pojawienie się b oznacza, że również znaleziony został koniec rdzenia, który należy zredukować według produkcji B→b.

Postaramy się teraz sformalizować przeprowadzone wyżej rozważania. W tym celu zauważmy, że informacje o tym, jakie na danej pozycji mogą wystąpić symbole tworzyliśmy zawsze na podstawie pewnej liczby produkcji, z których każda wnosiła do tych możliwości jeden symbol. Dla pierwszego symbolu produkcjami tymi były produkcje A→BC, B→aB oraz B→b, zaś wnoszonymi przez nie symbolami było odpowiednio B, a oraz b. Natomiast informacje o tym, czy dany symbol występujący na danej pozycji jest ostatnim symbolem rdzenia tworzyliśmy na podstawie tego, czy w produkcji, która go wnosiła, znajdował się on na końcu jej prawej

strony czy też nie. W dalszym ciągu wnoszony symbol będziemy w produkcjach zaznaczali pisząc pod nim kropkę. Wszystkie produkcje wyznaczające zbiór możliwości dla jednej pozycji będziemy grupowali w jednej tablicy a dokładnie w pierwszej kolumnie tej tablicy. Zbiór wszystkich takich tablic utworzonych dla danej gramatyki G nazywać będziemy zbiorem tablic LR(0) dla tej gramatyki.

Poszczególne tablice tego zbioru będziemy oznaczali przez T_o, T_1, T_2... itd. Poza produkcjami w tablicach tych będziemy również umieszczali: w drugiej kolumnie poszczególne symbole, a w trzeciej, na wysokości odpowiedniego symbolu produkcję, według której należy zredukować rdzeń i tablice dalszych możliwości.

Zgodnie z tym wyniki naszych uprzednich, odnoszących się do gramatyki G3, rozważań możemy zapisać w czterech tablicach tak jak na rys. 7.

- Tablica T_o określa możliwości dla pierwszej pozycji prawostronnej formy zdaniowej w gramatyce G3
- Tablica T_1 określa możliwości dla drugiej pozycji pod warunkiem, że na pierwszej znajduje się symbol B
- Tablica T_2 określa możliwości dla trzeciej pozycji pod warunkiem, że na pierwszej pozycji znajduje się symbol B, a na drugiej symbol b
- Tablica T_3 określa możliwości dla drugiej i następnych pozycji pod warunkiem, że na wszystkich poprzednich pozycjach znajduje się symbol a

T_o

$A \rightarrow BC\cdot$	B	T_1
$B \rightarrow aB\cdot$	a	T_3
$B \rightarrow b\cdot$	b	$B \rightarrow b$

T_1

$A \rightarrow BC\cdot$	C	$A \rightarrow BC$
$C \rightarrow bb\cdot$	b	T_2

T_2

$C \rightarrow bb\cdot$	b	$C \rightarrow bb$

T_3

$B \rightarrow aB\cdot$	B	$B \rightarrow aB$
$B \rightarrow aB\cdot$	a	T_3
$B \rightarrow b\cdot$	b	$B \rightarrow b$

Rys. 7. Zbiór tablic LR(0) dla gramatyki G3

Podamy teraz ogólny algorytm konstruowania tablic LR(0). Załóżmy, że mamy je skonstruować dla gramatyki $G = \langle V_T, V_N, P, S \rangle$. O produkcjach tej gramatyki założymy dodatkowo, że symbol startowy występuje w nich wyłącznie po lewej stronie. Gdyby warunek ten nie zachodził, wówczas dla jego spełnienia wystarczy do G wprowadzić nowy symbol nieterminalny S' oraz produkcję $S' \longrightarrow S$ przyjmując jednocześnie, że symbolem startowym G jest S'. To założenie pozwala nam w prosty sposób określić moment zakończenia pierwszej fazy analizy syntaktycznej, polegającej na znalezieniu ciągu produkcji użytych w prawostronnym wywodzie analizowanego zdania. Przy tym bowiem założeniu pierwsza faza zostaje zakończona w chwili zredukowania rdzenia według produkcji, w której po lewej stronie występuje symbol startowy.

Konstruowanie zbioru tablic LR(0) rozpoczynamy od utworzenia tablicy T_0, do której na początku wstawiamy wszystkie produkcje gramatyki G, w których po lewej stronie występuje symbol startowy. W każdej z tych produkcji zaznaczamy kropką pierwszy symbol jej prawej strony.

Następnie tak skonstruowaną tablicę uzupełniamy jeszcze dalszymi produkcjami.

Przeglądamy mianowicie kolejno wszystkie umieszczone dotychczas w konstruowanej tablicy produkcje (również i te, które zostały tam umieszczone w wyniku wykonania opisywanej czynności) sprawdzając, czy w którejś z nich kropką został oznaczony symbol nieterminalny. Dla każdego takiego symbolu umieszczamy w tablicy wszystkie produkcje, których lewa strona jest równa temu symbolowi nieterminalnemu. W umieszczonych tym sposobem produkcjach kropką oznaczamy pierwszy symbol ich prawych stron. Jeżeli jednak w trakcie opisanego wyżej uzupełniania tablicy okaże się, że znajduje się w niej już identyczna produkcja z kropką pod pierwszym symbolem, to produkcji takiej po raz drugi już nie wstawiamy. Uzupełnianie konstruowanej tablicy zostaje zakończone w chwili, gdy nie da się do niej wstawić już żadnej nowej produkcji.

Dalsze tablice tworzymy rozpatrując kolejno wszystkie dotychczas utworzone, a w każdej z nich wszystkie symbole wskazywane kropką w co najmniej jednej z produkcji zawartych w rozpatrywanej tablicy. Wykonujemy przy tym czynności opisane poniższym algorytmem.

1) Niech T oraz X oznaczają odpowiednio rozważaną tablicę oraz rozważany symbol (o X zakładamy, że jest zaznaczony kropką przynajmniej w jednej z produkcji zapisanych w T). Wstaw X do drugiej kolumny tablicy T.

2) Znajdź w pierwszej kolumnie T wszystkie takie produkcje, w których kropką oznaczony jest symbol X i w których to oznaczone kropką X jest ostatnim symbolem prawej strony. Jeżeli takich produkcji nie ma to przejdź do p. 4.

3) Wstaw do trzeciej kolumny T, na prawo od X, wszystkie produkcje znalezione w p. 2 (usuwając kropkę pod X).

4) Znajdź w pierwszej kolumnie T wszystkie takie produkcje, w których kropką oznaczony jest symbol X i w których to oznaczone kropką X nie jest ostatnim symbolem prawej strony. Jeżeli takich produkcji nie ma, to przejdź do p. 10.

5) Utwórz nową tablicę T' i do pierwszej kolumny T wstaw wszystkie produkcje znalezione w p. 4.

6) We wszystkich produkcjach wstawionych do T' przesuń kropkę o jeden symbol w prawo.

7) Uzupełnij T' dalszymi produkcjami w identyczny sposób, jak to miało miejsce w przypadku tablicy T_o.

8) Sprawdź, czy w skonstruowanym dotychczas zbiorze tablic znajduje się tablica T" taka, że zawartość pierwszej kolumny T" jest identyczna z zawartością pierwszej kolumny T' (w sensie produkcji i kropek). Jeżeli tak, to w trzeciej kolumnie T, na prawo od X zapisz T". T" jest tablicą dalszych możliwości dla symbolu X. Przejdź do p. 10.

9) Dołącz T' do konstruowanego zbioru tablic. Jednocześnie zapisz T' w trzeciej kolumnie tablicy T, na prawo od X. T' jest tablicą dalszych możliwości dla X.

10) Sprawdź, czy rozpatrzono już wszystkie symbole oznaczone kropkami w produkcjach zapisanych w T. Jeżeli nie, to weź pod uwagę następny taki symbol i przejdź do p. 1.

11) Sprawdź, czy rozpatrzono już wszystkie tablice zawarte w konstruowanym zbiorze tablic. Jeżeli nie, to weź pod uwagę następną tablicę oraz symbol oznaczony kropką w co najmniej jednej spośród zapisanych w niej produkcji. Przejdź do p. 1.

12) Zakończ konstruowanie zbioru tablic.

Na rys. 8 przedstawiono zbiór tablic LR(0) utworzony na podstawie podanego wyżej algorytmu dla gramatyki G2'. Ponieważ w gramatyce tej symbol startowy A występuje również po prawej stronie produkcji, konieczne było wprowadzenie do niej nowego symbolu startowego S' oraz produkcji S'$\longrightarrow$A. Produkcja ta z symbolem A oznaczonym kropką jest jedyną produkcją, którą na początku wstawiamy do T_0. Ponieważ w produkcji tej kropką jest teraz oznaczony symbol nieterminalny więc wykonując czynności uzupełniania, wstawiamy do T_0 najpierw produkcje $A \longrightarrow \dot{A}+B$ oraz $A \longrightarrow \dot{B}$, następnie $B \longrightarrow \dot{B}*C$ oraz $B \longrightarrow \dot{C}$ i wreszcie $C \longrightarrow \dot{(}A)$ oraz $C \longrightarrow \dot{1}$. W tym momencie dalszych produkcji nie da się już do T_0 dołączyć i w związku z tym przystępujemy do konstruowania dalszych tablic oraz do wypełniania drugiej i trzeciej kolumny T_0. W tym celu rozpatrujemy T_0 a w niej kolejno symbole A, B, C, 1 oraz (.

Weźmy na przykład pod uwagę symbol A. Jest on zaznaczony kropką w dwóch produkcjach: $S' \longrightarrow \dot{A}$ oraz $A \longrightarrow \dot{A} + B$. Zgodnie z podanym wyżej algorytmem wstawiamy A do drugiej kolumny T_0, a następnie na prawo od A, w trzeciej kolumnie zapisujemy produkcję S$\longrightarrow$A (por. rys. 8). Ponadto ponieważ A jest również oznaczone kropką w $A \longrightarrow \dot{A} + B$ więc dla A tworzymy także nową tablicę, którą tutaj oznaczymy przez T_1. Do pierwszej kolumny T_1 wstawiamy produkcję $A \longrightarrow \dot{A} + B$ przesuwając w niej jednocześnie kropkę o jeden symbol w prawo. W efekcie pierwsza kolumna T_1 zawiera produkcję $A \longrightarrow \dot{A} + B$. Jak widać jest to jedyna produkcja, którą udaje nam się wstawić do T_1. Proces uzupełniania nie wnosi już bowiem żadnych nowych produkcji (w produkcji $A \longrightarrow \dot{A} + B$ kropką oznaczony jest symbol terminalny). Musimy teraz jeszcze sprawdzić, czy w skonstruowanym dotychczas zbiorze tablic nie występuje już tablica, której pierwsza kolumna jest identyczna z pierwszą kolumną T_1. W tym momencie zbiór tablic składa się jedynie z tablicy T_0, a pierwsza kolumna T_0 różni się od pierwszej kolumny T_1. Dołączamy zatem T_1 do konstruowanego zbioru tablic zapisując jednocześnie T_1 w trzeciej kolumnie T_0, na prawo od A (por. rys. 8).

T_0

$S \longrightarrow A\cdot$	A	$S' \longrightarrow A,\ T_1$
$A \longrightarrow A\cdot+B$		
$A \longrightarrow B\cdot$	B	$A \longrightarrow B,\ T_2$
$B \longrightarrow B\cdot*C$		
$B \longrightarrow C\cdot$	C	$B \longrightarrow C$
$C \longrightarrow 1\cdot$	1	$C \longrightarrow 1,\ T_3$
$C \longrightarrow (\cdot A)$		

T_1

$A \longrightarrow A\cdot+B$	+	T_4

T_2

$B \longrightarrow B\cdot*C$	*	T_5

T_3

$C \longrightarrow (\cdot A)$	A	T_6
$A \longrightarrow A\cdot+B$	B	$A \longrightarrow B,\ T_2$
$A \longrightarrow B\cdot$		
$B \longrightarrow B\cdot*C$	C	$B \longrightarrow C$
$B \longrightarrow C\cdot$	1	$C \longrightarrow 1$
$C \longrightarrow 1\cdot$	(	T_3
$C \longrightarrow (\cdot A)$		

T_4

$A \longrightarrow A+B\cdot$	B	$A \longrightarrow A+B,\ T_2$
$B \longrightarrow B\cdot*C$		$A \longrightarrow A+B$
$B \longrightarrow C\cdot$	C	$B \longrightarrow C$
$C \longrightarrow 1\cdot$	1	$C \longrightarrow 1$
$C \longrightarrow (\cdot A)$	(	T_3

T_5

$B \longrightarrow B*C\cdot$	C	$B \longrightarrow B*C$
$C \longrightarrow 1\cdot$	1	$C \longrightarrow 1$
$C \longrightarrow (\cdot A)$	(	T_3

T_6

$C \longrightarrow (A)\cdot$	)	$C \longrightarrow (A)$
A $\longrightarrow A\cdot+B$	+	T_4

Rys. 8. Tablice LR(0) dla gramatyki G2'

Zajmiemy się obecnie opisem algorytmu LR(0) wykorzystującego tablice LR(0) skonstruowane opisanym wyżej sposobem. W świetle przeprowadzonych wyżej rozważań poprawność tego algorytmu nie powinna budzić zastrzeżeń. Algorytm ten opiszemy w punktach podających kolejne czynności niezbędne do wykonania procesu analizy.

1) Weź pod uwagę tablicę $T = T_o$ oraz pierwszy symbol analizowanego napisu.

2) Jeżeli symbol ten - oznaczmy go przez X - nie występuje w tablicy T wówczas analizowany napis nie jest prawostronną formą zdaniową gramatyki G, dla której skonstruowany został dany zbiór tablic.

3) Sprawdź, czy w trzeciej kolumnie T dla symbolu X występuje nazwa tablicy, czy też produkcja. W pierwszym wypadku przejdź do p. 4 - w drugim do p. 5.

4) Niech T' oznacza tablicę podaną dla symbolu X w T. Podstaw $T = T'$ oraz weź pod uwagę następny symbol analizowanego napisu i przejdź do wykonania czynności opisanych w p. 2.

5) Niech p oznacza produkcję podaną dla symbolu X w tablicy T. Zapamiętaj p jako kolejną produkcję poszukiwanego ciągu oraz usuń z analizowanego napisu l ostatnio badanych symboli (gdzie l oznacza liczbę symboli prawej strony produkcji p) a na ich miejsce wstaw symbol stanowiący lewą stronę produkcji p.

6) Jeżeli lewą stroną produkcji p jest symbol startowy to przystąp do konstruowania drzewa wywodu na podstawie zapamiętanego ciągu produkcji.

7) Jeżeli jednak lewą stroną produkcji p nie jest symbol startowy wówczas weź pod uwagę tablicę $T = T_o$ oraz pierwszy symbol otrzymanego w punkcie 5 napisu i przejdź do wykonywania czynności opisanych w p. 2.

Należy tutaj zwrócić uwagę na fakt, że wykonanie punktu 6 nie oznacza wcale, że cały podany do analizy napis jest zdaniem języka L(G) a jedynie, że jest nim ta część, która w wyniku kolejnych redukcji została przekształcona w symbol startowy. Aby nie dopuścić do sytuacji, w której następuje zakończenie pierwszego etapu analizy, mimo że faktycznie została zbadana tylko część poddanego analizie napisu można np. w pkt. 6 dodatkowo sprawdzić czy na prawo od symbolu startowego nie ma jeszcze dalszych symboli. Ich obecność świadczyłaby o tym, że mimo wszystko dany napis nie jest zdaniem języka L(G).

Studiując podany wyżej algorytm nie trudno zauważyć, że nie może on być stosowany np. w przypadku gramatyki G2'. Prawidłowe działanie tego algorytmu jest bowiem tylko wtedy możliwe, gdy we wszystkich skonstruowanych dla danej gramatyki tablicach, dla każdego symbolu podana jest bądź tylko nazwa tablicy, bądź tylko jedna produkcja. Tylko wtedy bowiem można dla każdego symbolu jednoznacznie określić czy należy wykonać pkt 4 algorytmu, czy też pkt. 5.

Jak to widać na rys. 8 warunek powyższy nie jest spełniony w przypadku G2'. W T_0 mamy bowiem dla A podaną zarówno nazwę tablicy jak i produkcję. Podobną sytuację mamy również w tablicach T_0, T_3 i T_4 dla symbolu B.

Na podstawie tych obserwacji możemy powiedzieć, że gramatyka G2' nie jest gramatyką LR(0)

Algorytm LR(0) daje się jednak uogólnić tak, że można go zastosować również w przypadku szerszej klasy gramatyk. Nie wdając się w szczegóły można powiedzieć, że w uogólnionym algorytmie, w celu znalezienia rdzenia prawostronnej formy zdaniowej oraz produkcji, według której rdzeń ten należy zredukować, bada się jeszcze dodatkowo jeden, dwa lub ogólnie k następnych symboli. Zależnie od liczby badanych dodatkowych symboli odpowiedni algorytm nosi nazwę algorytmu LR(1), względnie LR(2) ... względnie LR(k) a gramatyka, dla której taki algorytm funkcjonuje odpowiednio - gramatyki LR(1), LR(2)... LR(k).

Omawiana tutaj gramatyka G2' jest gramatyką LR(1). W celu prawidłowego przeprowadzenia analizy składni wystarczy tutaj badać dodatkowo jeszcze jeden symbol. Jak bowiem łatwo zauważyć w przypadku tablicy T_0, jeżeli po symbolu A następuje symbol + to należy wykonać pkt 4 przyjmując $T = T_1$, natomiast jeżeli A jest ostatnim symbolem badanego napisu, to należy wykonać pkt 5 redukując A według produkcji S ⟶ A. Podobnie w tablicach T_0, T_3 i T_4, gdy po B następuje * to oznacza to, że należy wykonać pkt 4, natomiast gdy B jest ostatnim symbolem badanego napisu lub gdy po B następuje + lub), to należy wykonać pkt 5.

Oczywiście dla $k \geqslant 1$ algorytm analizy składniowej, a zwłaszcza algorytm konstruowania tablic, są o wiele bardziej skomplikowane niż to mia-

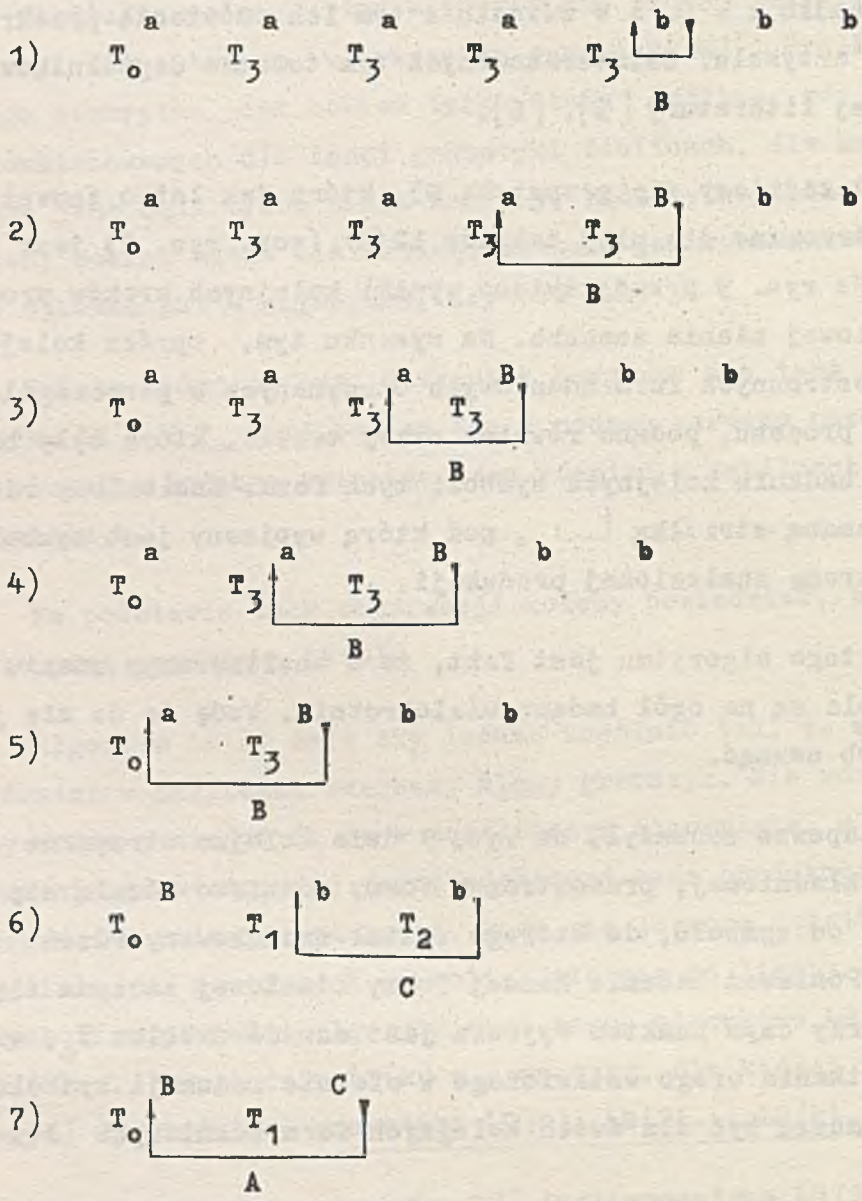
ło miejsce w przypadku k = 0, i w związku z tym ich omówienie przekraczałoby ramy tego artykułu. Zainteresowanych tym tematem Czytelników odsyłamy do bogatej literatury [5], [6].

Tutaj natomiast zajmiemy się gramatyką G3, która jak łatwo zauważyć sprawdzając skonstruowane dla niej tablice LR(0) (zob. rys. 7) jest gramatyką LR(0). Na rys. 9 przedstawiono wyniki kolejnych kroków procesu analizy składniowej zdania aaaabbb. Na rysunku tym, oprócz kolejnych symboli prawostronnych form zdaniowych otrzymanych w poszczególnych krokach tego procesu, podano również nazwy tablic, które były brane pod uwagę przy badaniu kolejnych symboli tych form. Znaleziony rdzeń jest zaznaczony łamaną strzałką ⌊⌋ , pod którą wypisany jest symbol stanowiący lewą stronę znalezionej produkcji.

Oczywistą wadą tego algorytmu jest fakt, że w analizowanym zdaniu poszczególne symbole są na ogół badane wielokrotnie. Wadę tę da się jednak w prosty sposób usunąć.

Jak czytelnik zapewne zauważył, na rys. 9 dwie kolejne otrzymane w procesie analizy składniowej, prawostronne formy zdaniowe różnią się dopiero poczynając od symbolu, do którego został zredukowany rdzeń pierwszej z nich. Ponieważ badanie każdej formy zdaniowej zaczyna się od jej początku, przy czym punktem wyjścia jest zawsze tablica T_0, więc aż do momentu napotkania owego wstawionego w efekcie redukcji symbolu - wyniki tych badań muszą być dla dwóch kolejnych form zdaniowych identyczne.

Nasuwa to myśl, aby za każdym razem nie powtarzać już raz wykonanej pracy i badanie każdej następnej prawostronnej formy zdaniowej rozpoczynać od symbolu, do którego został zredukowany rdzeń poprzedniej. Badanie to jednak musi być przeprowadzone za pomocą tablicy określonej w wyniku badania ostatniego z symboli poprzedzającego zredukowany rdzeń. Tak więc trzeba aby po zbadaniu tego ostatniego symbolu dodatkowo zapamiętać otrzymaną nazwę tablicy. Niestety, w opisywanej metodzie analizy składniowej nigdy nie jesteśmy w stanie stwierdzić, czy badany symbol poprzedza rdzeń czy też nie. Nie stanowi to jednak większej przeszkody, możemy bowiem zapamiętywać w formie ciągu kolejno nazwy wszystkich tablic określanych w p. 4 algorytmu, a następnie określać pożądaną tabli-



Rys. 9. Wyniki kolejnych kroków procesu analizy składniowej zdania aaaabbb

cę w momencie znalezienia końca rdzenia, kiedy to znana staje się produkcja p, według której należy zredukować rdzeń. Jeżeli bowiem z zapamiętanego ciągu nazw tablic usuniemy wówczas tyle ostatnich jego elementów, ile wynosi długość prawej strony produkcji p pomniejszona o 1, to ostatnia z pozostałych nazw jest nazwą tej tablicy, którą należy użyć do zbadania symbolu otrzymanego w efekcie redukcji rdzenia.

Zmodyfikowany zgodnie z powyższymi sugestiami algorytm LR(0) ma następującą postać:

1) Przyjmij $T = T_0$, zapamiętaj T_0 jako pierwszy element ciągu nazw tablic oraz weź pod uwagę pierwszy symbol analizowanego napisu.

2) Jeżeli w tablicy T dla rozważanego symbolu - oznaczymy go przez X - nie występuje ani nazwa tablicy ani produkcja, wówczas analizowany napis nie jest zdaniem języka L(G)(G oznacza gramatykę, dla której skonstruowany został zbiór wykorzystywanych w algorytmie tablic LR(0)).

3) Sprawdź, czy w trzeciej kolumnie tablicy T dla symbolu X występuje nazwa tablicy, czy też produkcja. W pierwszym wypadku przejdź do p.4, w drugim do p. 5.

4) Niech T' oznacza nazwę tablicy podaną w T dla symbolu X. Zapamiętaj T' jako kolejny element ciągu nazw tablic, podstaw $T = T'$ oraz weź pod uwagę kolejny symbol analizowanego napisu i przejdź do wykonania czynności opisanych poczynając od p. 2.

5) Niech p oznacza produkcję podaną w tablicy T dla symbolu X. Zapamiętaj p jako kolejny element poszukiwanego ciągu produkcji.

6) Jeżeli lewa strona produkcji p jest symbolem startowym, to sprawdź, czy zostały zbadane wszystkie symbole analizowanego napisu. Jeżeli tak, to przystąp do konstruowania drzewa wywodu. W przeciwnym wypadku analizowany napis nie jest zdaniem języka L(G).

7) Jeżeli lewa strona produkcji p nie jest symbolem startowym, to usuń z zapamiętanego ciągu nazw tablic l - 1 jego elementów - gdzie l oznacza liczbę symboli po prawej stronie produkcji p. Niech dla

ustalenia uwagi ostatnia z pozostałych nazw tablic jest T": podstaw T = T" oraz weź pod uwagę symbol stanowiący lewą stronę produkcji p i przejdź do wykonania czynności opisanych poczynając od p. 2.

Posługując się tym algorytmem - jak łatwo zauważyć - badamy każdy symbol analizowanego napisu, zgodnie z tym co sugerowano w rozdziale 2, tylko raz.

Nie będziemy tutaj podawali przykładu działania zmodyfikowanego algorytmu LR(0) sugerując, aby czytelnik wykonał tę pracę samodzielnie, np. dla napisu aaaabbb oraz zbioru tablic LR(0) dla gramatyki G3 (przedstawionych na rys. 7).

Literatura

[1] BLIKLE A.: Automaty i gramatyki. Warszawa 1971 PWN

[2] HOPCROFT J.E., ULLMAN J.D.: Formal Languages and Their Relation to Automata. Mass 1969, Addison - Wesley Reading

[3] MAŁUSZYŃSKI J.: Algorytm lewostronnej redukcji słów. Algorytmy 1969, nr 10, s. 59-75

[4] KRAWCZYK T.: Metody analizy składniowej. ETO Nowości 1974, nr 1, 2

[5] KNUTH D.: On the Translation of Languages from Left to Right. Information and Control 1965, t. 8, nr 6, s. 607-639

[6] AHO A.V., ULLMAN J.D.: The Theory of Parsing, Translation and Compiling. T. 1 i 2. New York 1973. Prentice-Hall, Inc. Englewood Cliffs

Spis treści rocznika XIII/1974
kwartalnika "Elektroniczna Technika Obliczeniowa Nowości"

„Intrograf" — Lublin, zam. 570 B-23 nakł. 1500×4