БЪЛГАРСКА АКАДЕМИЯ НА НАУКИТЕ * ACADÉMIE BULGARE DES SCIENCES
ИЗВЕСТИЯ НА МАТЕМАТИЧЕСКИЯ ИНСТИТУТ
BULLETIN DE L'INSTITUT DE MATHÉMATIQUES
Том (Tome) XIII

СИСТЕМА ЗА СИМВОЛИЧНО ПРОГРАМИРАНЕ НА МАШИНАТА „МИНСК-2"

Петър Бърнев и Димитър Тошков

В тази статия се разглежда една система за символично програмиране на машината „Минск-2" [3] с оперативна памет 8192 клетки. Тя може да се използва непосредствено и на машини „Минск-22". Системата се състои от език за символично програмиране и транслатор. Тя е създадена в 1967 г. и внедрена през 1968 г. в Математическия институт с Изчислителен център при БАН. Системата за символично програмиране е съобразена с диспечерската система МИД-2 [1] за машината „Минск-2", но съществува вариант, който може да работи и самостоятелно.

Системата за символично програмиране е предназначена да замени изцяло използването на машинния език, като запазва качеството на програмите при транслация. Тя дава следните преимущества в сравнение с непосредственото програмиране на машинен език:

1. Разпределя автоматично оперативната памет.
2. Допуска по-удобни означения на променливите и дава по-голяма прегледност на програмите.
3. Допуска използването на константи непосредствено в командите.
4. Позволява да се избегне преадресацията при внасяне на корекции в програмите.
5. Опростява използването на стандартни подпрограми (СП) — те се използват посредством системата МИКС-4 [2].
6. Опростява използването на периферните устройства на машината.
7. Открива допуснатите формални грешки в програмата.
8. Създава удобства при проверка на програмите.

За машината „Минск-2" са изработени различни символични езици и автокодове [4], [5], [6], [7]. Някои от тях не са достатъчно близки до машинния език, което води до загуба на ефективност при транслация. Освен това изразните средства на някои от тези езици са сравнително тежки, което усложнява тяхното използване, а всички автокодове използват служебни думи на чужди езици, което затруднява практическото им използване у нас.

В първата част на тази статия се описва езикът за символично програмиране, а във втората част се дават някои сведения за транслатора.

1. ЕЗИК ЗА СИМВОЛИЧНО ПРОГРАМИРАНЕ

1.1. Обща структура на програмата

Програмата, написана на езика за символично програмиране, се състои от една или няколко глави. Разделянето на една програма на глави се налага, когато тя не може изцяло да се помести в оперативната памет. Изпълнението на програмата започва винаги от началната глава. След това могат да се използват многократно всички участвуващи в програмата глави в произволен ред. Управлението може да се предава от една глава на друга от произволно място с помощта на специален оператор. Изпълнението на всяка глава започва с нейния пръв оператор. При повторно изпълнение на дадена глава тя се намира в това състояние, в което е останала при последното изпълнение.

Променливите и масивите (вж. 1.2), които участвуват в програмата, могат да бъдат локални или глобални. Локалните променливи и масиви са свързани с отделна глава и техните стойности не могат да бъдат използвани в други глави, но означенията им, използвани в една глава, могат да бъдат употребени и в други глави.

Глобалните променливи и масиви се отнасят за всички глави и употребените за тях означения не могат да се използват за други цели.

Предаването на управлението вътре в една глава става посредством вътрешни етикети, които са локални в описания по-горе смисъл. Външните и служебните етикети (вж. 1.2) са глобални.

Всяка част се състои от описание и операторна част. В описанието се дава информация за локалните масиви, участвуващи в главата. Операторната част се състои от оператори, написани на последователни редове или разделени помежду си със символа =. Началната глава съдържа пред своето описание и глобално описание, в което се дава информация за броя на главите, за глобалните променливи и масиви, за служебните етикети и за максималното време, необходимо за работа на програмата.

Началните данни за работа на програмата се записват посредством входната система на МИД-2 в специално определени участъци на външната памет, наречени външни масиви. Обменът между оперативната и външната памет, т. е. между вътрешните и външните масиви, се осъществява с помощта на специален оператор.

1.2. Величини

В езика се използват три типа величини: константи, променливи и етикети. Смисълът на първите два типа е очевиден, а етикетите се използват за отбелязване на определени оператори. Допуска се използването на следните видове константи: десетични (цели, дробни и с порядък), осмични и специални константи, свързани с използването на някои оператори (порядъци за изместване, параметри на СП и др.).

Променливите величини се разделят на прости и променливи с индекси. Допуска се само един индекс. Представянето на променливите величини в машината може да бъде с фиксирана или с плаваща запетая. Освен това се допуска използването на работни и индексни променливи. Към променливите се отнасят и външните масиви, но тяхното използване е възможно само посредством оператора за обмен.

В програмата може да се използват три вида етикети: вътрешни, външни и служебни. Външните етикети означават номерата на главите. С вътрешните етикети се отбелязват такива оператори в главата, които се използват от други оператори в същата глава. Служебните етикети служат за включване на определени оператори от програмата и се използват най-често при проверка на програмите. Означените с тях оператори се включват в транслираната програма само при специално указание на програмиста.

Символиката за означаване на променливите и константите не се отличава съществено от общоприетата. Пред числата, които са представени с фиксирана запетая (цели и дробни), се поставя точка. Осмичните константи (съответстващи на машинни кодове) се разделят на три групи по четири цифри, като знакът се включва в първата група. Незначещите нули в групите се пропускат и всяка група се предшествува от запетая. В табл. 1 се дават няколко примера за записване на константи в езика за символично програмиране.

Ще забележим, че ако е необходимо в даден оператор да участвува етикет на следващия го оператор, то може да се използва символът /, без да са въвежда такъв етикет. Със същия символ може да се замести кодът ,0,0,0 или нулево съдържание на адресна част.

Таблица 1

Константи	Записване в символичен език	Тип на константите
3,14159	3,14159	число с плаваща запетая
—0,0125	.—0,0125	число с фиксирана запетая
$-213 . 10^{-11}$	—213Ю—11	число с порядък
—120	.—120	цяло число
7777 0000 0000	,7777,0,0	осмична константа
—0001 0000 0002	,— 1,0,2	осмична константа

Простите променливи, представени с фиксирана запетая, се означават с буквата F, след която следват не повече от три цифри. Простите променливи, представени с плаваща запетая, се означават с произволна буква (отлична от F, M, R и I), след която могат да следват не повече от три цифри. Масивите се означават с буквата M, след която следва произволна буква и евентуално една цифра. Когато буквата е F, елементите на масива се представят с фиксирана запетая. Означенията на масивите се използват само в описанията. Означенията на променливите с индекс се образуват от означенията на съответните масиви, последвани в скоби от естествено число, представляващо номера на елемента от съответния масив. Когато това число е единица, то може да се изпусне заедно със скобите. Изменението на индекса на дадена променлива с индекс може да се извършва само чрез съответна индексна променлива. Външните масиви се означават чрез номерата си — естествени числа.

Примери за означения на променливи и масиви се дават в табл. 2.

Вътрешните етикети се означават с буквата Э, последвана от число в интервала 1—149, външните етикети — с буквата Ш, последвана от една

Таблица

Означение	Тип
X10	проста променлива с плаваща запетая
A2	проста променлива с плаваща запетая
Y	проста променлива с плаваща запетая
F123	проста променлива с фиксирана запетая
MB2	масив с плаваща запетая или променлива с индекс MB2 (1)
R14	работна променлива
I3	индексна променлива
MF2	масив с фиксирана запетая или променлива с индекс
MX(39)	променлива с индекс
14	външен масив

десетична цифра, а служебните — със символа ?, последван от число в интервала 0—35. Примери:

ЭО, Э120 — вътрешни етикети;

Ш4, Ш7 — външни етикети;

?6, ?29 — служебни етикети.

1.3. Описания

Описанията служат за задаване на известна информация, необходима за работата на транслатора във връзка с разпределението на оперативната памет и с някои други негови функции.

Всяка глава, която съдържа локални масиви, започва с описанието на тези масиви. Това описание съдържа наименованието на масивите и съответните им дължини, зададени в скоби като цели десетични числа. Така например MA(7), срещнато в описанието, означава, че масивът MA има 7 елемента, а не променливата с индекс MA (7).

Началната глава съдържа и глобално описание. Например глобалното описание

CH2 X113 MX7(100) 3—6 T=7 A2

означава, че програмата се състои от две глави (CH2), максималното време за нейното изпълнение е 7 минути (T=7), масивът MX7 е глобален (общ за двете глави) и броят на неговите елементи е 100, променливите X113 и A2 са глобални и операторите със служебни етикети 3, 4, 5 и 6 (3—6) ще бъдат включени в програмата.

1.4. Оператори

В езика за символично програмиране се допускат три типа оператори. Всеки оператор може да бъде предшествуван от служебен и вътрешен етикет и да завършва с коментар.

Към първия тип се отнасят операторите, които съответствуват директно на машинните команди и имат аналогична структура. Тези оператори се състоят от код на операцията (идентичен с машинния код на съответната операция), индексна част и два аргумента. Аргументите на операторите могат да бъдат величини сред участвуващите в езика. При

необходимост от индексация в индексната част се поставя съответна индексна променлива. Присвояването на стойност на индексните променливи се извършва от програмиста посредством оператори на езика. Поради характера на езика за символично програмиране специални средства, облекчаващи индексацията, не са включени. Подобни средства има смисъл да се включват в алгоритмични езици от по-високо ниво.

За да се избегне допускането на някои често срещащи се грешки при съставянето на програми, в езика са включени някои естествени изисквания за съответствие между вида на операциите, индексната част и аргументите. Така например индексна променлива в индексната част не се допуска, ако сред аргументите на оператора не участвува променлива с индекс.

Операторите от втория тип — микрооператори — имат аналогична структура, но се превеждат с няколко машинни команди. Тези оператори служат за означаване на сравнително кратки последователности от машинни команди, реализиращи типични действия, за които в машината не са предвидени специални команди. Например за преобразуването на дадена величина от един тип в друг при използването на машинен език програмистът е принуден да използва винаги една и съща група команди. За удобство тази група команди се обединява в един макрооператор от езика. Ефективността на транслацията при това не се намалява, а съставянето на програмите се облекчава.

Операторите от третия тип служат за използване на стандартни подпрограми (СП), включени в библиотеката. В тях се указва номерът на СП и съответните ѝ параметри. Към този тип оператори спада и операторът, който осъществява обмен на информацията между външните и вътрешните масиви.

ПРОГРАМА: СКАЛАРНО ПРОИЗВЕДЕНИЕ НА ДВА ВЕКТОРА

Шифър на задачата: 012001

Шифър на паметта:

0					
1		* T=1 *			
2		MA(10)	MB(10)		
3		SP0			
4		:+ /	1	MA	INPUTA
5		SP0			
6		:+ /	2	MB	INPUTB
7		−10 /	/	X	
8		−10 /	,11,0,0	I1	
9	Э1	+35 I1	MA	MB	AI.BI
0		+16 /	X	X	AI.BI+X
1		−20 I1	Э1	,0,1,1	
2		SP12			
3		:+ /	X	/	PRINTX
4		−00 /	/	/	STOP
5					
6					
7					

Фиг. 1

1.5. Пример

За илюстрация на езика за символично програмиране на фиг. 1 се привежда програма за пресмятане на скаларното произведение на два 10-мерни вектора A и B. Координатите на векторите A и B се намират съответно във външните масиви 1 и 2 и се присвояват на масивите МА и МВ от оперативната памет. В случая програмата се състои от една глава и глобалното описание съдържа само максималното време за работата на програмата.

2. ТРАНСЛАТОР

2.1. Обща структура на транслатора

Транслаторът превежда програмата, написана на езика за символично програмиране, в програма на машинен език с еднократен преглед на текста.

Последователните етапи на работата на транслатора са следните:

1. Прочитане от външната памет на глобалното описание и началната глава, разпределение на тази част от оперативната памет, която ще бъде обща за всички глави, съгласно глобалното описание, създаване таблица на указаните служебни етикети и задаване на диспечерската система на максималното време за работата на програмата.

2. Превеждане на началната глава.

3. Ако програмата съдържа и други глави — записване на началната глава във външната памет, създаване каталог на главите и последователно прочитане, превеждане и записване във външната памет на следващите глави.

4. Ако по време на транслацията в програмата не се забележат грешки и програмата се състои само от една глава, управлението се предава на МИКС-4, която включва необходимите СП, след което се преминава към изпълнение на транслираната програма. Когато програмата се състои от повече глави и нито една глава не съдържа грешки, специален блок организира прочитането на началната глава и осигурява понататъшното предаване на управлението между главите. При това преди първото изпълнение на всяка глава блокът се обръща към МИКС-4 за включване на необходимите за съответната глава СП.

Ако при транслацията се открият грешки в програмата, посредством перфорация се отпечатва на телетайп съответна информация и транслацията продължава до цялостния превод на програмата с цел да се открият евентуалните грешки в останалата част.

Преводът на всяка глава започва с обработката на локалното описание, чрез която на масивите, участвуващи в главата (ако има такива), се отделя съответно място в оперативната памет. След това се обработват последователно операторите на главата, при което всеки от тях се превежда в машинна команда (или група команди при макрооператорите) с абсолютни адреси. Изключение правят само вътрешните етикети, информация за които се запазва в специална таблица и съответствуващите адреси се запълват от транслатора след окончателния превод на главата.

При превода на адресните части на всеки оператор от първи тип в зависимост от началните им символи и кода на операцията се влиза в специална таблица, чрез която се установява дали съответната величина е допустима. В този случай управлението се предава на блок, който обработва съответния тип величини.

Преводът на макрооператорите се извършва чрез заместване с обработените адресни части в предварително подготвени групи команди за всеки макрооператор.

Всеки оператор за обръщение към СП се замества със съответна псевдокоманда за системата МИКС-4, последвана от обработените параметри.

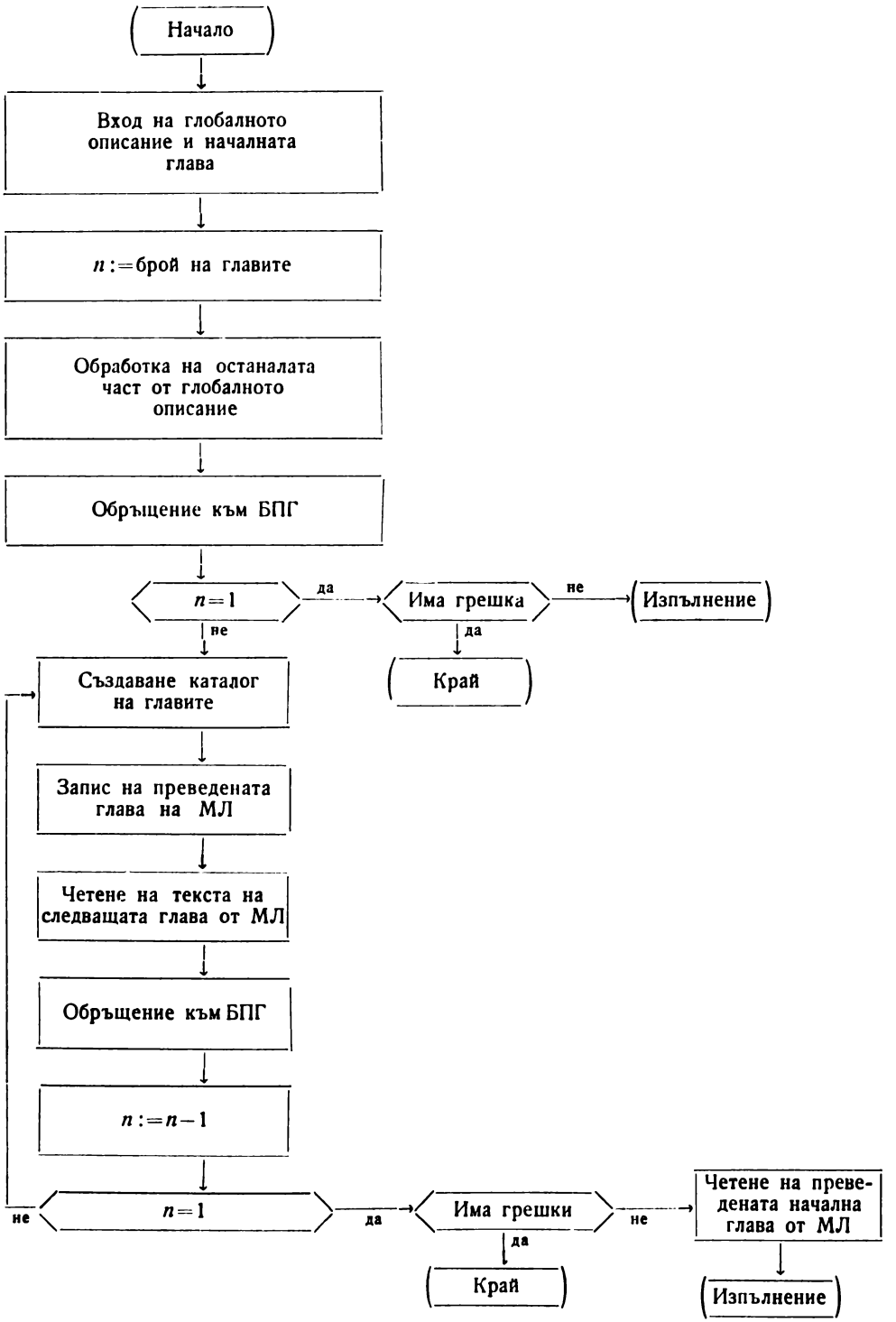
2.2. Блок-схема, разпределение на оперативната памет и параметри на транслатора

На фиг. 2 е дадена общата блок-схема, по която работи транслаторът. Със съкращението БПГ в схемата е означен блокът за превод на една глава. Основните функции на този блок са описани в блок-схемата от фиг. 3. Както се вижда от тази фигура, при превеждането и на трите типа оператори се използува специален блок за обработка на адресните части. Този блок от своя страна включва блокове за обработка на различните типове величини, както и блок, извеждащ информация, при срещане на грешки. Към блока на грешките при необходимост предават управлението и останалите блокове.

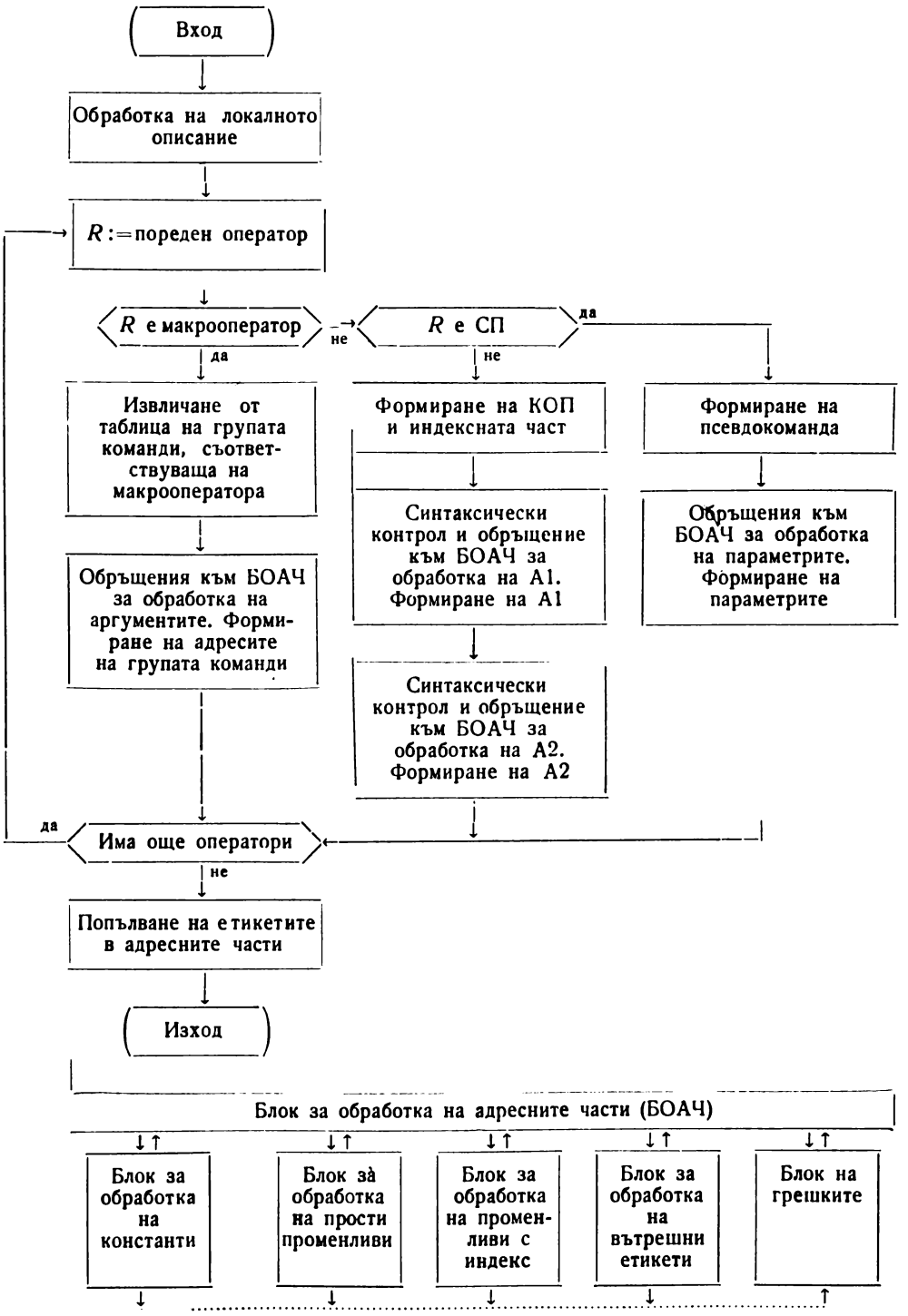
На фиг. 4 и 5 е показано в общи черти разпределението на оперативната памет съответно по време на транслация и по време на изпълнение на преведената програма. По време на превода транслаторът заедно с необходимите таблици, заемащи общо около 2000 клетки, се намират изцяло в оперативната памет.

Разпределението на оперативната памет е извършено така, че прехвърлянето на групи информация от едно място на друго е избягнато което ускорява транслацията. Така например при превода всяка глава се записва на мястото, където ще бъде изпълнена. Взети са мерки за рационалното използване на оперативната памет — например преведената програма се записва на същото място, където се намира текстът на главата на езика за символично програмиране. За намаляване броя на полетата с променливи граници някои от тях (като полетата за числа и прости променливи) са обединени. По този начин ограниченията за дължините на отделните полета се заменят с по-слабо ограничение, което се отнася само за сумарната дължина на обединените полета. Прави се пълна икономия на използваните константи. Даже константи от различни типове, чието двоично представяне съвпада, заемат само една клетка от оперативната памет.

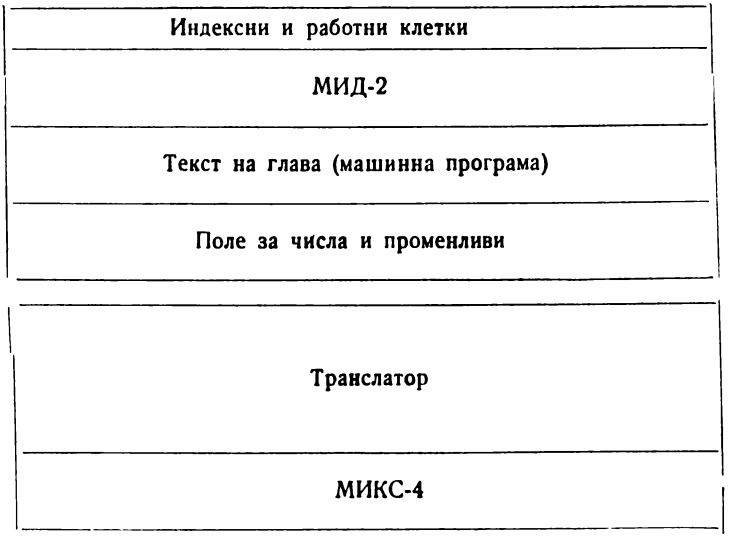
Въпреки това обемът на оперативната памет налага въвеждането на някои сравнително слаби ограничения. По-съществени от тях са: в програмата могат да участвуват до 10 глави, до 20 глобални масива, до 20 локални масива в една глава, до 256 етикета в адресните части на операторите. Общият брой на простите променливи, константите, командите на транслираната програма и командите на използваните СП не трябва да надминава за една глава 3400. Също така общият брой на еле-


Начало
Вход на глобалното описание и началната глава
n:=брой на главите
Обработка на останалата част от глобалното описание
Обръщение към БПГ
n=1
да
Има грешка
не
Изпълнение
не
да
Край
Създаване каталог на главите
Запис на преведената глава на МЛ
Четене на текста на следващата глава от МЛ
Обръщение към БПГ
n:=n-1
не
n=1
да
Има грешки
не
Четене на преведената начална глава от МЛ
да
Край
Изпълнение

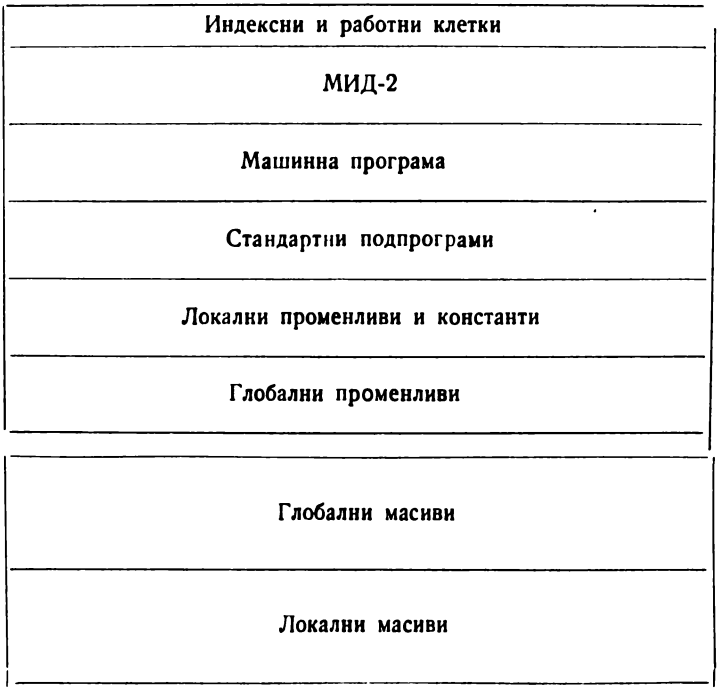
Фиг. 2


Вход
Обработка на локалното описание
R := пореден оператор
R е макрооператор
не
R е СП
да
да
не
Извличане от таблица на групата команди, съответствуваща на макрооператора
Формиране на КОП и индексната част
Формиране на псевдокоманда
Обръщения към БОАЧ за обработка на аргументите. Формиране на адресите на групата команди
Синтаксически контрол и обръщение към БОАЧ за обработка на А1. Формиране на А1
Обръщения към БОАЧ за обработка на параметрите. Формиране на параметрите
Синтаксически контрол и обръщение към БОАЧ за обработка на А2. Формиране на А2
да
Има още оператори
не
Попълване на етикетите в адресните части
Изход
Блок за обработка на адресните части (БОАЧ)
Блок за обработка на константи
Блок за обработка на прости променливи
Блок за обработка на променливи с индекс
Блок за обработка на вътрешни етикети
Блок на грешките

Фиг. 3


Индексни и работни клетки
МИД-2
Текст на глава (машинна програма)
Поле за числа и променливи
Транслатор
МИКС-4

Фиг. 4


Индексни и работни клетки
МИД-2
Машинна програма
Стандартни подпрограми
Локални променливи и константи
Глобални променливи
Глобални масиви
Локални масиви

Фиг. 5

ментите на глобалните масиви и локалните масиви на коя да е глава не трябва да надминава 4095.

Транслаторът превежда около 50—100 оператора в секунда, което съответствува приблизително на изпълнението на 150—300 машинни команди за превода на един оператор.

Системата за символично програмиране се експлоатира в Математическия институт с Изчислителен център на БАН повече от две години. През това време голям брой потребители се запознаха и прилагат успешно езика за символично програмиране за решаване на разнообразни задачи. С изключение на няколко случая, свързани с реализацията на сложни алгоритми от логически характер, при които допустимият брой на етикетите се оказа недостатъчен и се наложи сегментиране на програмите, както и на някои задачи, изискващи намеса от пулта по време на изпълнението им, на авторите не е известно наложените ограничения в системата да са били пречка при решаването на задачи.

Досегашният опит показа, че изучаването на езика за символично програмиране и използването на транслатора не изисква особени усилия. Индикациите, които се дават от транслатора, улесняват значително откриването на грешките, а средствата в езика опростяват отстраняването им.

ЛИТЕРАТУРА

1. Бърнев, П. Диспечерска система МИД-2 (под печат).
2. Бырнев, П. и Д. Тошков. Компилирующая система МИКС для использования библиотеки стандартных программ для машины Минск-2. Algoritmy, 4, (1967), No. 7.
3. Савинков, В. М. Программирования для ЭЦВМ Минск-2. М., 1966.
4. Немейман, М. Е. и др. Автокод для решения инженерных задач на машине Минск-2. Минск, 1965.
5. Hucl, C. Autocode for scientific computations on the Minsk computer programming and operating guide. Kancelarske stroje n. p., Praha, 1966.
6. Formandl a kol. Programovaci Jazik MXA3-MAT2. Kancelarske stroje n. p., Praha, 1966.
7. Бърнев, П. Кратко ръководство по програмиране на езика МИКОД. С. 1968.

Постъпила на 26. III. 1970

СИСТЕМА СИМВОЛИЧЕСКОГО ПРОГРАММИРОВАНИЯ МАШИНЫ МИНСК-2

Петр Бырнев и Димитр Тошков

(Резюме)

Рассматривается система символического программирования машины Минск-2, состоящая из языка символического программирования и соответствующего транслятора. Система предназначается для целостной замены использования машинного языка при сохранении качества транслируемых программ.

A SYSTEM FOR SYMBOLIC PROGRAMMING FOR MINSK-2

Petâr Bârnev, Dimitâr Toškov

(Summary)

A system for symbolic programming for MINSK-2 consisting of a language for symbolic programming and a corresponding translator is considered. The system is intended to replace entirely the machine language usage retaining the quality of the programmes translated.