СООБЩЕНИЯ
ОБЪЕДИНЕННОГО
ИНСТИТУТА
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ
ДУБНА

11 - 7396

О Ен Ир, В.П. Шириков

НОВЫЕ ВОЗМОЖНОСТИ ИСПОЛЬЗОВАНИЯ ЛОГИЧЕСКИХ ВЫРАЖЕНИЙ ПРИ ПРОГРАММИРОВАНИИ НА ЯЗЫКЕ ФОРТРАН ДЛЯ ЭВМ БЭСМ-6

1973
ЛАБОРАТОРИЯ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ И АВТОМАТИЗАЦИИ

11 - 7396

О Ен Ир, В.П. Шириков

НОВЫЕ ВОЗМОЖНОСТИ ИСПОЛЬЗОВАНИЯ ЛОГИЧЕСКИХ ВЫРАЖЕНИЙ ПРИ ПРОГРАММИРОВАНИИ НА ЯЗЫКЕ ФОРТРАН ДЛЯ ЭВМ БЭСМ-6

Введение

В течение нескольких лет эксплуатации транслятора с ФОРТРАНа, содержащегося в системе математического обеспечения БЭСМ-6, имевшиеся недостатки этого транслятора в основном устранены.

Однако ввиду специфической особенности машины БЭСМ-6 и алгоритма ее транслятора с ФОРТРАНа, использованного при его разработке, возможность использования логических выражений при программировании на ФОРТРАНе имела некоторые ограничения.

В настоящей работе описываются причины, ограничивавшие возможность использования логических выражений, и изменения в трансляторе, сделанные для устранения таких ограничений.

Новый вариант транслятора введен в эксплуатацию (версии транслятора от 8 июня 1972 года и от 5 марта 1973 года).

I. Расширение возможности использования логических операций .FQ. и .NE. между целыми операндами.

В ряде программ (особенно заимствованных из ЦЕРНа) в один и тот же массив засылаются значения самых различных типов и, в частности, текстовые константы перемешиваются с вещественными и целыми в одном массиве, как правило, описанном в операторе COMMON ; далее программист устраивает сравнение между элементами этого массива и некоторыми константами, и в зависимости от результата сравнения указывает передачи управления.

Пример:

```
      SUBRØUTINE  A
      CØMMØN  M(5O)
      DØ 1O I=1,5O
      IF ( M(I) .EQ.O) GØ TØ 1O
      :
      :
1O CØNTINUE
      :
      :
      END
```

В приведенном примере в случае, когда значениями M(I) оказываются текстовые константы, указанные сравнения, как правило, выполнялись неверно. Такое неверное сравнение возникало из-за следующей особенности машины БЭСМ-6 и алгоритма ее транслятора с ФОРТРАНа:

(I) в машине БЭСМ-6 числа представляются с плавающей запятой и все арифметические операции производятся с числами с плавающей запятой /5/, но нормализация налево результата операции между целыми операндами блокируется (целые величины считаются ненормализованными).

(2) алгоритм старого варианта транслятора составлен так, что логические операции заменяются на арифметические, и результат логической операции .EQ. или .NE. определяется в зависимости от сравнения между нулевым кодом и мантиссой (со знаком) результата выполнения арифметической операции вычитания, соответствующей операции.EQ.или .NE. между целыми операндами /2/.

Следовательно, проверка M(I).EQ.N сводилась к определению равенства нулю выражения M(I) -N . При выполнении вычитания предварительно выполняется выравнивание порядков за счет сдвига мантисс (нормализация вправо) для величин M(I) и N .

Целое N (даже если N =0) всегда имеет большой порядок;

если M(I) - текстовая константа, то выравнивание порядков приводит практически к ее исчерпыванию, и M(I) всегда оказывалась равной целому нулю.

Поэтому в варианте транслятора от 8.6.1972 года операция сравнения .EQ. или .NE. между целыми операндами делается не вычитанием, как ранее, а машинной операцией логического поразрядного сравнения.

Это реализовалось изменением блока BØØLEANa /2/.

2. Реализация возможности использования логических фактических параметров в арифметическом операторе присваивания.

Если в правой части арифметического оператора присваивания применяется в качестве фактического параметра логическое выражение, то при трансляции этого оператора старый транслятор приписывает правой части тип логического выражения.

Пример

```
      PRØGRAM  C
      :
      :
      P=1O+F(.TRUE.)
      :
      :
      END
      FUNCTIØN F(L)
      LØGICAL  L
      :
      :
      END
```

При трансляции оператора P=1O+F(.TRUE.)транслятор печатал информацию

"НЕПРАВИЛЬНОЕ ПРИСВАИВАНИЕ В АРИФМЕТИЧЕСКОМ ОПЕРАТОРЕ"

и запрещал его трансляцию.

Это устранялось добавлением к блоку CØNTRØL части, делающей дополнительные проверки и сравнения типов левой и правой частей операторов присваивания.

3. Реализация возможности использования логических выражений в качестве фактических параметров .

Нам было известно, что программа не пройдет тогда, когда выражение содержит в фактическом параметре логические операции, операции отношения.

Пример

```
      PRØGRAM  AB
      LØGICAL  L
        ⋮
      L =10 .GT. F (A.EQ.B)
```

```
24104
НЕПРАВИЛЬНО ПОСТРОЕНО ЛОГИЧЕСКОЕ ВЫРАЖЕНИЕ
      IF (5.EQ.F(L.ØR.FALSE.)) GØ TØ 100
24104
НЕПРАВИЛЬНО ПОСТРОЕНО ЛОГИЧЕСКОЕ ВЫРАЖЕНИЕ
        ⋮
  100 CØNTINUE
      IF (5-F(A.NE.B)) 200,300,400
22212
ЛОГИЧЕСКАЯ ОПЕРАЦИЯ В ПАРАМЕТРАХ CALL
        ⋮
  200
  300
  400
      CALL S(A.NE.B,R)
22212
ЛОГИЧЕСКАЯ ОПЕРАЦИЯ В ПАРАМЕТРАХ  CALL
        ⋮
      END
      FUNCTIØN F (L)
      LØGICAL L
        ⋮
      END
      SUBRØUTINE S(L,D)
      LØGICAL L
        ⋮
      END
```


Причиной запрета было то, что в процессе трансляции рекурсивное обращение к блоку транслятора невозможно [2].

В описываемом случае происходит следующее:

I). Когда блок CØNTRØL , управляющий обработкой выполняемых операторов на ФОРТРАНе, встречается с логическим оператором присваивания, он вызывает блок BØØLEAN , который делает обработку логического оператора присваивания.

Блок BØØLEAN вызывает свою подпрограмму LØGEX для обработки логического выражения.

С другой стороны, если блок CØNTRØL принимает логический IF -оператор, то он вызывает блок IFLØGIC , который обрабатывает логический IF -оператор.

Блок IFLØGIC обращается к подпрограмме LØGEX . Однако, в этом случае, если в логическом выражении находится вызов функции, имеющей логическое выражение в качестве фактического параметра, то для обработки такого фактического параметра необходимо рекурсивное обращение к подпрограмме LØGEX . Поэтому транслятор выдает информацию

" 24104 НЕПРАВИЛЬНО ПОСТРОЕНО ЛОГИЧЕСКОЕ ВЫРАЖЕНИЕ"

(через подпрограмму LØGEX) и запрещает его трансляцию.

2). Когда управляющий блок CØNTRØL принимает арифметический оператор присваивания или обращение к подпрограмме, этот CØNTRØL вызывает блок PRØPI , который выполняет первичную обработку арифметических выражений.

Если внутри арифметического оператора присваивания или вызова подпрограммы находится фактический параметр, который имеет вид логического выражения, то блок PRØPI должен вызывать подпрограмму LØGEX для обработки этого логического выражения.

В этом случае старый транслятор печатает информацию

"22212 ЛОГИЧЕСКАЯ ОПЕРАЦИЯ В ПАРАМЕТРАХ CALL"

(через блок PRØPI) и прекращает трансляцию оператора.

Причина заключается в том, что подпрограмма LØGEX преобразует логические операции (кроме операции .EQ. или .NE. между целыми операндами) в последовательность арифметических

выражений и вынуждена, в свою очередь, вызывать блок PRØPI .

По этой же причине блок PRØPI не может вызывать блок BØØLEAN . Такая трудность в трансляции устраняется тем, что вставляется обращение к новому блоку SLØP в блоке CØNTRØL перед обращениями к блокам PRØPI, BØØLEAN, IFLØGIC и в блоке IFARITH перед обращением к PRØPI (см. /2/, /7/).

Блок SLØP.

Основная функция блока SLØP заключается в том, что он выделяет фактические параметры, имеющие логические операции, из выражения,
образует последовательность промежуточных логических операторов присваивания,
вместо выделенных параметров вставляет соответствующие ему промежуточные переменные присваивания в выражении и,
прежде чем транслировать результирующее выражение,
транслирует последовательность промежуточных операторов присваивания.

Например, с помощью блока SLØP оператор

```
IL= A + F (P, A.EQ.B, F1(D.GT. F2(K.ØR.N))),
```

где F, F1, F2 - функции,
приводится к последовательности промежуточных логических операторов присваивания:

```
*V0001 = K.ØR.N
*V0002 = D.GT.F2(*V0001)
*V0003 = A.EQ.B
```

и результирующего оператора

```
IL = A + F(P, *V0003, F1 (*V0002)).
```

Объектом работы блока SLØP являются PI-строки операторов присваивания, обращений к подпрограммам и выражений, находящихся внутри IF-операторов (арифметических и логических).

Исходной информацией для блока SLØP являются начальный адрес рассматриваемого объекта, находящийся на сумматоре, и его конечный адрес, находящийся в магазине.

Блок SLØP состоит из следующих частей:

1) входная (хранящая) часть (SLØP)

2) часть поиска обращения к функции (или подпрограмме) в просматриваемой информации (SEARFUN)

3) часть выбора фактических параметров (SEARLEP-SEARRIG)

4) часть поиска логической операции в фактическом параметре (SEARCHLØ)

5) часть восстановления размеченных скобок в фактическом параметре, имеющем логические операции (RECPAREN)

6) часть выделения и трансляции фактических параметров, имеющих логические операции (EXTLEP)

7) выходная (восстанавливающая) часть (RECØN)

Описание отдельных частей SLØPа.

1) SLØP :

Производится засылка начального и конечного адресов просматриваемого объекта в рабочие ячейки.
Содержимое индексных регистров (I-I3) хранится в магазине.
Стираются индикатор (указатель) функции, индикатор ошибки и счетчик числа фактических параметров, содержащих логическую операцию.

2) SEARFUN:

Производится поиск вызовов (функций или подпрограмм). Для просматриваемого объекта такой поиск производится в обратном направлении (справа налево), что дает возможность использовать малое число рабочих ячеек. Это особенно удобно при обработке объектов, которые содержат обращения к функциям нескольких уровней.

3) SEARLEP-SEARRIG

Производится выбор фактического параметра, найденного с помощью SEARFUN .

При этом просмотр производится в прямом направлении (слева направо).
Вместе с выбором фактических параметров производится разметка для скобок списка параметров.
На прямом просмотре при встрече с размеченной открывающей скобкой часть, которая находится между этой скобкой и соответствующей ей закрывающей скобкой, не просматривается.
В этой части проверяется соответствие скобок.

4). SEARCHLØ:

Производится поиск знака логической операции в каждом параметре, выбранном с помощью SEARLEP- SEARRIG.
Направление поиска при этом прямое.

5). RECPAREN:

В фактическом параметре, выбранном с помощью SEARCHLØ , восстанавливаются размеченные скобки.

6). EXTLEP:

Для выбранного фактического параметра, который имеет логический знак операции и восстановленные скобки, производится следующий процесс обработки.

(I). Вводится новая промежуточная локальная переменная, имеющая логический тип.

а). Составляется новое промежуточное обозначение (идентификатора) в Σ -коде.
Составляется промежуточный внутренний идентификатор ⁎VN с помощью подпрограммы SH блока CØNTRØL , где ⁎ - символ, V - буква, N - четыре десятичных цифры. N определяется по содержимому ячейки RLVCNT , в которой регистрируется число фактических параметров, содержащих логическую операцию.

в). Переводится полученная Σ -строка в PI -строку. Σ -код ⁎VN засылается в таблицу (IDENTLIS) не описанных ни в одном из декларативных операторов простых переменных и переводится в PI -код.

Тип ⁎VN автоматически становится вещественным, так как первая буква ⁎ не есть одна из букв I,J,K,L,M,N.
Для этого перевода сделано дополнительное обращение к входу CØNALFA в начале той части блока PRØSIGa (переводящего Σ -коды в PI -коды), которая проверяет, не использовался ли уже данный идентификатор в Σ -строке, записывает этот идентификатор в соответствующую таблицу и составляет соответствующую ему PI -строку /2/.

с). Смена типа.
В полученном PI -коде тип REAL на тип LØGICAL.
Таким образом, в PI -строке, наконец, получается новая промежуточная внутренняя переменная, имеющая логический тип.

(2). Из PI-строк выделяется фактический параметр, являющийся логическим выражением, и на его место вставляется PI-код, соответствующий переменной ⁎VN.
Затем составляется следующий промежуточный оператор присваивания:

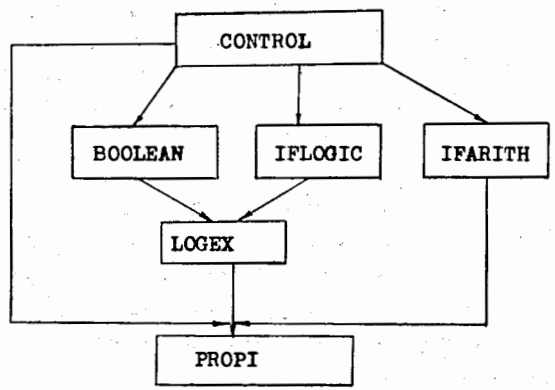
⁎VN = выделенный фактический параметр.

(3). Для обработки этого оператора производится обращение к блоку BØØLEAN .

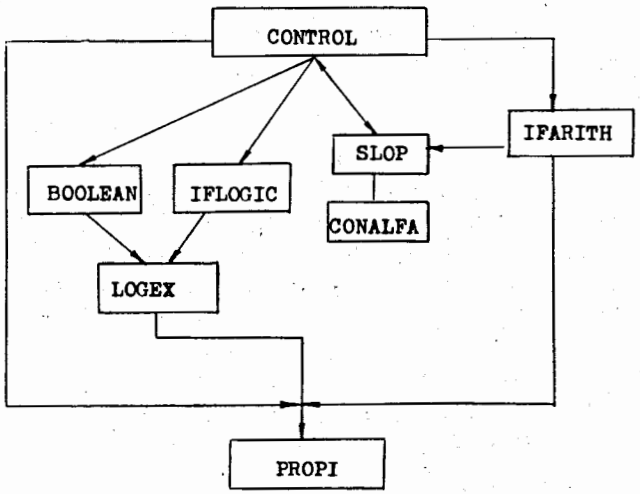
7). RECØN:

Проверяется, был ли вызов (функции, подпрограммы) в данном выражении. Если да, восстанавливаются размеченные скобки, которые являются скобками списков функций, не содержащихся в фактических параметрах, имеющих логические знаки операции.
Во всех случаях (за исключением случаев, когда имеются ошибки) при выходе из SLØP восстанавливаются индексные регистры (I-I3).
Для включения блока SLØPa в транслятор подправлялись следующие блоки транслятора: CØNTRØL, PRØSIG, PRØPI, BØØLEAN, IFARITH, LISTD, подпрограмма LØGEX.

Блок-схема работы транслятора, связанной с обработкой логических операций.



a) Старый вариант.

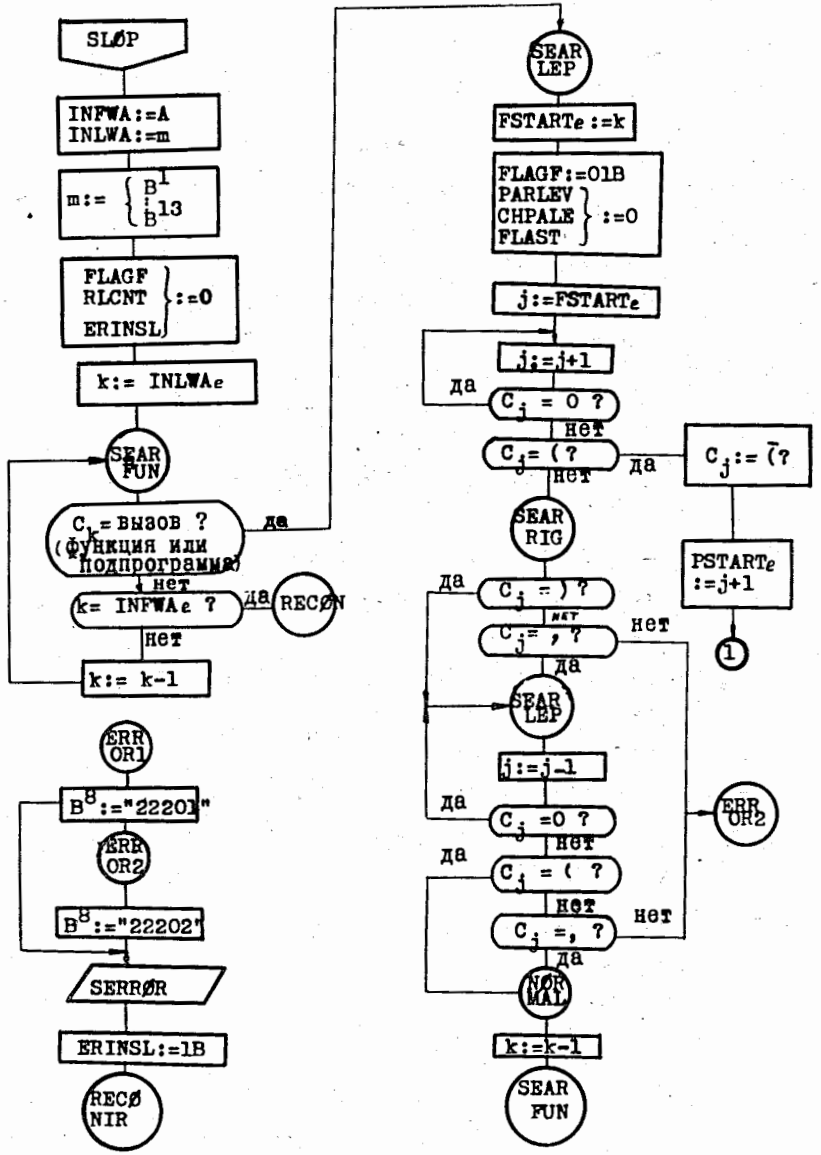


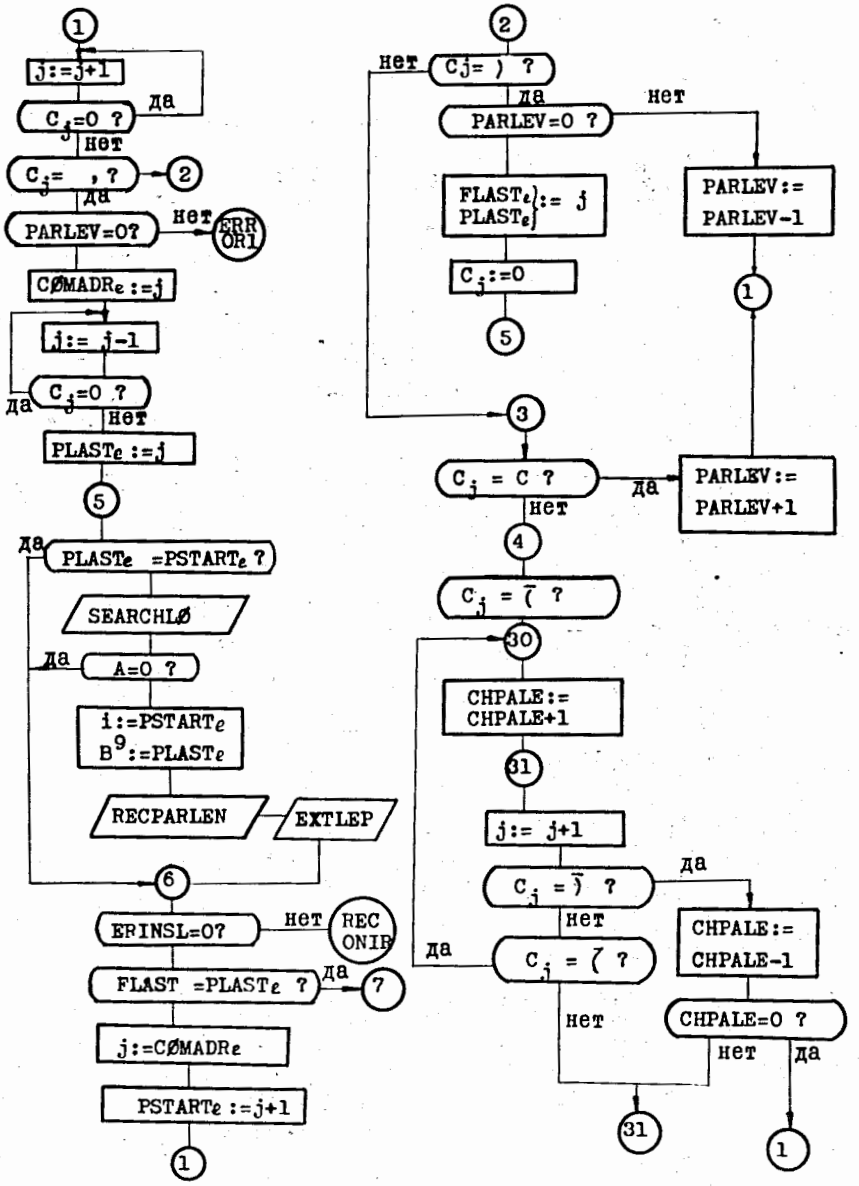
b). Новый вариант.

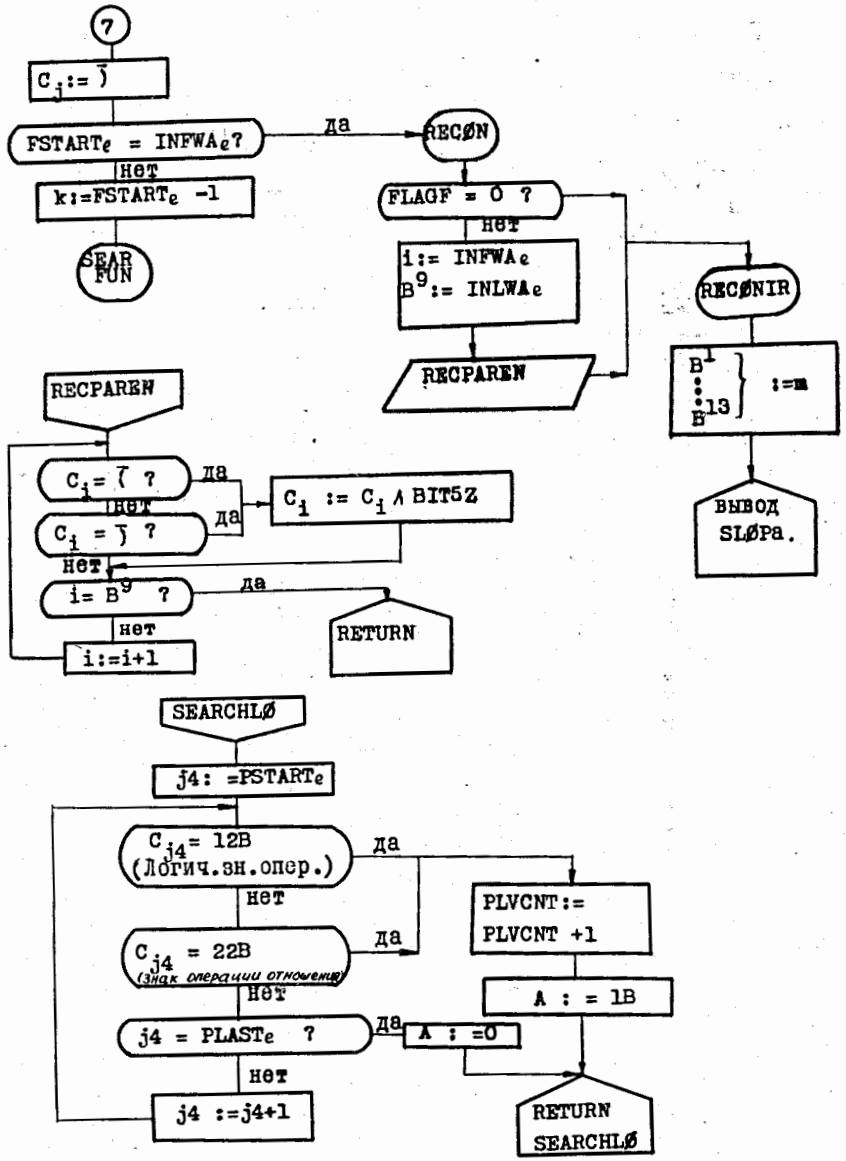
Обозначения, использующиеся в блок-схеме

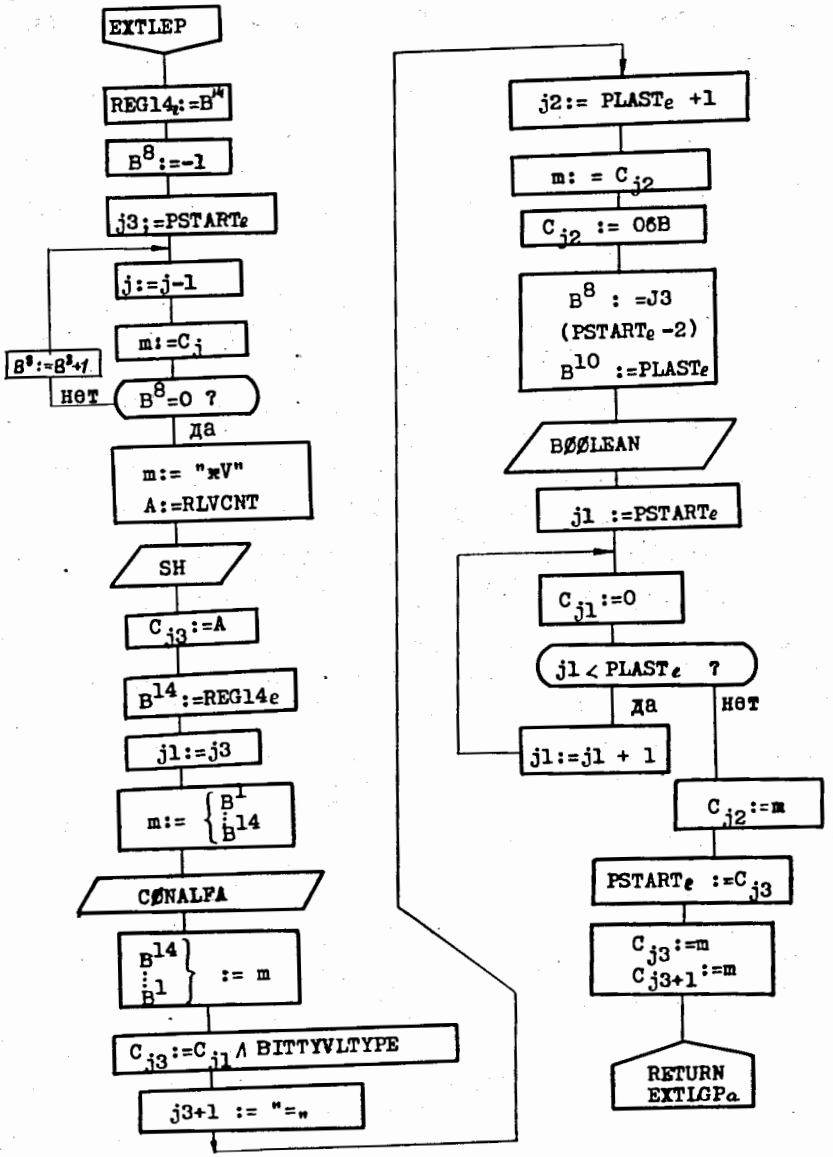
A	– сумматор,
m	– магазин,
c_k	– содержимое ячейки с номером К,
$\alpha := \beta$	– содержимое идентификатора β записывается в ячейку с идентификатором α.
B^i	– i-ый индексный регистр.
	– обращение к подпрограмме, входу или внутренней метке.
=	– сравнение.
),(	– скобки.
$\overline{)},\overline{(}$	– размеченные скобки.
IFFWA	– начальный адрес рассматриваемого объекта.
INLWA	– конечный –"– –"– –"–.
FLAGF	– идентификтор функции или подпрограммы.
RLCNT	– счетчик логических выражений в параметрах.
ERINSL	– индикатор ошибок.
$FSTART_e$	– начальный адрес вызовов функций или подпрограмм (адрес правой скобки список параметров).
PARLEV	– счетчик пар скобок.
CHPALE	– счетчик пар размеченных скобок.
$PSTART_e$	– начальный адрес параметра.
$PLAST_e$	– конечный адрес параметра.
$CØMADR_e$	– адрес запятой.

Схема блока SLØPa.






7
C_j:=)
FSTART$_e$ = INFWA$_e$?
да
RECØN
нет
k:=FSTART$_e$ -1
SEAR FUN
FLAGF = 0 ?
нет
i:= INFWA$_e$
B^9:= INLWA$_e$
RECPAREN
RECØNIR
B^1 ... B^{13} :=m
ВЫВОД SLØPa.
RECPAREN
C_i= (?
да
нет
C_i =) ?
да
C_i := C_i ∧ BIT5Z
нет
i= B^9 ?
да
нет
i:=i+1
RETURN
SEARCHLØ
j4: =PSTART$_e$
C_{j4}= 12B (Логич.зн.опер.)
да
нет
C_{j4} = 22B (знак операции отношения)
да
нет
j4 = PLAST$_e$?
да
нет
j4 :=j4+1
PLVCNT:= PLVCNT +1
A : = 1B
A : =0
RETURN SEARCHLØ


EXTLEP
REG14$_e$:=B^{14}
B^8:=-1
j3:=PSTART$_e$
j:=j-1
m:=C_j
B^8:=B^8+1
нет
B^8=0 ?
да
m:= "*V"
A:=RLVCNT
SH
C_{j3}:=A
B^{14}:=REG14$_e$
j1:=j3
m:= B^1 ... B^{14}
CØNALFA
B^{14} ... B^1 := m
C_{j3}:=C_{j1} ∧ BITTYVLTYPE
j3+1 := "=,
j2:= PLAST$_e$ +1
m: = C_{j2}
C_{j2} := 06B
B^8 : =J3 (PSTART$_e$ -2) B^{10} :=PLAST$_e$
BØØLEAN
j1 :=PSTART$_e$
C_{j1}:=0
j1 < PLAST$_e$?
да
нет
j1:=j1 + 1
C_{j2}:=m
PSTART$_e$:=C_{j3}
C_{j3}:=m
C_{j3+1}:=m
RETURN EXTLGPa

В заключение авторы выражают сердечную благодарность Пак Хон Чору, Г.Л.Мазному, И.Н.Силину, Н.С.Заикину за помощь и замечания.

ЛИТЕРАТУРА

1. Язык ФОРТРАН (под редакцией В.П.Ширикова), ОИЯИ, Дубна, 1969.
2. З.Бродцински и др. Транслятор с языка ФОРТРАН для системы математического обеспечения БЭСМ-6. Труды первой всесоюзной конференции по программированию, г. Киев, 1968.
3. Г.Л.Мазный. Мониторная система "Дубна", 11-5974, ОИЯИ, 1971.
4. А.И.Волков. Автокод MADLEN. Б4-11-4654, ОИЯИ, 1969.
5. Инструкция по программированию на БЭСМ-6.ИТМ ВТ АН СССР,М.1967.
6. Б.Рендделл, Л.Рассел. Реализация АЛГОЛ-60. "Мир", М, 1967.
7. З.Бродцински, Н.Н.Говорун и др. Система математического обеспечения ЭВМ БЭСМ-6. Транслятор с языка ФОРТРАН (части I, П,Ш,IУ). Депонированные публикации ОИЯИ, Б1-11-7160 и Б1-11-7162, Дубна, 1972.

Рукопись поступила в издательский отдел
7 августа 1973 года.