СООБЩЕНИЯ
ОБЪЕДИНЕННОГО
ИНСТИТУТА
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ
ДУБНА

11 - 9766

Г.Е.Бабаян, Г.А.Ососков

ТРАНСЛЯТОР С ЯЗЫКА **COPLAN** В МАШИННЫЙ КОД ЭВМ ТИПА "ЭЛЕКТРОНИКА-100"

1976

11 - 9766

Г.Е.Бабаян, Г.А.Ососков

ТРАНСЛЯТОР С ЯЗЫКА **COPLAN** В МАШИННЫЙ КОД ЭВМ ТИПА "ЭЛЕКТРОНИКА-100"

1. Общее описание транслятора

Специализированный язык COPLAN [1] был разработан в целях написания управляющих программ для малых ЭВМ.

Настоящий транслятор для малой ЭВМ типа "Электроника-100" [2,3], выполненный на языке ФОРТРАН, предназначен для перевода исходной программы на COPLANe в объектную программу на языке ЭВМ.

Фортранная реализация этого транслятора позволяет легко адаптировать его для любой ЭВМ, снабженной транслятором с ФОРТРАНа.

Практика переноса работ по трансляции языков для малых ЭВМ на большие машины, так же как эмуляция на последних работы мини-ЭВМ, весьма распространилась в последние годы [4]. Особенно широкий опыт в этом отношении имеется в вычислительных центрах ОИЯИ [5,6] и ЦЕРНа [7].

Вариант транслятора, описываемый в настоящей работе, существенно модернизирован по сравнению с описанным в работе [1] транслятором, переводящим программы с COPLANa в автокод и ориентированным на последующее применение уже имевшихся фортранных ассемблеров для мини-ЭВМ РДР-9 и РДР-8 [1].

Основными целями этой переделки были:

- объединение в одной программе транслятора COPLAN–автокод и ассемблера, переводящего автокодную программу в текст на машинном языке, годный к загрузке;
- сохранение возможности простого перехода к другому автокоду.

Этих целей удалось достичь главным образом путем использования при трансляции метода интерпретации [8],

позволившего значительно упростить транслятор за счет исключения промежуточного функционального языка.

Кроме этого, были сделаны следующие полезные изменения:

- применена более простая процедура разбиения памяти на страницы;
- организована независимая трансляция подпрограмм;
- введены дополнительные средства языка COPLAN /9/, такие как одномерные массивы, общие блоки памяти /типа COMMON в ФОРТРАНе/ и др.

Метод интерпретации реализован путем применения программного стека /магазина/ /10/.

Условно транслятор можно разделить на три основные части:

- транслятор COPLAN -автокод;
- ассемблер /выполняющий перевод автокод - язык загрузки/;
- загрузчик /выполняющий перевод язык загрузки - машинный код/.

Результат работы транслятора - программа на машинном языке в абсолютных адресах - выдается на магнитную или бумажную ленту. Для исполнения программа должна быть введена с соответствующего носителя в память ЭВМ и управление должно быть передано на первую исполнимую команду программы. Наличие эмулятора для ЭВМ типа "Электроника-100" позволяет обойтись без этапа выдачи программы на внешний носитель и исполнить ее прямо на большой ЭВМ.

2. Перевод исходной программы на внутренний язык

Процесс трансляции в первой части производится в один просмотр транслируемой информации в соответствии со следующей схемой:

чтение оператора - опознавание оператора - лексический анализ - генерация соответствующих команд на автокоде.

Для обработки выражений языка COPLAN служит метод, основанный на применении обратной польской записи, а для перевода выражений в обратную польскую запись - алгоритм, использующий магазинную память для операндов и операций и названный методом двойного старшинства /11/.

Все выражения исходной программы переводятся в обратную польскую запись с одновременной генерацией команд языка автокода.

Это позволило полностью исключить подпрограмму COPCOM, которая переводила исходную программу во внутренний функциональный язык транслятора, описанный в работе /1/.

Обработка декларативных операторов

Эти операторы преобразуются в соответствующие таблицы. При этом каждая строка таблицы блоков / DIMEN или COM / соответствует одному оператору описания массива /bss или com/ автокодного текста транслированной программы. Строка таблицы эквивалентностей соответствует оператору EQU на языке автокода. Трансляция операторов CON проводится без заполнения таблицы.

С помощью декларативных операторов EXTERN и ENTRY происходит связывание подпрограмм в COPLANe /9/. Для каждого имени из таблицы, соответствующей команде EXTERN, отводится ячейка в нулевой странице, куда в дальнейшем будет заноситься адрес метки с тем же именем, описанным в команде ENTRY.

Когда в исходной программе встречается оператор CALL SUBROUT, то генерируется команда безусловного перехода с косвенной адресацией к ячейке с именем SUBROUT в нулевой странице. Эта ячейка с именем SUBROUT должна быть уже зарезервирована с помощью декларативного оператора EXTERN SUBROUT в вызывающей программе, и в ней уже должен находиться адрес первой выполняемой команды подпрограммы. Последнее условие выполняется с помощью оператора ENTRY SUBROUT подпрограммы SUBROUT.

3. Первый проход ассемблера

При переводе автокодного текста программы в машинный код ассемблер должен, в частности, распределить память, то есть каждому имени и литералу поставить в соответствие абсолютный адрес, перевести команды автокода в машинные инструкции с учетом этого распределения памяти, найти возможные ошибки и выдать их диагностику, сформировать программу и выдать ее на перфо- или магнитную ленту, напечатать листинг.

В данной работе была использована схема обычного двухпроходного ассемблера /10/, в первом проходе которого выявляются имена и литералы и распределяется память.

Задача автоматического распределения памяти в случае мини-ЭВМ с короткими словами весьма осложняется тем, что из-за малости адресной части команд непосредственная адресация допустима только в пределах коротких страниц, на которые разбивается память таких ЭВМ. Например, в ЭВМ типа "Электроника-100", имеющей 12 разрядов в слове памяти, под адресную часть отводится только 7 разрядов, что определяет длину страницы /всего 128 слов/, в пределах которой допустима прямая адресация.

При ручном программировании в автокоде эта проблема просто решается аппаратом косвенной адресации. Однако при трансляции с языка COPLAN в автокод даже довольно простых выражений часто возникает последовательность команд, превышающая длину отдельной страницы. Процедура разбиения памяти на страницы и организации косвенной адресации выполнялась автоматически в варианте транслятора для ЭВМ типа РДР-8, упомянутого в работе /1/, с помощью подпрограммы PAGE . PAGE осуществляла для этого отдельный проход до работы ассемблера, рассчитанного на программы, уже разбитые на страницы или подготовленные для этого вручную с помощью оператора базирования /12/.

Алгоритм, реализованный в программе PAGE, был основан на следующих соображениях. Программа в символических адресах, получаемая после трансляции с исходного текста на COPLANe, разбивалась на отрезки по 128 слов. Очередной отрезок многократно просматривался для определения операндов, требующих косвенной адресации, которая осуществлялась путем резервирования ячеек в конце этого отрезка за счет выталкивания хвостовых команд отрезка на следующую страницу. Когда сумма просматриваемых и резервированных слов составляла 126 /2 слова необходимы для передачи управления на другую страницу/, просмотр опять повторялся, чтобы обнаружить возможное появление вверху отрезка операндов, которые начали требовать косвенной адресации из-за вытеснения соответствующих имен внизу отрезка на следующую страницу. Такое повторение просмотров продолжалось до тех пор, пока заявки на косвенную адресацию не были исчерпаны.

Совмещение в одном проходе процедур распределения памяти и автоматического разбиения ее на страницы значительно ускорило общее время трансляции как за счет исключения лишнего прохода, так и благодаря ускорению самого алгоритма разбиения, реализация которого в рамках ассемблера много проще, чем в программе PAGE. Остановимся на этом подробнее.

Сущность предлагаемого метода разбиения памяти на страницы состоит в формировании буфера для косвенной адресации в случае, когда команда и используемый ею адрес находятся на разных страницах. Команда использует ячейку буфера, находящуюся на той же самой странице.

Для данного сегмента программы /программа, подпрограмма, подпрограмма-функция/ образуется таблица имен IPP, общая для всех страниц /минимально может быть только одна страница памяти/, на которых будет размещен этот сегмент программы. Для каждой страницы размером в 128 слов образуются две таблицы: таблица имен IM всех идентификаторов и вторая таблица IPS тех имен, для которых потребуется организация косвенной адресации. На каждой странице резервируются две последние ячейки для того, чтобы поместить туда две команды, реализующие переход от одной страницы на другую. После этого последовательно просматривается

команда за командой и ведется учет количества резервированных ячеек. Для каждой команды на данной странице резервируется одна ячейка памяти для нее самой и, если требуется, еще одна - для организации возможной косвенной адресации. Если команда имеет метку и/или/ операнд, то проверяется: описаны и/или/находятся ли они в таблице имен IM . При необходимости описывается метка или резервируется ячейка для организации косвенной адресации. В случае дважды описанной метки делается сообщение об ошибке. Если в качестве операнда имеется литерал, то резервируется дополнительная ячейка, куда будет занесено его двоичное значение.При этом для одного и того же литерала на данной странице резервируется только одна ячейка памяти.

Каждый раз при переходе к следующей команде проверяется, равно ли количество уже резервированных ячеек числу 125 либо 126, то есть использована ли страница памяти полностью.

ЭВМ типа "Электроника-1ОО" - машины с одноуровневой косвенной адресацией, поэтому возникают дополнительные трудности, если встречается команда с косвенной адресацией на другую страницу. Назовем такие команды "плохими". При переходе исходной программы на языке COPLAN во внутренний язык в двух случаях может быть генерирована "плохая" команда: 1/ при обработке команд EXIT SUBR /где SUBR - имя подпрограммы/подпрограммы-функции/ и 2/ при обработке команд замены формальных параметров на фактические, то есть команд вида

SUBR FNCT (A, B,..., C).

"Плохая" команда может встретиться также в случае 3/, когда в исходной подпрограмме была команда с косвенной адресацией и при трансляции /загрузке/ в абсолютной форме сама команда и ее адрес оказались на разных страницах.

В первом случае /он неизбежно возникает, когда размер подпрограммы превышает 200_8 ячеек/ трудность можно преодолеть: как только встречается первая команда подпрограммы /она имеет вид SUBR ϕ /, на данной

странице резервируется ячейка, куда заносится команда JMP I SUBR. Когда же встречается команда JMP I SUBR /в которую была переведена команда EXIT SUBR при переводе во внутренний язык/, она заменяется командой безусловного перехода JMP к уже резервированной ячейке.

В двух других случаях делается следующее: данная подпрограмма полностью переводится на начало следующей страницы. Если из-за "плохих" команд, возникших при обработке SUBR FNCT(A,B,..,C), максимальная потеря ячеек памяти равна n * 3 , где n - число параметров FNCT, то для "плохих" команд другого вида потеря может быть больше. Трансляция подпрограммы такой "плохой" командой становится невозможной, если при загрузке подпрограммы с новой страницы возникнет та же самая ситуация. В этом случае выдается соответствующая диагностика, и программист должен учесть это.

Прежде чем реализовать переход с одной страницы на другую, надо проверить, не является ли последняя команда страницы "неразделимой" /команда условной передачи управления, состоящая фактически их двух слов, или группа слов, являющаяся списком параметров вызываемой подпрограммы и обязанная следовать за командой JMS ухода на подпрограмму/. "Неразделимые" команды переводятся на следующую страницу памяти. Если такие команды не имеют метки, то вместо них вставляется команда автокода NOP. Если же метка имеется, то ее признак описания в таблице имен IM ликвидируется, а освободившаяся ячейка используется для организации косвенной адресации тех команд, операндом которых являлось имя этой метки.

После этого просматривается таблица имен IM данной страницы. Имена с признаком описания этой таблицы заносятся в таблицу IPP,общую для данного сегмента программы, остальные - в таблицу IPS - таблицу имен, для которых требуется косвенная адресация на данной странице. Для каждого имени этой таблицы образуется новое имя, которое будет являться меткой для резервированной ячейки, в которую будет помещен действительный адрес этого имени из таблицы IPP.

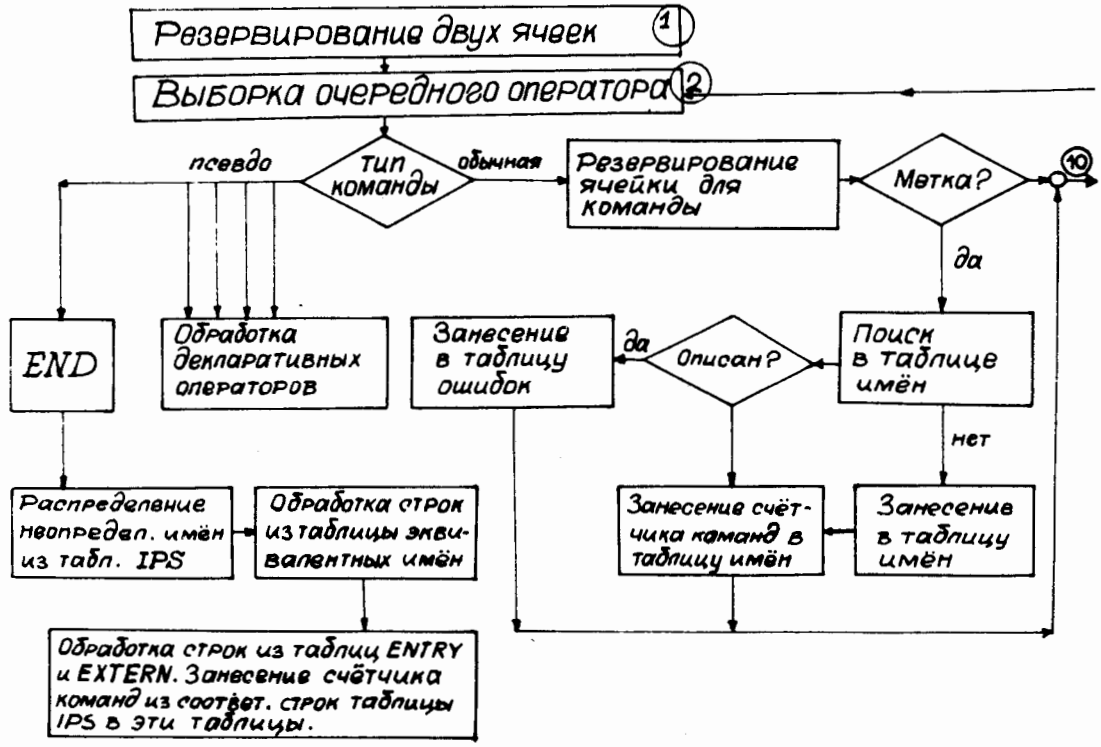


Рис. 1. Блок-схема первого прохода ассемблера /начало/.

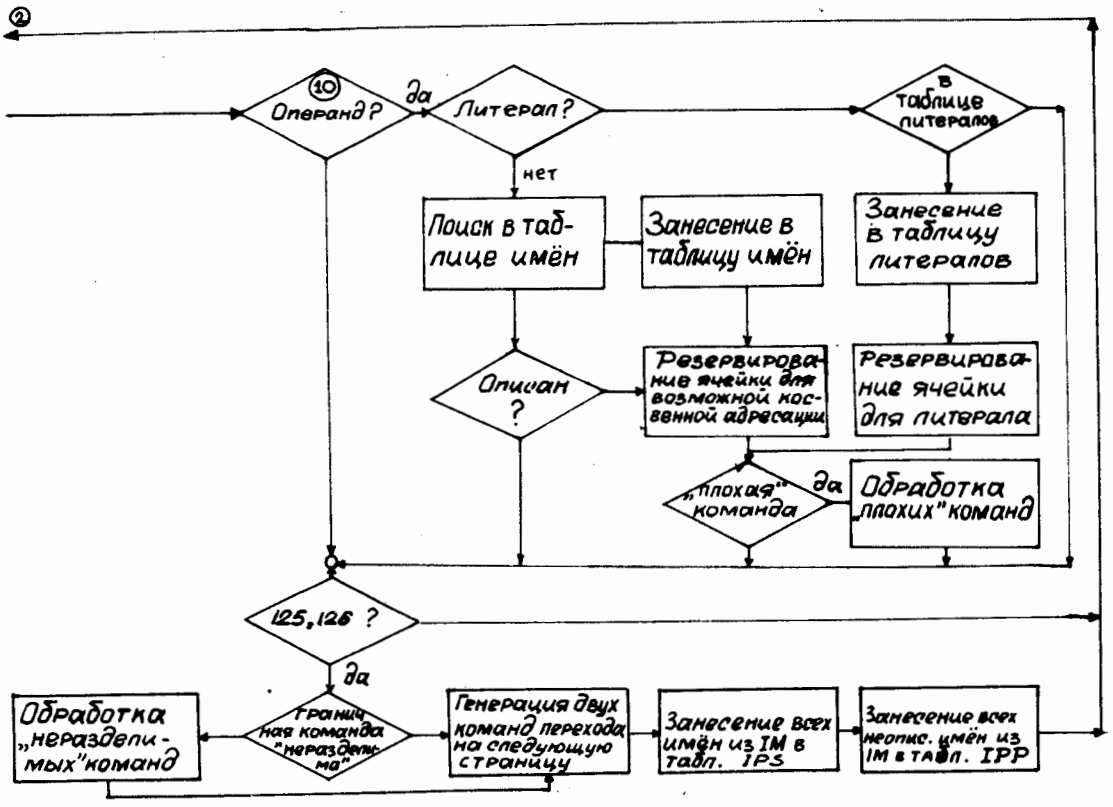


Рис. 1 /продолжение/.

Эти таблицы решают вопрос распределения памяти и разбиения памяти на страницы и используются во время второго прохода ассемблера.

На *рис. 1* приведена блок-схема первого прохода ассемблера.

4. Второй проход ассемблера

Назначение этого прохода - разобраться в каждой команде и заменить ее на эквивалентную команду машинного языка с соответствующим кодом операции, признаком косвенной адресации, если это потребуется, и адресом.

Схема генерирования машинных команд и формирования выходной информации показана на *рис. 2.*

В начале второго прохода устанавливается в нуль счетчик команд и общий счетчик длины программы. Затем выбирается очередной оператор исходной программы и заносится в символическую часть массива "печать". По коду, записанному в поле операции, с помощью таблицы операций определяется тип команды.

При обработке машинных команд просматривается поле операндов, выделяется операнд и ищется в таблице P соответствующей страницы. Наличие имени операнда в таблице означает, что в этой команде требуется организация косвенной адресации /устанавливается признак косвенной адресации/ на ячейку, зарезервированную во время первого прохода ассемблера. В соответствии с этим формируется соответствующий машинный операнд /по таблице IPP /. Сформированная машинная команда заносится в массивы "программа" и "печать" для последующей печати в одной строке мнемокода операции и соответствующего машинного эквивалента.

Псевдокоманды EQU, ENTRY , EXTERN и др. не обрабатываются, поскольку содержащаяся в них информация перенесена в соответствующие таблицы при первом проходе ассемблера.

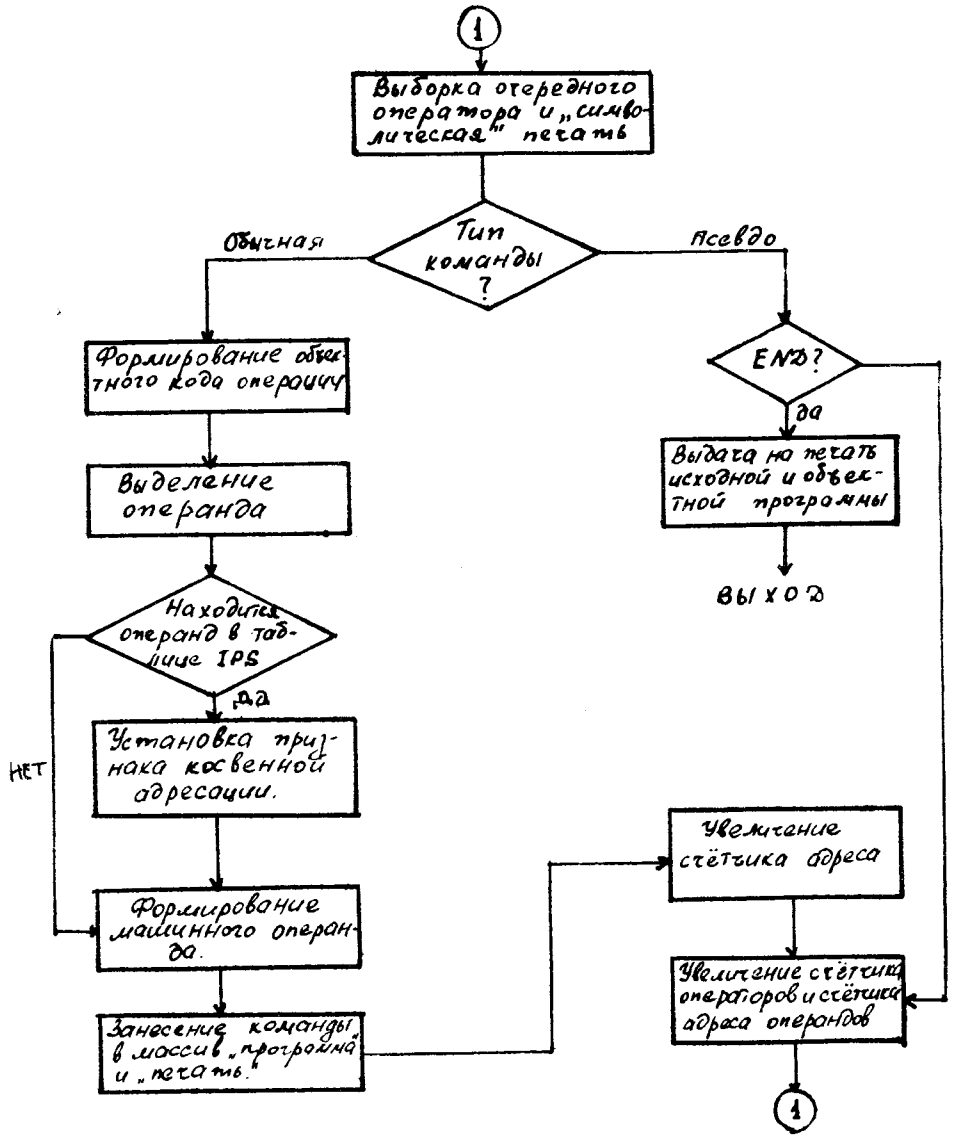


Рис. 2. Блок-схема второго прохода ассемблера.

5. Директивы транслятора

Директивы транслятора служат для управления транслятором. В первой колонке перфокарт, содержащих операторы с этими директивами, пробивается знак "+".

Имеется два оператора директив транслятора:

```
+      CPL     μ1, μ2, μ3, μ4
+              FIN
```

Первый из них - управляющий оператор транслятора - управляет режимами работы транслятора. Код CPL пробивается с десятой колонки, а параметры $\mu_1, \ldots, \mu_4$ - с 20-й колонки перфокарт. Количество параметров - один и более. Параметры отделяются друг от друга запятыми, пробелы не допускаются. Они могут комбинироваться любым образом. Смысл параметров следующий:

- B - выдается на печать и перфорируется колода перфокарт, на которых пробиты две колонки восьмеричных цифр, обозначающих адреса ячеек памяти и содержимое загруженной в абсолютной форме программы, начиная с ячейки памяти, указанной в декларативном операторе CORG .
- C - выдается листинг COPLANa.
- L - выдается листинг ассемблера.
- T - выдается на печать таблица имен с присвоенными им абсолютными адресами.

Второй оператор - оператор конца трансляции - FIN пробивается с 20-й колонки на перфокарте и служит для задания конца работы транслятора.

6. Сообщения об ошибках

Транслятор находит определенные в исходной программе ошибки и выдает соответствующую диагностику. В зависимости от того, где определена ошибка - при переводе во внутренний язык, во время первого прохода или второго, - двухбуквенное обозначение ошибки имеет соответственно вид: CX, AX, DX.

- CS - ошибка в управляющем операторе: отсутствует знак "+" в первой колонке или встретился недопустимый параметр.

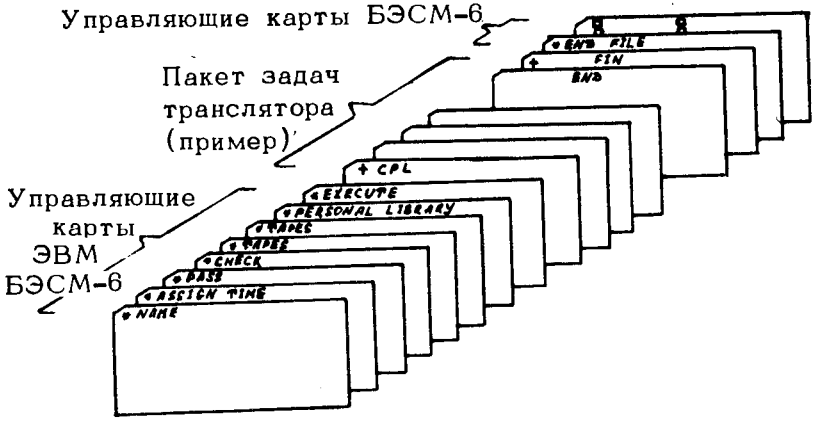
- CO - отсутствует декларативный оператор CORG .
- CY - дважды описан массив, либо массив с данным именем не описан ни в DIMEN , ни в COM .
- CP - несоответствие правых и левых скобок.
- CA - несоответствие количества операндов и количества знаков операций.
- CD - в адресной части команды автокода содержится неотносительное выражение, см. /9/
- CQ - в левой части оператора присваивания для обозначения индекса массива используется переменная, состоящая более чем из одной буквы.
- CF - ошибка в операторе IF ; используется знак отношения, отличный от EQ , GT, LT.
- AP - нет декларативного оператора NAM.
- AD - дважды определенная метка.
- AM - в данной программе один и тот же идентификатор применен для обозначения массива и переменной.
- AC - программа не разбивается на страницы, требуется изменить подпрограмму, содержащую команду с косвенной адресацией.
- DD - восьмеричное число в коде операции - недопустимое для машины типа "Электроника-100".
- DL - ошибка при использовании литерала.

Описанный транслятор был полностью реализован в виде фортранной программы для работы на ЭВМ БЭСМ-6 /см. *Приложение*/ и проверен на примерах. Общий объем программы в неоптимизированном варианте составляет 1228 фортранных операторов. Скорость трансляции - 11 *опер/с*. Полный текст транслятора, а также примеры программ на языке COPLAN и результаты их трансляции планируется опубликовать в отдельной работе после оптимизации транслятора.

Авторы считают своим приятным долгом поблагодарить В.В.Галактионова и А.А.Корнейчука за помощь и полезные замечания по тексту данной работы.

Приложение

Организация пакета с задачей пользователя на языке для трансляции и выдачи объектной программы на ЭВМ БЭСМ-6 в мониторной системе "Дубна" /13/.



Литература

1. *G.Ososkov. CERN/D.Ph. 11/PROG. 70-6. 1970.*
2. *PDP 8/C. Small Computer Handbook, 1972.*
3. *TPA. User"s Manual. KFKI Budapest, 11. Pf. 49.*
4. *Мини-ЭВМ /под редакцией Опперля/. Изд. "Мир", Москва, 1975.*
5. *В.В.Галактионов. Сообщение ОИЯИ, 10-9322, Дубна, 1975.*
6. *А.В.Кавченко, А.А.Карлов и др. Сообщение ОИЯИ, 11-7828, Дубна, 1974.*
7. *G.Brandon . CERN/D. Ph. 11/GB/IK, 1969.*
8. *Дж-Донован. Системное программирование. Изд. "Мир", 1975.*
9. *Г.Е.Бабаян, Г.А.Ососков. Сообщение ОИЯИ, 11-9767, Дубна, 1976.*
10. *А.Флорес. Программное обеспечение. Изд. "Мир", Москва, 1971.*
11. *С.Я.Виленкин, Э.А.Трахтенгерц. Математическое обеспечение управляющих вычислительных машин. Изд. "Энергия", Москва, 1972.*
12. *В.В.Галактионов. Сообщение ОИЯИ, 10-5911, Дубна, 1971.*
13. *Г.Л.Мазный. Мониторная система "Дубна". ОИЯИ, 11-5974, Дубна, 1972.*

Рукопись поступила в издательский отдел
5 мая 1976 года.