СООБЩЕНИЯ
ОБЪЕДИНЕННОГО
ИНСТИТУТА
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ
ДУБНА

10 - 9953

Г.Е.Бабаян

ФОРТРАННЫЙ ТРАНСЛЯТОР СО СПЕЦИАЛИЗИРОВАННОГО ЯЗЫКА **COPLAN** ДЛЯ ЭВМ М-6000

1976

10 - 9953

Г.Е.Бабаян

ФОРТРАННЫЙ ТРАНСЛЯТОР СО СПЕЦИАЛИЗИРОВАННОГО ЯЗЫКА **COPLAN** ДЛЯ ЭВМ М-6000

Бабаян Г.Е. 10 - 9953

Фортранный транслятор со специализированного языка COPLAN для ЭВМ М-6000

Описан фортранный транслятор с языка COPLAN -специализированного языка, предназначенного для написания управляющих программ в машинный код ЭВМ М-6000.

Работа выполнена в Лаборатории вычислительной техники и автоматизации ОИЯИ.

Сообщение Объединенного института ядерных исследований
Дубна 1976

I. Введение

В настоящей работе содержится описание транслятора со специализированного языка *COPLAN*, предназначенного для разработки управляющих программ / I/.

Транслятор воспринимает на входе программу, написанную на языке *COPLAN*, переводит её эквивалент на язык машины М-6000, пригодный к загрузке (так называемая объектная программа). Сам транслятор как программа написан на языке ФОРТРАН, расширенном путём введения процедур для операций над кодами и символами. Это даёт возможность данный транслятор легко перенести на любую ЭВМ, в матобеспечение которой входит язык ФОРТРАН, а также использовать всё более популярную теперь в физических центрах практику программирования для малых ЭВМ на больших ЭВМ / 2 /. Можно использовать также готовый эмулятор ЭВМ М-6000 - программу, имитирующую работу малой ЭВМ М-6000 на ЭВМ БЭСМ-6 путём интерпретации её команд /3 /.

При трансляции исходного текста программы, написанной на *COPLAN'е*, используется метод интерпретации, реализованный применением программного стека (магазина) / 4 /.

В соответствии со стадиями обработки программы, транслятор *COPLAN* - машинный код можно разделить на следующие основные части:

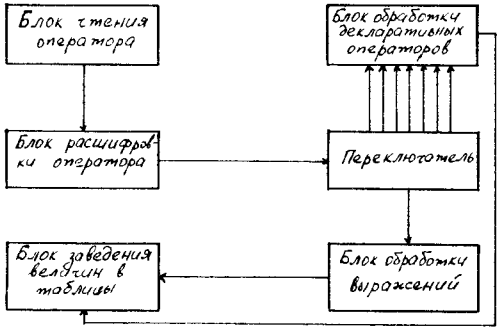
- Транслятор *COPLAN* - автокод
- Ассемблер (выполняющий перевод автокод - язык загрузки)
- Загрузчик (выполняющий перевод язык загрузки - машинный код)

Результат работы транслятора *COPLAN* - машинный код -программа на машинном языке в абсолютных адресах - выдаётся на магнит-

ную или бумажную ленту. Для исполнения программа должна быть введена с соответствующего носителя в память ЭВМ, и управление передано на первую исполнимую команду программы. Наличие эмулятора позволяет обойтись без этапа выдачи программы на внешний носитель, а исполнить её прямо на большой ЭВМ.

2. Перевод исходной программы с языка *COPLAN* в автокод.

Структура этой части, выполняющей перевод *COPLAN* - автокод, может быть описана следующей схемой:



Процесс трансляции в этой части производится за один просмотр текста на языке *COPLAN* в соответствии со следующей схемой:

чтение оператора - опознавание оператора - лексический анализ - генерация соответствующих команд на языке автокода.

Благодаря фиксированному формату карт *COPLAN*а / 1 /, грамматический разбор исходной программы сводится к выделению идентификаторов, литералов и терминальных символов (операции, ключевые слова).

Опознавание оператора выполняется блоком расшифровки, а потом с помощью переключателя производится передача управления на подпрограммы, обрабатывающие операторы языка *COPLAN* (фаза интерпретации). Каждая программа обработки, наряду с генерированием серий команд на автокоде, при трансляции данного оператора делает и синтаксический анализ этого оператора, и с нахождением ошибки выдаёт соответствующую диагностику.*)

2.1. Обработка выражений

В данное время классическим методом трансляции выражений стал метод, основанный на использовании промежуточной обратной польской записи / 4/. В обратной польской записи операнды располагаются в том же порядке, что и в исходном выражении, а знаки операций при просмотре слева направо встречаются в том порядке, в котором нужно выполнять соответствующие действия.

Пример

а) арифметическое выражение $A-B*C+D/L$

б) обратная польская запись арифметического выражения $ABC*-DL/+$.

Отсюда и вытекает основное преимущество обратной польской записи перед обычной записью выражений со скобками: выражение можно выполнить в процессе однократного просмотра слева направо. Учитывая это, для трансляции выражения предложен следующий алгоритм:

выражение переводится в обратную польскую запись с одновременной генерацией команд языка автокода.

*) Список диагностических сообщений приведён в /7/.

Для перевода выражений в обратную польскую запись в данной работе был применён алгоритм, основанный на использовании магазинной памяти - стеков для операндов и операций, названный методом двойного старшинства /5 /.

Стек представляет собой одномерный упорядоченный список элементов данных (в нашем случае - элементов языка, появляющихся при расшифровке выражения и его интерпретации), для которого операции внесения и удаления элементов, а также обращения к ним осуществляются с одного конца по принципу: последний пришедший извлекается первым.

Когда в стек вводится новый элемент, то элемент,находящийся в верхней ячейке стека,проталкивается вниз и становится вторым его элементом, тогда как новый становится верхним элементом стека. При извлечении выбирается элемент, находящийся в верхней ячейке стека, а второй элемент занимает его место, и т.д.

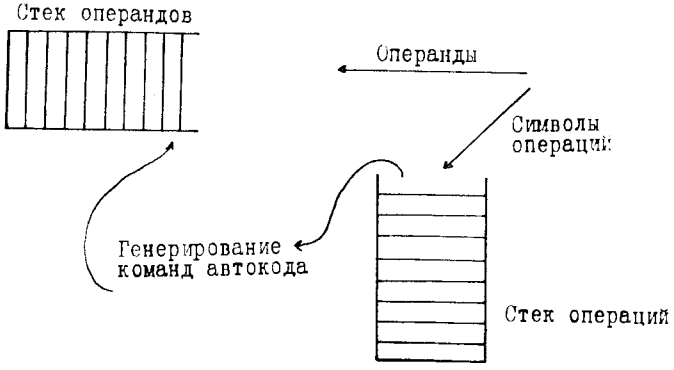


Рис. 1. Использование стека операций.

В методе двойного старшинства для операций вводится два типа приоритета: приоритет операции и сравнительный приоритет операции,

а алгоритм перевода в обратную польскую запись может быть записан в следующем виде:

Символ операции	Приоритет операции	Сравнительный приоритет
(	0	4
)	4	0
,	0	0.5
+	1	1
-	1	1
*	2	2
/	2	2

Табл. I. Приоритеты ограничителей.

0). Читается очередной символ строки - он может быть операндом, либо символом операции;

I) если очередной символ является операндом, то

I.I) он заносится в стек операндов

I.2) выполняется переход на пункт 0).

2).Очередной символ - символ операции. Если приоритет символа операции, находящийся в вершине стека операций, больше или равен сравнительному приоритету очередного символа операции, то:

2.I) символ операции, записанный в вершине стека операций, удаляется из неё и генерируется соответствующая команда (может быть сразу несколько команд) на языке автокода с нужным количеством операндов из вершины стека операндов;

2.2) если в процессе генерации команд автокода потребовалась промежуточная ячейка, то ей присваивается имя и заносится в вершину стека операндов;

2.3) выполняется переход на пункт 0);

3).Символ входной операции заносится в вершину стека операций и выполняется переход на пункт 0).

Процесс работы алгоритма для выражения $(A-F(X,Y,Z))*C-D$ может быть записан в виде следующей таблицы:

	Входной символ	Состояние стека операций	Состояние стека операндов	Генерирование команд (серий команд) автокода
1	(	(		
2	A	(	A	
3	-	(-	A	
4	F	(-	AF	
5	(	(-(	AF	
6	X	(-(	AFX	
7	,	(-(,	AFX	
8	Y	(-(,	AFXY	
9	,	(-(,,	AFXY	
10	Z	(-(,,	AFXYZ	
11	)	(-	$A\tau_1$	$\tau_1 = F(X,Y,Z)$
12	)		τ_2	$\tau_2 = A-\tau_1$
13	*	*	τ_2	
14	C	*	$\tau_2 C$	
15	-	-	τ_3	$\tau_3 = \tau_2 * C$
16	D	-	$\tau_3 D$	
17	⊥		τ_4	$\tau_4 = \tau_3 - D$

Таким образом, использование обратной польской записи позволяет строить эффективную входную программу. В фазу генерации команд автокода была включена машинно-зависимая оптимизация двух типов:

1) если какое-либо значение уже содержится в регистре, то нет необходимости загружать его вновь;

2) если мы собираемся немедленно использовать

временный операнд, который используется только в данном операторе и значение его находится в регистре, то нет необходимости записывать его значение в память.

2.2. Обработка декларативных операторов.

Язык COPLAN допускает использование как отдельных машинных команд автокода данной ЭВМ, так и целых сегментов (подпрограмм, подпрограмм-функций). Разрешается также использование основных псевдокоманд / I /.

Данный транслятор допускает использование следующих псевдокоманд автокода:

REP, BSS, DEC, OCT, DLD, DST

Использование псевдокоманд MPY, DIV, FMP, FDV, FAD, FSB можно заменить использованием соответствующих стандартных функций COPLAN'а, а основных псевдокоманд управления листингом – с помощью задания соответствующих параметров в операторе управления транслятором (см. ниже п. 6).

Остальные псевдокоманды либо имеют эквивалент в языке COPLAN (NAM, CORG, COM, EQU, EXTERN, ENTRY),либо они не используются в управляющих программах, для которых и предназначен язык COPLAN .

Декларативные операторы языка COPLAN, в список которых входят и все псевдокоманды, допустимые для данного транслятора, либо преобразуются в соответствующие таблицы, которые используются как при обработке выражений, так и при проходах ассемблера, либо транслируются без заполнения таблиц (CON, DEC, OCT, REP, DLD, DST).

Каждая строка таблицы блоков (DIMEN, COM) соответствует

одному оператору описания массива автокодного текста транслированной программы. Информация из этих таблиц используется и при обработке выражений для того, чтобы отличить имя массива от имени подпрограммы.

Строка таблицы эквивалентных имён соответствует оператору *EQU* на автокоде.

С помощью декларативных операторов *EXTERN* и *ENTRY* будет происходить связывание подпрограмм. Это делает связующий загрузчик, который описан ниже (п.5).

3. Первый проход ассемблера.

Ассемблер получает на входе исходную программу на автокоде и перерабатывает её в программу на языке машины. В зависимости от параметра оператора управления транслятором объектная программа либо оформляется в виде модуля загрузки, подлежащего дальнейшей обработке загрузчиком, либо составляется в абсолютных адресах в виде, непосредственно готовом к исполнению.

В ходе трансляции ассемблер

а) распределяет память;

б) переводит на машинный язык команды автокода и константы с учётом распределения памяти;

в) выявляет и выдаёт на печать ошибки в исходной программе;

г) формирует объектный модуль загрузки или готовую к исполнению объектную программу;

д) формирует и выдаёт печатный документ о программе.

В данной работе был использован обычный двухпроходной ассемблер / 5 /, в первом проходе которого выявляются все имена и литералы и распределяется память.

При последовательном просмотре команд каждая команда может порождать строку в одной или нескольких таблицах ассемблера. Когда команда проанализирована, ей присваивается номер, который заносится в соответствующую таблицу, если команда имеет связанную с с ней метку.

В конце первого прохода делается просмотр некоторых таблиц (подборка первого прохода), так как к этому моменту для некоторых величин память могла быть не распределена. Так же обрабатываются строки эквивалентных, внутренних и внешних имён. Блок-схема первого прохода приведена на рис. 2.

После первого прохода ассемблера, при отсутствии строк в таблице ошибок и при наличии соответствующего параметра в операторе управления транслятором, выдаются на печать строки таблицы имён с перемещаемыми или абсолютными адресами. В противном случае печатаются строки из таблицы ошибок с номером оператора и соответствующей диагностикой.

4. Второй проход ассемблера.

Назначение этого прохода - разобраться в каждой команде и заменить её эквивалентной командой машинного языка с соответствующими кодом операции, признаком косвенной адресации (если это потребуется) и адресом. К этому моменту все адреса (кроме адресов внешних имён, описанных в декларативных операторах *EXTERN*) уже распределены и поэтому могут быть заменены числовыми значениями. Для команд,в адресных частях которых встречаются внешние имена, во время второго прохода ассемблера резервируются ячейки в нулевой странице, а в адресные команды записываются адреса ячеек ну-

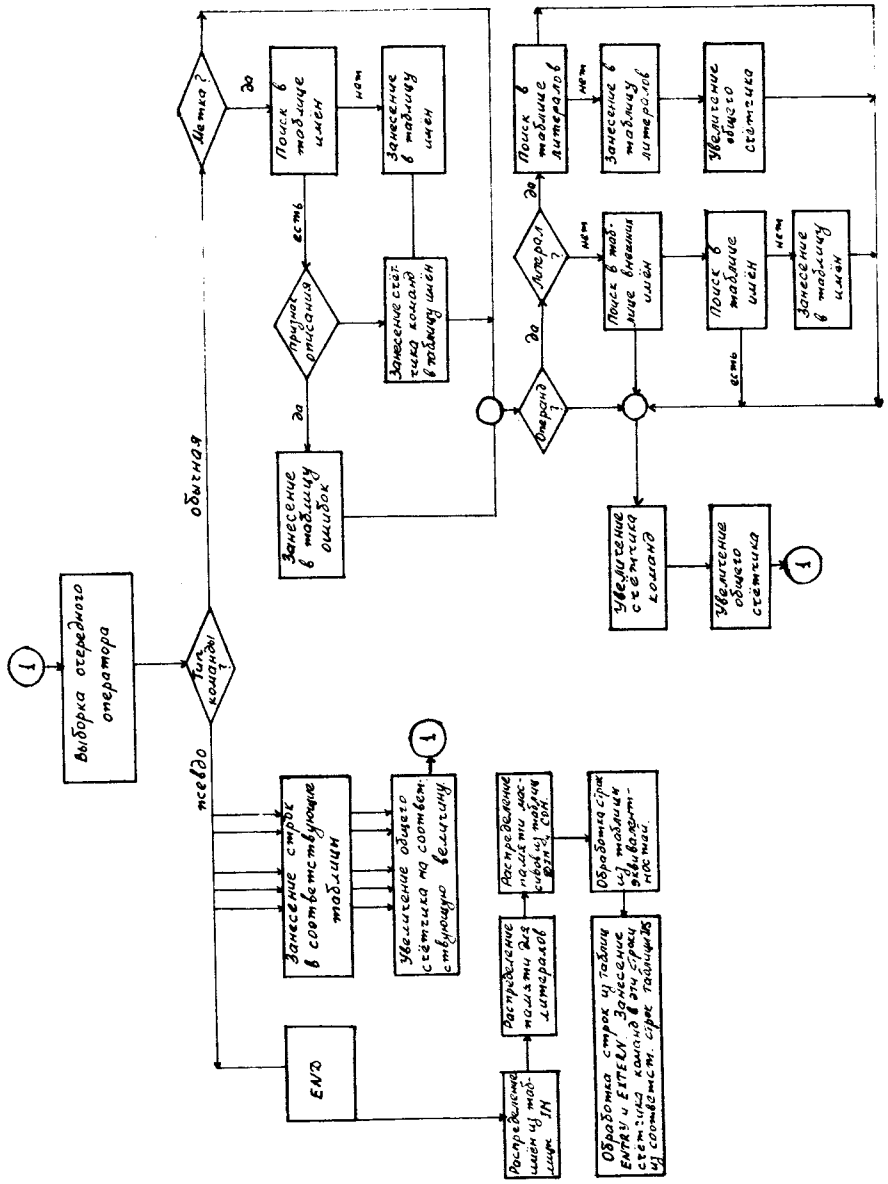


Рис. 2. Блок-схема первого прохода ассемблера.

левой страницы с признаком косвенной адресации. В соответствующие строки таблицы внешних имён заносятся адреса запасённых ячеек в нулевой странице. Как известно, под адресную часть команды ЭВМ М-6000 отводится 10 разрядов машинного слова, то есть непосредственно можно адресовать не более чем 1024 слов памяти (страница памяти). Обращение к ячейкам, выходящим за пределы страницы, обеспечивается во время второго прохода ассемблера.

При трансляции с языка *COPLAN* в автокод может быть генерирована также и команда с косвенной адресацией на ячейку, выходящую за пределы текущей страницы. Аппарат многоуровневой косвенной адресации ЭВМ М-6000 позволяет обрабатывать и такие команды. В обоих случаях во время второго прохода ассемблера полный 15-разрядный адрес ячеек, выходящих за пределы страницы, помещается в нулевую страницу (для первого случая без признака, для второго – с признаком косвенной адресации), а в адресную команду записывается адрес этой ячейки с признаком косвенной адресации.

Таким образом, полностью обеспечивается автоматическое распределение памяти при трансляции программ с языка *COPLAN*.

Задачей второго прохода ассемблера является также формирование листинга, обеспечивающего документирование и отладку программы.

Исходная и объектная программы представляются в листинге последовательностью строк, каждая из которых состоит из символической части (оператор исходной программы) и объектной (машинный код), представляющей собой соответствующую команду (константу) на машинном языке. Блок-схема второго прохода ассемблера приведена на рис. 3.

В начале второго просмотра ассемблера выполняются некоторые восстановления. Например, счётчик команд и общий счётчик должны быть установлены в нулевое или первоначальное состояние,

заданное командой *CORG / I /*, в зависимости от параметра оператора управления транслятором.

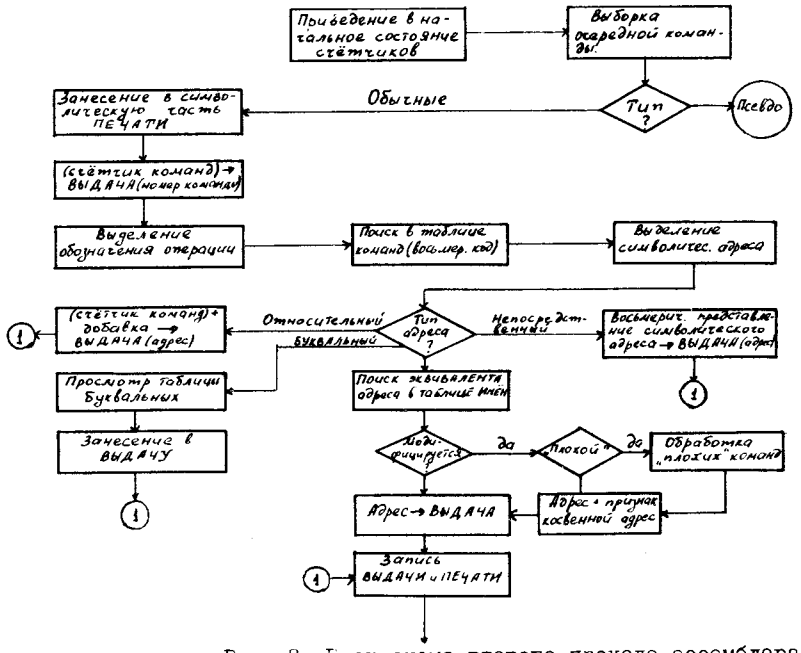


Рис. 3. Блок-схема второго прохода ассемблера.

Псевдокоманды не обрабатываются, поскольку содержащаяся в них информация перенесена в соответствующие таблицы при первом просмотре.

5. Загрузчик.

Информация в таблицах *EXTERN* и *ENTRY*, наличие косвенной адресации и нулевой страницы, на которую можно ссылаться из любого места памяти, используется загрузчиком для связывания подпрограмм *COPLAN'а*, а также для окончательного составления программ в абсолютной форме, готовой для непосредственного выполнения на машине.

После второго просмотра ассемблера в нулевой странице памяти зарезервированы ячейки для всех внешних имён всех подпрограмм, а в самих командах, использующих внешние имена, уже организованы ссылки (путём косвенной организации адресации) на соответствующие ячейки нулевой страницы. Загрузчик должен теперь занести в эти ячейки значения из таблицы внутренних имён.

Для программ, которые уже были транслированы сразу же в абсолютную форму, на этом роль загрузчика кончается. Для перемещаемых программ загрузчик должен ещё настроить соответствующим образом все перемещаемые адреса. Это делается путём прибавления к каждому перемещаемому адресу величины, равной 2000$_8$.

6. Директивы транслятора.

Директивы транслятора служат для управления транслятором. В первой колонке перфокарт этих операторов пробивается знак „+".

Имеется два оператора-директив транслятора:

+CPL $\mu_1, \mu_2, \ldots, \mu_5$

+ FIN

Первый из них - управляющий оператор транслятора - управляет различными режимами работ транслятора. Код *CPL* пробивается с десятой колонки, а параметры $\mu_1, \mu_2, \ldots, \mu_5$ с 20-й колонки перфокарт. Количество параметров может быть два и более, внутри параметров пробелы не допускаются. Параметры могут комбинироваться любым образом, но в первой позиции должен стоять либо *A*, либо *R*.

Смысл параметров следующий:

R – перемещаемая рабочая программа загружается загрузчиком, начиная с ячейки 2000

A – абсолютная форма; адреса, выделенные транслятором, являются абсолютными. Транслятор загружает программу с ячейки, указанной в адресной части декларативного оператора CORG .

B – выдаётся на печать и перфорируется колода перфокарт, состоящая из двух колонок восьмеричных цифр: адреса ячеек памяти и их содержимое.

C – выдаётся листинг на COPLANe .

L – выдаётся листинг ассемблера.

T – выдаётся на печать таблица идентификаторов.

+ FIN служит концом для транслятора, то есть он прекращает работу транслятора и пробивается с 20-й колонки на перфокарте.

7. Сообщения об ошибках.

Транслятор находит определённые в исходной программе ошибки и выдаёт двухбуквенное обозначение ошибки с порядковым её номером и номером перфокарты, если ошибка обнаружена при переводе во внутренний язык, или номером оператора автокода во время первого или второго прохода ассемблера.

Диагностические слова имеют вид СХ, если ошибка относится к работе транслятора COPLAN– автокод, АХ – к работе первого прохода и ДХ – к работе второго прохода ассемблера.

CS – ошибка в управляющем операторе; отсутствует знак „+" в первой позиции; отсутствует сам оператор; имеется недопустимый параметр; при наличии параметра A отсутствует декларативный оператор CORG .

CУ – дважды описанный массив, либо массив с данным именем не описан ни в DIMEN , ни в COM .

CP – несоответствие левых и правых скобок в выражении.

CA – несоответствие количества операндов с количеством знака операции.

CД – в адресной части команды автокода содержится неотносительное выражение.

CQ – в левой части оператора присваивания содержится неоднобуквенное переменное, обозначающее индекс массива.

CF – ошибка в операторе IF; используется знак отношения, отличный от EQ , GT , LT .

AP – нет декларативного оператора NAM .

AF – после команды END – конца подпрограммы отсутствует либо SUBR XXXX, либо FUNCT XXXX, либо + FIN .

AД – дважды определённая метка.

AM – в данной подпрограмме один и тот же идентификатор применён для обозначения массива и переменной.

ДД – в коде операции восьмеричное число – недопустимое для машины М-6000.

ДL – ошибка при использовании литерала.

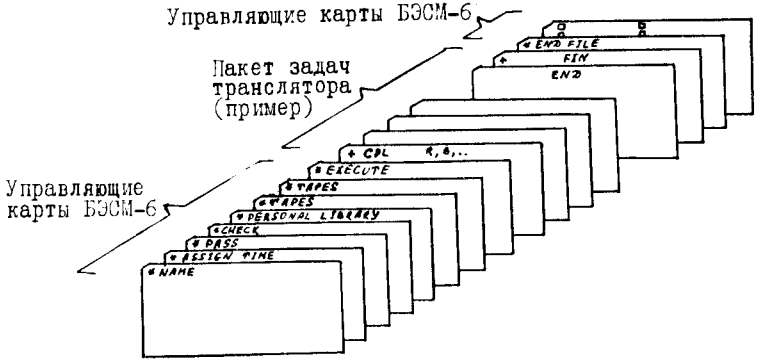
Описанный транслятор был полностью реализован в виде фортранной программы для работы на ЭВМ БЭСМ-6 (см. Приложение) и проверен на ряде примеров. Общий объём программы в неоптимизированном варианте составляет 1145 фортранных операторов. Скорость трансляции – 15 опер/с. За недостатком места в настоящем сообщении не приводится полный текст транслятора, а также примеры программ на языке

COPLAN и результатов их трансляции. Это планируется сделать в виде отдельной работы после оптимизации транслятора.

Автор считает своим приятным долгом поблагодарить А.А. Корнейчука за полезные замечания по тексту данной работы.

Приложение I.

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



Литература:

1. Бабаян Г.Е., Ососков Г.А. Сообщение ОИЯИ 10-9768 , Дубна, 1976г.
2. Мини-ЭВМ (под редакцией Опперля). Изд-во "Мир", Москва, 1975 г.
3. Жавченко А.В., Карлов А.А., и др. Сообщение ОИЯИ 11-7328, Дубна, 1974г.
4. Флорес А. "Программное обеспечение". Изд-во "Мир", Москва, 1971г.
5. Виленкин С.Я., Трахтенгерц Э.А. "Математическое обеспечение управляющих вычислительных машин". Изд-во "Энергия", Москва, 1972г.
6. Мазный Г.А. Мониторная система "Дубна", 11-5974, Дубна, 1972г.

Рукопись поступила в издательский отдел
8 июля 1976 года.