Serdica
Bulgariacae mathematicae publicationes

Сердика
Българско математическо списание

ТРАНСЛЯТОР ФОР32 С ЯЗЫКА ФОРТРАН ДЛЯ ЭВМ „МИНСК-32“

П. Х. БЪРНЕВ, М. Р. БЪРНЕВА, Д. Д. ДОБРЕВ, В. Т. ТОМОВ

Рассматривается транслятор с алгоритмического языка ФОРТРАН для ЭВМ „Минск-32“. Проводится анализ особенностей используемой машины и соответствующей операционной системы. Обсуждаются решения, принятые при проектировании входного языка и транслятора.

Описывается общая структура транслятора и некоторые характеристики его реализации.

Сведения о машине „Минск-32“ и ее операционной системе. Основной комплект ЭВМ „Минск-32“ [1, с. 6—31; 2, с. 4—31] включает оперативную память (ОП) 32К ячеек, устройство ввода с перфокарт, устройство ввода с перфолент, печатающее устройство вывода на перфокарты, два устройства вывода на перфоленту, внешнюю память (ВП) из пяти накопителей на магнитной ленте (МЛ) и пультовую пишущую машинку.

Ячейка содержит 37 разрядов, в которых записывается двухадресная команда, число или пяти семиразрядных символов. Цикл обращения к ОП — 5μs. Через базисы и относительные адреса доступны одновременно четыре поля ОП — по 2К каждое.

Так как в машине нет индексных регистров, индексация [1, с. 60—62] производится при помощи ячеек ОП, применяемых как индексные. В одной команде можно употреблять только одну индексную ячейку. Одновременно в программе доступны 15 индексных ячеек.

Система команд для работы с символами развита недостаточно.

Обмен информацией с периферийными устройствами [1, с. 135—148] проводится через один мультиплексный и один селекторный канал. Применяется код ГОСТ — 10859-64. Обмен с накопителями на магнитных лентах осуществляется последовательным доступом только в одном направлении. Повторная запись на магнитной ленте делает недоступной информацию, находящуюся после новой записи. Не существует аппаратной возможности управления печатью по страницам.

Операционная система состоит из управляющей программы „Диспетчер“ [1, с. 209—231; 2 с. 17—31] с системой для программирования [1, с. 182—208].

Диспетчер занимает постоянно для своих целей 6,5К ячеек из ОП и два накопителя на магнитных лентах. Принципиально он допускает параллельную работу четырех программ, но вследствие ограниченных возможностей параллельного обмена информацией с периферийными устройствами, и обстоятельства, что закрепление и освобождение устройств предоставлено программистам, на практике применяется, главным образом, однопрограммный режим работы.

СЕРДИКА. Българско математическо списание. Том 2, 1976, с. 12—28

Система „Диспетчер" выдает абсолютный адрес команды если при ее выполнении произойдет аварийная ситуация. В системе не предусмотрены средства, позволяющие следить за правильным взаимодействием программных частей.

В период проектирования транслятора ФОР32 (начало 1972 г.) система программирования включала: язык символического кодирования (ЯСК) и соответствующий транслятор (ТСКМЛ) [1, с. 46—181]; язык КОБОЛ и соответствующий транслятор (ТК32) [3]; общую организацию ввода, хранения и корректирования программ на магнитных лентах и соответствующие сервизные программы; общий язык для полученных в результате трансляции программ (язык загрузки) и сервизные программы для объединения программных частей и для загрузки программ в ОП; программы для отладки программ на уровне машинного языка; небольшую библиотеку сервизных программ.

Позднее в СССР были разработаны транслятор ТФ1 с ФОРТРАНа [4] и трансляторы с языка АЛГАМС.

2. Основные решения при проектировании и общая структура транслятора. Проект транслятора ФОР32 основан на принципах, упомянутых в [5], на особенностях „Минск-32" (аппаратуры и операционной системы) и на некоторых специальных условиях.

В соответствии с принципом о комплексном подходе, проектирование транслятора и языка программирования было сделано одновременно, т. е. была проектирована система программирования, состоящая из языка и соответствующего транслятора.

При проектировании были приняты во внимание следующие условия:

(У1) Система программирования должна быть основана на языке ФОРТРАН и должна включать единственный транслятор;

(У2) Не вносить изменения в структуру и операционную систему „Минск-32".

(У3) Технология эксплуатации транслятора ФОР32 должна соответствовать воспринятой технологии при работе с Диспетчером [1, с. 209 — 231, 2 с. 78—98] и с созданными уже трансляторами ТК32 и особенно ТСКМЛ.

Соображения (У2) и (У3) связаны с обстоятельством, что предназначение транслятора ФОР32 — дополнить систему математического обеспечения „Минск-32".

При проектировании были приняты следующие основные решения.

(Р1) Язык должен содержать основные выразительные средства ФОРТРАНа IV, но не должен быть перегружен редко используемыми конструкциями, которые могут быть заменены другими.

(Р2) Язык должен включать средства для обмена информацией со всеми наличными периферийными устройствами.

(Р3) Деятельность по отладке программ производится полностью средствами языка.

(Р4) Транслятор должен быть оформлен в виде библиотечной программы в системе „Диспетчер" и должен транслировать независимо части программы на ФОРТРАНе (подпрограммы, функции и основную программу).

(Р5) Каждая программная часть предварительно должна быть записана на МЛ. Первоначальная запись и внесение коррекций в нее должны

производиться при помощи сервизных программ, используемых при работе с ТСКМЛ.

(Р6) Транслированные программные части должны быть оформлены согласно правилам для библиотечных программ в ДИСПЕТЧЕРе и должны быть записаны на языке загрузки на МЛ.

(Р7) Передача дополнительной информации транслятору и манипуляции с пульта во время его работы производятся как и в ТСКМЛ.

(Р8) Распределение оперативной памяти при трансляции должно осуществляться так, чтобы при наличии максимально возможного объема памяти она была использована наиболее эффективно.

(Р9) При трансляции внешняя память должна использоваться как можно реже.

(Р10) Не должен использоваться ЯСК, как промежуточный язык при трансляции.

(Р11) Реализация операторов для обмена информацией, связи между программными частями и обработка аварийных ситуаций должны производиться во время выполнения путем интерпретации.

(Р12) Оптимизация в транслируемых программах должна производиться на уровне машинного языка, преимущественно в связи с использованием сумматора и с особенностями при индексации.

Ниже с целью пояснения комментируются некоторые из принятых решений. Большая часть из них вытекают из принципов, указанных в [5], из особенностей „Минск-32“ и условий (У1), (У2) и (У3). Номера комментариев соответствуют номерам решений (Р), к которым они относятся.

(К1) Включение в язык не особенно необходимых средств увеличило бы объем транслятора, в результате чего ухудшатся общие эксплуатационные характеристики системы „язык-транслятор“.

(К5) Условие, при котором транслируемая программа предварительно должна быть записана на МЛ, соответствует воспринятой практике в операционной системе Минск-32. По этой же причине для идентификации каждой строки программы служат первые ее 11 символов.

(К6) Из обстоятельства, что транслированные программы являются библиотечными (в смысле операционной системы) проистекают три важные следствия:

— каждая транслированная программная часть целиком умещается в ОП;

— транслированные транслятором ФОР32 подпрограммы и функции могут быть использованы программами, написанными на других языках;

— в программах на ФОРТРАНе могут использоваться программные части, написанные на других языках и транслированные соответствующими трансляторами на язык загрузки.

В двух последних случаях можно выполнять обмен информацией и с помощью общей области.

(К7) Транслятор ФОР32 работает как в однопрограммном, так и в многопрограммном режиме.

(К8) Эффективность использования оперативной памяти при трансляции определяется объемом транслируемой программы соответствующим единице предоставленной памяти.

а) большая эффективность при минимальной памяти (насколько ее реализация возможна) позволяет транслятору переводить более крупные программные части;

б) при предоставлении части оперативной памяти во время трансляции эвентуальный ее недостаток потребует повторной трансляции при большей памяти, но не потребуется сегментирование программы.

в) при использовании транслятора в однопрограммном режиме, который типичен для ДИСПЕТЧЕРА, используется вся оперативная память.

(К9) Решение обуславливается неудобствами внешней памяти: последовательный доступ в одном направлении, небольшое количество располагаемых лентопротяжек и небольшая скорость.

(К10) Транслятор ТСКМЛ работает в 6 последовательных фазах и занимает значительную память и время. ЯСК не используется в качестве промежуточного языка в трансляторах, созданных в СССР.

(К11) Интерпретация ведет к экономии оперативной памяти при выполнении сложных программ за счет некоторого перерасхода при выполнении программ, которые требуют небольшой памяти.

Интерпретация производится для операций обмена информацией и связи между программными частями, так как особенно важен контроль во время исполнения, а, с другой стороны, перерасход времени на интерпретацию незначителен по сравнению с временем, необходимым для фактического выполнения соответствующих операций.

(К12) Оптимизация на уровне языка программирования нецелесообразна из-за его относительной простоты и обстоятелства, что транслятор должен быть сбалансирован по отношению к занимаемой памяти, скорости трансляции, точности анализа ошибок и качеству транслируемой программы.

Решения (Р1)—(Р12) дают возможность определения общей структуры системы ФОР32 и обуславливают основные направления при ее детальном проектировании.

Система ФОР32 состоит из языка программирования (рассмотренного в т. 3) и транслятора, транслирующего программы на машинный язык. Со своей стороны транслятор состоит из компилятора (рассмотренного в т. 4) и исполнительной системы — ИС (рассмотренной в т. 5).

Программы на ФОРТРАНе транслируются компилятором. Полученная транслированная программа может выполняться машиной, только при помощи ИС. ИС можно рассматривать с трех точек зрения:

а) как дополнение к машине „Минск-32", которое позволяет ей выполнять транслированную компилятором программу;

б) как продолжение компилятора, позволяющее довести до конца трансляцию программы на понятный для машины язык;

в) как стандартную группу сервизных пропрограмм, присоединенную к транслируемой программе.

Эти три способа третирования ИС изображены схематично на рисунках 1а, 1б, 1в.

3. Язык. Язык, использованный в системе ФОР32, разработан на базе ФОРТРАНа. При этом были приняты во внимание соответствующие принципы из [5], решения (Р1), (Р2) и (Р3) и результаты из проведенных в ИММ с ВЦ статистических исследований по использованию различных языковых средств при программировании на ФОРТРАНе [10].

Разработанный язык представляет значительное расширение стандартного языка BASIC FORTRAN IV [6, 7] и включает достаточно широкую подсовокупность языка FORTRAN IV [6,7], а также и некоторые дополнительные средства. Подробно этот язык описан в [8,9]. По этой причине

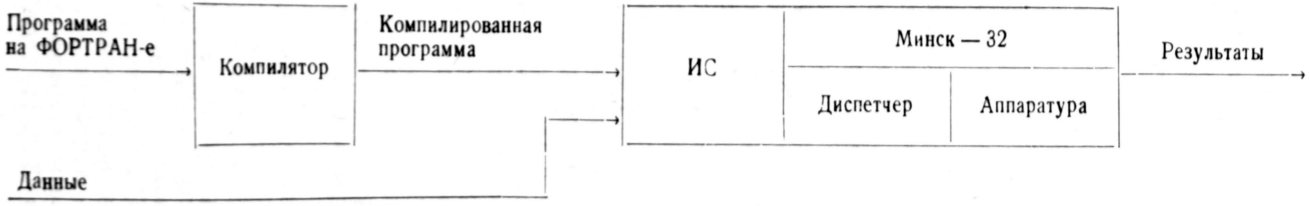


Рис. 1а

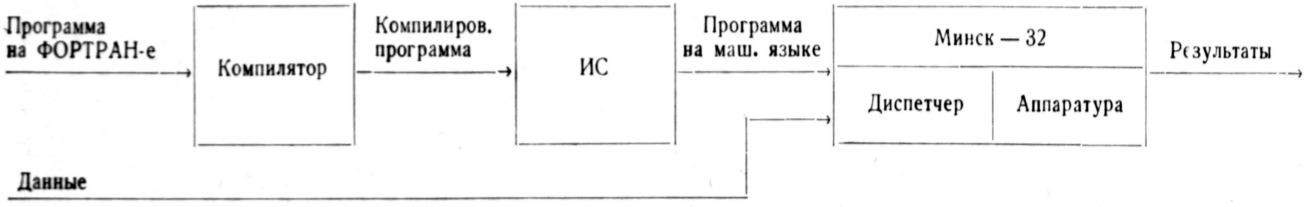


Рис. 1б

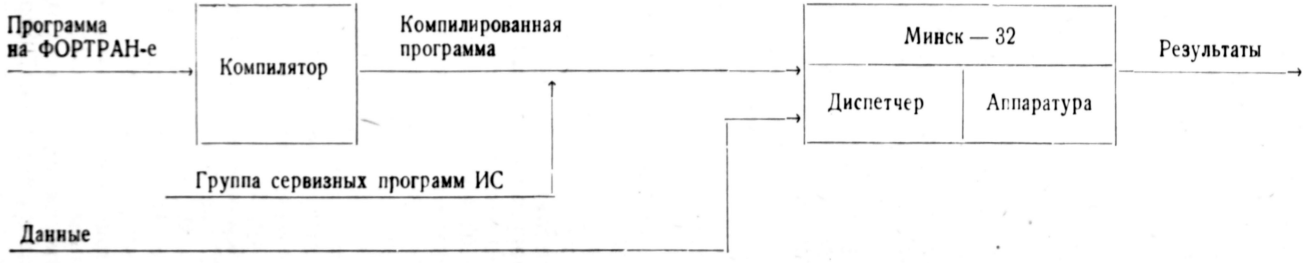


Рис. 1в

здесь он не излагается полностью, а рассматриваются только некоторые его особенности.

Язык ФОР32 в дополнение к FORTRAN IV позволяет использование:

(Я1) Разнотипных величин в арифметических выражениях и в операторе присваивания.

(Я2) Литералов, как фактических параметров.

(Я3) Расширения общей области в двух направлениях при помощи оператора ***EQUIVALENCE.***

(Я4) Оператора

LIST *A*1, *A*2,..., *Ar*,

который вызывает печать имени и значения каждой переменной *A*1, *A*2, ..., *An*, каждый раз когда ей присваивается значение.

(Я5) Букв русского альфавита (отличающихся графически от латинских букв).

(Я6) Литералов, обозначенных кавычками.

В ФОР32 не включены возможности работи с двойной точностью, с комплексными и логическими величинами. Не допускается использование именованных общих областей (имеется в виду ВП), а так же и некоторых редко используемых и легко заменимых средств, как, например: описания массивов в операторе ***COMMON***, оператор перехода вида ***GO TO*** *N* (*E1*, *E2*,..., *En*), форматная спецификация G, оператор ***DATA*** и др.

Программы ФОР32 состоят из одной основной и возможно дополнительных программных частей. Основная программная часть начинается оператором ***ROUTINE*** A, где A наименование программной части, рассматриваемой в качестве библиотечной программы на языке загрузки. Дополнительными программными частями могут быть, или библиотечные программы на языке загрузки, которые используются в качестве подпрограмм и функций.

При выполнении программы первоначально в ОП размещается основная программная часть и функции, которые она использует. Вызываемые подпрограммы размещаются в ОП вместе с функциями, использованными в них при каждом обращении.

Перед выполнением программы, группы из ее программных частей (или все программные части) могут быть объединены с тем, чтобы при выполнении в ОП размещалась сразу вся группа.

Распределение позиций в строке бланка производится по принятому. в заводской системе способу программирования (I) или по традиционному методу (II). Во втором случае программным путем производится преобразование в первый вид. Два распределения имеют следующий вид:

	I	II
поле для идентификации	1—11	73—80
поле для метки	12—16	1— 5
символ переноса	16	6
поле для оператора	18—80	7—69
символ комментария	12	1

В ФОР32 наименования величин могут содержать произвольное количество символов, но во внимание принимаются первые пять символов

В программах можно использовать два допустимых в коде ГОСТ 10859-64 интервала.

В операторах ***FORMAT*** после окончания списка форматных спецификаций, повторение производится с самой левой открывающей скобки. Символ "/" в этих операторах не играет роли разделителя. Управление печатью происходит по общим правилам для других устройств. Переход на новую страницу и оформление страницы производится автоматически. Вынужденный переход на новую страницу задается с помощью спецификации 1H1.

Допускается использование перфоленты в качестве носителя программ и информации ввода и вывода.

Накладываемые ограничения в системе ФОР32 и индикации, выдаваемые транслятором, являются неотделимой частью языка.

В связи с оптимальным использованием ОП, ограничения по возможности имеют суммарный характер, т. е. они относятся к группам различных языковых средств. Исключения из этого правила связаны с конструкциями, которые используют незначительный объем ОП, из-за чего на них могут быть наложены твердые, но не сильные ограничения.

Одно из основных ограничений требует, чтобы сумма числа простых переменных и констант и 6-кратное число массивов в одной программной части не превышало определенной константы, равной 3500 в данной реализации.

Слабые ограничения накладываются, например, на глубину вложения циклов (до 20), на количество параметров подпрограмм (до 32) и др.

Во время компиляции программа анализируется с целью установления ошибок, проистекающих не только из-за нарушения формальных и синтаксических правил и ограничений, но и ошибок вида: отсутствие метки после операторов ***GO TO*** и ***IF***, наличие метки перед операторами ***DIMENSION***, ***COMMON*** и т. д.

Для каждого оператора, содержащего ошибки, компилятор печатает индикацию, содержащую номер страницы и номер строки в ней, с которой начинается оператор и список номеров ошибок с соответствующими короткими объяснительными текстами.

Оператор ***LIST***, включенный в язык в соответствии с решением (Р3) предназначен для облегчения проверки программ. Для этой же цели, особое внимание уделяется контролю и индикации ошибок во время выполнения транслированных программ.

При выполнении осуществляется анализ ошибок, связанных с операциями обмена информацией, с корректностью вводимой и выводимой информации, контролируется соответствие между фактическими и формальными параметрами при использовании подпрограмм и функций и корректность параметров в операторах цикла. Наблюдается за некорректными аргументами встроенных функций, за выходом за пределами отведенной для программы ОП (вследствии некорректных индексов), за возникновением переполнения при выполнении арифметических операций и за выполнением рекурсивных обращений. Индикация об ошибках, обнаруженных во время выполнения, определяет характер ошибки и оператор, в котором она появилась. Оператор указывается посредством его месторасположения в нетранслированной программной ча-

сти и следа, представляющего цепь обращений, при помощи которых фактически было достигнуто выполнение этой программной части.

4. Компилятор. Основной характеристикой компиляторов является число просмотров. Главные соображения в пользу многопросмотровой компиляции следующие:

— уменьшается объем оперативной памяти, занятой в данный момент программой, реализирующей компиляцию, так как она делится на части;

— увеличивается возможность анализа программы с целью ее оптимизации и обнаружении ошибок;

— создается возможность для распределения работы по реализации компилятора между большим числом исполнителей.

Многопросмотровая компиляция ведет к увеличению в некоторой степени объема работы по реализации компилятора и требует достаточно быстрой и удобной промежуточной памяти.

Компилятор ФОР32 двухпросмотровый. При определении числа просмотров компилятора ФОР32 были приняты во внимание следующие обстоятельства:

— внешняя память медлена и неудобна;

— помещение всего компилятора в оперативной памяти очень ограничило бы объем программ. При разделении компилятора на две части он не занимает большой объем оперативной памяти. Эффект в этом отношении уменьшается при разбиении компилятора на более чем двух частей;

— проектированная оптимизация не требует многократного просмотра информации;

— исполнительная система представляет собой третий просмотр с точки зрения анализа самых существенных ошибок;

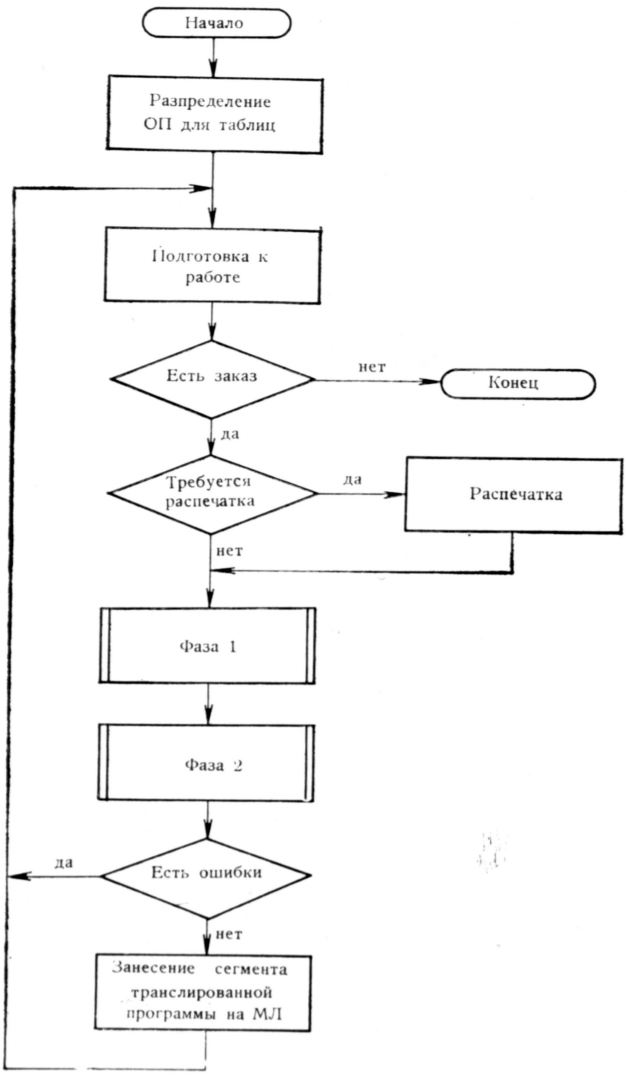
— реализацию компилятора предусматривалось поручить немногочисленному коллективу квалифицированных программистов.

При первом просмотре компиляции осуществляется перевод с ФОРТРАНа на внутренний язык, а при втором — с внутреннего языка на язык загрузки. Компилятор работает в двух фазах, которые управляются специальной программой (УП). УП резервирует часть оперативной памяти для таблиц, с помощью которых фазы обмениваются информацией, обрабатывает поступающие заказы на трансляцию и вызывает выполнение этих заказов, применяя последовательно Фазу 1 и Фазу 2 и возможно программу для выдачи распечатка.

На рис. 2 дана общая блок-схема УП. Управляющая программа присутствует все время в оперативной памяти, а Фаза 1 и Фаза 2 располагаются последовательно на одном и том же месте в ОП.

На рис. 3 представлено в общих чертах распределение оперативной памяти во время компиляции (стрелка показывает направление возрастания информации). Размеры отдельных участков даны приблизительно в листах (1 лист = 512 ячеек).

Фаза 1 делает перевод с ФОРТРАНа на внутренний язык и одновременно с этим производит синтаксический анализ и распределение памяти для величин, которые применяются в программе. Программа на внутреннем языке представляет текст, который состоит из описателей, соответствующих выполняемым оператором и ряда таблиц. Таблицы включают информацию, содержащуюся в выполняемых операторах, а так же ин-



Фиг. 1

формаций об исходной программе, полученную во время трансляции. Точнее, внутренний язык включает следующие таблицы:

(Т1) Таблица местоположения в ОП других таблиц внутреннего языка.

(Т2) Таблица, представляющая „заголовок“ исходной программы.

(Т3) Таблица переменных и констант.

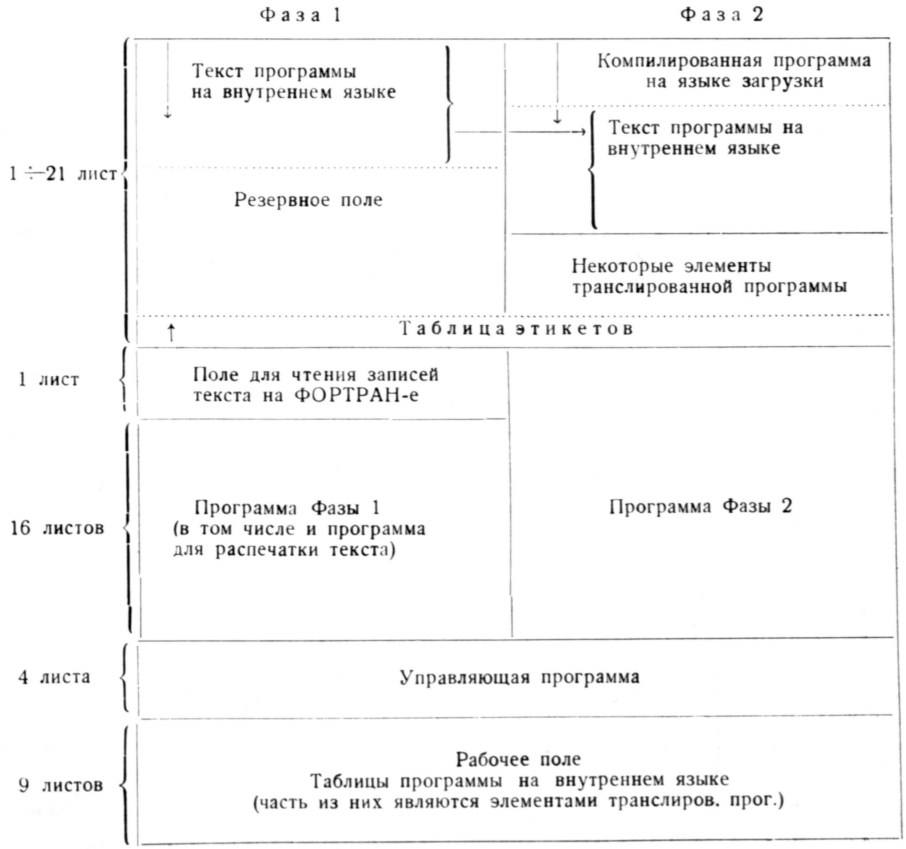


Рис. 3

(Т4) Таблица информационных векторов массивов.

(Т5) Таблица переменных, которые описаны в операторах ***COMMON*** и ***EQUIVALENCE***.

(Т6) Таблица переменных, которые описаны в операторе ***LIST***.

(Т7) Таблица меток перед оператором ***FORMAT***.

(Т8) Таблица форматных спецификаций.

(Т9) Таблица спецификаций „Холлерит“ и литерал.

(Т10) Таблица списков в операторах ***READ*** и ***WRITE***.

(Т11) Таблица меток.

(Т12) Таблица встроенных функций и использованных арифметических функций, подпрограммы функций.

(Т13) Таблица формальных параметров.

(Т14) Таблица местоположения (страница и строка в исходном тексте) операторов программы.

Во время выполнения Фазы 1 просматривается (только один раз) исходная программа на ФОРТРАНе. Она читается с магнитной ленты порциями, содержащими по 1920 символов. Операторы разграничиваются между собой и определяется их вид.

Следится о наличии заглавного и конечного оператора и о соблюдении принятой в языке последовательности: заглавный оператор, операторы описания величин, операторы описания арифметических функций, выполняемые операторы, конечный оператор.

Производится синтаксический контроль описания величин и распределяется память для массивов и переменных, которые описаны в операторах.

Для каждого распознанного выполняемого оператора или оператора описания функций выполняется синтаксический анализ и формируется индикация об обнаруженных ошибках (индикация, возможно дополненная и уточненная, печатается при выполнении Фазы 2). При этом оператор заменяется очередным описателем в тексте программы на внутреннем языке.

Во время перевода выполняемых операторов распределяется память для встречающихся в них простых переменных и констант.

Для обеспечения быстрого доступа и ввиду обстоятельства, что однобуквенные идентификаторы встречаются чаще всех остальных, таблица простых паременных организуется в виде набора списков. Каждый список включает имена переменных, начинающихся одной и той же буквой. При этом первый элемент одного списка кодируется в единственной ячейке и поэтому первый символ имени отбрасывается, а остальные кодируются в специальном внутреннем представлении для букв и цифр при помощи 6 битов (в коде ГОСТ они представляются 7-битовыми символами).

Фаза 1 продолжает свою работу при обнаружении ошибки кроме случая, когда некоторая из таблиц на внутреннем языке заполнена до конца или когда число ошибок превышает 150.

Фаза 1 использует ряд вспомагательных блоков. Среди них:

— блок чтения очередного символа с фильтрацией интервалов;
— бкок синтеза и определения типа идентификатора;
— блок обработки числа (целого или реального);
— блок сегментации операторов;
— блок определения вида оператора;
— блок обработки арифметического выражения;
— блок определения характеристик данного объекта, представленного идентификатором.

Дальнейший перевод программы осуществляется Фазой 2. Результатом ее работы является программа на языке загрузки, подготовленная для записи на МЛ. Во время работы этой фазы, печатаются индикации констатированных во время компиляции ошибок (если были обнаружены). В этом случае генерированная программа не сохраняется.

Программа на языке загрузки является библиотечной программой, которую можно выполнять только в присутствии Исполнительной системы. Она оформлена как „сегмент“ на языке загрузки и состоит из „заголовока“, пяти областей и „словаря“. (ОС ЭВМ „Минск-32“).

(Э1) „Заголовок“ содержит информацию о сегменте, необходимую „Загрузчику“ при загрузке сегмента в оперативную память. Описанная ниже внутренняя структура областей (см. рис. 4а, б, в, г, д) определяет

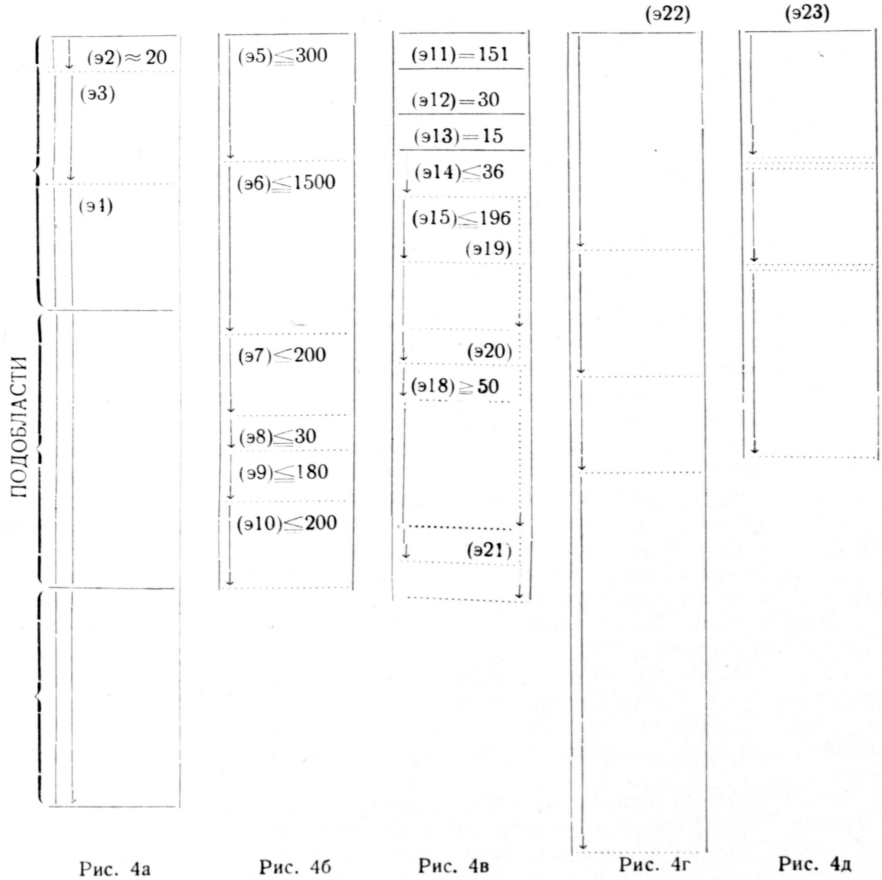


Рис. 4а Рис. 4б Рис. 4в Рис. 4г Рис. 4д

место расположения элементов переведенной программы во время ее выполнения. Там же даются ограничения длин отдельных элементов, вытекающие из воспринятого языка и метода трансляции. Область скомпилированной программы является „основной областью“. Она состоит из следующих элементов (см. рис. 4а):

(Э2) Блок инициализации программной части;

(Э3) Блоки, реализирующие арифметические функции, если есть такие;

(Э4) Программа, реализирующая выполнение операторов программной части.

Тело области делится на подобласти, не больше восьми. При этом каждая подобласть не превышает четыре листа (2048 ячеек). Подобласти компилированной программы доступны при помощи базиса „О“, и в данный момент действительна только одна из них. При необходимости передачи управления из одной подобласти команде другой подобласти, базис „О“ меняет свое значение соответствующим образом.

Область таблиц переведенной программы является „основной областью". Она состоит из следующих элементов (см. рис. 4б):

(Э5) Таблица форматных спецификаций;

(Э6) Таблица текстов из спецификаций „Холлерит" и „литерал"

(Э7) Таблица списков в операторах ***READ*** и ***WRITE***;

(Э8) Таблица переменных в операторах ***LIST***;

(Э9) Таблица формальных параметров арифметических функций;

(Э10)Таблица характеристик параметров в обращениях к подпрограммам и функциям.

Наличие этих таблиц в переведенной программе не обязательно и зависит от конкретного вида исходного текста. Во время выполнения программы доступ к таблицам осуществляется путем индексирования абсолютными адресами относительно базиса „3", который имеет значение 0.

Область с постоянным доступом является „основной областью". Она состоит из следующих элементов (см. рис. 4в):

(Э11) Поле взаимодействия скомпилированной программы с блоками Исполнительной системы.

(Э12) Таблица расположения в оперативной памяти Исполнительной системы и элементов переведенной программы.

(Э13) Группа констант общего предназначения.

(Э14) Таблица эталонных характеристик, соответствующих виду программной части и ее формальным параметрам.

(Э15) Таблица описаний используемых подпрограмм и функций (включая тех, чьи имена являются формальными параметрами).

(Э16) Таблица информационных векторов массивов (включая массивов формальных параметров).

(Э17) Таблица абсолютных адресов простых переменных, принадлежащих рабочей и общей области.

(Э18) Таблица простых переменных и констант, используемых в исходной программе (включая переменных, являющихся формальными параметрами).

(Э19) Таблица вспомагательных переменных и констант, участвующих в блоках, реализирующих арифметические функции.

(Э20) Таблица вспомагательных переменных и констант, участвующих при реализации циклов и арифметических выражений, выполняемых операторов.

(Э21) Таблица переходов осуществляющих передач управления между подобластями скомпилированной программы.

Наличие этих таблиц, за исключением (Э11), (Э12), (Э13) и (Э14) не обязательно. Оно зависит от конкретного вида исходного текста. Таблицы (Э19), (Э20) и (Э21) рассредоточены в свободных элементах таблиц (Э16), (Э17) и (Э18). На рис. 4 в это обозначено пунктирными стрелками.

Суммарная длина элементов этой области не должна превышать 8 листов (4096 ячеек). Во время выполнения переведенной программы к таблицам этой области всегда возможен прямой доступ при помощи базисов „1" и „2".

(Э22) Область массивов является „рабочей областью". Она состоит (см. рис. 4г) не более чем из 30 полей для описанных в программной части массивов (исключая те массивы, которые определены оператором ***COMMON*** или являются формальными параметрами) в таком порядке, в

каком они встречаются в операторах ***DIMENSION***, с учетом и операторов ***EQUIVALENCE***. Этой области принадлежат и простые переменные эквивалентные элементам массивов. Длина области вычисляется как сумма длин включенных в ней массивов. В случае отсутствия массивов, область состоит из одной фиктивной ячейки. Во время выполнения программы, элементы области доступны путем индексирования абсолютными адресами относительно базиса „3" или средствами специального базирования.

(Э23) Область величин, описаны в операторах ***COMMON***, является „общей областью". Она состоит (см. рис. 4д) не более чем из 30 полей, соответствующих массивам и простым переменным, в том порядке, в котором они встречаются в операторах ***COMMON*** и имея в виду операторы ***EQUIVALENCE***.

Эта область называется ***COMMON***. Ее длина равняется сумме длин, включенных в ней величин.

Если в программе не используются величины общей области, эта область заменяется рабочей областью длиной в 1 ячейку. Во время выполнения программы, элементы области доступны путем индексирования абсолютными адресами относительно базиса „3" или специальным базированием.

(Э24) „Словарь" используется для запоминания таблицы соответствий между командами скомпилированной программы и операторами (страница и строка в исходном тексте), которых они реализируют. Эта таблица не присутствует в оперативной памяти во время выполнения переведенной программы. Исполнительная система вызывает ее только во время обработки и индикации аварийных ситуаций.

Часть элементов переведенной программы подготовлены в окончательном виде еще во время работы Фазы 1. В частности (Э5)=(Т8), (Э6)=(Т9), (Э7)=(Т10), (Э16)=(Т4) и (Э18)=(Т3). Для других элементов, как (Э22) и (Э23) в программе на языке загрузки не остается ничего кроме информации об их длине, подсчитанной Фазой 1 и указанной в (Т2).

Компиляцию предшествует формирование элементов: (Э8) — путем конкретизации (Т6); (Э14) и (Э9) путем пересмотра (Т13); (Э17) — пересмотром (Т5); (Э15) и (Э10) — просмотром (Т12), при этом для (Э14) и (Э10) — только отводится место. Все эти элементы располагаются в памяти непосредственно после текста на внутреннем языке, после чего вместе с ними сдвигаются к (Т11) с целью освобождения наличной памяти, необходимой для компиляции программы (см. рис. 3).

Компиляция осуществляется однократным просмотром внутреннего текста (в котором есть ссылки почти на все таблицы на внутреннем языке), причем операторы анализируются (в том число проводится синтаксический анализ арифметических выражений) и генерируется программа (Э2), (Э3), (Э4), (Э19) и (Э20). Трансляция арифметических функций включена в общую схему компиляции. Во время компиляции заполняются элементы (Э10) и (Э12), а также (Э24) путем конкретизации (Т14). Ошибки, индицированные Фазой 1, как и ошибки, констатированные анализом, проводимым во время компиляции, печатаются в ходе просмотра текста на внутреннем языке, причем используется и (Т14).

Работа Фазы 2 заканчивается пополнением (Э14) и настройкой команд перехода по меткам, причем генерируется таблица (Э21).

Фаза 2 состоит из следующих более важных блоков:

— Блоки генерации таблиц, работающих до компиляции
— Блок инициализации компилятора
— Блок управления компилятором
— Блок анализа и генерации арифметических функций
— Блок анализа меток, предшествующих операторам
— Блок анализа и генерирования, выполняемых оператором
— Блок генерации окончания цикла
— Блок печати ошибок
— Блок настройки по меткам

В ходе работы компилятор использует следующие более важные блоки общего предназначения:

— Блок чтения описателя
— Блок записи команды
— Блок записи директивы
— Блок генерации обращения к ИС
— Блок синтаксического анализа и генерации арифметических выражений (реализует двухпросмотровую стековую обработку переходом через ПОЛИЗ)
— Блоки анализа и генерации доступа до переменных с индексами
— Блок генерации обращения к процедурам
— Блок включения прототипов команд перехода.

При компиляции арифметических выражений осуществляется оптимизация, которая выражается в следующем:

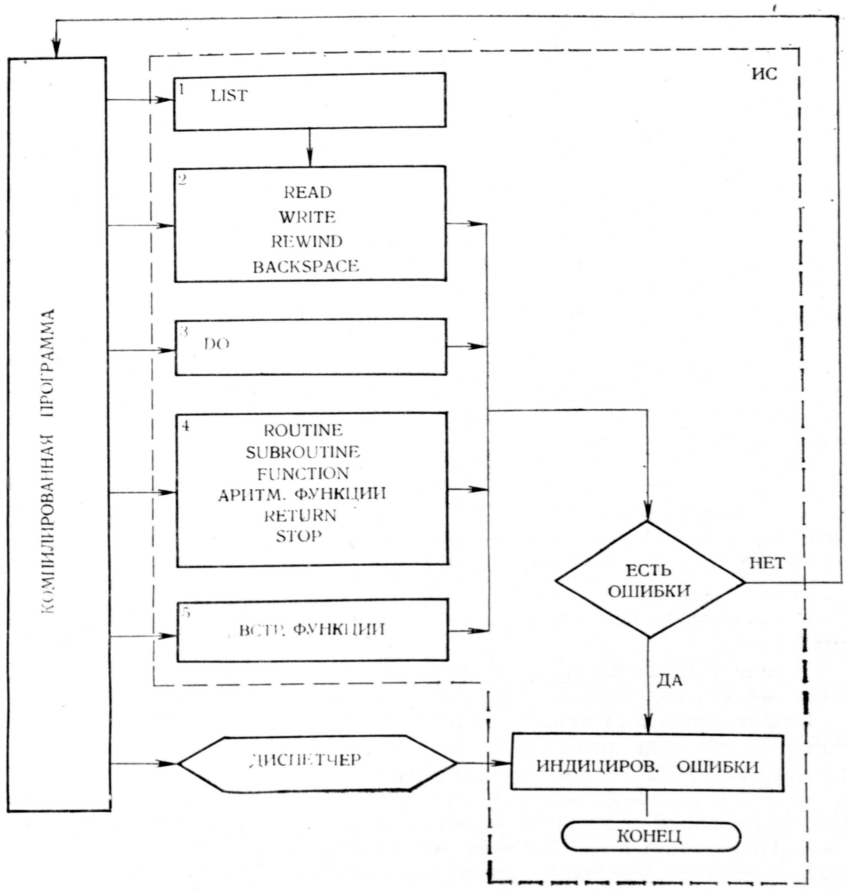
— предварительным вычислением индексов во время выполнения программы;
— использованием всех четырех разновидностей арифметических команд на основе коммутативности переводимых операций.

При анализе индексных выражений в зависимости от их конкретных особенностей, они распределяются в 25 группах. При компиляции они транслируются по разному и степень оптимизации растет с частотой использования соответствующей группы. Во всех случаях инвариантная часть индексного выражения вычисляется еще при трансляции.

5. Исполнительная система (ИС). Исполнительная система функционально является интерпретатором. Она управляется переведенной компилятором программой, в которую включены обращения к ИС.

Принципы действия Исполнительной системы иллюстрируются рис. 5. ИС состоит из пяти основных блоков (на рисунке они пронумерованы). Каждому из этих блоков передается управление из скомпилированной программы при появлении обозначенных в соответствующем блоке операторов (или обращении к функциям). Блок выполняет специфические для случая операции и возвращает управление скомпилированной программе.

Если во время работы некоторого из блоков найдена ошибка, то печатается соответствующая индикация и дальнейшее выполнение программы перерывается. Блок для печати индикации использует таблицу (Э24) и оформлен как отдельный вспомагательный сегмент, который вызывается в ОП только при возникновении сбойной ситуации. Этот блок используется и в случае, когда при выполнении скомпилированной программы выявляется ошибка, на которую реагирует непосредственно Диспетчер (переполнение, опыт использования недопустимых участков ОП путем неправильной индексации, при косвенной рекурсии).



Фиг. 5

Блок 1 действует при каждом присваивании значения переменной, описанной в операторе ***LIST***. Необходимый перевод и редактирование в связи с печатью имени и значения переменной выполняется при помощи блока обработки операторов обмена с периферийными устройствами (блок 2) предоставляя ему таблицу (Э8).

Блок 2 является программой системой, управляющей вводом-выводом информации. Блок контролирует корректность номеров периферийных устройств и соответствие между элементами списков операторов ***READ*** и ***WRITE*** и соответствующими форматными спецификациями. Блок осуществляет закрепление периферийных устройств, редактирование и перевод данных в соответствии с операторами, ***FORMAT***, передвижение МЛ и фактический обмен с периферийными устройствами.

Во время работы блок использует таблицы (Э5), (Э6), (Э7) и (Э16).

Блок 3 действует только в случаях, когда некоторые из параметров в операторе цикла являются переменными. Блок проверяет корректность фактического значения параметров и подсчитывает число повторений тела цикла.

Блок 4 проверяет корректность обращения к подпрограммам и функциям, как и соответствие между формальными и фактическими параметрами. Блок осуществляет инициализацию ИС, загрузку функций и процедур, описанных в операторах ***EXTERNAL***, перенос значений между фактическими формальными параметрами. Блок управляет состоянием стека, в котором содержится след ведущий до выполняемой в данный момент программы. Во время работы блок использует (Э10) вызывающей программы и (Э9), (Э14), (Э15), (Э16).

Блок 5 состоит из программ реализирующих вычисления значений встроенных функций, как и операций возведения в степень и деления целых. Каждая программа имеет два входа, соответствующих аргументам двух типов — целых и реальных. Перед вычислением производится контроль аргументов, чтобы избежать переполнения.

* * *

Транслятор разработан в Институте математики и механики БАН в 1972/73 г. под общим руководством и по идейному проекту П. Бърнева. Входной язык был разработан П. Бърневым и В. Томовым. Детальный проект и реализация первой фазы и управляющей программы транслятора сделаны М. Бърневой. Детальный проект и реализация второй фазы и исполнительной системы (без операций ввода — вывода и встроенных функций) были сделаны Д. Д. Добревым. Детальный проект реализации операторов обмена информацией сделан В. Томовым.

Авторы высказывают благодарность М. Шишковой за программную реализацию части исполнительной системы, относящейся к операторам ввода-вывода. Авторы благодарят так же М. Русеву за адаптацию встроенных функций и П. Петроза за реализацию некоторых вспомогательных блоков.

ЛИТЕРАТУРА

1. Н. Т. Кушнерев, М. Е. Неменман, В. И. Цагельский. Программирование для ЭВМ „Минск 32". Москва, 1972. 6—31.
2. И. А. Белокурская, Н. Т. Кушнерев, М. Е. Неменман. Диспетчер ЭВМ „Минск 32". Москва, 1973. 4—31.
3. В. П. Кулаковская, Л. М. Романовская, Т. А Савченко, Л. С. Фельдман. Кобол ЭВМ „Минск-32". Москва, 1973.
4. Математическое обеспечение ЭВМ „Минск 32". Институт математики АН БССР, вып. 9, Минск, 1973.
5. П. Бърнев. Някои принципи при създаване на системи за програмиране на базата на алгоритмичен език. Национална конференция по ФОРТРАН (сборник доклади). София 1975
6. A programming Language for Information Processing on Automatic Data Processing Systems. *Communications of the ACM*, 7, 1964 591—625.
7. American Standard FORTRAN. X. 3—9—1966.
8. ФОР32. Транслатор от ФОРТРАН за машината „Минск 32". Ръководство за програмиста — IV редакция. София, 1973.
9. ФОР32 Изменения и допълнения към IV редакция на ръководството за програмиста. София, 1974.
10. П. Бърнев, В. Томов. Емпирично разпределение на видовете оператори и форматни спецификации в програми на ФОРТРАН. Национална конференция по ФОРТРАН (сборник доклади). София, 1975.

Единый центр науки и подготовки кадров по математике и механике
София 1000 п. я. 373

Поступила 13. 11. 1974 г.