СООБЩЕНИЯ
ОБЪЕДИНЕННОГО
ИНСТИТУТА
ЯДЕРНЫХ
ИССЛЕДОВАНИЙ

Дубна

11 - 5923

В.А.Загинайко

КОМПИЛИРУЮЩАЯ СИСТЕМА ЭВМ БЭСМ-4

ЛАБОРАТОРИЯ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ
И АВТОМАТИЗАЦИИ

1971

В.А.Загинайко

КОМПИЛИРУЮЩАЯ СИСТЕМА ЭВМ БЭСМ-4

В настоящее время в ЛВТА ОИЯИ опробован вариант системы математического обеспечения на ЭВМ БЭСМ-4. Система включает в себя трансляторы с автокода (ассемблер), транслятор с машинно-ориентированного языка (макроассемблер), библиотечную (компилирующую) систему и ряд обслуживающих программ.

Входными языками системы являются машинно-ориентированный язык (Макрос) и автокод, близкий к автокоду ассемблер /2/. Отличия состоят в том, что в новом автокоде исключена трансляция арифметических выражений и добавлен ряд операторов, облегчающих использование автокода в рамках компилирующей системы. В частности, операторы *ident* и *end* дают возможность сегментировать задачу в автокоде на ряд библиотечных подпрограмм, транслируемых независимо, операторы *subr* и *call* дают возможность обращаться из одной библиотечной подпрограммы к другой, операторы *block* и *common* описывают массивы величин, общих для библиотечных подпрограмм.

Входной машинно-ориентированный язык Макрос совпадает с языком, описанным в /I/.

I. Взаимодействие программ системы при обработке задачи пользователя

Для счета задачи в рамках данной системы пользователь формирует пакет карт, расположенных в следующей последовательности:

1. Карта вызова задачи ввода.
2. Текст задачи в языках системы.
3. Карта вызова макроассемблера.
4. Таблица соответствий макроассемблера.

5. Карта вызова ассемблера.
6. Карта вызова компилирующей системы.
7. Подпрограммы в языке загрузки (стандартные массивы).
8. Информация к загрузчику о разделах.

При отсутствии системной ленты карты вызова заменяются соответствующими подпрограммами. Текст задачи пользователя может иметь следующие системные карты

```
*упр
*tlt
*mac
*ass
*end
```

Карта *упр предполагает, что следующая за ней информация пробита на устройстве УПП БЭСМ-4.

Карта *tlt означает, что следующая за ней информация пробита на телетайпе тем же способом, что для автокода "Ассемблер".

Карта *mac указывает, что следующая за ней программа – в языке Макрос.

Карта *ass указывает, что следующая за ней программа – в автокоде.

Карта *end означает конец информации.

При отсутствии карт система работает так, как будто задача начинается с карт *упр , *mac .

Обработка текста задачи происходит следующим образом:

I. Задача ввода записывает информацию на барабан в телетайпном коде частями по 177_8 кодов. Карты *упр ; *tlt исчезают.

2. Макроассемблер перерабатывает текст в Макросе в автокодный текст. Автокодный текст оставляется без изменения. После работы макроассемблера вся информация получается в автокоде. Карты *mac , *ass , *end исчезают.

3. Ассемблер перерабатывает текст задачи в серию стандартных массивов и пишет их в библиотеку.

4. Компилирующая система производит дозапись ранее полученных стандартных массивов, с помощью информации к загрузчику формирует рабочую программу в виде взаимодействия разделов (частей задачи) при счете.

Информация на барабане, полученная в результате работы задачи ввода и макроассемблера (транслятора с Макроса) может обрабатываться обслуживающими программами автокода Ассемблер.

2. Входной язык автокода

Входной язык автокода представляет собой расширение варианта автокода"Ассемблер" без арифметического блока. Расширение выполнено за счет добавления в язык следующих операторов:

I. *ident*
2. *end*
3. *call*
4. *subr*
5. *block*
6. *common*
7. *entry*
8. *set*
9. *data*
10. *real*
11. *text*

Операторы *ident* и *end* обозначают начало и конец отдельной библиотечной программы; метка при операторе *ident* определяет название программы и хранится в каталоге библиотеки.

Оператор *subr* указывает, что метка при этом операторе является названием библиотечной подпрограммы отличной от той, где встретился этот оператор.

Оператор *call* выполняет обращение к библиотечной подпрограмме. При этом оператор *call* , *s* ; эквивалентен следующим двум

s = *subr* ; *16* , *1+1* , *s+1* , *s* ;

Оператор *block* обозначает описание ячеек, общих для нескольких библиотечных подпрограмм. Имеет вид *l* : *block* ;

Оператор *common* является описанием части массива общих ячеек, имеет вид *l* : *common* , *n* ; . При этом комбинация операторов:

```
z : block ;
l1: common , n1 ;
l2: common , n2 ,
```

означает, что длина блока *z* равна *n1 + n2 + ····*, кроме того, *l1: equ , z ;* *l2 : equ , z+n1* и т.д.

Оператор *entry* обозначает передачу управления в библиотечную подпрограмму извне.

Оператор *data* указывает, что информация, расположенная между ним и следующим оператором *set* при загрузке рабочей программы исчезает. Она используется только в операторе рассылки данных (*set*).

Оператор вида *set , indata , ondata , ldata , cixdat ;* эквивалентен следующей процедуре на Алголе:

```
procedure set (indata, ondata, ldata, cixdat);
  integer ldata, cixdat, i, j, k;
    array indata [1: ldata * cixdat],
          ondata [1: ldata];
  begin  k:=1;
    for j:=1 step 1 until cixdat do
      for i:=1 step 1 until ldata do
        begin indata [k]:= ondata [i];
              K:=K+1;
        end
  end
```

Примечание: следует иметь в виду, что в Алголе элемент массива *a[i]* соответствует адресу в автокоде вида

$a + \langle i \rangle - 1$, где $\langle i \rangle$ - содержимое ячейки *i*.

Операторы *real* и *text* описывают десятичное число и текстовую константу.

3. Транслятор с автокода и язык загрузки

Транслятор с автокода является двухпроходным ассемблером. При первом просмотре составляется таблица меток и адресов к ним. При втором просмотре происходит программирование, выдача программы в языке загрузки (стандартного массива).

В транслятор входят следующие основные блоки (части программы):

1. блок опознавания операторов,
2. блок обработки операторов,
3. блок вычисления адреса,
4. блок формирования стандартного массива.

После окончания трансляции отдельной библиотечной подпрограммы происходит запись ее в библиотеку и выдача на перфорацию (по требованию пользователя).

Стандартный массив состоит из следующих основных частей:

1. длины групп,
2. программная часть,
3. таблица признаков,
4. таблица описаний,
5. таблица внешних идентификаторов,
6. таблица подпрограмм,
7. таблица *common*
8. таблица входов.

4. Компилирующая система

Компилирующая система состоит из программ работы с библиотекой, загрузчика и резидентной части.

Программы работы с библиотекой производят запись и считывание программ по наименованию, а также формирование каталога библиотечных программ и разделов.

Загрузчик производит формирование (загрузку) рабочей программы из стандартных массивов, имеющихся в библиотеке. Рабочая программа представлена в виде библиотеки разделов с каталогом разделов. Под разделом понимается часть рабочей программы, настроенная на определенное место памяти ЭВМ. Сегментация (разбиение) программы на разделы определяется информацией о разделах, задаваемой пользователем загрузчику в виде текста. Информация о разделах

указывает, какие из подпрограмм, вызываемых данной, являются разделами. Например, информация вида

$a(b, c(d, e))$ указывает, что подпрограммы b и c раздела a являются, также, разделами. Раздел c, в свою очередь, включает в себя разделы d и e. При загрузке разделы, подчиненные одному и тому же разделу, загружаются на одно и то же место в памяти, что дает возможность формировать рабочую программу, превышающую размеры оперативной памяти БЭСМ-4.

В процессе загрузки происходит формирование и печать таблицы загрузки, указывающей расположение начала подпрограммы или массива общих ячеек (*common*).

Если загружаемая программа является разделом, вместо нее загружается подпрограмма, вызывающая соответствующий раздел во время счета задачи через резидентную часть.

Подпрограмма вызова разделов имеет следующий вид (в автокоде):

```
name :   ident;
return : bss, 1;
         call, resid;
         00,0, alfraz, nraz;
         56, 0, return;
         end
```

Здесь *nraz* - номер вызываемого раздела, *alfraz* - адрес его начала.

После формирования библиотеки разделов с каталогом загрузчик формирует подпрограмму вызова головного раздела вместе с резидентной частью и передает на нее управление.

Резидентная часть при своей работе осуществляет вызов из библиотеки и взаимодействие разделов друг с другом.

ЛИТЕРАТУРА

I. В.А.Загинайко.
Инвариантное программирование на машины М-20, Минск-22 и БЭСМ-6.
Препринт ОИЯИ РII-3993, I968.

2. В.А.Загинайко, И.Н.Силин.
Ассемблер. БI-II-45I4, I969.

Рукопись поступила в издательский отдел
7 июля I97I года.