国防科技大学学报
JOURNAL OF NATIONAL UNIVERSITY OF DEFENSE TECHNOLOGY
第13卷第3期 1992年9月 Vol. 14 No. 3

交叉汇编器设计

张钦伍

（电子计算机系）

摘 要 本文讨论了交叉汇编器设计中的两个关键问题：先引用、后定义内部符号的处理和利用外部符号进行模间通信。提出了一种统一的处理办法——随机文件代真法。实践证明十分简洁 有效。

关键词 汇编语言，汇编器，交叉汇编器

分类号 TP313

1 交叉汇编器的作用

交叉汇编器属于工具性软件，是系统软件和通用应用软件研制和开发过程中常用的工具。在计算机系统的研制中，最初能上裸机调试的软件（常称先行软件）必须已是目标指令代码的形式。这种代码从哪里来？用交叉汇编器产生就是简便易行的一种。

设X是正在研制的机器，U是一台可用的异于X的机器。为产生X机上的先行软件的目标代码，可用U上的高级语言（如PASCAL，C等）写一交叉汇编器UXAM，它在U上运行，对X的汇编语言源程序进行汇编，直接生成可在X上运行的目标代码。于是X的先行软件只要用X的汇编语言编写就行了。

如果再写一个模拟器UXSM，在U上来模拟执行交叉汇编器UXAM产生的目标代码文件，那么U、UXAM、UXSM三者便构成了如下一个小系统（图1）.

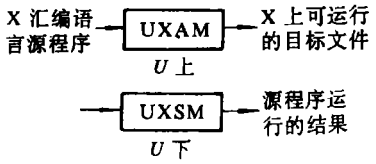

图 1

这个小系统的作用相当于一台安装了汇编器的X机。X上的任何软件，只要是用X的汇编语言编写，都可在这个小系统上调试。这个小系统与X硬件无关。这意味着X软件的调试与硬件的调试可以同时分头进行，从而大大缩短了工程的研制周期。

2 交叉汇编器的实现流程

交叉汇编器自身运行的机器与其产生的目标代码运行的机器不是同一机种，而且其目标代码是在一台暂时还无任何系统软件支持的环境下运行。这导致了作为工具软件的交叉汇编器与作为系统软件的汇编器在实现方法上有很大的差异。

目前，汇编器的汇编技术相当成熟。实现时通常采用两遍扫描，生成半目标文件的办法。在半目标

* 1991年8月26日收稿

文件一级已同高级语言的编译器生成的半目标文件一致。最后可运行程序的目标文件是通过一个单独的作业步（批处理时）或命令（交互时）对半目标文件进行装配或链接而形成的。[1]~[4]

交叉汇编器的实现比较灵活，也常受环境的制约。通常采用一遍扫描，直接生成可运行目标码的简单流程。实践证明，图 2 所示的流程是简洁高效的。

3 交叉汇编器设计中两个关键问题

交叉汇编器设计中有两个重要问题。其解决的好坏将直接影响总体方案的确定和交叉汇编器的质量。目前尚无完整统一的处理办法。两个问题是：

（1）后定义内部符号的处理

汇编语言中的表达式从语义看可分为两类：一类是表达式的值只用于指导汇编，并不用它产生目标代码。如数据生成伪指令中的重复因子，空间保留伪指令中的保留单位数。对这种表达式中的符号，在语法上要求定义出现在引用之前，否则汇编器无法继续工作。称之为先定义、后引用的表达式。另一类表达式在语法上是不能做这种要求的，如转移指令中的转移地址，访内指令中的内存地址。否则，语言本身就失去了灵活性。因此必须允许这种表达式中的符号是后定义的，即在引用之后的某个地方定义。其特点是表达式的值将被装入相应指令目标码的某个字段中，直接影响目标的运行。于是，在一遍扫描的处理中，这种后定义的内部符号如何参与表达式的计值？表达式的值如何装配？遇到该符号的定义时，其值又如何代真？这些是首先要解决的问题。

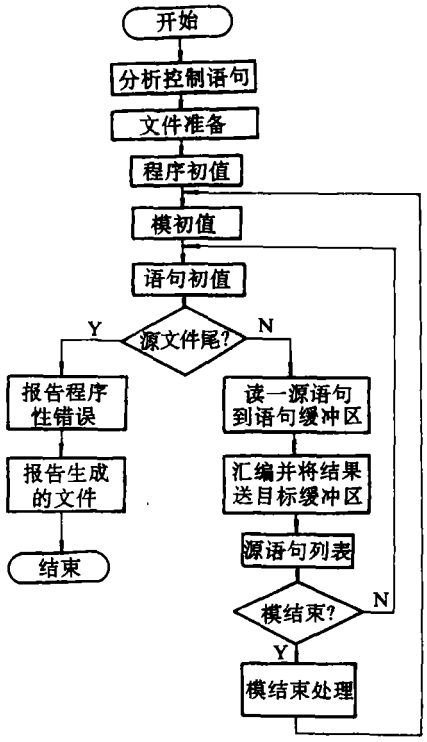

图 2 交叉汇编器流程

（2）程序模间的通信

构成源程序的各程序模之间必定有某种内在的联系，即有某种通信关系。不与其它程序模发生关系的程序模是不存在的。因此，汇编语言中模间通信机制的设置和如何处理是必须很好解决的另一问题。

目前，汇编语言中广泛使用的模间通信机制有两种：一种是公用块机制，一种是入口符号和外部符号说明机制。交叉汇编器设计中通常只实现后一种，而且只设置入口符号说明伪指令 ENTRY，并不设置外部符号说明伪指令 EXT. 这实际是一种向后全局的概念。这种不完全的机制造成了模间调用只能向前的定向性，使某些系统用户（如 OS）遇到无法克服的困难。因此，在交叉汇编器中如何完整的实现这一机制也是待解决的。

4 随机文件代真法

在此把提出的两个问题结合在一起，给出一种统一的处理办法——随机文件代真法。

第一个问题是解决表达式中出现的后定义内部符号的计值、装配和代真问题。第二个问题实质上是解决表达式中出现的外部符号的计值、装配和代真问题。两者的差别仅在于：前者是内部符号，后者是外部符号。内部符号是只能在该符号定义的模中引用的符号。后定义时，其延迟限不超过模结束。当汇

编器扫描遇到它的定义时，其引用指令的目标码还在目标缓冲区中。外部符号的引用和定义不在同一模中。后定义时，其延迟限可到源文件尾。汇编器扫描到其定义时，其引用指令所在模的目标码已经写到目标文件中去了。这一差别反映在处理过程中，就是遇到符号定义时代真的区域不同。前者是目标缓冲区，后者是目标文件。对于后定义的内部符号，需记住其在目标缓冲区中准确的引用位置；对于外部符号，需记住其在目标文件中准确的引用位置，便可代真。代真时要访问被代真指令的目标码。在目标文件中做这件事情方便吗？只要目标文件采用的是随机文件（或称直接文件），回答则是肯定的。处理过程与在目标缓冲区中的代真是完全一样的。从而可作如下合并处理。

(1) 创建目标文件时，要求随机文件的记录与目标缓冲区的元素大小相同。目的是使对两者的访问读写的单位一致。于是代真时，除了地点不同外，其它动作都相同。

(2) 在符号表的每个符号项下，引出两个引用链：一个是内部符号引用链，一个是外部符号引用链。每个引用链项记载着该符号一次引用的详细情况。这些信息都是代真时所需要的。当然，对于一个具体符号，只能结一种链。用 PASCAL 语言表示如下：

```
符号表的定义：
SYM_CHAIN = ↑SYMBOL_TABLE_ENTRY;
SYM BOL_TABLE_ENTRY =
    RECORD
      NAME: PACKED ARRAY [1…8] OF CHAR;
      VALUE: UNSIGNED;
      ATTRIBUTE: (W, P, V);
      GLOBAL: BOOLEAN;
      EXTERNAL: BOOLEAN;
      DEFINITION: BOOLEAN;
      NEXT: SYM_CHAIN;
      INNER_SYM_REFERENCE: INNER_CHAIN;
      EXT_SYM_REFERENCE: EXT_CHAIN;
    END;
```

这里的特性项，只假定符号有字、字片和值三种特性。

```
内部符号引用链的定义：
INNER_CHAIN = ↑INNER_SYMBOL_REFERENCE_ENTRY;
INNER_SYMBOL_REFERENCE_ENTRY =
  RECORD
    BEGIN_WORD: UNSIGNED;
    BEGIN_BIT: UNSIGNED;
    FIELD_WIDTH: UNSIGNED;
    EXP_SIGN: (POS, NEG);
    EXP_ATT: (E_W, E_P, E_V);
    OPERATOR: (ADD, SUB);
    NEXT: INNER_CHAIN;
  END;
```

其中，前三项记着该符号在目标缓冲区中准确的引用位置和宽度；接着的两项是表达式已求值的属性及正负，以便在代真时，把已装字段中部分表达式的值取出，进行符号扩展恢复其原来面貌。值得注意的是，每经过一次代真这两项值都可能变化，下一次代真应使用变化后的值。

外部符号引用链的定义：

```
EXT_CHAIN= ↑EXT_SYMBOL_REFERENCE_ENTRY;
EXT_SYMBOL_REFERENCE_ENTRY=
  RECORD
    BEGIN_WORD_RECORD_NUMBER: INTEGER;
    BEGIN_BIT: UNSIGNED;
      ⋮
    NEXT: EXT_CHAIN;
  END;
```

这里的第一项直接指向目标文件的一个记录。其它项与内部链相同。

(3) 遇到 ENTRY 伪指令如下处理：若符号表中已有该 ENTRY 说明的符号，就在符号表的该符号项内填上全局标志。若符号表中还未有该符号，就将该符号入表，填上全局标志和未定义标志。

(4) 只要规定 EXT 伪指令出现在被说明的符号引用之前，就能分清一个未定义的符号是内部的还是外部的。其处理与 ENTRY 类似。若符号表中已有被 EXT 说明的符号，就在符号表的该符号项中填上外部标志。否则，就将该符号入表，填上外部标志和未定义标志。

(5) 在模结束处理中，调整符号表时，应把有定义且有全局标志的符号留下，同时也要把有外部标志的符号留下，其余的符号全抹掉。

(6) 表达式计值时，遇到符号倒查符号表。若表中没有该符号，则将该符号入表，并填上非全局标志、非外部标志和未定义标志。对于尚未定义的符号，一律以零值和值特性参与表达式计值。若表达式中的符号均已有定义，则算出的值是最终的，否则算出的值只是表达式已知部分的值。无论哪种情形，都将该值装入相应目标指令的相应字段。同时，将表达式中出现的每一个非外部符号（不管有无定义）结内部引用链；将其中出现的每一个外部符号（不管有无定义）结外部引用链。

(7) 遇到符号定义时，查符号表。若表中没有该符号，则将该符号名、值及特性入表，并填上非全局、非外部、已定义标志。若符号表中已有该符号，先看是否有定义标志。若已有定义，则报重复定义错。若尚未定义，则把该符号的值和特性入表，并置上已定义标志。接着根据内部链和外部链对后定义先引用的符号进行代真。

(8) 对每个引用链项的代真工作可用两步归纳：

①根据链项中指明的准确位置和宽度把原表达式的装配值取出，并根据正负号对该值进行符号扩展，恢复该表达式值装配前的形态。

②根据操作符，把表达式值与符号值进行指定的运算，结果按原宽度截取后再送回原装配字段。

这一方法的优点是把看起来不相关的两个问题用同一办法解决，简单而通用。表达式中后定义的符号可以是多个。引用链的作用不只是为了代真，可以看到表达式中已定义的符号也结了引用链，目的是利用引用链输出交叉引用列表。这张表将提供每个符号确切的引用位置，对用户查错极为有利。

参 考 文 献

1. Cray 1 and Cray X_MP Computer Systems Cal Assembler Version 1 Reference Manual. Cary Reearch, Ins. 1983
2. Cary 1and Cray X_MP Computer Systems Macros And Opdefs ReferenceManoal. Cray Research, Inc, 1983
3. Vxamacro and Instruction Set Referevce Manual. Digital Equipment Corporation. 1984
4. Convex Assembly_Language User's Guide. Convex Computer Corporation. 1988

(下接第 50 页)

Experimental Investigation of Impulse Radar for Mitigation of Effects of Radar Absorbing Materials

He Jianguo Lu Zhongliang Su Yi
(Department of Electronic Technology)

Abstract

The response of UWB signal acting on the coating radar absorbing material (RAM) targets have been investigated experimentally by time—domain method and frequency—domain method in this paper. It is shown that the UWB signal is 10~12dB superior to the narrowband signal of conventional radar for anti—coating RAM'S targets, and it has been clearly indicated that UWB signal has indeed good capabilities for mitigation of effects of RAM.

Key words radar, radar signals, absorbing material, UWB radar, impulse response

---

(上接第 45 页)

The Design of a Cross Assembler

Zhang Qinwu
(Department of Computer Science)

Abstract

In this paper, the author disscussed two key problems in the design of a cross assembler: the processing of "define after use" internal symbols and the communication among modules by external symbols. The author also presented and unified processing method —— the random file substitute algorithm. It is proven in practice that the method is simple and efficient.

Key words assembly language , assembler, cross assembler