微机系统软件分析

朱 禹 著

北京航空航天大学出版社

微机系统软件分析

朱 禹 著

北京航空航天大学出版社

图书在版编目(CIP)数据

微机系统软件分析/朱禹编著. —北京:北京航空航天大学出版社,1995.6

ISBN 7-81012-551-6

Ⅰ.微… Ⅱ.朱… Ⅲ.微型计算机-程序系统-软件-分析 Ⅳ.TP31

中国版本图书馆 CIP 数据核字(95)第 03789 号

内 容 简 介

本书 1～3 章介绍了系统软件的分析方法、常用软件分析工具和分析文档的书写方法。第 4 章介绍了 *.COM 和 *.EXE 文件的结构和加载过程。第 5、6 章分析了 GWBASIC 语言,并在此基础上对其进行了改造,把非结构化的 GWBASIC 语言改进成为结构化语言。新增加了结构化的条件语句、循环语句、子程序定义与调用语句,子程序可以递归调用。第 7 章分析了 Wordstar,并介绍了 Wordstar 词典的压缩存储方法、文本文件的编辑方法和单词拼读检查的过程等。

通过对这些内容的学习,读者可掌握开发一种算法语言的基本方法、开发文本文件全屏幕编辑的方法和改进 *.COM 文,*.EXE 文件的基本方法。

本书可作为计算机系高年级学生和研究生及软件工作者学习参考。

●书　　名:　**微机系统软件分析**

WEIJI XITONG RUANJIAN FENXI

●编 著 者:朱　禹

●责任编辑:韦秋虎

●出 版 者:北京航空航天大学出版社

●印 刷 者:朝阳科普印刷厂

●发　　行:新华书店总店科技发行所

●经　　售:北京航空航天大学出版社发行科　电话　201 5720

全国各地新华书店

●开　　本:787×1092　1/16

●印　　张:12.5

●字　　数:411 千字

●印　　数:5000 册

●版　　次:1995 年 8 月第 1 版

●印　　次:1995 年 8 月第 1 次印刷

●书　　号:ISBN 7-81012-551-6/TP・150

●定　　价:17.00 元

前　言

随着计算机产业的飞速发展，每月、每天都有新的软件出现。这些新的软件在为人类提供更多的服务和便利的同时也为社会创造了巨大的财富。为了掌握这些最新的软件设计思想、实现方法，事实上我们不可能事事都亲自动手去研制、开发，既没有这个必要，也没有这种可能。为了跟上时代迅速地发展，就要掌握这些先进的方法，但是这些最先进的方法在书本上和论文中是找不到的。各大软件公司为了他们自身的利益，不可能把这些核心的先进技术过早地公布出来。只有在各大公司和大多数软件开发者都已基本掌握了这些技术时，它们才会出现在各种杂志和书籍中，这主要是为了普及的目的，同时也是对开发者的一种宣传。这时对于软件开发者来说为时已晚。那么如何掌握这些新技术呢？最好的方法是分析这些最先进的软件，进而掌握其程序设计思想和实现方法，为我所用。只有站在巨人的肩膀上，才能比巨人看得更远。

本书的1～3章介绍了系统软件的分析方法、常用的软件分析工具和分析文档的书写方法。第4章介绍了*.COM和*.EXE文件的结构和加载过程。第5、6章分析了GWBASIC语言，并在此基础上对其进行了改造，把非结构化的GWBASIC语言改进成为结构化语言。新增了结构化的条件语句、循环语句、子程序定义与调用语句，子程序可以递归调用。第7章分析了Wordstar，并介绍了Wordstar词典的压缩存储方法、文本文件的编辑方法和单词拼读检查的过程等。第5～7章软件分析与改进的部分工作是经过几届学生共同努力完成的，GWBASIC的改进经历了四年的时间才完成。

通过对这些内容的学习，读者可以掌握构造、开发一种算法语言的基本方法，可以掌握开发文本文件的编辑方法。读者还可学到改进*.COM和*.EXE文件的基本方法和技巧。通过对上述两个程序的分析可以掌握大量先进的程序设计思想和实现方法。如果掌握了这些，读者的程序设计水平就向这两个世界著名软件开发者靠近了一大步。

本书是作者根据自己多年分析、开发软件的经验写出的，它是我国第一本《微机系统软件分析》工具书。作者的本意是以此起到抛砖引玉

的作用，推动我国软件的发展。但限于作者的水平有限，书上的缺点错误在所难免。恳请广大读者和同行不断提出改进意见。

本书在编写的过程中曾得到多方面的帮助和鼓励，在此我谨向他们表示衷心地感谢。

作　者

1994.7.10

目　录

第一章　微机系统软件分析方法

§1.1　系统软件分析的目的

在当前计算机飞速发展的年代，每月、每天都有新的软件出现，为了掌握这些最新的软件设计思想、实现方法，我们不可能事事都亲自动手去开发，既没有这个必要，也没有这种可能。为了跟上时代迅速地发展，就要掌握这些先进的方法，但是这些最先进的方法在书本上和论文中是找不到的。各大软件公司为了他们自身的利益，不可能把这些核心的先进技术过早地公布出来。只有在各大公司和大多数软件开发者都已基本掌握了这些技术时，它们才会出现在各种杂志和书籍中，这主要是为了普及的目的，同时也是对开发者的一种宣传。这时对于软件开发者来说为时已晚。那么如何掌握这些新技术呢？最好的方法是分析这些最先进的软件，进而掌握其程序设计思想和实现方法，为我所用。只有站在巨人的肩膀上，才能比巨人看得更远。

分析国外的先进软件，可以对其进行改造以适应我国的需要，改正软件中的错误，完善软件的功能，增加软件的可靠性、可维护性、进行汉化等等，并在此基础上开发出适合我国需要的软件。

§1.2　系统软件分析的准备

1.2.1　软件的分类

计算机软件主要可以分为系统软件和应用软件两类。系统软件有时也简称为系统。它主要是解决计算机自身的配置和管理，实现计算机本身功能的扩充，为大多数计算机用户所共同使用的软件，如操作系统及各种语言等。一般把 Link、Edlin、Debug、Wordstar 也都归类到系统软件之中。

应用软件主要是满足用户在某些特定领域里的需要，一般由最终用户基于本专业的实际需要而提出，如财务管理系统、民航定票系统、图书管理系统、工厂的生产控制系统等等。

应用软件的分析方法与系统软件的分析方法基本是相同的。我们之所以讲述系统软件的分析方法，只是因为它更具有代表性而已。

1.2.2　获得原代码及其文档

为了分析一个软件，最好是在有源程序代码和各种软件文档的情况下进行，但实际情况这往往是不可能的，因为软件开发者常常不肯提供程序清单，其原因这是众所周知的。

如果得不到源程序代码，我们可以用高级反汇编程序将 *.COM 或 *.EXE 文件生成汇编程序或用反编译程序把它们生成相应的高级语言源程序。

如有源程序代码，应详细阅读有关资料文档，在对系统有了初步了解之后，选择动态跟踪

工具软件，如 DEBUG、T—DEBUG、CODE VIEW 等等。对于大的系统来说，要把人员作以初步的分工，为进行分析应准备更多工具软件，如反框图程序、测试程序等等。

§1.3 系统软件的分析方法

一个系统的主文件至少几十 K，多则几百 K，加之其它辅助文件、覆盖文件，一般都要达到几百 K，甚至几兆。如果单凭一条一条指令，一个子程序一个子程序地去读，去理解，工作量不堪设想，预定的目标也难以达到。为了提高分析的速度，一般应采用静态分析、动态跟踪、动静态结合验证的分析方法。

1.3.1 静态分析

在有原始文档的情况下，这一步较之无文档容易进行得多。分析方法也有所不同。若有原始文档，一般要选择入口路径，即分析的起点，应根据有关文档资料从子程序、模块、子系统直至系统逐级分析，按分析的目的顺序地进行，它是自底向上、自内向外逐步扩展路径的一种分析方法，这一步分析强调的是各子程序的入口条件、出口条件及其完成的功能及它们在模块中的地位和作用，进而理解一个模块、子系统、系统完成的功能。

在没有软件文档的情况下，这一步一般应和动态跟踪结合起来进行。分系的方法如前所述。

1.3.2 动态分析

动态分析是为了掌握系统各部分的主要功能和实现方法。如果已进行了静态分析，则动态分析是为验证和加强对系统的功能、设计思想和实现方法的理解，这一步分析是强调对系统整体的理解，各模块确切的功能。

它是自顶向下进行的。跟踪其调用关系，依次画出它们的调用关系图，从而掌握系统的结构和组成，进而猜出模块的主要功能。分析的方法主要是动态跟踪软件的执行路径，即软件的控制流，随时查看其执行结果即软件的数据流。完成动态分析以掌握系统的全貌，加强对系统执行的路径（软件的控制流）和执行的结果（数据流）的理解。以验证静态分析设想的正确性或修改静态分析的设想，进而掌握系统的总体结构和设计思想、完成功能和实现的部分方法。这是一种自顶向下的分析方法。

1.3.3 动静态分析相结合

用上述两种方法要想对软件各部分功能的具体的实现方法做到确切地掌握，达到细微的程度，进而对其进行移植和改进是有困难的。要达到这一目的要求，最好是采用动静态相结合的分析方法，也就是自顶向下与自底向上相结合的分析方法。它是自顶向下地跟踪程序的执行、掌握其调用关系，以确定你所要移植或改进模块的入口，再以此为出发点，详细地跟踪程序。将其执行的程序反汇编。对调用的树节点自底向上地进行静态的分析。将执行的结果格式化（一般要连续记录执行的结果）。若有源程序文本可以与之一一对应起来达到“运行系统源程序化”、“程序设计思想具体化”、“源程序文本动态化”、“具体细节的实现动态化”。

若没有源程序文本，在分析的过程中应充分地利用在动态跟踪过程中得到的控制流和数

据流。对反汇编后的程序代码进行详细地静态分析，以达到对“运行系统代码化”、“程序的设计思想具体化”、“反汇编程序代码动态化”。因为代码的静态分析容易掌握其细节，动态运行整体性强，将它们结合起来，容易理出系统各部分的功能和实现方法以及相关的细节，为进一步工作打下良好的基础。

1.3.4 宏观推测与微观处理相结合

在上述的分析过程中，特别是分析某一功能具体实现的过程中，要找出它所采用的数据结构和算法等具体问题时，宏观推测与微观处理相结合的方法是经常采用的。

从目前已知的初态、环境、目态等进行分析、推测这一功能实现的方法，并对程序进行动态跟踪和静态分析，用以证明推测的正确性，在分析的过程中，实际的结果往往与所推测的结果并不完全相同。根据掌握的新情况，进一步作出新的推测……，这样，反反复复直到最后动态跟踪与静态分析完全证明其推测的正确性为止。

上面所述的是分析软件的基本方法，由于各种软件的设计思想和实现方法是千差万别的，分析的方法也自然不能千篇一律，针对不同的软件灵活地运用上述方法，把它们有机地结合起来，并作到有所发展和创新。这些方法同许多新知识一样，非实践所不可得。“只能在游泳中，才能学会游泳。”

第二章　软件分析工具

软件分析的常用工具有反汇编、反编译、框图生成程序、动态跟踪程序、测试程序等等。这些工具有的是现成的，有的需要软件分析者自行开发。对于大的系统进行分析时，一般都要自行开发一部分软件工具，自行开发虽然会增加软件分析的成本，但由于所开发工具的实用性会使分析软件的效率得到较大的提高。下面我们简介几个较为实用软件分析工具。

§2.1　DEBUG

DEBUG.COM 是专为 8086/8088 汇编语言设计的一种调试工具，是搞汇编语言软件工作者必须掌握的工具。它是最早、最基本的软件分析工具。它具有显示、修改、动态跟踪等功能，使用它可以对 *.EXE、*.COM 文件进行动态地跟踪。当然对于那些具有反跟踪能力的软件应事先清除其反跟踪功能，才能对其进行分析。

有些软件要去除“反跟踪”是相当困难的，这时我们可以用 RDT 调试程序去跟踪它们。采用 RDT 调试程序基本上无需考虑“反跟踪”问题。使用 RDT 的调试者能较轻易地越过“反跟踪”这道巨大的屏障。RDT 包括了 Debug 的全部功能，其基本功能的使用与 Debug 相同。

介绍汇编语言的书籍中，都介绍了一些 Debug 的使用，但讲授汇编语言的书籍中对 Debug 介绍得都很不全面，只是为满足汇编语言教学的需要，讲述了 Debug 的部分功能；随着 Debug 版本的不断增高，Debug 也增加了新的功能，为了使读者对 Debug 较新版本有个全面的了解，以便在调试程序中查阅，我们本节对 Debug 的命令做以全面的介绍。

2.1.1　Debug 的启动

为了启动 Debug 我们可以按下列格式键入信息：

〔驱动器〕〔路径〕DEBUG 〔驱动器〕〔路径〕〔〈文件名〉〕〔参数 1〕〔参数 2〕

说明：1.〔　〕中的内容可以省略

2.〈文件名〉包括文件名和扩展名

3.参数 1、参数 2 表示正在调试的程序的输入和输出的说明。例如：

DEBUG DISKCOPY.COM A：　B：

4.Debug 装入后的提示符为“-”，表示在该提示符下可以接收 Debug 的任何命令。

5.如果是键入 DEBUG 而无文件说明，可以对内存的内容进行工作，或用 N（命名）和 L（装入）命令把需要的文件装入内存。

2.1.2　Debug 命令中的约定

在 Debug 的所有命令中都有如下的约定

1.命令参数的分隔符

除 q 和 xs 命令以外所有 Debug 命令都可带有参数，各参数之间可以用空格和逗号做为

分隔符，但只有在两个连续的数值之间，分隔符才是必须的，因此下面三条命令是等同的：

```
dcs:100 110
d cs:100 110
d,cs:100,110
```

2. 地址(address)的有效值

Debug 命令中的地址参数是指内存中的一个位置。地址分为两部分，段地址和偏移量，段地址可以是段寄存器或 4 位数字(小于 4 位也可以)，段地址有时可以省略。a、g、l、t、u 和 w 命令的段地址为 CS 寄存器的值，所有其它命令的段地址隐含为 DS 寄存器的值。所有数字均为 16 进制。

3. 范围(range)的有效值

在 Debug 中范围参数指定内存中的一个范围。它可以有两种形式表示地址范围。一种是由起始地址和结束地址构成，另一种是由起始地址及 L 表示范围长度构成。例如指定从 CS:100 开始的 10H 个字节范围可写成：

```
CS:100 10F
CS:100 L10
```

4. 命令键入

Debug 的所有命令以大写或小写键入具有相同的功能，每个命令均以回车作为结束符，也就是说只有打回车计算机才响应该命令。

2.1.3 Debug 的命令集

1. A(汇编)

功能： 对用 8086/8087/8088 汇编语言写的程序进行汇编，并直接装入内存。

格式： A〔地址〕

说明： ①地址可是下列任意一格式

```
CS:0100
03B0:0100
100
```

命令分别表示从 CS:100 处开始汇编，从 03B0:0100 开始汇编，从 CS:100 开始汇编。

②FAR 返回助记符为 RETF。

③转移及调用的汇编指令，Debug 会自动汇编成 short、near 或 far 转移或调用指令。

可以通过使用 near 或 far 前缀重新指定转移或调用。

例：

```
A 0B00:0500
0B00:0500 jmp 502              ;2 字节的 short 转移
0B00:0502 jmp near 505         ;3 字节的 near 转移
0B00:0505 jmp far 50           ;5 字节的 far 转移
```

④区分字和字节内存位置

Debug 无法识别 word(字)或 byte(字节)所以必须用前缀 word ptr 或 byte ptr 区分数据类型时，二者可分别简化成 wo 和 by。

⑤伪操作 db,dw 可以在 A 命令中使用

例： db 1,2,"this is an example"
dw 1000,2000,"ABCD"

⑥改变指定省略段地址寄存器时，用 CS:、DS:、ES:、SS:

例： CS:
mov AX,[110]

是把 CS:[110]的内容赋给 AX，若没有 CS:，则 mov AX,[110]是把 DS:[110]内容赋给 AX。

⑦使用 8087 操作码时，必须指定 wait 或 fwait 前缀。

2. C(比较)命令

功能： 比较内存中两个数据块的内容

格式： C 范围 地址

举例： C 100 110 200
C 100 L10 200

这两条命令都是将 100～110 的内存块的内容同 200～210 内存块的内容进行比较，Debug 用类似下面格式应答上一个命令(假定 DS:1200)

```
1200:100 4D    E4 1200:0200
1200:101 5A    20 1200:0201
1200:102 67    88 1200:0202
1200:103 4B    27 1200:0203
1200:104 8A    99 1200:0204
1200:106 77    3A 1200:0206
1200:107 44    41 1200:107
1200:110 5B    51 1200:110
```

这里缺少 1200:105，1200:108，1200:109～1200:10F 的内容，表示这些地址的内容与其对应地址的内容相同。

3. D(卸出)命令

功能： 显示内存中指定地址范围的内容。

格式： D[地址]或 D[范围]

说明： 显示出的内容分为两部分，前面是 16 进制数部分，用 16 进制数显示每个字节的内容，80 列显示每行 10H 个字节；后面是 ASCII 部分，显示每个字节所代表的 ASCII 码，不可显示的字符(0～31 和 127～255)用"."代替。

举例：
```
D CS:0100        ;显示 CS:100 开始的 80H 个字节的内容
D 1200:0100      ;显示从 1200:100 开始的 80H 个字节的内容
D 100            ;显示从 DS:100 开始的 80H 个字节的内容
D 100 300        ;显示从 DS:100 到 DS:300 的内容
```

4. E(输入)命令

功能： 向内存指定地址输入数据，输入的数据可以十六进制的数，也可以是 ASCII 字符。新输入的数据替换了原来的内容，因此该命令也经常被称为修改命令。

格式： E 地址〔清单〕

说明： 若将清单省略，比如：

E 地址

输入一个 16 进制数(1 位或 2 位)来代替原来地址的内容，然后可选择下列三个动作之一：

①按空格键是进行下一个地址，并显示它的内容，如果要修改它，就键入新的内容；若不想修改就再次按空格键，步进到下一个地址。

②输入一个连字符“—”就返回到前一个地址，并在新行上显示前一个地址和它的内容。如果你想修改它就键入新的内容，若不修改并想返回到上一个字节，就再次按连字符“—”。

③若想结束该命令，按 Enter 键。

举例：Eds:100 FF"ABC"8D

用清单中 5 个字节的内容替换由 ds:100 开始的 5 个字节的内容。

5. F(填写)命令

功能： 用清单中的内容填写指定范围内的内存单元。

格式： F 范围 清单

说明： 若清单中的字节数少于范围中的字节数，就重复使用清单的内容，直到所有被指定的存储单元填满为止。如果清单中包含的字节数大于地址范围，那么就忽略多余的清单项。

举例： F DS:110:114 41"BCD"45

把 DS:100 至 DS:114 的内容填成 41 42 43 44 45 即 ASCII 码 ABCDE。

6. G(执行)命令

功能： 执行你正在调试的程序。

格式： G〔=地址〕〔地址〔地址〕...〕

说明： ①当省略所有地址参数时，程序从 CS:IP 所指的地址开始执行，一直执行到程序的断点或终点。

②当省略〔=地址〕时，程序从 CS:IP 处开始，执行到〔地址〔地址〕...〕中一个断点地址停止，若遇不到上述的段点，执行同①中说明。在〔地址〔地址〕...〕最多可设置 10 个段点地址。主程序执行到先遇到的一个断点处停止。Debug 在每个 断点处都用中断指令 INT 3(代码 CC)代替断点地址的指令，程序执行到任何一 个断点处，所有断点处的指令都恢复成原来的指令(包括遇到程序中本

身设置断点，也是如此）。程序停止后，显示所有寄存器的内容，标志位的状态以及断点处的指令代码和指令。

③一旦程序执行完毕，则显示“program terminated normally”，表示程序已正常结束，若要再次执行必须重新装入。

④断点地址只能放在8086等操作码的第一个字节上，如果不是在第一个字节上，将会产生不可预料的结果。

⑤断点最多只能设置10个，如超过10个将会给出错误信息"bp Error"。

⑥用户的堆栈指针必须是有效的，且必须有6个字节可供G命令使用。

举例：G 105,110 220

程序从CS:IP处开始执行，程序中设置了3个断点CS:105,CS:110,CS:220。

G=110,CS:1175,1190

程序从CS:110处开始执行，程序中所设置的段点为CS:1175,CS:1190。

7. H(16进制算术运算)命令

功能：执行用户定义两个参数的十六进制数的加减运算。

格式：H 参数1　参数2

说明：参数1或参数2代表任意一个0到FFFFH范围的十六进制数。

Debug执行时首先将两个指定的十六进制参数值进行相加，然后再从第一个数值中减去第二个数值，在同一行上显示计算的结果，先是和后是差。

举例：H 0f 8

0017 0007

8. I(输入)命令

功能：从指定端口输入(读入)并显示一个字节，字节是16进制的数。

格式：I 端口地址

举例：I 2F8　　;从2F8H端口读入一个字节

6A　　;读入的字节内容为6A，并显示在屏幕上。

9. L(装入)命令

功能：将文件指定扇区的内容读入到内存。

格式：L〔地址〔驱动器号 扇区号 扇区数〕〕

说明：①地址为把磁盘上的内容读到内存的首地址。

②驱动器号是指要读入的磁盘所在的驱动器号，其数值为0=A,1=B,2=C

③扇区号是指要读入内容的起始扇区号。

④扇区数是指由起始扇区开始要读入的扇区数。

⑤如省略了所有参数，L命令将装入以N命令命名的文件到内存CS:100处。