

OOP 技术丛书

计算机程序调试与分析工具

——Turbo Debugger 与 Turbo Profiler

殷镇良 萧柔 烛华 等 编 潘金贵 顾铁成 审 校

南京大学出版社

OOP 技术丛书

计算机程序调试与分析工具

——Turbo Debugger 与 Turbo Profiler

殷镇良 萧 柔 烛 华 等编

潘金贵 顾铁成 审校

南京大学出版社

1994 · 南京

(苏)新登字 011 号

内 容 简 介

Borland 公司推出的 Turbo Debugger 和 Turbo Profiler 是程序开发的强有力工具。Turbo Debugger 是一个源程序级调试程序,用于 Turbo 系列语言的编程调试。而 Turbo Profiler 则是程序性能分析程序,是一个通过寻找程序瓶颈来度量程序性能指标的软件工具,它使用户能够监视计算机的重要活动,并向用户提供有关程序性能各方面的详细统计报告。

本书详细说明了 Turbo Debugger 和 Turbo Profiler 的工作原理与使用方法,是从事软件开发的人员和工程技术人员学习和使用这两个工具的一本重要的参考书。

QOP 技术丛书

计算机程序调试与分析工具

——Turbo Debugger 与 Turbo Profiler

殷真良 蔡 荣 烛 华 等编

潘金贵 顾铁成 审校

南京大学出版社出版

(南京大学校内 邮编:210093)

江苏省新华书店发行 江苏印刷厂印刷

*

开本:787×1092 1/16 印张:26.62\$字数:664 千

1994 年 10 月第 1 版 1994 年 10 月第 1 次印刷

印数:1—5 000

ISBN 7-305-02737-5/TP·110

定价:20.00 元

前 言

一、Turbo Debugger 与 Turbo Profiler 概述

1. Turbo Debugger

Turbo Debugger 是由 Borland 公司随 Borland C++ 软件包一起推出的一个源程序级调试工具,用于调试用 Turbo 系列语言编写的源程序。它的多重覆盖窗口、下拉菜单和弹出菜单的结合以及鼠标支持等等,为用户提供了一个快速友好的交互环境。在整个调试过程中,随时可以提供与上下文有关的联机帮助信息。

Turbo Debugger 具有以下性能:

- 在调试大程序时使用扩展内存规范(EMS)。
- 可对 C, C++, Pascal 和汇编语言表达式求值。
- 可重新设置屏幕格式。
- 必要时可进行汇编语言/CPU 级的调试。
- 断点和记载功能强。
- 击键记录(宏)。
- 能进行反向跟踪。
- 可使用远程系统对大型程序进行调试。
- 支持 80386 和其他销售商的调试硬件。
- 全面支持 Turbo Pascal 5.5 和更高版本的面向对象的程序设计。
- 全面支持 Borland 的 C++ 系列编译器。
- 可调试 TSR 程序与设备驱动程序。
- 可调试 Microsoft Windows 应用程序。

在 3.0 版本中, Turbo Debugger 修改了对 DOS 和 Windows 都适用的某些性能,增加了只适用于 Windows 调试的一些性能。

对 DOS 和 Windows 都适用的性能包括:

- 利用剪辑板可在对话框的文本输入框和窗口之间,或在诸窗口之间传递信息。
- 增加了新的断点功能,允许用户:
 - * 在断点上设置多重条件和多种操作;
 - * 成组设置和取消断点;
 - * 设置和取消某一模块里所有函数和过程中的断点;
 - * 设置和取消在某一对象类型中所有方法和某一类中所有成员函数上的断点。
- 支持 C++ 模板和嵌套类。
- 可在网络上或者在串行连接上进行远程调试。

- 通过 DOS 的 COUNTRY 命令和 Windows 的 Language 设置,支持国际定制,以符合各个国家的使用习惯。

只适用于 Windows 环境调试的特性如下:

- CPU 窗口有一显示保护模式选择符和观察由这些选择符引用的内存地址的内容的分区。
- 设备驱动程序 TDDEBUG. 386 提供支持 Ctrl—Alt—SysRq 程序断点。另外,该设备驱动程序还支持 Intel 80386 处理器(或更高的处理器)硬件调试寄存器。
- 由于 Turbo Debugger for Windows(TDW)同时装入应用程序的符号表和有关动态连接库(DLL)的符号表,因此 DLL 的调试比以前版本快。

2. Turbo Profiler

Turbo Profiler 是一个程序性能分析软件工具,它通过寻找程序瓶颈来度量程序性能指标,例如,它能测出:

- 程序主要时间开销在何处。
- 某行代码被执行多少次。
- 某个例程被调用多少次和被谁调用。
- 程序主要访问哪个文件和每次访问所花的时间。

该工具还可监视几类关键的计算机资源:

- 处理器时间。
- 磁盘存取活动。
- 键盘输入。
- 打印机输出。
- 中断活动。

使用 Turbo Profiler,用户便能够监视计算机的重要活动,获得程序性能各方面的详细设计报告,这样,用户就能对自己的程序进行“微调”。

Turbo Profiler 通过揭示程序的内幕情况和隐蔽操作(例如执行时间、语句执行次数、中断调用和文件访问)来能帮助用户精炼代码和加快程序执行速度。

Turbo Profiler 在功能和易用性方面比其他程序分析工具更胜一筹,这主要表现在下列几个方面:

- 交互式分析能迅速找出程序中的低效代码。
- 在分析过程中,用户可以读出或编辑任何文本文件。
- 可以分析在 DOS 或 Windows 下运行的任意大小的程序。
- 可用于任何 Turbo Pascal, Borland 的 C++ 编译器和 Turbo Assembler 程序。
- 提供一个易用界面,带有多重重叠窗口,鼠标器支持和上下文相关帮助。
- 报告单行或例程的执行时间及次数。
- 追踪哪些代码块被执行,哪些未被执行。
- 对所有例程的调用路径作完整的追踪,使用完整的调用堆栈追踪分析调用频率。
- 通过 File 窗口监视 DOS 文件活动,包括处理和文件打开、关闭、读出、写入的时间。使用事件列表记录读入或写出的字节的序号。
- 支持选择性中断监视,监视所有的视频、键盘、磁盘、鼠标器和一般中断。提供一个完

整的事件列表或频率监视结果。以符号名形式给出常用 DOS 调用列表,可以使用户迅速识别各类中断。

- 支持对 Turbo Pascal 和 Borland 的 C 或 C++ 覆盖的完整跟踪。
- 386 虚拟机分析方式占用最少的 RAM,使用户可使用更多的 RAM。
- 允许远程串行和网络分析。

Turbo Profiler 在有些方面实现了代码优化器没能做到的功能,它直接告诉用户哪段代码慢,何处是运行瓶颈,何时重构算法等等。

二、硬件及软件需求

1. Turbo Debugger

Turbo Debugger 能够在 IBM PC 系列微机(包括 XT、AT、PS/2 等)以及所有与 IBM 真正兼容的微机上运行,它运行于 DOS 3.1 或更高版本的 DOS 操作系统下至少需要 384K 的 RAM,可以在任何彩色或者单色 80 列监视器上运行。最好使用有硬盘的计算机,如果使用的计算机只有双软驱,则必须使用高密盘。也可以使用 3.5 英寸的 720K 磁盘,但 INSTALL 实用程序不能在这些磁盘上安装 Turbo Debugger,所以用户必须手工拷贝这些文件。

Turbo Debugger 不需要 80x87 数学协处理器芯片。

Turbo Debugger 可与 Borland 公司的下列产品一起使用: Borland C++ 系列编译器、Turbo Pascal 5.0 或更高版本、Turbo Assembler 1.0 或更高版本。在使用 Turbo Debugger 前,必须先把源文件编译成可执行的 .EXE 文件,并带有全部调试信息;或者编译成带相关的 .TDS 文件的 .COM 文件。

运行 Turbo Debugger 时,同时需要 .EXE 文件与相应的源文件。Turbo Debugger 在四个目录下搜索源文件:首先在编译器编译源文件时源文件所在的目录下查找;其次在 Options Path for Source 命令所指定的目录下查找;然后在当前目录下查找;最后在 .EXE 文件所在目录下查找。

2. Turbo Profiler

Turbo Profiler 可在 IBM PC 系列机上运行,包括 XT、AT、PS/2 系列和所有与 IBM 全兼容的机型。DOS 版本要求 2.0 或更高版本,并要求至少有 384K RAM。

Turbo Profiler 在任何 80 列监视器上都能执行。最好有一个硬盘,当然也可以使用只有双软盘驱动器的机器。

Turbo Profiler 不必有 80x87 数学协处理器。

三、术语解释

为了叙述上的简洁,在本书中使用了几个术语,其意义都略微超出了它们的一般含义。这些术语是:模块(Module)、函数(Function)或例程(Routine)、参数(Argument)。

模块(Module):代表 C 和汇编语言中通常所指的模块,也代表 Pascal 中的单元(unit)。

函数(Function)或例程(Routine):代表 C 中的函数和 Pascal 中的子程序,包括函数(Function)、过程(Procedure)和对象方法(Object Method)。在 C 中,函数既可以返回一个值(相

当于 Pascal 中的函数),又可以不返回值(相当于 Pascal 中的过程)。当一个 C 函数不返回值时,它被称为 void 函数。本书中使用的函数(Function)既代表 C 的函数也代表 Pascal 的函数与过程。当然,在书中与特定语言相关的部分里,这一点不适用。

参数(Argument):Argument 在本书中可与 Parameter 互换使用,它们都代表命令行参数或传递给过程和函数的参数。

四、内容安排

为了使我国广大的计算机工作人员能够更好地利用 Turbo 系列程序设计语言进行程序设计工作,我们根据长期从事 Turbo 系列语言编程的经验,参阅了国内外的有关软件、资料,编译了这本《计算机程序调试与分析工具——Turbo Debugger 与 Turbo Profiler》,它全面、系统地介绍了 Turbo Debugger 和 Turbo Profiler 的基本概念和高级程序设计技术,是一本读者使用程序调试与分析工具的非常实用的参考书。

本书由两篇组成,第一篇“使用和学习 Turbo Debugger”共 23 章,第二篇“使用和学习 Turbo Profiler”共 10 章。

介绍 Turbo Debugger 的各章及附录内容如下:

第一章“Turbo Debugger 的安装与启动”介绍了如何把 Turbo Debugger 文件安装到系统上, Turbo Debugger 软件所包含的文件。说明如何在 DOS 提示符下运行 Turbo Debugger。

第二章“程序调试过程与 Turbo Debugger 环境”介绍了 Turbo Debugger 的环境、菜单和窗口,并且演示了如何响应各种提示与错误信息,何时使用命令行选项,以及如何在配置文件中保存常用的设置。

第三章“程序调试示范”引导用户浏览一个样本调试过程(使用 Pascal 或者 C 程序),这个调试示例演示了 Turbo Debugger 许多强大的功能。

第四章“程序执行的控制”演示各种启动与终止被调试程序的方法,同时也介绍了如何重新开始一个调试会话或者重演上一次调试会话。

第五章“检查和修改数据”介绍了 Turbo Debugger 所特有的检查和修改程序中数据的方法。

第六章“断点的设置和利用”介绍了动作(action)概念,并解释所谓断点(breakpoint)、观察点(watchpoint)和跟踪点(tracepoint)。在该章中,对条件动作和无条件动作都作了解释,同时也说明了动作被触发时可能出现的各种情况。

第七章“考查和修改文件”描述如何检查与修改程序源文件或者任何磁盘文件。这些文件可以是文本形式,也可以是二进制形式。

第八章“表达式”描述 Turbo Debugger 所能接受的 C, C++, Pascal 以及汇编语言表达式的句法,同时也介绍如何使用格式控制字符来改变表达式值的显示方式。

第九章“调试 C++ 与面向对象的 Pascal 程序”解释用来检查对象(在用 Turbo Pascal 5.5 或更高版本编写的程序中)和类(在用 Borland C++ 系列编译器编写的程序中)的 Turbo Debugger 特殊性能。

第十章“汇编级调试”介绍如何按原始的十六进制数据察看和修改内存,如何使用嵌入式汇编与反汇编功能,以及如何检查和修改 CPU 寄存器值及其标志位。

第十一章“80x87 协处理器芯片与仿真器”讨论如何检查和修改浮点硬件或仿真器的内容。

第十二章“调试技巧”介绍了一些有效地调试程序的技巧。

第十三章“在 80386 处理器上的虚拟调试”描述如何利用扩充内存与 80386 计算机的能力,使被调试程序能够使用所有的常规 640K 地址空间,就好像没有安装 Turbo Debugger 一样。

第十四章“保护方式下的程序调试”介绍了如何利用 TD286 在保护方式下运行 Turbo Debugger,以便释放更多的内存空间,供大型被调试程序使用。

第十五章“面向 Windows 的 Turbo Debugger(TDW)”描述了如何运行 TDW 以及如何使用其特殊功能。

第十六章“调试标准的 Windows 应用程序”引导用户逐步调试用 C 语言编写的样本 Windows 程序。

第十七章“调试 ObjectWindows 应用程序”引导用户逐步调试一个用 Borland C++ 和 ObjectWindows 类库编写的样本 Windows 应用程序。

第十八章“调试 TSR 与设备驱动程序”解释如何调试终止并驻留(TSR)程序以及在 Turbo Debugger 启动时驻留的程序,还说明了如何手工装入一个符号表。

第十九章“远程调试”解释了如何利用 WREMOTE 实用程序来远程调试 Windows 应用程序,如何利用 TDREMOTE 实用程序来远程调试 DOS(非 Windows)应用程序。借助于这些实用程序,可在一台计算机上运行 Turbo Debugger,而在另一台计算机上运行所调试的程序。

第二十章“Turbo Debugger 的定制”解释了如何使用安装程序(TDINST)来定制屏幕颜色和改变缺省选项。

第二十一章“命令一览表”完整地列出了所有主菜单命令和对应于每个窗口类型的局部菜单命令。

第二十二章“Turbo Debugger 命令行选项和技术说明”总结了 Turbo Debugger 的所有命令行选项,并给出了 Turbo Debugger 与被调试程序和 DOS 打交道时的一些实现细节。

第二十三章“Turbo Debugger 对话框与错误消息”列出了所有可能出现的 Turbo Debugger 提示和错误消息,同时对如何响应它们给出了建议。

介绍 Turbo Profiler 的各章及附录内容如下:

第二十四章“分析器简介与使用示范”。该章首先简单介绍分析器是什么样的工具,概括 Turbo Profiler 的特点,并介绍在系统上运行 Turbo Pdrofiler 的一些准备知识,然后介绍了分析全过程的一个入门的典型例子。

第二十五章“Turbo Profiler 环境”对 Turbo Profiler 环境中每个菜单项和对话框选择作了详细的解释。

第二十六章“程序分析的策略和方法”提供了一般性指导和一些窍门,以求能得到丰富多彩的分析结果。

第二十七章“Turbo Profiler 工作原理”用模拟的方法解释分析器是如何在用户程序运行期间收集执行时间及执行次数方面的数据。

第二十八章“在 Windows 环境下使用 Turbo Profiler”介绍了如何使用 Windows 版的分析程序——Turbo Profiler for Windows。

第二十九章“程序的远程分析”叙述如何在两台机器上进行分析,用户可以在一台机器上运行程序而在另一台上运行 Turbo Profiler。

第三十章“在 80386 处理器上作虚拟分析”解释如何在 80386 的扩展内存上运行分析器,而把全部的 640K 常规内存让给用户程序。

第三十一章“修改 Turbo Profiler 配置”解释如何使用 TFINST 来改变 TPROF 的缺省配置。

第三十二章“Turbo Profiler 的命令行选项”列出了所有的 Turbo Profiler 命令行选项,并解释其完成的功能。

第三十三章“Turbo Profiler 的提示与出错信息”列出所有可能出现的提示信息 and 出错信息,并提示如何去处理这些信息。

参加本书编写工作的人员有殷镇良(编写第一篇第二部分共 11 章)、萧柔(编写第一篇第一部分共 9 章)、烛华(编写第二篇第二部分共 4 章)、朴建仁(编写第二篇第一部分共 4 章)、吴卫华(编写第二篇第三部分共 2 章)、朴世枫(编写第一篇第三部分共 3 章)等同志,王勇、朱训衷、顾铁成、冀惠刚等同志也参加了部分编写工作及辅助工作,殷镇良、潘金贵和萧柔同志对全书进行了仔细的修改和统编。

本书由潘金贵主审,顾铁成同志协助校阅了有关章节,值此谨表谢意。

由于水平、时间所限,不妥之处在所难免,欢迎读者批评指正。

编 者

1994 年 1 月于南京

目 录

第一篇 学习和使用 Turbo Debugger

第一部分 Turbo Debugger 基本功能和用法

第一章 Turbo Debugger 的安装和启动	3
1.1 安装 Turbo Debugger	3
1.2 准备待调试的程序	5
1.2.1 准备 Borland C++ 程序	5
1.2.2 准备 Turbo Pascal 程序	5
1.2.3 准备 Turbo 汇编程序	6
1.3 启动 Turbo Debugger	6
1.4 命令行选项	6
1.4.1 装载配置文件(-c)	7
1.4.2 显示更新方式(-d)	7
1.4.3 获取帮助(-h 与-?)	7
1.4.4 进程 ID 转换(-i)	7
1.4.5 按键记录(-k)	7
1.4.6 汇编模式启动(-l)	8
1.4.7 设置堆大小(-m)	8
1.4.8 鼠标器支持(-p)	8
1.4.9 远程调试(-r)	8
1.4.10 源代码处理(-s)	8
1.4.11 视频硬件(-v)	9
1.4.12 远程窗口调试(-w)	9
1.4.13 覆盖池大小(-y)	9
1.5 配置文件	10
1.6 选项菜单	10
1.6.1 Language(语言)命令	10
1.6.2 Macros(宏)菜单	10
1.6.3 显示选项命令(Display Options)	11

1.6.4	Path for Source 命令	12
1.6.5	保存选项命令(Save Option)	12
1.6.6	恢复选项命令	13
1.7	在 Turbo Debugger 中运行 DOS	13
1.8	返回 DOS	13
1.9	阅读建议	13
1.9.1	正在学习 Turbo 语言的程序员	13
1.9.2	已经用过 Turbo 语言的程序员	14
第二章	程序调试过程与 Turbo Debugger 环境	15
2.1	程序调试过程	15
2.1.1	有错误吗	15
2.1.2	错误在哪儿	15
2.1.3	是什么错误	15
2.1.4	改正错误	15
2.2	Turbo Debugger 的功能和限制	16
2.2.1	Turbo Debugger 能做什么	16
2.2.2	Turbo Debugger 不能做什么	17
2.3	Turbo Debugger 的环境	17
2.3.1	菜单与对话框	18
2.3.2	了解当前位置	19
2.3.3	历史特性	21
2.3.4	递增匹配	22
2.3.5	生成宏	22
2.3.6	窗口介绍	23
2.3.7	复制和粘贴	29
2.3.8	获取帮助	32
第三章	程序调试示范	35
3.1	演示程序	35
3.2	使用 Turbo Debugger	36
3.2.1	菜单	36
3.2.2	状态行	36
3.2.3	窗口	37
3.3	使用 C 演示程序	38
3.3.1	设置断点	39
3.3.2	使用监视	39
3.3.3	检查简单的 C 数据对象	40
3.3.4	检查复合 C 数据对象	41
3.3.5	改变 C 数据值	42
3.4	使用 Pascal 示例程序	43

3.4.1	设置断点	44
3.4.2	使用监视	44
3.4.3	检查简单的 Pascal 数据对象	45
3.4.4	检查复合的 Pascal 数据对象	46
3.4.5	改变 Pascal 数据值	46
第四章	程序执行的控制	48
4.1	检查当前程序状态	48
4.1.1	变量窗口	48
4.1.2	堆栈窗口(Stack Window)	51
4.1.3	源(Origin)局部菜单命令	51
4.1.4	获取信息(Get Info)命令	52
4.2	运行菜单	55
4.2.1	运行(Run, F9)	55
4.2.2	执行到光标位置(Goto Cursor, F1)	56
4.2.3	跟踪进入(Trace Into, F7)	56
4.2.4	单步执行(Step Over, F8)	56
4.2.5	执行到某处(Execute to..., Alt F9)	56
4.2.6	执行到返回(Until Return, Alt F8)	56
4.2.7	连续慢速执行(Animate...)	57
4.2.8	反向跟踪(Back Trace, Alt F1)	57
4.2.9	指令跟踪(Instruction Trace, Alt F7)	57
4.2.10	参数(Argument...)	57
4.2.11	程序复位(Program Reset, Ctrl F2)	57
4.3	执行历史窗口	58
4.3.1	指令区	58
4.3.2	按键记录区	59
4.4	中断程序执行	60
4.5	程序终止	60
4.6	重新开始调试	60
4.6.1	重新加载用户程序	61
4.6.2	按键宏记录和重用	61
4.7	打开新的待调试程序	61
4.8	改变程序参数	62
第五章	检查和修改数据	63
5.1	数据菜单	63
5.1.1	检查(Inspect...)	63
5.1.2	求值/修改(Evaluate/Modify...)	63
5.1.3	增加监视项(Add Watch...)	65
5.1.4	函数返回值(Function Return)	65

5.2	指向源文件中的数据对象.....	65
5.3	监视窗口.....	66
5.4	检查窗口.....	67
5.4.1	C 数据检查窗口	67
5.4.2	Pascal 数据检查窗口	70
5.4.3	汇编语言数据检查窗口.....	72
5.5	检查窗口的局部菜单.....	74
5.5.1	范围(Range...)	74
5.5.2	修改(Change...)	74
5.5.3	检查(Inspect...)	75
5.5.4	下降(Descend...).....	75
5.5.5	新表达式(New Expression...)	75
5.5.6	强制类型转换(Type Cast...)	75
第六章	断点的设置和利用	76
6.1	断点菜单.....	77
6.1.1	设置或清除断点(Toggle)	77
6.1.2	在...(At...)	77
6.1.3	内存变化全局断点(Changed Memory Global)	77
6.1.4	全局表达式为真(Expression True Global)	77
6.1.5	硬件断点(Hardware Breakpoint)	77
6.1.6	全清>Delete All)	77
6.2	断点窗口.....	77
6.2.1	断点窗口的局部菜单	78
6.2.2	Breakpoint Options 对话框	80
6.2.3	Conditions and Actions 对话框	81
6.3	定制断点.....	84
6.3.1	简单断点	84
6.3.2	全局断点	84
6.3.3	改变内存断点	85
6.3.4	条件表达式	85
6.3.5	硬件断点.....	86
6.3.6	记载变量值	86
6.3.7	断点和模板	86
6.4	Log 窗口	87
第七章	考察和修改文件	89
7.1	考察源程序文件.....	89
7.1.1	模块窗口	89
7.1.2	模块窗口的局部菜单	90
7.2	考察其他磁盘文件.....	92

7.2.1	文件窗口	92
7.2.2	文件窗口的局部菜单	92
第八章	表达式	94
8.1	为计算表达式选择语言	94
8.2	代码地址、数据地址和行号	95
8.3	存取当前作用域以外的符号	95
8.3.1	作用域重载语法	95
8.3.2	表达式的隐含范围	99
8.4	字节列表	99
8.5	C 表达式	99
8.5.1	C 符号	100
8.5.2	C 寄存器伪变量	100
8.5.3	C 常量和数值格式	101
8.5.4	转义序列	101
8.5.5	C 运算符优先级	101
8.5.6	在程序中执行 C 函数	103
8.5.7	带副作用的 C 表达式	103
8.5.8	保留字和类型转换	103
8.6	Pascal 表达式	104
8.6.1	Pascal 符号	104
8.6.2	Pascal 常量和数值格式	104
8.6.3	Pascal 字符串	104
8.6.4	Pascal 运算符和优先级	104
8.6.5	调用 Pascal 函数和过程	105
8.7	汇编表达式	105
8.7.1	汇编符号	105
8.7.2	汇编常量	105
8.7.3	汇编运算符	106
8.8	格式控制	106
第九章	调试 C++ 和面向对象的 Pascal 程序	108
9.1	层次窗口	108
9.1.1	对象类型列表(Object Type List) 区	108
9.1.2	层次树区	109
9.2	对象类型/类检查窗口	110
9.3	对象实例检查窗口	111
9.3.1	对象/类实例检查窗口的局部菜单	112
9.3.2	中区和底区	113

第二部分 高级调试技术

第十章 汇编级调试	114
10.1 源程序级调试	114
10.2 CPU 窗口	114
10.3 代码区	116
10.3.1 反汇编	116
10.3.2 代码区局部菜单	116
10.4 寄存器和标志位区	119
10.4.1 寄存器区的局部菜单	119
10.4.2 标志区局部菜单	120
10.5 选择符区	120
10.6 数据区	121
10.7 堆栈区	123
10.8 汇编	124
10.8.1 操作数地址大小重载限定符	124
10.8.2 操作数据长度前缀	125
10.8.3 串指令	125
10.9 转储窗口	125
10.10 寄存器窗口	126
10.11 嵌入汇编关键字	126
10.11.1 8086/80286 指令助记符	126
10.11.2 80386 指令助记符	127
10.11.3 80486 指令助记符	127
10.11.4 80386 寄存器	127
10.11.5 CPU 寄存器	128
10.11.6 特殊关键字	128
10.11.7 8087/80287 数学协处理器指令助记符	128
10.11.8 80387 指令助记符	128
第十一章 80x87 协处理器芯片和仿真器	129
11.1 80x87 芯片和仿真器	129
11.2 数学处理器窗口	129
11.2.1 寄存器区	130
11.2.2 控制区	130
11.2.3 状态区	131
第十二章 调试技巧	133
12.1 出现问题	133
12.2 调试风格	133

12.2.1	运行整个程序	133
12.2.2	增量测试	134
12.3	错误类型	134
12.3.1	一般性错误	134
12.3.2	C 特有的错误	135
12.3.3	Pascal 特有错误	139
12.3.4	汇编特有的错误	144
12.4	精确性测试	148
12.4.1	测试边界条件	148
12.4.2	无效的数据输入	148
12.4.3	空数据输入	148
12.5	作为程序设计一部分的调试	148
12.6	样例调试过程	149
12.7	C 调试过程	149
12.7.1	寻找错误	149
12.7.2	确定解决方案	150
12.7.3	启动 Turbo Debugger	150
12.7.4	检查	150
12.7.5	断点	150
12.7.6	监视窗口	151
12.7.7	求值/修改对话框	151
12.7.8	错误所在	151
12.8	Pascal 调试过程	152
12.8.1	寻找错误	152
12.8.2	确定解决方案	153
12.8.3	启动 Turbo Debugger	153
12.8.4	巡查程序	153
12.8.5	求值/修改对话框	154
12.8.6	检查	154
12.8.7	监视	155
12.8.8	最后一个错误	155
12.9	如何在不同语言下使用 Turbo Debugger	156
12.9.1	Borland C++ 的提示	156
12.9.2	Turbo Assembler 提示	157
12.9.3	Turbo Pascal 的提示	158
第十三章	在 80386 处理器上的虚拟调试	161
13.1	虚拟调试所需设备	161
13.2	安装虚拟调试设备驱动程序	161
13.3	启动虚拟调试	161

13.4	一般调试和虚拟调试的区别	163
13.5	解决问题的一些诀窍	163
13.6	TD386 错误信息	164
13.7	TDH386.SYS 错误信息	164
第十四章	保护模式下的程序调试	165
14.1	保护模式调试程序的硬件要求	165
14.2	安装保护模式调试程序	165
14.3	启动保护模式调试程序	165
14.4	保护模式的调试程序与 Turbo Debugger 的不同点	165
14.5	在不同的机器上运行 TD286	166
第十五章	面向 Windows 的 Turbo Debugger	167
15.1	运行 TDW 的软硬件需求	167
15.2	安装 TDW	167
15.3	配置 TDW	168
15.3.1	使用 TDW 命令行选项	168
15.3.2	使用 TDINST	169
15.4	使用 TDW	169
15.4.1	TDW 和 Turbo Debugger 之间的区别	169
15.4.2	TDW 特点	170
15.5	记录窗口消息	170
15.5.1	为标准的 Windows 应用程序选择窗口	170
15.5.2	为 ObjectWindows 应用程序选择窗口	171
15.5.3	指定消息类及其动作	173
15.5.4	观察消息	175
15.5.5	观察消息要诀	175
15.6	获取内存及模块列表	176
15.7	调试动态连接库	178
15.7.1	使用 Load Modules or DLLs 对话框	179
15.7.2	向 DLLs&Programs 列表添加 DLL	180
15.7.3	针对 DLL 设置调试选项	181
15.7.4	控制 TDW 装入 DLL 的符号表	181
15.7.5	调试 DLL 启动代码	181
15.8	将内存句柄转换成地址	182
15.9	调试诀窍	183
15.10	TDW 错误消息	184
第十六章	调试标准的 Windows 应用程序	185
16.1	样本程序	185
16.2	编译并连接演示程序	185
16.2.1	调试 BCWDEMOA	186