


第2章 动态分析技术

 电子工业出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY



声明：本电子文档是《加密与解密(第三版)》的配套辅助电子教程！电子工业出版社依法对本作品享有专有出版权。任何未经权利人书面许可，复制、销售或通过信息网络传播本作品的行为；歪曲、篡改、剽窃本作品的行为，均违反《中华人民共和国著作权法》，其行为人应承担相应的民事责任和行政责任，构成犯罪的，将被依法追究刑事责任。

为了维护市场秩序，保护权利人的合法权益，我社将依法查处和打击侵权盗版的单位和个人。欢迎社会各界人士积极举报侵权盗版行为，本社将奖励举报有功人员，并保证举报人的信息不被泄露。

举报电话：(010) 88254396；(010) 88258888

传 真：(010) 88254397

E-mail: dbqq@phei.com.cn

通信地址：北京市万寿路 173 信箱

电子工业出版社总编办公室

邮 编：100036

说明：本篇文档是对《加密与解密》第三版“第2章 动态分析技术”补充，为了降低书的价格，直接以电子文档提供给图书购买者。

看雪软件安全网站

<http://www.pediy.com>
kanxue

2008-6-1

2.2 SoftICE调试器

SoftICE 是 Compuware NuMega 公司 (www.compuware.com) 的产品，是目前公认最好的系统级调试工具！兼容性和稳定性极好，可在源代码级调试各种应用程序和设备驱动程序，也可使用 TCP/IP 连接进行远程调试。

NuMega 公司已在 2006 年 4 月宣布 Driver Studio 停止开发，目前支持到 Windows 2003，这意味着 SoftICE 将在新系统中消失。本节只简单介绍一下 SoftICE 基本用法，详细的请参考其自带的文档 Using SoftICE.pdf 和 CommRef.chm。光盘映像文件也提供了 TiANWEi 翻译的 SoftICE 的中文命令手册。

在开始之前先谈谈 ICE 的含义，ICE (In Circuit Emulator) 即实体电路模拟器，是用来跟踪软件执行动作细节的一个模拟 CPU 的电子设备。当然，这种设备价格昂贵，不是常人所能拥有的。NuMega 公司推出的 Soft-“ICE”，意思是靠软件实现 ICE 的功能。

SoftICE 单独发行的最高版本是 4.05，针对不同平台推出了相应的版本：DOS, Windows 3.x, Windows 9x, Windows NT 和 Windows 2000。

现在，Compuware NuMega 公司将 SoftICE 捆绑进驱动开发套件 DriverStudio 和 SoftICE DriverSuite 中发行。DriverStudio 是一个设备驱动程序和应用软件开发工具包，在这个套件中包括久经考验的工具 SoftICE, DriverWorks, VtoolsD 和 DriverAgent，以及基于应用层技术（这种技术曾用于 BoundsChecker, TrueTime, TrueCoverage 和 FailSafe）的新型设备驱动程序工具。SoftICE DriverSuite 是一套用来加速微软 Windows 设备驱动程序的开发和调试的核心工具，里面也包含 SoftICE。

2.2.1 安装

SoftICE 4.05 在 Windows 9x 系统上很稳定，但其 Windows 2000 版本或多或少总会出现点问题，如断点失效、不能返回应用程序领空等。另外，该版本不支持 USB 键盘与鼠标，因此建议为 Windows 2000 以上系统选择更高版本的 SoftICE。

Windows ME 是 Windows 9x 的升级版本，其核心与 Windows 9x 是一样的，因此 SoftICE 4.05 for Win 9x 版本可以在此系统上安装，安装时需要一个补丁程序“Winice Loader”。

DriverStudio 内置的 SoftICE 支持 USB 键盘与鼠标, DriverStudio 2.6 版以上支持 Windows XP 系统。

1. SoftICE 安装

本章以 SoftICE DriverSuite 为例讲述 SoftICE 的安装与配置, SoftICE 4.05 与之差不多, 只是界面形式不同。DriverSuite 可自动识别 Windows 不同版本, 安装时双击 setup.exe 进入标准安装向导界面, 单击“Next”后经过几个界面来到安装类型窗口, 如图 2.46 所示。

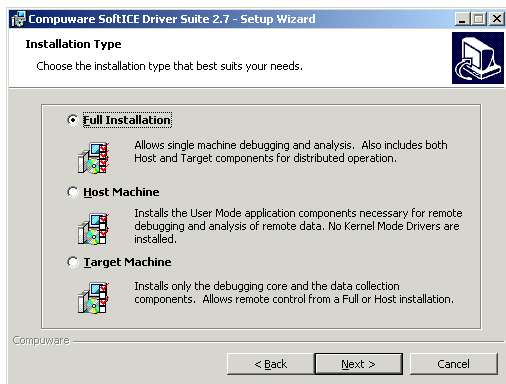


图 2.46 安装类型选择

有 3 种安装类型选择。

- Full Installation: 完全安装, 允许单机调试和分析, 包括 Host 和 Target 两部分;
- Host Machine: 仅安装远程调试和分析的组件;
- Target Machine: 仅安装调试核心部分, 以供 Full 或 Host 类型主机远程调试。

若是单机安装, 按默认选择 Full Installation 类型, 单击“Next”按钮来到组件选择框(见图 2.47)。

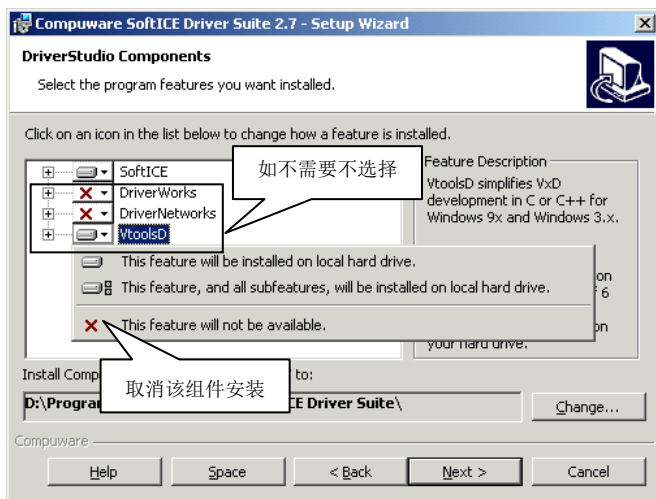


图 2.47 安装组件选择

在此选择 SoftICE 组件，其他组件是有关驱动开发方面的。单击“Next”按钮，SoftICE 将开始安装。安装完毕，自动打开选项配置框（见图 2.48），具体配置在下一节中专门介绍。

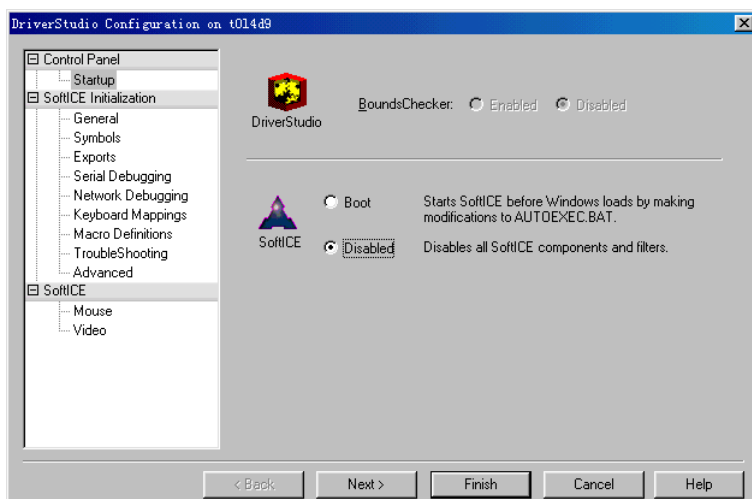


图 2.48 Windows 9x 系统启动模式设置

安装的最后一步是电子注册画面，其中有四个选项可供选择。

- Register using internet (requires Web browser): 通过浏览器进行因特网在线注册；
- Register using E-mail: 通过电子邮件进行注册；
- Print registration for faxing or mailing: 将注册信息打印出来，通过传真或邮寄注册；
- Register later: 以后注册。

上面的注册选项并不影响 SoftICE 的使用，一般直接选择“Register later”。

2. 启动方式

SoftICE for Windows 9x 核心是两个 VxD 文件，而 SoftICE for Windows NT/2000/XP 核心是两个 NT 内核设备驱动程序，具体见表 2-5。

表 2-5 SoftICE 核心文件

Windows 9x (VxD)	Windows ME	Windows NT/2000/XP	说 明
WINICE.EXE SIWVID.386	WINICE.EXE SIWVID.386 WINICE.VXD DEBUGGER.EXE	NTICE.SYS SIWVID.SYS	调试核心文件 显示驱动程序

在 Windows 9x 系统中，winice.exe 必须在 win.com 程序运行之前运行（即在纯 DOS 下运行），它会自动调用 Win.com 程序来启动 Windows 系统。

在 Windows NT/2000/XP 系统中，SoftICE 作为内核设备驱动程序必须在系统内核启动后才能被装载。因此，不能跟踪调试硬件抽象层（HAL）等的初始化过程。

(1) Windows 9x 系统启动模式

- Boot: 将在 C 盘根目录的 Autoexec.bat 中加入一行：c:\windows\winice.exe；

- Disabled: 禁止 SoftICE 启动。


SoftICE 在 Windows 9x 系统中是由 winice.exe 来启动系统的, 建议自己配置 Autoexec.bat 和 Config.sys 文件以增加启动菜单, 然后通过启动菜单来确定是否运行 SoftICE。

Autoexec.bat 配置样例:

```
@ECHO OFF
goto %config%
: SICE
c:\windows\winice.exe                // DriverStudio 用此命令行
C:\PROGRA~1\NUMEGA\SOFTIC~1\WINICE.EXE // SoftICE 4.05 用此命令行
goto common
: NORM
goto common
: common
```

Config.sys 配置样例:

```
[MENU]
MENUITEM NORM, Windows 9x
MENUITEM SICE, Windows 9x with SoftICE
MENUDEFAULT NORM, 2           // 菜单停留 2 秒
[NORM]
[SICE]
[common]
```

 **注意:** 由于 SoftICE 工作在 CPU 和操作系统之间, 所以 SoftICE 装载后就不能被停止, 只有重新启动系统。

(2) Windows NT/2000/XP 系统启动模式

SoftICE 在 Windows NT/2000/XP 平台上的安装与在 Windows 9x 上差不多, 不同的是在安装过程中有四个启动模式供选择 (见图 2.49)。

- Boot 模式: SoftICE 先于 Windows 装载, 这种模式主要用于调试内核驱动程序;
- System 模式: SoftICE 后于 Windows 加载, 这个模式用于调试一般的应用软件开发;
- Automatic 模式: 自动模式, SoftICE 先于 Windows 加载, 但不能调试内核驱动程序;
- Manual 模式: 手动模式, 此种模式就是进入 Windows 平台后, 需要手动执行快捷方式 “Start SoftICE” 或在命令行上运行 “net start ntice” 来装载 SoftICE;
- Disabled: 禁止所有的 SoftICE 组件服务。

Boot, System, Automatic 和 Manual 分别对应于一个驱动 (服务) 启动模式, 其中 Boot 模式启动最早, Manual 最晚。SoftICE 只有早于驱动 (服务) 启动, 才能调试该驱动 (服务)。一般情况下将启动模式设置为 Manual 模式, 此时启动 SoftICE 方法是执行快捷方式 “Start SoftICE”。它对应的文件是 ntice.bat, 其内容为: “net start ntice”。也就是说, Ntice.sys 是一个内核设备驱动类型的服务, 服务名称是 NTICE。

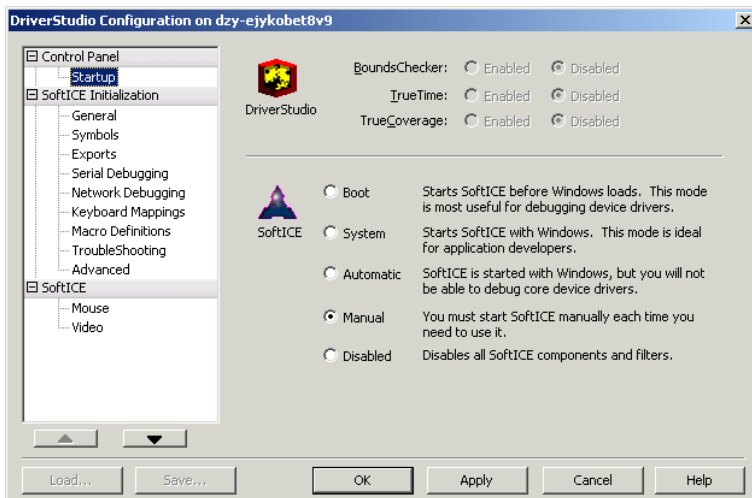


图 2.49 Windows NT/2000/XP 系统启动模式设置

若启动模式为 Boot 或 System, Windows NT/2000/XP 启动时, 在屏幕下方有一条蓝字的提示信息: “Press Esc to cancel loading SoftICE”, 此时按 Esc 键取消 SoftICE 的装载。此功能对 Automatic 模式无效。

Boot 模式时, SoftICE 先于 Windows 装载, 使其不能正常加载配置文件 winice.dat。NuMega 公司为 Windows NT/2000/XP 下的 SoftICE 设计了一个驱动程序: SoftICE Symbol Driver (SIWSYM)。当 SoftICE 启动时, SIWSYM 为其提供文件映像表地址, SoftICE 根据这个表寻找所需的配置文件, 从而能将其正常加载到系统中。

系统默认地将 Siwsym.sys 安装到 %SystemRoot%\System32\Drivers 目录中。若未发现, 需手动在 SoftICE 安装目录里将其复制过去。然后将下面一段文本存为 .reg 文件, 导入注册表。

```
[REGEDIT4]
```

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Siwsym]
```

```
"Type"=dword:00000001
```


```
"Group"="NTICE"
```

```
"ErrorControl"=dword:00000001
```

```
"Start"=dword:00000000
```

```
"Tag"=dword:00000001
```

最后运行 Icepack.exe, 它会将 Winice.dat 里的配置放进 Siwsym.sys 文件里, 最后重新启动系统。

 **注意:** 在 Boot 模式下, 每次改变 SoftICE 的配置 (直接编辑 Winice.dat 或通过 Loader32 改变配置), 都要重新运行 Icepack.exe, 这样才能保证新的修改起作用。由于 SoftICE 的不同版本设置可能稍有不同, 安装时请阅读安装目录下的 Siwsym.txt 说明。

3. 鼠标的配置

单击“Mouse”选项卡打开鼠标配置框（见图 2.50）。

- Serial (COM1): COM1 端口上的串口鼠标;
- Serial (COM2): COM2 端口上的串口鼠标;
- PS/2 compatible or USB: PS/2 或 USB 接口鼠标;
- None: 不安装鼠标;
- Enhanced Mouse: 鼠标有滚轮或新类型的鼠标。

若鼠标配置不正确,可能会导致 SoftICE 工作不正常,所以必须确认鼠标类型选择正确。若还不能正常工作,建议选择“None”选项来取消鼠标功能。

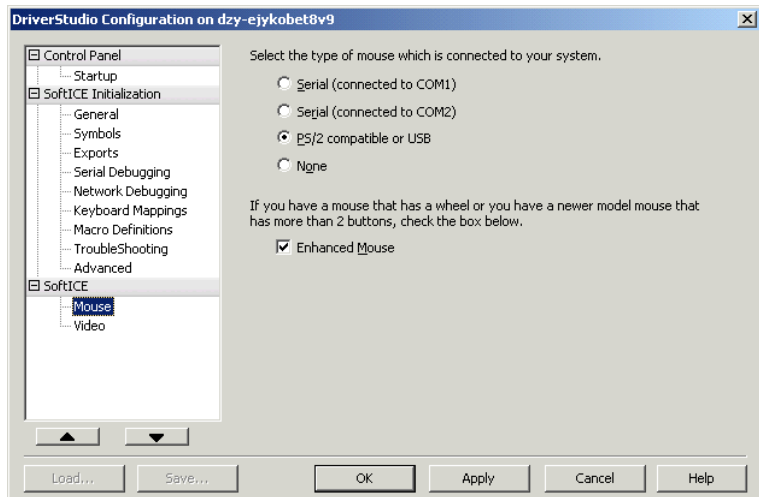


图 2.50 鼠标配置

4. 显卡的配置

单击“Video”选项卡打开显卡配置框（见图 2.51）。有两种模式供选择：一种是“Full screen VGA mode”模式，另一种是“Universal Video Driver”模式。

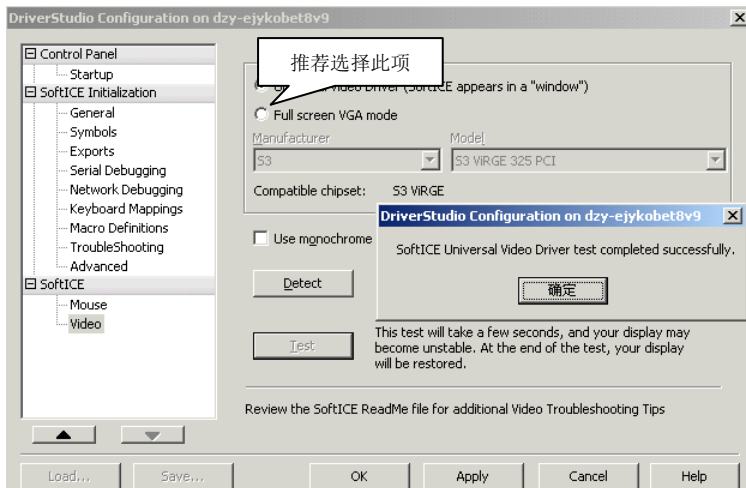


图 2.51 显卡配置

(1) Full screen VGA mode (全屏字符模式)

此项使 SoftICE 在激活状态下类似于 DOS 全屏 (也就是字符模式状态)。配置时, 先单击 “Detect” 自动检测显卡类型, 然后单击 “Test” 按钮测试。在测试过程中能够看到各种颜色的字符, 说明显卡测试通过。

SoftICE 不能运行的情况大多是与显卡配置有关。因为 SoftICE 直接与硬件打交道, 虽然大多数硬件设备是标准的, 但显卡则不同 (只有 VGA 显卡实现标准化), 所以 SoftICE 提供了常用显卡的支持。和显卡有关的文件存放在 SoftICE 目录中, 即 SIWVID.386 或 SIWVID.SYS 文件。

若全屏模式工作不正常, 解决方法有如下这些。

- 下载最新的显卡驱动程序或试用公版驱动程序;
- 从 NuMega 公司主页下载最新显卡支持文件 SIWVID.386 或 SIWVID.SYS;
- 在显卡配置项重新设置显卡类型。

(2) Universal Video Driver 模式 (通用显卡驱动模式)

Microsoft DirectX 技术的出现使得 SoftICE 从 3.2 版开始实现了显卡通用驱动程序 (Universal Video Driver, UVD), 此技术让 SoftICE 以窗口形式显示在屏幕上, 摆脱了字符模式显示方式。UVD 技术使支持 DirectX 的新显卡在 SoftICE 上都可正常工作 (注意: 显卡和显卡驱动都必须支持 Direct Draw, 否则请选择 Full screen VGA mode 模式)。

强烈推荐采用此模式, 这样可避免显示器不停地在图形和字符模式之间转换。配置时, 先单击 “Detect” 自动检测显卡类型, 然后选择 “Universal Video Driver” 这一项。确认无误后, 单击 “Test” 按钮测试。若弹出如图 2.51 所示的对话框, 测试通过, 否则重新设置。

如果 SoftICE 在 UVD 模式下不能正常工作, 请试着按如下步骤解决。

- ☐ 打开显卡配置框;
- ☐ 从显卡制造厂商 (Manufacturer) 中选择 “STANDARD VGA”;
- ☐ 选择 “Universal Video Driver (UVD)”;

☐ 按“Test”按钮测试。

若不成功, 则可能 SoftICE 将显卡驱动认错了 (该问题在老显卡里出现的比较多):

☐ 打开显卡配置框;

☐ 选择“Universal Video Driver (UVD)”;



图 2.52 显卡驱动程

☐ 打 开 注 册 表

[HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\NTTice], 查看“InstalledDisplayDrivers”键值与显卡驱动程序名是否相同。若不同, 纠正过来就能正常工作。

例如, 对于 S3 ViRGE DX 芯片显卡, 在 Windows 2000 系统上 SoftICE 显示不正常。查看显卡驱动程序, 选择“控制面板/系统/硬件/设备管理器/显示卡/属性/驱动程序详细信息”打开如图 2.52 所示的属性框。S3 ViRGE DX 驱动主程序是 s3mtrio.dll, s3mvirge.dll 和 s3m.sys。打开注册表, “InstalledDisplayDrivers”项的

值为“s3mtrio”, 现将此值换成显卡驱动的另一 DLL 文件名“S3mvirge”, 重新启动系统即可。如下次再进入 Setting 配置框时, SoftICE 可能会提示检测到显卡是否要修正, 选择“否”。

如果还是不能工作, 并且确信显卡支持 DirectDraw, 建议做如下操作。

☐ 下载最新的显卡驱动程序, 升级显卡驱动是因为厂商可能增加了对 Direct Draw 的支持;

☐ Windows NT/2000/XP 系统则尝试改变 Automatic 或 Manual 启动模式。

2.2.2 调试窗口简介

成功装载 SoftICE 后, 按“Ctrl+D”键激活并打开一个调试窗口 (见图 2.53)。当需要返回 Windows 系统时, 再按“Ctrl+D”键, 也可用 X 命令或按 F5 键。

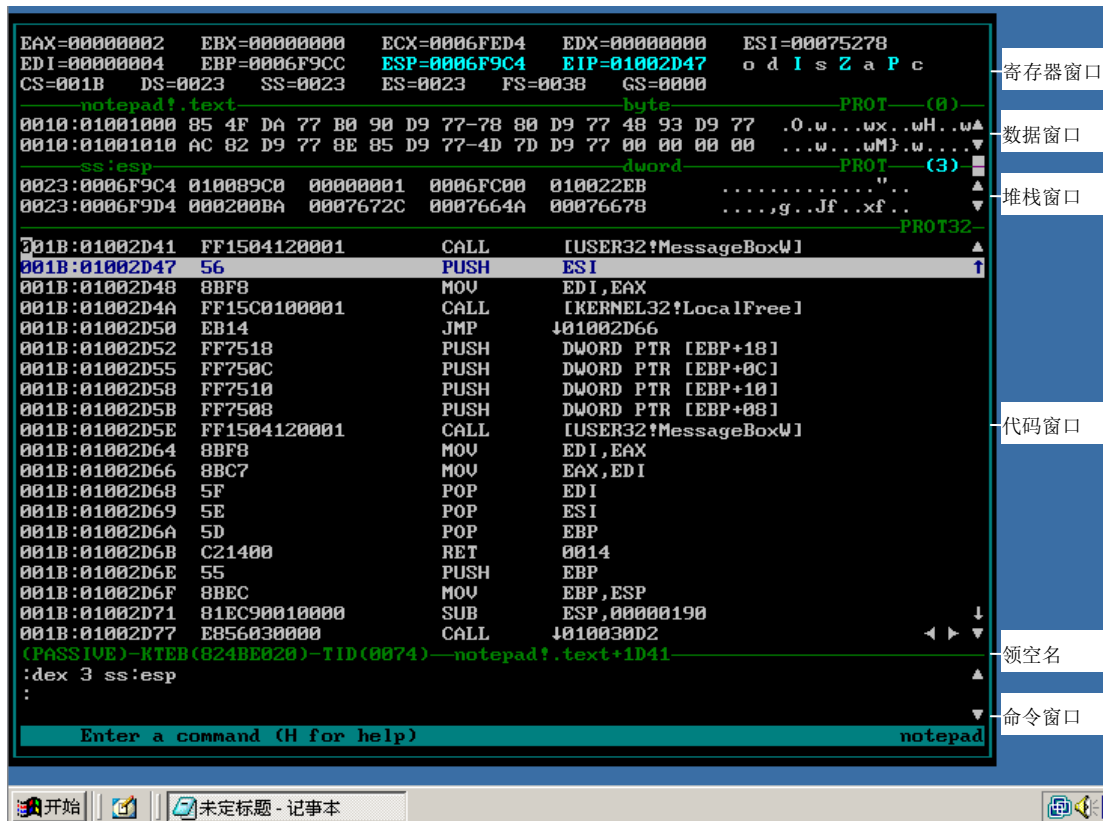


图 2.53 SoftICE 的调试窗口

注意：如果 SoftICE 不能成功激活，按如下步骤检查配置：

- ☐ 确保已装载 SoftICE。若是 Windows 9x，检查一下 Autoexec.bat 中的 Winice.exe 路径是否正确。若是 Windows NT/2000/XP，检查是否执行了快捷方式“Start SoftICE”？或换成其他的启动模式。
- ☐ 按“Ctrl+D”键黑屏，此时建议再按“Ctrl+D”键一次，以将控制权返回到 Windows 系统，参照显卡配置一节检查。
- ☐ 能激活进入 SoftICE，但光标失灵乱跳，这是因为 SoftICE 一直存在着对 PS/2 接口鼠标支持不太好的问题，用命令“Set Mouse Off”关闭鼠标，等正常后用命令恢复：“Set Mouse On”。
- ☐ 运行快捷方式 Settings，将其 General 选项的 Video Memory Size 值调大些。

调试窗口分为寄存器、浮点、数据、代码、堆栈和命令窗口等几部分。

1. 寄存器窗口

用 WR 命令打开或关闭，按“Alt+R”键将光标移到此窗口。在这里可以观察到各种寄

寄存器的当前值，如数据寄存器 EAX, EBX, ECX, EDX 和控制寄存器（EIP 和 PSW）等。当跟踪调试程序时，SoftICE 将高亮显示数据发生变化的寄存器。

标志寄存器：O (OF)、D (DF)、I (IF)、S (SF)、Z (ZF)、A (AF)、P (PF) 和 C (CF)。小写状态显示时，表示置 0；大写状态将高亮显示，表示置 1。

按 Tab 或“Shift+右光标键”指向下一个寄存器；按“Shift+Tab”键或“Shift+左光标键”使光标退回到上一个寄存器。修改后，按回车键确认，按 Esc 键取消。

相关命令如下：

CPU：可以显示所有 CPU 寄存器的内容（普通、控制、除错、段）。

2. 浮点窗口

用 WF 命令打开或关闭。在这里显示了 8 个浮点数据寄存器（FPU），编号为 ST0~ST7，也能显示 MMX 寄存器。

相关命令如下所示：

数 据 格 式	说 明	适 用 范 围
WF F	仅指向浮点	仅适合浮点
WF B	以字节 (Byte) 显示	仅适合 MMX
WF W	以字 (Word) 显示	
WF D	以双字 (Dword) 显示	

3. 数据窗口

用 WD 命令打开或关闭，按“Alt+D”键将光标移到此窗口（见图 2.54）。在数据窗口中显示指定内存中的数据，以十六进制和 ASCII 字符形式同时显示。

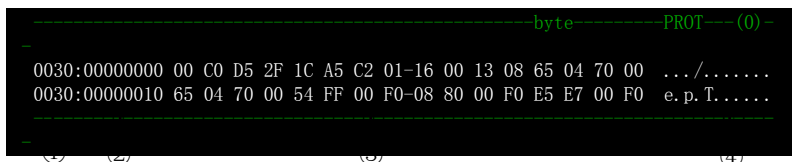


图 2.54 数据窗口

- ①号位置 0030：这是段选择器；在 Windows 的保护模式下，CS, DS, ES, SS, FS 和 GS 的名称不是段寄存器，是段选择器。
- ②号位置 00000000：此处表示内存的地址；
- ③号位置的值是当前内存的数据，以十六进制表示；
- ④号位置的值是“③”处值的另一种表示方法——ASCII 码。

相关命令如下所示：

命 令	操 作 结 果
WD.n [lines] n = (1、2、3、4)	打开或关闭另一数据窗口，最多可打开 4 个
DATA n n = (1、2、3、4)	在几个数据窗口间切换
FORMAT (Shift+F3)	改变数据窗口格式
DEX [window-number] [expression]	显示或设置数据窗口表达式



技巧：dex 3 ss:esp：设置 3 号数据窗口显示堆栈信息（将其作为堆栈窗口用）。

4. 代码窗口

用 WC 命令打开或关闭，按“Alt+C”键将光标移到此窗口（见图 2.55）。

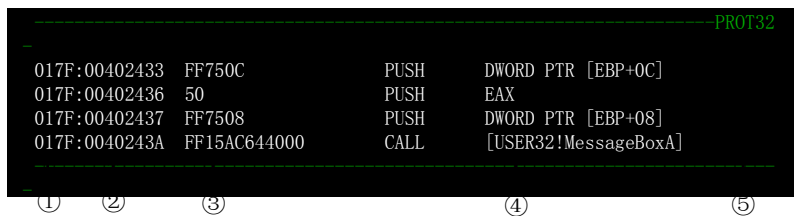


图 2.55 代码窗口

- ①段选择器：同一程序在不同系统环境下此值可能不同，一般也不需要关心此值；
- ②虚拟地址：一般情况下，同一程序的同一条指令在不同系统环境下此值相同。
- ③机器码：这就是 CPU 执行的机器代码，可用“code on/off”命令打开或关闭；
- ④汇编指令：和机器码对应的程序代码；
- ⑤PROT32：32 位保护模式，寻址方式 selector:offset；PROT16 为 16 位保护模式。

5. 系统堆栈窗口

用 WS 命令打开或关闭，这个堆栈窗口和 dex 3 ss:esp 设置的不同，显示形式如下：

FrameEBP	RetEIP	Symbol
0012FFC0	77E7CA90	NOTEPAD!.text+00CF
0012FFF0	00000000	KERNEL32!CreateProcessw+15A6

- FrameEBP：当前的 EBP 值；
- RetEIP：EBP 指向的下一个数据；
- Symbol：当前代码语句的偏移量，如“NOTEPAD!.text+00CF”表示此时位置相对.text 块偏移量为 0x00CF。

操作时，用鼠标双击某个显示的项目，即可跳到相关代码处。

6. Pentium III/IV 寄存器窗口

用 WX 命令打开或关闭。

7. 命令窗口

命令窗口是各种命令执行的地方，屏幕底部的动作状态行显示执行命令的各种提示，并提供指令语法帮助。按上下光标键可查阅命令历史记录。

8. 领空

所谓领空，实际上是指在某一时刻，CPU 的 CS:EIP 所指向的某段代码的所有者。图 2.53 中当前领空是 NOTEPAD!.text，表示 SoftICE 停下来时光标所指的那一句指令位于

Notepad.exe 程序的.text 区块里。在 SoftICE 里，领空名是不显示扩展名的。

2.2.3 窗口操作

1. 窗口属性

SoftICE 使用通用显卡驱动程序（UVD）将调试窗口显示在用户的桌面上，UVD 允许 SoftICE 直接在屏幕上绘制信息。表 2-6 列出的命令可以调整窗口位置、大小。

表 2-6 窗口属性

命 令	操 作 结 果
LINES <i>n</i>	范围是 25~128 线性调整，设置显示行数
WIDTH <i>n</i>	范围是 80~160 线性调整，设置宽度
SET FONT <i>n</i>	(<i>n</i> =1, 2, 3) 设置字符大小
SET ORIGIN <i>x y</i>	设置窗口位置， <i>x</i> , <i>y</i> 是窗口左上角的坐标（坐标原点为 0, 0）
SET FORCEPALETTE [ON OFF]	当置 ON 时，SoftICE 防止系统颜色的 8-位模式被改变，这样确保 SoftICE 的显示可见。默认为 OFF
SET MAXIMIZE [ON OFF]	当置 ON，SoftICE 重新调整窗口以最大化屏幕，受字体、显示行数、显存影响
Ctrl+Alt+光标键	移动窗口
Ctrl+Alt+Home(End, Page Down, Page Up)	重设窗口位置原点（操作时不要碰到旁边的 Delete 键，否则系统重启）
Ctrl+Alt+C	将窗口定位到屏幕中间
Ctrl+L	当调试窗口与应用窗口部分图像混杂时，按此键可刷新调试窗口

初次激活 SoftICE 时窗口很小，此时 LINES 默认为 25。如是字符模式，LINES 可以设置为 43, 50 或 60；如是 UVD 模式，LINES 可在 25~128 之间线性设置。UVD 模式下，常用的调整范围如下：

800×600 分辨率：SET FONT 2, LINES 40；

1024×768 分辨率：SET FONT 2, LINES 50。

2. 窗口打开或关闭命令

表 2-7 的命令可以打开窗口，再次执行可以关闭该窗口，也可直接用鼠标调整窗口。

表 2-7 窗口打开或关闭命令

命 令	窗 口
WC <i>n</i>	打开或关闭代码窗口，或改变代码窗口大小
WD <i>n</i>	打开或关闭数据窗口，或改变数据窗口大小
WR	打开或关闭寄存器窗口
WF	以浮点形式显示浮点栈
WS <i>n</i>	打开或关闭堆栈窗口
WW <i>n</i>	打开或关闭监视窗口，或改变监视窗口的大小
WT <i>n</i>	打开或关闭线程监视窗口
WX	打开或关闭 Pentium III/IV 寄存器窗口
WL <i>n</i>	打开或关闭本地窗口（局部变量窗口）

3. 窗口切换

光标默认在命令窗口中，若想移动光标到其他窗口，可用鼠标操作。如果光标在命令窗口或代码窗口中，也可以用“Alt + 热键”移动光标（见表 2-8）。重复此组合键光标可在两窗口间切换。

表 2-8 窗口切换命令

窗 口	Alt 键组合
代码窗口	Alt+C
数据窗口	Alt+D
寄存器窗口	Alt+R
本地窗口（局部变量窗口）	Alt+L
系统堆栈窗口	Alt+S
线程窗口	Alt+T
监视窗口	Alt+W

4. 窗口滚屏

SoftICE 支持窗口滚屏，这个功能特别有用。当需要在各小视窗中滚屏观看被挡住的区域时，可用鼠标或热键来控制屏幕。命令参见表 2-9。

表 2-9 窗口滚屏命令

组 合 键	操 作
Alt+ ↑ (↓)	数据窗口向上（下）滚动一行
Alt+Page Up(Page Down)	数据窗口向上（下）滚动一屏
Ctrl+ ↑ (↓)	代码窗口向上（下）滚动一行
Ctrl+ Page Up(Page Down)	代码窗口向上（下）滚动一屏
Shift+ ↑ (↓)	命令窗口向上（下）滚动一行

5. 鼠标右键输入命令

鼠标右键菜单可以在初始选项配置里设定。常用的命令有 D, U, WHAT 等命令。菜单命令参见表 2-10。

表 2-10 鼠标右键命令

鼠标右键命令	SoftICE 执行命令	功 能
Copy		复制
Paste		粘贴
Copy&Paste		将鼠标所选的数据复制到命令窗口中
Display	D	显示指令地址内存数据
Un-Assemble	U	反汇编
What	What	用来确定一个名字或表达式是否为已知类型
Previous	N/A	撤销上一个鼠标命令

Reip	r eip %cp%	对 EIP 赋值，不要乱用，否则系统崩溃
Add Watch	watch %cp%	增加监视窗口

6. 功能键

功能键可以代替一串 SoftICE 指令，可用命令行设定或在 Winice.dat 中自定义。SoftICE 默认的功能键定义见表 2-11。

表 2-11 功能键列表

热 键	命 令	功 能
F1	H	显示一般帮助信息
F2	WR	在寄存器窗口中切换
F3	SRC	在源程序、反汇编代码，两者混合之间切换显示
F4	RS	恢复屏幕内容
F5	X	从 SoftICE 窗口中退出
F6	EC	在命令窗口和代码窗口中切换
F7	^HERE	执行到光标所在行
F8	T	单步执行，可跟进 CALL 内
F9	BPX	在光标所在行设置断点
F10	P	单步执行，跳过 CALL
F11	G @SS: SP	执行到返回地址
F12	p ret	直到出现 RET (xxxx) 命令中断
Shift+F3	FORMAT	改变数据窗口格式，按字节、字、双字、短实型、长实型、10 字节实型循环
Alt+F1	WR	打开或关闭寄存器窗口
Alt+F2	WD	打开或关闭数据窗口
Alt+F3	WC	打开或关闭代码窗口
Alt+F4	WW	打开或关闭监视窗口
Alt+F5	CLS	清屏命令（命令窗口）
Alt+F11	dataaddr->0	取得数据窗口第一个双字所指的双字节值
Alt+F12	dataaddr->4	取得数据窗口第二个双字所指的双字节值

2.2.4 SoftICE 配置

用户可以自定义 SoftICE 的环境参数，运行 Symbol Loader 快捷方式，选中“Edit”菜单，选择“SoftICE Initialization Settings”打开初始化设置框（见图 2.56），或单击“Settings”快捷方式也可打开。

1. 常规选项（General）

General 选项设置 SoftICE 初始化命令和一些变量参数（见图 2.56）。

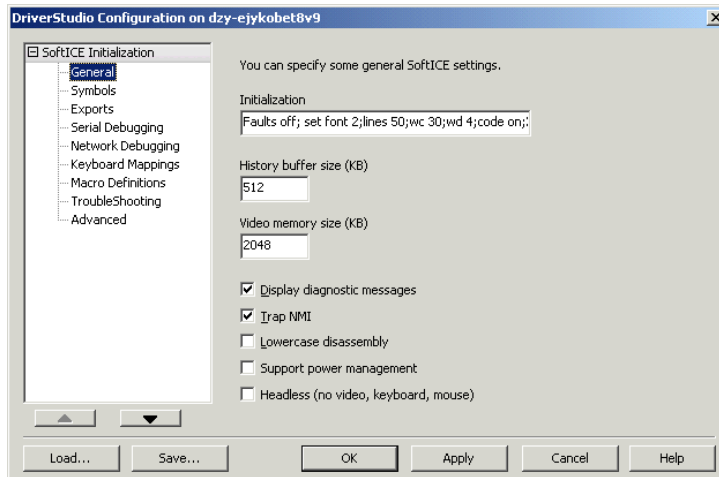


图 2.56 General 选项卡

- **Initialization:** SoftICE 加载时的初始化命令，默认值是“X;”，例如初始化字符串“INIT=“Faults off;set font 2;lines 40;data 3;dd;dex 3 ss:esp;data 0;wc 20;code on;X;””，其中 Faults off 是关闭错误检测功能，set font 2 设置 2 号字体，lines 40 显示 40 行，data 3 打开 3 号数据窗口，dex 3 ss:esp 设置 3 号窗口显示堆栈信息，data 0 将 0 号窗口设为当前窗口，wc 20 代码窗口显示 20 行，code on 显示机器码。
- **History buffer size:** SoftICE 历史缓冲区的大小，默认为 256KB。历史缓冲区用来缓存所有显示在命令窗口的信息，可把这个信息保存在一个文件中。需要时运行 Symbol Loader，在“File”菜单中，选择“Save Softice History As”存到一个文件里。
- **Trace Buffer Size (仅 Windows 9x):** 跟踪缓冲区，由 BPR 和 BPRW 命令用来存放跟踪信息，默认值为 8KB。
- **Total RAM (仅 Windows 9x):** 物理内存大小。
- **Video Memory Size:** 当用 UVD 驱动时，SoftICE 必须将当前屏幕内容储存起来以备恢复，这就需要占用一定的显存。SoftICE 默认地占用 2MB 内存。若按上一节调整显卡配置，SoftICE 不能正常工作。建议将此值设大些或许能解决问题。
- **Display Diagnostic Messages:** 是否在命令窗口中显示额外信息，如加载、卸载的模块等。建议不要选上，否则利用历史缓冲区抓取汇编代码时提示信息太多。
- **Trap NMI:** 是否跟踪不可屏蔽中断 NMI，默认时为标记该选择项。
- **Lowercase Disassembly:** 反汇编时是否使用小写字母表示指令，默认时没有标记该选择项。
- **Support power management:** 支持节能管理。

2. 符号选项 (Symbols)

SoftICE 启动时可以预装符号和代码。这些对调试设备驱动程序，比如静态 VxD 或系统的 SDK（软件开发工具包）很有用。SoftICE 可直接使用的调试符号文件只有*.NMS。符号生成过程参考符号调试技术一节。单击“Add”按钮（见图 2.57）可把.NMS 文件加进去。

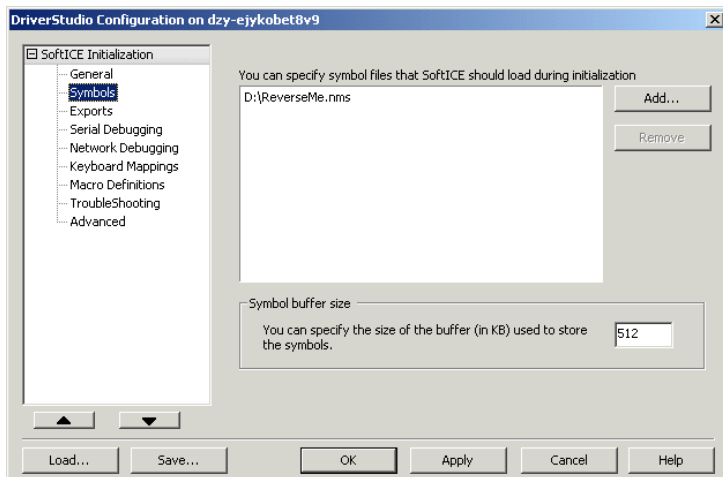


图 2.57 预装调试符号

3. 输出选项（Exports）

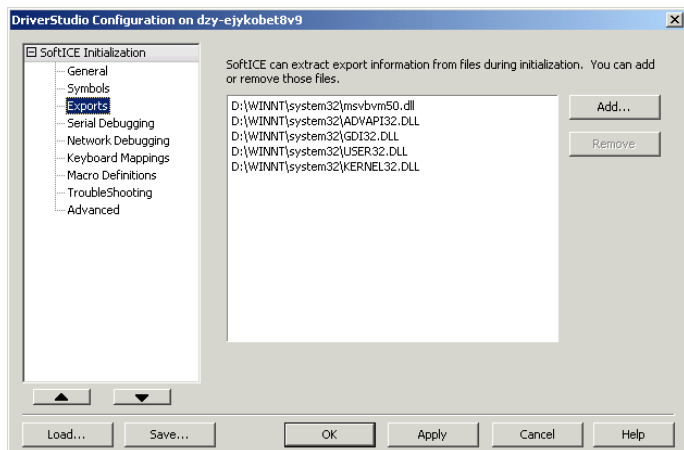


图 2.58 预装 DLL 信息

SoftICE 默认时加载 KERNEL32.DLL, USER32.DLL 和 GDI32.DLL 的输出函数表，也可通过符号预装加载更多的出口函数列表（见图 2.58）。当需要预装输出 DLL 时，单击“Add”按钮，在弹出的对话框中选择相应的 DLL 文件。反之，选择列表框中的文件，然后单击“Remove”按钮删除。

只有装载了相应的 DLL 文件，SoftICE 才能在其输出的函数中设置断点。在调试窗口下可以用“EXP 函数名”命令查看函数，此处函数可以是函数名的前缀，SoftICE 自动列出相近的函数。例如，exp message 将把所有的 message 前缀的函数列出。

只有将 advapi32.dll, comctl32.dll, comdlg32.dll, gdi32.dll, kernel32.dll, msbvm(50/60).dll, msvcrt.dll, oleaut32.dll, shell32.dll, user32.dll 等 DLL 文件通过 exports 选项配置好，才能设断点拦截相关 DLL 中的函数。

4. 功能键设定 (Keyboard Mappings)

此处可以设置 SoftICE 的功能键, 也可自定义功能键 (见图 2.59)。热键 “Alt” 缩写成 “A”, “Ctrl” 缩写成 “C”, “Shift” 缩写成 “S”。格式: 热键=^命令, 比如 “AF1” 表示 “Alt+F1”, “CF2” 表示 “Ctrl+F2”。设置时单击 “Add” 或 “Edit” 按钮, 并通过打开的对话框直接设置各功能键。

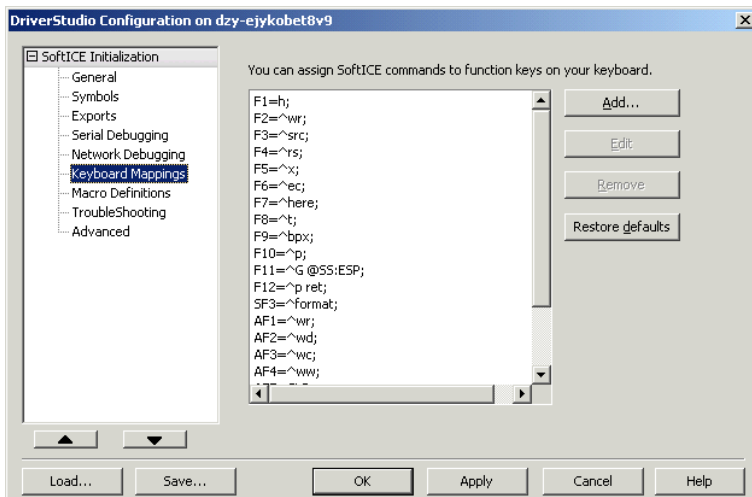


图 2.59 定义功能键

5. 宏定义 (Macro Definitions)

可用宏定义来方便使用 SoftICE, 如图 2.60 所示。宏定义有两种方法: 一种是在 SoftICE 启动后, 运行时用命令来定义; 另一种是在初始化设置中定义, 这样在 SoftICE 启动时自动加载宏定义信息。

可按如下方法定义宏。

- 单击 “Add” 按钮, 弹出添加宏定义窗口。
- 在名字域中填入宏的名字。注意宏的名字长度为 3~8 个字符, 可以包括字母、数字和下划线, 但必须包含一个字母字符, 而且不能与 SoftICE 中已有命令冲突。
- 在定义域中填入宏的定义。宏定义可以包含 SoftICE 的命令和其他的宏, 它们之间用分号隔离。必须用分号结束最后一个命令。传递命令行参数可在宏的定义体中任何地方使用 %number 的语法来引用, 其中 number 是一个 1~8 的数字。假设定义了一个宏: uasm=“u %1”, 这个宏表示反汇编命令。%1 将被 uasm 后的命令行中的第一个参数代替, 或者当没有参数时简单地被删除掉。需要在宏定义体中使用分号或者百分号时, 可用 “\;” 或者 “\%” 来表示, 也就是说, 在它们的前面加一个反斜杠 “\”。需要在宏定义体中使用反斜杠时, 可用两个反斜杠 “\\” 表示。
- 单击 “OK” 按钮添加到列表。

宏的使用: 在 SoftICE 下输入宏的名字即执行, 按 “Esc” 键终止宏的执行。

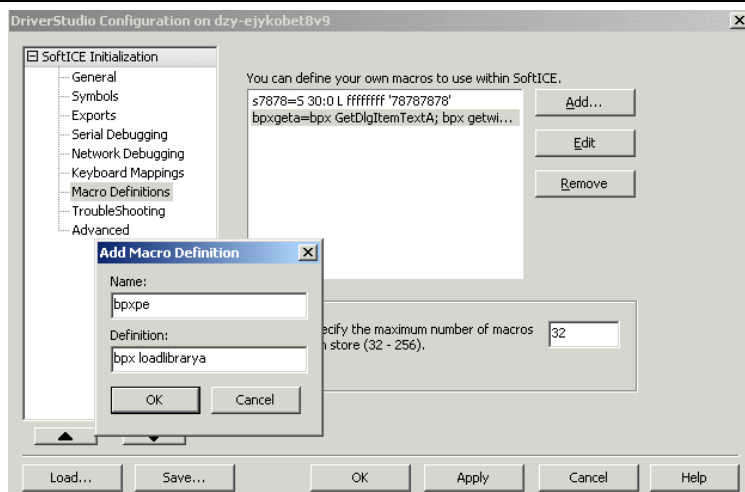


图 2.60 自定义宏命令

6. Troubleshooting

这个选项卡允许用户进行一些高级控制，这些控制的更改必须遵照 NuMega 公司的技术标准。通常，所有这些选项都是关闭的，如图 2.61 所示。

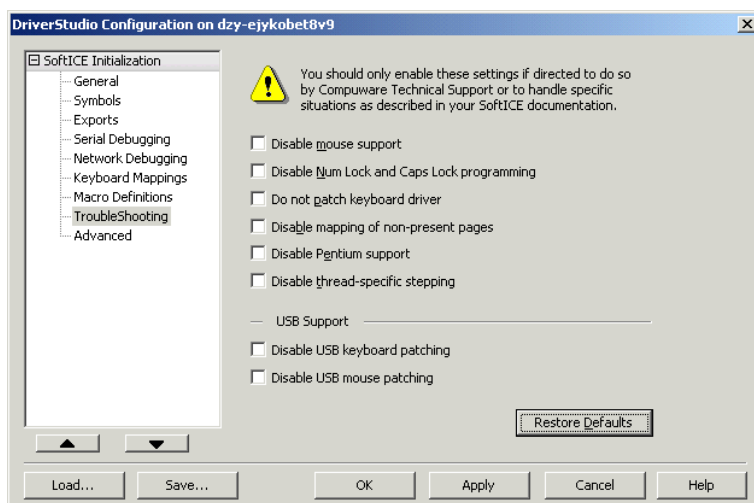


图 2.61 自定义宏命令

- **Disable Mouse support:** 如果鼠标和 SoftICE 不兼容，可以通过此选项关闭鼠标。
- **Disable Num Lock and Caps Lock programming:** 如果键盘和 SoftICE 不兼容，可以选择此项。
- **Do Not Patch Keyboard Driver (仅 Windows NT/2000/XP):** 当装载 SoftICE 后，键盘被锁或出现异常，选择此项。如果问题依旧，请选择“Disable Num Lock and Caps Lock programming”项。
- **Disable mapping of non-present pages:** 即使页表项标明不在内存中，SoftICE 也会试图在物理内存中寻找页面。选择该项可以关闭 SoftICE 的这个特性。

- **Disable pentium support:** SoftICE 自动检查是否使用 Pentium 处理器。如果正在使用 SoftICE 不熟悉的一个新型 CPU, 那么 SoftICE 就会错误地认为你正在使用 Pentium 处理器, 可在此关掉对 Pentium 的支持。
- **Disable thread-specific stepping:** P 单步执行指令, 此命令是线程敏感的。P 命令设置的返回断点只是针对当前活动线程。

7. Advanced

SoftICE 允许自定义调试窗口中的鼠标右键菜单, 菜单可以在 Advanced 选项里编辑修改 (见图 2.62)。

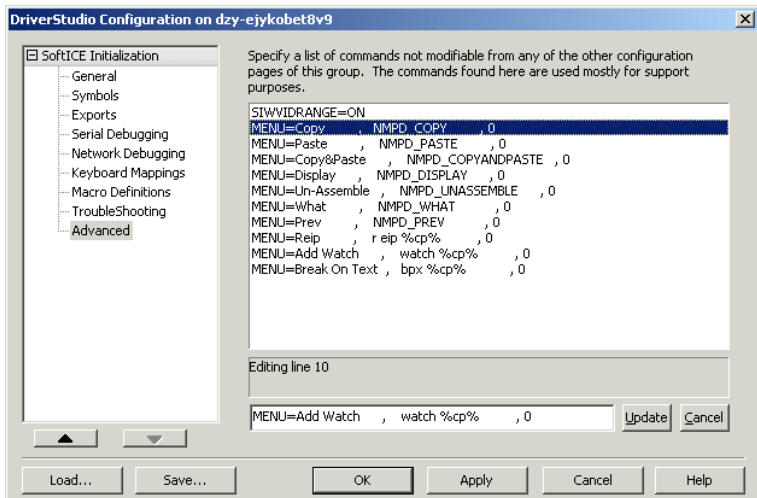


图 2.62 自定义鼠标右键菜单

菜单的格式为 MENU=Description,Command Field,[Modifier], 其中:

- **Description** 是菜单名, 可以包括字母、数字和空格, 字符长度不能超过 13;
- **Command Field** 是 SoftICE 各种命令、宏、表达式或预定义命令, 必须用完全命令, 不能用快捷键。符号 “%cp%” 指的是复制光标当前数据或文本到指定命令处。NMPD_COPY, NMPD_PASTE, NMPD_DISPLAY 等是 SoftICE 预定义命令。

例如, MENU=Break On Text, bpx %cp%, 0 表示光标移到某指令处, 执行后, 将所选数据复制到 bpx 后, 然后执行此命令。

8. Serial Debugging和Network Debugging

可用 SoftICE 通过网卡、串口线、MODEM, Internet 网进行远程调试。例如, 若用 MODEM 远程调试, 可在菜单中设置串口号、拨号号码以及拨号和应答的初始化命令字符串等。

9. Winice.dat配置

SoftICE 初始化设置 (SoftICE Initialization Settings) 的各项配置最后都保存于文件 Winice.dat 中。

- **Windows 9x 系统:** 若是 SoftICE 4.05 以前的版本, Winice.dat 位于 SoftICE 安装目录。


DriverStudio 内置的 SoftICE, Winice.dat 位于 Windows 目录, 如 c:\Windows。

- Windows 2000/XP 系统: Winice.dat 位于系统目录下的 system32\drivers 里。

可用任何文本编辑软件打开 winice.dat。光盘:\chap04 提供了 winice.dat 样例。在此列出 Windows 2000/XP 系统上的 winice.dat 的部分内容:

```
PENTIUM=ON                ; 支持奔腾指令
NMI=OFF                    ; 支持非屏蔽中断
LOWERCASE=OFF              ; 关闭小写反汇编
HST=512                    ; 历史缓存区大小, 单位是 KB
DRAWSIZE=2048              ; UVD 使用的显存大小, 单位是 KB, 默认 2 MB
SYM=512                    ; 符号缓存
; 系统默认已装载 KERNEL32.DLL, USER32.DLL, GDI32.DLL 三个文件
EXP=\SystemRoot\system32\NTDLL.DLL
EXP=\SystemRoot\system32\ADVAPI32.DLL          ; 拦截注册表相关 API 函数
INIT="Faults off;set font 2;lines 40 (更多的见光盘);x;" ; 初始化命令
;-----
;以下是 SoftICE 预定义的功能组合键
F1="h;"
F2="PAGEIN B ProcDump32 - Dumper Server;"      ; Nticedump 命令, 脱壳用
.....
;-----
;以下是宏命令, 根据需要自定义的
MACRO argesp8  ="D esp->8 ; p ret"
MACRO argespc  ="D esp->c ; p ret"
MACRO mon      ="Set Mouse on"                  ; 打开鼠标
MACRO moff     ="Set Mouse off"                 ; 关闭鼠标
MACRO bpxgeta  ="bpx GetDlgItemTextA DO \"argesp8\"; bpx GetWindowTextA DO \"argesp8\""; 拦截字符
MACRO bpxgetw  ="bpx GetDlgItemTextW DO \"argesp8\"; bpx GetWindowTextW DO \"argesp8\"
MACRO bpxma    ="bpx MessageBoxA ;bpx MessageBoxExA ;bpx MessageBeep DO \"p ret\""; 拦截窗口
MACRO bpxmw    ="bpx MessageBoxExW ;bpx MessageBeep DO \"p ret\"
MACRO bpxdiaa  ="bpx DialogBoxParamA ; bpx CreateWindowExA ; bpx ShowWindow " ; 拦截对话框
MACRO bpxdiaw  ="bpx DialogBoxParamW ; bpx CreateWindowExW ; bpx ShowWindow "
MACRO bpxrega  ="bpx RegQueryValueExA IF *(ESP->8) == '%1' DO \" D ESP->14 \"; 拦截读注册表
MACRO bpxregw  ="bpx RegQueryValueExW IF *(ESP->8) == '%1' DO \" D ESP->14 \";
MACRO regqva   ="BPX RegQueryValueExA IF(*(ESP->8)=='%1')||(ESP->8)=='%2'DO \"argesp14 \"
MACRO regqvw   ="BPX RegQueryValueExW IF(*(ESP->8)=='%1')||(ESP->8)=='%2'DO \"D ESP->14 \"
MACRO bpxpe    ="bpx LoadLibraryA DO \" D ESP->4 \"
MACRO bpxfi    ="bpx CreateFileA DO \" D ESP->4 \"
MACRO lbpm     ="bpm %1 x do \"bc bindex\" " ; 一次性断点
MACRO bpxseh   ="bpx KiUserExceptionDispatcher" ; 对系统异常处理代码设断
```

```
;-----  
;其他一些 DLL, 按需要配置装载, 装载时只须去除前面的分号即可  
; ***** Examples of export symbols that can be included *****  
; EXP=\SystemRoot\system32\msvbvm50.dll ; 装载 VB5 运行库  
; EXP=\SystemRoot\system32\msvbvm60.dll ; 装载 VB6 运行库  
; EXP=\SystemRoot\System32\hal.dll  
; EXP=\SystemRoot\System32\ntdll.dll  
; EXP=\SystemRoot\System32\csrssrv.dll
```

 **注意：**如果设置了断点，SoftICE 提示“Symbol not Defined (xxxx)”，此时打开 Winice.dat 配置文件，检查 API 所在的 DLL 是否装载。


2.2.5 SoftICE 常用命令简介

由于 SoftICE 命令较多，在此介绍几个常用的命令，其他的请参考附录的“SoftICE 手册”。

1. G 命令

语法：G [中断地址]

作用：执行程序，如果后面加地址，则执行到该地址为止；该命令属于一次性断点。F7 功能键有点类似 G 命令，也是一次性断点。

 **注意：**SoftICE 中的 G 命令必须是在当前实例的虚拟空间里，即必须在当前应用程序领空里设断，这时 EIP 为指定值才中断。

2. P 命令

语法：P [ret]

作用：单步执行程序。

只执行 P 时，相当于按下 F10 键。在汇编模式中，当遇到 CALL, INT, LOOP 或 REP 指令时，P 将不跟踪进去，直到这些指令执行完毕，控制才返回 SoftICE。换句话说，P 命令是“跨”过这些指令的。

“P RET”命令相当于快捷键 F12。SoftICE 将一直单步执行，直至找到一条返回语句(RET, RETF)。也就是说，让 SoftICE 一直执行代码，直到出现 RET (xxxx) 命令，再跳出来拦截。这时，当前 EIP 停在 RET (xxxx) 后的某一条语句上，通常是在某一个 CALL xxxxxxxx 后面。由于经常用 SoftICE 在某些底层的 Windows 函数上设置断点，所以 F12 功能键是很管用的。因为程序的作者用的是高级语言，Windows 又是提倡“透明”，不希望程序员知道底层的操作，而只提供给他们高层的接口，而相当多的高级函数调用某个一定的底层函数，所以当在底层函数上设置断点后，再用 F12 功能键，就可以知道程序员用的是什么函数了。

3. T 命令


语法：T [=start-address] [count]

start-address: 执行起始地址。

count: 指定 SoftICE 将单步跟踪多少次才停止。

作用: 单步跟踪。

T 命令相当于功能键 F8。如果没有指定起始地址, 将从 CS:EIP 指向的指令开始执行, 若遇到 CALL, LOOP 等指令, T 将跟踪进去。

 **注意:** F8 和 F10 功能键的主要差别就在于若遇到 CALL, LOOP 等指令, F10 是路过, 而 F8 是跟进去。

4. BPX命令

语法: BPX [address]

作用: 在可执行语句上设置 (或清除) 断点。

BPX 用来在指令处设置断点。程序一旦执行到此, SoftICE 就会弹出。当光标在代码窗口中时, 直接键入 BPX 就会在光标所在语句处设置断点, 再键入 BPX 就取消。BPX 的快捷键是 F9, 当光标在代码窗口中时, 按 F9 键就是设定 (取消)。另一种方法是用鼠标在代码上双击来设置断点。BPX 也可以用函数名来做地址参数, 格式为: BPX 函数名。

5. BL命令

语法: BL

作用: 显示当前所设的断点。

 **注意:** SoftICE 断点编号从 0 开始。

6. BC命令

语法: BC list | *

list: 可以清除指定编号的断点, 多个断点时中间用空格或逗号隔开。

* : 清除所有的断点。

作用: 清除一个或多个断点。

7. BD命令

语法: BD list | *

list: 可以是单个断点, 也可以是一系列断点, 中间用空格或逗号隔开。

* : 禁止所有的断点。

作用: 使一个或多个断点失效。

8. BE命令

语法: BE list | *

作用: 使一个或多个断点恢复有效。

用来恢复上次用 BD 命令使之失效的断点 (每当新定义断点或编辑断点时, 系统自动将其置为有效)。

9. BH命令

语法: BH

作用: 在 SoftICE 历史数据库中显示或选择曾经设过的断点。

用 BH 命令后, SoftICE 将显示一个表, 每一行是一个断点, 都是使用者曾经设过的断点。这时可用上下光标键来定位, 用 Insert 键选择, 再用 Enter 键来确定, 用 Esc 键取消。SoftICE 只记录最近的 32 个断点, 而且只在 Windows 正常退出后记录 (记录到 WINICE.BRK 中)。

10. BPE命令

语法: BPE index_number (断点索引号)

作用: 编辑一个已经存在的断点。

11. R命令

语法: R 寄存器名

作用: 显示或更改寄存器的内容。

它可更改所有寄存器的值。此命令较常用的一个功能是更改状态寄存器 (PSW) 的值, 格式为 “R FL 标志位”。比如, 当前 Z 标志位 (零位) 为置位状态, 执行 “r fl z” 之后会被清除; 如果 C 标志位为清除状态, 那么 “r fl c” 将使之置位。也可以在寄存器窗口中用鼠标直接单击相应的标志位, 然后用键盘上的 “Insert” 键来改变其状态。


 **注意:** 利用此命令可以很方便地和一些跳转指令上改变执行方向。

12. A命令

语法: A[地址]

作用: 进入小汇编状态, 可直接写入汇编代码。

如果不加地址值, 直接在当前 CS:EIP 处汇编。用 SoftICE 内置的汇编器在内存中写入汇编代码。汇编器支持标准的 80x86 指令集, 包括 386, 486, Pentium, Pentium-Pro 及 MMX 协处理器。新版的 SoftICE 还支持 AMD 的 3D Now!, PII, PIII 和 P4 等指令集。

 **注意:** A 命令只是修改了内存中的数据, 磁盘文件数据保持原样。要修改磁盘文件数据, 请用 Hiew 等十六进制工具操作。另外, A 命令有“记忆”效应, 被修改的指令只有在文件被修改或系统重启后才能恢复正常或用 A 命令改回。

13. D命令

语法: D[size] [address [l length]]

size: B 字节, W 字, D 双字, S 短实型, L 长实型或 T 10b 长实型。

作用: 显示某内存区域的内容。

14. S 命令

语法: S [address L length data-list]

address: 搜索的起始地址。

length : 搜索的长度 (字节数)。

data-lis: 可以是一系列字节, 也可以是字符串; 字符串可用单引号, 也可以用双引号括住。

作用: 在内存中搜寻特定数据, 如果找到数据, 将在数据窗口中显示出来; 如果找到后, 还要继续查找, 使用不带参数的 S 命令。由于 S 命令忽略不在内存中的页面, 可以使用 32 位平面地址数据段描述符 (Windows 9x 是 30h, Windows NT/2000/XP 是 10h) 在整个 4GB 空间内查找。例如:

S 30:0 L FFFFFFFF '78787878' //在 Win9x 系统内存中查找字符串 “78787878” 位置

15. E 命令

语法: E[size] [address [data-list]]

作用: 修改内存单元。

16. U 命令


语法: U [address [l length]] | [symbol-name]

address : 段, 偏移量或选择符。

symbol-name : 将从指定的函数开始反汇编。

length : 反汇编的长度 (字节数)。

作用: 反汇编指令。例如, U Messageboxa; U 401000。

 **技巧:** 可以利用此命令抓取汇编代码。运行 Symbol Loader 后, 将历史缓冲区 (history buffer) 调大些 (默认为 256, 不能存放足够多的缓冲数据); 然后切换到 SoftICE 调试画面下, 转到需要抓取的地方, 反汇编这些代码, 比如: U CS:EIP L 100; 立即按 Ctrl+D 返回到 Windows 环境, 再次转到 Symbol Loader 程序, 选择 File/Save SoftICE History As..., 反汇编代码保存在 *.log 文件里。若屏幕太杂乱, U 命令之前可以用 CLS 命令清屏。另外, 用 “D 地址 L 长度” 可得到数据窗口的数据。

17. FAULTS 命令

语法: FAULTS [on | off]

作用: 打开或关闭错误跟踪功能。

作为一个调试器, 由于 SoftICE 的 FAULTS 默认为 ON, 所以一旦 CPU 有非法指令, SoftICE 就会不停地弹出。此时用 “FAULTS off” 命令关闭错误功能。

 **注意:** 初学者一般不知道有这个命令, 一旦发生非法指令, 只会按 R 键。

18. ? 命令

语法: ? 表达式

作用: 计算一个表达式的值。

是一个非常高级的计算器, 可以很方便地在各种数制之间查看。

19. . 命令

语法: .

作用: 在代码窗口中定位当前指令。


当在代码窗口中上下浏览时, 有可能走得很远。一个“.”命令会在下一瞬间回到 SoftICE 当前所在的 CS:EIP 处。

20. EXP命令

语法: EXP [函数名]

作用: 显示 DLL 中的出口函数。

函数名可以指定其前几个字符, 可以用“?”来替代不定字符。这样, 就可查找相关函数以及它是哪个 DLL 文件。

 **注意:** 当记不清函数命名时, 可用此命令查询。例如, exp message 就可列出 message 前缀的函数。

21. WATCH命令

语法: watch 表达式

作用: 该命令可以设定内存和寄存器监视, 可以在所有时间监视某个数据。

表达式可以是符号、寄存器、内存数据和你希望的任何东西。每次 SoftICE 弹出时, 表达式的当前值和表达式类型 (SoftICE 无法确定大小时用 DWORD) 就会在监视窗口中显示出来。想删除一个 watch 变量, 按 Del 键删除。使用 WW 命令来开启/关闭监视窗口。例如:

watch eax; watch 401000+edi

2.2.6 SoftICE调试技术

建议读者自编一些应用程序开始练习。跟踪时注意 API 函数是如何调用的, 参数是如何传递的, 内存变量如何存放等。

在这里, 以一个用 VC 编写的小程序 TraceMe 来讲解 SoftICE 的调试技术, 编译时优化选项按默认设置为“Maximize speed”。读者也可以按“Minimize Size”优化选项编译一下, 并与本文比较。

拆解一个 Windows 程序要比拆解一个 DOS 程序困难得多, 因为在 Windows 中, 只要 API 函数被使用, 想对寻找蛛丝马迹的人隐藏是比较困难的。先简单地看一下 TraceMe 的序列号验证流程, 如图 2.63 所示。

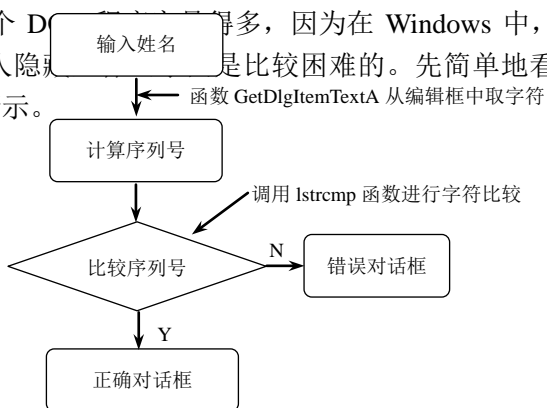
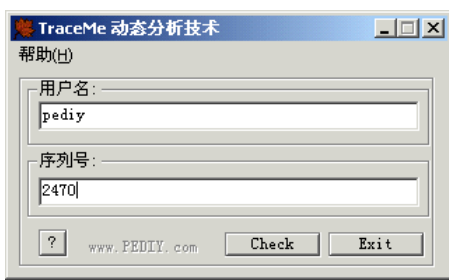


图 2.63 TraceMe 程序序列号验证过程


输入姓名与序列号到文字框中，程序调用 `GetDlgItemTextA` 函数把字符读出来，然后进行计算，最后用函数 `Lstrcmp` 进行比较。因此，这些调用的函数就是解密跟踪的目标，用这些函数作为断点，跟踪程序的序列号验证过程就能找出正确的序列号。

1. 捕捉字符

字符通常利用 Windows 文字框输入。为了检查输入的密码，程序常采用下面这些函数把文字框中的内容读出来：

16 位	32 位 (ANSI 版)	32 位 (Unicode 版)
<code>GetDlgItemText</code>	<code>GetDlgItemTextA</code>	<code>GetDlgItemTextW</code>
<code>GetWindowText</code>	<code>GetWindowTextA</code>	<code>GetWindowTextW</code>

一般事先不会知道程序具体是调用了什么函数来处理字符的，只好多试几遍，找出相关的函数。也可先静态分析，找出可疑的函数，再动态调试。

 **注意：**在 Windows 9x 系统下也可用万能断点函数 `Hmemcpy`。由于它工作在系统底层，大部分函数都会调用此函数来复制字符。本书以 Windows NT/2000/XP 为讲解主平台，用其他相关的 API 函数设断。

首先需要在 SoftICE 中设定一个“陷阱”（或称断点）。按“`Ctrl+D`”键进入调试状态，然后用命令 `BPX`，后跟函数的名字或者内存地址。因为这个 `TraceMe` 是 32 位 ANSI 版的程序，所以在 `GetDlgItemTextA` 处设一个断点。如果这个方法不行，可以再试其他方法。

像这样在 SoftICE 中输入：

```
:bpx GetDlgItemTextA
```

此时若 SoftICE 提示“`Symbol not Defined (xxxx)`”，请打开 `Winice.dat` 文件，将所有 DLL 前的分号去除（确保 `KERNEL32.DLL`，`USER32.DLL`，`GDI32.DLL` 三个 DLL 都被装载），然后重新启动。


好了，断点设置成功了。下一步可以列出所有断点来检查一下：

```
:bl
```


此时会看到这样的信息：

```
00) BPX USER32! GetDlgItemTextA
```

再按一次“`Ctrl+D`”键，从 SoftICE 中退出。

 **注意：**有些情况下，设置断点后，由于系统中其他进程频繁调用 `GetDlgItemTextA` 函数，因此 SoftICE 被激活，不能返回到 Windows 环境。此时查看 SoftICE 当前领空，将干扰的应用程序关闭或用其他函数设断。


现在已经设定了断点，可以捕捉任何对 `GetDlgItemTextA` 函数的调用。然后输入姓名和序列号，如姓名为“`pediy`”，序列号为“`1212`”。按下“`Check`”按钮，屏幕一闪，SoftICE 窗口弹出，程序中断在 SoftICE 中，就在函数 `GetDlgItemTextA` 开始的地方。

 **注意**：本书提供的 Winice.dat 文件中自定义了一个宏 MACRO bpxgeta="bpx GetDlgItemTextA; bpx getwindowtexta"。可以直接用 bpxgeta 宏命令设一批断点。

2. 在调试器中

按 F12 键，回到调用函数的地方，此时代码窗口和数据窗口之间的一行（又称领空）如下：

```
-----TraceMe!.text+01B4-----
```

 **注意**：要理解 F12 键的作用，在此可以用 F10 键模拟 F12 键跟踪，直到遇到“RET 10”指令，返回到上一层代码。

来到 TraceMe 领空后，可以这样禁止一个断点：

```
: bd 0 (或 bd * 禁止所有断点)
```

以后想再开启它的话，可以这样：

```
: be 0 (或 be * 开启所有断点)
```

很多时候必须重复跟踪同一段代码，因此可以先设置一个断点。将光标移到 04011AE 一行，按 F9 键设置新的断点（或 bpx 4011AE），以方便反复跟踪调试。现在有一个问题，现在已知 4011AE 这个断点，是不是任何时候拦截 TraceMe 时都可以用“bpx 断点地址（如 bpx 4011AE）”设断呢？答案是否定的，因为在设置断点时必须处于“程序的领空下”才能为某个指定的程序设置断点。遇到这种情况可以利用 Symbol Loader（后面会讲解）先加载应用程序，再使用命令“bpx 4011AE”。或先用其他 API 函数来拦截程序，先来到程序的领空，再对此地址设置断点。

在 SoftICE 中执行到这一句：

```
:004011B0 push eax ; 文本缓冲区指针
```

输入命令：D eax

此时数据窗口中没什么有价值的东西，继续按 F10 键单步执行完下面一句：

```
:004011B4 call edi ; GetDlgItemTextA 函数取姓名
```

此时 GetDlgItemTextA 函数已将字符串取出，放到 eax 所指的地址里。数据窗口右边字符段显示出刚输入的字符“pediy”，如图 2.64 所示。

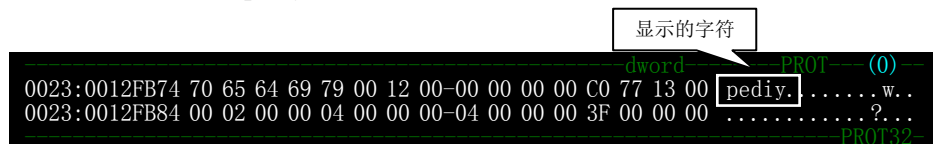



图 2.64 数据窗口查看字符

 **注意**：如果发现正在跟踪的代码段不是要跟踪的进程名，而是诸如 KERNEL32, GDI32, USER32 等常见的动态链接库，此时要按 F12 键多次直到来到跟踪的进程领空。因为无目的地跟踪上述动态链接库，除了浪费时间，不会带来任何好处。

2.2.7 SoftICE的符号调试技术

高级语言编译器都附带源代码级的调试器，如 Visual C++，Delphi 等。SoftICE 除了进行系统级调试外，也可在源代码状态下调试各种应用程序和设备驱动程序。

1. 生成NMS符号文件

除了可以直接从 PE 文件中提取出其输出表 (Export Table) 中的函数符号名之外，SoftICE 可以直接使用的调试符号文件只有 *.NMS。

得到可执行程序的 NMS 文件有如下途径。

(1) PDB 转 NMS

SoftICE 自带的 Symbol Loader 工具可将含有 PDB 调试信息的 EXE, DLL 等文件转换成 NMS 调试符号文件。

运行 SoftICE 中的 Symbol Loader 工具，单击菜单 “File/Open” 打开含有调试信息的 TraceMe.exe 文件，单击菜单 “Module/Setting” 打开设置对话框（见图 2.65），这些设置包括 General, Debugging, Translation 和 Modules and Files。

① General 选项

General 选项指定命令行参数和源文件搜索路径。

- **Command line:** 表示传递参数给加载的程序。
- **Source file search path:** 指出 Symbol Loader 用于定位加载该程序的源文件的搜索路径，如果 Symbol Loader 在该搜索路径上仍然找不到源文件，那么 Symbol Loader 就在 “Default source file path” 指定的路径中继续查找。
- **Default source file path:** 指出 Symbol Loader 查找源文件所使用的默认路径，这个设置是一个全局设置。
- **Prompt for missing source files:** 表示当 Symbol Loader 找不到源文件时是否给出提示，这个设置是全局的。
- **Minimize Loader on successful load:** 表示 Symbol Loader 成功加载 EXE 文件时是否自动最小化。

② Debugging 选项

Debugging 选项的作用是指定 Symbol Loader 工具加载文件的内容以及是否停在程序入口点上（见图 2.66）。

- **Load symbol information only:** 表示只加载 NMS 文件，而不加载执行文件映像。若在 Translation 选项卡中选择 “Symbols and source code”，那么还将加载源文件，默认情况下 Symbol Loader 为 DLL, SYS 和 VxD 文件类型选择该项设置。
- **Load executable:** 表示加载 NMS 文件和执行文件，如果在 Translation 选项卡中选择了 Symbols and source code，那么还将加载源文件，默认情况下 Symbol Loader 为 EXE 文件类型选择该项设置。



图 2.65 General 选项

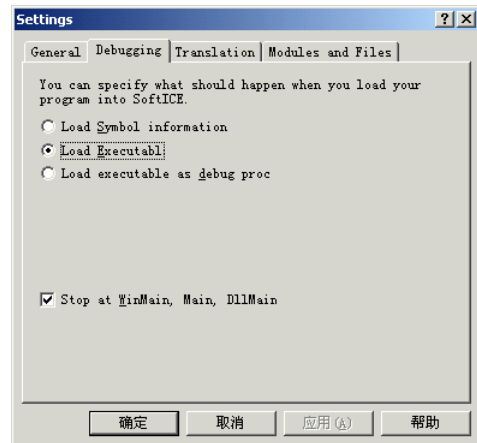


图 2.66 Debugging 选项

- Stop at Winmain main dllmain etc: 表示是否在程序的入口处创建一个断点。

③ Translation 选项

Translation 的作用是指定生成 NMS 文件的信息内容（见图 2.67）。

- Publics only: 只提供公用符号名，不包括类型信息和源代码。
- Type information only: 表示只提供类型信息，类型信息给出了数据结构信息。
- Symbols and sourc code: 表示提供所有有效的调试信息，包括源代码和行号信息。
- Package source with symbol table: 表示是否把源代码存储在符号文件 NMS 中，默认情况下设置此项。

④ Modules and Files 选项

选择加载文件所需要的符号文件或调试文件（见图 2.68）。

在 Translate 选项中选择“Symbols and source code”和“Packge source with symbol table”。单击菜单“Modul/Translate”命令，此时将加载符号文件和源文件。在默认情况下，会试图加载在调试信息中涉及的所有文件。因此，Symbol Loader 将提示不能定位源程序目录，TraceMe 是用 VC6 编译的，此时可以手工帮其定位到 VC6 安装目录\Microsoft Visual Studio\VC98\CRT\SRC 下。当然，也可以限制文件的加载，单击“Skip All”跳过所有源程序即可。经过转换就能生成一个 TraceMe.nms 的符号文件。

如果由 Winice.dat 来装载符号名，该符号文件的文件名不能超过 8 个字符，因为 SoftICE 是在实模式下启动的。

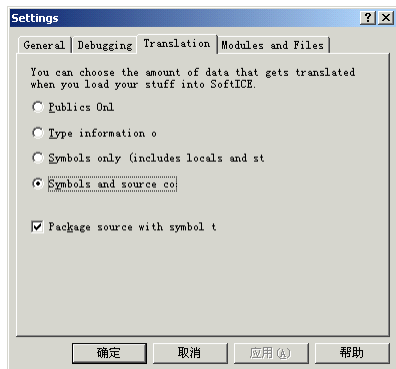


图 2.67 Translation 选项

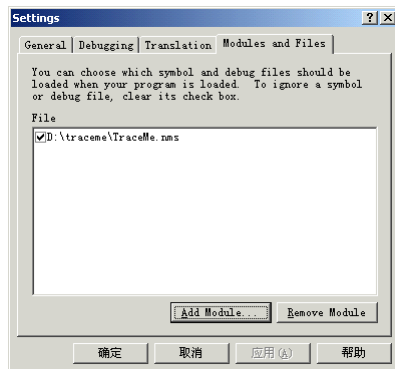


图 2.68 Modules and Files 选项

(2) SYM/DBG 转 NMS

用 Symbol Loader 工具也可将 SYM 或 DBG 调试符号转换成 NMS 调试符号，方法与转换 PDB 格式一样。

SoftICE 自带的命令行工具 Nmsym.exe 也可将*.SYM 转换成*.NMS，命令格式如下：

```
NMSYM [option(s)] <module-name>
      /TRANS(LATE): 此选项为符号转换
      [PUBLICS|TYPEINFO|SYMBOLS|*SOURCE*]
      [,PACKAGE|NOPACKAGE]
      [,ALWAYS]
```

例如：nmsym /translate:package,always myfile.sym

Nmsym.exe 功能强大，除了能转换符号外，还能装载程序。充分利用可以大大提高工作效率。

(3) MAP 转 NMS

如果没有可执行程序的源码，也没有*.DBG/*.PDB 文件，可以先用 IDA 对程序进行反汇编，IDA 可以识别出常用的库函数并生成*.MAP 文件（编译时，链接器 link.exe 和 tlink.exe 也可以生成 MAP 文件）。

SoftICE 的 Util16 目录中提供一个小程序 MSYM.EXE，可以将 MAP 转换成 SYM 文件，然后参考上面的方法，将 SYM 文件转换成 NMS 文件。

用法：MSYM file_name

Windows 9x DDK 开发包带有一命令行工具 Mapsym.exe，也可实现同样的功能。

用法：mapsym -ms file_name

2. 更新系统符号

每当升级安装了服务包后，记住要更新系统符号。SoftICE 每次升级都会支持当前最新 Windows 系统，包括系统调试符号。如果 Windows 推出 SP 之类的升级包，而 SoftICE 没升级，读者可以自己更新系统符号。

在开始菜单的 SoftICE 组里执行 Symbol Retriever 工具（见图 2.69），该工具可以连接到微软公司主页下载相应系统链接库的 DBG 文件。单击“Add File(s)”按钮选择系统文件，如

NTOSKRNL.EXE, HAL.DLL, KERNEL32.DLL 等, 单击 “Get symbols” 按钮将下载相应 DBG 文件。

- Translate To NMS After Download: 下载结束后, 将 DBG 格式转换成 NMS 格式;
- Load symbols into SoftICE: 自动将符号文件装载进 SoftICE;
- Clear Completed Files: 清除列表里已完成的符号文件;
- Display Download Path: 显示保存的路径。

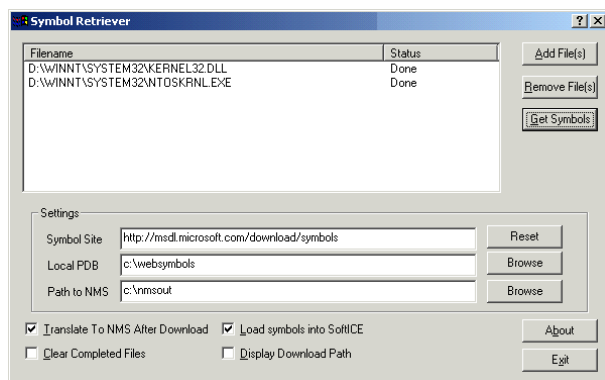


图 2.69 Symbol Retriever 工具

3. 符号文件的加载

符号装载有两种方法。一种是静态装载, 在初始化设置框 (SoftICE Initialization Settings) 中增加符号文件。此法要系统重新启动才能生效, 但可以调试驱动程序。

另一种是用 Symbol Load 动态装载, 方法是单击菜单 “File/Open” 打开符号文件, 单击菜单 “Module/Load” 加载符号文件。使用这种方法, 符号一装载, 立即生效, 但每次启动后必须重新加载。

Symbol Load 可以直接管理当前的符号文件, 如图 2.70 所示。

当有多个符号表装入内存的时候, 可以用 TABLE 命令显示符号表, 如下所示:

```
:TABLE
TraceMe [NM32]
0000493095 Bytes Of Symbol Memory Available
```

如果要用某个符号表中的符号, 必须先用 TABLE 命令选中, 如下所示:

```
:TABLE TraceMe
```

显示符号表中的符号可以用 SYM 命令, 格式如下:

```
SYM [区块名] 符号名
```

比如: sym .text。

显示 TraceMe 文件中的 .text 区块的所有符号。

此命令支持通配符, 可以用 “*” 代替末尾的字符, 用 “,” 代替某一个字符。

对于每个被加载的符号名, 可以直接在上面设置断点, 格式如下:

```
:bpx 符号名
```


在源码级调试时，如果装载符号和真实代码地址不符，可以用 SYMLOC 命令重定位符号基址。

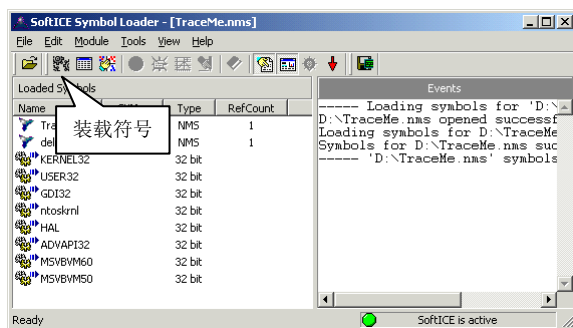


图 2.70 装载符号表

4. 用符号文件调试



图 2.71 符号调试技术

按上述要求设置好后，单击 Symbol Loader 的菜单“File/Open”来打开含有调试信息的 TraceMe 文件，单击菜单“Module/Load”加载文件。Symbol Loader 自动完成符号转换过程，

并将其装载。一旦装载成功,就会中断在程序的入口点。也可直接装载光盘提供的完整符号文件(TraceMe.nms)。这时,代码窗口显示的不再是汇编代码了,而是以源代码形式显示(见图 2.71)。

在 SoftICE 中输入如下命令可查看加载的源文件:

:File *

例如,要查看 VC 的启动文件 Crt0.c,输入下列命令(前提是符号转换定位源码时,不能单击“Skip All”按钮跳过)将会在代码窗口中看到该文件中的源码,按 F6 键将光标切换到代码窗口中,用上下光标键翻阅:

:File Crt0.c

现在对启动源码的 Getversion 函数设置断点。由于代码太长,可以用命令 SS 查找字符“Getversion”:

:ss 'Getversion'

找到 Getversion 函数后,按 F9 键或双击鼠标设置断点,关闭 TraceMe。再重新运行,将会中断在 Getversion 函数上,此时就可在源码级跟踪程序是如何调用 VC 启动函数的。

通过按 F3 键或用命令“SRC”,代码窗口将在源程序、反汇编代码、两者混合之间来回切换显示。

局部变量用 wl 命令查看。局部变量窗口给出了堆栈的变量和数据结构,当在局部窗口显示堆栈中的结构变量时,它会在结构体名字旁显示一个“+”,此时可以用鼠标双击展开“+”号,就会显示出结构体中各个变量的值。

全局变量用 watch 命令查看,格式如下:

:watch 表达式

也可用? 命令查看,格式如下:

:? 表达式


5. 装载外部函数

SoftICE 可提取 DLL 和 EXE 文件输出表中的函数符号名。

第一种方法是静态装载,在初始化设置框(SoftICE Initialization Settings)中增加 DLL 文件,系统重新启动才能生效。

第二种方法是用 Symbol Loader 动态装载,方法是单击菜单“Load/Exports”来打开要装载的 DLL 文件,该 DLL 输出函数立即生效,但每次启动后必须重新加载。

6. 中断在入口点上

可以用 Symbol Loader 装载普通发行版的 EXE 文件,以中断在程序 WinMain 处。首先在菜单“Module/Setting/Debugging”里,将“Stop at Winmain main dllmain etc”一项选上。单击菜单“File/Open”打开记事本程序(Notepad.exe),最后单击菜单“Module/Load”或按  按钮加载记事本程序。由于程序没有调试信息,SoftICE 会出现图 2.72 所示的提示窗口。

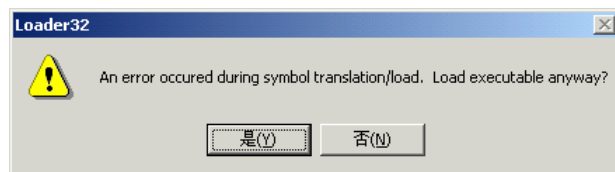


图 2.72 Symbol Loader 提示信息

可以不理睬这个警告提示，单击“是（Y）”按钮。屏幕一闪，SoftICE 将中断在记事本程序的入口点处。此时 SoftICE 可能停在一个 INVALID 代码处，表示数据未初始化，此时按 F10 或 F8 键就可显示出汇编代码。

DriverStudio 2.x 系列的 Symbol Loader 不能中断在 WinMain 处，有三种办法来解决这个问题。

（1）修补 Nmtrans.dll 文件

Nmtrans.dll 文件有个 bug，可以用工具 Nmtranspath 修补。在 DriverStudio v2.7 版本里的具体修补代码如下：

:1001CD7A 25FFFF0000	AND EAX, 0000FFFF	; 改成 MOV EAX, 0000FFFF
:1001CD7F 50	push eax	
:1001CD80 8B4DDA	mov ecx, dword ptr [ebp-26]	
:1001CD83 51	push ecx	
:1001CD84 8D55B0	lea edx, dword ptr [ebp-50]	
:1001CD87 52	push edx	

（2）更新系统符号

例如，在 Windows XP SP1 系统里用 DriverStudio v2.7，打 Nmtranspath 补丁后仍不能中断在入口点。可以用 Symbol Retriever 工具下载 Ntoskrnl.exe 的符号文件，并将其转换成 NMS 格式。

- ☐ 运行 Symbol loader，单击菜单“Edit/Softice Initialization Settings”打开配置窗口；
- ☐ 将刚生成的符号文件 Ntoskrnl.nms 增加到 Symbols 选项里；
- ☐ 在 Advanced 选项里，输入“NTSYMBOLS=ON”，并单击“Add”按钮；
- ☐ 重新启动。

（3）插入 INT3 指令

一些加壳的程序用 Symbol Loader 是不能中断在入口点处的，此时必须用插入 INT 3 断点的方法。INT 3 指令的机器码是“CC”。插入的方法很多。可以用十六进制工具，也可用专门的工具，如 LordPE 等。

现以光盘\chap04\int3\int3.exe 为例，具体步骤如下：

- ☐ 首先在 SoftICE 里设断点：bpint 3。
- ☐ 运行 LordPE，选上“Options/ Register shell extension”，即可将“Break 'n'Enter”添加到鼠标右键菜单上。

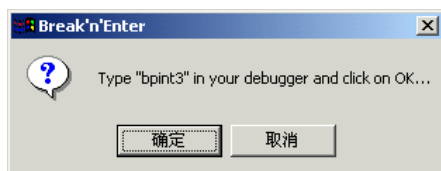


图 2.73 Break 'n'Enter

- ☐ 选中需要调试的程序，单击右键菜单执行“Break 'n' Enter (LordPE)”命令，出现图 2.73 所示的对话框。此时单击“确定”按钮，LordPE 将在执行文件入口点处插入一个“INT 3”指令并执行该程序。
- ☐ SoftICE 将中断在 INT 3 指令处，具体代码如表 2-12 所示。同时，LordPE 会将提示消息显示在 SoftICE 的命令窗口里：“Type " eb eip 60 ””，这句命令是还原被替换成 INT3 的机器码。

表 2-12 插入 INT3 指令前后的代码

原指令代码		在入口点处硬插 INT 3 指令	
60	PUSHAD	CC	INT 3
BE15B04000	MOV ESI,0040B015	BE15B04000	MOV ESI,0040B015
8DBEEB5FFFFFF	LEA EDI,[ESI+FFFF5FEB]	8DBEEB5FFFFFF	LEA EDI,[ESI+FFFF5FEB]
57	PUSH EDI	57	PUSH EDI

- ☐ 再用“eb eip 60”命令恢复程序的本来指令，然后就可按 F10 键进行单步跟踪了。

2.2.8 断点

可以用 SoftICE 在程序代码、内存、I/O 端口等处设置断点，SoftICE 断点编号从 0 到 FF，可以用断点编号进行删除、禁止、开启、编辑断点。所有的 SoftICE 断点命令(BPX, BPM, BPIO, BMSG 和 BPINT) 都接受如下条件断点格式：

breakpoint-command [breakpoint options] [IF conditional expression][DO “commands”]

- conditional expression: 条件表达式，符合标准的 C 语言逻辑算术语法；
- DO “commands”: 断点触发后调试器将执行的动作，这个动作包括用户定义的宏等命令。

1. 断点类型

(1) 代码定位断点

代码定位断点是最常用的断点，可以设置代码地址断点、函数名断点等。

语法：BPX [address] [IF expression] [DO “command1;command2;...”]

作用：在可执行语句上设置（或清除）断点。

用法：

address: 断点所在的线性地址；

IF expression: 条件表达式，只有条件为“真”时，SoftICE 才在断点处弹出；

Do command: 当 SoftICE 弹出时, 自动执行的一些命令。

BPX 实际上是在设置断点处插入了一个 INT 3 指令, 到这条指令时就弹出来。当然, 调试器已将插入的 INT3 指令隐藏起来了, 因此代码窗口里看不到。

用 **BPX** 可以在一个程序中设多个断点, 而不必使用少得可怜的调试寄存器。当在 ROM 中设断点时, SoftICE 自动用断点寄存器 (DRx) 来设置断点。

(2) 数据断点

调试过程中, 跟踪内存数据能极大地提高工作效率, 建议尽量使用数据断点。数据断点利用了 DR3~DR0 寄存器, 所以最多只能设四个断点。

作用: 在内存单元上设置断点。

语 法 : **BPM[B|W|D]** address [R|W|RW|X] [debug register] [IF expression][DO “command1;command2;...”]

用法:

BPM 和 **BPMB**: 内存单元为字节 (byte);

BPMW: 内存单元为字;

BPMD: 内存单元为双字;

R|W|RW|X: 所进行的操作, R 为读, W 为写, RW 为读写 (默认), X 为执行。

(3) 中断向量断点

作用: 在某个中断向量上设断点。

语法: **BPINT interrupt-number** [IF expression] [DO “command1;command2;...”]

用法:

interrupt-number: 中断向量号, 从 0 到 FFH。

(4) IO 端口断点

作用: 在输入输出 (IO) 端口处设置断点。

语法: **BPIO port-number** [R|W|RW] [IF expression][DO “command1;command2;...”]

用法:

port-number: 端口号。

(5) 消息断点

消息断点使得当某个特定窗口函数接收到某个特定消息时程序中断。

作用: 在 Windows 的消息上设断点。

语 法 : **BMSG window-handle** [L] [begin-message [end-message]][IF expression] [DO “command1;command2;...”]

用法:


window-handle: 消息发向的窗口句柄;

begin-message: 消息标识字的范围, 如果没有 **end-message**, 只有 **begin-message** 消息将被中断, 否则在区域内所有消息都会被中断断点。

Windows 本身是由消息驱动的, 所以跟踪一个消息会得到相当底层的答案。


要得到窗口句柄, 就需要 **HWND** 命令, 如下所示:

HWND [进程名]

 **注意：**在 Windows 9x 下，单独输入一个 HWND 命令将列出当前进程的所有句柄。在 Windows NT/2000/XP 下，必须进入当前应用程序领空执行，或 HWND[进程名]。

下面是一个具体示例。

- ☐ Windows 2000 上执行记事本程序 (Notepad);
- ☐ 按 “Ctrl+D” 键激活 SoftICE，输入：“HWND Notepad” 命令（或 proc 命令得到 PID 的值，再用 “HWND PID 值”），结果显示 “Edit” 子窗口的句柄是 060246;
- ☐ 再输入令：bmsg 060246 wm_char (Notepad 每次运行的句柄值都不一样);
- ☐ 回到 Notepad 中，随便键入一字符，SoftICE 就被激活了；原因在于在按键消息上设置了断点（退出 SoftICE 时别忘了输入命令 “BC *” 来清除刚才设的断点）。

 **注意：**在 Windows2000/XP 下，Driversuite v2.7 更高的版本可能存在 bug，有时用上述方法中断不下来。

如果要显示或查询 Windows 的消息函数，可以用如下 WMSG 命令：

WMSG wmsg [partial-name | message-number]

其中，partial name 参数可以是 Windows 消息的前几个字符，后面用星号 “*” 。例如：

WMSG wm_get*

2. 运算符

SoftICE 的运算符和 C/C++ 程序运算符有着紧密联系。SoftICE 支持的运算符有：

间接运算符	示 例
->	ebp->8（取得 ebp+8 所指向的双字节值）
.	eax.1c（取得 eax+1c 所指向的双字节值）
*	*eax（取得 eax 所指向的双字节值）
@	@eax（取得 eax 所指向的双字节值）

算术运算符	示 例
+（正号）	+42（十进制）
-（负号）	-42（十进制）
+（加法运算符）	eax + 1
-（减法运算符）	ebp - 4
*（乘法运算符）	ebx * 4
/（除法运算符）	Symbol / 2
%（模运算符或称求余运算符）	eax % 3
<<（逻辑左移）	bl << 1（逻辑左移一位）
>>（逻辑右移）	eax >> 2（逻辑右移二位）

位 运 算 符	示 例
---------	-----

& (AND, 按位与)	eax & F7
(OR, 按位或)	Symbol 4
^ (XOR, 按位异或)	ebx ^ 0xFF
~ (NOT, 取反)	~dx

逻辑运算符	示 例
! (逻辑非, NOT)	!eax
&& (逻辑与, AND)	eax && ebx
(逻辑或, OR)	eax ebx
== (等于)	Symbol == 4
!= (不等于)	Symbol != al
< (小于)	eax < 7
> (大于)	bx > cx
<= (小于或等于)	ebx <= Symbol
>= (大于或等于)	Symbol >= Symbol

特殊操作符	示 例
. (行号)	.123 (该值是当前源文件中 123 行的地址)
(,) (分组符号)	(eax+3) * 4 左括号和右括号的作用是用来将符号分组
, (参数列表)	函数名 (eax, ebx)
: (段操作符)	es:ebx
# (保护模式选择子)	#es:ebx (该地址是保护模式下的选择子: 偏移量的形式)
\$ (实模式段)	\$es:di (该地址是实模式下的段: 偏移量的形式)

3. 内置函数

SoftICE 的内置函数有如下这些。

(1) BPCOUNT

BPCOUNT 函数是计算当前断点表达式为“真”的次数。也就是说, 条件表达值是“真”, BPCOUNT 自动加 1, BPCOUNT 函数从 0 开始计数, 例如:


```
bpx myaddr IF (bpcount==5)
```

第五次触发到此断点时, BPCOUNT=5, 因此条件表达式的值是真, 断点满足, SoftICE 被激活, 此时 BPCOUNT 函数复位清零。

如果是在组合表达式中, BPCOUNT 函数需要放到右边, 例如:

```
bpx myaddr if (eax==1) && (bpcount==5)
```

如果经过此断点时 EAX=1, 则为“真”, BPCOUNT 自动加 1; 否则为“假”, BPCOUNT 不加, 函数 BPCOUNT 自动加 1。直到 EAX=1 为“真”5 次, BPCOUNT=5, 并且经过断点时的 EAX=1, SoftICE 才触发。

 **注意:** 不要把 BPCOUNT 函数放在表达式前面, 否则不能正确计数, 例如:

```
bpx myaddr if (bpcount==5) && (eax==1)
```

(2) BPMISS

BPMISS 函数是计算当前断点表达式为“假”的次数。也就是说，若条件表达值是“假”，BPMISS 自动加 1，BPMISS 函数从 0 开始计数，例如：

```
bpx myaddr if (eax==43) || (bpmis>=5)
```

如果表达式值中的 EAX=43 是“真”，整个条件就是“真”，SoftICE 触发；否则由于表达式每次是“假”，BPMISS 每次自动加 1，直到失败 5 次，BPMISS=5，并且条件为“真”时，SoftICE 才触发。

(3) BPTOTAL

每次触发断点一次，BPTOTAL 加 1，直到满足条件时中断。中断后其值不清零，继续自动加 1。

(4) BPLOG

BPLOG 表达式可把断点 log 记录到历史缓冲区中，SoftICE 不弹出，例如：

```
bpx myaddr if ((eax==1) && bplog)
```

每次经过断点时，若 EAX=1，SoftICE 记录信息到历史缓冲区中，SoftICE 不触发跳出。

4. 条件断点

在用 SoftICE 调试过程中，经常在 GetWindowText 和 GetDlgItemText 上设置一条 bpx 指令，并且“希望”程序调用该函数时 SoftICE 弹出。大多数情况下会这样，问题是当按下 F12 键时，常常迷失在代码里面。

(1) 读取字符

首先看一看 GetWindowText 函数，这个函数定义如下：

```
int GetWindowText(  
    HWND hWnd,           // 窗口或文本控件句柄  
    LPTSTR lpString,     // 缓存区地址  
    int nMaxCount        // 复制的最大字符数  
);
```

GetWindowText 采用标准调用约定，参数按从右到左的顺序入栈。因为在起始代码执行前，SoftICE 将弹出，EBP 堆栈结构还未建立，只得用 ESP 访问这些参数。运行光盘提供 FirstTarget，bpx GetWindowTextA 设断，单击“Check It”按钮，图 2.74 是当 SoftICE 弹出时从堆栈中看到的。

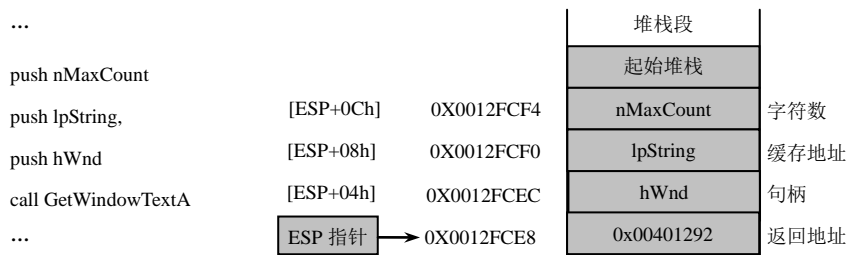


图 2.74 GetWindowTextA 函数参数入栈图

当函数返回时，GetWindowText 将把取出的文本放到由 lpString (LPTSTR 是一个长指针，指向由空字符终止的字符串) 指定的位置。因此，必须用 SoftICE 的间接运算符 (*)。例如，下列命令：

D * (esp+8) (在数据窗口显示由 “esp+8” 指定地址的内容)

也可用另一个快捷符：esp->8。可这样设断点：

bpx GetWindowTextA do "D esp->8;"

当运行该程序中断时，数据窗口中直接显示输入的字符，按一下 F12 键就返回程序领空中。当然也可这样设置断点，让它自动返回：

bpx GetWindowTextA do "D esp->8;p ret"

再看一看 GetDlgItemText 的定义：

```
UINT GetDlgItemText(
    HWND hDlg,           // 对话框句柄
    int nIDDlgItem,       // 控件标识 (ID 号)
    LPTSTR lpString,      // 文本缓冲区指针
    int nMaxCount         // 字符缓冲区的长度
);
```

惟一不同的是 nIDDlgItem，它是从文本那里获取的控制标识。堆栈如下：

```
[ESP+10h] - nMaxCount
[ESP+0Ch] - lpString    << 这里
[ESP+08h] - nIDDlgItem
[ESP+04h] - hwnd
[ESP+00h] - 返回的 EIP
```

然后设断点：

BPX getdlgitemtext DO "D esp->C;"

(2) 注册表操作

在注册表的 API 函数中设断点是很容易的。首先，必须确定 SoftICE 已经装载 ADVAPI32.DLL。运行光盘\chap01\Regtest\Regtest.exe，可如下设断点：

bpx RegQueryValueExA

单击“读取键值”按钮，SoftICE 将中断。一般的程序在这儿将会有几百次读注册表的调用。在此必须找一种方法，使 SoftICE 直接中断在相关的值上。

RegQueryValueEx 函数的定义：

```
LONG RegQueryValueEx(
    HKEY hKey,           // 查寻项的句柄
    LPTSTR lpValueName,  // 要获取值的名字
    LPDWORD lpReserved,  // 保留项，未用
    LPDWORD lpType,      // 装载取回值的类型的一个变量
    LPBYTE lpData,       // 数据缓存地址
    LPDWORD lpDataSize
```

```
LPDWORD lpcbData      // 数据缓存的大小
);
```

用 SoftICE 中断时，堆栈是这样的：

```
[ESP+18h] - lpcbData
[ESP+14h] - lpData      ; 这里是由窗口的装入返回数据的地方
[ESP+10h] - lpType
[ESP+0Ch] - lpReserved
[ESP+08h] - lpValueName ; 这里是将被取回的数据名字的地址
[ESP+04h] - hKey
[ESP+00h] - 返回的 EIP
```

在此可以设置一个动作断点 `do "d esp->14"` 来显示装入的数据。

例如，若 RegMon 注册表监视程序分析出 Regtest.exe 程序从注册表读取的信息：

HKEY_LOCAL_MACHINE\SOFTWARE\PEDIY\reguser

那么可以在 SoftICE 中设这样一个条件断点：

BPX RegQueryValueExA IF *(ESP->8) == 'Regu' DO "D ESP->14"

SoftICE 将只在读取注册表“Regu”键值时中断，并且显示堆栈[ESP+14]的值，即返回注册项的值。

- ESP->8 相当于[ESP+8]，*(ESP+8)是指[ESP+8]指定的地址中存储的值。
- 表达式*(ESP+8)=='Regi'表示 [ESP+8]指定的地址中存储的值等于“Regu”时为真。
“Regu”选用 4 个字符是因为 SoftICE 仅能返回 DWORD 值（32 位），所以只能用开始的 32 位，也就是前四个字符。

如果程序在两个地方读取字符串，怎么设置断点呢？例如：

HKEY_CURRENT_USER\Software\Applied Insights\AI Explorer\UN

HKEY_CURRENT_USER\Software\Applied Insights\AI Explorer\SN

可以在 SoftICE 中如此设断点：

BPX RegQueryValueExA IF (*(ESP->8) == 'UN') || (*(ESP->8) == 'SN') DO "D ESP->14;"

 **注意：**由于 SoftICE 命令行有字符限制，因此必须创建一个宏输入上述命令。

将下面的宏命令放到 Winice.dat 里：

```
MACRO regqva    ="BPX RegQueryValueExA IF(*(ESP->8)=='%1')||(*(ESP->8)=='%2')DO
\"D ESP->14\""
```

当需要时，用命令设断点：

```
:regqva un sn
```

此时，命令行由于字符数限制，设断后，后面部分不会显示，但命令仍有效。

下面是其他一些范例：

```
BPX loadlibraryA do "dd esp->4"      // 中断时执行命令：dd esp+4
```

```
BPX GetWindowTextA if EAX==00000008 // 当 EAX=8 时中断 GetWindowTextA
```

```
BPX GetWindowTextA do "x"           // 当中断时 SoftICE 自动回到 Windows 界面
```

BPX GetWindowTextA do "d EAX" // 当中断时，自动显示 EAX 的值

(3) 中断指定进程的 API 函数

有时，希望断点被指定进程调用时才触发 SoftICE。


例如，运行 Windows 98 的记事本程序 (Notepad)，切换到 SoftICE 下，用 PROC 命令查看 Notepad 进程 ID 号：

```
:proc Notepad (或直接输入 proc)
```

显示记事本程序的 PID=65C

```
:bpx MessageBoxA if (PID==65C)
```

在记事本编辑区内随便输入几个字符，关闭它，此时将跳出一个对话框提示是否保存文件，该对话框调用的是 MessageBoxA(W)函数，SoftICE 将马上中断。此时运行其他调用 MessageBoxA(W)函数的程序不会触发 SoftICE。

 **注意：**程序每次运行的进程 ID 值是不同的。

2.2.9 SoftICE 远程调试

SoftICE 远程调试按照连接方式分为 MODEM、网络和串口这 3 种远程调试。

1. 网络远程调试

远程 SoftICE 允许通过互联网进行远程控制，因而使调试工作更加容易和方便。Windows 9x, Windows NT/2000/XP 都支持远程 SoftICE。

(1) 远程 SoftICE 要求的配置

- 目标计算机必须运行 SoftICE；
- 目标计算机需要一块以太网卡，能连接上本地 IP 网络；
- 支持的以太网卡类型：NE2000 兼容网卡、3Com3C90X 网卡和 Intel E100；
- 远程控制的计算机必须和目标计算机网络互通。

(2) 设置 SoftICE

检查目标计算机是否使用 SoftICE 支持的网卡，然后在控制面板里用 SoftICE 自带的驱动程序文件 (\program files\compuware\driverstudio\softice\network\) 更新网卡自带的驱动程序文件。Windows 2000 上的安装过程为：

控制面板->网络拨号->本地连接->属性->配置->驱动程序->更新驱动程序

 **注意：**删除 SoftICE 前，必须先恢复原网卡驱动程序，操作与安装过程类似。

通用网卡驱动程序 (Universal Network Driver, UND) 是为工作在 Windows 2000/XP 系统上的 PCI 网卡准备的。使用 UND，主机可以是任何网卡，目标机必须是 PCI 接口网卡。

(3) 目标计算机的配置

网卡和驱动程序成功后，在 SoftICE 中执行 NET 命令，才允许对方的连接。

(4) NET 命令集

```
NET START <IP address|DHCP> [MASK=<subnet mask>] [GATEWAY=<IP address>]
```

```
NET ALLOW <IP address|ANY> [AUTO] [PASSWORD=<password>]
```

NET PING <IP address>

NET RESET

NET STOP

NET HELP

NET STATUS

NET START <IP address>[DHCP] [MASK=<subnet mask>] [GATEWAY=<IP address>]

(5) NET 命令讲解

- **NET START:** 为目标计算机指定 IP 地址, 并启动网络。

NET START <IP address>[DHCP] [MASK=<subnet mask>] [GATEWAY=<IP address>]

IP address	本机 IP 地址, 格式: xxx.xxx.xxx.xxx
DHCP	动态分配 IP 地址, 由网络上的 DHCP 服务器分配, 一般不常用
MASK	子网掩码。通常情形为 255.255.255.0
GATEWAY	网关。通俗说就是网络的出口

例如:

NET START 10.10.10.1

NET START 10.10.10.1 MASK=255.255.255.0 GATEWAY=10.10.10.10

- **NET ALLOW:** 允许拥有某个 IP 地址的计算机进行远程调试。

NET ALLOW <IP address>[ANY] [AUTO] [PASSWORD=<password>]

IP address	IP 地址
ANY	允许任何机器进行远程调试
AUTO	自动处理调试进程
PASSWORD	验证密码

例如:

NET ALLOW 10.10.10.2 AUTO

NET ALLOW 10.10.10.2 AUTO PASSWORD=123

- **NET PING:** 网络诊断使用, 用来进行网络测试。类似于 Windows 下的 ping 命令。

NET PING <IP address>

IP address	IP 地址, 一般是指本机的 IP 地址
------------	----------------------

例如:

NET PING 10.10.10.2

- **NET RESET:** 中断调试进程, 需要重新使用 NET ALLOW 命令指定 IP 地址。
- **NET STOP:** 中断调试进程, 中断网络, 需要使用 NET START 命令重新启动。
- **NET HELP:** 帮助命令。显示 NET 命令集及它们的语法。
- **NET STATUS:** 显示网络状态 (IP 地址、掩码及网关) 及连接状态。

(6) 建立远程调试进程

当目标计算机已经调试设置以后, 远程计算机就可使用 SINET 命令。

SINET 命令语法:

```
SINET <target IP address> [<password>]
```

target IP address 目标 IP 地址由目标计算机的 TCP/IP 协议中设置。如果目标计算机使用密码，在命令行；

password 指定密码（大小写自适应）

SINET 尝试和目标计算机连接。如果目标计算机有回应，SINET 需要通过远程计算机的密码验证（如果没有密码，就不需要输入）。如果远程计算机通过目标计算机的验证，SoftICE 将和目标计算机连接成功。出现一个控制台窗口来模拟 SoftICE 显示，对双方操作都是透明的。

目标和远程计算机键入 SoftICE 命令都会发生相应动作。远程计算机使用 Ctrl+D 作为中断热键，除非在目标计算机中重新定义。

中断远程调试进程的方法是在远程计算机中输入 Ctrl+Break，或者在目标计算机中使用 NET RESET 命令。

（7）应用实例

假设目标计算机的 IP 为 10.10.10.1，远程计算机的 IP 为 10.10.10.2。

① 在网卡和驱动程序安装成功后，首先需要配置目标计算机：

```
NET START 10.10.10.1
```

```
NET ALLOW 10.10.10.1 AUTO PASSWORD=123
```

```
NET PING 10.10.10.2
```

② 在远程计算机上执行如下命令：

```
SINET 10.10.10.1 123
```

③ 连接成功后，出现的信息是：

```
c:\nugega\ice4.05>SINET 10.10.10.1
```

```
Target is 10.10.10.1
```

```
Local host name is 'ddxia'
```

```
Initialized
```

```
Opened
```

```
Authenticating...O.K.
```

④ 在远程计算机中输入 Ctrl+Break，或者在目标计算机中使用 NET RESET 命令，使调试结束。

 **注意：**也可在 SoftICE 初始化设置（SoftICE Initialization Settings）里设置连接命令。

2. MODEM 远程调试

（1）简介

可以利用 MODEM 进行远程调试。这个功能非常适合于在离用户比较远的情况下进行程序调试。远程调试的时候，当地的计算机必须运行 SoftICE 和被调试的应用程序。远程的计算机就充当一台哑终端，为用户的 SoftICE 进程显示输出和接受键盘输入。SoftICE 在远程计算机上不支持鼠标。

(2) 硬件要求

- MODEM 可以执行标准的 AT 指令集, 比如 ATZ 和 ATDT, 以及返回标准结果码, 比如 RING 和 CONNECT。
- 需要 MODEM 执行可靠的纠错协议, 比如 V.42 或 MNP5, 因为 SoftICE 使用的通信协议不执行纠错。

(3) 建立连接


通过 MODEM 远程调试, 本地或远端都可进行拨号初始化连接。本地用户拨远端用户来建立连接, 按照如下步骤进行。

- ☐ 远端用户运行 SERIAL32.EXE。
- ☐ SoftICE 用户调用 DIAL 命令。
- ☐ 于是本地和远端建立起一个连接, 远端的用户就会处在 SoftICE 的监控下。
- ☐ 远端用户拨本地用户建立连接, 按照如下步骤进行:
- ☐ 本地用户调用 ANSWER 命令, 准备应答;
- ☐ 远程用户运行 SERIAL32.EXE。

于是远端和本地建立起一个连接, 远端的用户就会处在 SoftICE 的监控下。

(4) SERIAL32.EXE 使用说明

SERIAL32.EXE 是一个 DOS 程序, 为 SoftICE 充当一个哑终端, 并显示输出和接受键盘输入。在本地计算机上运行 SoftICE, 看到的屏幕正是 SERIAL32.EXE 所在计算机的屏幕显示。

 **注意:** 通过串口或 MODEM 建立连接, 都可使用 SERIAL32.EXE。

如果远端的计算机运行 MS-DOS, 老版本的 MS-DOS 中的 SERIAL 也是可以使用的(DOS 的 SERIAL.EXE 程序在 UTIL16 目录下)。

- SERIAL32.EXE 命令语法如下:

```
serial32 [-l=25|43|50|60] [<port name> [<baud rate>]]
```

- SERIAL.EXE 命令语法如下:

```
SERIAL.EXE [r] [ com-port] [ baud-rate] [I "init-string" ] [P number]
```

r	使用 r 选项, 是当远程计算机在 Windows NT 系统 DOS 窗口下运行的时候。选项可以忽略 FIFO 和重置波特率、停止位和奇偶位;
com-port	串口端口号, 1~4; 它必须指定;
baud-rate	是 SERIAL32.EXE 和 modem 通信的速率, 必须指定。它和在 SoftICE 连接中使用 DIAL 和 ANSWER 命令时键入的速率不必相同;
I	用于指定 modem 的初始化命令;
init-string	是初始化的 modem 命令;
P	指定一个电话号码。如果使用 P 选项指定一个号码, SERIAL 会拨这个号码试图和 SoftICE 用户建立连接, 需要注意的是在 SoftICE 中已经执行了 ANSWER

	命令;
number	是要拨的电话号码 (当 SERIAL 用于拨号模式时), 是 ATDT 拨号的号码。对于脉冲拨号, 把 P 字符作为号码的第一个数字 (这样做, 也许所有的 MODEM 都不能识别)。

当连接建立后, 利用 SERIAL32.EXE 可以看见和 SoftICE 相似的文字界面, 也能远程控制 SoftICE。例如, SERIAL 1 57000 p1-603-555-1212。

如果 P 选项没有指定, SERIAL 运行在应答模式中, 等待拨入。例如, SERIAL 1 57000。

(5) DIAL 命令

在 SoftICE 中运行 DIAL 命令向外拨号。远程用户必须在应答模式下运行 SERIAL32.EXE。命令语法如下:

DIAL [ON [com-port] [baud-rate] [I= init-string] [P= number]][OFF]	
ON	开始拨号;
OFF	终止一个远程连接;
Com-port	串口端口号, 1~4; 默认为 1;
Baud-rate	是 SERIAL32.EXE 和 modem 通信的速率。默认为 57000。它不需要和另一端运行 SERIAL32.EXE 时设的速率相同。如果 baud-rate 参数被指定, com-port 参数也要被指定;
I	用于指定 MODEM 的初始化命令;
init-string	是初始化的 MODEM 命令。如果在命令行没有设定初始化串, 那么这里就得设置, 或者在 SoftICE 初始化配置文件中指定;
P	指定一个电话号码。如果用 P 选项指定一个号码, SoftICE 会拨号试图和对方建立连接, 需要注意的是对方必须在 SERIAL32.EXE 配置为应答模式;
Number	是要拨的电话号码 (当 SERIAL 被用于拨号模式时), 是 ATDT 拨号的号码。对于脉冲拨号, 把 P 字符作为号码的第一个数字 (这样做, 也许所有的 MODEM 都不能识别)。如果在命令行没有设定号码, 那么这里就得设置, 或者在 SoftICE 初始化配置文件中指定。

例如, DIAL ON 1 57000 p=603-555-1212。

(6) ANSWER 命令

ANSWER 命令用来把 SoftICE 配置成能够自动应答另一台电脑使用 SERIAL32.EXE 所发出的呼叫。命令语法如下:


ANSWER [ON [com-port] [baud-rate] [I= init-string]][OFF]	
ON	设置为应答模式;
OFF	取消应答模式;
com-port	串口端口号, 1~4; 默认为 1;
baud-rate	是 SERIAL32.EXE 和 MODEM 通信的速率, 默认为 57000。它不需要和另一端运行 SERIAL32.EXE 时设的速率相同。如果 baud-rate 参数被指定, com-port 参数也要被指定;

init-string	是初始化的 MODEM 命令。如果在命令行没有设定初始化串, 那么这里就得设置, 或者在 SoftICE 初始化配置文件中指定。
-------------	--

例如, ANSWER ON 1 57000。

3. 串口连接 (仅 Windows 9x)

如果在 Windows 9x 下运行 SoftICE 来调试远程系统, 可以通过串口 (COM1, COM2, COM3 和 COM4) 进行通信。当调试远程系统时, 需要改变设置为 None; SERIAL CONNECTION 被默认为 None。

 **注意:** 如果在 Windows NT 上运行 SoftICE, SoftICE 自动确定串口连接。

- **DIAL** 初始化字符串

初始化字符串——MODEM 通过 DIAL 命令执行的参数, 比如 ATX0。

- **ANSWER** 初始化字符串

初始化字符串——MODEM 通过 ANSWER 命令执行的参数, 比如 ATX0。

2.2.10 在虚拟机里使用 SoftICE

现代处理器强大的能力使得虚拟机的使用逐渐广泛, 常见的虚拟机有 VMware 等。虚拟机的好处显而易见, 可以调试病毒、木马而不用担心其会破坏本机环境。

从 DriverSuite 3.1 和 VMware 4.0 开始, SoftICE 就能很好的在 VMware 中工作了, 不过, 还必须在虚拟机的配置文件 (*.vmx) 添加如下的字符串。

```
svga.maxFullscreenRefreshTick = "2"
```

2.2.11 IceExt 插件

IceExt 是一款配合 SoftICE 扩展其命令的操作工具, 并能有效地隐藏 SoftICE, 以防被一些软件发现被调试。

IceExt 从 0.70 开始, IceExt 由自己的装载器而不再是系统服务管理器负责载入。如果启动 IceExt 发生错误, 这次需要修改注册如下键值。

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NTice
```

新建 KDExtensions, 类型为 REG_SZ。设置 KDHeapSize, KDStackSize 键值大于或等于 0x08000。

设置好后, 如是 IceExt 0.70, 直接带参数运行 iceext -s 即可启动 SoftICE, 并支持如下扩展命令, 更多请用 !HELP 命令获得。

!BC	清除内存区块中所有的断点
!BL	列出内存区块中所有的断点
!BPR	在一个内存范围里下断点
!DUMP	抓取内存数据到磁盘
!DUMPSCREEN	抓取 SoftICE 屏幕数据以 RAW 格式保存到磁盘

!FPOS	显示文件指针
!INTSAVE	存储中断句柄
!INTREST	恢复中断句柄
!INTXCHG	置换中断句柄
!INTSHOW	显示内部中断句柄表
!LOADFILE	加载文件到内存
!PROTECT	隐藏 SoftICE，非常有用，防止其他程序检测到 SoftICE
!SUSPEND	挂起当前线程（仅 ring-3 线程）
!RESUME	唤醒被挂起的线程