DriverStudio™ and SoftICE™ Driver Suite Installation Guide

Release 2.7

Windows® 95
Windows® 98
Windows® Me
Windows NT®
Windows® 2000
Windows® XP

Part Number 11551

June, 2002

Restricted Rights Legend

Software License Agreement

This Software License Agreement is not applicable if Licensee has a valid Compuware License Agreement and has licensed this Software under a Compuware Product Schedule.

COMPUWARE CORPORATION ("COMPUWARE") IS WILLING TO LICENSE THIS SOFTWARE (SOFTWARE) TO YOU ONLY ON THE CONDITION THAT YOU ACCEPT ALL OF THE TERMS CONTAINED IN THIS AGREEMENT. BEFORE YOU CLICK ON THE "ACCEPT" BUTTON, CAREFULLY READ THE TERMS AND CONDITIONS OF THIS AGREEMENT. BY CLICKING ON THE "ACCEPT" BUTTON, YOU ARE CONSENTING TO BE BOUND BY THESE TERMS AND CONDITIONS. IF YOU DO NOT AGREE TO ALL OF THE TERMS AND CONDITIONS OF THIS AGREEMENT, THEN COMPUWARE IS UNWILLING TO LICENSE THE SOFTWARE TO YOU, IN WHICH EVENT YOU SHOULD CLICK THE "DO NOT ACCEPT" BUTTON TO DISCONTINUE THE INSTALL PROCESS AND CONTACT YOUR COMPUWARE SALES REPRESENTATIVE.

Software License Agreement

Please Read This License Carefully

You, either an individual or entity (Licensee), are purchasing a license (Agreement) to use this Compuware Corporation software, related user manuals (Documentation), and other materials provided hereunder (collectively, the Software). The Software is the property of Compuware Corporation (Compuware) and/or its licensors, is protected by intellectual property laws, and is provided to Licensee only under the license terms set forth below. This Agreement does not transfer title to the intellectual property contained in the Software. Compuware reserves all rights not expressly granted herein. By clicking on the "Accept" button and/or installing the Software indicates your acceptance of these terms. If you do not agree to all of the terms and conditions of this Agreement, do not install the Software and contact a Compuware sales representative.

Title and Proprietary Rights: Licensee acknowledges and agrees that the Software is proprietary to Compuware and/or its licensors, and is protected under the laws of the United States and other countries. Licensee further acknowledges and agrees that all rights, title and interest in and to the Software, including intellectual property rights, are and shall remain with Compuware and/or its licensors. Unauthorized reproduction or distribution is subject to civil and criminal penalties.

Evaluation Copy: This section shall apply **only** if the Software has been provided for Licensee's evaluation of the Software (Evaluation Copy). An Evaluation Copy is provided AS IS, with no warranties, express or implied, or maintenance service; for the sole and exclusive purpose of enabling Licensee to evaluate the Software. The Evaluation Copy will automatically time-out at the end of the evaluation period. If Licensee elects to continue to use the Software at the end of the evaluation period, Licensee must contact a Compuware representative.

Use Of The Software: Compuware grants Licensee the limited right to use the Software included in the package with this license, subject to the terms and conditions of this Agreement. Licensee agrees that the Software will be used solely for internal purposes. Licensee may not use the Software to offer data processing services to third parties, including but not limited to timesharing, facilities management, outsourcing or service bureau use, or any other third party commercial purpose or gain. Only one copy of the Software may be installed on a single computer at any one time unless:

(i) The Software is designed and intended by Compuware for use in a shared network client server environment, as set forth in the Documentation; and

(ii) Licensee agrees to provide technical or procedural methods to prevent use of the Software, even at different times, by anyone other than Licensee; and

(iii) Licensee has purchased a license for each individual user of the Software and/or for each computer that will have access to the Software. Any unauthorized use of this Software may cause termination of this Agreement.

Licensee may make one machine-readable copy of the Software for BACK UP PURPOSES ONLY. This copy shall display all proprietary notices, be labeled externally to show that it is the property of Compuware, and that its use is subject to this Agreement. Documentation may not be copied in whole or part. Licensee agrees to provide technical or procedural methods to prevent use of the Software by anyone other than Licensee, even at different times.

Licensee may not use, transfer, assign, export or in any way permit the Software to be used outside the country of purchase, unless authorized in writing by Compuware. Except as expressly provided in this Agreement, Licensee may not modify, reverse engineer, decompile, disassemble, distribute, sub-license, sell, rent, lease, give or in any way transfer the Software, by any means or in any medium, including telecommunications. Licensee will use its best efforts and take all reasonable steps to protect the Software from unauthorized use, copying or dissemination, and will retain all proprietary notices intact.

Maintenance Service: Licensee may subscribe to maintenance service on an annual basis by paying the maintenance fee then in effect. If Compuware provides maintenance service while connected to Licensee's system or network either remotely or in person, Compuware shall not be responsible for any damage or loss, including but not limited to server failure, data loss, virus/worm infection and network downtime. Licensee is responsible for maintaining adequate backup and virus intrusion software for its server, data and network systems.

Redistribution Rights of Device Driver Development Software: This section shall **only** apply if the Software is device driver development software, used by Licensee to develop application or device driver programs (User Software), as specified in the Documentation. The User Software may include run-time components (RTC's) that have been extracted by the Software from the library files of the Software, programs to remotely test the User Software, and compiled code examples. These RTCs, examples, and programs are specifically designated as redistributable in the Documentation. Licensee has a non-exclusive, royalty-free, restricted license to:

(i) Modify, compile, and distribute the driver code examples;

(ii) Distribute the remote testing program for testing purposes only;

(iii) Embed the RTCs and driver code examples in its User Software, in object code form only; and

(iv) Reproduce and distribute the RTCs and driver code examples embedded in its User Software, in object code form only, provided that:

(a) Licensee distributes the RTCs and driver code examples only in conjunction with and as a part of its User Software;

(b) Licensee will be solely responsible to anyone receiving its User Software for any updates, technical and other support obligations, and any other liability which may arise from the distribution of its User Software;

(c) Licensee does not use Compuware's or its licensors' names, logos, or trademarks to market or distribute its User Software;

(d) Licensee includes Compuware's and its licensors' copyright and/or proprietary notices and legends within the executable images of its User Software and on Licensee's software media and documentation; and

(e) Licensee agrees to indemnify, hold harmless and defend Compuware and its licensors from and against any claims or lawsuits, including attorney's fees, that arise or result from the use or distribution of its User Software.

Government Users: With respect to any acquisition of the Software by or for any unit or agency of the United States Government, the Software shall be classified as "commercial computer software," as that term is defined in the applicable provisions of the Federal Acquisition Regulation (the "FAR") and supplements thereto, including the Department of Defense (DoD) FAR Supplement (the "DFARS"). If the Software is supplied for use by DoD, the Software is delivered subject to the terms of this Agreement and either (i) in accordance with DFARS 227.7202-1(a) and 227.7202-3(a), or (ii) with restricted rights in accordance with DFARS 252.227-7013(c)(1)(ii) (OCT 1988), as applicable. If the Software is supplied for use by a Federal agency other than DoD, the Software is restricted computer software delivered subject to the terms of this Agreement and (i) FAR 12.212(a); (ii) FAR 52.227-19; or (iii) FAR 52.227-14(ALT III), as applicable. Licensor: Compuware Corporation, 31440 Northwestern Highway, Farmington Hills, Michigan 48334.

Limited Warranty and Remedy: Compuware warrants the Software media to be free of defects in workmanship for a period of ninety (90) days from purchase. During this period, Compuware will replace at no cost any such media returned to Compuware, postage prepaid. This service is Compuware's sole liability under this warranty. COMPUWARE DISCLAIMS ALL EXPRESS AND IMPLIED WARRANTIES, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. SOME STATES DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO THE ABOVE EXCLUSION MAY NOT APPLY TO LICENSEE. IN THAT EVENT, ANY IMPLIED WARRANTIES ARE LIMITED IN DURATION TO THIRTY (30) DAYS FROM THE DELIVERY OF THE SOFTWARE. LICENSEE MAY HAVE OTHER RIGHTS, WHICH VARY FROM STATE TO STATE.

Infringement of Intellectual Property Rights: In the event of an intellectual property right claim, Compuware agrees to indemnify and hold Licensee harmless, provided Licensee gives Compuware prompt written notice of such claim, permits Compuware to defend or settle the claim, and provides all reasonable assistance to Compuware in defending or settling the claim. In the defense or settlement of such claim, Compuware may obtain for Licensee the right to continue using the Software or replace or modify the Software so that it avoids such claim, or if such remedies are not reasonably available, accept the return of the infringing Software and provide Licensee with a pro-rata refund of the license fees paid for such Software based on a three (3) year use period.

Limitation of Liability: LICENSEE ASSUMES THE ENTIRE RISK AS TO THE RESULTS AND PERFORMANCE OF THE SOFTWARE. IN NO EVENT WILL COMPUWARE BE LIABLE TO LICENSEE OR TO ANY THIRD PARTY FOR ANY SPECIAL, INDIRECT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING BUT NOT LIMITED TO, LOSS OF USE, DATA, REVENUES OR PROFITS, ARISING OUT OF OR IN CONNECTION WITH THIS AGREEMENT OR THE USE, OPERATION OR PERFORMANCE OF THE SOFTWARE, WHETHER SUCH LIABILITY ARISES FROM ANY CLAIM BASED UPON CONTRACT, WARRANTY, TORT (INCLUDING NEGLIGENCE), PRODUCT LIABILITY OR OTHERWISE, AND WHETHER OR NOT COMPUWARE OR ITS LICENSORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE. SOME STATES DO NOT ALLOW THE LIMITATION OR EXCLUSION OF

LIABILITY FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE LIMITATION OR EXCLUSION MAY NOT APPLY TO LICENSEE. IN NO EVENT SHALL COMPUWARE BE LIABLE TO LICENSEE FOR AMOUNTS IN EXCESS OF PURCHASE PRICE PAID FOR THE SOFTWARE.

Term and Termination: This License Agreement shall be effective upon Licensee's acceptance of this Agreement and shall continue until terminated by mutual consent, or by election of either Licensee or Compuware in case of the other's unremediated material breach. In case of any termination of this Agreement, Licensee will immediately return to Compuware the Software that Licensee has obtained under this Agreement and will certify in writing that all copies of the Software have been returned or erased from the memory of its computer or made non-readable.

General: This Agreement, the Compuware invoice and Licensee purchase order, are the complete and exclusive statement of the parties' agreement and supersede any prior agreement or understanding whether oral or written, relating to the subject of this Agreement. The preprinted terms of any purchase order accepted by Compuware are expressly superseded by this Agreement. Should any provision of this Agreement be held to be invalid by any court of competent jurisdiction, that provision will be enforced to the maximum extent permissible and the remainder of the Agreement shall nonetheless remain in full force and effect. This Agreement shall be governed by the laws of the State of Michigan and the United States of America.

YOU ACKNOWLEDGE THAT YOU HAVE READ THIS AGREEMENT, UNDERSTAND IT AND AGREE TO BE BOUND BY ITS TERMS AND CONDITIONS.

Table of Contents

5 Installing SoftICE

6 Host and Target Configurations

Chapter 1
Quick Install

This chapter provides a brief description of a basic DriverStudio installation, as well as helpful installation tips. More detailed instructions for specific DriverStudio/SoftICE Driver Suite components are provided in later chapters in this manual.

Quick Installation Instructions

The following instructions guide you through the process of installing the DriverStudio/SoftICE Driver Suite tools.

1 Insert the product CD into your CD-ROM drive.

2 If you have the autorun feature enabled, Setup runs automatically. If not, open the **Add/Remove Programs** control panel and click **Install**.

3 If you choose not to register electronically during installation, fill out your registration and Compuware Upgrade Subscription program cards and send them to Compuware.

Tips for Installing Your Compuware Products

Entering Your Serial Number

Find the product serial number on your registration card or CD envelope. It is a twelve-digit number that includes hyphens. Enter the serial number exactly as shown.

Serial Number Not Working?

Check that you are installing the correct product and product version. Serial numbers are specific to a particular release of the product.

Problems During Installation?

A common cause of installation problems is the presence of “leftover” files from previous installations in your system **temp** directory. If you encounter errors during the installation process, try deleting the contents of your system **temp** directory, then re-install the product.

Choosing a Display Adapter for use with SoftICE

During installation, if your display adapter is not supported, try selecting the Universal Video Driver (Display Adapter Selection in the SoftICE Setup Wizard).

If your adapter fails the display adapter test you should try downloading the latest display drivers from your display adapter’s manufacturer. (See Chapter 5, *Installing SoftICE,* on Page 27.)

Chapter 2
Installing DriverWorks

Building the Libraries

In order to reduce the distribution size (and ensure that your libraries are built with a DDK and compiler consistent with your development environment) **DriverWorks** does not ship with binary versions of the libraries. You must build the libraries prior to building any drivers. However, before building the libraries, you must have the appropriate DDK installed. Refer to the following table for information on which DDK to use.

Table 2-1 DriverWorks Library Configurations

Configuration	DDK to Use	Library File	Target Drivers
VdwLibs – Win32 WDM Free or Checked	Windows 2000 ***or*** Windows Me ***or*** Windows XP	vdw_wdm.lib	WDM *style* drivers targeting Win2K, WinXP, Win98, WinMe
VdwLibs – Win32 NT4 Free or Checked	NT4, Windows 2000, or Windows XP	vdw.lib	NT4 *style* drivers targeting NT4, Win2K, WinXP, Win98, WinMe
VdwLibs – IA64 WDM Free or Checked	Windows XP	vdw_wdm.lib	WDM *style* drivers targeting WinXP for IA64 platform (64 bit only)
NdisWdm – Win32 NDIS WDM Free or Checked	Windows 2000 ***or*** Windows XP	kndiswdm.lib	NDIS WDM drivers targeting Win2K, WinXP, Win98, WinMe. (See DriverNetworks Help.)

Table 2-1 DriverWorks Library Configurations

Configuration	DDK to Use	Library File	Target Drivers
NdisWdm – Win32 IA64Free or Checked	Windows XP	kndiswdm.lib	NDIS WDM drivers targeting WinXP for IA64 platform (64-bit only.) (See DriverNetworks Help.)

Building from within Visual Studio

Use the following procedure to build the libraries from within Microsoft Visual Studio.

1 **Set up the correct environment for building device drivers.**

There are two ways to set up the correct build environment. The first way is to start Visual C++ using SetDDKGo. SetDDKGo is designed to make it easy to select a DDK to use, and to start Visual Studio with the environment set up correctly for that DDK. To launch Visual Studio with SetDDKGo, select the shortcut located in the start menu under **Compuware DriverStudio | Tools | Setup DDK and Start MSVC.**

It is also possible to set up the driver environment from within Visual Studio using the **DDK Settings** option located in the **DriverStudio** menu. This allows the environment to be changed without exiting Visual Studio.

2 **Load the workspace VdwLibs in Visual Studio.**

Select **File | Open Workspace** and browse to **DriverStudio\DriverWorks\source\vdwlibs.dsw.**

3 **Select to build or batch build the configurations you desire.**

When building with the Windows XP DDK, the DriverStudio add-in for Visual C++ will automatically switch to the correct XP DDK compiler to build the library.

Note DriverStudio provides an option to build device drivers using the DDK BUILD utility from within the Visual Studio IDE. Select **Build using BUILD.EXE** from the DriverStudio menu or toolbar. (See "How to Run a DDK Build from within the Visual Studio IDE" in the DriverWorks online Help for more details.)

Building from the Command Line

Use the following procedure to build libraries from the command line.

1 **Install the appropriate DDK for your platform.**

2 **Verify that the BASEDIR environment variable specifies the correct path to the DDK.**

Note The following step (Step #3) is not required when building with the **Windows XP DDK** because it provides a complete build environment.

---

3 **Ensure the appropriate version of the Microsoft compiler is on the path and starts correctly from the command line.**

You might need to run the VCVARS32.BAT batch file to define the path and environment variables correctly for command line operation. This .BAT file is in the MICROSOFT_VISUAL_STUDIO\VC98\BIN directory for MSVC++ 6.0. Type "CL" from the command line to ensure the compiler will run.

4 **Change to the root of the DriverWorks installation.**

Example:

```
C:> cd\Program Files\Compuware\DriverStudio\DriverWorks
```

5 **Run BLDLIB.BAT to build the libraries.**

This batch file takes three parameters, with the first two required. The allowed parameter values are:

BLDLIB [checked or free] [target] [64]

[checked or free] -[required] select a checked or free build.

[target] - [required] Possible values are:

NT	build NT 4.x libs
WDM	builds the WDM lib (vdw_wdm .lib)
NDIS	builds the NDIS WDM lib for use with DriverWorks (kndiswdm.lib)

[64] - (optional) If 64 is provided, a 64-bit version of the library will be built. The Windows XP DDK is required.

Note 64-bit configurations are not available in the NT 4 library.

Examples:

BLDLIB checked WDM (Builds the checked version of the WDM library)

BLDLIB free WDM 64 (Builds the free 64-bit version of the WDM library)

BLDLIB checked NDIS (Builds the checked 32-bit NDIS WDM library)

BLDLIB free NT (Builds the free version of the NT 4 library)

Testing Your Installation

Preparing to Build From a Command Prompt

If you want to build a sample driver from the command prompt (as opposed to from Microsoft Visual Studio), then you must correctly set up the build environment. The environment may be set up by running **setenv** for the DDK, or by selecting one of the build environment shortcuts in the Start menu under:

```
Start | Programs | Development Kits | DDK.
```

Running SETENV.BAT

In the BIN subdirectory of the DDK, there is a file named SETENV.BAT that sets up additional environment variables required by the BUILD utility. Run this file from the DOS prompt. The batch file requires the following two parameters:

- Enter the **root directory** for the DDK
- Enter **free** or **checked**, to select a free or checked (debug) build

Example: `C:TEST> setenv c:\ddk checked`

Running VCVARS32.BAT

A second .BAT file that sets up the paths for the compiler is named VCVARS32.BAT. (This file is in the `\MICROSOFT_VISUAL_STUDIO \VC98\BIN` directory for MSVC++ 6.0). You must run this .BAT file as well.

Note This step is not required when building with the Windows XP DDK because it provides a complete build environment.

You must run SETENV.BAT and VCVARS32.BAT each time you set up the environment for building drivers. When building from inside Developer Studio, this is done automatically.

Checking the Environment

After installing DriverWorks, it is recommended you reboot your system.

If you plan to build from the command line, confirm that the DDK and DRIVERWORKS environment variables are correct. Start a command

shell and type “set”. You should have the following symbols defined in your environment after running setenv.bat.

- **DRIVERWORKS** = Root directory of your DriverWorks installation.
  Example: `c:\DriverStudio\DriverWorks`
- **BASEDIR** = Root directory of your DDK installation.
  Example: `c:\ddk`
- **CPU** = i386 for 32-bit drivers or IA64 for 64-bit drivers.

Your PATH should also include the `DDK\BIN` directory and the MSVC paths. If the SDK binaries are on your path, they must not precede the DDK or MSVC paths.

Testing Samples

For Windows NT 4.0 drivers, the simplest example is in the HELLO folder under `DriverStudio\DriverWorks\Examples\NT`. You can build this sample and load it with the DriverMonitor utility.

For WDM drivers (Windows 98, Windows Me, Windows 2000, and Windows XP), the simplest sample is in the HELLOWDM folder under `DriverStudio\DriverWorks\Examples\WDM`. After building this sample, you can install the driver with the **Add New Hardware** wizard on the system Control Panel.

Building the Test Sample

Confirm that you can build one or more DriverWorks examples. If you have Microsoft Visual C++ 6.0 or later, you can open the workspace file `%DRIVERWORKS%\examples\nt\hello\hello.dsw` and build the driver in the Developer Studio environment. For WDM, choose the HELLOWDM sample.

Alternatively, if you want to build from the command prompt, connect to one of the example directories and run the BUILD program:

```
cd \Program Files\Compuware\DriverStudio\DriverWorks\
   examples\nt\hello
build
```

The example should build without errors. Review the output file “build.log” (or “buildchk.log” or “buildfre.log”) to confirm there are no errors.

Confirming that the Driver Loads and Runs

For non-WDM drivers on NT systems, run DriverMonitor to confirm that the new HELLO driver works correctly. Please note that the driver

`HELLO.SYS` will be created in the `objchk\i386` or `objfre\i386` directory depending on your build environment.

When you load HELLO.SYS from DriverMonitor, you should see a load confirmation on the Monitor screen, followed by a HELLO message generated by the driver.

In order to load device drivers under Windows NT, you need to be logged into an account that has Administrator privileges on your system. You will receive error messages from DriverMonitor if you do not have sufficient privileges to install kernel-mode device drivers.

Unload the driver before exiting from DriverMonitor.

To load HELLOWDM on Windows 98, first copy file HELLOWDM.INF to the directory where the .SYS file resides. Then open the Control Panel and run the **Add New Hardware** wizard. When prompted, tell the wizard that you want to select hardware from a list. HELLOWDM is in class "Other Devices." When prompted for the Manufacturer, select **Have Disk**, and browse to the directory where the INF and SYS file reside. If you have DriverMonitor running during the installation, you will see messages emitted by the driver. After installation, open the System icon in the Control Panel. The device should appear on the Device Manager tab.

Testing the DriverWizard Installation

Use this procedure to confirm that DriverWizard is available from Microsoft Visual C++.

1 Start Developer Studio.

2 Select **DriverStudio | DriverWizard** from the menu, or start DriverWizard from the toolbar.

3 Enter a project name and location, and click **Next** to begin to build your device driver.

4 Choose driver parameters, then select **Finish**.

5 Ensure that your driver builds properly by selecting **Build | Rebuild All**.

Chapter 3
Installing DriverNetworks

Building Libraries and Drivers

Since the introduction of the Windows XP DDK, network driver developers have had to contend with the possibility of multiple DDK environments and two or more target platforms on the same development machine: NT 4, Win2K, and/or WinXP.

Choosing the Correct Build Environment

It is critical for DriverNetworks driver project developers to choose the correct build environment. The recommended DDK to use for a given driver is one of the following.

- Windows NT 4 DDK for NDIS 4 miniport drivers (target: NT 4, Win95)
- Windows XP or Windows 2000 DDK for NDIS 5 miniport drivers (target Win98, WinMe, Win2K, WinXP)
- Windows XP or Windows 2000 DDK for Intermediate drivers and TDI Clients (target: Win98, WinMe, NT 4, Win2K, WinXP)

The DriverNetworks Library Configurations table on Page 10 lists the project configurations to be used in the various DDK environments.

Table 3-1 DriverNetworks Library Configurations

Configuration	DDK to Use	Library File	Target Drivers
KNdisLib - Win32 NDIS 4 Miniport Free or Checked	Windows NT 4	kndis4mp.lib	NDIS 4 Miniports
KNdisLib - Win32 NDIS 5 Miniport Free or Checked	Windows 2000 or Windows XP	kndis5mp.lib	NDIS 5 Miniports and IM Drivers
KNdisLib - Win32 NDIS 5 Protocol Free or Checked	Windows 2000	kndis5pt.lib	NT 4, Win2K TDI Clients; Protocol Drivers
KNdis64 – Win32 NDIS 5 Miniport Free or Checked	Windows XP	kndis5mp.lib	NDIS 5 Miniports & IM Drivers for XP IA64
KNdis64 – Win32 NDIS 5 Protocol Free or Checked	Windows XP	kndis5pt.lib	TDI Clients; Protocols for XP IA64
Tdiclient - Win32 NT 5 Debug or Release	Windows 2000 or Windows XP	tdint5.lib	NT 4, Win2K, WinXP TDI Clients
Tdi64 - Win32 NT 5 Debug or Release	Windows XP	tdint5.lib	WinXP IA64 TDI Clients
Tdiclient - Win32 W9x SYS Debug or Release	Windows 2000 VTOOLSD	tdiw9sys.lib	Win9x TDI Clients NDIS Miniports
Tdiclient - Win32 W9x VxD Debug or Release	Windows 2000 VTOOLSD	tdi9vxd.lib	Win9x TDI Clients VxDs (VTOOLSD)

Notes

The respective output .LIB files reside in \lib\$(CPU)\checked\ or \lib\$(CPU)\free\ subdirectories, which are branched from the DriverNetworks root directory. The $CPU value is i386 for 32-bit targets and IA64 for 64-bit ones. The DriverNetworks root directory is specified by the environment variable $(DRIVERNETWORKS) created by the DriverStudio install.

As the table indicates, both Windows 2000 DDK and VTOOLSD must be installed in order to build the TDI Client libraries for Win9x drivers.

Building Libraries from within Visual Studio

Complete the following steps to build the libraries from within Visual Studio.

1 **Set up the correct environment for building device drivers.**

There are two ways to set up the correct build environment. The first way is to start Visual C++ using SetDDKGo. SetDDKGo is designed to make it easy to select a DDK to use, and to start Visual Studio with the environment set up correctly for that DDK. To launch Visual Studio with SetDDKGo, select the shortcut located in the start menu under **Compuware DriverStudio | Tools | DDK Build Settings (SetDDKGo).**

It is also possible to set up the driver environment from within Visual Studio using the **Driver Build Settings** option located in the **DriverStudio** menu. This allows the environment to be changed without exiting Visual Studio.

2 **Load the workspace DNW in Visual Studio.**

Select **File | Open Workspace** and browse to **DriverStudio\DriverNetworks\source\dnw.dsw**.

3 **Select to build or batch build the configurations you desire.**

When building with the Windows XP DDK, the DriverStudio add-in for Visual C++ will automatically switch to the correct XP DDK compiler to build the library.

Note DriverStudio provides an option to build device drivers using the DDK BUILD utility from within the Visual Studio IDE. Select 'Build using BUILD.EXE' from the DriverStudio menu or toolbar. (See "How to Run a DDK Build from within the Visual Studio IDE" in the online DriverNetworks Help for more details.)

Building Libraries from the Command Line

Complete the following steps to build libraries from the command line.

Note It is not necessary to launch one of the DDK build environments from the start menu.

---

1 **Install the appropriate DDK.**

2 **Verify that the BASEDIR environment variable specifies the correct path to the DDK.**

3 **Run BuildNdis.bat or BuildTdi.bat to build the desired configuration.**

The command line for **BuildNdis** is as follows:

BuildNdis [checked | free] [miniport4 | miniport5 | protocol5] [IA64]

Checked | free

Selects rather to build a checked or free configuration

miniport4 | miniport5 | protocol5

miniport4 – Build the NDIS 4 miniport library

miniport5 – Build the NDIS 5 miniport library

protocol5 – Build the NDIS 5 protocol library

IA64

Specify IA64 to build a 64-bit library.

Note The IA64 option is only available on NDIS 5 libraries.

The command line for **BuildTDI** is as follows:

BuildTDI [checked | free] [NT | 9xVxD | 9xSYS] [IA64]

Checked | free

Selects rather to build a checked or free configuration

NT | 9xVxD | 9xSYS

NT – Build the Windows NT TDI client library

9xVxD – Build the VxD TDI client library

9xSYS – Build the TDI client library for WDM drivers running on Windows 9x.

IA64

Specify IA64 to build a 64-bit library.

Note The IA64 option is only available on the NT library.

Chapter 4
Installing VTOOLSD

The installation procedure performs two basic functions:

- It copies the files that comprise VTOOLSD to a directory tree of your choice.
- It creates a configuration file, USER.MAK, that defines symbols used by the VTOOLSD make system.

You must install VTOOLSD onto your system (or to a network directory accessible from your system.) You cannot use VTOOLSD directly from the distribution CD.

Creating USER.MAK

USER.MAK is a configuration file that the VTOOLSD make system uses to locate tools required to build VxDs. It consists of a list of symbol definitions, each of which informs NMAKE where a particular program resides on your system. SETUP prompts you to enter the paths for various required tools.

A correct USER.MAK is the key to a successful installation. Double check this file after installation to verify its concerns.

Some of the symbols defined in USER.MAK are files, and some are tokens used to control the make process. The following table explains the symbols that USER.MAK defines.

Table 4-1 Symbols in USER.MAK

Symbol	Meaning
COMPILER	Symbol to select compiler: MS5 or BCB3
DEBUGGER	Symbol to select debugger: SOFTICE3, WDEB386, or NONE
C32	Full pathname of 32-bit command line compiler

Table 4-1 Symbols in USER.MAK

Symbol	Meaning
LINKMSVC2	Full pathname of MS linker executable
TLINK32	Full pathname of Borland linker executable
ASM6	Full pathname of Microsoft MASM 6 executable
TASM32	Full pathname of the Borland assembler
NMSYM	Full pathname of the symbol file generator
EDITBIN	Full pathname of Microsoft EDITBIN utility

Each symbol definition line is of the form *SYMBOL* = <string>.

Example:

```
C32 = C:\BCC32\BIN\BCC32.EXE
```

The VTOOLSD make system uses additional symbols that are described in Chapter 5 of the *Using VTOOLSD* document.

VTOOLSD Environment Variable

The VTOOLSD make system requires you to define the environment variable **VTOOLSD**. The value of the variable must be the root directory of the VTOOLSD installation.

Example:

```
set VTOOLSD=C:\VTOOLSD
```

SETUP offers to insert this line into your AUTOEXEC.BAT file. If you opt not to have SETUP do it, you should add it yourself or set the variable each time you build a VxD with the VTOOLSD make system.

Chapter 5
Installing SoftICE

This chapter explains how to install and configure SoftICE on Windows 95, Windows 98, Windows ME, Windows NT, Windows 2000, and Windows XP. Both of these tasks are accomplished using the DriverStudio installation program. After installation, the SoftICE configuration pages can be accessed through **Start | Compuware DriverStudio | Settings**.

SoftICE Display Options

Your display configuration for SoftICE depends on your requirements. SoftICE supports the following video configurations:

- One display adapter and monitor using the SoftICE Universal Video Driver.
- One display adapter and monitor using the SoftICE adapter-specific drivers.
- Secondary monochrome card and monitor.
- Secondary VGA card and monitor.
- Second computer connected through a serial port or network.

Universal Video Driver

SoftICE supports virtually all modern video adapters using a "universal" video driver. This driver allows SoftICE to display in a movable window directly on top of the current display.

If the video adapter is placed in text mode (e.g., by a full-screen DOS box), SoftICE will display in full-screen text mode as well. Although this involves device-dependent register access, in most cases the Standard VGA driver can be used successfully, since the display is in a standard VGA mode.

Unless you experience difficulties with the Universal Video Driver, this is the method of choice for single-monitor debugging.

Adapter-Specific Drivers

SoftICE includes a number of adapter-specific drivers. These drivers are for the most part intended to support older video adapters, and should only be used if your video hardware does not function properly with the Universal Video Driver. Adapter-specific drivers are selected from a drop-down list on SoftICE's video configuration page.

Secondary Monochrome Card and Monitor

Use a secondary MDA (Monochrome Display Adapter) or Hercules-compatible display adapter coupled to a monochrome monitor to display both your application and the SoftICE debugging session simultaneously. The primary monitor displays your application while the secondary monochrome monitor displays your SoftICE debugging session. In this configuration, the SoftICE screen is limited to 43 lines.

Note Most display adapters can coexist with mono cards; however, there are exceptions. Review your display adapter's documentation to verify that your adapter can coexist with a monochrome card.

Using a secondary monochrome card and monitor is particularly helpful for debugging under the following circumstances.

- Debugging with an unsupported display adapter.

  If the SoftICE Universal Video Driver does not work properly with your video hardware, and there is no adapter-specific driver available, you can use a secondary monochrome card and monitor as a display alternative.
- Debugging video drivers.

  When SoftICE assumes control of the display adapter, it may change one or more states and affect debugging. Using a secondary adapter lets you avoid this situation.

Secondary VGA Monitor Debugging

On Win9x, Win2K, and WinXP systems, it is possible to install two VGA video cards and have SoftICE display on the secondary video card. This offers a replacement for the monochrome monitor configuration, as monochrome adapters and monitors are become increasingly rare. Using a secondary VGA adapter, Windows will display the desktop on

the primary, AGP video board, and SoftICE will display on the PCI-based board, in text mode.

1. To configure your system for Secondary VGA Monitor debugging requires that you first add a second, PCI-based video board to the system. Then follow the instructions below. Note that this procedure will not work for all combinations of system BIOSes and video boards; you may need to try more than one combination to find one that does work.
   a. If after installing the secondary VGA board you receive a BIOS error beep code (typically one long beep followed by two short beeps), you will need to power down, remove one video card, and then reboot.
   b. Upon reboot, enter your BIOS setup. From here you will need to find your video configuration settings. It is typically on the "Advanced" page.
   c. In the Video Configuration settings will typically be an entry entitled "Primary" or "Primary Video". Choose **PCI**.
   d. Save your configuration and reboot.
   e. Now, power down your machine and reinstall your PCI video card.
2. Boot into Windows.
   a. At this point in time, you should see the VGA screen which has the OS loader message (i.e., the output from /sos) and the Windows Desktop displayed on your AGP video card. (If you see a DOS text screen on one monitor and the Windows desktop on the other monitor, proceed to Step 3.)
   b. If you see the Windows desktop on both machines, you will need to make an additional change to Windows. Right-click on the desktop, choose properties; then, choose Settings.
   c. On the Settings page, choose the PCI monitor by clicking on the proper monitor.
   d. Uncheck "Extend my Windows Desktop onto this monitor."
3. Go to the SoftICE configuration page in the Configuration dialog and select Video Settings.
   a. Deselect Universal video driver and choose Full Screen VGA.
   b. For Video Driver, be sure to select "Standard VGA."
4. Go to the General SoftICE settings page. On the init line, add "altscr vga" at the start of the line.
5. Reboot your machine.

Now when you start SoftICE, it should always be visible just as if you were using a mono monitor.

Connecting a Second Computer

Connect a second computer through a serial port or a network to display both your application and the SoftICE debugging session simultaneously. The local primary computer displays your application while the secondary computer displays your SoftICE debugging session.

In essence, the remote computer is a dumb terminal that serves to display output and accept keyboard and mouse input.

Using a second computer connected through a serial port or network is particularly helpful for debugging under the following circumstances:

- **Debugging with an unsupported display adapter**

  If SoftICE does not support your display adapter or you are developing a new display adapter, you can use a second computer as a display alternative. This option is particularly useful for laptop computers.
- **Debugging display adapters**

  When SoftICE assumes control of the display adapter, it may change one or more states. Using a second computer lets you avoid this situation.
- **Debugging keyboard drivers**

  SoftICE uses the same keyboard controller as the driver, so the driver can result in an unanticipated state. Using a second computer lets you avoid this situation.
- **Debugging when the system is at a reduced power state**

Using a Secondary VGA Display Adapter and Monitor

Use a secondary VGA display adapter and monitor to obtain the greatest flexibility for displaying and debugging an application simultaneously. The primary monitor displays your application while the secondary monitor displays your SoftICE debugging session.

Note Only certain display adapters are designed to support this multiple display adapter option.

Refer to your display adapter documentation for an explanation of how to disable the VGA aspects of your video card.

SoftICE Pre-Installation Checklist

Before you install SoftICE, make sure you do the following items.

1 If you are installing SoftICE on Windows NT-based Operating Systems, verify that you have an account with administrator privileges.

2 If you are installing SoftICE on Windows Me, you must download and install the "Microsoft Debugging Tools for Windows Me" **before** you install SoftICE. This debugging toolkit replaces the retail version of io.sys with the checked version (which allows a debugger to be loaded). Follow these steps.

   a Download the "Micrsosoft Debugging Tools for Windows Me" kit from **http://www.microsoft.com/ddk/**.

   b Install the "Microsoft Debugging Tools for Windows Me" kit on the Millennium system on which you wish to install SoftICE. (Do **NOT** reboot after the install.)

   c Install SoftICE (as described below).

   d Reboot.

3 Below are listed requirements for each display option. Once you have determined which option best suits your needs, gather the information listed.

   ◇ **One display adapter and monitor using the SoftICE Universal Video Driver.** No additional information is needed. This is the preferred way to debug with one display adapter.

   ◇ **One display adapter and monitor using the SoftICE adapter-specific drivers.** Determine the display adapter's manufacturer and model number.

Note Use the Control Panel Display properties to determine your manufacturer and model number.

   ◇ **Secondary VGA display adapter**

   Verify that the two display adapters can coexist with one another.

   ◇ **Second computer**

   Determine the serial port characteristics or (if you will be using a network) the network characteristics.

4 Determine the type of mouse you are using, Serial, PS/2, or USB. If you are using a serial mouse, determine whether it is connected to COM1 or COM2.

5 Exit all Windows programs.

Installing SoftICE

Use the following procedure to install SoftICE.

1 In the Select Install Directory window, select the directory where you want to install SoftICE..

The default directory is:

```
C:\PROGRAM FILES\COMPUWARE\DRIVERSTUDIO\SOFTICE
```

If the directory you choose does not already exist, the wizard creates the directory for you.

2 The **Settings** dialog (**Programs | Compuware DriverStudio | Settings**) displays the settings pages for all DriverStudio components. Installation settings for other components are discussed in the section on Installing DriverStudio. This section discusses the options for the SoftICE startup mode, and video and mouse setup. These settings are located in the Control Panel and SoftICE program groups on the left side of the window.

3 If you are installing SoftICE on Windows NT-based Operating Systems, select one of the options in the Startup Mode Selection window to determine when SoftICE loads. (Table 5-1 describes the Startup Mode selections for the NT4, Win2K, and WinXP platforms.)

Table 5-1 Startup Mode Selections for Windows NT-family Platforms

Windows NT-based OS Startup Mode Options	Description
BOOT	Boot drivers such as disk controller drivers and certain file system drivers are critical to system booting. To ensure that it can load files such as initialization and symbol files, SoftICE always loads as the last boot driver.
SYSTEM	System drivers are loaded after boot drivers. If this option is selected, it is usually preferred to set SoftICE to start at Boot Mode.

Table 5-1 Startup Mode Selections for Windows NT-family Platforms

Windows NT-based OS Startup Mode Options	Description
AUTOMATIC	Automatic drivers are loaded by the Service Controller Manager during the last phase of system startup. The system is essentially done booting at this point. If you want to load SoftICE every time Windows starts, but you are not interested in debugging a core device driver, you could load it here. However, it is usually preferred to set SoftICE to start at Boot Mode.
MANUAL	SoftICE is not started automatically at boot time. This mode offers the greatest safety and flexibility, but it precludes debugging device drivers during the system boot. (See the *Using SoftICE* document.) The **net start ntice** command can be issued to start SoftICE in this mode.

4 In the Display Adapter Selection window, select the display option you wish to use and enter the appropriate information from Step 3 of the pre-installation:

- **To use the universal video driver**, select the "Universal Video Driver" checkbox.
- **To use SoftICE with one display adapter and monitor**, select the Manufacturer and Model of the display adapter. If your display adapter is not listed, select a display adapter with the same graphics chip as indicated in the COMPATIBILITY box. If you cannot find a display adapter with the same graphics chip, select STANDARD VGA (640x480 pixels), finish the installation, and refer to *Using SoftICE,* Appendix B, "Supported Display Adapters."
- **To use SoftICE on a secondary monochrome screen**, select DISPLAY SOFTICE ON ATTACHED MONOCHROME MONITOR, then select the Manufacturer and Model of the primary display adapter. If your display adapter is not listed, select a display adapter with the same graphics chip as indicated in the COMPATIBILITY box. If you cannot find a display adapter with the same graphics chip, select STANDARD VGA (640x480 pixels) and finish the installation.
- **To use SoftICE with a secondary VGA card**, select STANDARD VGA (640x480 pixels). SoftICE ignores the primary display adapter and uses the secondary VGA adapter and monitor.

- ◇ **To use SoftICE with a second computer**, select the Manufacturer and Model for the primary display adapter. If your display adapter is not listed, select a display adapter with the same graphics chip as indicated in the COMPATIBILITY box. If you cannot find a display adapter with the same graphics chip, select STANDARD VGA (640x480pixels) and finish the installation.

Note To select a different display adapter after installing SoftICE, start the DriverStudio settings program, and use the Video page in the SoftICE program group.

---

5 Once you have configured the video adapter settings properly, click TEST to test the display adapter settings.

6 In the Mouse Selection window, select one of the following options:

- ◇ Serial (connected to COM1)
- ◇ Serial (connected to COM2)
- ◇ PS/2 compatible or USB
- ◇ None
- ◇ IntelliMouse

Note If you are using a PS/2 or USB mouse, SoftICE will detect the presence of an Intellimouse automatically. To select a different mouse type after installation, use **SETTINGS** | **MOUSE** in the SoftICE program group.

---

7 If you are installing under Windows 95 or Windows 98, select one of the SoftICE System Configuration window options described in Table 5-2.

Table 5-2 Startup Mode Selections for Win95/Win98 Platforms

Windows 95 & Windows 98 Startup Mode Options	Description
Let Setup modify AUTOEXEC.BAT	This option appends the statement C:\DriverStudio\SoftIce\WINICE.EXE to the end of your AUTOEXEC.BAT and replaces C:\DriverStudio\SoftIce\WIN95 with the SoftICE installation directory.
Save the required changes to AUTOEXEC.BAT	This option gives you the opportunity to view the changes in a temporary file before implementing them. The wizard copies the AUTOEXEC.BAT to a file named AUTOEXEC.ICE and adds the WINICE.EXE statement to it. After you review the changes, rename your existing AUTOEXEC.BAT (to, for example. AUTOEXEC.OLD) and rename the AUTOEXEC.ICE file to AUTOEXEC.BAT.
Do not make any changes	This option does not modify your AUTOEXEC.BAT. If you select this option, you need to configure your system to load SoftICE before WIN.COM. If you are debugging a device driver, you can configure your system to decrease significantly the amount of time it takes you to switch from developing to debuggin. (Refer to *Configuring SoftICE Loading for Windows 95 and Windows 98* later in this document.)

8 Read the README file for last-minute product information.

Post-Installation Checklist

After you install SoftICE, perform the following steps as needed.

1 If you are running Windows 95 or Windows 98, and you selected LET SETUP MODIFY AUTOEXEC.BAT, Windows 95 and Windows 98 do not return control to SoftICE when it shuts down. Thus, SoftICE cannot save the breakpoint history file. To avoid this, set the BootGUI option in MSDOS.SYS to BootGUI=0. (Refer to the BH command in the *SoftICE Command Reference* for more information about the breakpoint history file and *Configuring SoftICE Loading*

from Windows 95 and Windows 98 (the next topic) for more information about the BootGUI statement.)

2 If you are running Windows 95 or Windows 98, and you selected DO NOT MAKE ANY CHANGES in the SoftICE System Configuration window, configure Windows 95 or Windows 98 to load SoftICE before WIN.COM. (Refer to the next topic, *Configuring SoftICE Loading for Windows 95 and Windows 98*.)

3 If you are using a second computer to display your SoftICE debugging session, configure SoftICE as described in *Connecting a Second Computer* (Page 30).

4 If you are running Windows 95, Windows 98, and the amount of physical memory installed on your PC exceeds 32 MB of RAM, modify the TOTAL RAM setting in your SoftICE Initialization settings to the correct value. (Refer to *Modifying General Settings* in the *Using SoftICE* document.)

Configuring SoftICE Loading for Windows 95 and Windows 98 Platforms

SoftICE is a kernel-mode debugger, so it must load before WIN.COM. By default, Windows 95 and Windows 98 boot directly into the new shell without giving you the opportunity to invoke WIN.COM explicitly. If you are debugging applications, run WINICE.EXE at the end of your AUTOEXEC.BAT. If you are developing static VxDs or other drivers, use the following method to optimize loading SoftICE before WIN.COM.

1 Do one of the following to prevent Windows 95 and Windows 98 from automatically loading and to force a DOS shell interpreter load.

- ◇ Press F8 while booting. When the Windows boot menu appears, select the option COMMAND PROMPT ONLY. Repeat this process each time you boot your PC.
- ◇ Append a PAUSE command at the end of your AUTOEXEC.BAT file and press Ctrl-C when it pauses to escape to DOS.
- ◇ Create a dummy batch file called WIN.BAT. When Windows starts, it executes WIN.BAT instead of WIN.COM and displays the DOS prompt.
- ◇ Modify the hidden file MSDOS.SYS (an INI text file) as follows:
  - Use the MS-DOS command ATTRIB to remove the hidden, read-only, and system attributes.
  - Edit the file to change the statement `BootGUI=1` to `BootGUI=0`.

2 Execute the command WINICE.EXE to load SoftICE, which loads Windows 95 or Windows 98.

3 Debug your code.

4 When you need to *restart* the computer, select either SHUT DOWN THE COMPUTER or RESTART THE COMPUTER. Any other shut down option destabilizes SoftICE.

When the final shutdown screen appears, SoftICE resets the display mode back to standard 80 x 25 text mode and displays the DOS prompt from Step 1. To continue debugging, repeat Step 2. When you are finished debugging, you can shut down Windows 95 and Windows 98 normally.

Connecting a Second Computer using a Null-Modem Cable

This section explains how to use the NET command and SIREMOTE.EXE to connect a second computer through a serial port using a null-modem cable. SIREMOTE.EXE is a client program that lets the second computer act like a dumb terminal, displaying output and accepting keyboard and mouse input.

To configure SoftICE to work with a second computer, do the following:

1 Use a null modem cable to connect the debugger machine to the client machine. (The figure on Page 38 shows “pin” diagrams for both a 9-pin and 25-pin null-modem cable.)

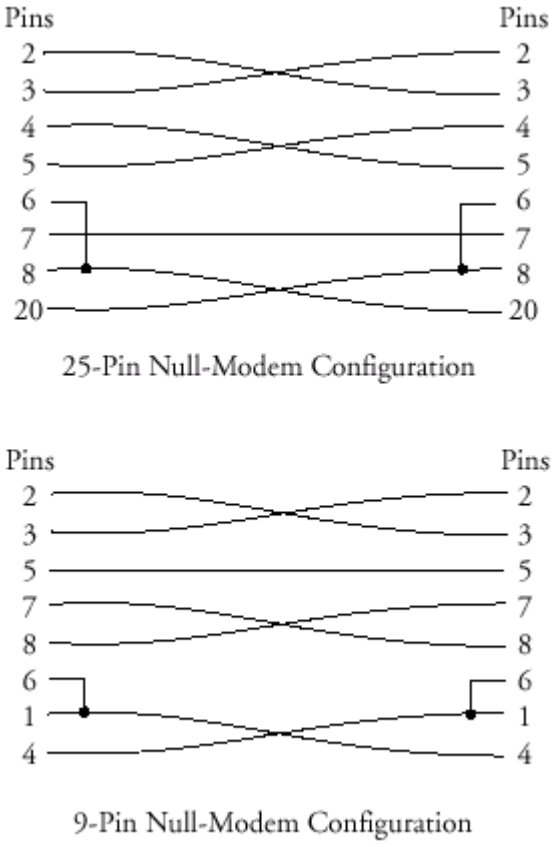

Figure 5-1 Null-Modem Configurations

2 Choose the **Edit | SoftICE Initialization Settings** menu item from the Symbol Loader (LOADER32.EXE).

3 Select the **Remote Debugging** tab from the **SoftICE Initialization Settings** dialog.

4 Select a COM port from the drop-down listbox. (This is the port that will be reserved for use by SoftICE.)

5 If you want SoftICE to automatically establish the connection upon loading, select the **Auto Connect** (via null modem) checkbox. Also, select a connection speed (the default is 57600).

6 Select **OK**.

7 To initiate the connection at SoftICE's first popup, remove the "X" from the INIT string in LOADER32's **Edit | SoftICE Initialization Settings** dialog on the debugger machine.

8 Reboot the debugger machine.

9 Once the debugger machine has rebooted, run SIREMOTE.EXE from the client machine with the following command line:

```
SIREMOTE.EXE COMx BAUD
```

10 SIREMOTE should initiate a connection and pop up SoftICE on both machines.

11 Using SoftICE from SIREMOTE is the same (as with a single-system execution of SoftICE) with the exception of the hot key. This is best explained by example. Suppose you are running SoftICE on two machines: machine A and machine B. At some point, you connect the two with a serial cable in order to debug A from B. You can run SIREMOTE on B; this allows you to bring up SoftICE on A by pressing CTRL-Z on B. If you want to start the *local* copy of SoftICE (on machine B), you can do that with the default hot key, CTRL-D.

12 If you do not want to configure SoftICE to automatically establish a connection upon loading, follow Steps 1 through 4 and reboot. After starting SoftICE, use the NET COMx BAUD command line from the SoftICE command line. The x denotes the COM port number that was reserved in Step 4 above.

Remote Debugging with SoftICE

For information about performing remote debugging over a network with SoftICE, refer to Chapter 9, "Remote Debugging with SoftICE," *in* the *Using SoftICE* document.

Troubleshooting Tips

Setting the Video Memory Size

When using the universal video driver, SoftICE must save the existing contents of the frame buffer so it can be restored later. The amount of memory required depends on the video mode, the number of lines used by SoftICE, and (for DirectDraw applications) how many flipping surfaces are in use. In any case, the amount of memory required can not exceed the amount of memory on your video card. By default, SoftICE reserves 2MB, but you can modify this using the Symbol Loader (on the **Edit** menu, select **SoftICE Initialization Settings** and change the "Video memory size" setting).

Solving Display Adapter Problems

SoftICE supports three different video modes: Universal Video Driver (UVD), Legacy, and Miscellaneous. Video functions reside in the

SIWVID driver. Proper loading of the SoftICE video driver is key to correctly hooking the video functions.

Video problems rarely occur on the Win9x platform because SoftICE is loaded prior to the Windows OS. Consequently, the following information applies primarily to the NT kernel; those cases that are applicable to the Win9x platforms are noted.

Universal Video Driver (UVD)

The preferred method of running SoftICE is to use the UVD. This method waits for an interrupt 2Dh from the loading of the video driver and hooks into the direct draw functions of the video card. This method should work on any card that supports DirectDraw.

Note Both the video card and the video card driver must support DirectDraw for the UVD to work.

If you're having problems using the UVD mode, try the following.

- Run the SoftICE video setup. If your card does not appear in the list, select VGA in the two list boxes (otherwise, choose your card in the two list boxes). Check the UVD selection box. When you click the TEST button, your video selection will be recorded. This will usually fix several problems; in particular, it will usually fix problems that occur when you have installed a new video driver (or updated an existing one).
- To give SoftICE a better chance of hooking the video, it is important to get the SIWVID driver to load as early as possible. Doing this requires checking the registry. Run REGEDIT and open the key `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Siwvid`. The value for "start" should be 0.
- Download and install the most current version of your video card driver provided by the manufacturer. Restart your system and try SoftICE again.
- Check the registry to be sure that SoftICE is using DirectDraw and is attempting to get notification from interrupt 2Dh.
- Using REGEDIT, open the key `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NTice`. Examine these values:
  - **Ddraw** - This is a DWORD value, and should be set to 1.
  - **DoInt2DPatch** - This is also a DWORD value, and should also be set to 1. This forces SoftICE to hook DirectDraw functions at the time the video driver loads.

- If SoftICE is running but not displaying, run the symbol loader from the Start menu and save the SoftICE history to a file. Review the file for errors and perform the corrective action indicated.
- A common error occurs when SoftICE requests NTSYMBOLS. This can be corrected by following the steps below.
  - Manually edit the WINICE.DAT file or use **SETTINGS | ADVANCED** and add the statement `NTSYMBOLS=ON`.
  - Find the symbols for the NT version you have. See the Microsoft website at http://www.microsoft.com/DDK/debugging/symbols.asp#Windows%202000%20Symbols.
  - Run the SoftICE symbol loader and from the **File** menu open the NTOSKRNL.DBG file (if using NT4 or Win2K) or load up NTOSKRNL.EXE (if using WinXP). From the **Module** menu choose **Translate**. This will create an NMS file. From the **Edit | SoftICE Initialization Settings** menu, select the **Symbols** tab and add NTOSKRNL.NMS.
  - Set up SoftICE to load at boot time. The SoftICE start up mode application is located in the Windows Start menu.

The above steps should solve most video card problems involving DirectDraw. Recently, machines have appeared with multiple video cards that support multiple monitors. If SoftICE has problems on such a system, the first thing to try is to simply re-run the SoftICE video setup from the Start menu. If that doesn't solve the problem, you must check that the video setup has correctly identified the video card driver. To do this, follow these steps.

1. Using REGEDIT, open the key `HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\VIDEO`. You will see multiple keys with the names `Device\VideoX`. These keys contain the video driver registry key names. They have the format: `\REGISTRY\Machine\System\ControlSet001\Services\MyVideo\Device0`.
2. Follow the video driver registry key names in the registry to locate the video driver's executable. (For example, if the video driver registry key name is: `\REGISTRY\Machine\System\ControlSet001\Services\myvideo\Device0`, then open the registry key `HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\myvideo`. You should see an item named "ImagePath" that will be the path to the video driver.) Write down all the video driver names that you find.
3. Tell SoftICE what video drivers to hook and what video drivers not to hook. The driver(s) that SoftICE will hook should be added

without the extension (i.e., without *.sys*) to the key `HKEY_LOCAL_MACHINE\SYSTEM\ControlSet\Services\NTice\InstalledDisplayDrivers`. Add all other drivers to the key `HKEY_LOCAL_MACHINE\SYSTEM\ControlSet\NTice\ExcludedDisplayDrivers`. (On a typical machine, the two values under this key should be `VGA; MNMDD`.)

Note For systems with multiple monitors, make sure that the order of the included monitors is the same as the order shown in the Control Panel's display settings.

If UVD still fails, there are still a few things that you can do to get SoftICE working on your machine. Try starting SoftICE in boot mode, then manual mode, and see if one works. You can change the start mode by running the Start Mode Setup from the Start menu. A final option would be to launch SoftICE from a command prompt with the command

```
net start ntice
```

This command will always launch SoftICE using the VGA driver.

Legacy Video Driver

To correctly configure SoftICE to use the Legacy Video Driver, follow these steps.

- Run the SoftICE video setup. If your card appears, select it in the two list boxes. Otherwise, select VGA in the two list boxes. DO NOT check the UVD selection box. When you click the TEST button, the changes will be recorded.
- Run the OS's display configuration (using the Display entry in the control panel) and select standard VGA at a resolution of 600 x 800 x 256.

Miscellaneous Configuration

This configuration allows the OS and system applications to use one video card (and driver), while SoftICE displays on the alternate card (and driver). These alternate displays can be VGA or Monochrome.

To use these drivers, place the command `altscr mono` or `altscr vga` in the SoftICE initialization settings. This tells SoftICE to write directly to the monochrome or VGA space. (This works as long as the primary video card does not write into the same space.)

Remote Debugging

If all of the above attempts fail, then, as a last alternative, use SoftICE's remote debug capability. Check out the README and Help files; also, explore our knowledge base at:

http://frontline.compuware.com/nashua/kb/

Chapter 6
Host and Target Configurations

This chapter explains how DriverStudio and SoftICE Driver Suite use the concept of Host and Target for remote debugging.

User and Kernel Mode

Contemporary computer processors implement the difference between ***User*** and ***Kernel*** operation. The hardware reserves a small but key portion of its functionality as "Kernel Mode," available only to the Operating System. When a user program attempts to access kernel mode functionality, a machine error ensues.

User Mode refers to operating the machine to run user programs, which are isolated from each other and from the operating system to prevent cross-interference and to keep user programs from interfering with the Operating System. Programs running in User Mode cannot exercise functionality classified as "Kernel Mode". Operating systems run user programs in User Mode, preventing them from directly accessing Kernel Mode functionality.

Kernel Mode is reserved for the Operating System. When operating in Kernel Mode, the full processor functionality is available. Kernel Mode is reserved for the Operating System and its components. Device Drivers are considered to be an integral part of the Operating System, therefore they run in Kernel Mode.

DriverStudio includes both User and Kernel Mode components. DriverStudio applications such as TrueCoverage or TrueTime usually include a User Mode component and a Kernel Mode component. (See Table 6-1.)

Table 6-1 DriverStudio Components

PRODUCT	USER MODE COMPONENTS	KERNEL MODE COMPONENTS
SoftICE	SIREMOTE, Symbol Loader	ntice.sys, siwvid.sys, nmfilter.sys
BoundsChecker	DriverWorkbench	bchkd.sys
TrueTime	DriverWorkbench	x9tt.sys
TrueCoverage	TrueCoverage Application	x9.sys

Frameworks wizards used by DriverWorks and DriverNetworks operate mostly in User Mode, but generate Device Drivers that will run in Kernel Mode. These Frameworks products also include many sample drivers, which also run in Kernel Mode.

Hosts and Targets

DriverStudio is an integrated suite that aims at Kernel Mode software development. Because Kernel Mode software is a part of the Operating System, sometimes it is difficult to develop such software without interfering with the operation of the development machine itself! So, the concept of ***Host*** and ***Target*** machines was born. Simply stated, the concept is this: Software is developed at a Host machine, but will run at a Target machine.

Host machines will, therefore, host the development environment. This includes the Microsoft DDK, the C compiler and linker used to compile the new code, the build tools and batch files required to build the product, and a host of development aids. The kernel developer who is in charge of developing the driver sits at the Host machine and controls both development and testing from his system.

Target machines receive the drivers to be tested. There's no need to have a full development environment on a Target machine. The DDK, compiler, and linker belong in the host system. But once the developer has a built driver that needs to be tested, that driver gets loaded into the Target machine, and the developer can control the Target remotely from his Host machine and do the debugging and testing work there.

A typical sequence of events would be:

1 The developer implements new code at the Host machine, then compiles, links and builds the code into a new device driver.

2 The developer uploads the device driver executables from the Host to the Target machine.

3 The developer installs the driver at the Target machine, rebooting it if necessary.

4 The developer proceeds to debug the Target machine from the Host.

In its simplest incarnation, this concept only involves one *Host* and one *Target*, and this is how many debuggers work.

DriverStudio's Approach

Prior to Release 2.5, DriverStudio only supported a simple serial or TCP/IP connection, and only with SoftICE. When not operating in single machine mode, SINET (now **SIREMOTE)** was run on the Host, and that allowed us to 'pop up' SoftICE on a target machine and to debug over a Serial or TCP/IP link.

With the advent of DriverStudio Release 2.5, a much wider networking concept was implemented. The functionality of DriverStudio was split into Host and Target functionality, by and large along the lines of User and Kernel mode operation. DriverStudio was redesigned to have User Mode components operating mostly in the Host machine, while Kernel Mode components operated in the Target machine.

Now, for example, we can run DriverWorkbench in a Host machine, and run TrueTime or BoundsChecker analysis at the Target system, while before Release 2.5, we were limited to running both User and Kernel functionality on the same system. DriverStudio 2.5 brought the separation of functionality along Host and Target lines, and now three kinds of configurations are allowed:

- ***Host Configurations*** including User Mode components of DriverStudio, such as the DriverWorkbench, the TrueCoverage application, the Frameworks Wizards, the SoftICE Symbol Loader, and the SoftICE remote user program, SIREMOTE. You can debug any Target machines visible over the network. Control of these remote Targets can be done through the Namespace Extension. (See *Using SoftICE*, Chapter 9, for more information and screen shots.)
- ***Target Configurations*** include Kernel Mode components such as the SoftICE, TrueCoverage and TrueTime drivers. Note that this configuration requires a Host machine to do debugging.
- ***Single Machine Configurations***, where both host and target functionality are configured. With this configuration you can do single machine debugging.

DriverStudio now implements a Distributed Protocol that allows it to have multiple Hosts and multiple Targets on the same network. In addition, DriverWorkbench is now equipped with a remote Target choice window, so that, for example, a developer sitting at a Host machine can link to remote Target machines and perform BoundsChecker and TrueTime analysis remotely at the Target. Virtually all functionality available through the DriverWorkbench is now available both at the local machine and at a remote Target, as long as the Host machine can "see" that target over the network. To help with this, DriverStudio now includes the Namespace Extension (see *Using SoftICE*, Chapter 9), which enhances the level of remote control a Host developer can exercise over multiple remote targets accessible through a TCP/IP network.

However, some User Mode components still run the old way, that is, they must be run at the Target Machine. (An example is TrueCoverage operation.) The Symbol Loader can be run on both the Host and Target machines.

What Runs in the Target

DriverStudio Target components mostly consist of Kernel Mode Drivers. These include:

- The SoftICE driver, ntice.sys.
- SoftICE support drivers, for example, siwvid.sys and nmfilter.sys.
- The BoundsChecker driver, bchkd.sys.
- The TrueTime Data Collection driver, x9tt.sys.
- The TrueCoverage Instrumentation driver, x9.sys.
- The Symbol Loader. (This is the only GUI component that can run on both the Host and Target sides.)
- The DriverStudio configuration components.
- Debug Message handling components.

Target components mostly perform Kernel Mode functions at the Target machine. This includes most of SoftICE, plus the Kernel Mode drivers for BoundsChecker, TrueTime and TrueCoverage, and other Kernel Mode functions. None of the User Mode GUI programs (with the exception of the Symbol Loader) can run in the Target system; neither are the SIREMOTE components of SoftICE a part of the Target Configuration.

What Runs in the Host

Host configurations run User Mode components of DriverStudio. These include:

- SoftICE user-side programs such as SIREMOTE.
- The SoftICE Symbol Loader. (This application can also run on the Target side.)
- Namespace Extension.
- DriverWorkbench, which provides access to the following User Mode components:
  - BoundsChecker functionality
  - TrueTime functionality
  - Remote Configuration
- Frameworks Wizards:
  - DriverWorks
  - DriverNetworks
  - VtoolsD
- Frameworks Tools:
  - DriverMonitor
  - EzDrvInstaller
  - Src2Dsp
  - SetDDKGo
  - Symbolic Link Viewer
- Help Components.

With the exception of the Symbol Loader, these User Side, GUI-intensive components, do not run on a Target machine. You need a Host Machine in order to run the GUI components of DriverStudio.

SoftICE Operation

SoftICE runs on the Target side, except for the SIREMOTE user interface. The SoftICE Symbol Loader also runs on the Target Side. However, a user sitting at a Host Machine can popup SoftICE at a remote Target, either by invoking SIREMOTE directly, or by invoking it through the Namespace Extension. The Host/Target architecture allows

a Host user to popup SoftICE at several different Target Machines, as long as the Target is visible over the TCP/IP network.

Symbol Loader Operation

The SoftICE Symbol Loader allows the Kernel Developer to integrate executable code, source code, and symbols into a single package that can be loaded into SoftICE. The Symbol Loader is built around a workspace concept to allow tailoring to the individual developer's directory structures. The new SoftICE Symbol Retriever provides access to any symbol site accessible over a network. This allows organizations to host symbol servers that can be accessed by employees or others over a network or the Internet.

Namespace Extension

The Namespace Extension is a host-resident component. It does not run on the target machine, although it can present to the host user a unified view of all hosts and targets that are reachable through the communications internet. This view also includes remote control over SoftICE running on a DriverStudio target.

DriverWorkbench Operation

DriverWorkbench runs in the Host Machine as a user shell that accommodates the functionality of BoundsChecker, TrueTime, and the DriverStudio Remote Configuration Utility (DSConfig.exe). Driver-Workbench also has the ability to connect to a remote Target Machine: when this is done, any Kernel Mode functionality of BoundsChecker, TrueTime or DSConfig.exe runs at the Target Machine. So, for example, you can open DriverWorkbench at the Host, connect it to a Target Machine, and run BoundsChecker or TrueTime analysis and data collection at the remote Target, controlling the whole process from the Host. You can also remotely configure your Target by invoking DSConfig.exe at the Host Machine after you have connected to the appropriate Target.

Remote Configuration Operation

DriverStudio allows a Host user to connect to a remote Target and remotely configure that Target Machine by running DSConfig.exe at the Host. The Target settings are displayed at the Host, and any changes made are saved and implemented on the Target. Remote configuration can be exercised either through DriverWorkbench or through the Namespace Extension.

BoundsChecker Operation

BoundsChecker functionality is implemented through a Target Side driver and a Host Side DriverWorkbench component. If you install the bchkd.sys driver in the Target system, you can connect to that Target from a Host running DriverWorkbench and exercise the available BoundsChecker functionality by picking up the data that has been collected by the BoundsChecker driver. BoundsChecker Target-side files are visible to the DriverStudio Namespace Extension, which runs on the Host side. Target-side information can also be captured by SoftICE and displayed from a Host system through the normal SoftICE mechanisms such as a local CTRL-D or a remote access through SIREMOTE.

TrueTime Operation

TrueTime data collection is performed at the Target by the x9tt.sys driver. That data is sent to the Host where it is reduced and built into analysis reports by DriverWorkbench. All the control and reporting is done on the Host side, while data collection is performed on the Target side. The Target driver is capable of saving TrueTime data files (with the .ttd extension) under control of the Host: these files can be retrieved by the Host Machine and analyzed by DriverWorkbench. TrueTime .ttd files are visible through the DriverStudio Namespace Extension on the Host.

TrueCoverage Operation

TrueCoverage is not included in the DriverWorkbench frame, therefore it cannot take advantage of DriverWorkbench's new Host/Target architecture. We must have both User-side and Kernel-side components of TrueCoverage running on the same machine – by definition, a Target Machine as far as TrueCoverage is concerned.

Frameworks Operation

There are three frameworks products: DriverWorks, DriverNetworks, and VtoolsD. DriverWorks supports the rapid development of C++ driver code. DriverNetworks offers the same support for network driver development. VtoolsD allows development of VxDs for Win9x. All of these products have wizards to simplify initial code development. The Frameworks wizards and utilities run on the Host side. The interface to the compiler and to the DDK run on the Host machine.

Frameworks Tools

DriverStudio provides the following Frameworks Tools:

- **DriverMonitor** - Displays user and kernel mode debug messages and loads/unloads drivers.
- **SetDDKGo** - Sets up a device driver build environment.
- **EzDriverInstaller** - Installs WDM drivers on Windows 2000/XP.
- **Symbolic Link Viewer** - Shows a list of symbolic links on Windows NT/2000/XP.
- **SrcToDsp** - Generates a DSP for a driver that builds with a SOURCES file.
- **BoundsChecker Custom Events Library** - Puts BoundsChecker events in a DDK driver.
- **DspToDsp** - Convert older DSPs to build with the XP DDK.
- **MSVC DDK Build Settings** - Change the driver build environment from within Visual Studio.

Summary

There are two basic DriverStudio remote debugging configurations: Host Systems and Target Systems. In addition, there's also the possibility of configuring a machine to be both a Host and a Target for use in Single Machine debugging. Host functionality mostly consists of User-side GUI-intensive programs. Target functionality mostly consists of device drivers that run in Kernel Mode. Depending on the particular application, it can be done mostly in the Host (for example, DriverWorks or DriverNetworks), mostly in the Target (for example, SoftICE), or distributed across Host and Target (Remote Configurator, TrueTime, BoundsChecker). The user sits at the Host Machine, from where he or she can control multiple Target Systems, over a TCP/IP Network.

Index