Windows NT Embedded

Step-by-Step

Sean D. Liming

Annabooks
San Diego

Windows NT Embedded Step-by-Step

by

Sean D. Liming

PUBLISHED BY

Annabooks
12860 Danielson Court
Poway, CA 92064
USA

858-391-5600
http://www.annabooks.com

Printed in the United States of America
ISBN 0-929392-68-X
First Printing May 2000

Resource Requirements

Development System:

Hardware Requirements:

The minimum development system requirements are a 200 MHz or higher Pentium-class processor, 64 megabytes (MB) of RAM, and 20 MB of free disk space to install the development system. Additional disk space will be required to store the generated systems, which would also need to include any page file space. Therefore, 70 MB of free disk space is recommended. A CD-ROM is required to hold the OS repository. A CD-ROM burner is required for Exercise 10.

Software Requirements:

Development system software requirements include Windows NT version 4.0 with Service Pack 4, or greater. A Windows NT Embedded Evaluation Kit can be purchased from an authorized distributor, such as Annasoft (www.annasoft.com) for a nominal price. In addition, the online Help requires IE 4.0 SP1 or IE 5.0. To complete Exercise 6 which covers the creation of new Target Designer eXtensions (TDXs) for new components, you must have the Microsoft Visual Studio 6.0 development environment. To complete Exercise 10, you'll need the WinOnCD CD creator software package (see the Resources section). Winternal's Remote Recover is highly recommended for downloading NT Embedded images to the target.

Target System:

We strongly recommend that you purchase the Aaeon 5894 platform and associated hardware (see Section 1.9). It's the best way to take full advantage of Exercises 1-9, 11, 12, and 14. Additional hardware will be required for Exercises 9, 10, 12, and 13.

Acknowledgements

There are a number of people I wish to thank, for without their support this book might not have been possible. First, I would like to thank Paul Morse and Jon Fincher of Microsoft who helped answer many of my questions regarding the new features of Windows NT Embedded, and for the feedback they provided with regard to contents of the book. Without their support this book would have never been possible.

There are a number of people at Intel I wish to thank, specifically Troy Smith, Byron Gillespie, Yvette Maldonado, and especially Sheeba Bedi and P. Sean Key. The various Pentium development kits, fast response to my questions, and quick delivery of the reference designs helped in the development of the multiprocessor section, among others.

Fred Fischer of AAEON was a lifesaver when it came time to get boards for the NT training class. Most of the exercises in the book are designed around the AAEON 5894.

Raz Dan of M-Systems helped explain the intricacies of the DiskOnChip NT driver. I am very grateful to Kevin Wilson of Winternals and Stephen Thomas of PowerQuest for working with Annasoft to create special price offerings for their software products. I would also like to thank many of my students who provided feedback and inputs to the course material.

The Annabooks sales team has been very patient while waiting for me to finish this book .... and waiting and waiting....well, it's finally here. Finally, I am very grateful to have the best engineering staff in the world. The entire group was able to put up with my constant travels, classes, and other absences, both physically and mentally.

Dedication

To my parents, and to overachievers everywhere.

Contents

Foreword

The computing landscape is rapidly evolving to include new types of computing devices, many of which are created for a specific function and contain an embedded operating system. These devices include telecommunications equipment like edge-of-network equipment, office equipment like copiers and printers, industrial equipment like robots and plant control devices, television set-top boxes, and a wide variety of Internet appliances.

Because of this new computing evolution, embedded system design engineers are being required to expand their knowledge and meet the design needs of increasingly complex and interconnected devices. Windows NT became popular with engineers when they discovered it provides a solid foundation for building embedded systems with its high-performance kernel, pre-emptive multitasking, SMP support, built-in security, scalable asynchronous I/O model, rich layered services, built-in networking, and the ubiquitous Win-32 API. However, there were still issues that needed to be addressed, including fault resilience, error recovery, target footprint management, modularity, solid-state operation, remote device management and troubleshooting, and the ability to either replace the existing shell or run it as a completely headless system. Windows NT Embedded 4.0 adds these, and other, capabilities to Windows NT, which enables designers to employ a very powerful embedded operating system in their next generation of solutions.

Embedded system designers around the world are using Windows NT Embedded in very clever and exciting ways. These myriad designs, which are quickly coming to market, are testament to the power and rich functionality of the Windows NT Embedded product and the platforms that are created with it. As with any general purpose embedded operating system and development platform that is gaining wide acceptance, additional knowledge about how to use the product is always in demand.

This demand for information is why *Windows NT Embedded Step-by-Step* will be of significant value to Windows NT Embedded developers. Sean's book takes a refreshing approach to seemingly complex subject matter. Rather than simply providing line after line of text and code, the book actually walks you through exercises that can be completed with a desktop development system and target device. There's no substitute for learning by doing, and the breadth of the exercises contained within provides a knowledge base that will assist you in quickly creating new, functionally rich solutions.

Paul Morse

Product Manager
Windows NT Embedded
Microsoft Corporation

Preface

The embedded market has changed dramatically over the past five years. The technology in the embedded PC market used to lag technology in the desktop and server market by three years. Today the embedded market is right in step with the desktop technology and driving the technology limits further.

There are several driving forces that are propelling the embedded market. The Internet has blossomed to connect everyone in the world to everyone else. The ability to access information anywhere and any time has spawned the growth in mobile PCs and wireless connectivity technologies. There has been an explosion of low powered 32-bit RISC CPUs for these mobile systems. And not to be forgotten, Microsoft has a larger presence in the market than ever before. As we say here, we are entering the age of Everywhere Computing!

When I first joined Annasoft in 1995, Microsoft was paying relatively little attention to the embedded market, even though many OEMs were embedding MS-DOS and Windows 3.1. There was a growing interest from OEMs who wanted to use Windows 95 because they wanted a 32-bit OS that had a familiar API. I wrote a white paper that discussed how to shrink Windows 95 for smaller read / write flash systems. The interest in Windows 95 and Windows NT began to grow, and many developers were calling to ask if Windows 95 could be made ROMable (as was done for the MS-DOS and Windows 3.1 predecessors).

Microsoft had been developing a small OS for embedded systems since the early 90s, and when Windows CE first arrived the market interest was overwhelming. Microsoft first introduced Windows CE for handheld systems, and had to immediately plan the road maps to extend support for embedded devices. While Windows CE supported the x86 instruction set, there was very little driver support, with the result that Windows CE wasn't quite ready for embedded PCs. Annasoft created Jump Start, which contained various drivers and a loader to support a wide variety of PC-compatible board manufacturers. Windows CE has continued to grow and create new markets and opportunities.

Windows NT took a different path. Microsoft had designed Windows NT to be portable to different underlying hardware, although only Intel Architecture was readily available. OEMs used Windows NT for robust communications, security, and inexpensive off-the-shelf platforms. A few years ago the large footprint and the high cost of flash made it difficult to justify using NT in an embedded system. At a 1996 Embedded Systems Conference, there were some engineers that walked by our booth and laughed at the idea of embedding Windows NT.

The OEM Pre-installation Kit (OPK) was provided to OEMs to pre-install Windows NT on target systems. VenturCom created a tool called Component Integrator that allowed you to create small (under 20 MB), custom Windows NT images from the distribution CD. Component Integrator interfaced to a database that had a list of the components that make up the OS. However, the high cost of the licensing the OS, the price of the tools, and the expense to embed NT in non-rotating media was not cost effective for OEMs building a thousand units or less. Most of the OEMs licensing Windows NT for embedded or dedicated applications were using it "as is", and storing it on conventional rotating media.

Windows CE paved the way for Windows NT Embedded. The timing was perfect. Windows CE helped Microsoft create a large presence in the embedded market. The price of flash has come down, and there are a number of inexpensive ATA flash cards and other flash drive solutions. More and more companies are implementing systems with 32-bit microprocessors, especially Pentium processors. What drew people to Windows CE was the familiar programming interface, and the ability to find programmers to develop applications. The same is true for Windows NT Embedded.

The biggest appeal NT Embedded provides is the potential for a short development cycle. Instead of recompiling the OS, NT Embedded allows you to select individual features that are already built and tested. Microsoft made a number of enhancements to Component Integrator and included some of the popular NT Resource kit utilities, new embedded components, Component Designer, and released the product as Windows NT Embedded.

Through my years at Annasoft, I have seen all of the Microsoft operating systems, MS-DOS, Windows, Windows 95, CE, and NT, being implemented in some device or another. I have talked and met with a wide range of developers, who are under constant pressure to design quickly and cost effectively. There is very little time for them to learn a new product. I have created classes for Windows CE and NT and have written various articles to shorten the learning cycle, but nothing beats working with a product hands-on.

The best way to learn a new software package is by doing. There are many books that teach an express method for learning an application or a computer language (i.e., Learn Visual Studio in 30 Days). For the most part, this book is designed in the express learning or "cookbook" method. Each exercise builds on the next to introduce new features and point out particular methods. The exercises are designed to be generic enough to reach a wide audience.

Like the title of the book, the exercises are in a step-by-step presentation. Every effort has been made to remove any mistakes, but of course there are always a few that slip by. The exercises were developed largely from customer feedback, student inputs, and the need to cover the basics and as many features as possible. There are many features, components, and combinations that can make an NT Embedded image, and the exercises cover the main features from a high level. It is difficult to go into detail of each component and feature so your feedback is important. If there is a topic you're interested in or have any comments or questions, please e-mail me at seanl@annasoft.com. Also check the Annasoft website (www.annasoft.com) for more information on special offers.

Best of luck on your NT Embedded Development!

Sean D. Liming

San Diego, California

Chapter 1 INTRODUCTION

A few years ago Microsoft launched Windows CE as a compact version of Windows intended for use in applications for embedded systems and consumer electronics. The goal was to bring value added software to applied computing systems. Almost immediately, OEMs wanted real-time performance, multimedia support and other features found only in Windows 95 and Windows NT. The overwhelming response to Windows CE, the continual growth of the PC in the embedded market, and future Internet appliance possibilities encouraged Microsoft to take a closer look at the dedicated / embedded market.

Before Windows CE, those who wanted an off-the-shelf OS and didn't want to pay for a specialized RTOS had a choice of MS-DOS, Windows 95, OS/2, UNIX, or Windows NT. If you were looking for an OS that had a familiar API, security, and communication robustness, the choice was Windows NT. Typical systems that use Windows NT are communication systems, security systems, factory automation, call centers, multimedia servers, network printers and copiers, and database servers. Some of these systems require headless operation, small footprint, and ROMable support. Windows NT was not originally designed for any of these requirements. Many third party tools were developed to help embedded developers create and implement Windows NT systems. Much to their credit, these third-party tool developers created products that brought Windows NT to the applied computing market, but combining the price of the tools with the price of Windows NT made implementing NT an expensive proposition. Price sensitive products and markets often went with another OS.

Microsoft purchased several pieces of technology to develop Windows NT Embedded 4.0. The goal of Windows NT Embedded is to support those of you who want a rich API environment, want to pay little for OS development, need special embedded features, and want the ability to use off-the-shelf software. Windows NT Embedded is poised as an ideal solution for applied computing OEMs who want to implement their systems using a robust and well-tested OS.

1.1 What is Windows NT Embedded?

Windows NT Embedded is a new version of the Microsoft Windows NT 4.0 operating system specifically designed to address the needs of embedded systems OEMs. Windows NT Embedded is a combination of Windows NT 4.0 with Service Pack 5, embedded features, authoring tools, and various tools from the Windows NT Resource kits.

Microsoft didn't change a single binary or code base to create Windows NT Embedded. By not changing the binaries, you have the ability to run any off-the-shelf software and driver packages designed for Windows NT 4.0 Workstation and/or Server.

Windows NT Embedded allows you to select the components that you want as part of your system. The operating system is broken down and grouped into a variety of capabilities and components that are listed in a database called Target Designer System Database (TDSD). Microsoft created tools that interface to the TDSD in order to build NT Embedded images and add more components. The two authoring tools are called Target Designer and Component Designer.

- Target Designer - Target Designer is the main interface to the TDSD. Target Designer is used to create the configurations, import new components, set the boot options, check the dependencies, manage locations of all of the component repositories, and build NT Embedded images.
- Component Designer - Component Designer is used to create components that can be added to the TDSD via Target Designer.

Microsoft provides other helpful tools to help with system development. These other tools help you diagnose your target hardware, provide remote connections to your target, debug your OS image, and assist with deployment.

1.2 Development Overview

Development work is simplified because NT Embedded is the exact same OS as NT 4.0 with Service Pack 5. You use the off-the-shelf version of NT 4.0 with Service Pack 5 to develop and test your application before you commit to purchasing the NT Embedded tools.

Unlike other embedded operating systems, Windows NT's popularity helps you regarding development tools and device drivers. NT device drivers are readily available for almost all PC devices sold on the market today.

The basic development steps are as follows:

1. Develop your application in Visual Studio.
2. Build your product with your application and an off-the-shelf version of NT 4.0 with Service Pack 5.
3. Test the system to see if it meets the product requirements.
4. If the product requirements are met, purchase NT Embedded.
5. Create components for your application, third party applications, and/or device drivers with Component Designer.
6. Use Target Designer to build a custom NT OS.
7. Download OS to target and test final system.

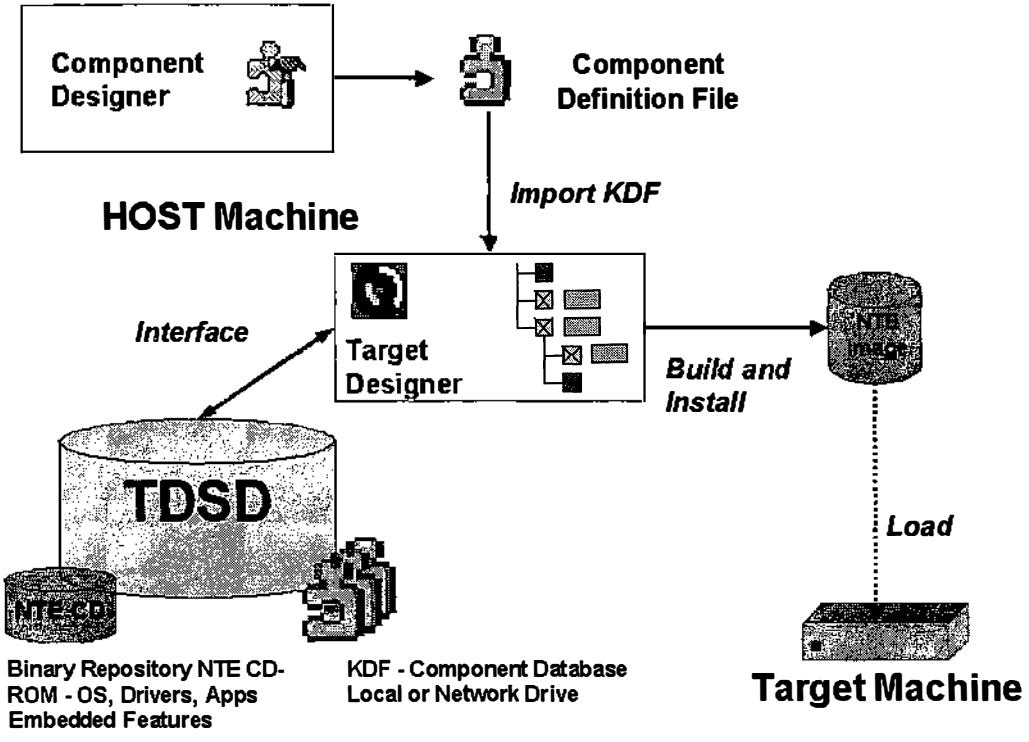


Figure 1.1 NTE Development Overview

1.3 Embedded Systems Support

As part of the TDSD, Microsoft has included some embedded components that are not part of the standard Windows NT release. These components allow Windows NT to be deployed in a variety of applied/embedded systems. These components include headless operation, read-only boot support, solid-state media, remote management, virtual page sizing, SID and PID cloning, and error recovery. Some of the Windows NT Resource kit's utilities are also included. Chapter 3 will discuss these components in greater detail.

1.4 What About the Size?

As you would expect, Windows NT is not a small operating system. Yet with the price of flash drives, hard drives, and RAM continuing to fall, Windows NT more often becomes an affordable solution. Target Designer allows you to select only the components you need for your system. A smaller footprint is the result. Table 3 lists preliminary sizes of some typical configurations. Your final OS image will vary depending on the application.

Target	Features	RAM	Persistent Storage
Headless Standalone Workstation with No Networking	Minimal OS and Command Console	12 MB	9.5 MB
LAN Router	Minimal OS + Networking + Routing and RAS	16MB	25MB
Server Appliance	Minimal OS + Networking+ IIS+ PDC + DHCP+WMI	32 MB	40MB

Table 1.1: OS Sizes for Various Applications (Courtesy Microsoft Corp.)

1.5 NTE Summary

Windows NT already provides an excellent foundation for building embedded systems: a high-performance kernel, pre-emptive multitasking, scalable asynchronous I/O model, rich layered services, and the well-known and widely-used Win-32 API.

Windows NT Embedded includes all of these features and brings three important features to applied systems OEMs. The first is the authoring tools, which automate most of the development work. The second is the price, which is cheaper than the mainstream tool and licensing cost. Third are the embedded components that provide the support for taking Windows NT to the applied computing market.

The biggest benefit is that the Windows NT code base is the most widely tested OS over the past seven years. These features and benefits make Windows NT Embedded an ideal OS solution for applied computing systems.

1.6 About This Book

There are many topics to cover within Windows NT. This book is focused on Windows NT Embedded and the use of Target Designer, Component Designer, and other tools. Chapter 7 lists a variety of publications that provide help on more detailed topics such as IIS or networking.

Because Windows NT Embedded is an easy-to-use tool set, Annasoft Systems has implemented the Learn-Do-Ship program for Windows NT Embedded. Hands-on exercises are the best way to learn any new software package. Each chapter and exercise builds on the former in order to teach a different concept or feature of NT Embedded. Although you may find that the exercises may not cover your specific development requirement, they will cover the general development process and concepts.

Exercise 1a, 1b, 1c, 1d - Exercise 1 is broken into four parts that will take you from selecting the components, building the OS, preparing the target, downloading the OS from host to target, backing-up configurations, and booting from either FAT or NTFS file systems.

Exercise 2 and Exercise 3 - These two exercises cover some of the embedded components that are only provided in NT Embedded.

Exercises 4, 5 and 6 - Both of these exercises cover the development of Kit Definition Files, also known as Component Definition Files or "KDFs", and introduce different methods and tools required to create KDFs. Exercise 4 covers the creation of a KDF for an application. Exercise 5 covers the development of a KDF for a hardware (network) driver. Exercise 6 covers Target Designer extensions (TDX) development for KDFs.

Exercises 7 -14 – The last set of exercises use various components and features of NTE and reinforce the earlier exercises. The exercises feature different topics such as dual network cards, graphical remote administration, KDF development, dual processing support, and cloning. Most of these exercises are dependent on Exercise 5.

1.7 Development System Requirements

1.7.1 Hardware Requirements:

The minimum development system requirements are a 200 MHz or higher Pentium-class processor, 64 megabytes (MB) of RAM, and 20 MB of free disk space to install the development system. Additional disk space will be required to store the generated systems, which would also need to include any page file space. Therefore, 70 MB of free disk space is recommended. A CD-ROM is required to hold the OS repository.

1.7.2 Software Requirements:

Development system software requirements include Windows NT version 4.0 with Service Pack 4, or greater. Windows NT Embedded can be purchased from an authorized distributor, such as Annasoft (www.annasoft.com). (By the time this book is published, an NTE evaluation kit should be available at very low cost.) In addition, the online Help requires IE 4.0 SP1 or IE 5.0. If you wish to create new Target Designer eXtensions (TDXs) for new components (Exercise 6), you must have the Microsoft Visual Studio 6.0 development environment.

The exercises also make use of third party software that can be purchased/downloaded from the Internet. You may or may not choose to use these, but these tools have been found to help speed the NTE development process:

Company/Individual	Product	Web site
Winternals	Remote Recover	www.winternals.com
	FileMon	or www.annasoft.com (special offer)
	RegMon	
Jean-Claude Bellamy	Scanbin	http://www.bellamyjc.net/
Marco Koetsved	PCISNIFF	http://members.hyperlink.net.au/~chart/index.htm

Table 1.2: Third Party Tools

You will also need a DOS boot disk with FORMAT and FDISK. A DOS boot disk image can be found on the first NTE CD under MUNGEBOOT. The disk contains the tools necessary to patch a hard drive's boot sector, but doesn't contain FDISK or FORMAT utilities.

1.8 Target System Requirements

Target system hardware requirements vary, depending on the selected operating system features and applications that you select for your configuration. The TDSD lists a few embedded PC platforms that have been tested with NTE. There are other off the shelf PC platforms available from several embedded PC manufacturers. As a minimum configuration, you will need a 100 MHz 486-class processor with 16 MB of RAM. Windows NT Embedded will support 486DX and Pentium-class processors.

1.9 The Target for the Exercises

Any PC platform with a 486 processor or greater can be a target for NTE. The exercises use the following target platform hardware:

- Pentium Processor
- VGA
- IDE Hard drive
- Floppy
- 1 Serial Port
- PCI Bus
- 10/100 Network Card
- NE2000 PCI Network Card
- DiskOnChip

The AAEON 5894 platform was chosen to develop most of the exercises because of the feature set. The AAEON 5894 supports the following:

- Pentium Processor
- C&T 65550 Video
- IDE
- Floppy
- DiskOnChip
- 4 Serial ports
- Parallel Port
- USB (OHCI)
- 10/100 Ethernet RealTek 8139
- PCI Bus

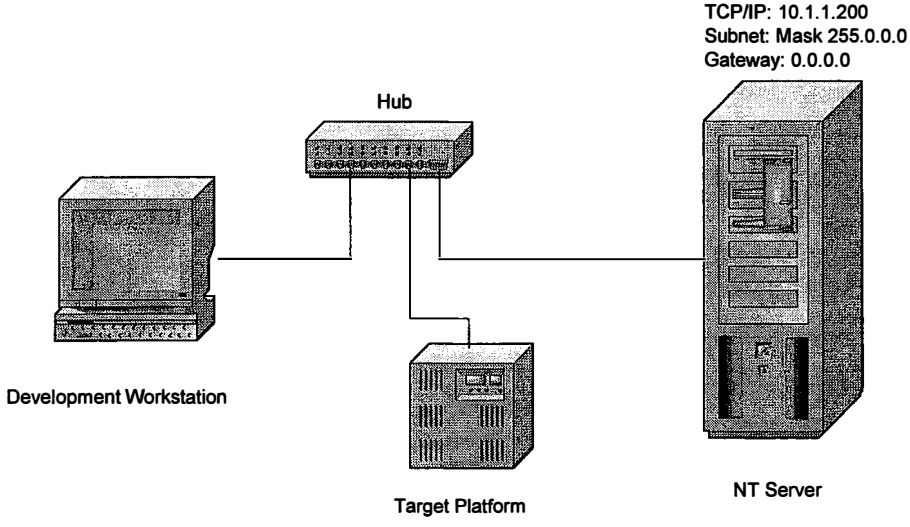
Although some exercises may require this platform, the general concepts apply to all systems. In addition to the AAEON 5894 or equivalent system, you will need an NE2000 card, LAN cables, NULL modem cable, keyboard, PS/2 mouse, 8 MB DiskOnChip, hub to create a network, and a DHCP server. Exercises 9, 10, 12, and 13 will require specific hardware for CD-Boot, SCSI, dual network cards, and Dual Processing.

You may purchase the AAEON platform from:

EMAC Inc.
www.emacinc.com,
phone: (618) 529-4525
e-mail: info@emacinc.com.

1.10 Development Setup

The basic development setup requires a network connection between the development system and the target hardware. The illustration below shows a typical installation.



Typical Development Setup

Chapter 2 TARGET DESIGNER - CREATING NTE IMAGES

2.1 Target Designer Overview

Target Designer is the primary authoring tool for Windows NT Embedded. Target Designer allows you to define the Windows NT functionality that makes up the target system. Once you define your configuration, Target Designer is used build the bootable NT target image. Unlike other RTOS approaches, which use a compile-to-target model, Target Designer takes the selected components and builds an image from binaries. Of course, you will have to verify that all of the components are in place for applications to run correctly. Target Designer lets you create server or workstation, headless, small OS or standard, and disk boot or flash boot images.

2.1.1 Target Designer Breakdown

As discussed in Chapter 1, Target Designer interfaces to the Target Designer System Database (TDSD). The TDSD is a database of collections, configurations, capabilities, and components. Like Explorer, the TDSD is displayed graphically on the left pane of the Target Designer screen as a hierarchy tree.

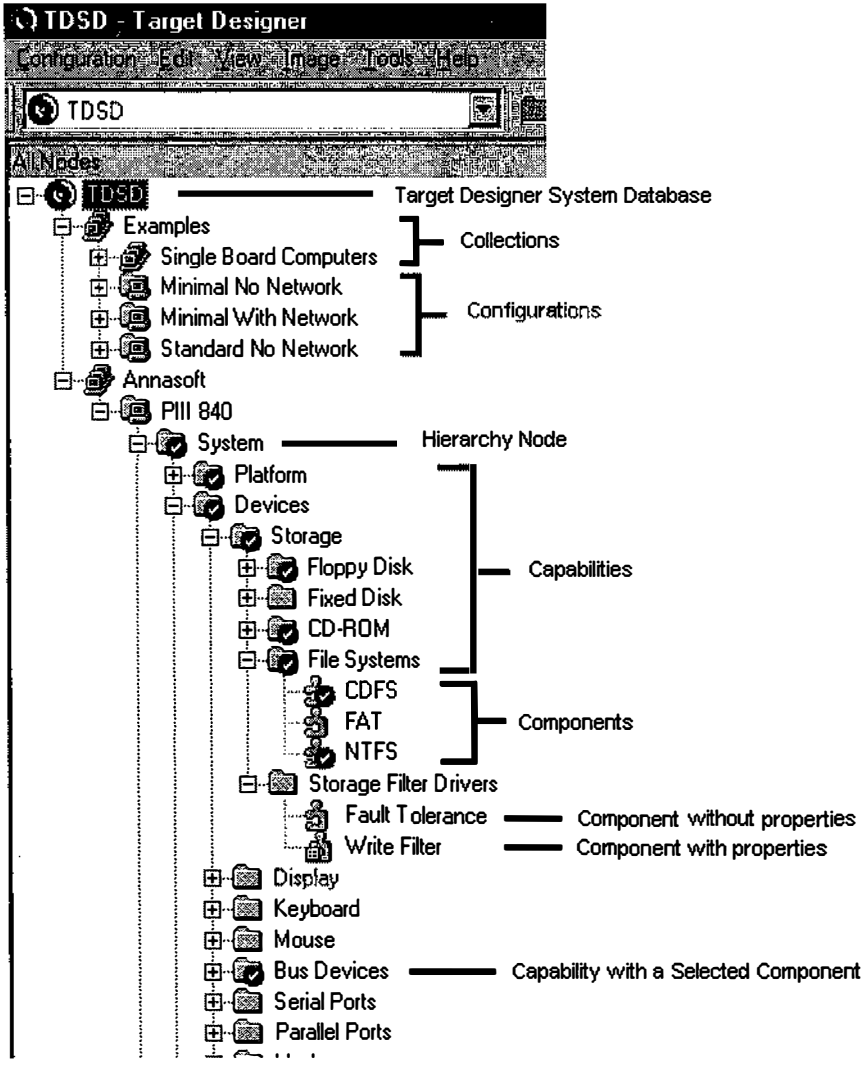


Figure 2.1: Target Designer Tree

Hierarchy Term	Symbol	Description
TDSD		The root of the tree, which includes the entire database.
Collection		A set of configurations or other collections that are grouped together for administrative or management convenience. For example, **Annasoft** is a collection).
Configuration		Defines the target image that is created. The configuration contains a tree of pre-defined hierarchy nodes. For example, **PIII 840** is a configuration.
Hierarchy node		A predefined, static node in the configuration tree that defines a sub-tree with a common functionality. Hierarchy nodes include capabilities. For example, **System** is a hierarchy node.
Capability		Includes a set of components. For example, **Fixed Disk** is a capability.
Component (Component without Properties)		The basic element in the configuration that defines files and registry entries. For example, **NTFS** and **CDFS** are components
Component with Properties		Components with properties have an added document symbol within the puzzle icon. When a component is selected, the properties become available for editing.
Capability with a Selected Component		When a component is selected, a green check icon is placed next to the component and the Capability that the component is under.

Table 2.1: Target Designer Terminology

2.1.2 Menu and Toolbars

Depending on where you are in the scope of the hierarchy, the menu and toolbars provide the operations and commands to add components, files, registry keys, cut, paste, etc. The menu bar is broken down as follows:

Configuration - The configuration menu provides the management options for configurations. You can create new collections or configurations, set a configuration's properties, rename a configuration, import, and export.

Edit - The edit menu provides the basic cut, copy, and paste functions. Depending on the scope of the TDSD hierarchy, you can add, delete, restore, and edit registry entries, components, or files.

View - The view menu allows you to select the right pane view: files, components, or registry.

Image - The image menu is used to manage the component repository locations and provides the build options.

- Check Dependencies - Verifies that the relationships between all of the components are satisfactory.
- Build and Install - This is a single operation that compiles the registry and installs a configuration's image to the target directory set in the Image->Settings menu.
- Build Only - Compiles the registry and performs the bindings, but doesn't install the OS image.
- Install Only - Installs the image to the target directory set in the Image->Settings menu.
- Clone Target - One step operation to make a copy of an image and provide a unique SID.
- Settings - Manages the drive / directory locations for the OS and component repositories. Also, sets the target image path.

Tools - The tools menu has two options: Component Management allows you to add and remove KDFs (components) to the TDSD. Component Designer is the second authoring tool set that allows you to create components.

Help - Target Designer Help

The Toolbar provides quick button access to the most frequently used menu operations.

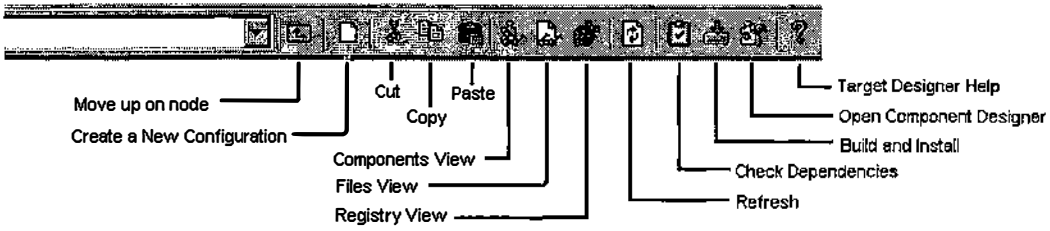


Figure 2.1b: Target Designer's Tool Bar

You can add capabilities and components to the database with the help of the Component Designer. To create an OS configuration, all you need to do is select the components you need by walking down the hierarchy nodes on the left pane. The right pane provides three views for files, registry entries, or components. As components and capabilities are added to the configuration, the right pane will update the configuration.

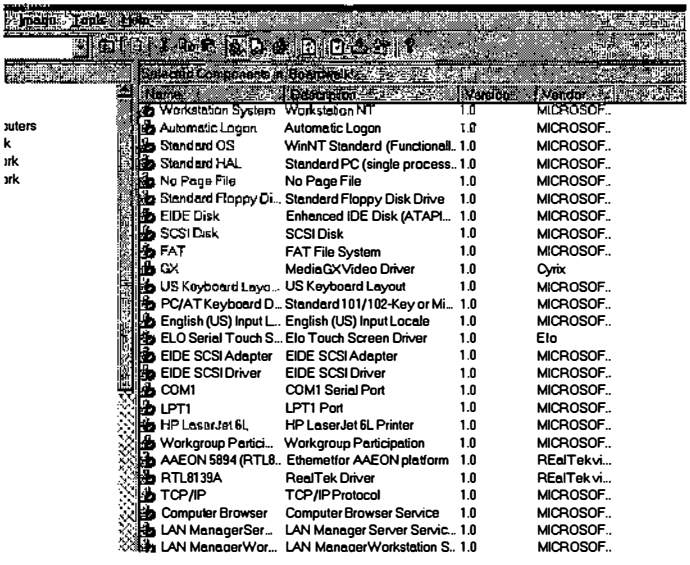


Figure 2.2: Component View

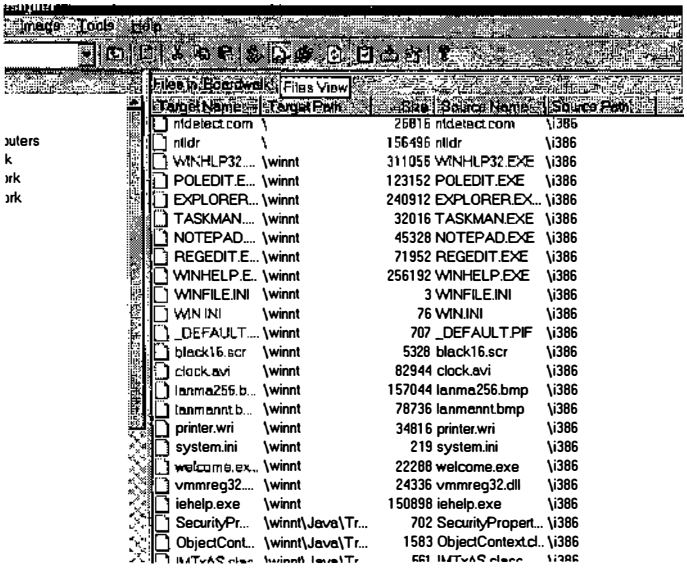


Figure 2.3: File View

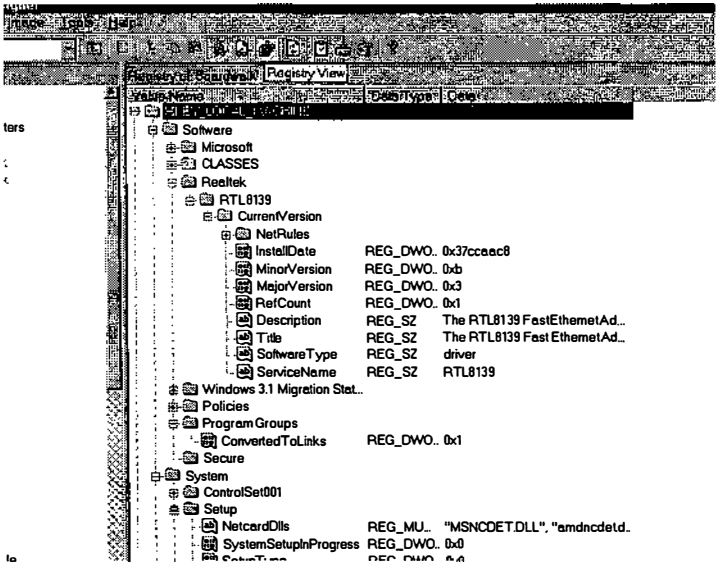


Figure 2.4: Registry View

When you build the OS image, the Target Designer will traverse through the TDSD and pull the components and files from a pre-selected file repository (typically a CD-ROM). The TDSD is stored in MDB format, which can be viewed with Microsoft Access.

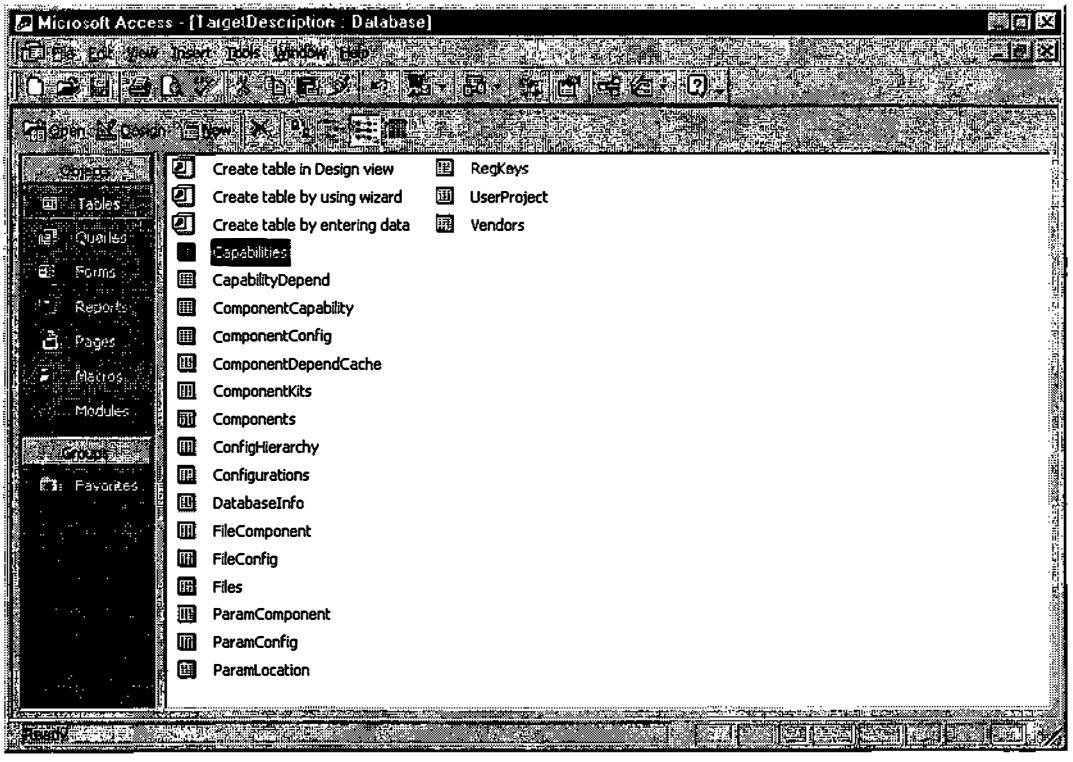


Figure 2.5: TDSD viewed from MS Access

2.2 Target Designer Step-by-Step

Building the OS image is easy using Target Designer. The interface is simple and easy to use. The following are the steps to build a Windows NT Embedded Image:

1. Select a target platform that will support your configuration. There are many manufacturers of embedded PC hardware. Which platform is best for your target system will depend on your total system requirements.
2. Use the Hardware Query tool and the SCSI tool to diagnose your target system's hardware. The information gathered by these tools will help you select the components for your configuration.

3. Use Component Designer to create component definition files for your applications and drivers. You will need to create components for applications and drivers not listed or supported in the standard Windows NT configuration. NTE comes with the basic driver components that originally ship with NT 4.0. For the latest drivers you will need to either create a component or install the driver after you have built and downloaded an NTE image.
4. From Target designer, import the component definition files into your database hierarchy.
5. Create a new configuration for your target.
6. Walk through the configuration tree and select the components for your system. You can select your system's role by selecting Primary Domain Controller, Server System, or Workstation System. Some components are mutually inclusive or exclusive from other components. Dialog boxes will appear when a selected component is required or is excluded from other non-selected or selected components. Some components allow you to set up their properties.

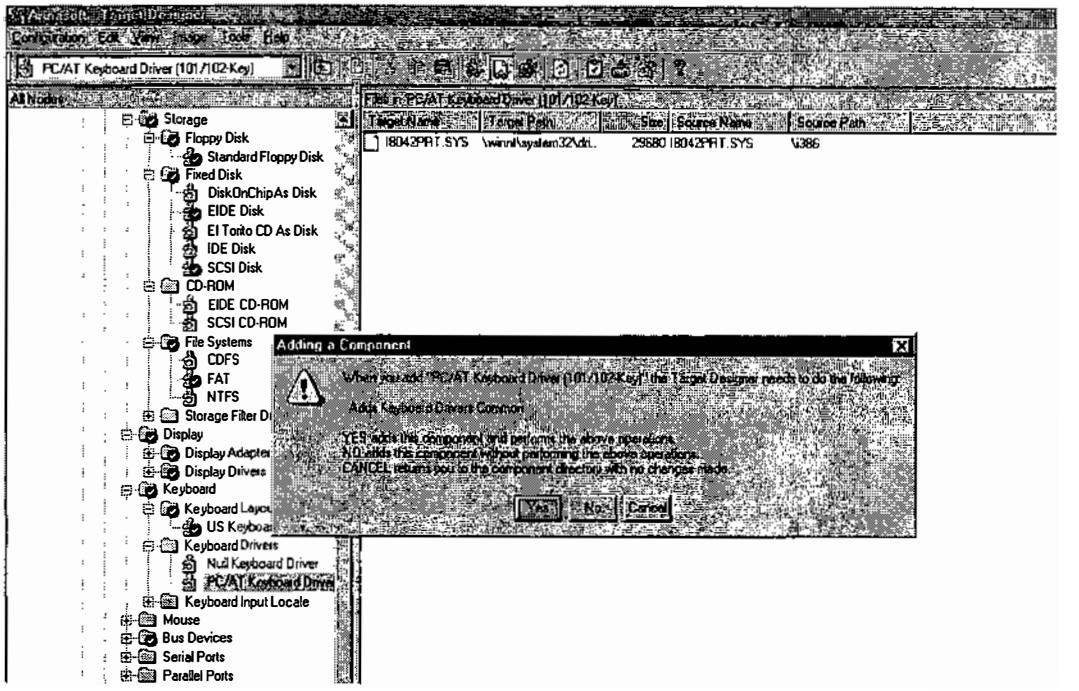


Figure 2.6: Warning on Component Conflicts

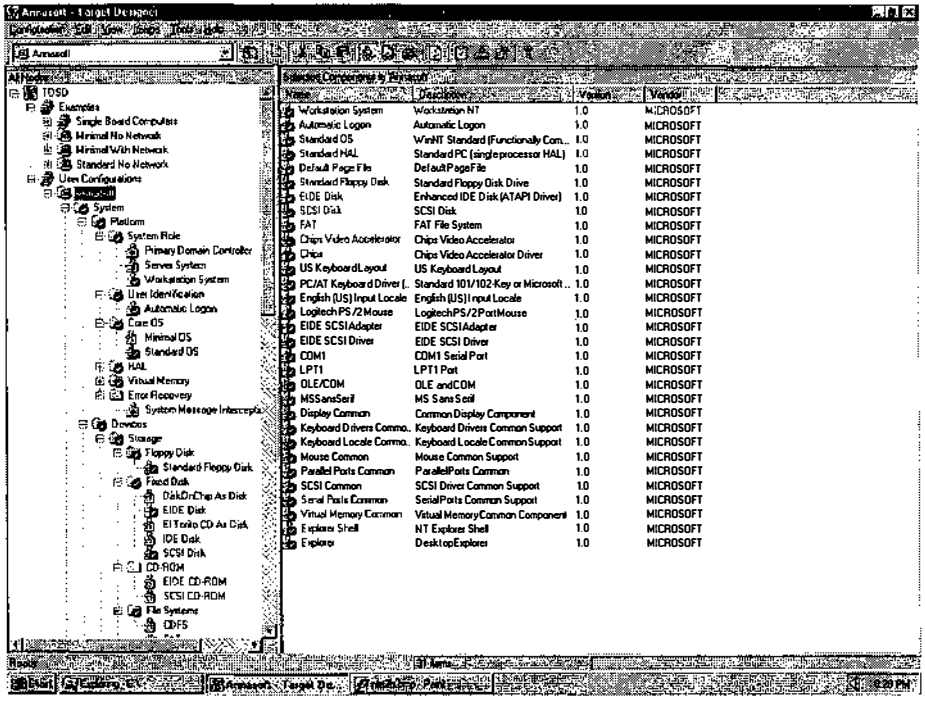


Figure 2.7: Target Designer Complete List of Components

7. After you have selected all of the components, you can run a check on your build to find any other dependencies that are missing. When checking is completed, either a message box appears telling you all of the dependencies and conflicts are okay or a message box appears that lists missing components, conflicts, or dependencies.

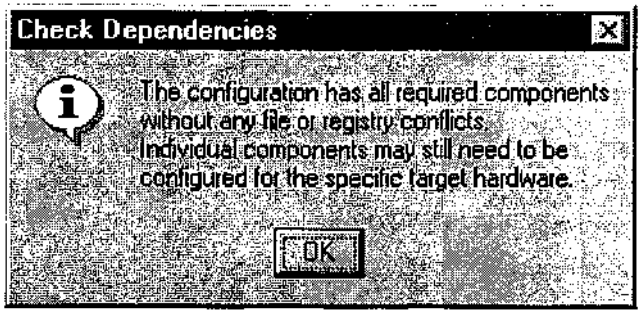


Figure 2.8: Dependency Checking Complete Dialog

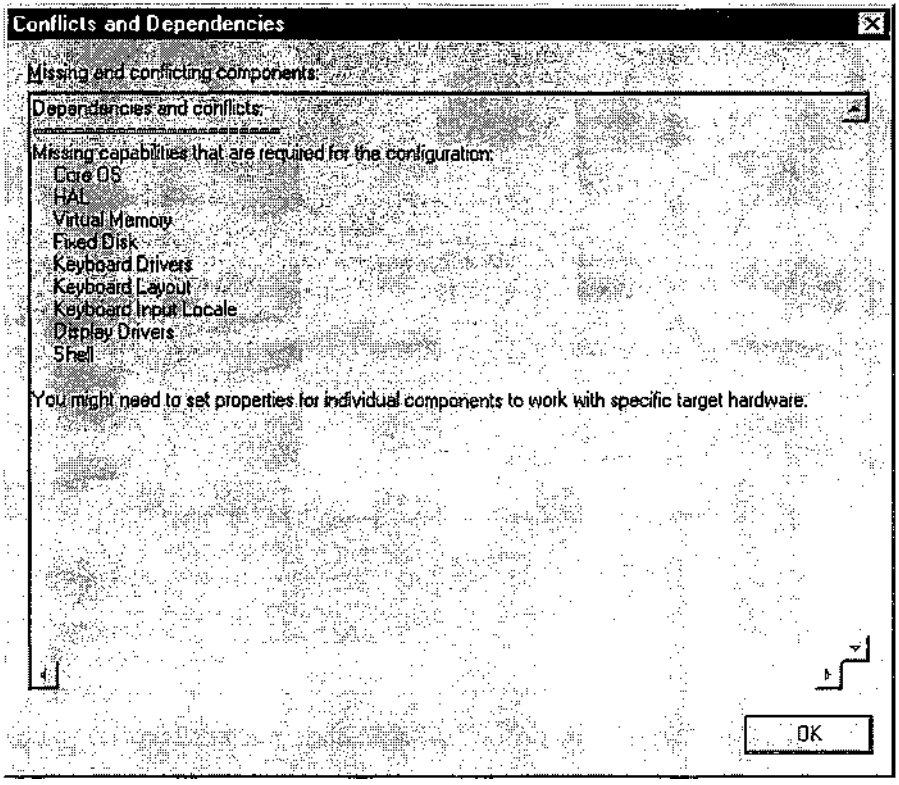


Figure 2.9: Dependency Checking Fail Dialog

8. Once the dependencies have been checked, you are ready to build and install the OS. Under the Image menu, check that the settings of your repository (the repository is the first NTE CD-ROM) and target subdirectory are set. You may select to target a second hard drive that can be removed and installed into the target system (in this case you must make sure that the NT master boot record is formatted on the second hard drive), or you can choose a subdirectory where later you can copy the files to the final boot media.

9. Build the OS. Target Designer will traverse the database and pull in the components that you have selected from the Windows NT Embedded CD-ROM and any other local directories you have previously defined.

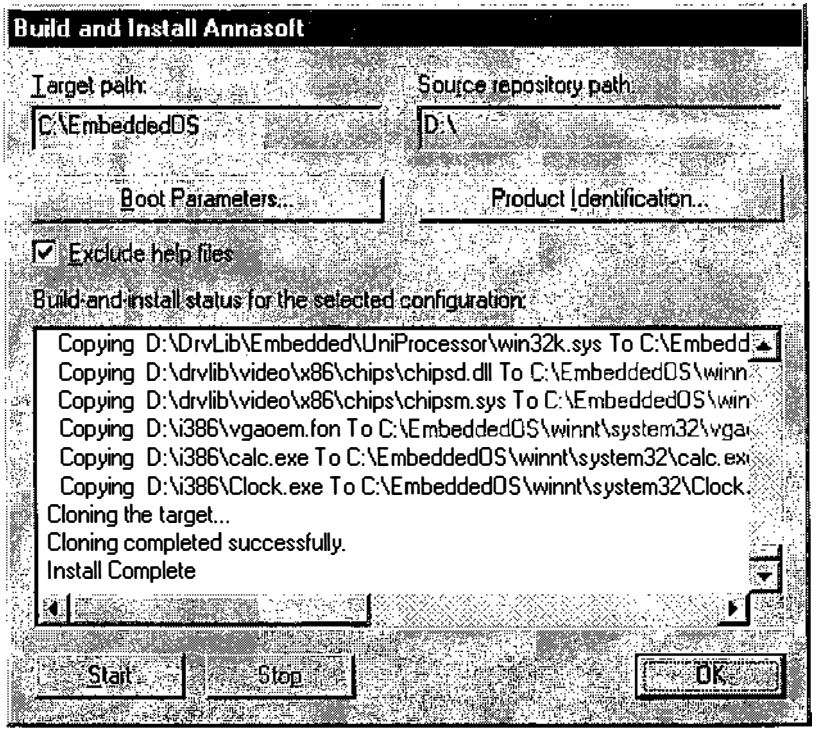


Figure 2.10: Building the OS

10. Once the build is complete, if you targeted a second hard drive, you can easily move the hard drive to your target device. If you installed to a subdirectory, you will need to copy the files to your target device. The target system build may

contain long file names that can be truncated when copied by MS-DOS. Make sure that your target device has the Master NT Boot record.

2.3 TDSD Breakdown - Components and Capabilities

As you traverse the hierarchy tree, you can see a variety of capabilities and components to choose for a configuration. At a minimum, a Windows NT Embedded configuration needs the following capabilities: System Role, Core OS, HAL, Virtual Memory, Storage Device, Display, keyboard, mouse, and shell. The smallest Windows NT Embedded size is about 9.55 MB.

Some of the capabilities are self explanatory, and some are not. Most of the components are recognizable if you're familiar with Windows NT. Below is a high level summary for some of the capabilities. A detailed list is provided in Chapter 10.

2.3.1 System Role

The system role defines what kind of system you're building. The options are Primary Domain Controller, Server System, and Workstation.

2.3.2 Core OS

There are two options for the core OS: Standard (74MB) and Minimal (7.81MB). By switching back and forth between the file views for both components, you can see that the minimal has just the components needed to boot and run NT. MFC, control panel applets, utilities, event log, registry editor, and fonts have been removed to create the Minimal component. Most of the capabilities that are not part of the Minimal OS are part of individual components. Each of these components can be added to build up the features of a minimal OS configuration.

2.3.3 HAL

The HAL is the hardware abstraction layer that makes Windows NT portable to other platforms. Windows NT supports 13 types of systems. The HAL capability contains both the kernel and the HAL for each system supported. The Standard HAL component is used for most single processor systems. MPS is used for multiprocessor system. Windows NT was designed to handle up to 32 processors, but only the 2 processor Workstation version and 4 processor Server version are shipped to consumers. The other HALs are for specific computers like Compaq's System Pro or IBM's Micro Channel Architecture. Computer board manufacturers will ship a HAL if their platform is not one of the standard set. A

component would have to be created for any new HAL provided by the computer manufacturer.

2.3.4 Devices

The Devices capabilities contains all of the basic I/O hardware capability found on most boards: hard drive, floppy, display, COM ports, SCSI Bus, etc.

2.3.5 Network

The Network capability breaks the network stack and services into a variety of components. A configuration contains an adapter, a driver, protocol, basic services, and basic utilities. The Windows NT resource kits explain the various services and utilities in detail.

2.3.6 Windows Services

Windows Services contains components that are needed by other components in order to run under Windows NT: OLE, DCOM, TAPI, and ODBC.

2.3.7 Management

The Management capability provides all of the necessary components to administer a system. Remote administration, user management, performance, storage management, diagnostics, and event monitoring can individually be added to a system.

Most of these components are not part of the Minimal OS component. If an NTE minimal image is having problems, these components can be individually added so the tools to diagnose the problem are available.

2.3.8 Applications

Applications capability contains the components for the shell, utilities, and accessory applications.

2.4 Exercise 1a – Building an Image

The goal of this exercise is to introduce Target Designer fundamentals. Exercise1 is broken into four parts. Exercise 1a will concentrate on building a Standard Workstation image with disk access support, video support, mouse, keyboard, and COM ports. Exercise 1b will

introduce a third party for remote downloads to the target platform. Exercise 1c will cover exporting a configuration. The configuration will be used in later exercises to create KDFs and install drivers. Exercise 1d will show you how to boot from an NTFS partition.

2.4.1 Creating the Configuration

1. Open Target Designer.
2. Click on the TDSD root so that it has focus.
3. Right click on TDSD and select ADD New Collection.
4. Right click on the new collection and RENAME the collection to **NTE Class.**
5. Right click on NTE Class and select ADD New Configuration.
6. Right click on new configuration and RENAME the configuration to **Exercise1.**
7. Once you have created the new collection and configuration, you now add components to your configuration. Walk the Configuration tree, right click and select ADD to add components. As you add components the dynamic dependency check will inform you if the component you select requires you to add additional components. Selecting "Yes" will include the dependent components.

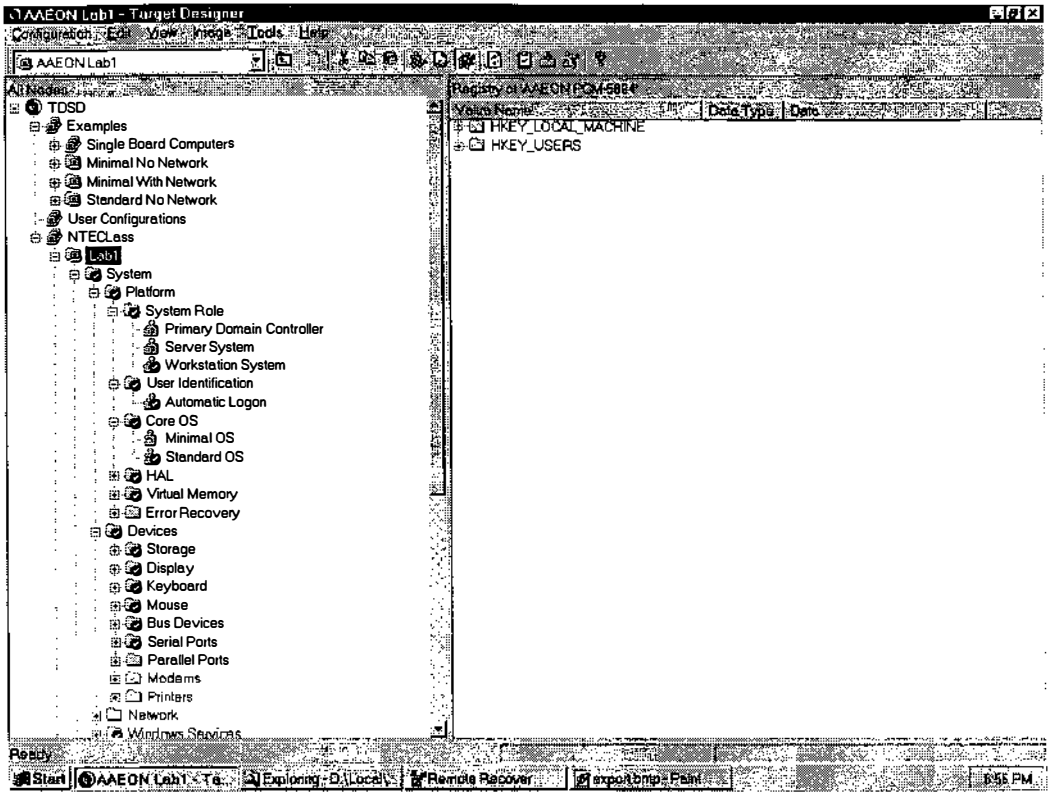


Figure 2.11: Target Designer Exercise1 Configuration

Following is the component list for this configuration. Select the bolded components and the rest of the dependent components will be added.

Capability	Component	Size	Check Box
System / Platform / SystemRole	**Workstation system**	0	
System / Platform / User Identification	**Automatic Login**	0	
System / Platform / Core OS	**Standard OS**	74.32MB	
System / Platform / HAL	**Standard HAL**	958KB	

System / Platform / Virtual Memory	**No Page File**	0MB	
System / Devices / Storage / Floppy Disk	**Floppy**	18KB	
System / Devices / Storage / Fixed Disk	**EIDE**	0	
System / Devices / Storage / Fixed Disk	SCSI Disk	15KB	
System / Devices / Storage / CD-ROM	EIDE CD-ROM	0	
System / Devices / Storage / CD-ROM	SCSI CD-ROM	22KB	
System / Devices / Storage / File Systems	**FAT**	139KB	
System / Devices / Display / Display Drivers	**VGA**	99KB	
System / Devices / Keyboard / Keyboard Layout	**US Keyboard Layout**	9KB	
System / Devices / Keyboard / Keyboard Drivers	**PC/AT Keyboard**	29KB	
System / Devices / Keyboard / Keyboard Input Local	**English (US) Input Locale**	7KB	
System / Devices / Mouse	**Logitech**	29KB	
System / Devices / BUS Devices SCSI / SCSI Adapters	EIDE SCSI Adapter	0	
System / Devices / BUS Devices SCSI / SCSI Drivers	EIDE SCSI Driver	27KB	
System / Devices / Serial Ports	**COM1**	0	
System / Windows Services / OLE and COM	OLE/COM	2.18MB	
System / Desktop Settings / Fonts	MS Sans Serif	151KB	

System / Desktop Settings / Application Links	**Explorer Links**	36KB	
System / Shared System Components	Display	25KB	
System / Shared System Components	Keyboard Drivers	9KB	
System / Shared System Components	Keyboard Locale	135KB	
System / Shared System Components	Mouse Common	9KB	
System / Shared System Components	SCSI	58KB	
System / Shared System Components	Serial Ports	44KB	
System / Shared System Components	Virtual Memory	0	
Applications / Shell	**Explorer Shell**	0	
Applications / Utilities	Explorer	322KB	
Applications / Accessories	**Calculator**	124KB	
Applications / Accessories	**Notepad**	225KB	

8. When completed, right-click on the Workstation system component and change the Computer Name to **NTE1**.

9. Perform a dependency check on the configuration. Either click the Check Dependencies button or select IMAGE/Check Dependencies. Dependency checking will find any other dependencies that are missing. A message box appears when dependency checking is completed. If you get a dialog indicating a conflict or a dependency missing make the corrections and re-check the configuration.

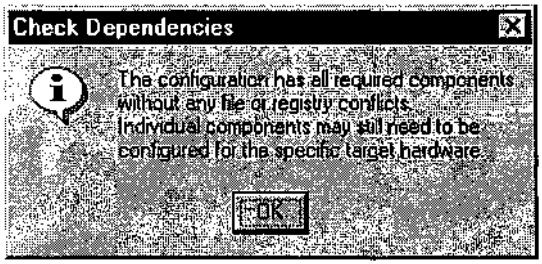


Figure 2.12: Check Dependencies Completed Dialog Box

10. To see the total size of the image left-click on Exercise1, and then select File view. The total image size will be listed at the boot along with the total number of files.
11. Check the IMAGE/Settings... make sure that the repository path is set to the root of the first NTE CD. Also, make sure the target path is pointing to a subdirectory like c:\EmbeddedOS. Make sure the NTE CD is in the CD-ROM drive.
12. Click Image Build and Install and click **START** in the Build and Install dialog box.
13. When prompted about the undefined PID, click **YES.** You will be using the TEST PID in these exercises. The test PID is good for 60 days. The image will have to be rebuilt after 60 days if you want to continue testing. Target Designer will traverse the database and pull in the components that you have selected from the Windows NT Embedded CD-ROM and any other local directories you have previously defined.

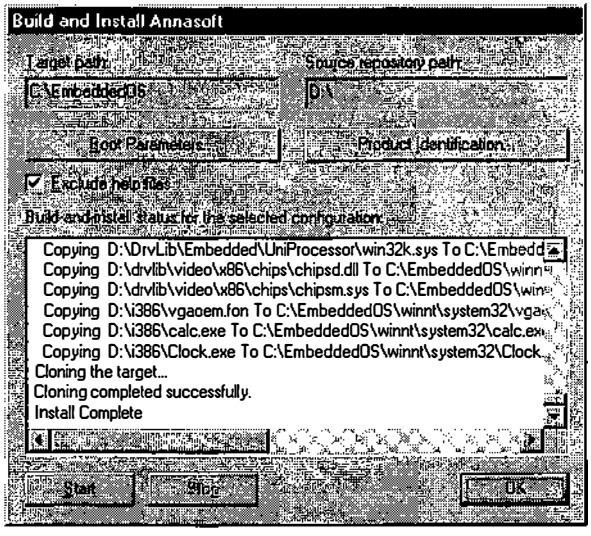


Figure 2.13: Building the Image

Once the image is built, you can look in the installation directory to see the NT image. BOOT.ini, NTDETECT.COM, and NTLDR are added to the image automatically.

2.5 Build Image Dialog Options

The Build and Install dialog is the final step to creating the NTE image. When you build and install the image, files are copied from the main repository and the local repository to a target

subdirectory. You can change these locations in the Image->Settings dialog. Typically the main repository is the first NTE CD. Local repositories are KDFs that were added to the TDSD. The image is built to a subdirectory to prevent the accidental replacement of your development machine's NT files. Build and Install are also broken down into two separate operations. Build will create the registry hives for the image. Install will copy all files of the configuration to the target location.

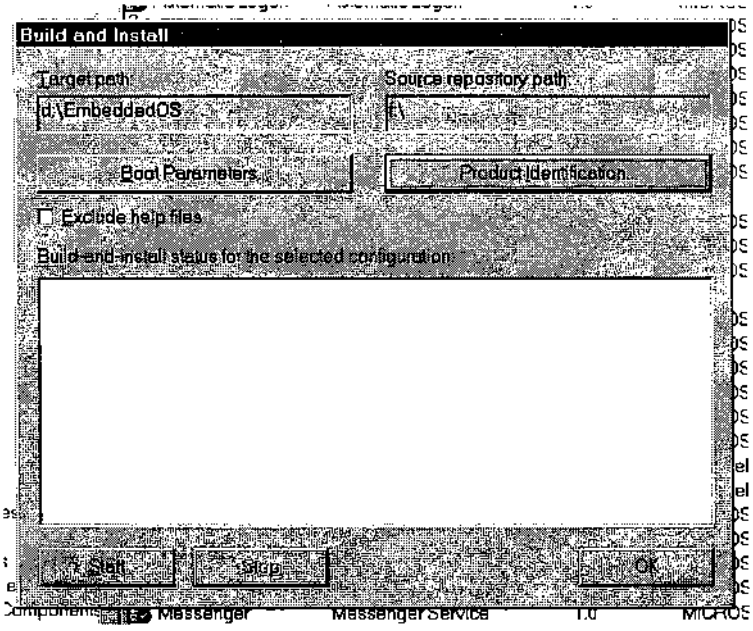


Figure 2.14: Build and Install Window

Click on Boot Parameters..., to modify the boot.ini settings to allow for debug checking (see Windows NT DDK for more information about debugging).

The options available are:

- Limit physical memory detection to - Allows you to check for bad memory chips within a system.
- Use standard video driver - OS will use the standard NT video driver.
- Show device driver names - Allows you to see what drivers load at start-up. Helpful if a driver added to TDSD via KDF fails to function.

- Disable serial mouse detection - When selected, NT will not detect a mouse on any serial COM ports.

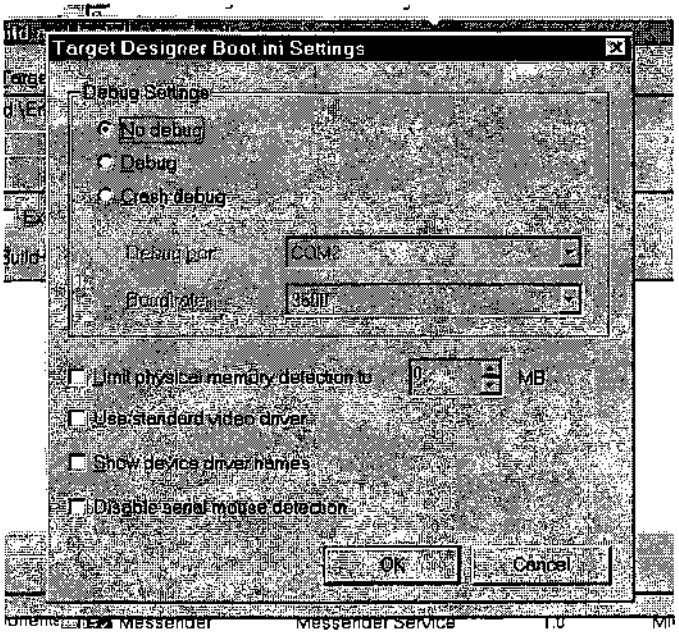


Figure 2.15: Boot.ini Settings

The Product Identification button allows you to enter the COA number according to your licensing agreement with Microsoft. Or, if you're in development, you can click on the Use Test Build to use the Test PID to create an image. Once an image is built with Target Designer using the Test PID, the image will only last 60 days. You can rebuild the image and continue development for another 60 days, until the system is developed. The 60-day limitation is to prevent unlicensed (illegal) copies of NT going into production.

After you develop your image, you can build the final image with your Product key ID number. The product key is limited to your licensing agreement. If you try to build with components outside of your license the build will fail. The product key is different than the licensing numbers (COA). The product key is obtained from your licensor and is unique to your signed license agreement.

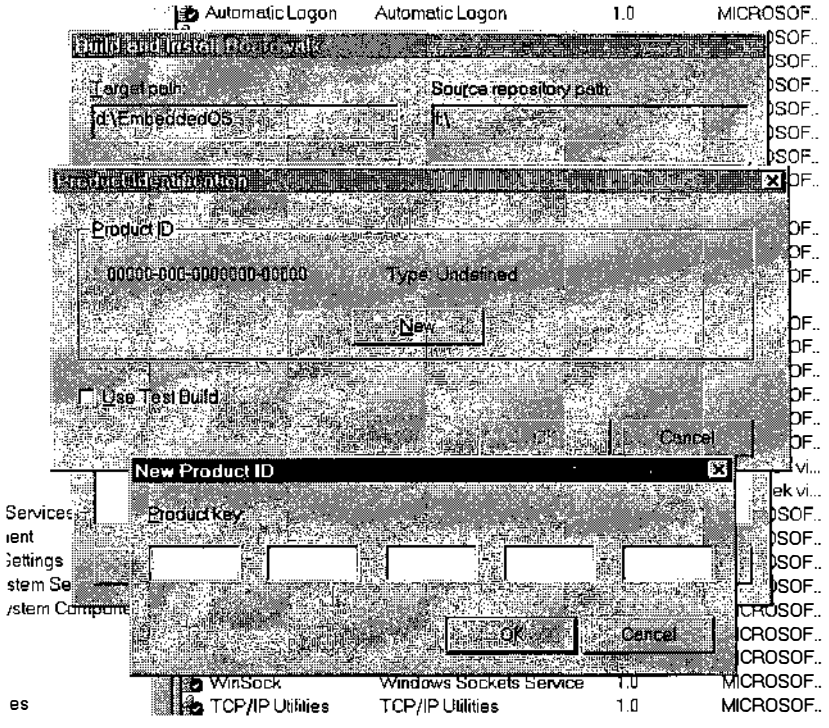


Figure 2.16: Product Identification

Windows NT Embedded licensing is broken down into four classes:

Class 1	Class 2	Class 3	Class 4
Headless Workstation	Full Workstation	Headless Sever	Full Server

The build will check to see if the PID entered matches the specification. If there is a mismatch the build will fail.

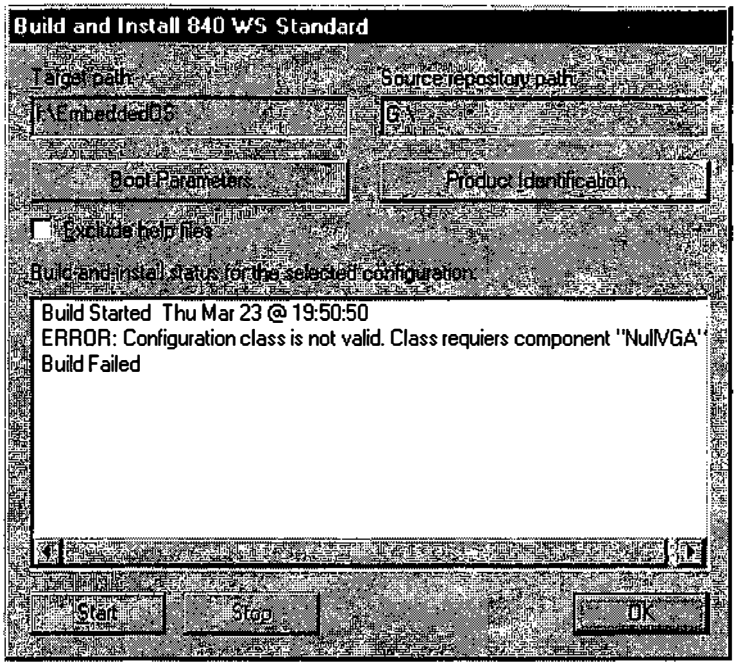


Figure 2.16b: PID and Configuration Mismatch.

2.6 Prepare Target and Remote Download

Windows NT Embedded doesn't come with a solution for transferring the image from the target subdirectory to the target platform. The transfer is left completely up to you and your development process. The following sections provide you with some useful alternatives.

2.6.1 Option 1. Second Hard Drive

You can have a second HDD drive on your development system. Format the HDD with the correct partition and file system using disk administrator. Copy the NT image from the target subdirectory to the second HDD. Shut down the development system, remove the second HDD, and re-install HDD on the target system. Now you can boot the target.

2.6.2 Option 2. Remote Download

You can transfer the NT image files over a network if you can connect your development and target to a network. Network transfer introduces two problems: boot sector and long file name support.

2.6.3 Preparing the Target

There are two problems to keep in mind when transferring an NTE image. The first is the NT boot sector. The target's hard drive must have the NT boot sector in order to boot the OS. In these exercises, we will be using the FAT file system in the primary boot partition. Format the hard drives using MS-DOS. Microsoft provides a NTBOOT patch under the MUNGEBOOT directory that can be used to patch the first sector on the disk. This disk image doesn't come with FORMAT or FDISK utilities. You will need these executables in order to prepare the hard drive for each exercise. Once the patch is applied to the hard drive, the HDD is ready to boot the NT image.

The initial boot sector - BIOS Parameter Block (BPB)

The first sector on a disk is known as the boot record, and contains the start-up code to boot an OS. You can use DEBUG to examine the boot record of any DOS disk volume.

```
C:>debug
-l 100 2 0 1
-d 100 2ff

1633:0100  EB 3C 90 4D 53 44 4F 53-35 2E 30 00 02 40 01 00   .<.MSDOS5.0..@..
1633:0110  02 00 02 00 00 F8 FB 00-3F 00 F0 00 3F 00 00 00   ........?...?...
1633:0120  B1 85 3E 00 80 00 29 3C-DD 37 92 4E 4F 20 4E 41   ..>...)<.7.NO NA
1633:0130  4D 45 20 20 20 20 46 41-54 31 36 20 20 20 33 C0   ME    FAT16   3.
1633:0140  8E D0 BC 00 7C 68 C0 07-1F A0 10 00 F7 26 16 00   ....|h.......&..
1633:0150  03 06 0E 00 50 91 B8 20-00 F7 26 11 00 8B 1E 0B   ....P.. ..&.....
1633:0160  00 03 C3 48 F7 F3 03 C8-89 0E 08 02 68 00 10 07   ...H........h...
1633:0170  33 DB 8F 06 13 02 89 1E-15 02 0E E8 90 00 72 57   3.............rW
1633:0180  33 DB 8B 0E 11 00 8B FB-51 B9 0B 00 BE DC 01 F3   3.......Q.......
1633:0190  A6 59 74 05 83 C3 20 E2-ED E3 37 26 8B 57 1A 52   .Yt... ...7&.W.R
1633:01A0  B8 01 00 68 00 20 07 33-DB 0E E8 48 00 72 28 5B   ...h. .3...H.r([
1633:01B0  8D 36 0B 00 8D 3E 0B 02-1E 8F 45 02 C7 05 F5 00   .6...>....E.....
1633:01C0  1E 8F 45 06 C7 45 04 0E-01 8A 16 24 00 EA 03 00   ..E..E.....$....
1633:01D0  00 20 BE 86 01 EB 03 BE-A2 01 E8 09 00 BE C1 01   . ..............
1633:01E0  E8 03 00 FB EB FE AC 0A-C0 74 09 B4 0E BB 07 00   .........t......
1633:01F0  CD 10 EB F2 C3 50 4A 4A-A0 0D 00 32 E4 F7 E2 03   .....PJJ...2....
1633:0200  06 08 02 83 D2 00 A3 13-02 89 16 15 02 58 A2 07   .............X..
1633:0210  02 A1 13 02 8B 16 15 02-03 06 1C 00 13 16 1E 00   ................
1633:0220  F7 36 18 00 FE C2 88 16-06 02 33 D2 F7 36 1A 00   .6........3..6..
1633:0230  88 16 25 00 A3 04 02 A1-18 00 2A 06 06 02 40 3A   ..%.......*...@:
1633:0240  06 07 02 76 05 A0 07 02-32 E4 50 B4 02 8B 0E 04   ...v....2.P.....
```

```
1633:0250   02 C0 E5 06 0A 2E 06 02-86 E9 8B 16 24 00 CD 13   ............$...
1633:0260   0F 83 05 00 83 C4 02 F9-CB 58 28 06 07 02 76 11   .........X(...v.
1633:0270   01 06 13 02 83 16 15 02-00 F7 26 0B 00 03 D8 EB   ..........&.....
1633:0280   90 A2 07 02 F8 CB 42 4F-4F 54 3A 20 43 6F 75 6C   ......BOOT: Coul
1633:0290   64 6E 27 74 20 66 69 6E-64 20 4E 54 4C 44 52 0D   dn't find NTLDR.
1633:02A0   0A 00 42 4F 4F 54 3A 20-49 2F 4F 20 65 72 72 6F   ..BOOT: I/O erro
1633:02B0   72 20 72 65 61 64 69 6E-67 20 64 69 73 6B 0D 0A   r reading disk..
1633:02C0   00 50 6C 65 61 73 65 20-69 6E 73 65 72 74 20 61   .Please insert a
1633:02D0   6E 6F 74 68 65 72 20 64-69 73 6B 00 4E 54 4C 44   nother disk.NTLD
1633:02E0   52 20 20 20 20 20 20 20-00-00 00 00 00 00 00 00   R       .........
1633:02F0   00 00 00 00 00 00 00 00-00-00 00 00 00 00 00 55 AA   ..............U.
-q
```

Listing 2.1: Using DEBUG to View DOS Boot Records

The BPB is broken into three parts. The first is the initial jump instruction. The second section contains the information about the disk such as bytes per sector, sector per cluster, number of FAT entries, etc. The last section is the actual loader code. The MUNGEBOOT disk patches a hard drive boot sector using DEBUG and a boot sector image. The patch replaces the first and third section with the appropriate NT boot sector, and keeps the FAT/disk information.

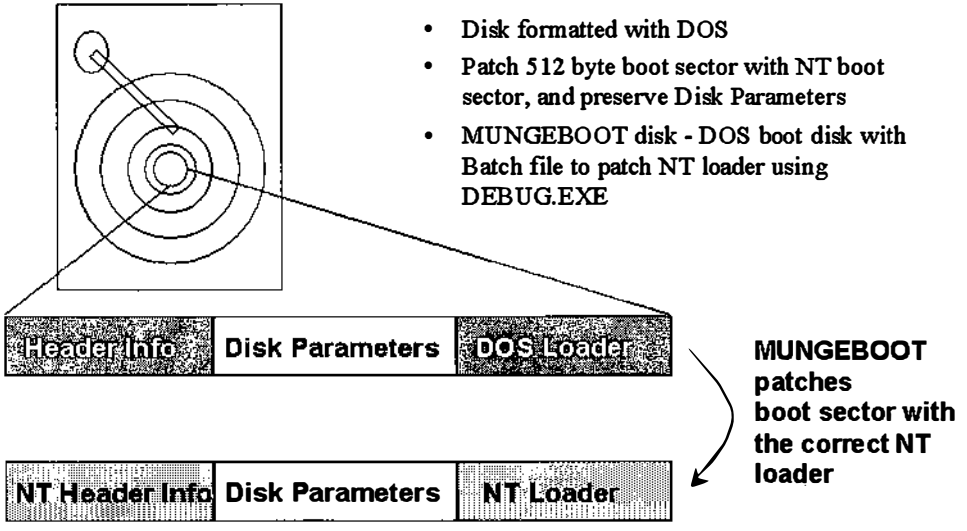


Figure 2.17: Mungeboot Processes

Long File Names and Winternals' Remote Recover

The second problem is Long File Names (LFN). Files and directories will be truncated when transferred over a straight DOS network connection. Winternals created the Remote Recover program to help IS departments recover files from a corrupted HDD. As a side benefit, Remote Recover allows you to transfer files over a network to a target system without losing long file name support. The NT HOST system sees the target system's HDD as another drive in the system.

The exercises in this book will use Remote Recover as the download transport to the target. You may choose to use a different download method. If you do choose to use a different method, you can skip this next exercise.

2.7 Exercise 1b - Remote Download

In this exercise, we will use NTBOOT patch and Remote Recover to transfer the image built in Exercise 1a to the target system. You will need two floppy disks.

2.7.1 Create the Mungeboot Disk

1. On your target's hard drive, create a clean 2GB formatted FAT partition. You can do this by attaching the target drive to your host machine and using Disk Administrator, or you can use an MS-DOS boot floppy with Format and FDISK. Later you can use Remote Recover to connect the target hard drive remotely to the host machine and re-format the drive to clean the partition between builds.
2. Insert a blank disk, and open a COMMAND Window (DOS Box).
3. Microsoft provides a pre-built DOS boot disk image on the NTE CD ROM. This image can be found in the D:\MUNGEBOOT directory along with an application to extract the image. The MUNGEBOOT image has the files needed to perform the NT boot sector patch that the HDD Prep DOS boot disk performs. We are going to extract the image only to get the DOS boot files. In the command Window go to the D drive, and go to the D:\MUNGEBOOT directory.
4. Type the following at the DOS prompt: d:\mungeboot>**webimgnt munge622.144.**
5. Follow the directions to extract the image to the A drive.
6. When completed close the DOS box.
7. Boot your target with the MUNGEBOOT disk that you just created and run the batch file to patch the hard drive with an NT boot sector.

2.7.2 Creating the Remote Recover Boot Disk

If you have Remote Recover installed on your system:

8. The Remote Recover requires an MS-DOS boot disk. Use the second floppy and repeat steps 3 through 7 to create another MUNGEBOOT disk.
9. Next, run the Remote Recover Client Setup Wizard from the Start menu. Follow the instructions to create the Remote Recover boot disk. The MSClient files are on the NTE CD under "clients". You will need the MSClient driver for your network card. (In the case of the AAEON platform, the MSClient driver is supplied by AAEON.)
10. Enter a TCP/IP address that will allow you to connect to your LAN. (In this exercise and the following exercises, the target's TCP/IP address will be 10.1.1.250. subnet 255.0.0.0. Your address may be different.)
11. When complete, remove the Remote Recover disk from the host and insert the disk into the floppy drive in the target. Make sure that you set up the TCP/IP address so that you can connect to your network.
12. Re-boot the target with the Remote Recover boot disk. Wait for the Remote Recover client to launch. Start Remote Recover host on the development system. Once the target is booted and waiting for host connection, you will see your target machine with the TCP/IP address in the Remote Recover Host's dialog box. Select your target machine (make sure you do not select anyone else's target system). A beep will sound on the target. Select Disk 0\partition 1 and click on the **MOUNT** button. Select a drive letter when prompted.

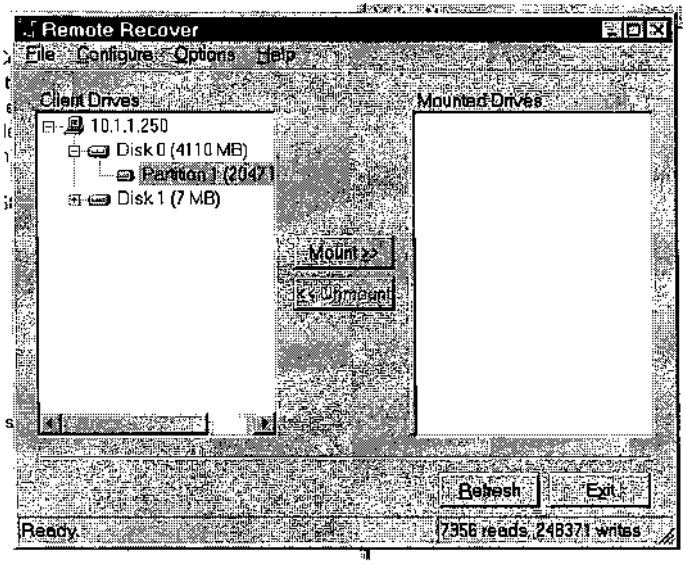


Figure 2.18: Remote Recover

13. Open Explorer. You will see the new drive letter.
14. Go to c:\EmbeddedOS and hit CTRL-A to select all of the files in the directory.
15. Copy the files under c:\EmbeddedOS, and drag them to the remote drive. Remote Recover Client will indicate that writes are being performed on the disk. Do not copy the EmbeddedOS folder. Only copy the files in c:\EmbeddedOS.
16. When the transfer is complete and the **Remote Recover client is waiting,** click the **UNMOUNT** button in the Remote Recover host.
17. Remove the Remote Recover floppy disk, and hit the ESC key on the target system to reboot the target.
18. The target system will reboot into the NT image.

TIP: You may use Remote Recover to re-format and set up the target disk from your host NT system without having to run the Mungeboot patch files.

2.8 Exercise 1c – Exporting and Importing Configurations

Managing large NT images or multiple images can be cumbersome. Many embedded systems can last as long as 10 years. Departments can change; people come and go over that time. To

ease the saving of images, Microsoft allows you to export Configurations information. The CFG file can be exported and imported from Target Designer. In this exercise we will export the configuration from Exercise 1a.

1. Left click on Exercise1.
2. In the Configuration menu, select Export...
3. You can rename and change storage locations of the CFG file. Click Export.

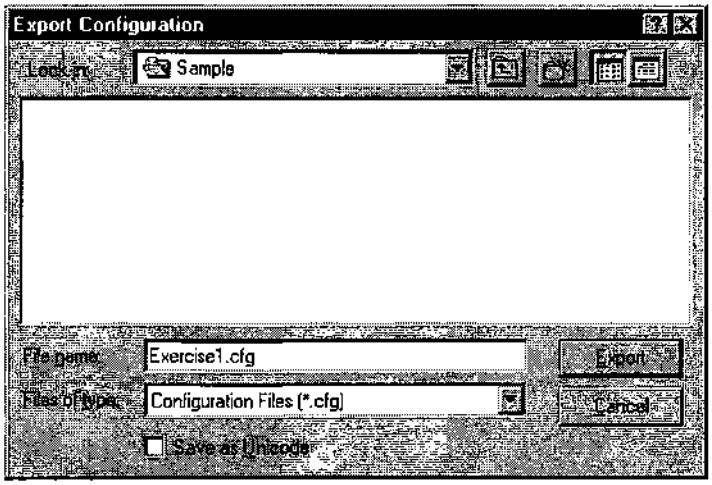


Figure 2.19: Export Configuration

4. The CFG file size will be 4KB. You can use Wordpad to view/edit the contents of a CFG file.

The CFG file can be viewed with a text editor.

```
[Version]

Signature = "$Windows NT$"
CFGVersion = 1.0

[Header]

Name = "Exercise1"
ModificationDate = 09/08/1999
OS = "WINNT"
```

```
OSVersion = "4.0"
Platform = "i386"
HierarchyLocation = "%%TDSD%%\NTECLass"
ExcludeHelpFiles = N
TestPID = Y
ProductID = "00000-000-0000000-00000"
DigitalPID = \
ff,ff,ff,ff,66,00,69,00,63,00,61,00,74,00,69,00,6f,00,6e,00,2e,00,2e,00,\
2e,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,01,00,00,00,00,00,00,00,\
00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,00,\
00,00,00,00,00,00,00,00,00,00,00,00,80,9a,89,5f,00,00,00,00,00,00,00,00,\
00,00,00,00,20,f0,43,00,60,00,10,01,f0,d8,14,00,5c,00,01,01,00,00,00,00,\
88,eb,12,00,d1,19,80,5f,19,00,00,00,00,00,00,00,c4,eb,12,00,00,00,00,00,\
19,00,00,00,08,00,01,00,3c,26,ed,77,5c,00,01,01,40,06,ed,05,5c,00,01,01,\
00,00,00,00,0a,62,00,00,49,01,0a,62,8e,10,01,00,31,00,00,00,48,a7,eb,77,\
c0,c0,c0,00,01,01,00,00,40,ec,12,00,d8,03,e8,77,5c,00,01,01,31,00,00,00,\
48,a7,eb,77,c0,c0,c0,00,40,ec,12,00,e2,03,e8,77,18,10,80,5f,18,10,80,5f,\
01,00,00,00,64,04,00,00,40,f0,12,00,0e,11,80,5f

Prebuilder =
Preinstaller =
Postinstaller =
CodePage = 1252


[BootParameters]

Debug = Off
DebugPort = "COM2"
BaudRate = 9600
MaxMemoryDetection = N
MaxMemory = 0
BaseVideo = N
DisplayDevice = N
SerialMouseDetection = N


[ComponentKits]

1 = "WinNT_CK_SystemRole", 1.00
2 = "WinNT_CK_AutoLogon", 1.00
3 = "WinNT_CK_HAL", 1.00
4 = "WinNT_CK_Pagefile", 1.00
5 = "WinNT_CK_Floppy", 1.00
6 = "WinNT_CK_Disk", 1.00
7 = "WinNT_CK_Keyboard", 1.00
8 = "WinNT_CK_Mouse", 1.00
9 = "WinNT_CK_Ports", 1.00
10 = "WinNT_CK_UTILS", 1.00
11 = "WinNT_CK_DCOM", 1.00
12 = "WinNT_CK_MSSansSerif", 1.00
13 = "WinNT_CK_Shell", 1.00
14 = "WinNT_CK_AppLinks", 1.00
15 = "WinNT_CK_Calc", 1.00
16 = "WinNT_CK_Notepad", 1.00
17 = "WinNT_CK_Display", 1.00
18 = "WinNT_CK_SCSI_Adapter", 1.00
```

```
19 = "WinNT_CK_StandardNTOS", 1.00

[Components]

"Workstation System", 1, Y
"Automatic Logon", 2, Y
"Standard HAL", 3, Y
"Virtual Memory Common", 4, N
"Standard Floppy Disk", 5, Y
"FAT", 6, Y
"US Keyboard Layout", 7, Y
"Keyboard Drivers Common", 7, N
"PC/AT Keyboard Driver (101/102-Key)", 7, Y
"Keyboard Locale Common", 7, N
"English (US) Input Locale", 7, Y
"Mouse Common", 8, N
"Logitech PS/2 Mouse", 8, Y
"Serial Ports Common", 9, N
"COM1", 9, Y
"Explorer", 10, N
"OLE/COM", 11, N
"MS Sans Serif Font", 12, N
"Explorer Shell", 13, Y
"Explorer Links", 14, Y
"Calculator", 15, Y
"Notepad", 16, Y
"Display Common", 17, N
"EIDE SCSI Adapter", 18, N
"SCSI Disk", 6, N
"EIDE SCSI Driver", 18, N
"SCSI Common", 18, N
"EIDE Disk", 6, Y
"Standard OS", 19, Y
"008 MB Page File", 4, Y
"VGA", 17, Y

[Directories]


[FileOverrides]


[FileDeletions]


[FileAdditions]


[RegistryOverrides]

KeyName = "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Ports"
ValueName = "COM1:"
Type = REG_SZ
Data = "38400,None,8,1,None"

KeyName = "HKEY_LOCAL_MACHINE\System\ControlSet001\Control\ProductOptions"
ValueName = "ProductSuite"
```

```
Type = REG_MULTI_SZ
Data = \
"EmbeddedNT", \
"Embedded Class: Temporary"

KeyName = "HKEY_LOCAL_MACHINE\System\ControlSet001\Control\Session Manager"
ValueName = "BootExecute"
Type = REG_MULTI_SZ
Data = \
"autocheck autochk *"

[RegistryDeletions]

[RegistryAdditions]

KeyName = "HKEY_LOCAL_MACHINE\System\ControlSet001\Control\ProductOptions"
ValueName = "ProductEvalInstallTime"
Type = REG_BINARY
Data = \
c0,31,84,29,70,f2,be,01,3c,e8,ea,1d,6e,1b,6b,e1
```

Once you have exported the configuration to a CFG file, you can save the CFG file to version control. If you created any KDFs for your configuration, you should also have those checked into your version control system. When you want to recover your configuration, you will need to install the KDFs first before importing the CFG file. Collections can also be exported to a binary file (CIX).

2.9 Exercise 1d - Booting from an NTFS Partition

Most of the exercises in this book have the targets booting from a FAT partition. However, NTE can also boot from an NTFS partition.

1. Make a copy of Exercise1 configuration, and rename the configuration to Exercise1d.
2. Using Exercise1d configuration, remove the FAT component, and add the NTFS component.
3. Build the image.
4. Using Remote Recover, mount the target's hard drive to the host system.
5. Format the target's boot partition as an NTFS partition from within Disk Administrator.
6. Copy the files from the host to the target over the Remote Recover link.

7. Un-mount the target drive and reboot the target system. The OS will boot from the NTFS partition.

2.10 Summary

This chapter has covered the basics of Target Designer: creating a configuration, building the OS, preparing the target system, downloading to the target, and booting the target as either a FAT or NTFS system. The rest of the exercises in this book re-use these basic steps.

Chapter 3 EMBEDDED FEATURES

As part of the TDSD, Microsoft has included some embedded components that are not part of the standard Windows NT release. These components allow Windows NT to be deployed in a variety of applied computing systems.

3.1 Headless Support

This enables Windows NT to be used in devices that boot and run without a mouse, keyboard, or display device. Many embedded systems do not expose either a traditional user interface (e.g., Windows-based or DOS-based PC) or, in many instances, any local user interface whatsoever. Windows NT requires a display driver to interface between the graphics sub-system and the video hardware. All currently available video display drivers assume and rely on the existence of underlying video hardware. Microsoft has included a NULL video driver component that eliminates the need for video hardware.

3.1.1 NULL Video Component

The NULL Video Driver acts as a stubbed driver. GUI calls are made to the video driver, but the driver does not access any video hardware. The video information is kept in memory and an acknowledgement is returned that the video command was completed successfully. Exercise 6b shows that you can access a headless system video interface remotely.

3.1.2 NULL Keyboard and NULL Mouse

The NULL Keyboard and NULL Mouse drivers are dependent on each other. When you select the NULL Keyboard, you are asked for the NULL mouse component, and vice-versa. These drivers are similar to the NULL video, but operate in reverse since these are input drivers. Connection status is returned "okay", faking NT into believing that there is a mouse and keyboard present.

TIP: Most PC platforms expect a video card and keyboard. The BIOSes in these platforms will halt if a device like a VGA card and keyboard are missing. Make sure that the BIOS can turn off detection of the video card and/or keyboard if you want to use the board in a headless system.

3.2 Read-Only Boot Support

Many embedded devices use ROM to support solid state operation, which lowers the unit cost and improves reliability. Windows NT Embedded supports a variety of read-only media for boot-strapping itself in a manner that is transparent to the applications and system binaries that access the media (e.g., El Torito support with the write-through filter component allows you to boot from CD-ROMs).

3.2.1 Write Filter Component

The Write Filter driver allows any writes to a partition to be redirected to a RAM disk. The driver allows partitions and whole disks to act as read-only media. The filter can be used for protecting the life of flash devices such as DiskOnChip or ATA flash devices. Any files written to a partition will be saved to memory, and while the system is still running the files can be transferred, opened, and deleted just like any file. When you shut down the system the files will be gone. The component's properties must be modified to list all of the read-only partitions. This will allow you to write-protect a partition via software. You do not need to list any read-only partitions, such as CD-ROMs.

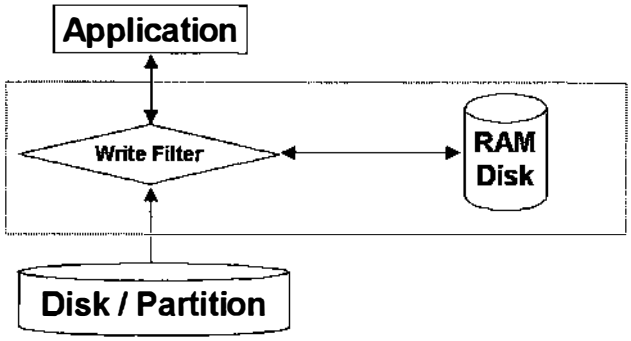


Figure 3.1: Write Filter Architecture

3.2.2 El Torito Component

The El Torito driver allows OEMs to build bootable NT CDs. The El Torito component must be used with the write filter and the No Page file components.

3.2.3 No Page File Component

NT Embedded allows you to select the virtual memory page file size from 0 to 512 MB. The No Page file can be used to reduce the constant read/writes to disks. The No Page file can extend the life of flash disks if read/write access is required. You may need to increase the amount of RAM in your system to improve performance. You should profile your application(s) to determine what memory resources are required.

3.3 Solid-State Media Support

Due to environmental factors such as shock and vibration, many embedded devices use bootable non-volatile storage media with no moving parts to increase mean time between failure (MTBF) ratings and to improve overall system robustness. NT Embedded supports solid state disks and flash memory.

3.3.1 DiskOnChip Driver

The M-Systems' DiskOnChip has become a popular device for many embedded systems. Almost all of the single board computer manufacturers support DiskOnChip. Windows NT

Embedded supplies the DiskOnChip driver to allow read-write access for DiskOnChip 2000 up to 144MB.

3.3.2 ATA (PCMCIA, PC Card) Flash Disks

NT can boot from ATA flash disks and Compact flash cards as long as the system's BIOS makes these devices look like regular hard drives.

3.4 Remote Management

In an embedded system, access to the operating system and/or application is often through a remote system of some type, since a user interface on the device itself may not exist or may not be practical to access. Windows NT Embedded provides both character-mode and graphical management solutions using serial, dial-up, and network connections.

3.4.1 Serial/Modem Remote Management

The first of the character mode accesses, Serial/Modem Remote management allows for the connection over a telephone line or through a direct cable connect. Hyperterm can be used as the interface to the remote system. Files and other basic administration functions can be performed.

3.4.2 Graphical Remote Management

Microsoft supplies NetMeeting 3.0 within NT Embedded. CD 1 contains a full release. You can connect to a remote system over a Network-TCP/IP connection, and take control of the remote desktop. Graphical Remote Management allows you to operate a remote system as if you were in the same room. Transfer of files, installing files, and administration can all be performed.

3.4.3 Telnet

Telnet is a popular method to access remote systems. Windows NT Embedded provides a simple Telnet host. You can connect remotely, and use FTP to transfer files. FTP client is supplied in the TCP/IP utilities component.

3.4.4 Shutdown Utility

The shutdown utility is one of the most popular utilities taken from the NT resource kits. The shutdown utility allows you to remotely shut down and/or reboot a headless system. The help for the shutdown utility shows the following parameters:

Usage: SHUTDOWN [/?] [\\Computer] [/L] [/A] [/R] [/T:xx] ["Msg"] [/Y] [/C]

/?	Shows this screen.
\\Computer	Specifies a remote computer to shut down.
/L	Specifies a local shutdown.
/A	Aborts a system shutdown. This is only possible during the timeout period. If this switch is used, all others are ignored.
/R	Specifies that the machine should reboot after shutdown.
/T:xx	Sets the timer for system shutdown in seconds (20 sec. default).
"Msg"	Specifies an additional message
/Y	Answer all following questions with "yes"
/C	Forces running applications to close.
	ATTENTION: If you use the /C parameter NT ignores the applications option to save data which may have changed. You will see no File-Save dialog box, because NT will force the application to close. This will result in a loss of all data not previously saved !!!

3.5 Error Recovery

Robustness and fault-resilience have always been design criteria for Windows NT. For example, Windows NT possesses characteristics such as protected virtual memory and object-based security. Many embedded applications have more extensive error recovery requirements since they are often used in 24/7 applications in which operator intervention and assistance is not practical. Windows NT Embedded solves the issues associated with "blue screens," and system messages normally displayed on the console that require user interaction. Not only can these errors be logged, but they can be used to trigger application software events.

3.5.1 System Message Interception

Though headless, the NULL video driver is still processing graphical information. If a computer that uses DHCP client to get a TCP/IP address is cut off from the network, a dialog box appears to notify the user. In a headless system, the user is unaware of the box,

and has no way to close it. Redirecting error dialogs to a text log is ideal for headless systems. NT Embedded includes a system message intercept component that can respond to system messages.

3.5.2 Blue Screen

Notice that when you shut down an NT target the system reboots. The auto-rebooting feature allows systems to recover if a fault occurs. If a configuration built by Target Designer fails, the system reboots by default. You can edit the AutoReboot registry value to display a Stop error, commonly called "blue screen."

You will need to edit the auto-reboot registry value in order to prevent the auto-reboot from occurring.

- Select the configuration that you want to change.
- On the toolbar, click the **Registry View** button.
- Locate the registry path: HKLM\System\ControlSet001\Control\CrashControl.
- Right click the **AutoReboot** value and select modify.
- In the **DWORD Editor** dialog box, change the setting from 1 (reboot on failure) to 0 (stop responding or "hang" on failure).

Note: When set to 0 (stop responding on failure), a failing target displays a Blue Screen (aka Blue Screen of Death - BSOD) and does not reboot.

3.6 SID Cloning

Each system must have a unique Security ID if it is going to be on a network. The SID is created from the Product Identification Number that identifies a licensed copy of Windows NT and other values within the system. Windows NT Embedded includes a low-volume cloning solution to change the SID for each image created by Target Designer.

The SID cloning utility will change the SID for the platform upon the first boot. The HKEY_USERS key can be used to verify that the SID has changed from one copy of the image to another. The image must be cloned to create individual images for read only boot partitions (i.e., El Torito, write filter boot partitions).

3.7 Exercise 2 - Headless System

Microsoft has added features to help OEMs run Windows NT in non-traditional desktop systems. Being able to build a system without a VGA card or mouse/keyboard support is a key feature for many embedded developers. Microsoft has created NULL-VGA, NULL-Mouse, and NULL-Keyboard drivers to allow you to build headless NT systems. In this exercise, we will build a headless system and connect to the system via Serial Remote Administration and Hyperterm. The exercise will also have a reduced footprint with the selection of the Minimal OS and No Page File.

Keep the keyboard, mouse, and video display attached to the target platform. Even though this is a headless image, the goal is to prove that the system is truly headless by seeing what comes out on the display and what you can or cannot type in on the keyboard.

3.7.1 Build Image

1. Add a new configuration under the NTE Class collection.
2. Rename the configuration to Exercise2.
3. Select the following components. (This time nothing is in bold, walk the tree from the top down):

Capability	Component	Size	Check Box
System / Platform / SystemRole	Workstation System	0	
System / Platform / User Identification	Automatic Login	0	
System / Platform / Core OS	Minimal	7.87MB	
System / Platform / HAL	Standard HAL	958KB	
System / Platform / Virtual Memory	No Page File	0MB	
System / Devices / Storage / Floppy Disk	Floppy	18KB	
System / Devices / Storage / Fixed Disk	EIDE	0	
System / Devices / Storage / Fixed Disk	SCSI Disk	15KB	

System / Devices / Storage / CD-ROM	EIDE CD-ROM	0	
System / Devices / Storage / CD-ROM	SCSI CD-ROM	22KB	
System / Devices / Storage / File Systems	FAT	139KB	
System / Devices / Display / Display Drivers	Null VGA	99KB	
System / Devices / Keyboard / Keyboard Layout	US Keyboard Layout	9KB	
System / Devices / Keyboard / Keyboard Drivers	Null Keyboard Driver	4KB	
System / Devices / Keyboard / Keyboard Input Local	English (US) Input Locale	7KB	
System / Devices / Mouse	Null Mouse	0	
System / Devices / BUS Devices SCSI / SCSI Adapters	EIDE SCSI Adapter	0	
System / Devices / BUS Devices SCSI / SCSI Drivers	EIDE SCSI Driver	27KB	
System / Devices / Serial Ports	COM1	0	
System / Management / Management Applications / Remote Administration	Serial Console Administration	143KB	
System / Management / Management Services	Event Logging	377KB	
System / Shared System Components	Console Administrator	56KB	
System / Shared System Components	Display	25KB	
System / Shared System Components	Keyboard Drivers	9KB	
System / Shared System Components	Keyboard Locale	135KB	

System / Shared System Components	Mouse Common	9KB	
System / Shared System Components	SCSI	58KB	
System / Shared System Components	Serial Ports	44KB	
System / Shared System Components	Virtual Memory	0	
Applications / Shell	Command Shell	0	
Applications / Utilities	Windows NT Command Prompt	262KB	

4. View the Properties of the Serial Admin Component. Make sure that the properties are set to the following:

   Port should be set to 1.
   Baud Rate: 19200
   Data bits: 8
   Parity: None
   Stop Bits: 1

5. Run a dependency check on the configuration. If there are any missing dependencies or conflicts, double check the list above with the components selected.
6. Build the image. A dialog box will indicate that the x:\EmbeddedOS directory is not empty. Click YES to process and erase old contents.
7. Once the image is built, use a download method to download the new OS to the target. See Exercise 1b for the Remote Recover method.

When the target boots it will stop at the NTDETECT blue screen. The NULL video driver is loaded, but video data is not going to the monitor. A mouse icon is not available so the mouse will not work. Try typing on the keyboard and verify that nothing happens. Now reboot the target without the video card, keyboard, and mouse attached (your BIOS should be able to disable keyboard and video checking at POST).

3.7.2 Prepare HyperTerminal on the Host Machine

1. Make sure that the NULL modem cable is connected between your host system and your target (COM1).

2. In your host machine, open a HyperTerminal session and setup the session. When the New Connection Dialog pops-up, call the session NTE Serial Connect.
3. In the "Connect to" dialog, make sure that the "connect using" is set to COM1 (or COM2).
4. Set the COM port settings to match the above setting for Serial Remote Administration.

When the session is open and connected, you will be asked for a user name and password. The User and Password are case sensitive.

User name: **Administrator**
Domain:
Password: **Administrator**

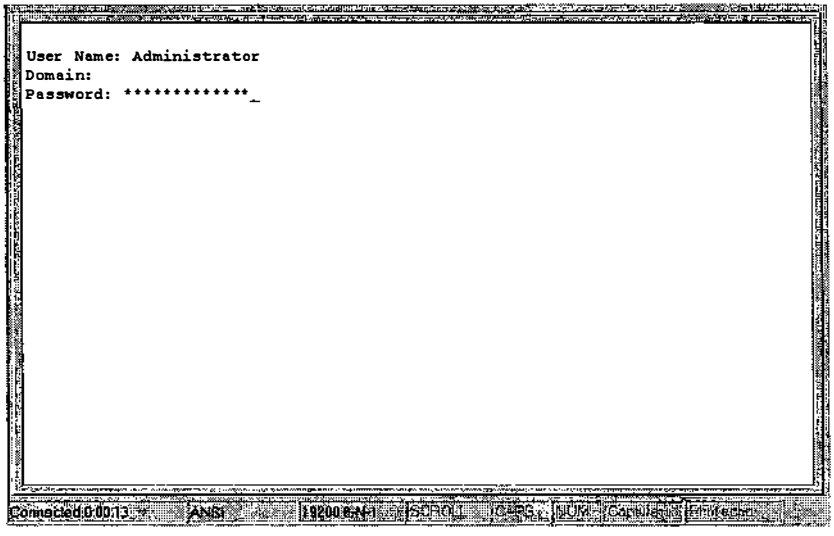


Figure 3.2: Logging into Remote Target

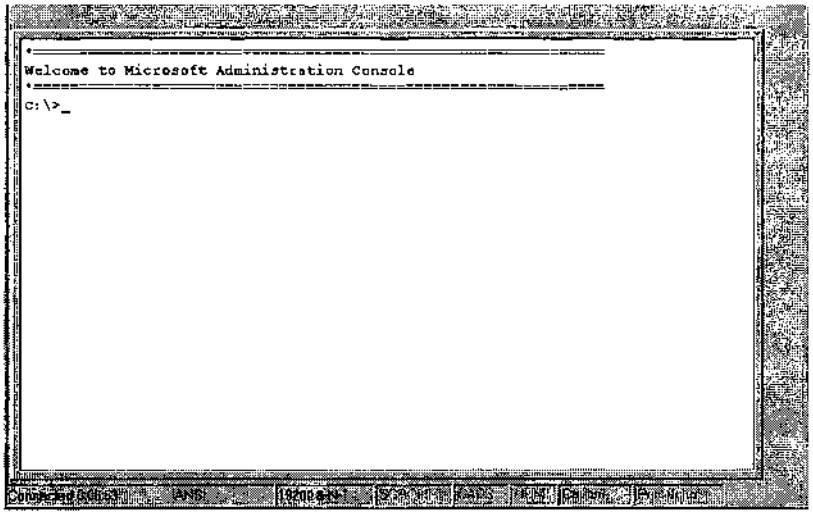


Figure 3.3: Welcome Screen - Login successful

3.7.3 File Transfers

Once you have a connection, you can perform DOS commands to view directories and copy and delete files. You can also copy files from one machine to another. HyperTerminal can send files to other terminals, using various protocols. The Serial Remote Administration Service implements ZMODEM protocol. To specify the protocol HyperTerminal uses for sending or receiving files, choose ZMODEM as the current protocol.

Sending Files from Host to Target Machine Using HYPER TERMINAL

To send a file from the host to the target machine, use "Transfer/Send File" from the menu or press the *send* icon. You can specify a file to send. As the file is sent to the target, a send progress dialog will be opened.

You can also cancel sending a file in the middle. The file will be saved in the current directory (of the *cmd.exe* shell) and will **replace** any existing file on the target machine.

For the *cmd.exe* shell to be ready to receive files, the line buffer should be empty and no application should be running in the console of the *cmd.exe* shell. (The HyperTerminal actually activates *rz.exe* in the *cmd.exe* shell before starting file transfer. The *rz.exe* utility is the utility on the target responsible for file receiving.)

If the send process continues uninterrupted and no error is reported, the file is guaranteed to reach its destination correctly (by CRC). If a file transfer is interrupted, a

partial file may reside in the target machine. The file name will be the same as in the host machine. (Path is stripped from the file.) More than one file can be sent this way (using wildcards).

Sending a File from Target to Host Machine - SZ.EXE Utility

Before sending a file from the target to the host machine, define in the HyperTerminal the target directory on the host for the files to arrive at (Transfer/Receive File on the menu). All received files will be saved in this directory. Files received will not replace existing files (instead a Number will be attached to the file name). HyperTerminal ignores the file path.

Activate the *sz.exe* utility to send a file by entering the following:

Sz.exe /s<FileName>

HyperTerminal will notice that a file is being sent and will open a file transfer progress Dialog. The file transmission can be halted at any time by pressing the Cancel button. File transmission is protected by CRC and the file is guaranteed to arrive correctly from target to host if no error is reported.

Sz.exe utility options:

Sz.exe has many options. A brief description follows ([ ] indicates default values):

Option	Description
/sSourceFileName	Source file name (can include a full path). No Space between the 's' and the file name.
/tTargetFileName [SourceFileName]	Name of the sent file on host (HyperTerminal will ignore the path).
/xEscapeControlCodes : + or [-]	Specify whether binary chars will be sent within the ZMODEM stream.
/cCrcType: [32] or 16	Specify the type of CRC used for transfer (32 bit or 16 bit CRC).
/oOverWrite:	Specify the file overwrite behavior for HyperTerminal to use.
/iIncludePath : [+] or –	Specify if path is to be sent to HyperTerminal (HyperTerminal will ignore path information).
/eEolConvert: + or [-]	Specify if Eol characters are to be converted before being sent.
/mXferMethod : WINDOW or [STREAMING]	Specify the transfer method of data packets (STREAMING has better throughput but WINDOW is better for noisy lines).

3.7.4 Transferring a File

1. Make sure that you are connected to your target system.
2. Create a text file using NotePad or WordPad on the desktop. Save the file to your Host's C drive root.
3. In HyperTerminal, select Send File from the Transfer menu.

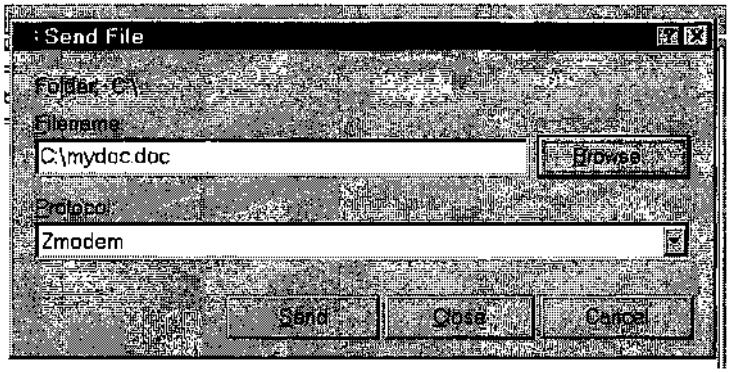


Figure 3.4: Transfer File from Host to Target

4. Locate and send the text file you created in Step 2 to your target machine. You will see a transfer dialog appear indicating the status of the transfer. The file will be located in the target's C drive root.
5. Now let's transfer the file back to your host machine. In the command window type the following:

   C:>sz /smydoc.doc (where mydoc.doc is the name of your text file) Once you hit return the file transfer will begin.

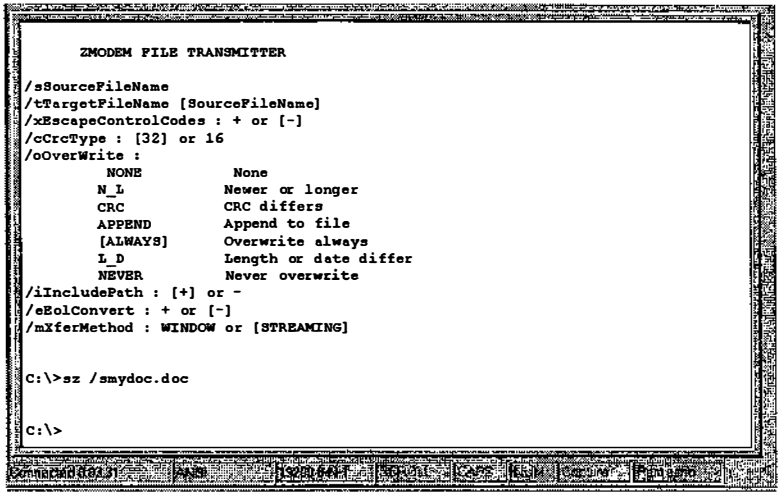


Figure 3.5: Transfer File back to Host

3.8 Exercise 3 – Embedded Features: DiskOnChip, Command Shell, Shutdown Utility, and the Write Filter

In the last exercise, you experimented with a few of the embedded features that Windows NT Embedded provides. In this exercise, we will continue to explore a few more features.

M-Systems' DiskOnChip is a popular device supported by many Embedded PC manufacturers. Flash technology has taken off dramatically over the past few years, resulting in small flash disks that are economical, viable, and robust enough for most embedded systems. OEMs are concerned about hard drive corruption, which can happen if a system is unsafely shut down. Windows NT can boot from a read-only partition. The write filter component directs all writes to a RAM disk, thus protecting a boot partition from data corruption.

Windows NT allows you to replace the explorer shell with an alternative shell. NT Embedded also includes some of the popular utilities that are found in the resource kits. The shutdown utility is very helpful in headless systems, allowing remote administrators to connect over a modem, access a remote system, and shutdown/restart a system.

In this exercise, we will implement a combination of these embedded components. The DiskOnChip will be write-protected to prove that drives can be made read-only. The command shell will be used to run a batch file to start other NT applications, and to shut down the system when the last application is closed.

3.8.1 Hardware for Exercise

The AAEON 5894 has support for the DiskOnChip 2000. You may choose to use another platform as long as the DiskOnChip is supported.

3.8.2 Building Image

1. Add a new configuration under the NTE Class collection.
2. Rename the configuration to Exercise3.
3. Select the following components:

Capability	Component	Size	Check box
System / Platform / SystemRole	Workstation System	0	
System / Platform / User Identification	Automatic Login	0	
System / Platform / Core OS	Minimal	7.87MB	
System / Platform / HAL	Standard HAL	958KB	
System / Platform / Virtual Memory	No Page File	0MB	
System / Devices / Storage / Floppy Disk	Floppy	18KB	
System / Devices / Storage / Fixed Disk	DiskOnChip	31KB	
System / Devices / Storage / Fixed Disk	EIDE Disk	0	
System / Devices / Storage / Fixed Disk	SCSI Disk	15KB	
System / Devices / Storage / File Systems	FAT	139KB	
System / Devices / Storage / Storage Filter Drivers	Write Filter	17KB	
System / Devices / Display / Display Drivers	VGA	10KB	

System / Devices / Keyboard / Keyboard Layout	US Keyboard Layout	9KB	
System / Devices / Keyboard / Keyboard Drivers	PC/AT Keyboard	29KB	
System / Devices / Keyboard / Keyboard Input Local	English (US) Input Locale	7KB	
System / Devices / Mouse	Logitech	29KB	
System / Devices / BUS Devices SCSI / SCSI Adapters	EIDE SCSI Adapter	0	
System / Devices / BUS Devices SCSI / SCSI Drivers	EIDE SCSI Driver	27KB	
System / Devices / Serial Ports	COM1	0	
System / Shared System Components	Display	25KB	
System / Shared System Components	Keyboard Drivers	9KB	
System / Shared System Components	Keyboard Locale	135KB	
System / Shared System Components	Mouse Common	9KB	
System / Shared System Components	SCSI	58KB	
System / Shared System Components	Serial Ports	44KB	
System / Shared System Components	Virtual Memory	0	
Applications / Shell	Command Shell	0	
Applications / Utilities	Windows NT Command Prompt	262KB	
Applications / Utilities	Shutdown Utility	34KB	
Applications / Accessories	Calculator	124KB	
Applications / Accessories	Notepad	225KB	

4. Create a Batch file at the root of your host's C:\ drive and call it **MYBATCH.BAT**.
5. Edit the MYBATCH.BAT file to contain the following:

```
Echo off
CLS
Echo Hello World
Start calc.exe
Start /wait notepad.exe
Shutdown /L /Y /C
```

The Start command will launch applications Calculator and Notepad. The WAIT option pauses the batch file from continuing until Notepad is closed. The system will shut down within 20 seconds.

6. In Target Designer click on the Exercise3 configuration, and select files view.
7. Right click in the files view pane and select Add File.
8. Use the browse button, and select the Mybatch.bat file.
9. In the add file dialog, set the target directory to \.

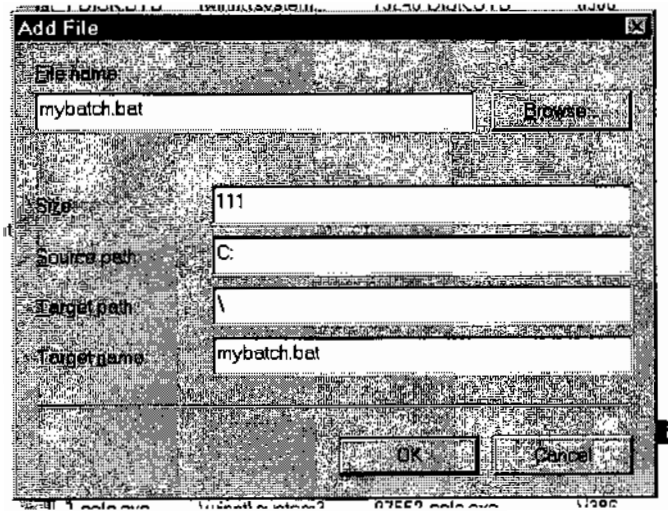


Figure 3.6: Add a File to the Configuration within Target Designer

10. Click OK, and the file will be added to the list in a different color.

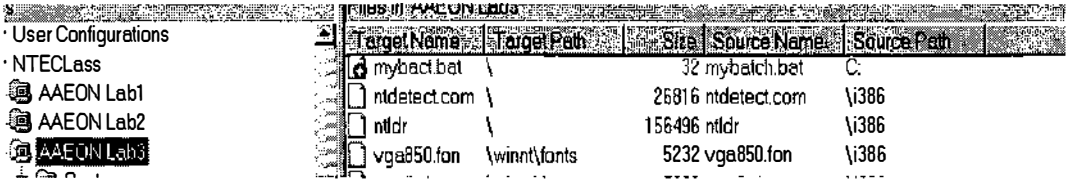


Figure 3.7: File Added Listed in a Different Color

11. Change the properties of the command shell component:

Cmd.exe /k c:\mybatch.bat runs the script and keeps command window open.

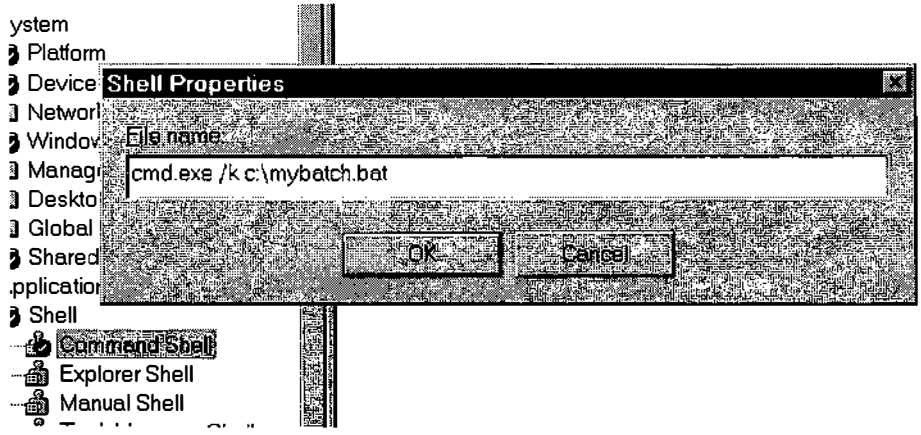


Figure 3.8: Running a Batch File upon Boot-up

12. Right click on the Write Filter component and select properties.
13. Add the DiskOnChip driver to the list of read-only drives:

\Device\Harddisk0\Partition1

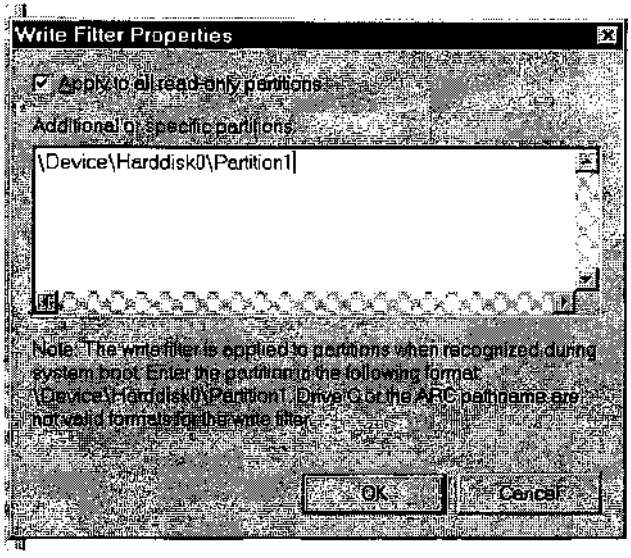


Figure 3.9: Adding a Partition to the Write Filter

Warning: Make sure that you spell the partition format information correctly. Capital letters are a must. The write-filter and Target Designer will not catch format errors.

14. Run a dependency check on the configuration. If there are any missing dependencies or conflicts, double check the list above with the components selected.
15. Build the image. A dialog box will indicate that the x:\EmbeddedOS directory is not empty. Click YES to process and erase old contents.
16. Once the image is built, use a download method to download the new OS to the target's hard drive, and copy MYBATCH.BAT to the DiskOnChip. See Exercise 1b for the Remote Recover method.

3.8.3 Running the Image

When the system boots, Hello World will echo to the command window screen, and Calculator and Notepad applications will run.

1. Create a file using notepad and save the file to the DiskOnChip. Notice that the file can be re-opened and moved around like any file, until you shut down the system.
2. Close Notepad and watch the system shut down.
3. Reboot the target.
4. Try to open the file using note pad. Notice that the file is not on the DiskOnChip.

The write filter can also be used on hard drive partitions to protect sensitive data from being overwritten.

Running a batch file from the command shell is handy for a headless system. The batch file can be used to start the primary application, or you can use a batch file to run the NET.EXE utility to share printer and file resources on a network.

3.9 Summary

The embedded features added to Windows NT allow it to reside in non-desktop systems. NTE can exist as a headless system with a minimal footprint. Exercise 3 introduced a method to run applications from the command line in batch mode. Later exercises will introduce other embedded features such as graphical remote administration and CD-boot.

Chapter 4 COMPONENT DESIGNER - DEVELOPING KDFS

4.1 Component Designer Overview

The second authoring tool is Component Designer, a companion tool to Target Designer, which is used to define and add new components into the TDSD component tree. The files created by Component Designer are called component (or "kit") definition files (KDFs). KDFs created within Component Designer are imported into the TDSD via Target Designer. Once the component is added to the TDSD, the component is available for all configurations, old and new.

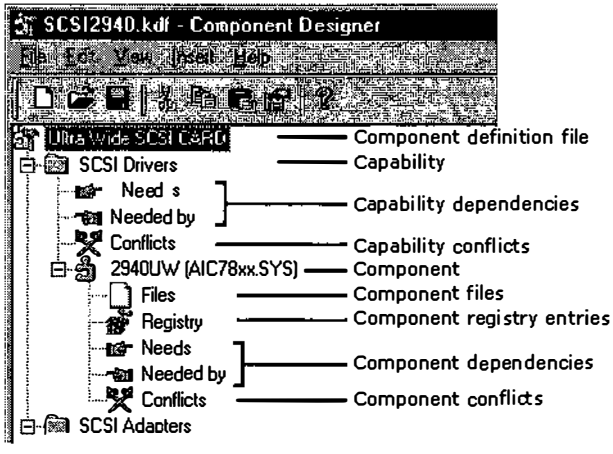


Figure 4.1: Component Designer

Because Windows NT Embedded already comes compiled, most of the work is in the development of component definition files (KDFs) and deployment of the final target. You can create a KDF file by editing a sample KDF file in any text editor, or by using the Component Designer tool for viewing and manipulating the data.

KDFs information is in a format that can be imported into the TDSD, which is one of the major reasons why INFs are not used. Information to create KDFs can be taken from INF files, but an installation and gathering method is easier to prove that an application or driver is working. KDFs contain sections that define components, capabilities, data, and relations.

Sections	Symbol	Description
Component Definition File		The root of the KDF. You can add Vendor Information and Comments (i.e., Copyright, detailed description of KDF), which are the properties
Capability		Each component must reside within a capability. You can create new capabilities with Component Designer (i.e., Sound Card)
Capability's Dependencies and Conflicts		Relationships can be established between a capability and other capabilities or components.
Component		Basic element in a configuration, which is comprised of files, registry hives, and relationships with other capabilities and components.
Files		Files list for the component. Each file has a source directory location and a target directory location.
Registry		Registry entries for the component.
Component's Dependencies and Conflicts		Relationships can be established between a component and other capabilities or components.

Table 2: KDF Sections

4.2 To KDF or Not to KDF

The big question that most of my students ask is: Why do we need to create KDFs? Applications and drivers can be installed on an NTE system without having to go through KDFs. You do not need to create components, but there are advantages.

Creating components are essential for upgrades and support. Applied or embedded systems can last for 10 years or more. In order to support a product over its lifetime, tools, development work, and other elements of a product must be backed up so that support personnel can reproduce problems. Like the CFG file discussed in Chapter 2, the KDF is a small text file that is easy to place into version control. Unlike CFG files, KDFs are comprised of binaries, registry hive files, and Target Designer Extensions (TDX), which must be all kept with the KDF for future recovery.

Some systems may not allow you to install applications or components very easily. Headless systems, systems without removable media, or specially designed hardware require the use of KDFs.

This chapter covers the development of KDFs for both applications and device drivers. KDFs can be quite complicated; the exercises in this section provide an overview of the development involved. Your specific KDF may not be the same, but the exercises provide methods that can be used to develop any KDF.

4.3 Component Designer Step-by-Step

Component Designer's GUI interface helps to ease the creation of components. Let's go through the steps to create a KDF for NOTEPAD.EXE. Since NOTEPAD.EXE is a self-contained application, the KDF will be very short.

1. Start Component Designer
2. Right click on the Component1 in the left pane and select Properties. Change the KDF name to Notepad2 and click OK.
3. Right click on Notepad2 and select Add Component.
4. A dialog box will appear asking for component information. In this dialog you will provide a name for the component, any Target Designer extension or TDX files, and the capability the component will be listed under. In this case Notepad2 will be under Accessories.

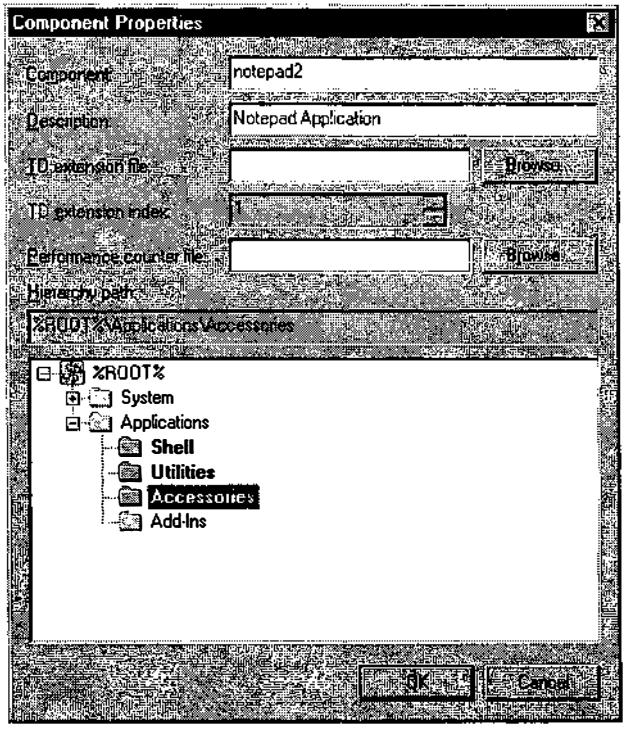


Figure 4.2: Adding a Component Dialog

5. Once you click OK, a hierarchy tree will appear. Next you will enter the file location for your component. In this example, go to the c:\winnt directory and locate your local NOTEPAD.EXE application.

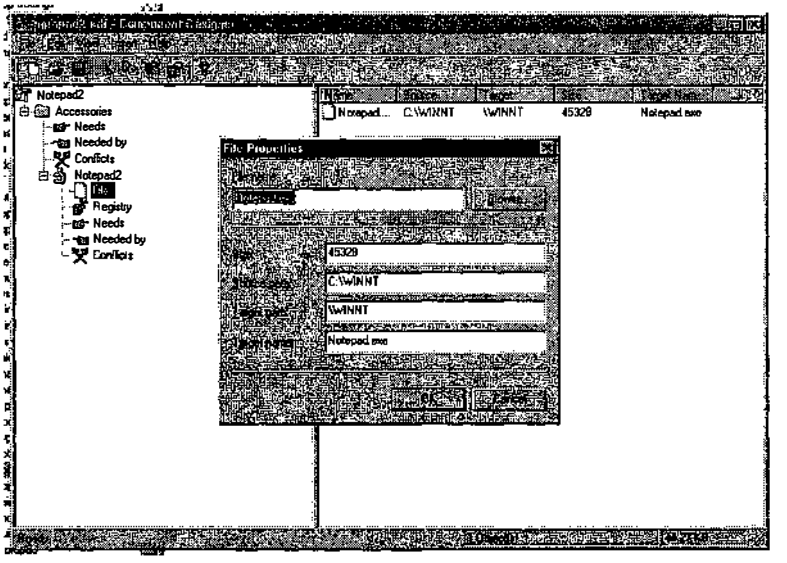


Figure 4.3: Entering the File Information

6. The next step is to enter the registry entries. NOTEPAD.EXE doesn't need any registry entries so this section will be left blank.
7. Next, list any dependencies or conflicts; in this case, the current NOTEPAD.EXE component.

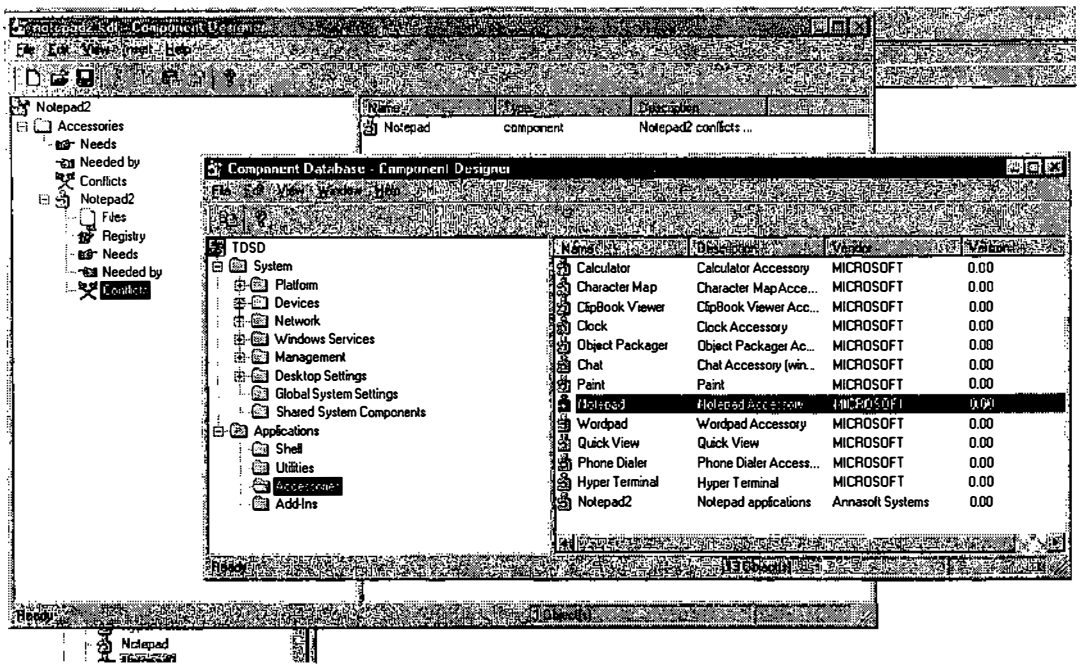


Figure 4.4. Adding Conflicting or Dependent Component Information

8. Save the component as Notepad2.kdf.
9. Now let's import the KDF. Go to the Tools menu and select Component Management. Click the add button and find NOTEPAD2.KDF.
10. Click OK and the component will be added to the database. The component will now be listed in the configuration.

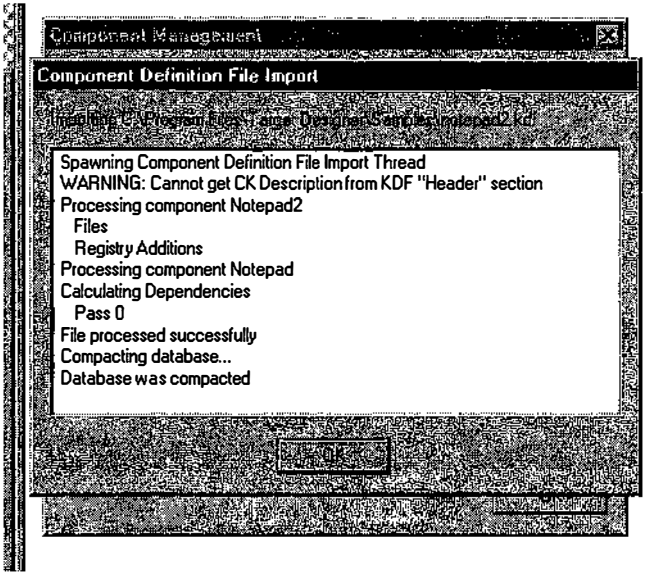


Figure 4.5: Importing a Component into Target Designer

Once you start adding components outside of the CD-ROM repository, the directory in which the file is located will become the repository for that component. If you change the location of the file, you must also change the repository location in Target Designer. This can be accomplished by selecting Settings from the Image Menu. Highlight the source of the repository and edit the entry.

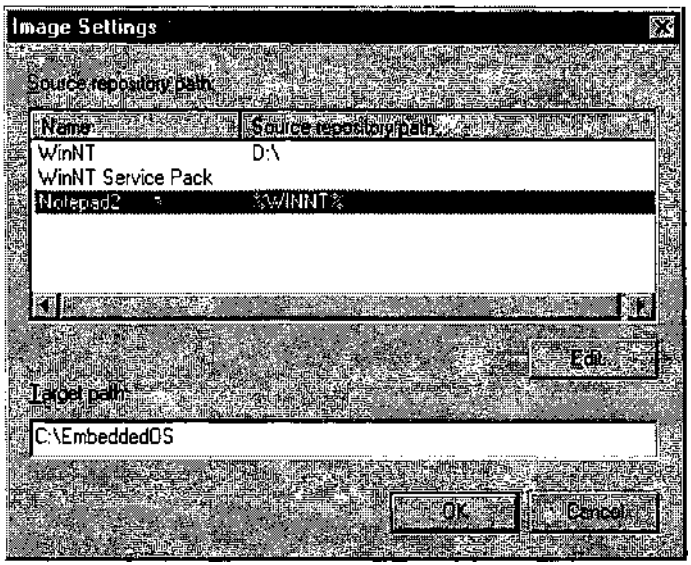


Figure 4.6: Source Paths to Repositories

The following is the KDF file for NOTEPAD2.KDF view in a text editor:

```
[Version]

Signature = "$Windows NT$"
KdfVersion = 1

[Header]

Name = Notepad2
Version = 4.0
ReleaseDate = 04/12/1999
Vendor = Annasoft Systems
OS = WINNT
OSVersion = 4.0
Platform = i386
Repository = "%WINNT%"
Description = ""
CodePage = 1252


[Capabilities]

1 = "Accessories"
```

```
[Components]

1 = "Notepad2", "Notepad applications", 1
2 = "Notepad"

[CapabilityDependencies]

[CapabilityComponentDependencies]

[ComponentCapabilityDependencies]

[ComponentDependencies]

1,2,N


[Directories]
1 = "\WINNT"
2 = "C:\WINNT"

[Notepad2.Files]
"Notepad.exe", 2, 1, 45328, "Notepad.exe"

[Notepad2.RegistryAdditions]
```

If the original Notepad component is selected within a configuration, selecting the new Notepad2 component will pop up a dialog to indicate the conflict between the two components.

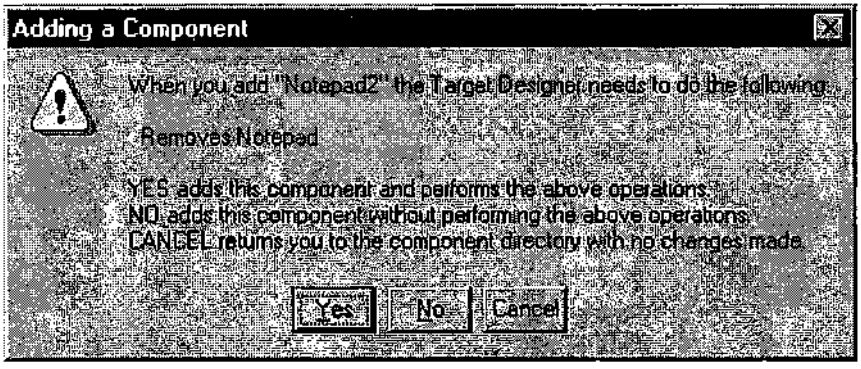


Figure 4.7: Dynamic Checking Indicates Conflicts and Dependencies

Once the component is imported into the TDSD, the new component is now available for all configurations. Any past configuration or new configurations have the Notepad2 component available. The ability to access new components in old configurations becomes very useful when trying to upgrade hardware and drivers.

4.4 TDSD Rules

Because you can add and remove components and capabilities from the database, there are rules that set how the Target Designer tree should be structured. The TDSD node can contain collection and configuration nodes.

- A collection can contain collection and configuration nodes.
- A configuration contains a tree of standard, predefined, non-extensible hierarchy nodes. This tree cannot be changed or extended.
- A hierarchy node can contain hierarchy nodes and capabilities. In other words, hierarchy nodes and capabilities can be siblings. However, the KDF can define only capabilities; the hierarchy tree is predefined.
- A capability can contain only components. That is, a capability cannot include other capabilities. Components and capabilities cannot be siblings in the tree.
- A component is always a leaf in the tree. Notice that capabilities or hierarchy nodes can also be leaves.

All hierarchy nodes, capabilities, and components defined in the tree must have a unique name.

Target Designer will fail to import a KDF that doesn't follow these rules.

4.5 Gathering KDF Information

The notepad KDF is the simplest example. If you develop your own applications and device drivers, you know all of the information required to create a KDF. Third party application and device driver KDFs are going to take the most work. There are a number of shareware utilities that can help with extracting information from applications and drivers, but no single tool fits all. The exercises in this section demonstrate the creation of KDFs from third party applications.

There are two types of KDFs: application and driver. Each KDF type requires a different approach for development.

Applications require that certain OS services or features are included in the image. MFC and OLE are required components for most Windows Applications. Finding all of the static

and dynamic DLLs is required for application KDFs. Registry entries for applications can be quite complex, and some of the keys are generated at boot time. Understanding which keys to choose takes a deeper understanding of the registry.

Device drivers require registry keys in order to load and set up parameters such as TCP/IP addresses or baud rates. There are dependences on other components and features, but a majority of the work is in gathering the registry keys. Two methods are used to gather registry keys: HIV (hive) File Method and the Local Method. Exercise 5 will introduce the HIV File Method, and Exercise 11 will introduces the Local Method.

There are a number of third party tools that can help gather the information dynamically and statically:

- Scanbin - Developed by Jean-Claude Bellamy, Scanbin can analyze an EXE file to determine the dependent DLLs and support files required by the executable. Visual Studio has a similar tool.
- NTFileMon - Developed by Winternals, this tracks files that are touched by the system as applications run.
- NTRegMon - Developed by Winternals, this tracks registry keys that are touched by EXEs and DLLs as the applications run.

4.6 Target Designer Extensions (TDX)

Some components allow you to set up the parameters before the OS build takes place. Components with properties are identified with a document in the puzzle piece icon. Components like the mouse driver and network adapter have property options in Target Designer. When you add a mouse component to your configuration, you can right-click on the component and select properties. A dialog that is similar to the mouse control panel dialog will appear. Here you can change the point speed, shape, and orientation. Microsoft calls these property boxes Target Designer eXtensions or TDXs. With the aid of Visual Studio 6.0 and the TDX SDK provided in Windows NT Embedded, you can create your own TDXs for your components. The TDX SDK and demo TDXs can be found in the \program files\Windows NT Embedded 4.0\SDK directory.

Exercise 6 walks through the development of a TDX, but there are a number of pre-built TDXs available. The TDXs can be found in the \program files\Windows NT Embedded 4.0\res\tdx directory. Most of the TDXs found in this directory are used by components already built in as part of the TDSD. Display.tdx, NetAdapter.tdx, modem.tdx, and mouse.tdx are TDXs that can be used with your components. The TDX must reside in the same directory as the KDF in order for Target Designer to import the component. Once a TDX has been built, the TDX can be added to the component properties section. The diagram below shows the Display.tdx added to a display component.

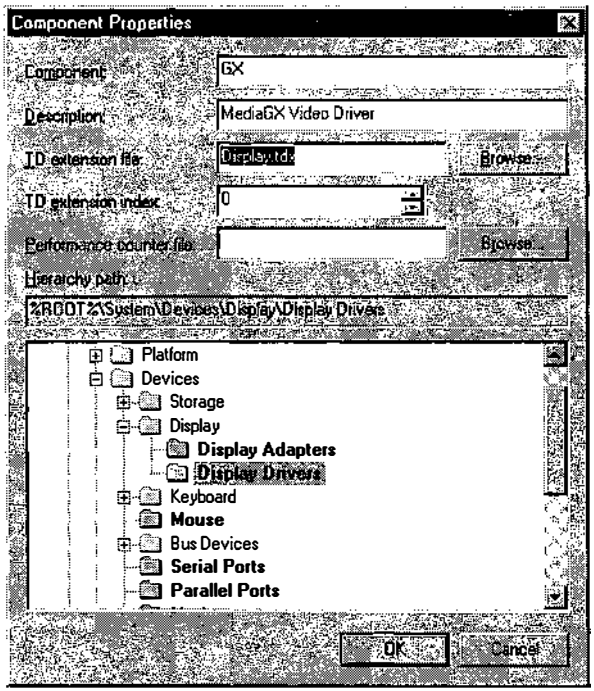


Fig 4.8 Display TDX for VGA Driver

The following is a description list of the TDXs found in the \program files\Windows NT Embedded 4.0\res\tdx directory

TDX	Description
AppleTalk	Used by the AppleTalk Protocol
AutoLogon	Used by the Autologon component
AutoReboot	Sets / Unsets the auto-reboot upon failure
Display	Pre-Set the display settings
FPNW	File and Print Services for Netware
IIS	Anonymous User Selection, used by IIS component
IPXSPX	Frametype setting for IPX/SPX Protocol
Modem	Modem settings
Mouse	Mouse settings
MsgBoxHandling	Used by System Management Interception component
NetAdapter	Set the network adapter settings: IRQ, BUS type, etc.

NetworkId	Sets the computer name and domain/workgroup name
SerialAdmin	Used by Serial administration component
SNMP	Pre-setup the Simple Network Management Protocol
SRVMGR	Used by the Service Settings component
Startup	Used by the Shell components
TCPIP	Pre-sets the TCP/IP address used by TCP/IP component.
Timezone	Time and date settings, used by the Time Zone Settings component
Winmodem	Modem setting for the WinModem component
Wrfilter	Used by the write filter component

4.7 Exercise 4 - Application KDF for MSPaint

In this exercise, we will create a KDF for MSPaint in order to demonstrate the basics for creating a component. Microsoft already includes Paint as a component, but in this exercise we will create a second MSPaint component that has a relationship with the old component. SCANBIN will be used to find all of the files necessary to run the MSPaint application.

4.7.1 Application KDF Basics

Although this exercise focuses on one particular application, the steps to gathering the KDF information are the same for any application.

A. Install the application on a system with Windows NT 4.0 SP5 or an NTE image.

B. Use utilities like Scanbin, FileMon, or RegMon to gather dependent files and registry entries. Most registry entries are created at run time so you can limit your registry search to HKEY_LOCAL_MACHINE.

C. Use Component Designer to create the final KDF.

4.7.2 Create KDF Structure

1. In Explorer create a Directory called c:\Local.
2. Under the local directory create a directory called c:\local\MSPaint.
3. Open Component Designer.

4. Right click on Component1 found in the left pane, and Select Add Component.
5. A dialog will appear asking for the name of the component, **MSPaint.**
6. Type in a description in the description text box: **Application KDF.**

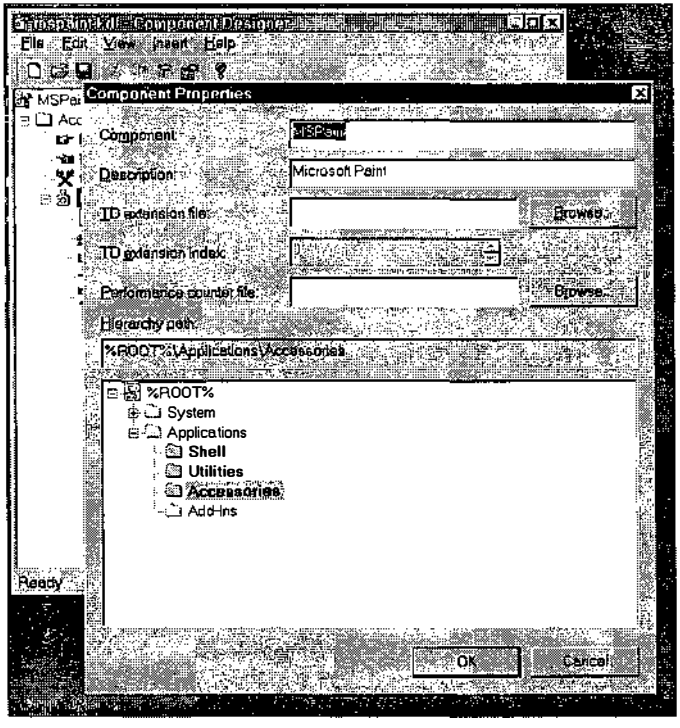


Figure 4.9: Create MS Paint Component

7. Target Designer (TD) Extension File and Performance Counter File can be left blank.
8. Finally, select the hierarchy path within Target Designer, Highlight Accessories under %ROOT%\Applications\Acessories.
9. Click OK when finished. The left pane will show a subset of the TD hierarchy with MSPaint as a component. Now we need to gather information about MSPaint.
10. Right click on Component1, and add the properties for Component name, Vendor, and Description:

    Component name: **MY MS Paint APP**
    Vendor: **Microsoft**

Description: **Microsoft Paint**

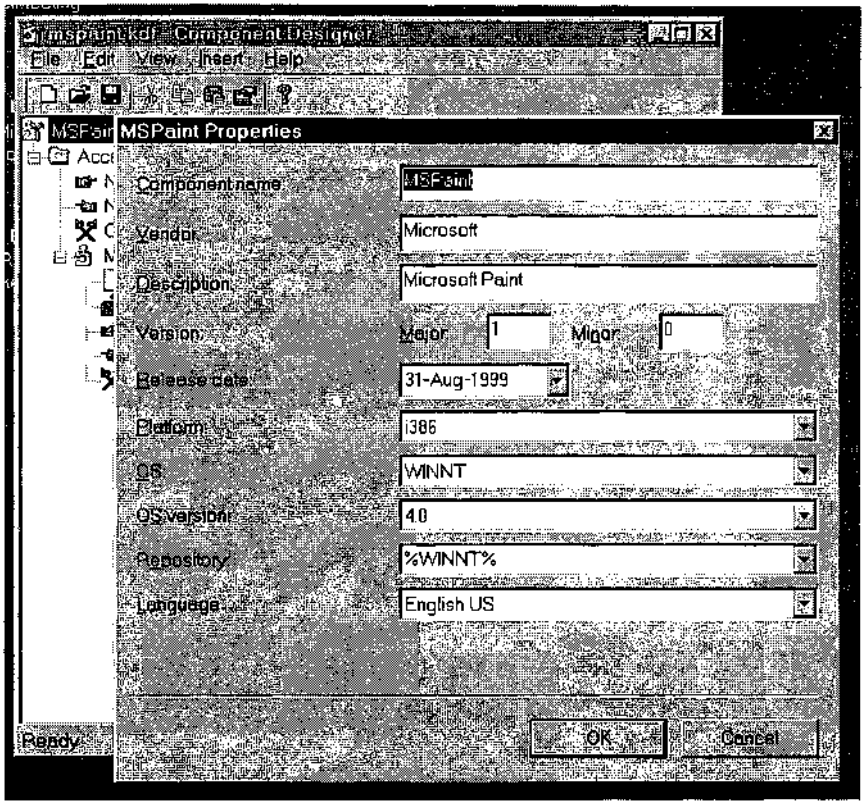


Figure 4.10: Set the Component Properties

11. Right click on MSPaint and select Comments

    Enter: **Copyright © 1999 ACME Corp.**

12. Save the mspaint.KDF to the c:\local\MSPaint directory.
13. Copy the MSPAINT.EXE file to the c:\local\MSPaint directory. **(Be careful not to MOVE MSPAINT.EXE file or create a short cut).**
14. Under the MSPaint Component, right click on Files and select "Drag for Explorer". Select the MSPAINT.EXE file that is in the e:\local\Mspaint directory.
15. Change the target directory to **\winnt\system32.**

16. Next, since we already have Paint as a component, there is a conflict between the Paint component already in Target Designer. Under the MSPaint Component, right click on Conflicts and select "Drag from Database", and drag the Paint component into the Conflicts pane within Component Designer.
17. SCANBIN is a shareware utility that can collect static information about your application. Open SCANBIN and click OK on the next two dialog boxes. See Chapter 1 for the website reference to download SCANBIN.

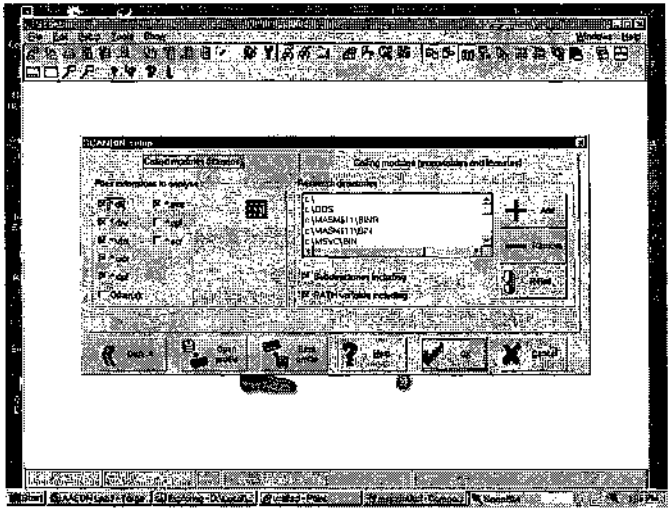


Figure 4.11: Scanbin

18. From the File menu, select Binary File Open ...
19. Locate and select the MSPAINT.EXE file found in \WINNT\SYSTEM32\MSPAINT.EXE. When you click OK the static analysis will be performed. When completed you will get a summary of the analysis.

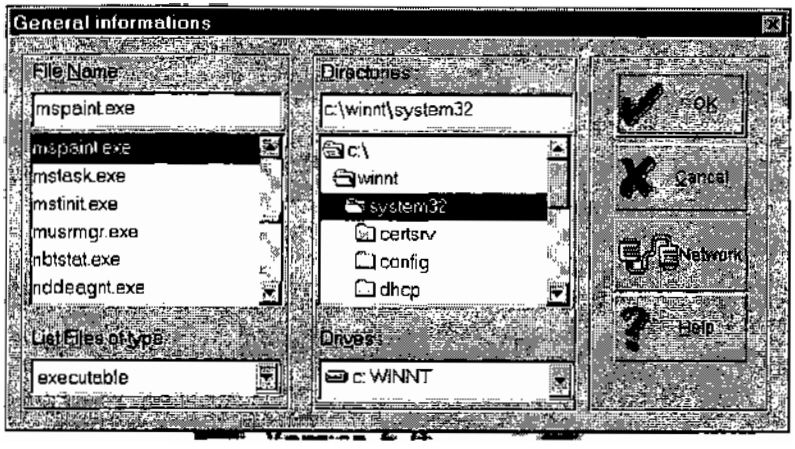


Figure 4.12: Open mspaint.exe within Scanbin

20. Click on the DLL Used tab. Notice all of the DLL files that are called direct and undirect. We will use this list to find the dependent files and components. Some of these files are already part of other components in the database, and some of these components are required to build the OS. Using the Files view in Target Designer, walk the selected components in Exercise3 and find all the components that contain the files used by MSpaint. Some files will be found in more than one location.

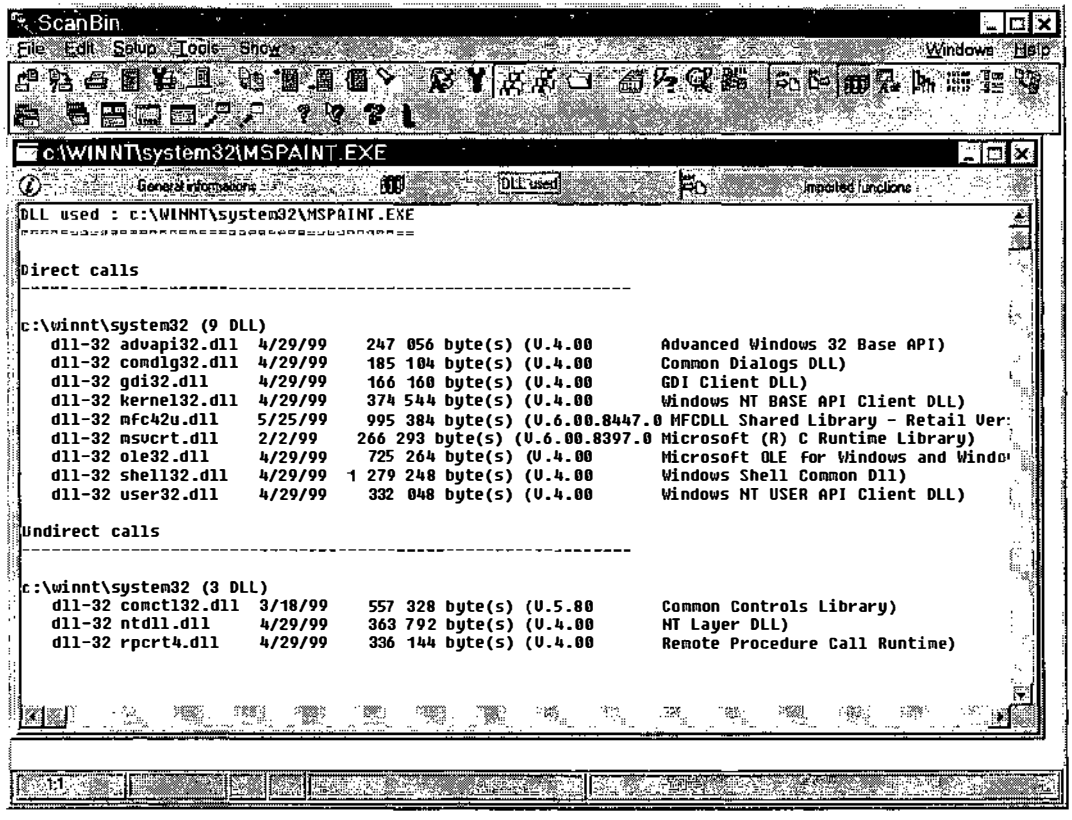


Figure 4.13: Static DLL Analysis of mspaint.exe

Dependent DLL	Component
Advapi32.dll	
Comdlg32.dll	
gdi32.dll	
Kernel32.dll	
mfc42u.dll	
msvcrt.dll	
ole32.dll	
shell32.dll	
user32.dll	
comctl32.dll	
ntdll.dll	
rpcrt4.dll	

No, I am not kidding. I strongly recommend that you walk through each of the selected components. The purpose of step 20 is to become more familiar with the components. You will see that files can be found in more than one component. As you continue to work with the various components and KDFs you will become familiar with the component relationships.

21. There should be two files that are not part of a selected component OLE32.DLL and MFC42U.DLL. These two files are part of components that are not selected. The goal and the challenge of NTE is to create small footprints of the NT OS where all of the components are fully functional. The goal here is to use the Minimal OS core component. OLE32.DLL is part of the OLE/COM component. In Component Designer, right click on the NEEDS section under MSPaint component, and add the OLE/COM component.
22. MFC42U.DLL is the only file left to include as part of the KDF. MFC42U.DLL is part of the Standard OS Component. Since we want to keep the minimal footprint, we will add MFC42U.DLL to the FILES section. In Explorer, locate and copy the MFC42U.DLL to the c:\local\mspaint directory. **(Be careful not to MOVE MFC42U.DLL).**
23. In Component Designer, right click on the FILES section under MSPaint component, and Drag Explorer MFC42U.DLL from the c:\local\mspaint location.
24. Edit properties of the MFC42U.DLL listing in the right pane, and change the Target Path to \winnt\system32.
25. Save the Mspaint.KDF file, and close Component Designer.
26. Close Scanbin and do not save any Scanbin database information.

If you open the Mspaint.kdf in a text file it should look like the following:

```
;Copyright (c) 1999 Annasoft Systems

[Version]
Signature = "$Windows NT$"
KdfVersion = 1

[Header]
Name = MSPaint
Version = 1.0
ReleaseDate = 08/31/1999
Vendor = Microsoft
OS = WINNT
```

```
OSVersion = 4.0
Platform = i386
Repository = "%WINNT%"
Description = "Microsoft Paint"
CodePage = 1252

[Capabilities]
1 = "Accessories"

[Components]
1 = "MSPaint", "Microsoft Paint", 1
2 = "OLE/COM"
3 = "Paint"

[CapabilityDependencies]

[CapabilityComponentDependencies]

[ComponentCapabilityDependencies]

[ComponentDependencies]
1,2,Y
1,3,N

[Directories]
2 = "D:\Local\MSpaint"
3 = "\winnt\system32"

[MSPaint.Files]
"MSPAINT.EXE", 2, 3, 339728, "MSPAINT.EXE"
"Mfc42u.dll", 2, 3, 995384, "Mfc42u.dll"

[MSPaint.RegistryAdditions]
```

27. In Target Designer, select Component Management under the Tools menu.
28. Click the Add... button and open the mspaint.kdf. The KDF will be imported into the Target Designer database. The location for the two binary files listed in the KDF must stay in the c:\local\mspaint directory. If they are moved, the configuration that has the MSPAINT component will crash during a build.
29. In Exercise3, select the new MSpaint component.
30. Edit the Command Shell, and remove the MYBATCH.BAT from executing at start-up.
31. Rebuild the Exercise3, prepare target's hard drive, and download to the target.

32. Reboot the target with the new image and run mspaint.exe from the command prompt.

We see that the new mspaint component contains only a minimal set of files in order to run. The MFC DLL is only found in the Standard OS, but adding the file to the mspaint component allowed for a smaller image to be built.

The local subdirectory was created so that all of the pieces of the KDF can be kept in one location, which simplifies Version Control.

4.8 Exercise 5 - Driver KDF for Network Card, Hive File Method

Exercise 4 provided a basic overview of some of the KDF features. Exercise 5 will expand on Exercise 4 and introduce a method to gather registry hives. Drivers depend on registry settings, and one of the most commonly used devices in an NT system is the network adapter. In this exercise, we will build a KDF for a network adapter using the Hive (HIV) File method.

The AAEON 5894 uses a RealTek 8139A 10/100 Ethernet controller. A Realtek 8029 driver comes with NTE, but this will not work with the 8139A. AAEON supplies the Windows NT 4.0 driver with the 5894 board. In this exercise we will create an 8139 KDF and build a Workstation image that connects to a network. If you do not have the AAEON 5894 or the RealTek 8139, the concepts presented here are the same for any Ethernet network card.

Registry entries will be captured to create the KDF. There are hundreds of registry entries in an NT system. To shorten this exercise, the registry keys will be located for you. Understanding how to locate registry keys will be discussed later in this section.

In this exercise you will gather the registry keys in HIV files to create the final KDF. To create the KDF, you will perform the following:

A. Build a Standard OS image that supports Control Panel apps. Access to the Control Panel will allow you to add network drivers and services.

B. Copy over the i386 subdirectory so the Network setup Wizard can install the network services that are not part of the configuration.

C. Create the basic structure of the KDF.

D. Install the 8139A NT driver on the target.

E. Capture the 8139A registry entries using REGEDT32

F. Copy the HIV (HIVE) files to your development machine.

G. Finish the KDF, and import the components into the Target Designer Database.

H. Rebuild a new configuration with the new network components.

4.8.1 Device Driver KDF Creation Basics

Although this exercise focuses on one particular device driver, the steps to gathering the registry entries are the same for any hardware device.

A. Install NT 4.0 SP5 or an NTE image to your target system.

B. Make sure that you have access to the i386 NT installation directory so that NT can load services that your device driver needs. Typically when you install device drivers such as network device drivers for the first time, network setup wizard will ask for the NT installation CD to add the network services. If your target doesn't have access to a CD, than you need to copy the i386 directory to your hard drive.

C. Install the device driver in question.

D. Use REGEDT32 to capture the registry entries. (You could install NTE on your target system, so that you can Drag-N-Drop the registry entries directly into the KDF.)

E. Use Component Designer to create the final KDF.

4.8.2 Prepare Target to Add Network Support

1. Rebuild and download Exercise1. After the OS is downloaded, DO NOT DISCONNECT Remote Recover.
2. Copy the \i386 subdirectory found on the NTE CD-ROM to the target. We need the files found in the i386 directory to add networking support and capture information needed to create the KDF.
3. Disconnect Remote Recover when the copy is completed, and reboot the target.
4. In the **host** system create a new directory, c:\local\8139.
5. Copy the Win NT 4.0 driver installation files (OEMSETUP.INF, RTL8139A.SYS, and RTSNTHLP.HLP) for the 8139A to a blank disk and to c:\local\8139.
6. In the **target** system, open Control Panel and select Networking. The computer will pop a dialog asking if you want to install networking now. Click YES.

7. The Network Wizard will search your system for connection types. Make sure that "Wired to network" is checked and click NEXT.
8. Insert the floppy containing the RealTek driver and click Select from List.
9. A list of adapters will appear. Click Have Disk.
10. Select drive A and click OK.
11. Select RealTek Driver and click OK.
12. Realtek driver should be checked and listed in the text box, and click NEXT.
13. The next dialog asks for the protocol, check TCP/IP, and click NEXT.
14. Network Service, Keep Defaults, and click NEXT.
15. Click Next again.
16. The Wizard will install the necessary files. A dialog pops up asking for the location of the NT files. Type c:\i386 in the text box and click CONTINUE.
17. The dialog will pop up again. Type c:\i386 in the text box and click CONTINUE.
18. Set the duplex mode to Auto.
19. Installation will continue, a dialog will pop-up asking if you want to use DHCP, and click NO.
20. The TCP/IP Setup Screen will appear:

    IP Address: 10.1.1.250

    Subnet mask: 255.0.0.0

    Or select the TCP/IP address that connects the target to your LAN.

Click Apply, and then Click OK

21. In the Bindings screen, click NEXT.
22. Click NEXT.
23. Change Workgroup Name to NTECLASS.
24. Click finish and click YES to restart computer.

4.8.3 Capture Registry Entries for KDF

The reason for installing the network driver was to capture the registry information required by the driver. There are a lot of registry entries, and it can be quite difficult to locate which registry entries are required for a given driver. You can use the Components in Target Designer as a guide to help locate registry entries for the network cards.

To capture registry entries you have three options:

1. Type each entry in by hand, which is cumbersome and takes too much time.
2. You can install NTE on your target machine and Drag from Local Registry the entries into Component Designer.
3. You can use REGEDT32.EXE to capture HIV (HIVE) files.

We will use Option 3 for this exercise. Network Adapters are broken into two Components: the adapter and the driver. Each component has distinct registry entries.

25. On the target machine, make sure that networking was installed and you can connect to the NTECLASS work group.
26. Open REGEDT32.EXE.

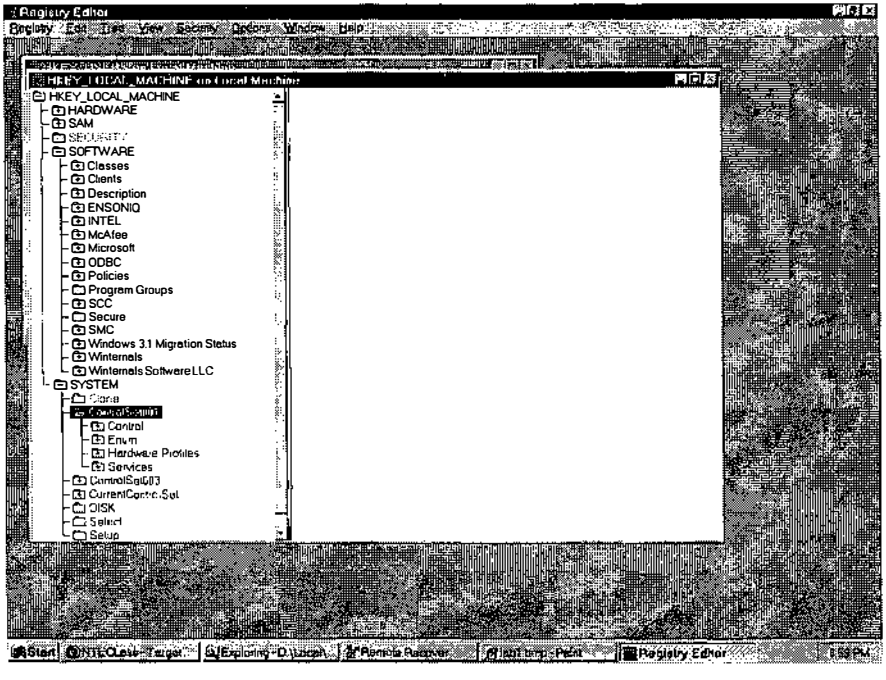


Figure 4.14: Access the Hives from Regedt32

27. We will use the SAVE KEY to save a branch of registry entries. Transverse the HKEY_LOCAL_MACHINE (HKLM) and locate the following registry entries and save each entry as a .HIV file. When you save the HIV file, make sure that you have highlighted the entry listed in the table below. (Highlight the end of the branch that is in bold.) If you do not do this correctly you may get registry entries that are not needed or you may not capture all of the entries.

Component	Registry Entry	File
AAEON 5894 (RTL8139A)	HKLM\Software\Microsoft\Windows NT\CurrentVersion**NetworkCards**	C:\RTL1.HIV
AAEON 5894 (RTL8139A)	HKLM\System\ControlSet001\Services**RTL81391**	C:\RTL2.HIV
RTL8139A	HKLM\Software**Realtek**	C:\RTL3.HIV
RTL8139A	HKLM\System\ControlSet001\Services**RTL8139**	C:\RTL4.HIV

You may also want to capture the errorlog registry key, but the errorlog registry key will not be needed for this exercise.

28. Copy the HIV files to the c:\local\8139 using the blank transfer disk.

4.8.4 Creating the KDF on the Host Machine

29. Open Component Designer, and create **two** new components within the same KDF:

Component: **AAEON 5894 (RTL8139A)**
Description: **Ethernet for AAEON Platform**
Hierarchy Path: **Network Adapters**

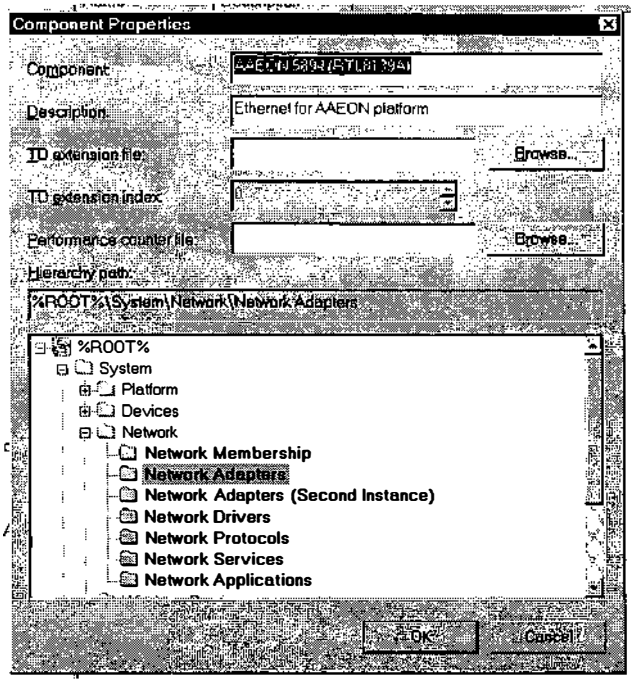


Figure 4.15: Create the Adapter Component.

Component: **RTL8139A**
Description: **RealTek Driver**
Hierarchy Path: **Network Drivers**

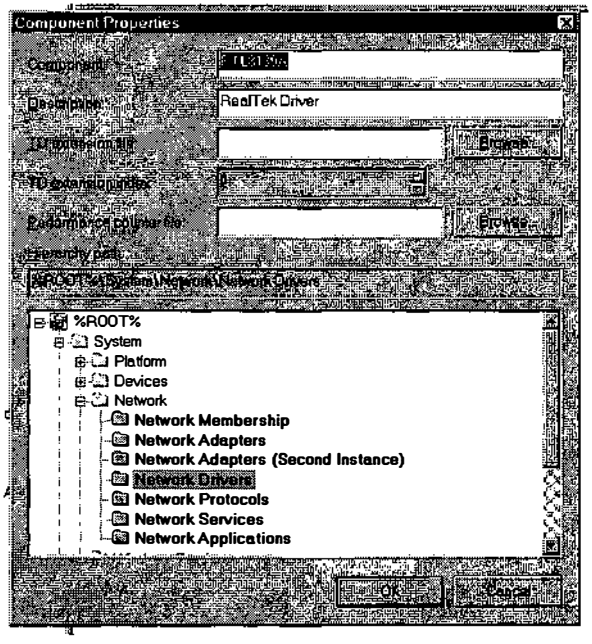


Figure 4.16: Create the Driver Component

30. Right click on Component1 (Component Root) and select properties.

    Component Name: **AAEON RealTek**
    Vendor: **RealTek via AAEON**
    Description: **8139A Driver**

31. Right click on AAEON RealTek and select Comments

    Enter: **Copyright © 1999 ACME Corp.**

32. Save the KDF as RTL8139.kdf. Make sure that you save the KDF to the same subdirectory as the HIV files (e:\local\8139).

33. Under the Registry entries for AAEON 5894 (RTL8139A), add a new key by right clicking in the right pane and selecting Add Registry Key:

Key path: HKLM\Software
Key name: Microsoft\Windows NT\CurrentVersion\NetworkCards

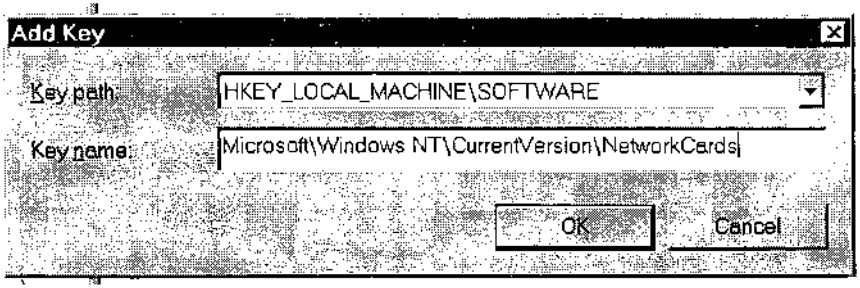


Figure 4.17: Creating the NetworkCard Hive Root

Note: There is a space between "Windows" and "NT"

34. Now we will associate the HIV file with the Registry key. Right click on HKLM\Software\Microsoft\Windows NT\CurrentVersion\NetworkCards and select Add HIV file.

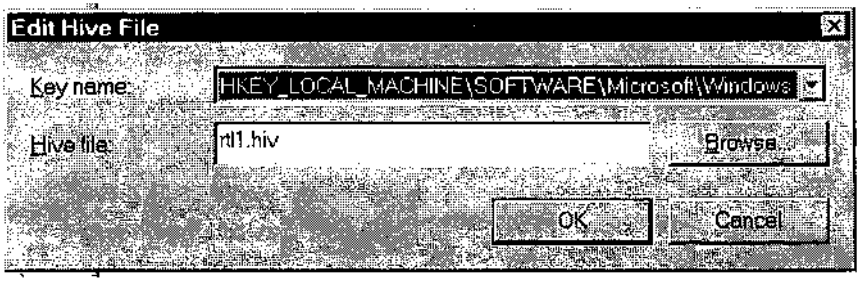


Figure 4.18: Adding the HIV File to the NetworkCard Root

35. Click Browse and select c:\local\8139\RTL1.HIV. Click OPEN.
36. Now repeat the same steps for the last adapter registry key and the two driver registry keys.

Component: AAEON 5894 (RTL8139A)

Step1: Add RegistryKey
Key path: **HKLM\System**
Key name: **ControlSet001\Services\RTL81391**

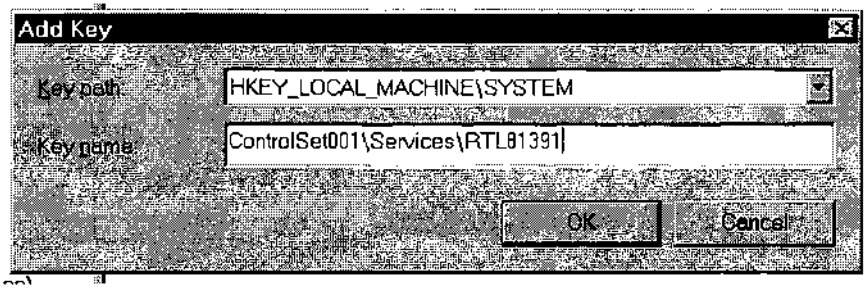


Figure 4.19: Create the RTL81391 Root

Step2: Add HIV Files: Right click on the registry key created in step one, and select ADD Hiv File

HIV File: **HKLM\System\ControlSet001\Services\RTL81391**
RTL2.HIV

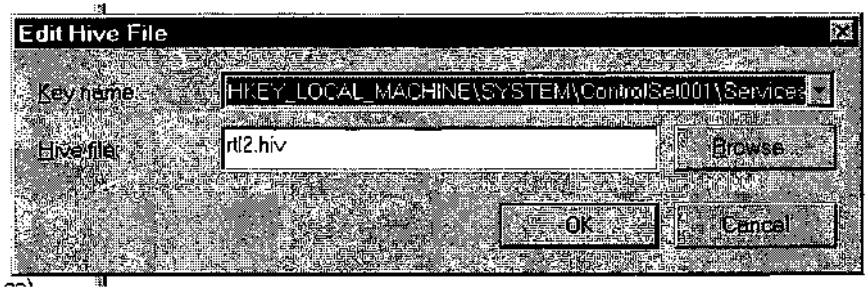


Figure 4.20: Add RTL2.HIV to the RTL81391 Root

Now repeat the steps for the other registry keys for the driver component:

Component: RTL8139A

Step1: Add Registry Key
Key path: **HKLM\Software**
Key name: **RealTek**

Step2: Add HIV File
HIV File: **HKLM\Softeware\Realtek RTL3.HIV**

Component: RTL8139A

Step1: Add Registry Key
Key path: **HKLM\System**
Key name: **ControlSet001\Services\RTL8139**

Step2: Add HIV File
HIV File: **HKLM\System\ControlSet001\Services\RTL8139 RTL4.HIV**

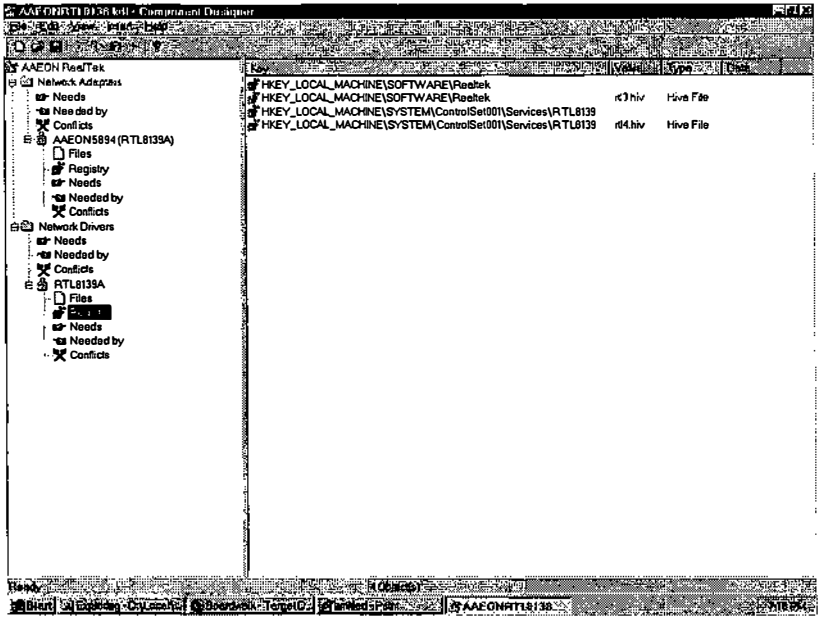


Figure 4.21: Two Registry HIV Files for the Driver Component

You should have two registry HIV files for each component.

37. In Files under RTL8139A, Drag from Explorer: RTL8139A.SYS, and RTSNTHLP.HLP.

38. Next, change the Target path for each entry. Right click and select properties for each entry:

RTL8139A.SYS
Target path: **\winnt\system32\drivers**

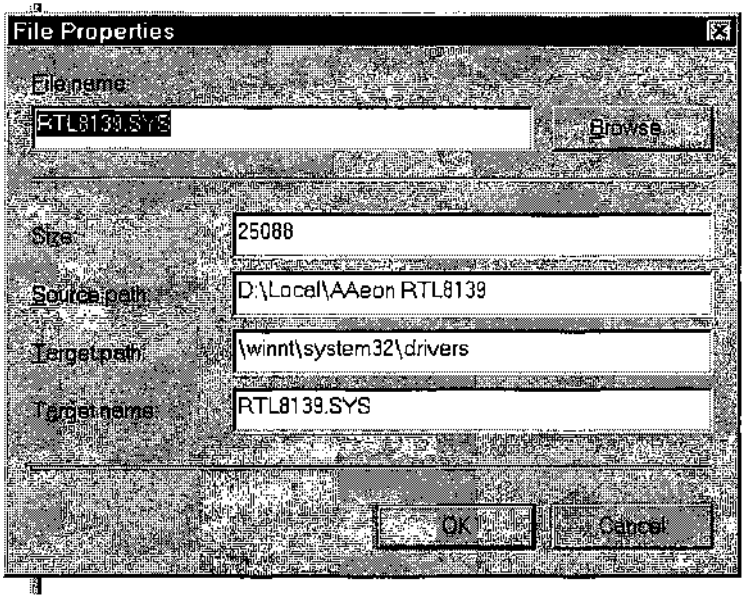


Figure 4.22: Changing the Device Driver File's Target Path

RTSNTHLP.HLP
Target path: **\winnt\system32**

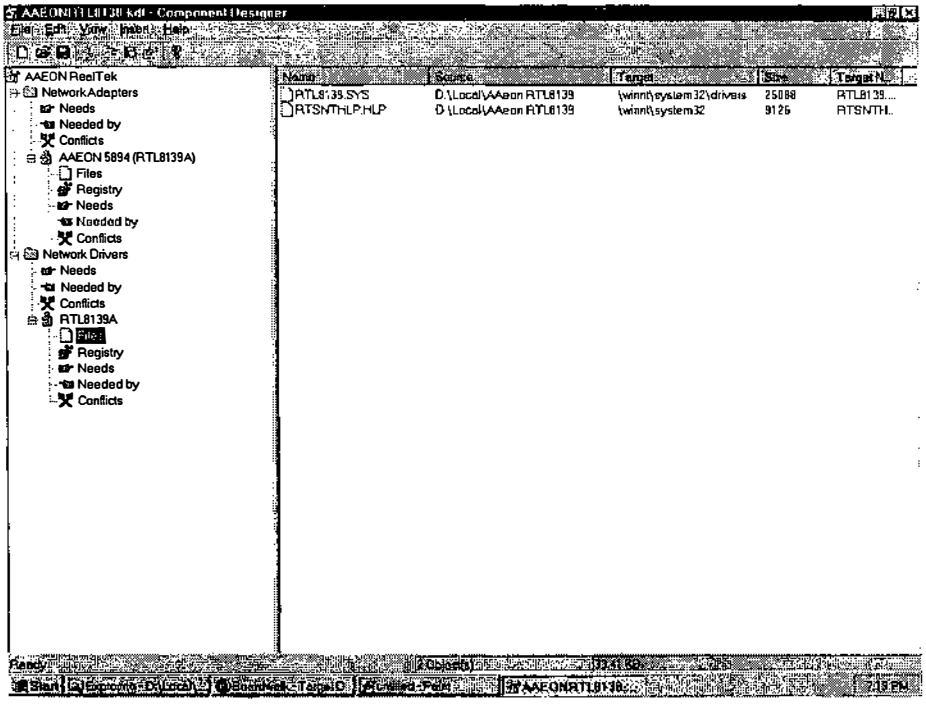


Figure 4.23: Two files: Driver and Help are Part of the Driver Component

```
;Copyright (c) ACME Corp.

[Version]

Signature = "$Windows NT$"
KdfVersion = 1


[Header]

Name = AAEON RealTek
Version = 1.0
ReleaseDate = 08/31/1999
```

```
Vendor = REalTek via AAEON
OS = WINNT
OSVersion = 4.0
Platform = i386
Repository = "%WINNT%"
Description = "8139A Ethernet Driver"
CodePage = 1252


[Capabilities]

1 = "Network Adapters"
2 = "Network Drivers"


[Components]

1 = "AAEON 5894 (RTL8139A)", "Ethernet for AAEON platform", 1
2 = "RTL8139A", "RealTek Driver", 2


[CapabilityDependencies]


[CapabilityComponentDependencies]


[ComponentCapabilityDependencies]


[ComponentDependencies]


[Directories]

2 = "D:\Local\AAeon RTL8139"
3 = "\winnt\system32\drivers"
4 = "\winnt\system32"


[AAEON 5894 (RTL8139A).Files]


[AAEON 5894 (RTL8139A).RegistryAdditions]


KeyName = "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\NetworkCards"

FileName = "rtl1.hiv"
```

```
KeyName = "HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RTL81391"

FileName = "rtl2.hiv"


[RTL8139A.Files]

"RTL8139.SYS", 2, 3, 25088, "RTL8139.SYS"
"RTSNTHLP.HLP", 2, 4, 9126, "RTSNTHLP.HLP"


[RTL8139A.RegistryAdditions]


KeyName = "HKEY_LOCAL_MACHINE\SOFTWARE\Realtek"

FileName = "rtl3.hiv"


KeyName = "HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RTL8139"

FileName = "rtl4.hiv"
```

TIP: You will want to keep the files and the KDF together so that all of the files can be checked into version control. This method will allow anyone to use the KDF without having to hunt down loose files.

TIP: If you have to recover a system from version control, make sure that you install the KDF first before importing the configuration files (.CFG). An error will appear if a component is missing when a CFG is being imported.

39. Save KDF - AAEON RTL8139.kdf.
40. In Target Designer, select Component Management under the Tools menu.
41. Click the Add... button and open the AAEON RTL8139.kdf. The KDF will be imported into the Target Designer database.

When a KDF with HIV files is added to the database, the HIV files are copied to the \program files\Windows NT Embedded 4.0\Repository\hives directory. The HIV files must exist in the same directory as the KDF for the copy to take place correctly, or a copy error will display during the import into the TDSD. When the network component is added

to the TDSD, you can use the Registry view to see if the correct registry entries are in place. You will have to go back to the KDF and possibly the target system to recapture HIV files to make any corrections. The KDF will have to be removed and re-imported into the TDSD if you make any changes to the KDF in Component Designer.

4.8.5 Build Image with Network support

42. Create a new Configuration and rename it to Exercise5
43. Select the following components

Capability	Component	Size	Check Box
System / Platform / SystemRole	Workstation System	0	
System / Platform / User Identification	Automatic Login	0	
System / Platform / Core OS	Minimal	7.87MB	
System / Platform / HAL	Standard HAL	958KB	
System / Platform / Virtual Memory	8	8MB	
System / Devices / Storage / Floppy Disk	Floppy	18KB	
System / Devices / Storage / Fixed Disk	EIDE	0	
System / Devices / Storage / Fixed Disk	SCSI Disk	15KB	
System / Devices / Storage / CD-ROM	EIDE CD-ROM	0	
System / Devices / Storage / CD-ROM	SCSI CD-ROM	22KB	
System / Devices / Storage / File Systems	FAT	139KB	
System / Devices / Display / Display Drivers	VGA	99KB	
System / Devices / Keyboard / Keyboard Layout	US Keyboard Layout	9KB	
System / Devices / Keyboard / Keyboard Drivers	PC/AT Keyboard	29KB	
System / Devices / Keyboard / Keyboard	English (US) Input	7KB	

Input Local	Locale		
System / Devices / Mouse	Logitech	29KB	
System / Devices / BUS Devices SCSI / SCSI Adapters	EIDE SCSI Adapter	0	
System / Devices / BUS Devices SCSI / SCSI Drivers	EIDE SCSI Driver	27KB	
System / Devices / Serial Ports	COM1	0	
System / Network / Network Membership	Workgroup Participation	0	
System / Network / Network Adapters	AAEON 5894 (RTL8139A)		
System / Network / Network Drivers	RTL8139A		
System / Network / Network Protocols	TCP/IP	514KB	
System / Network / Network Services	Computer Browser	0	
System / Network / Network Services	LAN Manager Server	329KB	
System / Network / Network Services	LAN Manager Workstation	322KB	
System / Network / Network Services	Messenger	38KB	
System / Network / Network Services	NetBIOS	39KB	
System / Network / Network Services	NT LM Security Support Provider	35KB	
System / Network / Network Services	RPC	1.015MB	
System / Network / Network Services	WinSock	178KB	
System / Network / Network Applications	Net Command (NET.EXE)	305KB	
System / Network / Network Applications	TCP/IP Utilities	733KB	
System / Windows Services / OLE and COM	OLE/COM	2.18MB	
System / Desktop Settings / Fonts	MS Sans Serif	151KB	

System / Desktop Settings / Application Links	Explorer Links	36KB	
System / Shared System Components	Display	25KB	
System / Shared System Components	Keyboard Drivers	9KB	
System / Shared System Components	Keyboard Locale	135KB	
System / Shared System Components	Mouse Common	9KB	
System / Shared System Components	Network Common	1.91MB	
System / Shared System Components	SCSI	58KB	
System / Shared System Components	Serial Ports	44KB	
System / Shared System Components	Virtual Memory	0	
Applications / Shell	Explorer Shell	0	
Applications / Utilities	Explorer	322KB	
Applications / Accessories	Calculator	124KB	
Applications / Accessories	Notepad	225KB	

44. When completed, Right click on the Workstation component and change the Computer Name to NTE1. **IF YOU ARE ASKED TO USE THIS NAME FOR DNS, SELECT NO.**
45. Right Click on the Workgroup Participation component, select Properties, and change the Workgroup to NTECLASS.
46. Right Click on the TCP/IP component, and select Properties. You should see the TCP/IP address that you set in the previous steps:

    IP Address: 10.1.1.250
    Subnet Mask: 255.0.0.0
    Or select the TCP/IP address that connects the target to your LAN.

If you do not see the TCP/IP address or the network adapter in the adapter drop down box, you will need to double check the adapter components registry entries.

47. Build the Exercise5 image, prepare the target's hard drive, and download the image to the target.

Tip: Notice that there is an extra step in the build process to enable/disable Network Services (i.e., Workstation, server, computer browser, etc.). If you do not have any service listed, you have an error in the registry entries of the KDF.

48. When download is completed, reboot the target.
49. You can use Explore to browse the network and locate your target computer.
50. You can go back and complete the KDF by adding the NetAdapter.TDX to the Second Adapter component properties. When you make changes to the KDF, you must remove the previous component from the TDSD, and re-import the updated KDF. To remove a component, select the component in the Component Management dialog and click the REMOVE button.

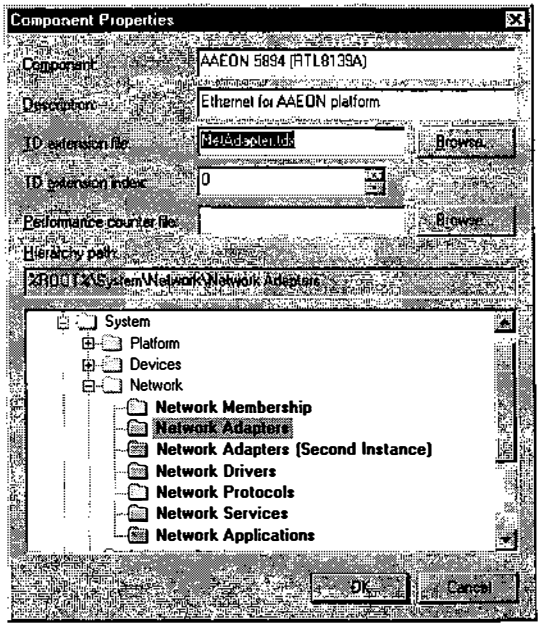


Figure 4.24: Add the NetAdapter TDX to the Adapters Component Properties.

4.9 Understanding the Registry

The registry contains a large amount of information, and it can be difficult to decipher which hives are needed for a KDF. Understanding the registry is key to developing KDFs. The registry was designed for security, multiple users, ease of programmability, and as a single source for platform configuration. Applications and drivers use the registry to store and retrieve configuration settings and linkages with other resources.

4.9.1 Registry Hierarchy

The registry is divided into several files found in the \winnt\system32\config directory. The registry can be viewed through Control Panel applets or registry editors. Regedt32 presents the registry as five root keys. Below is a table describing each of the root keys:

Root Key	Description
HKEY_CLASSES_ROOT	Provides backward compatibility for Windows 3.x applications for OLE and DDE support. The key points to the HKEY_LOCAL_MACHINE\SOFTWARE\Classes subkey, which has the same information.
HKEY_CURRENT_USER	Points to the HKEY_USERS key for the current logged in user profile.
HKEY_LOCAL_MACHINE	Contains the computer-specific information for applications and drivers. The information in this root key is independent of who is logged in.
HKEY_USERS	Contains the user profiles for each user listed in the system. The main two subkeys are .DEFAULT and SID. The SID subkey is unique for each user that logs into the system. When cloning a computer, the SIDs get modified to prevent security problems when more than one device is attached to a network.
HKEY_CURRENT_CONFIG	Points to the HKEY_LOCAL_MACHINE\SYSTEM subkey. Provides easy access for application developers to system information.

By process of elimination you can see that KDFs only require the following keys:

```
HKEY_LOCAL_MACHINE\SOFTWARE
HKEY_LOCAL_MACHINE\SAM
HKEY_LOCAL_MACHINE\SYSTEM
HKEY_LOCAL_MACHINE\SECURITY
HKEY_USERS\.DEFAULT
```

You can visually see the above list by looking at the Key Path drop down list when inserting a Registry Key item in Component Designer. There is more information about each sub key and hives, but a deeper discussion of the Windows NT registry would require another book. For more information please review the following book:

Windows NT Registry Troubleshooting, Rob Tidrow, New Riders, 1996.

4.9.2 Registry Patterns

Exercise 5 gave you the registry keys needed to build the KDF. The registry keys were located by following the pattern of similar network card and adapter components already in

Target Designer. As you get familiar with the registry, you can see that families of driver registry keys tend to follow a particular pattern. The patterns can be seen by transversing the various driver components within Target Designer.

The following is a general listing of registry hives for each driver family. Although each listing below contains most of the registry hives required, some drivers may have all of the hives, some drivers will have more, and some will have less. Once you get familiar with the registry, you can see the linkages between hives and guess where to locate the correct hives for your driver and/or application.

VGA Adapters / Drivers

The Display capabilities are divided into two components: adapter and driver. The adapter component contains nothing but a dependency for the driver component. So if your adapter matches a VGA adapter component, you can select from the list, and the correct driver component will be pulled in. The driver component contains all of the files and registry settings. The following are the typical registry keys:

Driver	Keys
VGA Drivers	HKLM\Software\Microsoft\Windows NT\CurrentVersion\OpenGLDrivers
	HKLM\System\ControlSet001\Services\<VGA>
	HKLM\System\ControlSet001\EventLog\System\<VGA>
	HKLM\System\ControlSet001\Hardware\0001\System\CurrentControlSet\Services\<VGA>
	HKLM\System\ControlSet001\ENUM\ROOT\LEGACY_<VGA>

Network Adapters / Drivers

Like the display, the network capability has the hardware divided between the adapter and the driver, but in this instance the adapter plays an important role. The adapter component contains the registry entries that hold the TCP/IP address information and other linkage information. The driver registry entries contain the hardware settings such as IRQ, bus type, and device number.

Driver	Keys
Network Adapter	HKLM\Software\Microsoft\Windows NT\CurrentVersion\NetworkCards\1
	HKLM\System\controlSet001\Services\<driver>1
Network Driver	HKLM\Software\<Manufacturer>\<driver>
	HKLM\System\ControlSet001\Services\<driver>
	HKLM\System\ControlSet001\Services\EventLog\System\<driver>
	HKLM\System\ControlSet001\ENUM\ROOT\LEGACY_<driver>

Modem Cards

Modems are listed as a single component. Most of today's modems follow the unimodem standard. Instead of a driver, an INF is listed in the files for the component. The registry entries provide information for connection speeds and hardware settings.

Driver	Keys
Modems	HKLM\Software\Microsoft\Windows\currentVersion\Unimodem\devicespecific\<modem>
	HKLM\System\ControlSet001\Services\<modem>
	HKLM\System\ControlSet001\ENUM\ROOT\<modem>
	HKLM\System\ControlSet001\ENUM\ROOT\LEGACY_<modem>
	HKLM\System\ControlSet001\Class\{4D36E96D-E325-11CE-BFC1-08002BE10318}

SCSI Cards

SCSI Cards are divided between adapter and driver. Most Adapter components contain only a dependancy on the driver component.

Driver	Keys
SCSI	HKLM\System\ControlSet001\Enum\ROOT\SCSIADAPTER
	HKLM\System\ControlSet001\Services\<driver>

4.9.3 Sample KDFs

Of course samples are a good guide for locating the correct registry values. Microsoft includes a set of sample KDFs on NTE CD-1.

Sample KDF	Capability	Description
Ati	Display	ATI Technologies Graphics Adapter KIT
Chips	Display	Chips Display Driver KIT
Matrox	Display	Matrox Millennium KIT
Cpqnf3	Network	Compaq NetFlex-3 NIC
E100bnt	Network	Intel EtherExpress PRO Adapter
Isdn	Network	DigiBri Isdn card
Buslogic	SCSI	BUS Logic SCSI adapter component Kit
Alerter	Services	Alerter
SAP	Services	SAP Agent
Schedule	Services	Scheduler Service

The companion disk provides some sample KDFs for common devices like video, network, SCSI, and sound. The KDFs are for the latest hardware available on the market, but since hardware changes quickly in the desktop market, the exact hardware may not exist. You will need the drivers that go with each KDF in order to fully use the KDF.

Sample KDF	Capability	Description
3COM US Robotics	Modem	3COM 56K Winmodem PCI Model# 5699
AAEON RTL8139	Network	RealTek 8139A Network Driver version 3.2
Diamond SupraModem	Modem	Diamond SupraMax 56K Modem
Intel	Network	Intel PRO/100+ Management Adapter
LNE100TX	Network	Linksys Etherfast 10/100 LAN Card Model: LNE100TX version 2.0
MediaGX	Display	Media GX Video Driver for Unicorn Platform
MediaPlayer	Accessories	Microsoft Media Player
Sbaudio	Sound	Sound Blaster 16 PCI Model: CT4740

4.10 Exercise 6 - Developing the "Hello World" TDX

4.10.1 TDX Overview

KDFs create capabilities and components within the TDSD. Target Designer Extensions (TDXs) manipulate the registry data within the TDSD. Section 4.6 introduced TDXs and the TDX SDK, which was installed with Target Designer and Component Designer. TDXs are MFC DLLs that can be written to add, delete, and edit registry values and keys. The TDX SDK supports Visual C++ 6.0.

When a component with properties support is added to a configuration, the properties option becomes visible. Target Designer makes calls to a component's TDX through several entry points in several operations:

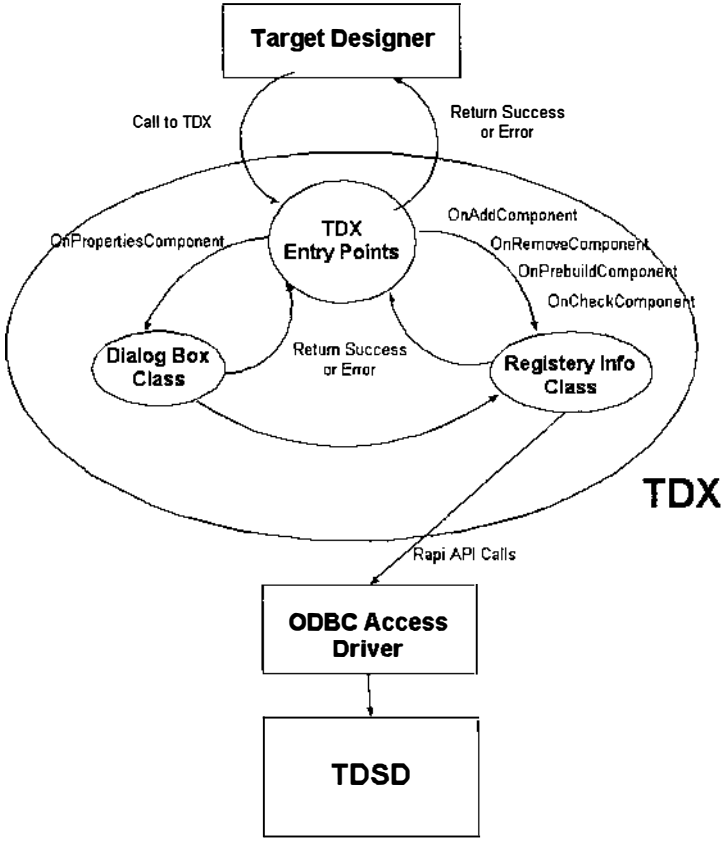
TDX Entry Point	Description
OnPropertiesComponent	Called when the component is configured
OnAddComponent	Called when the component is added to the configuration
OnRemoveComponent	Called when the component is removed from the configuration
OnPrebuildComponent	Called before the Build operation combines the registry
OnCheckDepedencyComponent	Called when the check component dependencies is started by the user

Each entry point can cause certain actions to take place. The most used entry point, OnPropertiesComponent, is used to launch the dialog interface to the user. The TDX's functionality is left up to the designer. The TDRAPI.LIB contains several registry calls that are used to access the registry within the TDSD. These calls are similar to the REG API calls for ease of programmability (the W implies Unicode). The Reg key information can be found in the Microsoft Developers Network Library, and the prototypes are found in the TDXFACE.H:

Rapi API Call	Equivalent Reg API Call	Description
RapiOpenKeyW	RegOpenKey	Opens the specified Key
RapiOpenKeyExW	RegOpenKeyEx	Opens the specified Key. (Win32-based applications)
RapiCreateKeyW	RegCreateKey	Creates specified key. If key already exists, the function opens the key.
RapiCreateKeyExW	RegCreateKeyEx	Creates specified key. If key already exists, the function opens the key. (Win32-based applications)
RapiQueryValueW	RegQueryValue	Retrieves the data of a specified registry key. The data must be a null-terminated string.
RapiQueryValueExW	RegQueryValueEx	Retrieves the data of a specified registry key. The data must be a null-terminated string. (Win32-based applications)
RapiSetValueW	RegSetValue	Sets the data for the specified registry key. The data must be a text string.
RapiSetValueExW	RegSetValueEx	Sets the data and type of a specified registry key. The data must be a text string. (Win32-based applications)
RapiEnumKeyW	RegEnumKey	Enumerates subkeys of the specified open registry key.
RapiEnumKeyExW	RegEnumKeyEx	Enumerates subkeys of the specified open registry key. Also retrieves the class name of the subkey and the last time the subkey was modified. (Win32-based applications)
RapiEnumValueW	RegEnumValue	Enumerates values for a specified open registry key.
RapiDeleteValueW	RegDeleteValue	Deletes a subkey
RapiDeleteKeyW	RegDeleteKey	Removes a named value from the specified registry key

4.10.2 SAMPLE.TDX

There is a sample TDX cleverly called "Sample" in the TDX SDK directory. There is also a SAMPLE.KDF that comes with the Sample TDX. When the component is added to the configuration (OnAddComponent), a registry subkey is created with a default value. The new registry key can only be seen from the top of the configuration. The key has the blue plus sign, which implies the key was added within Target Designer. You can change the properties via the dialog interface (OnPropertiesComponent), and the change will be reflected after OK is clicked. The subkey is removed when the component is removed from the configuration (OnRemoveComponent). A registry-information MFC class is created to manage the editing of the registry value as well as the creation and removal of the subkey.



Any changes to the key are reflected only within the scope of the current configuration. If you open the TDSD in MS Access, the configuration table lists a Configuration ID associated with each configuration. When the TDX is called, a configuration ID is passed to the TDX. The registry info class uses the configuration ID to make changes only within the current configuration scope.

4.10.3 Exercise 6 Overview

Exercise 6 is a variant of the Sample TDX; the "Hello World" TDX. The lesson is to walk through the steps required to set up, create, and implement a TDX. You will need Visual Studio to complete this exercise.

A sample component (KDF) with a single test registry key and value will be used to test the TDX and demonstrate how and when the TDX's entry points are called. When the sample component is added to or removed from the configuration, a message box will pop up with a message. A Dialog box will be used to change the known registry key's value.

4.10.4 Create the New Project

1. Open Visual C++, and go to Tools->Options.
2. Click on the Directories tab, and add the path to the TDXFACE.INC: \Program Files\Windows NT Embedded 4.0\SDK\INC
3. Change the Show directories to Library Files, and add the path to the TDRAPI.LIB: \Program Files\Windows NT Embedded 4.0\SDK\LIB
4. File->New, and select Projects tab is selected.
5. Click on MFCAppWizard(dll), title the project **Test**.

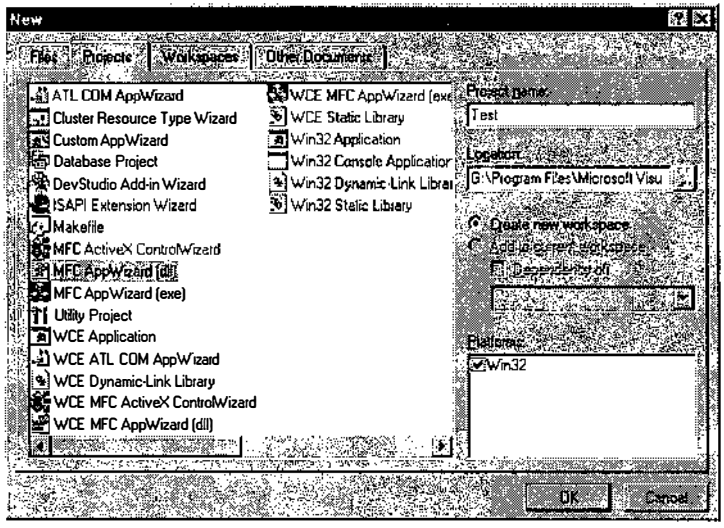


Figure 4.25: Creating a New MFC DLL

6. The MFC AppWizard Step 1 of 1 pops up, so we select shared MFC DLL. Click Finish.

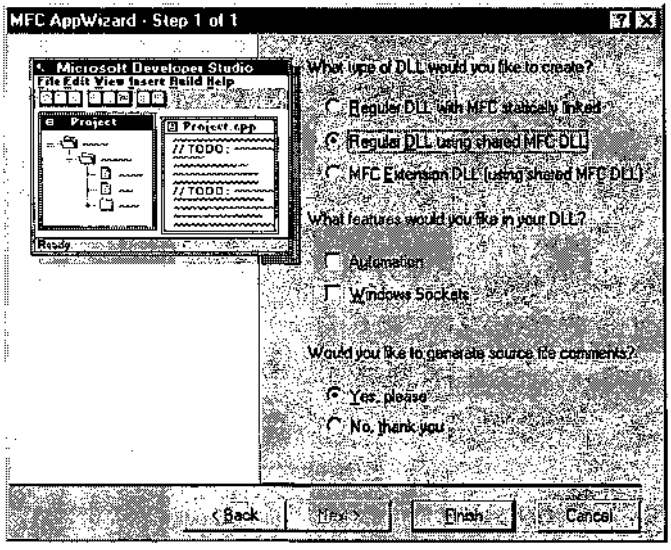


Figure 4.26: MFC AppWizard

7. The summary dialog pops up. Click OK.

4.10.5 Project Setup

Change to Output a TDX. The project needs to be set up to support Unicode and build a TDX.

8. Project->Settings
9. Goto the C/C++ tab.
10. Under the General Category, Change the Preprocessor definitions from:

    WIN32,_DEBUG,_WINDOWS,_WINDLL,_AFXDLL,_MBCS,_USRDLL

    To:

    WIN32,_DEBUG,_WINDOWS,_WINDLL,_AFXDLL,_UNICODE,UNICODE,_AFXEXT

11. Select the Link tab.
12. Under the General Category for both Debug and Release versions, change the output file to Test.TDX.
13. In the Objects/library modules box for both Release and Debug, add **tdrapi.lib** for both Debug and Release. The TDRAPI.LIB contains the RAPI API calls to the data base.
14. Select the Input category, for both Debug and Release add **odbccp32.lib** in the Ignore libraries.

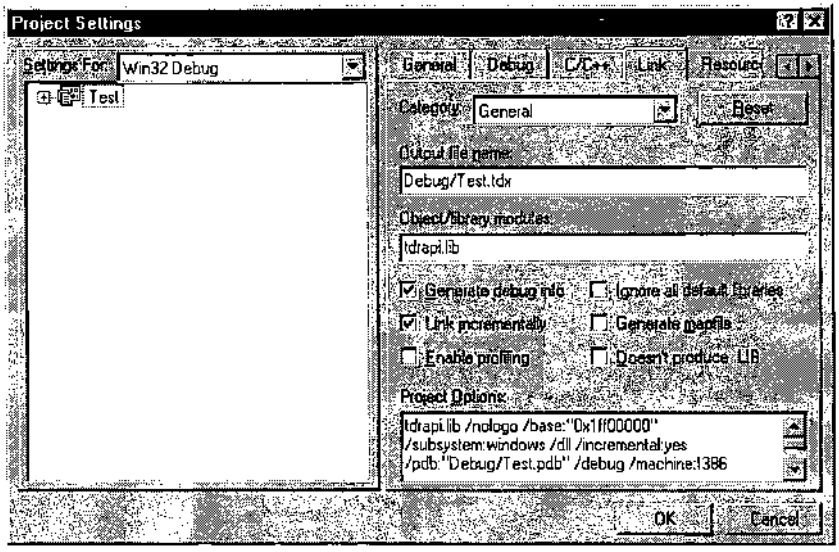


Figure 4.27: Project Settings Dialog

15. Select the Output category for both Debug and Release under Base address, and add the following: **0x1ff00000**. The re-basing of the DLL helps to avoid collisions when multiple TDXs are loading. Multiple TDX loading is a possibility during a build or a dependency check.
16. Click OK to close Project Settings dialog.

4.10.6 Create a Registry Information Class

The Registry Information class is the work horse of the TDX. Here you have created the member functions and variables that will access the test registry key.

17. Go to Insert-> NewClass
18. Select the ClassType: **Generic Class**
19. The class name is **CTestRegInfo**

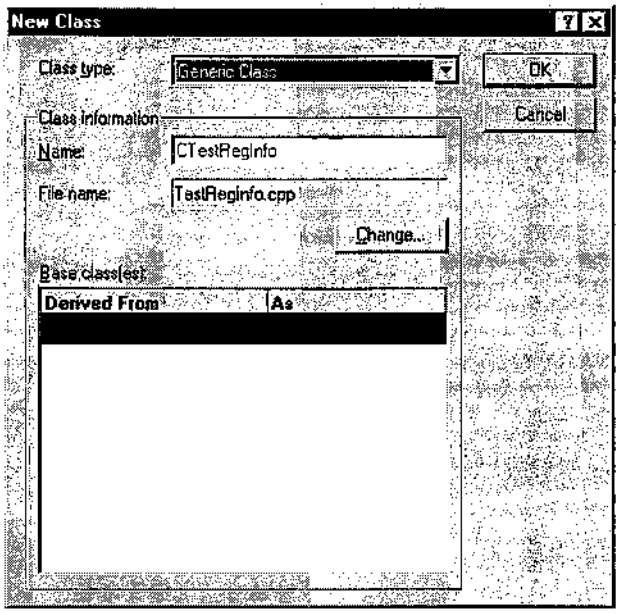


Figure 4.28: Adding the Registry Information Class

20. Click OK
21. Go to File->Save Workspace
22. Go to File->Save All
23. Open TestRegInfo.h and add LONG lConfigID to the CTestRegInfo declaration.

```
class CTestRegInfo
{
public:
        CTestRegInfo(LONG lConfigID);
        virtual ~CTestRegInfo();

};
```

24. Open TestRegInfo.cpp and LONG lConfigID to the CTestRegInfo definition.

```
////////////////////////////////
// Construction/Destruction
////////////////////////////////

CTestRegInfo::CTestRegInfo(LONG lConfigID)
{

}

CTestRegInfo::~CTestRegInfo()
{

}
```

25. Save All
26. In ClassView, right click on CTestRegInfo class, and select Add Member Function.
27. In the Dialog box fill in the following:

    Function type: **BOOL**
    Function declaration: **GetTestValue(CString &)**
    Access: **Public**

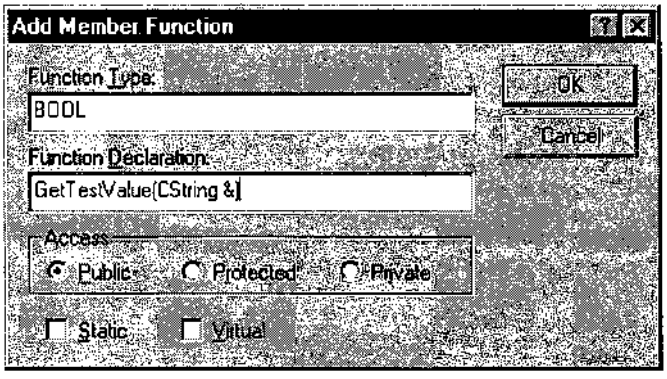


Figure 4.29: GetTestValue Member Function

This function will retrieve the registry key value.

28. Click OK
29. Add another Member Function:

    Function type: **BOOL**
    Function declaration: **SetTestValue(const CString &)**
    Access: **Public**

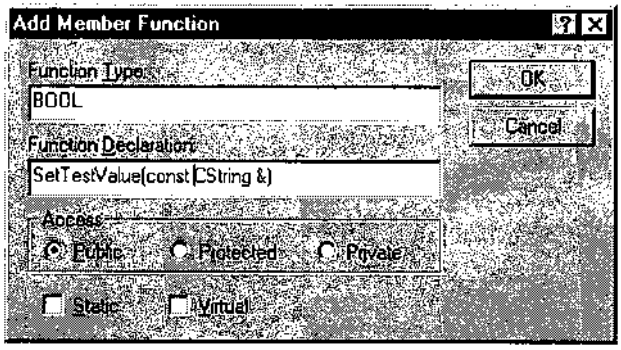


Figure 4.30: SetTestValue Member Function

This function will set the registry value.

30. Click OK
31. Add another Member Function:

    Function type: **BOOL**
    Function declaration: **Initialize**
    Access: **Public**

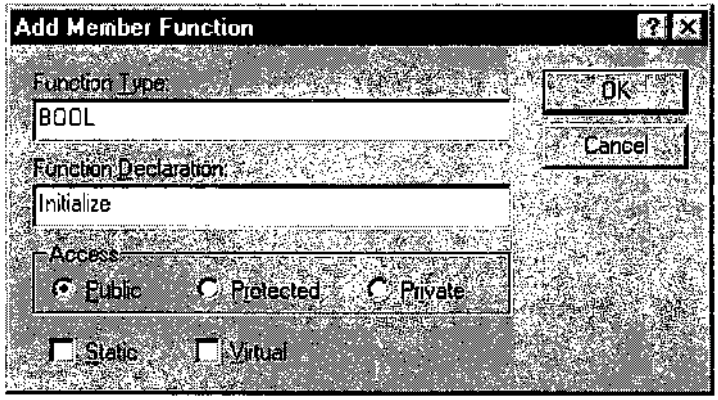


Figure 4.31: Initialize Member Function

32. Click OK

33. Right click on CTestRegInfo class, and select Add Member Variable.

Variable Type: **LONG**
Variable Name: **m_lConfigID**
Access: **Private**

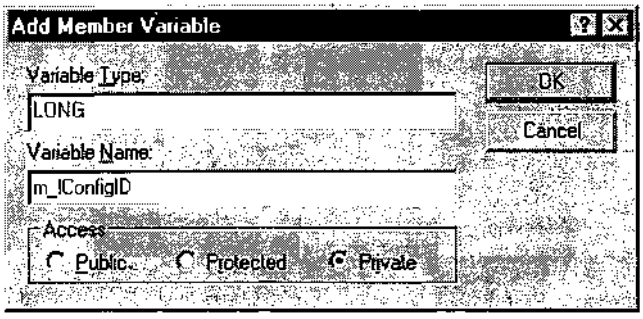


Figure 4.32: Add the Configuration ID Member Variable

34. Click OK.
35. Right click on CTestRegInfo class, and select Add Member Variable.

Variable Type: **CString**
Variable Name: **m_strTestValue**
Access: **Private**

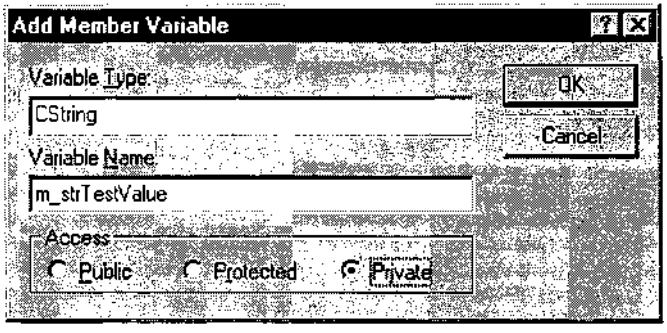


Figure 4.33: Add the TestValue Member Variable

36. Click OK

37. Right click on CTestRegInfo class, and select Add Member Variable

    Variable Type:**HKEY**
    Variable Name: **m_hTestKey**
    Access: **Private**

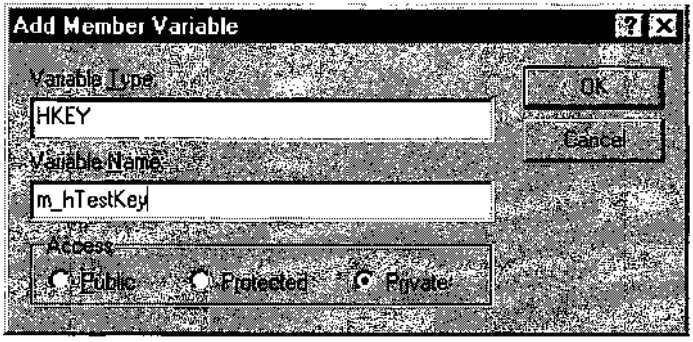


Figure 4.34: Add the TestKey Member Variable

38. Click OK

39. Open TestRegInfo.cpp and Add the following include files in the declarations section:

```
#include "tdxface.h"
#include "definitions.h"
```

40. Now add the following code block for each Member Function added by the previous step.

```
////////////////////////////////
// Construction/Destruction
////////////////////////////////

CTestRegInfo::CTestRegInfo(LONG lConfigID)
{
        m_hTestKey = NULL;
        m_strTestValue = _T("");
        m_lConfigID = lConfigID;

}
```

```
CTestRegInfo::~CTestRegInfo()
{
        if (m_hTestKey != NULL)
                RapiCloseKey(m_hTestKey);

}


BOOL CTestRegInfo::Initialize()
{

        HKEY hKey;
        LONG lRetVal = RapiOpenKeyEx(m_lConfigID,
                                     HKEY_LOCAL_MACHINE,
                                     szTestKey,
                                     NULL,
                                     KEY_ALL_ACCESS,
                                     &hKey);

        if (lRetVal != ERROR_SUCCESS)
        {
                return FALSE;
        }

        m_hTestKey = hKey;
        return TRUE;

}


BOOL CTestRegInfo::SetTestValue(const CString &strTestValue)
{

        ASSERT(m_hTestKey != NULL);

        DWORD  dwSize = (strTestValue.GetLength() + 1) *
sizeof(TCHAR);
        LONG lRet = RapiSetValueEx(m_hTestKey,
                                   szTestValue,
                                   NULL,
                                   REG_SZ,
                                   (LPBYTE)(LPCTSTR)strTestValue,
                                   dwSize);

        return (lRet == ERROR_SUCCESS);


}

BOOL CTestRegInfo::GetTestValue(CString &strTestValue)
```

```
{

        //the key must be initialized already
        ASSERT(m_hTestKey != NULL);

        LONG lRet;
        DWORD dwSize = RAPI_MAX_DATA_SIZE;
        TCHAR buffer[RAPI_MAX_DATA_SIZE];

        lRet = RapiQueryValueEx(m_hTestKey,
                                szTestValue,
                                NULL,
                                NULL,
                                (LPBYTE) (LPCTSTR) buffer,
                                &dwSize);

        if (ERROR_SUCCESS == lRet)
        {
                m_strTestValue = (LPCTSTR)buffer;
        }
        else
        {
                m_strTestValue = szDefaultTestValue;
        }

        strTestValue = m_strTestValue;
        return TRUE;


}
```

41. Save All.

4.10.7 Create the Properties Dialog

The dialog box provides a convenient interface to make registry changes. In this next section, you will create the dialog to change the value of one test registry key. The various functions in the dialog will call the Registry Information class to perform the Rapi function calls.

The On Property Component entry point will call the dialog class.

42. Now let's add the Form. Insert->New Form
43. Type in the name of the form: CTestTDXDialog. Click OK.

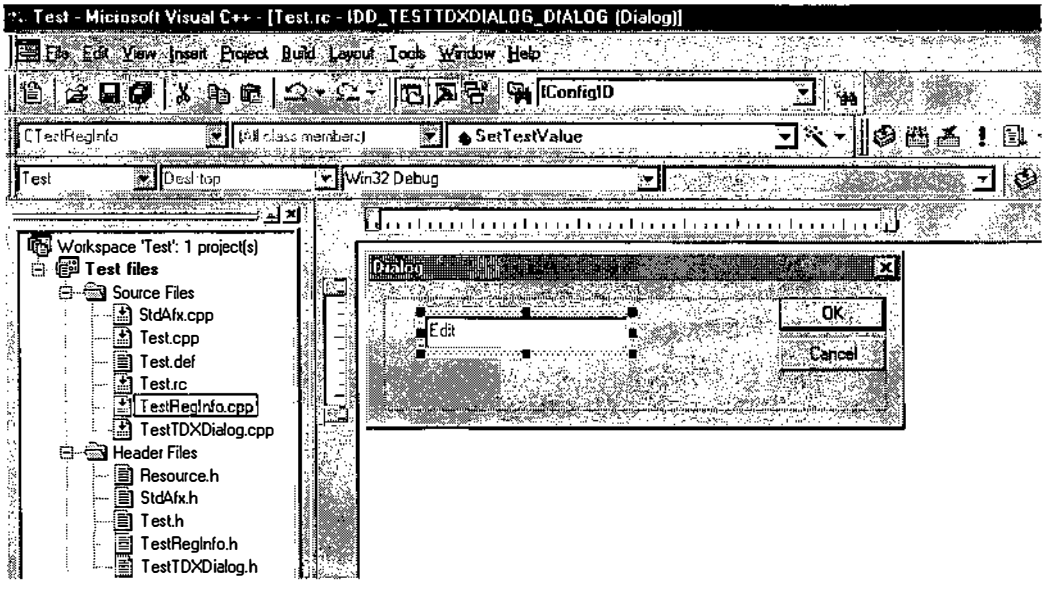


Figure 4.35: New Properties TestTDXDialog

44. Resize the form, and add an edit box.
45. Change the caption of the edit box to MY TDX.
46. Go to View->Class Wizard.
47. In Message Maps Tab, make sure that CTestTDXDialog is the selected class name, in the ObjectIDs highlight CTestTDXDialog, and double click on WM_CREATE in the Messages box to add the OnCreate member function.

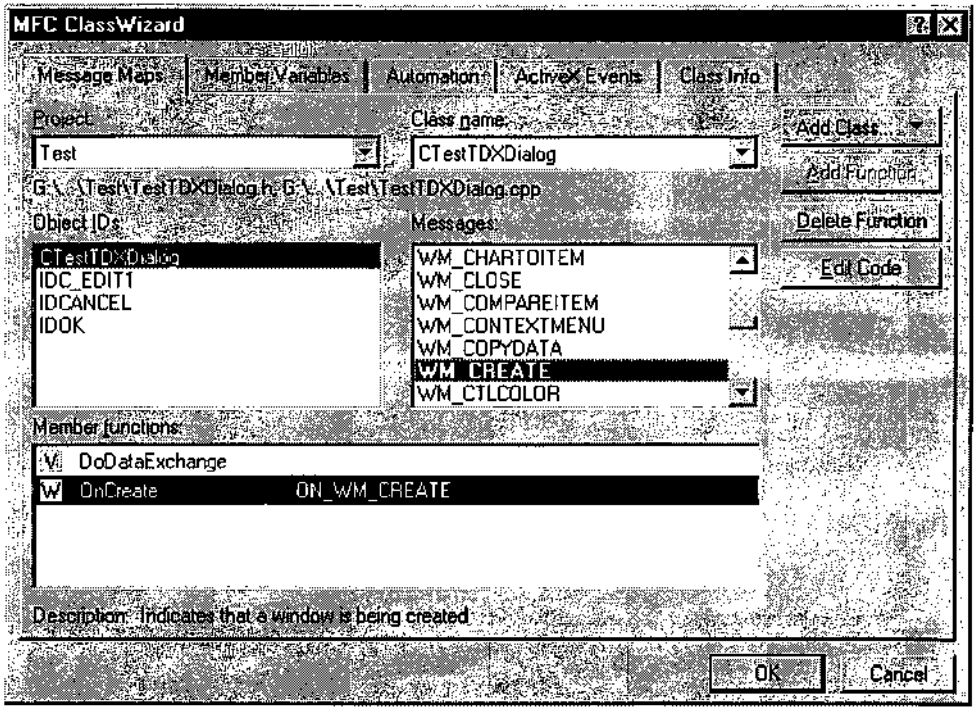


Figure 4.36: MFC Class Wizard

48. In Message Maps Tab, make sure that CTestTDXDialog is the selected class name, in the ObjectIDs highlight CTestTDXDialog, and double click on WM_INITDIALOG in the Messages box to add the OnInitDialog member function.

49. In Message Maps Tab, make sure that CTestTDXDialog is the selected class name, in the ObjectIDs highlight IDOK, and double click on BN_CLICKED in the Messages box to add the OnOK member function.

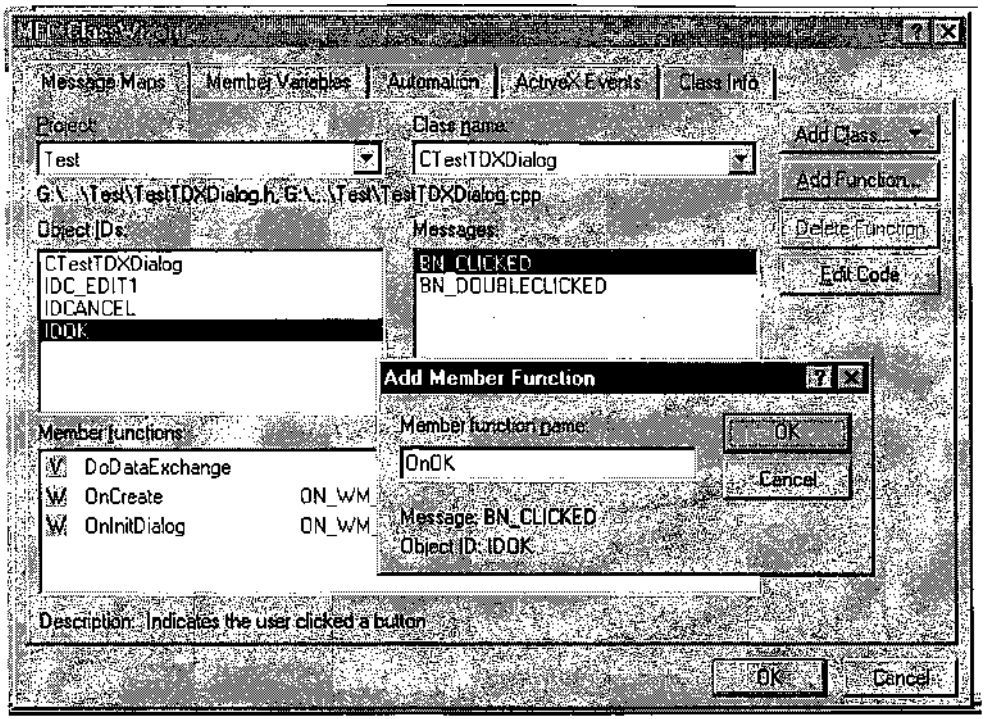


Figure 4.37: Adding the OnOK Member Function

50. In the Member Variables tab add m_strEdit to the IDC_EDIT1 (Edit box).

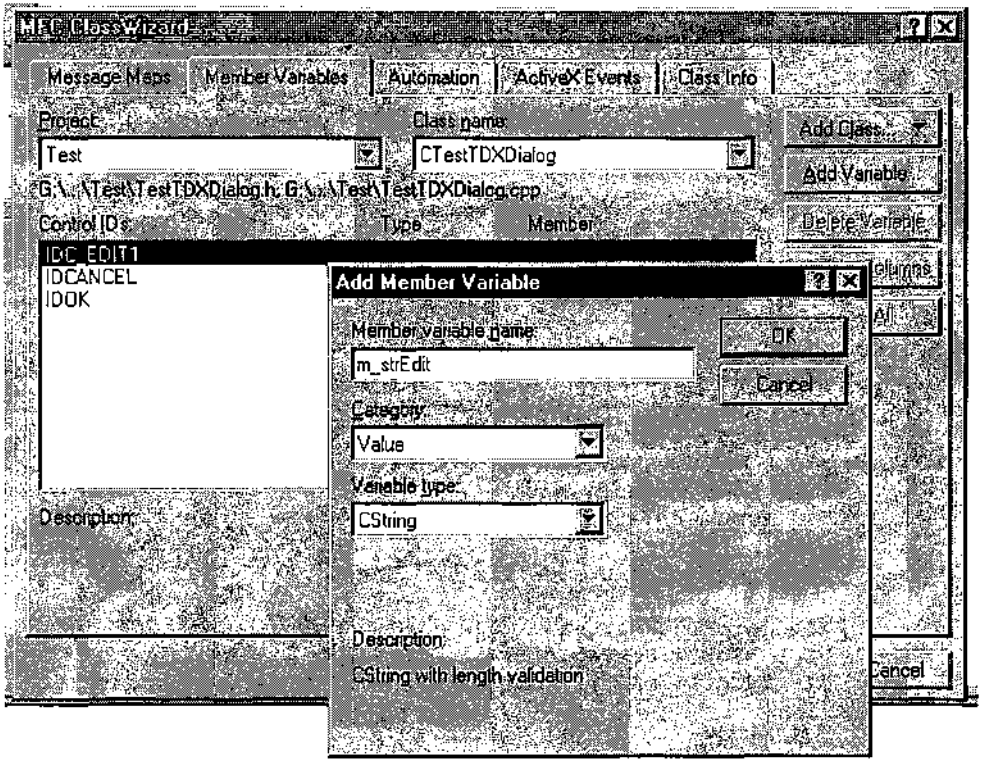


Figure 4.38: Adding the Edit Box Member Variable

51. Close the MFC Class Wizard and Click OK.
52. Open TestTDXDialog.CPP, and add the code to the new member functions as shown below.

```
////////////////////////////////////
// CTestTDXDialog message handlers

BOOL CTestTDXDialog::OnInitDialog()
{
	CDialog::OnInitDialog();

	// TODO: Add extra initialization here
	if (FALSE == GetRegistry()) AfxMessageBox(_T("Can not
initialize TDX"));
	UpdateData(FALSE);
```

```
        return TRUE;  // return TRUE unless you set the focus to a
control
                      // EXCEPTION: OCX Property Pages should
return FALSE
}

int CTestTDXDialog::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
        if (CDialog::OnCreate(lpCreateStruct) == -1)
                return -1;

        // TODO: Add your specialized creation code here
        if (m_regInfo.Initialize() == FALSE) return -1;

        return 0;
}

void CTestTDXDialog::OnOK()
{
        // TODO: Add extra validation here

        CDialog::OnOK();
        BOOL bRet = SetRegistry();
        if (TRUE != bRet) AfxMessageBox(_T("Cannot save data"));


}
```

53. Save All.
54. In the ClassView, right click on CTestTDXDialog, and select Add Member Variable.

Variable Type: **CTestRegInfo**
Variable Name: **m_regInfo**
Access: **Private**

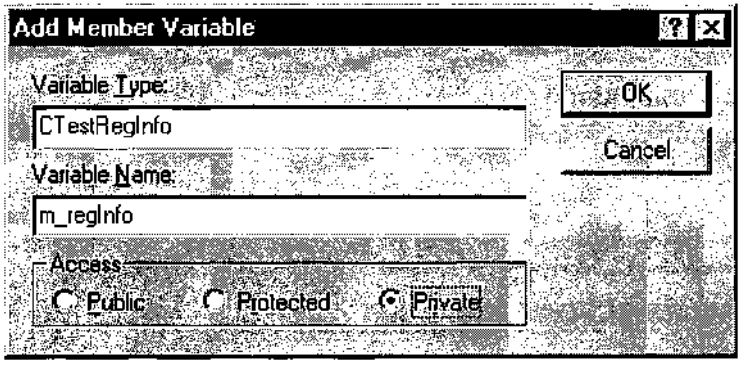


Figure 4.39: Create a Variable with the CTestRegInfo Type

55. In the ClassView, right click on CTestTDXDialog, and select Add Member Function.

Function Type: **BOOL**
Function Declaration: **SetRegistry**
Access: **Public**

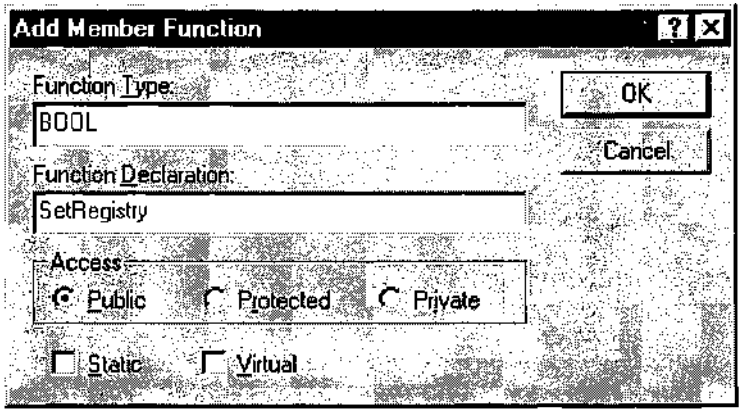


Figure 4.40: Add the SetRegistry Function

56. In the ClassView, right click on CTestTDXDialog, and select Add Member Function.

    Function Type: **BOOL**
    Function Declaration: **GetRegistry**
    Access: **Public**

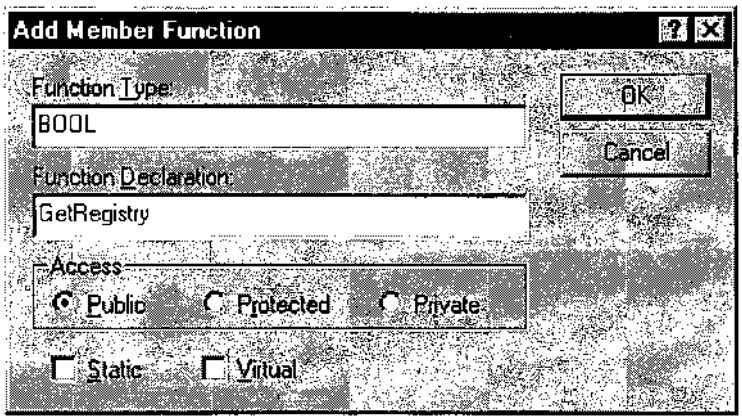


Figure 4.41: Add the Get Registry Function

57. Write the block for the two new member function shells generated by Visual Studio as shown below:

```
BOOL CTestTDXDialog::SetRegistry()
{

        return m_regInfo.SetTestValue(m_strEdit);


}

BOOL CTestTDXDialog::GetRegistry()
{

        CString strValue;
        if(m_regInfo.GetTestValue(strValue) == FALSE) return FALSE;

        m_strEdit = strValue;
        return TRUE;


}
```

58. Save All.

59. At the top of TestTDXDialog.cpp, edit the CTestTDXDialog definition to include the lConfigID, and pass the lConfigID to the m_regInfo class as shown below.

```
//////////////////////////////////
// CTestTDXDialog dialog


CTestTDXDialog::CTestTDXDialog(LONG lConfigID, CWnd* pParent
/*=NULL*/)
        : CDialog(CTestTDXDialog::IDD, pParent),
m_regInfo(lConfigID)
```

60. Open TestTDXDialog.h to add the lConfigID to the CTestTDXDialog declaration.

```
//////////////////////////////////
// CTestTDXDialog dialog

class CTestTDXDialog : public CDialog
{
// Construction
public:
        BOOL GetRegistry();
        BOOL SetRegistry();
        CTestTDXDialog(LONG lConfigID, CWnd* pParent = NULL);    //
standard constructor
```

61. Select Save All from the tool bar.

4.10.8 Add the Test Key Definition

Now let's define the test registry key and default value.

62. The test registry key and default value will be hard coded within the TDX, File->New, select C/C++ Header File, and type the name **Definitions.**

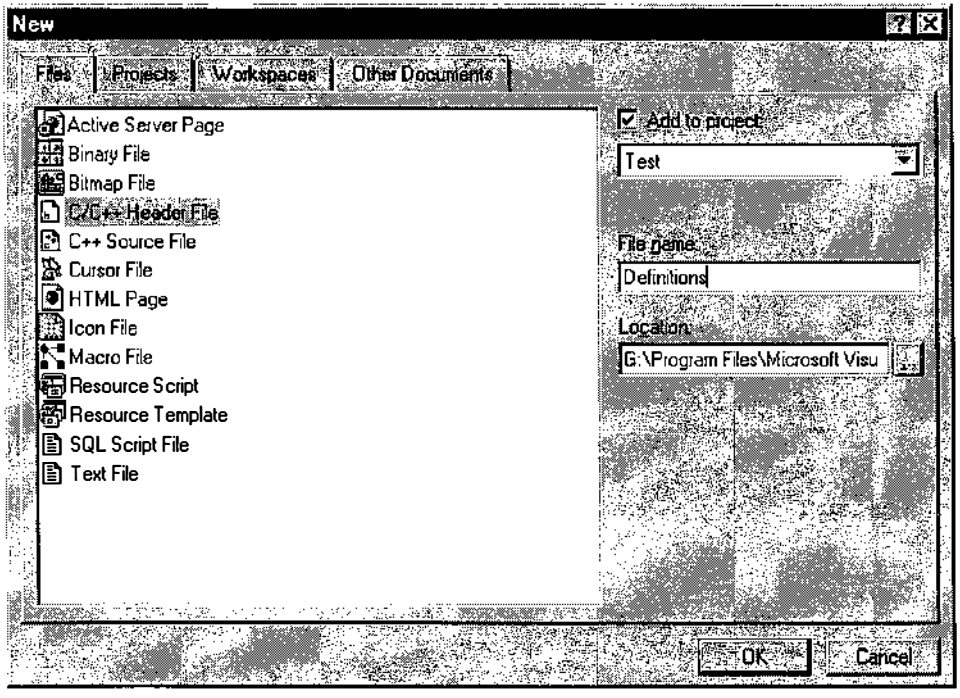


Figure 4.42: Adding a New Header File

63. Add the following to the Definitions.h file:

```
static const TCHAR szTestKey[]        = _T("Software\\TestTDX\\TestTDXKey");
static const TCHAR szTestValue[]      = _T("TestValue");
static const TCHAR szDefaultTestValue[] = _T("Hello World");
```

64. Select Save All from the tool bar.

4.10.9 Add the Entry Points

Finally, create the main entry points to the TDX.

65. Open Test.CPP, and Add the following include to the declarations section:

```
#include "tdxface.h"
#include "TestTDXDialog.h"
```

66. Add the following code after the App

```
////////////////////////////////////
// The one and only CTestApp object

CTestApp theApp;



DWORD
APIENTRY
OnPropertiesComponent(
       LONG lConfigID,            // id of the active config
       LPCWSTR szCapabilityName,  // address of name of selected
capability
       LPCWSTR szComponentName,   // address of name of selected
component
       UINT nAppNum,              // index of the activated tdx
       LPCWSTR szHelpFilePath     // pointer to the full path name
of the help file
       )
/*++
Function Description:
       Runs a custom tdx applet
Return code:
       TDX_SUCCESS - the applet was run successfully
       TDX_FAILED - the applet was not run successfully
       TDX_FAILED_NONOTIFY - the applet was not run successfully
--*/
{
       UNREFERENCED_PARAMETER(szCapabilityName);
       UNREFERENCED_PARAMETER(szComponentName);
       UNREFERENCED_PARAMETER(nAppNum);
       UNREFERENCED_PARAMETER(szHelpFilePath);

       AFX_MANAGE_STATE(AfxGetStaticModuleState());
    CTestTDXDialog dlg(lConfigID, AfxGetMainWnd());
    dlg.DoModal();
       return TDX_SUCCESS;
}


DWORD
APIENTRY
OnAddComponent(
       LONG lConfigID,            // id of the active config
       LPCWSTR szCapabilityName,  // address of name of selected
capability
       LPCWSTR szComponentName,   // address of name of selected
component
       UINT nAppNum               // index of the activated tdx
```

```
        )
/*++
Function Description:
        Performs the operations needed when the component is added
to the configuration.

Return code:
        TDX_SUCCESS - the applet was run successfully
        TDX_FAILED - the applet was not run successfully
        TDX_FAILED_NONOTIFY - the applet was not run successfully
--*/
{
        UNREFERENCED_PARAMETER(szCapabilityName);
        UNREFERENCED_PARAMETER(szComponentName);
        UNREFERENCED_PARAMETER(nAppNum);

        AFX_MANAGE_STATE(AfxGetStaticModuleState());
        CTestRegInfo regInfo(lConfigID);

        AfxMessageBox(_T("Component is added"), MB_OK |
MB_ICONINFORMATION);

        return TDX_SUCCESS;
}

DWORD
APIENTRY
OnRemoveComponent(
        LONG lConfigID,             // id of the active config
        LPCWSTR szCapabilityName,  // address of name of selected
capability
        LPCWSTR szComponentName,   // address of name of selected
component
        UINT nAppNum                // index of the activated tdx
        )
/*++
Function Description:
        Performs the operations needed when the component is
removed from the configuration.
Return code:
        TDX_SUCCESS - the applet was run successfully
        TDX_FAILED - the applet was not run successfully
        TDX_FAILED_NONOTIFY - the applet was not run successfully
--*/
{
        UNREFERENCED_PARAMETER(szCapabilityName);
        UNREFERENCED_PARAMETER(szComponentName);
        UNREFERENCED_PARAMETER(nAppNum);

        CTestRegInfo regInfo(lConfigID);
```

```
        AfxMessageBox(_T("Component is removed"));

        return TDX_SUCCESS;
}


DWORD
APIENTRY
OnPrebuildComponent(
        LONG lConfigID,             // id of the active config
        LPCWSTR szCapabilityName,  // address of name of selected
capability
        LPCWSTR szComponentName,    // address of name of selected
component
        UINT nAppNum                // index of the activated tdx
        )
/*++
Function Description:
        Performs the operations needed when the build of the
configuration is performed.

Return code:
        TDX_SUCCESS - the applet was run successfully
        TDX_FAILED - the applet was not run successfully
        TDX_FAILED_NONOTIFY - the applet was not run successfully-
--*/
{
        UNREFERENCED_PARAMETER(lConfigID);
        UNREFERENCED_PARAMETER(szCapabilityName);
        UNREFERENCED_PARAMETER(szComponentName);
        UNREFERENCED_PARAMETER(nAppNum);

        return TDX_SUCCESS;
}

DWORD
APIENTRY
OnCheckDependencyComponent(
        LONG configID,              // id of the active config
        LPCWSTR szCapabilityName,  // address of name of selected
capability
        LPCWSTR szComponentName,    // address of name of selected
component
        UINT nAppNum,               // index of the activated tdx
        LPBYTE lpDependency,              // address of data buffer
to include MULTI_SZ list of depended components
        LPDWORD lpcbDependency,     // address of data buffer size
        LPBYTE lpConflict,          // address of data buffer to
include MULTI_SZ list of conflicted components
```

```
        LPDWORD lpcbConflict              // address of data buffer
size
        )
/*++
Function Description:
        Gives the information of dynamic dependencies of the
component

Return code:
        TDX_SUCCESS - the applet was run successfully
        TDX_FAILED - the applet was not run successfully
        TDX_FAILED_NONOTIFY - the applet was not run successfully
--*/
{
        UNREFERENCED_PARAMETER(configID);
        UNREFERENCED_PARAMETER(szCapabilityName);
        UNREFERENCED_PARAMETER(szComponentName);
        UNREFERENCED_PARAMETER(nAppNum);
        UNREFERENCED_PARAMETER(lpDependency);
        UNREFERENCED_PARAMETER(lpcbDependency);
        UNREFERENCED_PARAMETER(lpConflict);
        UNREFERENCED_PARAMETER(lpcbConflict);

        return TDX_SUCCESS;
}
```

67. Select Save All from the tool bar.
68. Open the Test.def, and add the Exports for the entry points, leave OnCheckDependancyComponent and OnPrebuildComponent commented out.

```
OnPropertiesComponent
OnAddComponent
OnRemoveComponent
;OnCheckDependencyComponent  //Disabled entry point
;OnPrebuildComponent         //Disabled entry point
```

69. Select Save All from the tool bar.
70. Go to Build -> Configuration, and select Win32 Release.
71. Go to Build -> Build Test.tdx (and hope there are no errors).

4.10.10 Create Test KDF

Create a new KDF with the following registry entry:

HKLM\\SOFTWARE\TestTDX\TestTDXKey\TestValue Hello World

72. Add the TDX to the TDX properties.

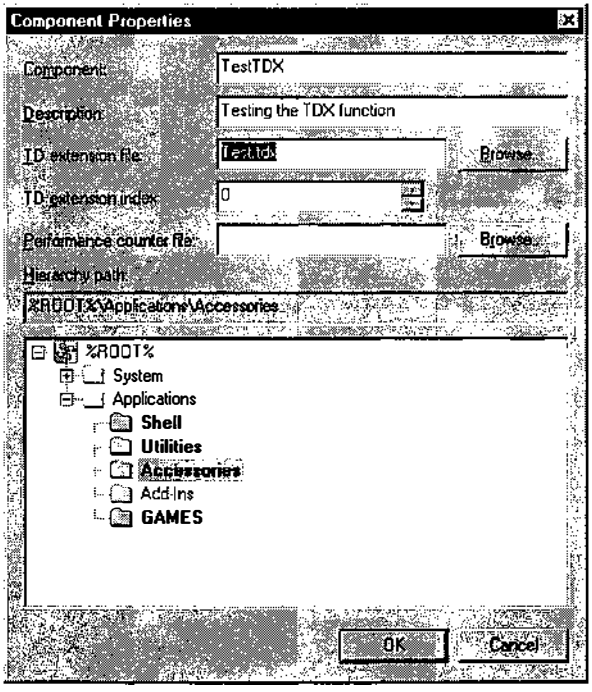


Fig. 4.43 Create the TestTDX Component

73. Right click on the KDF's root and change the properties and comments:

Properties:

Component Name: **TestTDXKDF**

Vendor: **Annasoft**

Description: **Test KDF for Test TDX**

Comments:

Copyright © 2000 Annasoft Systems

```
;Copyright (c) 2000 Annasoft Systems


[Version]

Signature = "$Windows NT$"
KdfVersion = 1


[Header]

Name = Test TDX
Version = 1.0
ReleaseDate = 03/22/2000
Vendor = Annasoft
OS = WINNT
OSVersion = 4.0
Platform = i386
Repository = "%WINNT%"
Description = "Test KDF for Test TDX"
CodePage = 1252


[Capabilities]

1 = "Accessories"


[Components]

1 = "TestTDX", "Testing the TDX function", 1, Test.tdx, 0


[CapabilityDependencies]


[CapabilityComponentDependencies]


[ComponentCapabilityDependencies]


[ComponentDependencies]
```

```
[Directories]


[TestTDX.Files]


[TestTDX.RegistryAdditions]


KeyName = "HKEY_LOCAL_MACHINE\SOFTWARE\TestTDX\TestTDXKey"

ValueName = "TestValue"
Type = REG_SZ
Data = "Hello World"
```

74. Save the KDF as TestTDX.KDF.
75. Copy the TDX and the new KDF to a directory called SAMPLE.
76. In Target Designer add the Test KDF.
77. Add the new component to any configuration. The OnAddComponent entry is called and a message box pops up.

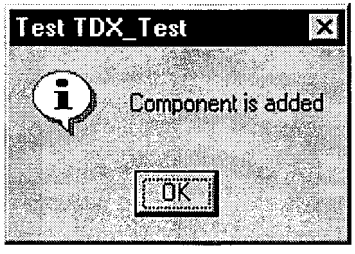


Figure 4.44: OnAddComponent Message Box

78. Open the Properties for the Component.

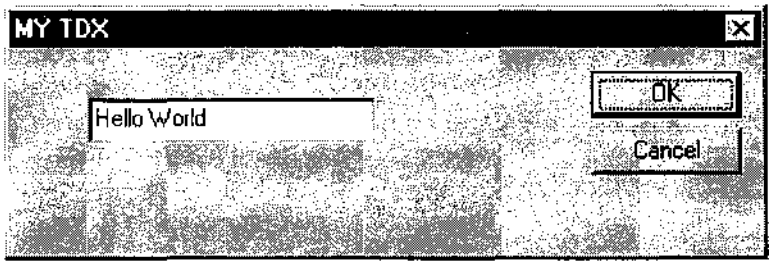


Figure 4.45: Change the TestValue Data

79. Change the Registry to something like: **TDXs are fun**. (Okay, type anything you want.)
80. Click OK
81. Walk the registry tree to see that the change took effect.

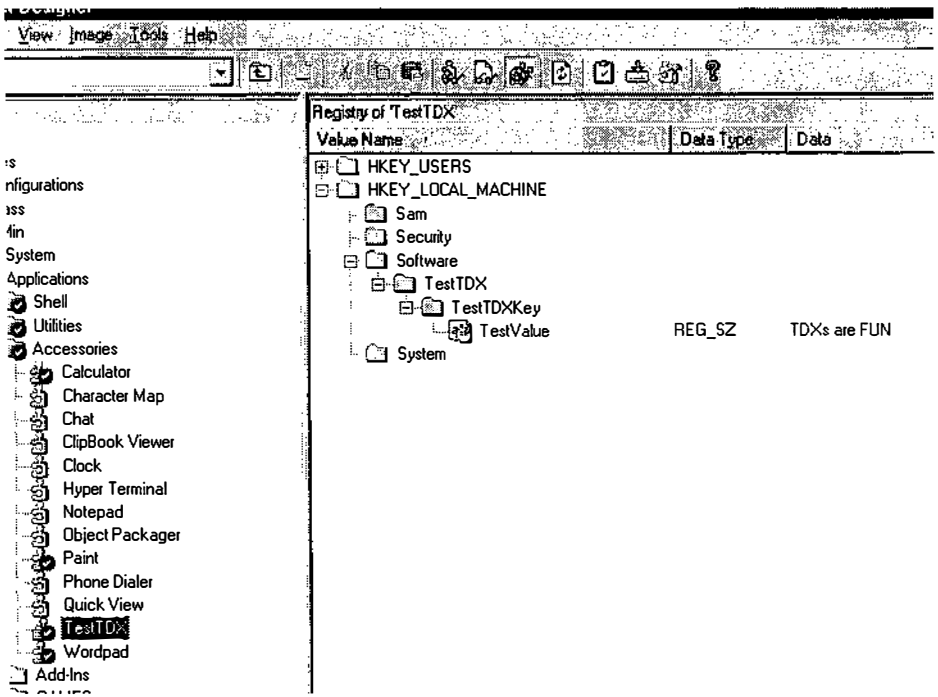


Figure 4.46: The Registry Key will Turn Blue when a Change is Made

82. Remove the component. The OnRemoveComponent entry is called.

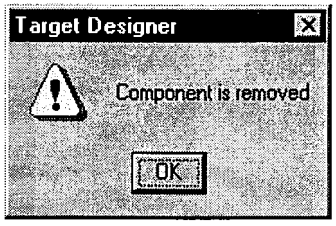


Figure 4.47: Remove Component Message Box

4.11 Summary

The "Hello World" application keeps getting more complicated. Well... that's not really a summary, but it does open more conversations.

In this section we explored the two types of KDFs: device driver and application. Each requires tools or methods to gather information to create the KDF. Exercise 4 focused on application KDFs, and Exercise 4 demonstrated that smaller images could be created even if certain support files are not part of a particular configuration. That exercise also stressed that familiarity with each of the components is important to understand how smaller images can be created and how dependencies effect KDF development.

Exercise 5 focused on a network driver KDF. If you have a network card or single board computer that uses the same RealTek 8139 network chip, you can use the component that you just created. Any new system will have different PCI settings for the chip, so you must change the PCI SLOT number, IRQ, and other settings to match the new system. The driver settings are found within the registry. In order to interrogate a new system and find the correct PCI settings, there are many free PCI sniff utilities available on the Internet. One such tool is called PCISNIFF, developed by Marco Koetsved. The utility can be found at:

http://members.hyperlink.net.au/~chart/index.htm

The method used to create the Driver KDF is called the HIV File method. The disadvantage of this method is that you must keep track of each HIV files for the KDF. REGEDT32 can be used to load and manipulate the registry entries so that they can be dragged directly into Component Designer to make a standalone KDF. Exercise 11 will introduce the Local method to create the KDF.

Exercise 6 implemented a Hello World TDX. The TDX provides a convenient user interface to make registry changes. The TDX can also perform other registry operations depending on which of five entry points are called. Three entry points were implemented, and Add /Remove message boxes were demonstrated. Each of the entry points could easily create or change registry keys.

Chapter 5 EXPLORING OTHER COMPONENTS AND FEATURES

Chapters 1 through 4 have covered the basics of NT Embedded. Chapter 5 will explore various features of NTE and reinforce the usage of Target Designer and Component Designer. There are several topics covered in this section: communications, web servers, remote administration, CD booting, error recovery, and multi-processor systems. Each exercise introduces a new feature, component, or concept. After exploring each exercise, you can create different combinations to build more realistic configurations.

5.1 Exercise 7 - Remote Administration, Graphical and Telnet

Now that you have added a network driver to your configuration, the next excercises will implement various features provided in Windows NT Embedded. NTE comes with several different types of remote administration features. Exercise 2 demonstrated a remote connection via serial. The serial connection is a text-based administration that provides simple access to a remote machine. In the world of Windows it would be nice to see what a user is seeing on their desktop without physically being there. Some third parties came up with graphical remote tools so that a remote administrator can see and take control of a remote machine's desktop. To provide this capability, Microsoft provides NetMeeting 3.0 as part of NTE as a Graphical Remote Administration component.

NetMeeting was designed to connect people in different locations over a Network. Video, audio, white board, file exchange, and remote desktop are a few features that NetMeeting provides. The remote desktop provides graphical remote administration.

Telnet is a popular terminal connection for UNIX systems, though it is new to Windows NT 4.0. NTE supports Telnet as a simple text-based remote management system. The

TCP/IP Utilities contain the FTP client, ping, Telnet client, arp, ipconfig, and other TCP/IP utilities. The FTP client is used to transfer files from within the remote system.

In this exercise, we will implement both Graphical Remote Administration and Telnet, and then repeat the exercise for a headless system.

5.1.1 Building the Image

1. Right click on Exercise5 and select Copy.
2. Right click on NTE Class and click Paste.
3. Rename the copy of Exercise5 to Exercise7
4. Select the following Components:

Capability	Component	Size	Check Box
System / Platform / SystemRole	Workstation System	0	
System / Platform / User Identification	Automatic Login	0	
System / Platform / Core OS	Minimal	7.87MB	
System / Platform / HAL	Standard HAL	958KB	
System / Platform / Virtual Memory	8	8MB	
System / Devices / Storage / Floppy Disk	Floppy	18KB	
System / Devices / Storage / Fixed Disk	EIDE	0	
System / Devices / Storage / Fixed Disk	SCSI Disk	15KB	
System / Devices / Storage / CD-ROM	EIDE CD-ROM	0	
System / Devices / Storage / CD-ROM	SCSI CD-ROM	22KB	
System / Devices / Storage / File Systems	FAT	139KB	
System / Devices / Display / Display Drivers	VGA	99KB	
System / Devices / Keyboard / Keyboard Layout	US Keyboard Layout	9KB	
System / Devices / Keyboard / Keyboard Drivers	PC/AT Keyboard	29KB	
System / Devices / Keyboard / Keyboard	English (US) Input	7KB	

Input Local	Locale		
System / Devices / Mouse	Logitech	29KB	
System / Devices / BUS Devices SCSI / SCSI Adapters	EIDE SCSI Adapter	0	
System / Devices / BUS Devices SCSI / SCSI Drivers	EIDE SCSI Driver	27KB	
System / Devices / Serial Ports	COM1	0	
System / Network / Network Membership	Workgroup Participation	0	
System / Network / Network Adapters	AAEON 5894 (RTL8139A)		
System / Network / Network Drivers	RTL8139A		
System / Network / Network Protocols	TCP/IP	514KB	
System / Network / Network Services	Computer Browser	0	
System / Network / Network Services	LAN Manager Server	329KB	
System / Network / Network Services	LAN Manager Workstation	322KB	
System / Network / Network Services	Messenger	38KB	
System / Network / Network Services	NetBIOS	39KB	
System / Network / Network Services	NT LM Security Support Provider	35KB	
System / Network / Network Services	RPC	1.015MB	
System / Network / Network Services	WinSock	178KB	
System / Network / Network Applications	Net Command (NET.EXE)	305KB	
System / Network / Network Applications	TCP/IP Utilities	733KB	
System / Windows Services / OLE and COM	OLE/COM	2.18MB	
System / Windows Services / Distributed COM	**DCOM**	0	
System / Windows Services / Cryptography	**Cryptography Core**	1.42MB	
System / Management / Management Applications / Remote Administration	**Graphical Remote Management**	900KB	
System / Management / Management Applications / Remote Administration	**Telnet Server**	475KB	

System / Management / Management Services	**Event Logging**	377KB	
System / Desktop Settings / Fonts	MS Sans Serif	151KB	
System / Desktop Settings / Application Links	Explorer Links	36KB	
System / Shared System Components	**Console Administrator**	56KB	
System / Shared System Components	Display	25KB	
System / Shared System Components	Keyboard Drivers	9KB	
System / Shared System Components	Keyboard Locale	135KB	
System / Shared System Components	**Logon Screen Saver**	37KB	
System / Shared System Components	Mouse Common	9KB	
System / Shared System Components	Network Common	1.91MB	
System / Shared System Components	SCSI	58KB	
System / Shared System Components	Serial Ports	44KB	
System / Shared System Components	Virtual Memory	0	
Applications / Shell	Explorer Shell	0	
Applications / Utilities	Explorer	322KB	
Applications / Utilities	**Windows NT Command**	262KB	
Applications / Accessories	Calculator	124KB	
Applications / Accessories	Notepad	225KB	

5. Build the image, prepare the target's hard drive, and download to target. (Please be sure that you are still using the same TCP/IP address to connect to your LAN.)

5.1.2 Graphical Remote Administration - Connection

1. Open NetMeeting 3.0 on your host system.
2. Select New Call from the Call menu

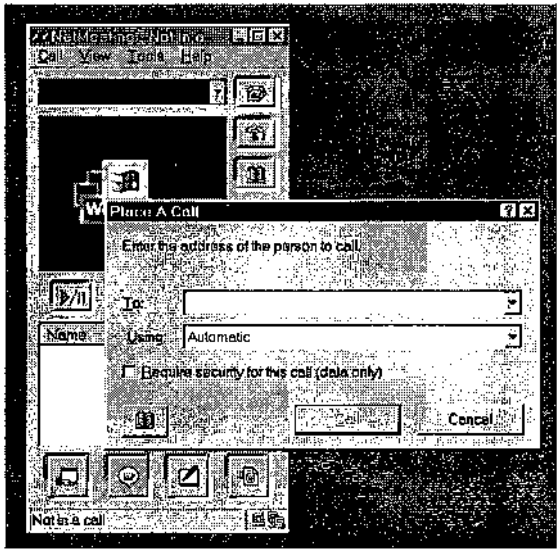


Figure 5.1: NetMeeting - Graphical Remote Administration

3. In the "To" text box enter the name of your target system (NTE1), check the Required security for this call, and click OK
4. You will be asked for a username and password. Enter Administrator for both.
5. You should see your target's desktop appear in the Window. The target's mouse will have a text box underneath indicating who has mouse control. Try moving your target's mouse.

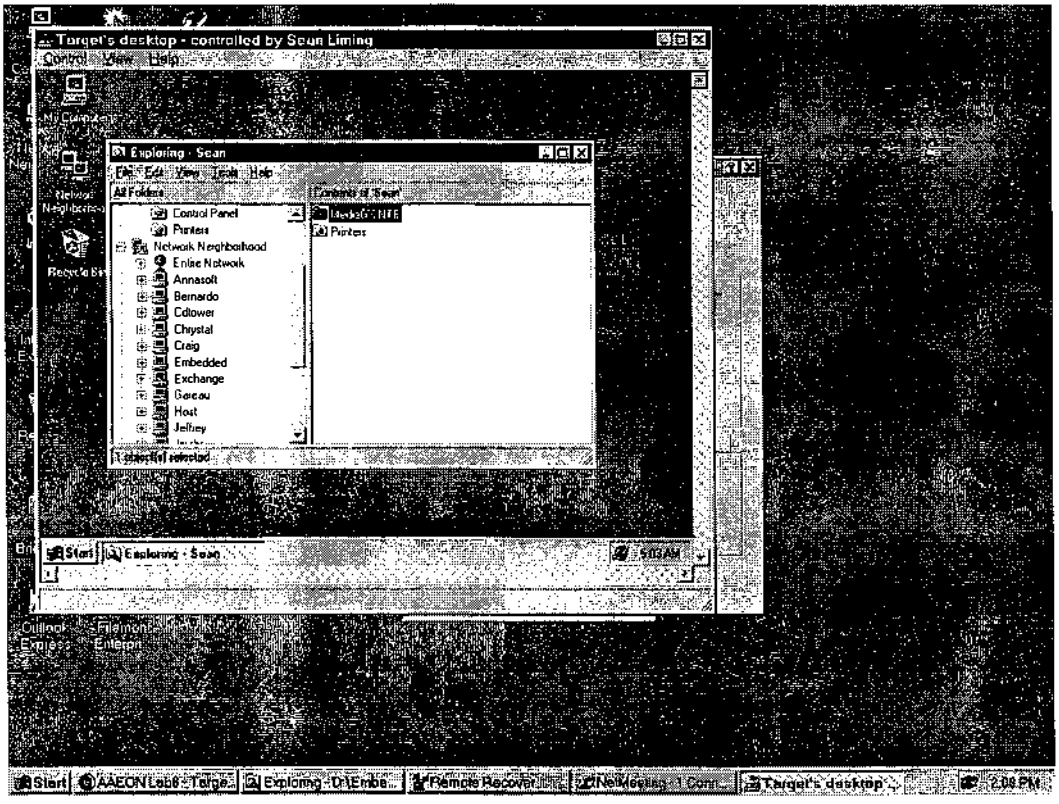


Figure 5.2: Viewing a Remote System on the Host

You can transfer files between the target and the host. Right click on the NetMeeting ICON found in the Taskbar. Select a file to transfer to the target or the host.

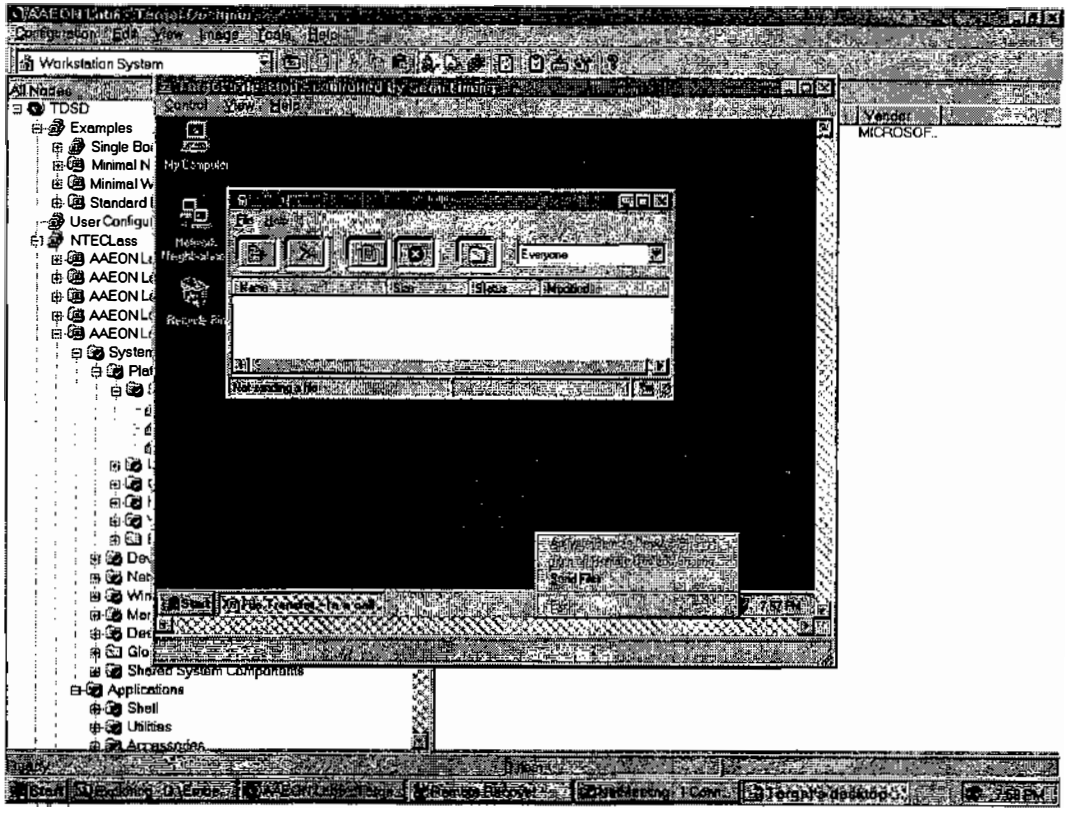


Figure 5.3: Sending a File from Target to Host

6. When completed, end the call. Notice that the target brings up the logon Screen Saver so that an unauthorized person cannot access the system when you are not present.

5.1.3 Telnet Connection

7. On your host machine, run TELNETC.EXE from the NTE CD-ROM. You can find the application in d:\tlntcnt.
8. You will see a DOS box with a **Microsoft Telnet>** Prompt.

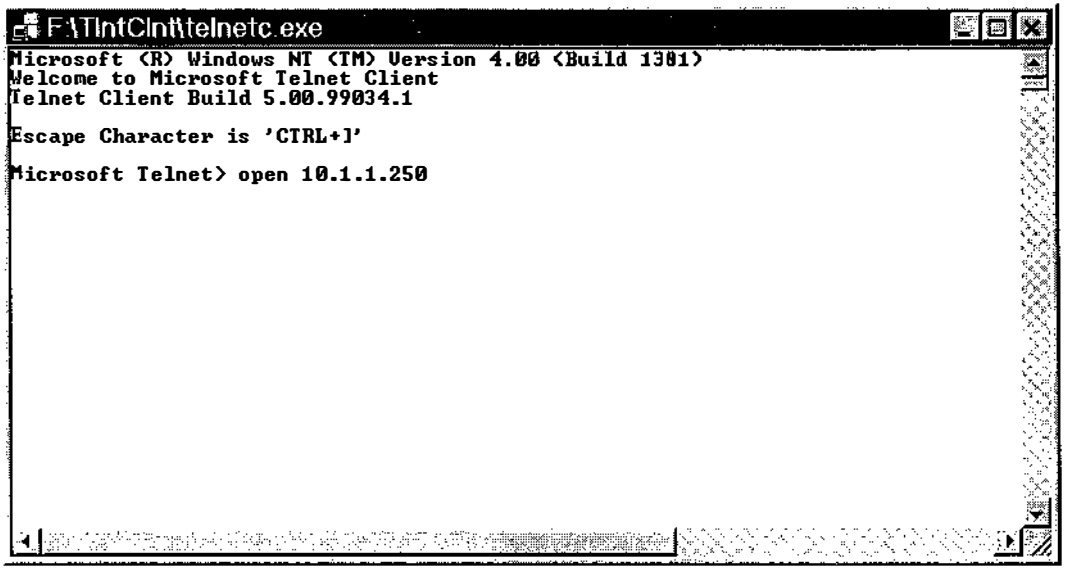


Figure 5.4:. Open a Telnet Session with the Target System

9. Type **open 10.1.1.2xx**, where xx is your workstation number.
10. You will be asked for user name and password. Type in **Administrator** for both.

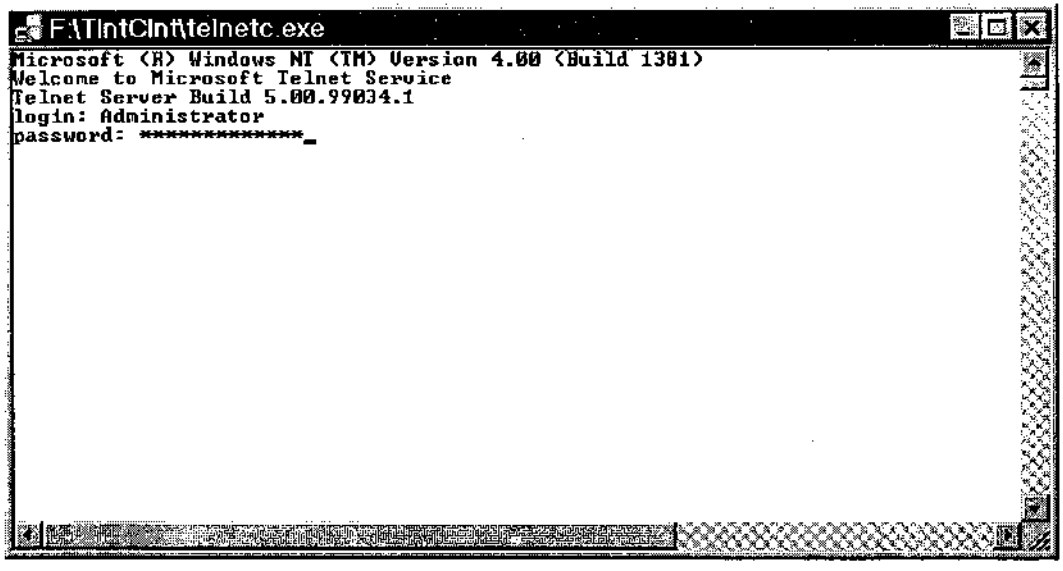


Figure 5.5: Enter User Name and Password

11. You will then get a DOS prompt. Now you can access your target computer.

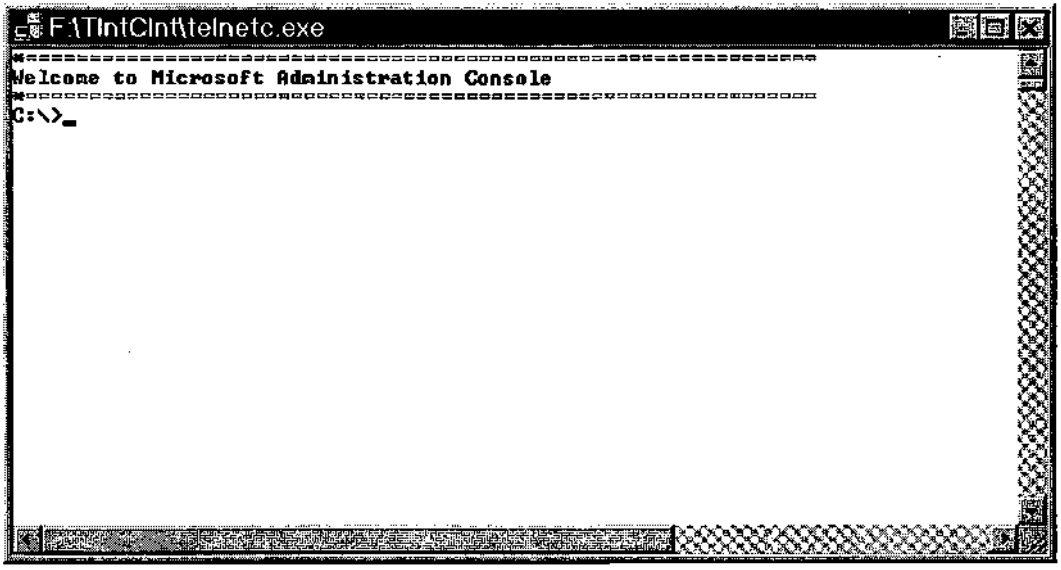


Figure 5.6: Login Completed

From here, you can view files, check error logs, change network settings, or transfer files using FTP client.

5.2 Exercise 7b - Graphical Remote Administration on a Headless System

In Exercise 6, we implemented Graphical Remote Administration. Some systems may not have a video card, but does this mean you can't perform Graphical Remote Admin? In this exercise, we will see if NetMeeting will connect to a headless system.

5.2.1 Build the Image

1. Right click on Exercise7 and select copy.
2. Right click on NTE Class and click paste.
3. Rename the copy of Exercise7 to Exercise7b.
4. Remove the VGA, PC/AT Keyboard, and Logitech components.

5. Add the NULL Video, Null Keyboard, and Null Mouse components.
6. Rebuild and download the OS to the target.
7. Reboot the target and try connecting to the target via NetMeeting 3.0 and Telnet.

You should verify that you made the connection and that you can view the desktop remotely.

5.3 Exercise 8 - Internet Information Server 3.0 and FrontPage 2000

The Internet has become part of our daily lives. Many third parties have developed web servers for a variety of implementations. Microsoft provides Internet Information Server as a standard feature in Windows NT Server. IIS supports WWW and FTP access. IIS 3.0 is included in Windows NT Embedded so developers can create embedded Internet appliances. In this exercise, we will implement IIS and create a web page using FrontPage 2000. We will also learn about modifying registry entries within Target Designer. Your target device will be an embedded web server, and you will view your posted website on your host's Internet Explorer.

IIS 4.0 can be used with NT Embedded. IIS 4.0 and other web utilities can be found in Option pack 4 of the MSDN subscription.

5.3.1 Build the Image

1. Right click on Exercise5 and select copy.
2. Right click on NTE Class and click paste.
3. Rename the copy of Exercise5 to Exercise8
4. Add the Server System, IIS 3.0, and Inet Service Manager components.
5. You should have the following components selected:

Capability	Component	Size	Check Box
System / Platform / SystemRole	Server System	0	
System / Platform / User Identification	Automatic Login	0	
System / Platform / Core OS	Minimal	7.87MB	

System / Platform / HAL	Standard HAL	958KB	
System / Platform / Virtual Memory	8	8MB	
System / Devices / Storage / Floppy Disk	Floppy	18KB	
System / Devices / Storage / Fixed Disk	EIDE	0	
System / Devices / Storage / Fixed Disk	SCSI Disk	15KB	
System / Devices / Storage / CD-ROM	EIDE CD-ROM	0	
System / Devices / Storage / CD-ROM	SCSI CD-ROM	22KB	
System / Devices / Storage / File Systems	FAT	139KB	
System / Devices / Display / Display Drivers	VGA	99KB	
System / Devices / Keyboard / Keyboard Layout	US Keyboard Layout	9KB	
System / Devices / Keyboard / Keyboard Drivers	PC/AT Keyboard	29KB	
System / Devices / Keyboard / Keyboard Input Local	English (US) Input Locale	7KB	
System / Devices / Mouse	Logitech	29KB	
System / Devices / BUS Devices SCSI / SCSI Adapters	EIDE SCSI Adapter	0	
System / Devices / BUS Devices SCSI / SCSI Drivers	EIDE SCSI Driver	27KB	
System / Devices / Serial Ports	COM1	0	
System / Network / Network Membership	Workgroup Participation	0	
System / Network / Network Adapters	AAEON 5894 (RTL8139A)		

System / Network / Network Drivers	RTL8139A	37KB	
System / Network / Network Protocols	TCP/IP	514KB	
System / Network / Network Services	Computer Browser	0	
System / Network / Network Services	IIS 3.0	722KB	
System / Network / Network Services	LAN Manager Server	329KB	
System / Network / Network Services	LAN Manager Workstation	322KB	
System / Network / Network Services	Messenger	38KB	
System / Network / Network Services	NetBIOS	39KB	
System / Network / Network Services	NT LM Security Support Provider	35KB	
System / Network / Network Services	RPC	1.015MB	
System / Network / Network Services	WinSock	178KB	
System / Network / Network Applications	Inet Service Manager	2.45MB	
System / Network / Network Applications	Net Command (NET.EXE)	305KB	
System / Network / Network Applications	TCP/IP Utilities	733KB	
System / Windows Services / OLE and COM	OLE/COM	2.18MB	
System / Management / Management Applications / System Configuration Management / Registry Management	Registry Editor	3.12MB	
System / Management / Management Applications / System Configuration Management / User Administration	USER Manager	915KB	
System / Management / Management Services	Event Logging	377KB	
System / Desktop Settings / Fonts	MS Sans Serif	151KB	
System / Desktop Settings / Application Links	Explorer Links	36KB	

System / Shared System Components	Display	25KB	
System / Shared System Components	FTP Performance Counters	0	
System / Shared System Components	Keyboard Drivers	9KB	
System / Shared System Components	Keyboard Locale	135KB	
System / Shared System Components	Mouse Common	9KB	
System / Shared System Components	Network Common	1.91MB	
System / Shared System Components	SCSI	58KB	
System / Shared System Components	Serial Ports	44KB	
System / Shared System Components	Virtual Memory	0	
System / Shared System Components	WWW Performance	0	
Applications / Shell	Explorer Shell	0	
Applications / Utilities	Explorer	322KB	
Applications / Accessories	Calculator	124KB	
Applications / Accessories	Notepad	225KB	

6. FrontPage 2000 will be used to create a web site. In order to publish the web site, an FTP connection must be set up so that the web files get posted to the correct location. In short, we need to point the FTP server's home directory to the WWW home directory. We also need to disable Anonymous Login and change the WWW default load file to match the name of the file provided by FrontPage 2000. In the IIS 3.0 component from within Target Designer, change the data in the following registry entries:

Registry Key	Data
HKLM\System\ControlSet001\Services\MSFTPSVC\ Parameters\VirtualRoots	**c:\InetPub\wwwroot,,3**
HKLM\System\ControlSet001\Services\MSFTPSVC\ Parameters\AllowAnonymous	**0x0**
HKLM\System\ControlSet001\Services\W3SVC\ Parameters\DefaultLoad File	**index.htm**

TIP: These values were found by first installing a full NTE configuration, and tracking the changes in the registry to changes in the Internet service manager setup dialogs. You will have to re-open regedt32 each time you make a new change to the registry.

7. Build the image, prepare your target's hard drive, and download the OS to the target platform.
8. A Default.HTM file comes as part of the build. Open Internet Explorer and type in your Target's TCP/IP address followed by the default.htm. The web page displays Embedded NT 4.0.

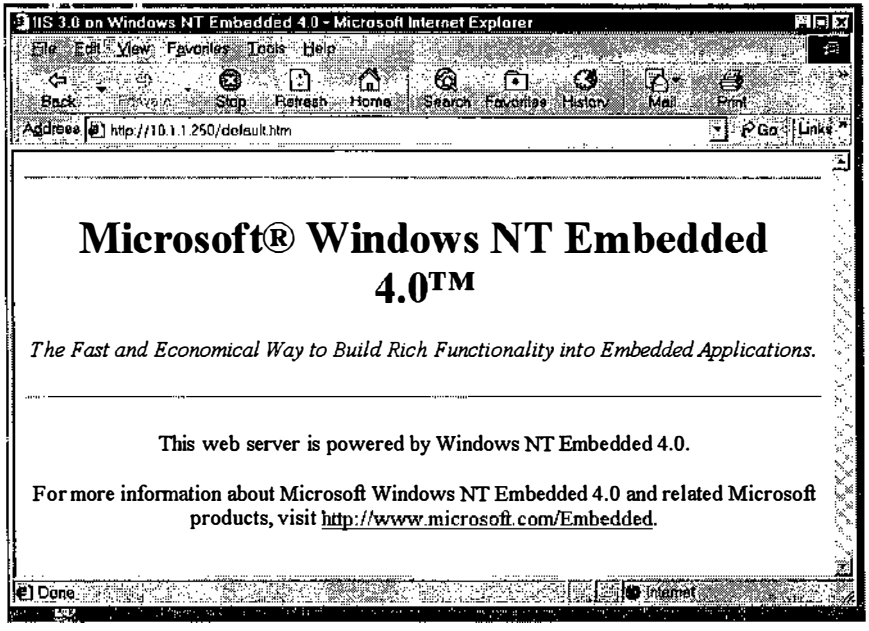


Figure 5.7: Accessing the Default HTM that comes with Windows NT Embedded

9. Open FrontPage 2000, and select FILE->New->WEB from menu.
10. FrontPage will create a new website with a new page ready for editing.
11. Create a design on the new page. (Now, I cannot be held responsible for the content you create.)

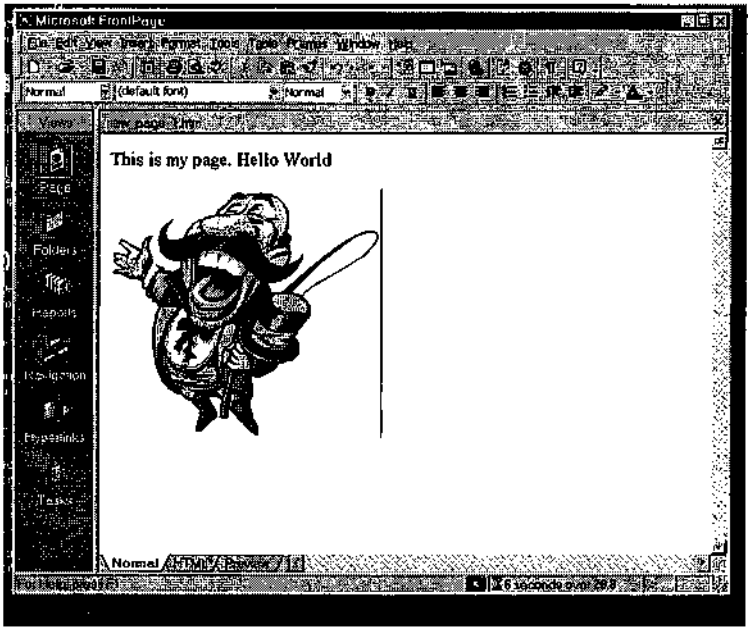


Figure 5.8: Creating a Web Site using MS FrontPage 2000

12. Save the page as **index.htm**.
13. Select Publish Web from the File menu

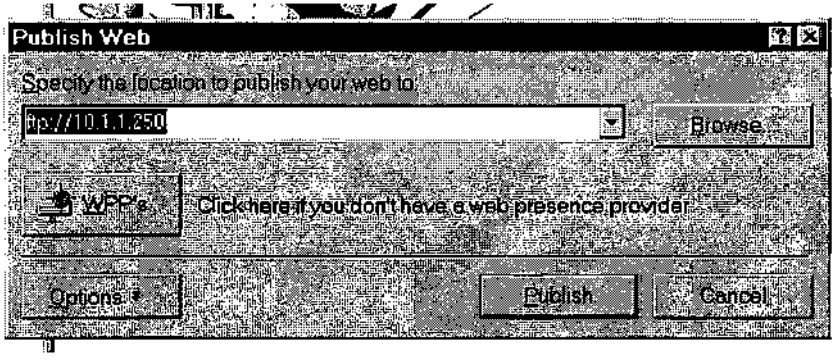


Figure 5.9: Publishing and Downloading WEB to Target Platform

14. Type in the FTP address for your target platform: ftp://x.x.x.x where x.x.x.x is the workstation TCP/IP address.
15. You will be asked for a user name and password. Use Administrator for both.

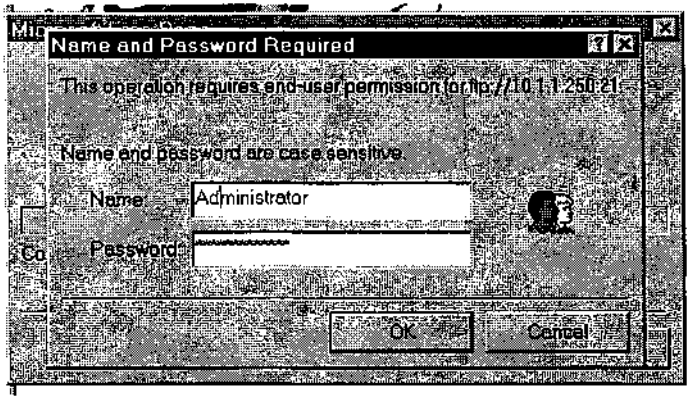


Figure 5.10: Enter Security Information to Access Remote Computer

16. Open Internet Explore and Type the following: http://x.x.x.x/index.htm (where x.x.x.x is your target's TCP/IP address). You should see your published web page.

5.4 Exercise 8b - Headless Web Server

Now let's make a headless web server. Using the same configuration from the last exercise, do the following:

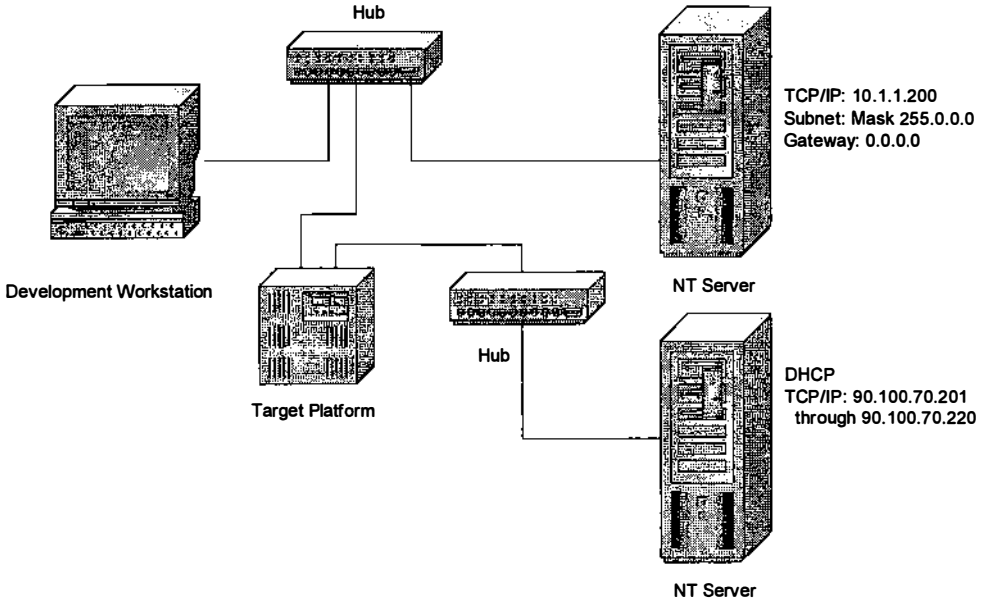
1. Add Null Video, Null Mouse, and Null Keyboard components.
2. Rebuild and download the image to the target.
3. Repeat steps 9-15 of Exercise 8. You should be able to connect and download new web pages to the target.

5.5 Exercise 9 - Dual Network Cards

Windows NT can support multiple network cards in a system. A multi-network card system can act as routers connecting multiple networks through one system. Another application is to dedicate one card for an internal network and a secured card for outside access.

In this exercise, we will build a dual-network card implementation. The primary card will be the NE2000 PCI adapter, and the secondary card will be the on-board AAEON 5894. The PCI card is NE2000 compatible, but has the Realtek 8029 Ethernet chip so the Realtek 8029 component will be used. You will need to access two different networks with two different TCP/IP addresses. This exercise will assume that you are using the same network as you have done in the previous exercises and a second network that supports TCP/IP. We will add to the 8139 KDF created in Exercise 5. To do so we will perform the following:

A. Create an NTE image that supports the 8029 adapter as the primary,
B. Install the 8139A as a secondary adapter,
C. Capture the registry settings for the secondary adatper (8139A),
D. Modify the 8139 KDF created in Exercise 5, to add a new component and add registry settings, and
E. Build a new configuration with primary and secondary adapters.



Setup for Dual Networks

5.5.1 Build an Image with Realtek 8029 Support

1. Right click on Exercise5 and select copy.
2. Right click on NTE Class and click paste.
3. Rename the copy of Exercise5 to Exercise10.
4. Add the Standard Core OS, which will remove the Minimal Core OS.
5. Add the Primary Domain Controller Component, right click on the component and select properties, and change the DOMAIN name to NTEDx, where x is your workstation number. **IF YOU ARE ASKED TO USE THIS NAME FOR DNS, SELECT NO.**
6. Remove the AAEON 5894 (RTL8139A) adapter and remove the RTL8139A driver from the configuration.
7. Add the Realtek RTL8029 PCI Adapter and the RTL8029 driver components
8. Select the following components:

Capability	Component	Size	Check Box
System / Platform / SystemRole	Primary Domain Controller	184KB	
System / Platform / User Identification	Automatic Login	0	
System / Platform / Core OS	Standard OS	74.32MB	
System / Platform / HAL	Standard HAL	958KB	
System / Platform / Virtual Memory	8	8MB	
System / Devices / Storage / Floppy Disk	Floppy	18KB	
System / Devices / Storage / Fixed Disk	EIDE	0	
System / Devices / Storage / Fixed Disk	SCSI Disk	15KB	
System / Devices / Storage / CD-ROM	EIDE CD-ROM	0	
System / Devices / Storage / CD-ROM	SCSI CD-ROM	22KB	
System / Devices / Storage / File Systems	FAT	139KB	

System / Devices / Display / Display Drivers	VGA	99KB	
System / Devices / Keyboard / Keyboard Layout	US Keyboard Layout	9KB	
System / Devices / Keyboard / Keyboard Drivers	PC/AT Keyboard	29KB	
System / Devices / Keyboard / Keyboard Input Local	English (US) Input Locale	7KB	
System / Devices / Mouse	Logitech	29KB	
System / Devices / BUS Devices SCSI / SCSI Adapters	EIDE SCSI Adapter	0	
System / Devices / BUS Devices SCSI / SCSI Drivers	EIDE SCSI Driver	27KB	
System / Devices / Serial Ports	COM1	0	
System / Network / Network Adapters	Realtek RTL8029 PCI Adapter		
System / Network / Network Drivers	RTL8029	37KB	
System / Network / Network Protocols	TCP/IP	514KB	
System / Network / Network Services	Computer Browser	0	
System / Network / Network Services	DHCP Client	56KB	
System / Network / Network Services	DNS Server	309KB	
System / Network / Network Services	LAN Manager Server	329KB	
System / Network / Network Services	LAN Manager Workstation	322KB	
System / Network / Network Services	Messenger	38KB	
System / Network / Network Services	NetBIOS	39KB	
System / Network / Network Services	NT LM Security Support Provider	35KB	
System / Network / Network Services	RPC	1.015MB	
System / Network / Network Services	WINS Server	1.89MB	

System / Network / Network Services	WinSock	178KB	
System / Network / Network Applications	Net Command (NET.EXE)	305KB	
System / Network / Network Applications	TCP/IP Utilities	733KB	
System / Windows Services / OLE and COM	OLE/COM	2.18MB	
System / Management / Management Applications / System Configuration Management / Registry Management	Registry Editor	3.12MB	
System / Management / Management Applications / System Configuration Management / User Administration	USER Manager for Domains	961KB	
System / Management / Management Services	Event Logging	377KB	
System / Desktop Settings / Fonts	MS Sans Serif	151KB	
System / Desktop Settings / Application Links	Explorer Links	36KB	
System / Shared System Components	Display	25KB	
System / Shared System Components	Keyboard Drivers	9KB	
System / Shared System Components	Keyboard Locale	135KB	
System / Shared System Components	Mouse Common	9KB	
System / Shared System Components	Network Common	1.91MB	
System / Shared System Components	SCSI	58KB	
System / Shared System Components	Serial Ports	44KB	
System / Shared System Components	Virtual Memory	0	
Applications / Shell	Explorer Shell	0	
Applications / Utilities	Explorer	322KB	
Applications / Accessories	Calculator	124KB	
Applications / Accessories	Notepad	225KB	

9. Right click on the TCP/IP component and change the TCP/IP address to support DHCP.
10. Right click on the Realtek RTL8029 PCI Adapter and set the properties to the following:

    Bus Number: **0**

    Bus Type: **PCI**

    Media Type: **Ethernet**

11. Build and Install the OS on your target platform.

5.5.2 Capture 8139A Second Instance Registry Keys

12. You will need the disk containing the Realtek 8139A driver.
13. Insert the disk in the floppy drive, and in your Target System open the Networking Control panel.
14. Go to the Adapters tab and click ADD.
15. Click on Have Disk.
16. Go to the A drive and select the Realtek 8139 driver.
17. Next, set the duplex mode to auto.
18. Click CLOSE, the bindings will begin.
19. The TCP/IP setup screen will pop-up. Set the TCP/IP settings for the Realtek 8139A adapter to the following:

    IP Address: 10.1.1.250
    Subnet mask: 255.0.0.0
    Or select the TCP/IP address that connects the target to your LAN.

20. Click OK
21. Click Apply.
22. Click OK.
23. The target computer will ask to reboot now. Click YES.
24. Once the target computer reboots, you want to capture two registry keys and store them as HIV files. Use REGEDT32.EXE to capture the following registry keys (the keys are different than before, so highlight the correct branch):

Component	Registry Entry	File
AAEON 5894 (RTL8139A) 2nd Adapter	HKLM\Software\Microsoft\Windows NT\CurrentVersion\NetworkCards\2	C:\RTL5.HIV
AAEON 5894 (RTL8139A) 2nd Adapter	HKLM\System\ControlSet001\Services\ **RTL81392**	C:\RTL6.HIV

25. Copy both HIV files from the target machine to the host machine c:\local\8139.
26. Open the AAEON RTL8139.kdf.
27. Add a new component to the KDF:

    Capability: Network Adapters (Second Instance)
    Component: AAEON 5894 (RTL8139A) 2nd Adapter
    Description: Ethernet for AAEON Platform

28. In the Registry Entry under the new component add the two new HIV files

Component: AAEON 5894 (RTL8139A) 2nd Adapter

Step1: Add Registry Key

Key path: **HKLM\Software**
Key name: **Microsoft\Windows NT\CurrentVersion\NetworkCards\2**

Step2: Add HIV File

HIV File: **HKLM\Software\Windows NT\CurrentVersion\NetworkCards\ 2RTL5.HIV**

Component: AAEON 5894 (RTL8139A) 2nd Adapter

Step 1: Add Registry Key

Key path: **HKLM\System**
Key name: **ControlSet001\Services\RTL81392**

Step2: Add HIV File

HIV File: **HKLM\System\ControlSet001\Services\RTL81392 RTL6.HIV**

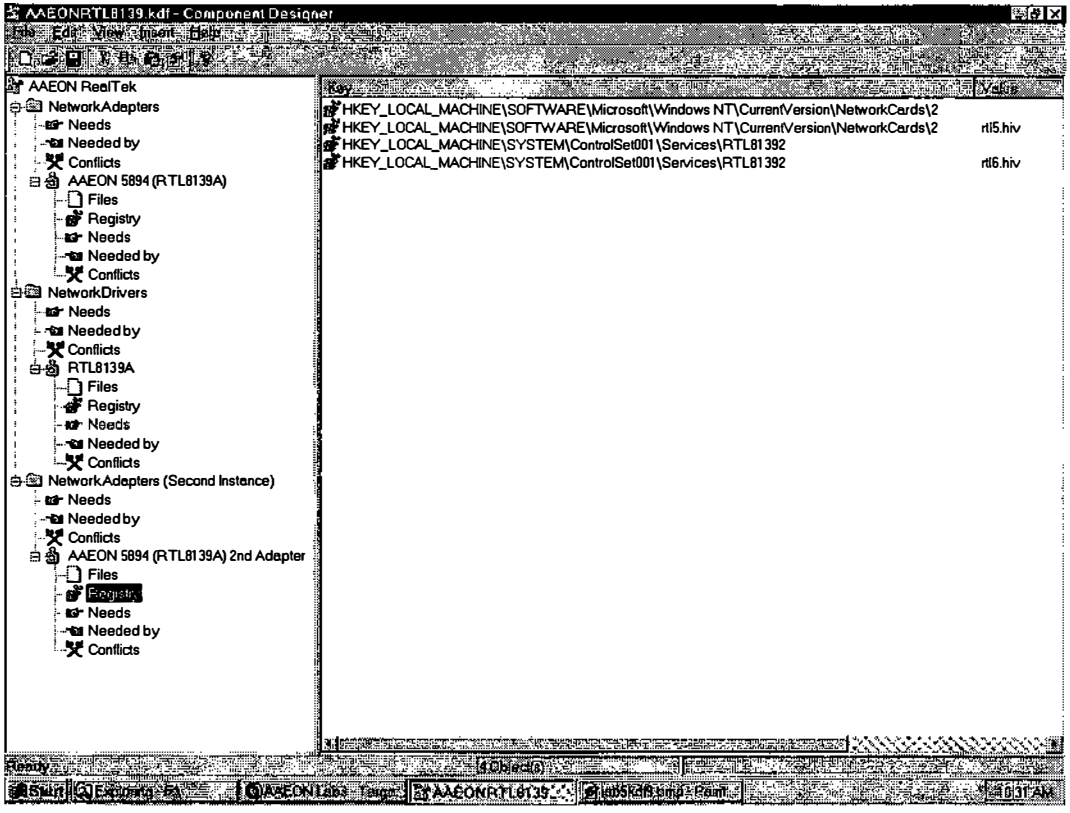


Figure 5.11: Two Registry HIVs for the Second Adapter

29. Save the KDF.
30. Now we must update the database with the updated KDF. In Target Designer, select Component Management under the Tools menu.
31. Select the 8139 entry and click the Remove button.
32. Add the updated 8139 KDF.
33. In Exercise9 add both the AAEON 5894 (RTL8139A) 2nd Adapter and RTL8139A components.
34. Build and install the new OS.

```
;Copyright (c) ACME Corp.

[Version]

Signature = "$Windows NT$"
KdfVersion = 1

[Header]

Name = AAEON RealTek
Version = 1.0
ReleaseDate = 08/31/1999
Vendor = REalTek via AAEON
OS = WINNT
OSVersion = 4.0
Platform = i386
Repository = "%WINNT%"
Description = "8139A Ethernet Driver"
CodePage = 1252

[Capabilities]

1 = "Network Adapters"
2 = "Network Drivers"
3 = "Network Adapters (Second Instance)"

[Components]

1 = "AAEON 5894 (RTL8139A)", "Ethernet for AAEON platform", 1
2 = "RTL8139A", "RealTek Driver", 2
3 = "AAEON 5894 (RTL8139A) 2nd Adapter", "On board 8139 as second
controller", 3

[CapabilityDependencies]

[CapabilityComponentDependencies]

[ComponentCapabilityDependencies]

[ComponentDependencies]
```

```
[Directories]

2 = "D:\Local\AAeon RTL8139"
3 = "\winnt\system32\drivers"
4 = "\winnt\system32"


[AAEON 5894 (RTL8139A).Files]


[AAEON 5894 (RTL8139A).RegistryAdditions]

KeyName = "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\NetworkCards"

FileName = "rtl1.hiv"


KeyName = "HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RTL81391"

FileName = "rtl2.hiv"


[RTL8139A.Files]

"RTL8139.SYS", 2, 3, 25088, "RTL8139.SYS"
"RTSNTHLP.HLP", 2, 4, 9126, "RTSNTHLP.HLP"


[RTL8139A.RegistryAdditions]

KeyName = "HKEY_LOCAL_MACHINE\SOFTWARE\Realtek"

FileName = "rtl3.hiv"


KeyName = "HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RTL8139"

FileName = "rtl4.hiv"


[AAEON 5894 (RTL8139A) 2nd Adapter.Files]


[AAEON 5894 (RTL8139A) 2nd Adapter.RegistryAdditions]
```

```
KeyName = "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\NetworkCards\2"

FileName = "rtl5.hiv"


KeyName = "HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\RTL81392"

FileName = "rtl6.hiv"
```

From within your target, you should be able to access two networks. A routing table can be set up on the target system between the two networks.

5.6 Exercise 10 - Creating a Bootable CD-ROM System

One of the many embedded features found in NTE is the ability to make Windows NT boot from CD-ROMs. Booting from a CD-ROM allows you to make upgrades easier, prevents data from being destroyed by end users, and you do not have to worry about hard drive corruption when a system is shut down prematurely.

The only drawback to booting from a CD-ROM is networking. If your target systems do not connect to a DHCP server, you will have to have a unique TCP/IP address for each CD. End users will not be able to set up a TCP/IP address, since the image and the TCP/IP address are built into a CD. Also, a system will have to be cloned one-by-one to create unique SIDs for each CD.

5.6.1 CD-ROM Software

You must have a software package that supports bootable CD-ROMs. Not all packages support Windows NT bootable CDs. The recommended software is CeQaudrat's WinOnCD (v3.6 or greater).

5.6.2 El Torito and WriteFilter Components

This exercise will feature the El Torito and WriteFilter components. As the NTE-CD boots, the PC BIOS looks into the El Torito partition for the bootstrap loader that is typically found on the first sector of the hard drive. The WriteFilter creates the scratch pad area in RAM as the OS continues to boot. All dynamic registry information will be created in RAM.

5.6.3 Exercise Overview

A computer that can boot from a CD will be needed for this exercise. The NTE image must be downloaded to a hard drive partition that is big enough to fit on a CD-ROM (650 MB). Most of the NTE images you have implemented thus far have been less than 100 MB. The following is an overview of the steps for the exercise:

A. Create a new configuration that includes the write filter and El Torito components.

B. Create a 100 MB partition on the target hard drive and format the partition as a FAT partition. (El Torito supports FAT file system partitions.)

C. One-by-one, copy OS boot components, \winnt subdirectory, and application files from the \EmbeddedOS directory to the target partition.

D. Boot the target system to verify that the image is working.

E. Remove the hard drive from the target and attach the hard drive to a computer that has a CD-ROM burner and the WinOnCD software.

F. Use WinOnCD to create the bootable El Torito partition.

G. Place the new CD in the target, and set the target's BIOS to boot from the CD-ROM drive.

5.6.4 Creating the Image

1. Right click on NTE Class, and select New Configuration.
2. Rename the New Configuration to Exercise10.
3. Select the following components to the configuration:

Capability	Component	Size	Check Box
System / Platform / SystemRole	Work Station	0	
System / Platform / User Identification	Automatic Login	0	
System / Platform / Core OS	Standard OS	74.32MB	
System / Platform / HAL	Standard HAL	958KB	

System / Platform / Virtual Memory	No Page File	0MB	
System / Devices / Storage / Floppy Disk	Floppy	18KB	
System / Devices / Storage / Fixed Disk	EIDE	0	
System / Devices / Storage / Fixed Disk	El Torito CD as Disk	7KB	
System / Devices / Storage / Fixed Disk	SCSI Disk	15KB	
System / Devices / Storage / CD-ROM	EIDE CD-ROM	0	
System / Devices / Storage / CD-ROM	SCSI CD-ROM	22KB	
System / Devices / Storage / File Systems	FAT	139KB	
System / Devices / Storage / Storage Filter Drivers	Write Filter	17KB	
System / Devices / Display / Display Drivers	VGA	99KB	
System / Devices / Keyboard / Keyboard Layout	US Keyboard Layout	9KB	
System / Devices / Keyboard / Keyboard Drivers	PC/AT Keyboard	29KB	
System / Devices / Keyboard / Keyboard Input Local	English (US) Input Locale	7KB	
System / Devices / Mouse	Logitech	29KB	
System / Devices / BUS Devices SCSI / SCSI Adapters	EIDE SCSI Adapter	0	
System / Devices / BUS Devices SCSI / SCSI Drivers	EIDE SCSI Driver	27KB	

System / Windows Services / OLE and COM	OLE/COM	2.18MB	
System / Desktop Settings / Fonts	MS Sans Serif	151KB	
System / Desktop Settings / Accessibility	Standard Accessibility	0	
System / Shared System Components	Display	25KB	
System / Shared System Components	Keyboard Drivers	9KB	
System / Shared System Components	Keyboard Locale	135KB	
System / Shared System Components	Mouse Common	9KB	
System / Shared System Components	SCSI	58KB	
System / Shared System Components	Virtual Memory	0	
Applications / Shell	Explorer Shell	0	
Applications / Utilities	Explorer	322KB	
Applications / Accessories	Wordpad	2.40MB	

4. Make sure the all read-only partitions is selected in the WriteFilter properties.
5. Build the NTE image.

5.6.5 Preparing the Target Drive

Most likely the CD-ROM burner and software are on another system. In order to create the CD-ROM, the WinOnCD must have access to a bootable partition. Once the NTE files have been copied to the target's boot hard drive, you can move this hard drive to the CD-ROM burner computer. Moving a hard drive to another system can present problems. Partitioning a hard drive on one computer is setting the calibration for that particular computer's BIOS. Thus, another computer may not be able to read the hard drive. An alternative is to connect the target to the CD burner PC using Remote Recover.

1. Connect the target system to the host using Remote Recover.

2. The target's hard drive boot partition must be less than 600 MB (the capacity limit of CD-Rs). The size of the NTE image is approximatly 80 MB. On the host, run Disk Administrator, and create a 100 MB partition on the first hard drive.
3. Using Explorer, format the target's bootable partition as a FAT partition.
4. Some BIOSes may not be able to see the full El Torito partition. As a result, you must copy the boot files in a specific order so that the BIOS can boot the OS.

In the following order, copy the following files to the target's hard drive:

A. BOOT.INI
B. NTLDR
C. NTDETECT.COM
D. WINNT (Subdirectory) - You must copy the entire subdirectory.
E. Copy the rest of your files. In this exercise, the last files are Program subdirectory and Temp subdirectory.

5. Once the copy is complete, you can boot the target system to verify that the image is working properly.
6. Remove the hard drive and attach it to a computer that has a CD-R burner and the WinOnCD software.

5.6.6 Creating the CD

1. On your CD-burner station, start the WinOnCD program.
2. Under the Data tab, select bootable CD.

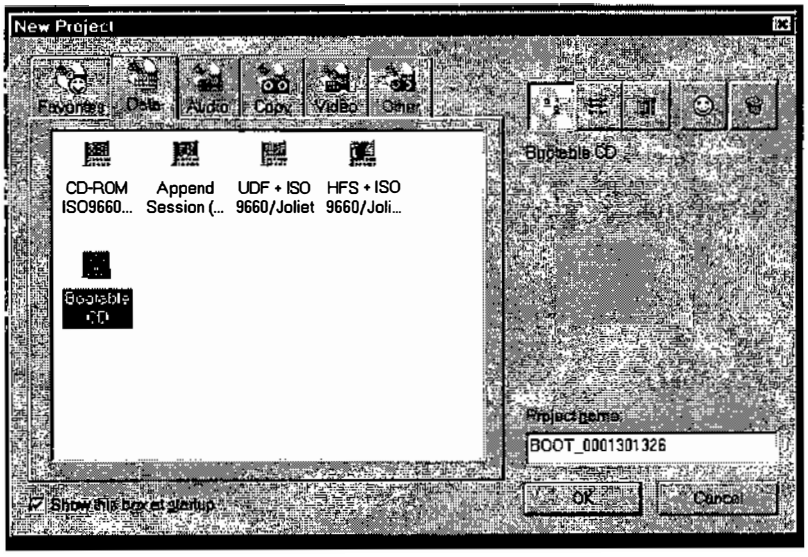


Figure 5.12: Creating a New WinOnCD Project

3. Make sure that you have access to the target's bootable hard drive. Click and drag the hard drive from the upper left Explorer pane to the boot source track found in the Bootable CD editor at the bottom of the screen.

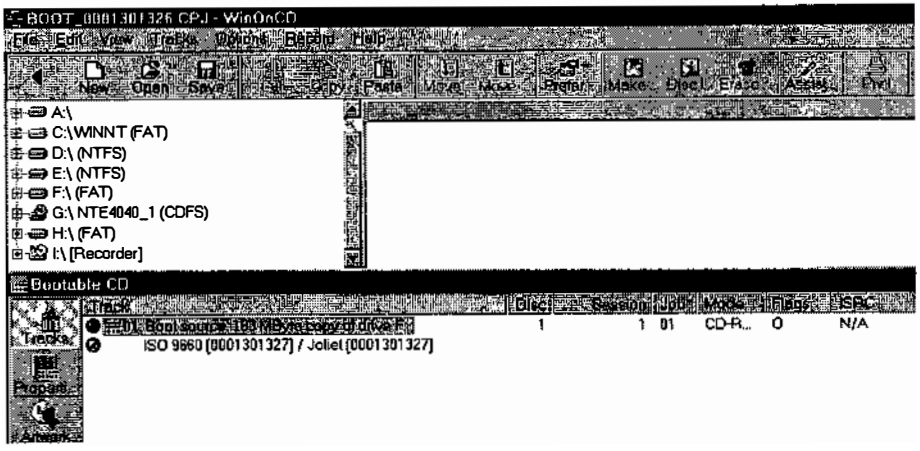
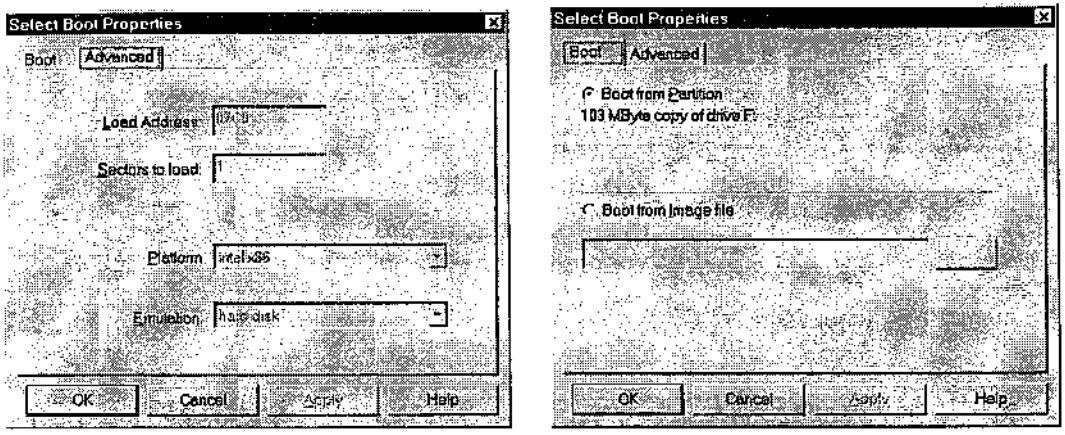


Figure 5.13: Drag and Drop the Boot Source Hard Drive to the Boot Source Section



Figures 5.14 and 5.15: Viewing the Boot Parameters

As you can see from the Properties view, the CD will emulate the hard drive boot.

4. With a blank CD-R in the CD-ROM burner, you are now ready to record the CD. With the record properties make sure that you select to Close Disk.

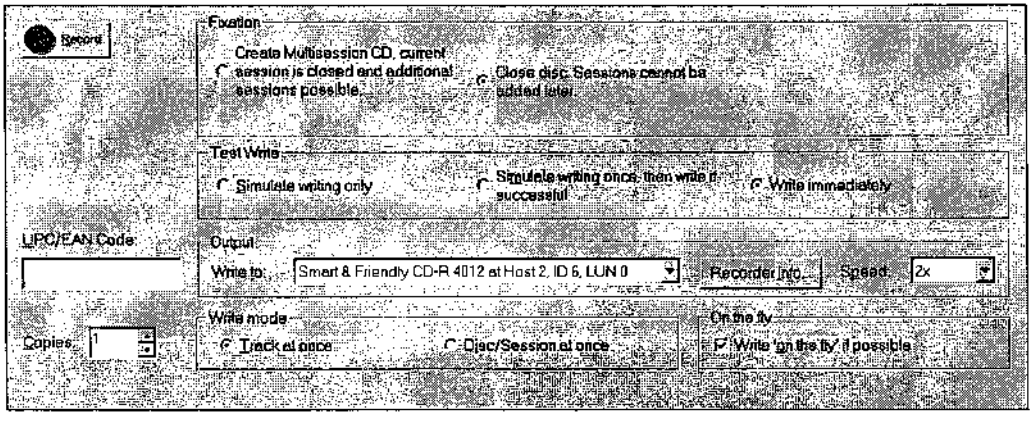


Figure 5.16: Recording the CD

5. Click on the Record button.
6. After the recording is completed, try booting the CD-ROM in the target system.

After the system boots, you will see a hard drive drive C and a CD-ROM drive D. Drive C will contain all of the system files that make up the image. You will not be able to access drive D.

5.7 Exercise 10b - Data / Applications on the ISO-9660 Track.

The OS image can be placed on a small partition, which leaves more room for data and other applications on an ISO-9660 track within the CD-ROM. In this exercise, let's add an application like Notepad to the ISO-9660 track.

1. Add the CDFS and Notepad components to the Exercise10 configuration.
2. Build the image and copy the image to the target hard drive as you did in the last exercise.
3. From within WinOnCD, copy the target's hard drive to the boot partition track, and copy Notepad to the ISO-9660 track.

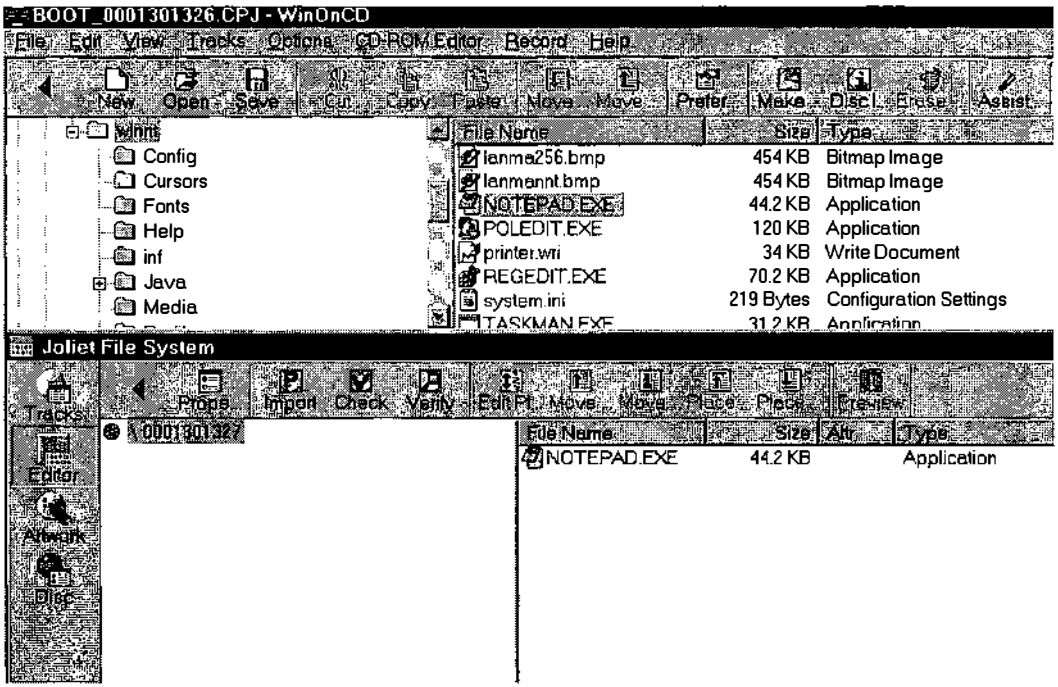


Figure 5.17: Adding Files to the ISO 9660 / Joliet Track

4. Record the CD and boot the target with the new CD.
5. When you boot, you should be able to access Drive D and run the Notepad application.

5.8 Exercise 11 - Error Recovery

System messages sometimes appear as dialog boxes requesting user action. Most of these messages are errors of which the user must be made aware; no disk in drive, DHCP server disconnect, and service not started or failed to start. The message dialogs are answered by the user, but in a headless system there is no video for a user to see (except via Graphical Remote Management). The System Message Interception (SMI) component can automatically respond to these messages. When an error message occurs, the SMI component can log the event to the Event Log and execute a DLL that can take further action.

For this exercise, a DHCP message will be invoked by disconnecting the target from the network. The exercise is broken down into two parts. The first part will use the SMI component to log the event to the Event log and display the DHCP message dialog on the screen. The second part will include the Automatic Reply feature so that the Event log occurs but the dialog box is not displayed.

5.8.1 Part 1 – Event Log

1. Copy the Exercise 5 configuration, and rename it Exercise 11.
2. Include the System Management Interception, DHCP Client, Event Logging, and Event Viewer components.
3. Modify the TCP/IP Component properties by selecting the "Obtain an IP address from a DHCP Server" radio button.
4. Make sure that a network card is selected.
5. Check the Event Logging box in the SMI properties dialog.

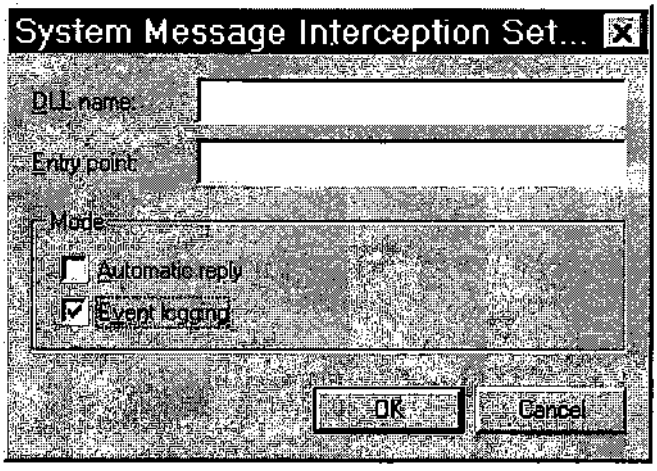


Figure 5.18: Selecting the Event Logging

6. Build and download the OS to the target system.
7. Remove the network cable from the target's Ethernet port.
8. Reboot the system.

The system will appear to take a longer time to finish booting because of the search to find the DHCP server. The DHCP search will eventually time out, and the DHCP message dialog will appear on the screen.

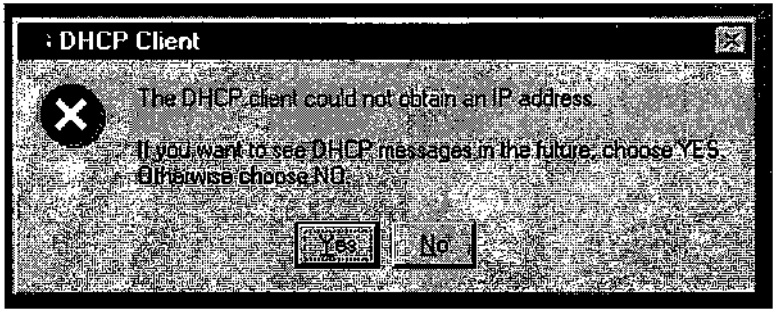


Figure 5.19: DHCP Error

9. Run the Event Viewer to see that the DHCP message is in the Event Log.

5.8.2 Part 2 – Automatic Reply

1. Reconnect the Network cable to the target.
2. Using the Exercise 11 Configuration, check the Automatic Reply box in the SMI properties dialog.

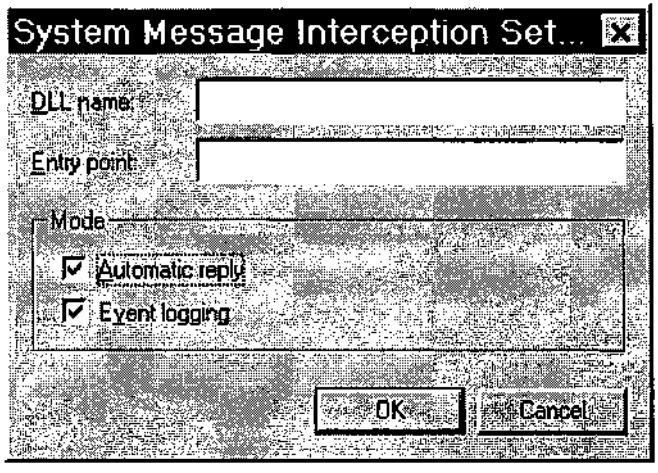


Figure 5.20: Select Automatic Reply

3. Build and download the OS to the target system.
4. Reboot the system.

The message will still be in the Event log, but the dialog box will not appear on the screen. The Automatic Reply selects the default reply. In the case of the DHCP message, the YES button was selected. In the real world, the system may need to do more than just auto respond to an error condition. The SMI component leaves room for a DLL to provide further action. The DLL could flash an LED or send a page to an administrator to indicate that an error occurred. (Please review Target Designer's online help for more information regarding the SMI component.)

5.9 Exercise 12 - SCSI KDF – Local Method, and SCSI Support

Now let's switch to a SCSI hard drive. In this exercise, we will create a KDF for a SCSI adapter. Exercises 5 and 9 introduced the HIV file method to create driver KDFs. A more

local approach will be used to create the KDF by installing Target Designer and Component Designer on the target platform. Drag-N-Drop from the local registry will be used to replace the HIV files.

5.9.1 Target Platform Hardware

The target hardware will consist of the following:

- Pentium processor
- 32 MB RAM
- Floppy drive
- IDE hard drive
- IDE CD-ROM
- Adaptec Ultra Wide 2940UW SCSI Adapter (PCI) or equivalent
- Ultra Wide SCSI hard drive
- VGA, keyboard, PS/2 mouse

We will first download the Standard NTE OS to the IDE hard drive and set up Component Designer on the target system. Once the SCSI Adapter KDF is created, the IDE hard drive and CD-ROM will be removed, and an image will be built with the new components and downloaded to the SCSI hard drive.

5.9.2 Build up the Target Platform

1. Make a copy of the Exercise1 configuration, and rename it to Exercise 11.
2. Include the NTFS and CDFS components.
3. Build and download the configuration to the target's IDE drive.
4. Insert the NTE CD1 and install IE 4.01 standard. The system will then reboot after installation is complete.
5. Next, run the NTE setup to install Target Designer and Component Designer. The system will then reboot. Now the target has everything needed to build the KDF.

5.9.3 SCSI Driver Installation

1. You will need the driver disk that contains the SCSI card driver (for the 2940UW the driver aic78xx.sys is required). Please see your SCSI card documentation for more information.
2. Go to Control Panel->SCSI Adapter.

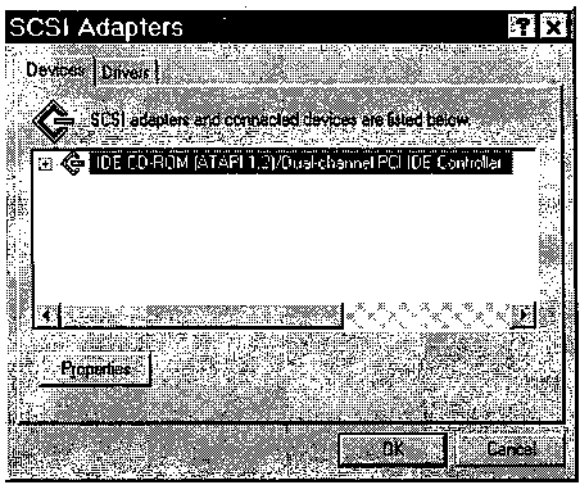


Figure 5.21: SCSI Adapter Applet

3. Click on the **Drivers** tab.
4. Click the **ADD** button.
5. Insert the SCSI adapter's driver disk into the floppy drive. Click the **Have Disk** button.
6. Select the SCSI adapter driver for installation. Once the SCSI driver has been installed the system will need to reboot.

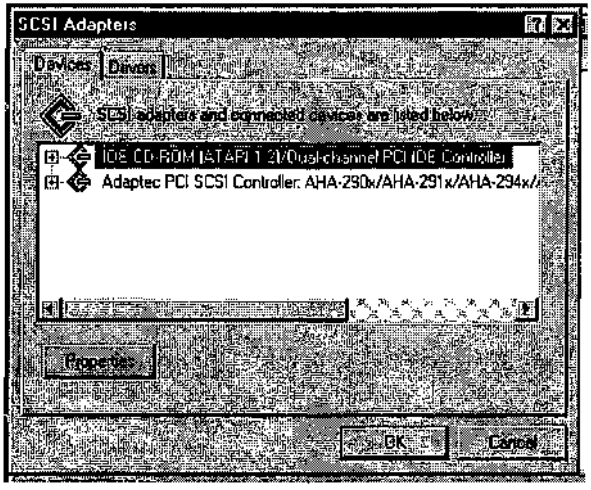


Figure 5.22: SCSI Adapter Added to the Device List

7. After the system reboots, open Disk Administrator.

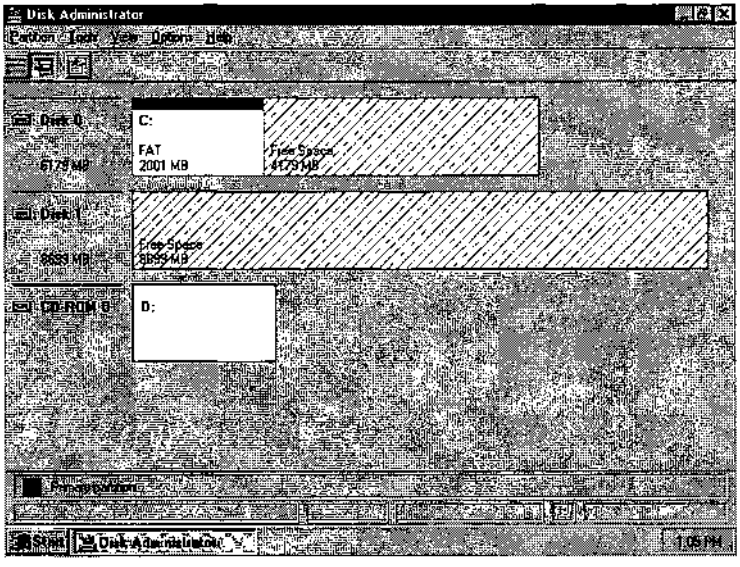


Figure 5.23: SCSI Driver is Shown as Free Space within Disk Administrator

8. Create a single partition on the SCSI drive.

9. Format the partition as an NTFS drive.

5.9.4 Creating the SCSI KDF

1. In the target system, open Component Designer.
2. Right click on Component1, and add the properties for Component name, Vendor, and Description:

   Component name: **Ultra Wide SCSI CARD**
   Vendor: **Adaptec**
   Description: **Ultra Wide SCSI PCI Card 2940UW Pro**

3. Right click on Mspaint and select Comments

   Enter: **Copyright © 2000 ACME Corp.**

The SCSI KDF will contain two components: one is the Adapter and the other is the driver. Each component will be placed in their respective Capabilities hierarchy.

4. First, let's create the drive component. Right click on **Ultra Wide SCSI CARD** found in the left pane, and select Add Component.
5. A dialog will appear asking for the name of the component, **2940UW (AIC78xx.SYS).**
6. Type in a description in the description text box: **Ultra Wide SCSI Driver.**
7. Select the hierarchy path within Target Designer. Highlight Accessories under %ROOT%\System\Devices\BUS Devices\SCSI Drivers
8. Click OK.

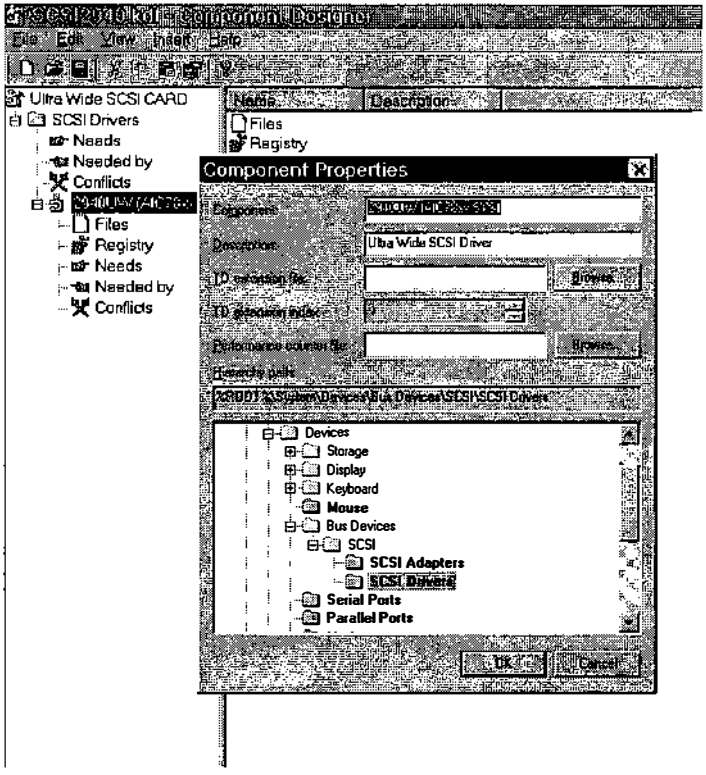


Fig 5.24. Creating the Driver Component

9. Now let's create the Adapter component. Right click on **Ultra Wide SCSI CARD** found in the left pane, and select Add Component
10. A dialog will appear asking for the name of the component, **AHA 2940UW Pro PCI Card**
11. Type in a description in the description text box: **Ultra Wide SCSI Card.**
12. Select the hierarchy path within Target Designer. Highlight Accessories under %ROOT%\System\Devices\BUS Devices\ SCSI Adapter

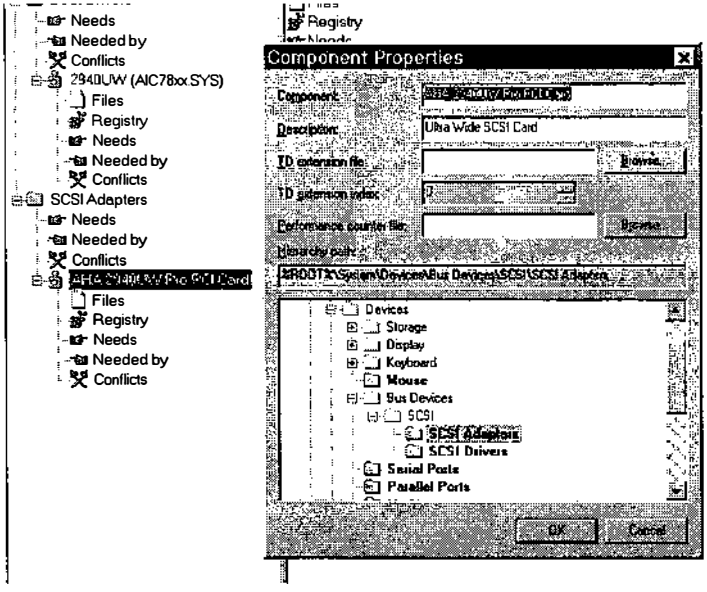


Figure 5.25: Creating the Adapter Component

13. The adapter component will not contain any files or registry entries. The adapter component will only contain a dependency on the driver component. Look at the SCSI adapters and SCSI drivers in Target Designer to confirm that the SCSI adapters contain nothing but a dependency on the SCSI driver. The same is true for VGA adapters and drivers. In the left pane in Component Designer, highlight the SCSI Drivers capability.

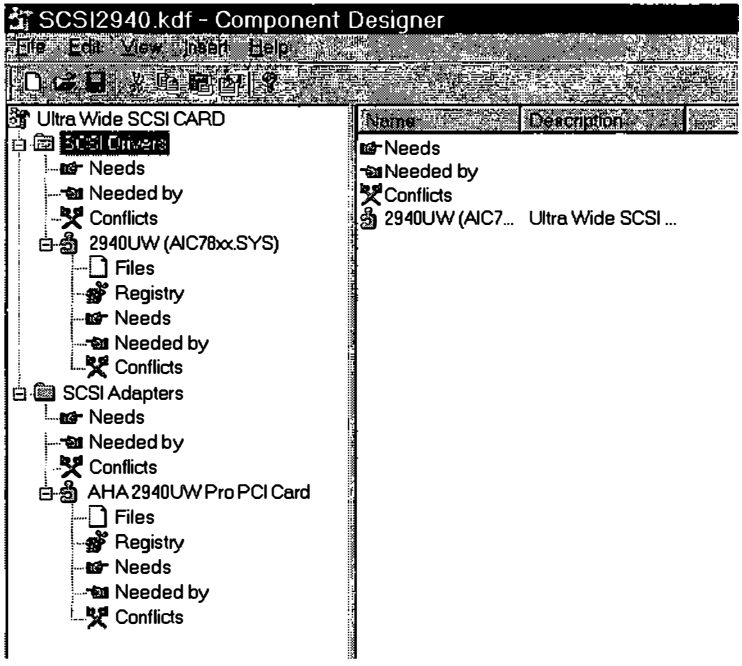


Figure 5.26: Highlight the Capability to Create a Dependency within the KDF between Components

14. The SCSI Driver component will be listed in the right pane. Drag the component to the NEEDS under the AHA 2940UW Pro PCI Card component. Now every time the adapter is selected in Target Designer the driver component will be required.

The driver component will contain the files and the registry entries. We will use the drag from local registry feature to capture the registry entries.

15. Highlight Registry Entries under the 2940UW (AIC78xx.SYS) component.
16. Right-click anywhere in the right pane and select drag from local registry. A REGEDIT application is started.
17. Transverse the registry to the bolded text in the table below. Click, drag, and drop the registry keys into Component Designer right pane.

Registry Entry
HKLM\System\ControlSet\Service**aic78xx**
HKLM\System\ControlSet\ENUM\ROOT**SCSIADAPTER**

Notice that the ControlSet001 Entries are grayed out. The ControlSet00X Entries are copies of ControlSet and are used to restore the previous configuration if a previous installation could boot the system successfully. When the ControlSet entries are brought over to Component Designer, ControlSet is changed automatically to ControlSet001.

18. Save the KDF as **Adaptec 2940.KDF** and exit Component Designer.
19. Copy the KDF to a blank floppy disk.
20. In Windows Explorer, locate the files: aic78xx.sys found in \winnt\system32\ drivers. Copy the driver file to the same floppy disk that holds the KDF.

The completion of the KDF will be on the host machine.

21. On the host machine, create a sub-directory under the d:\local directory called **AHA2940UW**.
22. Copy the KDF and the SYS files to the new AHA2940UW directory.
23. On the host, run Component Designer and open the **Adaptec 2940.KDF** file.
24. Highlight Files under the 2940UW (AIC78xx.SYS) component.
25. Right click anywhere in the right pane, and select drag from Explorer.
26. Drag-N-Drop the AIC78xx.SYS file found in d:\local\AHA2940UW to Component Designer's right pane.
27. Change the target directory to **\winnt\system32\drivers.**
28. Save the KDF, and close Component Designer.

The KDF should look like the following:

```
;Copyright (c) 2000 ACME Corp.

[Version]

Signature = "$Windows NT$"
KdfVersion = 1

[Header]
```

```
Name = Ultra Wide SCSI CARD
Version = 1.0
ReleaseDate = 02/26/2000
Vendor = Adaptec
OS = WINNT
OSVersion = 4.0
Platform = i386
Repository = "%WINNT%"
Description = "Ultra Wide SCSI PCI Card 2940UW Pro"
CodePage = 1252


[Capabilities]

2 = "SCSI Drivers"
1 = "SCSI Adapters"


[Components]

2 = "2940UW (AIC78xx.SYS)", "Ultra Wide SCSI Driver", 2
1 = "AHA 2940UW Pro PCI Card", "Ultra Wide SCSI Card", 1

[CapabilityDependencies]


[CapabilityComponentDependencies]


[ComponentCapabilityDependencies]


[ComponentDependencies]

1,2,Y


[Directories]

2 = "D:\Local\AHA2940UW"
3 = "\winnt\system32\drivers"


[2940UW (AIC78xx.SYS).Files]

"aic78xx.sys", 2, 3, 70000, "aic78xx.sys"
```

```
[2940UW (AIC78xx.SYS).RegistryAdditions]

KeyName = "HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\aic78xx"

ValueName = "Type"
Type = REG_DWORD
Data = 0x1


ValueName = "Start"
Type = REG_DWORD
Data = 0x0


ValueName = "Group"
Type = REG_SZ
Data = "SCSI miniport"


ValueName = "ErrorControl"
Type = REG_DWORD
Data = 0x1


ValueName = "Tag"
Type = REG_DWORD
Data = 0x7


ValueName = "ImagePath"
Type = REG_EXPAND_SZ
Data = "System32\drivers\aic78xx.sys"


ValueName = "PlugPlayServiceType"
Type = REG_DWORD
Data = 0x1


ValueName = "RequestedSystemResources"
Type = REG_RESOURCE_REQUIREMENTS_LIST
Data =
c8,00,00,00,05,00,00,00,00,00,00,00,0e,00,00,00,00,00,00,00,00,00,00,00,0
0,00,00,00,01,00,00,00,01,\
00,01,00,05,00,00,00,00,02,03,00,00,00,00,00,0a,00,00,00,0a,00,00,00,00,0
0,00,00,00,00,00,00,00,\
00,00,00,00,00,00,00,01,01,01,00,01,00,00,00,00,01,00,00,00,01,00,00,00,f
8,00,00,00,00,00,00,ff,\
```

```
f8,00,00,00,00,00,00,08,01,01,00,01,00,00,00,00,01,00,00,00,01,00,00,00,0
0,00,00,00,00,00,00,ff,\
ff,00,00,00,00,00,00,01,03,01,00,00,00,00,00,00,10,00,00,00,10,00,00,00,d
0,df,fe,00,00,00,00,ff,\
df,df,fe,00,00,00,00,08,03,01,00,00,00,00,00,00,10,00,00,00,10,00,00,00,0
0,00,00,00,00,00,00,ff,\
ff,ff,ff,00,00,00,00

KeyName = "HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Services\aic78xx\Enum"

ValueName = "NextInstance"
Type = REG_DWORD
Data = 0x1


ValueName = "Count"
Type = REG_DWORD
Data = 0x1


ValueName = "0"
Type = REG_SZ
Data = "Root\SCSIADAPTER\OEM0.INF&78XX40"


KeyName = "HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\SCSIADAPTER"

KeyName =
"HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\SCSIADAPTER\OEM0.INF&7
8XX40"

ValueName = "Problem"
Type = REG_DWORD
Data = 0x0


ValueName = "ConfigFlags"
Type = REG_DWORD
Data = 0x0


ValueName = "DeviceDesc"
Type = REG_SZ
Data = "Adaptec PCI SCSI Controller: AHA-290x/AHA-291x/AHA-294x/AHA-
394x/AHA-4944 or AIC-78xx(NT 4.0)"


ValueName = "Mfg"
Type = REG_SZ
```

```
Data = "(Additional models)"


ValueName = "Service"
Type = REG_SZ
Data = "aic78xx"


ValueName = "Driver"
Type = REG_SZ
Data = "{4D36E97B-E325-11CE-BFC1-08002BE10318}\0001"


ValueName = "StatusFlags"
Type = REG_DWORD
Data = 0x8


ValueName = "Class"
Type = REG_SZ
Data = "SCSIAdapter"


ValueName = "ClassGUID"
Type = REG_SZ
Data = "{4D36E97B-E325-11CE-BFC1-08002BE10318}"


ValueName = "FoundAtEnum"
Type = REG_DWORD
Data = 0x1


ValueName = "BaseDevicePath"
Type = REG_SZ
Data = "HTREE\ROOT\0"


KeyName =
"HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Enum\Root\SCSIADAPTER\OEM0.INF&7
8XX40\Control"

ValueName = "ActiveService"
Type = REG_SZ
Data = "aic78xx"


[AHA 2940UW Pro PCI Card.Files]
```

```
[AHA 2940UW Pro PCI Card.RegistryAdditions]
```

5.9.5 Implement the New SCSI KDF

1. Open Target Designer and select Component Management from the Tools menu.
2. Add the Adaptec 2940 KDF.
3. In Exercise 11 configuration, add the **AHA 2940UW Pro PCI Card** component.
4. A dialog box will pop-up listing the dependency on the **2940UW (AIC78xx.SYS)** component. Click Yes to add the driver **2940UW (AIC78xx.SYS)** component.
5. Add the Minimal OS component.
6. Build the Image.
7. Physically remove the IDE hard drive and CD-ROM drive.
8. Set the SCSI Adapter's BIOS to boot from the SCSI hard drive. (Please see your SCSI Adapter's instructions on how to set the boot device.)
9. Download the OS via Remote Recover to the SCSI NTFS partition. Keep the Remote Recover connection.
10. In Disk Administrator, select the target's SCSI drive partition and select Mark Active from the Partition menu.
11. Disconnect Target from Remote Recover, and reboot the system. NTE should boot from the NTFS / SCSI drive.

5.9.6 SCSI KDF Summary

This exercise introduced and implemented the Local method for building KDFs. It may not be possible to use the local method in all cases, but the advantage of having a standalone KDF can make it easy to keep all of the KDF pieces together.

5.10 Exercise 13 - Multiprocessor System

Windows NT was designed to support up to 32 processors. The two processor Workstation and four processor Server versions are available off the shelf. Windows NT Embedded supports both of these versions. The registry lists how many processors are in the system:

HKLM\System\ControlSet001\Session Manager\Licensed Processors. You can see that the Workstation Component supports up to two processors and the Server Components support up to four processors. Changing the value to increase the number of processors is illegal, and technically not advisable.

Multiprocessor operating systems support either asymmetric multiprocessing (ASMD) or symmetric multiprocessing (SMP). ASMP is a system where the OS runs on one processor and application (user) programs run on separate processors. SMP OS divides the kernel and application processes across all processors. NT was designed to support both architectures, but ships with support for SMP. There are strict rules to developing a multiprocessor board. Resource contention and other performance issues must be addressed for correct operation.

Multiprocessing systems are widely used in IT departments, but multiprocessing systems can also be used in a variety of embedded solutions: call-tracking servers with SQL databases, stock market tickers, routers, PBXs, switches, factory intranet servers, and multimedia devices.

Excercises 1 through 12 have supported single processor systems. The Standard HAL has been used in all configurations. The HAL allows the OS to be ported to other CPU and platform architectures. Windows NT has been ported to x86, MIPS, PowerPC, and the Alpha processor (Windows NT Embedded only supports the x86 architecture). There are a number of HALs that come with Windows NT.

HAL Component	Description
486c	Standard PC with C-Step 486
APIC	MPS Unitprocessor PC
AST	AST Manhattan SMP
Cbus	Corollary C-bus Architecture
Cbusm	Corollary C-bus icro Channel Architecture
MCA	IBM PS/2 or Other Micro Channel-based PC
MPS	MPS Multiprocessor PC
MPSM	Micro Channel Multi Processor PC
NCR	NCR System 3000 Model 3360/3450/3550
Olivetti	Olivetti LSX5030/40
Standard	Standard PC (single processor HAL)
SystemPro	Compaq SystemPro Multiprocessor or 100% Compatible
Wyse	Wyse Series 7000i Model 740MP/760MP

Table 5.1: Supported HALs in Windows NT

Windows NT allows board and PC manufacturers to supply their own NT HAL. If you're not sure which HAL to choose for your target, install the standard version of Windows NT 4.0 + SP5 to auto-detect the correct HAL.

5.10.1 Required Hardware

To show the difference between a single processor system and a multiprocessor system, you will need two platforms for this exercise. The Intel Pentium III/ 840 Development kit and the Intel Pentium III Evaluation Board are the platforms of choice for this exercise. These Intel evaluation platforms allow you to test applications on full-featured PC platforms. The Intel Pentium III Evaluation Board features the mobile Pentium III module, and comes with a Penitum III 500MHz CPU and 32 MB SDRAM.

Because of the demand for more computing power, PC board manufacturers are producing more multiprocessing platforms. To support applied computing developers, Intel has created the Pentium III / 840 Development kit. The development kit is based on the Multiprocessor Specification (MPS) v1.4 and supports the WTX specification. The evaluation board contains two Pentium III 600 MHz processors, 128 MB RDRAM, and the 840 chipset.

The MPS provides guidelines to build multiprocessor systems that are compatible with the standard PC architecture. The specification allows for support of operating systems from MS-DOS that run on a single processor to Windows NT that take advantage of multiple processors.

There are alternative platforms available with the 440BX, 440GX, 820, or 840 chipsets: The exercise is not limited to the Intel Development Kit.

Company	Website	Model	Chipset
ASUS	www.asus.com	P3C-D	i820
Aopen	www.aopen.com	DX6G Plus	440GX
		DX2G Plus	440GX
Intel	www.intel.com	Pentium III/840 DEV Kit	i840
Iwill	www.iwill.net	DCA200	i840
		DBD100	440BX
		DBL100	440BX

MSI	www.msicomputer.com	MS-6135	440GX
SuperMicro	www.supermicro.com	S2 DME	840
		S2 DM3	840
		PIII DME	840
		PIII DM3	840
		PIII DM6	840
		P6DBU	440BX
		P6DBS	440BX
		P6DBE	440BX
Soltek	www.soltek.com	SL-68A	440BX
Tyan	www.tyan.com	S2257 Thunder 2400	i840
		S1837DLUN	440BX
Tekram	www.tekram.com	P6B40D-5	440BX

Table 5.2: Dual Pentium III Platform Resources. More boards can be found at www.motherboards.com

5.10.2 Exercise Overview

Exercise 13 introduces some new components and features, and reinforces previous lessons. In this exercise we will create an image for a multiprocessor system and compare the performance to a single processor system. In an ideal world two processors would mean twice the computing power, but this is not the case. Applications have to be designed to fully support more than one processor. Windows NT will distribute the processes and threads between each processor in order to get the best possible performance. There are several tools that can help measure performance and control processes.

- Task Manager - Allows you to monitor and control what is running on the system. This provides a dynamic view of CPU and memory usage, and list of processes that can be broken down into various forms of information. Applications can be started, and applications and individual processes can be stopped.
- Performance Monitor - Performance Monitor is a graphical tool for measuring the performance of a computer or a computer that is on the network. You can view processes, threads, memory, and cache. Each of these measurements have

associated counters that provide information on device usage, throughput, and internal congestion. Performance Monitor allows you to chart, log, set alerts, and create reports on a system's activity.

- CPU Stress Test - This creates single to multiple threads that can be set at different priorities. This tool creates heavy usage on the processor for evaluating a system's performance.

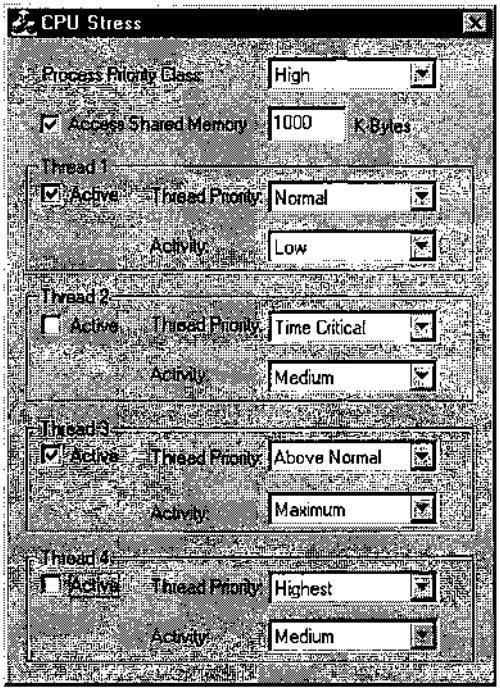


Figure 5.27: CPU Stress Utility

The CPU stress test is part of the Windows NT Resource kits and is not in the NT Embedded distribution CDs. You may not have access to this tool, so another application is needed to show the basic differences between single and multi processor systems. The TDSD doesn't contain all of the features, applications, and capabilities found in NT (i.e., games, multimedia, etc.), but the major components are listed in the database. NTE CD-1 contains the same applications as the regular distribution. You can create KDFs that will extract the files from the i386 directory. To make the exercise a little more fun, we will use 3D Pinball to act as the test application.

5.10.3 Creating the Pinball KDF

The local method will be used to create the KDF. The KDF information will be extracted from the host machine, so you must have Pinball loaded on your development machine. Pinball can be loaded via the Control Panel->Add/Remove Programs-> Windows Setup->Games. Pinball will be installed in the \Program Files\Windows NT\Pinball directory.

1. Open Component Designer, and add a new Capability called GAMES under the Applications capability hierarchy path.
2. Create a new component called Pinball under the Applications->Games hierarchy path.

   Component Name: **Pinball**
   Description: **Pinball by Maxis.**
   Hierarchical path: **Applications->Games**

3. Modify the Component Definition file properties and comments:

   Component Name: **3D Pinball**
   Vendor: **Maxis via Microsoft**
   Description: **Pinball**

   Comments: **Copyright © 2000 ACME Corp.**

4. Right click on Files under Pinball and select "Drag from Explorer".
5. Go to the \Program Files\Windows NT\Pinball directory and copy the entire contents to the KDF.
6. Perform a static analysis on Pinball.EXE using Scanbin (see Chapter 4 for more details). All of the DLLs needed by Pinball are part of every OS combination except WINMM.DLL.
7. Right click on Files and select Drag from Explorer.
8. Locate WINMM.DLL under \winnt\System32, and drag the file to the KDF.
9. The structure for the KDF is not complete, and yet the files will be extracted from NTE CD-1 and not from your local subdirectories. Each file must have a \i386 as the source path. To speed up the process, click to highlight the top most file in the list, scroll down to the bottom of the file list, hold the shift key down, and select the last file. All of the files should be highlighted.

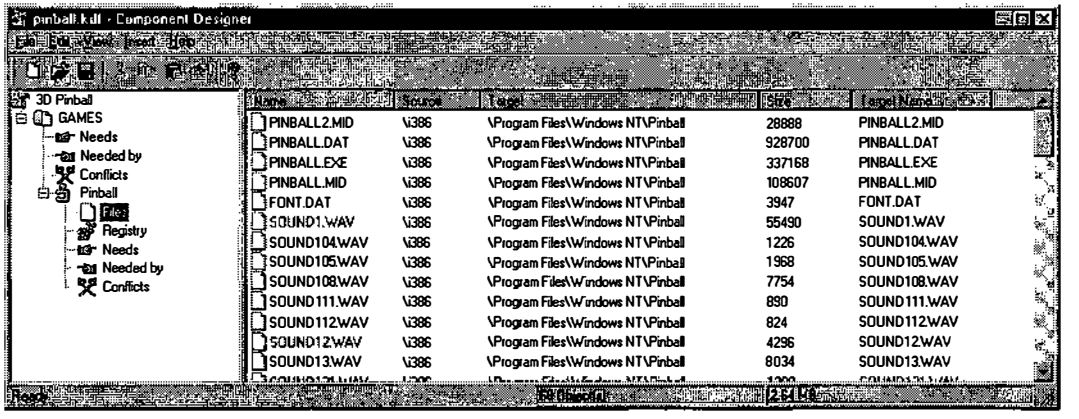


Figure 5.28: Pinball KDF

10. Go to Edit->Properties, change the source path to \i386, and click OK. All of the files should have a \i386 as the source path.
11. Save the KDF as PINBALL.KDF.
12. In Target Designer, use Component Management to ADD the Pinball KDF to the TDSD.

The KDF should look like the following under WordPad:

```
;Copyright (c) 2000 ACME Corp.

[Version]

Signature = "$Windows NT$"
KdfVersion = 1


[Header]

Name = 3D Pinball
Version = 1.0
ReleaseDate = 03/16/2000
Vendor = Maxis via Microsoft
OS = WINNT
OSVersion = 4.0
```

```
Platform = i386
Repository = "%WINNT%"
Description = "Pinball Game"
CodePage = 1252


[Capabilities]

1 = "GAMES", %ROOT%\Applications


[Components]

1 = "Pinball", "Pinball by Maxis", 1


[CapabilityDependencies]


[CapabilityComponentDependencies]


[ComponentCapabilityDependencies]


[ComponentDependencies]


[Directories]

1 = "\Program Files\Windows NT\Pinball"
3 = "\i386"
2 = "\winnt\system32"


[Pinball.Files]

"PINBALL2.MID", 3, 1, 28888, "PINBALL2.MID"
"PINBALL.DAT", 3, 1, 928700, "PINBALL.DAT"
"PINBALL.EXE", 3, 1, 337168, "PINBALL.EXE"
"PINBALL.MID", 3, 1, 108607, "PINBALL.MID"
"FONT.DAT", 3, 1, 3947, "FONT.DAT"
"SOUND1.WAV", 3, 1, 55490, "SOUND1.WAV"
"SOUND104.WAV", 3, 1, 1226, "SOUND104.WAV"
"SOUND105.WAV", 3, 1, 1968, "SOUND105.WAV"
"SOUND108.WAV", 3, 1, 7754, "SOUND108.WAV"
"SOUND111.WAV", 3, 1, 890, "SOUND111.WAV"
"SOUND112.WAV", 3, 1, 824, "SOUND112.WAV"
"SOUND12.WAV", 3, 1, 4296, "SOUND12.WAV"
```

```
"SOUND13.WAV", 3, 1, 8034, "SOUND13.WAV"
"SOUND131.WAV", 3, 1, 1290, "SOUND131.WAV"
"SOUND136.WAV", 3, 1, 19282, "SOUND136.WAV"
"SOUND14.WAV", 3, 1, 3002, "SOUND14.WAV"
"SOUND16.WAV", 3, 1, 1046, "SOUND16.WAV"
"SOUND17.WAV", 3, 1, 2090, "SOUND17.WAV"
"SOUND18.WAV", 3, 1, 3986, "SOUND18.WAV"
"SOUND181.WAV", 3, 1, 27472, "SOUND181.WAV"
"SOUND19.WAV", 3, 1, 5230, "SOUND19.WAV"
"SOUND20.WAV", 3, 1, 8650, "SOUND20.WAV"
"SOUND21.WAV", 3, 1, 9194, "SOUND21.WAV"
"SOUND22.WAV", 3, 1, 7376, "SOUND22.WAV"
"SOUND24.WAV", 3, 1, 12106, "SOUND24.WAV"
"SOUND240.WAV", 3, 1, 14600, "SOUND240.WAV"
"SOUND243.WAV", 3, 1, 20712, "SOUND243.WAV"
"SOUND25.WAV", 3, 1, 25704, "SOUND25.WAV"
"SOUND26.WAV", 3, 1, 7306, "SOUND26.WAV"
"SOUND27.WAV", 3, 1, 20242, "SOUND27.WAV"
"SOUND28.WAV", 3, 1, 8650, "SOUND28.WAV"
"SOUND29.WAV", 3, 1, 10364, "SOUND29.WAV"
"SOUND3.WAV", 3, 1, 22858, "SOUND3.WAV"
"SOUND30.WAV", 3, 1, 22570, "SOUND30.WAV"
"SOUND34.WAV", 3, 1, 1520, "SOUND34.WAV"
"SOUND35.WAV", 3, 1, 19498, "SOUND35.WAV"
"SOUND36.WAV", 3, 1, 33848, "SOUND36.WAV"
"SOUND38.WAV", 3, 1, 13024, "SOUND38.WAV"
"SOUND39.WAV", 3, 1, 28282, "SOUND39.WAV"
"SOUND4.WAV", 3, 1, 16626, "SOUND4.WAV"
"SOUND42.WAV", 3, 1, 29140, "SOUND42.WAV"
"SOUND43.WAV", 3, 1, 22796, "SOUND43.WAV"
"SOUND45.WAV", 3, 1, 9770, "SOUND45.WAV"
"SOUND49.WAV", 3, 1, 1876, "SOUND49.WAV"
"SOUND49D.WAV", 3, 1, 3330, "SOUND49D.WAV"
"SOUND5.WAV", 3, 1, 3180, "SOUND5.WAV"
"SOUND50.WAV", 3, 1, 12074, "SOUND50.WAV"
"SOUND528.WAV", 3, 1, 8932, "SOUND528.WAV"
"SOUND53.WAV", 3, 1, 9022, "SOUND53.WAV"
"SOUND54.WAV", 3, 1, 18250, "SOUND54.WAV"
"SOUND55.WAV", 3, 1, 21890, "SOUND55.WAV"
"SOUND560.WAV", 3, 1, 29004, "SOUND560.WAV"
"SOUND563.WAV", 3, 1, 24192, "SOUND563.WAV"
"SOUND57.WAV", 3, 1, 30502, "SOUND57.WAV"
"SOUND58.WAV", 3, 1, 3408, "SOUND58.WAV"
"SOUND6.WAV", 3, 1, 4376, "SOUND6.WAV"
"SOUND65.WAV", 3, 1, 17676, "SOUND65.WAV"
"SOUND68.WAV", 3, 1, 32402, "SOUND68.WAV"
"SOUND7.WAV", 3, 1, 26442, "SOUND7.WAV"
"SOUND713.WAV", 3, 1, 14592, "SOUND713.WAV"
"SOUND735.WAV", 3, 1, 27268, "SOUND735.WAV"
```

```
"SOUND8.WAV", 3, 1, 2102, "SOUND8.WAV"
"SOUND827.WAV", 3, 1, 47230, "SOUND827.WAV"
"SOUND9.WAV", 3, 1, 20098, "SOUND9.WAV"
"SOUND999.WAV", 3, 1, 6742, "SOUND999.WAV"
"table.bmp", 3, 1, 339178, "table.bmp"
"wavemix.inf", 3, 1, 2686, "wavemix.inf"
"winmm.dll", 3, 2, 149264, "winmm.dll"


[Pinball.RegistryAdditions]
```

5.10.4 Test Pinball on a Single CPU System (Intel Pentium III Evaluation Board)

To show the differences in performance and operation, the first run will be on Single CPU system (Intel Pentium III Evaluation Board). Task Manager and Performance Monitor will be used to measure the CPU usage while Pinball is running.

1. Create a new configuration and name it Exercise12.
2. Select the following components:

Capability	Component	Size	Check Box
System / Platform / SystemRole	Work Station	0	
System / Platform / User Identification	Automatic Login	0	
System / Platform / Core OS	Minimal OS	74.32MB	
System / Platform / HAL	Standard HAL	958KB	
System / Platform / Virtual Memory	Default Page File		
System / Devices / Storage / Floppy Disk	Floppy	18KB	
System / Devices / Storage / Fixed Disk	EIDE	0	
System / Devices / Storage / Fixed Disk	SCSI Disk	15KB	
System / Devices / Storage / CD-ROM	EIDE CD-ROM	0	
System / Devices / Storage / CD-ROM	SCSI CD-ROM	22KB	

System / Devices / Storage / File Systems	CDFS	60KB	
System / Devices / Storage / File Systems	FAT	139KB	
System / Devices / Display / Display Adapters	Chips Video Accelerator	0KB	
System / Devices / Display / Display Drivers	Chips Video Accelerator Driver	183KB	
System / Devices / Keyboard / Keyboard Layout	US Keyboard Layout	9KB	
System / Devices / Keyboard / Keyboard Drivers	PC/AT Keyboard	29KB	
System / Devices / Keyboard / Keyboard Input Local	English (US) Input Locale	7KB	
System / Devices / Mouse	Logitech	29KB	
System / Devices / BUS Devices SCSI / SCSI Adapters	EIDE SCSI Adapter	0	
System / Devices / BUS Devices SCSI / SCSI Drivers	EIDE SCSI Driver	27KB	
System / Devices / Serial Ports	COM1	0	
System / Windows Services / OLE and COM	OLE/COM	2.18MB	
System / Management / Management Applications / Performance Management and Troubleshooting / Performance Monitoring	Performance Monitor	255KB	
System / Management / Management Applications / Performance Management and Troubleshooting / Performance Monitoring	Task Manager	106KB	
System / Management / Management Applications / Performance Management and Troubleshooting / Performance Monitoring	Windows NT Diagnostics	195KB	

System / Management / Management Services	Performance Counters	359KB	
System / Desktop Settings / Fonts	Arial Font	612KB	
System / Desktop Settings / Fonts	MS Sans Serif	151KB	
System / Desktop Settings / Fonts	Times font	738KB	
System / Desktop Settings / Application Links	Explorer Links	36KB	
System / Shared System Components	Display	25KB	
System / Shared System Components	Keyboard Drivers	9KB	
System / Shared System Components	Keyboard Locale	135KB	
System / Shared System Components	Mouse Common	9KB	
System / Shared System Components	SCSI	58KB	
System / Shared System Components	Serial Ports	44KB	
System / Shared System Components	Virtual Memory	0	
Applications / Shell	Explorer Shell	0	
Applications / Utilities	Explorer	322KB	
Applications / Accessories	Calculator	124KB	
Applications / Accessories	Paint	1.44MB	
Applications / Accessories	Wordpad	2.40MB	
Applications / Games	Pinball	2.64MB	

3. Build and download the OS to the AAEON platform, Intel Pentium III Evaluation Board, or a single CPU system.
4. Start Task Manager, and click on the Performance Tab.
5. Start Performance Monitor, add % Processor Time to chart (see Performance Monitor Help for more details).
6. Start Pinball, and play the game (at first, one ball only). As you play, the CPU usage goes to 100 % on both activity monitors.

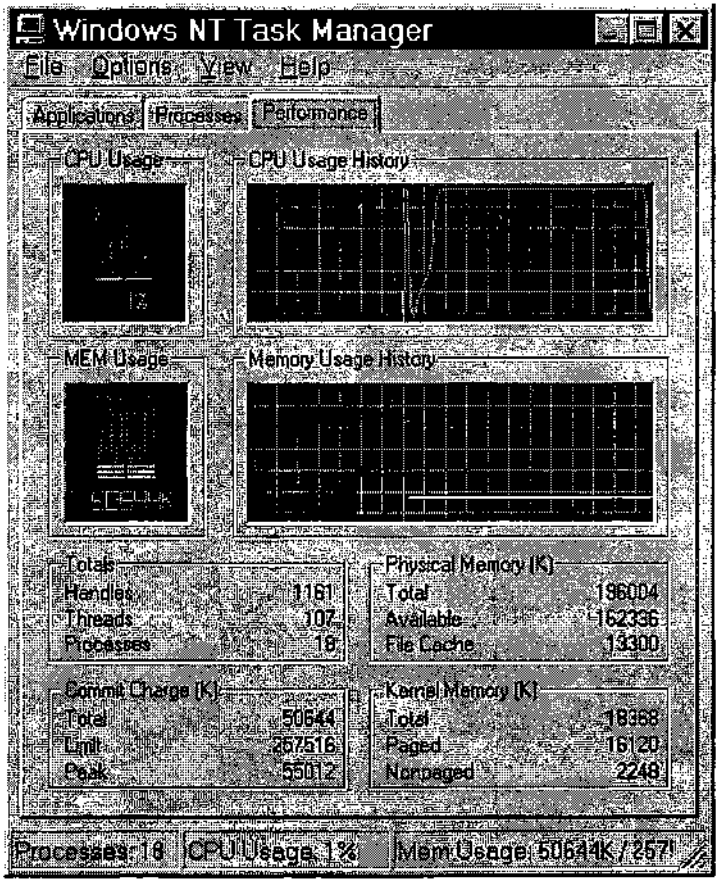


Figure 5.29: Task Manager Displaying a Single Processor Usage History

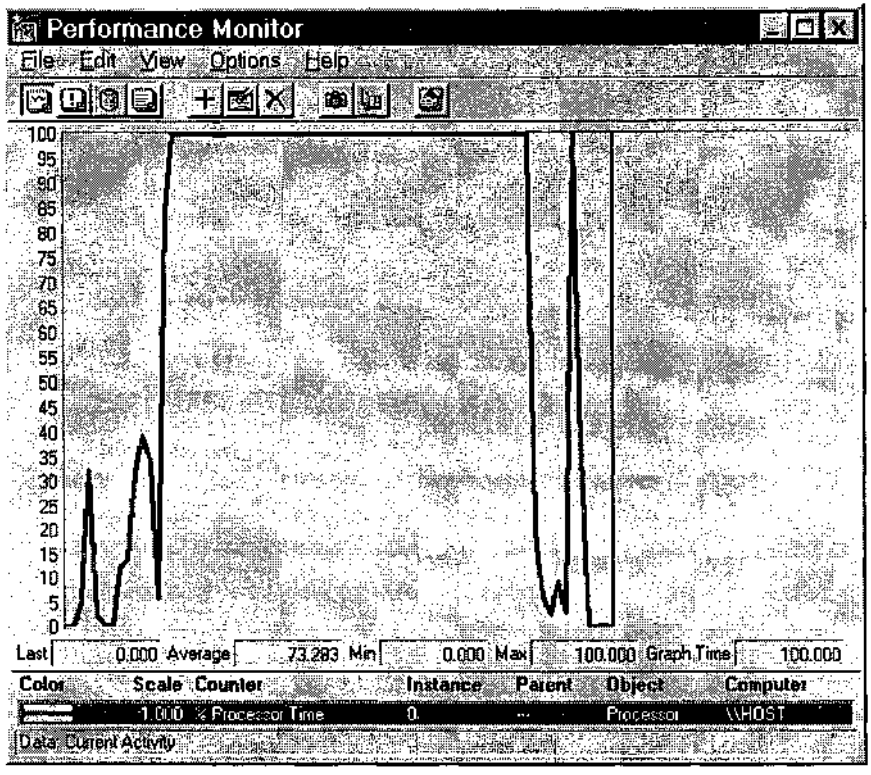


Figure 5.30: Performance Monitor Displaying CPU Usage

You can always add more counters to Performance Monitor to track process information, threads, and memory usage. The more processes running the more resources the system will require.

5.10.5 Test Pinball on a Multiprocessor System (Intel Pentium III / 840 Development Kit)

Now let's try the same test on a dual processor system. Perform the following:

1. Make a copy of Exercise13a, and rename it to Exercise13b
2. Add the MPS HAL, and remove the Standard HAL.
3. Build and download the image to the Dual processor target.

4. Open NT Diagnostics and select the System tab. You can see that there are two CPUs running in the system.
5. Close Windows NT Diagnostics.

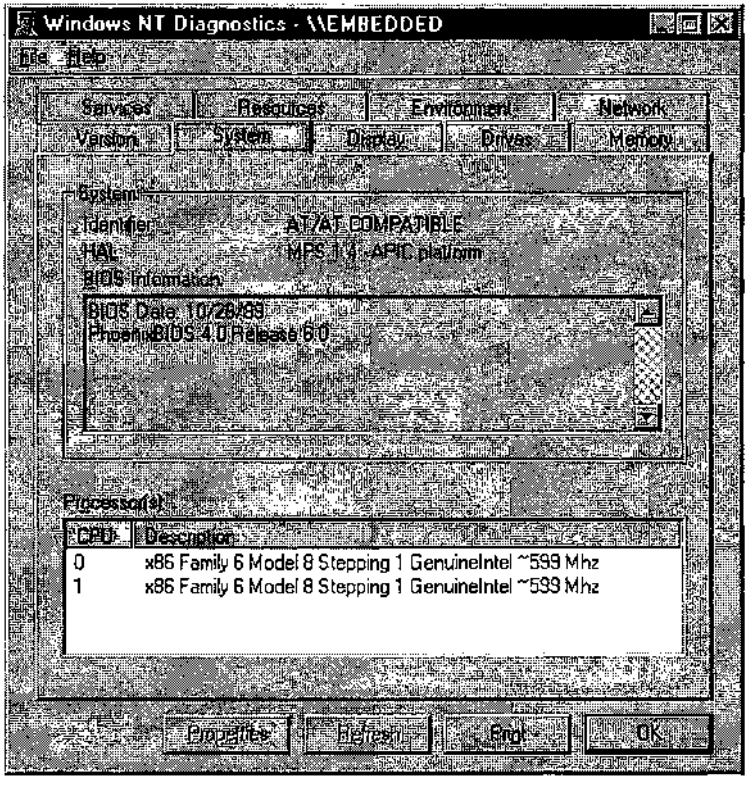


Figure 5.31: NT Diagnostics

6. Start Task Manager, and click on the Performance Tab. Task manager automatically displays the performance for both CPUs.
7. Start Performance Monitor, add % Processor Time for both CPUs to chart (see Performance Monitor Help for more details).
8. Start Pinball, and play the game (at first, one ball only). The graphs show that the kernel is distributing the Pinball process between both CPUs. Neither CPU runs at 100% for very long.

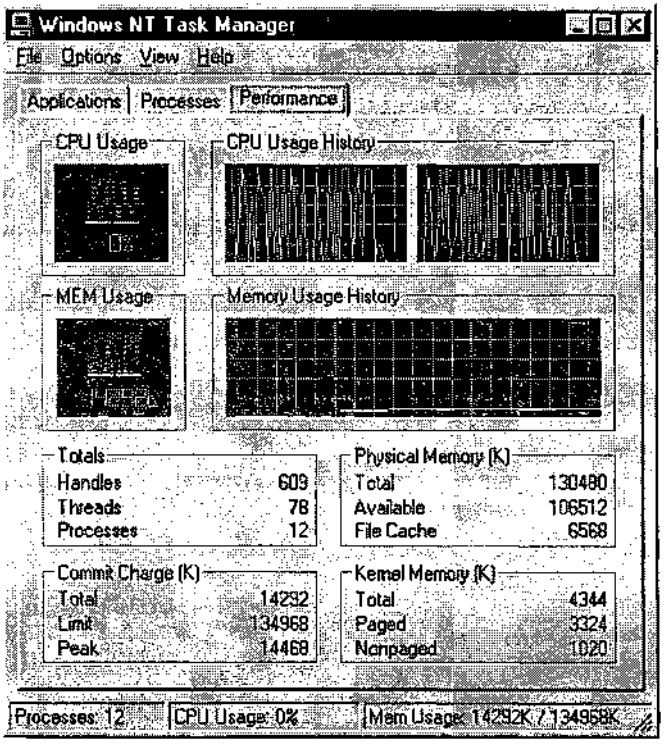


Figure 5.32: Task Manager Showing CPU History for Two Processors

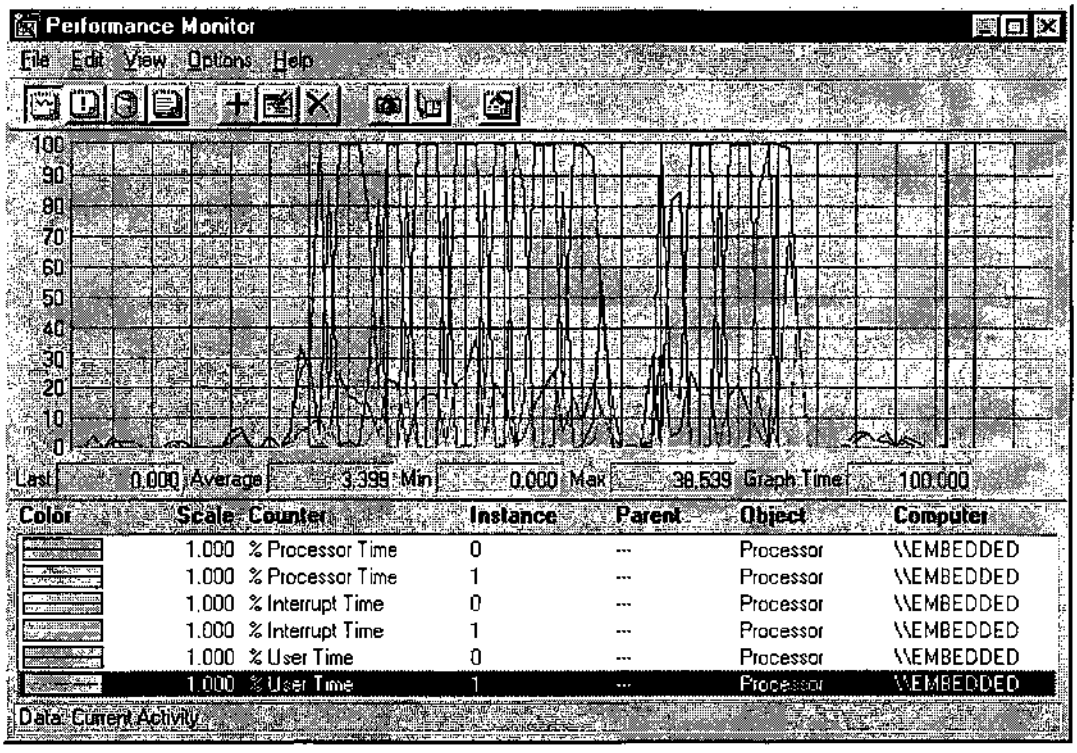


Figure 5.33: Performance Monitor Displaying Different Counters for each Processor Instance

Try running other applications like multimedia programs to stress the dual CPU platform. Use Performance Monitor to test your applications and optimize your system.

Task Manager can be used to control which processor the process can run.

9. Click on the Processes tab.
10. Select the Pinball.EXE, right-click to bring up the selection menu, and select Affinity.
11. Change the processes to CPU0.

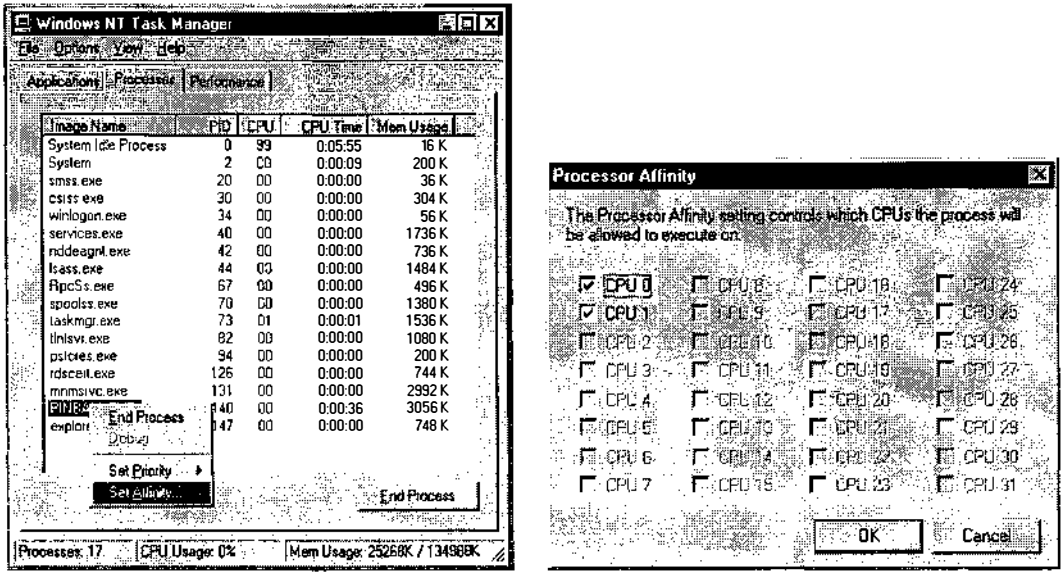


Figure 5.34: Setting the Processor Affinity for Pinball

12. In Task Manager, select the Performance tab.
13. Start playing pinball again. Observe how CPU0 is 100% out and CPU1 is available to run more processes. The CPU usage dropped to 0% while taking the screen shot, but as you can see from the graph, CPU0 was at 100%.

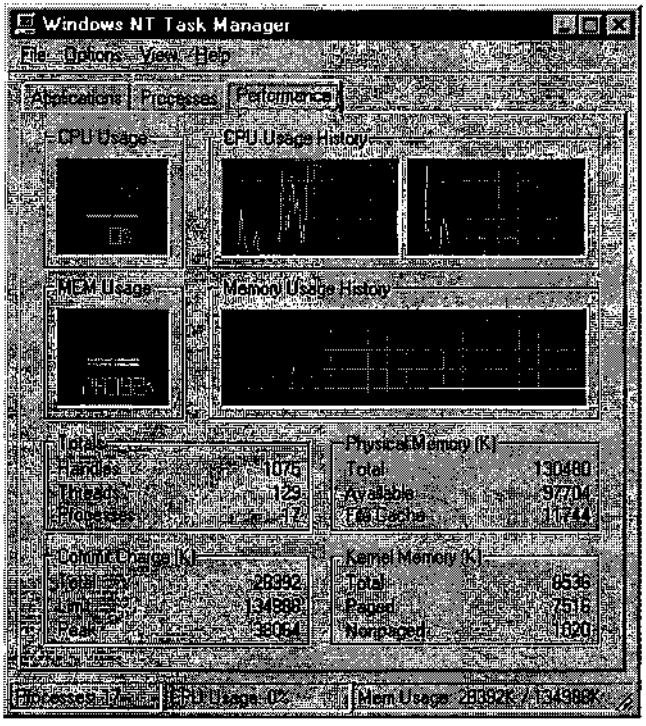


Figure 5.35: Task Manger Displaying Pinball Set to Run on Only CPU0

You can go back to change the affinity to CPU1. The affinity mask can be set within the program using the SetProcessAffinityMask and SetThreadAffinityMask functions.

5.10.6 Summary

This exercise introduced more features and components. Windows NT supports SMP for up to four processors in a server system and two processors on a workstation system. Another KDF was created for an application that resides in the i386 directory of NT CD-1. Task Manager, Performance Monitor, and Pinball were used to demonstrate the SMP capability of a Windows NT.

Pinball provided a simple mechanism to create threads and show how the kernel distributes the process between two CPUs. The CPU Stress utility provides a more flexible method to find bottlenecks in a system. The real stress test is with your own application.

Chapter 6 DEPLOYMENT STRATEGIES

Now that you have built and tested your image, you will need a deployment strategy for production and quality assurance. The deployment method that you choose is up to you and your company. The resources listed here are for your convience, and are not endorsements from either the author or Annabooks.

6.1 System Tests

Just running the application for many OEMs is enough to prove that the system is ready for manufacturing. Yet most ISO companies need to have a more rigorous test suite for QA. The testing methodology is left to you.

Visual Test from Rational Software Corp. (http://www.rational.com) can aid in the development of the applications. For a total system test that is non-intrusive, Btree Systems, Inc. (http://www.btree.com) offers their Validor Pro that remotely tests a system much like an end user.

6.2 Manufacturing - Disk Duplicators, SIDs, Network Addresses, and Computer Names

Target Designer allows you to preset most of the system parameters before an image is built and installed. Once you download the image to one hard drive, you can duplicate that drive image to other hard drives.

Networking is the biggest issue in manufacturing an NT system. Computers on a network must have a unique computer name, network address, and Security Identification number (SID).

6.2.1 Security ID (SID)

One of the biggest issues is the cloning of PIDs and SIDs in a networked environment. The PID is also known as the Certificate of Authenticity (COA) number. For NT Embedded, Microsoft doesn't require that each number be placed in each unit, but for accounting and tracking purposes the PID should be entered into the registry.

If multiples of your products can be attached to the same network, you need to have unique SIDs for each system in order for security to be effective. NTE comes with a cloning component to help with the duplication of an NTE image. When the Cloning Component is included in the image, the SETUPCL.EXE will execute and run once before the system completes the first login. After the first login, SETUPCL.EXE is removed from the Registry Entry to keep from executing again. You can verify that the Cloning component is working correctly by making copies of the same image. Boot each image for the first time and look at the HKEY_USERS SID. The HKEY_USERS SID changes when downloading the same image several times regardless of whether Standard or Minimal OS is implemented.

One item to note: if the cloning component is selected and you switch from the Standard OS to Minimal or Minimal to Standard, the registry entry HKLM\System\ ControlSet001\control\Session Manager\Boot Execute doesn't get updated with the setupcl.exe in the multi-data string. Only when the Cloning Component is removed and then re-added does the registry reflect the correct data value.

6.2.2 Computer Names and Network Addresses

Using Browser services, a computer is identified by its computer name. The computer name is set in one of the System Role components or Network Membership components before the image is built. The end user can change the computer name via Network control panel. You must provide other options for an end user to change the computer name if the network control panel applet is not available or the system is headless.

A Win32 program can be written to change the computer name from within the registry. The registry location for the computer name is:

HKLM\System\ControlSet001\Control\ComputerName\ComputerName\Computer Name

Changes to the TCP/IP address must also be changed by the end user in order for the system to be attached to a network. The TCP/IP address registry settings are found in the network adapter card registry settings:

HKLM\System\ControlSet001\Services\<net card>1\Parameters\TCP/IP

A change to the computer name requires that the system be shut down and restarted. The Shutdown component should be included to allow the end user to provide a clean shutdown and restart.

6.2.3 SystemPrep Tool

The SystemPrep tool is also available for use with Windows NT Embedded. The SytemPrep tool allows you to set up a system that allows the end user to walk through the standard NT setup screens. The end users enter in ownership information, computer names, and COA numbers. The SystemPrep automatically creates a unique SID for the system.

In order to use the SystemPrep tool, you will have to make some changes to the image after downloading to the target.

- Remove the password for the Administrator account.
- Remove the password for the Automatic login.
- Replace the PID (or COA) number found in HKLM\Software\Microsoft\Windows NT\CurrentVersion\ProductID with an NTE OEM COA number. The SystemPrep tool will fail with the default PID provided in Target Designer.
- Include the shutdown component for the end user to safely shutdown and restart the system.

Once these modifications are completed, you can run the SystemPrep tool and follow the on-screen instructions.

6.2.4 Write Protected Boot Partitions and El Torito CD-ROMs

There are work-arounds for you to change computer names and SIDs. These values cannot be changed when a boot partition is write-protected or the system is booting from an El Torito CD ROM. OEMs must build a unique image for each system when it comes to write-protected drivers and El Torito CD-ROMs.

6.2.5 Duplication

The method of duplication is up to you. There are some software duplication packages that feature the ability to change PIDs, SIDs, and computer names. There are a number of hard drive duplicators and various software packages available.

Software Duplicators

Company	Product	Website
Symantec	Norton Ghost	www.symantec.com
PowerQuest	Drive Image Pro	www.powerquest.com
Innovative Software	ImageCast IC3	www.innovativesoftware.com
Altiris	RapiDeploy	www.altiris.com

Hardware Duplicators

Company	Website
Douglas Electronics inc.	www.douglas.com
Intelligent Computer Solutions	www.ics-iq.com/main.htm
Lauran Electronics	www.lauran.com
Nucleas – CD-Dupe	www.cdupe.com
Kintronics	www.kintronics.com
Greystone Peripherals	www.grystone.com

6.2.6 Manufacturing Summary

The manufacturing of the Windows NT system is left up to the OEM. Allowing the end user to configure the system for network access in headless, minimal configurations must be designed into the NT image. Here are some guidelines:

- Use a DHCP address for the shipping unit, and the end user can connect the computer to a DHCP server and access the machine either directly or over remote administration.
- Make sure the computer name is unique; use your company's name as a computer name to prevent conflicts with other possible computer names.
- Provide a text mode application to allow for changes to the network settings, computer names, etc. over Serial/Modem remote administration.

6.3 Exercise 14 - System Cloning

This exercise will demonstrate the functionality of the System Clone component.

6.3.1 Image without System Cloning

1. Copy Exercise1 and rename the copy of Exercise1 to Exercise14
2. Build and download the OS to the target.
3. Using REGEDT32, write down the SID found in the HKEY_USERS SID.
4. Reformat the Target Drive, and download the same image.
5. Using REGEDT32 to view HKEY_USERS SID, the SID should be the same as before.

6.3.2 Image with System Cloning

6. Now, add the System Cloning component to the configuration.
7. Build and download the OS to the target.
8. Using REGEDT32, write down the SID found in the HKEY_USERS SID.
9. Reformat the Target Drive, and download the same image.
10. Using REGEDT32, the HKEY_USERS SID should be different. If you download the same image to different hard drive, or ghost the hard drive, the SID will be different for each copy.

6.4 Source Control

An embedded device can last as long as 10 years. It is important for most companies to be able to support a product for the extended life cycle. In all of that time people can come and go within an organization. Backing-up KDFs, CFGs, and drive images becomes an important part of the deployment strategy. There are many tools to help with source control: Microsoft's Visual Source Safe and Intersolv's PVCS are two popular packages on the market.

TIP: If you need to recover a system, make sure that the KDFs are installed before importing CFG files. You will get an error if a component is missing that the CFG depends on.

6.5 Field Update Issues

A typical Windows NT installation expects that the system will have a floppy (you cannot substitute a SuperDisk because NT setup will crash on the second disk), CD-ROM, and hard drive in the system. This is a limiting feature for embedded systems, especially when it comes to system recovery from backup media. From the previous exercises, we see that there can be a variety of ways to implement NTE. How field updates will take place must be designed into the device at the beginning of the system's development. There are tools that can help some general cases, but some situations may be more limiting.

6.5.1 Easy Restore

PowerQuest has developed EasyRestore from their patented DriveImage technology product. EasyRestore allows you to create a back-up CD-ROM or ZIP disk containing an exact image of the hard drive(s) of your target system. EasyRestore can create backup media regardless of the size of the hard drive, the number of partitions, or the file system. The following are the general steps for using EasyRestore with your target system:

1. Download the OS to your target and set up the target as it will be delivered to your customer.
2. EasyRestore is a DOS-based tool system. You use the CREATE.EXE application to create the disk image. PowerQuest provides a DOS boot floppy with all of the tools necessary to create an image.
3. As you create an image you will need to save the image to an alternative storage location. For example, if you are creating a backup image of your target's C drive, you must create the image on a D drive, ZIP disk, or a network drive (EasyRestore uses DR-DOS, so WorkGroup Add-on may or may not work).
4. Once the image is created, you create the backup media with the following options:

    A. **OPTION 1:** Create a bootable restore floppy with the hard drive image file located on the CD-ROM.

    B. **OPTION 2:** Create a bootable floppy with only the CD-ROM driver with all other files located on the CD-ROM.

C. **OPTION 3:** Create a bootable CD-ROM with all files located on the CD-ROM.

5. Your boot option will contain a copy of DOS and the PQER.EXE command which will restore the image to the hard drive thus returning the platform to a known good state.

EasyRestore provides other features such as splash screens and various command line settings. PowerQuest also offers other tools for software deployment and a changing partition sizes of a hard disk image. EasyRestore tools are free, but there is a run time royalty involved. Annasoft Systems has arranged a special price for customers who sign up for licensing through Annasoft.

6.5.2 Worst Case Situation

Extensibility and upgradeability must be accomodated in the early stages of the platform design. The size of NT requires field upgrades to have higher bandwidth. Section 6.3 discussed recovery media for NT in general, but there are cases where EasyRestore is not the solution. Bootable CD-ROMs can be easily updated by just replacing the CD, but a headless system without removable media presents the worst-case problem.

There are two types of upgrades: application upgrade and total system upgrades. Application upgrades can be handled by direct disk installation or over a remote administrative connection. A total system upgrade will require the storage medium to be reformatted to replace the OS, but in a headless system without a floppy, CD-ROM, ZIP, or other removable media, system updates can be a problem.

There are three options for this worst-case scenario:

- Have a dual OS boot - The system can either boot to Windows NT Embedded or MS-DOS to run a utility like Remote Recover to download the OS over Ethernet. The NTE system files will have to exist on a second partition so that the second partition can be re-formatted to remove the old NTE installation.
- Physically replace the hard drive - Open the box, disconnect cables, and unscrew the hard drive.
- Add removable media in the form of a PCMCIA drive - simple to eject the flash or PC Card storage medium.

Field upgrade methodologies for your system are up to you. Keep in mind the following:

- Type of person conducting the upgrade: will it be a field technician, or the end user?

- Hardware limitations: the marketing requirements for a system may not include removable media or an Ethernet connection.
- Reliability: systems with the Write Filter implementation cannot easily be updated.
- Time and Cost: down time can turn a profit into a loss.

6.6 High Availabitiy

High Availability (HA) systems are becoming more of a requirement in a connected world. Voice over IP and e-Commerce are driving forces for HA systems. Systems in which Windows NT appears unreliable usually have one or more of the following characteristics:

- Poorly written third party applications and device drivers
- Abuse by the user (the infamous unfriendly shutdown)
- Hardware incompatibilities or failures

Typical applied computing and embedded systems run a single application with little user intervention, which enhances the stability of the Windows NT system. NTE's embedded features such as error recovery, auto-reboot, write-filter, and CD-boot, give you important tools for implementing a stable NT platform. Hard drive corruption is an issue if users don't safely shut down a system. The embedded features can help prevent disk corruption and provide flash wear leveling. Below is a recommended list of reliable boot partitions using the embedded features:

Reliability	Boot Drive Properties	Drive	NTE Components to Select
1	EL Torito	CD-ROM	El Torito, Write Filter, No Page file
2	Boot partition is write protected	Hard drive, Flash drive	Write Filter, No page file
3	RAID (Server only)	Hard drive	Fault Tolerance
4	NTFS / No page file	Hard drive, Flash drive	NTFS, No Page File
5	FAT / No page file	Hard drive, Flash drive	No Page File

Chapter 7 RESOURCES

7.1 Third Party Tools

The book has referenced several third parties that offer products that can help in the development of Windows NT Embedded systems. The author and Annabooks do not endorse any of these products, but provide this small list to help you get started locating helpful resources more easily.

Remote Recover/RegMon/FileMon

Winternals	www.winternals.com

Hardware Disk Duplicators

Douglas Electronics Inc.	www.douglas.com
Intelligent Computer Solutions	www.ics-iq.com/main.htm
Lauran Electronics	www.lauran.com
Nucleas - CD-Dupe	www.cdupe.com
Kintronics	www.kintronics.com
Greystone Peripherals	www.grystone.com

Software Disk Duplicators

Norton Ghost	www.symantec.com/sabu/ghost/fs_ghost.html
PowerQuest	www.powerquest.com
Innovative Software	www.innovativesoftware.com
Altiris	www.altiris.com

Disk Duplication Services

Software Packaging Associates	www.softpack.com

Princeton Diskette	www.princetondiskette.com
Application Installation Tools	
InstallShield Software Corp.	www.installshield.com
Miscellaneous Tools	
WinOnCD	www.cequadrat.com
System Internals	www.systeminternals.com
Software Shelf International	www.softwareshelf.com
Sunbelt Software	www.sunbelt-software.com
WinAbility	www.winability.com
Real-Time Extensions	
RadiSys	www.radisys.com
Imagination Systems	www.imagination.com
USB Support	
Phoenix	www.phoenix.com
BlueWater Systems	www.bluewatersystems.com
BIOS	
General Software	www.gensw.com
PCI Detect Tools	
PCISNIFF (Product name)	http://members.hyperlink.net.au/~chart/index.htm
Registry Information	
Windows NT/2000 Tips and Tricks	http://www.jsiinc.com/reghack.htm

7.2 References & Recommended Reading

There was a wealth of information to draw from to create this book. There are hundreds of books covering different aspects of Windows NT. Below is a list of publications that I used in whole or part to create this book. You may find others that are helpful as well.

Windows NT Embedded Help Documentation, Microsoft, 1999.

"A First Look at Windows NT Embedded", Sean D. Liming, Windows CE Tech Journal, 1999.

"Windows NT In the Embedded Market", Bruce Beachman, Microsoft Embedded Review, 1998.

"Remote Console Administration", Microsoft, Windows NT Embedded, 1999.

Inside Windows NT, David A. Solomon and Helen Custer, Microsoft Press, 2nd Edition,1998.

Undocumented DOS, Andrew Schulman et. al., Addison Wesley, 2nd Edition, 1994.

IIS 4 Administrator's Handbook, David Iseminger, IDG Books Worldwide Inc., 1999.

Master Windows NT Server 4, Mark Minasi, Network Press, 6th Edition, 1999.

Microsoft Internet Information Server Resource Kit, Microsoft Press, 1998.

Microsoft Windows NT Server Resource Kit, Microsoft Press, 1996.

Microsoft Windows NT Workstation Resource Kit, Microsoft press, 1996.

Using Microsoft FrontPage, Randall and Jones, QUE, 2nd Edition, 1996.

Running Microsoft FrontPage98, Jim Buyens, Microsoft Press, 1997.

Networking Essentials, Microsoft Press, 1996.

Programming Windows, Charles Petzold, Microsoft Press, 5th Edition, 1998.

Windows NT Registry Troubleshooting, Rob Tidrow, New Riders, 1996.

Windows NT Registry, Sandra Osborne, New Rider, 1998.

Windows NT 4.0 Registry: A Professional Reference, Steven B. Thomas, McGraw-Hill, 1998.

7.3 Component List

Windows NT provides a robust set of features and capabilities. Some components in the TDSD are self-explanatory. The following list provides descriptions for the standard components in the TDSD. The information was compiled from the various resources found in Section 7.2.

Component	Capability	Size	Description
Primary Domain Controller	System / Platform / SystemRole	184KB	Primary Domain Controller System
Server System	System / Platform / SystemRole	0	Stand Alone Server System
Work Station	System / Platform / SystemRole	0	WorkStation System
Automatic Login	System / Platform / User Identification	0	Automatic login for an account. Target Designer built in accounts Administrator and Guest
Minimal	System / Platform / Core OS	7.87MB	Absolute minimal core OS files needed to boot system. Fonts, MFC libraries, system utilities, control panel applets, INF files, are not included
Standard	System / Platform / Core OS	74.32MB	Full feature core OS
486C	System / Platform / HAL	974KB	Standard PC with C-Step 486
APIC	System / Platform / HAL	992KB	MPS Unitprocessor PC
AST	System / Platform / HAL	972KB	AST Manhattan SMP
Cbus	System / Platform / HAL	1007KB	Corollary C-bus Architecture
Cbusm	System / Platform / HAL	1005KB	Corollary C-bus Micro Channel Architecture
MCA	System / Platform / HAL	972KB	IBM PS/2 or Other Micro Channel-based PC
MPS	System / Platform / HAL	994KB	MPS Multiprocessor PC
MPSM	System / Platform / HAL	993KB	Micro Channel Multi Processor PC
NCR	System / Platform / HAL	1004KB	NCR System 3000 Model 3360/3450/3550
Olivetti	System / Platform / HAL	996KB	Olivetti LSX5030/40
Standard	System / Platform / HAL	958KB	Standard PC (single processor HAL)
System Pro	System / Platform / HAL	982KB	Compaq SystemPro Multiprocessor or 100% Compatible
Wyse	System / Platform / HAL	966KB	Wyse Series 7000i Model 740MP/760MP

1	System / Platform / Virtual Memory	1MB	Requires Virtual Common Component
2	System / Platform / Virtual Memory	2MB	Requires Virtual Common Component
4	System / Platform / Virtual Memory	4MB	Requires Virtual Common Component
8	System / Platform / Virtual Memory	8MB	Requires Virtual Common Component
16	System / Platform / Virtual Memory	16MB	Requires Virtual Common Component
32	System / Platform / Virtual Memory	32MB	Requires Virtual Common Component
64	System / Platform / Virtual Memory	64MB	Requires Virtual Common Component
128	System / Platform / Virtual Memory	128MB	Requires Virtual Common Component
256	System / Platform / Virtual Memory	256MB	Requires Virtual Common Component
512	System / Platform / Virtual Memory	512MB	Requires Virtual Common Component
Default Page File	System / Platform / Virtual Memory		Requires Virtual Common Component
No Page File	System / Platform / Virtual Memory	0MB	Requires Virtual Common Component
System Message Interception	System / Platform / Error Recovery	0	Capture GUI and respond to system messages. A DLL can be called if an error occurs. Please see Target Designer help for conditional issues.
Floppy	System / Devices / Storage / Floppy Disk	18KB	Floppy driver
DiskOnChip	System / Devices / Storage / Fixed Disk	31KB	DiskOnChip driver
EIDE	System / Devices / Storage / Fixed Disk	0	Enhanced IDE hard drive driver
El Torito CD as Disk	System / Devices / Storage / Fixed Disk	7KB	El Torito CD boot support
IDE Disk	System / Devices / Storage / Fixed Disk	26KB	IDE hard drive Driver support
SCSI Disk	System / Devices / Storage / Fixed Disk	15KB	SCSI Disk support
EIDE CD-ROM	System / Devices / Storage / CD-ROM	0	Enhanced IDE CD ROM Support
SCSI CD-ROM	System / Devices / Storage / CD-ROM	22KB	SCSI CD-ROM support
CDFS	System / Devices / Storage / File Systems	60KB	CD File System

FAT	System / Devices / Storage / File Systems	139KB	File Allocation Table File System
NTFS	System / Devices / Storage / File Systems	370KB	NT File System
Fault Tolerance	System / Devices / Storage / Storage Filter Drivers	33KB	For use with Server or PDC system, provides fault tolerance / RAID support for hard drives
Write Filter	System / Devices / Storage / Storage Filter Drivers	17KB	Creates a RAM disk to redirect writes of a read-only partition to memory. Used to write protect HDD partitions and support NT CD booting.
ATI Graphics	System / Devices / Display / Display Adapters	0	
Chips Video Accelerator	System / Devices / Display / Display Adapters	0	
Cirrus Logic 5465 Graphics	System / Devices / Display / Display Adapters	0	
Cirrus Logic Graphics Adapter	System / Devices / Display / Display Adapters	0	
Matrox Graphics Adapter	System / Devices / Display / Display Adapters	0	
Matrox Millenium Graphics Adapter	System / Devices / Display / Display Adapters	0	
S3 Graphics	System / Devices / Display / Display Adapters	0	
SiS Graphics Adapter	System / Devices / Display / Display Adapters	0	
Trident Video Accelerator	System / Devices / Display / Display Adapters	0	
ATI	System / Devices / Display / Display Drivers	248KB	Driver for Adapter
Chips	System / Devices / Display / Display Drivers	183KB	Driver for Adapter
Cirrus	System / Devices / Display / Display Drivers	162KB	Driver for Adapter
CL546x	System / Devices / Display / Display Drivers	479KB	Driver for Adapter
MGA	System / Devices / Display / Display Drivers	192KB	Driver for Adapter
MGA64	System / Devices / Display / Display Drivers	802KB	Driver for Adapter

NullVGA	System / Devices / Display / Display Drivers	10KB	Headless / Display less system, no adapter component
S3	System / Devices / Display / Display Drivers	130KB	Driver for Adapter
SiSV	System / Devices / Display / Display Drivers	205KB	Driver for Adapter
Tgiul40	System / Devices / Display / Display Drivers	67KB	Driver for Adapter
VGA	System / Devices / Display / Display Drivers	99KB	Driver for Adapter
US Keyboard Layout	System / Devices / Keyboard / Keyboard Layout	9KB	Support US Keyboard layout
Null Keyboard Driver	System / Devices / Keyboard / Keyboard Drivers	4KB	Keyboard less system
PC/AT Keyboard	System / Devices / Keyboard / Keyboard Drivers	29KB	Standard keyboard support
English (US) Input Locale	System / Devices / Keyboard / Keyboard Input Local	7KB	Local input language localization
Logitech	System / Devices / Mouse	29KB	Logitech mouse support
Microsoft	System / Devices / Mouse	29KB	Microsoft mouse support
Null Mouse	System / Devices / Mouse	0	Mouse-less system
Serial Mouse	System / Devices / Mouse	14KB	Serial mouse support
Adaptec AHA-154X Plug-N-Play	System / Devices / BUS Devices SCSI / SCSI Adapters	0	Adaptec 154X SCSI bus support
Adaptec AHA-154X/AHA-164X	System / Devices / BUS Devices SCSI / SCSI Adapters	0	Adaptec 154XAHA-164X SCSI bus support
Adaptec AHA-294X/AHA-394X or AIC-78XX PCI	System / Devices / BUS Devices SCSI / SCSI Adapters	0	Adaptec 294X/AHA-394X or AIC-78XX SCSI bus support
BusLogic Multimaster PCI SCSI Host Adapter	System / Devices / BUS Devices SCSI / SCSI Adapters	0	BusLogic SCSI Bus Support
EIDE SCSI Adapter	System / Devices / BUS Devices SCSI / SCSI Adapters	0	SCSI bus support for EIDE SCSI system
Aha 154x	System / Devices / BUS Devices SCSI / SCSI Drivers	9KB	Driver for Adaptec 154X-164X SCSI bus adapter
Aic 78xx	System / Devices / BUS Devices SCSI / SCSI Drivers	26KB	Driver for Adaptec 294X/AHA-394X or AIC-78XX SCSI bus support
Buslogic	System / Devices / BUS Devices SCSI / SCSI Drivers	8KB	Driver for BusLogic SCSI Bus Support

EIDE SCSI Driver	System / Devices / BUS Devices SCSI / SCSI Drivers	27KB	Driver for
COM1	System / Devices / Serial Ports	0	COM1 Serial port
COM2	System / Devices / Serial Ports	0	COM2 Serial port
COM3	System / Devices / Serial Ports	0	COM3 Serial port
COM4	System / Devices / Serial Ports	0	COM4 Serial port
LPT1	System / Devices / Parallel Ports	0	LPT1 Parallel port
LPT2	System / Devices / Parallel Ports	0	LPT2 Parallel port
Dial-Up Networking Serial Cable Between 2 PCs	System / Devices / Modems	42KB	Serial networking
Sportster 28.8kbps Internal Modem	System / Devices / Modems	50KB	
Sportster 56Kbps External Modem X2	System / Devices / Modems	48KB	
Standard 28.b Kbps Modem	System / Devices / Modems	35KB	
Winmodem 56Kbps Modem	System / Devices / Modems	1.19MB	
HP LaserJet 6L	System / Devices / Printers	972KB	
HP LaserJet 6L - Shared	System / Devices / Printers	972KB	
Domain Participation	System / Network / Network Membership	229KB	Preset image to participate in a domain
Workgroup Participation	System / Network / Network Membership	0	Preset image to participate in a workgroup
	System / Network / Network Adapters		
	System / Network / Network Adapters (Second Instance)		
AMDPCN	System / Network / Network Drivers	51KB	
CpqNF3	System / Network / Network Drivers	70KB	
DC21X4	System / Network / Network Drivers	52KB	
Digi DataFire Driver	System / Network / Network Drivers	1.63 MB	

E100B	System / Network / Network Drivers	23KB	
E100BNT	System / Network / Network Drivers	28KB	
EI59X	System / Network / Network Drivers	50KB	
EL90x	System / Network / Network Drivers	41KB	
Elnk3	System / Network / Network Drivers	15KB	
IEEPRO	System / Network / Network Drivers	13KB	
N100	System / Network / Network Drivers	29KB	
NE2000	System / Network / Network Drivers	16KB	
RTL8029	System / Network / Network Drivers	37KB	
SMC1208	System / Network / Network Drivers	25KB	
SMC9xxx	System / Network / Network Drivers	29KB	
Apple Talk	System / Network / Network Protocols	409KB	Apple Talk protocol
IPX/SPX	System / Network / Network Protocols	421KB	Internetwork Packet Exchange / Sequenced Packet Exchange - Novell network support
NetBEUI	System / Network / Network Protocols	98KB	NetBEUI Protocol
PPTP	System / Network / Network Protocols	1MB	Point to Point Tunneling Protocol – Used as a wrapper for PPP and supports dial in connections to Virtual Private Networks
TCP/IP	System / Network / Network Protocols	514KB	Transmission Control Protocol / Internet Protocol (Just in case you didn't known)
Alerter	System / Network / Network Services	23KB	Notifications can be sent to selected users of system errors, important events, or resources of a server. Should have messenger service enabled

ASP	System / Network / Network Services	1.22MB	Active Server Pages Support for IIS – Server mechanism that provides dynamic content access by all browsers
Computer Browser	System / Network / Network Services	0	Maintains the current list of computers on a network, allows for browsing other computers on a network.
DHCP Client	System / Network / Network Services	56KB	Dynamic Host Control Protocol Client to connect to a Server with DHCP Server Services
DHCP Server	System / Network / Network Services	1.78MB	Dynamic Host Control Protocol Server service, contains DHCP utilities to create TCP/IP ranges for DHCP client computers
DNS Server	System / Network / Network Services	309KB	Domain Name Service - resolves inter-Domain communication
File and Print Services for Netware	System / Network / Network Services	2.31MB	File and Print Services for Netware
IIS 3.0	System / Network / Network Services	722KB	Internet Information Server 3.0
LAN Manager Server	System / Network / Network Services	329KB	Server Service – RPC support and file, print, and named pipe sharing
LAN Manager Workstation	System / Network / Network Services	322KB	Network connection and communications
Messenger	System / Network / Network Services	38KB	Sends out messages from administrator and the alerter service
NetBIOS	System / Network / Network Services	39KB	Network Basic Input / Output System provide programming commands to manipulate the network
NetDDE	System / Network / Network Services	227KB	Network transportation and security for Network Dynamic Data Exchange
NT LM Security Support Provider	System / Network / Network Services	35KB	Basic NT LAN Manager Security Service
RAS Client	System / Network / Network Services	2.64MB	Remote Access Server (ignore 'Client') for Workstation only
RPC	System / Network / Network Services	1.015MB	Remote Procedure Call Service

RRAS	System / Network / Network Services	6.72MB	Routing and Remote Access Services (aka Steelhead) Provide IP routing. For use with Server or PDC systems
SAP Agent	System / Network / Network Services	36KB	Service Advertising Protocol used in Novell networks
Service for Macintosh	System / Network / Network Services	1.36MB	Network support services for Macintosh
Simple TCP Service	System / Network / Network Services	43KB	Supports the TCP/IP services Character Generator, Day-time, Discard, Echo, and Quote of the Day. When started, the computer can re-spond to requests from other computers that support these protocols.
SNMP	System / Network / Network Services	169KB	Simple Network Management Protocol – NT system can be monitored by remote management tools
TCP/IP Printers	System / Network / Network Services	98KB	Allows for UNIX clients to connect to a printer connected to an NT system
WINS Server	System / Network / Network Services	1.89MB	Windows Internet Name Services
WinSock	System / Network / Network Services	178KB	Windows Socket Services
Inet Service Manager	System / Network / Network Applications	2.45MB	IIS 3.0 Service Manager
Net Command (NET.EXE)	System / Network / Network Applications	305KB	Network Command line Configuration Utility - NET.EXE
TCP/IP Utilities	System / Network / Network Applications	733KB	Contains various TCP/IP utilities such as ARP (address resolution Protocol), FTP client, Telnet client, Ping, IPCONFIG, ROUTE (Manipulates network routing tables), REXEC (Runs commands on remote computers running the REXEC service), Hostname (Display the name of the current computer), and Nbstat (Diagnostic command displays protocol statistics and current TCP/IP connections using NetBIOS over TCP/IP)

OLE/COM	System / Windows Services / OLE and COM	2.18MB	OLE and COM support
DCOM	System / Windows Services / Distributed COM	0	Distributed COM Support
Cryptography Core	System / Windows Services / Cryptography	1.42MB	Cryptography Core supports the additions of plug-ins and DLLs that encrypt messages
TAPI 2.1	System / Windows Services / TAPI	382KB	Telephony API 2.1 Support
Kernel TSP	System / Windows Services / Telephony Service Providers	20KB	Generic Kernel TSP
Unimodem	System / Windows Services / Telephony Service Providers	620KB	Universal Modem TSP
ODBC Access Driver	System / Windows Services / Database Access / Database Client Support / ODBC	372KB	ODBC for MS ACCESS
ODBC Core	System / Windows Services / Database Access / Database Client Support / ODBC	1.01MB	Open Database Connectivity Core Components
ODBC Jet Support	System / Windows Services / Database Access / Database Client Support / ODBC	3.95MB	ODBC Jet Support
ODBC SQL Server Driver	System / Windows Services / Database Access / Database Client Support / ODBC	500KB	ODBC for SQL
Graphical Remote Management	System / Management / Management Applications / Remote Administration	900KB	Graphically control a remote machine's desktop over TCP/IP
Modem Console Administration	System / Management / Management Applications / Remote Administration	0	Remote Command console access (Text) over a modem
Serial Console Administration	System / Management / Management Applications / Remote Administration	143KB	Remote Command console access (Text) over a NULL modem serial link
Telnet Server	System / Management / Management Applications / Remote Administration	475KB	Telnet Server support, connect via a Telnet client over TCP/IP
Registry Editor	System / Management / Management Applications / System Configuration Management / Registry Management	3.12MB	Regedit32
Disk Administrator	System / Management / Management Applications / System Configuration Management / Storage Administration / Windows Disk Management	812KB	DiskAdmin

USER Manager	System / Management / Management Applications / System Configuration Management / User Administration	915KB	
USER Manager for Domains	System / Management / Management Applications / System Configuration Management / User Administration	961KB	
MMC	System / Management / Management Applications / System Configuration Management / Microsoft Management Console	8.17MB	Microsoft Management Console
Event Viewer	System / Management / Management Applications / Performance Management and Troubleshooting / System Events	731KB	Event view (view system messages and error logs)
Performance Monitor	System / Management / Management Applications / Performance Management and Troubleshooting / Performance Monitoring	255KB	Measures system performance through a variety of counters
Task Manager	System / Management / Management Applications / Performance Management and Troubleshooting / Performance Monitoring	106KB	Control and monitor processes and dynamically display CPU and Memory usage.
Windows NT Diagnostics	System / Management / Management Applications / Performance Management and Troubleshooting / Performance Monitoring	195KB	Basic NT diagnostic program, view system resources and environment settings
Event Logging	System / Management / Management Services	377KB	Event Logging Service
Performance Counters	System / Management / Management Services	359KB	Performance Counters
Scheduler	System / Management / Management Services	22KB	Provides scheduling services, Must be enabled to support the AT utility
WMI	System / Management / Management Services	7.31KB	Windows Management Instrumentation - Web-based Enterprise Management (WBEM) initiative utilities tools. (see http://www.dmtf.org for more information)

Arial Font	System / Desktop Settings / Fonts	612KB	
Courier	System / Desktop Settings / Fonts	726KB	
Lucida Console	System / Desktop Settings / Fonts	112KB	
Lucida Unicode	System / Desktop Settings / Fonts	316KB	
Modem	System / Desktop Settings / Fonts	9KB	
MS Sans Serif	System / Desktop Settings / Fonts	151KB	
Roman	System / Desktop Settings / Fonts	13KB	
Script	System / Desktop Settings / Fonts	12KB	
Symbol	System / Desktop Settings / Fonts	64KB	
Times	System / Desktop Settings / Fonts	738KB	
Wing Dings	System / Desktop Settings / Fonts	79KB	
Standard Accessibility	System / Desktop Settings / Accessibility	0	Accessibility for those with disabilities
Explorer Links	System / Desktop Settings / Application Links	36KB	Shortcut links for the Explorer start bar
Device Settings	System / Global System Settings	0	Similar to the Control Panel Applet of the same name. Controls the startup of devices for select devices within the configuration
Services Settings	System / Global System Settings	0	Similar to the Control Panel Applet of the same name. Controls the startup of a service for selected services within the configuration
Time Zone Settings	System / Global System Settings	0	Set the time zone for the configuration
Console Administrator	System / Shared System Components	56KB	Common console administration
Display	System / Shared System Components	25KB	Common Display component
FTP Performance Counters	System / Shared System Components	0	FTP Performance counters for IIS 3.0
Keyboard Drivers	System / Shared System Components	9KB	Keyboard drivers common

Keyboard Locale	System / Shared System Components	135KB	Keyboard local common
Logon Screen Saver	System / Shared System Components	37KB	Logon Screen saver
Mouse Common	System / Shared System Components	9KB	Mouse Common
Network Common	System / Shared System Components	1.91MB	Network Common
Parallel Ports	System / Shared System Components	24KB	Parallel Common
Printer	System / Shared System Components	469KB	Printer Common
SCSI	System / Shared System Components	58KB	SCSI Common
Serial Ports	System / Shared System Components	44KB	Serial Common
Virtual Memory	System / Shared System Components	0	VM Common
WWW Performance	System / Shared System Components	0	WWW Performance counters for IIS 3.0
Command Shell	Applications / Shell	0	Command Console Shell
Explorer Shell	Applications / Shell	0	Explorer Shell
Manual Shell	Applications / Shell	0	Runs multiple shells upon login. Each application in the properties box is separated by a comma (,).
Task Manager Shell	Applications / Shell	0	Task Manager Utility as the shell
ACL	Applications / Utilities	797KB	Active Control List DLLs for used for sharing files or directories
Add Printer Wizard	Applications / Utilities	887KB	Add Printer Wizard
AT Command	Applications / Utilities	109KB	Network command for scheduling the execution of programs at a predetermined time. Requires Scheduling service.
Auto Check	Applications / Utilities	1.024MB	Checks disk if system wasn't shutdown properly
Copy File Command	Applications / Utilities	240KB	XCOPY command line utility
Explorer	Applications / Utilities	322KB	Explorer
File ACL Command	Applications / Utilities	64KB	CALC command line utility to change file/directory share permissions
File Attrib Command	Applications / Utilities	156KB	Attrib console command

Shutdown Utility	Applications / Utilities	34KB	System shutdown command line utility
System Cloning	Applications / Utilities	26KB	Changes the SID on the first boot of a system, allowing for one image to be replicated
Windows NT Command	Applications / Utilities	262KB	Command console
Windows NT Help Engine	Applications / Utilities	351KB	Windows NT Help Engine
Calculator	Applications / Accessories	124KB	
Character Map	Applications / Accessories	72KB	
Chat	Applications / Accessories	78KB	
Clipbook Viewer	Applications / Accessories	201KB	
Clock	Applications / Accessories	41KB	
Hyper Terminal	Applications / Accessories	838KB	
Notepad	Applications / Accessories	225KB	
Object Pacakge	Applications / Accessories	116KB	
Paint	Applications / Accessories	1.44MB	
Phone Dialer	Applications / Accessories	56KB	
Quick View	Applications / Accessories	1.12KB	
Wordpad	Applications / Accessories	2.40MB	

Index