Installing Microsoft Windows Driver Development Kit (DDK) for Microsoft Windows Server 2003 Service Pack 1 (SP1) on Windows XP Pro SP2

This step-by-step document also contains how-to install the Debugging Tools for Windows and Symbols Package.

Machine specification used in this scenario.

1. Operating System : Windows Xp Pro SP2
2. RAM : 2 GB DDR2
3. HDD : 160++G
4. Display : 128 MB ATI PCI Express
5. Processor : Intel Core 2 Duo 4400 2.00 GHz

Installing the DDK

First of all we need to download the ISO from Microsoft or you can buy the CD as well. The current version is Windows Server 2003 SP1 that covers Windows 2000, Windows XP and Windows 2003 Server family. This DDK has been superseded by the Windows Driver Kit (WDK). We will try to install the WDK later. While the DDK can be downloaded openly, you need to register for free and participate in the respective Microsoft community to download WDK. Microsoft said that the WDK should be used for the following reasons:

1. Use the Windows Vista build environments in the WDK to build drivers that use new features or functionality available only in Windows Vista.
2. Use the Windows Server 2003 build environments in the WDK to build drivers that use new features or functionality available only in Windows Server 2003.
3. Use the Windows XP build environments in the WDK to build drivers that do not use new functionality available only in Windows Vista or only in Windows Server 2003 and that are targeted for either Windows XP or Windows Server 2003 and Windows XP. The Windows XP build environments in the WDK contain minor updates to the Windows DDK that shipped with Windows XP SP1 and with Windows XP.
4. Use the Windows 2000 build environments in the WDK to build drivers designed to run on Windows Vista, Windows Server 2003, Windows XP, or Windows 2000. The Windows 2000 build environment in the WDK includes updated headers and libraries for Windows 2000 Service Pack 4 (SP4).

If you download the ISO, you need to burn it on the CD to make it useable. If needed, you can rename the .ISO extension to .IMG. If you want to get an idea how to install/extract the .IMG/.ISO files, go to this link

Then we are ready to install the DDK. Insert the previously burned ISO CD into the CD-ROM drive the autorun will be launched. You may want to read the following `Getting Started` page after a very short splash screen else just explore the CD through Windows explorer and run the **setup.exe**.

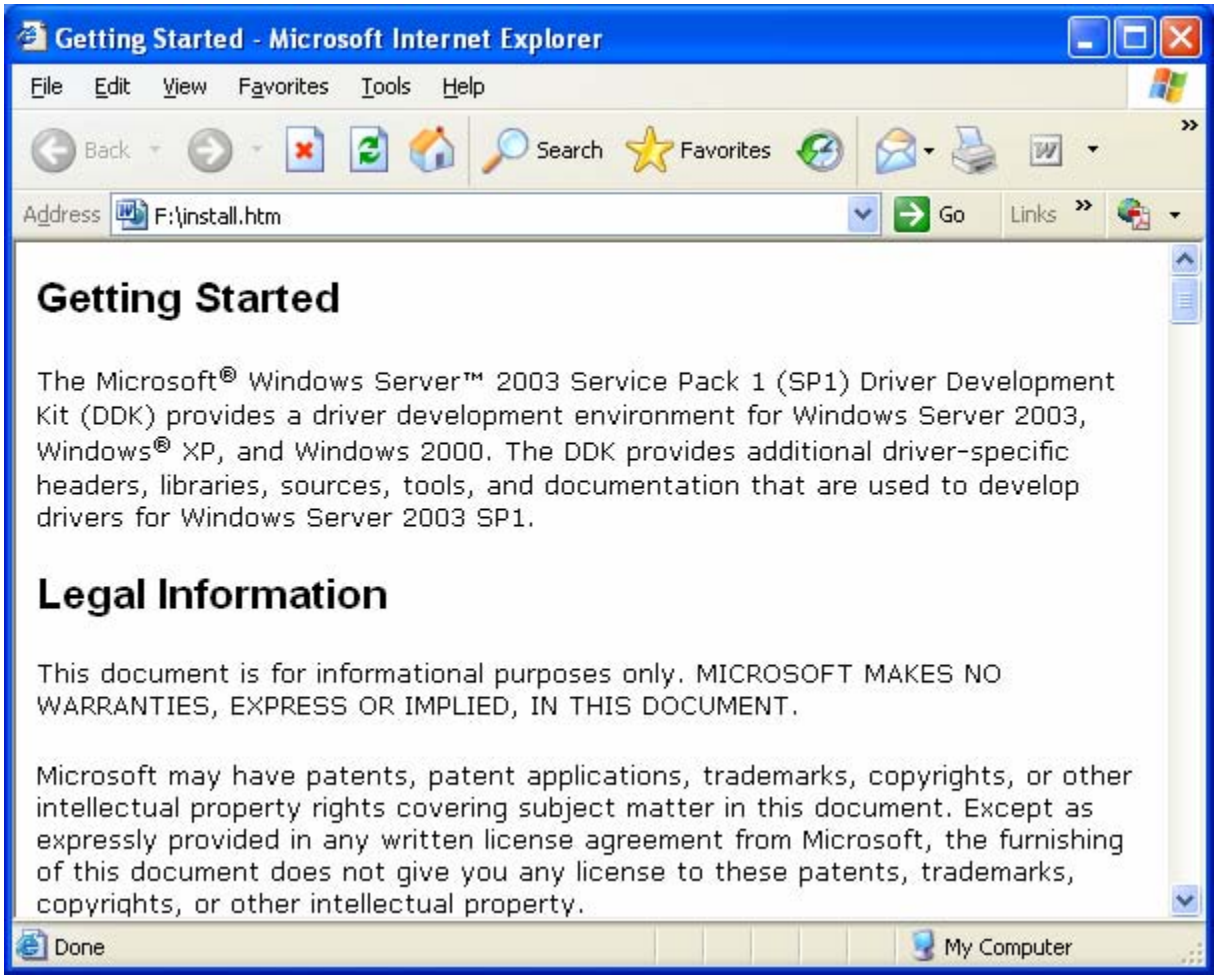

Figure 1: The Getting Started page for DDK

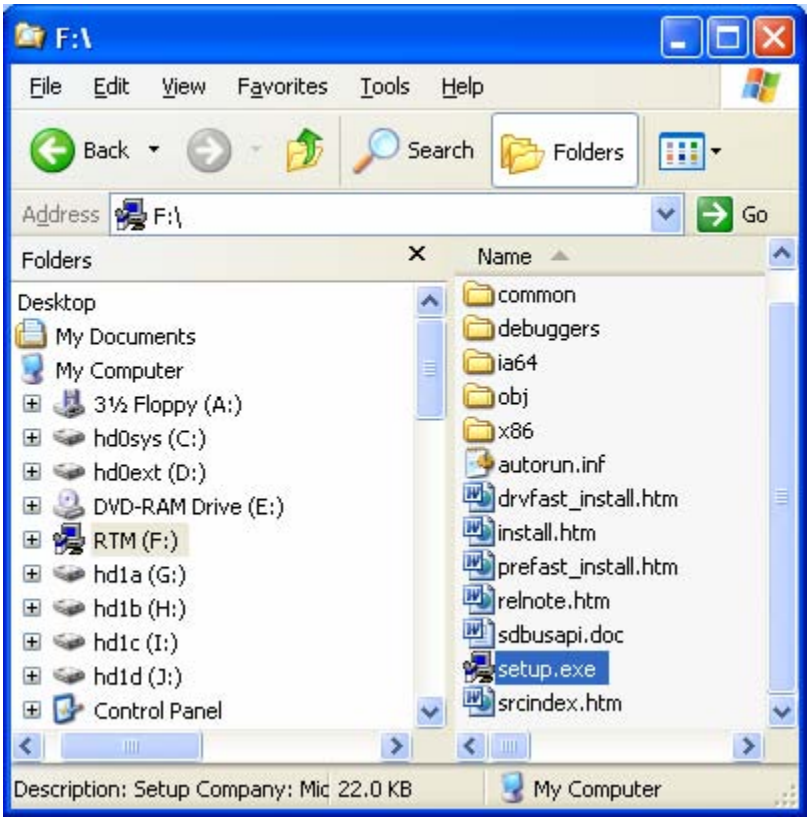

Figure 2: Launching the setup.exe file

Just click the `Next` button.

Figure 3: The Microsoft Windows Driver Development Kit, DDK installation welcome page

Read the `License Agreement` and select the `I Agree` radio button and then click the `Next` button.

Figure 4: Agreeing the License Agreement

Change the destination directory if needed. Here we just accept the default path given. Click the `Next` button.

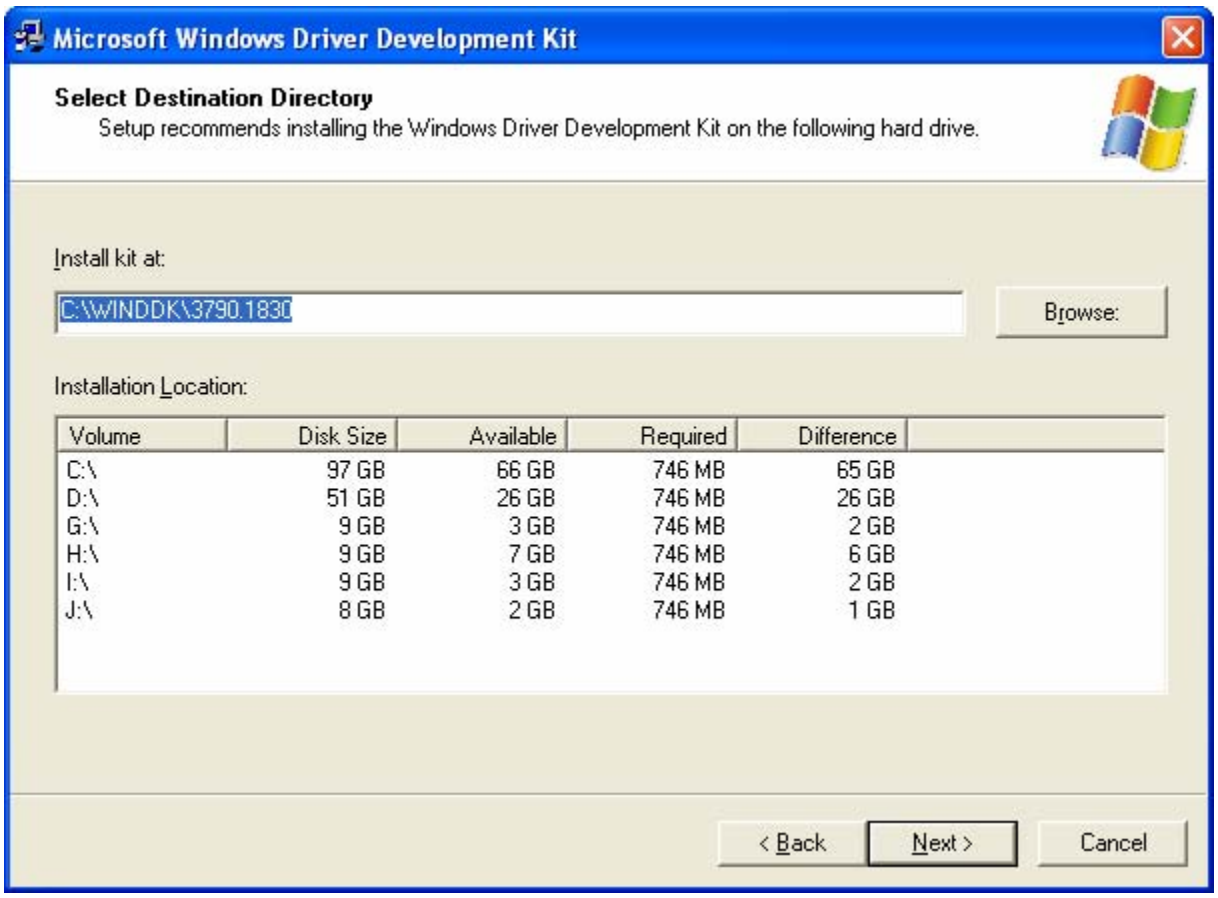

Figure 5: Setting the installation path

We select all the DDK component groups. You can select just what you want to use, others can be re-installed. Click the `Next` button. The required disk space is shown on the bottom-right, so make sure you have enough disk space.

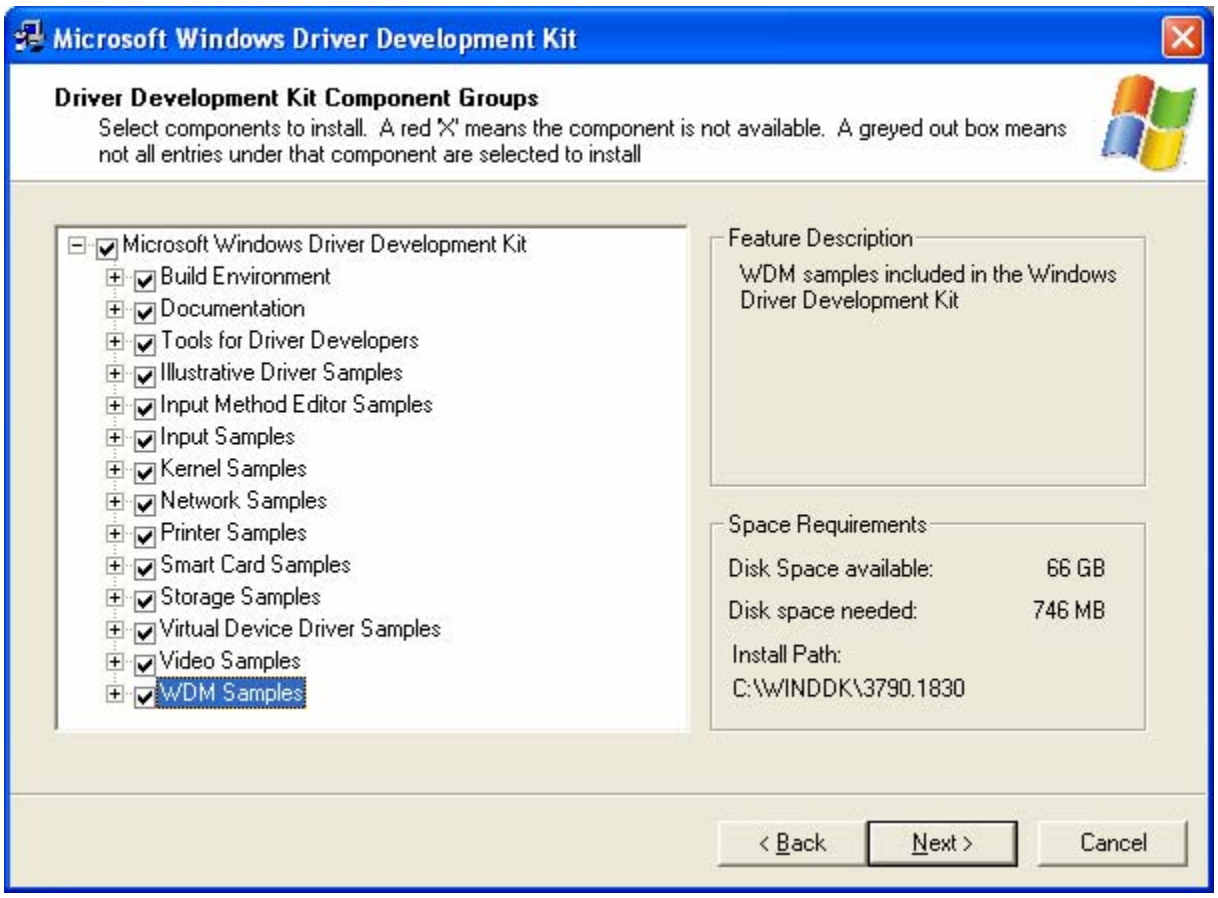

Figure 6: Selecting the DDK component groups

Click the `Next` button if there are no more changes else click the `Back` button.

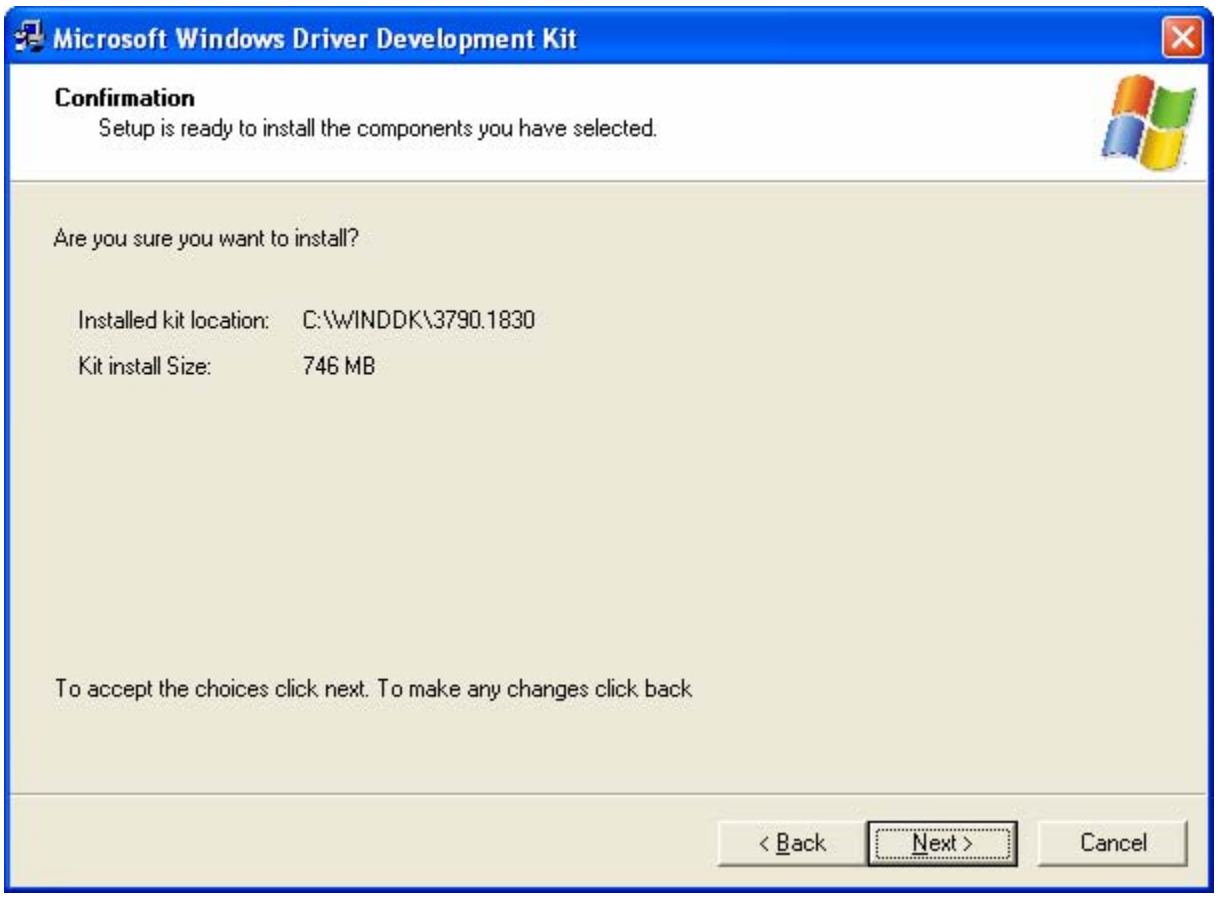

Figure 7: The DDK installation confirmation page

The DDK installation begins. Any error such as unreadable or corrupted file will be prompted during this process. Sit down and relax.

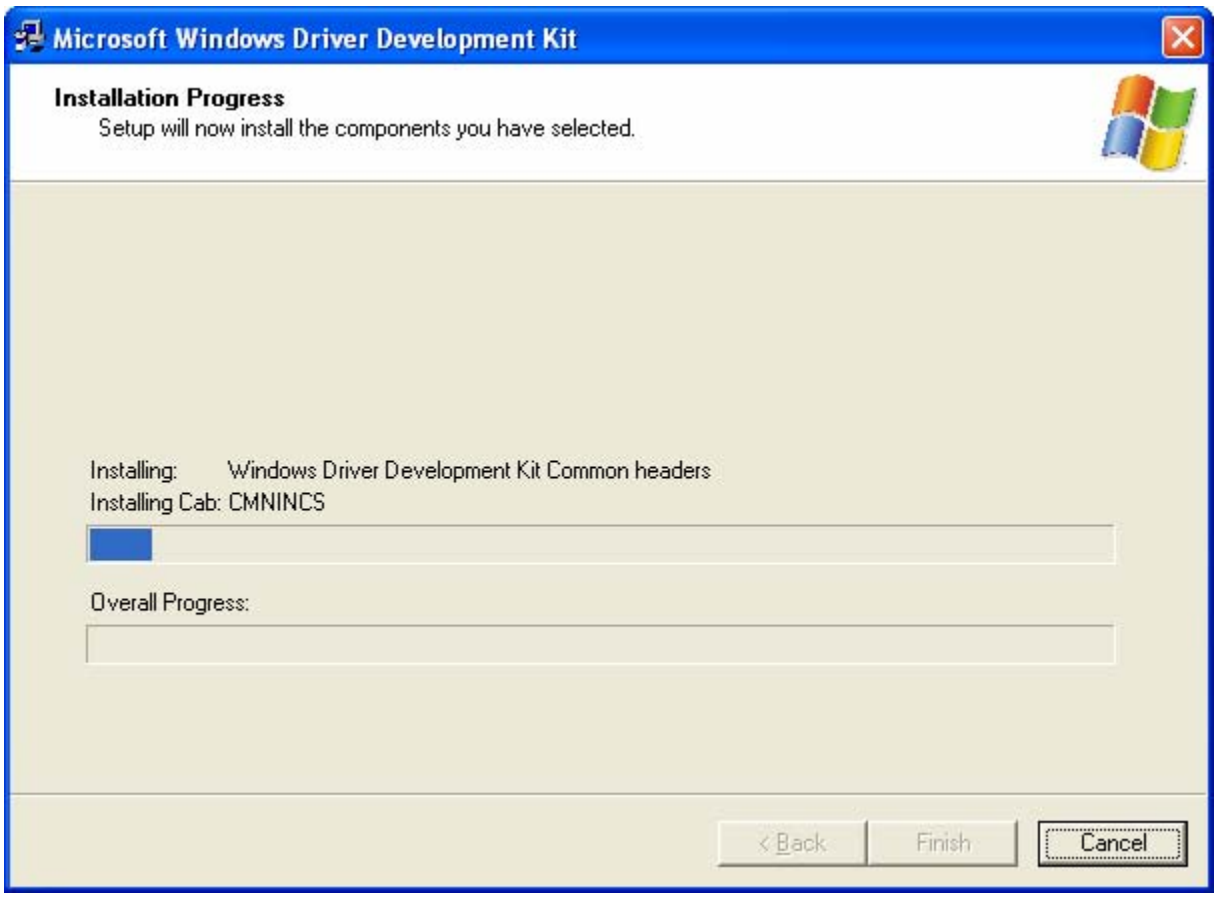

Figure 8: The Windows DDK installation begins

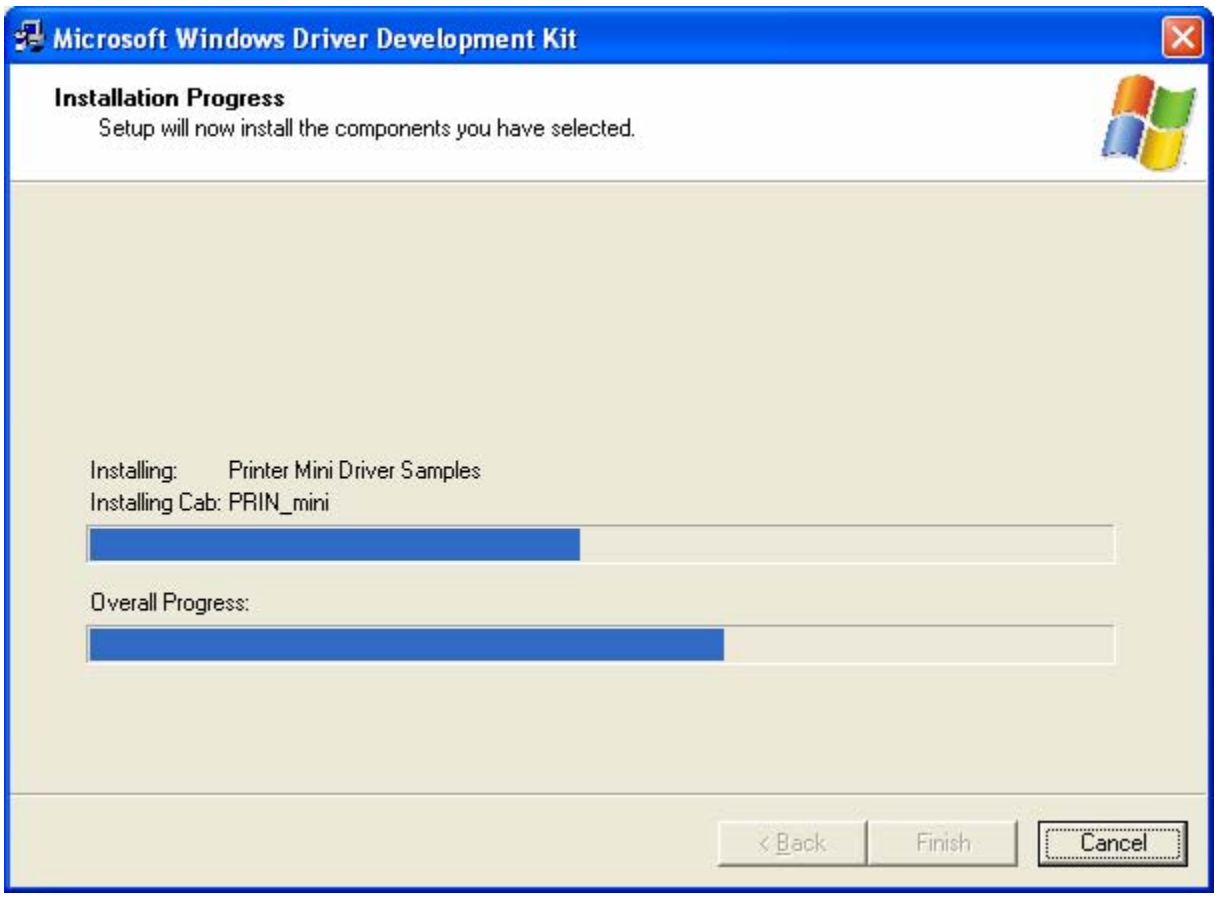

Figure 9: The Windows DDK installation in progress

The DDK installation is complete. Click the `Finish` button.

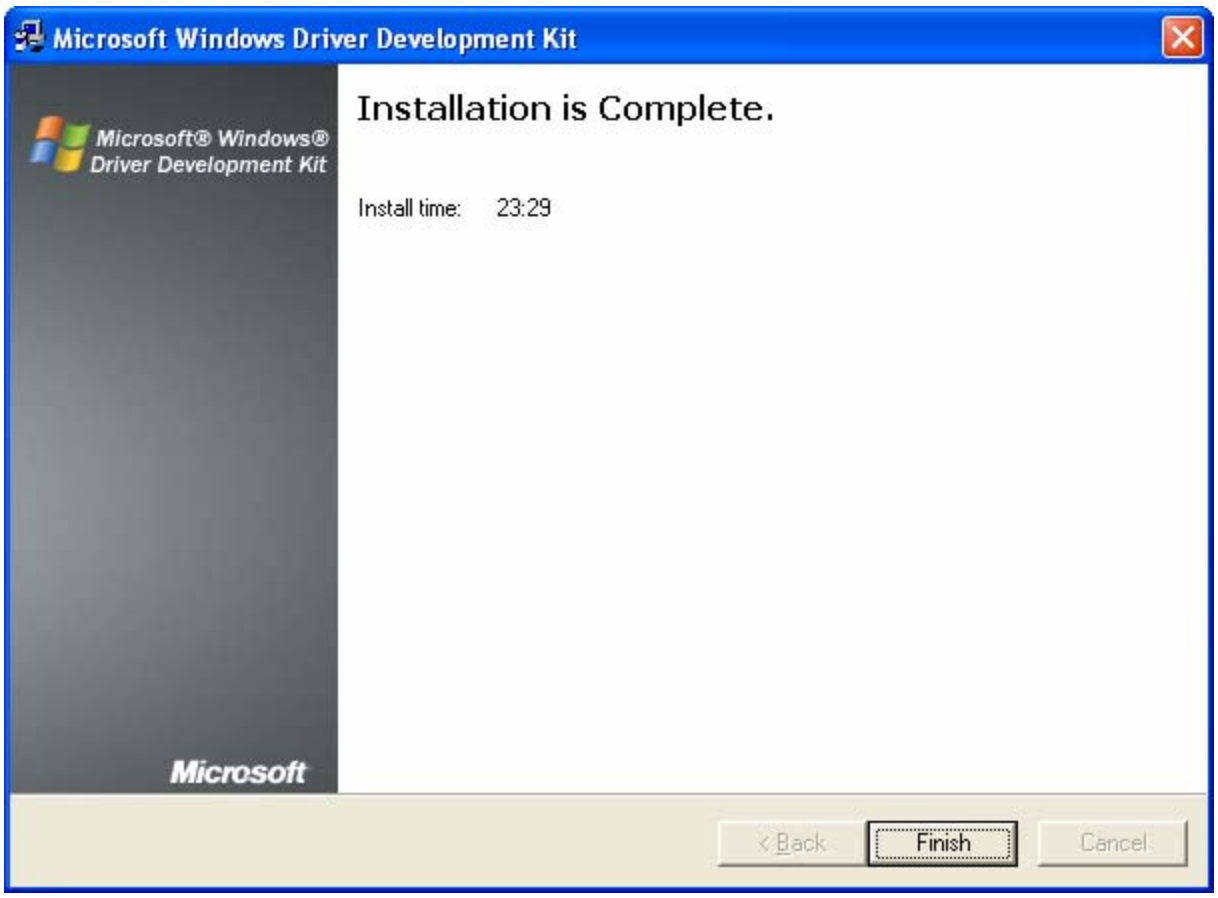

Figure 10: The Windows DDK installation is complete

You can see the created short cut on the `Start` → `All Programs` as shown below.

Figure 11: The Windows DDK start menu short cut

Installing the Debuggers

Debugging Tools for Windows features WinDbg, a powerful debugger with a graphical interface that can debug both user-mode and kernel-mode code. Debugging Tools for Windows also includes:

- KD - Command-line kernel debugger.
- NTSD - Command-line user-mode debugger.
- CDB - Command-line user-mode debugger (variant of NTSD).

and many additional tools. The documentation in Debugging Tools for Windows describes the use of these debuggers and includes tips for user-mode and kernel-mode debugging. Debugging Tools for Windows is available in three different versions:

1. A 32-bit version.
2. A native Intel Itanium version, and
3. A native x64 version.

The 32-bit version is appropriate for most users. If you are planning on debugging a user-mode application on an Itanium-based processor, you should install the Itanium version of the debuggers. If you are planning on debugging a user-mode application on an x64 processor, you should select the x64 version of the debuggers. The Itanium and x64 debuggers can be installed only on 64-bit versions of Windows. These debugging tools require approximately 25 MB of hard disk space.
To obtain the most current version of Debugging Tools for Windows, visit the Microsoft Debugging Tools Web site. If it is not convenient to visit this site, you can install Debugging Tools for Windows directly from this CD. However we found only the 32 bit, 64 bit and 64 bit for AMD processor in the CD. There is no Itanium version.
Since the version of Debugging Tools for Windows on this CD may not be the most recent version, it is recommended that you only use these links if you cannot access the Web site. The setup files can be found under the **`debuggers`** folder in the CD as shown below. Launch an appropriate setup file. In our case the 32 bit Debugging Tools for Windows already installed together when we install Windows Software Development Kit (SDK).

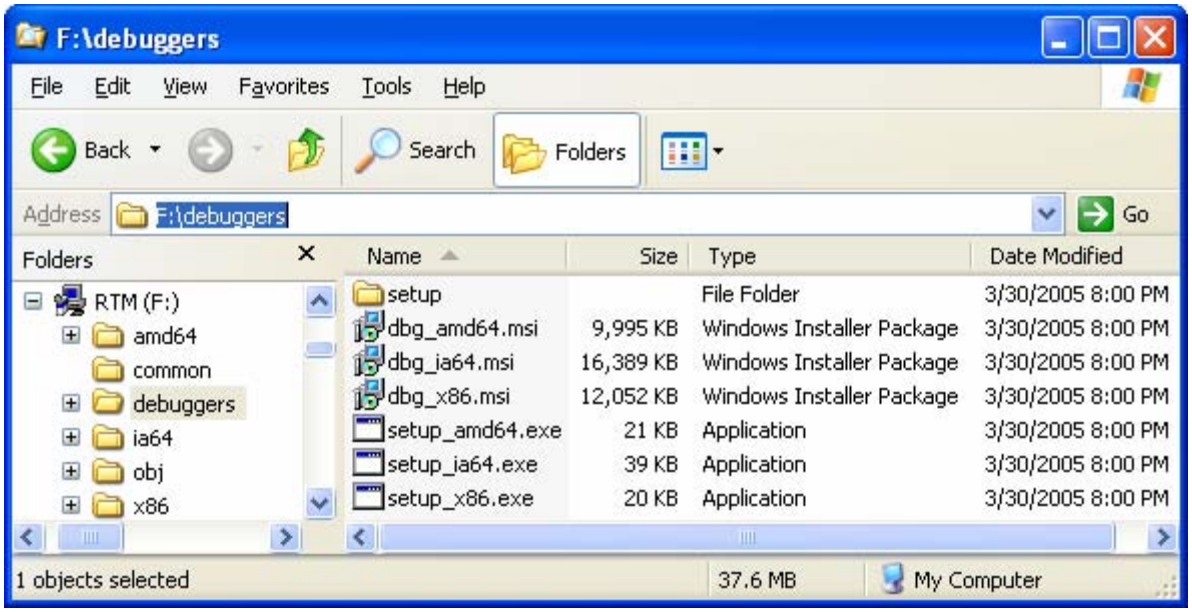

Figure 12: The Debugging Tools for Windows setup files

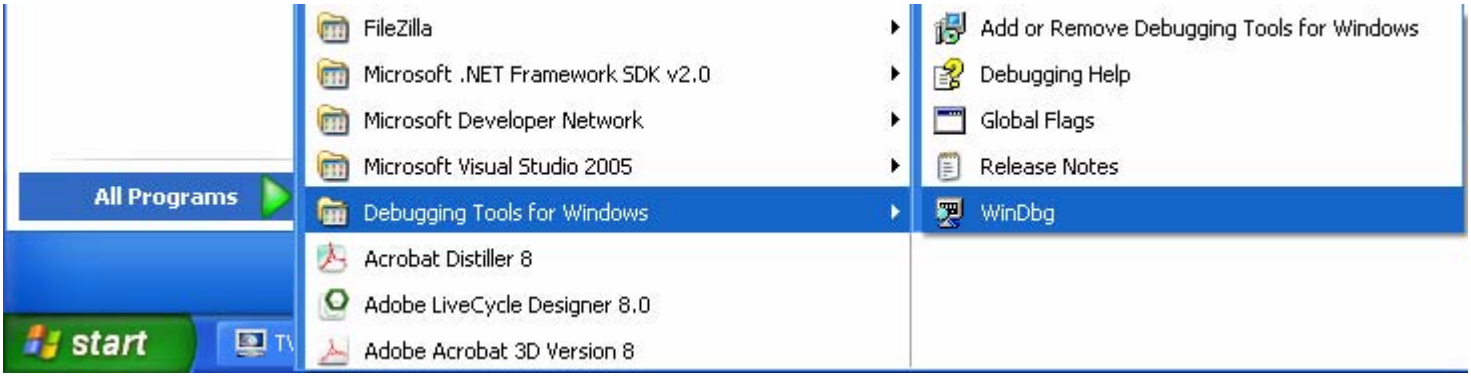

Figure 13: The Debugging Tools for Windows short cut menu

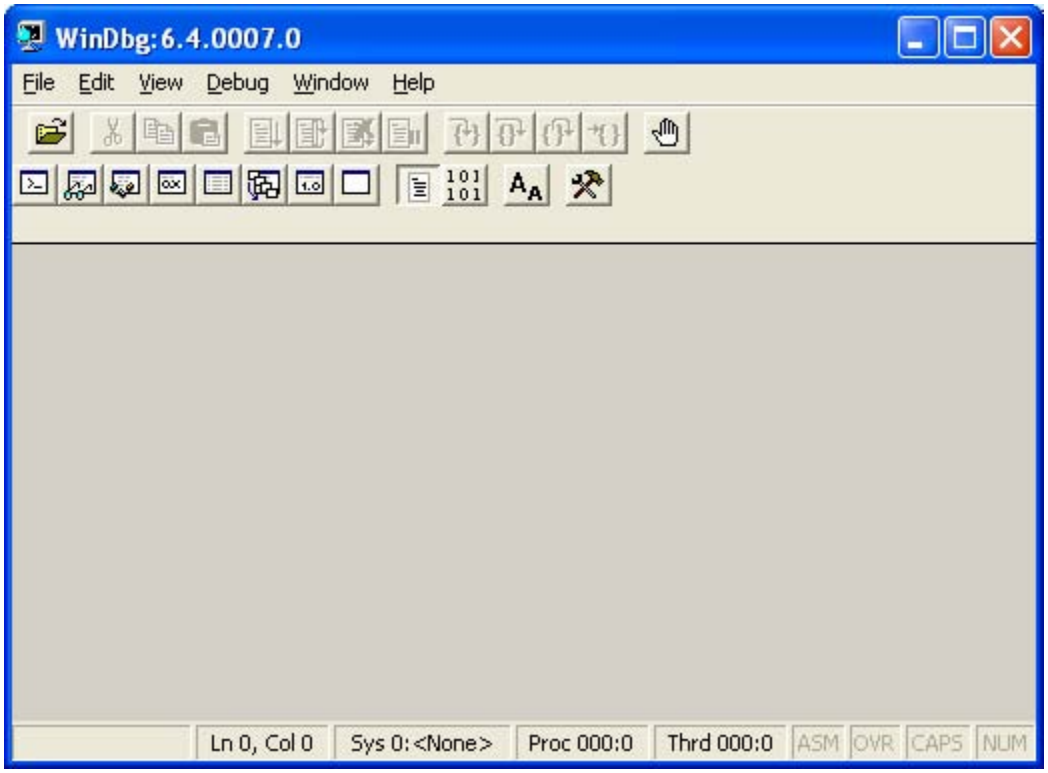

Figure 14: The naked WinDbg in action

Installing Symbols Package

As said by Microsoft, the new linker strips all debug information from the SYS file and moves the data into a PDB file. The PDB file should be copied to the symbols directory for debugging. Copying the SYS file will not provide debugging information. If you want the entire set of symbols for Windows Vista, Windows Server 2003, Windows XP, or Windows 2000, then you can download a symbol package and install it on your computer.

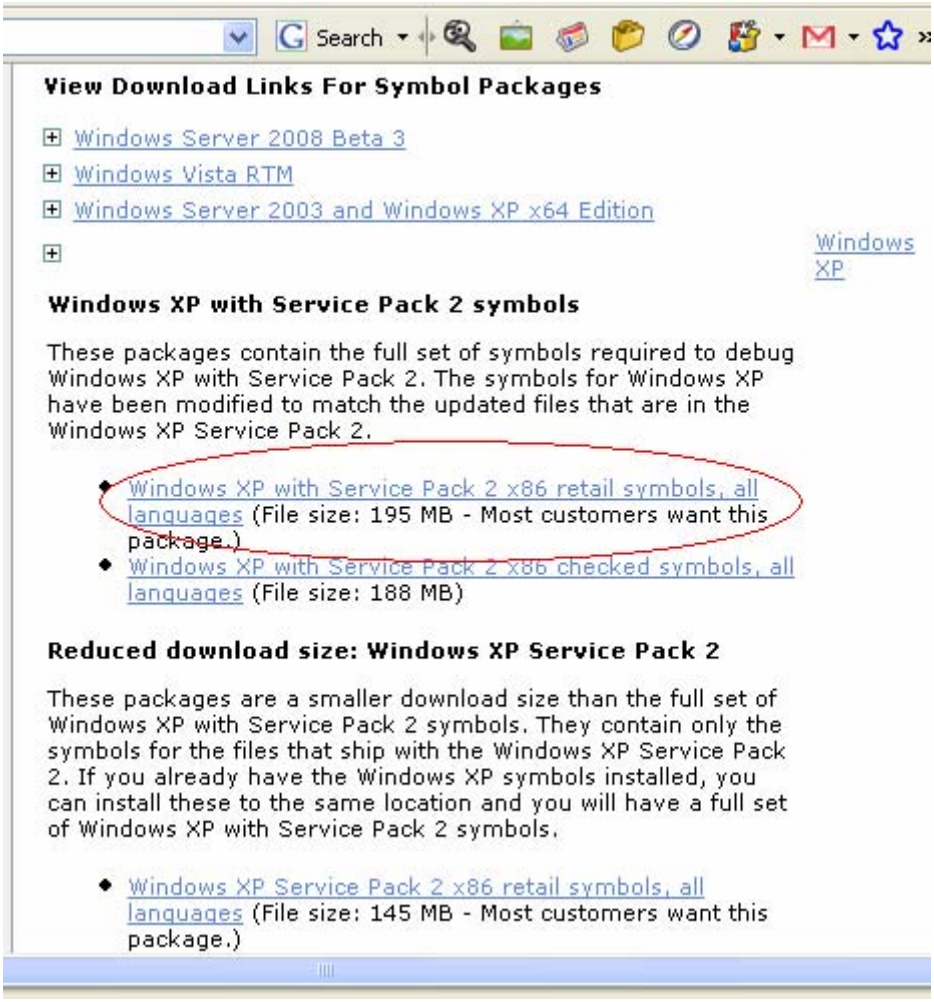

Figure 15: Selecting Windows XP with Service Pack 2 retail symbols for downloading

In our case, because we can't be online all the time, we have downloaded a package for Windows XP SP2 symbols package and run the executable file (~ 200MB).

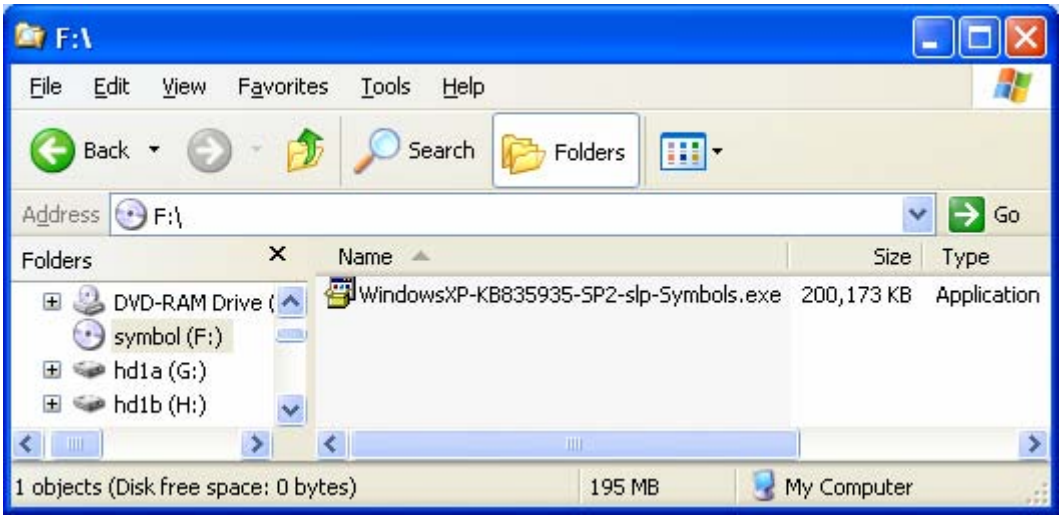

Figure 16: The Windows XP with Service Pack 2 retail symbols self-extraction file

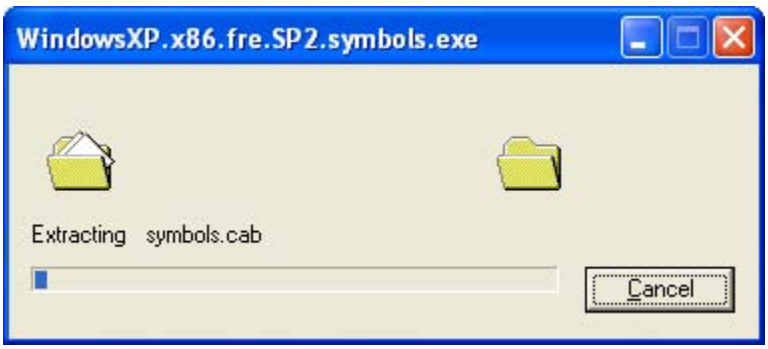

Figure 17: The Windows XP with Service Pack 2 retail symbols file extraction

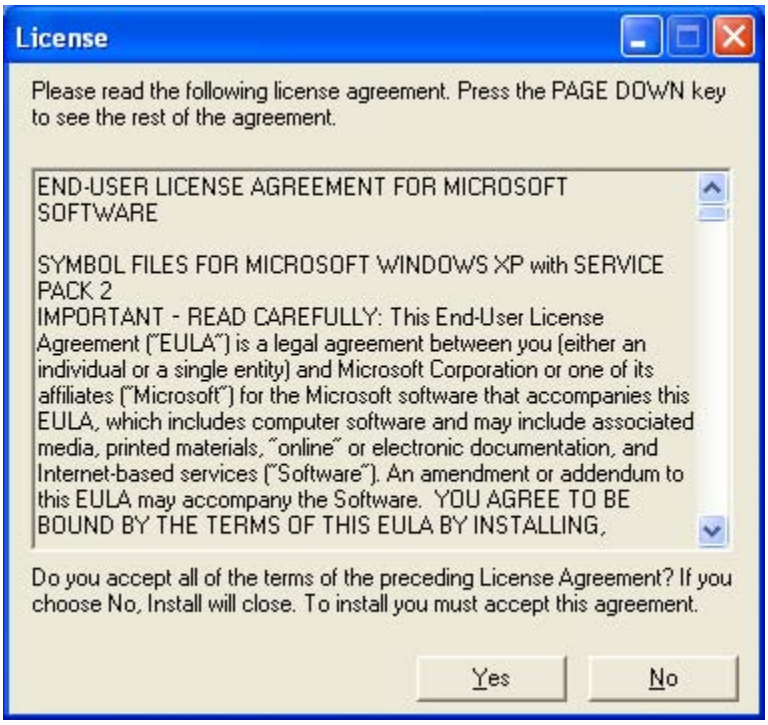

Figure 18: The Windows XP with Service Pack 2 retail symbols License agreement

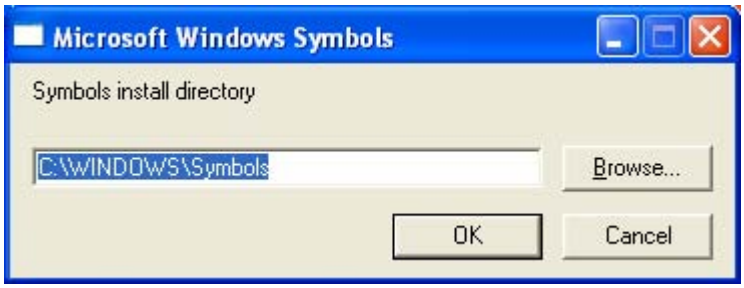

Figure 19: The Windows XP with Service Pack 2 retail symbols default installation path

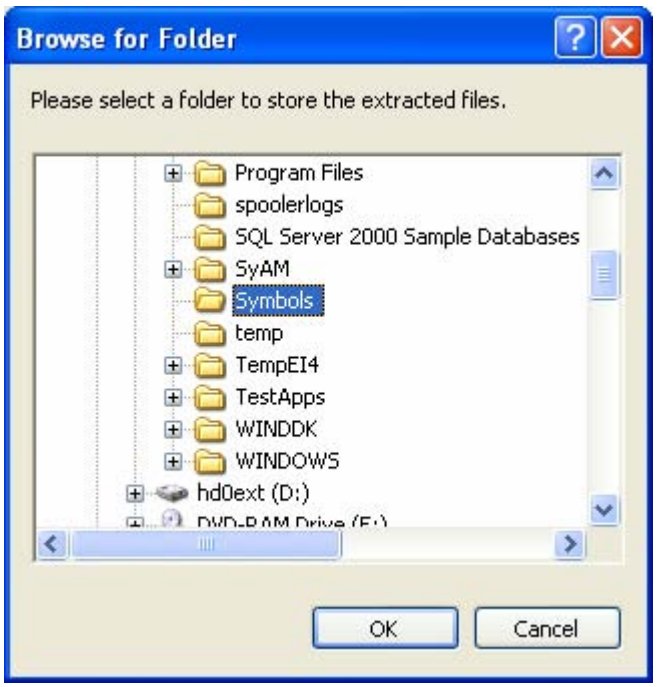

Figure 20: The Windows XP with Service Pack 2 retail symbols custom directory installation selection

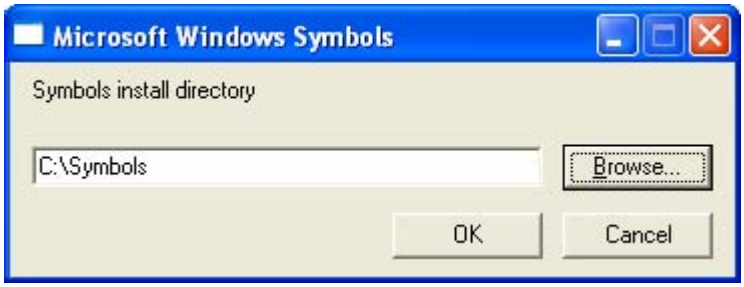

Figure 21: The Windows XP with Service Pack 2 retail symbols package installation path

Figure 22: The Windows XP with Service Pack 2 retail symbols files copying in action

Figure 23: The Windows XP with Service Pack 2 retail symbols package installation is complete

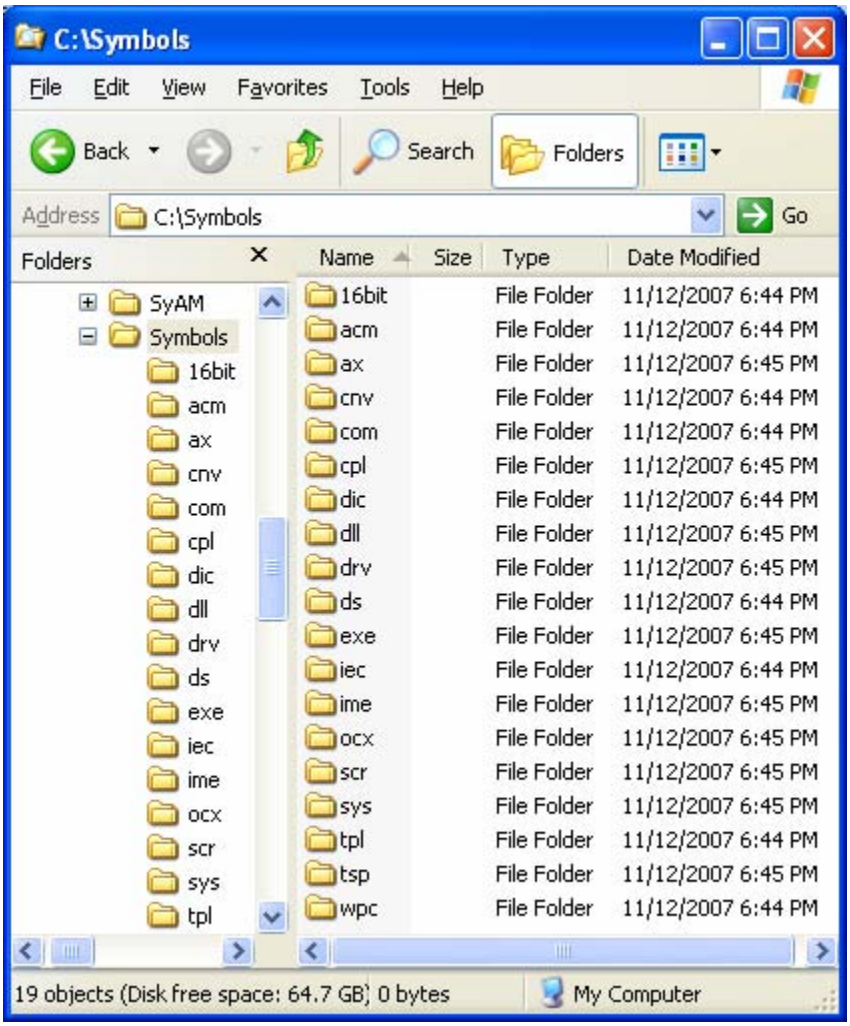

Figure 24: The Windows XP with Service Pack 2 retail symbols installed directories

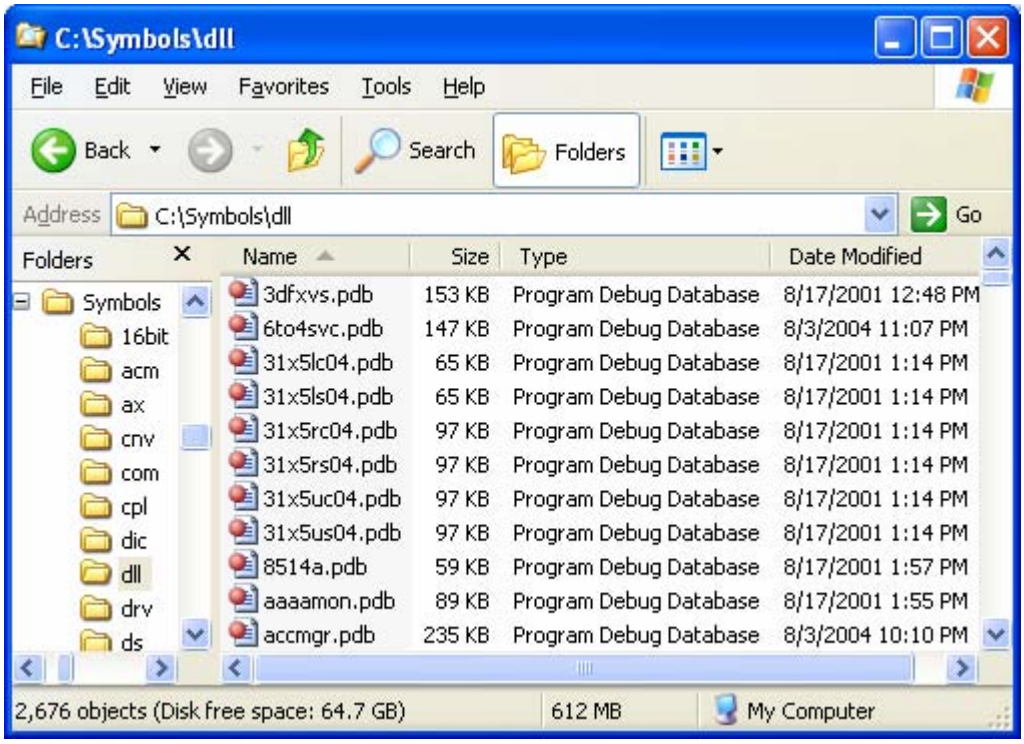

Figure 25: The Windows XP with Service Pack 2 retail symbols PDB files examples

After executing windgb, you need to set the path to the symbol files. This path points to the directories with the pdb files of your drivers. You can have different directories by separating them with a semicolon (`;`). You also need to point to the corresponding PDB files of all the windows components, if you want the call stacks that you'll see to include the functions from the components that are developed by Microsoft.
However, the problem in this case is that the windows PDB files change between service packs, hotfixes, etc. Fortunately, Microsoft has configured a symbol server, which can be used to download the needed files on-demand. This means that you just set the symbol path to the symbol server and windbg downloads only the PDB files that it needs. In order to do this, you need to add an entry to the windbg symbol path that's equal to:

```
SRV*DownstreamStore*http://msdl.microsoft.com/download/symbols
```

In the above line, `http://msdl.microsoft.com/download/symbols` is the symbol server then you don't need to modify this), and `DownstreamStore` is the path, where you want the pdb files to be downloaded. This needs to be substituted by a local directory, e.g. C:\Symbols, so the complete entry would be:

```
SRV*c:\Symbols*http://msdl.microsoft.com/download/symbols
```

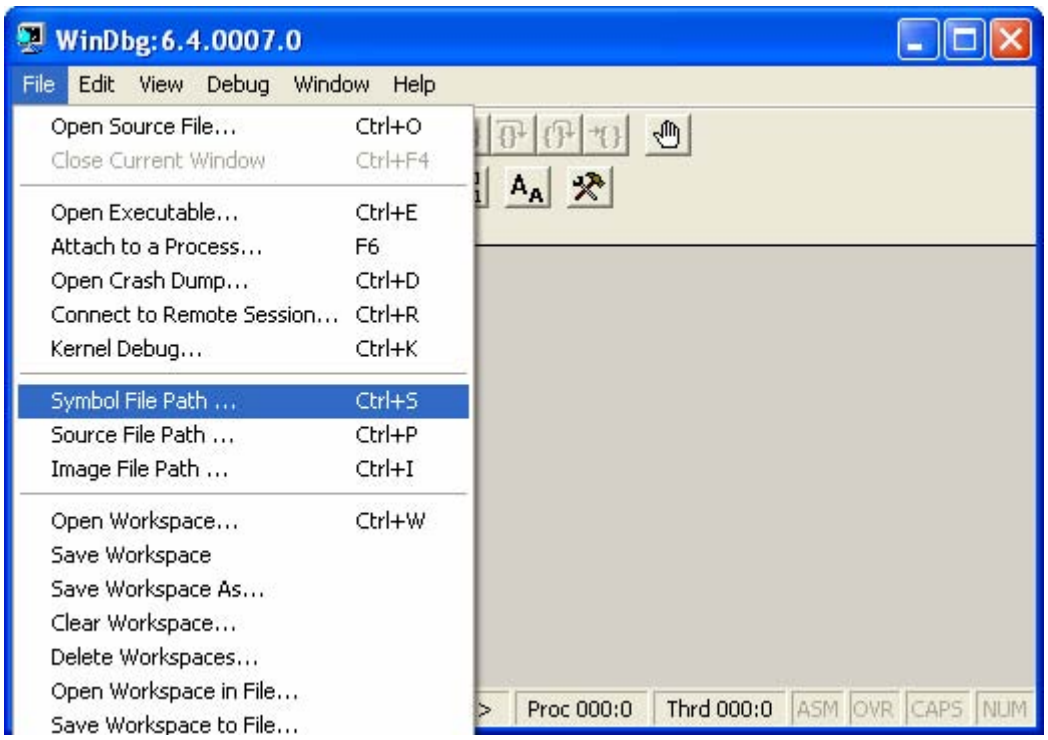

Figure 26: Setting the Symbol File Path

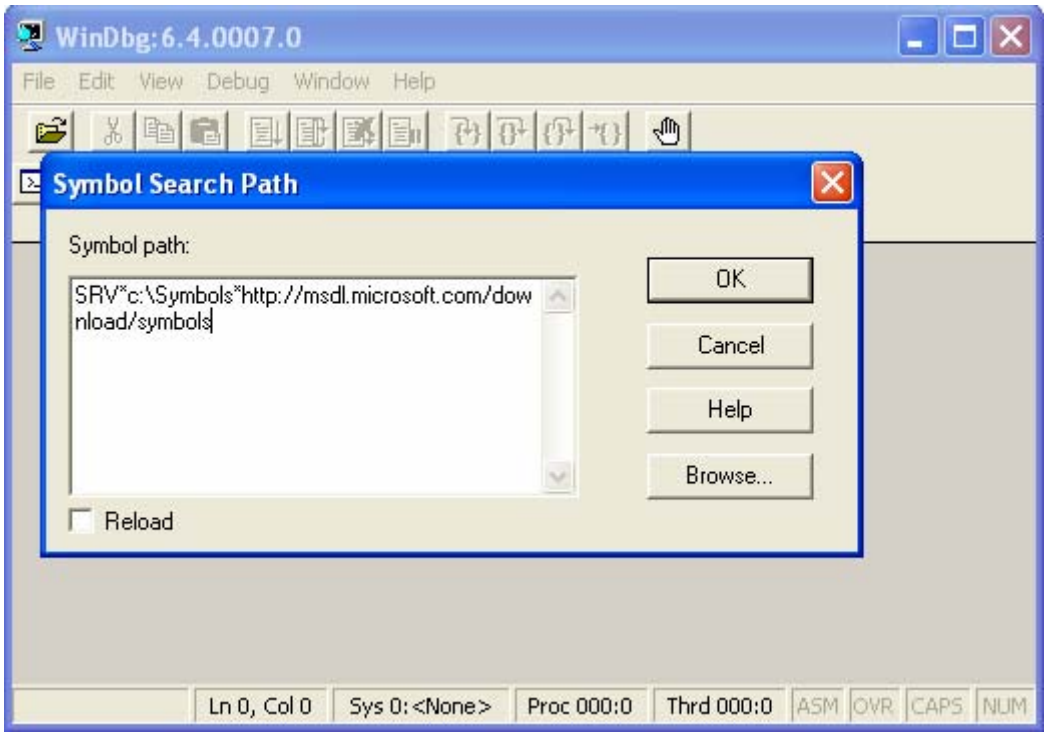

Figure 27: Setting the PDB Symbol File Path for the on demand download

Finally, if you have additional pdb files for the drivers that you are developing in directories `C:\mydrivers1`, `C:\mydrivers1\misc` and `C:\mydrivers2`, the complete symbol path would be:

```
SRV*c:\websymbols*http://msdl.microsoft.com/download/symbols;c:\drive
rs1;c:\drivers1\misc;d:\drivers2
```

Also, as it can be seen from the above example, the directories in the path aren't recursive, so if you have PDB files both in `C:\mydrivers1` and `C:\mydrivers1\misc`, then you need to include both of them, since the format doesn't imply the latter.
In order to set this line, you need to open windbg, go to `File` → `Symbol File Path` and paste the line in the textarea.
In our case the PDB files already installed under the `C:\Symbols` directory, so we just point to the directory in the Symbol File Path. Launch the Windbg program.

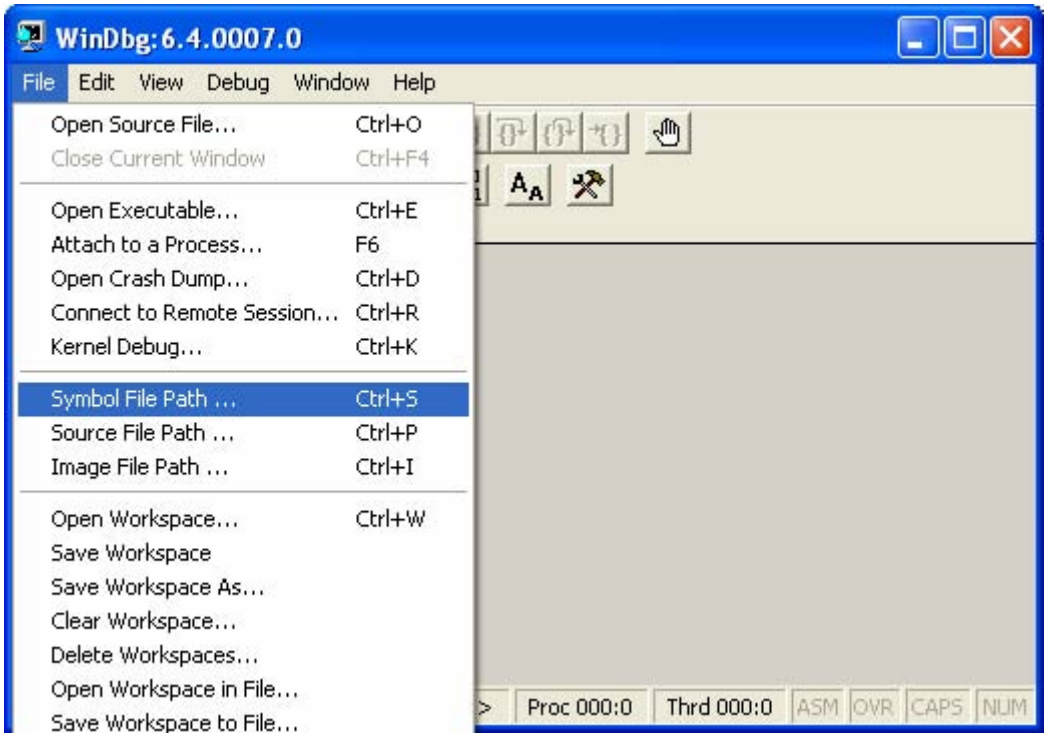

Figure 28: Invoking the Symbol File Path setting page

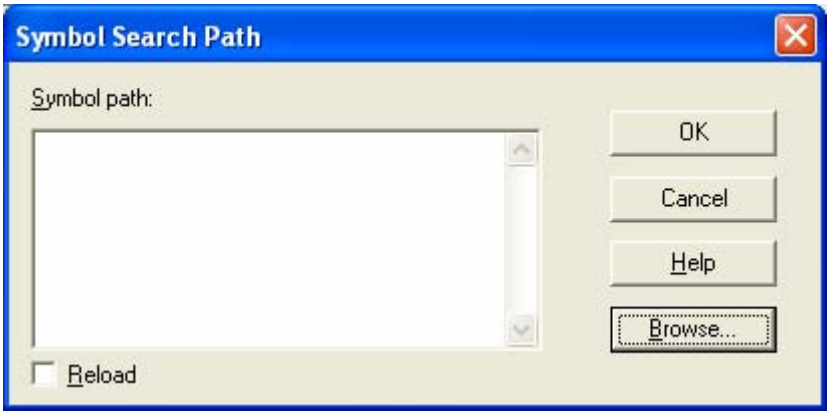

Figure 29: The Symbol File Path setting page

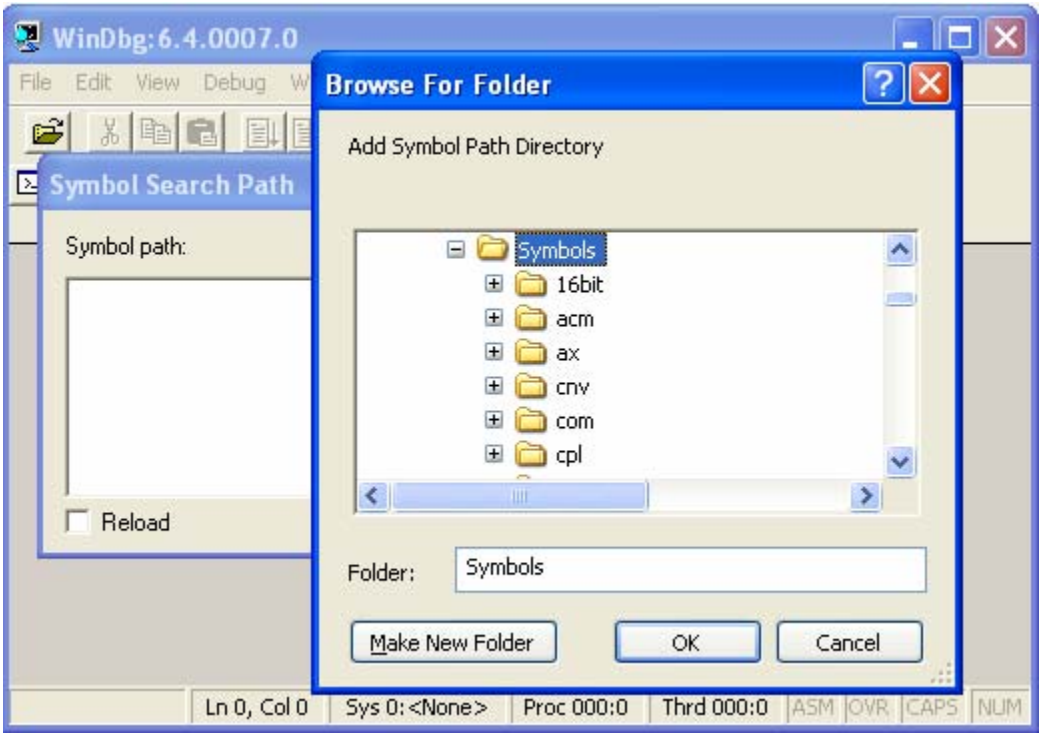

Figure 30: Selecting the `C:\Symbols` as the Symbol File Path

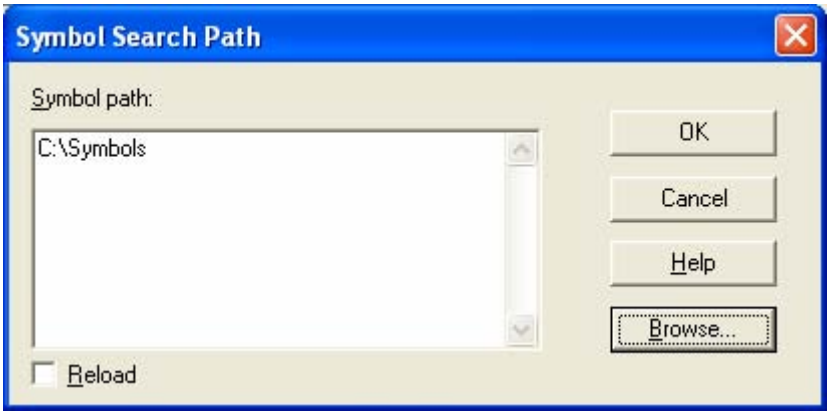

Figure 31: The Symbol File Path has been set to the local `C:\Symbols` directory

Don’t forget to save the changes that have been done.

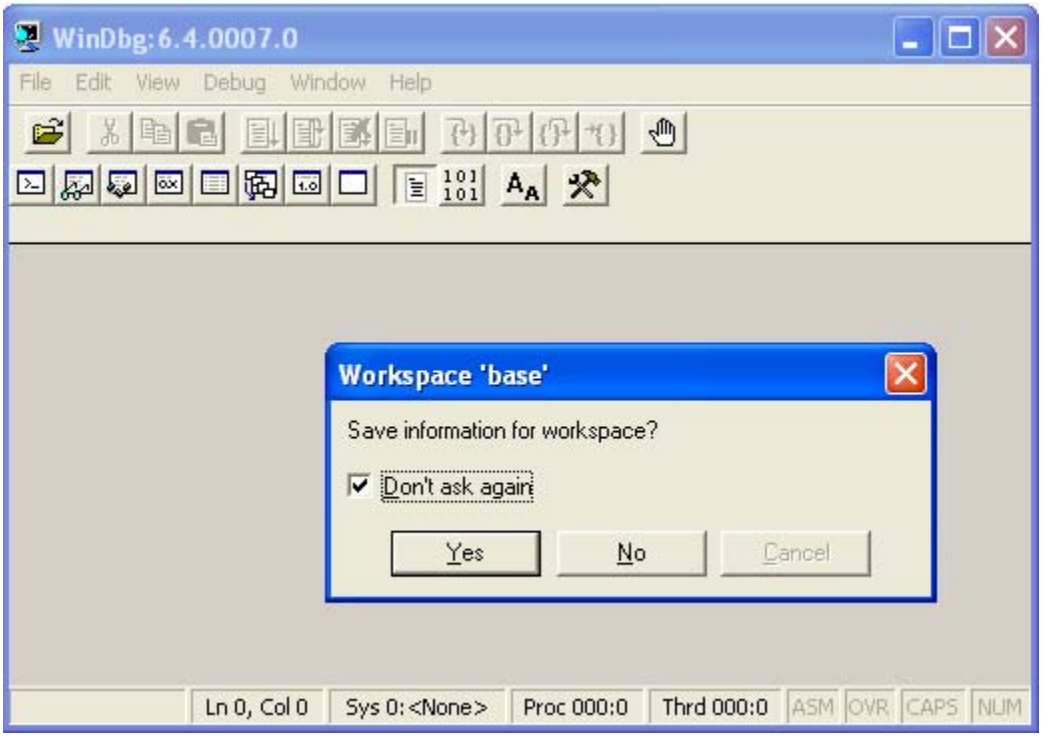

Figure 32: Saving the WinDbg changed workspace

More complete information should be found in the `Help`.

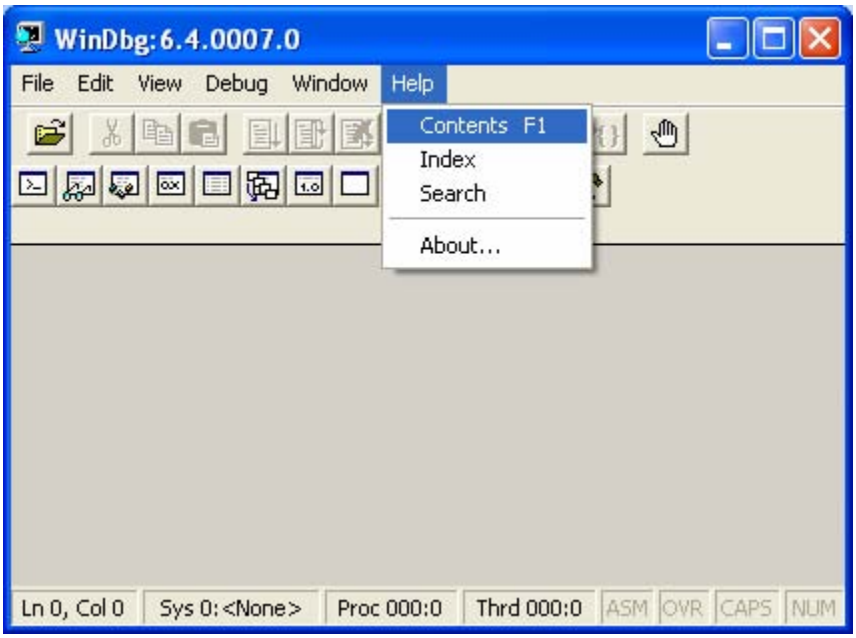

Figure 33: Invoking the WinDbg Help

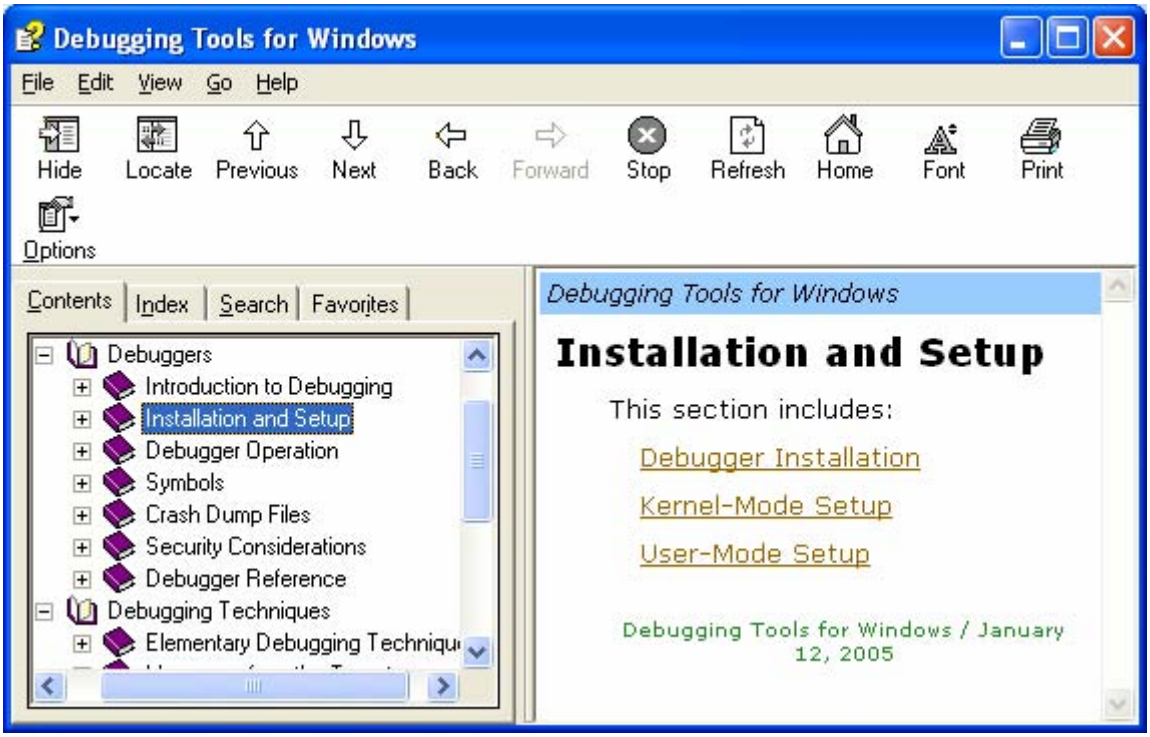

Figure 34: The WinDbg Help

Kernel-mode memory dump files can be analyzed by WinDbg. The processor or Windows version that the dump file was created on does not need to match the platform on which KD is being run.

Starting WinDbg

To analyze a dump file, start WinDbg with the `-z` command-line option:

```
windbg -y SymbolPath -i ImagePath -z DumpFileName
```

The `-v` option (verbose mode) is also useful. If WinDbg is already running and is in dormant mode, you can open a crash dump by selecting the `File` → `Open Crash Dump` menu command or pressing the CTRL+D shortcut key.

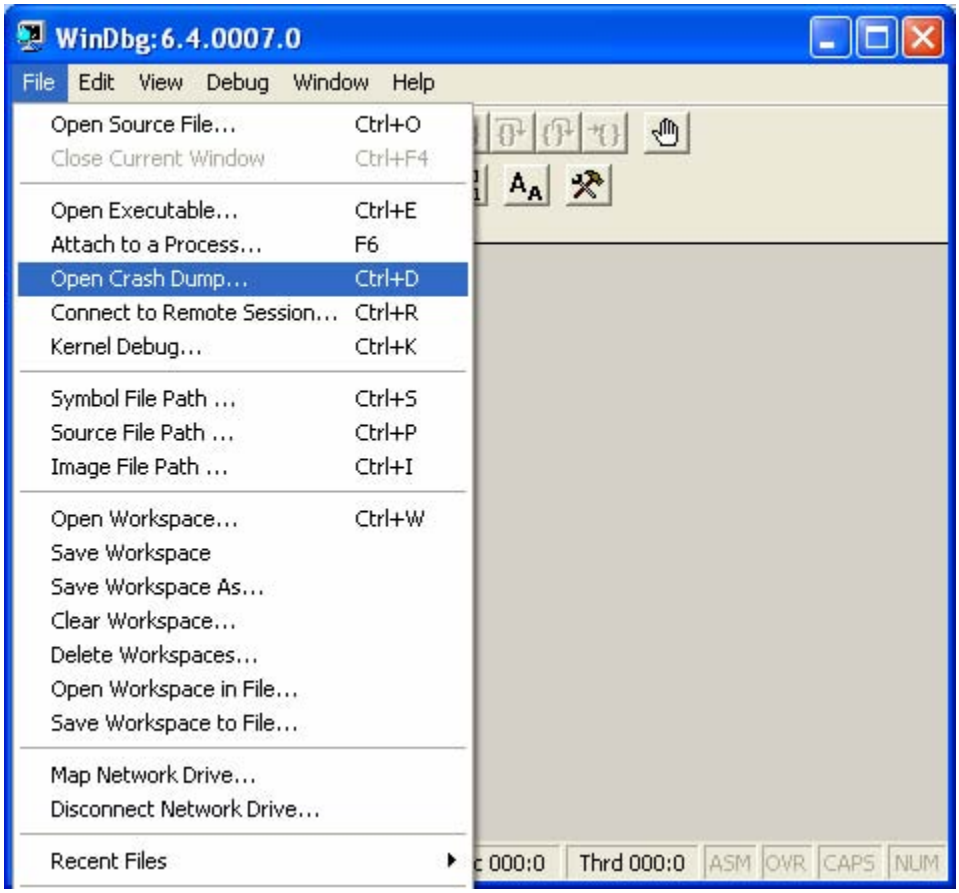

Figure 35: Opening the Windows Crash Dump file

When the `Open Crash Dump` dialog box appears, enter the full path and name of the crash dump file in the `File name` text box, or use the dialog box to select the proper path and file name. When the proper file has been chosen, click `Open`.

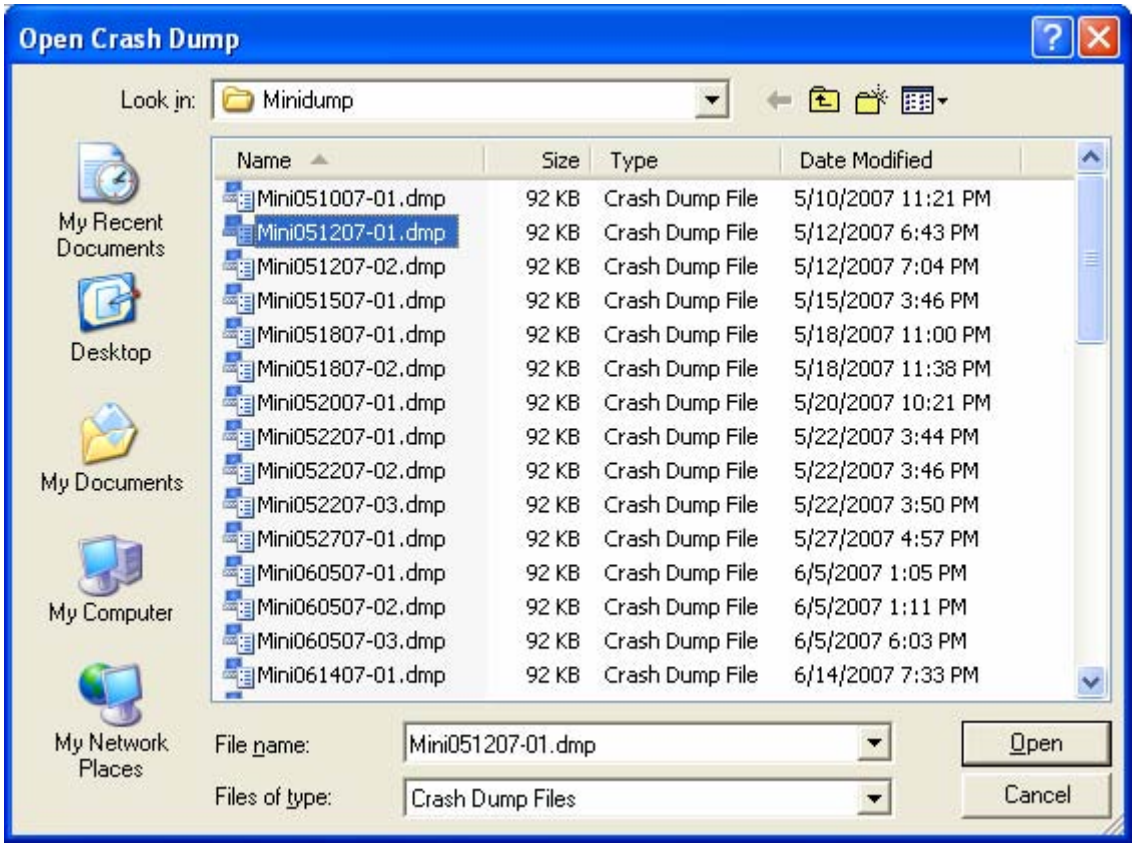

Figure 36: Selecting the Windows mini Crush Dump file

You can also open a dump file after the debugger is running by using the `.opendump` (`Open Dump File`) command, followed with `g` (`Go`).
Dump files generally end with the extension `.dmp` or `.mdmp`. You can use network shares or Universal Naming Convention (UNC) file names for the memory dump file. Well, it will take a long story to provide examples on how to debug either the user mode or kernel mode and why not you try the following links by Windows device driver developer for more information.

1. Windbg basic tutorials.
2. A Crash Dump analysis tutorials.
3. Tips on how to analyze strange Crash Dumps and uninstall the Windows hidden drivers.

Verifying DDK Installation

To verify that you have properly installed the Microsoft Windows Server 2003 Service Pack 1 (SP1) Driver Development Kit (DDK), complete the following steps:

1. Start a `Windows XP Checked Build Environment Command Prompt` window:

   - Click the `Start` button, and then click `All Programs`.
   - Point to `Development Kits`, point to your selected `Windows XXXX`, point to `Build Environments`, and then point to `Windows XP` (or your chosen Windows OS).

- Click `Windows XP Checked Build Environment`.

Figure 37: Running the Checked Build Environment for Windows XP

2. At the command prompt, type `build -cZ` to compile and link the complete set of installed sources. This command can take more than 30 minutes to complete, depending on which DDK components are installed and the system on which the build is being performed.

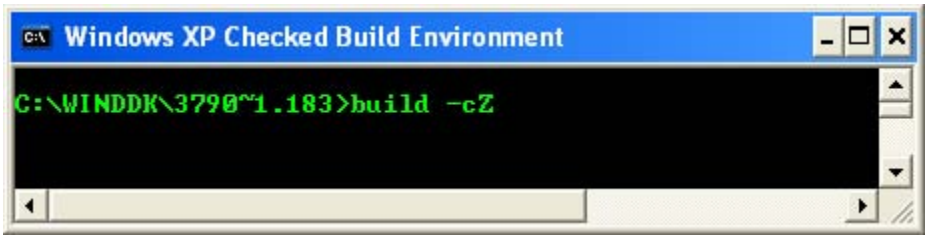

Figure 38: Running the `build` command

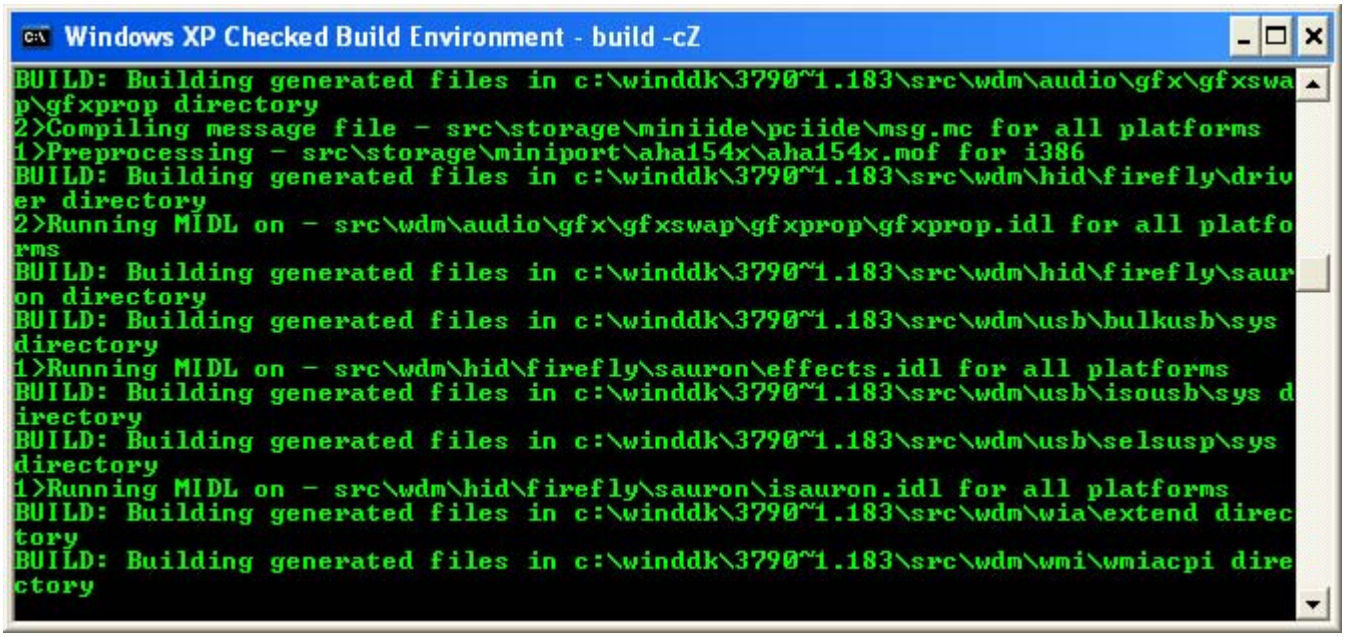

Figure 39: The Checked Build Environment for Windows XP in action

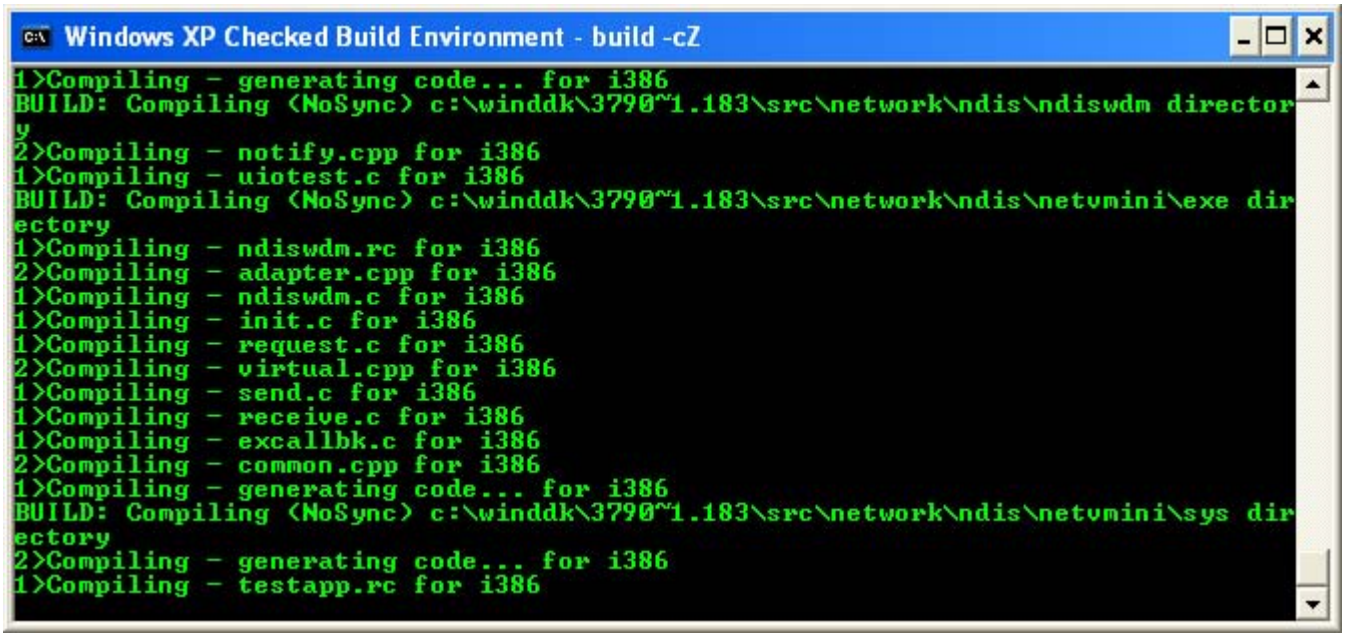

Figure 40: More action for the Checked Build Environment for Windows XP

3. The build command should complete with no errors and no warnings displayed but here we have some warnings.

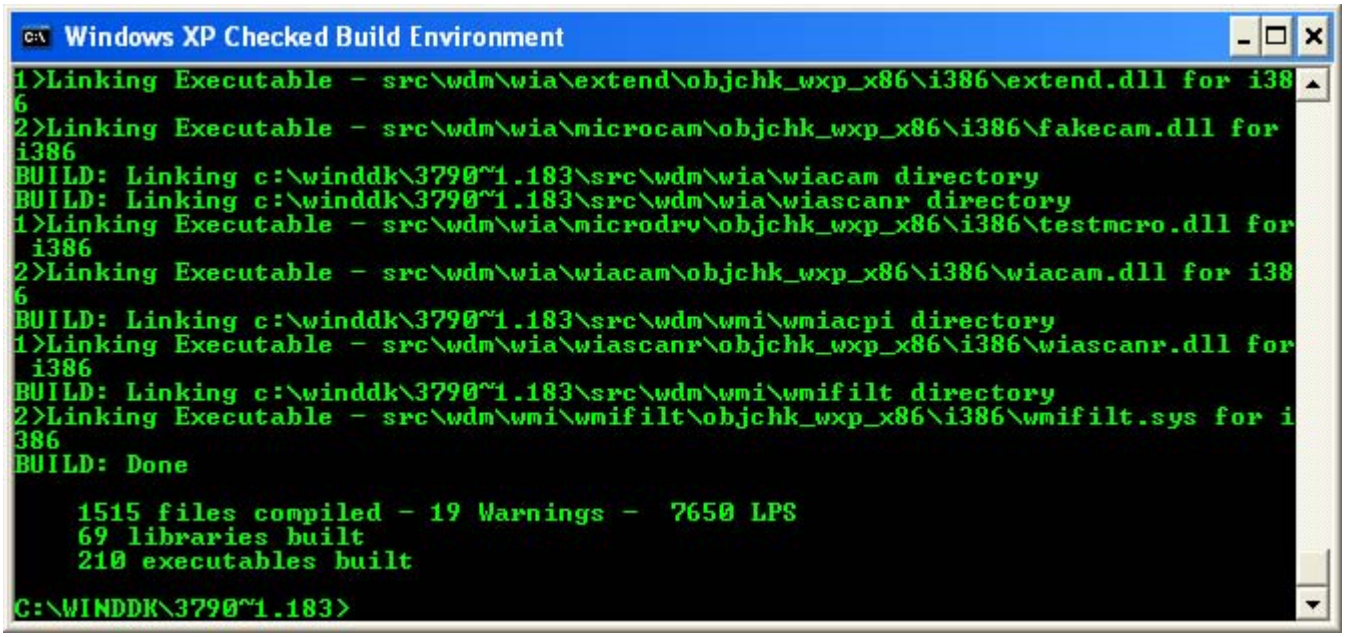

Figure 41: The Checked Build Environment for Windows XP is complete with warnings

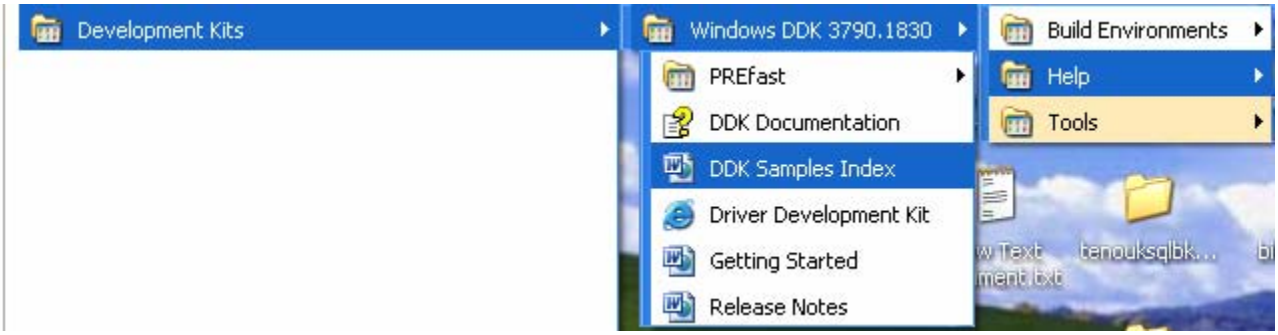

Figure 42: Viewing the DDK samples index files

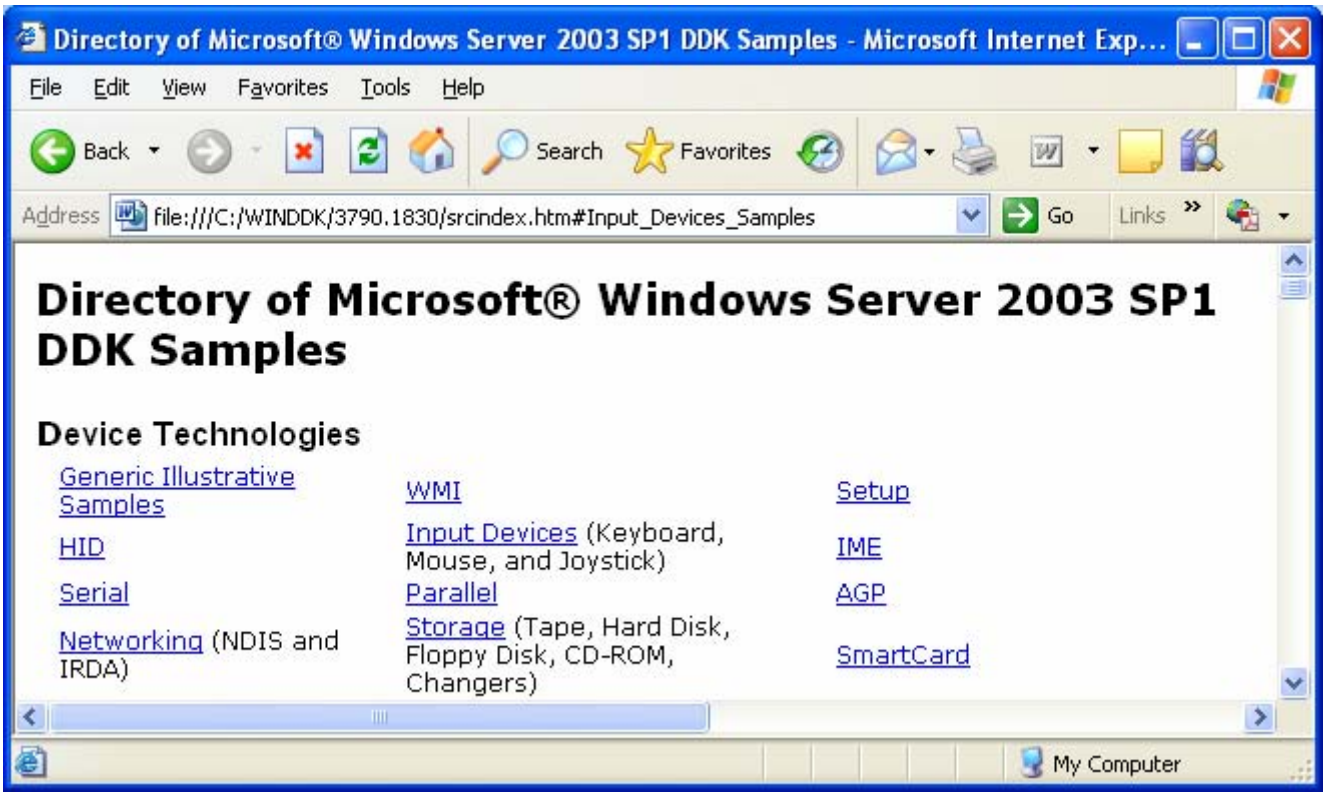

Figure 43: The DDK sample index files