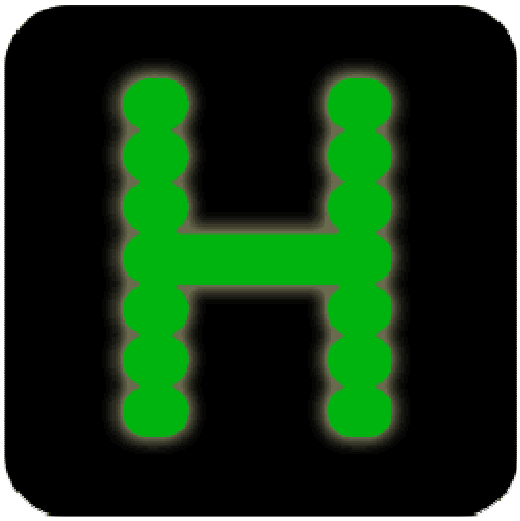
Hercules System/370, ESA/390, z/Architecture Emulator

Hercules - Installation Guide (Windows)

Version 3



Contents

Figures

Tables

1. Preface

1.1 Edition information

This edition applies to the Hercules S/370, ESA/390 and z/Architecture Emulator Release 3.05.0 and to all subsequent versions, releases and modifications until otherwise indicated in new editions. Make sure you are using the correct edition for the level of software you are using.

1.2 What this book is about

This book is a guide to installing the Hercules Emulator and related additional products (both required and optional) under the Microsoft Windows operating system. For guidance in operating or debugging Hercules or for a general overview additional manuals are available.

Please note that some information can be found in more than one manual. This redundancy is not intended to unnecessarily expand the manuals, rather to help to find all necessary information in one place.

1.3 Who should read this book

This book is mainly intended for people who are responsible for installation and maintenance of the Hercules Emulator. It may also be useful if you are responsible for operating the Hercules Emulator.

1.4 What you need to know to understand this book

To understand this book you should be familiar with installing software under the Windows™ (XP, W2K) operating system. You should also have experience with native DOS™ (Microsoft Disk Operating System) and the Microsoft Windows Command Shell. Some knowledge of TCP/IP configuration in a small network is required to configure network connectivity.

Last but not least you should be familiar with IBM mainframe environments (hardware and software) and the underlying ideas and concepts as Hercules emulates IBM mainframe hardware.

1.5 How to use this book

This book is designed as a step by step installation guide for the Hercules Emulator and related products. You should go through the book chapter by chapter and follow all the instructions given. This should lead to an easy and fast installation without major problems.

1.6 Revision Notice

Hercules Release:	Version 3 Release 05 Modification 0
Publication Number:	HEIW030500
SoftCopy Name:	HerculesWindowsInstall
Revision Number:	HEIW030500-00
Date:	September 1, 2007

1.7 Readers Comments

If you like or dislike anything about this book please send an email to the address below. Feel free to comment on any errors or lack of clarity. Please limit your comments on the information in this specific book and also include the “Revision Notice” just above. Thank you for your help.

Send your comments by email to the Hercules-390 discussion group:

hercules-390@yahoogroups.com

1.8 Legal Advice

Hercules implements only the raw S/370, ESA/390, and z/Architecture instruction set, it does not provide any operating system facilities. This means that you need to provide an operating system or standalone program which Hercules can load from an emulated disk or tape device. You will have to write the operating system or standalone program yourself unless you possess a license from IBM to run one of their operating systems on your PC or use IBM programs and operating systems which have been placed in the public domain.

NOTE: It is YOUR responsibility to comply with the terms of the license for the operating system you intend to run on the Hercules Emulator.

1.9 Trademarks

The following is a list of trademark acknowledgments and copyright notices for product and company names mentioned in this book. Other product and company names in this book that are not listed below may be the trademarks or registered trademarks of their respective owners.

- IBM, System/370, ESA/390, z/Architecture, MVS, OS/390, z/OS, VM, VM/ESA, z/VM, VSE, VSE/ESA, z/VSE are trademarks or registered trademarks of International Business Machines Corporation (IBM).
- Windows 95, Windows 98, Windows ME, Windows NT, Windows 2000, Windows XP, Windows Vista, Windows Server 2000, Windows Server 2003, Visual C++ Toolkit 2003, Visual C++ 2005 Express are trademarks of Microsoft Corporation.
- WinPcap is copyrighted by NetGroup, Politecnico di Torino (Italy).
- Cygwin is copyrighted by Red Hat, Inc.
- Vista tn3270 is copyrighted by Tom Brennan Software.
- Pentium, XEON are trademarks or registered trademarks of Intel Corporation.
- Athlon, Opteron are trademarks or registered trademarks of Advanced Micro Devices (AMD), Inc.
- Xmit Manager is copyrighted by Neal Johnston-Ward.
- FLEX-ES is a registered trademark of Fundamental Software, Inc.
- UMX Virtual Mainframe is a registered trademark of UMX Technologies.

1.10 Acknowledgements

The Hercules manuals would not have been possible without the assistance of many people and I would like to thank all those who helped me. In particular I would like to thank:

- The Hercules developers for their documentation on various websites from which I derived a great deal of information.
- Roger Bowler and Fish for proof-reading the manuals.
- Loris Degoianni for allowing me to use parts of the original WinPcap documentation.
- Tom Brennan for allowing me to use parts of his Vista tn3270 documentation.
- My colleagues for working with early previews of the documentation, beginning with just a few pages.
- Mike Cairns for reviewing and editing the manuals.

If anyone feels they have been forgotten on this list please let me know.

Peter Glanzmann

2. Related Publications

2.1 Hercules Emulator – General Information

The Hercules “General Information” manual provides an overview of the ideas and concepts of the Hercules Emulator as well as documentation of the emulator’s functionality. It explains what Hercules does and does not do. It helps you decide if the software fits your needs and fulfills your requirements.

2.2 Hercules Emulator – Installation Guide (Windows)

The Windows installation guide shows you how to install Hercules and all related optional and required software components under the Microsoft Windows operating system. After going through the installation guide you will have a working emulator environment ready to IPL the S370, S/390 or z/Architecture mainframe operating systems.

2.3 Hercules Emulator – Installation Guide (Linux)

The Linux installation guide shows you how to install Hercules and all related optional and required software components under the Linux operating system. After going through the installation guide you will have a working emulator environment ready to IPL the S370, S/390 or z/Architecture mainframe operating systems.

2.4 Hercules Emulator – Installation Guide (Mac OS X)

The Mac OS X installation guide shows you how to install Hercules and all the related optional and required software components under the Apple MacIntosh OS X operating system. After going through the installation guide you will have a working emulator environment ready to IPL the S370, S/390 or z/Architecture mainframe operating systems.

2.5 Hercules Emulator – User Reference Guide

The Hercules “User Reference” leads you through all aspects of the emulator’s operation. It provides instruction in the operation of the Hercules Emulator with and without the Windows GUI. The usage details for all Hercules utilities are also covered in this guide.

After reading this manual you should be able to work with Hercules and the Hercules console, create virtual devices, understand backup / restore procedures and general housekeeping within the Hercules environment.

2.6 Hercules Emulator – Messages and Codes

The “Messages and Codes” manual provides a detailed explanation of all Hercules related messages. It is the primary source for troubleshooting and debugging when you experience problems with Hercules.

3. Hardware Prerequisites

3.1 PC Hardware

The following section lists the requirements for various hardware components within the context of the following categories:

- Minimal (minimum required equipment)
- Moderate (average older PC equipment)
- Average (typical current PC equipment)
- Good (fast, newer PC environment)
- Optimal (recommended server equipment for best performance)

It is possible to run Hercules with less than the minimal recommended hardware but performance of any operating system executing under the emulator will be severely constrained. For acceptable performance you will need a configuration categorized as “average” or better. If you have access to PC hardware categorised as “optimal” the performance of installed OS’s will be adequate for most practical purposes.

Most modern PC systems are delivered with adequate RAM, Processor and Disk for Hercules use. It has been stated in previous Hercules documentation that a Pentium 200 MHz with 32 MB of RAM would be sufficient for the emulator, however today such a system is no longer practical.

3.1.1 Processor

Hercules does not necessarily depend on the Intel Pentium architecture. It also has been built, installed and run successfully on an Alpha 21164, SPARC and on z/Architecture Linux/390 systems. One of the most extravagant implementations for testing purposes has run OS/360 under Hercules under Linux/390 under Hercules under Linux/390 under VM/ESA. This is of course an extreme example of emulation layers. This document however only describes the implementation of the Hercules Emulator on an Intel based or compatible PC system.

Hercules will benefit greatly from a fast processor, the faster the processor the better Hercules will run. If you can employ a multiprocessor or dual-core system, so much the better. The Hercules Emulator makes extensive use of multithreading to overlap I/O with CPU activity. A multiprocessor board with two slower processors will in most cases outperform a uniprocessor board with a faster processor.

- Pentium with 500 MHz or equivalent processor (minimal)
- Pentium with 1 GHz or equivalent processor (moderate)
- Pentium with 2 GHz or equivalent processor (average)
- Pentium with 3 GHz and HyperThreading or equivalent processor (good)
- 2 (or more) Dual Core XEON's with 2.66 GHz or equivalent processors (optimal)

3.1.2 RAM

The more RAM installed in the system the better Hercules will perform. For maximum throughput you should set your main and expanded storage sizes in the Hercules configuration file high enough to eliminate installed operating system paging operations as much as possible. The S/390 guest system storage is allocated out of the Linux or Windows host operating system storage so try to provide enough RAM to your system to eliminate Linux or Windows paging as well.

- 128 MB RAM (minimal)
- 512 MB RAM (moderate)
- 1024 MB RAM (average)
- 2048 MB RAM (good)
- 4096 MB RAM (optimal)

There is a limit to the memory that Hercules can allocate under Windows (approximately 1 GB), normally 2 GB of installed RAM in the machine should be sufficient to allow for both Hercules and the host OS. If however you plan to run more than one instance of Hercules on the same machine, each instance can allocate 1 GB of RAM for its own use. In this case it makes sense to install more than 2 GB RAM.

Please note that you still can allocate more RAM than is physically available on the PC to the Hercules emulator for use of the installed operating system(s). But in this case the Linux or Windows host operating system has to page out the missing physical RAM to its own swap files which will seriously degrade Hercules performance.

3.1.3 Disk Storage

The disk storage requirements for Hercules depend entirely on your requirements. The runtime Hercules system requires only small amount of disk space. You will need a little more space if you plan to build Hercules from source code. But still the requirements are quiet modest.

Although Hercules and the necessary software components do not require very much disk storage, you need enough hard disk space to accommodate the emulated DASD volumes for the operating system you choose to run under Hercules.

For a minimal z/OS system with CICS, IMS, DB2, WAS etc., without any user data, you need at least 15 disks of type 3390 model 3 which requires 42.5 GB hard disk space. If you extend such a system with some user data for software development etc you may need 10 to 15 additional 3390 model 3 disks. This could lead to a total hard disk requirement of 85 GB. If you plan to do uncompressed backups of all the DASD volumes you can double this number.

The following table shows how much space is occupied for each virtual DASD volume on your PC hard disk(s) for some of the supported device types. If you make use of the compressed CKD DASD feature of Hercules these sizes will shrink dramatically. Usually to about 20 to 30 percent of the original size. Space savings depend on the actual used capacity within the virtual DASD volumes.

Model	Cylinder	Bytes/Track	Bytes/Cylinder	Bytes/Volume
3380-J	885	47’476	712’140	630 MB
3380-E	1’770	47’476	712’140	1.26GB
3380-K	2’665	47’476	712’140	1.89 GB
3390-1	1’113	56’664	849’960	946 MB
3390-2	2’226	56’664	849’960	1.89GB
3390-3	3’339	56’664	849’960	2.83 GB
3390-9	10’017	56’664	849’960	8.51 GB
3390-27	32’760	56’664	849’960	27.84 GB
3390-54	65'520	56'664	849'960	55.68 GB
9345-1	1’440	46’456	849’960	1.0 GB
9345-2	2’156	46’456	849’960	1.5 GB

Table 1: DASD Device Capacity

4. Software Prerequisites

4.1 Operating System

Hercules is an open source software implementation of the mainframe System/370, ESA/390 and z/Architecture hardware. Hercules itself is not an operating system nor does it emulate a mainframe operating system. The Hercules Emulator runs under Linux on several hardware platforms including the Intel Pentium PC, under various flavours of Microsoft Windows and under MAC OS X. From the point of view of the underlying operating system the Hercules Emulator is just an application program.

This guide focuses solely on the installation of Hercules under Microsoft Windows. Details of other host operating systems are not covered in this book. The installation of any hosted operating system and software utilities is beyond the scope of this book.

4.1.1 Windows Versions

The Hercules Emulator runs under Windows 98, Windows NT, Windows 2000, Windows XP, Windows Vista, Windows Server 2000 and Windows Server 2003. Although Windows 98 and Windows NT are possible candidates for running Hercules, their use is not recommended because of limitations in stability (especially Windows 98), limited networking capabilities and lack of support. It is generally a good idea to have a current operating system maintained with the latest fixes i.e. Windows XP / Vista or Server 2003.

Some users experienced problems with TCP/IP functionality when running under Windows XP with Service Pack 2 while other users did not encounter these issues. The presenting problem with this configuration is: It is possible to connect from the Host operating system to the Hercules machine and vice versa, however it is not possible to get a connection to another workstation on the LAN or the Internet. Most of these issues are related to configuration problems with the Windows XP firewall, introduced with Service Pack 2. Creating correct rules in either the Windows internal firewall or an alternative firewall product solves these problems.

4.1.2 Stability

As Hercules appears to Windows as just another application program, it is possible to run other applications simultaneously with Hercules, however this is not recommended. While the Hercules Emulator is extremely stable software it is impossible to guarantee that other applications will not degrade Hercules performance or even crash the Windows system. The less software you have installed on your system the more stable your emulated mainframe will be.

Intensive tests have proven the stability of Hercules, one of the test suites is repeated with every new Hercules release. The tests run a mixed online and batch workload continuously and at high volumes. The base system consists of a MVS 3.8J operating system with additional components. During the test the system is loaded with between four and eight self submitting (therefore continuous) batch jobs executing in parallel. These jobs consist of a mix of large sorts, assemblies, compilation and link jobs.

Additionally there are some TSO sessions active (up to ten sessions) on which a simulated user presses enter once a second in a performance monitor application.

This workload keeps the system (emulated mainframe as well as Hercules and the base operating system) busy at nearly 100% CPU and a produces a continuous I/O rate of more than 750 SIOs. In this state the machine runs 7x24 hours during several days. One of these tests was stopped after more than 12 weeks with the system still running perfectly.

4.1.3 Installed Software on a Hercules System

It was recommended earlier to have only a minimum of software installed on a Hercules host machine. Additional to the software for the Hercules Emulator presented in this guide, it is recommended to have the following software installed:

- Windows 2000, Windows XP, Windows Server 2003 with latest service packs and hot fixes
- Software firewall, especially if there is no hardware firewall in the LAN
- Antivirus software with on demand and on access checks active

Depending on your requirements other commonly used utilities include:

- Network sniffer
- Performance monitor / Task monitor
- FTP program

In general though the less software is installed the better the emulated mainframe will run.

4.2 Drivers

This Chapter describes the device drivers that are necessary specifically for the Hercules Emulator. A device driver is a routine or a set of routines that implement the device-specific aspects of generic I/O operations. Generally the Hercules Emulator does not need any special device drivers other than those provided by the operating system. An exception is WinPcap, which is required under the Microsoft Windows operating systems to enhance networking capabilities of the native operating system.

4.2.1 WinPcap Packet Capture Driver

WinPcap is an architecture for packet capture and network analysis for the Win32 platform developed at Politecnico di Torino in Italy. The packet filter is a device driver that adds to Windows 95, 98, ME, NT, 2000, XP and Server 2003 the ability to capture and send raw data from a network card with the possibility to filter and store the captured packets in a buffer.

WinPcap includes an API that can be used to directly access the functions of the packet driver offering a programming interface independent from the Windows operating system. It also exports a set of high level capture primitives that are compatible with libpcap, the well known Unix capture library. These functions capture packets in a way independent from the underlying network hardware and operating system.

WinPcap is a free, public system and is released under a BSD-style license. It can be downloaded from www.winpcap.org where the necessary documentation can also be found.

4.3 Runtime Environments

This chapter describes special runtime environments which act as a base for running the Hercules Emulator. Under a Linux-like operating system no additional runtime environment is needed. Under Microsoft Windows however, the necessary Linux POSIX threads support may be emulated as explained below.

4.3.1 Cygwin

Until release 3.02.0 the Hercules Emulator was designed to run only under a Linux system with POSIX threads (pthread) support. To be able to run Hercules releases below 3.03.0 under Windows it is necessary to install a runtime environment to provide this Linux compatible layer. This is where Cygwin steps in.

Cygwin is a Linux-like environment for Microsoft Windows. It consists of several DLLs which act as an emulation layer providing substantial POSIX system call functionality and a collection of common Linux tools. The Cygwin environment works with all x86 versions of Windows since Windows 95.

Cygwin is available from www.cygwin.com. The Cygwin development began 1995 at Cygnus Solutions which is now owned by RedHat Software.

Cygwin is mentioned in all current Hercules manuals for support purposes only. Since release 3.02.0 of Hercules it is recommended that Windows users install the Microsoft Visual C native binaries or Microsoft Installer (MSI) package.

Beginning with Release 3.03.0 the Hercules Emulator no longer requires Cygwin in order to run under Windows. It is highly recommended that Windows users of Hercules begin using this new MSVC Win-32 version instead of the Cygwin versions.

4.4 Hercules Emulator

The Hercules Emulator consists of the following mandatory and optional components:

- Hercules Emulator (mandatory)
- Hercules Windows GUI (optional)
- CTCI-W32 (optional)
- Additional utilities (optional)

4.4.1 Hercules

The Hercules executables are the heart of the emulator and a mandatory component. This is the software implementation of the System/370, ESA/390 and z/Architecture mainframe hardware and processor machine code instruction set.

Hercules runs as a DOS program and comes with a semi-graphical display in a DOS window (the Hercules Hardware Console - HMC) consisting of two screens, switched between using the ESC key.

The following figure shows the initial display, the Hercules console window:

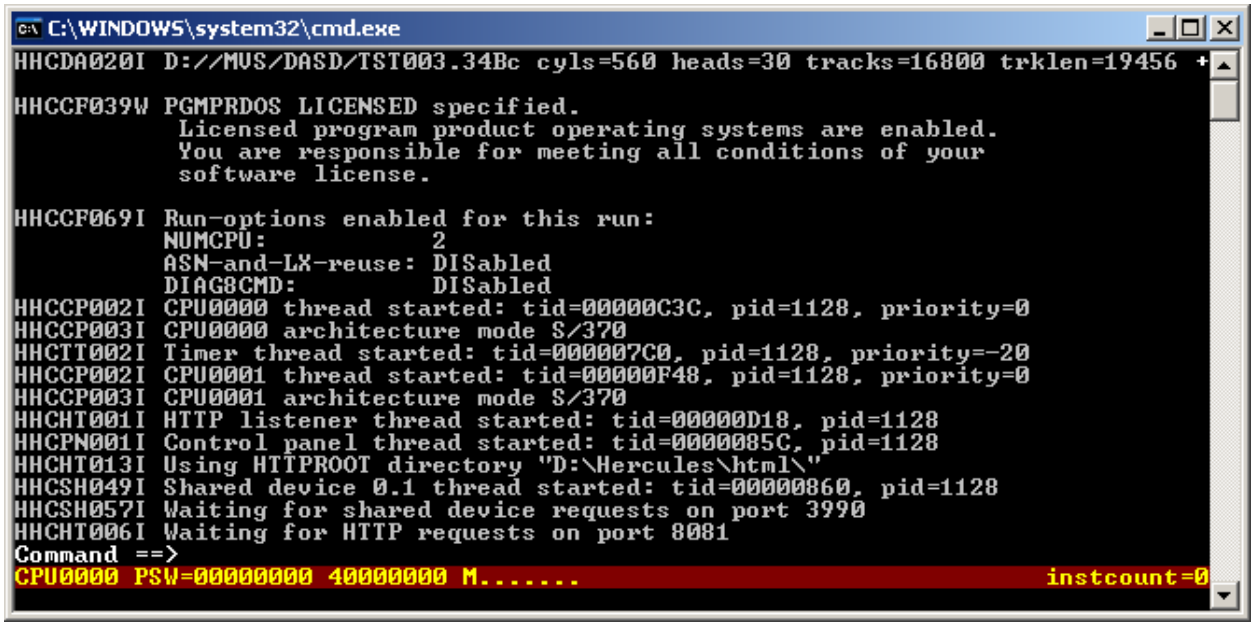


Figure 1: Hercules Hardware Console - Console window

The next figure shows the Hercules device and status display accessed with the ESC key:

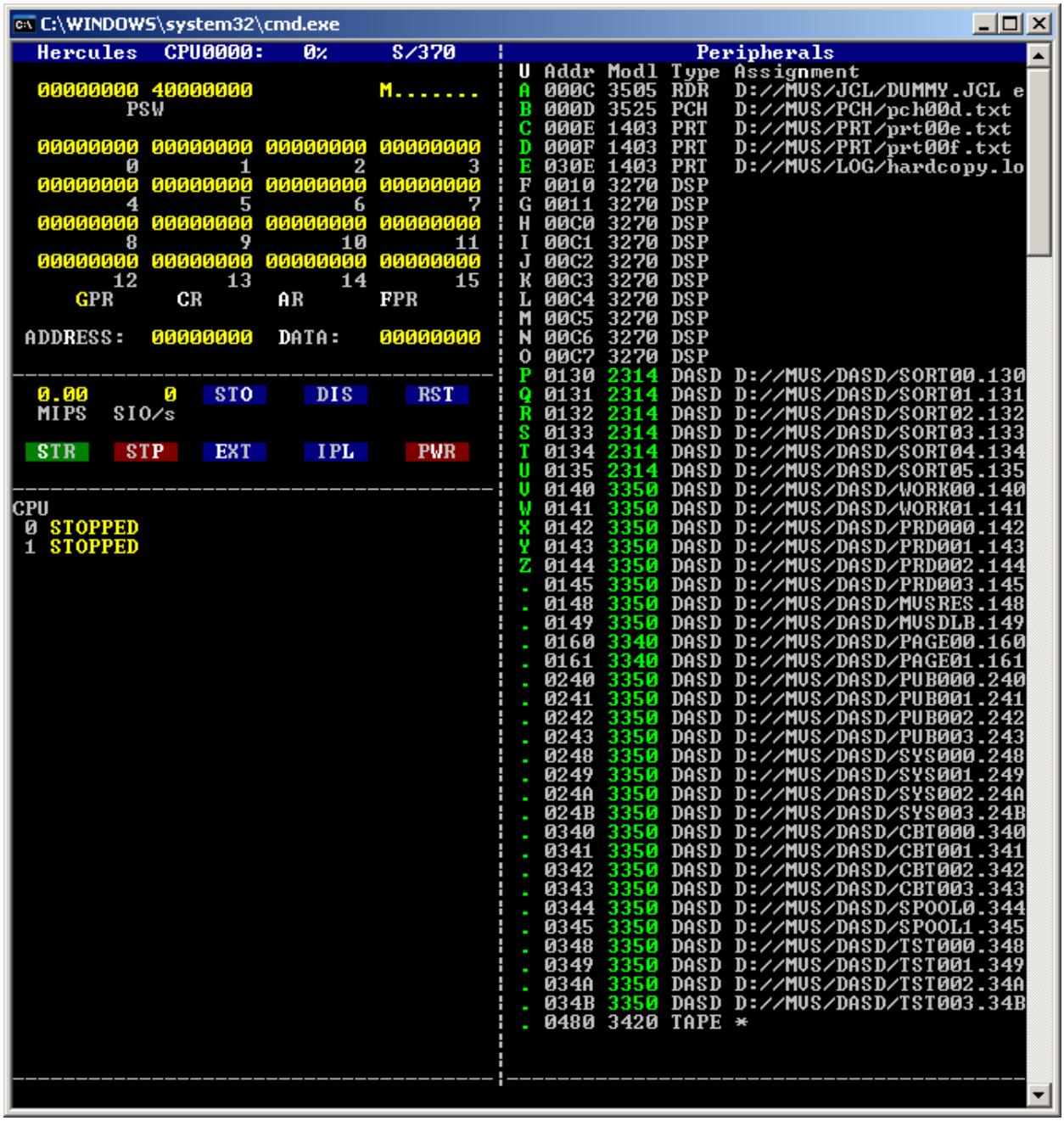


Figure 2: Hercules Hardware Console - Device and status display

4.4.2 Hercules Windows GUI

The Hercules Windows GUI (WinGUI) provides an optional graphical user interface to the Hercules Emulator replacing the native DOS console window of Hercules. The Windows GUI program interfaces with Hercules Emulator directly. It provides an easier way to work with the Hercules Emulator including interfaces to create / change the Hercules configuration files and the handling of log files.

The following figure shows the Hercules WinGUI main panel:

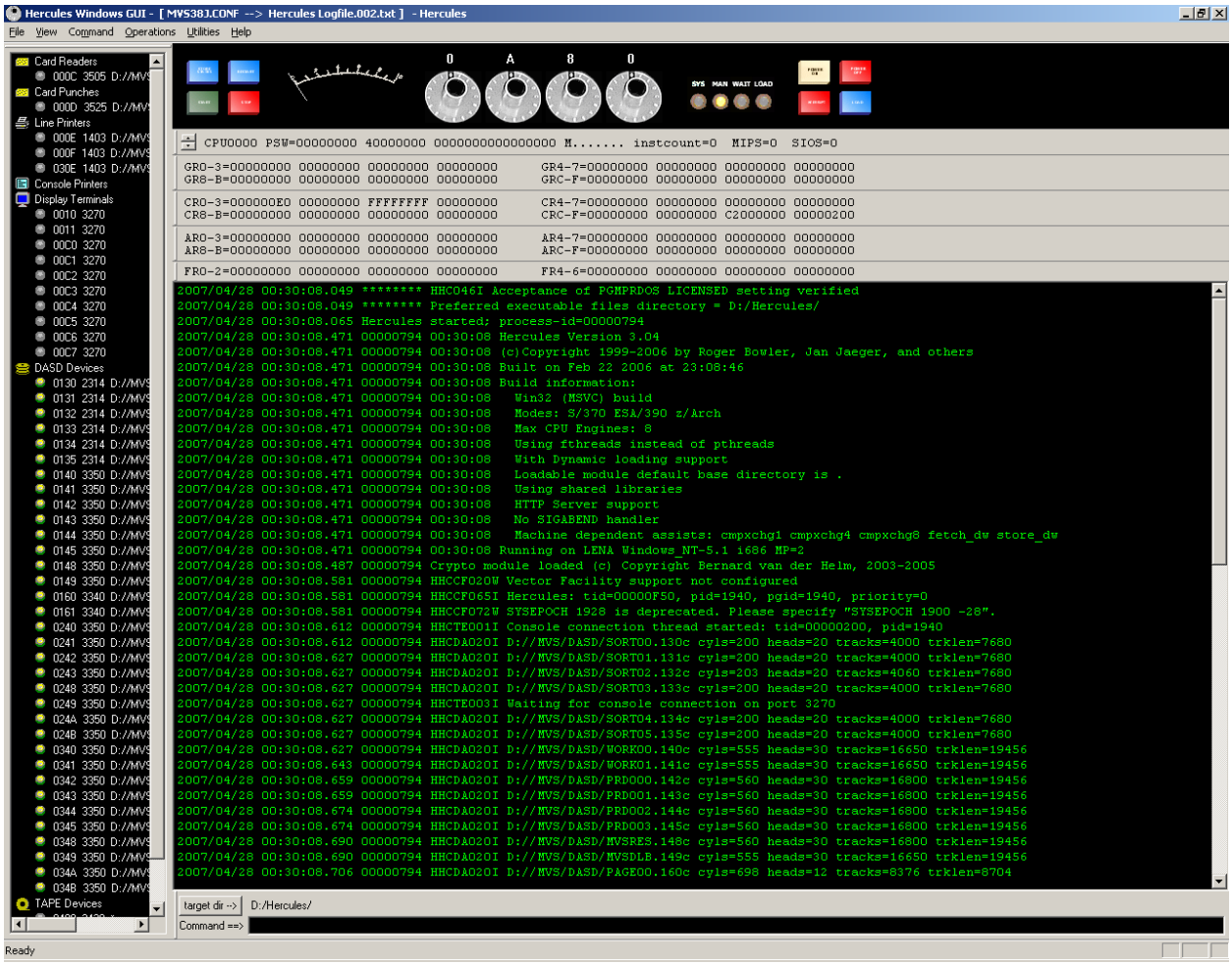


Figure 3: Hercules Windows GUI Main Panel

Using the GUI all of the Hercules DOS utilities are available, it is no longer necessary to know the exact syntax of each utility. Instead of having to issue cryptic command lines in native DOS such as

HETUPD -2 -b D:/MVS/TAPE/TLEV009.HET D:/MVS/TAPE/TLEV002.HET

a standard Windows dialog box can be used to call the utility. The following figure shows the pop-up window, used to provide information to one of the Hercules utility programs (HETUPD).

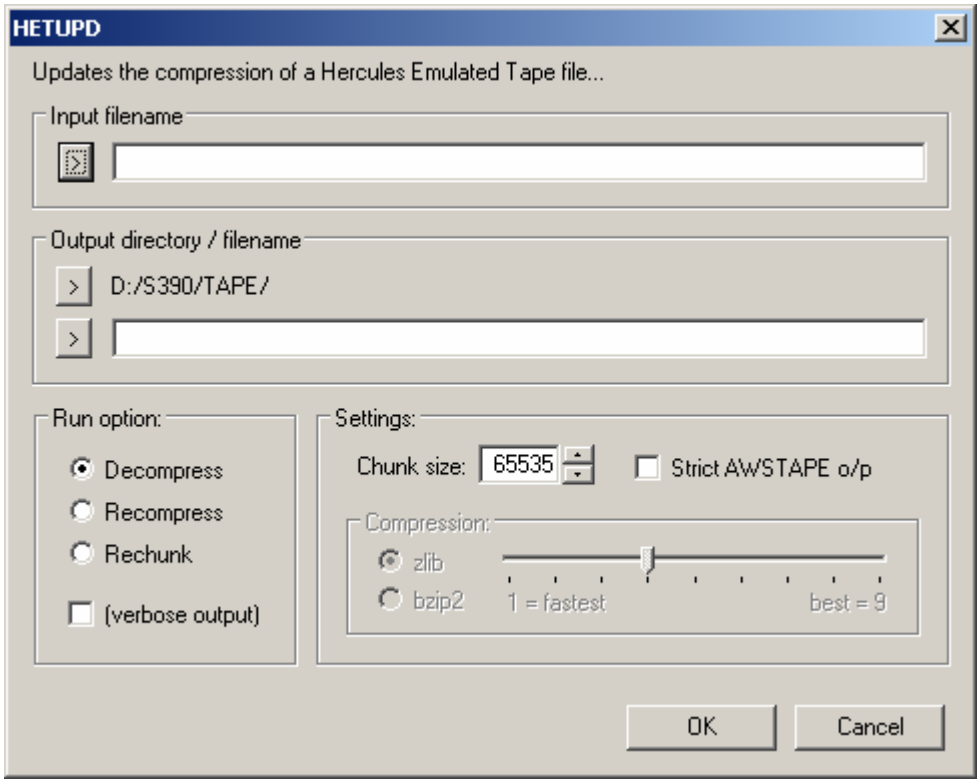


Figure 4: Hercules Utility Window

The WinGUI has been developed by David B. Trout (Fish). More detail about the functionality provided by the WinGUI can be found in the "User Reference Guide" and in Chapter 10 – Installing the Hercules WinGUI.

4.4.3 CTCI-W32

Since Hercules runs as a user process under the control of the host Windows system it does not normally have direct access to the driving system's network adapter. Until recently this presented a problem in establishing connectivity between the network and the TCP/IP stack of an operating system running under Hercules.

Since the development of device drivers by Fish, employing of the WinPcap's device driver, it is now possible to establish a virtual point-to-point link between the TCP/IP stack running under Hercules and Window's TCP/IP stack. This allows you to use Windows as a router to pass Ethernet frames between the Hercules TCP/IP stack and the rest of the network.

4.5 Additional required and optional Software

As well as the components described above other software, either required for practical use of Hercules (eg: tn3270 client) or that makes common tasks easier (eg: XMIT Manager, AWS Browse, ZZSA etc) may be used.

4.5.1 tn3270 Client (required)

For virtual 3270 consoles and 3270 terminals a tn3270 client software application is required. The tn3270 client can run on the same machine as Hercules or on any Unix or Windows box with a TCP/IP connection to the Hercules machine. The supported and recommended tn3270 client for Hercules under Windows is Vista tn3270.

Vista tn3270 can be obtained from www.tombrennansoftware.com. The license fee charged by the developer of the software, Tom Brennan, is very modest. A 30 day trial version can be downloaded from his web site.

Other tn3270 clients, such as QWS3270, IBM Personal Communications, Attachmate Extra, etc., should also work in most cases. Be aware that some tn3270 clients have a bug that makes them unusable as an MVS console.

Because the tn3270 client is an independent piece of software there are no version requirements. You can use any stable release of a tn3270 client although it is recommended to always run with a current release.

4.5.2 XMIT Manager (optional)

The XMIT Manager is a Windows based tool that allows for the manipulation of IBM mainframe created Xmit format files. With XMIT Manager you can open Xmit files and view or extract the data within them, whether binary or text, using a graphical interface. Xmit files containing partitioned or sequential datasets are supported.

4.5.3 AWS Browse (optional)

The AWS Browse Utility is used to view the contents of tapes from the Windows desktop without having to start a mainframe operating system and run tape reading utilities. There are currently two implementations of AWS browse.

The original one was created by Rob Storey. The second is an enhanced version written by Fish, which is faster and has more features.

5. Performance

As described previously the performance of the Hercules Emulator depends heavily on the underlying PC hardware. It is therefore not possible to give exact performance specifications of any particular operating system running under Hercules. Some practical values from user experiences across several machines presented here provide some guidelines though.

The performance of Hercules is measured in two values, MIPS and I/O rate. Both of these values are presented on the Hercules console and are refreshed every second independently of the PANRATE control statement. See Hercules User Reference for details on the PANRATE statement.

5.1 MIPS

MIPS is an abbreviation of “Million Instructions Per Second” and presents a measure of the number of instructions the CPU is executing in one second. MIPS is a measure of a computer's processor speed. However this measure is useful only among processors with the same instruction set as different instruction sets take different numbers of instructions to do the same job. Many of the reported MIPS values represent 'Peak' execution rates on artificial instruction sequences with few branches, whereas realistic workloads consist of a mix of instructions some of which take longer to execute than others.

The performance of the memory hierarchy greatly affects processor performance, an issue also not considered in simple MIPS comparisons. In an attempt to address these issues researchers have created standardized tests such as SpecInt to measure the real effective performance in commonly used applications. The use of raw MIPS as a measure of overall system performance has fallen into disuse. MIPS is sometimes pejoratively referred to as "Meaningless Indicator of Processor Speed" or "Meaningless Information Provided by Salespeople".

The Hercules console reports the MIPS rate for the emulated S/370, ESA/390 or z/Architecture instructions, not the underlying executed instructions of the hardware. As implied previously the MIPS rate can vary significantly depending on the executed instruction and whether the instructions can be processed entirely in cache, as can happen in a tight loop.

It is difficult to determine how the speed of the Hercules emulation corresponds to a real mainframe. This is due to difficulties in comparing real mainframe hardware to PCs (or servers) as well as the actual performance measurement itself. Hercules shows its processing speed in MIPS. Compared to the earlier IBM System/360 and System/370 hardware, it is safe to say that Hercules will outperform them when it is running on moderately powerful hardware, whereas newer IBM System/390 and z/Series hardware still is much faster than the emulation.

A fast dual core processor machine or a multi-processor system (equipped with enough RAM) is capable of reaching a sustained rate of 50 to 60 MIPS. When it is running in a tight loop (that means it is not doing real useful work) it can reach peaks around 100 to 130 MIPS. The speed depends greatly on the executed instructions. Instructions that are very expensive to emulate can still be - even on fast systems - below 1 MIPS, see the tables further below for details.

5.2 I/O Rate

The second value that gives us performance data is the I/O rate. This is the average number of SIOs (Start I/O per second) occurring to active devices, including DASD, TAPE, CTC (channel-to channel adaptor), local SNA and non-SNA devices etc.

This value also varies depending on the underlying hardware. A RAID system will easily outperform a non-RAID system. Many PC systems today can be ordered with a RAID-0 (Striping) or RAID-1 (Mirroring) adapter. While disk mirroring (RAID-1) gives fault tolerance it does not improve performance. However Data striping (RAID-0) spreads blocks of each file across multiple disks and can nearly double the performance of a single disk.

On a recent PC system as described above, incorporating two Serial-ATA (SATA) disk drives connected to a RAID adapter card with activated RAID-0 has shown peak I/O rates of more than 3250 SIOs per second. The sustained I/O rate delivered from such a system was more than 1500 SIOs per second.

5.3 Hercules Performance Measurements

The following sections show some performance index tables under various system configurations. Individual measured performance may vary from the figures shown as performance depends heavily on the hardware used and the Hercules release level.

All tests have been performed with the following measurement tools:

- IMON (Rate CPU Instruction Speed)
- CPU Instruction Timing Tool

Tests showing the performance of Hercules itself include measurements of the Hercules Emulator software and the difference between working with CKD or CCKD (compressed CKD) DASD emulation. Other tests, like the host CPU on which Hercules runs or the disk types of the host system, show the influence of the hardware used. The last test ("Hercules Overall Performance") represents the practical performance growth of the Hercules emulator as both hardware and software have improved.

5.3.1 Hercules Emulator Performance

The following diagram shows the software related performance index of the Hercules emulator only. The graphic shows the performance improvements that have been made over recent releases. The base version (index = 100%) for all measurements was Hercules V2.16.5.

All these tests were performed using the same hardware configuration. Results demonstrate the influence of the improvements in the performance of the Hercules Emulator software only.

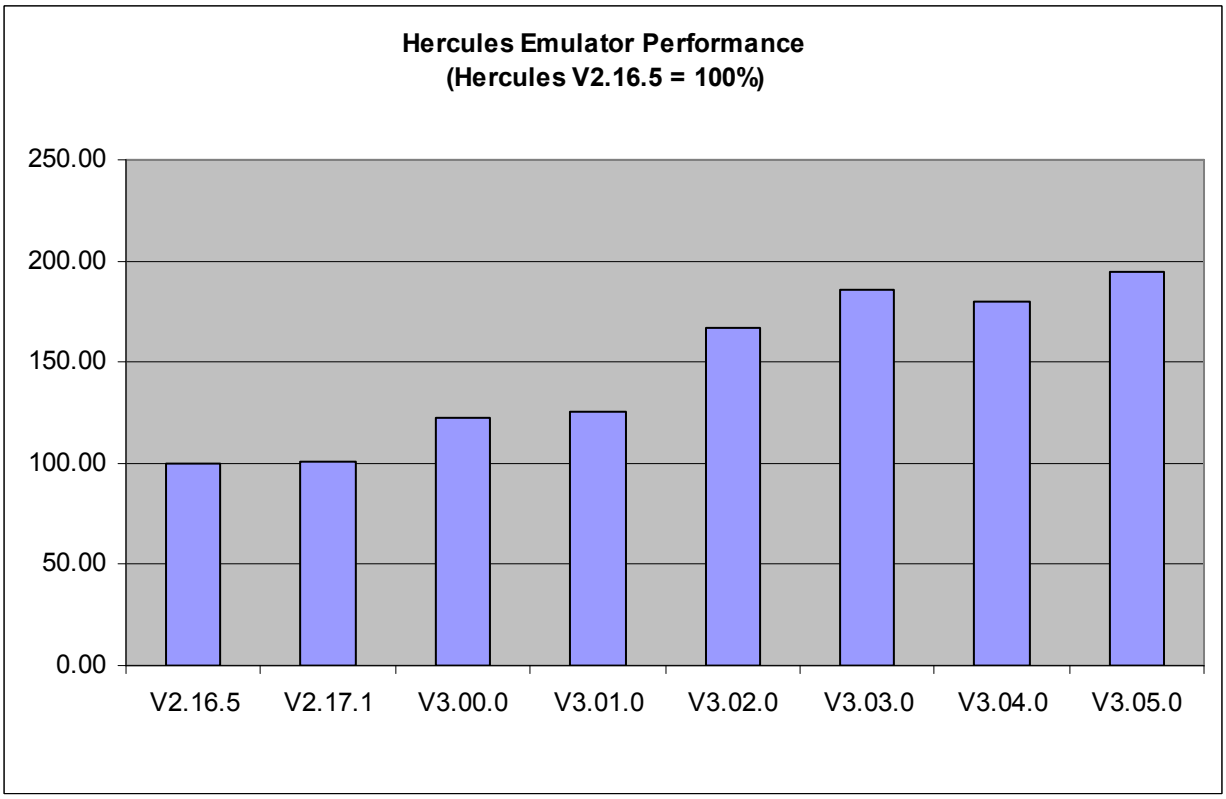


Figure 5: Hercules Emulator Performance

Although there have been massive performance improvements in the Hercules Emulator software in recent releases, the current rate of improvement cannot be expected to continue. On one hand, the maximum performance improvements available via software improvements will be reached, on the other hand the developers of Hercules have architectural frameworks that impose certain limitations, as explained below.

Firstly the Hercules software should be portable and is therefore written entirely in "C". The use of assembler to give additional performance is not exploited. Secondly the developers try to emulate real hardware as exactly as possible in accordance with IBM's "Principles of Operations" documentation. This leads to code that may be not optimal for the underlying hardware, but provides perfect emulation of real mainframe hardware, avoiding problems when running mainframe operating systems.

The performance of the Hercules Emulator software depends heavily on the set of executed (mainframe) instructions as detailed in below ("Emulated Instruction Performance").

5.3.2 Emulated CKD / CCKD DASD Performance

The type of the emulated DASD devices, either CKD or CCKD, has direct influence on performance. If using CKD devices there is more data to be read from the disk and transferred to memory, but the data is directly usable and no further processing needs to be done. Using CCKD (compressed CKD) devices the amount of data to be read from the disk and transferred to memory is greatly reduced but once in memory the data has to be uncompressed before it can be processed.

The performance characteristics of DASD types are irrelevant if host disk space limitations impose the choice of using compressed CKD only.

On a fast machine with fast disks CKD is the best choice. On a fast CPU with slow disks however, CCKD can give better overall performance. A fast processor can uncompress the data in less time than that

saved by transferring the data uncompressed. In the less common case of a slow CPU with fast disks CKD DASD can perform better as the uncompressed data is transferred faster than the processor could uncompress it. In the case of a slow machine with slow disks and possibly space constraints it is recommended to work with CCKD DASD.

Another point to consider is reliability. Although Hercules itself is very stable, occasional machine crashes such as the Windows "Blue Screen of Death" can occur. The CKD DASD emulation in these cases is very stable; it is usually possible to just restart the machine with no problems. However when working with CCKD DASD emulation, after a crash the CCKD routines have to perform a recovery during the restart of Hercules. This recovery takes from seconds to several minutes depending on what happened and the number of defined DASD devices. Although this recovery is normally successful cases have been reported where the recovery failed. In this case the only option is to restore DASD images from previously saved backups.

The following figure shows a performance comparison of compressed DASD (CCKD) versus CKD devices. The base (index=100%) is the IPL time with CKD devices. The IPL time with CCKD devices is only slightly higher.

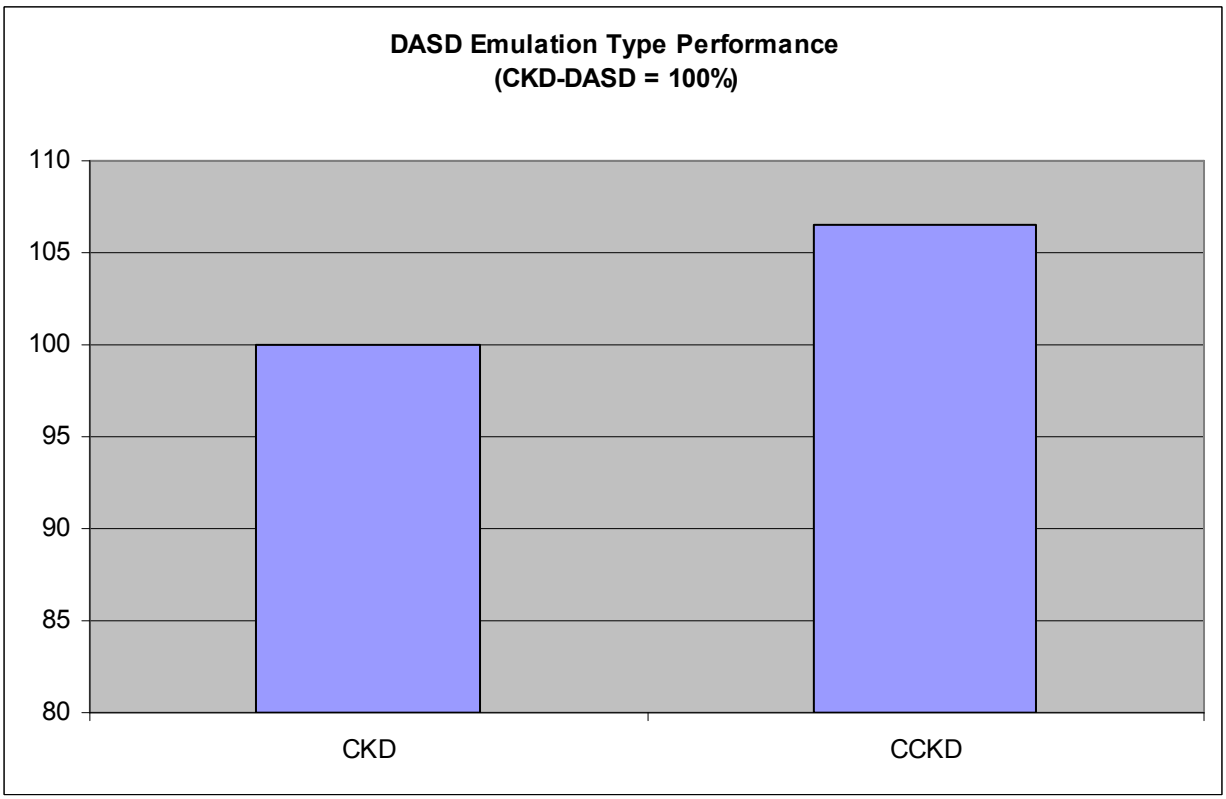


Figure 6: DASD Emulation Type Performance

Both tests were performed on the same machine (Intel P4, 3.2 GHz HT with a RAID-0 hard disk configuration). The measured performance degradation using CCKD emulation over CKD is only approximately 6.5 percent.

5.3.3 Host CPU Performance

The following diagram shows the CPU related performance results for Hercules. The base CPU at index = 100% is an Intel Pentium 3 with 500 MHz clock speed.

All these tests were performed with the same Hercules release and therefore show only the influence on performance of the real CPU and I/O devices.

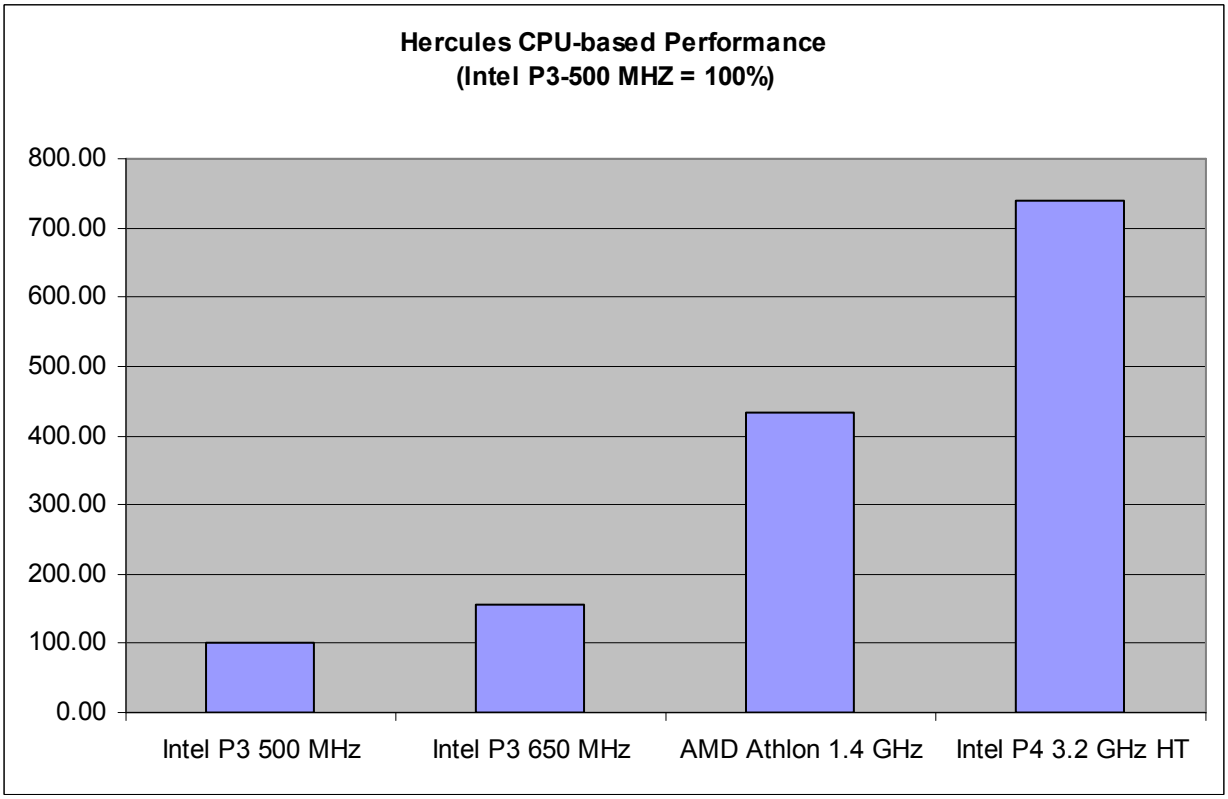


Figure 7: Hercules CPU-based Performance

The figures shown above are influenced by the I/O configuration used. As mainframe operating systems running under Hercules are often heavily I/O based, bare CPU speed is not the only relevant performance factor. An additional performance boost is achieved using RAID-0 striping and / or using several physical disks in the host system under which the emulator is running, as discussed in the following section.

5.3.4 Host Disk Performance

This next test shows the influence of the types of disk used on the host system where Hercules is running. Both measurements were made using the same hardware (Intel P4, 3.2 GHz), the only change being the type of disk employed.

The base for the comparison (index=100%) was an IBM Serial-ATA (SATA) disk, which reached a transfer rate of about 40 MB/s. For comparison one test used an ATA disk connected via USB-2 to the host system, reaching a maximum transfer rate of about 30 MB per second. The next test used two SATA disks connected to a RAID-0 adapter. This scenario reached a transfer rate of up to 80 MB per second.

The last two tests have been run with recent Western Digital Raptor SATA disk. They reached about 80 MB/s (non RAID) and 160 MB/s (RAID-0).

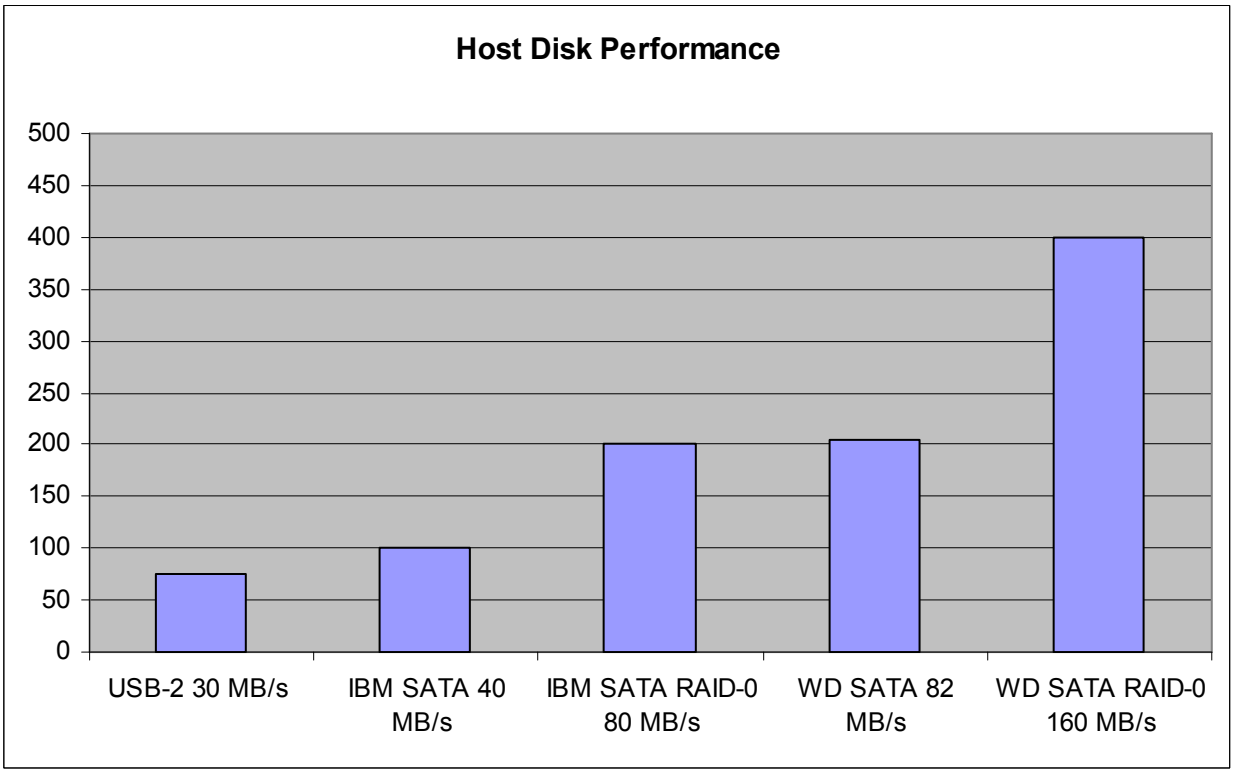


Figure 8: Host Disk Performance (Transfer Rates)

While these measurements only show the “raw speed” of the disks without direct relationship to Hercules, the next diagram shows the impact on the IPL time. The first test was made with a RAID-0 disk configuration which reached a transfer rate of 80 MB/s (100%). The second IPL was done with a disk with a transfer rate of only 30 MB/s. The IPL time with the slower disk is about 27% longer.

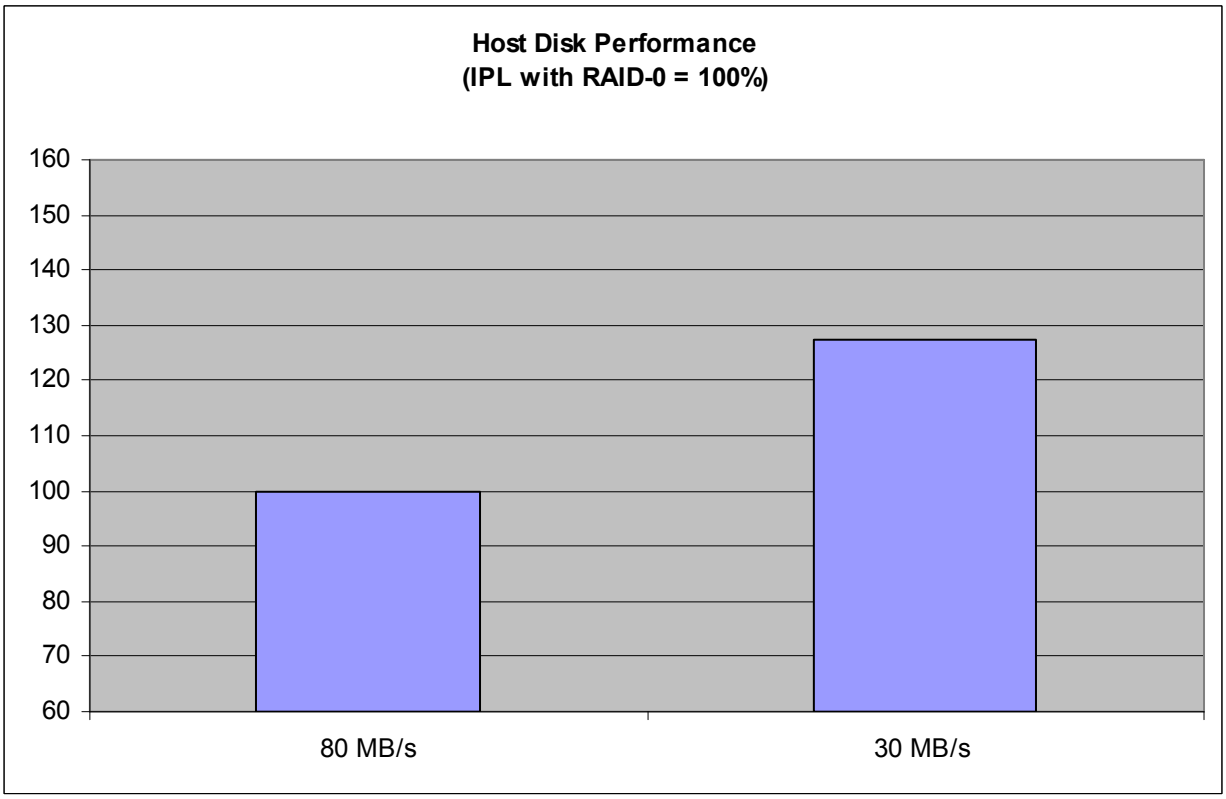


Figure 9: Host Disk Performance (IPL Comparison)

These tests demonstrate that fast disks and RAID-0 configurations provide a performance boost from which Hercules will benefit.

5.3.5 Hercules Overall Performance

The last diagram shows the real life performance improvements seen over time by Hercules users as both hardware and the Hercules software improve. Starting with a Pentium 3 Laptop at 500 MHz clock speed and Hercules release V2.16.5, through a series of intermediate steps both hardware and Hercules have been upgraded to a current Desktop Pentium 4 with 3.2 GHz (HT) and RAID 0 disk subsystem using Hercules release V3.05.0.

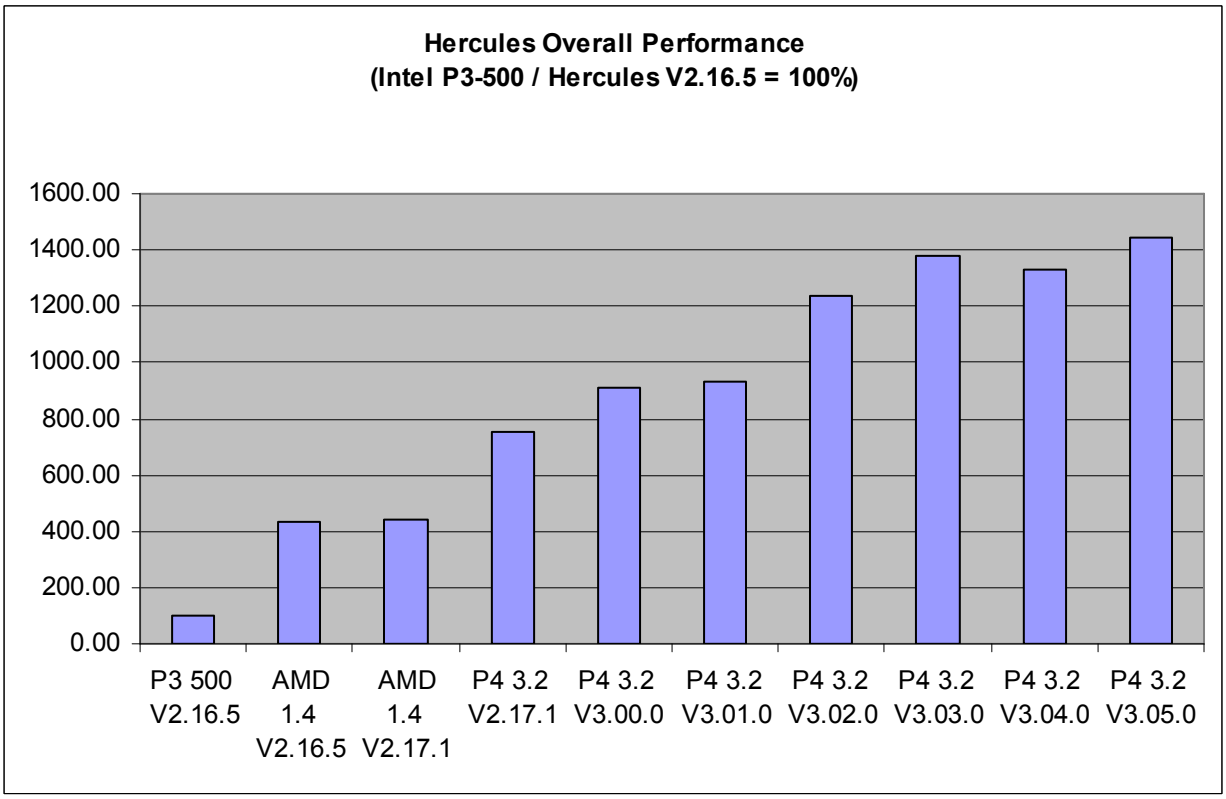


Figure 10: Hercules Overall Performance

Performance improvements over a five year period are impressive. As can be seen, significant improvement has occurred in purely software performance over the last four Hercules releases. However it is likely that most future performance improvements will be realized via improved hardware only.

5.3.6 Emulated Instruction Performance

As previously stated, Hercules Emulator performance depends heavily on the mainframe instructions being executed. While some S/3xx instructions are relatively easy to implement and require only a few "C" instructions in the emulator, others are quite complicated and require a lot of instructions to emulate the real hardware. Due to this fact the performance of each individual S/3xx instruction can vary greatly. There is a variance of more than 450 times between the fastest and slowest emulated instructions.

It is known though, that in mainframe operating systems, a small subset of the available instructions are employed to do most processing. In Hercules, if most of these instructions are emulated slowly, the overall performance of the emulator will be poor. Alternatively, if the most commonly used instructions are emulated quickly the performance of the emulator will benefit greatly.

An analysis of mainframe operating systems has shown that only 5 different instructions are responsible for up to 50% of the executed code. Expanding this analysis to 10-15 instructions will cover up to 75% of executed code and 15-20 instructions will cover more than 80% of all mainframe code.

The following table lists the performance ratios of individual instructions and tests under the Hercules Emulator. The first scenario is based on Hercules Release 3.05.0 on an Intel Pentium 3 running at 500 MHz with 128 MB RAM and Windows XP. Scenario 2 was run using an Intel Pentium 4 with 3.2 GHz (HT) with 2GB RAM on Windows XP and the same Hercules release. The last scenario used also the same Hercules Release, but run on 2 dual-core AMD Opterons (4 processors with 2.4 GHz) and 8 GB RAM on Windows XP-x64.

Executed Instruction	Scenario 1 Intel P3 500 MHz	Scenario 2 Intel P4 3.2GHz HT	Scenario 3 2x AMD Opteron 2.4 GHz
BCT Rx,LOOP	14.37 MIPS	99.55 MIPS	84.21 MIPS
BCTR RX,RLOOP	15.67 MIPS	109.10 MIPS	94.02 MIPS
NOP R0	23.09 MIPS	154.84 MIPS	145.58 MIPS
LR R1,R0	10.12 MIPS	66.33 MIPS	58.21 MIPS
LTR R1,R0	8.44 MIPS	64.76 MIPS	58.22 MIPS
L R1,0	1.82 MIPS	10.09 MIPS	12.43 MIPS
L R1,DATA	5.14 MIPS	39.00 MIPS	33.85 MIPS
L R1,DATA+1	4.75 MIPS	37.95 MIPS	33.47 MIPS
LH R1,DATA	4.47 MIPS	37.74 MIPS	33.97 MIPS
ICM R1,15,DATA	4.72 MIPS	35.50 MIPS	33.93 MIPS
ICM R1,1,DATA	2.70 MIPS	20.28 MIPS	18.60 MIPS
IC R1,DATA	5.02 MIPS	37.59 MIPS	33.83 MIPS
LD F0,DATA	4.64 MIPS	13.95 MIPS	25.20 MIPS
LM 8,6,SAVEREGS+(8*4)	2.35 MIPS	18.78 MIPS	18.07 MIPS
STM 1,14,DATA	2.44 MIPS	24.01 MIPS	19.29 MIPS
ST R1,DATA	4.83 MIPS	37.77 MIPS	32.94 MIPS
STH R1,DATA	3.88 MIPS	36.84 MIPS	33.95 MIPS
STCM R1,15,DATA	4.67 MIPS	35.87 MIPS	27.91 MIPS
STCM R1,1,DATA	2.64 MIPS	22.51 MIPS	19.33 MIPS
STCM R1,8,DATA	2.70 MIPS	23.05 MIPS	19.72 MIPS
STC R1,DATA	5.21 MIPS	39.97 MIPS	33.85 MIPS
MVI DATA,CHAR	5.06 MIPS	39.77 MIPS	35.57 MIPS
MVC DATA(8),DATA	2.41 MIPS	19.72 MIPS	17.74 MIPS
MVC DATA(32),DATA	1.79 MIPS	13.12 MIPS	12.24 MIPS
MVC DATA(32),DATAB	1.91 MIPS	2.25 MIPS	6.74 MIPS

MVC	DATA(32),0	1.24 MIPS	1.94 MIPS	5.13 MIPS
MVC	DATA(256),DATA	0.42 MIPS	3.56 MIPS	3.71 MIPS
MVC	DATA(256),DATAB	0.25 MIPS	0.31 MIPS	1.18 MIPS
XC	DATA(4),DATA	2.78 MIPS	17.58 MIPS	19.48 MIPS
XC	DATA(4),DATAB	2.81 MIPS	22.61 MIPS	21.99 MIPS
CLI	DATA,CHAR	5.14 MIPS	39.73 MIPS	32.51 MIPS
CLC	DATA(8),DATA	2.53 MIPS	26.42 MIPS	23.26 MIPS
CLC	DATA(32),DATA	1.56 MIPS	22.15 MIPS	17.92 MIPS
CLC	DATA(32),DATAB	1.49 MIPS	22.53 MIPS	17.94 MIPS
CLC	DATA(32),0	1.30 MIPS	10.01 MIPS	10.57 MIPS
CLC	DATA(256),DATA	0.30 MIPS	8.84 MIPS	7.93 MIPS
CLC	DATA(256),DATAB	0.30 MIPS	8.88 MIPS	7.92 MIPS
AR	R1,R0	6.81 MIPS	49.39 MIPS	37.61 MIPS
ALR	R1,R0	6.58 MIPS	59.98 MIPS	56.77 MIPS
A	R1,F1	4.75 MIPS	36.09 MIPS	33.01 MIPS
AH	R1,H1	4.21 MIPS	35.57 MIPS	30.77 MIPS
LA	R1,1(,R1)	7.31 MIPS	55.35 MIPS	45.27 MIPS
LA	R1,1(R1)	7.28 MIPS	55.46 MIPS	49.08 MIPS
AP	PL4,PL4	0.24 MIPS	1.67 MIPS	1.70 MIPS
AP	PL4,PL4B	0.24 MIPS	1.87 MIPS	1.81 MIPS
AP	PL16,PL16	0.19 MIPS	1.46 MIPS	1.53 MIPS
AP	PL16,PL16B	0.22 MIPS	1.64 MIPS	1.70 MIPS
CVD	R1,DATA	0.39 MIPS	2.82 MIPS	3.08 MIPS
CVD	R15,DATAB	0.94 MIPS	6.60 MIPS	7.11 MIPS
AER	F0,F4	4.40 MIPS	32.26 MIPS	30.21 MIPS
ADR	F0,F4	3.53 MIPS	25.78 MIPS	25.84 MIPS
AXR	F0,F4	2.56 MIPS	16.69 MIPS	17.44 MIPS
Move	256 Bytes by MVC	0.28 MIPS	0.31 MIPS	1.18 MIPS

Move 256 Bytes by MVCL	0.81 MIPS	3.25 MIPS	4.86 MIPS
Clear 256 Bytes by MVC	0.38 MIPS	0.93 MIPS	1.34 MIPS
Clear 256 Bytes by MVCL	0.91 MIPS	3.63 MIPS	5.46 MIPS

Table 2: Emulated Instruction Performance

The next table shows some performance data doing real work. The scenarios are the same as in the instruction tests before. The tests are the performance measurements of IMON (Interactive Monitor IMON/370 and IMON for OS/390 and z/OS, see http://www.prycroft6.com.au).

The second group of tests are real batch jobs (heavy sorts and some large assemblies) and a tight loop. These tests show the influence on raw processor speed (only one batch job running) as well as the influence of having a multi-processor system and running several batch jobs in parallel.

The last test shows the time used for an IPL of a mainframe operating system from the actual IPL command until TSO logon is possible. The IPL only contains the mainframe operating system itself, without any optional subsystems.

Executed Instruction	Scenario 1 Intel P3 500 MHz 1 Processor	Scenario 2 Intel P4 3.2GHz 2 Process. (HT)	Scenario 3 2 x AMD Athlon 2.4 GHz 4 Processors
IMON RR (Fast)	12.61 MIPS	41.98 MIPS	68.28 MIPS
IMON RR (Slow)	6.06 MIPS	36.80 MIPS	48.39 MIPS
IMON RX (Fast)	6.01 MIPS	37.17 MIPS	40.56 MIPS
IMON RX (Slow)	1.81 MIPS	11.20 MIPS	13.22 MIPS
IMON SS (Fast)	1.07 MIPS	6.65 MIPS	7.09 MIPS
IMON SS (Slow)	0.13 MIPS	0.90 MIPS	0.95 MIPS
IMON FP (Extended)	3.93 MIPS	22.81 MIPS	20.80 MIPS
IMON FP (Double)	0.71 MIPS	3.35 MIPS	4.94 MIPS
Assembly (1 Job)	4.62 MIPS	26.76 MIPS	34.52 MIPS
Assembly (2 Jobs)	4.42 MIPS	20.85 MIPS	61.53 MIPS
Assembly (3 Jobs)	4.09 MIPS	19.46 MIPS	81.75 MIPS
Assembly (4 Jobs)	(no response)	18.95 MIPS	104.07 MIPS

Sort (1 Job)	3.46 MIPS	21.82 MIPS	26.63 MIPS
Sort (2 Jobs)	3.36 MIPS	18.75 MIPS	49.89 MIPS
Sort (3 Jobs)	3.25 MIPS	18.64 MIPS	58.65 MIPS
Sort (4 Jobs)	(no response)	18.53 MIPS	71.10 MIPS
Loop (1 Job)	13.50 MIPS	116.04 MIPS	105.54 MIPS
Loop (2 Jobs)	13.23 MIPS	61.30 MIPS	209.20 MIPS
Loop (3 Jobs)	12.97 MIPS	60.78 MIPS	311.10 MIPS
Loop (4 Jobs)	(no response)	59.75 MIPS	414.06 MIPS
IPL time	82:51 Minutes	03:10 Minutes	01:03 Minutes

Table 3: Various Performance Tests

These tests show that, although the raw instruction speed of the four processor machine is somewhat slower as shown in table 2 above, the system greatly benefits from the four processors when doing real work, because it can handle more tasks in parallel.

6. Component Compatibility Tables

The various components that together make the full Hercules environment have some inter-dependencies on each other. It is recommended that these components be at a certain release level for every version of Hercules. The following tables list the combinations of components for several releases that have been successfully evaluated as working together.

If a combination is not listed in these tables this does not necessarily mean that it will not function. However you must asses this yourself and at your own risk. Although the listed combinations shown below have been thoroughly tested and proven to provide a stable environment, there is no guarantee that the components work in every environment and in any case.

Components that have been updated during the lifetime of a specific Hercules release are shown with their highest release level in each table. In general it is recommended to work always with the most current stable release of each component. Beta releases are not shown here. The date(s) in parenthesis show the release dates of the respective components. If there was no release of Hercules but updated components, then the word “Components” can be found before the release date.

In the case of Cygwin the release shown here is the one that Hercules was built with, rather than the highest. As Cygwin is a runtime environment, its release level must be the same or higher than the version Hercules was built with. Therefore the Cygwin release level listed for any level of Hercules is the minimum level needed for that Hercules release. Any higher level of Cygwin should also run but experience has shown that this is not always the case.

Please note that obsolete components, e.g. Cygwin, or components that were subsequently included in other packages, e.g. FishPack, TunTap32 and tt32info are mentioned in the table immediately after the release where the change occurred and are then removed from the tables.

6.1 Hercules V 3.05.0 (Release date: June 23, 2007)

Component	**Release**
Hercules Emulator	V 3.05.0
WinPcap	V 4.0
HercWinGUI	V 1.11.1.5265
CTCI-W32	V 3.2.1.160
FishLib	V 2.7.1.564
AWS Browse	V 1.5.1.1805

Table 4: Hercules Release V 3.05.0 Component Compatibility Table

6.2 Hercules V 3.04.0 (Components, Release date: March 01, 2007)

Component	Release
Hercules Emulator	V 3.04.0
WinPcap	V 4.0
HercWinGUI	V 1.11.1.5265
CTCI-W32	V 3.2.1.160
FishLib	V2.7.1.564
AWS Browse	V 1.5.1.1805

Table 5: Hercules Release V 3.04.0 Component Compatibility Table (changed components)

6.3 Hercules V 3.04.0 (Components, Release date: October 04, 2006)

Component	Release
Hercules Emulator	V 3.04.0
WinPcap	V 3.1
HercWinGUI	V 1.10.1.4909
CTCI-W32	V 3.1.7
FishLib	V2.2.4.668
AWS Browse	V 1.4.0.1483

Table 6: Hercules Release V 3.04.0 Component Compatibility Table (changed components)

6.4 Hercules V 3.04.0 (Components, Release date: August 16, 2006)

Component	Release
Hercules Emulator	V 3.04.0
WinPcap	V 3.1
HercWinGUI	V 1.10.0.4890
CTCI-W32	V 3.1.6
FishLib	V2.2.4.668
AWS Browse	V 1.4.0.1483

Table 7: Hercules Release V 3.04.0 Component Compatibility Table (changed components)

6.5 Hercules V 3.04.0 (Components, Release date: May 06, 2006)

Component	Release
Hercules Emulator	V 3.04.0
WinPcap	V 3.1
HercWinGUI	V 1.9.5.4734
CTCI-W32	V 3.1.2
FishLib	V2.2.1.605
FishPack	**Now included in new CTCI-W32**
TunTap32	**Now included in new CTCI-W32**
tt32info	**Now included in new CTCI-W32**
AWS Browse	V 1.3.0.1445

Table 8: Hercules Release V 3.04.0 Component Compatibility Table (with new CTCI-W32)

6.6 Hercules V 3.04.0 (Release date: February 24, 2006)

Component	Release
Hercules Emulator	V 3.04.0
WinPcap	V 3.0
HercWinGUI	V 1.9.5.4734
FishPack	V 1.3.0.323
TunTap32	V 2.1.0.404
tt32info	V 1.0.2.133
AWS Browse	V 1.2.0.1278

Table 9: Hercules Release V 3.04.0 Component Compatibility Table

6.7 Hercules V 3.03.1 (Release date: December 31, 2005)

Component	Release
Hercules Emulator	V 3.03.1
WinPcap	V 3.0
HercWinGUI	V 1.9.5.4734
FishPack	V 1.3.0.323
TunTap32	V 2.1.0.404
tt32info	V 1.0.2.133
AWS Browse	V 1.2.0.1278

Table 10: Hercules Release V 3.03.1 Component Compatibility Table

6.8 Hercules V 3.03.0 (Release date: December 20, 2005)

Component	Release
Hercules Emulator	V 3.03
Cygwin	**No longer needed!**
WinPcap	V 3.0
HercWinGUI	V 1.9.5.4734
FishPack	V 1.3.0.323
TunTap32	V 2.1.0.404
tt32info	V 1.0.2.133
AWS Browse	V 1.2.0.1278

Table 11: Hercules Release V 3.03.0 Component Compatibility Table

6.9 Hercules V 3.02.0 (Release date: December 11, 2004)

Component	Release
Hercules Emulator	V 3.02
Cygwin	V 1.5.12
WinPcap	V 3.0
HercWinGUI	V 1.8.8.4207
FishPack	V 1.3.0.323
TunTap32	V 2.0.3.379
tt32info	V 1.0.2.133
AWS Browse	V 1.2.0.1278

Table 12: Hercules Release V 3.02.0 Component Compatibility Table

6.10 Hercules V 3.01.0 (Release date: November 30, 2003)

Component	Release
Hercules Emulator	V 3.01
Cygwin	V 1.5.5
WinPcap	V 3.0
HercWinGUI	V 1.6.8.3981
FishPack	V 1.3.0.323
TunTap32	V 2.0.3.379
tt32info	V 1.0.2.133

Table 13: Hercules Release V 3.01.0 Component Compatibility Table

6.11 Hercules V 3.00.0 (Release date: October 2, 2003)

Component	Release
Hercules Emulator	V 3.00
Cygwin	V 1.5.5
WinPcap	V 3.0
HercWinGUI	V 1.6.8.3910
FishPack	V 1.3.0.323
TunTap32	V 2.0.3.379
tt32info	V 1.0.2.133

Table 14: Hercules Release V 3.00.0 Component Compatibility Table

6.12 Hercules V 2.17.1 (Release date February 12, 2003)

Component	Release
Hercules Emulator	V 2.17.1
Cygwin	V 1.3.20
WinPcap	V 3.0
HercWinGUI	V 1.6.0.3438
FishPack	V 1.3.0.323
TunTap32	V 2.0.3.379
tt32info	V 1.0.2.133

Table 15: Hercules Release V 2.17.1 Component Compatibility Table

6.13 Hercules V 2.17.0 (Release date February 1, 2003)

Component	Release
Hercules Emulator	V 2.17.0
Cygwin	V 1.3.19
WinPcap	V 2.3
HercWinGUI	V 1.5.0.3290
FishPack	V 1.1.0.296
TunTap32	V 2.0.0.367
tt32info	V 1.0.2.133

Table 16: Hercules Release V 2.17.0 Component Compatibility Table

6.14 Hercules V 2.16.5 (Release date July 8, 2002)

Component	Release
Hercules Emulator	V 2.16.5
Cygwin	V 1.3.10
WinPcap	V 2.3
HercWinGUI	V 1.5.0.3290
FishPack	V 1.1.0.296
TunTap32	V 2.0.0.367
tt32info	V 1.0.2.133

Table 17: Hercules Release V 2.16.5 Component Compatibility Table

6.15 Hercules V 2.16.4 (Release date July 3, 2002)

Component	Release
Hercules Emulator	V 2.16.4
Cygwin	V 1.3.10
WinPcap	V 2.3
HercWinGUI	V 1.5.0.3290
FishPack	V 1.1.0.296
TunTap32	V 2.0.0.367
tt32info	V 1.0.2.133

Table 18: Hercules Release V 2.16.4 Component Compatibility Table

6.16 Hercules V 2.16.3 (Release date July 2, 2002)

Component	Release
Hercules Emulator	V 2.16.3
Cygwin	V 1.3.10
WinPcap	V 2.3
HercWinGUI	V 1.5.0.3290
FishPack	V 1.1.0.296
TunTap32	V 2.0.0.367
tt32info	V 1.0.2.133

Table 19: Hercules Release V 2.16.3 Component Compatibility Table

6.17 Hercules V 2.16.2 (Release date May 20, 2002)

Component	Release
Hercules Emulator	V 2.16.2
Cygwin	V 1.3.10
WinPcap	V 2.3
HercWinGUI	V 1.5.0.3290
FishPack	V 1.1.0.296
TunTap32	V 2.0.0.367
tt32info	V 1.0.2.133

Table 20: Hercules Release V 2.16.2 Component Compatibility Table

6.18 Hercules V 2.16.1 (Release date May 4, 2002)

Component	Release
Hercules Emulator	V 2.16.1
Cygwin	V 1.3.10
WinPcap	V 2.3
HercWinGUI	V 1.5.0.3290
FishPack	V 1.1.0.296
TunTap32	V 2.0.0.367
tt32info	V 1.0.2.133

Table 21: Hercules Release V 2.16.1 Component Compatibility Table

6.19 Hercules V 2.16.0 (Release date April 20, 2002)

Component	Release
Hercules Emulator	V 2.16.0
Cygwin	V 1.3.10
WinPcap	V 2.3
HercWinGUI	V 1.5.0.3290
FishPack	V 1.1.0.296
TunTap32	V 2.0.0.367
tt32info	V 1.0.2.133

Table 22: Hercules Release V 2.16.0 Component Compatibility Table

7. Installation WinPcap

Figure 11: WinPcap Logo

7.1 WinPcap Packet Capture Driver

The first component you are required to install after the operating system is the WinPcap Packet Capture Driver. This can be downloaded from the website of the Politecnico di Torino (http://www.WinPcap.org) where you will also find comprehensive documentation. Save the executable in a directory of your choice.

Be sure to only use the latest version of WinPcap that is supported by CTCI-W32 supplied with the release of Hercules you are using. The most recent release of WinPcap may not be compatible with your version of Hercules. See chapter 6 (Component Compatibility Tables) for details of the supported release.

7.2 Installation Steps (Windows Setup)

The installation program for WinPcap creates and then initiates a kernel driver service. Please note that you need to have administrator privileges in order to create and to start the Netgroup Packet Filter (NPF) kernel driver service. You must be logged onto your Windows system as an "administrator" when you run the TESTAPP program for the first time and create the NPF service entry, or whenever you run a program that starts the NPF service. Once the NPF service has been created and / or started however, any non administrator user may use the service.

As with every installation under the Windows operating system, you should stop all running programs prior to starting the installation process. To start the installation wizard just double-click on the executable WINPCAP_v_r.EXE (where “v” is the version and “r” is the release) in the directory where you previously have saved the file.

The installer takes a moment to load into memory.

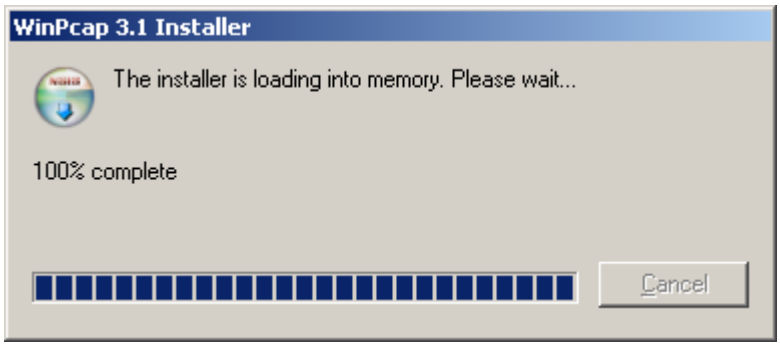


Figure 12: WinPcap Setup – Loading Installer

The welcome screen of the installation wizard appears.

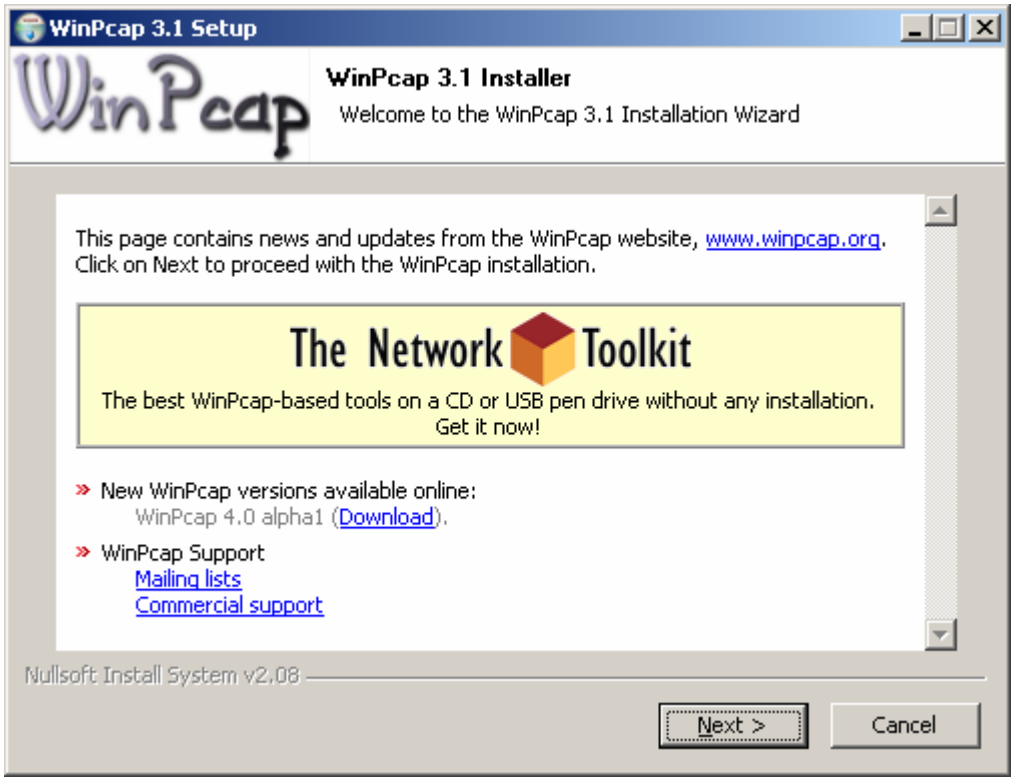


Figure 13: WinPcap Setup - Welcome Screen

This screen gives you the usual information presented when starting installation programs under Windows. Click on “Next >” to continue with the installation.

On the following screen you are presented with the license agreement. Read the license agreement carefully. You have to accept the license agreement in order to continue with the installation process.

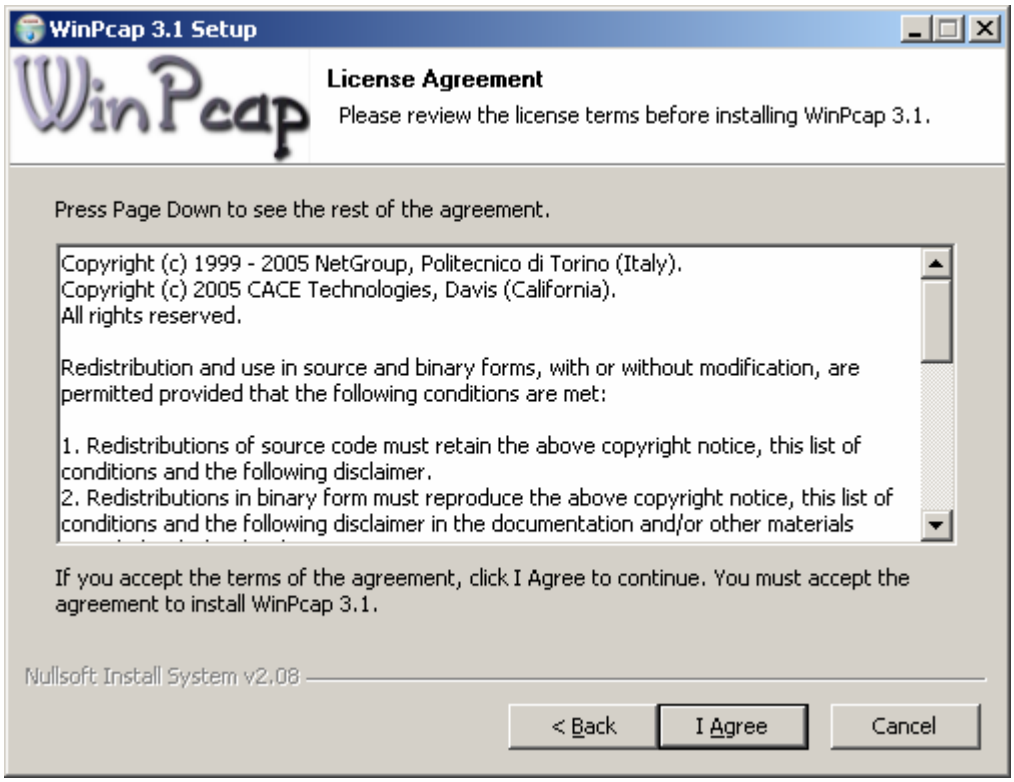


Figure 14: WinPcap Setup - License Agreement

After reading the license agreement, accept it by clicking on the “I Agree" button to continue the installation process.

The installation process now automatically begins. You do not have to select a target directory and there are no other decisions to make. A progress bar in a new window will appear. On a fast machine it is possible that the progress window just flickers for a short time, or that you will not see the window at all as the installation process is very fast.

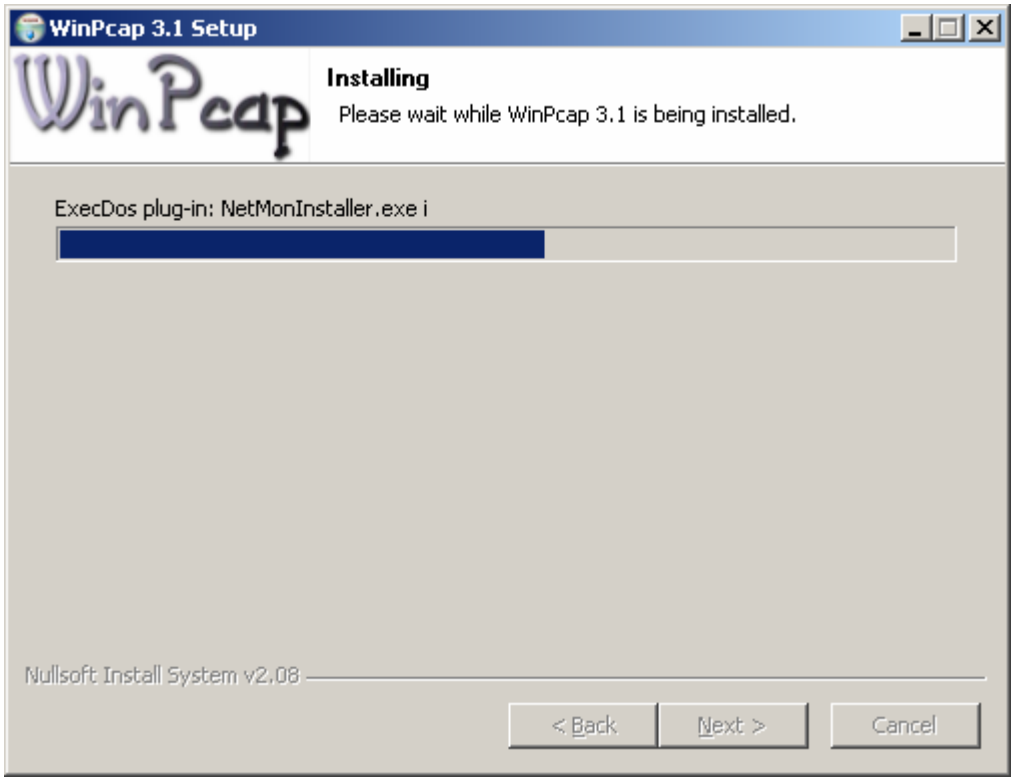


Figure 15: WinPcap - Setup Status

As soon as all files are copied to their default destination and the installation process is complete the final screen shown below is displayed.

Figure 16: WinPcap Setup - Installation Complete

If an older version of WinPcap was installed on the machine it is strongly recommended that you reboot the system. Click on "Finish" to complete the installation process, then continue with the customization steps.

7.2.1 Customization Steps

Unfortunately the WinPcap installation routine sometimes fails to successfully create the packet driver service (NPF) required by any program using WinPcap. In early CTCI-W32 implementations the FishPack DLL simply tried to start the driver service that it needed assuming that the WinPcap installer had been successful in creating the NPF service. In the current release of CTCI-W32, FishPack now makes a dummy call to the WinPcap DLLs “PacketOpenAdapter” function at startup to force WinPcap to finish its installation process and create the needed kernel driver service. This can be done by either starting Hercules using CTCI-W32 or can be forced with the stand-alone CTCI-W32 utility “tt32info”.

Therefore the customization steps described below should not be required, however it is a good idea to verify that the WinPcap installation did complete successfully. To see if the kernel driver service is installed properly, perform the following steps.

From the Windows desktop click on “START” then select “RUN…”, the run window will appear. Type the name of the Windows System Information program (“msinfo32”) and click on “OK”.

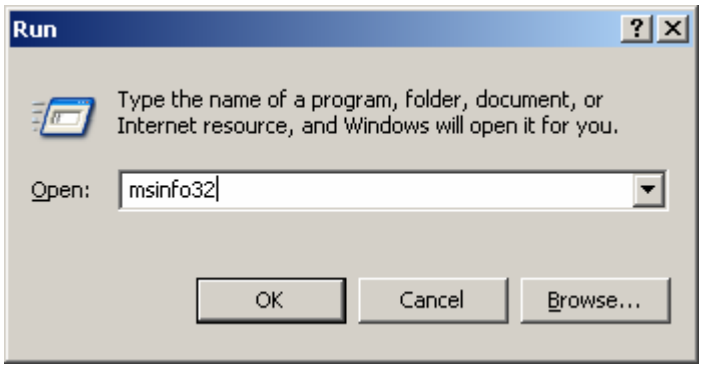


Figure 17: Run Program msinfo32

The System Information program starts and presents the main window shown below.

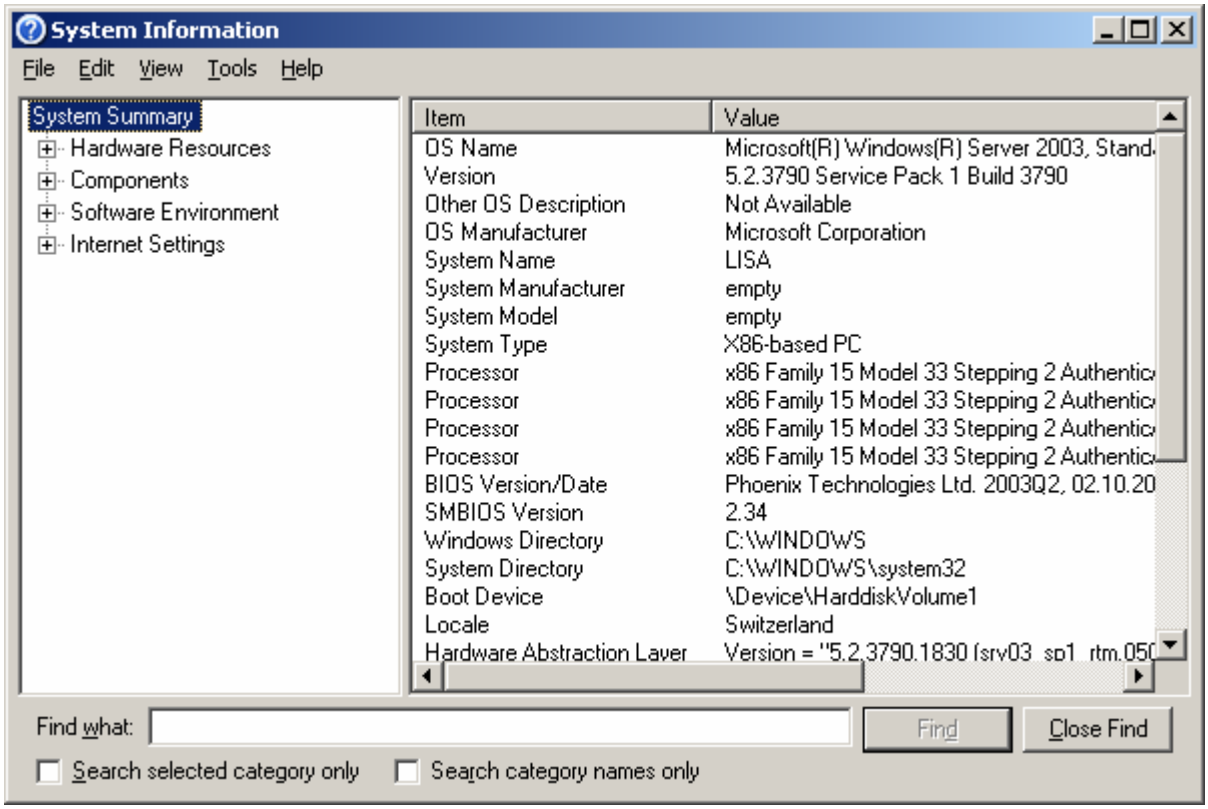


Figure 18: System Information Main Window

In this window select “Software Environment” and then “System Drivers”. After a few moments a list of all installed system drivers is presented.

Scroll through this list looking for the WinPcap kernel driver service which will be called “npf – NetGroup Packet Filter Driver”. If this driver is found in the list then WinPcap has successfully installed its service. If this entry is missing from the list it can be created using the CTCI-W32 utility “tt32info” (see chapter 11 for details).

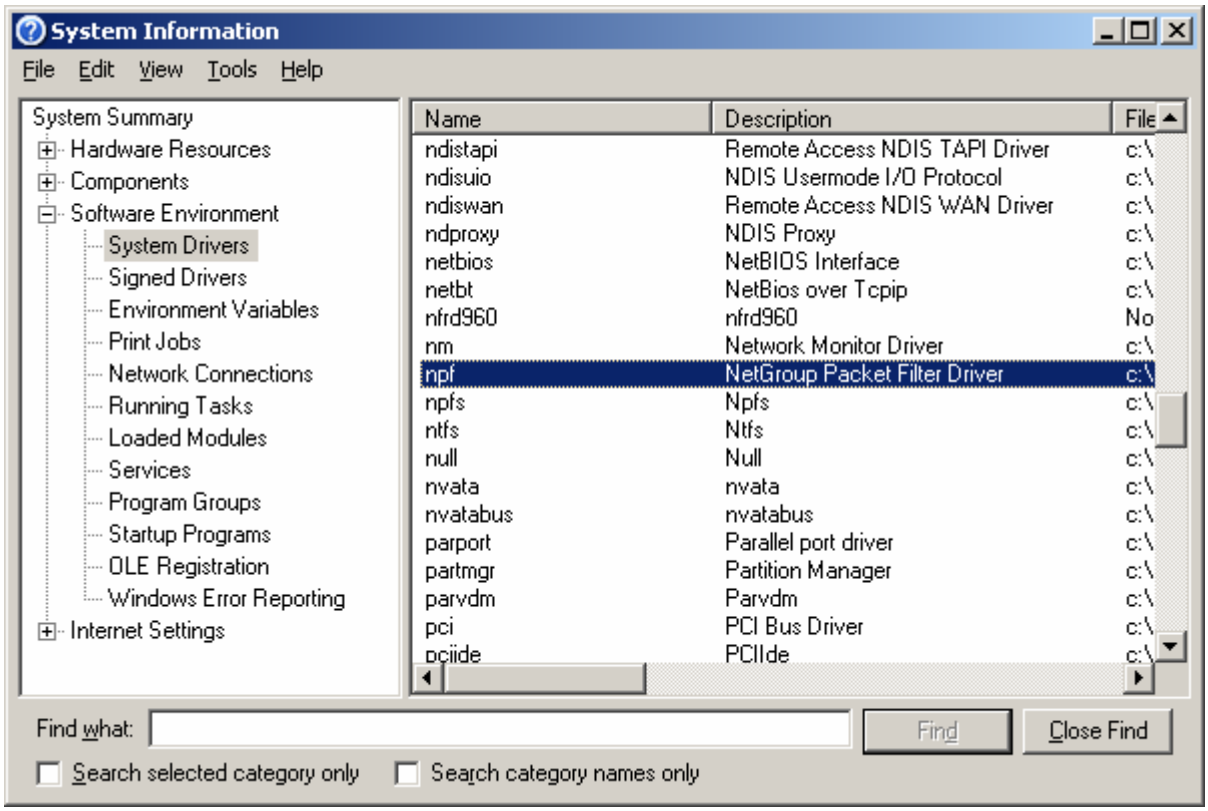


Figure 19: NPF - NetGroup Packet Filter Driver

The status of the NPF service is seen by scrolling to the right.

After WinPcap installation the column “started” should be set to “No”, the column “Start Mode” should show “Manual” and the column “State” shows that the service is “Stopped”. As soon as CTCI-W32 is active the values in these columns will change to “Started=Yes” and “State=Running”.

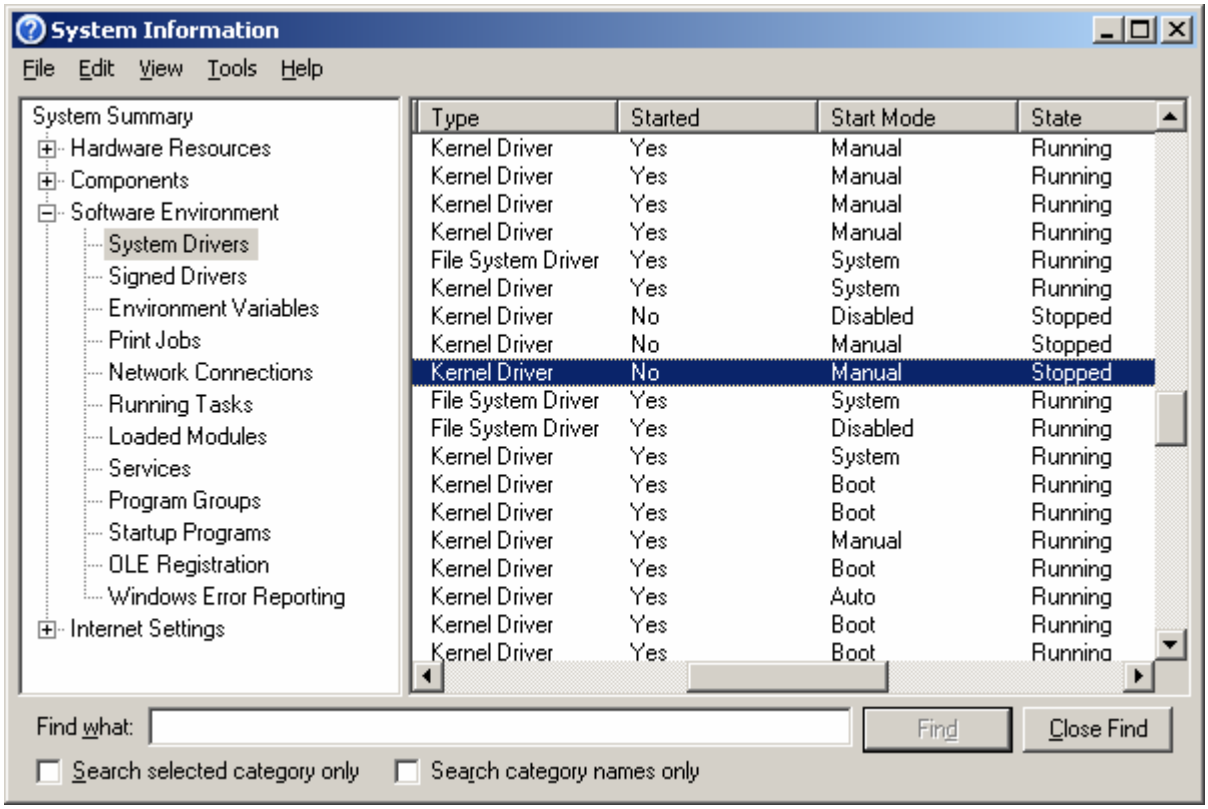


Figure 20: NetGroup Packet Filter Driver Status

Although the service will be started automatically on request of Hercules via CTCI-W32, some users may wish to have the service running automatically after Windows bootup. This behaviour can be controlled using the following registry setting:

`HKLM\System\CurrentControlSet\Services\NPF\Start`

`The key (REG_DWORD) may be one of the following values:`

- 0x0000001 (Disabled, the service is never running)
- 0x0000002 (Manual, the service is started on manual intervention)
- 0x0000003 (System, the service is started automatically on request)

By default this key is set to 0x00000003 and the service is started automatically on request by CTCI-W32.

8. Installation Cygwin

Figure 21: Cygwin Logo

8.1 Cygwin

Up to release 3.02.0 of Hercules the next component required to running Hercules under windows was the Cygwin environment. As Hercules was designed to run on a Linux system with POSIX threads (pthread) support, a Linux-like operating system is required as a base. Cygwin provides this Linux-like environment under Windows in the form of Unix system calls that translate to native Windows system calls, thus providing a native Unix layer under Windows.

Beginning with Release 3.03.0 the Hercules Emulator no longer requires Cygwin in order to run under windows, Hercules is now a pure Win-32 application. It is strongly recommended that Windows users of Hercules migrate from the Cygwin dependant versions to the new Microsoft Visual C (MSVC) version.

If you intend to install release 3.03.0 or later of Hercules you can skip this chapter and proceed directly with the installation of Hercules itself described in chapter 9.

8.2 Downloading the Setup Program

First create a directory \CYGWIN in one of your hard disk drives. In the following paragraphs we will assume you are using drive F: for the Cygwin installation. From here onwards replace references to drive F: with the actual drive letter where you created the Cygwin directory.

Next download the Cygwin setup program from RedHat and place it in F:\CYGWIN, you can download the setup program from:

http://sources.redhat.com/cygwin/

8.3 Downloading the necessary components

Start the setup program from F:\CYGWIN.

The Cygwin installation welcome screen will appear:

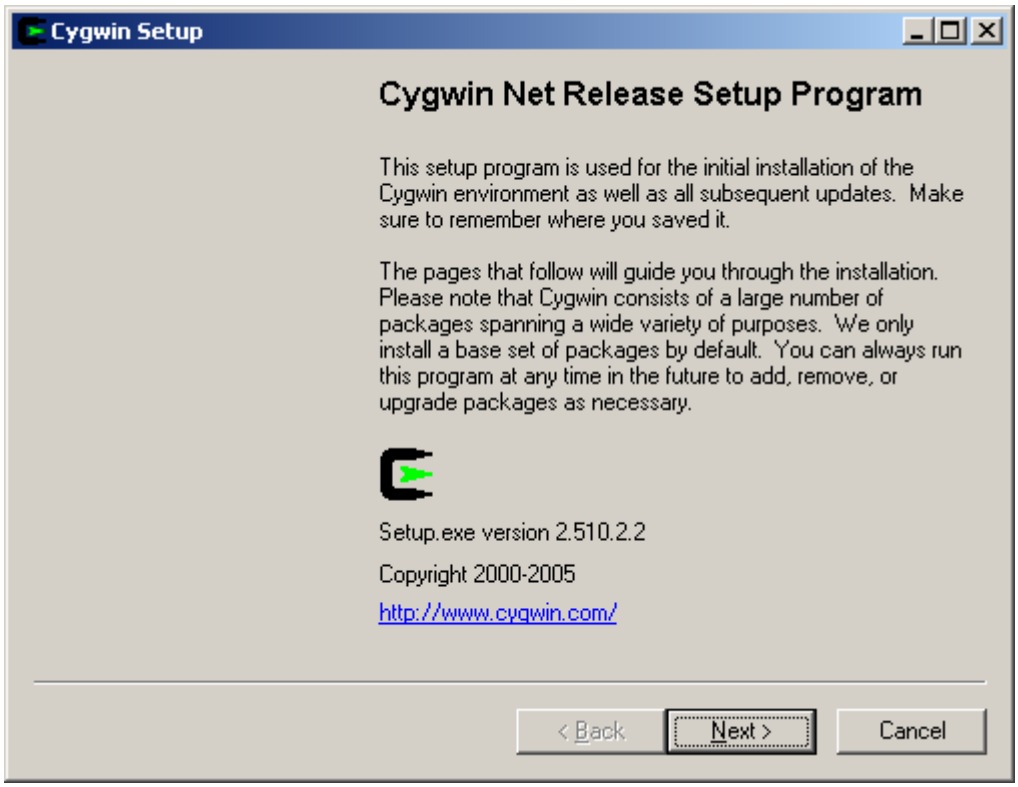


Figure 22: Cygwin Setup - Welcome Screen

On this screen click “Next >” to continue.

On the next screen you will be asked to choose between different download sources for Cygwin components, select “Install from Internet”.

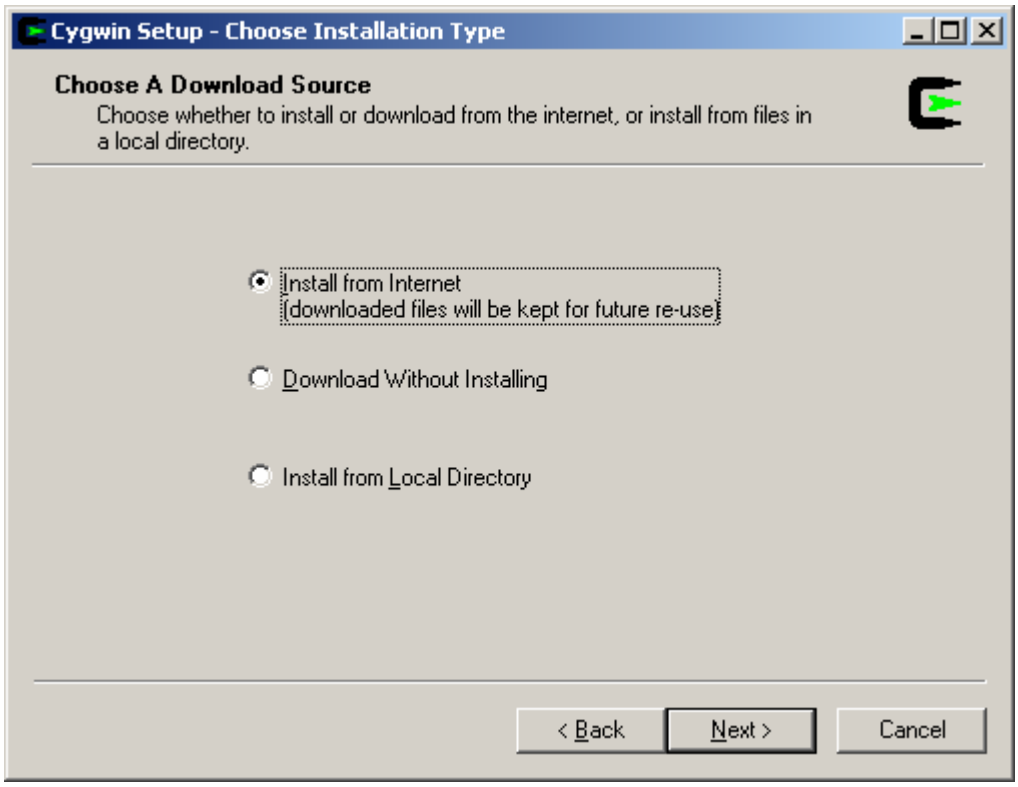


Figure 23: Cygwin Setup - Download Source

After clicking on “Next >” you must specify the root directory for Cygwin created earlier.

Enter the drive and the directory you created in the first step and leave other options as they are.

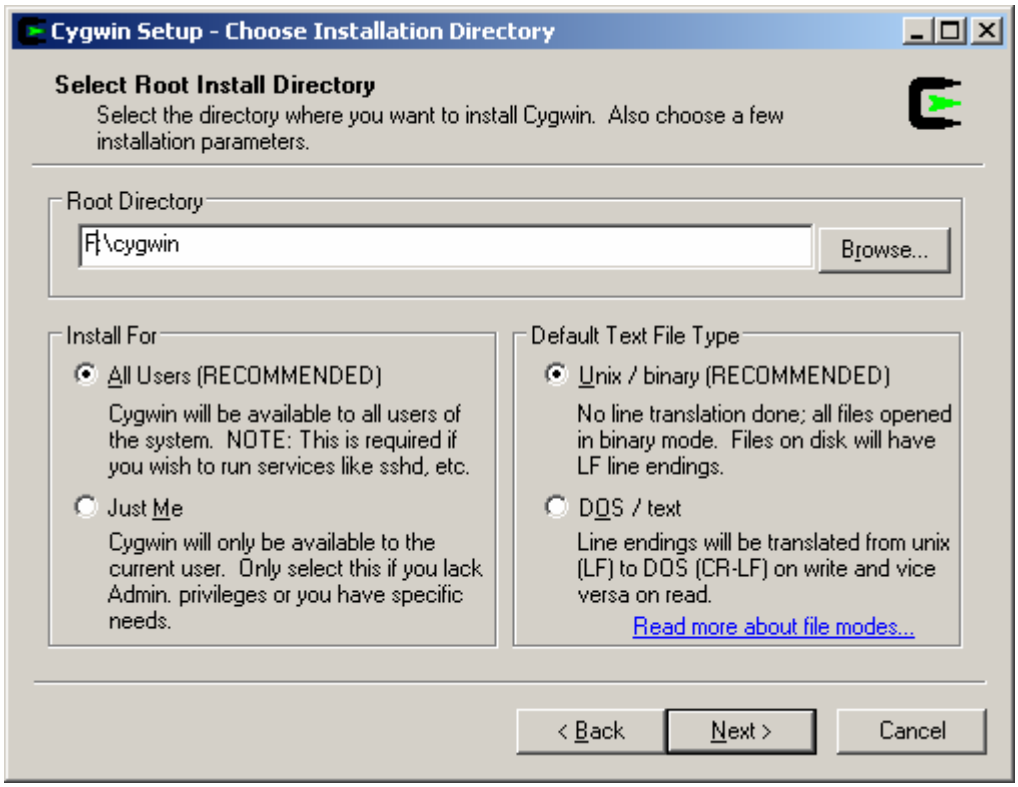


Figure 24: Cygwin Setup Installation Directory

Click “Next >” to continue.

You will now be asked for the local package directory. Enter a drive letter and a directory name. In this example we selected our F: drive again and chose CYGWIN as the name for our local package directory.

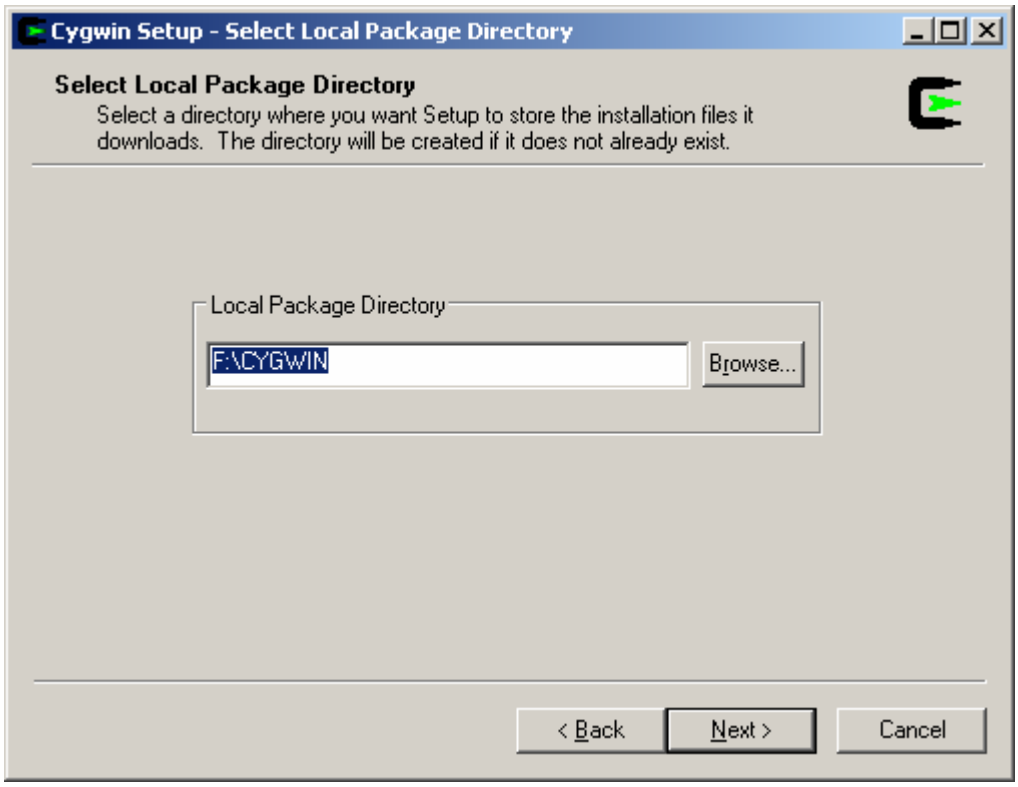


Figure 25: Cyqwin Setup – Local Package Directory

Again click on "Next >" to continue.

For the next step choose the connection type, usually “direct connection”.

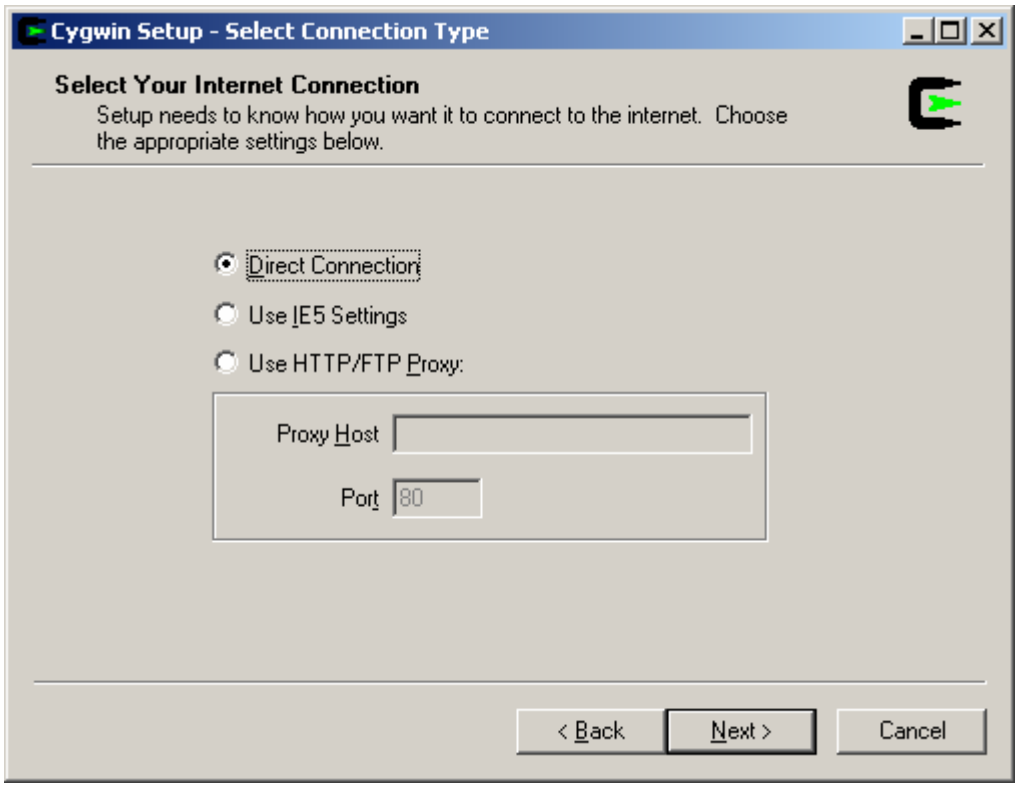


Figure 26: Cygwin Setup – Connection Type

After clicking on “Next >” the setup program will now connect to the internet and provide you with a list of download sites for the Cygwin system.

Choose a download site close to your current location.

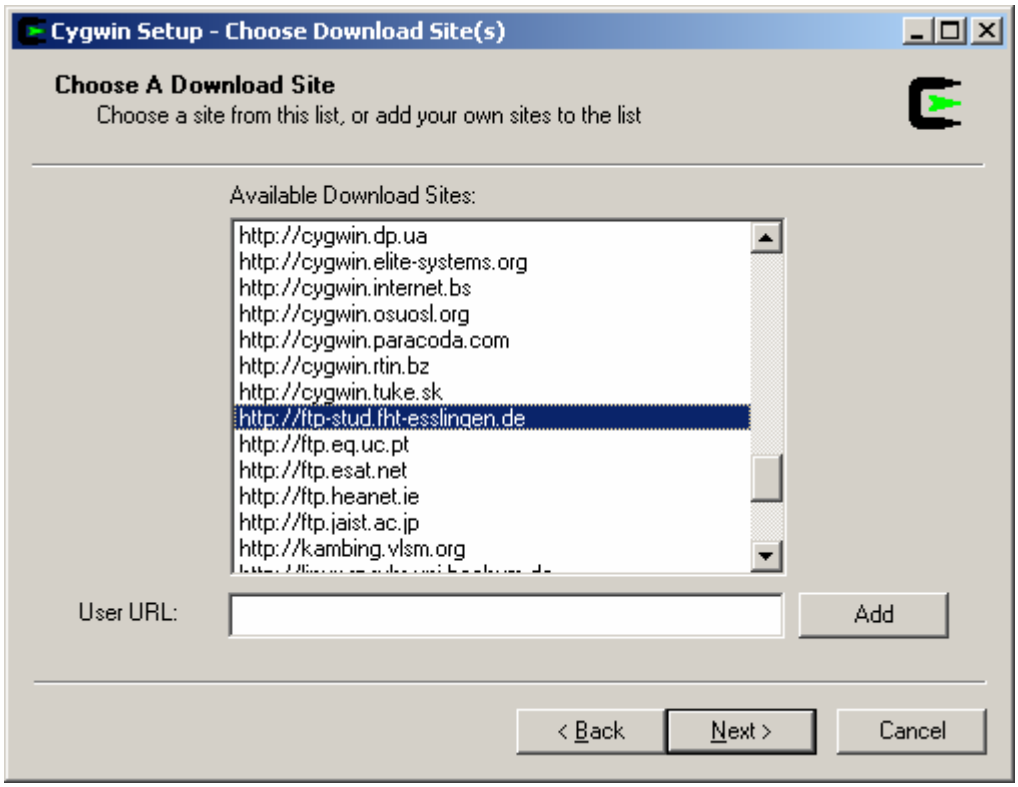


Figure 27: Cygwin Setup – Choose Download Site(s)

After selecting a download site click on "Next >" to continue.

You are now presented with a selection panel listing the available Cygwin categories.

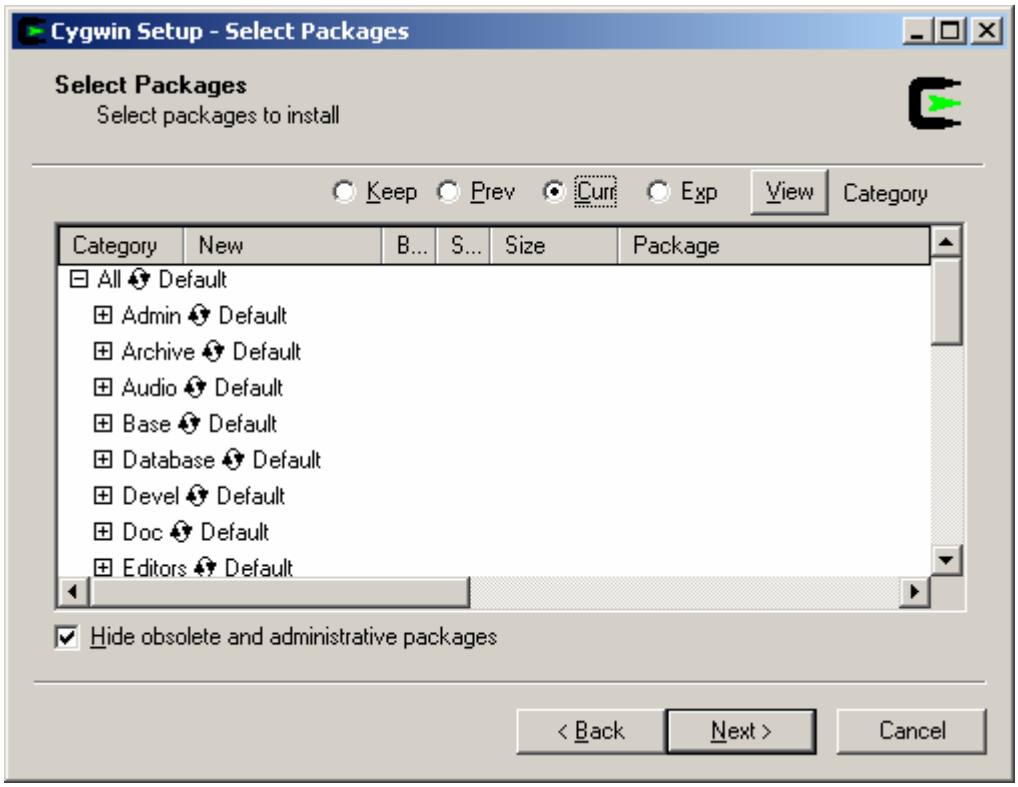


Figure 28: Cygwin Setup – Select Packages (1)

Hercules requires only packages from the ‘base’ category. Click on the circular arrow between the words “Base” and “Default”, this changes the selection from “Default” to “Install”.

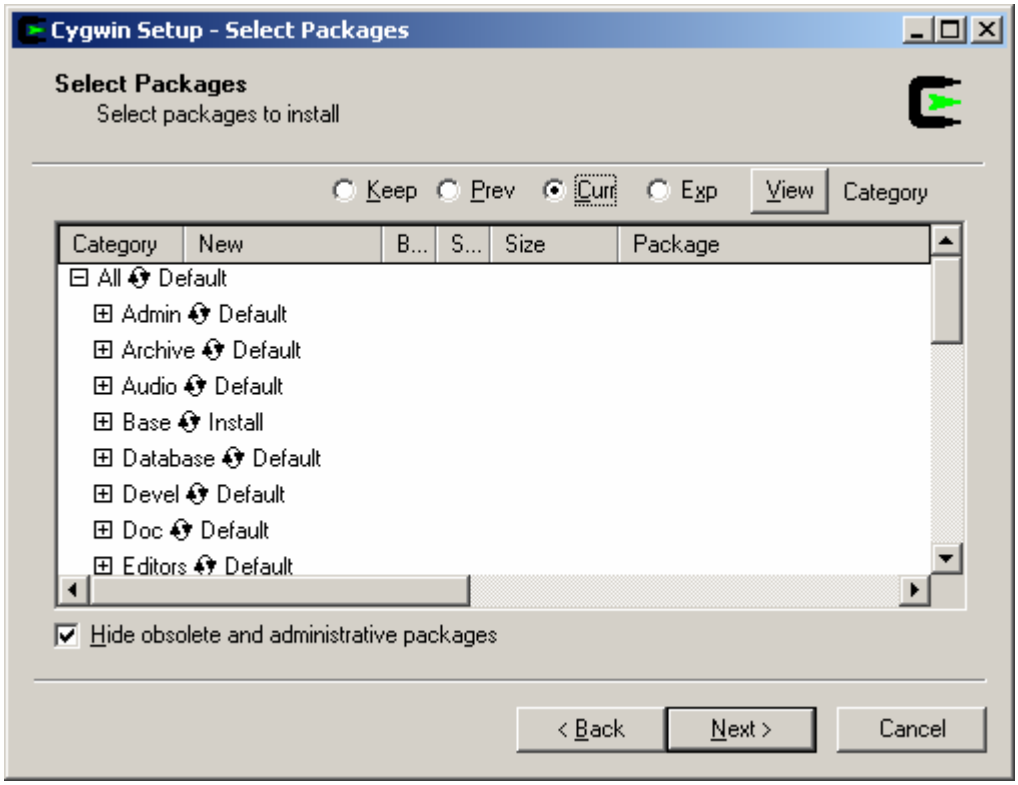


Figure 29: Cygwin Setup – Select Packages (2)

Expand the base package view by repeatedly clicking the “View” button until the view changes from “Category view” to “Full view”.

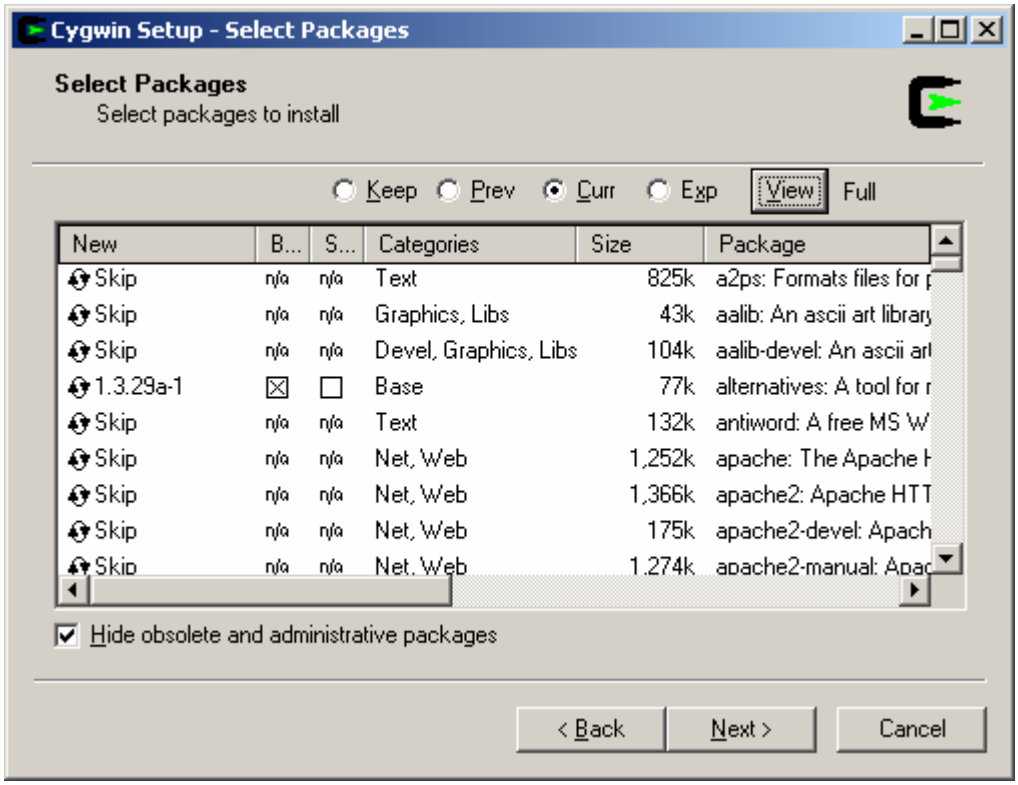


Figure 30: Cygwin Setup – Select Packages (3)

In this part of the Cygwin installation we must select the Unix equivalent packages that our application requires. Depending on your needs, select the appropriate packages listed in chapters 8.4 though 8.6. You have the choice of selecting the runtime environment only, the build environment or the development environment.

Be careful not to make any errors during this step. If you have to abort the installation you cannot go back and resume, you must restart the Cygwin installation from scratch.

When you are ready to continue click on “Next >” and the actual download will begin.

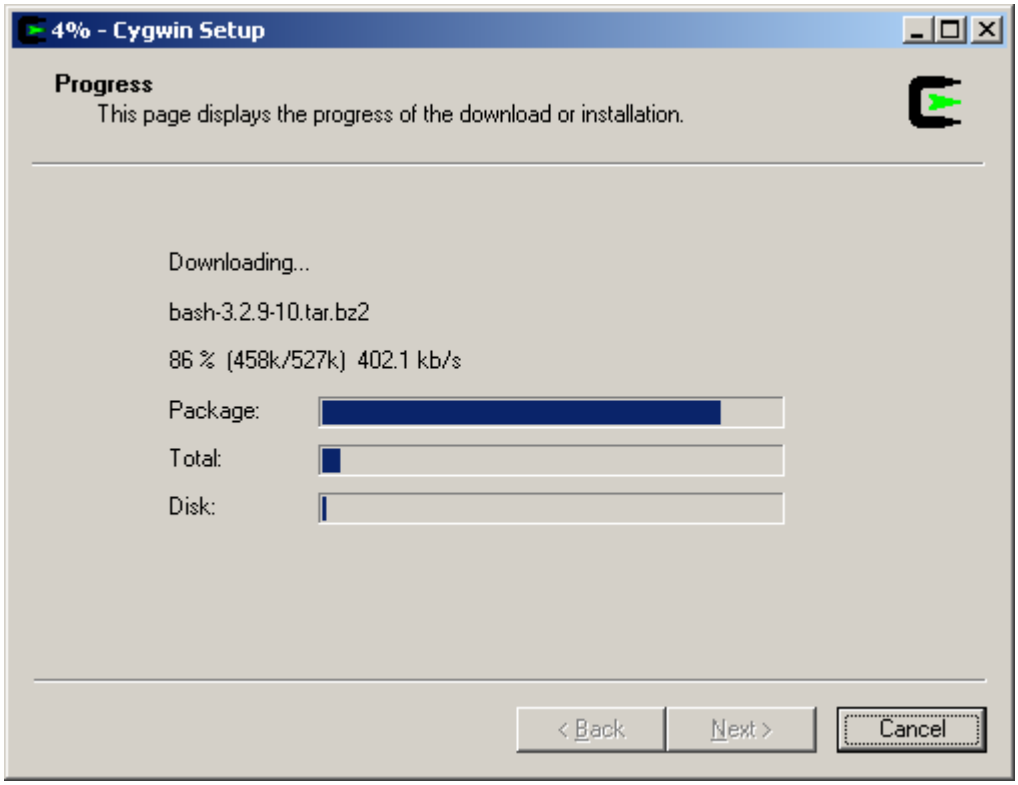


Figure 31: Cygwin Setup – Download process

After the Cygwin installer has downloaded the selected components the screen will be updated.

The Cygwin installation process will then unpack the downloaded files and perform the necessary installation tasks.

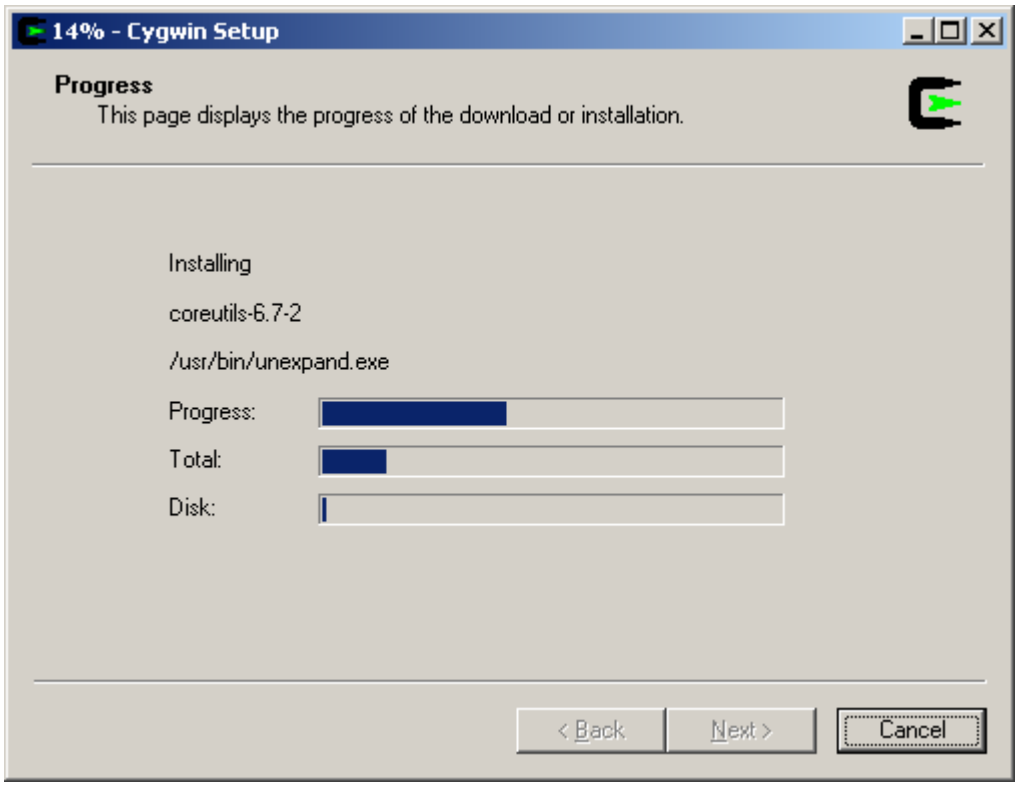


Figure 32: Cygwin Setup – Install process

This procedure may take some time to unpack and install all packages.

When Cygwin has finished this installation step it presents the option of creating icons in the startup menu and on the desktop. Hercules does not require these and the options can be safely unchecked.

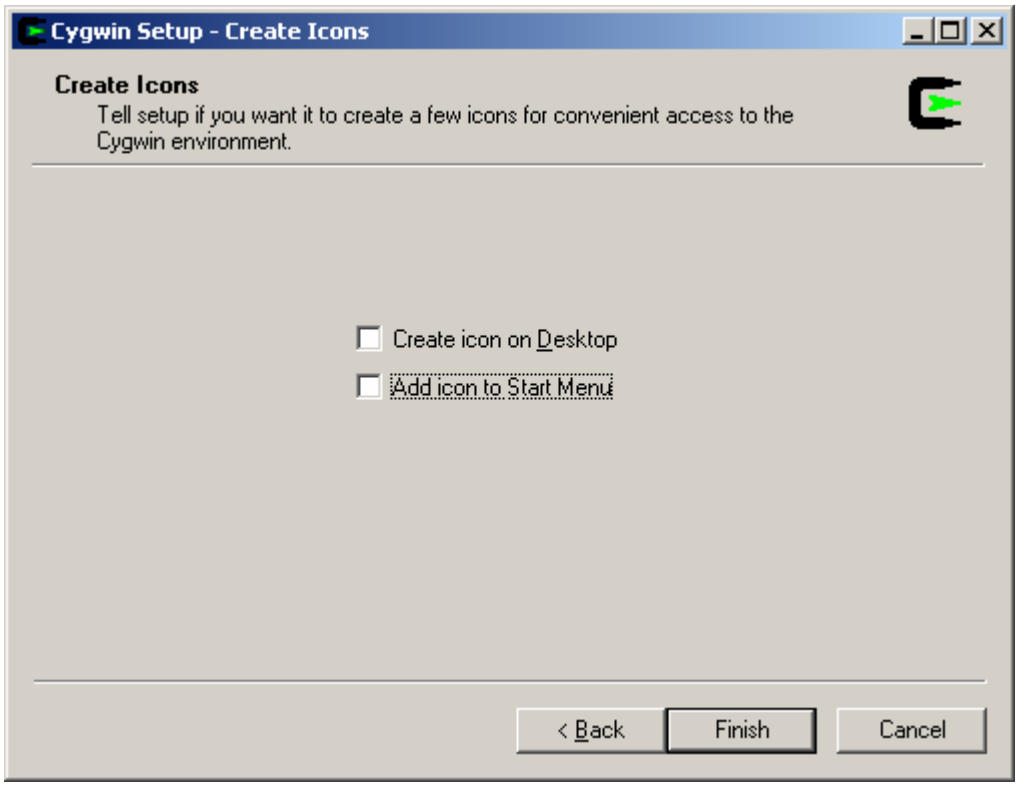


Figure 33: Cygwin Setup – Create Icons

After clicking the “Finish” button the installation of Cyqwin is complete. Cygwin leaves a directory beginning with “ftp...” in your Root Cygwin directory, containing the packed downloaded packages. These are no longer required and the directory can be safely deleted.

8.4 Components for the runtime environment

If you only want to run Hercules and do not intend to build your own binaries from the source code you only need select the following Cygwin packages for installation:

- bash
- bzip2
- cygwin
- cygwin-doc
- grep
- libbz2_1
- libiconv2
- libintl1
- libintl2
- libintl3
- login
- mc (not really required but provides an extremely useful file manager similar to Norton Commander for the native bash shell environment)
- netcat
- unzip
- zlib
- zip
- _postinstall

This is the complete Cygwin runtime environment for Hercules.

8.5 Components for the Cygwin build environment

If you want to be able to build your own Hercules binaries from the daily source snapshots you will have to add additional packages to those selected above. Start the Cygwin setup again, select the “FULL” view as before. Select the following packages in addition to those listed above:

- ash
- base-files
- base-passwd
- bash
- binutils
- clear
- fileutils
- file

- findutil
- gcc and everything related
- gzip
- less
- libiconv
- libpcre
- libtools
- make
- mktemp
- perl
- sh-utils
- tar
- textutils
- which

This will complete the Cygwin build environment for Hercules.

8.6 Components for the Cygwin development environment

If you want to work with the Hercules CVS repository instead of the daily developer snapshots, you will need these additional packages:

- cvs
- gettext
- gettext-devel
- libtool-devel (**not** required since Hercules Release 2.18)

This completes the Cygwin development environment for Hercules.

9. Installing the Hercules Emulator

9.1 Downloading the Binaries

The "ready-to-run" binaries can be downloaded from http://www.hercules-390.org. For Windows users there are currently three varieties available:

- MSVC Windows installer package (filename "hercules-*v.rr*-native.msi")
- MSVC self-extracting archive (filename "hercules-*v.rr*-native.exe")
- Cygwin self-extracting archive (filename "hercules-*v.rr*-cygwin-*xxxx*.exe")

Where *'v.rr'* in the filename specifies the version and release of the Hercules Emulator contained and '*xxxx'* in the Cygwin version specifies the CPU type for which the package has been built.

The difference between these binaries and how each is installed are described in the following sections.

9.2 Choosing a Package

Which of these packages is the right one for you depends on your needs. The Cygwin package although still available is no longer recommended and since Hercules release 3.03.0 Cygwin is no longer a pre-requisite.

The new MSVC packages, the self-extracting archive (.exe) and the Windows installer package (.msi), are very similar. The main difference being that the Windows installer package provides the standard Windows installation dialog for both installation and removal of the emulator. The Windows installer package also creates Start Menu entries for both the documentation and to start the DOS command prompt in the Hercules binary directory.

Some people want to have the full control over the whole installation process and therefore prefer the self-extracting archive (.exe). On the other hand however, some people prefer to have a simple installation routine, similar to other Windows software packages.

Note: If you haven't ever installed the MSVC (native) version of Hercules on your system before (i.e. if this is the first time you are installing it), please use the .msi Windows Installer package since it will ensure that the required Microsoft support DLLs are also installed. If you are simply upgrading from a previous MSVC (native) installation however, then since you only need the new Hercules binaries (and not the MS support DLLs), you may, if you wish, use the self-extracting archive instead.

9.3 Installation Steps (MSVC Windows Installer Package)

The "MSVC Windows Installer Package" is a standard Windows installation dialog as used for most Windows application installations. Using this method the 'C' runtime library files are installed if needed in a separate directory structure under the Windows system directory (i.e. "%windir%\WinSxS").

This new Windows technique (SxS means "Side by Side") allows multiple versions of the same DLL to coexist in the same system. For example should application “A” need foobar.dll version 1.12 and application “B” needs foobar.dll version 1.14, both DLLs can coexist in the same system.

To start the installation dialog run the .MSI executable file. A welcome window is presented first.

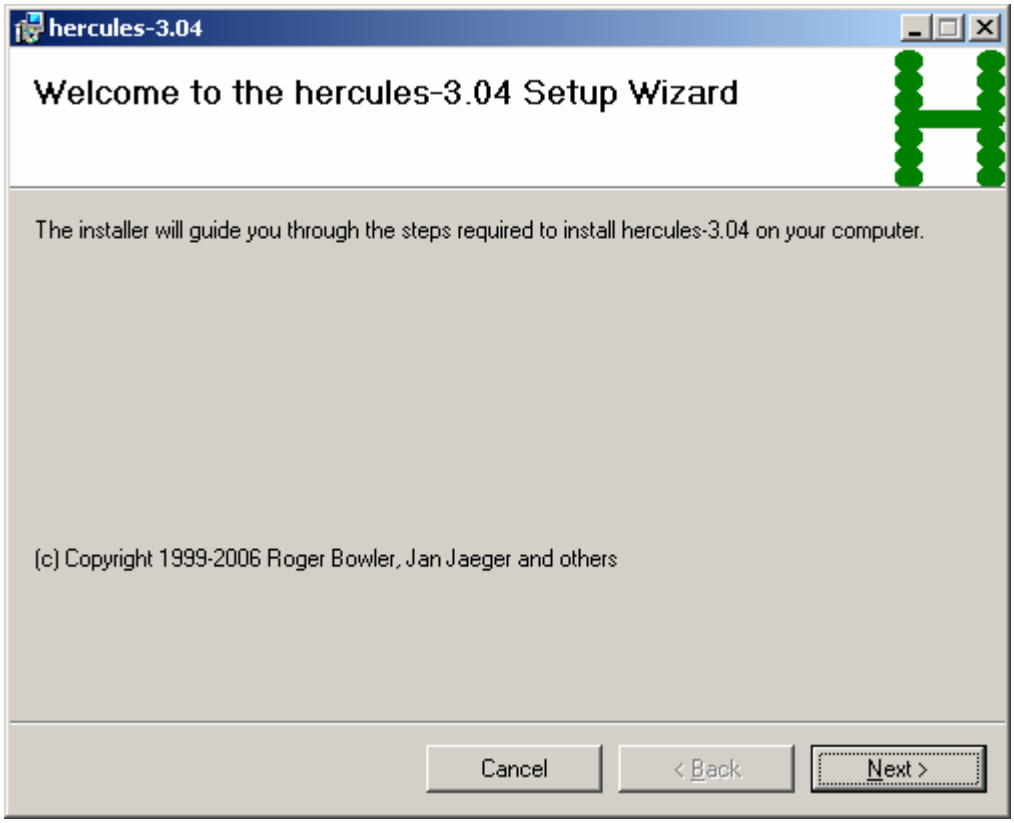


Figure 34: Welcome Window (MSVC Installer Package)

Clicking on "Next" continues the installation process.

Now a license agreement window is presented. It explains in detail the "Q Public License". Please read the license agreement.

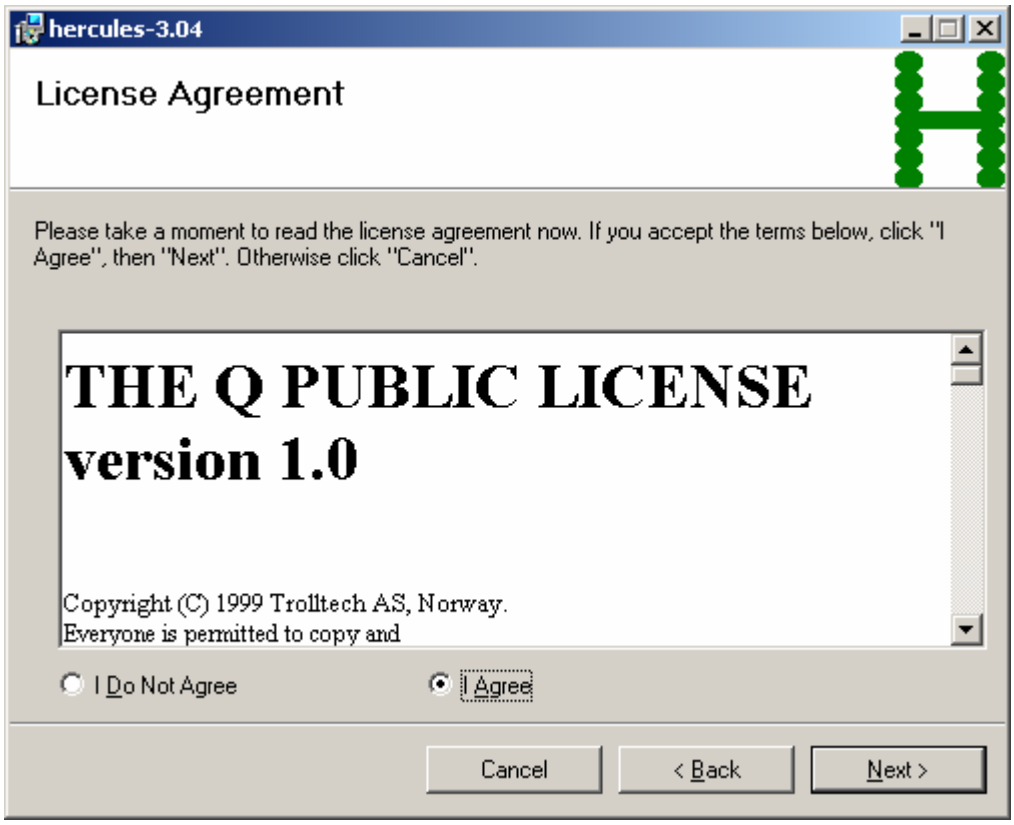


Figure 35: License Agreement (MSVC Installer Package)

After reading the license agreement if you accept the terms confirm this using the radio button "I agree" then click on "Next" to continue with the installation.

The next window lets you specify the installation folder. You can use the default or specify any directory of your choice.

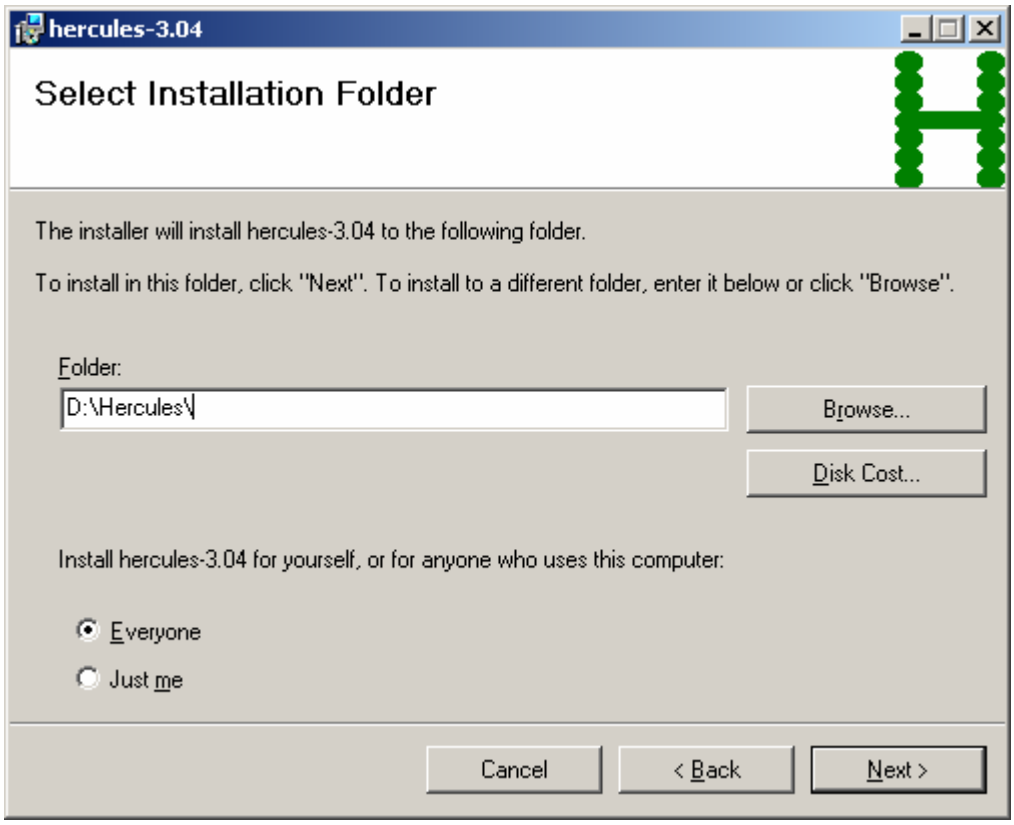


Figure 36: Installation Directory Selection (MSVC Installer Package)

If you plan to run different Hercules versions in parallel then the default naming convention which contains the version and release levels in the directory name is useful. If your intention is to run one Hercules version at a time then you may prefer to shorten the directory name to simply "hercules" as in the example above.

Before installing the package you can examine your available disk space by clicking on the button "Disk Cost". This will display all available disk partitions on which Hercules can be installed and shows the disk space before and after installing the package. Hercules itself, without a mainframe operating system installed, uses approximately 11 MB of disk space.

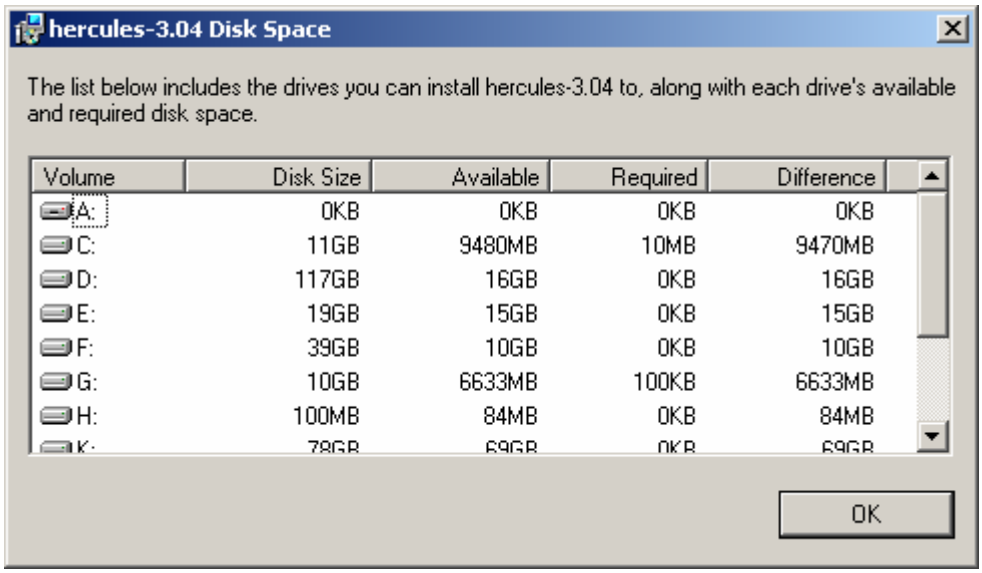


Figure 37: Disk Space Information (MSVC Installer Package)

When you are satisfied with the installation options, click on "OK".

A confirmation dialog is presented, this is a final opportunity to change any of your selections made in the previous screens.

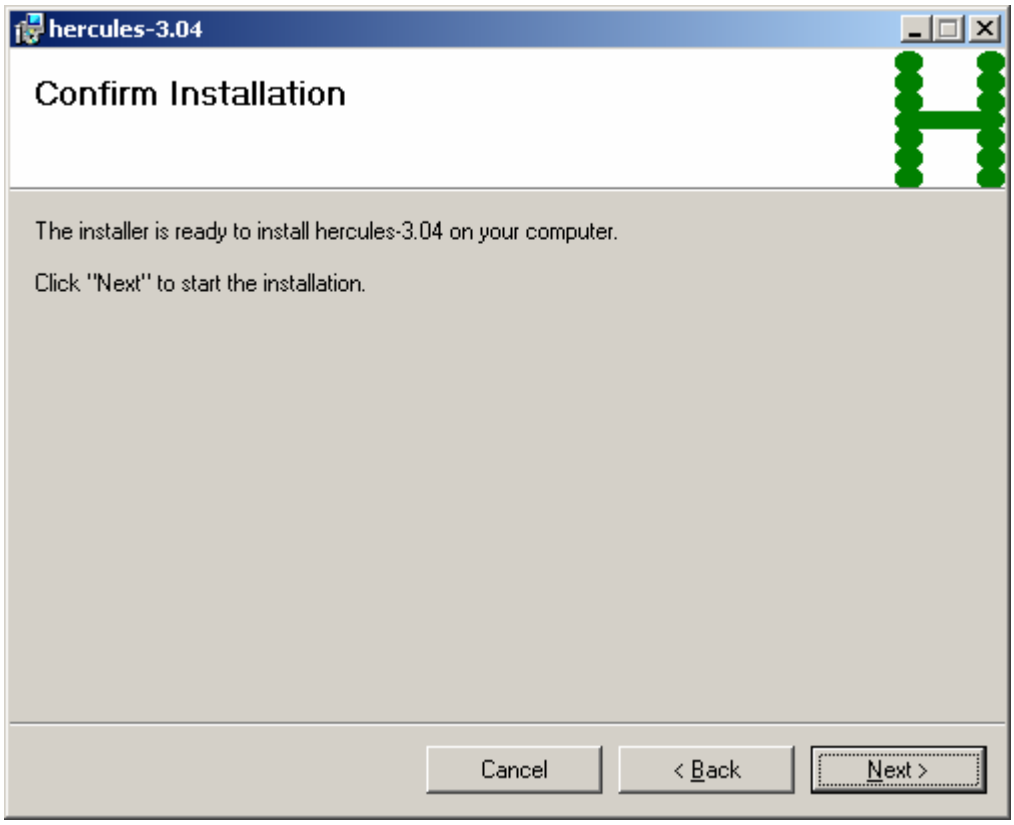


Figure 38: Installation Confirmation (MSVC Installer Package)

If you do not want to change any of your choices, click on "Next >" to start the actual installation process.

The installer now begins to copy files to the destination directories.

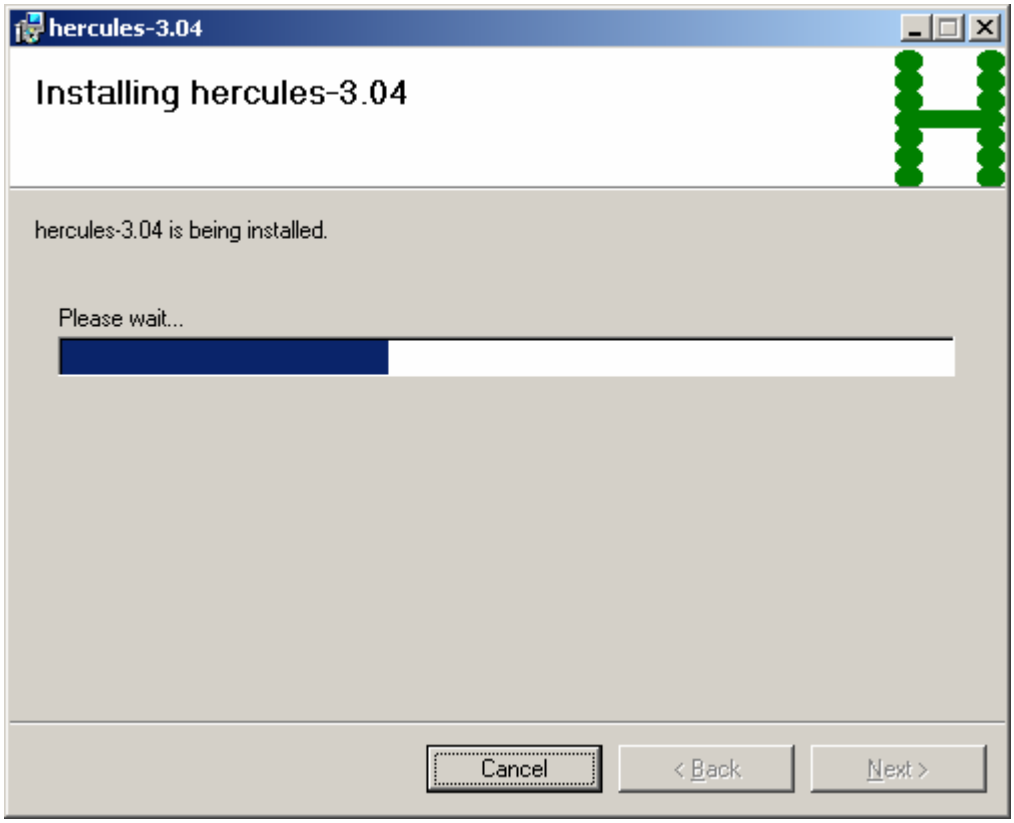


Figure 39: Installation Progress Bar (MSVC Installer Package)

If necessary the process of copying files can be stopped by clicking on "Cancel".

After a few seconds the installation process will finish and present the final window.

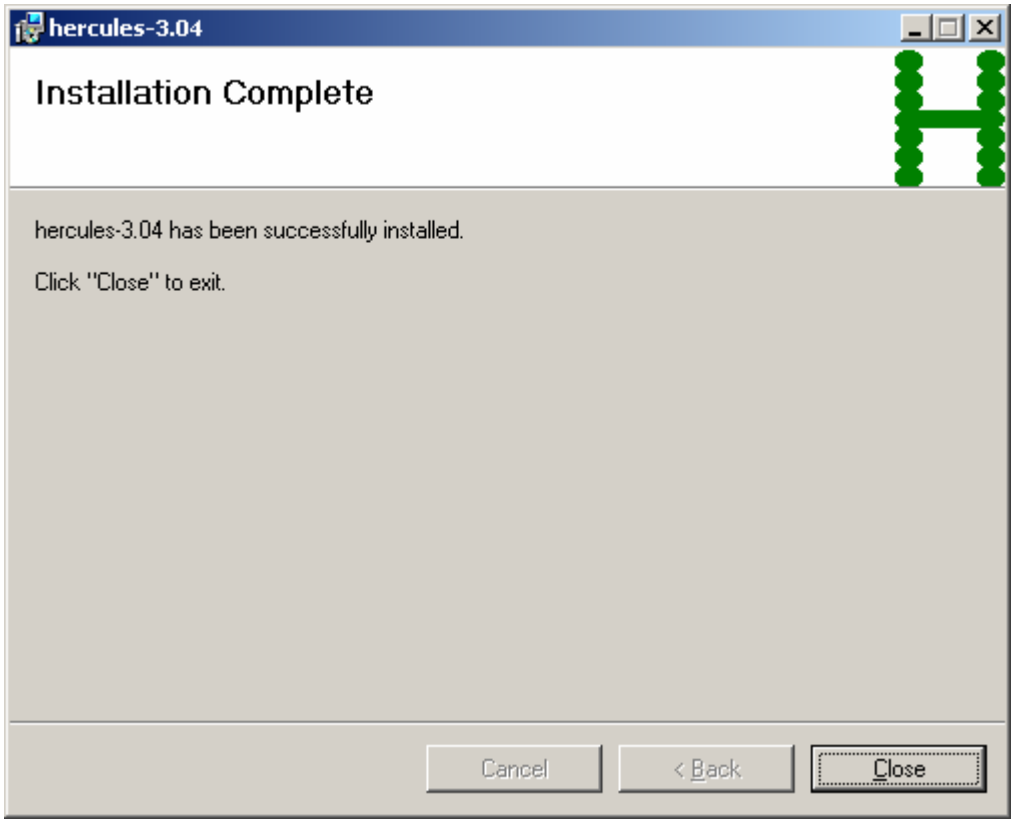


Figure 40: Installation Complete (MSVC Installer Package)

Click on "Close" to terminate the installation dialog and proceed to customise your installation as described in the following sections.

9.4 Installation Steps (MSVC Self-Extracting Archive)

The MSVC native version of the Hercules Emulator is delivered as a self-extracting archive. In this archive the binaries and all other necessary files (e.g. html files etc.) are placed in a predefined directory structure required to run Hercules.

The self-extracting .exe does not invoke the Windows installer therefore will not install the "Microsoft Visual Studio 8 'C' Runtime Library" as a system file. Instead a technique is used where the 'C' runtime DLLs are installed as a separate subdirectory of the application together with the manifest files. This is designed to ensure that Hercules will run even if the proper 'C' runtime library is not present on the system.

It is normal for the self-extracting archive to have a subdirectory named ‘Microsoft.VC80.CRT’ and for the Microsoft Installer version to not have this subdirectory.

To extract the files from the archive run the executable. This will present a window where you will be asked to specify the target directory for the installation. It is sufficient to specify only the drive letter of the target environment as the necessary subdirectory structure is created automatically.

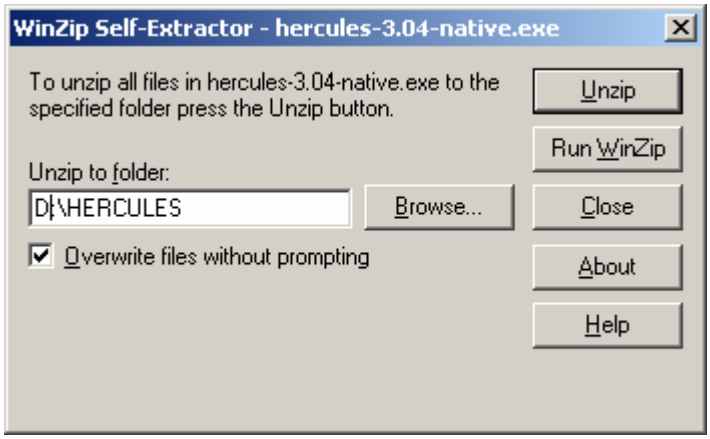


Figure 41: Specifying Target Directory (MSVC Self-Extracting Version)

After clicking the "Unzip" button the directories are created and files are extracted and copied to the proper locations. Alternatively WinZip can be called to extract the files manually. When all files are copied a confirmation window appears.

Figure 42: Confirmation Message (MSVC Self-Extracting Version)

The root directory that is created is called hercules-*v.rr*, where *v* stands for the current version and *rr* for the current release. This naming convention is useful if you plan to run several Hercules versions in parallel. If your intention is to run only one Hercules version at a time the directory can be renamed to "hercules" without a version or release indication.

Next proceed to section 9.6 ‘Customisation Steps’ below.

9.5 Installation Steps (Cygwin Self-Extracting Archive)

The Cygwin version of the Hercules Emulator is delivered as a self-extracting archive with three separate compilations for different CPU types available:

- P4 (Pentium 4 CPUs)
- i686 (Pentium Pro, Pentium II, Pentium III, AMD Athlon and up)
- i586 (Pentium and AMD K6)

Please download the correct package for your target environment. While some combinations will run with only degraded performance (i.e. the i586 package running on a P4 machine), other combinations may fail to execute at all (i.e. P4 package running on an AMD K6 CPU).

In these archives the binaries are placed in a directory structure that corresponds exactly to the Cygwin directory structure established during the Cygwin installation process.

To extract the files from the archive run the executable (.exe) file. This will first present an information window describing briefly the archive contents for your information. The following package details are listed:

- Hercules Emulator version
- Thread model
- CPU type for target system
- Maximum number of emulated processors

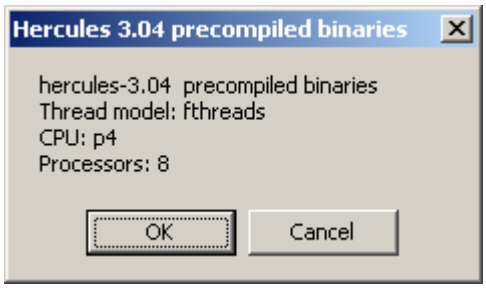


Figure 43: Hercules Archive Info Window (Cygwin Version)

After clicking OK you are asked to specify the target directory, this must be the root directory of the Cygwin installation. Alternatively it is possible to start WinZip from this directory and unzip the files manually.

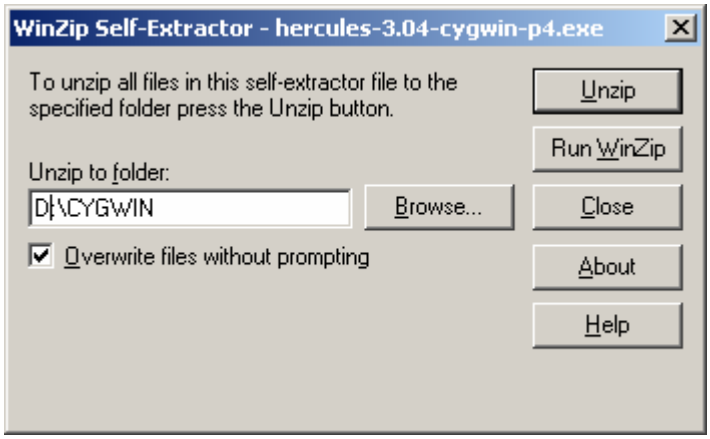


Figure 44: Specifying Target Directory (Cygwin Version)

After clicking to the "Unzip" button the files are extracted from the archive and copied to the specified directory. In the above example the Hercules files would be placed in several subdirectories in the path "D:\CYGWIN\usr\local\".

After unzipping all files a confirmation window is presented.

Figure 45: Confirmation Message (Cygwin Version)

At this point all necessary Hercules files are placed in their correct locations in the Cygwin directory structure. Next proceed to section 9.6 'Customisation Steps' below.

9.6 Customization Steps

After the actual installation is complete there are additional customization steps required, these include:

- Creating the Hercules Configuration File
- Creating the Hercules Startup Batch File
- Creating the Hercules Run-Commands File
- Creating the Terminal Batch File

These manual customization steps are explained in detail over the following sections.

9.6.1 Creating the Hercules Configuration File

When starting the Hercules Emulator from either a DOS command line or via the Hercules Windows GUI, you may specify the name of a configuration file as a parameter:

HERCULES [-f filename] …

HERCGUI [-f filename] …

where *filename* is the name of the configuration file. The default filename if none is specified during the startup is 'hercules.cnf'. The name of the default configuration file may be overridden via the environment variable HERCULES_CNF.

The configuration file is an ASCII text file that is used to describe the processor definition, the device layout and any runtime parameters. Details of the format and acceptable directives that can be made within the file are found in Chapter 3 of the Hercules User Reference Guide.

9.6.2 Creating the Hercules Startup Batch File

Although the Hercules Emulator can be started manually from a DOS command prompt it is often easier to establish a batch file to do this. The batch file can then be executed by double-clicking it from Windows Explorer, called directly from a DOS command prompt window or it can be integrated as a shortcut into the Windows start menu or desktop.

9.6.2.1 Hercules Startup Batch File

The following figure shows an example of a Hercules startup batch file using the native Hercules console.

```
@ECHO OFF
REM
REM ************************************************************************************
REM CHECK / SET HERCULES PATH                                                          (1)
REM ************************************************************************************
REM
IF %SETHERC%.==1. GOTO RUNIT
SET SETHERC=1
PATH D:\Hercules;%PATH%
SET HERCULES_RC=D:\MVS\CONF\HERCULES.RC
:RUNIT
REM
REM ************************************************************************************
REM START HERCULES EMULATOR                                                            (2)
REM ************************************************************************************
REM
START D:\Hercules\Hercules -f D:/MVS/CONF/MVS38J.CONF >D:\MVS\LOG\Hercules_Log.txt
EXIT
```

Figure 46: Hercules Startup Batch File

This example batch file contains two main sections:

(1) In the first section a check is made to determine if the path to the Hercules binaries has already been set. If so, then the rest of the first section is skipped. If not, then the path to the Hercules binaries is set and the environment variable %SETHERC% is set to '1' to indicate that the path has been successfully set.

(2) In the second section the Hercules Emulator is started with a configuration file located in a directory other than the emulator itself. Additionally the path and filename for the log file is set.

9.6.2.2 Hercules Windows GUI Startup Batch File

The following figure shows an example of a Hercules startup batch file using the Hercules Windows GUI.

```
@ECHO OFF
REM
REM ************************************************************************************
REM CHECK / SET HERCULES PATH                                                          (1)
REM ************************************************************************************
REM
IF %SETHERC%.==1. GOTO RUNIT
SET SETHERC=1
PATH D:\Hercules;%PATH%
SET HERCULES_RC=D:\MVS\CONF\HERCULES.RC
:RUNIT
REM
REM ************************************************************************************
REM START HERCULES WIN GUI AND HERCULES EMULATOR                                       (2)
REM ************************************************************************************
REM
```

```
START D:\Hercules\HercGui -f D:/MVS/CONF/MVS38J.CONF
EXITEXIT
```

Figure 47: Hercules Windows GUI Startup Batch File

This example batch file contains two main sections:

(1) In the first section a check is make to determine if the path to the Hercules binaries has been already set. If so, then the rest of the first section is skipped. If not, then the path to the Hercules binaries is set and the environment variable %SETHERC% is set to '1' to indicate that the path has been successfully set.

(2) In the second section the Hercules Windows GUI is started which will in turn start the Hercules Emulator, using a configuration file located in a separate directory to the emulator itself. Differently to the first example above, the filename and path of the log file are not given as a parameter to the Windows GUI, rather they are configured within the GUI itself.

9.6.3 Creating the Hercules Run-Command File

Hercules also provides the ability to automatically execute Hercules panel commands after startup via the 'run-commands' file. If the run-commands file exists when Hercules starts, each line contained in the file is read and interpreted as panel command.

In the following example file the Windows telnet program is started in a shell, all currently valid Hercules panel commands are displayed and MAXRATES is reset. Finally a couple of tn3270 sessions for consoles and TSO terminals are started.

```
sh startgui telnet localhost 3270
?
maxrates
sh startgui C:\Programs\Vista32\vista32.exe MVS_MST.ses
pause 3
sh startgui C:\Programs\Vista32\vista32.exe MVS_OPR.ses
pause 3
sh startgui C:\Programs\Vista32\vista32.exe MVS_TSO.ses
pause 3
sh startgui C:\Programs\Vista32\vista32.exe MVS_TSO.ses
pause 3
sh startgui C:\Programs\Vista32\vista32.exe MVS_TSO.ses
```

Figure 48: Hercules Run-Commands File

If the "sh" (shell) command is used in the run-commands file, then the "startgui" keyword should also be used when starting external applications if one of the following is true:

a) If starting Windows GUI applications (as opposed to command-line programs), regardless whether or not the Hercules Windows GUI is being used.

b) If starting any program, Windows GUI programs or command-line programs (either via the Hercules panel command-line or via the run-commands file), when running Hercules via the Hercules Windows GUI.*

* If you wish to capture the output of a shell command however (e.g. "sh dir"), then you should not use startgui since it prevents the output from being captured / piped back to Hercules or the Hercules Windows GUI.

Creative use of the run-commands file can completely automate Hercules startup and initiate the IPL of the mainframe operating system. For details about the usage of the run-commands file see the "Hercules User Reference Guide".

9.6.4 Creating the Terminal Batch File

When the Hercules Emulator is been started using one of the previously described batch files, terminals (consoles and terminals) must then be connected. If they are not automatically started via the rc-file as described in the section above, the easiest way to do this is also with a separate batch file, which can start several instances of a terminal emulation program.

In the example file below several terminals are connected to the Hercules emulator using tn3270 emulation software "Vista tn3270" from Tom Brennan. The initiation of terminal connections to Hercules is particular to the 3270 emulator you chose to use – consult the documentation of your emulator software for these details.

```
@ECHO OFF
REM
REM ************************************************************************
REM Start master console (0700) and wait for connection to Hercules                    (1)
REM ************************************************************************
REM
START "C:\PROGRAMS\VISTA32\VISTA32.EXE C:\PROGRAMS\VISTA32\MVS_MST.SES"
PAUSE 2
REM
REM ************************************************************************
REM Start operator console (0701) and wait for connection to Hercules                  (2)
REM ************************************************************************
REM
START "C:\PROGRAMS\VISTA32\VISTA32.EXE C:\PROGRAMS\VISTA32\MVS_OPR.SES"
PAUSE 2
REM
REM ************************************************************************
REM Start TSO terminals (0702-0704) and wait for connection to Hercules                (3)
REM ************************************************************************
REM
START "C:\PROGRAMS\VISTA32\VISTA32.EXE C:\PROGRAMS\VISTA32\MVS_TSO.SES"
PAUSE 2
START "C:\PROGRAMS\VISTA32\VISTA32.EXE C:\PROGRAMS\VISTA32\MVS_TSO.SES"
PAUSE 2
START "C:\PROGRAMS\VISTA32\VISTA32.EXE C:\PROGRAMS\VISTA32\MVS_TSO.SES"
EXIT
```

Figure 49: Terminal Batch File

This example terminal batch file contains three sections:

(1) Connecting the master console

(2) Connecting the alternate (operator) console

(3) Connecting the TSO terminals

Generally these three sections are identical except for the session profile used. The number of terminals to be started depends on your needs although it is recommended that at least three terminal sessions (master console, alternate console and TSO terminal) be started.

The WAIT command is used to define a one second delay and helps ensure that the terminal sessions connect in the correct sequence to the intended addresses. The length of this wait interval depends on the speed of the host system CPU and should be changed to a value appropriate for your environment. You may need to experiment with this setting in order to discover an appropriate value.

10. Installing the Hercules Windows GUI

10.1 Downloading the Binaries

The Hercules Windows GUI can be downloaded from www.softdevlabs.com. Several other programs from the developer (David B. Trout, aka 'Fish') require a custom DLL, FishLib.dll, it may be appropriate to download the FishLib package as well.

The GUI also requires some Microsoft Foundation Classes (MFC) and C runtime DLLs (MFC42.DLL, MSVCRT.DLL, MSVCP60.DLL and DbgHelp.DLL). It is possible that these DLLs are already present in the Windows system directory. If not though, they must be downloaded and copied to the Windows system directory. These DDLs are all available from www.softdevlabs.com.

The source code for the Hercules Windows GUI is no longer available due to current efforts to commercialise this product.

Note: Beginning with release 1.11.1.5265 of the Windows GUI additional DLLs are required. These are Microsoft MFC and VC Runtime DLLs that you can download from the address mentioned above. The installation takes a few seconds and does not require a reboot. There are a 32-bit and a 64-bit version of these DLLs. Please ensure you are using the correct one according to the product you are installing (32-bit or 64-bit version of the Windows GUI).

These DLLs are normally automatically installed as part of a standard Microsoft Installer program (.msi), however as the Windows GUI does not have an installer program you will need to do this manually. Note that you only need to install these C Runtime DLLs once even if new versions of HercGUI are subsequently installed.

10.2 Installation Steps

The installation involves simply unzipping the executables into the directory where the Hercules executables reside. No other installation steps are required. Uninstalling is the reverse, just delete the files.

On its first startup the Hercules Windows GUI adds a new key to the Windows registry to hold some configuration settings. To completely remove all trace of the GUI you must manually delete this key using a registry editor such as the Microsoft "regedit" utility.

The Hercules Windows GUI uses the following registry key:

```
HKEY_CURRENT_USER/Software/Software Development Laboratories/Hercules
```

10.3 Customization Steps

The first time the Hercules Windows GUI is started it will open the Preferences dialog. Completing this dialog is the only customization required, it adds necessary information (e.g. directory names) required for the GUI to operate.

The following topics describe the various dialogs in detail. Please note that this installation section describes many of the tasks involved in the use of the GUI in preference to separating this information into the Hercules User Reference Manual.

10.4 Main Screen

The various panels of the main screen can be re-located to suit the user's preferences. The device list bar can be docked on either side of the screen and the various status bars at either the top or bottom of the screen. You can float them in a window by themselves or you can hide them altogether.

To do this just grab the panel using your mouse and drag it to your preferred location. The screen layout is remembered across sessions. If you place the controls panel at the bottom of the screen, then it will be in the same place the next time you start Hercules.

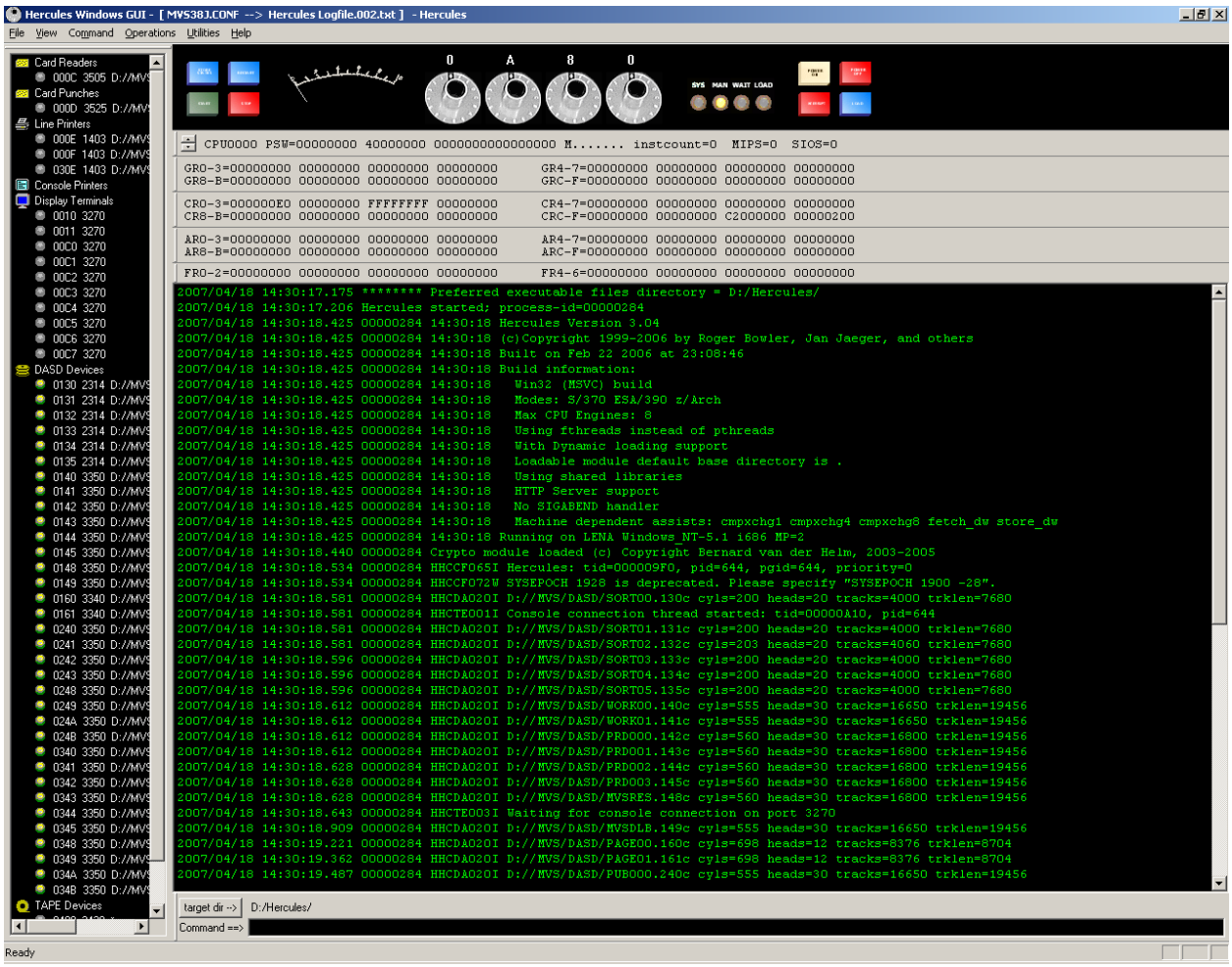


Figure 50: Hercules Windows GUI Main Panel

Since version 1.9.5 the Hercules Windows GUI allows you to specify the target directory that the 'sh' (shell) command will use. Simply browse to and select the desired directory using the GUI.

Once this is set any shell command entered will be processed using the defined directory as its current working directory. This provides a workaround for the fact that the Windows GUI's current directory changes depending on the dialog in use. This also compensates for the fact that prior to version 3.03 Hercules's current directory normally never changes.

10.5 Preferences

The Preferences dialog is where directories, file extensions, logging options etc are defined. The following subtopics describe each of the Preferences tabs in detail.

10.5.1 Directories

The Directories tab allows you to specify the preferred directories for each file type.

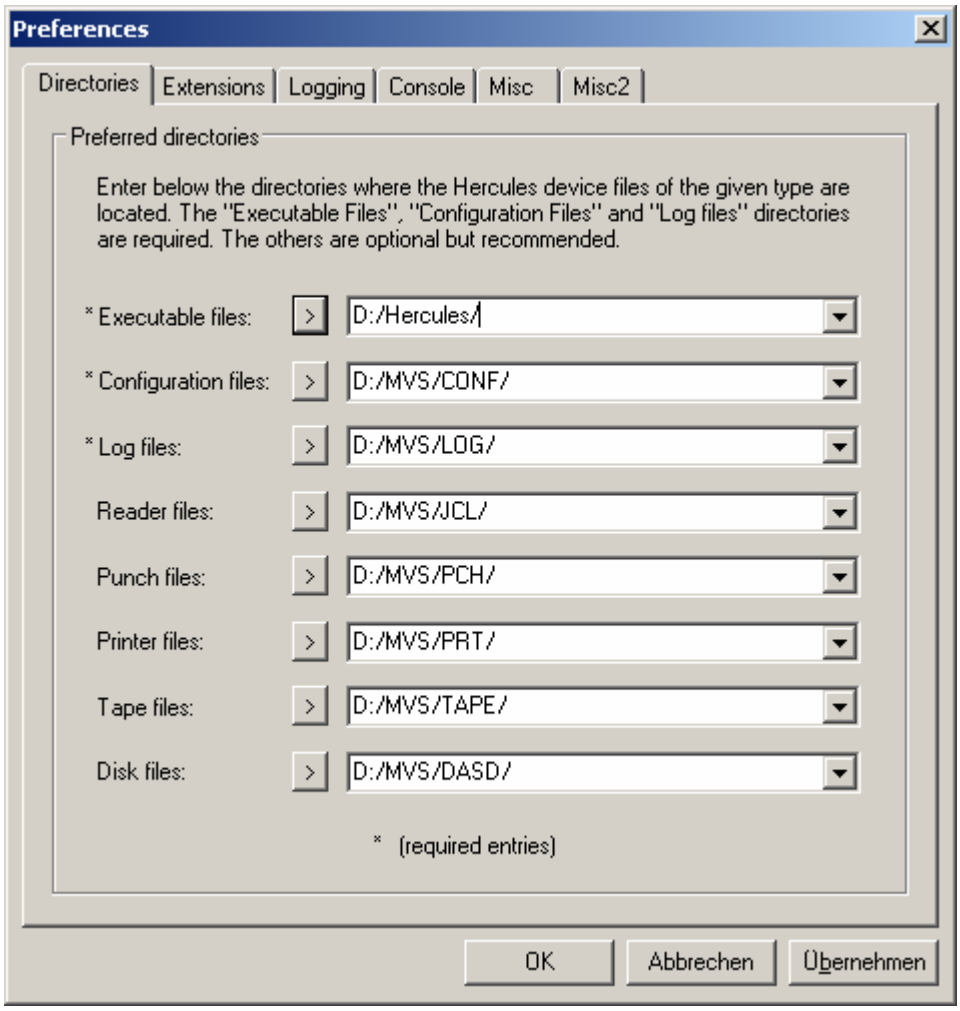


Figure 51: Preferences Directory Tab

The path for executable files, configuration files and log files are all required, all others are optional and provided for convenience. If specified, they are used as default directories by the various device configuration dialogs.

10.5.2 File Extensions

The File Extensions tab allows you to specify the preferred file extensions used in the "Files of type" dropdown list in all standard 'Open' and 'Save as' dialog boxes that the Hercules Windows GUI uses.

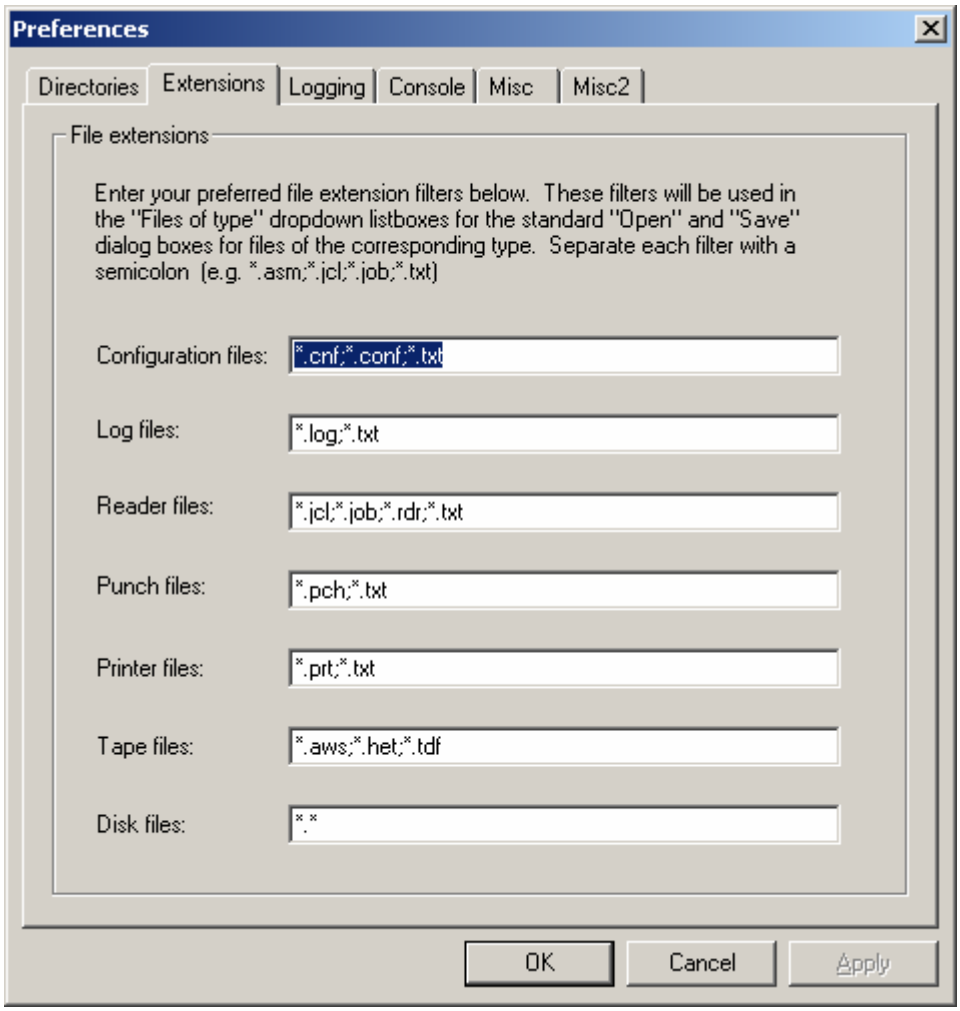


Figure 52: Preferences Extensions Tab

This feature allows each user to have different naming conventions for their reader, punch and disk files etc.

10.5.3 Logging

The Logging tab allows you to specify the preferred console logging options.

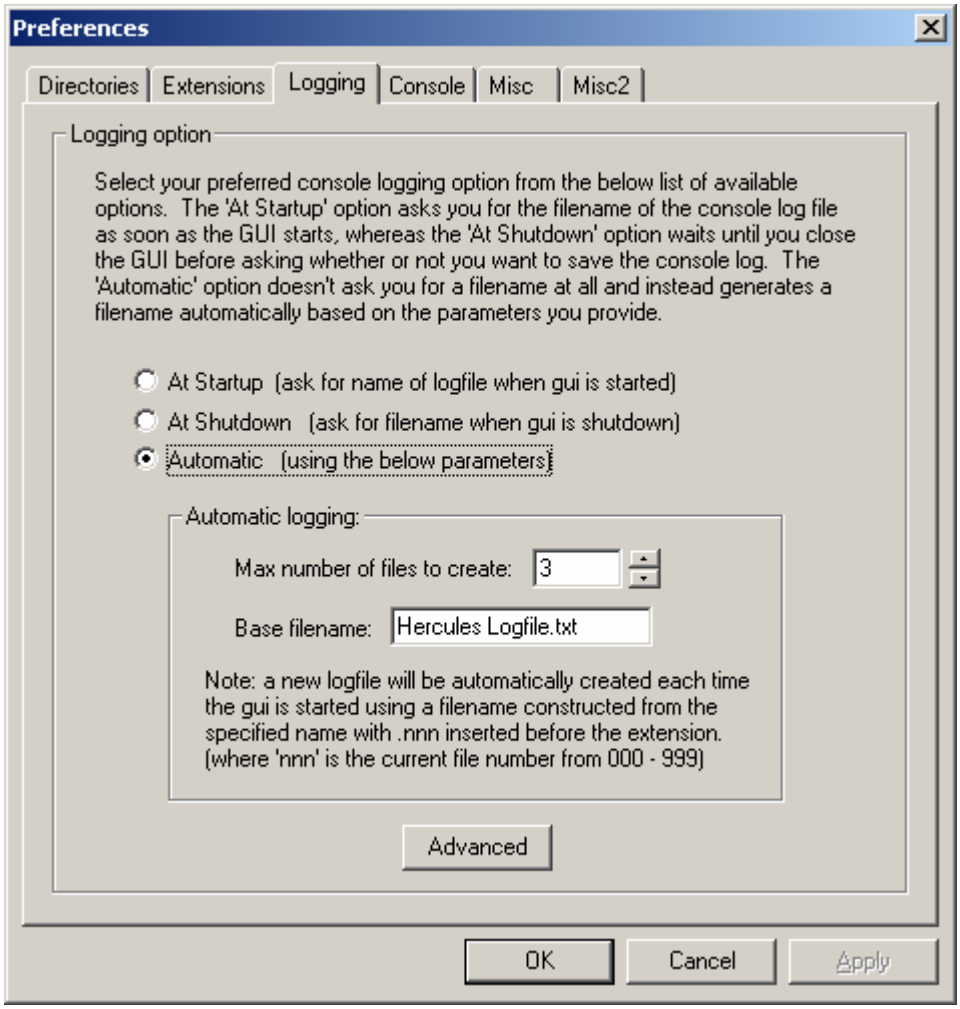


Figure 53: Preferences Logging Tab

The 'At Startup' option requests the filename of the console log file when the GUI is started. The 'At Shutdown' option asks if the console log file should be saved or not when the GUI is about to be closed.

The 'Automatic' option does not ask for a filename at all and instead generates a filename automatically based on the parameters provided. For example, if the base filename is "Hercules Logfile.txt" the generated log filename will be "Hercules Logfile.000.txt" the first time the GUI is started, then "Hercules Logfile.001.txt" for the next start of the GUI, etc.

Starting with version 1.4.0, once log files are created by whichever method, they are written to continuously as new messages arrive. This removes the need to do a periodic "Save Messages".

As new messages are now automatically written to the log file during Hercules execution, it is recommended to use the "Advanced Logging Options" dialog to specify the maximum log file size in number of lines. This is meant to prevent the log file from filling up your hard drive. Use the ‘Advanced’ button on the main Logging preferences tab to access these settings.

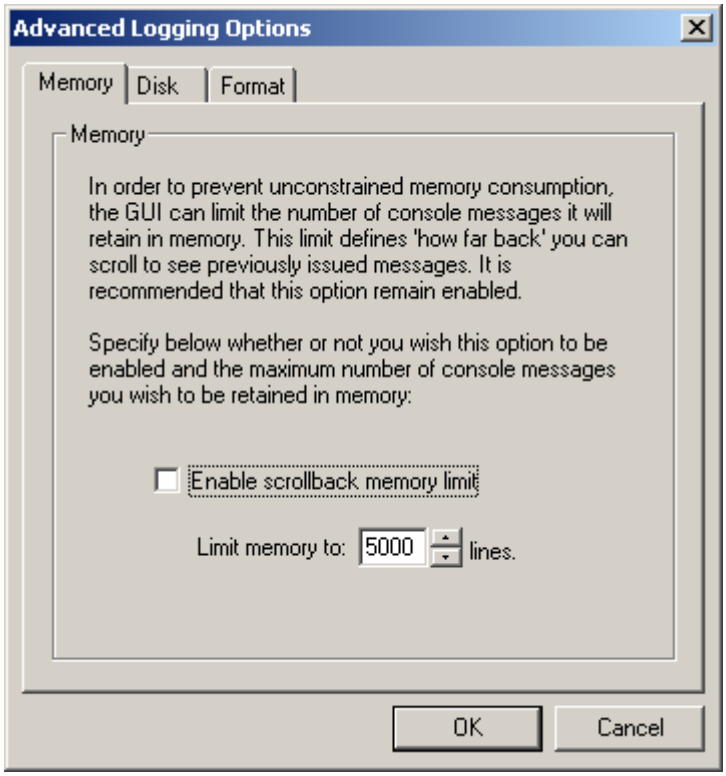


Figure 54: Advanced Logging Options Memory Tab

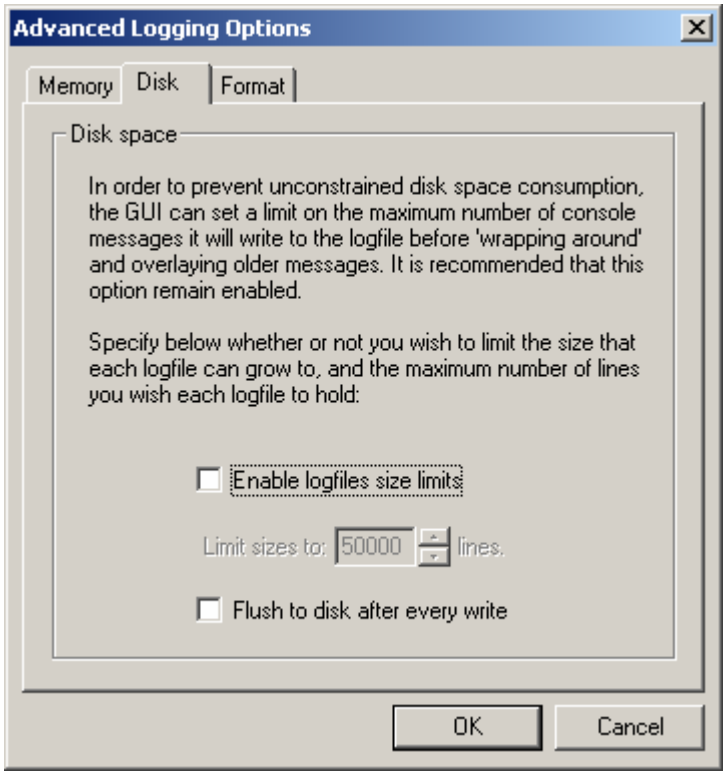


Figure 55: Advanced Logging Options Disk Tab

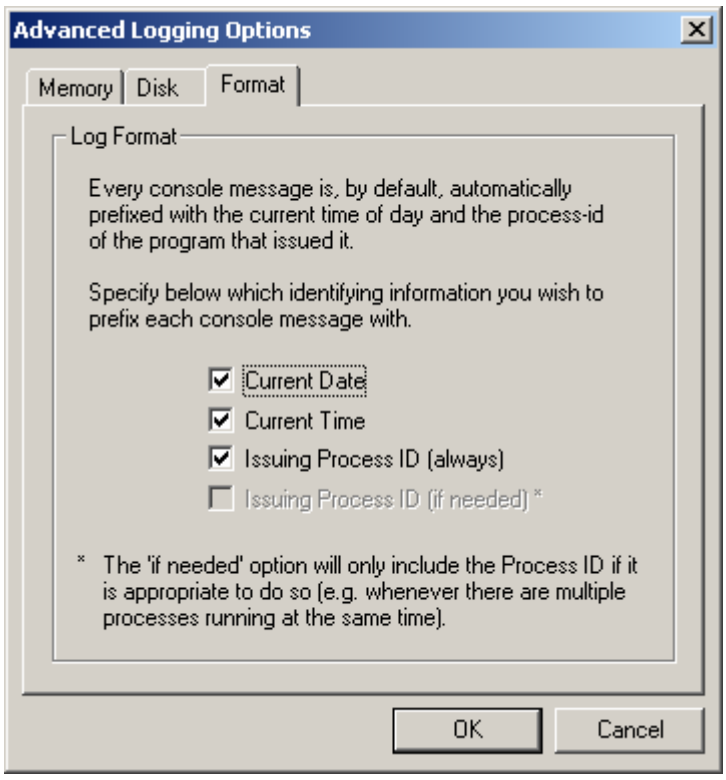


Figure 56: Advanced Logging Options Format Tab

The log file is a wrap-around file, where the oldest messages will be overwritten by newer messages when the specified size limit has been reached. Once the specified maximum number of lines has been written to the log file, the GUI will reset its file position pointer back to the beginning of the file and begin writing new log file messages over the top of older ones.

The file is not recreated when full. When it wraps around and begins to overlay older messages, the messages near the end of the log file are still present until eventually overwritten.

Note too that the Advanced Logging Options dialog lets you specify the maximum number of messages to be retained in memory. A limit is required to prevent run-away memory consumption. This memory limit is a separate value from the disk log file limit and essentially controls how far back you can scroll the console to see older messages.

10.5.4 Console

The Console Tab allows it to specify the preferred console font, font colour and background colour for the Hercules console message area.

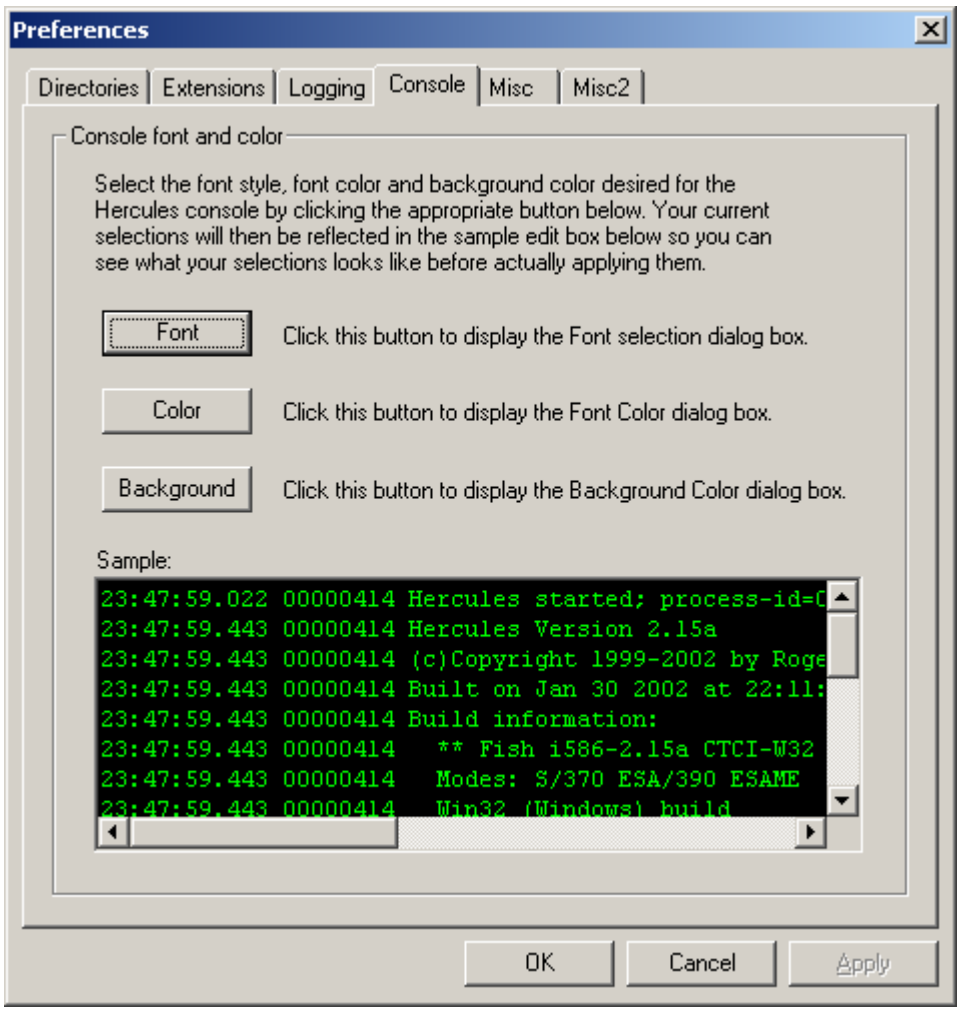


Figure 57: Preferences Console Tab

Simply click on the appropriate button and select the font or color. Your selection will then be reflected in the sample edit-box to let you see what the console would actually look like before your changes are applied.

10.5.5 Misc

The Misc Tab allows you to specify various miscellaneous preferences, such as how the GUI should react to your pressing the "Power Off" button.

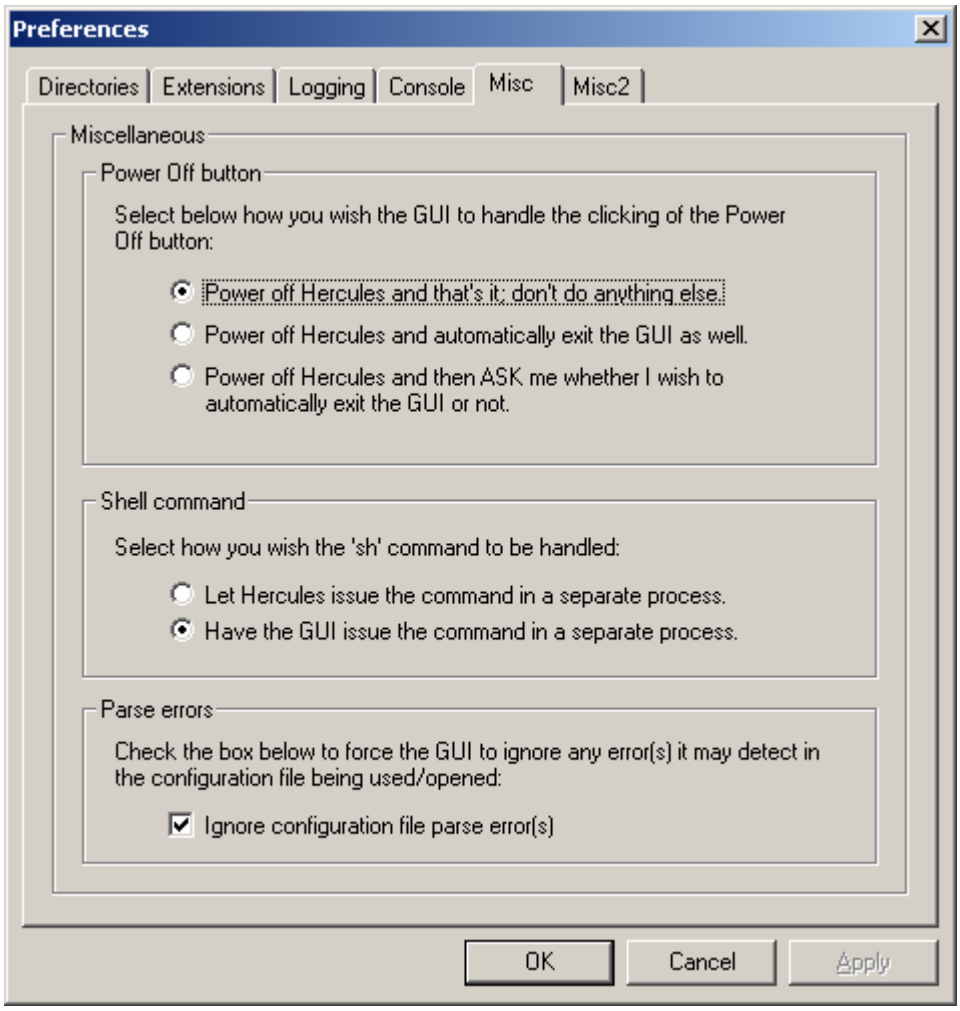


Figure 58: Preferences Misc Tab

When shell command support was added to Hercules for the Windows environment a bug was revealed in the manner that Cygwin processed the 'fork' command. Although a workaround was developed the GUI was also modified to be able to issue the shell commands directly instead.

In order for the GUI to be able to issue shell commands instead of Hercules itself a new program called 'conspawn.exe' was developed. The GUI passes the shell command to conspawn for execution. The conspawn program is included as part of the GUI package and must reside in the same directory as the other Hercules executables.

The Ignore Parse Errors option was created to allow you to open and use a Hercules control file that the GUI would otherwise fail to parse properly for whatever reason.

When a control file is opened the GUI parses the statements and saves the information in an internal control area. This information is used later to complete various fields in the dialogs such as the device configuration dialog. If the GUI cannot properly parse a given control file statement it throws a parse error and prevents you from opening or using what it considers to be a bad control file. This option can be used to bypass this error, for example when using a device statement containing a new parameter that Hercules understands but the GUI may not yet support.

The "Ignore Parse Errors" option instructs the GUI to ignore the parse error and open and use the control file anyway. When this option is set the GUI ignores all parse errors and will always successfully open whatever control file you instruct it to. As the GUI can update the Hercules control file, it is highly recommended that you leave this option disabled and enable it only when needed to avoid accidentally damaging a file that may not contain valid Hercules control specifications.

If you do need to enable this option, remember to disable it when no longer needed as once enabled it will stay that way until you purposely disable it. Like all preference options it is persistent across executions of the GUI.

10.5.6 Misc2

The Misc2 Tab provides additional options that would not fit on the original Misc options tab.

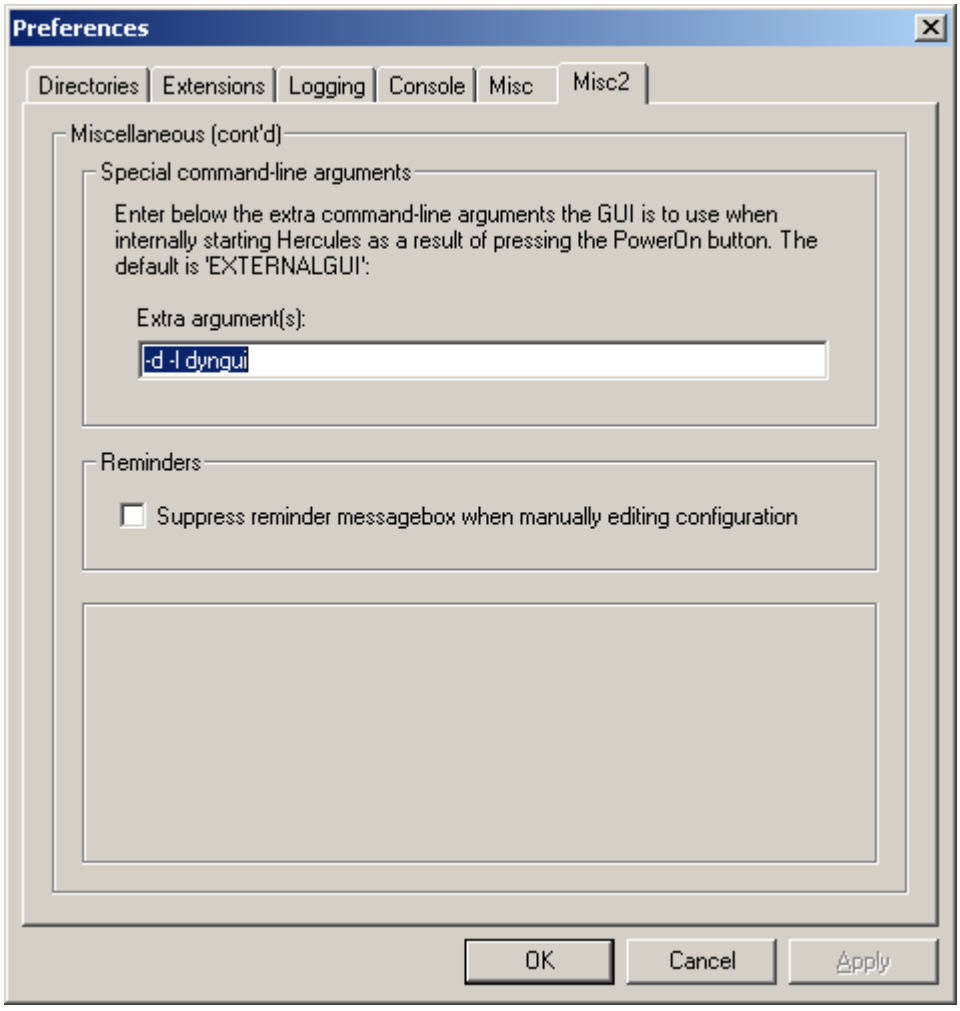


Figure 59: Preferences Misc2 Tab

Since version 3.0 of Hercules support has been added for command-line options beyond the existing -f (control file) option. Here you can enter any of the new command-line options you may wish to have issued whenever the GUI starts up ("Powers On") the Hercules emulator.

Note that you must not specify the -f option here. The GUI constructs the -f option automatically based on the configuration file previously defined in the Preferences dialog. You may enter any other command-line options in addition to the required ones - see the important note immediately below.

You must specify here either "EXTERNALGUI" (without quotes) or – starting with Hercules Version 3.0 – a new command line option "-d -l dyngui". This option must not be left blank. If you accidentally leave this option blank it is possible that both Hercules and the GUI will not operate at all and highly likely that they will not operate as expected.

This is an advanced option and should not be modified by end users. Please only change this if instructed to by Hercules Technical Support or Hercules developers.

10.6 System Configuration

When a Hercules control file is opened by the GUI it is parsed and then a "System Configuration" dialog is displayed showing the various configuration settings.

Since HercGUI version 1.9.5 the System Configuration dialog information is displayed in three separate property pages: the Architecture page, the O/S Tailor settings page and the Other / Misc page.

The Identification section simply displays the full pathname of the configuration file that was opened and provides an input field for a text description to be associated with this particular configuration file. This is in case you have multiple system configurations where it is useful to have some indication that you are modifying the correct one. The description you enter here is saved as a comment at the beginning of the configuration file.

10.6.1 Architecture Settings

This page of the System Configuration dialog is where you define various hardware and architectural settings, e.g. how many emulated CPUs, how much emulated main storage etc.

The Architecture radio buttons allow you to define the default architectural mode your emulated CPUs will be initially started in. For emulated z/Architecture machines the actual IPL takes place in ESA/390 mode and it is the responsibility of the IPL'ed operating system to switch the CPU(s) into z/Architecture mode when it is ready to use this mode. It does this using the SIGP instruction.

The following figure shows the Architecture Settings tab.

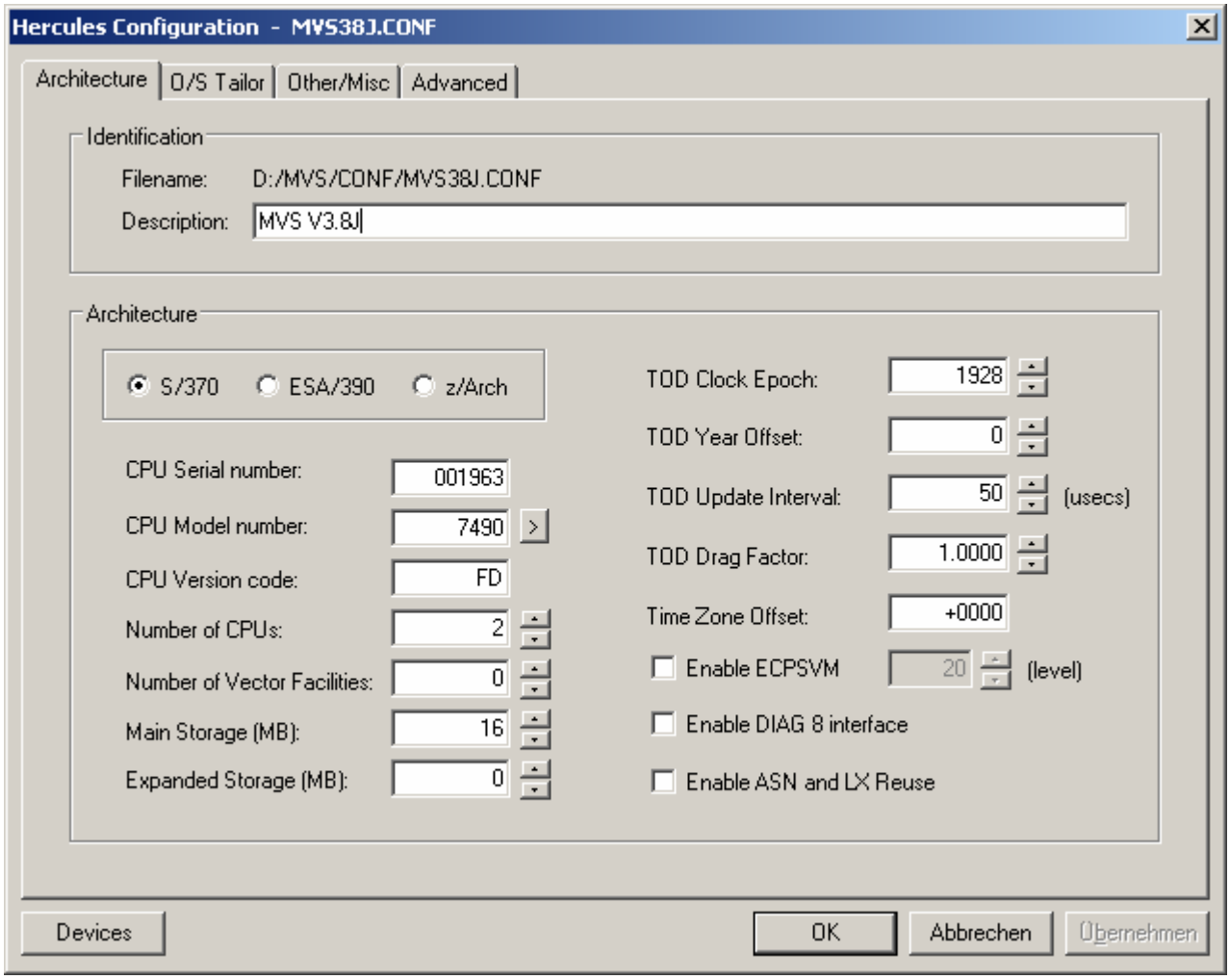


Figure 60: Architecture Settings Tab

In the main System Configuration dialog, if the GUI finds a specially formatted file called "cpu-types.txt" in your preferred configuration files directory, a small '>' button will appear next to the "CPU Model number" edit-box. This file may contain a list of the various CPU model numbers, their names and their corresponding STIDP (Store CPU ID instruction) values.

When the System Configuration dialog is initialized, if the GUI finds this file, it will parse the entries and use them to construct a drop-down box that will be displayed when you click the '>' button. This allows you to easily select the desired CPU Model you wish your virtual mainframe to be reported as.

Neither the GUI nor the Hercules Emulator itself makes any attempt to try and emulate all aspects or features of a given CPU model. This CPU model number simply specifies what value to use in the STIDP (Store CPU ID) instruction.

When your CPU Model is selected in this way the GUI automatically fills in the "CPU Model number" edit-box field with the corresponding value it finds in the "cpu-types" file. You have the option of manually overriding this value. A sample "cpu-types" text file is included with the distribution of the Hercules GUI.

10.6.2 O/S Tailor Settings

The O/S Tailor settings page is where you can establish certain settings related to the type of guest operating system you intend to actually run on your virtual mainframe. The purpose of the O/S Tailor radio buttons is to limit the amount of Hercules generated message traffic by selectively suppressing certain program check and trace type messages which are considered normal for the specified operating system.

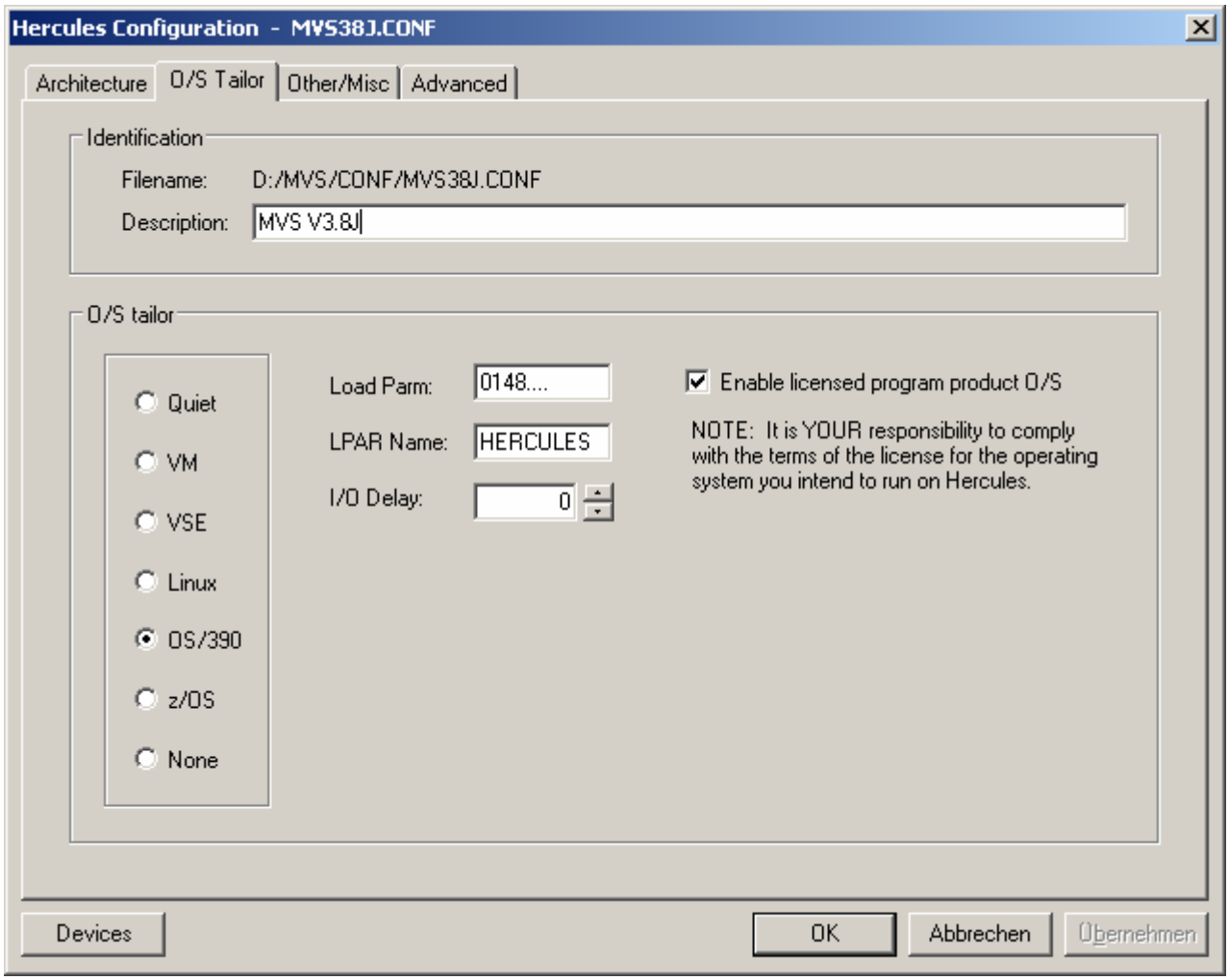


Figure 61: O/S Tailor Settings Tab

When the "Enable licensed program product O/S" option is specified for a given control file any attempt to power on Hercules using that control file will result in a dialog box being displayed that asks you to verify your true intentions.

Figure 62: PGMPRDOS LICENSED Acknowledgment

As soon as this dialog is displayed a 10-minute timer is started. If you respond before the timer expires your response is accepted as-is. If your response is "No, I do NOT honestly wish to run in unrestricted mode" then your virtual mainframe will not be powered on. If your response is "Yes" - whether explicit or presumed (see next paragraph) - then Hercules will be powered on.

Please note that if you fail to respond within the 10-minute time limit your response is presumed to be "Yes, I honestly DO wish to run in unrestricted mode". If you do not wish to wait the entire 10 minutes then you will have to respond to the dialog manually yourself. There is no way to disable or override this feature.

This dialog is displayed each time you attempt to power on Hercules during a given Hercules GUI session when using a configuration file with the "Enable licensed program product O/S" option checked. Although you only have to respond to this message once during a single session if you continue to use the same configuration file. If you switch to a different configuration file and then return later to the first one (or exit the GUI entirely and start it again), you will be asked once again to confirm your intent.

10.6.3 Other / Misc Settings

The "Other / Misc" window allows you to define more system configuration values, mostly related the internal functioning of the Hercules emulator. Please refer to the documentation for the Hercules emulator itself for more information regarding the various values that may be specified here.

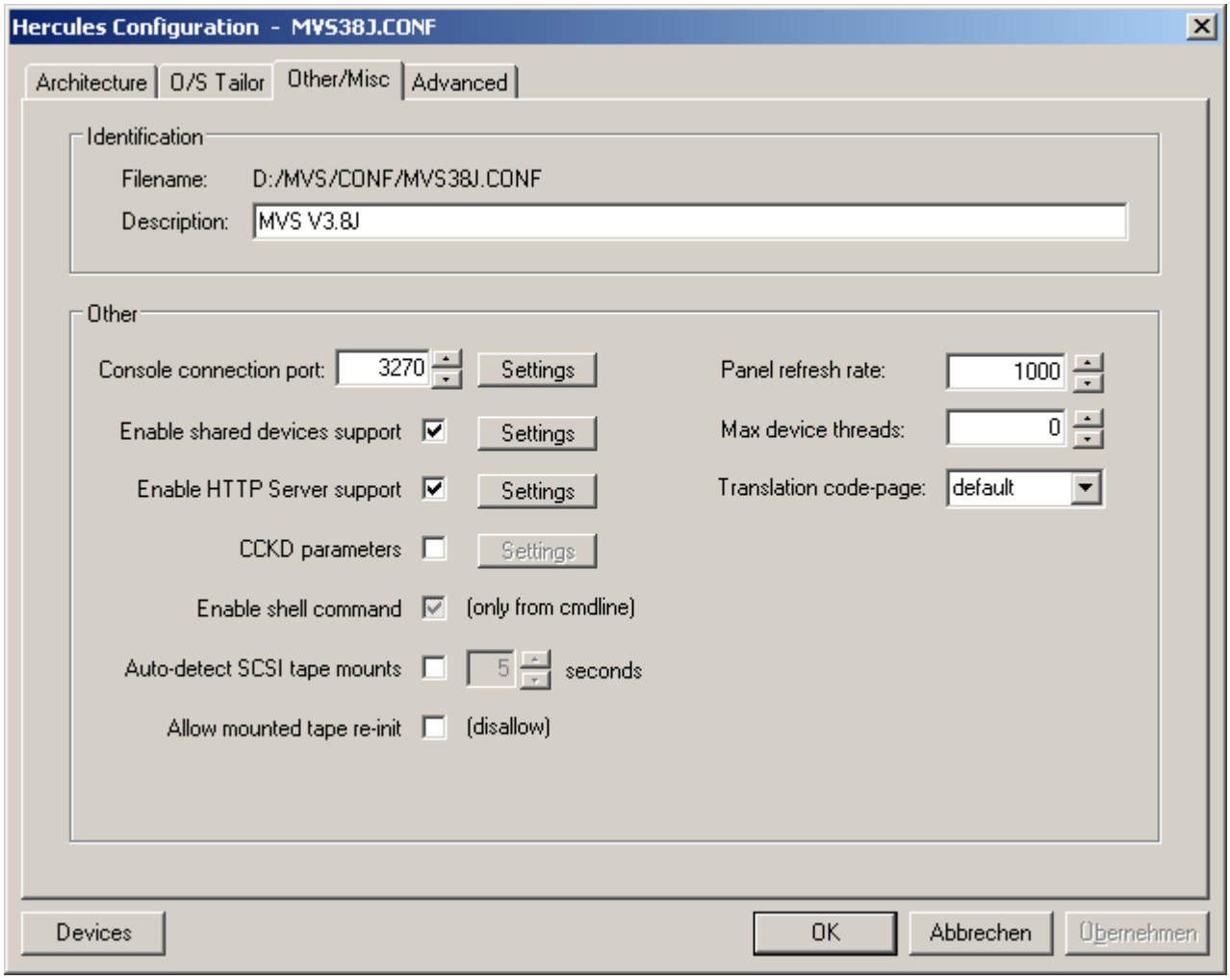


Figure 63: Other / Misc Tab

Hercules's HTTP Server support allows you to control Hercules via any standards compliant web browser. This dialog allows you to define the HTTP Server parameters that Hercules is to use.

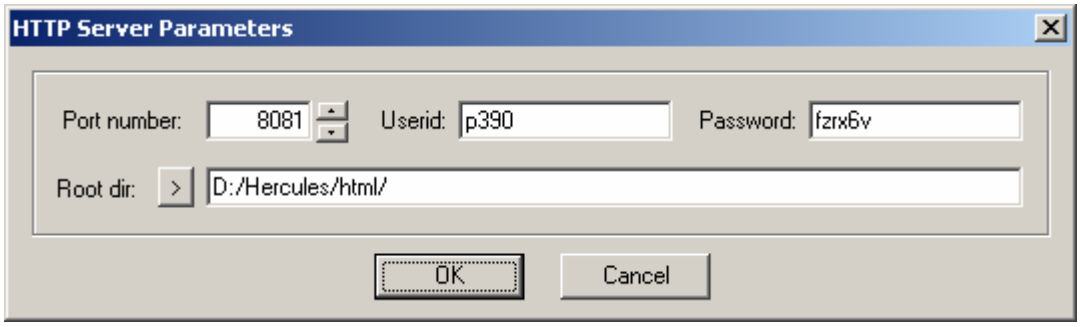


Figure 64: HTTP Server Parameters

Enter the port number for Hercules's HTTP Server to listen on and if desired specify the authentication criteria needed to connect to the server. You can also enter the root directory from which the web pages will be served.

If you enter a userid for authentication then you must also enter a password. If no userid and password are entered then anyone with a browser that is able to connect to your Windows host system will be able to control your Hercules system via the HTTP Server interface. The password you enter is not encrypted in any way and is stored in your Hercules control file, as well as passed through the network, in unencrypted plain text format. You should therefore take whatever steps are required to secure Hercules control file(s) that contain HTTP Server passwords.

Since Hercules version 2.17 the behavior Compressed CKD DASD (CCKD) functionality is controlled via the setting of certain global parameters. CCKD functionality is no longer adjustable on an individual device-by-device basis.

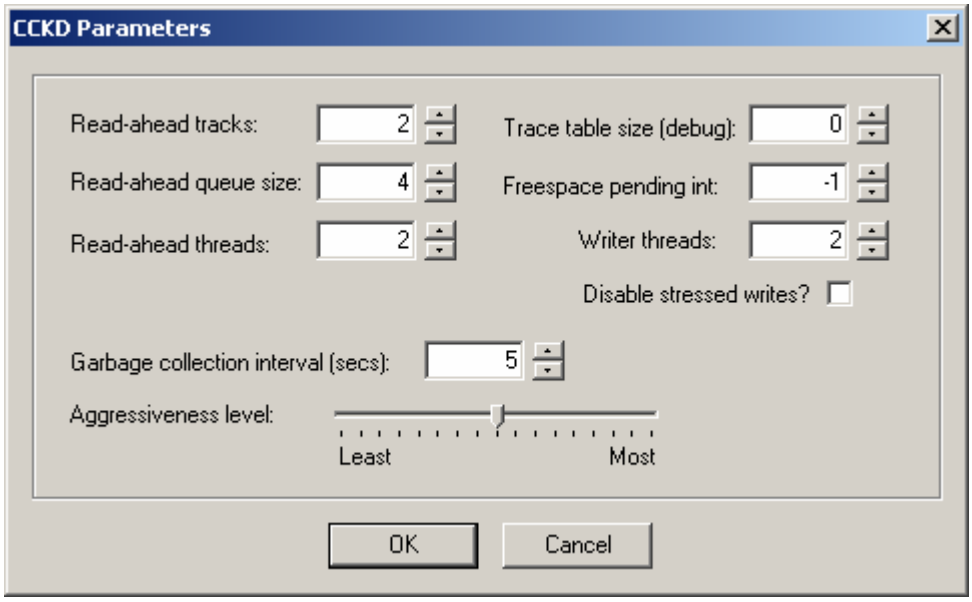


Figure 65: CCKD Parameters

10.6.4 Advanced Settings

The Advanced configuration page is where settings for features that are intended only for more advanced users may be made. If you have a custom dynamic module (DLL) you wish Hercules to use or wish to modify Hercules's default priority settings you would do that here.

Please see the “Hercules User Reference Guide” for more details on the options presented here.

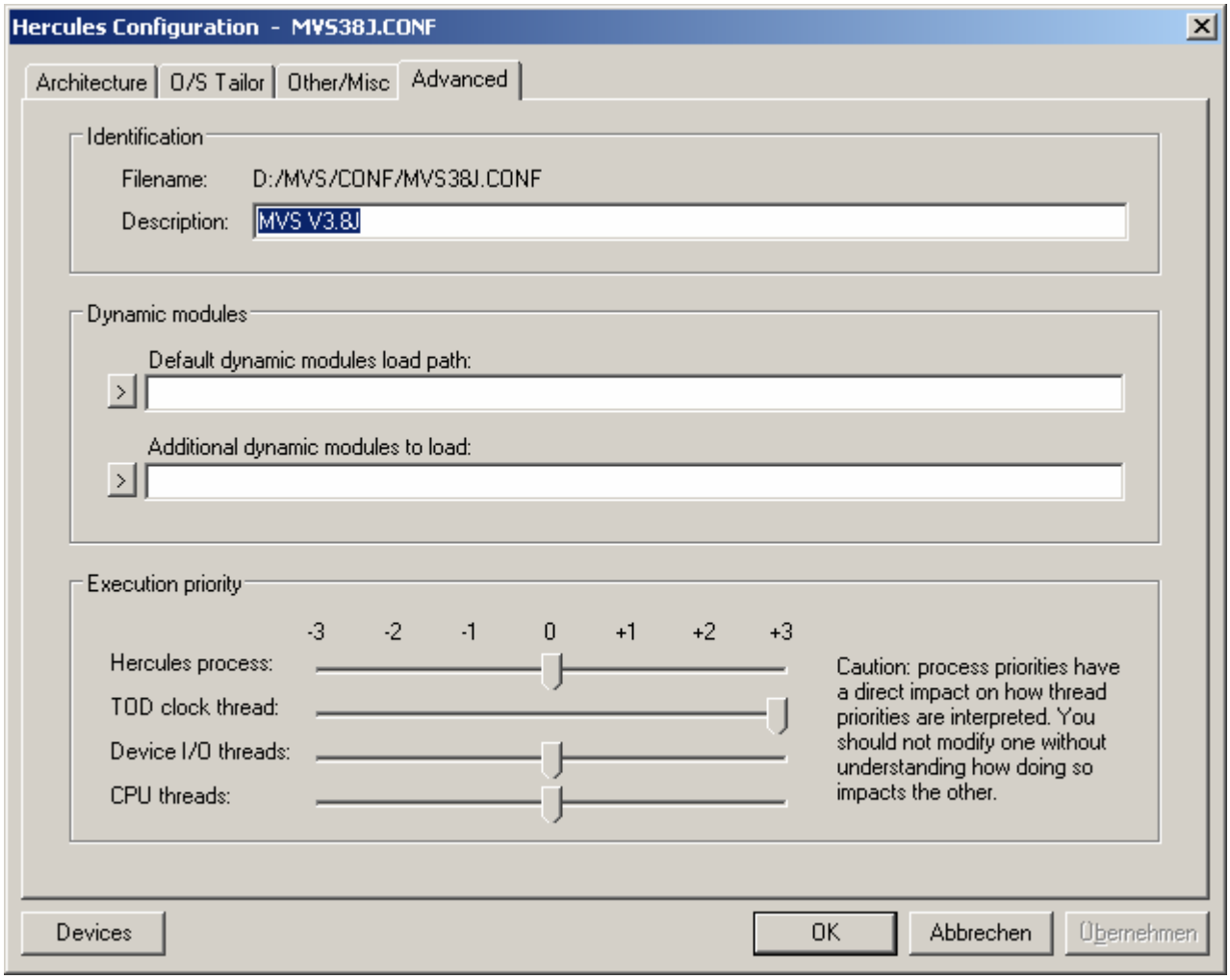


Figure 66: Advanced Tab

10.7 Device Settings

Clicking the "Devices" button from the System Configuration dialog or selecting "Modify Devices" from the File menu will take you to the Device Configuration dialog. From here you can add, delete or modify the devices in the current configuration.

This particular dialog is resizable as device configuration statements can be quite long.

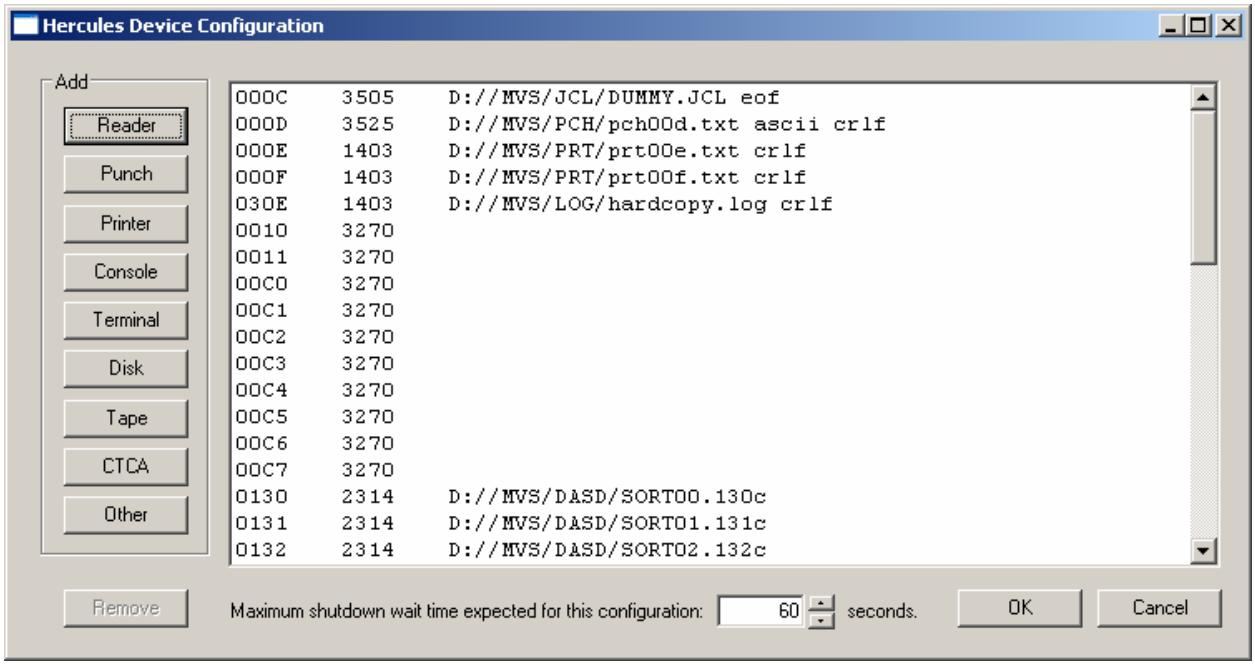


Figure 67: Device Configuration

The *"Maximum shutdown wait time expected for this configuration"* setting defines a time limit to the Hercules GUI. This limit is the amount of time the GUI is to expect between when the 'quit' command is issued (or the "Power Off" button is pressed) and when Hercules finally finishes exiting after completing its shutdown sequence.

When you use compressed disks (CCKD) Hercules needs time to write-back cached copies of track images and adjust the free space for each disk before it can safely exit. If the expected wait time is exceeded the GUI issues a warning asking whether to continue waiting or forcibly terminate the Hercules Emulator process. If many compressed disks are frequently updated it can take over a minute to write all cached data to disk.

It is safe to specify a value for this setting that is high enough for all likely cases in your environment. The Hercules GUI will terminate as soon as Hercules itself ends regardless of the wait time setting.

Right clicking on a device statement presents a context menu from which you can select 'Edit' or 'Properties'. Selecting 'Properties' presents the "Reinitialize Device" dialog, also displayed by double-clicking the device statement.

When 'Edit' is selected from the right-click context menu you are presented with a simple device statement edit dialog. From here you can directly modify raw device statements.

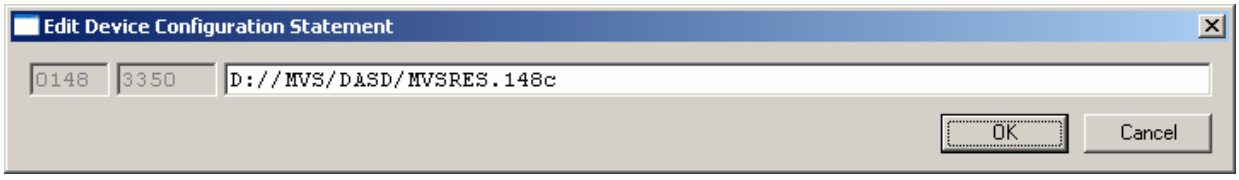


Figure 68: Edit Device Configuration Statement

To add a new device, click on one of the 'Add' buttons:

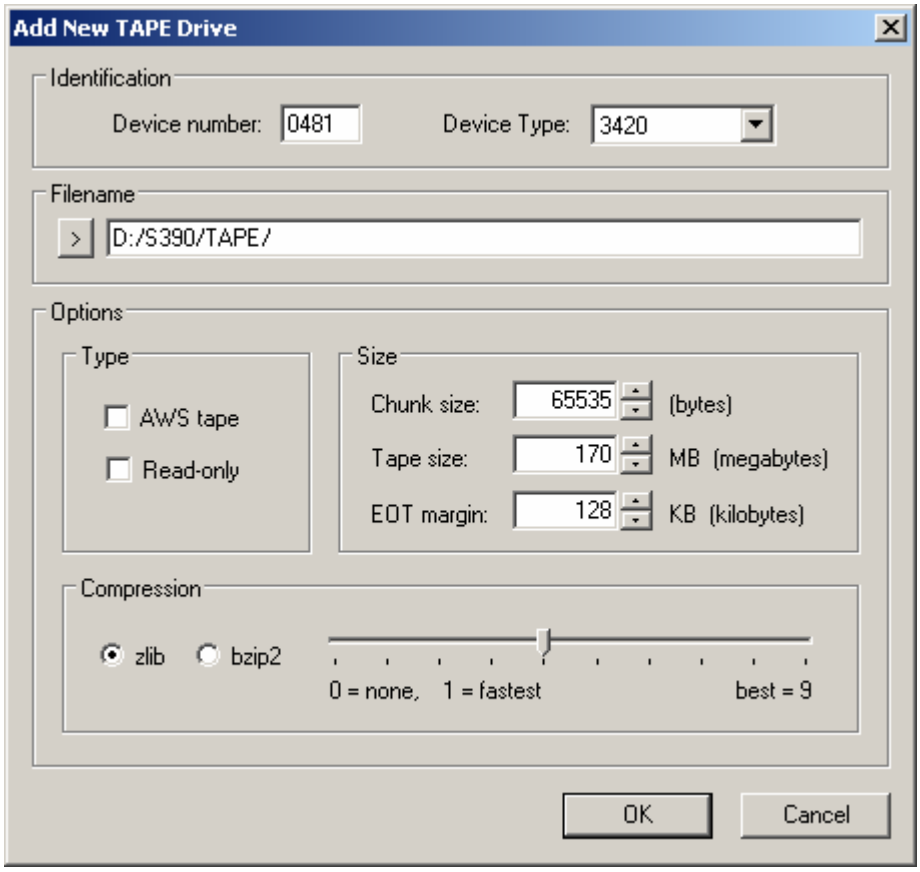


Figure 69: Add New Device

To delete a device select the desired device first to highlight it and then click on the 'Remove' button. To modify an existing device double-click on the entry for the desired device:

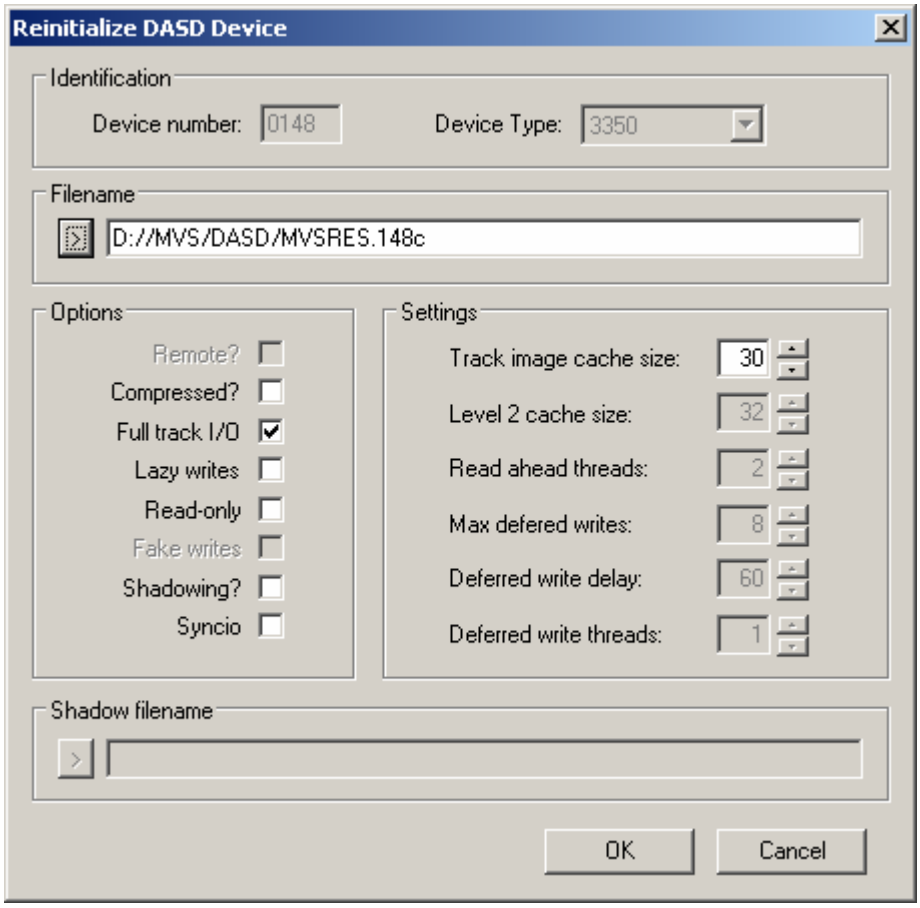


Figure 70: Reinitialize Device

Note: If you used any of the new CCKD parameters on the main System Configuration dialog, some controls in the above dialog will not be displayed. This occurs because the new System Configuration CCKD parameters modify CCKD functionality on a system-wide basis and remove the ability to specify these parameters on per-device basis.

10.8 Display / Alter Memory

The "Display / Alter Memory" item in the command menu allows you to display or modify absolute main storage.

It is very important that you keep in mind that when you alter absolute main storage via this dialog then neither the storage keys nor the CPU instruction and data caches are updated in the Hercules emulator itself. Instead the memory of the Hercules emulated operating system is directly modified without the Hercules Emulator knowing of this.

Please use this feature with caution. It is designed for examining and searching main storage for emulator debugging purposes and not as a safe means of modifying hosted operating system storage.

The following figure shows the "Display / Alter Memory" dialog.

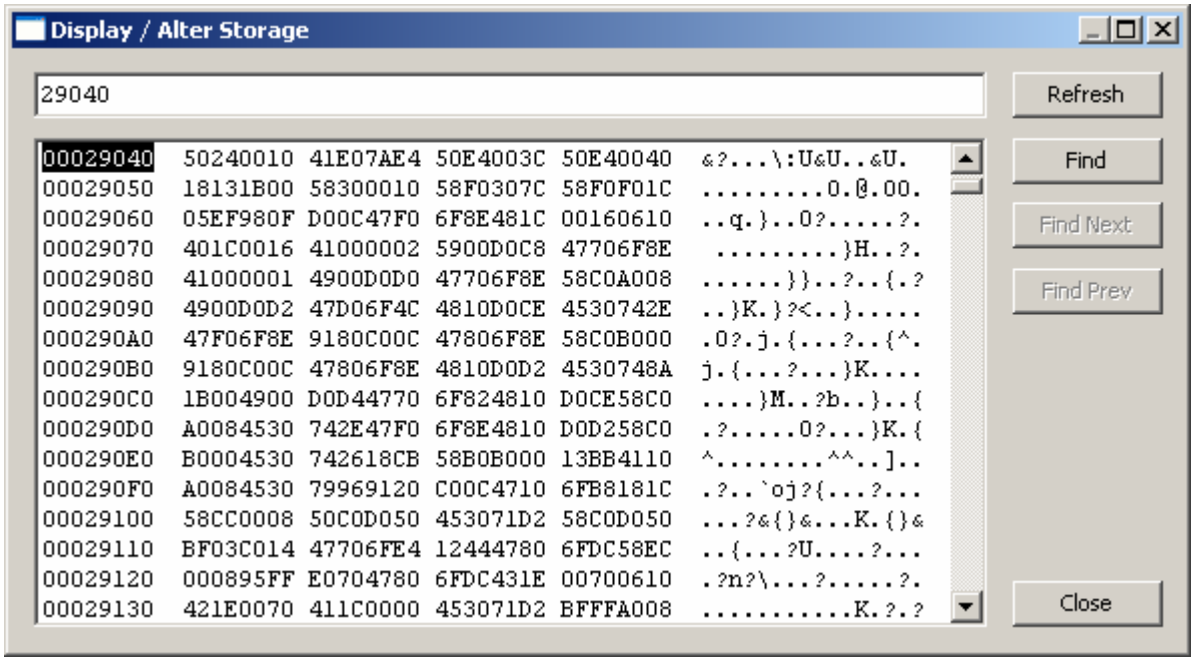


Figure 71: Display / Alter Memory Dialog

If you need to modify real or virtual storage it is highly recommended that you use the Hercules Emulator 'r' and 'v' panel commands. These ensure that the hardware emulator is aware of any changes.

10.9 Load Card Reader, Load Tape, Unload Tape

These menu items provide a quick and easy way to do just as their descriptions suggest. The 'Load Reader' command is provides an interface to submit jobs to the system. It displays the "Re-Initialize Device" dialog for the card reader. From here you can use standard Windows 'Open File' dialogs to browse for a file that you want to submit. The selected file will be loaded into the card reader. Clicking OK issues the appropriate Hercules devinit panel command.

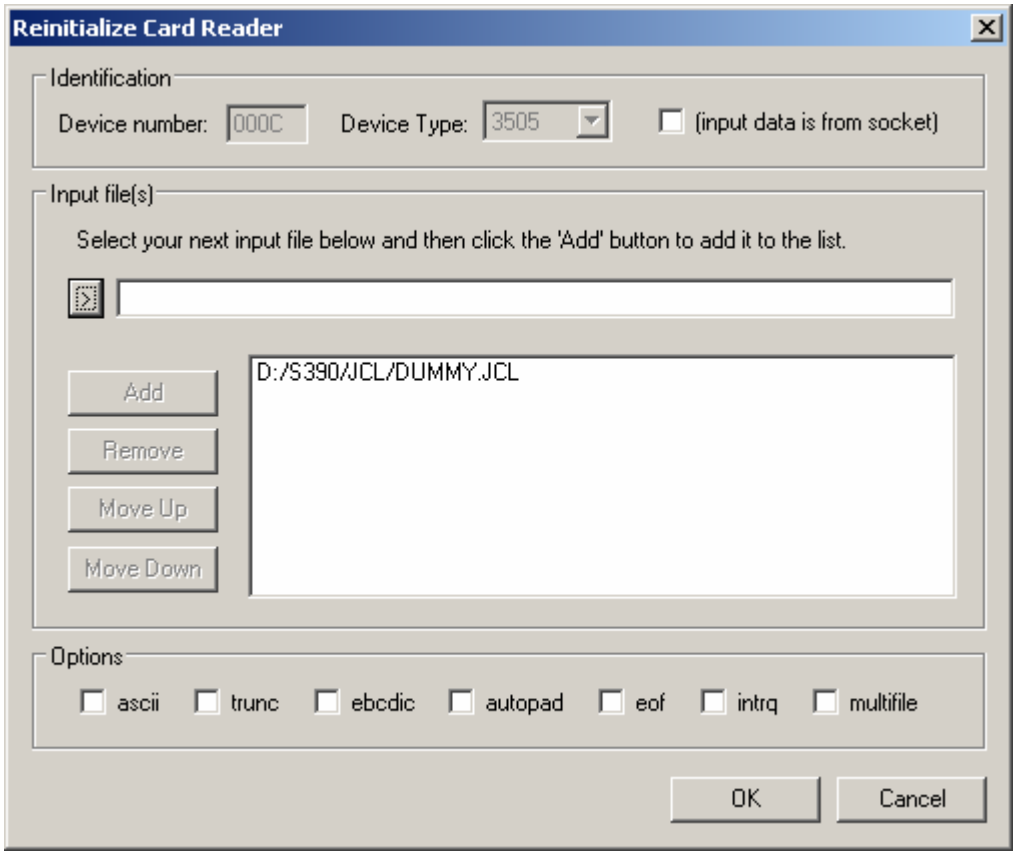


Figure 72: Reinitialize Card Reader Dialog

The "input data is from socket" option is found to the right of the device type field. It allows you to tell Hercules to obtain card reader input from a specified socket instead of a disk file as by default. This allows you to submit card decks remotely using a simple utility that connects to the specified socket and writes card images directly to Hercules. Recent releases of the GUI provide a DOS program "HercRdr" to support this capability. For more information on the HercRdr utility and the 'sockdev' option refer to the Hercules User Reference Guide.

10.10 Device List Bar

The "Device List" bar, similarly to its non-GUI console mode counterpart, lists the devices in the current configuration and their status.

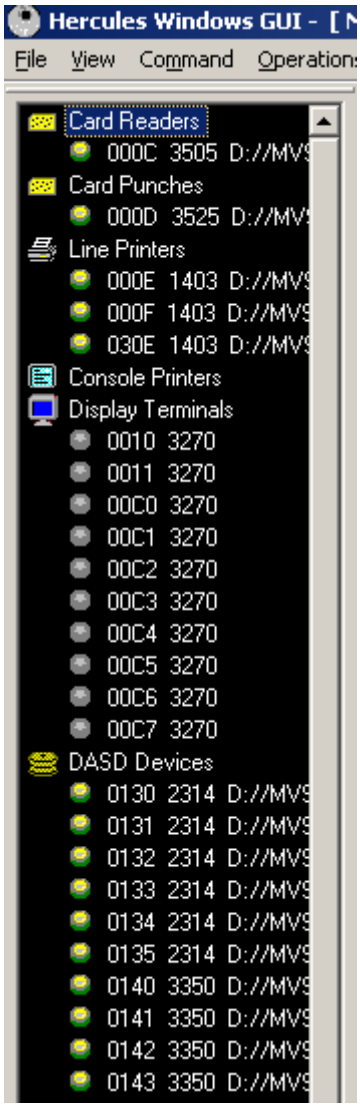


Figure 73: Device List Bar

A grey diode indicates the device is offline and not open. A green diode indicates it is online or open. The green diode changes to yellow whenever the device is busy and changes to red when there is an interrupt pending for the device.

As the hosted system runs and performs I/O to the devices in your configuration you will see the diodes change between yellow, red and green. This shows that there is I/O activity taking place on the device.

Devices are displayed in a tree-list with a branch for each class of device. Right-clicking the Devices within a branch presents a context sensitive menu. You can also right-click each branch.

To add a new disk drive to your configuration right-click on the "DASD Devices" branch and select 'Add device' from the menu that appears. To delete ("detach"), rename ("define"), reinitialize ("devinit") or present an attention interrupt ("i") for a particular device, right-click the device and select the appropriate option from the context menu presented.

10.11 Utilities Menu

All of the Hercules utility programs can be run by completing the appropriate dialog. Both Hercules and these utilities run as separate processes, so it is possible to run more than one utility at the same time as Hercules images.

A progress dialog is displayed as each utility runs, all messages generated by the utility are displayed on the GUI console just as Hercules messages are. Each message is prefixed with its process ID to differentiate between utility program messages and a timestamp.

The following figures show an example of the DASDINIT utility window.

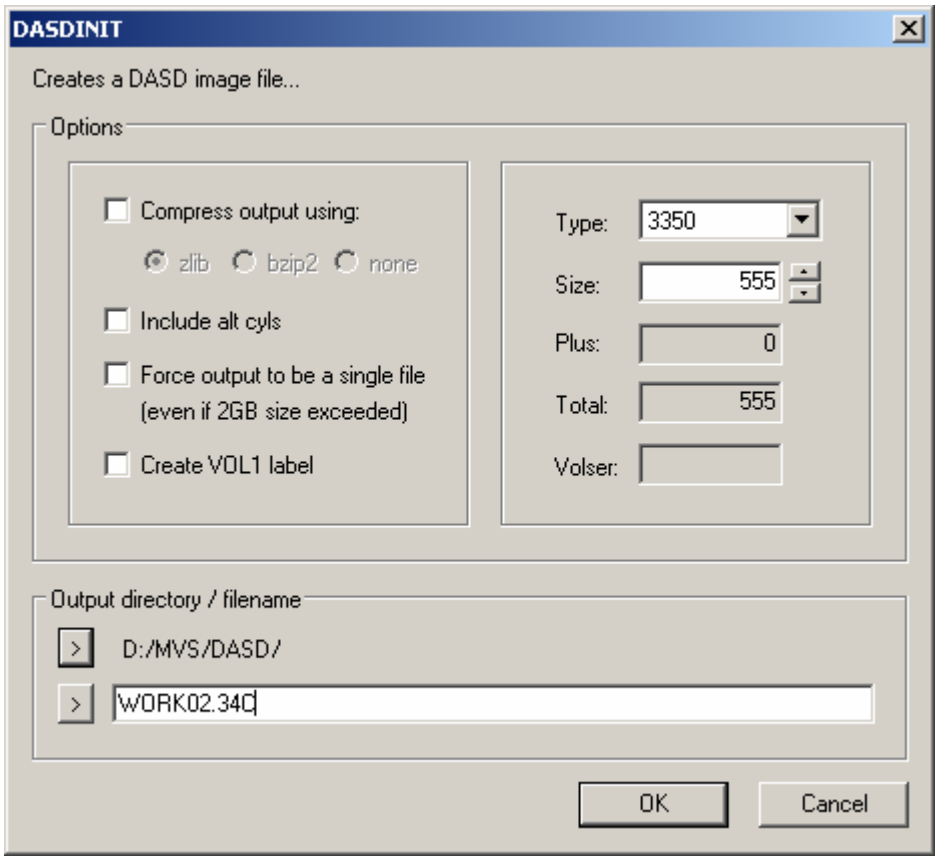


Figure 74: DASDINIT Utility Window

10.12 Registry Tweaks

Some of Hercules minimum, maximum and default values used by the GUI are stored in the windows registry. The following table shows the valid values.

Apart from the "MinTODDrag" and "MaxTODDrag" which are string values, all of the values below are DWORD values and are stored in the "Limits" branch of the main Hercules Windows GUI registry key:

HKEY_CURRENT_USER/Software/Software Development Laboratories/Hercules/Limits

Note that changing the values of these registry entries will not necessarily change the actual function of the Hercules Emulator. If the GUI accepts the new value this does not necessarily mean that Hercules itself will accept the value.

Name	Default	Description
MaxCPUs	32	Maximum allowable number of central processors
MaxVectors	4	Maximum allowable number of vector facilities
MaxMainMem	2048	Maximum allowable amount of central storage (MB)
MaxExpandedMem	1024	Maximum allowable amount of expanded storage (MB)
MinEpoch	1801	Minimum allowable clock epoch year
MaxEpoch	2099	Maximum allowable clock epoch year
MinPanRate	10	Minimum allowable panel refresh rate (milliseconds)
MaxPanRate	5000	Maximum allowable panel refresh rate (milliseconds)
MaxTapeSizeMB	2048	Maximum allowable emulated tape size (MB)
MaxEOTMarginKB	2048	Maximum allowable emulated tape 'end-of-tape warning area' margin-size (KB)
MaxECPSVMLevel	99	Maximum allowable ECPSVM value
DefECPSVMLevel	20	Default ECPSVM value
MinTODDrag	0.0001	Minimum allowable TOD clock drag factor
MaxTODDrag	10000	Maximum allowable TOD clock drag factor
MinCCKDgcparm	-8	Minimum allowable CCKD parameters garbage- collection aggressiveness level
MaxCCKDgcparm	+8	Maximum allowable CCKD parameters garbage- collection aggressiveness level

DefCCKDgcparm	0	Default CCKD parameters garbage- collection aggressiveness level
MaxCCKDfreepend	4	Maximum allowable CCKD freespace pending interval value
DefCCKDfreepend	-1	Default CCKD freespace pending interval value
MaxCCKDrat	16	Maximum allowable CCKD read-ahead tracks
DefCCKDrat	2	Default CCKD read-ahead tracks
MaxCCKDraq	16	Maximum allowable CCKD read-ahead queue size
DefCKDraq	4	Default CCKD read-ahead queue size
MaxCCKDra	9	Maximum allowable CCKD read-ahead threads
DefCCKDra	2	Default CCKD read-ahead threads
MaxCCKDwr	9	Maximum allowable CCKD writer threads
DefCCKDwr	2	Default CCKD writer threads
MaxCCKDgcint	60	Maximum allowable CCKD garbage-collection interval (seconds)
DefCCKDgcint	5	Default CCKD garbage-collection interval (seconds)
MaxCCKDtrace	200000	Maximum allowable CCKD trace table size (number of entries)
DefCCKDtrace	0	Default CCKD trace table size (number of entries)
MaxCCKDcache	64	Maximum allowable CCKD cache size (MB)
MaxCCKDl2cache	2048	Maximum allowable CCKD level-2 cache size (MB)
MeterThreadRate	1000	CPU percent utilization meter update interval (milliseconds)
NagleThreadNagleRate	75	Maximum 'continue buffering' (to prevent screen refresh) message reception rate (milliseconds)
NagleThreadMaxNagleRate	375	Maximum allowable delay before displaying buffered messages (milliseconds)
MinCaptureBuffsize	64	Minimum allowable TunTap32 WinPcap device driver capture buffer size (KB)
MaxCaptureBuffsize	16384	Maximum allowable TunTap32 WinPcap device driver capture buffer size (KB)
DefCaptureBuffsize	1024	Default TunTap32 WinPcap device driver capture buffer size (KB)

MinPacketBuffsize	16	Minimum allowable TunTap32 DLL I/O buffer size (KB)
MaxPacketBuffsize	1024	Maximum allowable TunTap32 DLL I/O buffer size (KB)
DefPacketBuffsize	64	Default TunTap32 DLL I/O buffer size (KB)

Table 23: Hercules Windows GUI Registry Keys

11. Installation of CTCI-W32

11.1 Downloading the Binaries

The CTCI-W32 components can be downloaded from http://www.softdevlabs.com. Two packages are required, CTCI-W32 and FishLib.

CTCI-W32 consists of FishPack.dll, TunTap32.dll and TT32Test.exe packaged together as one product. The FishLib package is required with all recent versions of Fish's software. It contains common routines used throughout the packages.

Note: Beginning with release 3.2.1.160 of CTCI-W32 additional DLLs are required. These are Microsoft MFC and VC Runtime DLLs that you can download from the address mentioned above. The installation takes a few seconds and does not require a reboot. There are a 32-bit and a 64-bit version of these DLLs. Please ensure you are using the correct one according to the product you are installing (32-bit or 64-bit version of the Windows GUI).

If you previously installed the Hercules Windows GUI as described earlier in this manual then these DLLs are already present on your system. Note that you only need to install these C Runtime DLLs once even if new versions of CTCI-W32 are subsequently installed.

11.2 Installation Steps

The installation of the CTCI-W32 packages is straightforward. Unzip the executables and DLLs from the downloaded ZIP files into the same directory as the Hercules executables and the installation is complete.

11.3 Customization Steps

The customization of the CTCI-W32 consists of three major steps described further below:

- Configuring Windows Networking
- Configuring Hercules
- Configuring the Guest Operating Systems TCP/IP Settings

For details on how CTCI-W32 works internally see the Hercules "General Information" manual.

11.3.1 Configuring Windows Networking

Only minor configuration of Windows networking is required in order to use CTCI-W32, beginning with WinPcap (see chapter 7). Next verify that the network adapter has an IP address assigned and default gateway assigned. In most installations this will already be the case and no further configuration will be required.

If you are using DHCP rather than assigning a static IP address to the network card, then it will be necessary to tell Hercules the exact hardware (MAC) address of the network adapter Hercules is to use. See section "Configuring Hercules" below for further information.

To verify the IP address of your network interface open the network card properties and double-click on the "Internet Protocol (TCP/IP)" component. The following properties dialog will appear:

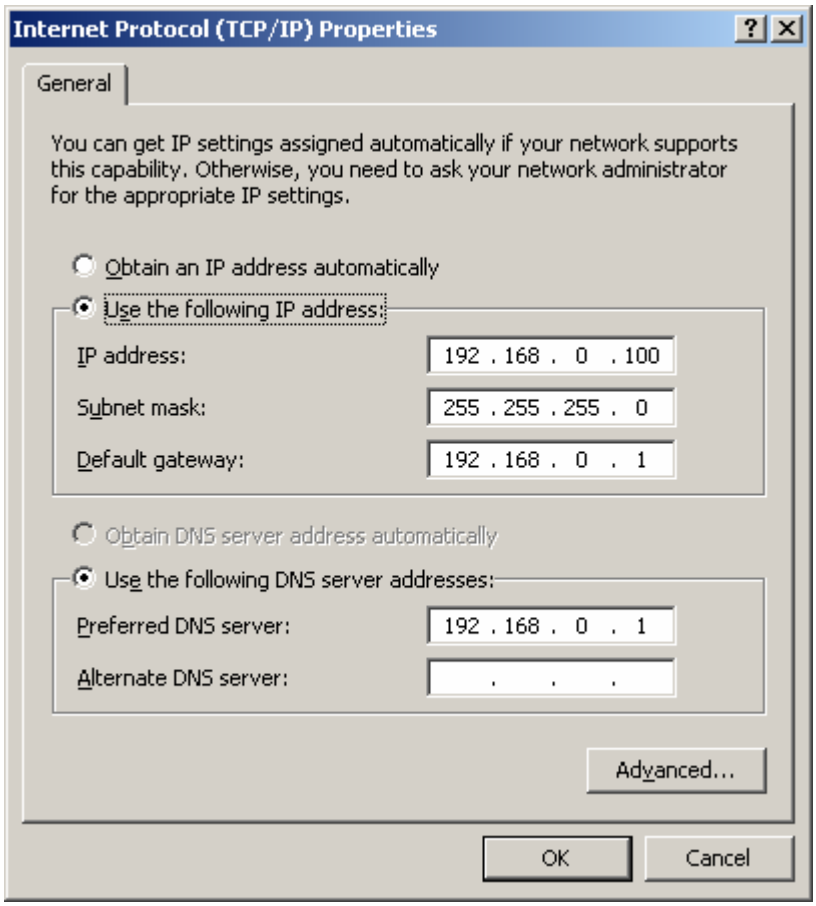


Figure 75: Windows TCP/IP Properties

Make sure you have entered a valid IP address, subnet mask and default gateway. This is all that is necessary on the Windows system side, the remaining configuration of the CTCI-W32 functionality is controlled via Hercules device statements.

11.3.1.1 IP Forwarding

You may or may not need to have "IP Forwarding" enabled on your Windows system. Whether this is required or not depends on your use of a router within your network. If you are using a router to define routes to hosts on your LAN, then you should not need "IP Forwarding" enabled on your Windows system. If however your are not using a physical hardware router then you will likely need to enable IP Forwarding.

To allow your Hercules guest operating system to communicate with hosts on the LAN apart from the host machine itself (where Hercules is running), TCP/IP requires routes to and from the Hercules hosted operating system. This is necessary so that the Hercules virtual guest OS's packets can be properly routed to their final destination. This is typically the role performed by a hardware router. If you do not use a hardware router then Windows "IP Forwarding", together with appropriate ROUTE statements, will perform this role.

Without a router or "IP Forwarding" enabled you will only be able to communicate with your Hercules guest OS from the same Windows computer that Hercules is running on and the hosted OS will only be able to communicate with the Windows system it is running on.

To enable "IP Forwarding" on Windows 2000 / Windows XP, make the following registry change:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters]

"IPEnableRouter"="1"
```

Figure 76: Windows 2000 / XP "IP Forwarding" Registry Key

For Windows 98 / Windows ME, make the following registry change instead:

```
[HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\VxD\MSTCP]

"EnableRouting"="1"
```

Figure 77: Windows 98 / ME "IP Forwarding" Registry Key

If the "IPEnableRouter" (or "EnableRouting") registry value does not yet exist under the mentioned registry key then you will have to add it manually. Select 'New' from the 'Edit' menu and add a new DWORD value with the indicated name.

11.3.2 Configuring Hercules

Two IP addresses must ultimately be assigned, one for the Hercules end of the link and another for the driving system's (host systems) end of the link.

Note that the format of the Hercules configuration file statements for networking emulation has changed from previous releases of Hercules. The older 'CTCI-W32' format has been deprecated. Please ensure your CTC device statements are updated to use the newer 'CTCI' or 'LCS' format in order to remain compatible with current and future releases of Hercules. For example:

```
0E20  3088  CTCI  192.168.0.4  192.168.0.2
0E21  3088  CTCI  192.168.0.4  192.168.0.2
```

Figure 78: Sample CTCI definition for static IP addresses

The first IP address (192.168.0.4) is the IP address of your Hercules system, i.e. the guest operating system running under Hercules. The second IP address (192.168.0.2) identifies the CTCI-W32 logic, i.e. the network adapter that CTCI-W32 is to use in order to reach the Windows TCP/IP stack.

It is recommended that the IP address you choose for your virtual guest (like 192.168.0.4 in the above example) is in the same subnet as your Windows host in order for CTCI-W32 to work properly. The simplest way to approach things is to configure your guest as if it were a real system connected to your

network. Thus it would normally be assigned an IP address within the same subnet as all the other workstations on your LAN.

Note that it is possible to place the guest OS in a separate subnet to your Windows host if you are prepared to define the proper routing entries. However it is usually a lot simpler and less problematic to define all hosts in the one IP subnet.

If your network adapter does not have a static IP address then instead of specifying an IP address as the second parameter in the Hercules device statement, you must specify the MAC address of the adapter you wish to use:

```
0E20  3088  CTCI  -n  00-80-B3-E1-DF-69  192.168.0.4  0.0.0.0
0E21  3088  CTCI  -n  00-80-B3-E1-DF-69  192.168.0.4  0.0.0.0
```

Figure 79: Sample CTCI definition for dynamic IP addresses

Note that the format of the CTCI definition statement is slightly different in this case. The '-n' parameter informs Hercules that this is a MAC address, however the parser still expects to see two IP addresses, hence the second address is specified as a 'dummy' address, i.e. all zeroes. The first IP address is still the address you have assigned to the OS hosted by Hercules.

11.3.3 Configuring the Guest Operating System TCP/IP Settings

The procedure for configuring TCP/IP on any guest operating system you plan to run under Hercules is different for each OS. For example the procedure is completely different for z/VM than it is for z/OS. For detailed instructions on how to configure TCPIP profiles for using a CTCI device please refer to the appropriate operating system manuals.

A sample configuration is shown in below:

```
TCP/IP configuration using CTCI-W32
-----------------------------------

The following values belong in the TCP profile.

The first entry in the TCP profile is the DEVICE entry.  This is the Para-
meter that defines the UCB address assigned to the interface within the MVS
environment.  It is related to the LINK statement to build the TCPIP connec-
tion between the hardware and the TCPIP stack.

        DEVICE     CTC1         CTC           E20
        LINK       CTC1L        CTC           0          CTC1

The second entry is the HOME entry.  This is where the IP address is
assigned to the device interface within the TCPIP stack.  The HOME entry
connects the IP address to the hardware through the LINK defined with the
DEVICE.

        HOME
        192.168.0.4   CTC1L

The GATEWAY describes how the IP stack gets to the rest of the network.  It
defines the physical first hop from the local interface out to the entire
network.  In our example, we are defining a gateway to one physical IP
address, 192.168.0.2.  1492 is the data packet size.

        GATEWAY
        192.168.0.2    =       CTC1L     1492     HOST
```

```
The DEFAULTNET statement instructs what path the IP stack should take to
look for address it doesn't know about.  IP assumes the next hop will be
able to resolve the IP address the stack is trying to connect to.

        DEFAULTNET 192.168.0.2 = CTC1L 1492 0


The final statement needed in the profile is the command to instruct TCPIP
to start the device.

        START CTC1
```

Figure 80: Sample TCP/IP Configuration for CTCI-W32

If you wish to try using an LCS (LAN Channel Station) device instead then the following sample may be useful. LCS can handle any Ethernet packet rather than just IP packets:

```
Sample regular / normal LCS device definitions

0E20-0E21  LCS  -n  172.16.9.163   -m  00-00-5e-90-09-5d   172.16.9.93

------------------------------------------------------------------

Sample LCS device definitions for Enterprise Extender

0E20-0E23  LCS  -n  172.16.9.163   -o  oatfile.txt

------------------------------------------------------------------

oatfile.txt:

*******************************************************
* Dev  Mode Port Entry specific information            *
*******************************************************
0E20  IP   00   PRI   172.16.9.93
HWADD      00         00-00-5E-90-09-5D

0E22  IP   01   SEC   172.16.10.93
HWADD      01         00-00-5E-90-0A-5D


------------------------------------------------------------------

Sample TCPIP PROFILE statements for Enterprise Extender

(Note: not all statements are shown)


IPCONFIG  DATAGRAMFWD  VARSUBNETTING  SYSPLEXROUTING

DEVICE  LCS1     LCS       E20  AUTORESTART
LINK    ETH1     ETHERNET  0    LCS1

DEVICE  VDEV1    VIRTUAL   0
LINK    VLINKA   VIRTUAL   0    VDEV1
```

```
DEVICE  IUTSAMEH MPCPTP
LINK    EELINK   MPCPTP         IUTSAMEH

START   LCS1
START   IUTSAMEH

HOME
172.16.9.93   ETH1
172.16.10.93  VLINKA

BEGINROUTES
ROUTE  172.16.0.0  255.255.0.0  =              ETH1  MTU 1492
ROUTE  DEFAULT                  172.16.13.1    ETH1  MTU 1492
ENDROUTES

PORT
12000 UDP VTAM
12001 UDP VTAM
12002 UDP VTAM
12003 UDP VTAM
12004 UDP VTAM

---------------------------------------------------------------

(Missing / not shown: VTAM definitions)
```

Figure 81: Sample LCS Configuration for CTCI-W32

11.3.3.1 Defining the Guest's Default Gateway

The correct definition of your guest operating systems default gateway depends on whether or not you are using a real router. If you do not have a real router and are instead using the Windows "IP Forwarding" (IP Routing) feature to perform routing, then your default gateway should be the physical adapter on your Windows system that your virtual interface is using.

Alternatively if you do have a real router then your default gateway should be the IP address of your actual router. In the sample LCS configuration immediately above 172.16.13.1 is a real network router and thus knows how to route traffic to the other end of the guest's Enterprise Extender link. Note that, as recommended, the guest's IP addresses (172.16.9.93 and 172.16.10.93) are within the same subnet (255.255.0.0) as the Windows host that Hercules is running under (172.16.9.163).

11.3.4 Tweaking CTCI-W32

The CTCI-W32 protocol supports some additional tuning parameters in addition to the required parameters.

You can adjust the size of the WinPcap kernel device driver's internal packet buffer, as well as the size of TunTap32.dll's own packet I/O buffer to try and increase the performance of your network:

```
0E20  3088  CTCI  -n 00-80-B3-E1-DF-69  -k 1024  -i 64  192.168.1.99  0.0.0.0
0E21  3088  CTCI  -n 00-80-B3-E1-DF-69  -k 1024  -i 64  192.168.1.99  0.0.0.0
```

Figure 82: CTCI-W32 Tuning Parameters

The numbers 1024 and 64 in the above statements are the size of the WinPcap kernel device driver's internal packet buffer (in KB), and the size of TunTap32 DLL's internal packet I/O buffer (in KB) respectively. Please note that memory for WinPcap's kernel device driver's packet buffer is taken from Windows "non-paged" memory pool (i.e. from your system's real physical memory). The size of this buffer has a direct impact on Windows performance. If you choose a ridiculously large number here Windows will have little memory left to work with and the overall performance of your Windows system will be degraded.

The second number (64 in the above example) is the size in KB of the I/O buffer allocated inside the TunTap32 DLL. This defines how much data TunTap32 will request from the device driver each time it needs to do an I/O operation to the physical adapter and therefore how much data will be transferred from the device driver's internal buffer to TunTap32's I/O buffer. The memory for this buffer is allocated in virtual memory from the Hercules process's address space and should not impact Windows's performance unless a ridiculously large number is specified. If too high a number is specified then Windows will likely 'thrash' attempting to page the entire Hercules address space and performance will be degraded.

The TunTap32 DLL passes packets to Hercules one at a time. When it runs out of packets in its internal I/O buffer it requests more data from the WinPcap device driver via the FishPack DLL. The TunTap32 I/O buffer size determines how much data and therefore how many packets it will receive from WinPcap for each request.

The larger this buffer, the fewer actual I/O's TunTap32 will perform to WinPcap via FishPack, but at the same time the longer the interval between those I/O's. This implies that the WinPcap device driver will have to buffer its data for a longer period between less frequent I/O's. This requires a larger kernel buffer in order to ensure that no packets are lost during periods of high network activity. Conversely, the larger the actual I/O (i.e. the more bytes transferred per I/O), the longer each I/O takes. Although this is measured in microseconds, the delay holds up the entire Windows operating system during transfers of data from kernel-space to a user-space. It is good to perform as little I/O as possible as these can be expensive, but the benefit drops quickly if each I/O has significant impact on the overall system, so choose these settings carefully.

Unless you are using a Gigabit or faster network it is usually best to leave these at their default settings. The currently implemented minimum, maximum and default values are:

Buffer Type	Minimum	Default	Maximum
WinPcap device driver capture buffer size	64 KB	1 MB	16 MB
TunTap32 DLL I/O buffer size	16 KB	64 KB	1 MB

Table 24: CTCI-W32 Buffer Sizes

If you decide to adjust these buffer size values to try and increase network throughput and performance, you may find the Hercules "tt32stats" command useful.

The tt32stats command shows the actual tt32 statistics as shown below:

```
23:59:11.098 0000066C tt32 stats e20
23:59:11.098 0000066C TunTap32.dll Statistics:
23:59:11.118 0000066C Size of Kernel Hold Buffer:       1024K
23:59:11.118 0000066C Size of DLL I/O Buffer:             64K
23:59:11.128 0000066C Maximum DLL I/O Bytes Received:      2K
23:59:11.138 0000066C            7 Write Calls
23:59:11.148 0000066C            8 Write I/Os
```

```
23:59:11.148 0000066C          319 Read Calls
23:59:11.158 0000066C          259 Read I/Os
23:59:11.168 0000066C          282 Packets Read
23:59:11.168 0000066C            8 Packets Written
23:59:11.178 0000066C        38172 Bytes Read
23:59:11.178 0000066C          542 Bytes Written
23:59:11.188 0000066C            1 Internal Packets
23:59:11.198 0000066C            2 Ignored Packets
```

Figure 83: tt32 Statistics

If the reported "Maximum DLL I/O Bytes Received" value is identical to the value specified for the "Size of DLL I/O Buffer", then each time TunTap32 requested more packets the WinPcap device driver had at least a full buffer worth waiting to be delivered to TunTap32.

This generally indicates that the default buffer size is too small or that your network is extremely busy. This would be the least common situation though. TunTap32 is rarely unable to deliver an entire buffer of packets to Hercules before another buffer arrives unless Hercules is performing extremely poorly or your network really is under heavy load.

12. Installation of Vista tn3270

Figure 84: Vista tn3270 Logo

12.1 Vista tn3270

Vista tn3270 is a Windows program designed to emulate IBM 3270 terminals connected to a host via an IP. It is written by Tom Brennan, is currently available as a free 30 day trial and a perpetual license costs approximately $30 US dollars. This emulator was created with mainframe programmers in mind and has some unique features unavailable on even the most expensive commercial emulators.

Vista has features designed especially for programmers such as built-in multiple cut and paste buffers, fully customizable keyboard, extensive select/copy/paste functions – especially the “SelectJCL” function that is used to select dataset names, parameters, and similar items with a single mouse click.

Vista uses bitmapped raster fonts for the clearest text possible. There are 2 sets, "Thick" and "Thin", in 73 sizes each from 4x6 to 16x36. With so many sizes you can easily setup Vista to suit your monitor size and terminal model preferences in either full-screen or windowed mode.

Parts of the following sections about the Vista tn3270 installation have been taken from the original Vista tn3270 documentation with the kind permission of Tom Brennan.

12.2 Downloading the Installation Routine

The binaries for the Vista tn3270 Terminal Emulation can be downloaded directly from Tom Brennan's Vista tn3270 webpage using the following link:

www.tombrennansoftware.com

The downloaded version of Vista tn3270 can be used for a free trial for 30 days following installation. After this free trial period a license key must be entered for continued use of the software.

12.3 Install Vista tn3270

To start the installation process, just double-click on the downloaded executable file.

A confirmation screen appears.

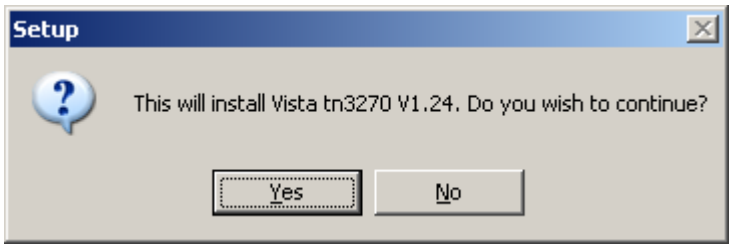


Figure 85: Vista tn3270 – Setup Confirmation Screen

Click “Yes” to start with the installation after which a welcome screen is presented:

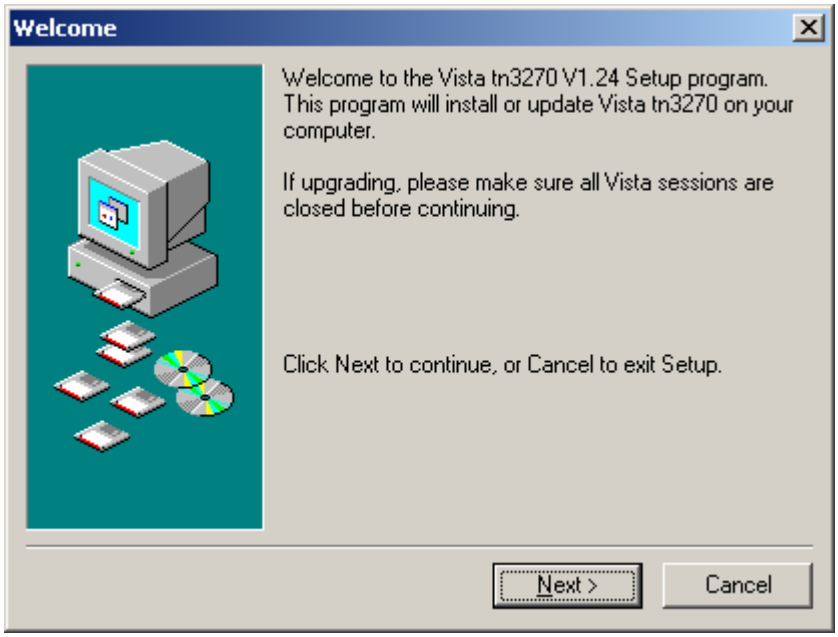


Figure 86: Vista tn3270 – Welcome Screen

Click on "Next >" to continue the installation process.

The following screen prompts you for the installation directory. Select a destination directory of your choice and click “Next” to continue.

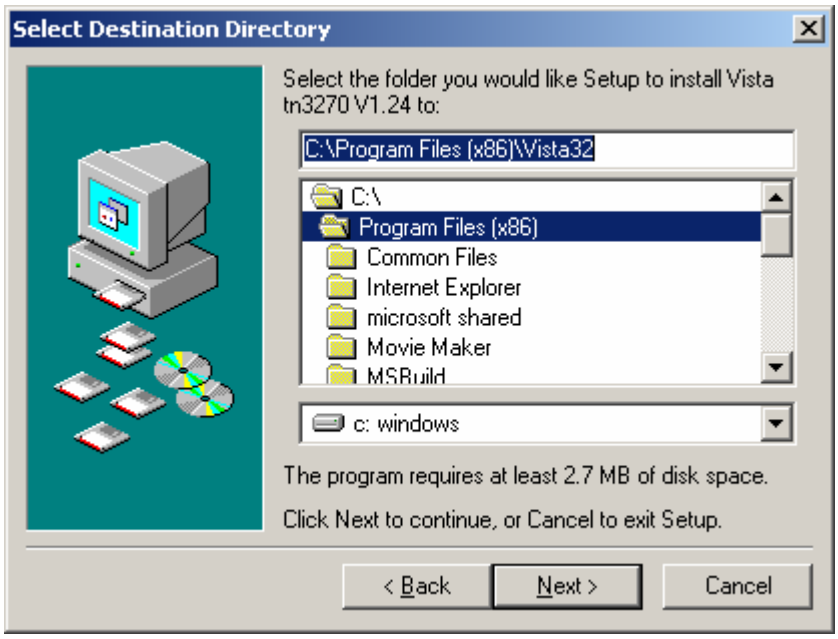


Figure 87: Vista tn3270 – Select Destination Directory

The screen presented next allows you to choose the Windows Start Menu Group to which the emulator icons should be added. The default group is usually acceptable. Click on “Next” again to continue.

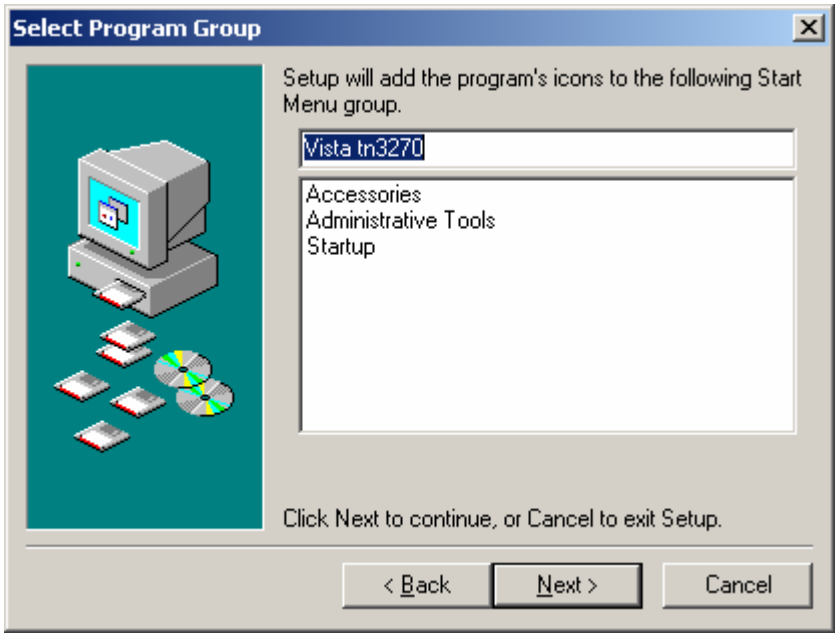


Figure 88: Vista tn3270 – Select Program Group

Now the installation program is ready to copy the necessary files to your harddisk. A confirmation screen appears where you can change your previous selections. If you do not want to change any of these click on "Install" and the setup program begins the actual installation.

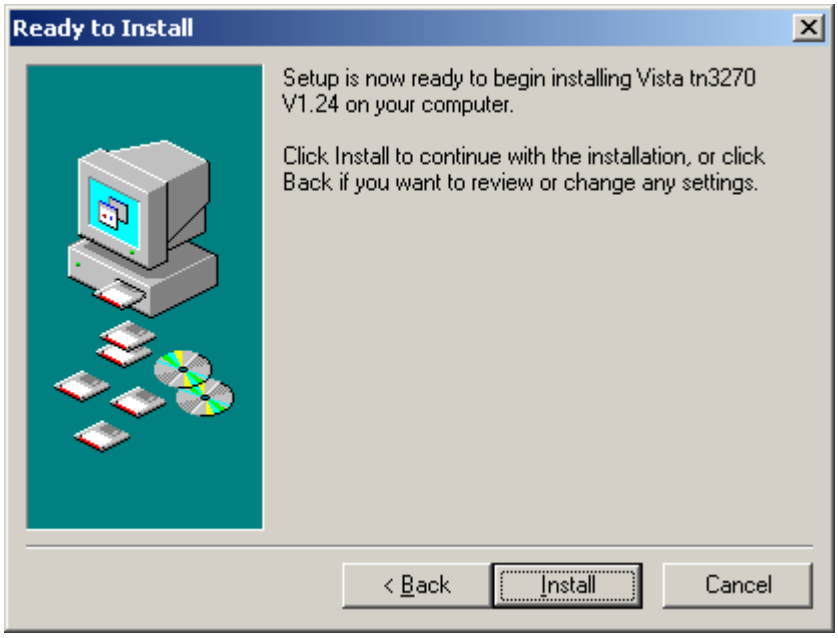


Figure 89: Vista tn3270 – Ready to Install Screen

When the installation process is finished the “Setup Completed” screen will appear. You must confirm by clicking “Finish”.

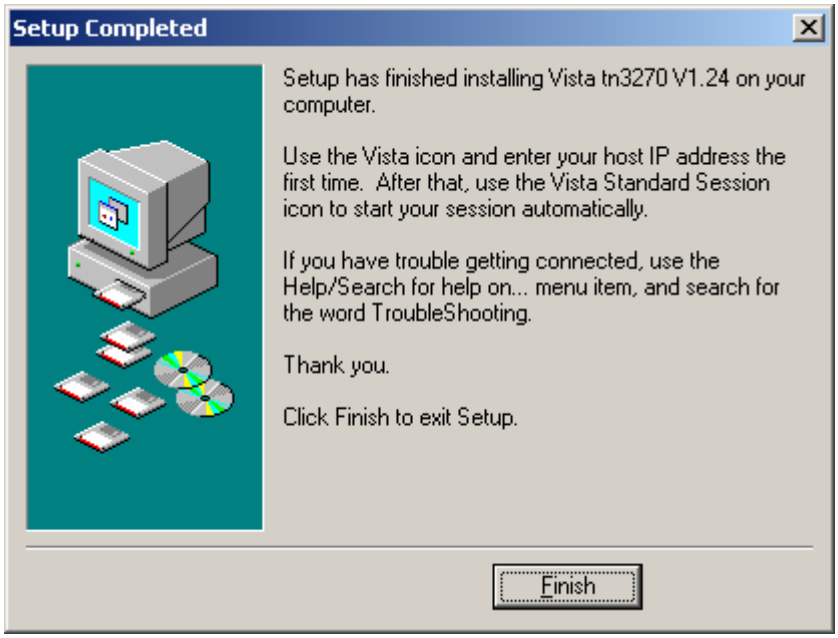


Figure 90: Vista tn3270 – Setup Completed

12.4 Activation of the Software

The Vista tn3270 software creator, Tom Brennan, allows you to try the product for 30 days. When this period expires a licence key must to be entered to enable the product for use again. You can purchase the product via the online purchasing system at the product website. After purchase a license key is sent by email, normally within 24 hours.

12.5 Create Sessions

The detailed process of creating terminal emulation sessions and the impact of all possible options within Vista tn3270 is beyond the scope of this manual. For details regarding these please refer to the printed Vista tn3270 documentation or the online help within the product. A short introduction to creating Vista sessions for connection to a Hercules hosted OS follows.

Start a Vista tn3270 window by clicking on "Vista" in the Vista program folder or wherever you chose to install the product. This creates a new Vista instance and immediately opens the "Start a new Terminal Session" dialog, where you can specify IP address and port number. If instead you click on “Vista Standard Session”, then Vista immediately attempts to connect to the last used IP address and port.

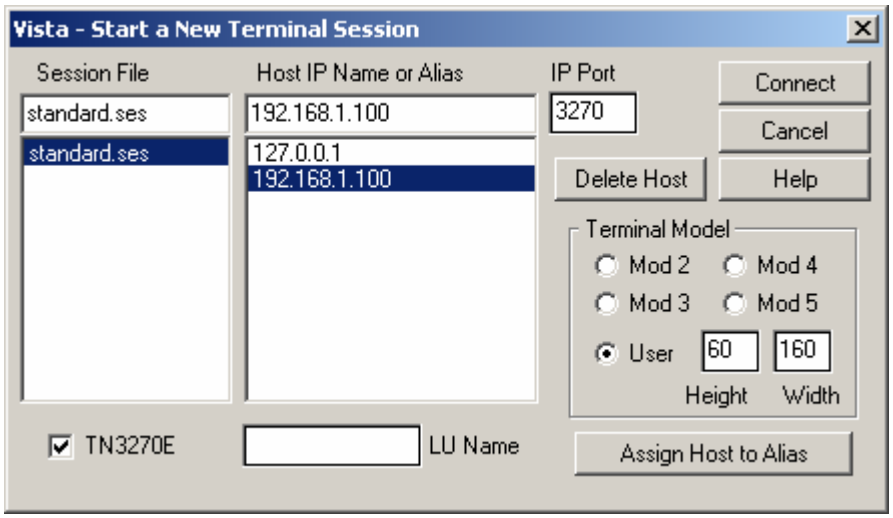


Figure 91: Vista tn3270 - New Terminal Session Dialog

The following options can be specified in this dialog:

Session File — Select an existing session file or type a new name. A session file contains most of the parameters for a session such as font size, screen colours and other options. Multiple Vista windows can share the same session file but that each window stores parameters when it is closed. This means that the parameters will be set to those of the last window closed.

Host IP Name/Alias — This name is usually a dot address like 206.85.100.23 or a DNS (Domain Name Server) name, such as "tn3270.company.com". It can also be an Alias name that points to either a DNS or dot address (see Assign Host to Alias below). IP names and their associated port numbers are stored in the VISTA.INI file rather than the session file so that they can be shared among sessions.

The special host name "localhost" or the IP address "127.0.0.1" means the local machine.

IP Port

Each Host IP Name has an associated IP Port number which you can change using this field. Normally TN3270 is defined to port 23 but in some cases your connection may need to use a different port number. This port number has to be the same as that specified for the CNSLPORT system parameter in the Hercules Configuration File.

If you need to logon to the same hostname using various port numbers create multiple Alias names which point to the same host but use different ports. Additionally you can define aliases for IP addresses. This can help you remember which host Vista is connecting to.

Delete Host

This button can be used to delete the selected Host IP Name if you want to clean up the list.

Terminal Model

Vista can emulate 5 standard terminal models:

Mod 2 (24 lines by 80 columns)

Mod 3 (32 lines by 80 columns)

Mod 4 (43 lines by 80 columns)

Mod 5 (27 lines by 132 columns)

User (variable up to 72 lines by 200 columns)

Assign Host to Alias

Dot addresses and DNS IP names can be cryptic to look at. Instead you can type a more descriptive name such as "P390" in the Host IP Name or Alias field then press the Assign Host to Alias button to relate the alias name to a real DNS name or dot address. Alias names can also provide the ability to logon to the same hostname using different port or LU names.

TN3270E

TN3270E is the default protocol for Vista connections.

LU Name

When in TN3270E mode, you have the option of specifying a VTAM 8 character LUNAME to be used when establishing the connection. If you need to logon to the same hostname using various LU names, create multiple Alias names which point to the same host but have different LU's.

Connect Button

When this button is pressed, Vista attempts to connect to the specified Host IP. If all is correct Vista connects to Hercules and presents the Hercules welcome screen. If however there is a problem you may see an error window such as the following one:

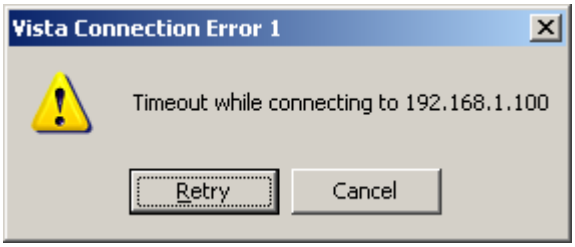


Figure 92: Vista tn3270 - Connection Error

13. Installation of XMIT Manager

Figure 93: XMIT Manager Logo

13.1 XMIT Manager Basics

The XMIT Manager is a Windows based application that allows you to manipulate Xmit format files generated on an IBM mainframe. With XMIT Manager you can open Xmit files and view or extract both, binary or text data contained within them. Xmit Manager will process both partitioned and sequential datasets using a graphical interface.

13.2 Downloading the Binaries

The binaries for the XMIT manager can be downloaded from various locations. The most reliable and therefore recommended source is the CBT website which can be accessed with the following link:

http://www.cbttape.org/njw/

13.3 Installation Steps

Installation of the XMIT Manager is straight forward. First extract the ZIP-file you downloaded from the CBT website to a directory on your harddisk. This creates some files after which you start the installation by running the SETUP.EXE program.

The setup program first presents a screen showing the Software License Agreement.

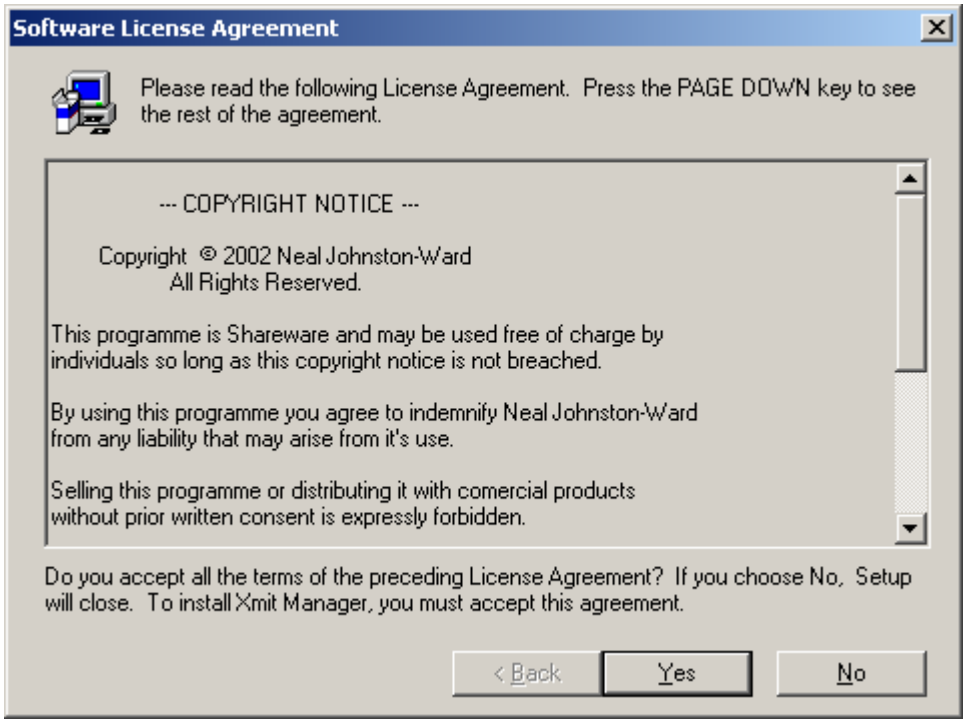


Figure 94: XMIT Manager Setup – Software License Agreement

Read the "Software License Agreement" and accept it by clicking on “Yes”.

The destination location screen will be shown, from here choose the directory where you want the setup program to install the software. The default location “C:\Program Files\Xmit Manager” is usually satisfactory. If you wish to install the software somewhere else click on “Browse” and chose your preferred drive and directory, or type a path directly in the panel.

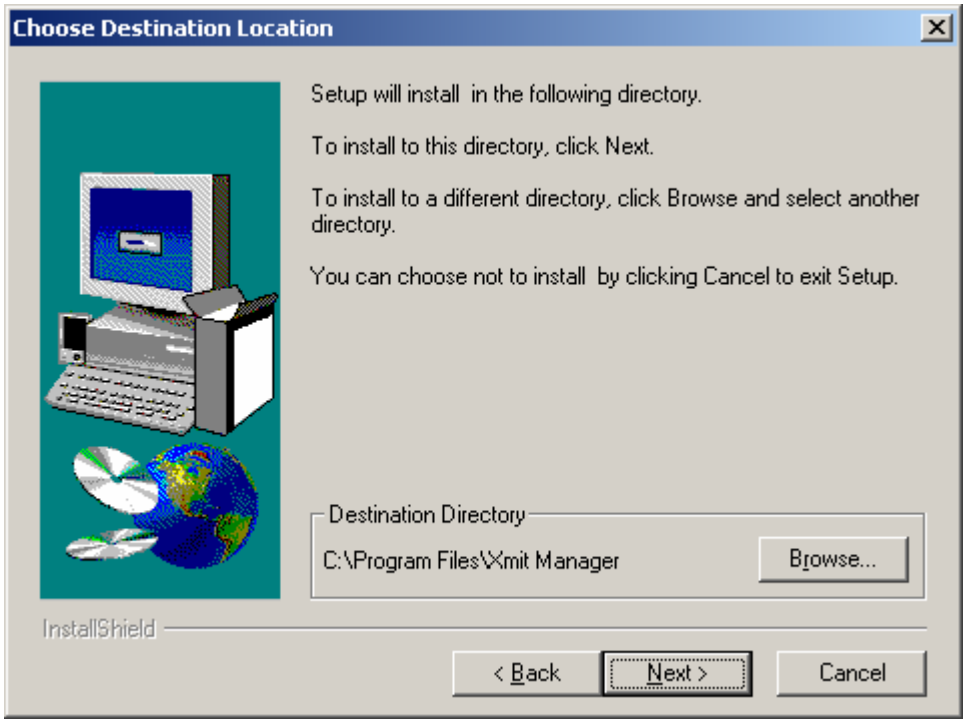


Figure 95: XMIT Manager Setup – Destination Location

When you are satisfied with your choice of the installation directory click on "Next >" to continue.

The setup program will ask you to select a program folder, the default is normally acceptable.

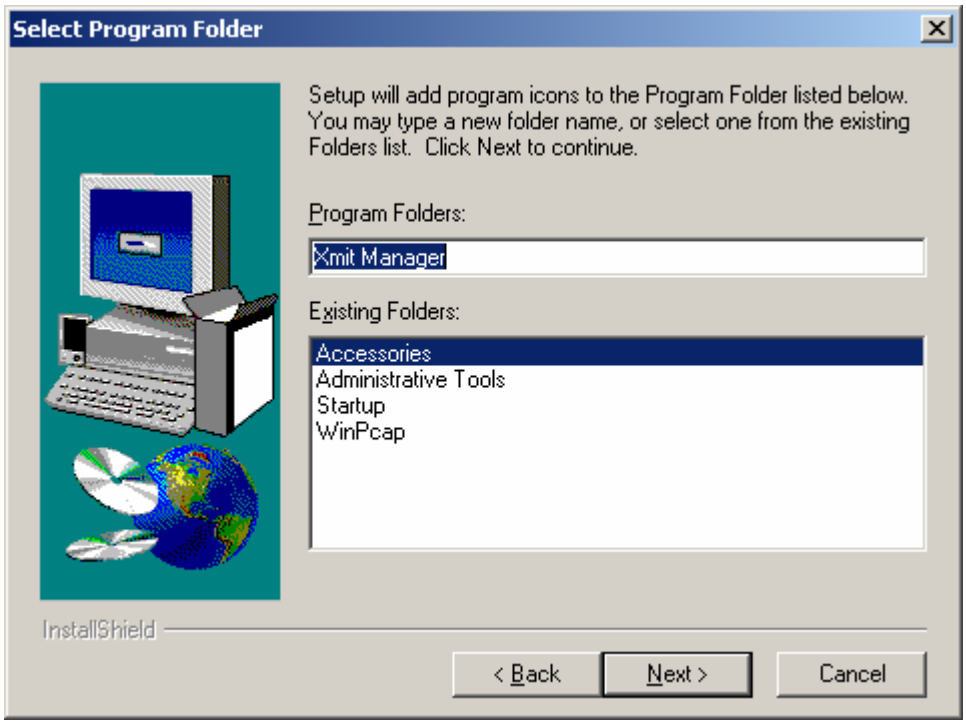


Figure 96: XMIT Manager Setup – Select Program Folder

Click on "Next >" again to proceed with the installation.

The following screen gives you a chance to review your selections and to go back and correct any, if necessary.

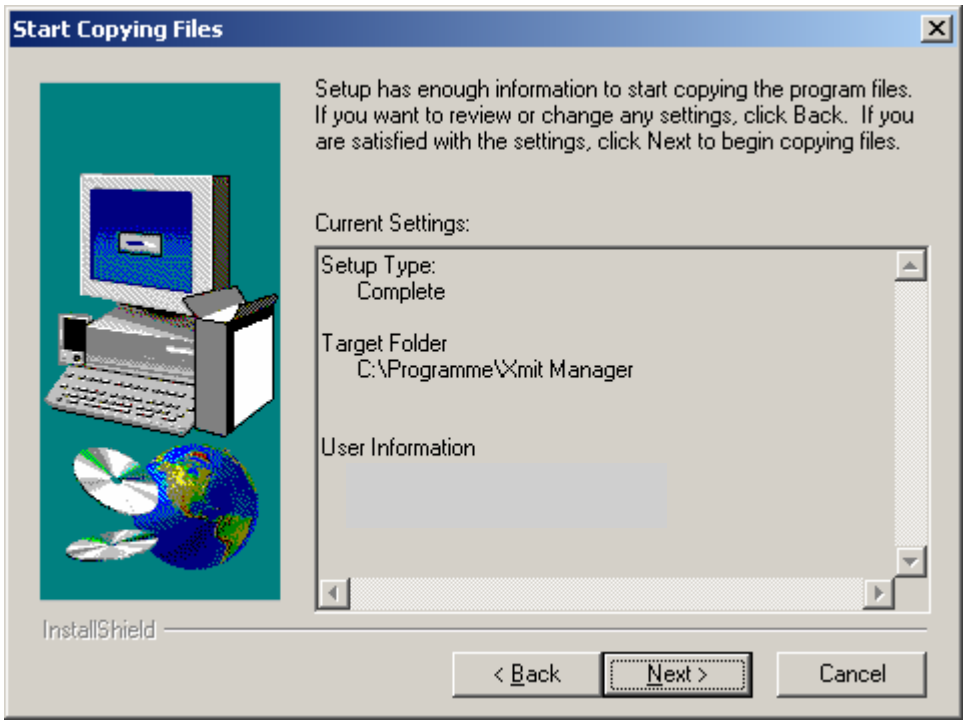


Figure 97: XMIT Manager Setup – Review Settings

When you are satisfied with your installation choices, continue by clicking “Next”. The installer will start copying files to your harddisk, creating icons and updating registry settings.

The final setup screen will then be displayed.

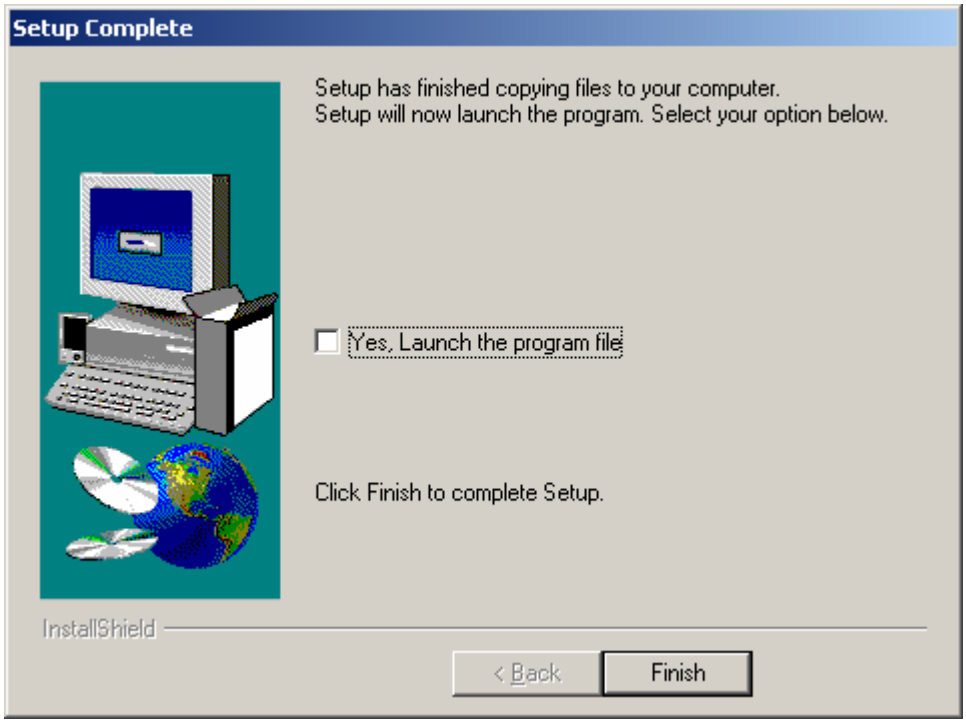


Figure 98: XMIT Manager Setup – Setup Complete

You can now either exit the setup program or optionally select to launch the application as you exit setup. Click on "Finish" to complete.

14. AWS Browse

14.1 AWS Browse Basics

The AWS Browse Utility can be used to view the contents of emulated mainframe tapes (AWS and HET) directly from the Windows desktop without having to start a mainframe operating system. There are currently two implementation of AWS browse.

The first and older one from Rob Storey is no longer supported. The second one is from David B. Trout (Fish), which has more features and is the subject of the rest of this chapter.

14.2 Downloading the Binaries

The binaries for the AWS Browse utility can be downloaded from various locations. However the recommended source is the developers to ensure you have the latest version. The following link leads you directly to the download page:

http://www.softdevlabs.com/Hercules/hercgui-index.html

Note: Beginning with release 1.5.1.1805 of AWS Browse additional DLLs are required. These are Microsoft MFC and VC Runtime DLLs that you can download from the address mentioned above. The installation takes a few seconds and does not require a reboot. There are a 32-bit and a 64-bit version of these DLLs. Please ensure you are using the correct one according to the product you are installing (32-bit or 64-bit version of the Windows GUI).

If you previously installed the Hercules Windows GUI as described earlier in this manual then these DLLs are already present on your system. Note that you only need to install these C Runtime DLLs once even if new versions of AWS Browse are subsequently installed.

14.3 Installation Steps

Download the file called “AWSBrowse-*version*.zip” to your harddisk. ‘Version’ is the version number of the utility in the form “v.r.m.b” (version.release.modification.build), i.e. *AWSBrowse-1.2.0.1278.zip.*

The archive contains several files; the actual AWSBrowse EXE files (32-bit and 64-bit versions) and some DLLs. Please check the README file for the latest instructions.

Unzip these files and copy them to either your Hercules directory (see Chapter 9. Hercules Emulator Installation) or a directory of your choice from where you wish to run the utility. Be sure to you place the executable and the DLLs in the same directory. The Installation is now complete.

Clicking on the desired executable starts AWS Browse with the initial screen showed below.

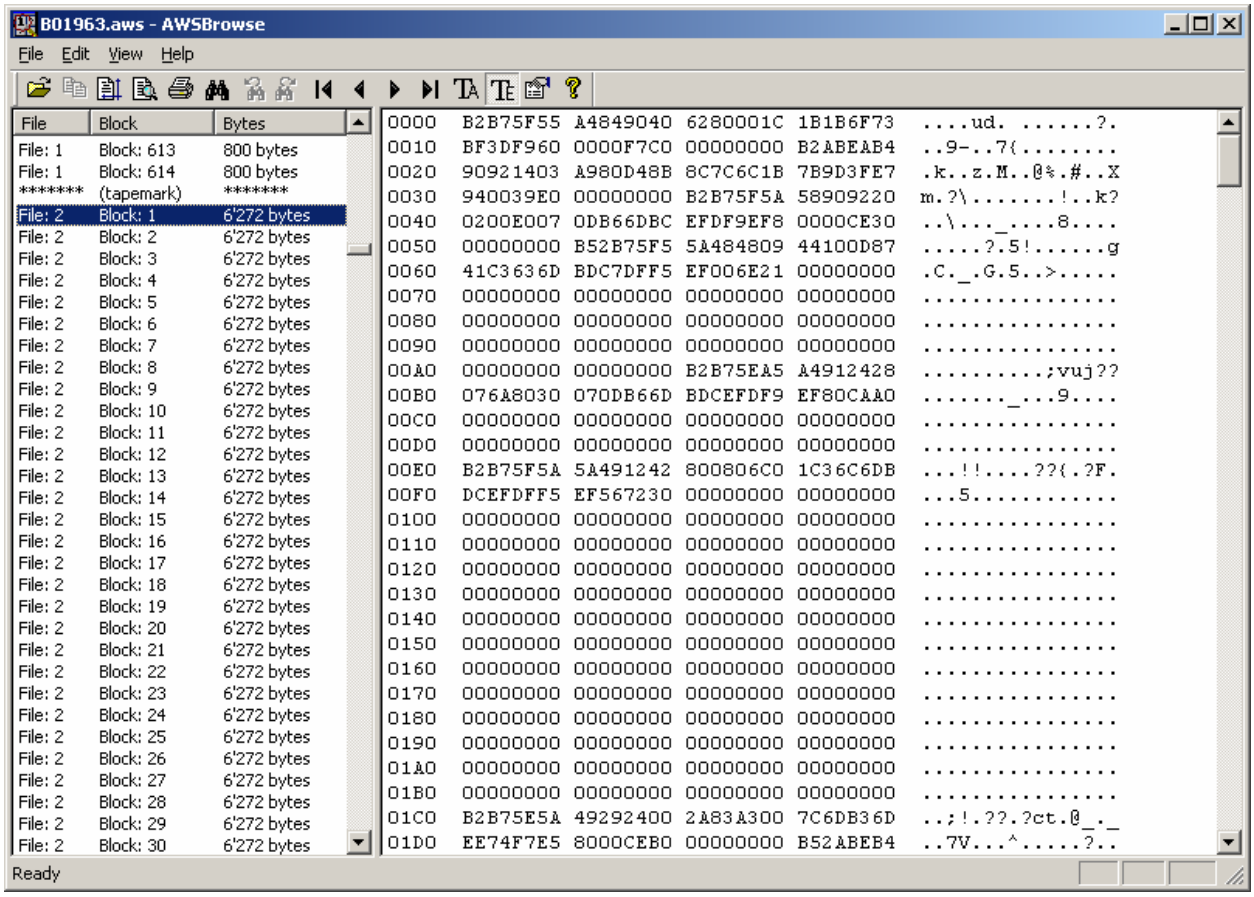


Figure 99: AWS Browse - Initial Screen

15. Hercules “MSVC” Build Instructions

15.1 Introduction

This section provides brief instructions and suggestions to assist you in building the “MSVC” version of Hercules (i.e. the version that does not require Cygwin in order to run under Windows). As it is now recommended to use the new MSVC version, the process of building the Cygwin version of Hercules is not described in this manual.

There is no need to build Hercules from source yourself unless you wish to change the Hercules sources. The following topics are only relevant if you plan to change or enhance Hercules in some way..

There are two supported methods of building Hercules:

- The Visual C++ Toolkit 2003 method, which is best if you are familiar and comfortable with using command-line tools
- The Visual C++ 2005 Express method, which is best if you are more comfortable with GUI Windows development tools and is the recommended method.

Microsoft’s “Visual C++ 2005 Express” product is a free C++ version of the MS Visual Studio product. On April, 19th 2006, Microsoft made an announcement that all Visual Studio 2005 Express Editions will now be completely free. The Visual C++ 2005 Express Edition is the environment used by the Hercules development team.

The alternate choice is the “Visual C++ Toolkit 2003”. This is the method used by the Hercules developers prior to Microsoft releasing the free Visual C++ Express product. With this method you must use a command prompt window and manually enter commands in order to build Hercules as opposed to simply clicking a few buttons when using the VC++ Express product.

The GUI method is the easier of the two and involves simply installing Microsoft’s the Visual C++ 2005 Express product followed by the Platform SDK product. The command line method also involves installing Microsoft’s Platform SDK but additionally requires more complicated temporary installation of the “.NET Framework Software Development Kit” in order to acquire a critical link library. This link library is provided and installed by default with the Visual C++ 2005 express product and is not provided with the Visual C++ Toolkit 2003.

All the download links mentioned in this chapter can be found in “Appendix B. Links” at the end of this document.

15.2 Setting up the Hercules Build Environment (Summary)

A summary of the setup process is presented in the next two sections. Once the setup is complete, the process of building Hercules is extremely simple. Open a command line window and enter the proper build command is all that is required. If you decide to use the Visual C++ Express Edition product instead, you can use the take advantage of the provided “Hercules.sln” solution file which will allow you to complete a Hercules build by just clicking the “build solution” button.

15.2.1 Visual C++ Toolkit 2003 Method

The following steps summarize the setup of the build environment using the Visual C++ Toolkit 2003:

1. Download and install the Visual C++ Toolkit 2003 to obtain the C++ compiler.

2. Download and install the Platform SDK to obtain the headers. ***Note:*** you should not require the full SDK. Only the basic windows components are necessary. If you are unsure as to what is required and what is optional you can install the entire SDK.

3. Download and *temporarily* install the .NET SDK in order to obtain the MSVC LIB and DLL.

   a. Download and install the .NET SDK.

   b. Copy the needed MSVCRT.LIB file to the Platform SDK “libs” directory.

   c. Copy the needed MSVCR71.DLL file to a location where windows can find it.

   d. Uninstall the .NET SDK.

4. Optional: Download and install the ZLIB package to obtain the DLL and headers.

   a. Download both the binary and the source packages.

   b. Unzip these to temporary directories.

   c. Create a permanent directory somewhere.

   d. Copy the required files to the permanent directory

   e. Delete the temporary directories.

   f. Define a ZLIB_DIR environment variable containing the path to the permanent directory.

5. Optional: Download and build the BZIP2 package from Cygwin to get the LIBBZ2.DLL and headers.

   a. Download the BZIP2 source package.

   b. Unzip it to a temporary directory.

   c. Fix their makefile.

   d. Run the makefile (to build the DLL).

   e. Copy the resulting DLL and associated header file to a permanent directory.

   f. Define a BZIP2_DIR environment variable containing the path to the permanent directory.

6. Optional: Download and install the “Perl-Compatible Regular Expressions” (PCRE) package to get the DLL and header files.

   a. Download the PCRE binaries and developer files packages.

   b. Unzip each of them in a temporary directory.

   c. Create a permanent directory for the packages.

   d. Create a ‘bin’, ‘include’ and ‘lib’ subdirectory under the permanent directory created above.

   e. Copy the necessary files into these subdirectories.

   f. Define a PCRE_DIR environment variable pointing to the permanent home directory.

7. Create a command (.bat) file to initialize and invoke the build process that will compile your Hercules Emulator.

15.2.2 Visual C++ 2005 Express Method

The following steps summarize the setup of the build environment using the Visual C++ Toolkit 2003 method:

1. Download and install the Visual C++ 2005 Express.

2. Download and install the Platform SDK to ensure you have a complete set of the latest Windows headers.

   Note: you should not require the full SDK. Only the basic windows components are necessary. If you are unsure as to what is required and what is optional you can install the entire SDK.

3. Add the Platform SDK directories to the Visual C++ 2005 Express build directories list. The last three steps are actually covered in more detail in the relevant Microsoft website.

4. Optional: Download and install the ZLIB package to get the DLL and headers.

   a. Download both the binary and the source packages.

   b. Unzip these to temporary directories.

   c. Create a permanent directory somewhere.

   d. Copy the needed files to the permanent directory

   e. Delete the temporary directories.

   f. Define a ZLIB_DIR environment variable containing the path to the permanent directory.

5. Optional: Download and build the BZIP2 package from Cygwin to get the LIBBZ2.DLL and headers.

   a. Download the BZIP2 source package.

   b. Unzip it to a temporary directory.

   c. Fix their makefile.

   d. Run the makefile (to build the DLL)

   e. Copy the resulting DLL and associated header file to a permanent directory.

   f. Define a BZIP2_DIR environment variable containing the path to the permanent directory.

6. Optional: Download and install the “Perl-Compatible Regular Expressions” (PCRE) package to get the DLL and header files.

   a. Download the PCRE binaries and developer files packages.

   b. Unzip each of them in a temporary directory.

   c. Create a permanent directory for the packages.

   d. Create a ‘bin’, ‘include’ and ‘lib’ subdirectory under the permanent directory created above.

   e. Copy the necessary files into these subdirectories.

   f. Define a PCRE_DIR environment variable pointing to the permanent home directory.

7. Start Visual C++ 2005 Express, open the provided “Hercules.sln” solution file and click on the “build solution” button.

15.3 Visual C++ Toolkit 2003 Method (Details)

The overall setup process is basically very simple and is explained in detail in the next two sections. Once the setup is complete, the process of building Hercules is extremely simple; open a command line window and enter the proper build command is all that is required. If you decide to use the Visual C++ Express Edition product instead, you can use the take advantage of the provided “Hercules.sln” solution file which will allow you to complete a Hercules build by just clicking the “build solution” button.

15.3.1 Download of C++ Toolkit Compiler and Platform SDK

The Microsoft Visual C++ Toolkit 2003 includes command line versions of the Visual C/C++ compiler and linker as well as the standard C/C++ Library.

Download the installation file (a 32MB download) and run it to install the Toolkit. The resulting directory “C:\Program Files\Microsoft Visual C++ Toolkit 2003” should occupy approximately 30MB of disk space. Once the Toolkit is installed run the 'cl' program at least once from the Visual C++ Toolkit command prompt window before installing the Platform SDK.

The “Platform SDK Web Install” contains include files and libraries required to create applications for Win95 / 98 / ME / NT / 2000 / XP.

Important: Before installing the Platform SDK you should first launch "Visual C++ Toolkit 2003 Command Prompt" and *run the 'cl' program at least once* in order to register the VC++ environment variables properly. Also verify that the environment variable "VCToolkitInstallDir" exists and is set correctly.

15.3.2 Obtaining MSVCRT.LIB and other Files

To build the DLL version of Hercules you will need a Microsoft file MSVCRT.LIB that is not included in the free Visual C++ Toolkit 2003. You can obtain this file by downloading and installing the ".NET Framework Version 1.1 Software Development Kit (SDK)". This is a 108 MB download and may take some time if using a slow Internet connection.

After installing the .NET Framework SDK proceed with the following steps:

- Move the file msvcrt.lib

  from directory:

  C:\Program Files\Microsoft Visual Studio .NET 2003\Vc7\lib

  to directory:

  C:\Program Files\Microsoft Visual C++ Toolkit 2003\lib

- Copy the file NMAKE.EXE

  from:

  C:\Program Files\Microsoft.NET\v1.1\Bin

  to:

  C:\Program Files\Microsoft Visual C++ Toolkit 2003\bin

- Copy the file SetEnv.cmd

  from:

  C:\Program Files\Microsoft Platform SDK

  to:

  C:\Program Files\Microsoft Visual C++ Toolkit 2003

- Copy the file cvtres.exe

  from:

  C:\WINNT\Microsoft.NET\Framework\v1.1.4322

  to:

  C:\Program Files\Microsoft Visual C++ Toolkit 2003\bin

When the above files have been copied you can uninstall the .NET v1.1 Framework SDK (via the Control Panel "Add/Remove Programs"). This will release approximately 250 MB of disk space. **Note:** do not un-install the Platform SDK or the Toolkit, only uninstall the .NET v1.1 SDK. You can also delete the 108 MB installation file as it is no longer required.

15.4 Visual C++ 2005 Express (Details)

Installation of the Visual C++ 2005 Express is via a standard Windows installation. Details can be found on Microsoft's web page "Using Visual C++ 2005 Express Edition with the Microsoft Platform SDK" (see Appendix B for links).

15.4.1 Adding Platform SDK Directories to Visual C++ 2005 Express

In this section we inform Visual C++ 2005 Express of the location of the Platform SDK "Executables", "Include" and "Library" directories. This is required in order to find various header and library (.LIB) files required for the build process.

- Select "Options" from the "Tools" menu and an 'Options' dialog should appear.
- In the "Show directories for:" combo-box select: "Executable files" (if this is not already selected) and then click on the yellow folder icon to create a new entry in the list. The new entry will be initially empty and a value will need to be specified.
- To specify this value click on the ellipsis button at the end of the field and browse to the directory where you installed the Platform SDK. Select the 'bin' directory and click 'Open'. That will add that particular directory to the list however it may not be in the proper position. To fix this click on the up / down arrow and move the entry so that it comes first, before all other entries.
- Then do the same thing again, but this time selecting the Platform SDK's "bin/winnt" directory instead. Position it in the list so that it comes the second one, immediately following the "bin" entry.
- Now click on the "Show directories for:" combo-box again and this time select "Include files". Add a new entry as previously but this time add Platform SDK\include as the new entry then move this to the proper position immediately following the entry for $(VCInstallDir)include.
- Lastly add your Platform SDK\lib directory to the "Libraries files" list immediately following the $(VCInstallDir)lib entry and click OK.

15.5 Building Hercules using Visual Toolkit 2003

Open a Command-Prompt window and navigate to the directory where the Hercules source code is then enter the following command:

```
makefile.bat  RETAIL  makefile-dllmod.msvc  2  -a
```

Figure 100: The Hercules Build Command (Visual Toolkit 2003)

The third argument ('2' in the above example) is the maximum number of CPU engines you wish your Hercules to support. There is no default, the argument must be specified on the makefile.bat command although Hercules itself defaults to 2 CPU engines if the MAX_CPU_ENGINES variable is not specified at build time.

The last argument ('-a' in the above example) requests a "full" re-build of everything. You may optionally leave off this argument to build only what is necessary or specify 'clean' instead to clean-out your build directories.

Note: some problems have been reported when a full build (i.e. a "rebuild all" with the '-a' option) has **not** been performed. This occurs sporadically under as-yet undetermined conditions. Due to this it is recommended to always do a full rebuild.

A possible cause for this behaviour may be that link-time optimizations are specified but then not re-linked (not rebuilt) across all modules, potentially confusing the Global Optimizer, however this has not yet been confirmed.

15.6 Building Hercules using Visual C++ 2005 Express

Default Solution and Project files for Visual Studio 7.0 and 8.0 (as well as Visual Studio 6.0 workspace and project files) are provided as part of the Hercules source-code distribution. Simply open one of these, i.e. either the Solution or Workspace file, depending on the version of Visual Studio you're using and click the 'Build' or 'Rebuild All'(*) button. This is all that is required to build the Hercules executables from source.

The default makefile project simply invokes "makefile.bat" which, after calling a few key batch files to setup the build environment, invokes the 'nmake' command for "makefile-dllmod.msvc".

Note: some problems have been reported when a full build (i.e. a "rebuild all" with the '-a' option) has **not** been performed. This occurs sporadically under as-yet undetermined conditions. Due to this it is recommended to always do a full rebuild.

A possible cause for this behaviour may be that link-time optimizations are specified but then not re-linked (not rebuilt) across all modules, potentially confusing the Global Optimizer, however this has not yet been confirmed.

15.7 Setting up ZLIB Support

ZLIB is a compression algorithm written by Jean Loup Gailly and Mark Adler and may be used in the Hercules project pursuant to the ZLIB license.

In source form the Hercules project does not contain any ZLIB source code at all. In binary form however the Hercules project may include an unmodified version of the ZLIB runtime DLL in addition to its own distribution binaries.

The “ZLIB_DIR” environment variable defines the location where the files required for building a version of Hercules with ZLIB compression support are found. The makefile used by the Hercules build process tests whether this environment variable is defined and acts accordingly to build Hercules with or without ZLIB compression support.

***Note*:** The environment variable can be defined temporarily via the “SET” command or permanently via the System Control Panel. Refer to Windows help for more information about environment variables.

If ZLIB_DIR is undefined when building Hercules then an attempt is made to locate the ZLIB library in the directory *winbuild\zlib\win32_32.* If ZLIB_DIR contains a non-existing path name, make exits with an error. The ZLIB_DIR variable may also be set to the special keyword “NONE” to indicate that support for ZLIB compression should not be generated. To summarize: You can leave the ZLIB_DIR environment variable undefined or set to “NONE” to disable ZLIB support. If however ZLIB_DIR is set to anything other than “NONE” then it must be set to a valid path.

The environment variable ZLIB_DIR should contain the *base path* for the ZLIB files. This means that the following files and directories are expected to exist in the specified ZLIB_DIR directory:

- $(ZLIB_DIR)\zlib1.dll
- $(ZLIB_DIR)\lib\zdll.lib
- $(ZLIB_DIR)\include\zlib.h
- $(ZLIB_DIR)\include\zconf.h

Remember to “make clean” if the build type is changed or the ZLIB location is modified.

Note: The *zlib1.dll* Dynamic Link Library (DLL) will be copied to the appropriate build target directory as part of the build process in order to ensure that the DLL is obtainable by the platform PE loader at run time. PE stands for Program Executable and designates the Windows EXE and DLL executable file formats.

15.7.1 Obtaining ZLIB

Building from the distributed Hercules source tree does not incorporate by default ZLIB as a compression mechanism as:

- ZLIB is an external project completely separate from the Hercules project itself.
- Windows does not provide any documented location for the ZLIB library runtime and/or header files.

In order to build Hercules with ZLIB support (for DASD, TAPE, etc) go to http://www.zlib.net and locate the download for the 'zlib compiled DLL'. This is only a small download of approximately 75 kB.

Extract that ZIP file to the *winbuild\zlib\win32_32* directory relative to the Hercules source code directory or alternatively, install the file according to the ZLIB_DIR environment variable instructions above. The Hercules project is currently known to have been successfully built using ZLIB version 1.2.2, the current version at time of writing.

15.8 Setting up BZIP2 Support

BZIP2 is a freely available open Source, BSD-style license, patent free, high-quality data compressor written by Julian R. Seward. It typically compresses files to within 10% to 15% of the best available techniques from the PPM family of statistical compressors, and performs approximately twice as fast at compression and six times faster at decompression than comparable implementations.

In source form the Hercules project does not contain any BZIP2 source code at all. In binary form however the Hercules project may include an unmodified version of the BZIP2 runtime DLL in addition to its own distribution binaries.

The "BZIP2_DIR" environment variable defines the location of the BZIP2 link library and header file. If the BZIP2_DIR environment variable is undefined then BZIP2 support will not be included when you build Hercules. Furthermore, if BZIP2_DIR defines a non-existing path, make will exit with an error. The BZIP2_DIR may also be set to the special keyword "NONE" producing the same effect as it not being defined.

Note: you can define an environment variable either temporarily via the "SET" command or permanently via the System Control Panel. Refer to Windows help for more information.

BZIP2_DIR should define the *base path* for BZIP2 and the following files are expected:

- $(BZIP2_DIR)\bzlib.h
- $(BZIP2_DIR)\libbz2.lib
- $(BZIP2_DIR)\libbz2.dll

Remember to "make clean" if the build type is changed or the BZIP2_DIR location is modified.

Note: The *libbz2.dll* Dynamic Link Library (DLL) will be copied to the appropriate build target directory as part of the build process in order to ensure that the DLL is obtainable by the platform PE loader at run time. PE stands for Program Executable and designates the Windows EXE and DLL executable file formats.

15.8.1 Obtaining BZIP2

Building from the distributed Hercules source tree does not by default incorporate BZIP2 as a compression mechanism as:

- BZIP2 is an external project completely separate from the Hercules project itself.
- Windows does not provide any documented location for the BZIP2 library runtime and/or header files.

In order to build Hercules with BZIP2 support (for DASD, TAPE, etc) go to http://www.bzip.org, click on the download link then locate the link titled:

"Here is the 1.0.3 source tarball, which includes full documentation"

near the beginning of the page. Unfortunately you have to manually build the BZIP2 DLL yourself from source or obtain a pre-built copy of it with associated header and lib files.

This is a small download of approximately 650 kB and the file is in *tar.gz* format. You will require WinZip or some other utility that understands *tar.gz* format in order to unzip it.

Extract the files to a directory of your choosing and build the DLL using ***a modified version*** of the supplied makefile.msc. The changes necessary in the makefile are listed here:

- Add libbz2.dll to the beginning of the build-all rule.

  change:

  all: lib bzip2 test

  to:

  all: libbz2.dll lib bzip2 test

- Define the libbz2.dll build rule by **inserting a blank line** following the above line and then code the following two lines after the blank line:

  libbz2.dll: $(OBJS)

  link /dll /def:libbz2.def /out:libbz2.dll $(OBJS)

  NOTE! *the tab preceding the "link" statement above is necessary.*

- Remove setargv.obj from the bzip2 compile-and-link statement.

  change:

  $(CC) $(CFLAGS) -o bzip2 bzip2.c libbz2.lib setargv.obj

  to:

  $(CC) $(CFLAGS) -o bzip2 bzip2.c libbz2.lib

- Change the LIB build statement in the lib: section

  change:

  lib /out:libbz2.lib $(OBJS)

  to:

link /lib /out:libbz2.lib $(OBJS)

- Finally add the following line as the last statement in the clean: section near the very end of the makefile:

  del libbz2.dll

Once the above changes to the makefile have been made open a "Visual C++ Toolkit 2003 Command Prompt" window from your Start menu and enter the following commands to build the DLL:

```
setenv
cd \
cd xxxxxx…     (i.e. to wherever the BZIP2 source files are)
nmake –f makefile.msc
```

Figure 101: BZIP2 build command.

Once it is built, copy the resulting DLL, lib and bzip2.h header file to the defined BZIP2_DIR location then re-build Hercules to access BZIP2 support.

15.9 Setting up PCRE Support

PCRE (Perl-Compatible Regular Expressions) is a set of functions that implement regular expression pattern matching using the same syntax and semantics as Perl 5. PCRE has its own native API as well as a set of wrapper functions that correspond to the POSIX regular expression API. The PCRE library is free, including use in commercial software.

In source form the Hercules project does not contain any PCRE source code at all. In binary form however, the Hercules project may include an unmodified version of the PCRE runtime DLLs in addition to its own distribution libraries.

The Perl-Compatible Regular Expressions library is needed only to support the Hercules Automatic Operator (HAO) Facility. If you do not plan to use the Hercules Automatic Operator Facility then you do not need to install PCRE support and you may skip this step.

15.9.1 PCRE Installation

Use the following procedure to install the PCRE package:

- Download the binaries and developer packages from the PCRE web site.
- Unzip each into a temporary directory.
- Create a directory with any name for the permanent home of the package.
- Create a ‘bin’, ‘include’ and ‘lib’ subdirectory under the just created permanent PCRE directory.
- Copy the following files from the previously mentioned temporary unzip directory to the permanent subdirectories:

  a. Copy the ‘pcre3.dll’ and ‘pcreposix3.dll’ DLLs from the “%temp%\pcre-6.4-1-bin\bin” directory into the permanent PCRE ‘bin’ subdirectory.

  b. Copy the ‘pcre.h’ and ‘pcreposix.h’ header files from the “%temp%\pcre-6.4-1-lib\include” directory

into the permanent PCRE ‘include’ subdirectory.

c. Copy the ‘pcre.lib’ and ‘pcreposix.lib’ link library files from the “%temp%\pcre-6.4-1-lib\lib” directory into the permanent PCRE ‘lib’ subdirectory.

- Define a PCRE_DIR environment variable pointing to the main permanent directory for the package.
- Delete the temporary directory created in the second step.

16. Maximizing Hercules Available Memory

16.1 Introduction

This chapter provides instructions and suggestions to help maximize the amount of available memory on Windows systems. It is the result of research conducted by Fish (David B. Trout) into this subject from a Hercules developer's perspective although it pertains to all general Windows application development.

Please note, that these steps are optional. They are not necessary to run Hercules.

16.2 Windows Memory Layouts

The following figures show how Windows divides each process's 4 GB address space on 32-bit Windows systems.

a) Win 9x memory layout:

```
0x00000000 – 0x00000fff       (4KB)       MS-DOS & 16-bit Windows
                                          (inaccessible; reserved
                                          for NULL ptr assignments)

0x00001000 – 0x0003ffff       (4MB)       MS-DOS & 16-bit Windows
                                          (read/write, but don't touch)

0x00400000 – 0x7fffffff       (2GB)       Private to Win32 processes
                                          (unreserved, usable)

0x80000000 – 0xbfffffff       (1GB)       memory-mapped files, shared
                                          Win32 DLLs, 16-bit apps and
                                          memory allocations; shared by
                                          all Win32 processes.
                                          (read/write, usable)

0xc0000000 – 0xffffffff       (1GB)       VxDs, memory manager and file
                                          system code; shared by all Win32
                                          processes.
                                          (read/write, but don't touch)
```

Figure 102: Windows 9x Memory Layout

b) Win NT memory layout:

```
0x00000000 – 0x0000ffff       (64KB)      inaccessible; reserved
                                          for NULL ptr assignments

0x00010000 – 0x7ffeffff       (2GB)       Private to Win32 processes
                                          (unreserved, usable)

0x7fff0000 – 0x7fffffff       (64KB)      inaccessible; reserved
                                          for NULL ptr assignments

0x80000000 – 0xffffffff       (2GB)       Operating system use
                                          (inaccessible)
```

Figure 103: Windows NT Memory Layout

As can be seen even though each process has 4GB of address space (addressable using 32 bits) only slightly less than 2 GB (31 bits) is actually available to any given application. The rest is reserved for the operating system and cannot be accessed.

To further complicate matters, just as with most operating systems, any application or binary image (i.e. .EXE, .DLL, etc.) can define at link time its preferred base address. Also known as a *load* address, this is the address at which the image in question prefers to be loaded.

The reason images (.EXEs and .DLLs, etc.) have defined preferred base or loading addresses is simply to relieve the operating system from having to relocate the image to another part of memory. At link-edit time - when the image gets built - all of the program's address constants etc. are pre-initialized with whatever preferred base address and offset was defined on the LINK statement. Using this method, if the image can be loaded at the preferred base address, the operating system has no other work to do for the image, specifically; all address constants etc are already set to the values they need to be for this binary image.

If however the image cannot be loaded at the preferred base address then the operating system, after loading it at a different address, must then adjust all of the address references in the entire image to reflect the new base. Applications that do a lot of image loading and may be running on slow systems can take a few extra microseconds to do this work. With the expansion of addressable memory and the speed of both RAM and CPU today such things are no longer really a concern, although Windows is still designed to cater for this.

The end result being that a default build always creates EXE's with a preferred base address of 0x00400000 (4MB) as this is the lowest address an image can be loaded at on Win 9x. If you try specifying a lower base address the linker will issue a warning that the resulting executable may not run on Win 9x.

Additionally the default preferred base address for DLLs is at 0x10000000 which is 256MB. Thus up to 256MB of valuable virtual address space will be wasted as it cannot be used for the "malloc heap" and serves only as space to load additional DLLs if required.

16.3 The "VADUMP" Report

The SDK utility VADUMP ("vadump -sv -p <pid>") can report a process's memory layout. Running it on a default MSVC build of Hercules reveals the following memory layout on a test system:

Memory Layout (before rebase)

```
00230000 : 0023f000  hsys.dll
00240000 : 00276000  hdasd.dll
00280000 : 00290000  hutil.dll
00290000 : 002a3000  zlib1.dll
002b0000 : 002ce000  LIBBZ2.dll
00400000 : 00406000  Hercules.exe
00bc0000 : 00bc6000  hdteq.dll
00bd0000 : 00bec000  dyncrypt.dll
00bf0000 : 00bf7000  dyngui.dll
00c00000 : 00c09000  hdt3505.dll
00c10000 : 00c16000  hdt3525.dll
00c20000 : 00c27000  hdt1403.dll
00c30000 : 00c3b000  hdt3270.dll
02800000 : 028c1000  DbgHelp.dll
10000000 : 101ba000  hengine.dll
43000000 : 43005000  GoogleDesktopNetwork1.dll
68590000 : 685a6000  rsvpsp.dll
68c40000 : 68c49000  RAPILIB.dll
74fd0000 : 74fee000  msafd.dll
75010000 : 75017000  wshtcpip.dll
75020000 : 75028000  WS2HELP.dll
75030000 : 75044000  WS2_32.dll
759b0000 : 759b6000  LZ32.dll
77820000 : 77827000  VERSION.dll
77d30000 : 77da8000  RPCRT4.dll
77e10000 : 77e6f000  USER32.dll
77f40000 : 77f7C000  GDI32.dll
77f80000 : 77ffc000  ntdll.dll
78000000 : 78045000  msvcrt.dll
7c2d0000 : 7c335000  ADVAPI32.dll
7c370000 : 7c409000  MSVCR80.dll
7c570000 : 7c623000  KERNEL32.dll
```

Figure 104: Memory Layout before Rebase

Note 1: It is likely that the default build for the Hercules MSVC version has been changed since the time this analysis was performed. Thus the information above may no longer be accurate and it is included for illustrative purposes only.

Note 2: The VADUMP utility does not display its report in any meaningful sequence. All these displays were manually sorted into ascending virtual address sequence.

16.4 Using "REBASE"

Adding the command (without the quotes of course) **"rebase -b 0x400000 $(X)*.dll"** immediately following the **"all: allzlib alllibbz2"** build-rule in the Hercules dllmod makefile results in the following memory layout:

Memory Layout (after rebase)

```
00370000 : 0038C000  dyncrypt.dll
00400000 : 00406000  Hercules.exe
00420000 : 00427000  dyngui.dll
00450000 : 00486000  hdasd.dll
004a0000 : 004a7000  hdt1403.dll
004d0000 : 004db000  hdt3270.dll
004f0000 : 004f9000  hdt3505.dll
00500000 : 00506000  hdt3525.dll
00510000 : 00516000  hdteq.dll
00530000 : 006ea000  hengine.dll
006f0000 : 006ff000  hsys.dll
00710000 : 00720000  hutil.dll
00720000 : 0073e000  LIBBZ2.dll
00740000 : 00753000  zlib1.dll
02800000 : 028c1000  DbgHelp.dll
43000000 : 43005000  GoogleDesktopNetwork1.dll
68590000 : 685a6000  rsvpsp.dll
68c40000 : 68c49000  RAPILIB.dll
74fd0000 : 74fee000  msafd.dll
75010000 : 75017000  wshtcpip.dll
75020000 : 75028000  WS2HELP.dll
75030000 : 75044000  WS2_32.dll
759b0000 : 759b6000  LZ32.dll
77820000 : 77827000  VERSION.dll
77d30000 : 77da8000  RPCRT4.dll
77e10000 : 77e6f000  USER32.dll
77f40000 : 77f7c000  GDI32.dll
77f80000 : 77ffc000  ntdll.dll
78000000 : 78045000  msvcrt.dll
7c2d0000 : 7c335000  ADVAPI32.dll
7c370000 : 7c409000  MSVCR80.dll
7c570000 : 7c623000  KERNEL32.dll
```

Figure 105: Memory Layout after Rebase

Note how hengine.dll has moved down from its default 0x10000000 slot thus freeing up valuable contiguous memory. This is exactly the outcome we desire.

Continuing these efforts further with a Windows 2000 Server virtual machine the following memory layouts were observed:

W2KSRVR VIRTUAL MACHINE

Using old Ivan snapshot; MAINSIZE 64

```
00230000 : 0023f000  hsys.dll
00240000 : 00278000  hdasd.dll
00280000 : 00290000  hutil.dll
00290000 : 002a2000  zlib1.dll
00400000 : 00406000  hercules.exe
00ba0000 : 00ba6000  hdteq.dll
00bb0000 : 00bb6000  hdt1052c.dll
00bc0000 : 00bc8000  hdt3505.dll
00bd0000 : 00bd6000  hdt3525.dll
00be0000 : 00be6000  hdt1403.dll
00bf0000 : 00bfB000  hdt3270.dll
10000000 : 101dd000  hengine.dll
68590000 : 685a6000  rsvpsp.dll
68c40000 : 68c49000  RAPILIB.dll
72a00000 : 72a2d000  DbgHelp.dll
74fd0000 : 74feeE000  msafd.dll
75010000 : 75017000  wshtcpip.dll
75020000 : 75028000  WS2HELP.dll
75030000 : 75044000  WS2_32.dll
77d30000 : 77da8000  RPCRT4.dll
77e10000 : 77e6f000  USER32.dll
77f40000 : 77f7c000  GDI32.dll
77f80000 : 77ffc000  ntdll.dll
78000000 : 78045000  MSVCRT.dll
7C2d0000 : 7c335000  ADVAPI32.dll
7c340000 : 7c396000  MSVCR71.dll
7c570000 : 7c623000  KERNEL32.dll
```

Using new rebased version; MAINSIZE 64

```
00400000 : 00406000  hercules.exe
00410000 : 0042C000  dyncrypt.dll
00430000 : 00436000  hdt1052c.dll
00440000 : 00447000  hdt1403.dll
00450000 : 00486000  hdasd.dll
004e0000 : 004eb000  hdt3270.dll
004f0000 : 004f9000  hdt3505.dll
00500000 : 00506000  hdt3525.dll
00510000 : 00516000  hdteq.dll
00530000 : 006ea000  hengine.dll
006f0000 : 006ff000  hsys.dll
00710000 : 00720000  hutil.dll
00720000 : 0073e000  LIBBZ2.dll
00740000 : 00753000  zlib1.dll
```

```
 68590000 : 685a6000  rsvpsp.dll
 68c40000 : 68c49000  RAPILIB.dll
 72a00000 : 72a2d000  DbgHelp.dll
 74fd0000 : 74fee000  msafd.dll
 75010000 : 75017000  wshtcpip.dll
 75020000 : 75028000  WS2HELP.dll
 75030000 : 75044000  WS2_32.dll
 77d30000 : 77da8000  RPCRT4.dll
 77e10000 : 77e6f000  USER32.dll
 77f40000 : 77f7c000  GDI32.dll
 77f80000 : 77ffc000  ntdll.dll
 78000000 : 78045000  msvcrt.dll
 7c2d0000 : 7c335000  ADVAPI32.dll
 7c370000 : 7c409000  MSVCR80.dll
 7c570000 : 7c623000  KERNEL32.dll


un-rebased snapshot: old max MAINSIZE = 1500

 00230000 : 0023f000  hsys.dll
 00240000 : 00278000  hdasd.dll
 00280000 : 00290000  hutil.dll
 00290000 : 002a2000  zlib1.dll
 00400000 : 00406000  hercules.exe
 00ba0000 : 00ba6000  hdteq.dll
 00bb0000 : 00bb6000  hdt1052c.dll
 00bc0000 : 00bc8000  hdt3505.dll
 00bd0000 : 00bd6000  hdt3525.dll
 00be0000 : 00be6000  hdt1403.dll
 00bf0000 : 00bfb000  hdt3270.dll
 10000000 : 101dd000  hengine.dll
 6e0b0000 : 6e0c6000  rsvpsp.dll
 6e0d0000 : 6e0d9000  RAPILIB.dll
 72a00000 : 72a2d000  DbgHelp.dll
 74fd0000 : 74fee000  msafd.dll
 75010000 : 75017000  wshtcpip.dll
 75020000 : 75028000  WS2HELP.dll
 75030000 : 75044000  WS2_32.dll
 77d30000 : 77da8000  RPCRT4.dll
 77e10000 : 77e6f000  USER32.dll
 77f40000 : 77f7c000  GDI32.dll
 77f80000 : 77ffc000  ntdll.dll
 78000000 : 78045000  MSVCRT.dll
 7c2d0000 : 7c335000  ADVAPI32.dll
 7c340000 : 7c396000  MSVCR71.dll
 7c570000 : 7c623000  KERNEL32.dll


rebased snapshot: New max MAINSIZE = 1800!

 00400000 : 00406000  hercules.exe
 00410000 : 0042c000  dyncrypt.dll
 00430000 : 00436000  hdt1052c.dll
 00440000 : 00447000  hdt1403.dll
```

```
00450000 : 00486000  hdasd.dll
004e0000 : 004eb000  hdt3270.dll
004f0000 : 004f9000  hdt3505.dll
00500000 : 00506000  hdt3525.dll
00510000 : 00516000  hdteq.dll
00530000 : 006ea000  hengine.dll
006f0000 : 006ff000  hsys.dll
00710000 : 00720000  hutil.dll
00720000 : 0073e000  LIBBZ2.dll
00740000 : 00753000  zlib1.dll
719c0000 : 719d6000  rsvpsp.dll
719e0000 : 719e9000  RAPILIB.dll
72a00000 : 72a2d000  DbgHelp.dll
74fd0000 : 74fee000  msafd.dll
75010000 : 75017000  wshtcpip.dll
75020000 : 75028000  WS2HELP.dll
75030000 : 75044000  WS2_32.dll
77d30000 : 77da8000  RPCRT4.dll
77e10000 : 77e6f000  USER32.dll
77f40000 : 77f7c000  GDI32.dll
77f80000 : 77ffc000  ntdll.dll
78000000 : 78045000  msvcrt.dll
7c2d0000 : 7c335000  ADVAPI32.dll
7c370000 : 7c409000  MSVCR80.dll
7c570000 : 7c623000  KERNEL32.dll
```

Figure 106: Windows 2000 Server Virtual Machine Memory Layouts

The above displays illustrate how a simple rebasing of DLLs helps to free up valuable virtual memory, over 300MB in this particular case, as well as the apparent reshuffling of already loaded DLLs caused by specifying a large MAINSIZE. It is an educated guess based on the observed evidence that Windows does this in order to try and make room for a heap of the specified size.

Continuing the same effort but on a production development system provided the following results:

PRODUCTION SYSTEM

With only Hercules rebased, max MAINSIZE = 1000

```
00370000 : 0038C000  dyncrypt.dll
00400000 : 00406000  Hercules.exe
00420000 : 00427000  dyngui.dll
00450000 : 00486000  hdasd.dll
004a0000 : 004a7000  hdt1403.dll
004d0000 : 004db000  hdt3270.dll
004f0000 : 004f9000  hdt3505.dll

00500000 : 00506000  hdt3525.dll
00510000 : 00516000  hdteq.dll
```

```
00530000 : 006ea000  hengine.dll
006f0000 : 006ff000  hsys.dll
00710000 : 00720000  hutil.dll
00720000 : 0073e000  LIBBZ2.dll
00740000 : 00753000  zlib1.dll
02800000 : 028c1000  DbgHelp.dll
43000000 : 43005000  GoogleDesktopNetwork1.dll
68590000 : 685a6000  rsvpsp.dll
68c40000 : 68c49000  RAPILIB.dll
74fd0000 : 74fee000  msafd.dll
75010000 : 75017000  wshtcpip.dll
75020000 : 75028000  WS2HELP.dll
75030000 : 75044000  WS2_32.dll
759b0000 : 759b6000  LZ32.dll
77820000 : 77827000  VERSION.dll
77d30000 : 77da8000  RPCRT4.dll
77e10000 : 77e6f000  USER32.dll
77f40000 : 77f7c000  GDI32.dll
77f80000 : 77ffC000  ntdll.dll
78000000 : 78045000  msvcrt.dll
7C2d0000 : 7c335000  ADVAPI32.dll
7c370000 : 7c409000  MSVCR80.dll
7c570000 : 7c623000  KERNEL32.dll
```

Rebasing DbgHelp.dll and GoogleDesktopNetwork1.dll, max MAINSIZE = 1800!

```
00370000 : 0038C000  dyncrypt.dll
00400000 : 00406000  Hercules.exe
00420000 : 00427000  dyngui.dll
00450000 : 00486000  hdasd.dll
004a0000 : 004a7000  hdt1403.dll
004d0000 : 004db000  hdt3270.dll
004f0000 : 004f9000  hdt3505.dll
00500000 : 00506000  hdt3525.dll
00510000 : 00516000  hdteq.dll
00530000 : 006ea000  hengine.dll
006f0000 : 006ff000  hsys.dll
00710000 : 00720000  hutil.dll
00720000 : 0073e000  LIBBZ2.dll
00740000 : 00753000  zlib1.dll
00bb0000 : 00c71000  DbgHelp.dll
00e80000 : 00e85000  GoogleDesktopNetwork1.dll
722e0000 : 722f6000  rsvpsp.dll
72300000 : 72309000  RAPILIB.dll
74fd0000 : 74fee000  msafd.dll
75010000 : 75017000  wshtcpip.dll
75020000 : 75028000  WS2HELP.dll
75030000 : 75044000  WS2_32.dll
759b0000 : 759b6000  LZ32.dll
77820000 : 77827000  VERSION.dll

77d30000 : 77da8000  RPCRT4.dll
77e10000 : 77e6f000  USER32.dll
```

```
77f40000 : 77f7c000  GDI32.dll
77f80000 : 77ffc000  ntdll.dll
78000000 : 78045000  msvcrt.dll
7c2d0000 : 7c335000  ADVAPI32.dll
7c370000 : 7c409000  MSVCR80.dll
7c570000 : 7c623000  KERNEL32.dll
```

Figure 107: Production System Memory Layout

As can be seen from the above displays the result of rebasing both Hercules's DLLs - and perhaps more importantly the Google desktop DLL - resulted in the recovery of over 800 MB of usable virtual address space.

16.5 Rebasing DLLs which are in use

If you encounter the error "*REBASE: *** RelocateImage failed (foobar.dll). Image may be corrupted*" when trying to rebase a particular DLL this means that the DLL is either marked read-only or is already loaded and in use by a currently executing process and thus cannot be rebased.

If you still wish to rebase such a module you will need to identify which process or processes are using the module and shut these down or in the case of a service, stop that service. Note that stopping a key system service when other system services depend on it may have negative ramifications for your system overall. Make changes such as this at your own risk carefully testing each individual change and be prepared to recover the entire system from backups if necessary.

As a practical example, in the test case, rebasing "DbgHelp.dll" was easy, as it was not in use. Rebasing "GoogleDesktopNetwork1.dll" is a more difficult process. The RpcSs ("Remote Procedure Call, RPC") service was using it and shutting down this particular critical system service requires several steps. You need to go into "REGEDIT" and set the services 'Start' value from 2 (Automatic) to 4 (Disabled) then reboot into Safe Mode. During this reboot the Windows Explorer may take much up to several minutes to start. Also, once disabled, you will **not** be able to re-enable it via the Services snap-in. You will need to use "REGEDIT" again to change its 'Start' value back from 4 to 2 and then reboot.

Once you have disabled the RpcSs service and have rebooted into Safe Mode go into the directory "Program Files\Google\Google Desktop" and rebase Google's network DLL as needed. This should work now that RpcSs is no longer running, use the following command:

rebase -b 0x400000 GoogleDesktopNetwork1.dll

Once you have rebased it go back into "REGEDIT", change RpcSs back to automatic as described above and reboot normally.

There is no known Microsoft Windows utility that will tell you what process is using a given DLL. The Company "Sysinternals" provides a free "Process Explorer" similar to the Windows' Task Manager with more functionality and features.

You may also want to make some registry tweaks to Hercules Windows GUI so it will accept MAINSIZE values larger than 1024. See the Registry Tweaks section of the Hercules GUI help pages for more information.

Finally the size of the Windows swap file limits the amount of memory a program can acquire. If the Windows swap file is too small then no matter what Hercules does it will never obtain the memory it requests. If possible, set "Maximum paging size" value to the maximum supported.

16.6 Bottom Line

Like the previous examples have shown, by rebasing different DLLs it is possible to recover a lot of available virtual memory. However it should be noted that this is a significant intervention in the Windows operating system. It is strongly recommended that you create an image copy backup of your operating system before experimenting with rebasing DLLs.

Appendix A. Links

- **Hercules Emulator**

  http://www.hercules-390.org

  http://www.bsp-gmbh.com/hercules/index.shtml

- **Hercules WinGUI, “Fish” (David B. Trout)**

  http://www.softdevlabs.com/Hercules/hercgui-index.html

- **CTCI-W32, “Fish” (David B. Trout)**

  http://www.softdevlabs.com/Hercules/ctci-w32-index.html

- **WinPcap, Politecnico di Torino**

  http://www.WinPcap.org

- **Cygwin, Cygnus Solutions (Red Hat Software)**

  http://www.cygwin.com

  http://www.redhat.com/software/cygwin/

- **Turnkey MVS**

  http://www.bsp-gmbh.com/turnkey/index.html

- **Vista tn3270, Tom Brennan Software**

  http://www.tombrennansoftware.com

- **X3270, Paul Mattes**

  http://www.geocities.com/SiliconValley/Peaks/7814

- **AWSBROWSE**

  http://www.softdevlabs.com/Hercules/hercgui-index.html

- **XMIT Manager**

  www.cbttape.org

- **Microsoft Visual C++ Toolkit 2003**

  http://www.microsoft.com/downloads/results.aspx?freetext=Microsoft+Visual+C%2b%2b+Toolkit+2003&productID=&DisplayLang=en

- **Microsoft Visual C++ 2005 Express**

  http://msdn.microsoft.com/vstudio/express/visualc/default.aspx

- **Microsoft Platform SDK**

  http://www.microsoft.com/downloads/results.aspx?freetext=Windows+Server+2003+SP1+Platform+SDK&productID=&DisplayLang=en

- **Microsoft .NET Framework Software Development Kit**

  http://www.microsoft.com/downloads/results.aspx?freetext=NET+Framework+Software+Development+Kit&productID=&DisplayLang=en

- **ZLIB**

  http://www.zlib.net

- **BZIP2**

  http://www.bzip.org

- **PCRE**

  http://gnuwin32.sourceforge.net/packages/pcre.htm
  http://gnuwin32.sourceforge.net/downlinks/pcre-bin-zip.php
  http://gnuwin32.sourceforge.net/downlinks/pcre-lib-zip.php