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

Hercules – User Reference Guide

Version 4 Release 00

Draft - November 21, 2015

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

Hercules – User Reference Guide

Version 4 Release 00

First Edition, November 21, 2015

HEUR040000-00

Contents

Figures

Tables

1. Preface

1.1 Edition information

This edition applies to the Hercules S/370, ESA/390 and z/Architecture Emulator, Release 4.00.0 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 for using and operating the Hercules Emulator and related additional products (both required and optional). For guidance in installation 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 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 operating the Hercules Emulator. It may also be useful if you are responsible for installing the Hercules Emulator.

1.4 What you need to know to understand this book

To understand this book, you should be familiar using software under the Linux, Microsoft Windows or Mac OS X operating systems. You should also have experience with Linux command shells or native DOS (Microsoft Disk Operating System) and the Microsoft Windows command shell.

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 reference for all aspects the Hercules Emulator and related products. You can go through the book chapter by chapter or you can use the book as a reference for all questions regarding the operation of Hercules.

1.6 Revision Notice

Hercules Release:	Version 4 Release 00 Modification 0
Publication Number:	HEUR040000
SoftCopy Name:	HerculesUserReference
Revision Number:	HEUR040000-00
Date:	November 21, 2015

1.7 Readers Comments

If you like or dislike anything of this book please send a mail or email to the address below. Feel free to comment 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 XP, Windows Vista, Windows 7, Windows Server 2003, Windows Server 2008, Visual C++ Toolkit, Visual C++ Express are trademarks of Microsoft Corporation.
- Linux is a trademark owned by Linus Torvalds. The Linux Mark Institute is the exclusive licensor of the Linux trademark on behalf of its owner Linus Torvalds.
- 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.

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.
- Robert Allan for providing the “Linux Installation” part.
- Lutz Mader for providing the “Mac OS X Installation” part.

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

Peter Glanzmann

2. Related Publications

2.1 Hercules – General Information

The Hercules “General Information” manual provides you an overview of the ideas and concepts of the Hercules Emulator as well as a 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 fulfills your requirements.

2.2 Hercules – Installation Guide

The Hercules "Installation Guide" shows you how to install Hercules and all related optional and required software components under the Microsoft Windows, Linux and Apple MacIntosh OS X operating systems.

After going through the installation guide you will have a working emulator environment ready to IPL a S370, S/390 or z/Architecture mainframe operating system.

2.3 Hercules – 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 Hercules GUIs. The usage details for the utilities are covered in the “Hercules Utilities” guide.

After reading this manual you should be able to work with Hercules, to create a configuration file and to use Hercules commands through the console.

2.4 Hercules – Operations and Utilities Guide

The Hercules “Operations and Utilities Guide” describes the operation of Hercules as well as additional utilities that are delivered together with the emulator. Selected utilities from third-party suppliers are also covered in this manual.

After reading this manual you should have the knowledge to operate Hercules and use the right utility for a certain housekeeping task within the Hercules environment. You should also be able to create virtual devices and understand backup / restore procedures.

2.5 Hercules – 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 if you experience problems with Hercules.

2.6 Hercules – Reference Summary

The Hercules "Reference Summary" booklet lists all the system parameters, device definitions, console commands, Hercules utilities etc. along with their arguments.

This booklet is intended as a quick reference guide for experienced users. Consult the Hercules "User Reference Guide" and “Utilities Guide” for more detailed and additional information.

3. Summary of changes

3.1 Version 4, First Edition (HEUR040000-00)

This section describes briefly the various changes that have been made in the “User Reference Guide” related to the previous edition. The most significant changes made in this edition of the manual are the following:

- Chapter 2 (Related Publications): New manual “Operations and Utilities Guide” added.
- Chapter 3 (Summary of changes) added.
- Chapter 4 (Hercules Configuration File): New sections for the following system parameters added:
    - ARCHLVL (Set architecture level)
    - AUTOINIT (Automatic creation of empty tape files)
    - CAPPING (CPU capping feature)
    - CMDLEVEL (Set current command group)
    - CMDSEP (Command line separator)
    - CP_UPDT (User character conversion table)
    - CPUIDFMT (Set format BASIC / 0 / 1 STIDP generation)
    - DEFSTORE (Define main and expanded storage)
    - HAO (Hercules Automatic Operator)
    - HTTP (HTTP server configuration)
    - MAXRATES (MIPS/SIO rate reporting interval)
    - MEMLOCK (Lock Hercules memory)
    - MSGHLD (Timeout value of held messages)
    - MSGLEVEL (Message display output)
    - QUITMOUT (Quit timeout value)
    - SCPECHO (Echo to console and history of SCP replies)
    - SCPIMPLY (Pass non-Hercules commands to the SCP)
    - SCSIMOUNT (Automatic SCSI tape mounts)
    - SHOWDVOL1 (Enable showing of DASD volsers in device list)
    - SRVPRIO (Server threads priority)
    - SYMPTOM (Instruction trace display option)
- Chapter 6 (Device Definition Descriptions): New section “PTP (MPCPTP/PCPTP6 Channel-to-Channel link)” added.
- Chapter 7 (Hercules Console): New section “Programmed Function Key (PF Key) Support” added.
- Chapter 8 (Console Command Descriptions): New sections for the following console commands added:
    - $LOCATE (Display and verify Hercules control blocks)

- $TEST (Custom test command)
- $ZAPCMD (Enable or disable system parameters and console commands)
- ABS (Display or alter absolute storage)
- ARCHLVL (Set architecture level)
- AUTOINIT (Display or set the automatic creation of empty tape files)
- CACHESTATS (Display cache statistics)
- CAPPING (Display or set CPU capping value)
- CMDLEVEL (Display or set current command group)
- CMDSEP (Display or set current command line separator)
- CNSLPORT (Display or set console port)
- CODEPAGE (Display or set codepage conversion table)
- CP_UPDT (Create or modify user character conversion table)
- CPUIDFMT (Display or set format BASIC / 0 / 1 STIDP generation)
- CPUMODEL (Display or set CPU model number)
- CPUPRIO (Display or set CPU thread process priority)
- CPUSERIAL (Display or set CPU serial number)
- CPUVERID (CPU version code)
- DEFSTORE (Display or define main and expanded storage values)
- DEVPRIO (Display or set device threads process priority)
- DIAG8CMD (Display or set DIAGNOSE 8 command option)
- DIR (Display file and directory listing)
- ENGINES (Set processor engines type)
- EXEC (Execute a REXX script)
- FCB (Display current FCB or load new FCB image)
- HERCPRIO (Display or set Hercules process priority)
- HTTP (Start, stop, modify and display HTTP server)
- ICOUNT (Display individual instruction counts)
- KD (Clear held messages)
- LEGACYSENSEID (Display or set SENSE ID CCW (x'E4') feature)
- LS (Display file and directory listing)
- MAINSIZE (Display or set main storage size)
- MANUFACTURER (Display or set STSI manufacturer code)
- MAXCPU (Display or set maximum number of CPUs)
- MEMLOCK (Lock Hercules memory)
- MODEL (Display or set STSI model code)
- MODPATH (Display or set dynamic load module path)
- MSGLEVEL (Display or set the current message display output)
- MT (Control magnetic tape operation)

- NUMCPU (Display or set number of emulated CPUs)
- NUMVEC (Display or set number of vector facilities)
- PANTITLE (Display or set console window title)
- PGMPRDOS (Set LPP license setting)
- PLANT (Display or set STSI plant code)
- PTP (Enable / disable PTP debugging)
- QCPUID (Display CPU ID)
- QPFKEYS (Display the current PF key settings)
- QPID (Display process ID of Hercules)
- QPORTS (Display TCP/IP ports in use)
- QPROC (Display processors type and utilization)
- QSTOR (Display main and expanded storage values)
- QUITMOUT (Display or set quit timeout value)
- SCPECHO (Display or set echo to console and history of SCP replies)
- SCPIMPLY (Display or set option to pass non-Hercules commands to the SCP)
- SHCMDOPT (Display or set shell command option)
- SHRDPORT (Set shared device server port)
- SRVPRIO (Display or set server threads process priority)
- SYMPTOM (Instruction trace display option)
- SYSEPOCH (Set base date for TOD clock)
- TODPRIO (Display or set timer thread process priority)
- TZOFFSET (Set TOD clock offset from GMT)
- XPNDSIZE (Display or set expanded storage size)
- YROFFSET (Set TOD clock offset from actual date)

- The description of the Hercules Utilities has been moved to the “Operations and Utilities Guide”.
- Chapter 12 (The Run-Commands File) added.
- Chapter 13 (The “Hercules Automatic Operator” (HAO) Facility) added.
- Chapter 14 (REXX Support) added.
- Appendix D. Hercules Command Groups added.
- Appendix E. Build Options for System Parameters and Console Commands added.
- Appendix F. Environment Variables added.
- Appendix H. Links: List of links updated.
- Many additional examples have been added.
- Most output samples from console commands have been replaced.

4. Hercules Configuration File

4.1 The Configuration File

This chapter describes the configuration file for the Hercules Emulator. By default the configuration file is named “hercules.cnf” or “hercules.conf”. It is located in the configuration directory, refer to the Hercules Installation Guide for details of the directory structure. The configuration file contains the processor definitions in the system parameters section, the device layout in the device definitions section and Hercules runtime parameters also in the system parameters section.

The configuration file - especially the device definition part - is roughly equivalent to the IOCDS definitions found on a real mainframe environment. It is an ASCII text file that is read and interpreted by the Hercules Emulator during its initialization phase.

4.2 System Parameters

The system parameters describe the processor definition and some Hercules internal runtime parameters.

The processor definitions include the processor model, CPU serial number, memory configuration and architecture mode. They describe the processor hardware that is to be emulated.

The Hercules runtime parameters define values that Hercules requires as an emulated environment. These parameters include values like port numbers, directory paths and priorities.

The following table shows an overview of all valid system parameters. Please note that the availability of certain system parameters depends on the build options used when Hercules was compiled. For a list of all build options and the related system parameters please consult “Appendix E. Build Options for System Parameters and Console Commands”.

System Parameter	Description
#	Comment line
*	Comment line
ARCHLVL	Set architecture level
ARCHMODE	Set initial architecture mode (alias for ARCHLVL system parameter)
ASN_AND_LX_REUSE / ALRF	ESAME ASN and LX REUSE feature (deprecated, use ARCHLVL instead)
AUTO_SCSI_MOUNT	Automatic SCSI tape mounts (deprecated, use SCSIMOUNT instead)
AUTOINIT	Automatic creation of empty tape files
AUTOMOUNT	Tape automount root directory
CAPPING	CPU capping feature

System Parameter	Description
CCKD	Compressed CKD DASD options
CMDLEVEL	Set command group
CMDLVL	Set command group (alias for CMDLEVEL)
CMDSEP	Command line seperator
CNSLPORT	Console port
CODEPAGE	Codepage conversion table
CONKPALV	Console and telnet clients keep-alive option
CP_UPDT	User character conversion table
CPUIDFMT	Set format BASIC / 0 / 1 STIDP generation
CPUMODEL	CPU model number
CPUPRIO	CPU thread process priority
CPUSERIAL	CPU serial number
CPUVERID	CPU version code
DEFSTORE	Define main and expanded storage
DEFSYM	Define a symbol
DEVPRIO	Device threads process priority
DEVTMAX	Maximum number of device threads
DIAG8CMD	DIAGNOSE 8 command option
ECPSVM	ECPS:VM support status (VM)
ENGINES	Processor engines type
HAO	Hercules Automatic Operator
HERCLOGO	Hercules logo file
HERCPRIO	Hercules process priority
HTTP	HTTP server configuration
HTTPPORT	HTTP server port (deprecated, use HTTP instead)
HTTPROOT	HTTP server root directory (deprecated, use HTTP instead)

System Parameter	Description
IGNORE	Ignore subsequent INCLUDE errors
INCLUDE	Include configuration file
IODELAY	I/O interrupt wait time (LINUX)
LDMOD	Additional dynamic load modules
LEGACYSENSEID	SENSE ID CCW (x’E4’) feature
LOADPARM	IPL parameter
LOGOPT	Logging options
LPARNAME	LPAR name returned by DIAG x'204'
LPARNUM	LPAR identification number
MAINSIZE	Main storage size
MANUFACTURER	STSI manufacturer code
MAXCPU	Maximum number of CPUs
MAXRATES	MIPS/SIO rate reporting interval
MEMLOCK	Lock Hercules memory
MODEL	STSI model code
MODPATH	Dynamic load module path
MOUNTED_TAPE_REINIT	Control tape initialization
MSGHELD	Timeout value of held messages
MSGLEVEL	Message display output
MSGLVL	Message display output (alias for MSGLEVEL)
NUMCPU	Number of emulated CPUs
NUMVEC	Number of vector facilities
OSTAILOR	Tailor trace information for specific oprating system
PANRATE	Console refresh rate
PANTITLE	Console window title
PGMPRDOS	LPP license setting

System Parameter	Description
PLANT	STSI plant code
QUITMOUT	Quit timeout value
REXX	REXX interpreter settings
SCLPROOT	SCLP base directory
SCPECHO	Echo to console and history of SCP replies
SCPIMPLY	Pass non-Hercules commands to the SCP
SCSIMOUNT	Automatic SCSI tape mounts
SHCMDOPT	Shell command option
SHOWDVOL1	Enable showing of DASD volsers in device list
SHRDPORT	Shared device server port
SRVPRIO	Server threads priority
SYMPTOM	Instruction trace display option (alias for TRACEOPT)
SYSEPOCH	Base date for TOD clock
TIMERINT	Internal timer update interval
TODDRAG	TOD clock drag factor
TODPRIO	Timer thread process priority
TRACEOPT	Instruction trace display option
TZOFFSET	TOD clock offset from GMT
XPNDSIZE	Expanded storage size
YROFFSET	TOD clock offset from actual date

Table 1: Hercules System Parameters

4.3 Device Definitions

The device definitions describe the layout of the emulated hardware i.e: DASD, tape, terminals and printers. It is comparable with the IOCDS found in a real mainframe environment. Each emulated device must be specified in this part of the configuration file.

The following table shows an overview of all valid devices:

Device Type	Device	Emulated by
3270, 3278	Local non-SNA 3270 display or printer	TN3270 client connection
SYSG	Integrated 3270 (SYSG) console	TN3270 client connection
1052, 3215	Console printer-keyboards	Telnet client connection
1052-C, 3215-C	Integrated console printer-keyboards	Integrated on Hercules console
1442, 2501, 3505	Card readers	Disk file(s), ASCII or EBCDIC
3525	Card punch	Disk file, ASCII or EBCDIC
1403, 3211	Line printers	Disk file, ASCII
3410, 3420, 3422, 3430, 3480, 3490, 3590, 9347, 8809	Tape drives	Disk file, CD-ROM or SCSI tape
3088	Channel-to-Channel Adapter	“CTCT” driver
((CTCI))	Channel-to-Channel link to host TCP/IP stack	“CTCI” TUN/TAP driver
((LCS))	IBM 2216 router, IBM 3172 running ICP, IBM 8232 LCS device, LCS3172 driver of a P/390, IBM Open Systems Adapter (OSA)	“LCS” (LAN channel station) TUN/TAP driver
((QETH))	OSA Express IP Layer 2 support only. Supported only for Linux guests. TAP adapter must be bridged to a local LAN	“QETH” (OSA/QDIO Ethernet Adapter) TUN/TAP driver
3310, 3370, 9313, 9332, 9335, 9336, 0671	FBA direct access storage devices	Disk file
2305, 2311, 2314, 3330, 3340, 3350, 3375, 3380, 3390, 9345	CKD direct access storage devices	Disk file
2703	Communication line	TCP socket

Table 2: Hercules Device Definitions

4.4 Coding Rules

There are only a few rules for creating configuration files. The file must be an ASCII text file. Blank lines, or lines beginning with a hash ("#") sign or an asterisk ("*"), are treated as comments and are not checked.

Except for the ARCHLVL stetements, the system parameters may appear in any order but must precede any device definitions. Each system parameter must be coded on a separate line.

The device records may also appear in any order but must follow the system parameters. There must be one device definition for each I/O device or for each group of identical I/O devices.

4.5 Record Format

The following sections describe the format of the Hercules configuration file records. The format is slightly different between system parameters and device definitions.

4.5.1 Format of the system parameters

The record format of the system parameters is as follows:

system-parameter argument

where:

system-parameter is the name of the system parameter.

argument is the value assigned to the system parameter.

Examples:

```
CPUMODEL  3090
ENGINES   4*CP,2*AP,2*IP,2*IL
HTTP      ROOT /usr/local/share/hercules/
PANRATE   FAST
```

4.5.2 Format of the device definitions

The format of device definition is as follows:

devnum(s) devtype [argument(s)]

where:

devnum(s) is either: a single devnum (see details below), a range of devnums separated by a dash ("-"), a count of devnums seperated by a dot ("."), or a comma (",") separated list of devnums.

Examples include:

- 0200-0210
- 0300.10

- 0400,0410
- 0100,0110-011F

If devnums specifies more than one device then all devices will have identical characteristics. All devices defined as a group must be defined on a single channel. A channel is defined as a continuous group of 256 (or hexadecimal 100) devices. For example devices 0010 and 0020 are on the same channel, whereas devices 0100 and 0210 are not on the same channel.

The devnum itself is either a 1 to 4 digit hexadecimal number in the range 0000 to FFFF for ESA/390 or 0000 to 0FFF for S/370. The device number uniquely identifies each device to the operating system.

devtype is the device type.

argument(s) is a list of parameters depending on the device type. These parameters are explained in the sections that describe each device type.

Examples:

```
0120       3380  mvsv5r.120
0230.16    3270  GROUP1 192.168.100.0 255.255.255.0
0583-0587  3420  *  maxsizeM=170  eotmargin=131072
```

4.6 Sample Configuration File

The following figure shows a real example of a Hercules configuration file, used for running the MVS 3.8J operating system. Please note, that not all possible parameter and definitions are contained in this sample.

```
# *************************************************************************************************
# Hercules V4.00.0 Emulator Control file
# MVS V3.8J
# *************************************************************************************************

# =================================================================================================
# System parameters
# =================================================================================================

ARCHLVL             S/370                        # Initial architecture mode
CAPPING             25                           # CPU capping (MIPS)
CNSLPORT            3270                         # Telnet client port
CODEPAGE            DEFAULT                      # Codepage conversion table
CPUMODEL            7060                         # CPU model
CPUSERIAL           001963                       # CPU serial number
DEFSYM              DASDPATH    D:/MVS/DASD      # Define symbol for CCKD DASD device path
DEFSYM              READERPATH  D:/MVS/READER    # Define symbol for READER device path
DEFSYM              PRINTPATH   D:/MVS/PRINTER   # Define symbol for PRINTER device path
DEFSYM              PUNCHPATH   D:/MVS/PUNCH     # Define symbol for PUNCH device path
DEVTMAX             0                            # Maximum number of device threads
ENGINES             CP                           # Processor engine types
HERCLOGO            HERCLOGO.TXT                 # Hercules 3270 logo file
HTTP                PORT 8089 NOAUTH             # HTTP server port / authorization data
```

```
HTTP                 ROOT D:\Hercules\html        # HTTP server root directory
HTTP                 START                        # Start the HTTP server
LOGOPT               TIMESTAMP                    # Hercules log options
LPARNAME             HERCULES                     # LPAR name returned by DIAG X'204'
MAINSIZE             16                           # Main storage in MB
MANUFACTURER         HRC                          # Manufacturer name returned by STSI
MAXCPU               1                            # Maximum number of CPUs
MODEL                7060                         # Model name returned by STSI
NUMCPU               1                            # Number of emulated CPUs
OSTAILOR             OS/390                       # Intended operating system
PANRATE              1000                         # Panel refresh rate [ms]
PANTITLE             "Hercules Emulator HMC"      # Hercules HMC window title
PGMPRDOS             RESTRICTED                   # Emulation of IFL hardware
PLANT                ZZ                           # Plant name returned by STSI
SYSEPOCH             1900 +28                     # Base date for TOD clock
TIMERINT             50                           # Internal timer update interval
TRACEOPT             TRADITIONAL                  # Instruction trace display option
TZOFFSET             +0000                        # TOD clock offset from GMT

# ===============================================================================================
# Device Definitions
# ===============================================================================================

# -----------------------------------------------------------------------------------------------
# Card Readers
# -----------------------------------------------------------------------------------------------

000C  3505  $(READERPATH)/DUMMY.JCL  eof  ascii  trunk

# -----------------------------------------------------------------------------------------------
# Card Punches
# -----------------------------------------------------------------------------------------------

000D  3525  $(PUNCHPATH)/PCH00D.TXT  ascii  crlf

# -----------------------------------------------------------------------------------------------
# Line Printers
# -----------------------------------------------------------------------------------------------

000E  1403  $(PRINTPATH)/PRT00E.TXT  crlf
000F  1403  $(PRINTPATH)/PRT00F.TXT  crlf

# -----------------------------------------------------------------------------------------------
# Local Non-SNA Terminals
# -----------------------------------------------------------------------------------------------

0010  3270  *  192.168.0.101
0011  3270  *  192.168.0.101
00C0  3270  *  192.168.0.101
00C1  3270  *  192.168.0.101
00C2  3270  *  192.168.0.101
```

```
# ----------------------------------------------------------------------------------------------
# CCKD DASD Devices
# ----------------------------------------------------------------------------------------------

0130  2314  $(DASDPATH)/SORT00.CCKD
0131  2314  $(DASDPATH)/SORT01.CCKD
0132  2314  $(DASDPATH)/SORT02.CCKD
0133  2314  $(DASDPATH)/SORT03.CCKD
0134  2314  $(DASDPATH)/SORT04.CCKD
0135  2314  $(DASDPATH)/SORT05.CCKD
0140  3350  $(DASDPATH)/WORK00.CCKD
0141  3350  $(DASDPATH)/WORK01.CCKD
0142  3350  $(DASDPATH)/PRD000.CCKD
0148  3350  $(DASDPATH)/MVSRES.CCKD
0149  3350  $(DASDPATH)/MVSDLB.CCKD
0160  3340  $(DASDPATH)/PAGE00.CCKD
0161  3340  $(DASDPATH)/PAGE01.CCKD
0240  3350  $(DASDPATH)/PUB000.CCKD
0248  3350  $(DASDPATH)/SYS000.CCKD
0344  3350  $(DASDPATH)/SPOOL0.CCKD
0345  3350  $(DASDPATH)/SPOOL1.CCKD
0348  3350  $(DASDPATH)/TST000.CCKD

# ----------------------------------------------------------------------------------------------
# TAPE Devices
# ----------------------------------------------------------------------------------------------

0480  3420  *
0481  3420  *

# ==============================================================================================
# (EOF)
# ==============================================================================================
```

Figure 1: Sample Configuration File

4.7 Symbol Substitution

In configuration statements, as well as in console commands and OAT files, symbols may be substituted for text. To substitute symbol *symbol* with its contents the symbol should be enclosed within parenthesis or braces and preceded by a $ sign. For example, if symbol "*foo"* contains the text string *"bar"* then "$(foo)" or "${foo}" will be substituted with the string *"bar"*.

Symbols are defined using a DEFSYM configuration statement or a DEFSYM panel command or can be operating system environment variables. There are several predefined symbols (see table below). To resolve the symbol substitution a symbol with that name defined via DEFSYM is searched first. If none can be found then a check is made to see if an environment variable with the same name exists.

If a symbol is not defined then an empty string will be substituted. By coding an additional default value following an equal sign (“=”) or a colon-equal-sign (":=") the default value is substituted instead of an empty string if the symbol is not defined.

Note that the default value is a literal string and no substitution is applied to it. Thus attempting to use the syntax "${foo=${bar}}" will not yield the expected results. It will not be substituted with the currently defined value of the "bar" environment variable, but rather by the literal string "${bar" followed immediately by the literal character '}'.

Symbols created with DEFSYM or environment variables can also be used as part of panel commands. For example the command "cd $(TAPEDIR)" will change the current directory to the resolved string for the symbol TAPEDIR.

It is important to note that symbol names, potentially being the names of environment variables, are subject to whatever case sensitivity the host operating system happens to enforce or allow. Under Windows environment variables are not case sensitive, whereas on other operating systems they may be. Thus "$(FOO)", "$(foo)" or "$(Foo)" all cause the same value to be substituted on Windows, whereas they could be substituted with completely different values under a case sensitive operating system.

4.7.1 Syntax

Descriptive

`$(`*`symbol`*`[[:]=`*`default`*`])`

`or`

`${`*`symbol`*`[[:]=`*`default`*`]}`

Diagram

```
►►── $ ──┬── ( ──┬── symbol ──┬──────────────────┬──┬── ) ──┬──►◄
         └── { ──┘            ├── :=default ─────┤  └── } ──┘
                              └── =default ──────┘
```

4.7.2 Parameter

symbol This is the name of the symbol.

default This is the default value that takes place if the symbol is not defined.

4.7.3 Special symbols

4.7.3.1 Predefined symbols

The symbols according to the following table are predefined and can be used without defining the symbol through a corresponding DEFSYM configuration statement or panel command.

Symbol name	Assigned value	Example string
BDATE	Hercules build date	‘Sep 22 2010’
BTIME	Hercules build time	’19:13:03’
CUU	3 digit device number, upper case hexadecimal digits	‘12A’
CUUU	4 digit device number, upper case hexadecimal digits	‘012A’
cuu	3 digit device number, lower case hexadecimal digits	‘12a’
cuuu	4 digit device number, lower case hexadecimal digits	‘012a’
DEVN	4 digit device number, upper case hexadecimal digits	‘012A’
HOSTARCH	Host system architecture	‘i686’
HOSTNAME	Name of the host system	‘GOOFY’
HOSTNUMCPUS	The number of CPUs of the host system	‘MP=8’
HOSTOS	Host operating system	‘Windows’
HOSTOSREL	Host operating system release	‘6.1.7600’
HOSTOSVER	Host operating system version	‘7 Ultimate Edition, 64-bit’
MODNAME	Module name of the startup program	‘hercules.exe’
MODPATH	Path name of the startup program	‘D:\Hercules\’
VERSION	Hercules version	‘3.07’

Table 3: Predefined symbols

4.7.3.2 Environment variables

If a symbol is not explicitly defined by a DEFSYM statement and an environment variable by the same name exists, then the string contents of that environment variable will be used for substitution.

4.7.3.3 Undefined symbols

If a symbol is not defined by an explicit DEFSYM, is not an automatically generated symbol or is not an environment variable, an empty string will be substituted.

4.7.4 Escaping substitution, recursion

To specify the '$’ string without incurring substitution an additional '$' sign should be used. For example, $$(FOO) will not be substituted. If substitution is required but the preceding text is to contain a '$' sign as the very last character then $$$(FOO) would be specified. Thus if symbol FOO contains "BAR", then $$(FOO) will remain "$$(FOO)" while $$$(FOO) will become "$BAR". Substitution is not recursive, only one substitution pass is made.

4.7.5 Examples

Example 1:

The DEFSYM configuration statement

```
DEFSYM TAPEDIR "/home/hercules/tapes"
```

and the symbol substitution in the device definition

```
0380 3420 $(TAPEDIR)/scratch.aws
```

results in the following device definition used by Hercules

```
0380 3420 /home/hercules/tapes/scratch.aws
```

Example 2:

The DEFSYM configuration statement

```
DEFSYM DASDPATH "D:/HERCULES/DASD"
```

and the following symbol substitution in the device configuration

```
0148 3350 $(DASDPATH:=D:/MVS/DASD)/MVSRES.CKD
```

results in the following device definition with resolved DASDPATH symbol

```
0148 3350 D:/HERCULES/DASD/MVSRES.CKD
```

If however there is no symbol DASDPATH (the DEFSYM statement is missing) then the symbol in the device definition is resolved with the defined default

```
0148 3350 D:/MVS/DASD/MVSRES.CKD
```

Example 3:

The device definition using the predefined symbol "CUUU"

```
0148.5 3350 D:/MVS/DASD/VOLSER.$(CUUU)
```

results in the following device definitions used by Hercules

```
0148 3350 D:/MVS/DASD/VOLSER.0148
0149 3350 D:/MVS/DASD/VOLSER.0149
014A 3350 D:/MVS/DASD/VOLSER.014A
014B 3350 D:/MVS/DASD/VOLSER.014B
014C 3350 D:/MVS/DASD/VOLSER.014C
```

5. System Parameter Descriptions

5.1 # (Comment line)

5.1.1 Function

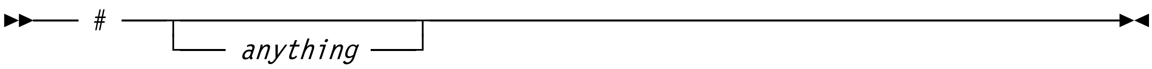
The hash (“#”) symbol marks the beginning of a comment or a full comment line.

5.1.2 Syntax

Descriptive

`# [anything]`

Diagram



5.1.3 Parameter

anything Any text following the asterisk symbol.

5.1.4 Examples

Example 1:

Add comment lines to the configuration file and add comment after the system parameter.

```
#
#                System Parameters
#

NUMCPU    2      # Number of emulated CPUs
MAINSIZE 1024   # Main storage in MB
```

5.2 * (Comment line)

5.2.1 Function

The asterisk (“*”) symbol marks the beginning of a comment or a full comment line.

5.2.2 Syntax

Descriptive

`* [`*`anything`*`]`

Diagram

```
►►── * ──┬────────────┬──────────────────────────────────────►◄
         └─ anything ─┘
```

5.2.3 Parameter

anything Any text following the hash symbol.

5.2.4 Examples

Example 1:

Add comment lines to the configuration file and add comment after the system parameter.

```
*  ┌──────────────────────────────────────────┐
*  │           System Parameters              │
*  └──────────────────────────────────────────┘
NUMCPU    2      * Number of emulated CPUs
MAINSIZE 1024    * Main storage in MB
```

5.3 ARCHLVL (Set architecture level)

5.3.1 Function

The ARCHLVL system parameter specifies the initial architecture mode. Additional ARCHLVL statements may be specified to enable or disable specific facilities, to override the STFLE (Store Facility List Extended) response and force it to return a certain (incorrect) bit pattern or to display the status (enabled or disabled) of facilities during startup. An overview of all architecture facilities can be found in “Appendix C. Architecture Facilities” on page 643.

5.3.2 Syntax

Descriptive

```
ARCHLVL {S/370 | ESA/390 | ESAME | z/ARCH}

   or

ARCHLVL {ENABLE | DISABLE} facility [S/370 | ESA/390 | ESAME | z/ARCH]

   or

ARCHLVL QUERY [facility | ALL]

   or

ARCHLVL {ENABLE | DISABLE} bitno [S/370 | ESA/390 | ESAME | z/ARCH]
```

Diagram

```
►►── ARCHLVL ──┬── S/370 ───┬──────────────────────────────────────────►◄
               ├── ESA/390 ─┤
               ├── ESAME ───┤
               └── z/ARCH ──┘

   or

►►── ARCHLVL ──┬── ENABLE ──┬── facility ──┬─────────────────┬─────────►◄
               └── DISABLE ─┘              ├── S/370 ────────┤
                                           ├── ESA/390 ──────┤
                                           ├── ESAME ────────┤
                                           └── z/ARCH ───────┘

   or

                              ┌──── ALL ──────┐
►►── ARCHLVL ── QUERY ────────┴── facility ───┴────────────────────────►◄
```

```
  or

►►── ARCHLVL ──┬── ENABLE ──┬── bitno ──┬─────────────┬──────────►◄
               └── DISABLE ─┘           ├── S/370 ────┤
                                        ├── ESA/390 ──┤
                                        ├── ESAME ────┤
                                        └── z/ARCH ───┘
```

5.3.3 Parameter

S/370 Use S/370 for OS/360, VM/370 and MVS 3.8.

ESA/390 Use ESA/390 for MVS/XA, MVS/ESA, OS/390, VM/ESA, VSE/ESA, Linux/390 and ZZSA. zOS can be run until version 1.2 with ESA/390 mode without installed bimodal feature or until version 1.4 if the bimodal feature is installed.

ESAME Use ESAME (Enterprise System Architecture, Modal Extensions) for z/OS, z/VM, z/VSE and z/Linux. The ESAME mode is equivalent to z/Archtecture mode at at Architecture Level 2.

When ESAME is specified, the machine will always be IPL'ed in ESA/390 mode but the system is capable of being switched into the z/Architecture mode after IPL. This is handled automatically by all z/Architecture operating systems.

z/ARCH Use z/ARCH for z/OS, z/VM, z/VSE and z/Linux. z/ARCH is similar to ESAME. The z/Arch mode is equivalent to z/Architecture mode at Architecture Level 3.

ENABLE Enable the specified facility. If no architecture mode is given as additional parameter the facility is enabled for all architecture modes.

DISABLE Disable the specified facility. If no architecture mode is given as additional parameter the facility is disabled for all architecture modes.

facility The name of the facility that has to be enabled, disabled or displayed. An overview of all architecture facilities can be found in "Appendix C. Architecture Facilities" on page 643.

bitno The bit number of the facility that has to be enabled or disabled. The format of *bitno* is BIT*nn*, e.g. "BIT44" (Bit 44 = PFPO Facility Bit). An overview of all architecture facilities can be found in "Appendix C. Architecture Facilities" on page 643.

QUERY Display the settings for the specified facility.

ALL Specifies that all facilities have to be displayed.

5.3.4 Examples

Example 1:

Set the architecture level to S/370.

```
ARCHLVL S/370
```

Example 2:

Disable the ‘DECIMAL_FLOAT’ facility in z/Architecture mode.

```
ARCHLVL DISABLE DECIMAL_FLOAT z/ARCH
```

Example 3:

Force the PFPO feature to be enabled in all architecture modes.

```
ARCHLVL ENABLE BIT44
```

Example 4:

Display the facility settings during the Hercules startup messages.

```
ARCHLVL QUERY ALL
```

5.4 ARCHMODE (Initial architecture mode)

5.4.1 Function

The ARCHMODE parameter has been deprececated. This parameter was used to specifies the initial architecture mode.

ARCHMODE is still accepted and is treated as an alias for the new ARCHLVL system parameter. All existing ARCHMODE statements should be changed to ARCHLVL. Please see ARCHLVL for details.

5.4.2 Syntax

See ARCHLVL system parameter.

5.4.3 Parameter

See ARCHLVL system parameter.

5.4.4 Examples

See ARCHLVL system parameter.

5.5 ASN_AND_LX_REUSE / ALRF (ESAME ASN and LX REUSE feature)

5.5.1 Function

The ASN_AND_LX_REUSE (ALRF) system parameter has been deprecated and is replaced by the ARCHLVL system parameter. Use “ARCHLVL ENABLE | DISABLE ASN_LX_REUSE” instead. See ARCHLVL system parameter for details.

5.5.2 Syntax

See ARCHLVL system parameter.

5.5.3 Parameter

See ARCHLVL system parameter.

5.5.4 Examples

See ARCHLVL system parameter.

5.6 AUTO_SCSI_MOUNT (Automatic SCSI tape mounts)

5.6.1 Function

The AUTO_SCSI_MOUNT system parameter has been deprecated and is replaced by SCSIMOUNT. See SCSIMOUNT system parameter for details.

5.6.2 Syntax

See SCSIMOUNT system parameter.

5.6.3 Parameter

See SCSIMOUNT system parameter.

5.6.4 Examples

See SCSIMOUNT system parameter.

5.7 AUTOINIT (Automatic creation of empty tape files)

5.7.1 Function

The AUTOINIT system parameter controls the automatic creation of empty tape files. AUTOINIT handles the "file not found" condition for a specified tape file. It controls if DEVINIT returns a "file not found" error or creates an empty tape file if the tape file could not be found. The default for AUTOINIT is OFF.

5.7.2 Syntax

Descriptive

```
AUTOINIT {ON | OFF}
```

Diagram

```
►►── AUTOINIT ──┬── ON ──┬──────────────────────────────────────►◄
                └── OFF ─┘
```

5.7.3 Parameter

ON When AUTOINIT is set ON, DEVINIT will initialize a blank, non-labeled tape if the specified tape file is not found. Next, DEVINIT writes two tapemarks, rewinds the tape and positions the tape to the beginning.

OFF When AUTOINIT is set OFF (which is the default), DEVINIT will return a "file not found" error if the specified tape file is not found.

5.7.4 Examples

Example 1:

Switch on the automatic creation of empty tape files.

```
AUTOINIT ON
```

5.8 AUTOMOUNT (Tape automount root directory)

5.8.1 Function

Specifies the host system directory where the guest is allowed or not allowed to automatically load virtual tape volumes from. Prefix allowable directories with a '+' (plus) sign and unallowable directories with a '-' (minus) sign. The default prefix if neither is specified is the '+' sign (an allowable directory).

All host system virtual tape volumes to be automounted by the guest must reside within one of the specified allowable host system directories or any of its subdirectories while not also being within any of the specified unallowable directories or any of their subdirectories, in order for the guest-invoked automount to be accepted.

Specifying a disallowed automount directory does not preclude the Hercules operator from manually mounting any file via the DEVINIT console command - even one in a currently defined disallowed automount directory. The AUTOMOUNT statement only controls guest-invoked automatic tape mounts and not manual tape mounts performed by the Hercules operator.

All directories must be specified on separate statements, but as many statements as needed may be specified in order to describe the desired allowable/unallowable directories layout. For convenience, an AUTOMOUNT console command is also provided to dynamically add new or remove existing automount directories at any time.

The automount feature is activated whenever you specify at least one allowable or unallowable directory. If only unallowable directories are specified, then the current directory becomes the only defined allowable automount directory by default.

All specified directories are always resolved to fully-qualified absolute directory paths before being saved.

Caution: Enabling this feature may have security consequences depending on which allowable host system directories you specify as well as how your guest operating system enforces authorized use of the Set Diagnose (X'4B') channel command code.

Refer to the description of the virtual tape device 'NOAUTOMOUNT' option for more information.

5.8.2 Syntax

Descriptive

AUTOMOUNT [<u>+</u> | -]*directory*

or

AUTOMOUNT {ADD *directory* | DEL *directory*}

Diagram

or

```
►►── AUTOMOUNT ──┬── ADD directory ──┬──────────────────────────────►◄
                 └── DEL directory ──┘
```

5.8.3 Parameter

ADD or + Add an entry to the list of allowable tape automount directories.

DEL or - Delete an entry from the list of allowable tape automount directories.

directory Specifies the host system directory where the guest is allowed or not to automatically load virtual tape volumes from.

5.8.4 Examples

Example 1:

Specify directory “D:\MVS\TAPE” as the host system directory from where the guest is allowed to automatically load virtual tape volumes.

```
AUTOMOUNT +D:/MVS/TAPE
   or
AUTOMAOUNT ADD D:/MVS/TAPE
```

5.9 CAPPING (CPU capping feature)

5.9.1 Function

The CAPPING system parameter is used to cap the CPUs. If *mips* is greater than zero then all of the ‘CP’ type processors are capped to this value. If *mips* is equal to zero or “OFF” or the CAPPING system parameter is not coded in the configuration file then the capping is disabled.

Only CPUs of type CP are capped. CPUs of type IL, AP or IP are never capped. The CPU string on the Hercules device and status panel which shows the CPU usage turns from white to red during the time the CPU is capped.

5.9.2 Syntax

Descriptive

`CAPPING {`*`mips`*` | `<u>`OFF`</u>` | 0}`

Diagram

5.9.3 Parameter

mips Maximum total number of MIPS for all the ‘CP’ type processors.

OFF Disables the CPU capping.

0 This is the same as OFF.

5.9.4 Examples

Example 1:

Set the CPU capping to 25 MIPS.

```
CAPPING 25
```

Example 2:

Disable the CPU capping feature.

```
CAPPING OFF
```

5.10 CCKD (Compressed CKD DASD options)

5.10.1 Function

The CCKD system parameter is used to alter CCKD processing. The CCKD system parameter supports the same options as the CCKD console command.

5.10.2 Syntax

Descriptive

CCKD *option=value* [,*option=value* [,*option=value* ...]]

where option can be:

[COMP={-1 | *n*}]

[,COMPPARM={-1 | *n*}]

[,RA={2 | *n*}]

[,RAQ={4 | *n*}]

[,RAT={2 | *n*}]

[,WR={2 | *n*}]

[,GCINT={10 | *n*}]

[,GCPARM={0 | *n*}]

[,NOSTRESS={0 | 1}]

[,FREEPEND={-1 | *n*}]

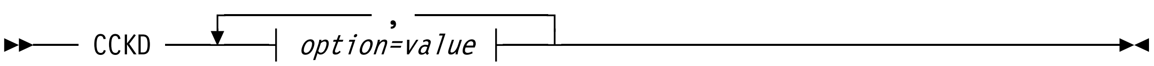
[,FSYNC={0 | 1}]

[,TRACE={0 | *n*}]

[,LINUXNULL={0 | 1}]

[,GCSTART={0 | 1}]

Diagram



where option can be:

```
├── COMP= ──────┬─ -1 ─┬────────────────────────────────┤
                └─ n ──┘

├── COMPPARM= ──┬─ -1 ─┬────────────────────────────────┤
                └─ n ──┘

├── RA= ────────┬─ 2 ──┬────────────────────────────────┤
                └─ n ──┘

├── RAQ= ───────┬─ 4 ──┬────────────────────────────────┤
                └─ n ──┘

├── RAT= ───────┬─ 2 ──┬────────────────────────────────┤
                └─ n ──┘

├── WR= ────────┬─ 2 ──┬────────────────────────────────┤
                └─ n ──┘

├── GCINT= ─────┬─ 10 ─┬────────────────────────────────┤
                └─ n ──┘

├── GCPARM= ────┬─ 0 ──┬────────────────────────────────┤
                └─ n ──┘

├── NOSTRESS= ──┬─ 0 ──┬────────────────────────────────┤
                └─ 1 ──┘

├── FREEPEND= ──┬─ -1 ─┬────────────────────────────────┤
                └─ n ──┘

├── FSYNC= ─────┬─ 0 ──┬────────────────────────────────┤
                └─ 1 ──┘

├── TRACE= ─────┬─ 0 ──┬────────────────────────────────┤
                └─ n ──┘

├── LINUXNULL= ─┬─ 0 ──┬────────────────────────────────┤
                └─ 1 ──┘

├── GCSTART= ───┬─ 0 ──┬────────────────────────────────┤
                └─ 1 ──┘
```

5.10.3 Parameter

option Set a CCKD option. Multiple options may be specified, separated by commas with no intervening blanks.

The CCKD options are:

COMP=*n* Specifies the compression type to be used. This overrides the compression used for all CCKD files. The default (-1) means don't override the compression. Valid compression types are:

-1 Default

0 None

1 Zlib

2 Bzip2

COMPPARM=*n* Overrides the compression parameter. A higher value generally means more compression at the expense of CPU and/or storage. The default (-1) means don't override the compression parameter. The value of *n* can be from -1 and 9.

RA=*n* Sets the Number of read ahead threads. When sequential track or block group access is detected, some number (RAT=n) of tracks or block groups are queued (RAQ=n) to be read by one of the read ahead threads. The default is 2, the value of *n* can be a number from 1 to 9.

RAQ=*n* Sets the size of the read ahead queue. When sequential track or block group access is detected, some number (RAT=n) of tracks or block groups are queued in the read ahead queue. The default is 4, the value of *n* can be a number from 0 to 16. A value of zero disables read ahead.

RAT=*n* Sets the number of tracks or block groups to read ahead when sequential track or block group access is detected. The default is 2, the value of ratn can be a number from 0 to 16. A value of zero disables read ahead.

WR=*n* Sets the number of writer threads. When the cache is flushed, updated cache entries are marked write pending and a writer thread is signalled. The writer thread compresses the track or block group and writes the compressed image to the emulation file.

A writer thread is CPU-intensive while compressing the track or block group and I/O-intensive while writing the compressed image. The writer thread runs one nicer than the CPU thread(s). The default is 2, a value from 1 to 9 can be specified

GCINT=*n* This is the number of seconds the garbage collector thread waits during an interval. At the end of an interval the garbage collector performs space recovery, flushes the cache and optionally 'fsyncs' the emulation file.

However, the file will not be 'fsynced' unless at least 5 seconds have elapsed since the last synchronization (FSYNC). The default is 10 seconds. You can specify a number between 1 and 60.

GCPARM=*n* A value affecting the amount of data moved during the garbage collectors space recovery routine. The garbage collector determines an amount of space to move based on the ratio of free space to used space in an emulation file and on the number of free spacesin the file. The garbage collector wants to reduce the free space to used space ratio and the number of free spaces.

The value is logarithmic; a value of 8 means moving 2^8 the selected value while a negative value similarly decreases the amount to be moved. Normally, 256K will be moved for a file in an interval. Specifying a value of 8 can increase the amount to 64M. At least 64K will be moved. Specifying a large value (such as 8) may not increase the garbage collection efficiency correspondingly. The default is 0. You can specify a number from -8 to 8.

NOSTRESS=*n* Indicates whether stress writes will occur or not. A track or block group may be written under stress when a high percentage of the cache is pending write or when a device I/O thread is waiting for a cache entry. When a stressed write occurs, the compression algorithm and/or compression parm may be relaxed, resulting in faster compression but usually a larger compressed image.

If NOSTRESS is set to one, then a stressed situation is ignored. You would typically set this value to one when you want create the smallest emulation file possible in exchange for a possible performance degradation. The default is 0. You can specify 0 (enable stressed writes) or 1 (disable stressed writes).

FREEPEND=*n* Specifies the free pending value for freed space. When a track or block group image is written, the space it previously occupied is freed. This space will not be available for future allocations until *n* garbage collection intervals have completed. In the event of a catastrophic failure, previously written track or block group images should be recoverable if the current image has not yet been written to the physical disk.

By default the value is set to -1 which means that if FSYNC is specified then the value is 1 otherwise it is 2. If 0 is specified then freed space is immediately available for new allocations. The default is -1. You can specify a number from -1 to 4.

FSYNC=*n* Enables or disables FSYNC. When FSYNC is enabled then the disk emulation file is synchronized with the physical hard disk at the end of a garbage collection interval (no more often than 5 seconds though).

This means that if FREEPEND is non-zero and a catastrophic error occurs, the emulated disks should be recovered coherently. However, FSYNC may cause performance degradation depending on the host operating system and / or the host operating system level. The default is 0 (fsync disabled), you can specify 0 (disable FSYNC) or 1 (enable FSYNC).

TRACE=*n* Specifies the number of CCKD trace entries. You would normally specify a non-zero value when debugging or capturing a problem in CCKD code. When the problem occurs, you should enter the "k" Hercules console command which will print the trace table entries. The default is 0. You can specify a number between 0 and 200000. Each trace entry represents 128 bytes. Normally, for debugging, it is recommended to use 100000.

LINUXNULL=*n* If set to 1 then tracks written to 3390 CCKD volumes that were initialized with the *-linux* option will be checked if they are null (that is if all 12 4096 byte user records contain zeroes). This is used by the DASDCOPY utility. The default is 0.

GCSTART=*n* If set to 1 then space recovery will become active on any emulated disks that have free space. Normally space recovery will ignore emulated disks until they have been updated. The default is 0.

Notes:

- *raq* should be at least as large as *ra*. Read ahead threads are scheduled from entries in the read ahead queue. Likewise *rat* should not exceed *raq* because only *raq* tracks or block groups can be queued at any time.

- The number of writer threads *wr* should usually be 1 more than the number of host processors. This is because one writer thread could be CPU-bound (compressing a track or block-group image) and the other could be i/o-bound (writing the compressed image).

- The garbage collection interval governs the maximum time in seconds an updated track or block group image will reside in storage before being written to the emulation file. A large value may mean more data loss if a catastrophic error occurs. A small value may mean that more CPU time is spent compressing images.

  For example, suppose that a particular image is updated several times each second. If the interval is changed from the default 5 seconds to 1 second, then that image will be compressed and written 5 times more frequently. A large value may cause more cache flushes within a garbage collection interval. These kinds of flushes mean that a read will wait because there are no available cache entries, slowing the emulated operating system. A large value will also cause more pending free space to build up (since free space is flushed each interval). This may mean that the garbage collector space recovery routine will perform more work and the resulting emulation file may be larger.

- Specify *fsync=1* and *gcint=5* if you are seriously concerned about your data being lost due to a failure. *fsync* will ensure your data on disk is coherent. However, fsync may cause a noticeable performance degradation. Note that an fsync will not be performed more often than every 5 seconds.

5.10.4 Examples

Example 1:

Set the CCKD options to use bzip2 as compression method and using maximum compression.

```
CCKD COMP=2,COMPPARM=9
```

Example 2:

Set the CCKD options to use the default compression method, as well as default compression. Disable stressed writes and set the number of trace entries to 100000.

```
CCKD COMP=-1,COMPPARM=-1,NOSTRESS=1,TRACE=100000
```

5.11 CMDLEVEL (Set current command group)

5.11.1 Function

The CMDLEVEL system parameter sets the current command group(s). A plus sign preceding the command group activates the console commands of this group, whereas a preceding minus sign deactivates the commands.

A table showing the affiliation of each console command to the various command groups can be found in ‘Appendix D. Hercules Command Groups’.

Some console commands are always active to keep Hercules operable, independent of the current active command group. These commands are listed under command group 'NONE' in the above mentioned table. Command group ‘NONE’ is the result of a ‘CMDLVL –ALL’ command.

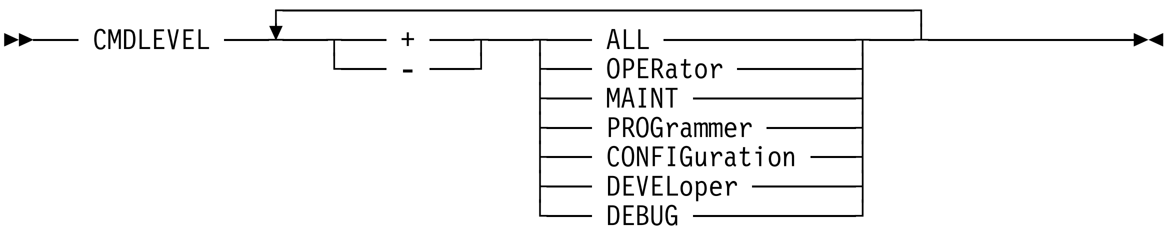
The default command level is set to operator, maintenance, programmer and configuration (which corresponds to the command "CMDLEVEL -ALL +OPER +MAINT +PROG +CONFIG"). Some of the arguments can be abbreviated as shown in the syntax section below.

5.11.2 Syntax

Descriptive

```
CMDLEVEL {{+ | -} {ALL | OPERator | MAINT | PROGrammer | CONFIGuration |
         DEVELoper | DEBUG} [...]}
```

Diagram



5.11.3 Parameter

+ The plus sign activates the commands of the following command group.

- The minus sign deactivates the commands of the following command group.

ALL Command group ‘ALL’ contains all Hercules console commands. Specifying command group ‘+ALL’ enables all console commands whereas command group ‘-ALL’ equals to NONE. This disables all console commands with the exception of commands necessary to keep Hercules operable.

OPER The ‘OPERator’ command group activates or deactivates all system operator commands.

MAINT Command group ’MAINT’ activates or deactivates all system maintainer commands.

PROG The ‘PROGrammer’ command group activates or deactivates all systems programmer commands.

CONFIG Command group ‘CONFIGuration’ activates or deactivates all system configuration commands.

DEVEL The ‘DEVELoper’ command group activates or deactivates all system developer commands.

DEBUG Command group ‘DEBUG’ activates or deactivates all debugging activity commands activities.

5.11.4 Examples

Example 1:

Set the command group to OPERATOR and PROGRAMMER.

```
CMDLEVEL –ALL +OPERATOR +PROGRAMMER
```

Example 2:

Allow all commands but exclude the DEBUG command group.

```
CMDLEVEL +ALL -DEBUG
```

5.12 CMDLVL (Set current command group)

5.12.1 Function

CMDLVL is an alias for CMDLEVEL. The CMDLVL system parameter sets the command group(s). See CMDLEVEL system parameter for details.

5.12.2 Syntax

See CMDLEVEL system parameter.

5.12.3 Parameter

See CMDLEVEL system parameter.

5.12.4 Examples

See CMDLEVEL system parameter.

5.13 CMDSEP (Command line separator)

5.13.1 Function

The CMDSEP system parameter sets the command line separator. The command line separator character is used to separate multiple panel commands on a single line. The default is OFF which means that there is no command line separator defined and therefore multiple panel commands on a single line are not supported.

5.13.2 Syntax

Descriptive

CMDSEP {*char* | OFF}

Diagram

```
►►── CMDSEP ──┬── char ──┬──────────────────────────────►◄
              └── OFF ───┘
```

5.13.3 Parameter

char Specifies a single character that is used for command separation. This character must not be the period ('.'), the exclamation mark ('!') or the hyphen ('-'). Although the command line separation character can be set to the number (hash) sign ('#'), this is not recommended because this could affect processing command lines that contain comments.

OFF OFF disables command separation. This is the default.

5.13.4 Examples

Example 1:

Set the command line separator to ‘;’.

```
CMDSEP ;
```

Example 2:

Disable command line separation.

```
CMDSEP OFF
```

5.14 CNSLPORT (Console port)

5.14.1 Function

The CNSLPORT parameter specifies the port number (in decimal), on which the telnet server will listen. The statement may also have the form host:port, where the telnet console server will bind to the specified address.

5.14.2 Syntax

Descriptive

CNSLPORT {<u>3270</u> | *port* | *host:port*}

Diagram

```
►►── CNSLPORT ──┬────── 3270 ──────┬────────────────────────────────────►◄
                ├────── port ──────┤
                └─── host:port ────┘
```

5.14.3 Parameter

host The IP address of the host to which the telnet server will bind to. If an IP address is given then it must be a valid IP address for the host system.

port The port number (decimal) on which the telnet server will listen. The port number must not be in use by any other server. The port number must be in the range of 0 to 65535. Ports below 1024 cannot be used unless Hercules is running as root or is otherwise authorized to use low ports.

5.14.4 Examples

Example 1:

Set the port number on which the telnet server will listen to 3270.

```
CNSLPORT 3270
```

Example 2:

Specify 192.168.1.10 as the IP address of the host to which the telnet server will bind to and set the port number on which the telnet server will listen to 3270.

```
CNSLPORT 192.168.1.10:3270
```

5.15 CODEPAGE (Codepage conversion table)

5.15.1 Function

This parameter specifies the codepage conversion table used for the ASCII / EBCDIC translation. If no codepage is given the environment variable HERCULES_CP will be inspected. The default codepage used is “DEFAULT”.

5.15.2 Syntax

Descriptive

CODEPAGE {DEFAULT | *codepage* | USER | MAINT *cmd* [*operands*]}

Diagram

```
►►── CODEPAGE ──┬── DEFAULT ─────────────────────────┬──────────────────►◄
                ├── codepage ────────────────────────┤
                ├── USER ────────────────────────────┤
                └── MAINT ── cmd ──┬─────────────────┤
                                   └── operands ─────┘
```

5.15.3 Parameter

DEFAULT “DEFAULT” specifies the traditional Hercules codepage.

codepage Specifies the codepage conversion table used for ASCII / EBCDIC translation. Supported codepage mappings are shown in the table below. Iconv single byte codepages may also be used (e.g. “UTF8/EBCDIC-CP-NL”).

USER This specifies that the user specific codepage conversion tables (see CPUPDT system parameter and console command) have to be activated.

MAINT MAINT and its arguments is the same as the CP_UPDT system parameter. Please see CP_UPDT for details.

Supported codepage mappings:

Mapping	ASCII	EBCDIC
437/037	437 PC United States	037 United States/Canada
437/500	437 PC United States	500 Latin 1
437/1047	437 PC United States	1047 Open Systems Latin 1
819/037	819 ISO-8859-1 Latin 1	037 United States/Canada

Mapping	ASCII	EBCDIC
819/037v2	819 ISO-8859-1 Latin 1	037 United States/Canada SHARE
819/273	819 ISO-8859-1 Latin 1	273 CECP Austria/Germany
819/277	819 ISO-8859-1 Latin 1	277 CECP Denmark/Norway
819/278	819 ISO-8859-1 Latin 1	278 CECP Finland/Sweden
819/280	819 ISO-8859-1 Latin 1	280 CECP Italy
819/284	819 ISO-8859-1 Latin 1	284 CECP Spain
819/285	819 ISO-8859-1 Latin 1	285 CECP United Kingdom
819/297	819 ISO-8859-1 Latin 1	297 CECP France
819/500	819 ISO-8859-1 Latin 1	500 CECP International
819/1047	819 ISO-8859-1 Latin 1	1047 Open Systems Latin 1
850/273	850 PC Latin 1	273 Austria/Germany
850/1047	850 PC Latin 1	1047 Open Systems Latin 1
1252/037	1252 Windows Latin 1	037 United States/Canada
1252/037v2	1252 Windows Latin 1	037 United States/Canada SHARE
1252/1047	1252 Windows Latin 1	1047 Open Systems Latin 1
1252/1140	1252 Windows Latin 1	1140 United States/Canada with Euro sign

Table 4: Supported codepage mappings

5.15.4 Examples

Example 1:

Set the codepage conversion table to the default.

```
CODEPAGE DEFAULT
```

Example 2:

Set the codepage conversion table to 437/500.

```
CODEPAGE 437/500
```

Example 3:

Activate the user specific codepage conversion tables.

```
CODEPAGE USER
```

5.16 CONKPALV (Console and telnet clients keep-alive option)

5.16.1 Function

The CONKPALV parameter specifies the tn3270 console and telnet clients keepalive option values that control automatic detection of disconnected tn3270/telnet client sessions.

This is a built-in feature of TCP/IP and allows detection of unresponsive TCP/IP connections and not idle clients. That is to say, your connection will not be terminated after 3 seconds of idle time. Your 3270 session can remain idle for many minutes or hours or days without any data being transmitted. If the TCP/IP stack at the other end of the connection (not your 3270 client itself) fails to respond to the internal keepalive probe packets however, then it means that the TCP/IP stack itself is down or there has been a physical break in the connection.

Thus, even if your 3270 client is completely idle, your system's TCP/IP stack itself should still respond to the keepalive probes sent by the TCP/IP stack at the Hercules end of the link. If it doesn't, then TCP/IP will terminate the tn3270/telnet session which will cause Hercules to disconnect the terminal.

The three values can also be modified on-demand via the conkpalv panel command, which has the exact same syntax. Note that the syntax is very unforgiving: no spaces are allowed anywhere within the parentheses and each value must be separated from the other with a single comma.

Please also note that not all systems support being able to modify all three values. That is, not all values may be able to be changed, and it is system dependent which values you can change and which values you cannot. On Windows for example, the *count* value is ignored and cannot be changed from its default value of 10. Other systems may ignore one or more or all three values and use platform defaults instead. This is entirely system dependent. Check you system's documentation for details regarding which values can be changed and which cannot as well as how to adjust your system's default values.

5.16.2 Syntax

Descriptive

CONKPALV {<u>(3,1,10)</u> | (*idle,intv,count*)}

Diagram

```
►►── CONKPALV ──┬──────── (3,1,10) ───────┬──────────────────────────►◄
                └─── (idle,intv,count) ───┘
```

5.16.3 Parameter

idle The *idle* value specifies the number of seconds of inactivity until the first keep-alive probe is sent. The default for the *idle* value is 3 seconds.

intv The *intv* value specifies the interval in seconds between the probes if no acknowledgement is received from the previous probe. The default for *intv* is 1 second.

count The *count* value specifies the number of unacknowledged keep-alive packets sent before the connection is considered to have failed. The default value is 9 for non-Windows platforms and 10 for Windows systems.

Note: On Windows platforms the count value is ignored and cannot be changed from its default value of 10.

5.16.4 Examples

Example 1:

Set the tn3270 console and telnet clients keep-alive option settings on a Unix platform to the following values: 5 seconds of inactivity until the first keep-alive probe is sent, 3 seconds for the interval between the probes and to 15 unacknowledged keep-alive packets sent before the connection is considered to have failed.

```
CONKPALV (5,3,15)
```

5.17 CP_UPDT (User character conversion table)

5.17.1 Function

The CP_UPDT system parameter creates or modifies the contents of the user codepage tables. The tables can be populated with a ‘CP_UPDT REFERENCE’ statement which copies one of the provided codepage tables to the user tables or by a ‘CP_UPDT IMPORT’ statement which imports a previously created and with ‘CP_UPDT EXPORT’ exported table.

Changes in the user tables are made through one or more ‘CP_UPDT ALTER’ statements. These allow for up to 16 modifications at a time. The current contents of the user tables can be displayed on the console (and written to the log file) with ‘CP_UPDT DISPLAY’. Finally the changed user tables are activated with a ‘CODEPAGE USER’ statement.

In all CP_UPDT statements that require the selection of a user table (EBCDIC or ASCII), the EBCDIC table refers to the ‘guest to host’ (g2h) translation and the ASCII table refers to the ‘host to guest’ (h2g) translation.

5.17.2 Syntax

Descriptive

CP_UPDT *command*

where *command* can be:

ALTER {EBCDIC | ASCII | G2H | H2G} (*pos*,*val*[,*pos*,*val*]...)

DISPLAY {EBCDIC | ASCII | G2H | H2G}

EXPORT {EBCDIC | ASCII | G2H | H2G} *filename*

IMPORT {EBCDIC | ASCII | G2H | H2G} *filename*

REFERENCE [*codepage*]

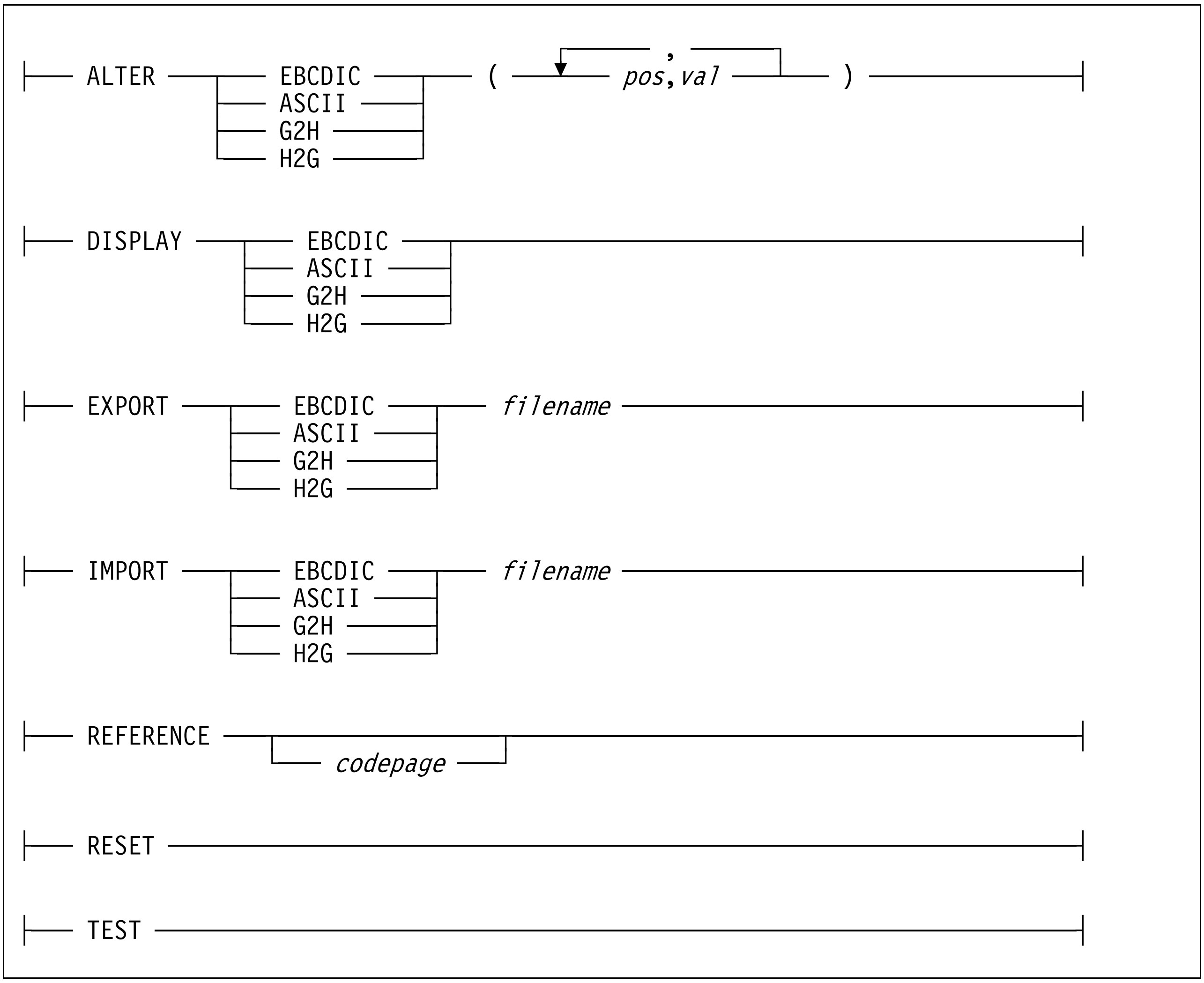
RESET

TEST

Diagram

```
►►── CP_UPDT ──┤ command ├──────────────────────────────────────►◄
```

where *command* can be:



5.17.3 Parameter

ALTER Alters the user EBCDIC or ASCII table value at hex position *pos* to hex value *val*. Up to 16 pairs of hex digits may be specified within the parenthesis. ALTER can be abbreviated as ‘ALT’.

DISPLAY Displays the user EBCDIC or ASCII codepage table. DISPLAY can be abbreviated as ‘DIS’ or ‘DSP’.

EXPORT Exports the contents of the user EBCDIC or ASCII codepage table to file *filename*. EXPORT can be abbreviated as ‘EXP’.

IMPORT Imports the contents of file *filename* to the user EBCDIC or ASCII codepage table. IMPORT can be abbreviated as ‘IMP’.

REFERENCE Copies the specified codepage to the user EBCDIC and ASCII tables. If no code-page is specified, a list of valid codepages is displayed on the console. ‘REFE-RENCE’ can be abbreviated as ‘REF’.

EBCDIC The target for the command is the EBCDIC table. The EBCDIC table refers to the

	‘guest to host’ translation. ‘EBCDIC’ can be abbreviated as ‘E’.
RESET	Reset the internal user tables to binary zero.
TEST	Verify that user tables are transparent, i.e. the value at position n in g2h used as an index into h2g will return a value equal n (g2h<=>h2g, h2g<=>g2h).
ASCII	The target for the command is the ASCII table. The ASCII table refers to the ‘host to guest’ translation. ‘ASCII’ can be abbreviated as ‘A’.
G2H	This is the same as ‘EBCDIC’.
H2G	This is the same as ‘ASCII’.
pos	Specifies the hex position within the selected table.
val	Specifies the hex value for the selected position.
filename	Specifies the file name of the file to which the specified codepage has to be exported or from which the codepage table has to be imported.
codepage	Specifies the codepage that has to be copied to the user tables.

5.17.4 Examples

Example 1:

Copy the Hercules default codepage to the user tables, alter the ASCII user table and activate the user tables.

```
CP_UPDT REFERENCE DEFAULT
CP_UPDT ALTER ASCII (5B,C0,5D,D0,7B,AD,7D,BD)
CODEPAGE USER
```

5.18 CPUIDFMT (Set format BASIC / 0 / 1 STIDP generation)

5.18.1 Function

The CPUIDFMT system parameter sets the STORE CPU ID (STIDP) format bit. The default STIDP format, if not explicitly set, is 'BASIC'. The format bit of the STIDP information specifies the format of the first two digits of the CPU identification number. When the format bit is ‘0’ then the contents of the CPU identification number identifies the CPU. When the format bit is ‘1’ then the CPU identification number identifies the system configuration as opposed to an individual CPU in the configuration and it identifies the logical partition in which the program is executed.

When the format is ‘BASIC’ the CPU identification number has the following hexadecimal format, where 'A' is the CPU address of the CPU.

- x'Annnnn' (Basic Mode)

When the format is ‘0’ the CPU identification number has the following hexadecimal format where 'L' is a logical CPU address and 'P' is a logical partition identifier.

- x'LPnnnn' (LPAR mode)

When the format is ‘1’ the CPU identification number has the following hexadecimal format where 'PP' is the user partition identifier (UPID). The UPID is an eight bit unsigned binary integer bound to a logical partition.

- x'PPnnnn' (LPAR mode)

In all cases *n* is a digit derived from the serial number of the CPU.

For more information on the STORE CPU ID (STIDP) instruction and the format bit see IBMs "z/Architecture Principles of Operation" manual.

5.18.2 Syntax

Descriptive

CPUIDFMT {BASIC | 0 | 1}

Diagram

```
►►── CPUIDFMT ──┬── BASIC ──┬──────────────────────────────────►◄
                ├──── 0 ────┤
                └──── 1 ────┘
```

5.18.3 Parameter

BASIC	Set the format to 'BASIC'. The STIDP format bit is set to ‘0’.
0	Set the format to '0'. The STIDP format bit is set to ‘0’.
1	Set the format to '1'. The STIDP format bit is set to ‘1’.

5.18.4 Examples

Example 1:

Set the STORE CPU ID format to ‘1’.

```
CPUIDFMT 1
```

Example 2:

Set the STORE CPU ID format to ‘BASIC’.

```
CPUIDFMT BASIC
```

5.19 CPUMODEL (CPU model number)

5.19.1 Function

The CPUMODEL parameter specifies the 4 hexadecimal digits CPU model number stored by the STIDP instruction.

5.19.2 Syntax

Descriptive

CPUMODEL {0586 | *model*}

Diagram

```
►►── CPUMODEL ──┬── 0586 ──┬──────────────────────────────────────►◄
                └── model ─┘
```

5.19.3 Parameter

model Any valid 4 digit hexadecimal CPU model number. A list of the valid model numbers can be found in the Hercules Windows GUI file “cpu-types.txt”. The default (“0586”) is not a mainframe CPU model number, but is related to the underlying PC architecture.

5.19.4 Examples

Example 1:

Specify a 7490 CPU model.

```
CPUMODEL 7490
```

Example 2:

Specify a 3090 CPU model.

```
CPUMODEL 3090
```

5.20 CPUPRIO (CPU thread process priority)

5.20.1 Function

CPUPRIO specifies the priority of the CPU thread. See section 5.82 “Process and Thread Priorities” for details. Default is a nice value of 15, which means a low priority such that I/O can be scheduled and completed in favour of CPU cycles. On multi-CPU systems a real CPU can be “dedicated” to Hercules by giving the CPU-thread a very high dispatching priority (-20).

Caution: CPUPRIO should not have a higher dispatching priority than the TOD clock and timer thread.

5.20.2 Syntax

Descriptive

```
CPUPRIO {15 | nn}
```

Diagram

```
►►── CPUPRIO ──┬── 15 ──┬──────────────────────────────────────────►◄
               └── nn ──┘
```

5.20.3 Parameter

15 Specifies a CPU thread prority of 15. This is the default

nn This value specifies the priority for the CPU thread. For details on the priority values see section 5.82 (“Process and Thread Priorities”). The default is 15.

5.20.4 Examples

Example 1:

Give the CPU-thread a very high dispatching priority of -20.

```
CPUPRIO -20
```

5.21 CPUSERIAL (CPU serial number)

5.21.1 Function

CPUSERIAL specifies the 6 hexadecimal digit CPU serial number stored by the STIDP instruction.

5.21.2 Syntax

Descriptive

CPUSERIAL {<u>000001</u> | *serial*}

Diagram

```
►►── CPUSERIAL ──┬── 000001 ──┬──────────────────────────────────────►◄
                 └── serial ──┘
```

5.21.3 Parameter

serial Any valid 6 digit hexadecimal CPU serial number. In BASIC mode, the high-order digit may be replaced with the processor number when MAXCPU is greater than one. In LPAR mode, the two high-order digits are replaced with either the LPAR number or the CPU number and LPAR number with the full serial number available via the STSI instruction. The default serial number is '000001'.

5.21.4 Examples

Example 1:

Set the CPU serial number to 001963.

```
CPUSERIAL 001963
```

5.22 CPUVERID (CPU version code)

5.22.1 Function

CPUVERID specifies the 2 hexadecimal digit CPU version code stored by the STIDP instruction. The default version code is “FD” when ARCHMODE S/370 or ARCHMODE ESA/390 is specified. For the z/ARCH (or ESAME architecture mode respectively) the version code is always stored as “00” and any value specified here is ignored.

5.22.2 Syntax

Descriptive

`CPUVERID 00` (For z/ARCH and ESAME)

`CPUVERID {FD | verid}` (For S/370 and ESA/390)

Diagram

For z/ARCH and ESAME:

```
►►── CPUVERID ── 00 ─────────────────────────────────────────►◄
```

For S/370 and ESA/390:

```
►►── CPUVERID ──┬──── FD ───┬─────────────────────────────────►◄
                └── verid ──┘
```

5.22.3 Parameter

verid Any valid 2 digit hexadecimal CPU version code. A list of valid version codes can be found in the Hercules Windows GUI file “cpu-types.txt”.

5.22.4 Examples

Example 1:

Set the CPU version code to FD.

```
CPUVERID FD
```

5.23 DEFSTORE (Define main and expanded storage)

5.23.1 Function

The DEFSTORE statement is used to specify the size of the main and expanded storage. Storage is allocated in megabytes, unless a specific unit is specified. The actual upper limit of the main and expanded storage is determined by the host system's architecture, operating system, and on some systems the amount of physical memory and paging space you have available.

The practical limit depends on the maximum amount of storage that can be obtained by the "malloc" function (usually around 1 GB on 32-bit platforms; on 64-bit platforms the value should only be limited by available paging space).

An additional optional argument determines the locking state of the allocated memory (page lock by host operating system). The LOCKED option indicates that the memory is to be locked into storage while UNLOCKED (the default) indicates that the memory is not locked into the storage.

Please note that Hercules preserves the last locking state of DEFSTORE for each type of storage. Once the storage is locked, any subsequent change to the storage size will honor the existing lock state of memory unless the lock state is specified again on the DEFSTORE command.

Caution: Do not lock storage unless sufficient real memory is available to back up the request. Failure to do so may require the host system to be rebooted.

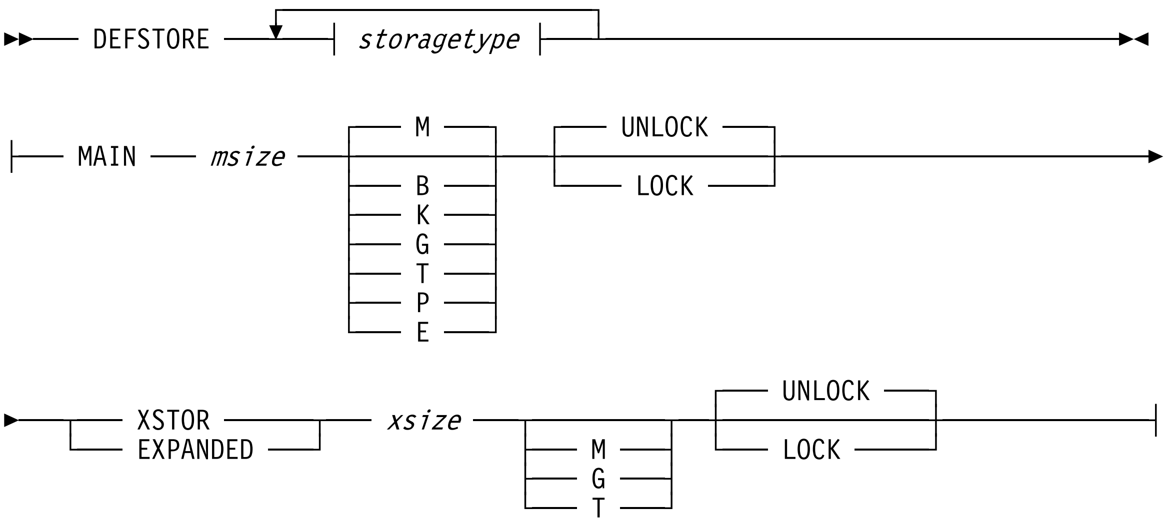
5.23.2 Syntax

Descriptive

DEFSTORE {[MAIN *msize*[B | K | <u>M</u> | G | T | P |E] [<u>UNLOCK</u> | LOCK]]

[{XSTOR | EXPANDED} *size*[<u>M</u> | G | T] [<u>UNLOCK</u> | LOCK]]}

Diagram



5.23.3 Parameter

msize The value of *msize* must be a valid decimal number. The actual upper limit is determined by the host system's architecture, the operating system and on some systems the amount of physical memory and paging space that is available.

For storage sizes less than 16M, sizes not on a 4K boundary are rounded up to the next 4K boundary. Otherwise, storage sizes not on a 1M boundary are rounded up to the next 1M boundary.

The minimum size is 4K for architecture levels ALS0 and ALS1 (S/370 and ESA/390), and 8K for architecture level ALS2 (ESAME) and higher. A maximum of 64M may be specified for architecture level ALS0 (S/370), 2048M (2G) for ALS1 (ESA/390) and 16E for architecture level ALS2 (ESAME) and higher.

The default on startup is 2M.

xsize The value of *xsize* must be a valid decimal number. The actual upper limit is determined by the host system’s architecture, the operating system and on some systems the amount of physical memory and paging space that is available.

Storage sizes not on a 1M boundary are rounded up to the next 1M boundary. The lower limit and default is 0.

B ‘B’ determines that the number given is specified in bytes (no multiplier). Specifying the storage in bytes is possible only for main storage.

K ‘K’ determines that the number given is specified in kilobytes (multiplier 2**10). Specifying the storage in kilobytes is possible only for main storage.

M ‘M’ determines that the number given is specified in megabytes (multiplier 2**20). This is the default if no unit is appended.

G ‘G’ determines that the number given is specified in gigabytes (multiplier 2**30).

T ‘T’ determines that the number given is specified in terabytes (multiplier 2**40). On 32-bit machines the unit terabytes is not available.

P ‘P’ determines that the number given is specified in petabytes (multiplier 2**50). Specifying the storage in petabytes is possible only for main storage. On 32-bit machines the unit petabytes is not available.

E ‘E’ determines that the number given is specified in exabytes (multiplier 2**60). Specifying the storage in exabytes is possible only for main storage. On 32-bit machines the unit exabytes is not available.

LOCK Attempt to lock the storage (pages locked by the host operating system).

UNLOCK Leave the store unlocked (no pages locked by the host operating system). This is the default.

Notes:

The actual upper limit is determined by the host system’s architecture and operating system and the amount of physical memory and available paging space. The total of MAINSIZE and XPNDSIZE on host systems with a 32-bit architecture will be limited to 4G; host systems with a 64-bit architecture will be limited to less than 16E.

Using minimum storage sizes, storage sizes less than or not on a 64K boundary for architecture level ALS0 (S/370) or not on a 1M boundary for architecture level ALS1 (ESA/390) and higher, it may be possible to generate error conditions not covered by the “Principles of Operations”.

Use of storage sizes greater than supported by the guest operating system may generate incorrect results or error conditions within the guest operating system.

5.23.4 Overview Storage Allocation Units

Unit	Multiplier	Name (Symbol)	IEC Name (IEC Symbol)	Restrictions
B	None	Byte (B)	Byte (B)	Main storage only
K	2**10	Kilobyte (kB)	Kibibyte (KiB)	Main storage only
M	2**20	Megabyte (MB)	Mebibyte (MiB)	
G	2**30	Gigabyte (GB)	Gibibyte (GiB)	
T	2**40	Terabyte (TB)	Tebibyte (TiB)	Not on 32-bit machines
P	2**50	Petabyte (PB)	Pebibyte (PiB)	Not on 32-bit machines
E	2*60	Exabyte (EB)	Exbibyte (EiB)	Not on 32-bit machines

Table 5: Storage Allocation Units

5.23.5 Examples

Example 1:

Set the size of the main storage to 1024 MB. Do not lock the memory into the storage.

```
DEFSTORE MAIN 1024
  or
DEFSTORE MAIN 1024 UNLOCK
  or
DEFSTORE MAIN 1024M
  or
DEFSTORE MAIN 1024M UNLOCK
```

Example 2:

Set the size of the expanded storage to 256 MB. Do not lock the memory into the storage.

```
DEFSTORE XSTORE 256
  or
DEFSTORE XSTORE 256M
  or
DEFSTORE XSTORE 256 UNLOCK
  or
DEFSTORE XSTORE 256M UNLOCK
```

Example 3:

Set the the main storage to 2 GB, the expanded storage to 512 MB and lock the memory into the storage.

```
DEFSTORE MAIN 2048 LOCK XSTORE 512 LOCK
  or
DEFSTORE MAIN 2048M LOCK XSTORE 512M LOCK
  or
DEFSTORE MAIN 2G LOCK XSTORE 512M LOCK
```

5.24 DEFSYM (Define a symbol)

5.24.1 Function

Defines symbol '*symbol'* is to contain value '*value'*. The symbol can then be the object of a substitution used later in the configuration file or for console commands. If *value* contains blanks or spaces it must be enclosed within quotes or apostrophes. See chapter "Symbol Substitutions" for a more in-depth discussion on this feature.

Substitution is available in configuration statements, meaning it is possible to perform substitution in the *DEFSYM* statement itself. However, symbols are always defined as the last step in the process, so attempting to self-define a symbol will result in an empty string.

5.24.2 Syntax

Descriptive

DEFSYM *symbol value*

Diagram

►►── DEFSYM ── *symbol* ── *value* ──────────────────────────►◄

5.24.3 Parameter

symbol The name of a symbol.

value The value that is assigned to the symbol.

5.24.4 Examples

Example 1:

Define a symbol "TAPEDIR" with the value "/home/hercules/tapes".

```
DEFSYM TAPEDIR "/home/hercules/tapes"
```

5.25 DEVPRIO (Device threads process priority)

5.25.1 Function

DEVPRIO specifies the priority of the device threads. See section 5.82 “Process and Thread Priorities” for details.

Caution: DEVPRIO should not have a higher dispatching priority than the TOD clock and timer thread.

5.25.2 Syntax

Descriptive

DEVPRIO {<u>8</u> | *nn*}

Diagram

```
►►── DEVPRIO ──┬── 8 ──┬────────────────────────────────────────►◄
               └── nn ─┘
```

5.25.3 Parameter

8 Specifies a device threads prority of 8. This is the default

nn This value specifies the priority for the device threads. For details on the priority values see section 5.82 (“Process and Thread Priorities”). The default is 8.

5.25.4 Examples

Example 1:

Set a priority of 10 for the device threads.

```
DEVPRIO 10
```

5.26 DEVTMAX (Maximum number of device threads)

5.26.1 Function

DEVTMAX specifies the maximum number of device threads allowed.

5.26.2 Syntax

Descriptive

`DEVTMAX {0 | -1 | 1-n}`

Diagram

```
►►── DEVTMAX ──┬── 0 ───┬──────────────────────────────────────►◄
               ├── -1 ──┤
               └── 1-n ─┘
```

5.26.3 Parameter

0 Specify 0 to create an unlimited number of ‘semi-permanent’ threads on an ‘as-needed’ basis. This is the default. With this option, a thread is created to service an I/O request for a device if one does not already exist. When the I/O is complete the thread enters an idle state waiting for new work. If a new I/O request for the device arrives before the timeout period expires the existing thread will be reused. The timeout value is currently hard coded at 5 minutes.

Note that this option can cause one thread (or possibly more) to be created for each device in your configuration. Specifying 0 means there is no limit to the number of threads that can be created.

-1 Specify -1 to cause ‘one time only’ temporary threads to be created to service each I/O request to a device. Once the I/O request is complete, the thread exits. Subsequent I/Os to the same device will cause another worker thread to be created again.

1-*n* Specify a value from 1 to n to set an upper limit to the number of threads that can be created to service any I/O request to any device. Like the “0” option, each thread once finished servicing an I/O request enters an idle state. If a new request arrives before the timeout period expires, the thread is reused.

If all threads are busy when a new I/O request arrives a new thread is created only if the specified maximum number of threads have not yet been reached. If the specified maximum number of threads already has been reached then the I/O request is placed in a queue and will be serviced by the first available thread (eg. by whichever thread becomes idle first).

This option was created to address a threading issue, possibly related to the Cygwin phtreads implementation on Windows systems. On Windows systems positive DEVTMAX values are currently not honoured and are treated identically as if the value 0 had been specified. The default for non-Windows systems is 0.

5.26.4 Examples

Example 1:

Set the upper limit for the number of device threads to 16.

```
DEVTMAX 16
```

Example 2:

Allow an unlimited number of device threads.

```
DEVTMAX 0
```

5.27 DIAG8CMD (DIAGNOSE 8 command option)

5.27.1 Function

This parameter specifies whether a command issued through Diagnose 8 will be executed by Hercules as a Hercules commands or not. An optional second argument can be specified to request whether the commands issued using the Diagnose 8 interface will be traced at the console. This may be useful for programs that routinely issue console commands using the Diagnose 8 interface.

Caution: Enabling this feature may have security consequences. When this feature is enabled it is possible for guest operating systems running under Hercules to issue commands directly to the host operating system by means of the Hercules 'sh' (shell) command. This ability may be disabled via the SHCMDOPT statement.

Note: There are some commands that are being prevented from being used by the Diagnose 8 interface. The list of commands that may not be executed by means of Diagnose 8 can be found in "Appendix D. Hercules Command Groups" under the column 'NODIAG8'.

5.27.2 Syntax

Descriptive

```
DIAG8CMD {DISABLE | ENABLE [ECHO | NOECHO]}
```

Diagram

```
►►── DIAG8CMD ──┬── DISABLE ──────────────────┬──────────────────►◄
                └── ENABLE ───┬───────────────┤
                              ├──── ECHO ─────┤
                              └─── NOECHO ────┘
```

5.27.3 Parameter

DISABLE Commands issued through the Diagnose 8 interface will generate a Specification Exception program interrupt on the issuing CPU. This is the default.

ENABLED Commands issued through the Diagnose 8 interface will be executed by Hercules as Hercules commands.

ECHO When ECHO is specified, a message is issued as the console is about to issue the command, the command is redisplayed as if it was entered through the console input line, and a final message is issued to indicate the command completed.

NOECHO When NOECHO is specified, no such messages are displayed and the command completes silently, except for the output of the command itself if the Diagnose 8 interface did not request a response buffer. This is the default.

The value of ECHO or NOECHO has no effect on command output being placed into a response buffer if the Diagnose 8 interface requested one.

5.27.4 Examples

Example 1:

Specify that commands issued through the Diagnose 8 interface are executed as Hercules commands. Additionally issue a message, as the console is about to execute the command, then redisplay the command itself and give a final message, indicating the command has completed.

```
DIAG8CMD ENABLE ECHO
```

5.28 ECPSVM (ECPS:VM support status (VM))

5.28.1 Function

This parameter specifies, whether ECPS:VM (Extended Control Program Support : Virtual Machine) support is to be enabled and – if it is enabled – to which level. The purpose of ECPS:VM is to provide to the VM/370 operating system a set of shortcut facilities to perform hypervisor functions (CP Assists) and virtual machine simulation (VM Assists).

Although this feature does not affect VM operating system products operating in XA, ESA or z/Architecture mode, it will affect VM/370 and VM/SP products running under VM/XA, VM/ESA or z/VM. Running VM/370 and VM/SP products under VM/XA, VM/ESA or z/VM should be done with ECPS:VM disabled. ECPS:VM should not be enabled in AP or MP environments. ECPS:VM has no effect on non-VM operating systems. It is however recommended to disable ECPS:VM when running native non-VM operating systems.

If a specific level is given this value will be reported to the operating system when it issues a Store ECPS:VM level but it does not otherwise alter the ECPS:VM facility operations. Please note that this is a partial implementation.

Because the ECPSVM system parameter is processed by the same command processor like the ECPSVM console command, the same parameters can be specified. This may not make sense for all of them however. As an example the STATS parameter in the configuration is useless, because there are no statistics to display at this time.

Some of the arguments of the ECPSVMsystem parameter can be abbreviated as shown in the syntax section below.

5.28.2 Syntax

Descriptive

ECPSVM {Help | STats | DISable | ENable | DEBUG | NOdebug | Level [*nn*]}

Diagram

```
►►── ECPSVM ──┬── Help ──────────┬──────────────────────────►◄
              ├── STats ─────────┤
              ├── DISable ───────┤
              ├── ENable ────────┤
              ├── DEBUG ─────────┤
              ├── NOdebug ───────┤
              └── Level ───┬─────┤
                           └── nn ┘
```

5.28.3 Parameter

Help	Display help with all available ECPSVM options.
STats	Show statistical counters.
DISable	Disable all ECPS:VM features. This is the default if ECPSVM is not coded.
ENable	Enable all ECPS:VM features.
DEBUG	Debug ECPS:VM features.
NOdebug	Turn debug modus off for ECPS:VM features.
Level	Set/show ECPS:VM level.
nn	The value *nn* specifies the support level that is reported to the operating system.

5.28.4 Examples

Example 1:

Enable “Extended Control Program Support : Virtual Machine” support.

```
ECPSVM ENABLE
or
ECPSVM EN
```

Example 2:

Enable “ECPS:VM” support and set the level reported to the operating system to 20.

```
ECPSVM LEVEL 20
or
ECPSVM L 20
```

Example 3:

Debug “ECPS:VM” features.

```
ECPSVM DEBUG
```

5.29 ENGINES (Processor engines type)

5.29.1 Function

The ENGINES parameter specifies the type of engine for each installed processor. The default engine type is CP. The number of installed processor engines is determined by the MAXCPU system parameter.

If the ENGINES system parameter specifies more than MAXCPU engines, the excess engines are ignored. If fewer than MAXCPU engines are specified, the remaining engines are set to type CP (the default). See the MAXCPU system parameter or console command for details regarding the compile time variable MAX_CPU_ENGINES.

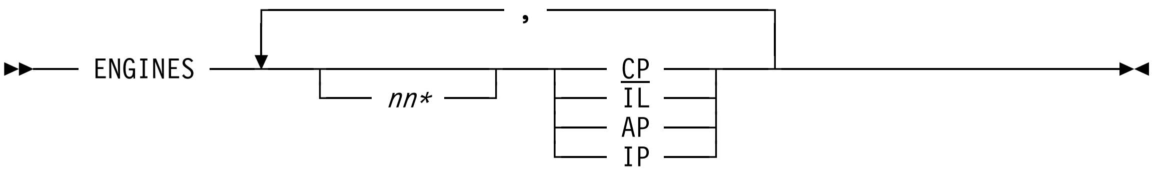
For detailed explanations on the interrelationship between ENGINES, MAXCPU and NUMCPU please see “Appendix B. Configuration of Emulated CPUs”.

5.29.2 Syntax

Descriptive

`ENGINES [nn*] {CP | IL | AP | IP} [, ... ]`

Diagram



5.29.3 Parameter

nn*	This is an optional repeat count.
CP	Specifies a processor engine of type CP. This is the default.
IL	Specifies a processor engine of type IL.
AP	Specifies a processor engine of type AP.
IP	Specifies a processor engine of type IP.

5.29.4 Examples

Example 1:

Specify 4 engines of type CP, 2 engines of type AP and 2 engines of type IP.

```
ENGINES CP,CP,CP,CP,AP,AP,IP,IP
   or
ENGINES 4*CP,2*AP,2*IP
```

Example 2:

Specify 4 engines of type CP, 1 engine of type AP, 1 engine of type IP and 2 engines of type IL.

```
ENGINES CP,CP,CP,CP,AP,IP,IL,IL
   or
ENGINES 4*CP,AP,IP,2*IL
```

Example 3:

Specify 3 engines of type CP and 1 engines of type IL.

```
ENGINES CP,CP,CP,IL
   or
ENGINES 3*CP,IL
```

5.30 HAO (Hercules Automatic Operator)

5.30.1 Function

The Hercules Automatic operator (HAO) feature is a facility that allows to automatically issue console commands in response to certain messages being issued. To use the HAO facility it is necessary to define a rule, consisting of a target and an associated command.

The Hercules Automatic Operator is only for those messages issued by Hercules to its console. It cannot be used for messages issued from the guest operating system.

The current implementation limits the total number of defined rules to 64. There is currently no way to define a command whose arguments varies based on actual message text. All of the defined HAO rules are checked for a match each time Hercules issues a message, there is no way at this time to stop the processing of subsequent rules.

HAO supports several commands. However when HAO is used in the configuration file, not all of the possible HAO commands make sense. As an example, there is no reason to define a rule in the configuration file and afterwards to delete it in one of the next statements. Therefore only those commands are listed here, that are useful during Hercules startup. For a list of all possible HAO commands see the HAO panel command.

5.30.2 Syntax

Descriptive

HAO *command* [*operands*]

where *command* can be:

TGT *target*

CMD *consolecmd*

LIST [*nn*]

Diagram

```
►►── HAO ──┤ command ├──────────────────────────────────────────►◄

   where command can be:

├── TGT ── target ──────────────────────────────────────────────┤

├── CMD ── consolecmd ──────────────────────────────────────────┤
```

```
├── LIST ──┬──────┬──────────────────────────────────────┤
           └─ nn ─┘
```

5.30.3 Parameter

TGT	This is the keyword to define a new rule (pattern).
target	Specifies the rule (pattern) to react on. The target is a regular expression pattern which is matched against the text of the messages that Hercules issues.
CMD	This is the keyword to specify a command for a previously defined rule.
consolecmd	Specifies the command to be executed if a target rule matches. The associated command must be a valid Hercules console command.
LIST	List all rules/commands or list only rule/command at index *nn*.
nn	Number of the index to be listed or deleted.

5.30.4 Examples

Example 1:

Define the following target rule and the related command to be issued:

Check for message HHC01600E ("Unknown Hercules command") and issue "?" command (List all valid commands).

```
HAO TGT HHC01600E

HAO CMD HELP
```

5.31 HERCLOGO (Hercules logo file)

5.31.1 Function

HERCLOGO specifies the logo text file which defines a welcome screen that is presented when a TN3270 terminal connects to a Hercules 3270 device. For details on how to code the Hercules logo file, see section 10 ("Hercules 3270 Logo").

5.31.2 Syntax

Descriptive

```
HERCLOGO filename
```

Diagram

```
►►── HERCLOGO ── filename ──────────────────────────────────►◄
```

5.31.3 Parameter

filename The name (and optionally path) of a logo text file. If no path is specified the logo file is first searched in the current working directory and second in the directory where the Hercules executable resides.

5.31.4 Examples

Example 1:

Use the logo text file "HERCLOGO.txt" for the welcome screen.

```
HERCLOGO D:\HERCULES\CONF\HERCLOGO.TXT
```

5.32 HERCPRIO (Hercules process priority)

5.32.1 Function

The HERCPRIO parameter specifies the process priority for Hercules. See section 5.82 “Process and Thread Priorities” for details.

Caution: HERCPRIO should not have a higher dispatching priority than the TOD clock and timer thread.

5.32.2 Syntax

Descriptive

HERCPRIO {0 | *nn*}

Diagram

```
►►── HERCPRIO ──┬── 0 ──┬──────────────────────────────►◄
                └── nn ─┘
```

5.32.3 Parameter

0 Specifies a process priority for Hercules of 0. This is the default

nn This value specifies the process priority for Hercules. For details on the priority values see section 5.82 (“Process and Thread Priorities”). The default is 0.

5.32.4 Examples

Example 1:

Set the Hercules process priority to 0.

```
HERCPRIO 0
```

5.33 HTTP (HTTP server configuration)

5.33.1 Function

The HTTP statement configures the HTTP server. More than one HTTP statement can be coded in the configuration file. Depending on the given arguments the port on which the HTTP server will listen and the authorization (if any) or the location of the HTTP server files can be specified. Additional arguments allow to start or to stop the HTTP server. Please note, that the HTTP server is not started by default (i.e. without a “HTTP START” statement).

5.33.2 Syntax

Descriptive

HTTP {START | STOP | ROOT *path* | PORT *port* {NOAUTH | AUTH *userid password*}}

Diagram

```
►►── HTTP ──┬── START ─────────────────────────────────────┬──────────►◄
            ├── STOP ──────────────────────────────────────┤
            ├── ROOT ── path ──────────────────────────────┤
            └── PORT ── port ──┬────────── NOAUTH ─────────┤
                               └── AUTH userid password ───┘
```

5.33.3 Parameter

START Start the HTTP server (if it is stopped).

STOP Stop the HTTP server (if it is started).

ROOT Keyword to specify the root directory of the HTTP server files. The HTTP root can only be set if the HTTP server is in the stopped state.

path The full path of the root directory where the HTTP server files reside. If this parameter is not specified the default value for Win32 builds of Hercules is the directory where the Hercules executables themselves reside. For non-Win32 builds it is the directory specified as the default package installation directory when the Hercules executables were built. This can vary depending on how the Hercules package was built, it is commonly “/usr/local/share/hercules”.

PORT Keyword to specify the port on which the HTTP server will listen (including optional authorization information). The HTTP port and authorization information can only be set if the HTTP server is in the stopped state.

port The port number must be either 80 or within the range of 1024 to 65535 inclusive.

NOAUTH NOAUTH indicates that no userid and password are required to access the HTTP server.

AUTH AUTH indicates that a userid and a password are required to access the HTTP server. The userid and password have to be coded after the AUTH parameter.

userid The userid can be any valid string.

password The password can be any valid string.

5.33.4 Examples

Example 1:

Activate the HTTP server under a Linux system listening on port 8081 and with the HTTP server files residing on “/usr/local/share/Hercules”. Specify that authorization is required to access the HTTP server. The userid should be UID0001 and the password should be PSWD0001.

```
HTTP PORT 8081 AUTH UID0001 PSWD0001
HTTP ROOT /usr/local/share/Hercules
HTTP START
```

Example 2:

Activate the HTTP server under a Windows system listening on port 8088 and with the HTTP server files residing on “D:\Hercules\HTML”. Specify that authorization is not required to access the HTTP server.

```
HTTP PORT 8088 NOAUTH
HTTP ROOT D:\Hercules\HTML
HTTP START
```

5.34 HTTPPORT (HTTP server port)

5.34.1 Function

The HTTPPORT parameter has been deprecated and is replaced by the HTTP system parameter. Use "HTTP PORT" instead. See the HTTP system parameter for details.

5.34.2 Syntax

See HTTP system parameter.

5.34.3 Parameter

See HTTP system parameter.

5.34.4 Examples

See HTTP system parameter.

5.35 HTTPROOT (HTTP server root directory)

5.35.1 Function

The HTTPROOT parameter has been deprecated and is replaced by the HTTP system parameter. Use "HTTP ROOT" instead. See the HTTP system parameter for details.

5.35.2 Syntax

See HTTP system parameter.

5.35.3 Parameter

See HTTP system parameter.

5.35.4 Examples

See HTTP system parameter.

5.36 IGNORE (Ignore subsequent INCLUDE errors)

5.36.1 Function

This system parameter indicates that errors caused by subsequent INCLUDE statements - for files which do not exist - should be ignored rather than causing the Hercules startup to be aborted as it would otherwise normally occur

5.36.2 Syntax

Descriptive

```
IGNORE INCLUDE_ERRORS
```

Diagram

```
►►—— IGNORE —— INCLUDE_ERRORS ——————————————————————————————►◄
```

5.36.3 Parameter

INCLUDE_ERRORS Indicates that subsequent INCLUDE errors will be ignored.

5.36.4 Examples

Example 1:

Specify that errors caused by subsequent INCLUDE statements should be ignored.

```
IGNORE INCLUDE_ERRORS
```

5.37 INCLUDE (Include configuration file)

5.37.1 Function

An INCLUDE statement instructs the Hercules initialisation process to treat the contents of the file specified by filepath as if its contents had appeared in the configuration file at the point where the INCLUDE statement appears.

The included file itself may contain yet another INCLUDE statement as long as the maximum nesting depth of 8 is not exceeded.

5.37.2 Syntax

Descriptive

INCLUDE *filepath*

Diagram

```
►►—— INCLUDE —— filepath ——————————————————————————►◄
```

5.37.3 Parameter

filepath Specifies the path and the filename of the configuration file that is to be included.

5.37.4 Examples

Example 1:

Specify that Hercules should include the contents of the file "D:\S390\CONF\CONFIG2.CONF" into the configuration.

```
INCLUDE D:\S390\CONF\CONFIG2.CONF
```

5.38 IODELAY (I/O interrupt wait time (LINUX))

5.38.1 Function

IODELAY sets the amount of time in microseconds to wait after an I/O interrupt is ready to be set pending. This value can also be set using the Hercules console. The purpose of this parameter is to circumvent a bug in the Linux/390 and z/Linux ‘dasd.c’ device driver. The bug is more likely to occur under Hercules than on a real machine as Hercules may present an I/O interrupt sooner than a real machine.

NOTE: OSTAILOR LINUX no longer sets IODELAY to 800 since the problem described above is no longer present in recent versions of the Linux kernel.

5.38.2 Syntax

Descriptive

IODELAY {<u>0</u> | *usecs* [NOWARN]}

Diagram

```
►►── IODELAY ──┬────────── 0 ──────────────┬──────────────────────►◄
               └── usecs ──┬───────────────┤
                           └──── NOWARN ───┘
```

5.38.3 Parameter

usecs Amount of time in microseconds to wait after an I/O interrupt is ready to be set pending.

NOWARN If the IODELAY value is non-zero then a warning message will be issued unless NOWARN is specified.

5.38.4 Examples

Example 1:

Set the amount of time to wait after an I/O interrupt is ready to be set pending to 1000 microseconds.

```
IODELAY 1000
```

Example 2:

Set the amount of time to wait after an I/O interrupt is ready to be set pending to 500 microseconds and suppress the warning message.

```
IODELAY 500 NOWARN
```

5.39 LDMOD (Additional dynamic load modules)

5.39.1 Function

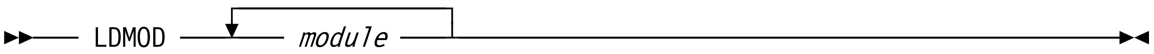
LMOD provides the capability of defining additional modules that will be loaded by the Hercules Dynamic Loader. The default search order is within the Hercules directory and in the default DLL search path. Most systems also support absolute filenames (e.g. names starting with “/” or “.”) in which case the default search path is not taken. Multiple LDMOD statements may be used.

5.39.2 Syntax

Descriptive

LMOD *module* [*module* [*module* ...]]

Diagram



5.39.3 Parameter

module A list of modules that are to be loaded by the Hercules Dynamic Loader.

5.39.4 Examples

Example 1:

Instruct Hercules to load the additional module named S37X (S/370 Extension) from the Hercules directory or the default DLL search path.

```
LDMOD S37X
```

5.40 LEGACYSENSEID (SENSE ID CCW (x'E4') feature)

5.40.1 Function

This option specifies whether the SENSE ID CCW (X'E4') will be honoured for the devices that originally did not support that feature. This includes (but may not be limited to) 3410 and 3420 tape drives, 2311 and 2314 direct access storage devices and 2703 communication controllers. Because those legacy devices didn't originally support this command, and for compatibility reasons, the default is OFF or DISABLE.

Specify ON or ENABLE, if your guest operating system needs the Sense ID support to dynamically detect those devices. Note that most current operating systems will not detect those devices even though Sense ID is enabled because those devices never supported the Sense ID in the first place. This mainly applies to custom built or modified versions of guest operating systems that are aware of this specific Hercules capability.

5.40.2 Syntax

Descriptive

LEGACYSENSEID {OFF | DISABLE | ON | ENABLE}

Diagram

```
►►── LEGACYSENSEID ──┬──── OFF ────┬──────────────────────────────────►◄
                     ├── DISABLE ──┤
                     ├──── ON ─────┤
                     └── ENABLE ───┘
```

5.40.3 Parameter

OFF Specify OFF or DISABLE if your guest operating system does not need the Sense ID support to dynamically detect devices that originally did not support that feature. This is the default.

DISABLE This is the same os ‘OFF’.

ON Specify ON or ENABLE if your guest operating system needs the Sense ID support to dynamically detect devices that originally did not support that feature.

ENABLE This is the same as ‘ON’.

5.40.4 Examples

Example 1:

Specify that the SENSE ID CCW will be honoured for devices that originally did not support that feature.

```
LEGACYSENSEID ENABLE
```

5.41 LOADPARM (IPL parameter)

5.41.1 Function

This parameter specifies the eight-character IPL parameter which is used by all MVS based operating systems (MVS 3.8J, MVS/SP, MVS/XA, MVS/ESA, OS/390, z/OS) to select the system start parameter.

5.41.2 Syntax

Descriptive

LOADPARM *ipl_parameter*

Diagram

```
►►── LOADPARM ── ipl_parameter ──────────────────────────────►◄
```

5.41.3 Parameter

ipl_parameter The system parameter used for the IPL of the intended operating system. The parameter is operating system dependent, consult the relevant operating system documentation for details.

5.41.4 Examples

Example 1:

Use 0A8200M1 as system parameter for the IPL of the operating system.

```
LOADPARM 0A8200M1
```

5.42 LOGOPT (Logging options)

5.42.1 Function

The LOGOPT statement specifies the Hercules log options. It allows to insert or suppress the time stamp in front of each log message

5.42.2 Syntax

Descriptive

```
LOGOPTS {TIMESTAMP | TIME | NOTIMESTAMP | NOTIME}
```

Diagram

```
►►── LOGOPTS ──┬──── TIMESTAMP ──┬──────────────────────────────────►◄
               ├────── TIME ─────┤
               ├── NOTIMESTAMP ──┤
               └───── NOTIME ────┘
```

5.42.3 Parameter

TIMESTAMP Insert a time stamp in front of each log message. This is the same as using TIME.

TIME Insert a time stamp in front of each log message. This is the same as using TIME-STAMP.

NOTIMESTAMP Display log messages without a timestamp. This is the same as using NOTIME.

NOTIME Display log messages without a timestamp. This is the same as using NOTIME-STAMP.

5.42.4 Examples

Example 1:

Specify that messages in the log are to be time stamped.

```
LOGOPTS TIMESTAMP
```

5.43 LPARNAME (LPAR name returned by DIAG x'204')

5.43.1 Function

LPARNAME defines the LPAR name returned by DIAG x’204’.

5.43.2 Syntax

Descriptive

```
LPARNAME {HERCULES | lparname}
```

Diagram

```
►►── LPARNAME ──┬── HERCULES ──┬──────────────────────────────►◄
                └── lparname ──┘
```

5.43.3 Parameter

lparname Maximum 8 byte character name for the LPAR. The default is HERCULES.

5.43.4 Examples

Example 1:

Specify HERCULES as LPAR name returned by DIAG x’204’.

```
LPARNAME HERCULES
```

5.44 LPARNUM (LPAR identification number)

5.44.1 Function

The LPARNUM system parameter sets the LPAR identification number. It specifies the one- or two-digit hexadecimal LPAR identification number stored by the STIDP instruction, or BASIC. If a one-digit number from 1 to F (hexadecimal) is specified, then STIDP stores a format-0 CPU ID, unless a subsequent "CPUIDFMT 1" statement is specified. If zero or a two-digit hexadecimal number, except 10 (hexadecimal), is specified, then STIDP stores a format-1 CPU ID. For LPARNUM 10 the current CPUIDFMT is not changed. If LPARNUM is BASIC, then the STIDP instruction stores a basic-mode CPU ID. The default is LPARNUM 1 with a format-0 CPU ID.

5.44.2 Syntax

Descriptive

LPARNUM {BASIC | 1 | *n* | *nn*}

Diagram

```
►►── LPARNUM ──┬── BASIC ──┬──────────────────────────────────────────►◄
               ├──── 1 ────┤
               ├── 1...F ──┤
               ├──── 0 ────┤
               └─── nn ────┘
```

5.44.3 Parameter

BASIC Specifies that STIDP stores a basic-mode CPU ID.

1 Specifies the one-digit hexadecimal LPAR identification number. The STIDP instruction stores a format-0 CPU ID. This is the default.

1 … F Specifies the one-digit hexadecimal LPAR identification number. The STIDP instruction stores a format-0 CPU ID, unless a subsequent "CPUIDFMT 1" statement is specified.

0 Specifies 0 as LPAR identification number. The STIDP instruction stores a format-1 CPU ID.

nn Specifies the two-digit hexadecimal LPAR identification number (except 10 hexadecimal). For LPARNUM 10 the current CPUIDFMT is not changed. The STIDP instruction stores a format-1 CPU ID.

5.44.4 Examples

Example 1:

Set the LPAR identification number to x’21’ (Format-1 CPU ID).

```
LPARNUM 21
```

Example 2:

Set the LPAR identification number to x’A’ (Format-0 CPU ID).

```
LPARNUM A
```

Example 3:

Set a basic-mode CPU ID.

```
LPARNUM BASIC
```

5.45 MAINSIZE (Main storage size)

5.45.1 Function

MAINSIZE specifies the size of the main storage. Storage is allocated in megabytes, unless a specific unit is specified. The actual upper limit of the main storage is determined by the host system's architecture, operating system, and on some systems the amount of physical memory and paging space you have available. The practical limit depends on the maximum amount of storage that can be obtained by “malloc” (usually around 1 GB on 32-bit platforms; on 64-bit platforms the value should only be limited by available paging space).

An additional optional argument determines the locking state of the allocated memory (page lock by host operating system). The LOCKED option indicates that the memory is to be locked into storage while UNLOCKED (the default) indicates that the memory is not locked into the storage.

Please note that Hercules preserves the last locking state of MAINSIZE. Once storage is locked, any subsequent change to the main storage size will honor the existing lock state of memory unless the lock state is specified again on the MAINSIZE command.

Caution: Do not lock main storage unless sufficient real memory is available to back up the request. Failure to do so may require the host system to be rebooted.

5.45.2 Syntax

Descriptive

MAINSIZE *msize*[B | K | M | G | T | P |E] [UNLOCK | LOCK]

Diagram

```
                              ┌── M ──┐   ┌── UNLOCK ──┐
►►── MAINSIZE ── msize ───────┼───────┼───┴── LOCK ────┴──────────────►◄
                              ├── B ──┤
                              ├── K ──┤
                              ├── G ──┤
                              ├── T ──┤
                              ├── P ──┤
                              └── E ──┘
```

5.45.3 Parameter

msize The value of *msize* must be a valid decimal number. The actual upper limit is determined by the host system's architecture, the operating system and on some systems the amount of physical memory and paging space that is available.

For storage sizes less than 16M, sizes not on a 4K boundary are rounded up to the next 4K boundary. Otherwise, storage sizes not on a 1M boundary are rounded up to the next 1M boundary.

The minimum size is 4K for architecture levels ALS0 and ALS1 (S/370 and

ESA/390), and 8K for architecture level ALS2 (ESAME) and higher. A maximum of 64M may be specified for architecture level ALS0 (S/370), 2048M (2G) for ALS1 (ESA/390) and 16E for architecture level ALS2 (ESAME) and higher.

The default on startup is 2M.

B ‘B’ determines that the number given is specified in bytes (no multiplier).

K ‘K’ determines that the number given is specified in kilobytes (multiplier 2**10).

M ‘M’ determines that the number given is specified in megabytes (multiplier 2**20). This is the default if no unit is appended.

G ‘G’ determines that the number given is specified in gigabytes (multiplier 2**30).

T ‘T’ determines that the number given is specified in terabytes (multiplier 2**40). On 32-bit machines the unit terabytes is not available.

P ‘P’ determines that the number given is specified in petabytes (multiplier 2**50). On 32-bit machines the unit petabytes is not available.

E ‘E’ determines that the number given is specified in exabytes (multiplier 2**60). On 32-bit machines the unit exabytes is not available.

LOCK Attempt to lock the storage (pages locked by the host operating system).

UNLOCK Leave the store unlocked (no pages locked by the host operating system). This is the default.

Notes:

The actual upper limit is determined by the host system’s architecture and operating system and the amount of physical memory and available paging space. The total of MAINSIZE and XPNDSIZE on host systems with a 32-bit architecture will be limited to 4G; host systems with a 64-bit architecture will be limited to less than 16E.

Using minimum storage sizes, storage sizes less than or not on a 64K boundary for architecture level ALS0 (S/370) or not on a 1M boundary for architecture level ALS1 (ESA/390) and higher, it may be possible to generate error conditions not covered by the “Principles of Operations”.

Use of storage sizes greater than supported by the guest operating system may generate incorrect results or error conditions within the guest operating system.

5.45.4 Overview Storage Allocation Units

Unit	Multiplier	Name (Symbol)	IEC Name (IEC Symbol)	Restrictions
B	None	Byte (B)	Byte (B)	
K	2**10	Kilobyte (kB)	Kibibyte (KiB)	
M	2**20	Megabyte (MB)	Mebibyte (MiB)	

Unit	Multiplier	Name (Symbol)	IEC Name (IEC Symbol)	Restrictions
G	2**30	Gigabyte (GB)	Gibibyte (GiB)	
T	2**40	Terabyte (TB)	Tebibyte (TiB)	Not on 32-bit machines
P	2**50	Petabyte (PB)	Pebibyte (PiB)	Not on 32-bit machines
E	2*60	Exabyte (EB)	Exbibyte (EiB)	Not on 32-bit machines

Table 6: Storage Allocation Units

5.45.5 Examples

Example 1:

Set the size of the main storage to 1024 MB. Do not lock the memory into the storage.

```
MAINSIZE 1024
  or
MAINSIZE 1024 UNLOCK
  or
MAINSIZE 1024M
  or
MAINSIZE 1024M UNLOCK
```

Example 2:

Set the size of the main storage to 4 GB. Lock the memory into the storage.

```
MAINSIZE 4096 LOCK
  or
MAINSIZE 4G LOCK
```

5.46 MANUFACTURER (STSI manufacturer code)

5.46.1 Function

MANUFACTURER specifies the manufacturer name returned by the STSI instruction. The default name is HRC.

5.46.2 Syntax

Descriptive

```
MANUFACTURER {HRC | name}
```

Diagram

```
►►── MANUFACTURER ──┬── HRC ──┬──────────────────────────────────────►◄
                    └── name ─┘
```

5.46.3 Parameter

name Any name with a maximum length of 16 characters. Valid characters are A-Z and 0-9. If MANUFACTURER is not specified in the configuration file then the default is HRC.

5.46.4 Examples

Example 1:

Set the manufacturer name returned by the STSI instruction to “HERC”.

```
MANUFACTURER HERC
```

5.47 MAXCPU (Maximum number of CPUs)

5.47.1 Function

The MAXCPU system parameter specifies the maximum number of installed processor engines. The combination of MAXCPU and NUMCPU controls the behaviour of how many CPU engines will be configured online upon startup and how many can be configured online later. The NUMCPU statement specifies the number of engines which will be configured online at startup time. All processors are CP engines unless otherwise specified by the ENGINES statement.

MAX_CPU_ENGINES is a compile-time variable which sets an upper limit on the value of MAXCPU. The value of MAX_CPU_ENGINES is displayed in the build information message on the Hercules control panel at startup time. To change the value of MAX_CPU_ENGINES you must rebuild Hercules. For Unix builds, specify "./configure --enable-multi-cpu=*nn*" before performing make. For Windows builds, specify "SET MAX_CPU_ENGINES=*nn*" before performing nmake.

MAX_CPU_ENGINES may be up to 128 on 64-bit Linux platforms. On Windows, and on all 32-bit platforms, the maximum value is 64. For performance reasons, values above 32 are not recommended for 32-bit platforms. If MAX_CPU_ENGINES is set to 1 then multiprocessing is disabled. See also the NUMCPU statement for a discussion of the performance implications of MAX_CPU_ENGINES.

The value of MAXCPU cannot exceed the value of MAX_CPU_ENGINES. If MAXCPU is not specified in the configuration file, then its initial value is equal to NUMCPU. If MAXCPU and NUMCPU are both omitted, then MAXCPU is set to 1.

For detailed explanations on the interrelationship between MAXCPU, ENGINES and NUMCPU please see “Appendix B. Configuration of Emulated CPUs”.

5.47.2 Syntax

Descriptive

```
MAXCPU {1 | nn}
```

Diagram

```
►►── MAXCPU ──┬─ 1 ──┬──────────────────────────────────────►◄
              └─ nn ─┘
```

5.47.3 Parameter

nn Specifies the maximum number of installed processor engines. The value of MAXCPU cannot exceed the value of MAX_CPU_ENGINES (see description above).

5.47.4 Examples

Example 1:

Set the maximum number of installed processor engines to 8.

MAXCPU 8

5.48 MAXRATES (MIPS/SIO rate reporting interval)

5.48.1 Function

MAXRATES sets the MIPS/SIO rate reporting interval. When the interval is expired a MAXRATES command is automatically issued. The current rates will also be displayed during shutdown of Hercules. The default for the MAXRATES interval (if not coded in the configuration file) is 1440 minutes (1 day).

5.48.2 Syntax

Descriptive

MAXRATES [1440 | *interval* | MIDNIGHT]

Diagram

```
►►── MAXRATES ──┬──── 1440 ────┬──────────────────────────────────────►◄
                ├── interval ──┤
                └── MIDNIGHT ──┘
```

5.48.3 Parameter

1440 This is the default that takes place if there is no MAXRATES interval is specified in the configuration file and sets the interval to 1440 minutes (1 day).

interval This specifies the interval time in minutes. Changes to the MAXRATES interval that are other than “MIDNIGHT” will set the current interval start time to the present; this includes a value of “1440”.

MIDNIGHT Sets the interval to 1440 minutes (1 day) and the start time for the interval timer to midnight of the current day. This will cause the MAXRATE statistics to be date aligned.

5.48.4 Examples

Example 1:

Set the MAXRATES interval to 60 minutes.

MAXRATES 60

Example 2:

Set the MAXRATES interval to one day and and the start time for the interval timer to midnight of the current day.

MAXRATES MIDNIGHT

5.49 MEMLOCK (Lock Hercules memory)

5.49.1 Function

The MEMLOCK system parameter is used to lock Hercules memory in storage. It defines the locking state of the allocated Hercules memory (page lock by host operating system). If the Hercules memory is locked in storage it cannot be paged out by the host operating system. This may result in some performance improvements.

This system parameter is available if Hercules is built with option “_HAVE_MLOCKALL”. Currently MEMLOCK is only supported under Linux host operating systems.

5.49.2 Syntax

Descriptive

`MEMLOCK {ON | OFF}`

Diagram

```
►►── MEMLOCK ──┬── ON ──┬──────────────────────────────────►◄
               └── OFF ─┘
```

5.49.3 Parameter

ON ON indicates that the memory is to be locked into storage.

OFF OFF indicates that the memory is not to be locked into the storage.

5.49.4 Examples

Example 1:

Lock the Hercules memory in the storage.

```
MEMLOCK ON
```

Example 2:

Do not lock the Hercules memory in the storage.

```
MEMLOCK OFF
```

5.50 MODEL (STSI model code)

5.50.1 Function

MODEL specifies the model names returned by the STSI instruction. The optional second, third and the fourth operands specify the capacity model name, the permanent capacity model name and the temporary capacity model name respectively.

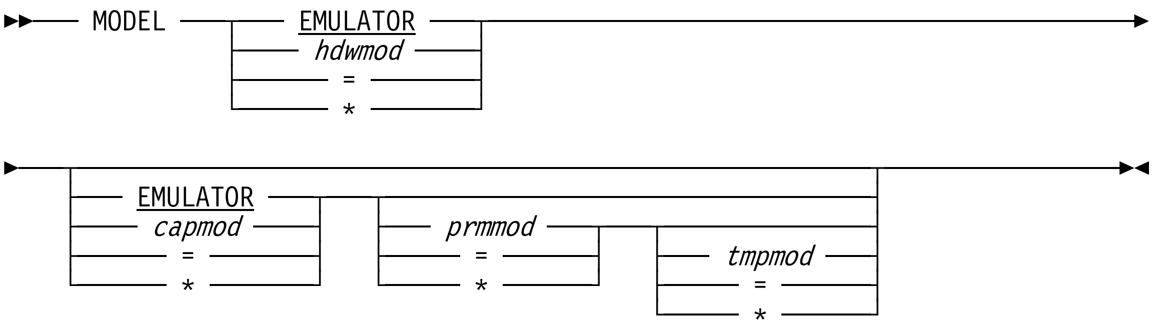
The default model name, if the MODEL system parameter is not coded, is “EMULATOR”.

5.50.2 Syntax

Descriptive

```
MODEL {EMULATOR | hdwmod | = | *}
      [EMULATOR | capmod | = | *
      [prmmod | = | *
      [tmpmod | = | *]]]
```

Diagram



5.50.3 Parameter

hdwmodel This specifies the hardware model name. This can be any name with a maximum length of 16 characters. Valid characters are A-Z and 0-9. Specifying a “=” resets the hardware model to “EMULATOR”; specifying an “*” leaves the current hardware model name intact. If the MODEL system parameter is not specified in the configuration file then the default name is “EMULATOR”.

capmodel This optional parameter specifies the capacity model name. The *capmodel* can be any name with a maximum length of 16 characters. Valid characters are A-Z and 0-9. Specifying a “=” copies the current hardware model name to the capacity model; specifying an “*” leaves the current capacity model name intact. The default capacity model name is “EMULATOR”.

prmmodel This specifies the permanent capacity model name. The *prmmodel* can be any name with a maximum length of 16 characters. Valid characters are A-Z and 0-9. Specifying a “=” copies the current capacity model name to the permanent capacity model; specifying an “*” leaves the current permanent capacity model name intact. The default permanent capacity model name is a null string.

tmpmodel This specifies the temporary capacity model name. The *tmpmodel* can be any name with a maximum length of 16 characters. Valid characters are A-Z and 0-9. Specifying a “=” copies the current permanent capacity model name to the temporary capacity model; specifying an “*” leaves the current temporary capacity model name intact. The default temporary capacity model name is a null string.

5.50.4 Examples

Example 1:

Set the model name returned by the STSI instruction to “EMULATOR”.

```
MODEL EMULATOR
```

Example 2:

Set all model names returned by the STSI instruction to “EMULATOR”.

```
MODEL EMULATOR EMULATOR EMULATOR EMULATOR
   or
MODEL EMULATOR = = =
```

Example 3:

Set all capacity model names returned by the STSI instruction to “HERCULES”, but leave the hardware model name as is.

```
MODEL * HERCULES HERCULES HERCULES
```

5.51 MODPATH (Dynamic load module path)

5.51.1 Function

MODPATH specifies the path where dynamic load modules are loaded from. If a MODPATH statement is coded then the path on the MODPATH statement is searched before the default path is searched. When a relative path is specified it is interpreted as a relative path within the default search path. If an absolute path is coded it is interpreted as such.

The default MODPATH is ‘hercules’ which means modules are loaded from the directory ‘hercules’ within the default LD_LIBRARY_PATH.

5.51.2 Syntax

Descriptive

MODPATH *path*

Diagram

```
►►── MODPATH ── path ──────────────────────────────────────────►◄
```

5.51.3 Parameter

path The path where dynamic load modules are loaded from.

5.51.4 Examples

Example 1:

Set the path from where dynamic load modules are loaded to “D:/HERCULES/MODIFICATIONS”.

```
MODPATH D:/HERCULES/MODIFICATIONS
```

5.52 MOUNTED_TAPE_REINIT (Control tape initialization)

5.52.1 Function

This system parameter specifies whether reinitialization of tape drive devices via the DEVINIT command in order to mount a new tape should be allowed if there is already a tape mounted on the drive. This option is meant as a safety mechanism to protect against accidentally unmounting a tape from the wrong device as a result of a simple typing error and thereby eventually cancelling a potentially important tape job. Please note that for SCSI tape drives the "DEVINIT nnnn *" command has no effect. The tape must be unmounted manually since it is a real physical device and not emulated via a disk file like '.AWS' or '.HET' tapes.

Specifying ALLOW, the default, indicates that new tapes may be mounted via "DEVINIT nnnn new-tape-filename" irrespective of whether or not there is already a tape mounted on the drive.

Specifying DISALLOW prevents new tapes from being mounted if one is already mounted. Before the new tape can be mounted the existing one has first to be unmounted (via the "DEVINIT nnnn *" command). Otherwise the DEVINIT attempt to mount the new tape is rejected.

5.52.2 Syntax

Descriptive

```
MOUNTED_TAPE_REINIT {ENABLE | ALLOW | DISABLE | DISALLOW}
```

Diagram

```
►►── MOUNTED_TAPE_REINIT ──┬── ENABLE ───┬──────────────────────────►◄
                           ├── ALLOW ────┤
                           ├── DISABLE ──┤
                           └── DISALLOW ─┘
```

5.52.3 Parameter

ENABLE Indicates that new tapes may be mounted irrespective of whether or not there is already a tape mounted on the drive. This is the default.

ALLOW This is the same as ENABLE.

DISABLE Prevents new tapes from being mounted if one is already mounted on the drive. Before the new tape can be mounted the currently mounted tape must first to be unmounted. Instead of DISABLE, the argument DISALLOW that has been used in earlier versions of Hercules can also be used.

DISALLOW This is the same as DISABLE.

5.52.4 Examples

Example 1:

Specify that new tapes are not being mounted if one is already mounted on the drive.

```
MOUNTED_TAPE_REINIT DISABLE
```

Example 2:

Specify that new tapes may be mounted even if one is already mounted on the drive.

```
MOUNTED_TAPE_REINIT ALLOW
```

5.53 MSGHLD (Timeout value of held messages)

5.53.1 Function

The MSGHLD system parameter is used to set the timeout value of held messages.

5.53.2 Syntax

Descriptive

MSGHLD *nn*

Diagram

```
►►—— MSGHLD —— nn ——————————————————————————————————————►◄
```

5.53.3 Parameter

nn This value specifies the new timeout value of held messages in seconds.

5.53.4 Examples

Example 1:

Set the timeout interval for held messages to 30 seconds.

```
MSGHLD 30
```

5.54 MSGLEVEL (Message display output)

5.54.1 Function

The MSGLEVEL system parameter specifies the setting of the message level. It decides how many and what kind of messages are written to the Hercules console (and to the log).

The message level is set per default to 'terse' which turns the verbose message level off. To display the additional messages during configuration file processing Hercules can be started with the "-v" option which sets the verbose message level. As an alternative, the MSGLEVEL system parameter can be set to activate the verbose message level. In this case however, MSGLEVEL must be coded as one of the first statements in the configuration file to take effect at an early stage during configuration file processing.

In addition to the 'terse' level the following options are set by default (if not otherwise overwritten through the MSGLEVEL system parameter or console command): 'nodebug', 'tape', 'dasd', 'comm', 'ur', 'scsi', 'ctca', 'graf', 'thread' and 'channel'.

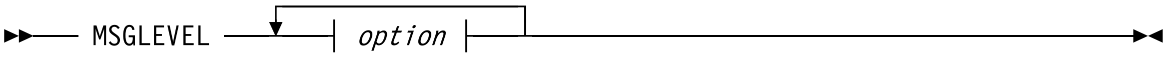
Certain levels (on, off, text, time) can only be set if Hercules is built with one of the following build options: OPTION_MSGCLR or OPTION_MSGHLD.

5.54.2 Syntax

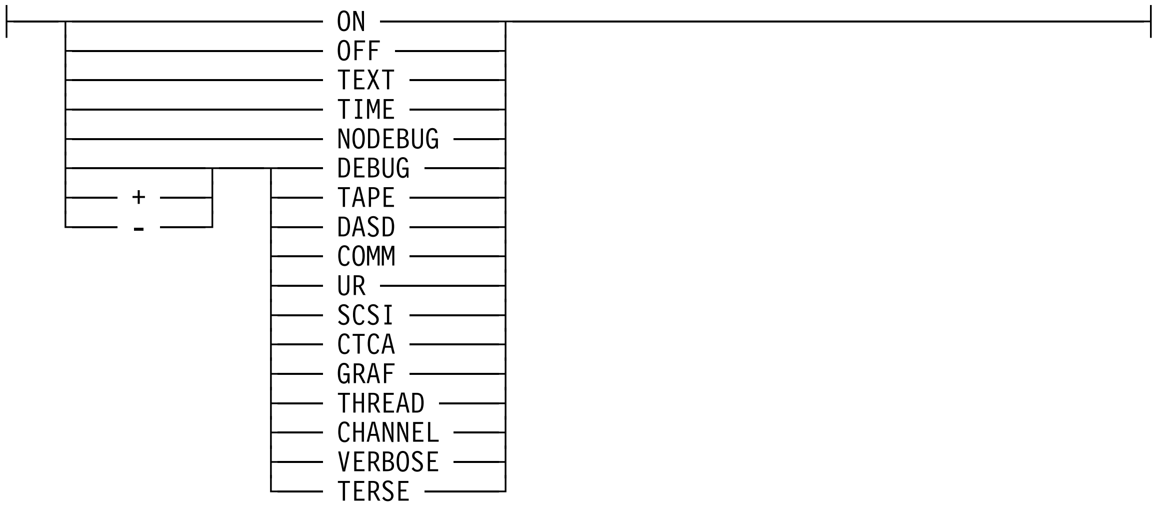
Descriptive

```
MSGLEVEL {option option ...}

   where option can be:

ON | OFF | TEXT | TIME | NODEBUG |
[+ | -] DEBUG   |
[+ | -] TAPE    |
[+ | -] DASD    |
[+ | -] COMM    |
[+ | -] UR      |
[+ | -] SCSI    |
[+ | -] CTCA    |
[+ | -] GRAF    |
[+ | -] THREAD  |
[+ | -] CHANNEL |
[+ | -] VERBOSE |
[+ | -] TERSE
```

Diagram



where *option* can be:



5.54.3 Parameter

ON	ON displays the messages in the default kind with message number followed by the message text.
OFF	No messages are displayed.
TEXT	Displays only the text part of the message (without message numbers).
TIME	Prefix the messages with a timestamp.
NODEBUG	The messages are not issued in debug mode (not prefixed additionally with the name of the source member and the line number that issues the message).
DEBUG	The messages are prefixed additionally with the name of the source member and the line number that issues the message.
TAPE	Display tape related messages.
DASD	Display DASD related messages.
COMM	Display communications related messages.
UR	Display unit record related messages.

SCSI Display SCSI related messages.

CTCA Display CTCA and LCS related messages.

GRAF Display graphics (3270) related messages.

THREAD Display threading related messages.

CHANNEL Display channel related messages.

VERBOSE Displays additional messages during configuration file processing.

TERSE This turns the verbose message level off. This is the default unless VERBOSE is specified either through the MSGLEVEL system parameter or panel command or if Hercules is started with the “-v” option.

5.54.4 Examples

Example 1:

Turn the verbose level off.

```
MSGLEVEL TERSE
```

Example 2:

Prefix messages with the name of the source member and the line number that issues the message (DEBUG level).

```
MSGLEVEL DEBUG
```

Example 3:

Display no messages at all.

```
MSGLEVEL OFF
```

Example 4:

Set a highly customized message processing. All messages have to be prefixed with a timestamp and additional messages during configuration file processing have to be displayed. SCSI related massaged as well as unit record related messages have to be suppressed.

```
MSGLEVEL TIME +DEBUG +VERBOSE –SCSI -UR
```

5.55 MSGLVL (Message display output)

5.55.1 Function

This parameter specifies the message level. MSGLVL is an alias for the MSGLEVEL system parameter. Please see MSGLEVEL for details.

5.55.2 Syntax

See MSGLEVEL command.

5.55.3 Parameter

See MSGLEVEL command.

5.55.4 Examples

See MSGLEVEL command.

5.56 NUMCPU (Number of emulated CPUs)

5.56.1 Function

NUMCPU specifies the number of emulated processor engines which will be configured online at startup time. The combination of NUMCPU and MAXCPU controls the behaviour of how many CPU engines will be configured online upon startup and how many can be configured online later.

NUMCPU cannot exceed the value of MAXCPU. If NUMCPU is less than MAXCPU then the remaining engines can be configured online later. The default NUMCPU value is 1. All processors are CP engines unless otherwise specified by the ENGINES system parameter.

Multiprocessor emulation works best if your host system actually has more than one physical CPU, but you can still emulate multiple CPUs nevertheless even on a uniprocessor system (and you might even achieve a small performance benefit when you do).

There is little point, however, in specifying NUMCPU greater than 1 unless your guest operating system (running under Hercules) is actually able to support multiple CPUs. If you do not actually need multiprocessor emulation, then setting MAX_CPU_ENGINES to 1 at compile time might even produce a slight performance advantage too.

For detailed explanations on the interrelationship between NUMCPU, MAXCPU and ENGINES please see “Appendix B. Configuration of Emulated CPUs”.

5.56.2 Syntax

Descriptive

NUMCPU {<u>1</u> | *nn*}

Diagram

```
►►── NUMCPU ──┬── 1 ──┬──────────────────────────────────────────►◄
              └── nn ─┘
```

5.56.3 Parameter

nn The number of emulated CPUs. NUMCPU must be less than or equal MAXCPU. If NUMCPU is larger than MAXCPU then an error message is issued, if it is less than MAXCPU then the remaining engines can be configured online later. The default for NUMCPU is 1.

5.56.4 Examples

Example 1:

Set the number of emulated CPUs to 4.

```
NUMCPU 4
```

5.57 NUMVEC (Number of vector facilities)

5.57.1 Function

NUMVEC defines the number of emulated vector facilities. Default is one per CPU (for the number of CPUs, see also the NUMCPU parameter). The vector facility is only available in ESA/390 mode by default.

5.57.2 Syntax

Descriptive

NUMVEC {0 | *nn*}

Diagram

```
►►── NUMVEC ───┬── 0 ──┬──────────────────────────────────────────►◄
               └── nn ─┘
```

5.57.3 Parameter

nn	The number of desired vector facilities.

5.57.4 Examples

Example 1:

Set the number of emulated vector facilities to 1.

```
NUMVEC 1
```

5.58 OSTAILOR (Tailor trace information for specific operating system)

5.58.1 Function

OSTAILOR lets you specify the intended operating system. The effect of this parameter is to reduce control panel message traffic by selectively suppressing trace messages for program checks which are considered normal in the specified environment.

The argument *QUIET* suppresses all exception messages whereas the argument *NULL* suppresses none of them. The other options do suppress some messages and do not suppress other messages depending on the specified operating system.

Prefix values with a plus (“+”) to combine them with existing values or with a minus (“-“) to exclude them from existing values. To combine values the OSTAILOR statement has to be coded several times. Excluding values (although possible) does not make sense in the configuration file.

See also the PGMTRACE console command which also to further fine tune the tracing of program interrupt exceptions.

5.58.2 Syntax

Descriptive

```
OSTAILOR {[+ | -] z/OS | OS/390 | VM | VSE | z/VSE | LINUX | OPENSOLARIS |
                  QUIET | NULL}
```

Diagram

```
►►── OSTAILOR ──┬───────┬──┬──── z/OS ─────┬──────────────────►◄
                ├── + ──┤  ├─── OS/390 ────┤
                └── - ──┘  ├───── VM ──────┤
                           ├──── VSE ──────┤
                           ├─── z/VSE ─────┤
                           ├─── LINUX ─────┤
                           ├─ OPENSOLARIS ─┤
                           ├─── QUIET ─────┤
                           └─── NULL ──────┘
```

5.58.3 Parameter

+ Specifies to combine the value withexisting values.

- Specifies to exclude the value from existing values.

z/OS Code z/OS if you intend to run z/OS.

OS/390	Code OS/390 if you intend to run MVS/370, MVS/XA, MVS/ESA, OS/390.
VM	Code VM if you intend to run VM/370, VM/ESA or z/VM.
VSE	Code VSE if you intend to run VSE/370 or VSE/ESA.
z/VSE	Code z/VSE if you intend to run z/VSE
LINUX	Code Linux if you intend to run Linux/390 or Linux for z/Series.
OpenSolaris	Code OpenSolaris you intend to run OpenSolaris for z/Series.
QUIET	QUIET discards all exception messages.
NULL	NULL allows all exception messages to be logged.

5.58.4 Examples

Example 1:

Specify Linux as the intended operating system and selectively suppressing trace messages for program checks which are considered normal in the Linux environment.

```
OSTAILOR LINUX
```

Example 2:

Do not specify the intended operating system and allow all exception messages to be logged.

```
OSTAILOR NULL
```

Example 3:

Specify VM, VSE and z/OS as the intended operating systems and selectively suppressing trace messages for program checks which are considered normal in these environments.

```
OSTAILOR VM
OSTAILOR +VSE
OSTAILOR +z/OS
```

5.59 PANRATE (Console refresh rate)

5.59.1 Function

PANRATE defines the rate, in milliseconds, at which the Hercules hardware console (HMC) will be refreshed. Please be aware that the panel refresh rate can reduce overall Hercules performance if PANRATE is set to fast.

5.59.2 Syntax

Descriptive

PANRATE {SLOW | FAST | *rate*}

Diagram

```
►►── PANRATE ──┬── SLOW ──┬──────────────────────────────────────────►◄
               ├── FAST ──┤
               └── rate ──┘
```

5.59.3 Parameter

SLOW SLOW is a synonym for a Hercules hardware console refresh rate of 500 milliseconds. This is also the default, if PANRATE is not specified.

FAST FAST is a synonym for a Hercules hardware console refresh rate of 50 milliseconds.

rate Any value between 1 (10) and 5000 milliseconds. A value less than the Linux system clock tick interval (10 on Intel platforms, 1 on Alpha platforms) or a value of more than 5000 will be rejected.

5.59.4 Examples

Example 1:

Set the panel refresh rate to 1 second.

```
PANRATE 1000
```

Example 2:

Set the panel refresh rate to 50 milliseconds.

```
PANRATE FAST
or
PANRATE 50
```

Example 3:

Set the panel refresh rate to 500 milliseconds.

```
PANRATE SLOW
or
PANRATE 500
```

5.60 PANTITLE (Console window title)

5.60.1 Function

PANTITLE specifies an optional console window title-bar string to be used in place of the default title supplied by the windowing system. This option allows one to distinguish between different Hercules sessions when running more than one instance of Hercules on the same machine.

The PANTITLE option takes effect only when the Hercules console is displayed on an ‘xterm’ terminal (commonly used on Unix systems) or in a Windows command prompt window. Note that this option has no effect when Hercules is run under the control of the Hercules Windows GUI since Hercules's console window is hidden in favour of using the GUI's window instead.

The default console title is a string consisting of the following information:

“LPARNAME – SYSTYPE * *SYSNAME* * *SYSPLEX* – System Status: *colour*”

SYSTYPE, SYSNAME and SYSPLEX are populated by the system call SCLP Control Program Identification. If any of these values is blank, then that field is not presented in the console title. The system status colour has following meanings:

RED	One or more CPUs are in wait state.
AMBER	One or more CPUs are not running.
GREEN	Everything is working correctly.

5.60.2 Syntax

Descriptive

PANTITLE {*text* | *"text text text"* | ""}

Diagram

```
►►── PANTITLE ──┬─────────── text ──────────┬──────────────────►◄
                ├─── "text text text" ──────┤
                └──────────── "" ───────────┘
```

5.60.3 Parameter

text Specifies the console window title-bar string to be used. If the value contains any blanks it must be enclosed within double-quotes ("). An empty string ("") will remove the default console title.

5.60.4 Examples

Example 1:

Set the console window title-bar string to “Hercules Emulator HMC”.

```
PANTITLE "Hercules Emulator HMC"
```

Example 2:

Set the console window title-bar string to “Hercules_HMC”.

```
PANTITLE Hercules_HMC
```

5.61 PGMPRDOS (LPP license setting)

5.61.1 Function

PGMPRDOS specifies whether or not Hercules will run licensed program product (LPP) ESA or z/Architecture operating systems.

5.61.2 Syntax

Descriptive

PGMPRDOS {RESTRICTED | LICENSED}

Diagram

```
►►── PGMPRDOS ──┬── RESTRICTED ──┬──────────────────────────────────────►◄
                └─── LICENSED ───┘
```

5.61.3 Parameter

RESTRICTED When PGMPRDOS is set to RESTRICTED, Hercules will stop all CPUs when a licensed program product operating systems is detected. RESTRICTED is the default.

LICENSED Setting PGMPRDOS to LICENSED will allow you to run licensed program product operating systems normally. This parameter has no effect on Linux/390, Linux for z/Series, or any 370-mode operating system.

If you are running Hercules under the Windows GUI a pop up window appears during startup which must be acknowledged before the startup continuous.

5.61.4 Examples

Example 1:

Allow licensed program product operating systems to run normally.

```
PGMPRDOS LICENSED
```

Example 2:

Disallow licensed program product operating systems to run.

```
PGMPRDOS RESTRICTED
```

5.62 PLANT (STSI plant code)

5.62.1 Function

PLANT specifies the plant name returned by the STSI instruction.

5.62.2 Syntax

Descriptive

PLANT {ZZ | *name*}

Diagram

```
►►── PLANT ──┬─── ZZ ───┬──────────────────────────────────────────────►◄
             └── name ──┘
```

5.62.3 Parameter

name Any name with a maximum length of four characters. Valid characters are A-Z and 0-9. If PLANT is not specified in the configuration file then the default name is ZZ.

5.62.4 Examples

Example 1:

Specify “HERC” as the plant name returned by the STSI instruction.

```
PLANT HERC
```

5.63 QUITMOUT (Quit timeout value)

5.63.1 Function

QUITMOUT is used for setting the timeout value for a second QUIT, EXIT or SSD command if Hercules is built with the option “OPTION_SHUTDOWN_CONFIRMATION”. If Hercules is built without this option, then the QUITMOUT system parameter cannot be specified in the configuration file.

5.63.2 Syntax

Descriptive

```
QUITMOUT nn
```

Diagram

```
►►── QUITMOUT ── nn ──────────────────────────────────────►◄
```

5.63.3 Parameter

nn This specifies the timeout value where *nn* must be in the range of 2 to 60 seconds. If the timeout value is 0 then no second QUIT, EXIT or SSD command is necessary.

5.63.4 Examples

Example 1:

Set the quit timeout value to 30 seconds.

```
QUITMOUT 30
```

5.64 REXX (REXX interpreter settings)

5.64.1 Function

The REXX system parameter is used to manage the Rexx interpreter settings. It allows to specify the paths where the Rexx executables can be found and what extensions for the executables are to be used. If Hercules is built with support for both Rexx environments (Regina Rexx and Open Object Rexx) then the desired environments can be enabled or disabled. The message prefixes to be used for Rexx messages (issued through the "say" command) and error messages (issued through the Rexx interpreter) can be set separately.

5.64.2 Syntax

Descriptive

REXX *option*

where *option* can be:

ENABLE | START [REGINA | OOREXX]

DISABLE | STOP

PATHS | REXXPATHS {*path* [*delimiter path ...*] | RESET}

SYSPATH {ON | OFF | RESET}

EXTENSIONS | SUFFIXES {*suffix* [*delimiter suffix ...*] | RESET}

RESOLVER {ON | OFF | RESET}

MSGLEVEL {0 | 1 | RESET}

MSGPREFIX {*messageprefix* | OFF | RESET}

ERRPREFIX {*errorprefix* | OFF | RESET}

MODE {COMMAND | SUBROUTINE}

Diagram

```
►►── REXX ── option ─────────────────────────────────────────────────►◄
```

where *option* can be:

```
├───┬── ENABLE ──┬───┬──────────────┬──────────────────────────────────┤
    └── START ───┘   ├── REGINA ──┤
                     └── OOREXX ──┘
```

```
├──┬── DISABLE ──┬─────────────────────────────────────────────────┤
   └─── STOP ────┘

├──┬──── PATHS ─────┬──┬─────┬── delimiter ──┐ ┬──────────────────┤
   └── REXXPATHS ───┘  │     ▼── path ───────┘ │
                       └──────── RESET ────────┘

├── SYSPATH ──┬── ON ─────┬──────────────────────────────────────┤
              ├── OFF ────┤
              └── RESET ──┘

├──┬── EXTENSIONS ──┬──┬─────┬── delimiter ──┐ ┬────────────────┤
   └─── SUFFIXES ───┘  │     ▼── suffix ─────┘ │
                       └──────── RESET ────────┘

├── RESOLVER ──┬── ON ─────┬─────────────────────────────────────┤
               ├── OFF ────┤
               └── RESET ──┘

├── MSGLEVEL ──┬── 0 ──────┬─────────────────────────────────────┤
               ├── 1 ──────┤
               └── RESET ──┘

├── MSGPREFIX ──┬── messageprefix ──┬────────────────────────────┤
                ├──────── OFF ──────┤
                └─────── RESET ─────┘

├── ERRPREFIX ──┬── errorprefix ──┬──────────────────────────────┤
                ├─────── OFF ─────┤
                └────── RESET ────┘

├── MODE ──┬──── COMMAND ────┬───────────────────────────────────┤
           └─── SUBROUTINE ──┘
```

5.64.3 Parameter

ENABLE Enables the Rexx environment that is specified as argument (Regina or ooRexx). This option is only available if Hercules is built with support for both Regina and ooRexx and cannot be used in a single Rexx environment.

If no environment is given as argument then the default Rexx environment (ooRexx) is started. ENABLE can be abbreviated as 'ENA'.

START This is the same as ENABLE. START can be abbreviated as 'STA'.

DISABLE Disables the currently active Rexx environment. This option is only available if Hercules is built with support for both Regina and ooRexx and cannot be used in a single Rexx environment.

DISABLE can be abbreviated as ‘DIS’.

STOP This is the same as DISABLE. STOP can be abbreviated as ‘STO’.

REGINA Given as argument to the ‘START’ or ‘ENABLE’ options, REGINA specifies that the Regina Rexx environment has to be started.

OOREXX Given as argument to the ‘START’ or ‘ENABLE’ option, OOREXX specifies that the Open Object Rexx (ooRexx) environment has to be started.

PATHS This is the keyword for specifying the search path(s) for the Rexx scripts. PATHS can be abbreviated as ‘PATH’.

REXXPATHS This is the same as PATHS. REXXPATHS can be abbreviated as ‘REXXP’.

path A path (or a list of paths, separated by a delimiter) in which Rexx executables will be searched. If the path is not specified when activating the Rexx environment then the default used is the current path taken from the environment variable ‘PATH’.

SYSPATH Keyword for specifying, if the search for the Rexx executables should be extended to the system paths (when set to on ‘ON’) or if it should be limited to the defined PATHS / REXXPATHS (when set to ‘OFF’).

SYSPATH is set to “ON” if not overwritten through the REXX system parameter in the configuration file or changed later on with the REXX console command.

EXTENSIONS This is the keyword for specifying the filename extension(s) to be used to search the Rexx executables. EXTENSIONS can be abbreviated as ‘EXT’.

If a given script name is in the format *filename.extension* then it is used as is without any further processing.

SUFFIXES This is the same as EXTENSIONS. SUFFIXES can be abbreviated as ‘SUF’.

suffix A filename extension (or a list of filename extensions, separated by a delimiter) that identifies a Rexx executable. A filename extension must be specified in the format “.*ext*”.

If there are no extensions specified the defaults used are ‘.REXX’, ‘.rexx’, ‘.REX’, ‘.rex’, ‘.CMD’, ‘.cmd’, ‘.RX’ and ‘.rx’.

delimiter This is the delimiter used for separating multiple paths or multiple extensions. For Linux and Mac OS-X systems this is the colon (“:”), for Windows systems it is the semicolon (“;”).

RESOLVER Keyword to define who will resolve the script name.

When set to “ON” then the Hercules Rexx interface will resolve the script name and issue appropriate messages in case the process fails. When set to ‘OFF’ then the script name will be passed as is to the Rexx interpreter.

RESOLVER is set to ‘ON’ if not overwritten through the REXX system parameter in the configuration file or changed later on with the REXX console command.

MSGLEVEL This is the keyword for specifying the message level to be used. MSGLEVEL can be abbreviated as ‘MSGL’.

0 Disables the display of the HHC17503I and HHC17504I messages. This is the default, if not explicitely set through the MSGLEVEL parameter.

1 Enables the display of the HHC17503I and HHC17504I messages when a script has finished:

HHC17503I REXX(*package name*) Exec/Script '*script name*' RetRC(0)
HHC17504I REXX(*package name*) Exec/Script '*script name*' RetValue'0'

MSGPREFIX This is the keyword used to set the prefix for standard messages (issued through ‘say’). MSGPREFIX can be abbreviated as ‘MSGP’. There is no message prefix set, unless exlicitely specified with the MSGPREFIX parameter.

msgprefix Specifies the Rexx standard message prefix to be used. *msgprefix* can be any string up to 9 characters. Embedded blanks are not allowed.

ERRPREFIX This is the keyword used to set the prefix for error messages. ERRPREFIX can be abbreviated as ‘ERRP’. There is no error prefix set, unless exlicitely specified with the ERRPREFIX parameter.

errorprefix Specifies the Rexx error message prefix to be used. *errorprefix* can be any string up to 9 characters. Embedded blanks are not allowed.

MODE This is the keyword used to specify the argument passing style to a Rexx script. If not explicitely specified, MODE is set to command style (‘COMMAND’).

COMMAND Specifies command style for passing arguments to a Rexx script. ‘COMMAND’ may be abbreviated as ‘COM’.

SUBROUTINE Specifies subroutine style for passing arguments to a Rexx script. ‘SUBROUTINE’ may be abbreviated as ‘SUB’.

ON Activates the specified option.

OFF Deactivates the specified option.

RESET Given as an argument to one of the options of the Rexx command this will reset the corresponding value to the default settings.

5.64.4 Examples

Example 1:

Enable ooRexx, set the filename extensions to be used to “.REXX” and “.REX” and set subroutine style for passing arguments to the Rexx script.

```
REXX ENABLE OOREXX
REXX EXTENSIONS .REXX;.REX
REXX MODE SUBROUTINE
```

Example 2:

Enable Regina Rexx and set the search path for the Rexx scripts to “D:\MVS\SCRIPTS”, but disable the search in the system path. Set the filename extensions to be used for the Rexx scripts to “.REXX”, “.REX” and “.RX”.

Finally set the message and error prefixes to RXMSG and RXERR and enable displaying the HHC17503I and HHC17504I messages after a Rexx script has been finished.

Use the default setting for the argument passing style.

```
REXX ENABLE REGINA
REXX PATH D:\MVS\SCRIPTS
REXX SYSPATH OFF
REXX EXTENSIONS .REXX;.REX;.RX
REXX MSGLEVEL 1
REXX MSGPREF RXMSG
REXX ERRPREF RXERR
```

5.65 SCLPROOT (SCLP base directory)

5.65.1 Function

The SCLPROOT system parameter sets the SCLP base directory. If a directory is given then SCLP disk I/O for the specified directory path is enabled. NONE disables SCLP disk I/O.

A subsequent list-directed IPL resets the path to the location of the .ins file, and a subsequent CCW-type IPL disables SCLP disk I/O.

5.65.2 Syntax

Descriptive

SCLPROOT {NONE | *directory*}

Diagram

```
►►── SCLPROOT ──┬─── NONE ──────┬──────────────────────────────────►◄
                └── directory ──┘
```

5.65.3 Parameter

NONE Disables SCLP disk I/O.

directory Specifies the directory from which SCLP disk I/O is allowed. A subsequent IPL of an ".ins" file or a subsequent CCW-type IPL will override this.

5.65.4 Examples

Example 1:

Disable SCLP disk I/O.

```
SCLPROOT NONE
```

Example 2:

Specifies "D:\SCLP\DISK" as the directory from which SCLP disk I/O is allowed.

```
SCLPROOT D:\SCLP\DISK
```

5.66 SCPECHO (Echo to console and history of SCP replies)

5.66.1 Function

SCPECHO allows it to route the SPC ('.') and priority SCP ('!') replies and responses to the Hercules console. The default is on (echo to the console) if SCPECHO is not coded.

5.66.2 Syntax

Descriptive

```
SCPECHO {OFF | ON}
```

Diagram

```
►►── SCPECHO ───┬── OFF ──┬──────────────────────────────────────────────►◄
                └── ON  ──┘
```

5.66.3 Parameter

OFF Do not route the SPC and priority SCP replies and responses to the Hercules console. This is the default, if SCPECHO is not coded in the configuration file.

ON Route the SPC and priority SCP replies and responses to the Hercules console.

5.66.4 Examples

Example 1:

Route SPC and priority SCP replies and responses to the Hercules console.

```
SCPECHO ON
```

5.67 SCPIMPLY (Pass non-Hercules commands to the SCP)

5.67.1 Function

SCPIMPLY allows it to pass all non-Hercules commands (commands unknown to Hercules) to the SPC if the SCP has enabled receipt of SCP commands. The default is off (no passing of unknown commands) if SCPIMPLY is not coded.

An example: The command "ping" is an unknown Hercules command. If the SCP running is SLES z/Linux then the ping would be sent as console input to SLES just as if the command has been prefixed with a '.' (period).

5.67.2 Syntax

Descriptive

`SCPIMPLY {OFF | ON}`

Diagram

```
►►── SCPIMPLY ──┬── OFF ──┬──────────────────────────────────────────►◄
                └── ON ───┘
```

5.67.3 Parameter

OFF Do not pass non-Hercules commands to the SCP. This is the default, if SCPIMPLY is not coded in the configuration file.

ON Pass all non-Hercules commands to the SCP if the SCP has enabled receipt of SCP commands.

5.67.4 Examples

Example 1:

Pass non-Hercules commands to the SCP.

```
SCPIMPLY ON
```

5.68 SCSIMOUNT (Automatic SCSI tape mounts)

5.68.1 Function

The SCSIMOUNT parameter specifies whether automatic detection of SCSI tape mounts is enabled or not.

NOTE! Enabling this option may negatively impact Hercules performance depending on how the host operating system (Windows, Linux etc.) handles SCSI attached tape drive status queries.

5.68.2 Syntax

Descriptive

SCSIMOUNT {NO | YES | *secs*}

Diagram

```
►►── SCSIMOUNT ──┬── NO ───┬──────────────────────────────►◄
                 ├── YES ──┤
                 └── secs ─┘
```

5.68.3 Parameter

NO NO indicates that the SCSIMOUNT option is disabled, forcing all SCSI tape mounts to be done manually via an appropriate DEVINIT command.

YES Yes enables the option and causes periodic queries of the SCSI tape drive in a five second interval to automatically detect when a tape is mounted. YES is equivalent with “SCSIMOUNT 5”.

secs A value from 1 to 99 seconds (inclusive) enables this option and causes periodic queries of the SCSI tape drive to automatically detect if a tape is mounted.

5.68.4 Examples

Example 1:

Enable automatic detection of SCSI tape mounts and set the periodic queries of the SCSI tape drive to a 30 seconds interval.

```
AUTO_SCSI_MOUNT 30
```

5.69 SHCMDOPT (Shell command option)

5.69.1 Function

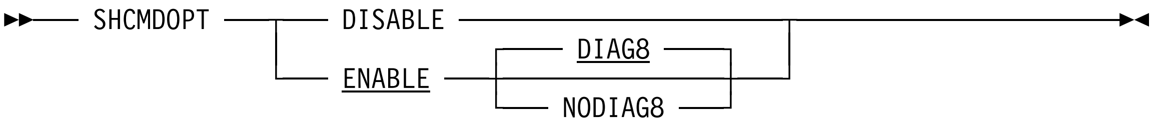
The SHCMDOPT system parameter defines the behaviour of the shell (sh) command. It specifies if shell commands (sh) are globally enabled or disabled either directly via the Hercules hardware console or programmatically via the DIAG8CMD interface.

5.69.2 Syntax

Descriptive

SHCMDOPT {DISABLE | ENABLE [DIAG8 | NODIAG8]}

Diagram



5.69.3 Parameter

DISABLE When set to DISABLE, shell commands (sh) are globally disabled and will result in an error if entered either directly via the Hercules hardware console or programmatically via the DIAG8CMD interface.

ENABLE When set to ENABLE, shell commands (sh) are globally enabled either directly via the Hercules hardware console or programmatically via the DIAG8CMD interface. This is the default.

DIAG8 When DIAG8 is specified (which is the default) the programmatic execution of shell commands (sh) via the DIAG8CMD interface is enabled. This is the default.

NODIAG8 When NODIAG8 is specified only the programmatic execution of shell commands via the DIAG8CMD interface is disabled; shell commands (sh) entered directly via the Hercules hardware console will still work.

NOTE: *"entered directly via the Hercules hardware console"* includes commands entered via the HTTP server facility or entered via "run command" (.rc) scripts.

5.69.4 Examples

Example 1:

Disable the programmatic execution of shell commands via the diagnose 8 interface but allow shell commands entered directly via the Hercules hardware console.

```
SHCMDOPT ENABLE NODIAG8
```

Example 2:

Globally disable the execution of shell commands.

```
SHCMDOPT DISABLE
```

Example 3:

Globally enable the execution of shell commands.

```
SHCMDOPT ENABLE DIAG8
```

5.70 SHOWDVOL1 (Enable showing of DASD volsers in device list)

5.70.1 Function

SHOWDVOL1 indicates whether to show the DASD VOL1 labels (volser) in the device list display. 'YES' shows the volser in addition to the usual filename, whereas 'NO' shows the device list in a traditional file-name only format. The 'ONLY' option shows only the volser; the filename is not shown at all. The default is 'NO', which results in a traditional device list display. Note: This system parameter is only available if Hercules is built with "OPTION_SHOWDVOL1".

5.70.2 Syntax

Descriptive

SHOWDVOL1 {<u>NO</u> | YES | ONLY}

Diagram

```
►►── SHOWDVOL1 ──┬── NO ───┬──────────────────────────────►◄
                 ├── YES ──┤
                 └── ONLY ─┘
```

5.70.3 Parameter

NO NO shows the device list in the traditional filename only format. This is the default.

YES YES shows the volser in addition to the usual filename.

ONLY ONLY shows only the volser, the filename is not shown at all.

5.70.4 Examples

Example 1:

Show the DASD VOL1 labels (volser) in the device list.

```
SHOWDVOL1 YES
```

Example 2:

Show only the DASD VOL1 labels (volser) in the device list, don't show the filenames.

```
SHOWDVOL1 ONLY
```

5.71 SHRDPORT (Shared device server port)

5.71.1 Function

SHRDPORT defines the port number (in decimal) on which the shared device server will listen. The shared device server will allow other Hercules instances to access devices on this instance. Currently only DASD devices may be shared. The default port is 3990. If you decide to use a different port number then you must specify this port number on the device statements for the other Hercules instances. If no SHRDPORT statement is present then the shared device server thread will not be activated.

5.71.2 Syntax

Descriptive

SHRDPORT {3990 | *port* | START | STOP}

Diagram

```
►►── SHRDPORT ──┬── 3990 ──┬──────────────────────────────────►◄
                ├── port ──┤
                ├── START ─┤
                └── STOP ──┘
```

5.71.3 Parameter

3990 This is the default port for the shared device server. If this port number is used then the port number is not required to be coded on the device statements for other Hercules instances.

port Any valid port number.

START Start the shared device server thread.

STOP Stop the shared device server thread.

5.71.4 Examples

Example 1:

Specify 3990 as the port used for the shared device server and start the shared device server thread.

```
SHRDPORT 3990
SHAREDPORT START
   or
SHRDPORT 3990
```

Example 2:

Specify 3990 as the port used for the shared device server, but do not start the shared device server thread upon Hercules startup so that it can be started later on manually through the SHRDPORT console command.

```
SHRDPORT 3990
SHRDPORT STOP
```

5.72 SRVPRIO (Server threads priority)

5.72.1 Function

The SRVPRIO parameter specifies the priority of the server threads. See section 5.82 “Process and Thread Priorities” for details.

Caution: SRVPRIO should not have a higher dispatching priority than the TOD clock and timer thread.

5.72.2 Syntax

Descriptive

SRVPRIO {4 | *nn*}

Diagram

```
►►── SRVPRIO ──┬── 4 ──┬──────────────────────────────────────────────────►◄
               └── nn ─┘
```

5.72.3 Parameter

4 Specifies a server threads process priority of 4. This is the default

nn This value specifies the process priority for the server threads. For details on the priority values see section 5.82 (“Process and Thread Priorities”). The default is 4.

5.72.4 Examples

Example 1:

Set the server threads process priority to 0.

```
HERCPRIO 0
```

5.73 SYMPTOM (Instruction trace display option)

5.73.1 Function

This parameter specifies the initial architecture mode. SYMPTOM is an alias for the TRACEOPTsystem parameter. Please see TRACEOPT for details.

5.73.2 Syntax

See TRACEOPT system parameter.

5.73.3 Parameter

See TRACEOPT system parameter.

5.73.4 Examples

See TRACEOPT system parameter.

5.74 SYSEPOCH (Base date for TOD clock)

5.74.1 Function

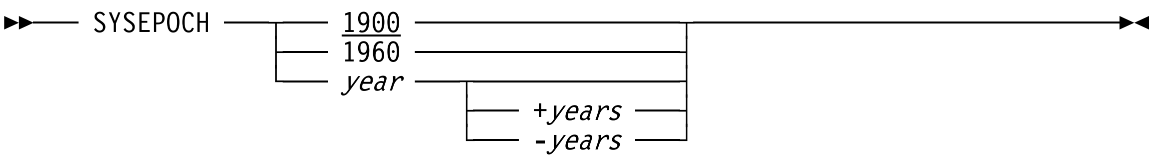
SYSEPOCH specifies the base date for the TOD clock. Use the default value (1900) for all systems except OS/360. Use 1960 for OS/360. Values other than these were formerly used to offset the TOD clock by a number of years to move the date before the year 2000 for non-Y2K-compliant operating systems. This use is deprecated and support will be removed in a future Hercules release after which only values of 1900 or 1960 will be accepted.

5.74.2 Syntax

Descriptive

`SYSEPOCH {`<u>`1900`</u>` | 1960 | `*`year`*` [`*`+years`*` | `*`-years`*`]}`

Diagram



5.74.3 Parameter

1900 Year 1900 is one of the two valid values for SYSEPOCH. 1900 is the default.

1960 Year 1960 is the second of the two valid values for SYSEPOCH.

year This is the base date for the TOD clock. The only supported values for SYSEPOCH are currently 1900 and 1960. Any other value will produce a warning message showing the equivalent values to specify in the SYSEPOCH statement. 1900 is the default.

+ *years* Specifies an optional positive year offset. It will be treated as though it had been specified using the YROFFSET statement.

- *years* Specifies an optional negative year offset. It will be treated as though it had been specified using the YROFFSET statement.

5.74.4 Examples

Example 1:

Specify year 1900 as the base date for the TOD clock.

```
SYSEPOCH 1900
```

Example 2:

Specify a positive year offset of 20 years to the default of 1900.

```
SYSEPOCH +20
```

5.75 TIMERINT (Internal timer update interval)

5.75.1 Function

The TIMERINT parameter sets the internal timer update interval in microseconds. This parameter specifies how frequently Hercules's internal timers-update thread updates the TOD clock, CPU Timer and other architectural related clock/timer values.

The default interval is 50 microseconds which strikes a reasonable balance between clock accuracy and overall host performance. The minimum allowed value is 1 microsecond and the maximum is 1,000,000 microseconds (one second).

CAUTION: *While a lower TIMERINT value may help increase the accuracy of the guest's TOD clock and CPU Timer values it could also have severe negative impact on the overall performance of the host operating system. This is especially true when a low TIMERINT value is coupled with a high HERCPRIO and TODPRIO priority setting. Exercise extreme caution when choosing your desired TIMERINT in relationship to your chosen HERCPRIO and TODPRIO priority settings.*

5.75.2 Syntax

Descriptive

TIMERINT {50 | *interval*}

Diagram

```
►►── TIMERINT ──┬──── 50 ─────┬─────────────────────────────────►◄
                └── interval ─┘
```

5.75.3 Parameter

interval Specifies the timer update interval in microseconds. The minimum allowed value for the interval is 1 microsecond and the maximum is 1'000'000 microseconds (one second.

CAUTION: While a lower TIMERINT value may help increase the accuracy of the guest's TOD clock and CPU Timer values, it could also have severe negative impact on the overall performance of the host operating system. This is especially true when a low value is coupled with a high HERCPRIO and TODPRIO priority setting.

Exercise extreme caution when choosing your desired TIMERINT in relationship to your chosen HERCPRIO and TODPRIO priority settings.

interval Specifies the timer update interval in microseconds. The minimum allowed value for the interval is 1 microsecond and the maximum is 1,000,000 microseconds (one second).

5.75.4 Examples

Example 1:

Set the timer update interval to 100 microseconds.

```
TIMERINT 100
```

5.76 TODDRAG (TOD clock drag factor)

5.76.1 Function

TODDRAG specifies the TOD clock drag factor. This parameter can be used to slow down or speed up the TOD clock by a factor of *factor*. A significant slowdown can improve the performance of some operating systems which consume significant amounts of CPU time processing timer interrupts. A drag factor of 2.0 slows down the clock by 50%, a drag factor of 0.5 doubles the speed of the clock, a drag factor of 1.01 slows down the clock by 1% and 0.99 speeds up the clock by 1%.

5.76.2 Syntax

Descriptive

TODDRAG {<u>1.000000</u> | *factor*}

Diagram

```
►►── TODDRAG ──┬── 1.000000 ──┬──────────────────────────────────────►◄
               └── factor ────┘
```

5.76.3 Parameter

factor The factor by which the TOD clock will be slowed down or sped up. The default factor is 1.000000.

5.76.4 Examples

Example 1:

Slow down the TOD clock by 50%.

```
TODDRAG 2
```

Example 2:

Double the speed of the TOD clock.

```
TODDRAG 0.5
```

5.77 TODPRIO (Timer thread process priority)

5.77.1 Function

With this parameter you can specify the priority of the TOD clock and the timer threads. See section 5.82 ("Process and Thread Priorities") for details.

Caution: TODPRIO should be given a dispatching priority equal to or higher than any other thread within Hercules (CPUPRIO, DEVPRIO, HERCPRIO, SRVPRIO).

5.77.2 Syntax

Descriptive

TODPRIO {-20 | *nn*}

Diagram

```
►►── TODPRIO ──┬── -20 ─┬──────────────────────────────────────►◄
               └── nn ──┘
```

5.77.3 Parameter

15 Specifies a TOD clock process prority of -20. This is the default

nn This value specifies the priority of the TOD clock and the timer thread. For details on the priority values see section 5.82 ("Process and Thread Priorities"). The default for TODPRIO is -20.

5.77.4 Examples

Example 1:

Set the priority of the TOD clock and the timer threads to -20.

```
TODPRIO -20
```

5.78 TRACEOPT (Instruction trace display option)

5.78.1 Function

TRACEOPT sets the Hercules instruction tracing display option. In addition to the TRACEOPT system parameter there is also a corresponding TRACEOPT console command to dynamically display and/or update the current setting at any time.

5.78.2 Syntax

Descriptive

TRACEOPT {TRADITIONAL | REGSFIRST | NOREGS}

Diagram

```
►►── TRACEOPT ───┬── TRADITIONAL ──┬──────────────────────────────────►◄
                 ├─── REGSFIRST ───┤
                 └──── NOREGS ─────┘
```

5.78.3 Parameter

TRADITIONAL TRADITIONAL (the default), displays the registers following the instruction about to be executed such that pressing enter (to execute the displayed instruction) then shows the next instruction to be executed followed by the updated registers display.

REGSFIRST REGSFIRST displays the current register contents followed by the instruction about to be executed such that pressing enter (to execute the displayed instruction) then shows the updated registers followed by the next instruction to be executed.

NOREGS NOREGS suppresses the registers display altogether and shows just the instruction to be executed.

5.78.4 Examples

Example 1:

Set the tracing display option to display the registers following the instruction about to be executed.

```
TRACEOPT TRADITIONAL
```

5.79 TZOFFSET (TOD clock offset from GMT)

5.79.1 Function

TZOFFSET is used to define the offset of the TOD clock from the current system time. For GMT use the default value (0000). For time zones west of Greenwich specify a negative value (example: -0500 for US Eastern Standard Time, -0800 for US Pacific Standard Time). For time zones east of Greenwich, specify a positive value (example: +0100 for Central European Time, +0930 for South Australian Time).

5.79.2 Syntax

Descriptive

TZOFFSET {<u>0000</u> | *+hhmm* | *-hhmm*}

Diagram

```
►►── TZOFFSET ──┬── 0000 ──┬──────────────────────────────────►◄
                ├── +hhmm ─┤
                └── -hhmm ─┘
```

5.79.3 Parameter

0000 GMT time (0000 is the default value). Please note that this is also the correct setting if your system time (the time of the operating system on which Hercules is running) is set to local time rather than GMT.

hhmm Use a positive time in hours and minutes for time zones east of Greenwich.

hhmm Use a negative time in hours and minutes for time zones west of Greenwich.

5.79.4 Examples

Example 1:

Set the offset of the TOD clock from the current system time to Central European Time.

```
TZOFFSET +0100
```

5.80 XPNDSIZE (Expanded storage size)

5.80.1 Function

XPNDSIZE specifies the expanded storage size. Storage is allocated in megabytes, unless a specific unit is specified. The actual upper limit of the expanded storage is determined by the host system's architecture, operating system, and on some systems the amount of physical memory and paging space you have available. The lower limit is 0.

The practical limit depends on the maximum amount of storage that can be obtained by "malloc" (usually around 1 GB on 32-bit platforms; on 64-bit platforms the value should only be limited by available paging space).

An additional optional argument determines the locking state of the allocated memory (page lock by host operating system). The LOCKED option indicates that the memory is to be locked into storage while UNLOCKED (the default) indicates that the memory is not locked into the storage.

Please note that Hercules preserves the last locking state of XPNDSIZE. Once storage is locked, any subsequent change to the expanded storage size will honor the existing lock state of memory unless the lock state is specified again on the XPNDSIZE command.

Caution: Do not lock expanded storage unless sufficient real memory is available to back up the request. Failure to do so may require the host system to be rebooted.

5.80.2 Syntax

Descriptive

XPNDSIZE *xsize*[M | G | T] [UNLOCK | LOCK]

Diagram

```
                                ┌──── M ────┐   ┌─── UNLOCK ───┐
►►─── XPNDSIZE ──── xsize ──────┼───────────┼───┴──── LOCK ────┴──────────────►◄
                                ├──── G ────┤
                                └──── T ────┘
```

5.80.3 Parameter

size The value of *xsize* must be a valid decimal number. The actual upper limit is determined by the host system's architecture, the operating system and on some systems the amount of physical memory and paging space that is available.

Storage sizes not on a 1M boundary are rounded up to the next 1M boundary. The lower limit and default is 0.

M 'M' determines that the number given is specified in megabytes (multiplier 2**20). This is the default if no unit is appended.

G ‘G’ determines that the number given is specified in gigabytes (multiplier 2**30).

T ‘T’ determines that the number given is specified in terabytes (multiplier 2**40). On 32-bit machines the unit terabytes is not available.

LOCK Attempt to lock the storage (pages locked by the host operating system).

UNLOCK Leave the store unlocked (no pages locked by the host operating system). This is the default.

Notes:

The actual upper limit is determined by the host system’s architecture and operating system, the guest operating system and the amount of physical memory and available paging space.

The total of MAINSIZE and XPNDSIZE on host systems with a 32-bit architecture will be limited to less than 4G; host systems with a 64-bit architecture will be limited to less than 16E.

Use of storage sizes greater than supported by the guest operating system may generate incorrect results or error conditions within the guest operating system.

5.80.4 Overview Storage Allocation Units

Unit	Multiplier	Name (Symbol)	IEC Name (IEC Symbol)	Restrictions
M	2**20	Megabyte (MB)	Mebibyte (MiB)	
G	2**30	Gigabyte (GB)	Gibibyte (GiB)	
T	2**40	Terabyte (TB)	Tebibyte (TiB)	Not on 32-bit machines

Table 7: Storage Allocation Units

5.80.5 Examples

Example 1:

Set the size of the expanded storage to 256 MB. Do not lock the memory into the storage.

```
XPNDSIZE 256
  or
XPNDSIZE 256M
  or
XPNDSIZE 256 UNLOCK
  or
XPNDSIZE 256M UNLOCK
```

Example 2:

Set the size of the expanded storage to 2 GB. Lock the memory into the storage.

```
XPNDSIZE 2048 LOCK
  or
XPNDSIZE 2G LOCK
```

5.81 YROFFSET (TOD clock offset from actual date)

5.81.1 Function

Specifies the number of years the TOD clock is offset from the actual date. Positive numbers will move the clock forward in time while negative numbers will move it backward. A common value for non-Y2K-compliant operating systems is YROFFSET -28 which has the advantage that the day of the week and the presence or absence of February 29 is the same as the current year.

5.81.2 Syntax

Descriptive

`YROFFSET {`*`+years`*` | `*`-years`*`}`

Diagram

```
►►── YROFFSET ──┬── +years ──┬──────────────────────────────────────►◄
                └── -years ──┘
```

5.81.3 Parameter

+years Specifies the number of years the TOD clock is offset positive from the actual date. This value may not be specified as greater than +/-142 years, the total range of the TOD clock. Specifying a value that causes the computed TOD clock year to be more than 142 years later than SYSEPOCH will produce unexpected results.

-years Specifies the number of years the TOD clock is offset positive from the actual date. This value may not be specified as greater than +/-142 years, the total range of the TOD clock. Specifying a value that causes the computed TOD clock year to be earlier than the value of SYSEPOCH will produce unexpected results.

5.81.4 Examples

Example 1:

Specify 28 years to offset the TOD clock from the actual date.

```
YROFFSET    -28
```

5.82 Process and Thread Priorities

This section covers details regarding the priority settings within Hercules. The relevant system parameters are:

- CPUPRIO
- DEVPRIO
- HERCPRIO
- SRVPRIO
- TODPRIO

5.82.1 Process Priorities

Under Linux a process is a thread and thread priority information applies instead.

For Windows the following conversions are used for translating Unix process priorities to Windows priority classes:

Unix Process Priority	**Windows Priority Class**	**Meaning**
-20 to -16	Realtime	Process that has the highest possible priority. The threads of the process preempt the threads of all other processes, including operating system processes performing important tasks. For example, a real-time process that executes for more than a very brief interval can cause disk caches not to flush or cause the mouse to be unresponsive.
-15 to -9	High	Process that performs time-critical tasks that must be executed immediately. The threads of the process preempt the threads of normal or idle priority class processes. An example is the Task List, which must respond quickly when called by the user, regardless of the load on the operating system. Use extreme care when using the high-priority class, because a high-priority class application can use nearly all available CPU time.
-8 to -1	Above Normal	Process that has priority above the Normal class but below the High class.
0 to 7	Normal	Process with no special scheduling needs.
8 to 15	Below Normal	Process that has priority above the Idle class but below the Normal class.
16 to 20	Low	Process whose threads run only when the system is idle. The threads of the process are preempted by the threads of any process running in a higher priority class. An example is a screen saver. The idle-priority class is inherited by child processes.

Table 8: Process Priority Conversions

Caution: On Windows, the value you choose for your process priority has a direct impact on how your thread priorities are interpreted! You should never modify one without understanding what impact you are doing so might have on the other!

5.82.2 Thread Priorities

On Linux/Unix hosts Hercules needs to be a setuid root program to allow it to reset its dispatching priority to a high (negative) value (i.e. "chown root.root hercules; chmod +s hercules").

For Windows the following conversions are used for translating Linux/Unix thread priorities to Windows thread priorities:

Unix Thread Priority	Windows Thread Priority	Meaning
-20 to -16	Time Critical	Base priority of 15 for Idle, Below Normal, Normal, Above Normal, or High class processes, and a base priority of 31 for Realtime class processes.
-15 to -9	Highest	Priority 2 points above the priority class.
-8 to -1	Above Normal	Priority 1 point above the priority class.
0 to 7	Normal	Normal priority for the priority class.
8 to 15	Below Normal	Priority 1 point below the priority class.
16 to 19	Lowest	Priority 2 points below the priority class.
20	Idle	Base priority of 1 for Idle, Below Normal, Normal, Above Normal, or High class processes, and a base priority of 16 for Realtime class processes.

Table 9: Thread Priority Conversions

Caution: On Windows, your Thread Priority is interpreted differently based on your chosen Process Priority setting! You should never modify your Thread Priority settings without first reviewing your chosen Process Priority setting!

6. Device Definition Descriptions

6.1 Local non-SNA 3270 Devices

6.1.1 Function

The local non-SNA 3270 device statements are used to define terminals to the Hercules configuration. There are no required arguments for this particular device type but there are several optional arguments that may be specified.

To use this device a tn3270 client must connect to the host machine via the port number specified on the CNSLPORT statement. A valid tn3270 device type such as IBM-3278 must be used.

If the tn3270 client software allows it to specify a device type suffix (e.g. *IBM-3278@GROUPNAME*) then the suffix can be used to connect to that specific device number, if defined. If no suffix is used then the tn3270 client will be connected to the first available and defined 3270 device.

If a specific terminal device address is specified via the device type suffix of the tn3270 client software then it must be eligible to connect at that device address or the connection is immediately rejected. An alternative terminal device for which the tn3270 client might be eligible will not be automatically selected.

6.1.2 Syntax

Descriptive

devaddr devtype [{*groupname* | *} [*ipaddr* [*mask*]]]

Diagram

```
►►── devaddr ── devtype ──────────────────────────────────────────►

►──┬─────────────────┬──┬──────────────────────────┬──────────────►◄
   ├─────── * ───────┤  └── ipaddr ──┬─────────────┤
   └── groupname ────┘               └── mask ─────┘
```

6.1.3 Parameter

devaddr This is the device address.

devtype This is the device type. Valid terminal device types are 3270 and 3278.

groupname If a terminal group name is given on the device statement then a device type suffix with this group name can be used to indicate that a device in this group is to be used. If it is specified as a terminal type suffix (e.g. IBM-3278@GROUPNAME) and there are no devices defined with that group name or no available devices remaining in that group then the connection is rejected. If no group name is specified as a terminal type suffix then the connection will only be eligible to terminal devices which also have no group name specified.

The terminal group name should be 1-8 alphanumeric characters in length, the first character being alphabetic and it should not be a hexadecimal number. Upper and lower case letters in the group name are considered to be equivalent.

***** The asterisk is used to indicate any terminal group name. It may be omitted if there are no additional arguments following the group name. If an IP address (and optionally mask) is specified and there is no specific group name desired then the asterisk must be coded to distinguish the terminal group name and the IP address arguments.

ipaddr [mask] The optional IP address and subnet masks specify the IP address(es) at which client(s) are allowed to connect via the device address identified by the device statement. This provides an additional means of specifying which device(s) a client tn3270 session may connect to.

If the IP address and mask (default 255.255.255.255 if not specified) of the tn3270 client attempting to connect match the IP address/mask entered on the device statement, then the client is eligible to connect at this device address. Otherwise the client is ineligible to connect at this device address and the next available device, if any, for which the client is eligible to connect, will be attempted.

If no permissible terminal devices remain (terminal devices for which the client is eligible to connect) or there are no more available terminal devices then the client connection is rejected.

The optional IP address and subnet mask may be specified in conjunction with the terminal group argument. In this case the terminal group argument must be specified ahead of the optional IP address and subnet mask arguments. To specify an IP address and a subnet mask without also specifying a terminal group use an asterisk (“*”) as the group name substitute or placeholder.

If an IP address / mask are not specified then any client tn3270 session is allowed to connect to the device (provided they are also a member of the specified terminal group, if any).

The terminal group name must match if specified, regardless of any optional IP address / mask. To summarize, the device number suffix always takes precedence over any group name and any group name always takes precedence over any IP address/mask value.

6.1.4 Examples

Example 1:

Define a 3270 device on device address 0200.

```
0200 3270
```

Example 2:

Define a 3270 device on device address 0200. Allow clients with any terminal group name but only from IP address 192.168.0.100 to connect.

```
0200 3270 * 192.168.0.100
```

Example 3:

Define a 3270 device on device address 0200. Allow clients with any terminal group name but only from IP address 192.168.0.100 with subnet mask 255.255.255.0 to connect.

```
0200 3270 * 192.168.0.100 255.255.255.0
```

Example 4:

Define a 3270 device on device address 0200. Allow only clients with terminal group name CONSOLE and from IP address 192.168.0.100 with subnet mask 255.255.255.0 to connect.

```
0200 3270 CONSOLE 192.168.0.100 255.255.255.0
```

Example 5:

Define a 3270 device on device address 0200. Allow only clients with terminal group name TSOTERM to connect.

```
0200 3270 TSOTERM
```

6.2 Integrated 3270 (SYSG) Console

6.2.1 Function

The integrated 3270 (SYSG) console is similar to a local non-SNA 3270 device, except that it is not addressed by a subchannel number and it is supported only by certain system control programs. The SYSG console is defined like a 3270 device except that the device type is SYSG and the device address is ignored. Only one SYSG console can be defined in a configuration.

Use tn3270 client software to connect to the SYSG console device via the port number specified on the CNSLPORT statement, just as you would connect to a regular local non-SNA 3270 device.

The SYSG console configuration statement recognizes optional arguments which specify group name and IP address in the same way as previously described for a local non-SNA 3270 device. These optional arguments provide a means to ensure that a given tn3270 client can connect directly to the SYSG console.

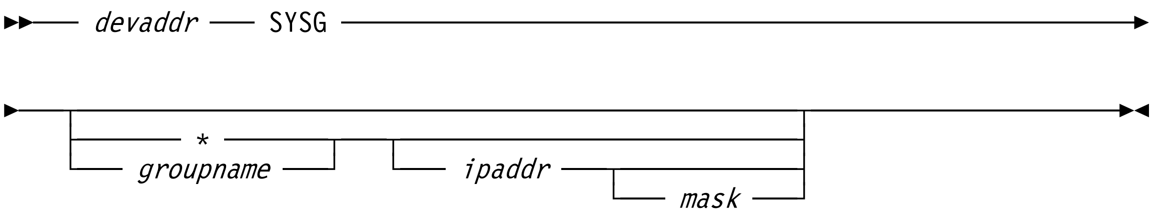
If the group name and IP address arguments are not specified, then the SYSG console is considered to be a member of the general pool of devices eligible for connection to any incoming tn3270 client.

6.2.2 Syntax

Descriptive

devaddr SYSG [{*groupname* | *} [*ipaddr* [*mask*]]]

Diagram



6.2.3 Parameter

devaddr This is the device address. In the case of the integrated 3270 (SYSG) console this address is ignored.

SYSG This is the device type. The only valid device type is SYSG.

groupname If a terminal group name is given on the device statement then a device type suffix with this group name can be used to indicate that a device in this group is to be used. If it is specified as a terminal type suffix (e.g. IBM-3278@GROUPNAME) and there are no devices defined with that group name or no available devices remaining in that group then the connection is rejected. If no group name is specified as a terminal type suffix then the connection will only be eligible to terminal devices which also have no group name specified.

The terminal group name should be 1-8 alphanumeric characters in length, the first character being alphabetic and it should not be a hexadecimal number. Upper and lower case letters in the group name are considered to be equivalent.

*****

The asterisk is used to indicate any terminal group name. It may be omitted if there are no additional arguments following the group name. If an IP address (and optionally mask) is specified and there is no specific group name desired then the asterisk must be coded to distinguish the terminal group name and the IP address arguments.

***ipaddr* [*mask*]**

The optional IP address and subnet masks specify the IP address(es) at which client(s) are allowed to connect via the device address identified by the device statement. This provides an additional means of specifying which device(s) a client tn3270 session may connect to.

If the IP address and mask (default 255.255.255.255 if not specified) of the tn3270 client attempting to connect match the IP address/mask entered on the device statement, then the client is eligible to connect at this device address. Otherwise the client is ineligible to connect at this device address and the next available device, if any, for which the client is eligible to connect, will be attempted.

If no permissible terminal devices remain (terminal devices for which the client is eligible to connect) or there are no more available terminal devices then the client connection is rejected.

The optional IP address and subnet mask may be specified in conjunction with the terminal group argument. In this case the terminal group argument must be specified ahead of the optional IP address and subnet mask arguments. To specify an IP address and a subnet mask without also specifying a terminal group use an asterisk (“*”) as the group name substitute or placeholder.

If an IP address / mask are not specified then any client tn3270 session is allowed to connect to the device (provided they are also a member of the specified terminal group, if any).

The terminal group name must match if specified, regardless of any optional IP address / mask. To summarize, the device number suffix always takes precedence over any group name and any group name always takes precedence over any IP address/mask value.

6.2.4 Examples

Example 1:

Define an integrated 3270 (SYSG) console on device address 0000 (the device address is ignored for SYSG consoles).

```
0000 SYSG
```

Example 2:

Define an integrated 3270 (SYSG) console on device address 0000. Allow only clients with terminal group name SYSGCONS and from IP address 192.168.0.100 with subnet mask 255.255.255.0 to connect.

```
0000 SYSG SYSGCONS 192.168.0.100 255.255.255.0
```

Example 3:

Define an integrated 3270 (SYSG) console on device address 0000. Allow clients with any terminal group name but only from IP address 192.168.0.100 to connect.

```
0000 SYSG * 192.168.0.100
```

Example 4:

Define an integrated 3270 (SYSG) console on device address 0000. Allow only clients with terminal group name SYSGCONS and from IP address 192.168.0.100 to connect.

```
0000 SYSG SYSGCONS 192.168.0.100
```

Example 5:

Define an integrated 3270 (SYSG) console on device address 0000. Allow only clients with terminal group name SYSGCONS to connect.

```
0000 SYSG SYSGCONS
```

6.3 Console Printer-Keyboard Devices

6.3.1 Function

These device statements are used to define “Console Printer-Keyboard” devices to Hercules. To use the Console Printer-Keyboard device a telnet client must connect to the host machine via the port number specified on the CNSLPORT statement in that configuration file.

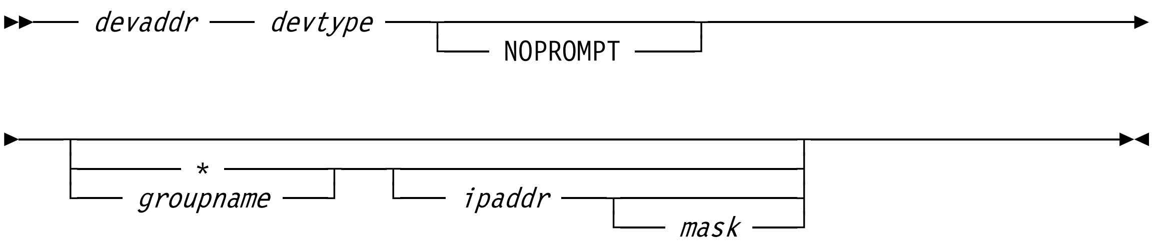
If the telnet client software allows it to specify a device type suffix (e.g. ansi@0009) then the suffix can be used to specify the specific 1052 or 3215 device to which the client should connect. If no suffix is used in the telnet client software (or the software does not allow it), then the client will be connected to the first available 1052 or 3215 device for which it is eligible.

6.3.2 Syntax

Descriptive

devaddr devtype [NOPROMPT] [{*groupname* | *} [*ipaddr* [*mask*]]]

Diagram



6.3.3 Parameter

devaddr This is the device address.

devtype This is the device type. Valid terminal device types are 1052 and 3215.

NOPROMPT The NOPROMPT argument may be specified to cause suppression of the “ENTER INPUT FOR CONSOLE DEVICE nnnn” prompt message which is otherwise normally issued to the device whenever the system is awaiting input from the device.

The NOPROMPT argument, if specified, must preceed the optional group name and any ipaddr mask arguments.

groupname If a terminal group name is given on the device statement then a device type suffix with this group name can be used to indicate that a device in this group is to be used. If it is specified as a terminal type suffix (e.g. IBM-3278@GROUPNAME) and there are no devices defined with that group name or no available devices remaining in that group then the connection is rejected. If no group name is specified as a terminal type suffix then the connection will only be eligible to terminal devices which also have no group name specified.

The terminal group name should be 1-8 alphanumeric characters in length, the first character being alphabetic and it should not be a hexadecimal number. Upper and lower case letters in the group name are considered to be equivalent.

***** The asterisk is used to indicate any terminal group name. It may be omitted if there are no additional arguments following the group name. If an IP address (and optionally mask) is specified and there is no specific group name desired then the asterisk must be coded to distinguish the terminal group name and the IP address arguments.

***ipaddr* [*mask*]** The optional IP address and subnet masks specify the IP address(es) at which client(s) are allowed to connect via the device address identified by the device statement. This provides an additional means of specifying which device(s) a client tn3270 session may connect to.

If the IP address and mask (default 255.255.255.255 if not specified) of the tn3270 client attempting to connect match the IP address/mask entered on the device statement, then the client is eligible to connect at this device address. Otherwise the client is ineligible to connect at this device address and the next available device, if any, for which the client is eligible to connect, will be attempted.

If no permissible terminal devices remain (terminal devices for which the client is eligible to connect) or there are no more available terminal devices then the client connection is rejected.

The optional IP address and subnet mask may be specified in conjunction with the terminal group argument. In this case the terminal group argument must be specified ahead of the optional IP address and subnet mask arguments. To specify an IP address and a subnet mask without also specifying a terminal group use an asterisk (“*”) as the group name substitute or placeholder.

If an IP address / mask are not specified then any client tn3270 session is allowed to connect to the device (provided they are also a member of the specified terminal group, if any).

The terminal group name must match if specified, regardless of any optional IP address / mask. To summarize, the device number suffix always takes precedence over any group name and any group name always takes precedence over any IP address/mask value.

6.3.4 Examples

Example 1:

Define a 1052 console printer-keyboard device on device address 0009.

```
0009 1052
```

Example 2:

Define a 1052 console printer-keyboard device on device address 0009 and suppress the “ENTER INPUT FOR CONSOLE DEVICE nnnn” prompt messages.

```
0009 1052 NOPROMPT
```

Example 3:

Define a 3215 console printer-keyboard device on device address 0009. Allow only clients with terminal group name CONSOLES and from IP address 192.168.0.100 to connect.

```
0009 3215 CONSOLES 192.168.0.100 255.255.255.0
```

Example 4:

Define a 3215 console printer-keyboard device on device address 0009. Allow only clients with terminal group name CONSOLES and from IP address 192.168.0.100 with subnet mask 255.255.255.0 to connect. Additionally suppress the “ENTER INPUT FOR CONSOLE DEVICE nnnn” prompt messages.

```
0009 3215 NOPROMPT CONSOLES 192.168.0.100 255.255.255.0
```

Example 5:

Define a 3215 console printer-keyboard device on device address 0009. Allow clients with any terminal group name but only from IP address 192.168.0.100 with subnet mask 255.255.255.0 to connect. Additionally suppress the “ENTER INPUT FOR CONSOLE DEVICE nnnn” prompt messages.

```
0009 3215 NOPROMPT * 192.168.0.100 255.255.255.0
```

6.4 Integrated Console Printer-Keyboard Devices

6.4.1 Function

These device statements are used to define “Integrated Console Printer-Keyboard” devices to the Hercules configuration. The statements have one optional argument, the default command prefix for sending input to the device. The default command prefix is “/”. All integrated devices must use a different command prefix.

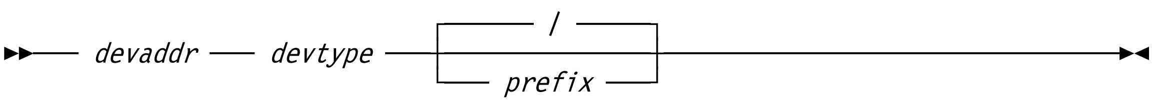
To send a logon command to a 1052-C or 3215-C enter “/logon” on the Hercules console.

6.4.2 Syntax

Descriptive

devaddr devtype [{*prefix* | /}]

Diagram



6.4.3 Parameter

devaddr This is the device address.

devtype This is the device type. Valid terminal device types are 1052-C and 3215-C.

prefix This is the command prefix for the device. The default command prefix is “/”. All devices must use a different command prefix.

6.4.4 Examples

Example 1:

Define a 3215-C integrated console printer-keyboard device on device address 0009. Specify "/" as command prefix.

```
0009 3215-C /
```

6.5 Card Reader Devices

6.5.1 Function

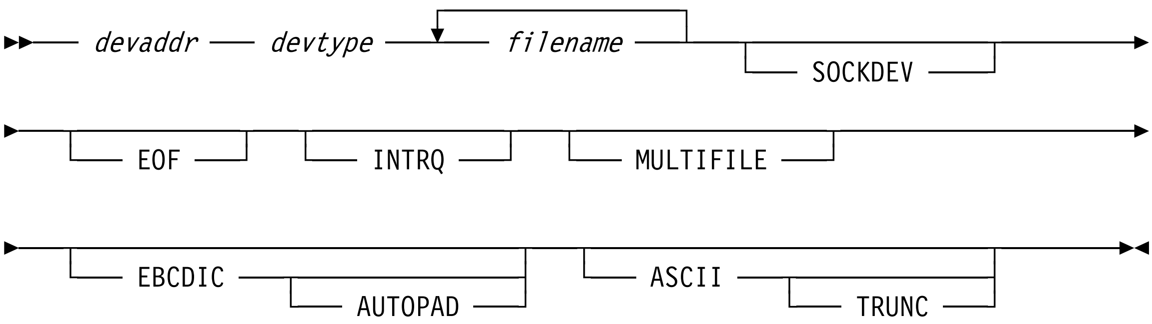
The card reader device statements are used to define card readers to the Hercules configuration. The argument specifies a list of file names containing card images. Additional arguments can be defined after the file names.

6.5.2 Syntax

Descriptive

devaddr devtype filename [*filename* ...] [SOCKDEV] [EOF] [INTRQ]
[MULTIFILE] [EBCDIC [AUTOPAD]] [ASCII [TRUNC]]

Diagram



6.5.3 Parameter

devaddr This is the device address.

devtype This is the device type. Valid card reader types are 1442, 2501, 3505.

filename The name (and optionally the path) of the card reader file.

SOCKDEV Indicates that the card reader is a socket device, wherein the filename is actually a socket specification instead of a device filename. When used, there must be only one filename specified in the form “*port*”, “*host:port*” or “*sockpath/sockname*”. The device then accepts remote connections on the given TCP/IP port or Unix domain socket and reads data from the socket instead from a device file. This allows automatic remote submission of card reader data.

See chapter 15 ("Submitting Jobs via the Socket Reader") for more details.

EOF EOF specifies that unit exception status is presented after reading the last card in the file. This option is persistent and will remain in effect until the reader is reinitialized with the INTRQ option.

INTRQ INTRQ specifies that unit check status with intervention required sense bytes is presented after reading the last card in the file. This option is persistent and will remain in effect until the reader is reinitialized with the EOF option.

MULTIFILE Specifies, when multiple files are entered, to automatically open the next input file and continue reading whenever EOF is encountered on a given file. If not specified reading stops once EOF is reached on a given file and an attention interrupt is required to open and begin reading the next file.

EBCDIC Specifies that the file contains fixed length 80-byte EBCDIC records with no line-end delimiters.

ASCII Specifies that the file contains variable length lines of ASCII characters delimited by LF (line feed) or CRLF (carriage return line feed) sequences at the end of each line.

If neither EBCDIC nor ASCII is coded then the device handler attempts to detect the format of the card image file when the device is first accessed. Auto-detection is not supported for socket devices and the default is ASCII if SOCKDEV is specified.

TRUNC With ASCII files, TRUNC defines, that lines longer than 80 characters are truncated instead of producing a unit check error.

AUTOPAD With EBCDIC files, AUTOPAD defines, that the file is automatically padded to a multiple of 80 bytes if necessary.

6.5.4 Examples

Example 1:

Define a 3505 card reader device on device address 000C. The name and path of the card reader file is "D:/JCL/DUMMY.JCL". A unit exception status has to be presented after reading the last card in the file.

```
000C 3505 D:/JCL/DUMMY.JCL EOF
```

Example 2:

Define a 1442 card reader device on device address 000C. The name and path of the card reader file is "D:/JCL/JCL.TXT". The file contains fixed length 80-byte EBCDIC records, automatically padded to a multiple of 80 bytes if necessary. A unit check status with intervention required sense bytes has to be presented after reading the last card in the file.

```
000C 1442 D:/JCL/JCL.TXT EBCDIC AUTOPAD INTRQ
```

Example 3:

Define a 2501 card reader device on device address 000C. The card reader is a socket device accepting remote connections from address 127.0.0.1 port 2501. The file contains variable length lines of ASCII characters delimited by line feed or carriage return line feed sequences at the end of each line. Lines longer than 80 bytes have to be truncated and a unit exception status has to be presented after reading the last card in the file.

```
000C 2501 127.0.0.1:2501 SOCKDEV ASCII TRUNC EOF
```

6.6 Card Punch Devices

6.6.1 Function

The card punch statement defines a card punch device to the Hercules configuration. The argument specifies the name of the file to which the punched output will be written. Additional arguments may be specified after the filename.

6.6.2 Syntax

Descriptive

devaddr devtype filename [ASCII] [CRLF] [NOCLEAR]

Diagram

```
►►── devaddr ── devtype ── filename ──┬─────────┬──────────────────►
                                      └─ ASCII ─┘

►──┬────────┬──┬───────────┬───────────────────────────────────────►◄
   └─ CLRF ─┘  └─ NOCLEAR ─┘
```

6.6.3 Parameter

devaddr This is the device address.

devtype This is the device type. A valid card punch type is 3525.

filename The name (and optionally the path) of the card punch file.

ASCII ASCII specifies that the file will be written as variable length lines of ASCII characters, delimited by line feeds or carriage return line feed sequences. Trailing blanks are removed from each line. If the ASCII argument is not specified, the file is written as fixed length 80-byte EBCDIC records with no line-end delimiters.

CRLF This optional parameter specifies that carriage return line feed sequences are written at the end of each line. If the CRLF argument is not specified, line feeds are only written at the end of each line.

NOCLEAR This argument specifies that the output file will not be cleared to zero bytes when it is opened. If NOCLEAR is not specified, then any previous content of the file is destroyed when the file is opened for output.

6.6.4 Examples

Example 1:

Define a 3525 card punch device on device address 000D. The punched output has to be written to the file "D/PCH/PCH1.TXT" as variable length lines of ASCII records delimited by carriage return line feed sequences. Any existing output in the file has to be kept when the file is opened for output.

```
000D 3525 D:/PCH/PCH1.TXT ASCII CRLF NOCLEAR
```

Example 2:

Define a 3525 card punch device on device address 000D. The punched output has to be written to the file "D/PCH/PCH2.TXT" as fixed length 80-byte EBCDIC records with no line-end delimiters. Any existing output in the file will be destroyed when the file is opened for output.

```
000D 3525 D:/PCH/PCH2.TXT
```

6.7 Line Printer Devices

6.7.1 Function

The line printer device statement defines a printer to the Hercules configuration. The argument specifies the name of a file to which the printer output will be written. The output is written in the form of variable length lines of ASCII characters, delimited by line feeds or carriage return line feed sequences. Trailing blanks are removed from each line. Carriage control characters are translated to blank lines or ASCII form feed characters. If the file exists it will be overwritten.

If the filename begins with the vertical bar '|' pipe character, then it is removed and the remainder of the filename is interpreted as a command line (the name of a program or batch file followed by any necessary arguments) to which to "pipe" the printer output to. This is known as the "print-to-pipe" feature. All printer output is then sent to the piped programs stdin input, and all of the piped programs stdout and stderr output is piped back to Hercules for displaying on the hardware console.

SOCKDEV indicates that the line printer is a socket device wherein the filename is actually a socket specification in the form *host:port* instead of a device filename. The device then accepts remote connections on the given TCP/IP port and writes data to the socket instead to a device file. A Windows software that supports a socket printer is Fish's "HercPrt" (Hercules Remote Printer Spooler). It spools the print output from a Hercules socket printer and creates either PDF, RTF or just plain text files. A control file enables HercPrt to split the printouts on job separator page boundaries and name the file according the actual job accounting field values.

6.7.2 Syntax

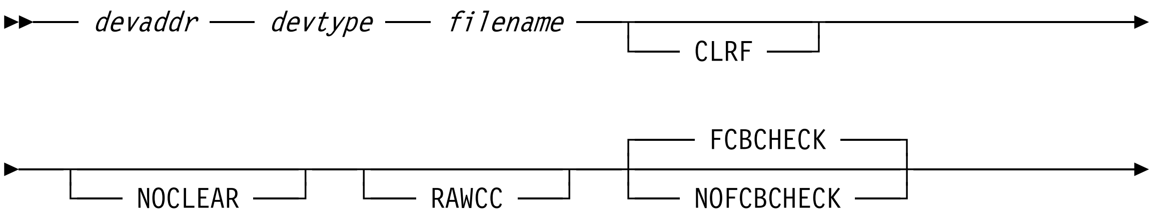
Descriptive

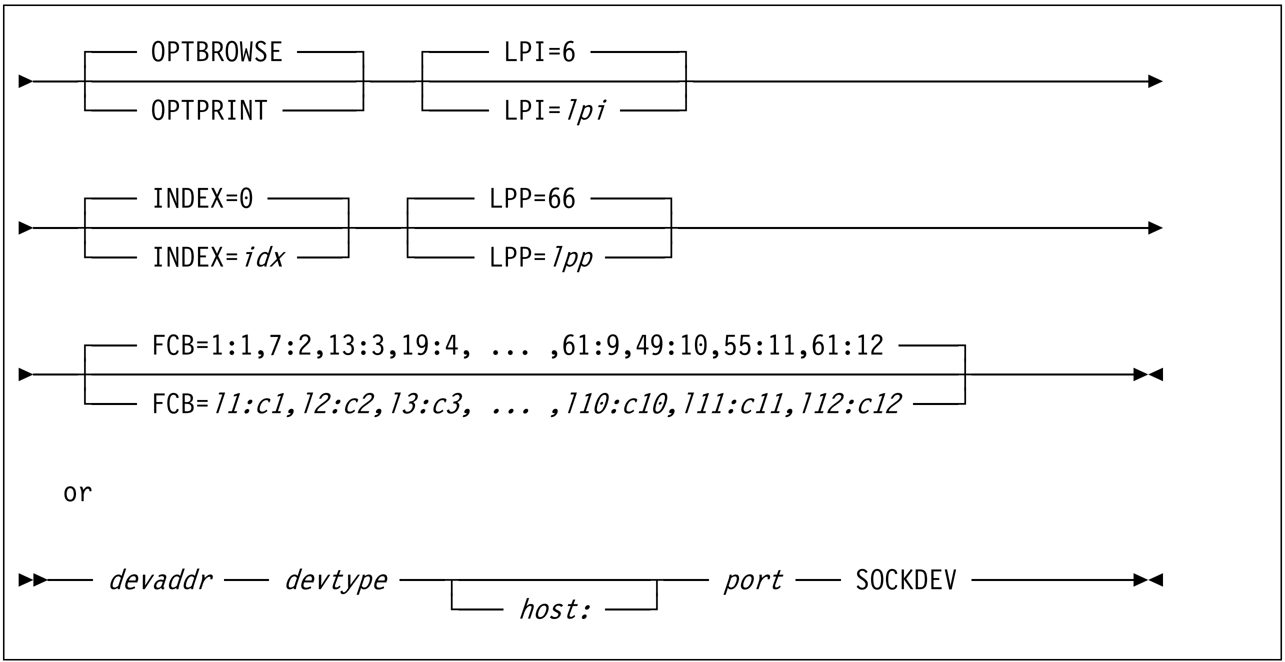
devaddr devtype filename [CRLF] [NOCLEAR] [RAWCC] [FCBCHECK | NOFCBCHECK]
[OPTBROWSE | OPTPRINT] [LPI={6 | *lpi*}] [INDEX={0 | *idx*}] [LPP={66 | *lpp*}]
[FCB={1:1,07:2,13:3,19:4,25:5,31:6,37:7,43:8,61:9,49:10,55:11,61:12 |
l1:c1,l2:c2,l3:c3, ... ,l10:c10,l11:c11,l12:c12}]

or

devaddr devtype [*host:*]*port* SOCKDEV

Diagram





6.7.3 Parameter

devaddr This is the device address.

devtype This is the device type. Valid printer types are 1403 and 3211.

filename The name (and optionally the path) of the printer file or the “print-to-pipe” command line. If the filename begins with the vertical bar '|' pipe character, then it is removed and the remainder of the filename is interpreted as a command line (the name of a program or batch file followed by any necessary arguments) to which to "pipe" the printer output to.

This is known as the "print-to-pipe" feature. All printer output is then sent to the piped programs stdin input, and all of the piped programs stdout and stderr output is piped back to Hercules for displaying on the hardware console.

If the "print-to-pipe" command line contains arguments, then quotes must be placed around the entire filename string including the vertical bar. If the "print-to-pipe" command line itself contains quotes, then the command line must be enclosed in apostrophes instead of quotes.

CRLF This optional parameter specifies that carriage return line feed sequences are written at the end of each line. If the CRLF argument is not specified then line feeds only are written at the end of each line.

NOCLEAR This argument specifies that the output file will not be cleared to zero bytes when it is opened. If NOCLEAR is not specified, then any previous content of the file is destroyed when the file is opened for output.

RAWCC RAWCC specifies that printer output CCWs are not to be interpreted, but simply dumped in hex to the printer output file. This is useful for debugging. Default is to interpret printer CCWs normally.

FCBCHECK This argument specifies that an attempt to skip to a FCB channel for which no line number has been set will cause the command to be rejected with a unit check. This is the default.

NOFCBCHECK This argument specifies that an attempt to skip to a FCB channel for which no line number has been set will cause the next line of output printed on the next line on the printer output. The opposite, FCBCHECK, is the default.

OPTBROWSE Printing is optimized for browsing. This is the default.

OPTPRINT Printing is optimized for printed output.

LPI= Specifies the number of lines per inch. The value of *lpi* must be 6 or 8. The default number of lines per inch is 6.

INDEX= Sets the 3211 indexing. The value of *idx* must be in the range of 0 - 31. The default is 0.

LPP= Specifies the number of lines per page. The value of *lpp* must be numeric but is not further checked. Any number of lines per page is allowed. The default number of lines per page is 66.

FCB= FCB specifies an initial FCB image to use for the printer. The argument must be given in the form *l1:c1, ... ,l12:c12* where 'l' and 'c' are numeric. 'l' is the line number and 'c' is the assigned channel. There is a maximum of 12' l:c' pairs allowed.

The default is FCB=1:1,07:2,13:3,19:4,25:5,31:6,37:7,43:8,61:9,49:10,55:11,61:12.

host This is the hostname or the IP address of the socket device.

port This is the port number of the socket device.

SOCKDEV SOCKDEV indicates that the line printer is a socket device wherein the filename is actually a socket specification instead of a device filename.

When used, there must only be one filename specified in the form *port* or *host:port*. The device then accepts remote connections on the given TCP/IP port and writes data to the socket instead to a device file.

This allows automatic remote spooling of line printer data. The sockdev option is mutually exclusive with all other printer options (like CRLF etc.) and must be specified alone.

6.7.4 Examples

Example 1:

Define a 1403 line printer device on device address 000E. The printed output has to be written to the file "D:\PRT\PRT1.TXT" with carriage return line feed sequences at the end of each line. Any existing output in the file has to be kept when the file is opened for output.

```
000E 1403 D:/PRT/PRT1.TXT CRLF NOCLEAR
```

Example 2:

Define a 3211 line printer device on device address 000F. The printed output has to be written to the file "D:\PRT\PRT2.TXT" with line feeds only at the end of each line. Any existing output in the file will be over-written when the file is opened for output.

```
000F 3211 D:/PRT/PRT2.TXT
```

Example 3:

Define a 1403 line printer device on device address 000E. The printed output has to be written to the file "D:\PRT\PRT1.TXT" with carriage return line feed sequences at the end of each line. A specific FCB has to be used and a skip to a FCB channel for which no line number has been set (channels 7 and 8) has to cause a unit check. The number of lines per inch must be 6 and the number of lines per page is 66. The output has to be optimized for printing.

```
000E 1403 D:/PRT/PRT1.TXT CRLF FCBCHECK LPI=6 LPP=66 OPTPRINT
          FCB=1:1,07:2,13:3,19:4,25:5,31:6,61:9,49:10,55:11,61:12
```

Example 4:

Define a 1403 line printer device on device address 000E. The line printer is a socket device with the IP address 192.168.0.199 and port 1403. Output is written to the socket instead to a device file.

```
000E 1403 192.168.0.199:1403 SOCKDEV
```

Example 5:

Define a 1403 line printer device on device address 000E. The command line for the print-to-pipe feature is “/usr/bin/lpr”, the argument for the ‘lpr’ program is “-Phplj” (Unix example). In the case of the Windows example the command line for the print-to-pipe feature is “C:\utils\pr”, the argument for the ‘pr’ program is “-PLPTP1:” . Each printed line will have a carriage return line feed sequence at the end.

```
000E 1403 "|/usr/bin/lpr -Phplj" crlf         (for Unix)

000E 1403 "|c:\utils\pr -s -PLPT1:" crlf      (for Windows)
```

6.8 Emulated Tape Devices

Tape device statements are used to define tape devices to the Hercules configuration. Five types of emulation are supported:

- SCSI tapes
- Optical Media Attach (OMA) virtual files
- AWSTAPE virtual files
- HET virtual files
- Fake Tape virtual files

6.8.1 SCSI Tapes

6.8.1.1 Function

When defining SCSI tapes the argument specifies the tape device name (usually **/dev/nst0**). SCSI tapes are read and written using variable length EBCDIC blocks and filemarks exactly like a mainframe tape volume (see also the AUTO_SCSI_MOUNT system parameter).

6.8.1.2 Syntax

Descriptive

devaddr devtype devname [--no-erg] [--blkid-32 | --blkid-22]

Diagram

```
►►── devaddr ── devtype ── devname ──┬──────────────┬──────────────►
                                     └── --no-erg ──┘

►──┬─────────────────┬─────────────────────────────────────────────►◄
   ├── --blkid-32 ───┤
   └── --blkid-22 ───┘
```

6.8.1.3 Parameter

devaddr This is the device address.

devtype This is the device type. Valid device types are 3410, 3420, 3422, 3430, 3480, 3490, 8809 and 9347.

devname The tape device name (usually “/dev/nst0” on Linux or “\\.\Tape0” on Windows).

--no-erg This option is intended to prevent issuance of the Erase Gap command to those SCSI tape drives that do not support it (e.g. Quantum DLT series). It causes Hercules's device emulation logic to ignore any ‘Erase Gap’ commands issued to the drive and to return immediate success instead.

This option should only be used (specified) for drives such as the Quantum which support switching from read mode to write mode in the middle of a data stream without the need of an intervening ‘Erase Gap’ command. Specifying it for any other model SCSI drive may cause incorrect functioning as a result of the Erase Gap command not being issued to the actual SCSI hardware.

Check the manufacturer information for your particular model of SCSI-attached tape drive (and/or use Fish's "ftape" Windows utility) to determine whether or not this option is needed for your particular drive.

--blkid-32 This option indicates that your SCSI-attached tape drive only supports 32-bit block-ids (as used by 3590 drives) and not the 22-bit format used by 3480/3490 drives. You should only specify this option if you intend to define the drive as a model 3480 or 3490 device and then only if your actual SCSI drive uses 32-bit block-ids. If you define your Hercules tape drive as a model 3590 device however this option is not required as 3590 drives are already presumed to use 32-bit block-ids.

Specifying this option on a 3480/3490 device statement will cause Hercules device emulation logic to automatically translate the actual SCSI tape drive's 32-bit block-id into 22-bit format before returning it back to the guest operating system (since that is the format it expects it to be in for a model 3480/3490 drive), and to translate the guest's 22-bit format block-id into 32-bit format before sending it to the actual SCSI hardware (since that is the format that the actual hardware requires it to be in).

--blkid-22 This is the opposite of the above --blkid-32 option.

6.8.1.4 Examples

Example 1:

Define a 3420 tape device on device address 0580. The tape device is a SCSI-attached tape drive which only supports 32-bit block-ids.

```
0580 3420 /dev/nst0 --blkid-32
```

Example 2:

Define a 3490 tape device on device address 0581. The tape device is a SCSI-attached tape drive which supports 22-bit block-ids. Ignore any ‘Erase Gap’ commands.

```
0581 3490 \\.\Tape0 --no-erg --blkid-22
```

6.8.2 *Optical Media Attach (OMA) virtual files*

6.8.2.1 Function

OMA device statements are used to define Optical Media Attach (OMA) virtual files to the Hercules configuration. OMA virtual files are read-only files which normally reside on CDROM. OMA virtual tapes consist of one CDROM file corresponding to each physical file of the emulated tape. An ASCII text file called the "Tape Descriptor File" (TDF) specifies the names of the files which make up the virtual tape. The argument specifies the name of the tape descriptor file.

Each file on the virtual tape can be in one of the following three formats:

TEXT TEXT files consist of variable length ASCII records delimited by carriage return line feed (CRLF) sequences at the end of each record. Each record is translated to EBCDIC and presented to the program as one physical tape block.

FIXED nnnnn FIXED files consist of fixed length EBCDIC blocks of the specified length (nnnnn).

HEADERS HEADERS files consist of variable length EBCDIC blocks. Each block is preceded by a 12-byte header.

If you have any IBM manuals in BookManager format on CDROM, you can see some examples of TDF files in the "\TAPES" directory on the CDROM.

6.8.2.2 Syntax

Descriptive

devaddr devtype tdf

Diagram

►►── *devaddr* ── *devtype* ── *tdf* ──────────────────────────────►◄

6.8.2.3 Parameter

devaddr This is the device address.

devtype This is the device type Valid device types are 3410, 3420, 3422, 3430, 3480, 3490, 8809 and 9347.

tdf The filename (and path) of the tape descriptor file.

6.8.2.4 Examples

Example 1:

Define a 3480 tape device on device address 0582. The tape device is an Optical Media Attach (OMA) virtual file (CD-ROM). The filename of the Tape Descriptor File (TDF) is "B00105.TDF" located in the current search path.

```
0582 3480 /CDROM/TAPES/UAA186.TDF
```

Example 2:

Define a 3490 tape device on device address 0581. The tape device is an Optical Media Attach (OMA) virtual file (CD-ROM). The filename of the Tape Descriptor File (TDF) is "UAA186.TDF" located in path "/CDROM/TAPES/".

```
0581 3490 B00105.TDF
```

6.8.3 AWSTAPE virtual files

6.8.3.1 Function

AWSTAPE device statements are used to define AWSTAPE virtual files to the Hercules configuration. AWSTAPEs contain a complete tape in one file. AWSTAPE files consist of variable length EBCDIC blocks. Each block is preceded by a 6-byte header. Filemarks are represented by a 6-byte header with no data. This is the same format as is used by the IBM P/390 systems. The argument specifies the location of the AWSTAPE file.

6.8.3.2 Syntax

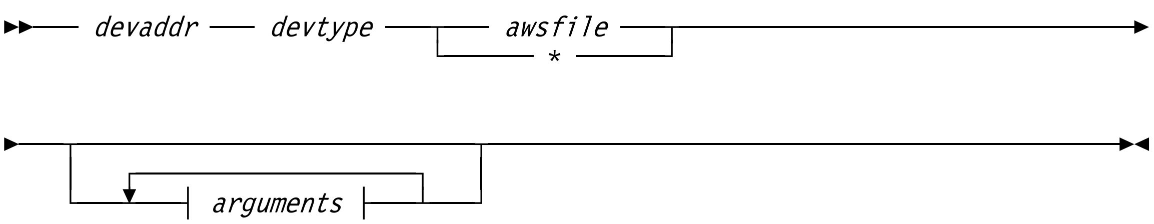
Descriptive

devaddr devtype {*awsfile* | *} [*arguments*]

where arguments can be:

[MAXSIZE={*n*[K | M | G | T] | 0} |
 MAXSIZEK={*n* | 0} |
 MAXSIZEM={*n* | 0}]
[EOTMARGIN=*n*[K | M | G | T]]
[READONLY={0 | 1}]
[RO | NORING | RW | RING]
[DEONIRQ={0 | 1}]
[NOAUTOMOUNT]

Diagram



where arguments can be:

```
├──┬── MAXSIZE= ──┬── 0 ─────────────────────┬──────────────────────────┤
   │              └── n ──┬── K ──┐          │
   │                      ├── M ──┤          │
   │                      ├── G ──┤          │
   │                      └── T ──┘          │
   ├── MAXSIZEK= ─┬── 0 ──┬──────────────────┘
   │              └── n ──┘     │
   └── MAXSIZEM= ─┬── 0 ──┬─────┘
                  └── n ──┘


├── EOTMARGIN= ── n ──┬─────────┬──────────────────────────────────────────┤
                      ├── K ────┤
                      ├── M ────┤
                      ├── G ────┤
                      └── T ────┘

├── READONLY= ──┬── 0 ──┬────────────────────────────────────────────────┤
                └── 1 ──┘

├──┬──── RO ──────┬──────────────────────────────────────────────────────┤
   ├── NORING ────┤
   ├──── RW ──────┤
   └─── RING ─────┘

├── DEONIRQ= ──┬── 0 ──┬─────────────────────────────────────────────────┤
               └── 1 ──┘

├── NOAUTOMOUNT ─────────────────────────────────────────────────────────┤
```

6.8.3.3 Parameter

devaddr This is the device address.

devtype This is the device type. Valid device types are 3410, 3420, 3422, 3430, 3480, 3490, 8809 and 9347.

awsfile The filename (and path) of the AWSTAPE file or "*". An asterisk defines an empty tape station. The tape must be manually be loaded on request.

If the filename starts with the "@" character (at sign), the file then describes a list of tape emulation files to be loaded in succession. The syntax of each line in this file is identical to the information that can be coded directly after the device type in the overall configuration file. Any emulation file name parameter specified in this file may be substituted by the character "*", in which case it specifies a set of options to be applied to all additional emulation files specified in the list.

This function emulates an Automatic Cartridge Feeder (ACF. The ACF is a feature on Cartridge type tape drives (3480, 3490, etc..) that automatically loads a new tape when a tape is removed from the drive. There is no real control over this device by the host as it just keeps on feeding tapes one after the other. Although the

ACF feature is unique to cartridge type systems, the emulation accepts the use of the same technique for emulated 1/2 inch tapes reel drives as well.

Parameters are appended in succession. In all cases, if the same parameter is coded more than once, the last instance takes precedence. Therefore it is possible to specify a set of parameters in the base configuration file, another on an “*” line and another for each individual line in the list of files. Parameters are then appended in that order. A SCSI tape device should not be given in a file list.

The remaining parameters are described in section 6.8.6 (“Common parameters for AWS, HET and FakeTape virtual files”).

6.8.3.4 Examples

Example 1:

Define a 3490 tape device on device address 0580. The tape device is an AWSFILE virtual file named "R0673A.AWS" located in path "/S390/TAPES/".

```
0580 3490 /S390/TAPES/R0673A.AWS
```

Example 2:

Define a 3490 tape device on device address 0580. The tape device is an AWSFILE virtual file named "R0674A.AWS" located in path "/S390/TAPES/". Mount the tape readonly and disable the support for guest-initiated automatic tape volume mounting for this device.

```
0580 3490 /S390/TAPES/R0674A.AWS READONLY=1 NOAUTOMOUNT
```

6.8.4 *HET virtual files*

6.8.4.1 HET virtual files

HET device statements are used to define HET virtual files to the Hercules configuration. These contain a complete tape in one file and have the same structure as the AWSTAPE format with the added ability to have compressed data. The first argument specifies the location of the HET file. The filename must end with ".HET" to be recognized by Hercules as a HET file (e.g. "023178.HET"). There are several additional arguments that control various HET settings.

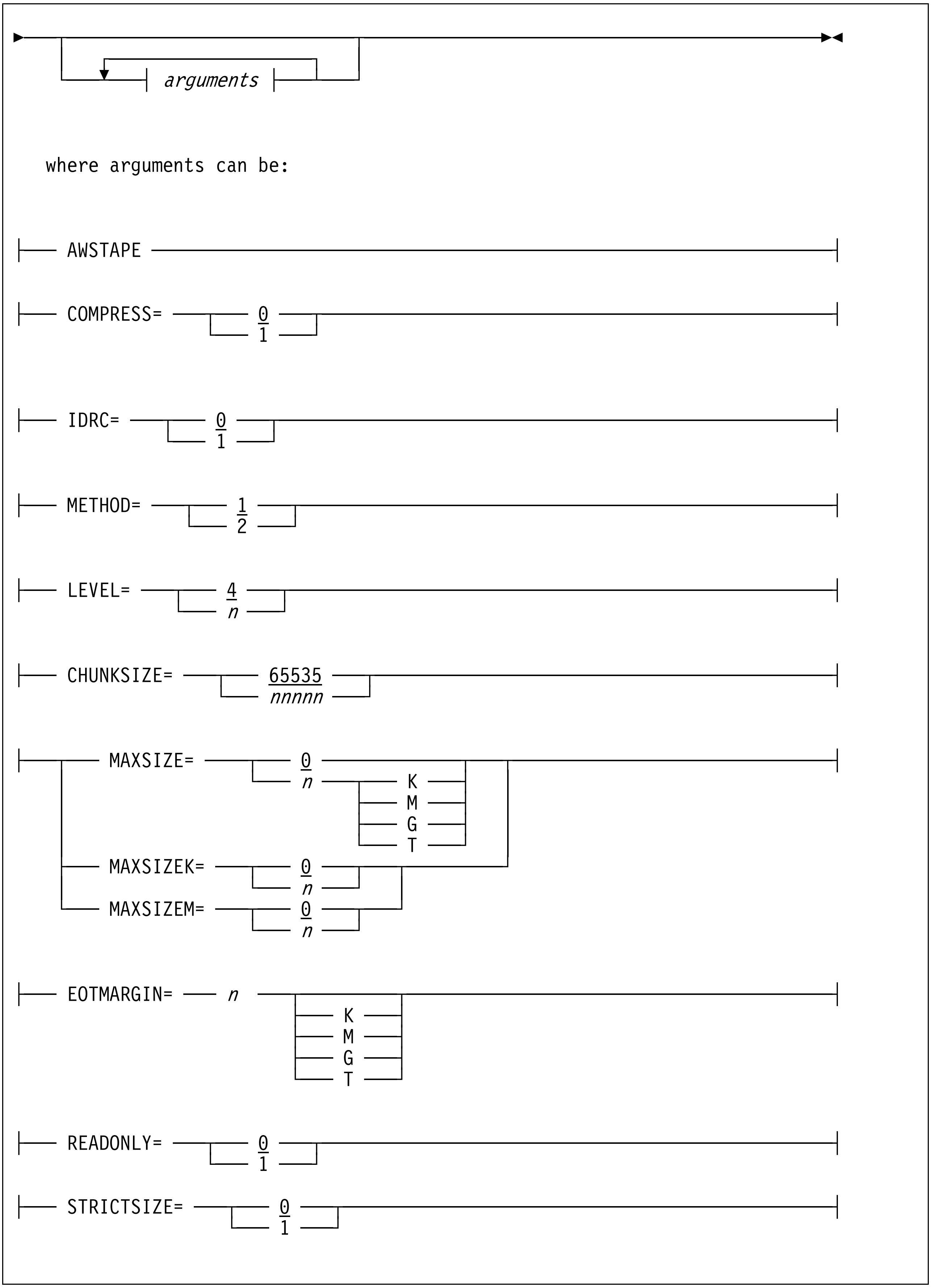
6.8.4.2 Syntax

Descriptive

```
devaddr devtype {hetfile | *} [arguments]

   where arguments can be:

[AWSTAPE]
[COMPRESS={0 | 1}]
[IDRC={0 | 1}]
[METHOD={1 | 2}]
[LEVEL={n | 4}]
[CHUNKSIZE={nnnnn | 65535}]
[MAXSIZE={n[K | M | G | T] | 0} |
 MAXSIZEK={n | 0} |
 MAXSIZEM={n | 0}]
[EOTMARGIN=n[K | M | G | T]]
[READONLY={0 | 1}]
[STRICTSIZE={0 | 1}]
[RO | NORING | RW | RING]
[DEONIRQ={0 | 1}]
[NOAUTOMOUNT]
```

Diagram

```
►►── devaddr ── devtype ──┬── hetfile ──┬──────────────────────────►
                          └───── * ─────┘
```


arguments
where arguments can be:
AWSTAPE
COMPRESS= 0 1
IDRC= 0 1
METHOD= 1 2
LEVEL= 4 n
CHUNKSIZE= 65535 nnnnn
MAXSIZE= 0 n K M G T
MAXSIZEK= 0 n
MAXSIZEM= 0 n
EOTMARGIN= n K M G T
READONLY= 0 1
STRICTSIZE= 0 1

```
├──┬──── RO ────┬──────────────────────────────────────────────────────┤
   ├── NORING ──┤
   ├──── RW ────┤
   └─── RING ───┘


├── DEONIRQ= ──┬── 0 ──┬──────────────────────────────────────────────┤
               └── 1 ──┘


├── NOAUTOMOUNT ──────────────────────────────────────────────────────┤
```

6.8.4.3 Parameter

devaddr This is the device address.

devtype This is the device type. Valid device types are 3410, 3420, 3422, 3430, 3480, 3490, 8809 and 9347.

hetfile The filename (and path) of the HET file or "*". An asterisk defines an empty tape station. The tape must be manually loaded on request.

If the filename starts with the "@" character (at sign), the file then describes a list of tape emulation files to be loaded in succession. The syntax of each line in this file is identical to the information that can be coded directly after the device type in the overall configuration file. Any emulation file name parameter specified in this file may be substituted by the character "*", in which case it specifies a set of options to be applied to all additional emulation files specified in the list.

This function emulates an Automatic Cartridge Feeder (ACF). The ACF is a feature on cartridge type tape drives (3480, 3490, etc.) that automatically loads a new tape when a tape is removed from the drive. There is no real control over this device by the host as it just keeps on feeding tapes one after the other. Although the ACF feature is unique to cartridge type systems, the emulation accepts the use of the same technique for emulated 1/2 inch tapes reel drives as well.

Parameters are appended in succession. In all cases, if the same parameter is coded more than once, the last instance takes precedence. Therefore it is possible to specify a set of parameters in the base configuration file, another on an "*" line and another for each individual line in the list of files. Parameters are then appended in that order. A SCSI tape device should not be given in a file list!

AWSTAPE The AWSTAPE argument causes HET files to be written in AWSTAPE format. This disables the additional feature provided by the HET format.

IDRC= COMPRESS and IDRC control whether compression should be used when writing to HET files. The value of *n* can be "1" to turn compression on (the default), or can be "0" to turn compression off. IDRC is currently a synonym for COMPRESS but may be used in the future to control another emulated tape drive feature. Therefore COMPRESS is the preferred method to turn compression on or off.

COMPRESS= This is the same as IDRC=.

METHOD= The METHOD argument allows you to specify which compression method to use. A value of “1” forces the use of ZLIB compression. A value of “2” forces the use of BZIP2 compression. The default is “1”.

LEVEL= The LEVEL option controls the level of compression. It ranges from “1” (for fastest compression) to “9” (for best compression). The default compression level is “4”.

CHUNKSIZE= The CHUNKSIZE option allows you to create HET files with different chunk sizes. The AWSTAPE (and therefore HET) formats allow each tape block to be logically broken up into smaller chunks. For instance, if the S/3x0 application creates tapes with a block size of 27998, those blocks would be broken down into *nnnnn* sized chunks. Although possible it is not recommended to change the chunk size, as decreasing this may reduce compression performance. The range of *nnnnn* is from 4096 to 65535. 65535 is the default.

STRICTSIZE= Upon reaching the tape file size limit, depending on strictsize, the tape file will or will not be truncated to enforce the maxsize limit. The limit is only enforced during a write type operation. If the file already exists and the program only reads the file, then the file will not be truncated, regardless of the strictsize setting.

This affects any write that starts below the limit, but that would extend beyond the limit. The strictsize parameter only affects compressed HET files. On AWS tapes, the limit is always enforced, but the file is not truncated (the write does not occur, because first AWS tapes are never truncated and second the effects of the write are known in advance).

Regardless of strictsize, any write operation (Write, Write TM) will return a Unit Check with Equipment Check to the program if the file size exceeds the predefined limit. If strictsize is “0”, the write will actually have been performed on the tape file. When strictsize is set to “1”, the file will be truncated on the preceding tape block boundary.

Care must be taken that regardless of the strictsize setting, the tape may become unusable for the guest program should such an event occur (absence of a Tape Mark for example). This option has no effect if maxsize is 0. The default is “0” (do not truncate).

The remaining parameters are described in section 6.8.6 (“Common parameters for AWS, HET and FakeTape virtual files”).

6.8.4.4 Examples

Example 1:

Define a 3490 tape device on device address 0580. The tape device is a HET virtual file with name "R0674A.HET" located in path "/S390/TAPES/". BZIP2 compression has to be turned on at a level of 9 (best compression).

```
0580 3490 /S390/TAPES/R0674A.HET COMPRESS=1 METHOD=2 LEVEL=9
```

Example 2:

Define a 3490 tape device on device address 0580. The tape device is a HET virtual file with name "R0675A.HET" located in path "/S390/TAPES/". Mount the tape as read/write.

```
0580 3490 /S390/TAPES/R0675A.HET RING
```

6.8.5 Fake Tape virtual files

6.8.5.1 Function

Fake Tape virtual files contain a complete tape in one file. FakeTape files consist of variable length EBCDIC blocks. Each block is preceded by a 12-byte ASCII-hex-character header. Filemarks are represented by a 12-byte character header with no data. The FakeTape format is used by the Flex-ES system from Fundamental Software Inc (FSI). The argument specifies the location of the FakeTape file (for example “ickdsf.fkt”). "FLEX-ES" and "FakeTape" are trademarks of Fundamental Software, Inc.

6.8.5.2 Syntax

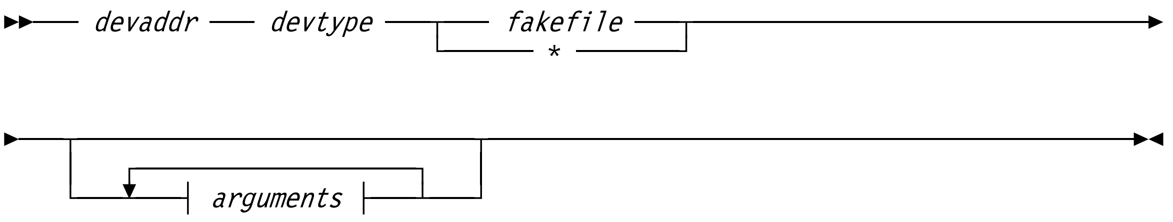
Descriptive

devaddr devtype {*fakefile* | *} [*arguments*]

where arguments can be:

```
[MAXSIZE={n[K | M | G | T] | 0} |
 MAXSIZEK={n | 0} |
 MAXSIZEM={n | 0}]
[EOTMARGIN=n[K | M | G | T]]
[READONLY={0 | 1}]
[RO | NORING | RW | RING]
[DEONIRQ={0 | 1}]
[NOAUTOMOUNT]
```

Diagram



where arguments can be:

```
├────┬──── MAXSIZE= ─────┬─── 0 ──────────────────┬──────────────────────────────┤
     │                   └─── n ─────┬─── K ───┐  │
     │                               ├─── M ───┤  │
     │                               ├─── G ───┤  │
     │                               └─── T ───┘  │
     ├──── MAXSIZEK= ────┬─── 0 ─────┬───────────┘
     │                   └─── n ─────┘   │
     └──── MAXSIZEM= ────┬─── 0 ─────────┘
                         └─── n ─────┘

├─── EOTMARGIN= ─── n ───┬──────────────────────────────────────────────────────┤
                         ├─── K ───┤
                         ├─── M ───┤
                         ├─── G ───┤
                         └─── T ───┘

├─── READONLY= ───┬─── 0 ───┬───────────────────────────────────────────────────┤
                  └─── 1 ───┘

├────┬──── RO ──────┬───────────────────────────────────────────────────────────┤
     ├─── NORING ───┤
     ├──── RW ──────┤
     └─── RING ─────┘

├─── DEONIRQ= ───┬─── 0 ───┬────────────────────────────────────────────────────┤
                 └─── 1 ───┘

├─── NOAUTOMOUNT ───────────────────────────────────────────────────────────────┤
```

6.8.5.3 Parameter

devaddr This is the device address.

devtype This is the device type. Valid device types are 3410, 3420, 3422, 3430, 3480, 3490, 8809 and 9347.

fakefile The filename (and path) of the Fake Tape virtual file or "*". An asterisk defines an empty tape station. The tape must be manually be loaded on request.

If the filename starts with the "@" character (at sign), the file then describes a list of tape emulation files to be loaded in succession. The syntax of each line in this file is identical to the information that can be coded directly after the device type in the overall configuration file. Any emulation file name parameter specified in this file may be substituted by the character "*", in which case it specifies a set of options to be applied to all additional emulation files specified in the list.

This function emulates an Automatic Cartridge Feeder (ACF. The ACF is a feature on Cartridge type tape drives (3480, 3490, etc..) that automatically loads a new tape when a tape is removed from the drive. There is no real control over this device by the host as it just keeps on feeding tapes one after the other. Although the ACF feature is unique to cartridge type systems, the emulation accepts the use of the same technique for emulated 1/2 inch tapes reel drives as well.

Parameters are appended in succession. In all cases, if the same parameter is coded more than once, the last instance takes precedence. Therefore it is possible to specify a set of parameters in the base configuration file, another on an "*" line and another for each individual line in the list of files. Parameters are then appended in that order. A SCSI tape device should not be given in a file list.

The remaining parameters are described in section 6.8.6 ("Common parameters for AWS, HET and FakeTape virtual files").

6.8.5.4 Examples

Example 1:

Define a 3490 tape device on device address 0580. The tape device is an Fake Tape virtual file named "R0528A.FKT" located in path "/S390/TAPES/".

```
0580 3490 /S390/TAPES/R0528A.FKT
```

Example 2:

Define a 3490 tape device on device address 0580. The tape device is an Fake Tape virtual file named "R0528A.FKT" located in path "/S390/TAPES/". Mount the tape as read-only and present a device end if intervention is required during tape motion.

```
0580 3490 /S390/TAPES/R0528A.FKT NORING DEONIRQ=1
```

6.8.6 Common parameters for AWS, HET and Fake Tape virtual files

MAXSIZE= This specifies the maximum number of bytes for the emulated file. The value is either *n* bytes or *nx* where x specifies the multiplier K, M, G, or T (see "Table 10" at the end of this section for details). Specifying zero for this parameter means "unlimited" (there is no limit on the file size). MAXSIZE defaults to "0".

MAXSIZEK= This is the same as MAXSIZE=, but specified in Kilobytes.

MAXSIZEM= This is the same as MAXSIZE=, but specified in Megabytes.

EOTMARGIN= Specifies the number of bytes remaining (before reaching *MAXSIZEx*) at which point the tape device will signal the presence of the "End-of-Tape" marker (reflector), thus allowing the program to switch to the next tape. The value is either *n* bytes or *nx* where x specifies the multiplier K, M, G, or T (see "Table 10" at the end of this section for details).

READONLY= Specifies whether the tape is mounted read-only (without a write ring or with the cartridge protect switch set to "write protect"). A parameter of 1 means read-only; a parameter of 0 means read-write. If READONLY=1, RO or NORING is not specified; READONLY=0 is the default. Note that READONLY=0 does not override the host system file permission settings for the underlying AWS or HET file. If the AWS or HET file is marked read-only, the tape will be mounted read-only despite specification of READONLY=0.

RO Specifies that the tape is mounted read-only (without a write ring or with the cartridge protect switch set to "write protect"). RO and NORING are equivalent to READONLY=1.

NORING This is the same as RO.

RW Specifies that the tape should be mounted read-write, if possible. RW and RING are equivalent to READONLY=0. This is the default if READONLY=1, RO or NORING is not specified. Note that RW and RING do not override the host system file permission settings for the underlying AWS or HET file. If the AWS or HET file is marked read-only, the tape will be mounted read-only despite specification of RW or RING.

RING This is the same as RW.

DEONIRQ= Specifies whether a device end is presented if intervention is required during tape motion. A parameter of 1 selects this option; a parameter of 0 turns it off.

NOAUTOMOUNT Indicates that support for guest-initiated automatic tape volume mounting is to always be disabled for this tape device. Automatic guest tape-mount support is automatically globally enabled for all virtual (non-SCSI) tape devices by default whenever an allowable automount directory is defined via the AUTOMOUNT system parameter or the automount console command. The NOAUTOMOUNT option allows you to specifically disable such support for a given device.

The automount feature enables software running in guest operating systems to automatically mount, unmount and/or query for themselves the host "virtual tape volume" filename mounted on a tape drive, via the use of special CCW opcodes (0x4B Set Diagnose and 0xE4 Sense ID) without any intervention on the part of the Hercules operator. An example of such a program for DOS/VSE called TMOUNT is provided in the util subdirectory of the distributed source code.

This is a sticky option. When specified, automount support for the device remains disabled until the option is specifically removed via a devinit command without the option specified. This means if NOAUTOMOUNT is enabled for a device while global automount functionality is currently disabled (because no AUTOMOUNT statement was specified at Hercules startup), then automount functionality remains disabled for the device even should global automount functionality be later manually enabled via an automount con-sole command.

When the 0x4B Set Diagnose CCW is used to auto-mount a virtual tape volume onto a given tape drive, an absolute (fully-qualified) pathname should normally always be specified, but need not be if a path relative to the currently defined "default allowable" automount directory is used instead.

The default allowable automount directory is always the first "allowable" directory that was defined, or else the current directory if no allowable directories were specifically defined. There is always a default allowable directory whenever any allowable or unallowable automount directories are defined.

Fully-resolved, absolute-full-path filenames are defined as being those which, for Windows, have a ':' (colon) in the second position or, for other host operating systems (e.g. Linux), have a '/' (slash) in the first position. Pathes which start with a '.' (period) are considered relative paths and will always be appended to the currently defined default allowable automount directory, before being resolved into fully-qualified paths by the host system (i.e. only fully-resolved absolute pathnames are used in the performance of the actual automatic tape volume mount).

For example, if more than one allowable automount directory is defined and the volume wishing to be mounted happens to reside in the second one, then a fully-qualified absolute pathname should of course be specified (or else one that is relative to the default directory which happens to resolve to the desired file).

All attempts to automount host files in any defined "disallowable" directory (or any subdirectory thereof [or otherwise *not* within any defined "allowable" directory or subdirectory]) will be rejected. An error message is always issued in such cases, just as one is whenever a successful mount or unmount is performed.

A sample guest automount program called TMOUNT for the DOS/VSE operating system is provided in the "util" subdirectory of the distributed source code.

6.8.6.1 Multipliers for ‘MAXSIZE=’ and ‘EOTMARGIN=’ parameters

Unit	Multiplier	Name (Symbol)	IEC Name (IEC Symbol)	Restrictions
K	2**10	Kilobyte (kB)	Kibibyte (KiB)	
M	2**20	Megabyte (MB)	Mebibyte (MiB)	
G	2**30	Gigabyte (GB)	Gibibyte (GiB)	
T	2**40	Terabyte (TB)	Tebibyte (TiB)	Not on 32-bit machines

Table 10: Multipliers for ‘MAXSIZE=’ and ‘EOTMARGIN=’ parameters

6.8.7 Basic ACF (Automatic Cartridge Feeder) Support

6.8.7.1 Function

The ACF (Automatic Cartridge Feeder) is a feature on cartridge type tape drives (3480, 3490, etc.) that automatically loads a new tape when a tape is removed (ejected) from the drive. There is no real control over this functionality by the host as the device just keeps on feeding tapes one after the other.

Although the ACF features is unique to cartridge type tape systems the emulation accepts to use the same technique for emulated ½ inch tapes reel drives as well. ACF is supported through a file that contains a list of files (emulated tapes or cartridges) that will be loaded one after the other.

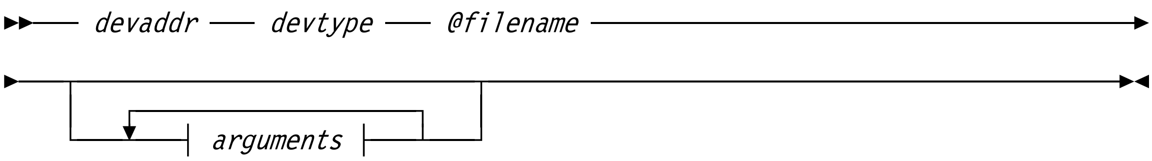
To manually reset the ACF to the top of the stack the DEVINIT panel command can be used to "reload" the ACF feature.

6.8.7.2 Syntax

Descriptive

devaddr devtype @filename [*arguments*]

Diagram



6.8.7.3 Parameter

devaddr This is the device address.

devtype This is the device type. Valid device types are 3410, 3420, 3422, 3430, 3480, 3490, 8809 and 9347.

@filename The filename (without the prefixing '@') contains a list of files that will be loaded one after the other. The filenames contained in the file list cannot describe another ACF file nor an SCSI tape handle (/dev/st*x*). However the files may be standard AWS, HET, Fake Tape or OMA files.

arguments These are the same arguments that apply to the AWS, HET, Fake Tape or OMA files, as described above.

6.8.7.4 Syntax ACF description file

```
# comment line
* arguments
filename [arguments]
```

6.8.7.5 Parameter

The hash symbol starts a comment line.

***** Any options following the asterisk are applied to each file of the list, followed by the options specified on the device configuration entry, followed by the options specified on each individual entry of the file list. Care must be taken that all '*' lines are processed at once.

filename The filename (and path) of the AWS, HET, Fake Tape or OMA tape file.

arguments These are the same arguments that apply to the AWS, HET, Fake Tape or OMA files, as described above. The arguments are processed in the order in which they appear. Any conflicting argument overrides the previous one.

6.8.7.6 Examples

Example 1:

Define a 3490 Automatic Cartridge Feeder (ACF) on device address 0589. ZLIB compression is turned on. The file "newstack" describes a list of tape emulation files to be loaded in succession.

a) Hercules configuration file:

```
0589 3490 @NEWSTACK COMPRESS=1
```

b) "NEWSTACK" file:

```
* MAXSIZEM=16 EOTMARGIN=131072
tape01.aws COMPRESS=0
tape02.het MAXSIZEM=32 EOTMARGIN=65536
tape03.het MAXSIZE=0
```

The above example is equivalent to issuing the following (one at the start and one after each tape unload):

a) in the configuration file:

```
180 3420 tape01.aws maxsizeM=16 eotmargin=131072 compress=0
```

b) via devinit panel command:

```
devinit 180 tape02.het compress=1 maxsizeM=32 eotmargin=65536
devinit 180 tape03.het eotmargin=131072 compress=1 maxsize=0
```

Example 2:

Resolution of the arguments using ‘*’ lines.

The following ACF file

```
* compress=0
tape01.aws
* compress=1
tape02.aws
```

is equivalent to this one

```
* compress=1
tape01.aws
tape02.aws
```

6.9 Channel-to-Channel Adapters

6.9.1 Introduction

The Channel-to-Channel Adapter device statements define CTC adapters to the Hercules configuration. All of the communications emulation implemented within Hercules use a CTCA (Channel-to-Channel Adapter) type device. Depending on the type, the CTCA device will provide either a point-to-point or a virtual network adapter interface to the driving system's TCP/IP stack or in the case of CTCT, a true CTCA connection to another instance of Hercules via a TCP/IP connection.

All current emulations, with the exception of VMNET and CTCT use the Universal TUN/TAP driver on Linux and TunTap32 (WinPCap) on the Windows platforms which creates a network interface on the driving system which allow Hercules to present frames to, and receive frames from the TCP/IP stack. This network interface is configured on Linux platforms by the 'hercifc' program which is invoked by Hercules after the TUN/TAP device is opened.

The following are the emulation types currently supported:

- CTCI (Channel-to-Channel link to TCP/IP stack)
- CTCT (Channel-to-Channel emulation via TCP Connection)
- LCS (LAN Channel Station emulation)
- PTP (MPCPTP/PCPTP6 Channel-to-Channel link)
- VMNET (Channel-to-Channel link via SLIP/VMNET)

6.9.2 CTCI (Channel-to-Channel link to TCP/IP stack)

6.9.2.1 Function

The CTCI is a point-to-point connection with the TCP/IP stack of the driving system on which Hercules is running.

6.9.2.2 Syntax

Descriptive

```
devaddr CTCI [{-n | --dev} name]
             [{-s | --netmask} mask]
             [{-m | --macaddr} mac | 00:00:5E:nn:nn:nn}]
             [{-t | --mtu} mtu | 1500]                    *Linux only
             [{-i | --ibuff} {ibuff | 64}]                *Windows only
             [{-k | --kbuff} {kbuff | 1024}]              *Windows only
             [-d | --debug]
             guestip hostip
```

Diagram

```
►►── devaddr ── CTCI ──┬───────────────────────────┬──────────────────►
                       ├────── -n ─────┬─── name ──┘
                       └──── --dev ────┘

►──┬──────────────────────────────┬────────────────────────────────────►
   ├────── -s ──────┬─── mask ────┘
   └─── --netmask ──┘

►──┬───────────────────────────────────────────────┬───────────────────►
   ├──── -m ──────┬─┬────────── mac ───────────────┤
   └──── --mac ───┘ └── 00:00:5E:nn:nn:nn ─────────┘

►──┬─────────────────────────┬─────────────────────────────────────────►
   ├──── -t ─────┬── mtu ────┘
   └──── --mtu ──┘

►──┬─────────────────────────────────────┬─────────────────────────────►
   ├────── -i ──────┬──┬──── ibuff ──────┤
   └──── --ibuff ───┘  └──── 64 ─────────┘

►──┬─────────────────────────────────────┬─────────────────────────────►
   ├────── -k ──────┬──┬──── kbuff ──────┤
   └──── --kbuff ───┘  └──── 1024 ───────┘

►──┬─────────────────┬── guestip ── hostip ───────────────────────────►◄
   ├────── -d ───────┤
   └──── --debug ────┘
```

6.9.2.3 Parameter

devaddr This is the device address.

CTCI This specifies the CTCI device/protocol type.

-n *name* For Linux: Specifies the name of the tunnel device to use. The default name is "/dev/net/tun", which is correct for version 2.4 and above of the Linux kernel.

For Windows: Identifies the host network adapter. If your network adapter does not have a permanent (static) IP address assigned to it (e.g. you use DHCP and have a dynamic IP assigned) then instead of specifying an IP address as the second parameter in the device statement, you must specify the MAC address of the adapter. In this case the second argument (hostip) must be coded as "0.0.0.0". This is simply a placeholder because two IP addresses are expected to satisfy the device definition syntax.

--dev *name* This is the same as "-n name".

-s *mask* This is the netmask to be configured on the link. Note: Since this is a point-to-point link netmask is meaningless from the perspective of the actual network device.

--netmask *mask* This is the same as "-s mask".

--m *mac*	This is the optional hardware address of the interface in the format of either “xx:xx:xx:xx:xx:xx” or “xx-xx-xx-xx-xx-xx”. The default is 00:00:5E:nn:nn:nn where the “*nn:nn:nn”* part is constructed from the last three octets of the specified guestip.
--macaddr *mac*	This is the same as “-m mac”.
-t *mtu*	Linux only: Specifies the maximum transmission unit size (default 1500 bytes).
--mtu *mtu*	This is the same as “-t mtu”.
-i *ibuff*	Windows only: This specifies the TunTap32 I/O buffer size in KB. *nnnn* hast to be between 16 and 1024 (1 MB). The default value is 64 KB.
--ibuff *ibuff*	This is the same as “-i ibuff”.
-k *kbuff*	Windows only: This specifies the WinPcap device driver capture buffer size in KB. *nnnn* hast to be between 64 and 16384 (16 MB). The default value is 1024 (1 MB).
--kbuff *kbuff*	This is the same as “-k kbuff”.
-d	Specifies that debugging output has to be produced on the Hercules control panel. This should normally be left unspecified.
--debug	This is the same as “-d”.
guestip	Specifies the IP address of the guest operating system running under Hercules.
hostip	Identifies the host network adapter to use. If the host system is configured with DHCP this should instead be the MAC address of the ethernet adapter you wish to have Hercules use to connect to the outside world. If this parameter is specified using the *-m* or *--dev* option, then the value here must be specified as "0.0.0.0" to satisfy the statement syntax.

6.9.2.4 Examples

Example 1:

Define 3088 CTC adapters on device addresses 0E20 and 0E21. The device/protocol type is CTCI (Channel-to-Channel link to Linux TCP/IP stack). The IP address of the guest operating system running under Hercules is 192.168.1.99, the host IP address (the network adapter) to use is 192.168.1.100.

```
0E20.2 3088 CTCI 192.168.1.99 192.168.1.100

   or

0E20 3088 CTCI 192.168.1.99 192.168.1.100
0E21 3088 CTCI 192.168.1.99 192.168.1.100
```

Example 2:

Define 3088 CTC adapters on device addresses 0E20 and 0E21. The device/protocol type is CTCI (Channel-to-Channel link to Win32 TCP/IP stack). The WinPcap device driver capture buffer has to be set to 2048 KB, the TunTap32 I/O buffer size has to be set to 128 KB. The IP address of the guest operating system running under Hercules is 192.168.1.99 and the MAC address of the network adapter to use is 00-80-B3-E1-DF-69.

```
0E20.2 3088 CTCI -n 00-80-B3-E1-DF-69 -k 2048 -i 128 192.168.1.99 0.0.0.0
```

```
   or
```

```
0E20 3088 CTCI -n 00-80-B3-E1-DF-69 -k 2048 -i 128 192.168.1.99 0.0.0.0
0E21 3088 CTCI -n 00-80-B3-E1-DF-69 -k 2048 -i 128 192.168.1.99 0.0.0.0
```

Example 3:

Define 3088 CTC adapters on device addresses 0E20 and 0E21. The device/protocol type is CTCI (Channel-to-Channel link to Linux TCP/IP stack). The IP address of the guest operating system running under Hercules is 192.168.1.99, the host IP address (the network adapter) to use is 192.168.1.100.

The maximum transmission size (MTU) should be 1492 bytes. The WinPcap device driver buffer size has to be set to 1024 KB and the TunTap32 I/O buffer size to 256 kB. Additionally debugging output has to be shown on the Hercules console.

```
0E20.2 3088 CTCI -k 1024 -i 256 –t 1492 –d 192.168.1.99 192.168.1.100
```

```
   or
```

```
0E20 3088 CTCI -k 1024 -i 256 –t 1492 –d 192.168.1.99 192.168.1.100
0E21 3088 CTCI -k 1024 -i 256 –t 1492 –d 192.168.1.99 192.168.1.100
```

6.9.3 CTCT (Channel-to-Channel emulation via TCP Connection)

6.9.3.1 Function

This is an emulated Channel-to-Channel adapter to another Hercules system. CTCT only supports IP traffic and cannot be used to transport NJE, SNA, PVM, etc. type payloads. This may change in the future.

6.9.3.2 Syntax

Descriptive

devaddr CTCT *lport rhost rport bufsize*

Diagram

►►── *devaddr* ── CTCT ── *lport* ── *rhost* ── *rport* ── *bufsize* ──────────►◄

6.9.3.3 Parameter

devaddr	This is the device address.
CTCT	This specifies the CTCT device/protocol type.
lport	Specifies the local TCP port. This is the TCP port that Hercules will listen on for this CTC adapter.
rhost	Specifies the remote host. This is the name or IP address of the remote host system that Hercules is running on, not the name or IP address of the OS running under this instance of Hercules.
rport	Specifies the remote TCP port. The rport parameter on this system must match the lport parameter on the remote system and vice versa.
bufsize	Specifies the buffer size for the link. If this link is used for IP traffic this parameter should be more than the MTU of the interface definition in the OS TCPIP stack.

6.9.4 Examples

Example 1:

Define 3088 CTC adapters on device addresses 0400 and 0401. The device/protocol type is CTCT (Channel-to-Channel emulation via TCP connection). The local TCP ports on which Hercules will listen are 30880 and 30881. The IP address of the remote host is 192.168.100.2, the remote ports are also 30880 and 30881. The buffer size for the link is 2048 KB.

```
0400.2 3088 CTCT 30880 192.168.100.2 30880 2048

   or

0400 3088 CTCT 30880 192.168.100.2 30880 2048
0401 3088 CTCT 30881 192.168.100.2 30881 2048
```

6.9.5 CTCE (Enhanced Channel-to-Channel emulation via TCP Connection)

6.9.5.1 Function

The CTCE device type will emulate a real 3088 Channel-to-Channel Adapter also for non-IP traffic, enhancing the CTCT capabilities. CTCE connections are also based on TCP/IP between two (or more) Hercules instances, and requires an even-odd pair of port numbers per device side.

Only the even port numbers are to be configured; the odd numbers are just derived by adding 1 to the configured even port numbers. The socket connection pairs cross-connect, the arrows are showing the the send -> receive direction:

```
InstanceA-lport-even -> InstanceB-rport-odd
InstanceB-lport-odd  <- InstanceA-rport-even
```

CTCE connected Hercules instances can be hosted on either Unix or Windows platforms, both sides do not need to be the same.

6.9.5.2 Syntax

<u>**Descriptive**</u>

devaddr CTCE *lport raddress rport* [*mtu* [*sml*]]

<u>**Diagram**</u>

```
►►── devaddr ── CTCE ── lport ── raddress ── rport ──────────►

►──┬─────────────────────┬──────────────────────────────────►◄
   └── mtu ──┬──────────┤
             └── sml ───┘
```

devaddr	This is the device address.
CTCE	This specifies the device/protocol type.
lport	Specifies the even TCP/IP port on the local system.
raddress	Specifies the TCP/IP address of the remote system.
rport	Specifies the even TCP/IP port on the remote system.
mtu	Optional MTU buffer size. The default size is 32778.
sml	Optional small minimum for the MTU buffer size. The default size is 8.

6.9.5.3 Examples

Example 1:

Define a 3088 CTC connection with the CTCE protocol between two Hercules instances. Instance A is using address 192.168.1.100, Instance B is using address 192.168.1.200.

Definitions on Hercules Instance A with IP address 192.168.1.100:

```
0E40 CTCT 30880 192.168.1.200 30880
0E41 CTCT 30882 192.168.1.200 30882
```

Definitions on Hercules Instance B with IP address 192.168.1.200:

```
0E40 CTCT 30880 192.168.1.100 30880
0E41 CTCT 30882 192.168.1.100 30882
```

6.9.6 LCS (LAN Channel Station)

6.9.6.1 Function

This statement defines an emulated LAN Channel Station adapter. This emulation mode appears to the operating system running in the Hercules machine as an IBM 8232 LCS device, an IBM 2216 router, an IBM 3172 running ICP (Interconnect Communications Program), the LCS3172 driver of an IBM P/390, or an IBM Open Systems Adapter (OSA).

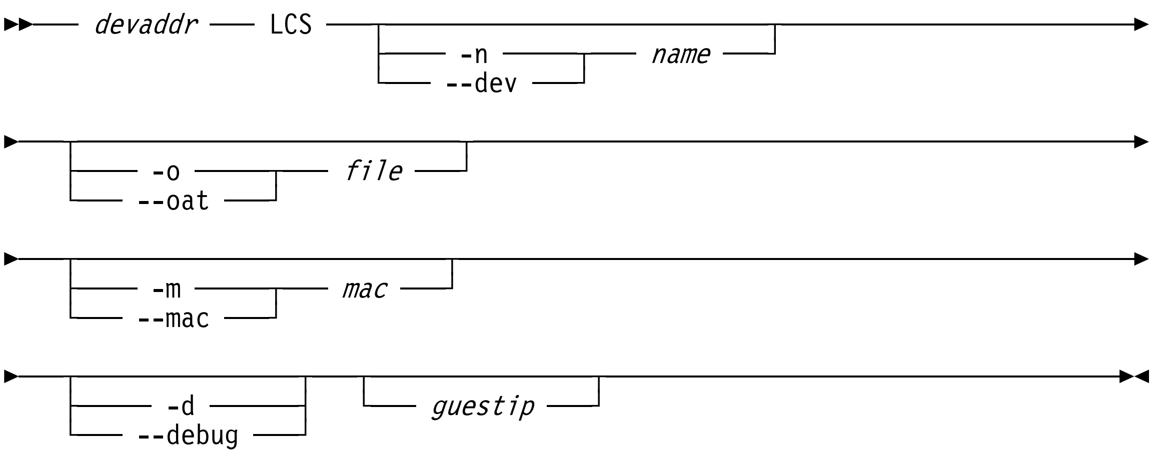
Rather than a point-to-point link, this emulation creates a virtual ethernet adapter through which the guest operating system running in the Hercules machine can communicate. As such, this mode is not limited to TCP/IP traffic, but in fact will handle any ethernet frame.

6.9.6.2 Syntax

Descriptive

devaddr `LCS [{-n | --dev}` *name*`]`
`[{-o | --oat}` *file*`]`
`[{-m | --mac}` *mac*`]`
`[-d | --debug]`
`[`*guestip*`]`

Diagram



6.9.6.3 Parameter

devaddr This is the address pair of the LCS device. This pair must be an even-odd address. This can be specified as two separate LCS statements with a single device address (“0E20 LCS…” and “0E21 LCS …”) or as a single statement with a device address count (“0E20.2 LCS …”).

LCS This specifies the LAN Channel Station (LCS) device/protocol type.

-n *name* Identifies the host network adapter to use:

On Linux systems this is name of the TUN/TAP special character device, normally /dev/net/tun.

On Windows this is either the IP or MAC address of the host systems network card. TunTap32 will automatically select the first network adapter it finds if the option is omitted, this may be not desirable for some users.

--dev *name* This is the same as "-n name".

-o *file* *file* specifies the filename of the OSA Address Table (OAT). If this option is specified the optional *--mac* and *guestip* entries are ignored in preference to statements in the OAT file. For a description of the OAT file see next section.

--oat *file* This is the same as "-o file".

-m *mac* *mac* is the optional hardware address of the interface coded in the format "xx:xx:xx:xx:xx:xx". If you use the "-o/--oat" option do not specify an address here, it will be ignored.

--mac *mac* This is the same "-m mac".

-d Specifies that debugging output has to be produced on the Hercules control panel. This should normally be left unspecified.

--debug This is the same as "-d".

guestip This is an optional IP address of the Hercules (guest OS) side. This is only used to establish a point-to-point routing table entry on the driving system. If you use the "-o/--oat" option do not specify an address here, it will be ignored.

6.9.6.4 OSA Address Table (OAT) Syntax

The syntax for the OSA Address Table file is as follows:

```
*********************************************************
* Dev   Mode  Port  Entry specific information...
*********************************************************
  0400  IP    00    PRI  172.021.003.032
  0402  IP    00    SEC  172.021.003.033
  0404  IP    00    NO   172.021.003.038
  0406  IP    01    NO   172.021.002.016
  040E  SNA   00


HWADD  00  02:00:FE:DF:00:42
```

```
HWADD  01  02:00:FE:DF:00:43
ROUTE  00  172.021.003.032  255.255.255.224
```

Figure 2: OSA Address Table (OAT) File Syntax

Dev	This is the base device address.
Mode	This is the operation mode: IP or SNA. (Note: The SNA operation mode is currently not implemented.
Port	This is the virtual (relative) adapter number.
Entry	This applies only for IP Mode and specifies where a packet with an unknown IP address is forwarded. PRI is the primary default entry, SEC is the entry to use when the primary is not available and NO specifies that this is not a default entry.
n.n.n.n	This specifies the home IP address.

When the operation mode is IP specify only the even (read) device number *dev.* The odd (write) address will be created automatically. Additionally two other statements can be included in the Address Translation file. These are the HWADD and ROUTE statements.

Use the HWADD statement to specify a hardware (MAC) address for a virtual adapter. The first parameter after the HWADD specifies the relative adapter for which the address is applied.

The ROUTE statement is included for convenience. It allows the 'hercifc' program to create a network route for this specific virtual adapter. Please note that it is not necessary to include point-to-point routes for each IP address in the table. This is done automatically by the emulation module.

The read / write devices can be swapped by coding the odd address of the even-odd pair in the OAT. Up to 4 virtual (relative) adapters (00-03) are currently supported.

If no Address Translation file is specified, the emulation module will create the following:

- An ethernet adapter (port 0) for TCP/IP traffic only.

- Two device addresses (devnum and devnum+1).

6.9.6.5 Examples

Example 1:

Define LCS (LAN Channel Station emulation) adapters on device addresses 0440 and 0441. The name of the TUN/TAP special character device is "/dev/net/tun". The IP address of the Hercules guest OS side is 192.168.200.2.

```
0440.2 LCS -n /dev/net/tun 192.168.200.2
   or
0440 LCS -n /dev/net/tun 192.168.200.2
0441 LCS -n /dev/net/tun 192.168.200.2
```

6.9.7 PTP (MPCPTP/PCPTP6 Channel-to-Channel link)

6.9.7.1 Function

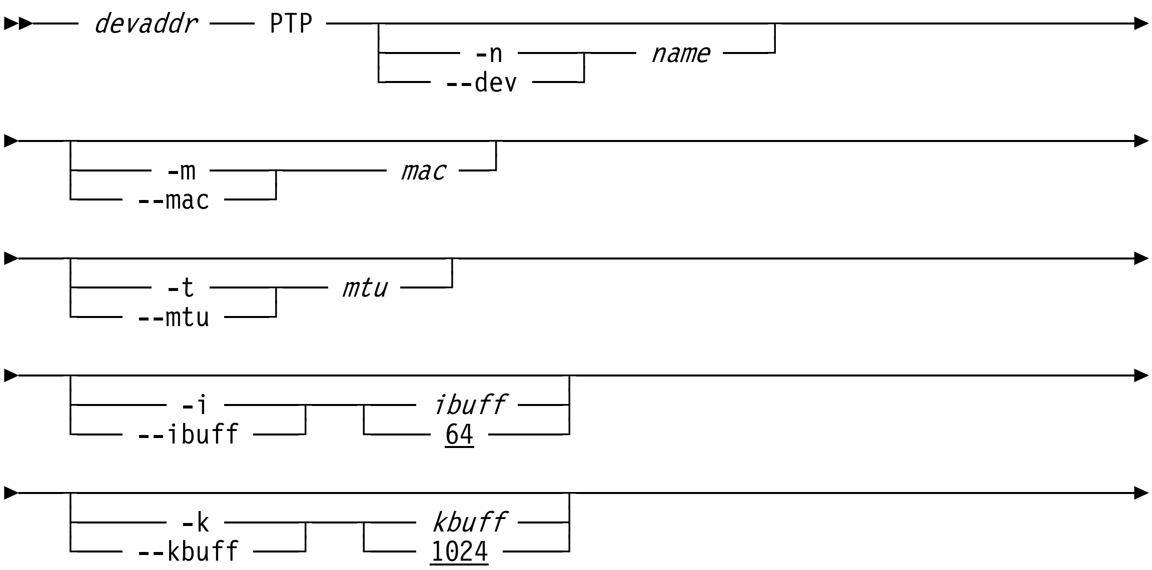
The PTP is a point-to-point link to the driving system's TCP/IP stack. From the point of view of the guest operating system in the Hercules machine it appears to be an MPCPTP and/or MPCPTP6 ESCON CTC link to another guest operating system.

6.9.7.2 Syntax

Descriptive

devaddr PTP [{-n | --dev} *name*]
[{-m | --macaddr} *mac*]
[{-t | --mtu} *mtu* | 1500] *Linux only
[{-i | --ibuff} {*ibuff* | 64}] *Windows only
[{-k | --kbuff} {*kbuff* | 1024}] *Windows only
[{-4 | --inet}]
[{-6 | --inet6}]
[-d | --debug]
guest1 host1
[*guest2 host2*]

Diagram



```
►──┬──── -4 ─────┬──┬──── -6 ──────┬──┬──── -d ──────┬──────────────────►
   └──── --inet ──┘  └──── --inet6 ──┘  └──── --debug ──┘

►── guest1 ── host1 ──┬──────────────────────────┬───────────────────►◄
                      └── guest2 ── host2 ───────┘
```

6.9.7.3 Parameter

devaddr1-devaddr2 These are the device addresses (address pair) of the PTP device.

PTP This specifies the PTP device/protocol type.

-n *name* For Linux: Specifies the name of the tunnel device to use. The default name is "/dev/net/tun", which is correct for version 2.4 and above of the Linux kernel.

For Windows: Identifies the host network adapter. If your network adapter does not have a permanent (static) IP address assigned to it (e.g. you use DHCP and have a dynamic IP assigned) then instead of specifying an IP address you must specify the MAC address of the adapter. TunTap32 will automatically select the first network adapter it finds if this option is omitted, this may not be desirable depending on your configuration.

--dev *name* This is the same as "-n name".

--m *mac* This is the optional hardware address of the interface in the format of either "xx:xx:xx:xx:xx:xx" or "xx-xx-xx-xx-xx-xx".

--macaddr *mac* This is the same as "-m mac".

-t *mtu* Linux only: Specifies the maximum transmission unit size (default 1500 bytes).

--mtu *mtu* This is the same as "-t mtu".

-i *ibuff* Windows only: This specifies the TunTap32 I/O buffer size in KB. *nnnn* hast to be between 16 and 1024 (1 MB). The default value is 64 KB.

--ibuff *ibuff* This is the same as "-i ibuff".

-k *kbuff* Windows only: This specifies the WinPcap device driver capture buffer size in KB. *nnnn* hast to be between 64 and 16384 (16 MB). The default value is 1024 (1 MB).

--kbuff *kbuff* This is the same as "-k kbuff".

-4 Indicates that when a host name is specified for *guest1*, it must resolve to an IPv4 address.

--inet This is the same as "-4".

-6 Indicates that when a host name is specified for *guest1*, it must resolve to an IPv6 address.

--inet6	This is the same as “-6”.
-d	Specifies that debugging output has to be produced on the Hercules control panel. Warning: This will produce a tremendous amount of output to the Hercules console and should therefore normally be left unspecified.
--debug	This is the same as “-d”.
guest1	Specifies the host name or the IP address of the guest operating system running under Hercules.
host1	Identifies the host network adapter to use.
guest2	Specifies the host name or the IP address of the guest operating system running under Hercules.
host2	Identifies the host network adapter to use.

The values for *guest1* and *host1* must both be of the same address family, i.e. both IPv4 or both IPv6.

The values for *guest2* and *host2*, if specified, must both be of the same address family, i.e. both IPv4 or both IPv6, and must not be of the same address family as *guest1* and *host1*.

If a host name is specified for *guest1* and the host name can be resolved to both an IPv4 and an IPv6 address, then use either the “-4/--inet” or the “-6/--inet6” option to specify which address family should be used. If neither the “-4/--inet” or the “-6/--inet6” option is specified then whichever address family the resolver returns first will be used.

If *guest1/host1* or *guest2/host2* are IPv4 addresses or are host names that will resolve to IPv4 addresses, *guest1/guest2* can be followed by the prefix size expressed in CIDR notation, for example 192.168.1.1/24. If the prefix size is specified it can have a value from 0 to 32; if it is not specified then a value of 32 is assumed. The prefix size is used to produce the equivalent subnet mask. For example, a value of 24 produces a subnet mask of 255.255.255.0.

If *guest1/host1* or *guest2/host2* are IPv6 addresses, or host names that resolve to IPv6 addresses, then *host1/host2* can be followed by the prefix size expressed in CIDR notation, for example 2001:db8:3003:1::543:210f/48. If the prefix size is specified it can have a value from 0 to 128; if not specified a value of 128 is assumed.

If *guest1*, *host1*, *guest2* or *host2* are numeric IPv6 addresses, they can be coded between braces, for example [2001:db8:3003:1::543:210f].

6.9.7.4 Examples

Example 1:

Define a PTP device on addresses 0E20 and 0E21. The IP address of the guest operating system running under Hercules is 192.168.1.99, whereas the host IP address (the network adapter) to use is 192.168.1.100.

```
0E20.2 PTP 192.168.1.99 192.168.1.100
```

or

```
0E20 PTP 192.168.1.99 192.168.1.100
0E21 PTP 192.168.1.99 192.168.1.100
```

Example 2:

Define PTP (MPCPTP ESCON CTC link) adapters on device addresses 0440 and 0441. The name of the TUN/TAP special character device is “/dev/net/tun”. The IP address of the Hercules guest OS side is 192.168.1.99, the host network adapter to use is 192.168.1.100

```
0440.2 PTP -n /dev/net/tun 192.168.1.99 192.168.1.100

   or

0440 PTP -n /dev/net/tun 192.168.1.99 192.168.1.100
0441 PTP -n /dev/net/tun 192.168.1.99 192.168.1.100
```

Example 3:

Define PTP (MPCPTP ESCON CTC link) adapters on device addresses 0E20 and 0E21. The WinPcap device driver capture buffer has to be set to 2048 KB and the TunTap32 I/O buffer size has to be set to 256 KB. The IP address of the guest operating system running under Hercules is 192.168.1.99 and the MAC address of the network adapter to use is 00-80-B3-E1-DF-69.

```
0E20.2 3088 PTP -n 00-80-B3-E1-DF-69 -k 2048 -i 256 192.168.1.99 0.0.0.0

   or

0E20 3088 PTP -n 00-80-B3-E1-DF-69 -k 2048 -i 256 192.168.1.99 0.0.0.0
0E21 3088 PTP -n 00-80-B3-E1-DF-69 -k 2048 -i 256 192.168.1.99 0.0.0.0
```

6.9.8 VMNET (Channel-to-Channel link via SLIP/VMNET)

6.9.8.1 Function

If the emulation mode is not specified on the configuration statement, it is assumed to be a point-to-point link to the driving system's TCP/IP stack using Willem Konynenberg's VMNET package. This provides the same function as the CTCI mode of operation, except that it uses a virtual SLIP interface instead of the TUN/TAP driver.

6.10 FBA DASD Devices

6.10.1 Function

This device statement is used to define a FBA DASD device to the Hercules configuration. The argument specifies the name of a file which contains the FBA DASD image or the INET address of a Hercules shared device server.

The file consists of fixed length 512-byte records each of which represents one physical block of the emulated disk. To allow access to a minidisk within a full-pack FBA DASD image file two additional arguments (origin and numblks) may be specified after the filename.

6.10.2 Syntax

Descriptive

devaddr devtype filename [*origin* | 0] [*numblks*] [sf=*shadowfile*] [SYNCIO]

or

devaddr devtype ipname [*:port* | :3990] [*:devnum*] [COMP=*n*]

Diagram

```
                                          ┌────── 0 ──────┐
►►── devaddr ── devtype ── filename ──────┼───────────────┼──────────────────────►
                                          └─── origin ────┘

►──┬────────────────┬──┬──────────────────┬──┬──────────────┬───────────────►◄
   └─── numblks ────┘  └── SF=shadowfile ─┘  └─── SYNCIO ───┘

   or

                                       ┌──── :3990 ────┐
►►── devaddr ── devtype ── ipname ─────┼───────────────┼──────────────────────►
                                       └──── :port ────┘

►──┬────────────────┬──┬─────────────┬──────────────────────────────────────►◄
   └─── :devnum ────┘  └── COMP=n ───┘
```

6.10.3 Parameter

devaddr This is the device address.

devtype This is the device type. Valid device types are 3310, 3370, 9332, 9335, 9336, 0671.

filename This specifies the filename (and path) of the file containing the DASD image.

origin Specifies the relative block number within the DASD image file at which the minidisk begins. The number must be less than the number of blocks in the file. The default origin is zero.

numblks Specifies the number of 512-byte blocks in the minidisk. This number must not exceed the number of blocks in the file minus the origin. If omitted or specified as an asterisk (“*”), then the minidisk continues to the end of the DASD image file.

SF= Specifies a shadow file for the DASD device. Please refer to the following CKD DASD section for information regarding the use of the SF=shadow file option.

shadowfile This is the base name of the shadow file. The handling of shadow files for FBA devices is identical as that for CKD devices. Please refer to the following CKD DASD section for information regarding the use of the SF=shadow filefile option.

SYNCIO SYNCIO enables possible synchronous I/O and is explained in detail in the following CKD DASD section.

ipname This is the host name or IP address of the system, where the Hercules shared device server is running.

port This is the port number the shared device server is listening on. If the port is omitted then the default is 3990.

devnum Devnum specifies the device number on the shared device server. If the device number is omitted then the default is the current device number on the local system.

COMP=*n* This keyword requests that the data has to be transferred compressed between the client and the server. The argument *n* specifies the compression level (1-9). Values closer to 1 mean less compression but also less processor time to perform the compression. Values closer to 9 mean the data is compressed more but also more processor time is required to compress the data.

6.10.4 Examples

Example 1:

Define a 3310 FBA DASD device on device address 0120. The name and path of the file containing the DASD image is "D:/DASD/TST001.FBA".

```
0120 3310 D:/DASD/TST001.FBA
```

Example 2:

Define a 3370 FBA DASD device on device address 0120. The DASD device is attached to a shared device server which is running on a machine with IP address 192.168.200.1 and listening on port 3990. The device address of the DASD on the shared device server is 0121.

```
0120 3370 192.168.200.1:3990:0121
```

6.11 CKD DASD Devices

6.11.1 Function

This device statement is used to define a CKD DASD device to the Hercules configuration. The argument specifies the name of a file that contains the CKD DASD image or the INET address of a Hercules shared device server.

The file consists of a 512-byte device header record followed by fixed length track images. The length of each track image depends on the emulated device type and is always rounded up to the next multiple of 512 bytes.

Volumes larger than 2 GB (for example the 3390, model 3) can be supported by spreading the data across more than one file. Each file contains a whole number of cylinders. The first file (which contains cylinders 0-2518 in the case of a 3390) usually has "_1" as the last two characters of its name. The ckddasd driver allocates the remaining files by replacing the last characters of the file name by the characters "_2", "_3" and so on.

If your operating system supports large file sizes (or 64-bit offsets) then volumes larger than 2GB can be kept in a single file. The two character suffix is not used in this case.

Alternatively the argument may specify the name of a file containg a compressed CKD DASD image (CCKD DASD files). The CKD driver will automatically detect whether the file contains a regular CKD image or a compressed CKD image.

6.11.2 Syntax

Descriptive

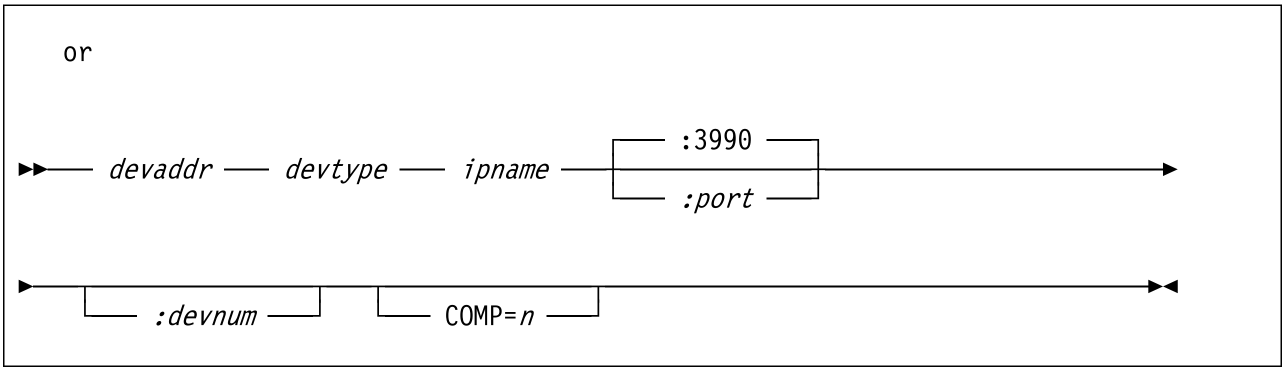
devaddr devtype filename [sf=*shadowfile*] [NOSYNCIO | SYNCIO] [READONLY]
[FAKEWRITE] [CU=*type*]

or

devaddr devtype ipname [*:port* | <u>:3990</u>] [*:devnum*] [COMP=*n*]

Diagram

```
►►── devaddr ── devtype ── filename ──┬─────────────────────┬──────►
                                      └── SF=shadowfile ────┘

►──┬───────────────┬──┬──────────────┬──┬───────────────┬──────────►
   ├── NOSYNCIO ───┤  └── READONLY ──┘  └── FAKEWRITE ──┘
   └── SYNCIO ─────┘

►──┬──────────────┬──────────────────────────────────────────────►◄
   └── CU=type ───┘
```



6.11.3 Parameter

devaddr This is the device address.

devtype This is the device type. Valid device types are 2311, 2314, 3330, 3350, 3375, 3380, 3390 and 9345.

filename This specifies the filename (and optionally the path) of the file containing the DASD image.

shadowfile A shadow file contains all the changes made to the emulated DASD device since it was created, maintained until the next shadow file is created. The moment of the shadow file's creation can be thought of a snapshot of the current emulated DASD at that time, as if the shadow file is later removed the emulated DASD reverts back to the state it was when the snapshot was taken.

Using shadow files, the base files can be kept on a read-only device like CDROM or can be defined as read-only thus ensuring that these files can never be corrupt-ted.

The shadow file does not have to actually exist when it is defined in the configu-ration file. The *shadow* operand of the sf= parameter is simply a filename template that will be used to name the shadow file whenever one is be created. Shadow files are created using the sf+ xxxx or sf+ * commands.

The shadow file name must have a position where the shadow file number will be set. This is either the character preceding the last period after the slash or the last character if there is no period. For example: *sf=shadows/linux1_*.dsk*.

Hercules console commands are provided to add a new shadow file, remove the current shadow file (with or without backward merge), compress the current shadow file and display the shadow file status and statistics.

Please note, that the "sf" parameter has to be coded in lowercase letters otherwise an error message is presented.

Details on how to work with shadow files can be found in chapter 8.153 ff.

[NO]SYNCIO SYNCIO enables possible "synchronous" I/O. This is a DASD I/O feature wherein guest I/O requests are completed "synchronously" during the actual emulated exe-cution of the SIO / SSCH (START IO / START SUBCHANNEL) instructions rather than being deferred and executed asynchronously in a separate device I/O thread.

Only I/O which are known to be able to be completed without actually needing to perform any actual host I/O are completed synchronously (e.g. whenever the data being requested is found already be in the cache). If the requested data could not be found in the cache then actual host I/O will need to be done and the request is passed to a device I/O thread to be completed asynchronously instead.

SYNCIO is the default for CKD devices. SYNCIO statistics may be displayed via the Hercules SYNCIO console command. SYNCIO may be abbreviated as *SYIO.*

Note: If you plan on using SYNCIO with Linux/390 and/or zLinux you might also want to take a look at the IODELAY system parameter as well.

READONLY READONLY returns "write inhibited" sense when a write is attempted. Note that not all the sense bits may be set absolutely correctly.

FAKEWRITE FAKEWRITE is a workaround for the READONLY sense problem that was mentioned above. In these cases the disk is not intended to be updated (e.g. MVS updates the DSCB last referenced field for a readonly file) and any writes appear to be successful even though nothing is actually written.

type Specifies the type of control unit to which this device is attached. The use of this parameter does not necessarily imply that all functions of the specified control unit are emulated. The purpose is to force a particular control unit type to be indicated in the data returned by SENSE ID and similar CCW's.

Normally the default value is appropriate and this parameter need not to be specified. Additional to the table below the following values may be specified: 3990-3 and 3990-6. The default value depends on the device type as shown in table below.

ipname This is the host name or IP address of the system where the Hercules shared device server is running.

port This is the port number the shared device server is listening on. If omitted the default is 3990.

devnum Devnum specifies the device number on the shared device server. If omitted the default is the current device number on the local system.

COMP=*n* This keyword requests that the data has to be transferred compressed between the client and the server. The argument *n* specifies the compression level (1-9). Values closer to 1 mean less compression but also less processor time to perform the compression. Values closer to 9 mean the data is compressed more but also more processor time is required to compress the data.

Two other options *LAZYWRITE* and *FULLTRACKIO* have been deprecated. They are still accepted to support compatibility with older configuration files but do actually nothing. It is strongly recommended to remove these statements from the configuration file.

The following table shows the default Control Unit types depending on on the device type:

Device Type	**Default CU Type**
2311	2841
2314	2314
3330, 3340, 3350, 3375, 3380	3880

3390	3990
9345	9343

Table 11: Default CU Types

6.11.4 Examples

Example 1:

Define a 3390 CKD DASD device on device address 0A8C. The name and path of the file containing the DASD image is "D:/DASD/SARES1.CKD".

```
0A8C 3390 D:/DASD/SARES1.CKD
```

Example 2:

Define a 3390 CKD DASD device on device address 0AA2. The name and path of the file containing the DASD image is "D:/DASD/SARES1.CKD". Synchronous I/O has to be enabled and the device has to return the "write inhibited" sense when a write is attempted.

```
0AA2 3390 D:/DASD/SARES1.CKD SYNCIO READONLY
```

Example 3:

Define a 3390 CCKD DASD device on device address 0A00. The name and path of the file containing the DASD image is "D:/DASD/SARES1.CCKD". All changes to the DASD device have to go to the shadow file(s) "D:/SHADOW/SARES1_*.CCKD".

```
0A00 3390 D:/DASD/SARES1.CCKD SF=D:/SHADOW/SARES1_*.CCKD
```

Example 4:

Define a 3390 CKD DASD device on device address 0A8C. The DASD device is attached to a shared device server which is running on a machine with IP address 192.168.200.1 and listening on port 3990. The device address of the DASD on the shared device server is also 0A8C.

```
0A8C 3390 192.168.200.1:3990:0A8C
```

6.12 Communication Lines

6.12.1 BSC (Preliminary 2703 BSC Support)

6.12.1.1 Function

This statement describes a BSC emulation line entry to either link 2 Hercules engines or a custom made program emulating a 2780, 3780 or 3x74, or a custom made program interfacing to a real BSC line.

The communication is emulated over a TCP connection. All bytes are transferred as-is (except for doubling DLE in transparent mode) just as over a real BSC link. Emulated EIA (DCD, DTR, CTS, etc.) or X.21/V.11 leads (C, T, etc.) are treated differently depending on the DIAL option selected.

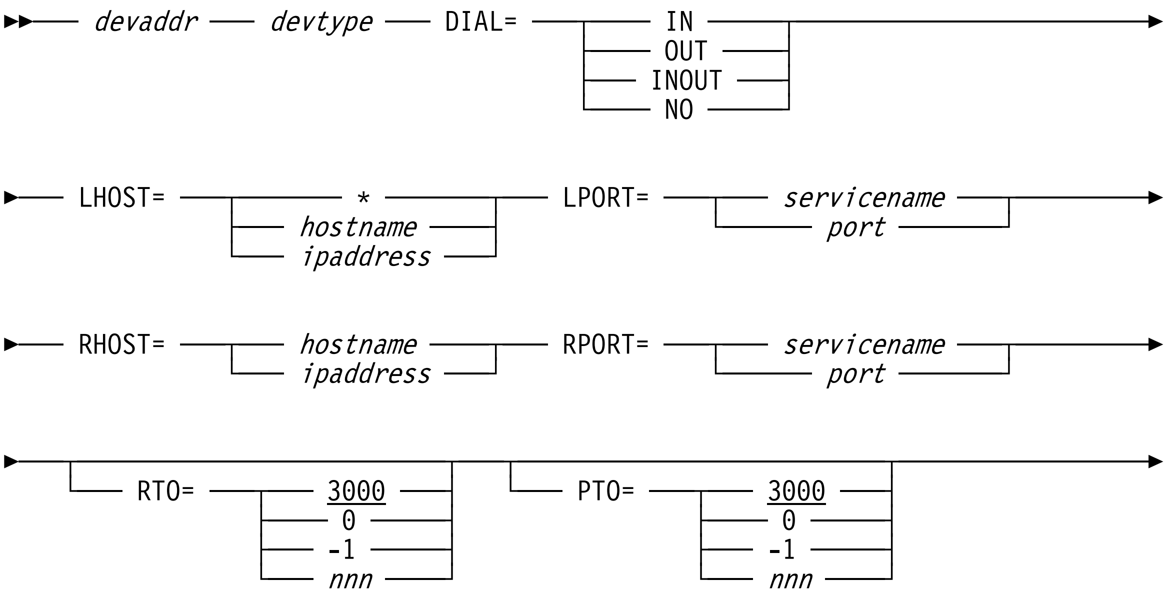
The line emulates a point-to-point BSC link. There is no point-to-multipoint handling.

6.12.1.2 Syntax

Descriptive

```
devaddr devtype DIAL={IN | OUT | INOUT | NO}
                LHOST={hostname | ipaddress | *} LPORT={servicename | port}
                RHOST={hostname | ipaddress } RPORT={servicename | port}
                [RTO={0 | -1 | nnn | 3000}]
                [PTO={0 | -1 | nnn | 3000}]
                [ETO={0 | -1 | nnn | 10000}]
```

Diagram



```
►──┬──────────────────────────────────────────────────────────►◄
   └── ETO= ──┬── 10000 ──┤
              ├──── 0 ────┤
              ├──── -1 ───┤
              └──── nnn ──┘
```

6.12.1.3 Parameter

devaddr This is the device address.

devtype This is the device type. A valid device type is 2703.

DIAL= This specifies the call direction (if any). If DIAL=NO is specified the TCP outgoing connection is attempted as soon as an ENABLE CCW is executed. Also in this mode, an incoming connection will always be accepted. If DIAL=IN or INOUT is specified, a TCP incoming call is accepted only if an ENABLE CCW is currently executing on the device. With DIAL=OUT the ENABLE CCW is rejected. When DIAL=IN or INOUT is coded a DIAL CCW allows the application to establish a TCP connection to a specific host. For other DIAL values the DIAL CCW is rejected.

LHOST= Specifies which IP address to listen on. This also conditions the network interface from which incoming calls will be accepted. An asterisk (“*”) means all incoming TCP calls are accepted regardless of the destination IP address or call origin. A given IP address when DIAL=OUT is specified has no effect.

LPORT= Specifies the TCP port for which to listen to incoming TCP calls. This value is mandatory for DIAL=IN, DIAL=INOUT or DIAL=NO. It is ignored for DIAL=OUT.

RHOST= Specifies the remote host to which to direct a TCP connection on a DIAL=NO line when an ENABLE CCW is executed. This value is mandatory when DIAL=NO is specified, It is ignored for other DIAL values.

RPORT= Specifies the remote port to which to direct a TCP connection on a DIAL=NO line when an ENABLE CCW is executed. This value is mandatory when DIAL=NO is specified, It is ignored for other DIAL values.

RTO= Specifies the number of milliseconds before terminating a read on a timeout when no read termination control character is received. A value of zero means the read ends immediately. “-1” specifies that there is no timeout.

PTO= Specifies the number of milliseconds before terminating a POLL on a timeout when no ACK or NACK sequence is received. A value of zero means the POLL ends immediately. “-1” specifies that there is no timeout.

ETO= Specifies the number of milliseconds before terminating an ENABLE operation on a timeout. The timeout applies when DIAL=NO, DIAL=IN or DIAL=INOUT is specified, , the outgoing TCP call fails (DIAL=NO) and there is no previously or currently established TCP connection for this line. When DIAL=NO is specified the timeout defaults to 10 seconds, for DIAL=IN or DIAL=INOUT, the timeout defaults to “-1”, meaning there is no timeout.

RTO, PTO and ETO are tuning options. In most cases using the default values will give the best result.

6.12.1.4 Examples

Example 1:

Define a 2703 BSC emulation line on device address 0023. The outgoing connection should be attempted as soon as an ENABLE CCW is expected, incoming connections should always be accepted (DIAL=NO). The IP address to listen on is 127.0.0.1 with port 3780. The remote host has the IP address 192.168.0.99 and the port to which to direct a TCP connection is 3781.

```
0023 2703 DIAL=NO LHOST=127.0.0.1 LPORT=3780 RHOST=192.168.0.99 RPORT=3781
```

Example 2:

Example 2 is basically the same as example 1, but needs optimized timing options. The timeout for terminating a read must be 5000 ms, the timeout for terminating a POLL must also be 5000 ms and the timeout for terminating an ENABLE operation must be 12000 ms.

```
0023 2703 DIAL=NO LHOST=127.0.0.1 LPORT=3780 RHOST=192.168.0.99 RPORT=3781
          RT=5000 PTO=5000 ETO=12000
```

6.12.2 TTY (Preliminary 2703 TELE2 TTY Support)

6.12.2.1 Function

This statement describes a 2703 Telegraph Terminal Control Type II (TTY 33/35) stop/start line, providing access to the Host OS via a standard telnet client.

To the host OS the line emulates an asynchronous TELE2 connection. The communication is emulated over a telnet connection.

6.12.2.2 Syntax

Descriptive

devaddr *devtype* LPORT=*port* DIAL=IN TTY=1

Diagram

►►── *devaddr* ── *devtype* ── LPORT=*port* ── DIAL=IN ── TTY=1 ──────────►◄

6.12.2.3 Parameter

devaddr This is the device address.

devtype This is the device type. A valid device type is 2703.

port Specifies the TCPIP port to listen on for incoming TCP calls.

DIAL=IN Specifies that this line is for in-bound calls. This parameter is required.

TTY=1 Specifies that this definition is for a TTY port. This parameter is required.

6.12.2.4 Examples

Example 1:

Define a 2703 TTY emulation line on device address 00C3 listening on port 32003.

```
00C3 2703 LPORT=32003 DIAL=IN TTY=1
```

7. Hercules Console

7.1 Hercules Hardware Console

Various aspects of Hercules can be controlled with console commands. These commands are entered from the command line of the Hercules hardware management console (HMC).

The Hercules hardware management console window is the panel that is active by default after starting Hercules. An alternate semi-graphical "New Panel" display showing devices, registers etc. is also available. Pressing the escape key switches between the two panels.

The main Hercules screen contains a scrollable list of messages with a command input area and a system status line at the bottom of the screen. To scroll through the messages, use either the Page Up or Page Down keys, the Ctrl + Up Arrow or Ctrl + Down Arrow keys, or the Home or End and/or the Ctrl + Home or Ctrl + End keys.

Important messages are highlighted in a different color (usually red) and are prevented from being scrolled off the screen for two minutes. If Extended Cursor handling is available then important messages currently at the top of the screen can be removed early by moving the cursor to the line containing the message and then pressing enter.

Use the Insert key to switch between insert and overlay mode when typing in the command input area. Use the Home and End keys to move to the first or last character of the command you are typing, or the use the left/right arrow keys to move to a specific character. Use the Escape key to erase the input area. Pressing "Escape" when the command input area is already empty causes the screen to switch to the semi-graphical "New Panel" display mode, which shows the overall status of the system and devices.

When in the semi-graphical "New Panel" display mode there is no command input area. Instead, single character "hot keys" are used to issue some of the more common functions such as starting or stopping the CPU. The hot-keys are those which are highlighted. Pressing the '?' key displays brief help information on how to use the semi-graphical panel.

When a command is prefixed with a hyphen ('-'), the command will not be redisplayed at the console. This can be used in scripts and is also used internally when commands are to be invoked without being redisplayed at the console.

Prefixing a command with a hyphen ('-') makes it a silent non-echoing command and prevents the command from appearing in the command recall history list (see 'HST' command).

Commands can be executed in the background by specifying an ampersand ('&') as last argument:

- "SCRIPT ABC" runs in the foreground

- "SCRIPT ABC &" runs in the background

- "SCRIPT ABC &&" runs in the foreground but passes '&' as last argument

- "SCRIPT ABC & &" runs in the background and passes '&'as last argument

Sample screenshots of the Hercules HMC are shown in the pictures on the following pages.

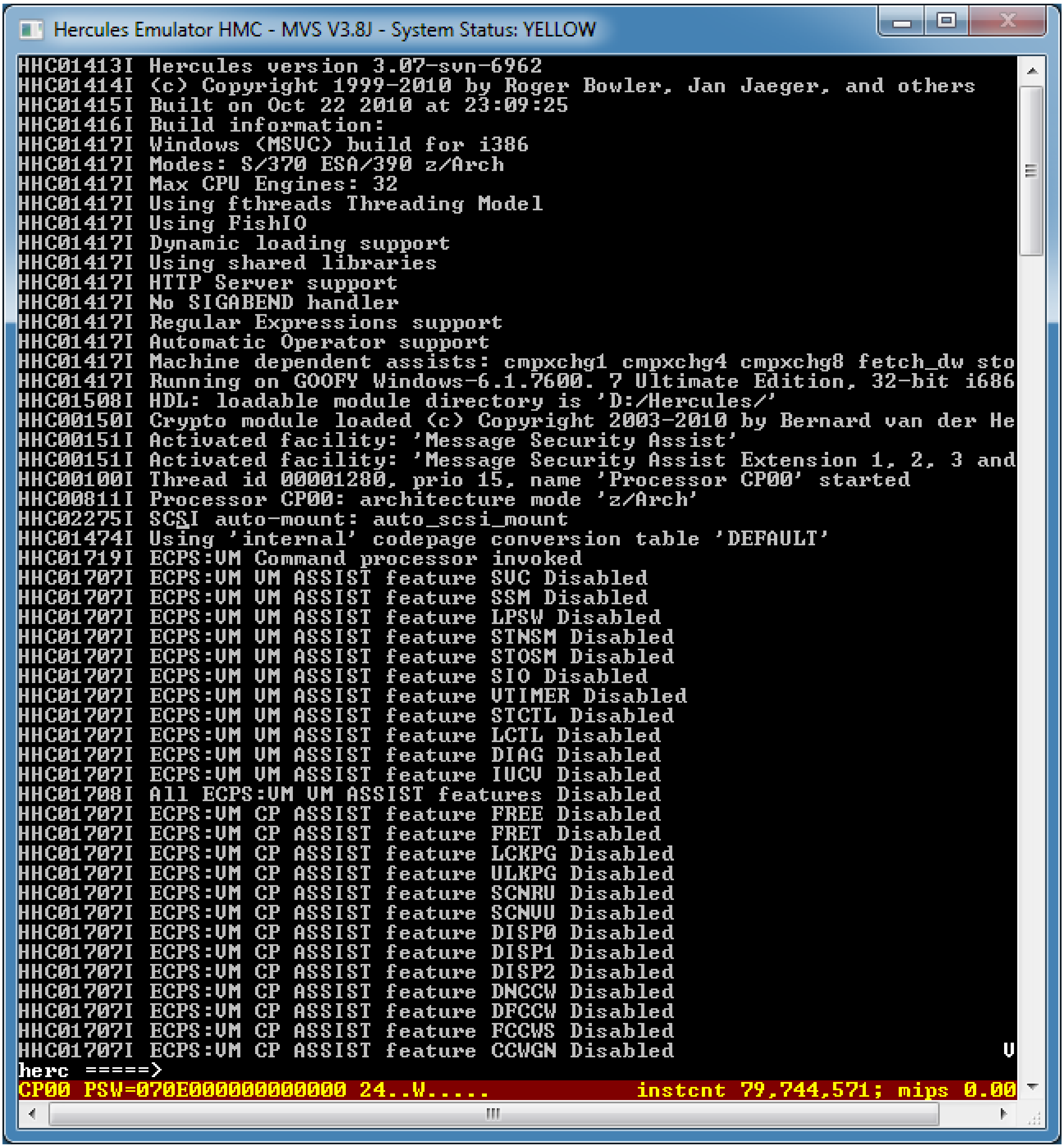


Figure 3: Hercules Hardware Console

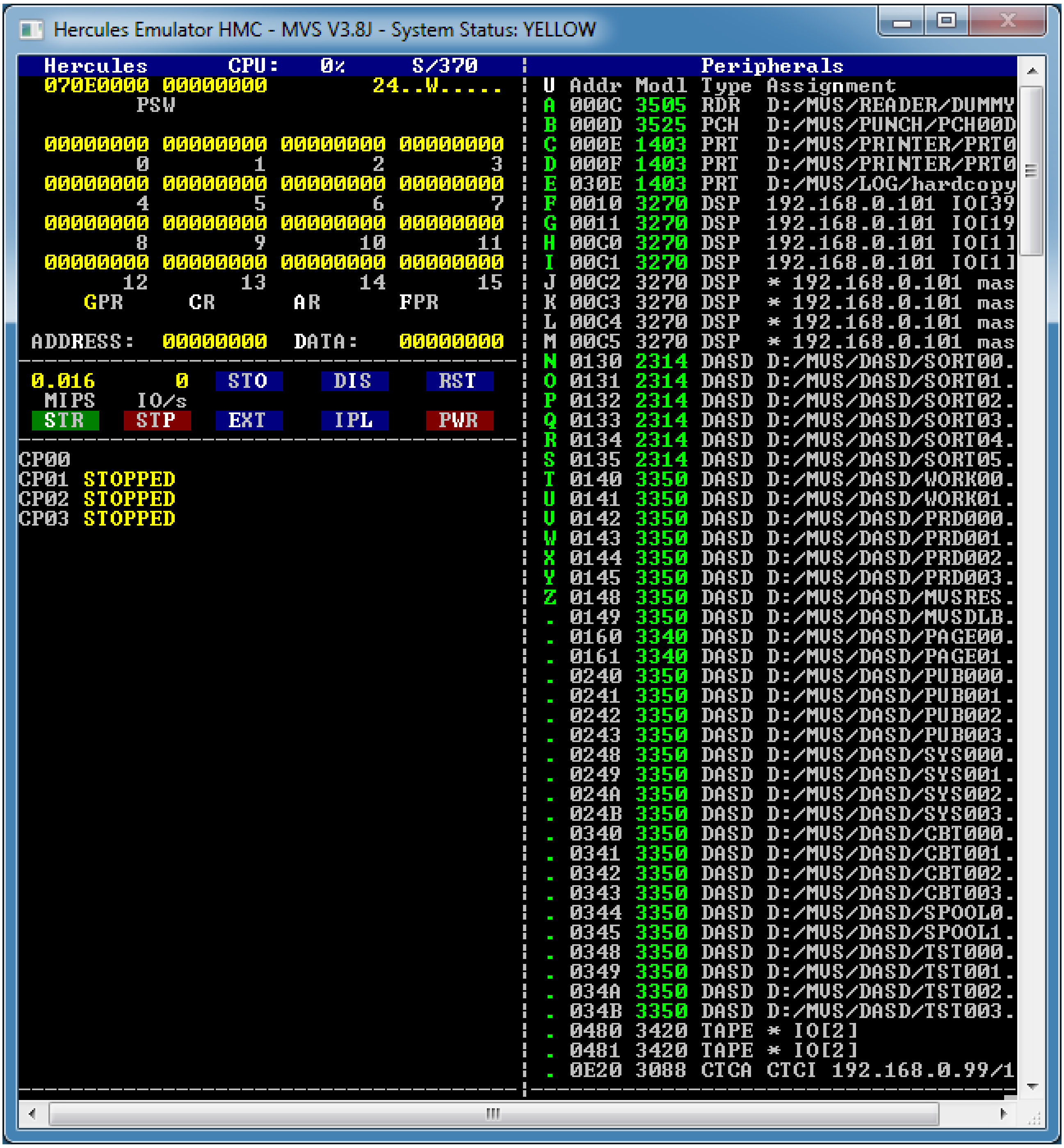


Figure 4: Hercules device and status panel

7.2 Web browser interface

It is not necessary to work directly on the PC where Hercules is running to enter the console commands. Instead of using the hardware management console on the Hercules server itself it is possible to work from a browser interface in a web browser.

The requirement to be able to work with the web browser interface is a running Hercules HTTP server. The HTTP server is started through the HTTPPORT and HTTPROOT system parameter. For details on configuring the HTTP server please see section "System Parameter Descriptions".

Once the HTTP server is started, the interface can be used from a web browser on any machine, that has a network connection to the Hercules server. With this interface you can operate Hercules nearly in the same way as you would do locally on the hardware management console.

The following picture shows the main panel of the web browser interface.

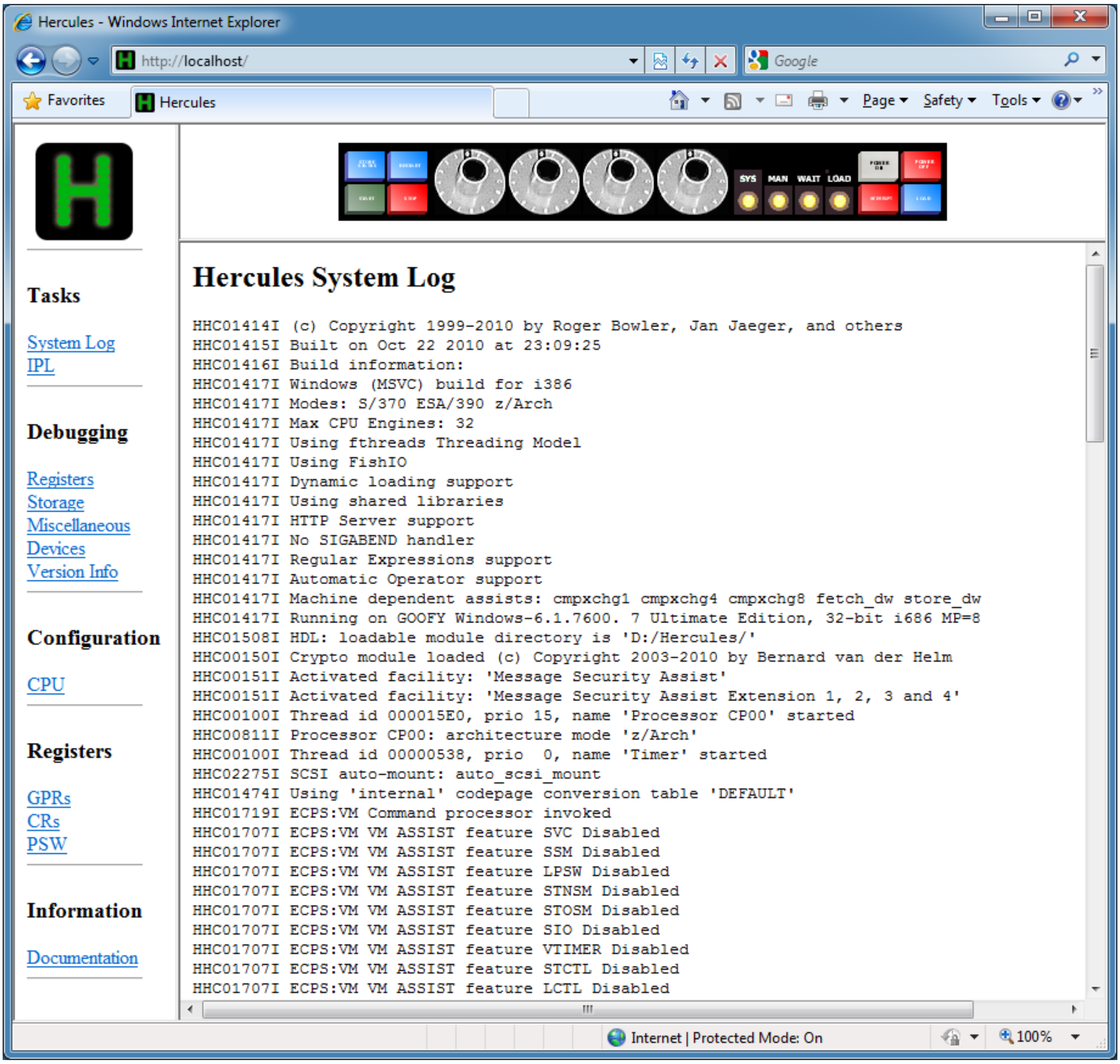


Figure 5: Hercules web browser interface

7.3 Using the keyboard

The usage of the keyboard in the Hercules hardware management console is distinguished between normal cursor handling and extended cursor handling.

7.3.1 Normal cursor handling

The normal cursor handling is available on all platforms (Windows and Unix).

Key	Action
Esc	Erases the contents of the command input area. If the command input area is already empty, switches to semi-graphical New Panel.
Del	Deletes the character at the cursor position.
Backspace	Erases the previous character.
Insert	Toggles between insert mode and overlay mode.
Tab	Attempts to complete the partial file name at the cursor position in the command input area. If more than one possible file exists, a list of matching file names is displayed.
Home	Moves the cursor to the start of the input in the command input area. If the command input area is empty, scrolls the message area to the top.
End	Moves the cursor to the start of the input in the command input area. If the command input area is empty, scrolls the message area to the bottom.
Page Up	Scrolls the message area up one screen.
Page Down	Scrolls the message area down one screen.
Up arrow	Recalls the previous command into the input area.
Down arrow	Recalls the next command into the input area.
Right arrow	Moves cursor to the next character of the input area.
Left arrow	Moves cursor to the previous character of the input area.
Ctrl + Up arrow	Scrolls the message area up one line.
Ctrl + Down arrow	Scrolls the message area down one line.
Ctrl + Home	Scrolls the message area to the top.
Ctrl + End	Scrolls the message area to the bottom.

Table 12: Normal cursor handling

7.3.2 Extended cursor handling

The following additional keyboard functions are effective when the Hercules Extended Cursor Handling feature (OPTION_EXTCURS) is activated at compile time. At present, this feature is activated on the Windows platform only.

Key	Action
Alt + Up arrow	Moves cursor up one row.
Alt + Down arrow	Moves cursor down one row.
Alt + Right arrow	Moves cursor right one column.
Alt + Left arrow	Moves cursor left one column.
Tab	If the cursor is outside the command input area, moves cursor to the start of the input in the command input area. Otherwise behaves like as described in the previous table.
Home	If the cursor is outside the command input area, moves cursor to the start of the input in the command input area. Otherwise behaves like as described in the previous table.
End	If the cursor is outside the command input area, moves cursor to the end of the input in the command input area. Otherwise behaves like as described in the previous table.

Table 13: Extended cursor handling

7.3.3 Windows event handler

Hercules has handlers for certain Windows events. The following table shows the trapped events and the according action(s).

Key or function	Action
CTRL-Break	Simulates the External Interrupt Key being pressed.
CTRL-C	CTRL-C is currently caught, but there is no action taken.
Close	The normal close button (the red "X" box) has been disabled to prevent an unintended shutdown of Hercules. The close function via the Windows menu ("File -> Exit") however is still available. In this case Hercules initiates an immediate shutdown in order to close all files correctly.
Shutdown	Shutdown ("Start -> Shut down -> Shut down") initiates an immediate shutdown of Hercules in order to close all files correctly.
Logoff	Logoff ("Start -> Shut down -> Log off") initiates an immediate shutdown of Hercules in order to close all files correctly.

Table 14: Windows event handler

7.4 Log formats

The appearance of the log in terms of prefixing the messages is dependent on the used environment. The most flexible ways of prefixing the messages is given when running Hercules using the WinGUI. Running Hercules with the native console gives only limited options. The following table show an overview of the log options, depending on the Hercules version used.

Message Prefix	Native Hercules Console	Hercules with WinGUI
Date	No	Yes
Time	Yes	Yes
Process ID	No	Yes
Prefix(es) visible on console	No	Yes
Prefix(es) visible on hardcopy log	Yes	Yes

Table 15: Message prefix overview

Most of the console command examples in this book are made without the WinGUI and with LOGOPT set to NOTIMESTAMP as described below. This is to ensure that the sample output from the commands fits into the text boxes in this book.

7.4.1 Hercules Console

Using Hercules under Linux, Mac OS or Windows (without WinGUI) lets you prefix the console messages with a timestamp. The difference between these options is shown in the next figure.

Switching the option can be done with the 'logopt' console command or can be set at startup time through the 'logopt' system parameter.

Please note that the console itself does not show the timestamp, independent of the 'logopt' settings. Only the hardcopy log does show the timestamp.

```
.
.
.
logopt notimestamp
HHCPN197I Log option set: NOTIMESTAMP
gpr
CPU0000: R0=0000000000000000 R1=0000000000000000
CPU0000: R2=0000000000000000 R3=0000000000000000
CPU0000: R4=0000000000000000 R5=0000000000000000
CPU0000: R6=0000000000000000 R7=0000000000000000
CPU0000: R8=0000000000000000 R9=0000000000000000
CPU0000: RA=0000000000000000 RB=0000000000000000
CPU0000: RC=0000000000000000 RD=0000000000000000
CPU0000: RE=0000000000000000 RF=0000000000000000
```

```
.
.
.
23:01:32 logopt timestamp
23:01:32 HHCPN197I Log option set: TIMESTAMP
23:01:36 gpr
23:01:36 CPU0000: R0=0000000000000000 R1=0000000000000000
23:01:36 CPU0000: R2=0000000000000000 R3=0000000000000000
23:01:36 CPU0000: R4=0000000000000000 R5=0000000000000000
23:01:36 CPU0000: R6=0000000000000000 R7=0000000000000000
23:01:36 CPU0000: R8=0000000000000000 R9=0000000000000000
23:01:36 CPU0000: RA=0000000000000000 RB=0000000000000000
23:01:36 CPU0000: RC=0000000000000000 RD=0000000000000000
23:01:36 CPU0000: RE=0000000000000000 RF=0000000000000000.
.
.
.
```

Figure 6: Message prefixing using the native Hercules console

7.4.2 Hercules console using the WinGUI

Using the WinGUI with the Windows version of Hercules gives you the possibility to prefix all messages with the current date, the current time and the process ID. These prefixing options can be selected individually in the "Format" tab of the "Advanced Logging Options", where they can be separately turned on or off as showed in the next figure.

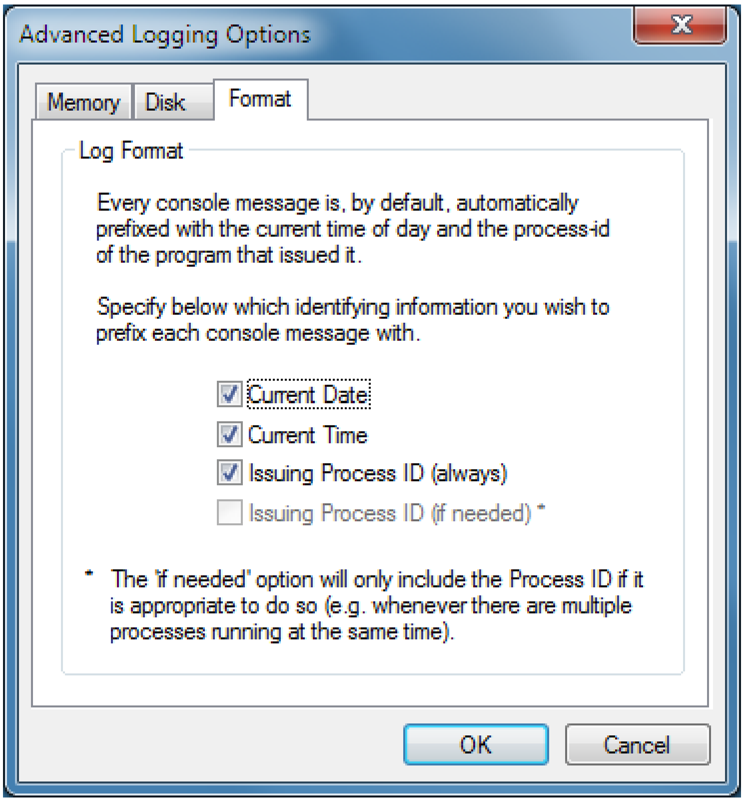


Figure 7: WinGUI Advanced Logging Options

The ‘logopt’ system parameter or console command has no effect when using the WinGUI. Prefixing the messages is only influenced over the GUI settings. The following figure shows the log messages using all prefixing options (date, time and process ID) as specified above.

```
.
.
.
2010/06/18 19:12:18.071 000008E4 HHC01416I Build information:
2010/06/18 19:12:18.071 000008E4 HHC01417I Windows (MSVC) build for AMD64
2010/06/18 19:12:18.071 000008E4 HHC01417I Modes: S/370 ESA/390 z/Arch
2010/06/18 19:12:18.071 000008E4 HHC01417I Max CPU Engines: 8
2010/06/18 19:12:18.071 000008E4 HHC01417I Using fthreads Threading Model
2010/06/18 19:12:18.072 000008E4 HHC01417I Using FishIO
2010/06/18 19:12:18.072 000008E4 HHC01417I Dynamic loading support
2010/06/18 19:12:18.072 000008E4 HHC01417I Using shared libraries
2010/06/18 19:12:18.072 000008E4 HHC01417I HTTP Server support
2010/06/18 19:12:18.072 000008E4 HHC01417I No SIGABEND handler
2010/06/18 19:12:18.072 000008E4 HHC01417I Regular Expressions support
2010/06/18 19:12:18.072 000008E4 HHC01417I Automatic Operator support
.
.
.
```

Figure 8: Message prefixing using the console of the Hercules WinGUI

7.5 Programmed Function Key (PF Key) Support

The Hercules console supports the usage of PF keys. The next sections show how to define and use PF keys with Hercules.

7.5.1 Usage

The command to be assigned to the PF key has to be defined with a DEFSYM statement. This can be done through a DEFSYM system parameter statement in the Hercules configuration file or through a panel command (manually from the console or coded in the Hercules run-commands file).

On Windows systems PF keys PF01 to PF48 are assignable, on non-Windows systems PF01 to PF20. The following special keys must be used to access the PF keys:

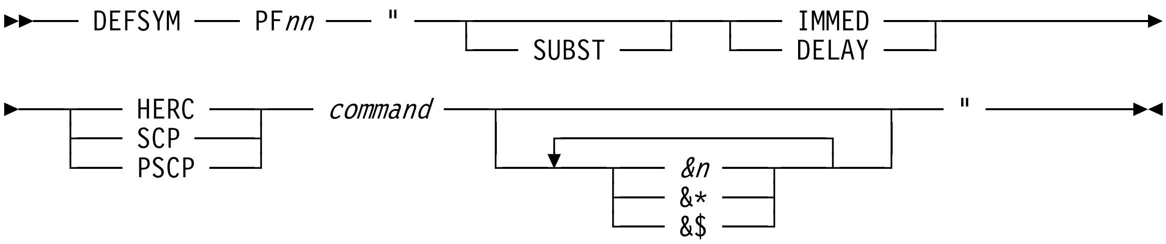
PF01-PF12	Press PF key only
PF13-PF24	Press SHIFT and PF key
PF25-PF36	Press CTRL and PF key
PF37-PF48	Press ALT and PF key

7.5.2 Syntax

Descriptive

DEFSYM PF*nn* "[SUBST] {IMMED | DELAY} {HERC | SCP | PSCP}
    command [*&n* | *&** | *&$* [...]]"

Diagram



7.5.3 Parameter

PFnn This identifies the PF key to be defined, where *nn* is a number from 01 to 48. This number must be specified with the leading zero for PF keys 1 - 9 (PF01, …, PF09).

SUBST This indicates that the PF key data might contain substitution placeholders (*&n,* see below).

IMMED This indicates the point in time when the PF key is processed. IMMED indicates that the PF data is processed when the PF key is pressed.

DELAY This indicates the point in time when the PF key is processed. DELAY indicates that the PF data is displayed on the command line when the PF key is pressed. The data may then be modified if necessary. To finally process the PF data the "ENTER" key must be pressed.

HERC The command target for the assigned command is the Hercules Emulator.

SCP The command target for the assigned command is the System Control Program (the guest operating system running under Hercules).

PSCP The assigned command is a System Control Program (guest operating system) priority command.

command This defines the command that is to be assigned to the PF key.

&n A substitution placeholder is identified by an ampersand ("&"), followed by a decimal number from 0 to 9. If two ampersands are detected ("&&") then a single ampersand is substituted.

&* Variable "&*" represents all of the remaining tokens beyond the last substitution variable used. A single space is added between each token. Tokens can be enclosed in paired single or double quotes. The quotes are included as part of the token string.

**&$** Variable "&$" represents the data remaining on the line past token 9. Variable "$*" (see below) does include the data from "&$".A single space is added between each token.

Tokens can be enclosed in paired single or double quotes. The quotes are included as part of the token string.

7.5.4 Default PF Key Assignments

The following PF keys are defined per default:

PF01 'SUBST IMMED herc help &0'

PF10 'SUBST DELAY herc devinit &*'

PF11 'IMMED herc devlist TAPE'

7.5.5 Examples

Example 1:

```
DEFSYM PF01 “SUBST IMMED HERC HELP &0”
```

Typing ‘MAXRATES’ on the command line and pressing PF01 would place the following on the command line and immediately execute the command:

```
HERC HELP MAXRATES
```

Example 2:

```
DEFSYM PF02 “SUBST DELAY HERC DEVINIT &0 &*”
```

Typing ‘580 “HETTAPE.HET” RO’ on the command line and pressing PF02 would place the following on the command line:

```
HERC DEVINIT 580 “HETTAPE.HET” RO
```

Pressing an additional “ENTER” does execute the command.

Example 3:

```
DEFSYM PF03 “SUBST DELAY HERC DEVINIT &0 &* &1 &* &2”
```

Typing ‘T1 T2 T3 T4 T5 T6’ on the command line and pressing PF03 would place the following on the command line:

```
HERC DEVINIT T1 T2 T3 T4 T5 T6 T2 T3 T4 T5 T6 T3
```

Pressing an additional “ENTER” does execute the command.

7.6 Hercules Console Commands (sorted alphabetically)

The following table shows an overview of the valid Hercules console commands that can be entered in the control panel (Hercules console display). In this table the commands are sorted alphabetically rather than grouped by functionality.

Please note that the availability of certain commands depends on the build options used when Hercules was compiled. For a list of all build options and their related console commands please consult “Appendix E. Build Options for System Parameters and Console Commands”.

Command	**Description**
!message	SCP priority message
#	Silent comment
$locate	Display and verify Hercules control blocks
$test	Custom test command
$zapcmd	Enable or disable system parameters and console commands
*	Loud comment
.reply	SCP command
?	List all commands / command specific help (alias for help)
abs	Display or alter absolute storage
aea	Display AEA (absolue-effective-address) tables
aia	Display AIA (absolute-instruction-address) tables
ar	Display access registers
archlvl	Set architecture level
archmode	Set architecture mode (alias for ARCHLVL command)
attach	Configure device
auto_scsi_mount	Automatic SCSI tape mounts (deprecated, use SCSIMOUNT instead)
autoinit	Display or set the automatic creation of empty tape files
automount	Display or update allowable tape automount directories
b	Set breakpoint (synonym for B+)
b+	Set breakpoint (synonym for B)
b-	Delete breakpoint

Command	Description
cache	Execute cache related commands
cachestats	Display cache statistics
capping	Display or set CPU capping value
cckd	CCKD command
cd	Change directory
cf	Configure current CPU online or offline
cfall	Configure all CPU's online or offline
clocks	Display TOD clock and CPU timer
cmdlevel	Display or set the current command group
cmdlvl	Alias for cmdlevel
cmdsep	Display or set command line separator
cmdtgt	Specify the command target
cnslport	Display or set console port
codepage	Display or set codepage conversion table
conkpalv	Display / alter console TCP/IP keep-alive settings
cp_updt	Create or modify user character conversion table
cpu	Define target CPU for console display and commands
cpuidfmt	Display or set format BASIC / 0 / 1 STIDP generation
cpumodel	Display or set CPU model number
cpuprio	Display or set CPU thread process priority
cpuserial	Display or set CPU serial number
cpuverid	Display or set CPU version code
cr	Display or alter control registers
cscript	Cancel a running script thread
ctc	Enable / disable CTC debugging
define	Rename device

Command	Description
defstore	Display or define main and expanded storage values
defsym	Define a symbol
delsym	Delete a symbol
detach	Remove device
devinit	Reinitialize device
devlist	List device, device class or all devices
devprio	Display or set device threads process priority
devtmax	Display or set max device threads
diag8cmd	Display or set DIAGNOSE 8 command option
dir	Display file and directory listing
ds	Display subchannel
ecpsvm	ECPS:VM commands
engines	Set processor engines type
exec	Execute a REXX script
exit	Terminate the emulator
ext	Generate external interrupt
f{+/-} addr	Mark frames usable / unusable
fcb	Display current FCB or load new FCB image
fpc	Display or alter floating point control register
fpr	Display or alter floating point registers
g	Turn off instruction stepping and start all CPUs
gpr	Display or alter general purpose registers
hao	Hercules Automatic Operator (HAO)
help	List all commands / command specific help
herc	Send Hercules command
herclogo	Read a new Hercules logo file

Command	Description
hercprio	Display or set Hercules process priority
hst	History of commands
http	Start, stop, modify or display HTTP server
i	Generate I/O attention interrupt for device
icount	Display individual instruction counts
iodelay	Display or set I/O delay value
ipending	Display pending interrupts
ipl	IPL Normal from device xxxx
iplc	IPL Clear from device xxxx
k	Display CCKD internal trace
kd	Clear held messages
ldmod	Load a module
legacysenseid	Display or set SENSE ID CCW (x'E4) feature
loadcore	Load a core image from a file
loadparm	Set IPL parameter
loadtext	Load a text deck file
log	Direct log output
logopts	Change log options
lparname	Display or set LPAR name
lparnum	Display or set LPAR identification number
ls	Display file and directory listing
lsdep	List module dependencies
lsmod	List dynamic modules
mainsize	Display or set main storage size
manufacturer	Display or set STSI manufacturer code
maxcpu	Display or set maximum number of CPUs

Command	Description
maxrates	Display highest MIPS/SIO rate or set new reporting interval
memlock	Lock Hercules memory
message	Display message on console like VM
model	Display or set STSI model code
modpath	Display or set dynamic load module path
mounted_tape_reinit	Control tape initialization
msg	Display message on console like VM
msghld	Display or set timeout value of held messages
msglevel	Display or set the current message display output
msglvl	Display or set the current message display output (alias for msglevel command)
msgnoh	Display message on console like VM, but without header
mt	Control magnetic tape operation
numcpu	Display or set number of emulated CPUs
numvec	Display or set number of vector facilities
ostailor	Tailor trace information for specific operating system
panrate	Display or set console refresh rate
pantitle	Display or set console window title
pgmprdos	Set LPP license seting
pgmtrace	Trace program interrupts
plant	Display or set STSI plant code
pr	Display prefix register
pscp	Send system control program priority message
psw	Display or alter program status word
ptp	Enable / disable PTP debugging
ptt	Display or set internal trace
pwd	Print working directory

Command	Description
qcpuid	Display CPU ID
qd	Query device information
qpfkeys	Display the current PF key settings
qpid	Display process ID of Hercules
qports	Display TCP/IP ports in use
qproc	Display processors type and utilization
qstor	Display main and expanded storage values
quiet	Toggle automatic refresh of console display data
quit	Terminate the emulator
quitmout	Display or set quit timeout value
r	Display or alter real storage
restart	Generate restart interrupt
resume	Resume Hercules
rexx	Display or set REXX interpreter settings
rmmod	Delete a module
s	Instruction stepping
s+	Instruction stepping on
s-	Instruction stepping off
s?	Instruction stepping query
s{+/-} dev	Turn CCW stepping on / off
savecore	Save a core image to a file
sclproot	Set or display SCLP base directory
scp	Send system control program command
scpecho	Display or set echo to console and history of SCP replies
scpimply	Display or set option to pass non-Hercules commands to the SCP
script	Run a sequence of console commands contained in a file

Command	Description
scsimount	Automatic SCSI tape mounts
sf+	Create a new shadow file
sf-	Delete a shadow file
sfc	Compress a shadow file
sfd	Display shadow file statistics
sfk	Perform a chkdsk on the active shadow file
sh	Shell command
shcmdopt	Display or set shell command option
showdvol1	Display or set enable showing of DASD volsers in device list
shrd	Display or set shared device server trace
shrdport	Set shared device server port
sizeof	Display size of structures
srvprio	Display or set server threads process priority
ssd	Signal Shutdown
start	Start CPU or printer / punch device
startall	Start all CPU's
stop	Stop CPU or printer / punch device
stopall	Stop all CPU's
store	Store CPU status at absolute zero
suspend	Suspend Hercules
symptom	Instruction trace display option (alias for TRACEOPT command)
syncio	Display syncio device statistics
sysclear	SYSTEM CLEAR RESET manual operation
sysepoch	Set base date for TOD clock
sysreset	SYSTEM RESET manual operation
t	Instruction trace

Command	Description
t+	Instruction trace on
t-	Instruction trace off
t?	Instruction trace query
t{+/-} CKD	Turn CKD_KEY tracing on / off
t{+/-} dev	Turn CCW tracing on / off
timerint	Display or set timers update interval
tlb	Display TLB tables
toddrag	Display or set TOD clock drag factor
todprio	Display or set timer thread process priority
traceopt	Instruction trace display option
tt32	Control / query CTCI-WIN functionality
tzoffset	Set TOD clock offset from GMT
u	Disassemble storage
uptime	Display Hercules Emulator uptime
v	Display or alter virtual storage
version	Display version information
xpndsize	Display or set expanded storage size
yroffset	Set TOD clock offset from actual date

Table 16: Hercules Console Commands (sorted alphabetically)

7.7 Hercules Console Commands (grouped by functionality)

The next table shows an overview of the valid Hercules console commands that can be entered in the control panel (Hercules console display). In this table the commands are grouped by functionality.

Please note that the availability of certain commands depends on the build options used when Hercules was compiled. For a list of all build options and the related console commands please consult “Appendix E. Build Options for System Parameters and Console Commands”.

Command	Description
help	List all commands / command specific help

Command	Description
?	List all commands / command specific help (alias for help)
#	Silent comment
*	Loud comment
cmdlevel	Display or set the current command group
cmdlvl	Alias for cmdlevel
cmdsep	Display or set command line seperator
message	Display message on console like VM
msg	Display message on console like VM
msgnoh	Display message on console like VM, but without header
quiet	Toggle automatic refresh of console display data
hst	History of commands
hao	Hercules Automatic Operator (HAO)
log	Direct logger output
logopt	Display or set logging options
msglevel	Display or set the current message display output
msglvl	Display or set the current message display output (alias for msglevel command)
ostailor	Tailor trace information for specific operating system
uptime	Display Hercules Emulator uptime
version	Display version information
exit	Terminate the emulator
quit	Terminate the emulator
quitmout	Display or set quit timeout value
cpu	Define target CPU for console display and commands
fcb	Display current FCB or load new FCB image

Command	**Description**
start	Start CPU or printer / punch device
stop	Stop CPU or printer / punch device
startall	Start all CPU’s
stopall	Stop all CPU’s
cf	Configure current CPU online or offline
cfall	Configure all CPU’s online or offline
.reply	SCP command
!message	SCP priority message
scpecho	Display or set echo to console and history of SCP replies
scpimply	Display or set option to pass non-Hercules commands to the SCP
ssd	Signal Shutdown
ptt	Display or set internal trace
i	Generate I/O attention interrupt for device
ext	Generate external interrupt
restart	Generate restart interrupt
archlvl	Set architecture level
archmode	Set architecture mode (alias for ARCHLVL command)
engines	Set processor engines type
defstore	Display or define main and expanded storage values
mainsize	Display or set main storage size
xpndsize	Display or set expanded storage size
cpuprio	Display or set CPU thread process priority
devprio	Display or set device threads process priority
hercprio	Display or set Hercules process priority
srvprio	Display or set server threads process priority
todprio	Display or set timer thread process priority

Command	Description
maxcpu	Display or set maximum number of CPUs
numcpu	Display or set number of installed CPUs
numvec	Display or set number of vector facilities
loadparm	Set IPL parameter
lparname	Display or set LPAR name
lparnum	Display or set LPAR identification number
cpumodel	Display or set CPU model number
cpuserial	Display or set CPU serial number
cpuverid	Display or set CPU version code
cpuidfmt	Display or set format BASIC / 0 / 1 STIDP generation
cnslport	Display or set console port
capping	Display or set CPU capping value
manufacturer	Display or set STSI manufacturer code
model	Display or set STSI model code
plant	Display or set STSI plant code
pgmprdos	Set LPP license setting
codepage	Display or set codepage conversion table
cp_updt	Create or modify user character conversion table
diag8cmd	Display or set DIAGNOSE 8 command option
shcmdopt	Display or set shell command option
legacysenseid	Display or set SENSE ID CCW (x’E4) feature
sysepoch	Set base date for TOD clock
tzoffset	Set TOD clock offset from GMT
yroffset	Set TOD clock offset from actual date
ipl	IPL Normal from device xxxx
iplc	IPL Clear from device xxxx

Command	Description
sysreset	SYSTEM RESET manual operation
sysclear	SYSTEM CLEAR RESET manual operation
store	Store CPU status at absolute zero
sclproot	Set or display SCLP base directory
http	Start, stop, modify or display HTTP server
psw	Display or alter program status word
gpr	Display or alter general purpose registers
fpr	Display or alter floating point registers
fpc	Display or alter floating point control register
cr	Display or alter control registers
ar	Display access registers
pr	Display prefix register
timerint	Display or set timers update interval
clocks	Display TOD clock and CPU timer
ipending	Display pending interrupts
ds	Display subchannel
abs	Display or alter absolute storage
r	Display or alter real storage
v	Display or alter virtual storage
u	Disassemble storage
devtmax	Display or set max device threads
k	Display CCKD internal trace
memlock	Lock Hercules memory
attach	Configure device
detach	Remove device
define	Rename device

Command	Description
devinit	Reinitialize device
devlist	List device, device class or all devices
qcpuid	Display CPU ID
qd	Query device information
qpfkeys	Display the current PF key settings
qpid	Display process ID of Hercules
qports	Display TCP/IP ports in use
qproc	Display processors type and utilization
qstor	Display main and expanded storage values
mounted_tape_reinit	Control tape initialization
auto_scsi_mount	Automatic SCSI tape mounts (deprecated, use SCSIMOUNT instead)
scsimount	Automatic SCSI tape mounts
autoinit	Display or set the automatic creation of empty tape files
automount	Display or update allowable tape automount directories
mt	Control magnetic tape operation
cd	Change directory
dir	Display file and directory listing
ls	Display file and directory listing
pwd	Print working directory
sh	Shell command
exec	Execute a REXX script
rexx	Display or set REXX interpreter settings
cache	Execute cache related commands
cachestats	Display cache statistics
cckd	CCKD command
shrd	Display or set shared device server trace

Command	Description
shrdport	Set shared device server port
conkpalv	Display / alter console TCP/IP keep-alive settings
t	Instruction trace
t+	Instruction trace on
t-	Instruction trace off
t?	Instruction trace query
s	Instruction stepping
s+	Instruction stepping on
s-	Instruction stepping off
s?	Instruction stepping query
b	Set breakpoint (synonym for B+)
b+	Set breakpoint (synonym for B)
b-	Delete breakpoint
g	Turn off instruction stepping and start all CPUs
icount	Display individual instruction counts
pgmtrace	Trace program interrupts
savecore	Save a core image to a file
loadcore	Load a core image from a file
loadtext	Load a text deck file
modpath	Display or set dynamic load module path
ldmod	Load a module
rmmod	Delete a module
lsmod	List dynamic modules
lsdep	List module dependencies
iodelay	Display or set I/O delay value
ctc	Enable / disable CTC debugging

Command	Description
ptp	Enable / disable PTP debugging
tt32	Control / query CTCI-WIN functionality
toddrag	Display or set TOD clock drag factor
syncio	Display syncio device statistics
kd	Clear held messages
msghld	Display or set timeout value of held messages
maxrates	Display highest MIPS/SIO rate or set new reporting interval
panrate	Display or set console refresh rate
pantitle	Display or set console window title
showdvol1	Display or set enable showing of DASD volsers in device list
defsym	Define a symbol
delsym	Delete a symbol
script	Run a sequence of console commands contained in a file
cscript	Cancel a running script thread
ecpsvm	ECPS:VM commands
aea	Display AEA (absolue-effective-address) tables
aia	Display AIA (absolute-instruction-address) tables
tlb	Display TLB tables
$locate	Display and verify Hercules control blocks
$test	Custom test command
$zapcmd	Enable or disable system parameters and console commands
sizeof	Display size of structures
suspend	Suspend Hercules
resume	Resume Hercules
herclogo	Read a new Hercules logo file

Command	Description
traceopt	Instruction trace display option
symptom	Instruction trace display option (alias for TRACEOPT command)
cmdtgt	Specify the command target
herc	Send Hercules command
scp	Send system control program command
pscp	Send system control program priority message
sf+	Create a new shadow file
sf-	Delete a shadow file
sfc	Compress a shadow file
sfd	Display shadow file statistics
sfk	Perform a chkdsk on the active shadow file
t{+/-} CKD	Turn CKD_KEY tracing on / off
t{+/-} dev	Turn CCW tracing on / off
s{+/-} dev	Turn CCW stepping on / off
f{+/-} addr	Mark frames usable / unusable

Table 17: Hercules Console Commands (grouped by functionality)

8. Console Command Descriptions

8.1 !message (SCP priority message)

8.1.1 Function

The “!message” (SCP) command is used to enter a system control program (i.e. guest operating system) priority command on the hercules console. The command to be issued has to be prefixed with an exclamation point '!'.

This command is similar to the “.reply” command. It is up to the operating system to differentiate between them.

8.1.2 Syntax

Descriptive

```
!prio_msg
```

Diagram

```
►►── !prio message ──────────────────────────────────────────►◄
```

8.1.3 Parameter

prio_msg The priority message that will be routed to the system control program running under the Hercules Emulator.

8.1.4 Examples

For a similar sample please see the “.reply” command.

8.2 # (Silent comment)

8.2.1 Function

The hash ("#") provides a convenient way of entering comments into the Hercules log file only with no other effect (“silent” comment). The comment is not processed in any way other than to echo it in the log file.

8.2.2 Syntax

Descriptive

 # anytext

Diagram

```
►►── # ── anytext ──────────────────────────────────────────►◄
```

8.2.3 Parameter

anytext This is the text that has to be written only to the Hercules log file (“silent” comment).

8.2.4 Examples

Example 1:

Write a text string to the log file only (“silent” comment).

```
# A comment that is written to the Hercules log file only
```

Figure 9: Silent comment

8.3 $LOCATE (Display and verify Hercules control blocks)

8.3.1 Function

The $LOCATE command displays and verifies various Hercules control blocks. The specified control block is displayed in a hex dump format.

8.3.2 Syntax

Descriptive

$LOCATE *controlblock*

Diagram

```
►►── $LOCATE ── controlblock ──────────────────────────────────►◄
```

8.3.3 Parameter

controlblock This specifies the control block to be verified and displayed. Currently the following control blocks are supported through the LOCATE command:

- sysblk (System Configuration Block)
- regs (CPU Register Context)
- hostinfo (Host System Information Block)

8.3.4 Examples

Example 1:

Locate the Hercules sysblk control block. Please note that not all hex columns can be displayed in the figure below. Missing columns have been marked with "∫∫".

```
HHC00013I Herc command: '$locate sysblk'
HHC90000D DBG: SYSBLK @ 0x002333C0 - Verified
HHC90000D DBG: sysblk+0x0000  53595342  ∫∫  002333C0  SYSBLK          4.00    .....#3.
HHC90000D DBG: sysblk+0x0020  00009DE0  ∫∫  00000000  ...../.......... ........k..L....
HHC90000D DBG: sysblk+0x0040  00000000  ∫∫  00000000  ................ ................
HHC90000D DBG: sysblk+0x0060  00000000  ∫∫  60140000  ..........;..... ..`pc.......`...
HHC90000D DBG: sysblk+0x0080  00000000  ∫∫  04000000  ........H....... ................
HHC90000D DBG: sysblk+0x00a0  04000000  ∫∫  387CA600  ........dnoC.{.. xtvM.|..xtvM8|..
HHC90000D DBG: sysblk+0x00c0  7874764D  ∫∫  307DA600  xtvMp|..xtvM.|.. xtvM.|..xtvM0}..
HHC90000D DBG: sysblk+0x00e0  7874764D  ∫∫  B00D0000  xtvMh}..xtvM.}.. ........x.......
```

```
HHC90000D DBG: sysblk+0x0100  00000000  ∫∫  508D6002  ................ ........xtvMP.`.
HHC90000D DBG: sysblk+0x0120  FC190000  ∫∫  00000000  .....g...g...g.. .g..............
HHC90000D DBG: sysblk+0x0140  00000000  ∫∫  00000000  ................ ................
HHC90000D DBG: sysblk+0x0160  00000000  ∫∫  00000000  ................ ................
HHC90000D DBG: sysblk+0x0180  00000000  ∫∫  00000000  ................ ................
HHC90000D DBG: sysblk+0x01a0  00000000  ∫∫  00000000  ................ ................
HHC90000D DBG: sysblk+0x01c0  80000000  ∫∫  00000000  ................ ................
HHC90000D DBG: sysblk+0x01e0  00008000  ∫∫  00000000  ................ ................
```

Figure 10: $LOCATE command (System Configuration Block)

Example 2:

Locate the Hercules regs control block. Please note that not all hex columns can be displayed in the figure below. Missing columns have been marked with "∫∫".

```
HHC00013I Herc command: '$locate regs'
HHC90000D DBG: REGS[00] @ 0x02BF6788 - Verified
HHC90000D DBG: regs+0x0000  52454753  ∫∫  00000000  REGS_CP00        4.00    ......g.
HHC90000D DBG: regs+0x0020  00009720  ∫∫  00000000  ... .3#......... K...............
HHC90000D DBG: regs+0x0040  00000000  ∫∫  00000000  ................ ................
HHC90000D DBG: regs+0x0060  00000000  ∫∫  00000000  ................ ................
HHC90000D DBG: regs+0x0080  AA8AE302  ∫∫  00000000  .........p...... ................
HHC90000D DBG: regs+0x00a0  00000000  ∫∫  00000000  ................ ................
HHC90000D DBG: regs+0x00c0  00000000  ∫∫  00000000  ................ ................
HHC90000D DBG: regs+0x00e0  00000000  ∫∫  00000000  ................ ................
HHC90000D DBG: regs+0x0100  00000000  ∫∫  00000000  ................ ................
HHC90000D DBG: regs+0x0120  00000000  ∫∫  00000000  ........@....... .|..............
HHC90000D DBG: regs+0x0140  00000000  ∫∫  00000000  ................ ................
HHC90000D DBG: regs+0x0160  00000000  ∫∫  00000000  ................ ................
HHC90000D DBG: regs+0x0180  00000000  ∫∫  00000000  ................ ................
HHC90000D DBG: regs+0x01a0  10BBFE00  ∫∫  00000000  ................ ................
HHC90000D DBG: regs+0x01c0  00000000  ∫∫  00000000  ................ ................
HHC90000D DBG: regs+0x01e0  00000000  ∫∫  00000000  ................ ................
```

Figure 11: $LOCATE command (CPU Register Context)

Example 3:

Locate the Hercules hostinfo control block. Please note that not all hex columns can be displayed in the figure below. Missing columns have been marked with "∫∫".

```
HHC00013I Herc command: '$locate hostinfo'
HHC90000D DBG: HOSTINFO @ 0x0036AE00 - Verified
HHC90000D DBG: sysname          = Windows
HHC90000D DBG: nodename         = GOOFY
HHC90000D DBG: release          = 6.1.7600
HHC90000D DBG: version          = 7 Ultimate Edition 64-bit
HHC90000D DBG: machine          = Intel(R) x64
HHC90000D DBG: cpu_brand        = Intel(R) Core(TM) i7 CPU       Q 720  @ 1.60GHz
```

```
HHC90000D DBG: trycritsec_avail  = YES
HHC90000D DBG:
HHC90000D DBG: num_procs         = 8
HHC90000D DBG: num_packages      = 1
HHC90000D DBG: num_physical_cpu  = 4
HHC90000D DBG: num_logical_cpu   = 8
HHC90000D DBG: bus_speed         =   0Khz
HHC90000D DBG: cpu_speed         =   1.6Ghz
HHC90000D DBG: vector_unit       = YES
HHC90000D DBG: fp_unit           = YES
HHC90000D DBG: cpu_64bits        = YES
HHC90000D DBG: cpu_aes_extns     = NO
HHC90000D DBG:
HHC90000D DBG: valid_cache_nums  = YES
HHC90000D DBG: cachelinesz       = 64
HHC90000D DBG: L1Dcachesz        =   32KiB
HHC90000D DBG: L1Icachesz        =   32KiB
HHC90000D DBG: L2cachesz         =  256KiB
HHC90000D DBG: L3cachesz         =    6MiB
HHC90000D DBG:
HHC90000D DBG: hostpagesz        =    4KiB
HHC90000D DBG: AllocGran         =   64KiB
HHC90000D DBG: TotalPhys         =    7.9263GiB
HHC90000D DBG: AvailPhys         =    6.1006GiB
HHC90000D DBG: TotalPageFile     =   15.9245GiB
HHC90000D DBG: AvailPageFile     =   13.9831GiB
HHC90000D DBG: TotalVirtual      =    1.9999GiB
HHC90000D DBG: AvailVirtual      =    1.8433GiB
```

Figure 12: $LOCATE command (Host System Information Block)

8.4 $TEST (Custom test command)

8.4.1 Function

The $TEST command is used for invoking a custom test command. It performs whatever function the user specifically coded it to do.

WARNING: Do not use this command unless you specifically coded the function that this command invokes. Unless you wrote this command you probably don't know what it is actually doing. It could perform any function, up to crashing Hercules!

8.4.2 Syntax

Descriptive

$TEST *req_parms* [*opt_parms*]

Diagram

```
>>── $TEST ── req_parms ─┬─────────────┬──────────────><
                         └─ opt_parms ─┘
```

8.4.3 Parameter

req_parms Any required parameter(s) for the specific implementation of the $TEST command.

opt_parms Any optional parameter(s) for the specific implementation of the $TEST command.

8.4.4 Examples

Example 1:

Invoke the Hercules test command. This implementation of $TEST does not require any parameter.

```
HHC00013I Herc command: '$test'
*** $test command: creating threads...
*** $test thread 00002590: sleeping for 2 seconds...
*** $test thread 00002594: sleeping for 4 seconds...
*** $test thread 000036BC: sleeping for 5 seconds...
*** $test thread 0000223C: sleeping for 1 seconds...
*** $test thread 000010C4: sleeping for 3 seconds...
*** $test thread 00002590: 2 second sleep done; rc=0
*** $test thread 00003070: sleeping for 3 seconds...
```

```
*** $test thread 0000223C: 1 second sleep done; rc=0
*** $test thread 00002AD8: sleeping for 6 seconds...
*** $test thread 00002594: 4 second sleep done; rc=0
*** $test thread 000028EC: sleeping for 4 seconds...
*** $test thread 000010C4: 3 second sleep done; rc=0
*** $test thread 00002F2C: sleeping for 2 seconds...
*** $test thread 00002494: sleeping for 1 seconds...
*** $test thread 00003070: 3 second sleep done; rc=0
*** $test command: waiting for threads to exit...
*** $test thread 00002494: 1 second sleep done; rc=0
*** $test thread 000036BC: 5 second sleep done; rc=0
*** $test thread 00002F2C: 2 second sleep done; rc=0
*** $test thread 000028EC: 4 second sleep done; rc=0
*** $test thread 00002AD8: 6 second sleep done; rc=0
*** $test command: test complete.
```

Figure 13: $TEST command

8.5 $ZAPCMD (Enable or disable system parameters and console commands)

8.5.1 Function

The $ZAPCMD console command allows it to enable or disable a given command name as a system parameter (configuration statement) or console command.

This is a console command used for debugging and requires that the debug command level has been set. This is the default for debug builds, but not for normal production builds.

To activate the $ZAPCMD console command for normal non-debug production release builds, use the following command sequence:

```
MSGLVL VERBOSE    (optional)
MSGLVL DEBUG      (optional)
CMDLVL DEBUG      (required)
$ZAPCMD cmdname option
```

Note: It is possible to disable the $ZAPCMD console command itself, so be careful!

8.5.2 Syntax

Descriptive

```
$ZAPCMD cmdname [CFG | CMD | NOCFG | NOCMD]
```

Diagram

```
►►── $ZAPCMD ── cmdname ──┬── CFG ───┬──────────────────►◄
                          ├── NOCFG ─┤
                          ├── CMD ───┤
                          └── NOCMD ─┘
```

8.5.3 Parameter

cmdname Specifies the command name of the system parameter (configuration statement) or the console command that has to be enabled or disabled.

CFG Enable the given command name as a system parameter (configuration statement).

NOCFG Disable the given command name as a system parameter (configuration statement).

CMD Enable the given command name as a console command.

NOCMD Disable the given command name as a console command.

8.5.4 Examples

Example 1:

Disable ARCHLVL as a console command.

```
HHC00013I Herc command: 'archlvl s/370'
HHC01603I $zapcmd archlvl nocmd
HHC01603I archlvl s/370
HHC01600E Unknown command archlvl, enter 'help' for a list of valid commands
```

Figure 14: $ZAPCMD command (disable console command)

8.6 * (Loud comment)

8.6.1 Function

The asterisk ("*") provides a convenient way of entering comments into the Hercules log file and to the Hercules console with no other effect (“loud” comment). The comment is not processed in any way other than to echo it in the log file and at the console.

8.6.2 Syntax

Descriptive

* *anytext*

Diagram

```
►►── * ── anytext ──────────────────────────────────────────►◄
```

8.6.3 Parameter

anytext This is the text that has to be written to the Hercules log file and displayed on the Hercules console (“loud” comment).

8.6.4 Examples

Example 1:

Write a text string to the log file and the console (“loud” comment).

```
* A comment that is written to the Hercules log file and displayed on the Hercules console
```

Figure 15: Loud comment

8.7 .reply (SCP command)

8.7.1 Function

The ".reply" (SCP) command is used to give a reply to a system control program (i.e. the guest operating system running under Hercules) message. To reply to a SCP message that gets issued to the Hercules console, prefix the reply with a period (dot).

8.7.2 Syntax

Descriptive

.reply

Diagram

```
►►── .reply ──────────────────────────────────────────────►◄
```

8.7.3 Parameter

reply The reply that will be routed to the system control program running under the Hercules Emulator.

8.7.4 Examples

Example 1:

Reply to message IGGN504A ("r 00,0148").

```
*   IGGN504A SPECIFY UNIT FOR CATALOG.MVS.MASTER ON MVSRES
  .r 00,0148
  HHCCP041I SYSCONS interface active
   IEE600I REPLY TO 00 IS;0148
```

Figure 16: SCP command

8.8 ? (List all commands / command specific help)

8.8.1 Function

The “?” command is an alias for the “help” command. Without any options it will display a sorted list of all commands matching the current command level with a short description. It is possible to specify a partial command name followed by an asterisk (“*”) to get a list of all commands matching the partial command name. For example ‘? msg*’ will list all commands beginning with ‘msg’ and matching the current command level.

If the list is enterd with a full command name it will display a long form of help information associated with that command if the command is available for the current command level. The list provided by the command without options shows with an asterisk in front of the description if there is additional information available.

The displayed help text may be limited to explaining the general format of the command and its various required or optional parameters and is not meant to replace the appropriate manual.

8.8.2 Syntax

Descriptive

? [*command* | *cmd**]

Diagram

```
►►── ? ───┬────────────┬──────────────────────────────────────────►◄
          ├─ command ──┤
          └── cmd* ────┘
```

8.8.3 Parameter

command	The Hercules console command to which additional information is desired.
cmd*	Provides a list of all commands beginning with the partial command name ‘cmd’ and matching the current command level.

8.8.4 Examples

Example 1:

List all valid console commands matching the current command level.

```
HHC00013I Herc command: '?'
HHC01603I
```

```
HHC01602I Command      Description
HHC01602I -------      ---------------------------------------------
HHC01602I !message    *SCP priority messsage
HHC01602I #            Silent comment
HHC01602I *            Loud comment
HHC01602I .reply      *SCP command
HHC01602I ?            alias for help
HHC01602I aea          Display AEA tables
HHC01602I aia          Display AIA fields
HHC01602I alrf         Command deprecated: Use "archlvl"  instead
HHC01602I ar           Display access registers
HHC01602I archlvl     *Set Architecture Level
HHC01602I archmode     Alias for archlvl
HHC01602I asn_and_l    Command deprecated: Use "archlvl" instead
HHC01602I attach      *Configure device
HHC01602I auto_scsi   *Command deprecated - Use "SCSIMOUNT"
HHC01602I autoinit    *Display/Set automatic create empty tape file switch
HHC01602I automount   *Display/Update allowable tape automount directories
HHC01602I b           *Set breakpoint
HHC01602I b+           Set breakpoint
HHC01602I b-          *Delete breakpoint
.
several lines not displayed
.
HHC01602I s{+/-}dev    Turn CCW stepping on/off
HHC01602I t           *Instruction trace
HHC01602I t+          *Instruction trace on
HHC01602I t-           Turn off instruction tracing
HHC01602I t?          *Instruction trace query
HHC01602I timerint    *Display or set timers update interval
HHC01602I tlb          Display TLB tables
HHC01602I toddrag      Display or set TOD clock drag factor
HHC01602I todprio      Set/Display todprio parameter
HHC01602I traceopt    *Instruction trace display options
HHC01602I tt32        *Control/query CTCI-WIN functionality
HHC01602I tzoffset     Set tzoffset parameter
HHC01602I t{+/-}CKD    Turn CKD_KEY tracing on/off
HHC01602I t{+/-}dev    Turn CCW tracing on/off
HHC01602I u            Disassemble storage
HHC01602I uptime       Display how long Hercules has been running
HHC01602I v           *Display or alter virtual storage
HHC01602I version      Display version information
HHC01602I xpndsize    *Define/Display xpndsize parameter
HHC01602I yroffset     Set yroffset parameter
HHC01603I
HHC01610I  (*)  More help available.
```

Figure 17: “?” command

Example 2:

Display a list of commands beginning with ‘cpu’ and matching the current command level.

```
HHC00013I Herc command: '? cpu*'
HHC01603I
HHC01602I Command      Description
HHC01602I -------      -----------------------------------------------
HHC01602I cpu         *Define target cpu for panel display and commands
HHC01602I cpuidfmt     Set format 0/1 STIDP generation
HHC01602I cpumodel     Set CPU model number
HHC01602I cpuprio      Set/Display cpuprio parameter
HHC01602I cpuserial    Set CPU serial number
HHC01602I cpuverid     Set CPU verion number
HHC01603I
HHC01610I  (*)  More help available.
```

Figure 18: “? CPU*” command

Example 3:

Display additional help text for the MAINSIZE command.

```
HHC00013I Herc command: '? mainsize'
HHC01603I
HHC01602I Command      Description
HHC01602I -------      -----------------------------------------------
HHC01602I mainsize    *Define/Display mainsize parameter
HHC01603I
HHC01603I Format: mainsize [ mmmm | nnnS [ lOCK | unlOCK ] ]
HHC01603I         mmmm    - define main storage size mmmm Megabytes
HHC01603I
HHC01603I         nnnS    - define main storage size nnn S where S is the multiplier
HHC01603I                   B = no multiplier
HHC01603I                   K = 2**10 (kilo/kibi)
HHC01603I                   M = 2**20 (mega/mebi)
HHC01603I                   G = 2**30 (giga/gibi)
HHC01603I                   T = 2**40 (tera/tebi)
HHC01603I                   P = 2**50 (peta/pebi)
HHC01603I                   E = 2**60 (exa/exbi)
HHC01603I
HHC01603I         lOCK    - attempt to lock storage (pages lock by host OS)
HHC01603I         unlOCK  - leave storage unlocked (pagable by host OS)
HHC01603I
HHC01603I       (none)    - display current mainsize value
HHC01603I
HHC01603I  Note: Multipliers 'T', 'P', and 'E' are not available on 32bit machines
```

Figure 19: “? MAINSIZE” command

8.9 ABS (Display or alter absolute storage)

8.9.1 Function

The ABS command allows you to display or alter absolute storage. Up to 64K of absolute storage can be displayed, up to 32 bytes of absolute storage can be altered.

8.9.2 Syntax

Descriptive

ABS {*addr* | *addr.length* | *addr-addr* | *addr=value*}

Diagram

```
►►── ABS ──┬───── addr ───────┬────────────────────────────────►◄
           ├── addr.length ───┤
           ├── addr-addr ─────┤
           └── addr=value ────┘
```

8.9.3 Parameter

addr Specifies the address of the absolute storage that is to be displayed. If *addr* is given without a length or without a second address for the end of the storage area then 64 bytes of absolute storage are displayed.

addr.length Specifies the address of the absolute storage area that is to be displayed with starting address and length (from address *addr* with the length of *length*). The *length* value must be given in hexadecimal. The maximum length that can be specified is 64K.

addr-addr Specifies an address range with start and end address (from begin address to end address) of the absolute storage area that is to be displayed.

addr=value Specifies the address of the absolute storage area that is to be altered. *value* is a hex-string of up to 32 pairs of hex digits (32 bytes) which will be written to the absolute storage address given by the *addr* parameter. After altering the storage, 16 bytes of absolute storage starting at *addr* are displayed.

8.9.4 Examples

Example 1:

to be done.

```
HHC00013I Herc command: 'abs 00000000.ff'
.
.
.
```

Figure 20: ABS command (display absolute storage)

Example 2:

to be done.

```
HHC00013I Herc command: 'abs 00000000=FFFFFFFF'
.
.
.
```

Figure 21: ABS command (alter absolute storage)

8.10 AEA (Display AEA absolute-effective-address tables)

8.10.1 Function

Display the Hercules AEA tables. The AEA table is an address lookup accelerator which saves absolute data addresses for further lookups to eliminate the calls to logical_to_main.This shortens the translation path length.

8.10.2 Syntax

Descriptive

```
AEA
```

Diagram

```
►►── AEA ──────────────────────────────────────────────────────────►◄
```

8.10.3 Parameter

None.

8.10.4 Examples

Example 1:

Display the AEA tables

```
HHC00013I Herc command: 'aea'
HHC02282I aea mode   DAT-Off
HHC02282I aea ar     0D 07 01 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
HHC02282I aea common             0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
HHC02282I aea cr[1]  0000000000000000
HHC02282I     cr[7]  0000000000000000
HHC02282I     cr[13] 0000000000000000
HHC02282I     cr[r]  00000000FFFFFFFF
HHC02282I     alb[0] 0000000000000000
HHC02282I     alb[1] 0000000000000000
HHC02282I     alb[2] 0000000000000000
HHC02282I     alb[3] 0000000000000000
HHC02282I     alb[4] 0000000000000000
HHC02282I     alb[5] 0000000000000000
HHC02282I     alb[6] 0000000000000000
HHC02282I     alb[7] 0000000000000000
```

```
HHC02282I     alb[8] 0000000000000000
HHC02282I     alb[9] 0000000000000000
HHC02282I     alb[10] 0000000000000000
HHC02282I     alb[11] 0000000000000000
HHC02282I     alb[12] 0000000000000000
HHC02282I     alb[13] 0000000000000000
HHC02282I     alb[14] 0000000000000000
HHC02282I     alb[15] 0000000000000000
```

Figure 22: AEA command

8.11 AIA (List AIA absolute-instruction-address fields)

8.11.1 Function

Display the Hercules AIA fields. The AIA fields are instruction fetch accelerators which save absolute instruction addresses for further lookups to shorten the translation path length.

8.11.2 Syntax

Descriptive

```
AIA
```

Diagram

```
►►── AIA ──────────────────────────────────────────────────────────────►◄
```

8.11.3 Parameter

None.

8.11.4 Examples

Example 1:

Display the AIA fields.

```
HHC00013I Herc command: 'aia'
HHC02283I AIV 000000000002f800 aip 0000000003AB0800 ip 0000000003AB0AAA aie 0000000000000000
          aim 0000000003A9F000
```

Figure 23: AIA command

8.12 AR (Display access registers)

8.12.1 Function

Display the current contents of the access registers.

8.12.2 Syntax

Descriptive

```
AR
```

Diagram

```
►►── AR ──────────────────────────────────────────────────────────────►◄
```

8.12.3 Parameter

None.

8.12.4 Examples

Example 1:

Display the access registers.

```
HHC00013I Herc command: 'ar'
HHC02272I Access registers
HHC02272I CP00: AR00=00000000 AR01=00000000 AR02=0000062A AR03=87465380
HHC02272I CP00: AR04=87461860 AR05=07F51080 AR06=87465190 AR07=00000000
HHC02272I CP00: AR08=8746531C AR09=070D0060 AR10=00000000 AR11=00000000
HHC02272I CP00: AR12=00000000 AR13=00000000 AR14=00000000 AR15=00000000
```

Figure 24: AR command

8.13 ARCHLVL (Set architecture level)

8.13.1 Function

The ARCHLVL console command is used to display or set the architecture mode. Given without any parameters the ARCHLVL command displays the current setting of the architecture mode.

Additional parameters may be specified to set a new value for the architecture mode, to enable or disable specific facilities, to override the STFLE (Store Facility List Extended) response and force it to return a certain (incorrect) bit pattern or to display the status (enabled or disabled) of facilities. An overview of all architecture facilities can be found in “Appendix C. Architecture Facilities” on page 643.

8.13.2 Syntax

Descriptive

ARCHLVL [S/370 | ESA/390 | ESAME | z/ARCH]

or

ARCHLVL {ENABLE | DISABLE} *facility* [S/370 | ESA/390 | ESAME | z/ARCH]

or

ARCHLVL QUERY [*facility* | ALL]

or

ARCHLVL {ENABLE | DISABLE} *bitno* [S/370 | ESA/390 | ESAME | z/ARCH]

Diagram

```
►►── ARCHLVL ──┬───────────────┬─────────────────────────────────────────►◄
               ├── S/370 ──────┤
               ├── ESA/390 ────┤
               ├── ESAME ──────┤
               └── z/ARCH ─────┘
```

or

```
►►── ARCHLVL ──┬── ENABLE ───┬── facility ──┬───────────────┬────────►◄
               └── DISABLE ──┘              ├── S/370 ──────┤
                                            ├── ESA/390 ────┤
                                            ├── ESAME ──────┤
                                            └── z/ARCH ─────┘
```

or

```
              ┌──── ALL ──────┐
►►── ARCHLVL ── QUERY ──┼───────────────┼─────────────────────►◄
              └── facility ───┘

  or

►►── ARCHLVL ──┬── ENABLE ──┬── bitno ──┬──────────────┬────────►◄
               └── DISABLE ─┘           ├── S/370 ─────┤
                                        ├── ESA/390 ───┤
                                        ├── ESAME ─────┤
                                        └── z/ARCH ────┘
```

8.13.3 Parameter

S/370 Use S/370 for OS/360, VM/370 and MVS 3.8.

ESA/390 Use ESA/390 for MVS/XA, MVS/ESA, OS/390, VM/ESA, VSE/ESA, Linux/390 and ZZSA. zOS can be run until version 1.2 with ESA/390 mode without installed bimodal feature or until version 1.4 if the bimodal feature is installed.

ESAME Use ESAME (Enterprise System Architecture, Modal Extensions) for z/OS, z/VM, z/VSE and z/Linux. The ESAME mode is equivalent to z/Archtecture mode at at Architecture Level 2.

When ESAME is specified, the machine will always be IPL'ed in ESA/390 mode but the system is capable of being switched into the z/Architecture mode after IPL. This is handled automatically by all z/Architecture operating systems.

z/ARCH Use z/ARCH for z/OS, z/VM, z/VSE and z/Linux. z/ARCH is similar to ESAME. The z/Arch mode is equivalent to z/Architecture mode at Architecture Level 3.

ENABLE Enable the specified facility. If no architecture mode is given as additional parameter the facility is enabled for all architecture modes.

DISABLE Disable the specified facility. If no architecture mode is given as additional parameter the facility is disabled for all architecture modes.

facility The name of the facility that has to be enabled, disabled or displayed. An overview of all architecture facilities can be found in "Appendix C. Architecture Facilities" on page 643.

bitno The bit number of the facility that has to be enabled or disabled. The format of *bitno* is BIT*nn*, e.g. "BIT44" (Bit 44 = PFPO Facility Bit). An overview of all architecture facilities can be found in "Appendix C. Architecture Facilities" on page 643.

QUERY Display the facility settings.

ALL Specifies that all facilities have to be displayed.

8.13.4 Examples

Example 1:

Display the current setting of the architecture mode.

```
HHC00013I Herc command: 'archlvl'
HHC02203I Value 'archmode': 'S/370'
```

Figure 25: ARCHLVL command (display architecture mode)

Example 2:

Set the architecture level to z/ARCH.

```
HHC00013I Herc command: 'archlvl z/arch'
```

Figure 26: ARCHLVL command (set new architecture mode)

Example 3:

Disable the ‘DECIMAL_FLOAT’ facility in z/Architecture mode.

```
HHC00013I Herc command: 'archlvl disable decimal_float z/arch'
HHC00898I Facility(DECIMAL_FLOAT) disabled for archmode z/Arch
```

Figure 27: ARCHLVL command (enable facility)

Example 4:

Force the enabling of the PFPO feature (bit number 44) for all architecture modes.

```
HHC00013I Herc command: 'archlvl enable bit44'
HHC00898I Facility(44) enabled for archmode S/370
HHC00898I Facility(44) enabled for archmode ESA/390
HHC00898I Facility(44) enabled for archmode z/Arch
```

Figure 28: ARCHLVL command (force on facility)

Example 5:

Query all facilities.

```
HHC00013I Herc command: 'archlvl query all'
HHC00890I Facility(N3                  ) Enabled
HHC00890I Facility(ESAME_INSTALLED     ) Enabled
HHC00890I Facility(ESAME_ACTIVE        ) Enabled
HHC00890I Facility(IDTE_INSTALLED      ) Enabled
```

```
HHC00890I Facility(IDTE_SC_SEGTAB      ) Disabled
HHC00890I Facility(IDTE_SC_REGTAB      ) Disabled
HHC00890I Facility(ASN_LX_REUSE        ) Disabled
HHC00890I Facility(STFL_EXTENDED       ) Enabled
HHC00890I Facility(ENHANCED_DAT        ) Disabled
HHC00890I Facility(SENSE_RUN_STATUS    ) Enabled
HHC00890I Facility(CONDITIONAL_SSKE    ) Enabled
HHC00890I Facility(CONFIG_TOPOLOGY     ) Enabled
HHC00890I Facility(IPTE_RANGE          ) Disabled
HHC00890I Facility(NONQ_KEY_SET        ) Disabled
HHC00890I Facility(TRAN_FAC2           ) Enabled
HHC00890I Facility(MSG_SECURITY        ) Enabled
HHC00890I Facility(LONG_DISPL_INST     ) Enabled
HHC00890I Facility(LONG_DISPL_HPERF    ) Enabled
HHC00890I Facility(HFP_MULT_ADD_SUB    ) Enabled
.
.
.
HHC00890I Facility(ETF3_ENHANCEMENT    ) Enabled
HHC00890I Facility(EXTRACT_CPU_TIME    ) Enabled
HHC00890I Facility(CSSF                ) Enabled
HHC00890I Facility(CSSF2               ) Enabled
HHC00890I Facility(GEN_INST_EXTN       ) Enabled
HHC00890I Facility(EXECUTE_EXTN        ) Enabled
HHC00890I Facility(ENH_MONITOR         ) Disabled
HHC00890I Facility(RESERVED_39         ) Disabled
HHC00890I Facility(SET_PROG_PARAM      ) Disabled
HHC00890I Facility(FPS_ENHANCEMENT     ) Enabled
HHC00890I Facility(DECIMAL_FLOAT       ) Enabled
HHC00890I Facility(DFP_HPERF           ) Enabled
HHC00890I Facility(PFPO                ) Disabled
HHC00890I Facility(FAST_BCR_SERIAL     ) Disabled
HHC00890I Facility(CMPSC_ENH           ) Disabled
HHC00890I Facility(RES_REF_BITS_MUL    ) Disabled
HHC00890I Facility(CPU_MEAS_COUNTER    ) Disabled
HHC00890I Facility(CPU_MEAS_SAMPLNG    ) Disabled
HHC00890I Facility(MSA_EXTENSION_3     ) Disabled
HHC00890I Facility(MSA_EXTENSION_4     ) Disabled
HHC00890I Facility(MOVE_INVERSE        ) Enabled
```

Figure 29: ARCHLVL command (query all)

Example 6:

Query the ‘MOVE_INVERSE’ facility.

```
HHC00013I Herc command: 'archlvl query move_inverse'
HHC00890I Facility(MOVE_INVERSE        ) Enabled
```

Figure 30: ARCHLVL command (query facility)

8.14 ARCHMODE (Set architecture mode)

8.14.1 Function

The ARCHMODE command has been deprecated. This command was used to specify the initial architecture mode.

ARCHMODE is still accepted and is treated as an alias for the ARCHLVL console command. However in the future the ARCHLVL command should be used instead. Please see ARCHLVL for details.

8.14.2 Syntax

See ARCHLVL console command.

8.14.3 Parameter

See ARCHLVL console command.

8.14.4 Examples

See ARCHLVL console command.

8.15 ATTACH (Configure device)

8.15.1 Function

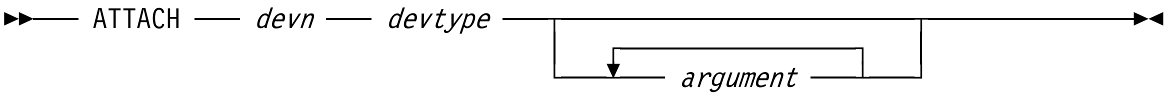
The ATTACH command configures a new device and makes it available in the active configuration. The effect of this command is immediately visible in the device display of the Hercules Windows GUI or the Hercules "Peripherals" display if the GUI is not used.

8.15.2 Syntax

Descriptive

ATTACH *devn type* [*argument* [*argument* ...]]

Diagram



8.15.3 Parameter

devn The device number of the device that is to be configured in the current configuration.

type The device type that is to be configured in the current configuration. For valid device types see chapter 6 (Device Definition Descriptions).

argument These are the additional arguments for the device that is to be configured in the current configuration. The arguments are dependent on the device type. For additional information on valid arguments see chapter 6 (Device Definition Descriptions).

8.15.4 Examples

Example 1:

Attach a printer device to the active configuration.

```
HHC00013I Herc command: 'attach 000f 1403 d:/mvs/prt/prt2.txt crlf'
HHC02198I Device attached
```

Figure 31: ATTACH command (configure printer)

Example 2:

Attach a display terminal to the active configuration.

```
HHC00013I Herc command ‘attach 0700 3270’
HHC02198I Device attached
```

Figure 32: ATTACH command (configure display terminal)

8.16 AUTO_SCSI_MOUNT (Automatic SCSI tape mounts)

8.16.1 Function

The AUTO_SCSI_MOUNT panel command has been deprecated and is replaced by SCSIMOUNT. See SCSIMOUNT panel command for details.

8.16.2 Syntax

See SCSIMOUNT panel command.

8.16.3 Parameter

See SCSIMOUNT panel command.

8.16.4 Examples

See SCSIMOUNT panel command.

8.17 AUTOINIT (Display or set the automatic creation of empty tape files)

8.17.1 Function

The AUTOINIT console command controls the automatic creation of empty tape files. AUTOINIT handles the "file not found" condition for a specified tape file. It controls if DEVINIT returns a "file not found" error or creates an empty tape file if the tape file could not be found. Given without an argument AUTOINIT displays the current setting.

8.17.2 Syntax

Descriptive

```
AUTOINIT [ON | OFF]
```

Diagram

```
►►── AUTOINIT ──┬──────────┬──────────────────────────────────────────►◄
                ├── ON ────┤
                └── OFF ───┘
```

8.17.3 Parameter

ON When AUTOINIT is set ON, DEVINIT will initialize a blank, non-labeled tape if the specified tape file is not found. Next, DEVINIT writes two tapemarks, rewinds the tape and positions the tape to the beginning.

OFF When AUTOINIT is set OFF (which is the default), DEVINIT will return a "file not found" error if the specified tape file is not found.

8.17.4 Examples

Example 1:

Display the current AUTOINIT setting.

```
HHC00013I Herc command: 'autoinit'
HHC02203I Value 'autoinit': 'off'
```

Figure 33: AUTOINIT command (display current setting)

Example 2:

Set the automatic creation of empty tape files to ‘ON’.

```
23:05:25.148 00000ABC HHC00013I Herc command: 'autoinit on'
23:05:25.148 00000ABC HHC02204I Value 'autoinit' set to 'on'
```

Figure 34: AUTOINIT command (change setting)

8.18 AUTOMOUNT (Display or update allowable tape automount directories)

8.18.1 Function

The AUTOMOUNT command allows it to add or delete entries from the list of allowable/unallowable tape automount directories, or lists all currently defined list entries, if any.

The format of the *directory* operand for add/del operations is identical to that as described in the documentation for the AUTOMOUNT system parameter (i.e. prefix with '+' or '-' as needed).

The automount feature is appropriately enabled or disabled for all tape devices as needed depending on the updated empty/non-empty list state.

8.18.2 Syntax

Descriptive

```
AUTOMOUNT {ADD directory | DEL directory | LIST}
   or
AUTOMOUNT [ + | - ]directory
```

Diagram

```
             ┌── + ──┐
►►── AUTOMOUNT ──┼───────┼── directory ──────────────────────────►◄
             └── - ──┘
or
►►── AUTOMOUNT ──┬── ADD directory ──┬───────────────────────────►◄
                 └── DEL directory ──┘
```

8.18.3 Parameter

ADD or +	Add an entry to the list of allowable tape automount directories.
DEL or -	Delete an entry from the list of allowable tape automount directories.
LIST	List all currently defined list entries of allowable tape automount directories.
directory	Specifies the host system directory where the guest is allowed or not to automatically load virtual tape volumes from.

8.18.4 Examples

Example 1:

Add entries to the list of allowable tape automount directories.

```
HHC00013I Herc command: 'automount add D:/MVS/Tape'
HHC02203I default allowed automount directory: 'D:/MVS/Tape/'
HHC00013I Herc command: 'automount add D:/MVS/Cart'
HHC02203I allowed automount directory: 'D:/MVS/Cart/'
```

Figure 35: AUTOMOUNT command (add entries)

Example 2:

List all currently defined list entries of allowable tape automount directories.

```
HHC00013I Herc command: 'automount list'
HHC02217I '+D:/MVS/Tape/'
HHC02217I '+D:/MVS/Cart/'
```

Figure 36: AUTOMOUNT command (list entries)

Example 3:

Delete an entry from the list of allowable tape automount directories.

```
HHC00013I Herc command: 'automount del D:/MVS/Cart'
HHC02220I Entry deleted
```

Figure 37: AUTOMOUNT command (delete entry)

8.19 B (Set breakpoint)

8.19.1 Function

The B (or B+) command sets a breakpoint. The argument gives an instruction address or a range of addresses where you wish to halt the execution. Once the breakpoint is reached, instruction execution is temporarily halted and the next instruction to be executed is displayed. You may then examine registers and / or storage etc. To continue execution after reaching a breakpoint, enter the “G” command.

The “B” command is synonymous with the “S+” command.

8.19.2 Syntax

<u>Descriptive</u>

B {*addr* | *addr-addr*}

<u>Diagram</u>

```
►►── B ──┬──── addr ─────┬──────────────────────────────────►◄
         └── addr-addr ──┘
```

8.19.3 Parameter

addr Instruction address where instruction execution is to be stopped.

addr-addr Range of instruction addresses where instruction execution is to be stopped.

8.19.4 Examples

Example 1:

Set a breakpoint at instruction address 01000000.

```
HHC00013I Herc command: 'b+ 01000000'
HHC02229I Instruction break on range 1000000-1000000
```

Figure 38: B command

8.20 B+ (Set breakpoint)

8.20.1 Function

The B+ (or B) command sets a breakpoint. The argument gives an instruction address or a range of addresses where you wish to halt the execution. Once the breakpoint is reached, instruction execution is temporarily halted and the next instruction to be executed is displayed. You may then examine registers and / or storage etc. To continue execution after reaching a breakpoint, enter the “G” command.

The “B+” command is synonymous with the “S+” command.

8.20.2 Syntax

Descriptive

B+ {*addr* | *addr-addr*}

Diagram

```
►►── B+ ──┬──── addr ────┬──────────────────────────────────────►◄
          └── addr-addr ─┘
```

8.20.3 Parameter

addr	Instruction address where instruction execution is to be stopped.
addr-addr	Range of instruction addresses where instruction execution is to be stopped.

8.20.4 Examples

Example 1:

Set a breakpoint at instruction address 01000000.

```
HHC00013I Herc command: 'b+ 01000000'
HHC02229I Instruction break on range 1000000-1000000
```

Figure 39: B+ command

8.21 B- (Delete breakpoint)

8.21.1 Function

The B- command removes any previously set breakpoint.

8.21.2 Syntax

Descriptive

```
B-
```

Diagram

```
►►—— B- ————————————————————————————————————►◄
```

8.21.3 Parameter

None.

8.21.4 Examples

Example 1:

Delete the previously set breakpoint.

```
HHC00013I Herc command: 'b-'
HHC02229I Instruction break off range 1000000-1000000
```

Figure 40: B- command

8.22 CACHE (Execute cache related commands)

8.22.1 Function

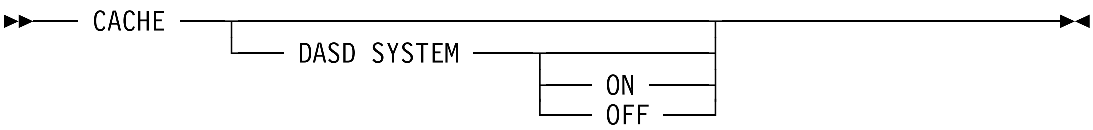
The CACHE command is used to execute various cache related commands. It allows to switch on or off the caching for the DASD system and to display the status of the system DASD caching. Given without an argument the CACHE command displays the cache statistics (see also CACHESTATS command).

8.22.2 Syntax

Descriptive

```
CACHE [DASD SYSTEM [ON | OFF]]
```

Diagram



8.22.3 Parameter

DASD SYSTEM Displays the status of the DASD system caching if given without an additional argument, otherwise enables or disables the caching for all DASD devices.

ON Enables the caching for all DASD devices.

OFF Disables the caching for all DASD devices.

8.22.4 Examples

Example 1:

Display the status of the DASD system caching.

```
HHC00013I Herc command: 'cache dasd system'
HHC02203I dasd system cache: on
```

Figure 41: CACHE command (display status of DASD system caching)

Example 2:

Disable caching for all DASD devices.

```
HHC00013I Herc command: 'cache dasd system off'
HHC02204I dasd system cache set to off
```

Figure 42: CACHE command (disable caching for DASD devices)

Example 3:

Display the current cache statistics.

```
HHC00013I Herc command: 'cache'
HHC02294I Cache...........          0
HHC02294I nbr ............        229
HHC02294I busy ...........          0
HHC02294I busy% ..........          0
HHC02294I empty ..........          0
HHC02294I waiters ........          0
HHC02294I waits ..........          0
HHC02294I buf size ........   2799616
HHC02294I hits ...........       1415
HHC02294I fast hits .......      1115
HHC02294I misses ..........     56004
HHC02294I hit% ...........          2
HHC02294I age ............      29617
HHC02294I last adjusted ... none
HHC02294I last wait ....... none
HHC02294I adjustments .....         0
HHC02294I Cache...........          1
HHC02294I nbr ............       1031
HHC02294I busy ...........         34
HHC02294I busy% ..........          3
HHC02294I empty ..........        776
HHC02294I waiters ........          0
.
several lines not displayed
.
HHC02294I hit% ...........         97
HHC02294I age ............      13261
HHC02294I last adjusted ... none
HHC02294I last wait ....... none
HHC02294I adjustments .....         0
HHC02294I Cache[2] ....... not created
HHC02294I Cache[3] ....... not created
HHC02294I Cache[4] ....... not created
HHC02294I Cache[5] ....... not created
HHC02294I Cache[6] ....... not created
HHC02294I Cache[7] ....... not created
```

Figure 43: CACHE command (display cache statistics)

8.23 CACHESTATS (Display cache statistics)

8.23.1 Function

The CACHESTATS command displays the current Hercules cache statistics.

8.23.2 Syntax

Descriptive

```
CACHESTATS
```

Diagram

```
►►── CACHESTATS ─────────────────────────────────────────────►◄
```

8.23.3 Parameter

None.

8.23.4 Examples

Example 1:

Display the current cache statistics.

```
HHC00013I Herc command: 'cachestats'
HHC02294I Cache...........          0
HHC02294I nbr ............        229
HHC02294I busy ...........          0
HHC02294I busy% ..........          0
HHC02294I empty ..........          0
HHC02294I waiters ........          0
HHC02294I waits ..........         39
HHC02294I buf size ........    2987008
HHC02294I hits ...........       2014
HHC02294I fast hits .......       1039
HHC02294I misses ..........       4056
HHC02294I hit% ...........         33
HHC02294I age ............       4569
HHC02294I last adjusted ... none
HHC02294I last wait ....... none
```

```
HHC02294I adjustments .....          0
HHC02294I Cache...........           1
HHC02294I nbr ............        1031
HHC02294I busy ...........          34
HHC02294I busy% ..........           3
HHC02294I empty ..........         887
HHC02294I waiters ........           0
HHC02294I waits ..........           0
HHC02294I buf size .......      294912
HHC02294I hits ...........        2706
HHC02294I fast hits ......        2436
HHC02294I misses .........         145
HHC02294I hit% ...........          94
HHC02294I age ............        2851
HHC02294I last adjusted ... none
HHC02294I last wait ....... none
HHC02294I adjustments .....          0
HHC02294I Cache[2] ....... not created
HHC02294I Cache[3] ....... not created
HHC02294I Cache[4] ....... not created
HHC02294I Cache[5] ....... not created
HHC02294I Cache[6] ....... not created
HHC02294I Cache[7] ....... not created
```

Figure 44: CACHESTATS command

8.24 CAPPING (Display or set CPU capping value)

8.24.1 Function

The CAPPING command displays or sets the CPU capping value. If *mips* is greater than zero then the CPUs are capped to this value. If *mips* is equal to zero or “OFF” then the capping is disabled.

Only CPUs of type CP are capped. CPUs of type IL, AP or IP are never capped. The CPU string on the Hercules device and status panel which shows the CPU usage turns from white to red during the time the CPU is capped.

8.24.2 Syntax

Descriptive

CAPPING [*mips* | OFF | 0]

Diagram

```
►►── CAPPING ──┬────────────┬──────────────────────────────────────►◄
               ├── mips ────┤
               ├── OFF ─────┤
               └──── 0 ─────┘
```

8.24.3 Parameter

mips Maximum total number of MIPS for all the ‘CP’ type processors.

OFF Disables the CPU capping.

0 This is the same as OFF.

8.24.4 Examples

Example 1:

Display the current CPU capping value (capping set).

```
HHC00013I Herc command: 'capping'
HHC00832I Central processors will be capped at 10 MIPS
```

Figure 45: CAPPING command (display current CPU capping value, capping set)

Example 2:

Set the CPU capping value to 25 MIPS.

```
HHC00013I Herc command: 'capping 25'
HHC00877I Central processors are capped at 25 MIPS
```

Figure 46: CAPPING command (set CPU capping value)

Example 3:

Disable the CPU capping.

```
HHC00013I Herc command: 'capping off'
```

Figure 47: CAPPING command (disable capping)

Example 4:

Display the current CPU capping value (capping not set).

```
HHC00013I Herc command: 'capping'
HHC00838I Capping is not enabled
```

Figure 48: **CAPPING command (display current CPU capping value, capping not set)**

8.25 CCKD (CCKD command)

8.25.1 Function

The CCKD console command can be used to affect CCKD processing. The CCKD console command supports the same options as the CCKD system parameter.

8.25.2 Syntax

Descriptive

CCKD [HELP | STATS | OPTS | *option=value* [,*option=value* ...]]

   where option can be:

[COMP={-1 | *n*}]

[,COMPPARM={-1 | *n*}]

[,RA={2 | *n*}]

[,RAQ={4 | *n*}]

[,RAT={2 | *n*}]

[,WR={2 | *n*}]

[,GCINT={5 | *n*}]

[,GCPARM={0 | *n*}]

[,NOSTRESS={0 | 1}]

[,FREEPEND={-1 | *n*}]

[,FSYNC={0 | 1}]

[,TRACE={0 | *n*}]

[,LINUXNULL={0 | 1}]

[,GCSTART={0 | 1}]

Diagram

```
                    ┌──────────── , ─────────┐
►►── CCKD ──┬──────▼┤ option=value ├─────────┴─┬──────────────────────────►◄
            ├────────────── HELP ──────────────┤
            ├────────────── STATS ─────────────┤
            └────────────── OPTS ──────────────┘
```

```
where option can be:

├── COMP= ──────┬── -1 ──┬───────────────────────────────────────┤
                └─── n ──┘

├── COMPPARM= ──┬── -1 ──┬───────────────────────────────────────┤
                └─── n ──┘

├── RA= ────────┬── 2 ───┬───────────────────────────────────────┤
                └── n ───┘

├── RAQ= ───────┬── 4 ───┬───────────────────────────────────────┤
                └── n ───┘

├── RAT= ───────┬── 2 ───┬───────────────────────────────────────┤
                └── n ───┘

├── WR= ────────┬── 2 ───┬───────────────────────────────────────┤
                └── n ───┘

├── GCINT= ─────┬── 5 ───┬───────────────────────────────────────┤
                └── n ───┘

├── GCPARM= ────┬── 0 ───┬───────────────────────────────────────┤
                └── n ───┘

├── NOSTRESS= ──┬── 0 ───┬───────────────────────────────────────┤
                └── 1 ───┘

├── FREEPEND= ──┬── -1 ──┬───────────────────────────────────────┤
                └─── n ──┘

├── FSYNC= ─────┬── 0 ───┬───────────────────────────────────────┤
                └── 1 ───┘

├── TRACE= ─────┬── 0 ───┬───────────────────────────────────────┤
                └── n ───┘

├── LINUXNULL= ─┬── 0 ───┬───────────────────────────────────────┤
                └── 1 ───┘

├── GCSTART= ───┬── 0 ───┬───────────────────────────────────────┤
                └── 1 ───┘
```

8.25.3 Parameter

HELP Display the CCKD help information.

STATS Display the current CCKD statistics.

OPTS Display the current CCKD options.

option Set a CCKD option. Multiple options may be specified, separated by commas with no intervening blanks.

The CCKD options are:

COMP=*n* Specifies the compression type to be used. This overrides the compression used for all CCKD files. The default (-1) means don't override the compression. Valid compression types are:

-1 Default

0 None

1 Zlib

2 Bzip2

COMPPARM=*n* Overrides the compression parameter. A higher value generally means more compression at the expense of CPU and/or storage. The default (-1) means don't override the compression parameter. The value of *n* can be from -1 and 9.

RA=*n* Sets the Number of read ahead threads. When sequential track or block group access is detected, some number (RAT=n) of tracks or block groups are queued (RAQ=n) to be read by one of the read ahead threads. The default is 2, the value of *n* can be a number from 1 to 9.

RAQ=*n* Sets the size of the read ahead queue. When sequential track or block group access is detected, some number (RAT=n) of tracks or block groups are queued in the read ahead queue. The default is 4, the value of *n* can be a number from 0 to 16. A value of zero disables read ahead.

RAT=*n* Sets the number of tracks or block groups to read ahead when sequential track or block group access is detected. The default is 2, the value of ratn can be a number from 0 to 16. A value of zero disables read ahead.

WR=*n* Sets the number of writer threads. When the cache is flushed, updated cache entries are marked write pending and a writer thread is signalled. The writer thread compresses the track or block group and writes the compressed image to the emulation file.

A writer thread is CPU-intensive while compressing the track or block group and I/O-intensive while writing the compressed image. The writer thread runs one nicer than the CPU thread(s). The default is 2, a value from 1 to 9 can be specified

GCINT=*n* This is the number of seconds the garbage collector thread waits during an interval. At the end of an interval the garbage collector performs space recovery, flushes the cache and optionally 'fsyncs' the emulation file.

However, the file will not be 'fsynced' unless at least 5 seconds have elapsed since the last synchronization (FSYNC). The default is 10 seconds. You can specify a number between 1 and 60.

GCPARM=*n* A value affecting the amount of data moved during the garbage collectors space recovery routine. The garbage collector determines an amount of space to move based on the ratio of free space to used space in an emulation file and on the number of free spacesin the file. The garbage collector wants to reduce the free space to used space ratio and the number of free spaces.

The value is logarithmic; a value of 8 means moving 2^8 the selected value while a negative value similarly decreases the amount to be moved. Normally, 256K will be moved for a file in an interval. Specifying a value of 8 can increase the amount to 64M. At least 64K will be moved. Specifying a large value (such as 8) may not increase the garbage collection efficiency correspondingly. The default is 0. You can specify a number from -8 to 8.

NOSTRESS=*n* Indicates whether stress writes will occur or not. A track or block group may be written under stress when a high percentage of the cache is pending write or when a device I/O thread is waiting for a cache entry. When a stressed write occurs, the compression algorithm and/or compression parm may be relaxed, resulting in faster compression but usually a larger compressed image.

If NOSTRESS is set to one, then a stressed situation is ignored. You would typically set this value to one when you want create the smallest emulation file possible in exchange for a possible performance degradation. The default is 0. You can specify 0 (enable stressed writes) or 1 (disable stressed writes).

FREEPEND=*n* Specifies the free pending value for freed space. When a track or block group image is written, the space it previously occupied is freed. This space will not be available for future allocations until *n* garbage collection intervals have completed. In the event of a catastrophic failure, previously written track or block group images should be recoverable if the current image has not yet been written to the physical disk.

By default the value is set to -1 which means that if FSYNC is specified then the value is 1 otherwise it is 2. If 0 is specified then freed space is immediately available for new allocations. The default is -1. You can specify a number from -1 to 4.

FSYNC=*n* Enables or disables FSYNC. When FSYNC is enabled then the disk emulation file is synchronized with the physical hard disk at the end of a garbage collection interval (no more often than 5 seconds though).

This means that if FREEPEND is non-zero and a catastrophic error occurs, the emulated disks should be recovered coherently. However, FSYNC may cause performance degradation depending on the host operating system and / or the host operating system level. The default is 0 (fsync disabled), you can specify 0 (disable FSYNC) or 1 (enable FSYNC).

TRACE=*n* Specifies the number of CCKD trace entries. You would normally specify a non-zero value when debugging or capturing a problem in CCKD code. When the problem occurs, you should enter the "k" Hercules console command which will print the trace table entries. The default is 0. You can specify a number between 0 and 200000. Each trace entry represents 128 bytes. Normally, for debugging, it is recommended to use 100000.

LINUXNULL=*n* If set to 1 then tracks written to 3390 CCKD volumes that were initialized with the *-linux* option will be checked if they are null (that is if all 12 4096 byte user records contain zeroes). This is used by the DASDCOPY utility. The default is 0.

GCSTART=*n* If set to 1 then space recovery will become active on any emulated disks that have free space. Normally space recovery will ignore emulated disks until they have been updated. The default is 0.

Notes:

- *raq* should be at least as large as *ra*. Read ahead threads are scheduled from entries in the read ahead queue. Likewise *rat* should not exceed *raq* because only *raq* tracks or block groups can be queued at any time.

- The number of writer threads *wr* should usually be 1 more than the number of host processors. This is because one writer thread could be CPU-bound (compressing a track or block-group image) and the other could be i/o-bound (writing the compressed image).

- The garbage collection interval governs the maximum time in seconds an updated track or block group image will reside in storage before being written to the emulation file. A large value may mean more data loss if a catastrophic error occurs. A small value may mean that more CPU time is spent compressing images.

  For example, suppose that a particular image is updated several times each second. If the interval is changed from the default 5 seconds to 1 second, then that image will be compressed and written 5 times more frequently. A large value may cause more cache flushes within a garbage collection interval. These kinds of flushes mean that a read will wait because there are no available cache entries, slowing the emulated operating system. A large value will also cause more pending free space to build up (since free space is flushed each interval). This may mean that the garbage collector space recovery routine will perform more work and the resulting emulation file may be larger.
- Specify *fsync=1* and *gcint=5* if you are seriously concerned about your data being lost due to a failure. *fsync* will ensure your data on disk is coherent. However, fsync may cause a noticeable performance degradation. Note that an fsync will not be performed more often than every 5 seconds.

8.25.4 Examples

Example 1:

Display the CCKD options panel.

```
HHC00013I Herc command: 'cckd opts'
HHC00346I cckd opts: comp=-1,compparm=-1,ra=2,raq=4,rat=2,wr=2,gcint=10
HHC00346I            gcparm=0,nostress=0,freepend=-1,fsync=0,linuxnull=0,trace=0
```

Figure 49: CCKD OPTS command

Example 2:

Display the CCKD help panel.

```
HHC00013I Herc command: 'cckd help'
HHC00345I Command parameters for cckd:
HHC00345I   help          Display help message
HHC00345I   stats         Display cckd statistics
```

```
HHC00345I   opts          Display cckd options
HHC00345I   comp=<n>      Override compression                (-1 ... 2)
HHC00345I   compparm=<n>  Override compression parm           (-1 ... 9)
HHC00345I   ra=<n>        Set number readahead threads        ( 1 ... 9)
HHC00345I   raq=<n>       Set readahead queue size            ( 0 .. 16)
HHC00345I   rat=<n>       Set number tracks to read ahead     ( 0 .. 16)
HHC00345I   wr=<n>        Set number writer threads           ( 1 ... 9)
HHC00345I   gcint=<n>     Set garbage collector interval (sec) ( 1 .. 60)
HHC00345I   gcparm=<n>    Set garbage collector parameter     (-8 ... 8)
HHC00345I   gcstart       Start garbage collector
HHC00345I   nostress=<n>  1=Disable stress writes
HHC00345I   freepend=<n>  Set free pending cycles             (-1 ... 4)
HHC00345I   fsync=<n>     1=Enable fsync
HHC00345I   linuxnull=<n> 1=Check for null linux tracks
HHC00345I   trace=<n>     Set trace table size           ( 0 ... 200000)
```

Figure 50: CCKD HELP command

Example 3:

Display the CCKD statistics panel.

```
HHC00013I Herc command: 'cckd stats'
HHC00347I cckd stats:
HHC00347I   reads....      2726 Kbytes...     10997
HHC00347I   writes...    141831 Kbytes...    179004
HHC00347I   readaheads      751 misses...        65
HHC00347I   syncios..      1149 misses...      1352
HHC00347I   switches.      3908 l2 reads.       322 strs wrt.       914
HHC00347I   cachehits      1929 misses...      1979
HHC00347I   l2 hits..     15250 misses...       322
HHC00347I   waits...........    i/o......       158 cache....         0
HHC00347I   garbage collector   moves....       984 Kbytes...    173612
```

Figure 51: CCKD STATS command

Example 4:

Set a CCKD parameter.

```
HHC00013I Herc command: 'cckd gcint=30'
HHC00346I cckd opts: comp=-1,compparm=-1,ra=2,raq=4,rat=2,wr=2,gcint=30
HHC00346I            gcparm=0,nostress=0,freepend=-1,fsync=0,linuxnull=0,trace=0
```

Figure 52: CCKD command (set options)

8.26 CD (Change directory)

8.26.1 Function

The CD command changes the current directory.

8.26.2 Syntax

Descriptive

CD *path*

Diagram

```
►►── CD ── path ──────────────────────────────────────────►◄
```

8.26.3 Parameter

path This specifies the new path. The path can be specified absolute ("D:\S390\DASD") or relative to the current path ("DASD").

8.26.4 Examples

Example 1:

Change the current directory (absolute path).

```
HHC00013I Herc command: 'cd D:\MVS'
HHC02204I working directory set to D:\MVS
```

Figure 53: CD command (absolute path)

Example 2:

Change the current directory (relative path).

```
HHC00013I Herc command: 'cd DASD'
HHC02204I working directory set to D:\MVS\DASD
```

Figure 54: CD command (relative path)

8.27 CF (Configure current CPU online or offline)

8.27.1 Function

The CF command is used to configure a CPU online or offline. The number of the CPU to be taken online or offline has to be specified first with the CPU command (refer to the CPU command for further information). If the CF command is issued without a parameter, the actual status of the CPU is displayed.

Use the CFALL command to display the status of all CPUs or to configure all CPUs online or offline.

8.27.2 Syntax

Descriptive

`CF [ON | OFF]`

Diagram

```
►►── CF ──┬─────────────┬──────────────────────────────────────────────►◄
          ├──── ON ─────┤
          └──── OFF ────┘
```

8.27.3 Parameter

ON Place the specified CPU online.

OFF Place the specified CPU offline.

8.27.4 Examples

Example 1:

Display the actual status of CPU 1.

```
HHC00013I Herc command: 'cpu 1'
HHC00013I Herc command: 'cf'
HHC00819I Processor CP01: online
```

Figure 55: CF command (display CPU status)

Example 2:

Configure CPU 1 offline.

```
HHC00013I Herc command: 'cpu 1'
HHC00013I Herc command: 'cf off'
HHC00101I Thread id 000010E0, prio  0, name 'Processor CP01' ended
HHC00820I Processor CP01: offline
```

Figure 56: CF command (configure CPU offline)

Example 3:

Configure CPU 1 online.

```
HHC00013I Herc command: 'cpu 1'
HHC00013I Herc command: 'cf on'
HHC00100I Thread id 000006F8, prio  0, name 'Processor CP01' started
HHC00811I Processor CP01: architecture mode 'S/370'
HHC00819I Processor CP01: online
```

Figure 57: CF command (configure CPU online)

8.28 CFALL (Configure all CPUs online or offline)

8.28.1 Function

The CFALL command is used to place all CPUs online or offline. If the CFALL command is issued without a parameter, the actual status of all CPUs is displayed

8.28.2 Syntax

Descriptive

```
CFALL [ON | OFF]
```

Diagram

```
►►── CFALL ──┬──────────┬──────────────────────────────────────────►◄
             ├── ON ────┤
             └── OFF ───┘
```

8.28.3 Parameter

ON Place all CPUs online.

OFF Place all CPUs offline.

8.28.4 Examples

Example 1:

Display the current status of all CPUs.

```
HHC00013I Herc command: 'cfall'
HHC17007I NumCPU = 04, NumVEC = 00, ReservedCPU = 00, MaxCPU = 04
HHC17008I Avgproc  001% 08; MIPS[   2.44]; SIOS[     2]
HHC17009I PROC CP00 - 003%; MIPS[   0.43]; SIOS[     0]
HHC17009I PROC CP01 - 001%; MIPS[   0.13]; SIOS[     2]
HHC17009I PROC CP02 - 001%; MIPS[   0.10]; SIOS[     0]
HHC17009I PROC CP03 - 000%; MIPS[   0.09]; SIOS[     0]
HHC17010I  - Started        : Stopping        * Stopped
```

Figure 58: CFALL command (display status of all CPUs)

Example 2:

Configure all CPUs offline.

```
HHC00013I Herc command: 'cfall off'
HHC00811I Processor CP00: architecture mode 'ESA/390'
HHC00101I Thread id 00000450, prio 15, name 'Processor CP00' ended
HHC00101I Thread id 0000035C, prio  0, name 'Processor CP01' ended
HHC00101I Thread id 000010C4, prio  0, name 'Processor CP02' ended
HHC00101I Thread id 000013E4, prio  0, name 'Processor CP03' ended
HHC00101I Thread id 00000D78, prio  0, name 'Processor CP04' ended
HHC00101I Thread id 00000510, prio  0, name 'Processor CP05' ended
HHC00101I Thread id 000006F8, prio  0, name 'Processor CP06' ended
HHC00101I Thread id 00000C64, prio  0, name 'Processor CP07' ended
HHC00101I Thread id 0000137C, prio  0, name 'Timer' ended
```

Figure 59: CFALL command (configure all CPUs offline)

Example 3:

Configure all CPUs online.

```
HHC00013I Herc command: 'cfall on'
HHC00100I Thread id 000001DC, prio  0, name 'Processor CP00' started
HHC00811I Processor CP00: architecture mode 'ESA/390'
HHC00100I Thread id 00001370, prio  0, name 'Processor CP01' started
HHC00811I Processor CP01: architecture mode 'ESA/390'
HHC00100I Thread id 00000DA0, prio  0, name 'Processor CP02' started
HHC00811I Processor CP02: architecture mode 'ESA/390'
HHC00100I Thread id 0000132C, prio -20, name 'Timer' started
HHC00100I Thread id 0000091C, prio  0, name 'Processor CP03' started
HHC00811I Processor CP03: architecture mode 'ESA/390'
HHC00100I Thread id 00000680, prio  0, name 'Processor CP04' started
HHC00811I Processor CP04: architecture mode 'ESA/390'
HHC00100I Thread id 000013C4, prio  0, name 'Processor CP05' started
HHC00811I Processor CP05: architecture mode 'ESA/390'
HHC00100I Thread id 000010AC, prio  0, name 'Processor CP06' started
HHC00811I Processor CP06: architecture mode 'ESA/390'
HHC00100I Thread id 00000950, prio  0, name 'Processor CP07' started
HHC00811I Processor CP07: architecture mode 'ESA/390'
```

Figure 60: CFALL command (configure all CPUs online)

8.29 CLOCKS (Display TOD clock and CPU timer)

8.29.1 Function

The CLOCKS command is used to display the actual values of the emulators various internal clocks and timers.

8.29.2 Syntax

Descriptive

```
CLOCKS
```

Diagram

```
►►── CLOCKS ──────────────────────────────────────────────►◄
```

8.29.3 Parameter

None.

8.29.4 Examples

Example 1:

Display the TOD clock and CPU timer.

```
HHC00013I Herc command: 'clocks'
HHC02274I tod = F8ED87F060EA3000    2038.282 00:19:05.843363
HHC02274I h/w = C6B345BF5EEA3000    2010.282 00:19:05.843363
HHC02274I off = 323A423102000000      28.001 00:00:00.000000
HHC02274I ckc = 0000000000000000    1900.001 00:00:00.000000
HHC02274I cpt = not decrementing
HHC02274I itm = FDA3714C                     15:23:28.164060
```

Figure 61: CLOCKS command

8.30 CMDLEVEL (Display or set current command group)

8.30.1 Function

The CMDLEVEL command displays or sets the current command group(s). A plus sign preceding the command group activates the console commands of this group, whereas a preceding minus sign deactivates the commands.

A table showing the affiliation of each console command to the various command groups can be found in ‘Appendix D. Hercules Command Groups’.

Some console commands are always active to keep Hercules operable, independent of the current active command group. These commands are listed under command group 'NONE' in the above mentioned table. Command group ‘NONE’ is the result of a ‘CMDLVL –ALL’ command.

The default command level is set to operator, maintenance, programmer and configuration (which corresponds to the command "CMDLEVEL -ALL +OPER +MAINT +PROG +CONFIG"). Some of the arguments can be abbreviated as shown in the syntax section below.

8.30.2 Syntax

Descriptive

```
CMDLEVEL [{+ | -} {ALL | OPERator | MAINT | PROGrammer | CONFIGuration |
         DEVELoper | DEBUG} [...]]
```

Diagram

```
                     ┌────────────────────────────────────────────┐
►►── CMDLEVEL ───────▼───┬─── + ───┬──┬── ALL ──────────────┬──────┴──────────────►◄
                         └─── - ───┘  ├── OPERator ─────────┤
                                      ├── MAINT ────────────┤
                                      ├── PROGrammer ───────┤
                                      ├── CONFIGuration ────┤
                                      ├── DEVELoper ────────┤
                                      └── DEBUG ────────────┘
```

8.30.3 Parameter

+ The plus sign activates the commands of the following command group.

- The minus sign deactivates the commands of the following command group.

ALL Command group ‘ALL’ contains all Hercules console commands. Specifying command group ‘+ALL’ enables all console commands whereas command group ‘-ALL’ equals to NONE. This disables all console commands with the exception of commands necessary to keep Hercules operable.

OPER The ‘OPERator’ command group activates or deactivates all system operator commands.

MAINT Command group ’MAINT’ activates or deactivates all system maintainer commands.

PROG The ‘PROGrammer’ command group activates or deactivates all systems programmer commands.

CONFIG Command group ‘CONFIGuration’ activates or deactivates all system configuration commands.

DEVEL The ‘DEVELoper’ command group activates or deactivates all system developer commands.

DEBUG Command group ‘DEBUG’ activates or deactivates all debugging activity commands activities.

8.30.4 Examples

Example 1:

Set the command group to OPERATOR and PROGRAMMER.

```
HHC00013I Herc command: 'cmdlevel -all +operator +programmer'
HHC01606I cmdlevel[85] is 'operator programmer '
```

Figure 62: CMDLEVEL command (set command group)

Example 2:

Display the current command group(s).

```
HHC00013I Herc command: 'cmdlvl'
HHC01606I cmdlevel[1F] is 'operator maintenance programmer configuration'
```

Figure 63: CMDLEVEL command (display current command group)

8.31 CMDLVL (Display or set current command group)

8.31.1 Function

CMDLVL is an alias for CMDLEVEL. The CMDLVL command displays or sets the current command group(s). See CMDLEVEL for details.

8.31.2 Syntax

See CMDLEVEL command.

8.31.3 Parameter

See CMDLEVEL command.

8.31.4 Examples

See CMDLEVEL command.

8.32 CMDSEP (Display or set current command line separator)

8.32.1 Function

The CMDSEP command displays or sets the command line separator. The command line separator character is used to separate multiple panel commands on a single line. The default is OFF which means that there is no command line separator defined and therefore multiple panel commands on a single line are not supported.

8.32.2 Syntax

Descriptive

CMDSEP [*char* | OFF]

Diagram

```
►►── CMDSEP ──┬── char ──┬──────────────────────────────────────►◄
              └── OFF ───┘
```

8.32.3 Parameter

char Specifies a single character that is used for command separation. This character must not be the period ('.'), the exclamation mark ('!') or the hyphen ('-'). Although the command line separation character can be set to the number (hash) sign ('#'), this is not recommended because this could affect processing command lines that contain comments.

OFF OFF disables command separation. This is the default.

8.32.4 Examples

Example 1:

Set the current command separator to ';'.

```
HHC00013I Herc command: 'cmdsep'
HHC02203I Value 'cmdsep': 'Not set'
```

Figure 64: CMDSEP command (display the current command line separator)

Example 2:

Set the current command separator to ';'.

```
HHC00013I Herc command: 'cmdsep ;'
HHC02204I Value 'cmdsep' set to ';'
```

Figure 65: CMDSEP command (set new command line separator)

8.33 CMDTGT (Specify the command target)

8.33.1 Function

The CMDTGT command specifies the target environment for all following commands entered on the command line of the Hercules hardware console.

8.33.2 Syntax

Descriptive

`CMDTGT {HERC | SCP | PSCP | ?}`

Diagram

```
►►── CMDTGT ──┬── HERC ──┬──────────────────────────────────►◄
              ├── SCP ───┤
              ├── PSCP ──┤
              └─── ? ────┘
```

8.33.3 Parameter

HERC The command target is the Hercules Emulator. This is also the default, when no CMDTGT target has been given.

SCP The target for the commands is the system control program (i.e. the guest operating system running under Hercules).

PSCP All entered commands are system control program (i.e. guest operating system) priority commands.

? Display the current target environment for commands.

8.33.4 Examples

Example 1:

Display the current target environment for commands.

```
HHC00013I Herc command: 'cmdtgt ?'
HHC02288I Commands are sent to 'herc'
```

Figure 66: CMDTGT command (display current setting)

Example 2:

Set the system control program as the target for commands.

```
HHC00013I Herc command: 'cmdtgt scp'
HHC02288I Commands are sent to 'scp'
```

Figure 67: CMDTGT command (set target to SCP)

Example 3:

Set the command target from SCP or PSCP back to Hercules (see also HERC command).

```
HHC00013I Herc command: 'herc cmdtgt herc'
HHC02288I Commands are sent to 'herc'
```

Figure 68: CMDTGT command (set target back to Hercules)

8.34 CNSLPORT (Display or set console port)

8.34.1 Function

The CNSLPORT command displays or sets the port number (in decimal), on which the telnet server will listen. The parameter of the command may also have the form host:port, where the telnet console server will bind to the specified address.

8.34.2 Syntax

Descriptive

CNSLPORT [*port* | *host:port*]

Diagram

```
►►── CNSLPORT ──┬──────────────────┬──────────────────────────────►◄
                ├────── port ──────┤
                └── host :port ────┘
```

8.34.3 Parameter

host The IP address of the host to which the telnet server will bind to. If an IP address is given then it must be a valid IP address for the host system.

port The port number (decimal) on which the telnet server will listen. The port number must not be in use by any other server. The port number must be in the range of 0 to 65535. Ports below 1024 cannot be used unless Hercules is running as root or is otherwise authorized to use low ports.

8.34.4 Examples

Example 1:

Display the current telnet client port.

```
HHC00013I Herc command: 'cnslport'
HHC17001I Console server listening for host 192.168.1.10 on port 3270
```

Figure 69: CNSLPORT command (display current telnet client port)

Example 2:

Set the port number on which the telnet server will listen to 3270.

```
HHC00013I Herc command: 'cnslport 3270'
```

Figure 70: CNSLPORT command (set telnet client port)

Example 3:

Specify 192.168.1.10 as the IP address of the host to which the telnet server will bind to and set the port number on which the telnet server will listen to 3270.

```
HHC00013I Herc command: 'cnslport 192.168.1.10:3270'
```

Figure 71: CNSLPORT command (set telnet client port bound to specific address)

8.35 CODEPAGE (Display or set codepage conversion table)

8.35.1 Function

The CODEPAGE command displays or sets the codepage conversion table. If an operand is given then the codepage is set to the specified page, if the page is valid. If no operand is specified, the current codepage is displayed.

With the QCODEPAGES panel command a list of all valid codepage mappings can be displayed.

8.35.2 Syntax

Descriptive

CODEPAGE [*codepage* | USER | MAINT *cmd* [*operands*]]

Diagram

```
►►── CODEPAGE ──┬──────────────────────────────────┬──────────────►◄
                ├── codepage ──────────────────────┤
                ├── USER ──────────────────────────┤
                └── MAINT ── cmd ──┬───────────────┤
                                   └── operands ───┘
```

8.35.3 Parameter

DEFAULT “DEFAULT” specifies the traditional Hercules codepage.

codepage Specifies the codepage conversion table used for ASCII / EBCDIC translation. Supported codepage mappings are shown in the table below. Iconv single byte codepages may also be used (e.g. “UTF8/EBCDIC-CP-NL”).

USER This specifies that the user specific codepage conversion tables (see CPUPDT system parameter and console command) have to be activated.

MAINT MAINT and its arguments is the same as the CP_UPDT system parameter. Please see CP_UPDT for details.

Supported codepage mappings:

Mapping	ASCII	EBCDIC
437/037	437 PC United States	037 United States/Canada
437/500	437 PC United States	500 Latin 1
437/1047	437 PC United States	1047 Open Systems Latin 1

Mapping	ASCII	EBCDIC
819/037	819 ISO-8859-1 Latin 1	037 United States/Canada
819/037v2	819 ISO-8859-1 Latin 1	037 United States/Canada SHARE
819/273	819 ISO-8859-1 Latin 1	273 CECP Austria/Germany
819/277	819 ISO-8859-1 Latin 1	277 CECP Denmark/Norway
819/278	819 ISO-8859-1 Latin 1	278 CECP Finland/Sweden
819/280	819 ISO-8859-1 Latin 1	280 CECP Italy
819/284	819 ISO-8859-1 Latin 1	284 CECP Spain
819/285	819 ISO-8859-1 Latin 1	285 CECP United Kingdom
819/297	819 ISO-8859-1 Latin 1	297 CECP France
819/500	819 ISO-8859-1 Latin 1	500 CECP International
819/1047	819 ISO-8859-1 Latin 1	1047 Open Systems Latin 1
850/273	850 PC Latin 1	273 Austria/Germany
850/1047	850 PC Latin 1	1047 Open Systems Latin 1
1252/037	1252 Windows Latin 1	037 United States/Canada
1252/037v2	1252 Windows Latin 1	037 United States/Canada SHARE
1252/1047	1252 Windows Latin 1	1047 Open Systems Latin 1
1252/1140	1252 Windows Latin 1	1140 United States/Canada with Euro sign

Table 18: Supported codepage mappings

8.35.4 Examples

Example 1:

Display the current codepage conversion table.

```
HHC00013I Herc command: 'codepage'
HHC01476I Codepage is 'default'
```

Figure 72: CODEPAGE command (display current codepage conversion table)

Example 2:

Set the codepage conversion table to 437/037.

```
HHC00013I Herc command: 'codepage 437/037'
HHC01474I Using 'internal' codepage conversion table '437/037'
```

Figure 73: CODEPAGE command (set codepage conversion table)

8.36 CONKPALV (Display / alter console TCP/IP keep-alive settings)

8.36.1 Function

The CONKPALV command specifies the tn3270 console and telnet clients keepalive option values that control automatic detection of disconnected tn3270/telnet client sessions.

This is a built-in feature of TCP/IP and allows detection of unresponsive TCP/IP connections and not idle clients. That is to say, your connection will not be terminated after 3 seconds of idle time. Your 3270 session can remain idle for many minutes or hours or days without any data being transmitted. If the TCP/IP stack at the other end of the connection (not your 3270 client itself) fails to respond to the internal keepalive probe packets however, then it means that the TCP/IP stack itself is down or there has been a physical break in the connection.

Thus, even if your 3270 client is completely idle, your system's TCP/IP stack itself should still respond to the keepalive probes sent by the TCP/IP stack at the Hercules end of the link. If it doesn't, then TCP/IP will terminate the tn3270/telnet session which will cause Hercules to disconnect the terminal.

The three values can also be modified on-demand via the conkpalv panel command, which has the exact same syntax. Note that the syntax is very unforgiving: no spaces are allowed anywhere within the parentheses and each value must be separated from the other with a single comma.

Please also note that not all systems support being able to modify all three values. That is, not all values may be able to be changed, and it is system dependent which values you can change and which values you cannot. On Windows for example, the *count* value is ignored and cannot be changed from its default value of 10. Other systems may ignore one or more or all three values and use platform defaults instead. This is entirely system dependent. Check you system's documentation for details regarding which values can be changed and which cannot as well as how to adjust your system's default values.

8.36.2 Syntax

Descriptive

CONKPALV (*idle,intv,count*)

Diagram

```
►►── CONKPALV ── (idle,intv,count) ──────────────────────────►◄
```

8.36.3 Parameter

idle The *idle* value specifies the number of seconds of inactivity until the first keep-alive probe is sent. The default for the *idle* value is 3 seconds.

intv The *intv* value specifies the interval in seconds between the probes if no acknowledgement is received from the previous probe. The default for *intv* is 1 second.

count The *count* value specifies the number of unacknowledged keep-alive packets sent before the connection is considered to have failed. The default value is 9 for non-Windows platforms and 10 for Windows systems.

Note: On Windows platforms the count value is ignored and cannot be changed from its default value of 10.

8.36.4 Examples

Example 1:

Set the TCP/IP keep-alive settings to 5 seconds idle time, 3 seconds interval between the probes and disconnect after 15 consecutive failed probes.

```
HHC00013I Herc command: 'conkpalv (5,3,15)'
```

Figure 74: CONKPALV command

8.37 CP_UPDT (Create or modify user character conversion table)

8.37.1 Function

The CP_UPDT console command creates or modifies the contents of the user codepage tables. The tables can be populated with a 'CP_UPDT REFERENCE' command which copies one of the provided codepage tables to the user tables or by a 'CP_UPDT IMPORT' command which imports a previously created and with 'CP_UPDT EXPORT' exported table.

Changes in the user tables are made through one or more 'CP_UPDT ALTER' commands. These allow for up to 16 modifications at a time. The current contents of the user tables can be shown on the console with 'CP_UPDT DISPLAY'. Finally the changed user tables are activated with a 'CODEPAGE USER' command.

In all commands that require the selection of a user table (EBCDIC or ASCII), the EBCDIC table refers to the 'guest to host' (g2h) translation and the ASCII table refers to the 'host to guest' (h2g) translation.

8.37.2 Syntax

Descriptive

CP_UPDT *command*

where *command* can be:

ALTER {EBCDIC | ASCII | G2H | H2G} (*pos*,*val*[,*pos*,*val*]...)

DISPLAY {EBCDIC | ASCII | G2H | H2G}

EXPORT {EBCDIC | ASCII | G2H | H2G} *filename*

IMPORT {EBCDIC | ASCII | G2H | H2G} *filename*

REFERENCE [*codepage*]

RESET

TEST

Diagram

```
►►── CP_UPDT ──┤ command ├──────────────────────────────────────►◄

   where command can be:

                                        ┌─────── , ─────┐
├── ALTER ────┬── EBCDIC ──┬── ( ──▼──── pos,val ────┴── ) ────────┤
              ├── ASCII ───┤
              ├── G2H ─────┤
              └── H2G ─────┘
```

```
├── DISPLAY ──┬── EBCDIC ──┬──────────────────────────────────────────┤
              ├── ASCII ───┤
              ├── G2H ─────┤
              └── H2G ─────┘

├── EXPORT ──┬── EBCDIC ──┬── filename ────────────────────────────────┤
             ├── ASCII ───┤
             ├── G2H ─────┤
             └── H2G ─────┘

├── IMPORT ──┬── EBCDIC ──┬── filename ────────────────────────────────┤
             ├── ASCII ───┤
             ├── G2H ─────┤
             └── H2G ─────┘

├── REFERENCE ──┬────────────────┬─────────────────────────────────────┤
                └── codepage ────┘

├── RESET ───────────────────────────────────────────────────────────┤

├── TEST ────────────────────────────────────────────────────────────┤
```

8.37.3 Parameter

ALTER Alters the user EBCDIC or ASCII table value at hex position *pos* to hex value *val*. Up to 16 pairs of hex digits may be specified within the parenthesis. ALTER can be abbreviated as ‘ALT’.

DISPLAY Displays the user EBCDIC or ASCII codepage table. DISPLAY can be abbreviated as ‘DIS’ or ‘DSP’.

EXPORT Exports the contents of the user EBCDIC or ASCII codepage table to file *filename*. EXPORT can be abbreviated as ‘EXP’.

IMPORT Imports the contents of file *filename* to the user EBCDIC or ASCII codepage table. IMPORT can be abbreviated as ‘IMP’.

REFERENCE Copies the specified codepage to the user EBCDIC and ASCII tables. If no code-page is specified, a list of valid codepages is displayed on the console. ‘REFE-RENCE’ can be abbreviated as ‘REF’.

RESET Reset the internal user tables to binary zero.

TEST Verify that user tables are transparent, i.e. the value at position n in g2h used as an index into h2g will return a value equal n (g2h<=>h2g, h2g<=>g2h).

EBCDIC The target for the command is the EBCDIC table. The EBCDIC table refers to the ‘guest to host’ translation. ‘EBCDIC’ can be abbreviated as ‘E’.

ASCII The target for the command is the ASCII table. The ASCII table refers to the ‘host to guest’ translation. ‘ASCII’ can be abbreviated as ‘A’.

G2H This is the same as ‘EBCDIC’.

H2G This is the same as ‘ASCII’.

pos Specifies the hex position within the selected table.

val Specifies the hex value for the selected position.

filename Specifies the file name of the file to which the specified codepage has to be exported or from which the codepage table has to be imported.

codepage Specifies the codepage that has to be copied to the user tables.

8.37.4 Examples

Example 1:

Copy the Hercules default codepage to the user tables.

```
HHC00013I Herc command: 'cp_updt reference default'
HHC01478I Codepage 'default' copied to 'user'
```

Figure 75: CP_UPD command (copy codepage to user tables)

Example 2:

Display user ASCII table.

```
HHC00013I Herc command: 'cp_updt display ascii'
HHC01484I Codepage: Displaying user table ascii/h2g
HHC01485I Codepage:    _0_1_2_3 _4_5_6_7 _8_9_A_B _C_D_E_F 0... 4... 8... C...
HHC01486I Codepage: 0_ 00010203 372D2E2F 1605250B 0C0D0E0F .... .... .... .... 0_
HHC01486I Codepage: 1_ 10111213 3C3D3226 18191A27 221D351F .... .... .... .... 1_
HHC01486I Codepage: 2_ 405A7F7B 5B6C507D 4D5D5C4E 6B604B61  !"# $%&' ()*+ ,-./ 2_
HHC01486I Codepage: 3_ F0F1F2F3 F4F5F6F7 F8F97A5E 4C7E6E6F 0123 4567 89:; <=>? 3_
HHC01486I Codepage: 4_ 7CC1C2C3 C4C5C6C7 C8C9D1D2 D3D4D5D6 @ABC DEFG HIJK LMNO 4_
HHC01486I Codepage: 5_ D7D8D9E2 E3E4E5E6 E7E8E9AD E0BD5F6D PQRS TUVW XYZ[ \]^_ 5_
HHC01486I Codepage: 6_ 79818283 84858687 88899192 93949596 `abc defg hijk lmno 6_
HHC01486I Codepage: 7_ 979899A2 A3A4A5A6 A7A8A9C0 6AD0A107 pqrs tuvw xyz{ |}~. 7_
HHC01486I Codepage: 8_ 68DC5142 43444748 52535457 56586367 .... .... .... .... 8_
HHC01486I Codepage: 9_ 719C9ECB CCCDDBDD DFECFCB0 B1B2B3B4 .... .... .... .... 9_
HHC01486I Codepage: A_ 4555CEDE 49690406 AB08BAB8 B7AA8A8B .... .... .... .... A_
HHC01486I Codepage: B_ 090A14BB 15B5B617 1BB91C1E BC20BEBF .... .... .... .... B_
HHC01486I Codepage: C_ 21232428 292A2B2C 3031CA33 343638CF .... .... .... .... C_
HHC01486I Codepage: D_ 393A3B3E 41464A4F 5962DA64 65667072 .... .... .... .... D_
HHC01486I Codepage: E_ 73E17475 76777880 8C8D8EEB 8FEDEEEF .... .... .... .... E_
HHC01486I Codepage: F_ 909A9B9D 9FA0ACAE AFFDFEFB 3FEAFAFF .... .... .... .... F_
```

Figure 76: CP_UPD command (display ASCII user table)

Example 3:

Export the ASCII user table to file ‘ascii.cp’.

```
HHC00013I Herc command: 'cp_updt export ascii d:\mvs\conf\ascii.cp'
HHC01490I Codepage: Exported user table ascii/h2g to file d:\mvs\conf\ascii.cp
```

Figure 77: CP_UPD command (export user table)

Example 4:

Import the ASCII user table from file ‘ascii.cp’.

```
HHC00013I Herc command: 'cp_updt import ascii d:\mvs\conf\ascii.cp'
HHC01490I Codepage: Imported user table ascii/h2g from file d:\mvs\conf\ascii.cp
```

Figure 78: CP_UPD command (import user table)

Example 5:

Alter the ASCII user table as follows, display the changed table and activate it:

- Change position x'5B' to value x'C0' and position x'5D' to value x'D0'
- Change position x'7B' to value x'AD' and position x'7D' to value x'B0'

```
HHC00013I Herc command: 'cp_updt alter ascii (5b,c0,5d,d0,7b,ad,7d,bd)'
HHC01487I Codepage: Altering user table ascii/h2g
HHC01488I Codepage: Pos[5B] was AD is C0
HHC01488I Codepage: Pos[5D] was BD is D0
HHC01488I Codepage: Pos[7B] was C0 is AD
HHC01488I Codepage: Pos[7D] was D0 is BD

HHC00013I Herc command: 'cp_updt display ascii'
HHC01484I Codepage: Displaying user table ascii/h2g
HHC01485I Codepage:    _0_1_2_3 _4_5_6_7 _8_9_A_B _C_D_E_F 0... 4... 8... C...
HHC01486I Codepage: 0_ 00010203 372D2E2F 1605250B 0C0D0E0F .... .... .... .... 0_
HHC01486I Codepage: 1_ 10111213 3C3D3226 18191A27 221D351F .... .... .... .... 1_
HHC01486I Codepage: 2_ 405A7F7B 5B6C507D 4D5D5C4E 6B604B61  !"# $%&' ()*+ ,-./ 2_
HHC01486I Codepage: 3_ F0F1F2F3 F4F5F6F7 F8F97A5E 4C7E6E6F 0123 4567 89:; <=>? 3_
HHC01486I Codepage: 4_ 7CC1C2C3 C4C5C6C7 C8C9D1D2 D3D4D5D6 @ABC DEFG HIJK LMNO 4_
HHC01486I Codepage: 5_ D7D8D9E2 E3E4E5E6 E7E8E9C0 E0D05F6D PQRS TUVW XYZ{ \}^_ 5_
HHC01486I Codepage: 6_ 79818283 84858687 88899192 93949596 `abc defg hijk lmno 6_
HHC01486I Codepage: 7_ 979899A2 A3A4A5A6 A7A8A9AD 6ABDA107 pqrs tuvw xyz[ |]~. 7_
HHC01486I Codepage: 8_ 68DC5142 43444748 52535457 56586367 .... .... .... .... 8_
HHC01486I Codepage: 9_ 719C9ECB CCCDDBDD DFECFCB0 B1B2B3B4 .... .... .... .... 9_
HHC01486I Codepage: A_ 4555CEDE 49690406 AB08BAB8 B7AA8A8B .... .... .... .... A_
HHC01486I Codepage: B_ 090A14BB 15B5B617 1BB91C1E BC20BEBF .... .... .... .... B_
HHC01486I Codepage: C_ 21232428 292A2B2C 3031CA33 343638CF .... .... .... .... C_
HHC01486I Codepage: D_ 393A3B3E 41464A4F 5962DA64 65667072 .... .... .... .... D_
HHC01486I Codepage: E_ 73E17475 76777880 8C8D8EEB 8FEDEEEF .... .... .... .... E_
HHC01486I Codepage: F_ 909A9B9D 9FA0ACAE AFFDFEFB 3FEAFAFF .... .... .... .... F_

HHC00013I Herc command: 'codepage user'
HHC01474I Using 'internal' codepage conversion table 'user'
```

Figure 79: CP_UPD command (alter ASCII user table)

Example 6:

Reset the internal user tables to binary zero.

```
HHC00013I Herc command: 'cp_updt reset'
HHC01479I Codepage 'user' is deleted

HHC00013I Herc command: 'cp_updt display ascii'
HHC01484I Codepage: Displaying user table ascii/h2g, table is invalid
HHC01485I Codepage:    _0_1_2_3 _4_5_6_7 _8_9_A_B _C_D_E_F 0... 4... 8... C...
HHC01486I Codepage: 0_ 00000000 00000000 00000000 00000000 .... .... .... .... 0_
HHC01486I Codepage: 1_ 00000000 00000000 00000000 00000000 .... .... .... .... 1_
HHC01486I Codepage: 2_ 00000000 00000000 00000000 00000000 .... .... .... .... 2_
HHC01486I Codepage: 3_ 00000000 00000000 00000000 00000000 .... .... .... .... 3_
HHC01486I Codepage: 4_ 00000000 00000000 00000000 00000000 .... .... .... .... 4_
HHC01486I Codepage: 5_ 00000000 00000000 00000000 00000000 .... .... .... .... 5_
HHC01486I Codepage: 6_ 00000000 00000000 00000000 00000000 .... .... .... .... 6_
HHC01486I Codepage: 7_ 00000000 00000000 00000000 00000000 .... .... .... .... 7_
HHC01486I Codepage: 8_ 00000000 00000000 00000000 00000000 .... .... .... .... 8_
HHC01486I Codepage: 9_ 00000000 00000000 00000000 00000000 .... .... .... .... 9_
HHC01486I Codepage: A_ 00000000 00000000 00000000 00000000 .... .... .... .... A_
HHC01486I Codepage: B_ 00000000 00000000 00000000 00000000 .... .... .... .... B_
HHC01486I Codepage: C_ 00000000 00000000 00000000 00000000 .... .... .... .... C_
HHC01486I Codepage: D_ 00000000 00000000 00000000 00000000 .... .... .... .... D_
HHC01486I Codepage: E_ 00000000 00000000 00000000 00000000 .... .... .... .... E_
HHC01486I Codepage: F_ 00000000 00000000 00000000 00000000 .... .... .... .... F_
```

Figure 80: CP_UPD command (reset user codepage tables)

Example 7:

Test the internal user tables.

```
HHC00013I Herc command: 'cp_updt test'
HHC01487I Codepage: Testing user table ebcdic/g2h vs. ascii/h2g
HHC01491I Codepage: g2h pos[15] = 0A; h2g pos[0A] = 25
HHC01491I Codepage: g2h pos[AD] = 5B; h2g pos[5B] = C0
HHC01491I Codepage: g2h pos[BD] = 5D; h2g pos[5D] = D0
HHC01491I Codepage: g2h pos[C0] = 7B; h2g pos[7B] = AD
HHC01491I Codepage: g2h pos[D0] = 7D; h2g pos[7D] = BD
HHC01487I Codepage: Testing user table ascii/h2g vs. ebcdic/g2h
HHC01492I Codepage: h2g pos[5B] = C0; g2h pos[C0] = 7B
HHC01492I Codepage: h2g pos[5D] = D0; g2h pos[D0] = 7D
HHC01492I Codepage: h2g pos[7B] = AD; g2h pos[AD] = 5B
HHC01492I Codepage: h2g pos[7D] = BD; g2h pos[BD] = 5D
HHC01492I Codepage: h2g pos[B4] = 15; g2h pos[15] = 0A
```

Figure 81: CP_UPD command (test user codepage tables)

8.38 CPU (Define target CPU for console displays and commands)

8.38.1 Function

The CPU command sets the target CPU for console displays and commands in a Hercules multi-CPU environment. If the Hercules configuration file sets a single CPU environment, then no CPU command is necessary. If the NUMCPU system parameter in the Hercules configuration file is set greater than one and no CPU command is given, then always the first CPU (CPU 0) is the target for console displays and commands.

If a CPU address is given through the CPU command then this specifies the CPU to which all subsequent panel commands will apply to. If an optional command follows the CPU address, the command will execute on this CPU and the target CPU will not be permanently changed.

For example entering command “CPU 0F” followed by command “GPR” will change the target CPU for panel displays and commands and then display the general purpose registers for CPU #15 whereas the command “CPU 0F GPR” will execute the GPR command on CPU #15, but will not change the target CPU for subsequent panel displays and commands.

Please note that the command “*CPU 1 CPU 2*” does no change the target to CPU 2 as the last command (CPU 2) is issued under the temporary CPU 1 environment.

8.38.2 Syntax

Descriptive

CPU *hh* [*cmd*]

Diagram

```
►►── CPU ── hh ──┬─────────┬──────────────────────────►◄
                 └── cmd ──┘
```

8.38.3 Parameter

hh The hexadecimal CPU address of the CPU in the multiprocessor configuration which you wish all console commands to apply to. For example, entering “CPU 0F” followed by a subsequent command “GPR” in a multi- CPU configuration will display the general purpose registers for CPU #15 in the configuration, as opposed to CPU #0. The CPU address is then permanently set to CPU #15, as long as no other CPU command is given.

cmd This is the Hercules command that will execute on CPU *hh.* In this case the CPU address applies only for this command and is not permanently set to CPU *hh*.

8.38.4 Examples

Example 1:

Set the target address for subsequent Hercules commands to CPU #4 in a multiprocessor configuration and display the general purpose registers for this CPU.

```
HHC00013I Herc command: 'cpu 4'
HHC00013I Herc command: 'gpr'
HHC02269I General purpose registers
HHC02269I CP04: GR00=00000000 GR01=062B7000 GR02=0178F500 GR03=040C1E18
HHC02269I CP04: GR04=00000000 GR05=00000021 GR06=00000000 GR07=0146B847
HHC02269I CP04: GR08=00000000 GR09=03EB0980 GR10=00F67400 GR11=04755D78
HHC02269I CP04: GR12=0146A848 GR13=040C0390 GR14=00000000 GR15=00000336
```

Figure 82: CPU command (set target CPU address permanently)

Example 2:

Set the target address temporarily to CPU #2 in a multiprocessor configuration and display the general purpose registers for this CPU.

```
HHC00013I Herc command: 'cpu 2 gpr'
HHC02269I General purpose registers
HHC02269I CP02: GR00=762DDDF8 GR01=00010016 GR02=00F52D28 GR03=0428E294
HHC02269I CP02: GR04=00FACD80 GR05=01175397 GR06=0428EF58 GR07=00F3C4B8
HHC02269I CP02: GR08=0428E268 GR09=81174398 GR10=0175F060 GR11=00FD12C0
HHC02269I CP02: GR12=03FCD790 GR13=0428E390 GR14=8101250C GR15=811744F4
```

Figure 83: CPU command (set target CPU address permanently)

8.39 CPUIDFMT (Display or set format BASIC / 0 / 1 STIDP generation)

8.39.1 Function

The CPUIDFMT command displays or sets the STORE CPU ID (STIDP) format bit. The default STIDP format, if not explicitly set, is 'BASIC'. The format bit of the STIDP information specifies the format of the first two digits of the CPU identification number. When the format bit is ‘0’ then the contents of the CPU identification number identifies the CPU. When the format bit is ‘1’ then the CPU identification number identifies the system configuration as opposed to an individual CPU in the configuration and it identifies the logical partition in which the program is executed.

When the format is ‘BASIC’ the CPU identification number has the following hexadecimal format, where 'A' is the CPU address of the CPU.

- x'Annnnn' (Basic Mode)

When the format is ‘0’ the CPU identification number has the following hexadecimal format where 'L' is a logical CPU address and 'P' is a logical partition identifier.

- x'LPnnnn' (LPAR mode)

When the format is ‘1’ the CPU identification number has the following hexadecimal format where 'PP' is the user partition identifier (UPID). The UPID is an eight bit unsigned binary integer bound to a logical partition.

- x'PPnnnn' (LPAR mode)

In all cases *n* is a digit derived from the serial number of the CPU.

For more information on the STORE CPU ID (STIDP) instruction and the format bit see IBMs "z/Architecture Principles of Operation" manual.

8.39.2 Syntax

Descriptive

```
CPUIDFMT [BASIC | 0 | 1]
```

Diagram

```
►►── CPUIDFMT ──┬──────────────┬──────────────────────────────────────►◄
                ├─── BASIC ────┤
                ├───── 0 ──────┤
                └───── 1 ──────┘
```

8.39.3 Parameter

BASIC Set the format to 'BASIC'. The STIDP format bit is set to ‘0’.

0 Set the format to '0'. The STIDP format bit is set to ‘0’.

1 Set the format to '1'. The STIDP format bit is set to ‘1’.

8.39.4 Examples

Example 1:

Display the current STIDP format.

```
HHC00013I Herc command: 'cpuidfmt'
HHC02203I Value 'CPUIDFMT': '0'
```

Figure 84: CPUIDFMT command (display current STIDP format)

Example 2:

Set STIDP format to '1'.

```
HHC00013I Herc command: 'cpuidfmt 1'
```

Figure 85: CPUIDFMT command (set STIDP format bit)

8.40 CPUMODEL (Display or set CPU model number)

8.40.1 Function

The CPUMODEL command displays or sets the 4 hexadecimal digits CPU model number stored by the STIDP instruction.

8.40.2 Syntax

Descriptive

CPUMODEL [*model*]

Diagram

```
►►── CPUMODEL ─┬───────────┬──────────────────────────────────────►◄
               └── model ──┘
```

8.40.3 Parameter

model Any valid 4 digit hexadecimal CPU model number. A list of the valid model numbers can be found in the Hercules Windows GUI file “cpu-types.txt”.

8.40.4 Examples

Example 1:

Specify a 7060 CPU model.

```
HHC00013I Herc command: 'cpumodel'
HHC02203I cpumodel      : 7060
```

Figure 86: CPUMODEL command

8.41 CPUPRIO (Display or set CPU thread process priority)

8.41.1 Function

The CPUPRIO command is used to change the priority of the CPU thread. See section 5.82 for details on process and thread priorities. On multi-CPU systems a real CPU can be “dedicated” to Hercules by giving the CPU-thread a very high dispatching priority (-20). Given without an argument the CPUPRIO command displays the current CPU thread process priority.

Caution: CPUPRIO should not have a higher dispatching priority than the TOD clock and timer thread.

8.41.2 Syntax

Descriptive

CPUPRIO [*nn*]

Diagram

```
►►── CPUPRIO ──┬──────────┬──────────────────────────────────────────►◄
               └─── nn ───┘
```

8.41.3 Parameter

nn This value specifies the priority for the CPU thread. For details on the priority values see section 5.82 (“Process and Thread Priorities”).

8.41.4 Examples

Example 1:

Set the CPU process priority to -20.

```
HHC00013I Herc command: 'cpuprio -20'
HHC02204I cpuprio        set to -20
```

Figure 87: CPUPRIO command

8.42 CPUSERIAL (Display or set CPU serial number)

8.42.1 Function

The CPUSERIAL command displays or sets the 6 hexadecimal digit CPU serial number stored by the STIDP instruction.

8.42.2 Syntax

Descriptive

`CPUSERIAL` *serial*

Diagram

```
►►── CPUSERIAL ──┬──────────┬──────────────────────────►◄
                 └─ serial ─┘
```

8.42.3 Parameter

serial Any valid 6 digit hexadecimal CPU serial number. In BASIC mode, the high-order digit may be replaced with the processor number when MAXCPU is greater than one. In LPAR mode, the two high-order digits are replaced with either the LPAR number or the CPU number and LPAR number with the full serial number available via the STSI instruction. The default serial number is ‘000001’.

8.42.4 Examples

Example 1:

Set the CPU serial number to 001963.

```
HHC00013I Herc command: 'cpuserial 001963'
HHC02204I cpuserial      set to 001963
```

Figure 88: CPUSERIAL command

8.43 CPUVERID (CPU version code)

8.43.1 Function

The CPUVERID command displays or sets the 2 hexadecimal digit CPU version code stored by the STIDP instruction. The default version code is “FD” when ARCHMODE S/370 or ARCHMODE ESA/390 is specified. For the z/ARCH (or ESAME architecture mode respectively), the version code is always stored as “00” and any value specified here is ignored.

8.43.2 Syntax

Descriptive

CPUVERID *verid*

Diagram

►►── CPUVERID ── *verid* ──────────────────────────────►◄

8.43.3 Parameter

verid	Any valid 2 digit hexadecimal CPU version code. A list of valid version codes can be found in the Hercules Windows GUI file “cpu-types.txt”.

8.43.4 Examples

Example 1:

Set the CPU version code to FD.

```
HHC00013I Herc command: 'cpuverid fd'
HHC02204I cpuverid       set to FD
```

Figure 89: CPUVERID command

8.44 CR (Display or alter control registers)

8.44.1 Function

The CR command displays or alters the actual contents of the control registers.

8.44.2 Syntax

Descriptive

CR [*nn=xxxxxxxx* | *nn=xxxxxxxxxxxxxxxx*]

Diagram

```
►►── CR ──┬──────────────────────────┬──────────────────────►◄
          ├────── nn=xxxxxxxx ───────┤
          └── nn=xxxxxxxxxxxxxxxx ───┘
```

8.44.3 Parameter

nn This specifies the optional register number (00-15) to be altered.

xxxxxxxx This is the register value in hexadecimal (1-8 hex digits for 32-bit registers, 1-16 hex digits for 64-bit registers).

8.44.4 Examples

Example 1:

Display the control registers.

```
HHC00013I Herc command: 'cr'
HHC02271I Control registers
HHC02271I CP00: CR00=5FB1EE40 CR01=3FFFE07F CR02=3F149FC0 CR03=00000001
HHC02271I CP00: CR04=00010001 CR05=3FFE7040 CR06=FE000000 CR07=3FFFE07F
HHC02271I CP00: CR08=00000000 CR09=00000000 CR10=00000000 CR11=00000000
HHC02271I CP00: CR12=01984E80 CR13=3FFFE07F CR14=FF8BF15F CR15=03150010
```

Figure 90: CR command (display control registers)

Example 2:

Alter control register number 15.

```
HHC00013I Herc command: 'cr 15=ffffffff'
HHC02271I Control registers
HHC02271I CP00: CR00=5FB1EE40 CR01=3FFFE07F CR02=3F149FC0 CR03=00000001
HHC02271I CP00: CR04=00010001 CR05=3FFE7040 CR06=FE000000 CR07=3FFFE07F
HHC02271I CP00: CR08=00000000 CR09=00000000 CR10=00000000 CR11=00000000
HHC02271I CP00: CR12=01984E80 CR13=3FFFE07F CR14=FF8BF15F CR15=FFFFFFFF
```

Figure 91: CR command (alter control register)

8.45 CSCRIPT (Cancel a running script thread)

8.45.1 Function

The CSCRIPT command cancels a currently running script or all scripts. If '*' or 'ALL' is given as argument then all running scripts are cancelled. If no argument is given only the first running script is cancelled. To cancel a specific script, then '*id*' can be given as argument, where '*id*' is the ID number of the script to be cancelled. The 'SCRIPT' command may be used to display a list of all currently running scripts. If no script is running, no action is taken.

8.45.2 Syntax

Descriptive

CSCRIPT [* | ALL | *id*]

Diagram

```
►►── CSCRIPT ──┬──────────────┬──────────────────────────────►◄
               ├──── * ───────┤
               ├─── ALL ──────┤
               └─── id ───────┘
```

8.45.3 Parameter

id The thread id of the script to be cancelled.

8.45.4 Examples

Example 1:

Cancel the first running script, given the case that currently five scripts (IDs 2,3,4,5,and 6) are running.

```
HHC00013I Herc command: 'cscript'
HHC02259E Script 2 aborted: 'user cancel request'
HHC00007I Previous message from function script_abort() at script.c[837]
HHC02265I Script 2: file 'D:/MVS/Conf/Script1.rc' aborted due to previous conditions
```

Figure 92: CSCRIPT command (cancel first script)

Example 2:

Cancel the script with the ID number 4 from the remaining four running scripts.

```
HHC00013I Herc command: 'cscript 4'
HHC02259E Script 4 aborted: 'user cancel request'
HHC00007I Previous message from function script_abort() at script.c[837]
HHC02265I Script 4: file 'D:/MVS/Conf/Script3.rc' aborted due to previous conditions
```

Figure 93: CSCRIPT command (cancel script with ID)

Example 3:

Cancel all still running scripts (with IDs 3,5 and 6).

```
HHC00013I Herc command: 'cscript all'
HHC02259E Script 3 aborted: 'user cancel request'
HHC00007I Previous message from function script_abort() at script.c[837]
HHC02265I Script 3: file 'D:/MVS/Conf/Script2.rc' aborted due to previous conditions
HHC02259E Script 5 aborted: 'user cancel request'
HHC00007I Previous message from function script_abort() at script.c[837]
HHC02265I Script 5: file 'D:/MVS/Conf/Script4.rc' aborted due to previous conditions
HHC02259E Script 6 aborted: 'user cancel request'
HHC00007I Previous message from function script_abort() at script.c[837]
HHC02265I Script 6: file 'D:/MVS/Conf/Script5.rc' aborted due to previous conditions
```

Figure 94: CSCRIPT command (cancel all scripts)

8.46 CTC (Enable / disable CTC debugging)

8.46.1 Function

The CTC command enables or disables the debug packet tracing for the specified CTCI / LCS / PTP device group(s) identified by *devnum* or for all CTCI / LCS / PTP device group(s) if *devnum* is not specified or specified as ‘ALL’.

8.46.2 Syntax

Descriptive

CTC DEBUG {ON | OFF} [*devnum* | ALL]

Diagram

```
                                        ┌── ALL ────┐
►►── CTC ── DEBUG ──┬── ON ──┬──────────┼───────────┼──────────────────────►◄
                    └── OFF ─┘          └── devnum ─┘
```

8.46.3 Parameter

ON Enables the debug packet tracing.

OFF Disables the debug packet tracing.

devnum Specifies the CTCI / LCS / PTP device group(s) for which debug packet tracing has to be enabled or disabled.

ALL Enables or disables the debug packet tracing for all CTCI / LCS / PTP device groups. 'ALL' is the default if no device group is specified.

8.46.4 Examples

Example 1:

Enable the debug packet tracing for all CTCI / LCS / PTP devices. Please note that not all hex columns can be displayed in the figure below. Missing columns have been marked with “∫∫”.

```
HHC00013I Herc command: 'ctc debug on all'
HHC02204I CTC debug      set to on ALL
HHC00910I 0:0E20 CTC: sending packet to device 'tun0'
HHC00964I CTC: packet trace: +0000< 4500011C  ∫∫  C0A80063  E....X..@......c ........ .7M{y..
HHC00964I CTC: packet trace: +0010< C0A80001  ∫∫  C6E342A9  ......y.U|....B. {y....`..@..FT.z
```

```
HHC00964I CTC: packet trace: +0020< 54C02001  ʃʃ  31617437  T. ..'......1at7 .{.....x.7.1./..
HHC00964I CTC: packet trace: +0030< 7514C6E1  ʃʃ  01B5DE57  u...'...=.3....W ..F....G...J....
HHC00964I CTC: packet trace: +0040< 4B1821C1  ʃʃ  18C5E477  K.!..P..p......w ...A.&.X.....EU.
HHC00964I CTC: packet trace: +0050< 681A40A1  ʃʃ  42220697  h.@.......Q.B".. .. ~.i...1.j...p
HHC00964I CTC: packet trace: +0060< 47DEA381  ʃʃ  52CAC4B7  G...i.j'Wy&qR... ..ta.{|..`....D.
HHC00964I CTC: packet trace: +0070< 7E38CA61  ʃʃ  0B309ED7  ~8.aBn.Gnf.Q.0.. =../.>w.>."....P
HHC00964I CTC: packet trace: +0080< 250C3541  ʃʃ  393414F7  %.5AD|>g8..194.. .....@...U.....7
HHC00964I CTC: packet trace: +0090< 0A4C6421  ʃʃ  1A25A717  .Ld!.A..D/...%.. .<.....g..]...x.
HHC00964I CTC: packet trace: +00A0< 1DFCD701  ʃʃ  7CC5D537  ....B...N...|..7 ..P..gbx+k~1@EN.
HHC00964I CTC: packet trace: +00B0< 20310DE1  ʃʃ  23451F57   1..t....i..#E.W .....f.G...J....
HHC00964I CTC: packet trace: +00C0< 0F0C88C1  ʃʃ  65440577  ......6.c.w.eD.w ..hA"W.X........
HHC00964I CTC: packet trace: +00D0< 54C2C7A1  ʃʃ  11D30797  T...?...].h..... .BG~.{..)..j.L.p
HHC00964I CTC: packet trace: +00E0< 35974A81  ʃʃ  1172A5B7  5.J._.['.$]q.r.. .p.a^.$...)...v.
HHC00964I CTC: packet trace: +00F0< 7DDD9161  ʃʃ  48135FD7  }..a]swGG..QH._. '.j/).....O...^P
HHC00964I CTC: packet trace: +0100< 6FF91C41  ʃʃ  3715B5F7  o..AK..g.@S17... ?9...[... .....7
HHC00964I CTC: packet trace: +0110< 725D6B21  ʃʃ            r]k!S$C.q.T. .),....g....
HHC00913I 0:0E20 CTC: received 284 bytes size packet from device 'tun0'
HHC00964I CTC: packet trace: +0000> 4500011C  ʃʃ  C0A80001  E.......@.Ih.... .....D.. ...{y..
HHC00964I CTC: packet trace: +0010> C0A80063  ʃʃ  C6E342A9  ...c....U|....B. {y....a..@..FT.z
HHC00964I CTC: packet trace: +0020> 54C02001  ʃʃ  31617437  T. ..'......1at7 .{.....x.7.1./..
HHC00964I CTC: packet trace: +0030> 7514C6E1  ʃʃ  01B5DE57  u...'...=.3....W ..F....G...J....
HHC00964I CTC: packet trace: +0040> 4B1821C1  ʃʃ  18C5E477  K.!..P..p......w ...A.&.X.....EU.
HHC00964I CTC: packet trace: +0050> 681A40A1  ʃʃ  42220697  h.@.......Q.B".. .. ~.i...1.j...p
HHC00964I CTC: packet trace: +0060> 47DEA381  ʃʃ  52CAC4B7  G...i.j'Wy&qR... ..ta.{|..`....D.
HHC00964I CTC: packet trace: +0070> 7E38CA61  ʃʃ  0B309ED7  ~8.aBn.Gnf.Q.0.. =../.>w.>."....P
HHC00964I CTC: packet trace: +0080> 250C3541  ʃʃ  393414F7  %.5AD|>g8..194.. .....@...U.....7
HHC00964I CTC: packet trace: +0090> 0A4C6421  ʃʃ  1A25A717  .Ld!.A..D/...%.. .<.....g..]...x.
HHC00964I CTC: packet trace: +00A0> 1DFCD701  ʃʃ  7CC5D537  ....B...N...|..7 ..P..gbx+k~1@EN.
.
.
.
```

Figure 95: CTC command (enable debug packet tracing)

Example 2:

Disable the debug packet tracing.

```
HHC00013I Herc command: 'ctc debug off'
HHC02204I CTC debug      set to off
```

Figure 96: CTC command (disable debug packet tracing)

8.47 DEFINE (Rename device)

8.47.1 Function

The DEFINE command can be used to “rename” a device (change the device number).

8.47.2 Syntax

Descriptive

DEFINE *olddevice newdevice*

Diagram

►►── DEFINE ── *olddevice* ── *newdevice* ──────────────────────────►◄

8.47.3 Parameter

olddevice The device number of the device that has to be renamed.

newdevice The new device number, to which the device should now be attached.

8.47.4 Examples

Example 1:

Rename two devices.

```
HHC00013I Herc command: 'define 00c5 00c6'
HHC00013I Herc command: 'define 0481 0482'
```

Figure 97: DEFINE command

8.48 DEFSTORE (Display or define main and expanded storage values)

8.48.1 Function

The DEFSTORE command is used to specify the size of the main and expanded storage. Given without an argument the DEFSTORE command displays the current size of the main and expanded storage. If only the type of storage is given, the current storage size of this type is displayed. Storage is allocated in megabytes, unless a specific unit is specified. The actual upper limit of the main and expanded storage is determined by the host system's architecture, operating system, and on some systems the amount of physical memory and paging space you have available.

The practical limit depends on the maximum amount of storage that can be obtained by the "malloc" function (usually around 1 GB on 32-bit platforms; on 64-bit platforms the value should only be limited by available paging space).

When increasing the storage size Hercules attempts to allocate first the new storage. If the new allocation is successful then the previously allocated memory will be freed. This is to prevent a situation where the old memory is freed first, then the new allocation fails and a reallocation of the memory in the previous size also fails because of storage fragmentation and therefore leaving Hercules without memory.

When decreasing the storage the memory will stay allocated in the previous size but the storage size will appear as decreased. Subsequent increases will not reallocate memory unless they go over the already allocated amount.

An additional optional argument determines the locking state of the allocated memory (page lock by host operating system). The LOCKED option indicates that the memory is to be locked into storage while UNLOCKED (the default) indicates that the memory is not locked into the storage.

Please note that Hercules preserves the last locking state of DEFSTORE for each type of storage. Once the storage is locked, any subsequent change to the storage size will honor the existing lock state of memory unless the lock state is specified again on the DEFSTORE command.

Caution: Do not lock storage unless sufficient real memory is available to back up the request. Failure to do so may require the host system to be rebooted.

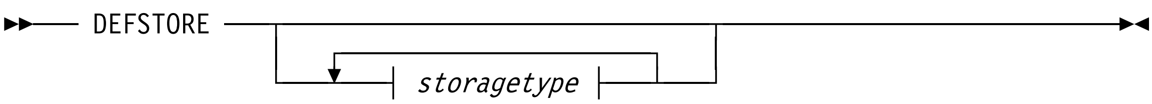
8.48.2 Syntax

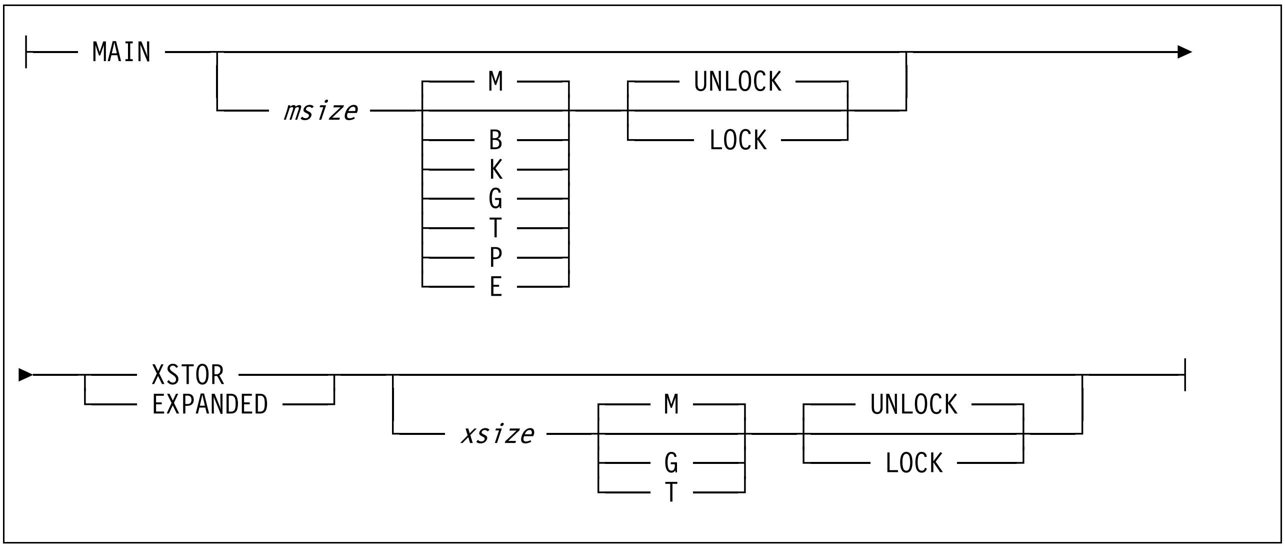
Descriptive

DEFSTORE [MAIN [*msize*[B | K | M | G | T | P |E] [UNLOCK | LOCK]]]

[{XSTOR | EXPANDED} [*xsize*[M | G | T] [UNLOCK | LOCK]]]

Diagram





8.48.3 Parameter

msize The value of *msize* must be a valid decimal number. The actual upper limit is determined by the host system's architecture, the operating system and on some systems the amount of physical memory and paging space that is available.

For storage sizes less than 16M, sizes not on a 4K boundary are rounded up to the next 4K boundary. Otherwise, storage sizes not on a 1M boundary are rounded up to the next 1M boundary.

The minimum size is 4K for architecture levels ALS0 and ALS1 (S/370 and ESA/390), and 8K for architecture level ALS2 (ESAME) and higher. A maximum of 64M may be specified for architecture level ALS0 (S/370), 2048M (2G) for ALS1 (ESA/390) and 16E for architecture level ALS2 (ESAME) and higher.

The default on startup is 2M.

xsize The value of *xsize* must be a valid decimal number. The actual upper limit is determined by the host system's architecture, the operating system and on some systems the amount of physical memory and paging space that is available.

Storage sizes not on a 1M boundary are rounded up to the next 1M boundary. The lower limit and default is 0.

B 'B' determines that the number given is specified in bytes (no multiplier). Specifying the storage in bytes is possible only for main storage.

K 'K' determines that the number given is specified in kilobytes (multiplier 2**10). Specifying the storage in kilobytes is possible only for main storage.

M 'M' determines that the number given is specified in megabytes (multiplier 2**20). This is the default if no unit is appended.

G 'G' determines that the number given is specified in gigabytes (multiplier 2**30).

T 'T' determines that the number given is specified in terabytes (multiplier 2**40). On 32-bit machines the unit terabytes is not available.

P 'P' determines that the number given is specified in petabytes (multiplier 2**50). Specifying the storage in petabytes is possible only for main storage. On 32-bit machines the unit petabytes is not available.

E 'E' determines that the number given is specified in exabytes (multiplier 2**60). Specifying the storage in exabytes is possible only for main storage. On 32-bit machines the unit exabytes is not available.

LOCK Attempt to lock the storage (pages locked by the host operating system).

UNLOCK Leave the store unlocked (no pages locked by the host operating system, pageable by host OS). This is the default.

Notes:

The actual upper limit is determined by the host system's architecture and operating system and the amount of physical memory and available paging space. The total of MAINSIZE and XPNDSIZE on host systems with a 32-bit architecture will be limited to 4G; host systems with a 64-bit architecture will be limited to less than 16E.

Using minimum storage sizes, storage sizes less than or not on a 64K boundary for architecture level ALS0 (S/370) or not on a 1M boundary for architecture level ALS1 (ESA/390) and higher, it may be possible to generate error conditions not covered by the "Principles of Operations".

Use of storage sizes greater than supported by the guest operating system may generate incorrect results or error conditions within the guest operating system.

8.48.4 Overview Storage Allocation Units

Unit	Multiplier	Name (Symbol)	IEC Name (IEC Symbol)	Restrictions
B	None	Byte (B)	Byte (B)	Main storage only
K	2**10	Kilobyte (kB)	Kibibyte (KiB)	Main storage only
M	2**20	Megabyte (MB)	Mebibyte (MiB)	
G	2**30	Gigabyte (GB)	Gibibyte (GiB)	
T	2**40	Terabyte (TB)	Tebibyte (TiB)	Not on 32-bit machines
P	2**50	Petabyte (PB)	Pebibyte (PiB)	Not on 32-bit machines
E	2*60	Exabyte (EB)	Exbibyte (EiB)	Not on 32-bit machines

Table 19: Storage Allocation Units

8.48.5 Examples

Example 1:

Display the current size of the main and expanded storage.

```
HHC00013I Herc command: 'defstore'
HHC17003I MAIN     storage is 256 M (mainsize); storage is not locked
HHC17003I EXPANDED storage is 256 M (xpndsize); storage is not locked
```

Figure 98: MAINSIZE command (display size of main and expanded storage)

Example 2:

Display the current size of the main storage.

```
HHC00013I Herc command: 'defstore main'
HHC17003I MAIN     storage is 256 M (mainsize); storage is not locked
```

Figure 99: MAINSIZE command (display size of main storage)

Example 3:

Set the size of the main storage to 512 MB. Lock the storage into the memory.

```
HHC00013I Herc command: 'defstore main 512 lock'
HHC01428I Locking main storage
HHC17003I MAIN     storage is 512 M (mainsize); storage is locked
```

Figure 100: MAINSIZE command (Set size of locked main storage)

Example 4:

Set the size of the main and expanded storage to 512 MB each. Lock the main storage into the memory, but do not lock the expanded storage.

```
HHC00013I Herc command: 'defstore main 512M lock xstor 512M unlock'
HHC01428I Locking main storage
HHC17003I MAIN     storage is 512 M (mainsize); storage is locked
HHC17003I EXPANDED storage is 512 M (xpndsize); storage is not locked
```

Figure 101: MAINSIZE command (Set size of main and expanded storage)

8.49 DEFSYM (Define a symbol)

8.49.1 Function

The DEFSYM command defines symbol '*symbol*' as to contain value '*value*'. The symbol can then be the object of a substitution for later panel commands. If the *value* contains blanks or spaces then it must be enclosed within quotes or apostrophes. See chapter "Symbol Substitutions" for a more in-depth discussion on this feature.

8.49.2 Syntax

Descriptive

DEFSYM [*symbol* [*value*]]

Diagram

```
►►── DEFSYM ──┬─────────────────────────────┬──────────────────────────────►◄
              └── symbol ──┬────────────────┤
                           └── value ───────┘
```

8.49.3 Parameter

symbol The name of a symbol. If no symbol name (and optionally a value) is given then all previously defined symbols are listed.

value The value that is assigned to the symbol. If the value is omitted then the previously defined symbol is cleared, the symbol itself is not deleted.

8.49.4 Examples

Example 1:

List all defined symbols.

```
HHC00013I Herc command: 'defsym'
HHC02199I Symbol VERSION      '3.0.7.6879'
HHC02199I Symbol BDATE        'Oct 12 2010'
HHC02199I Symbol BTIME        '18:23:14'
HHC02199I Symbol HOSTNAME     'GOOFY'
HHC02199I Symbol HOSTOS       'Windows'
HHC02199I Symbol HOSTOSREL    '6.1.7600'
HHC02199I Symbol HOSTOSVER    'NT '
HHC02199I Symbol HOSTARCH     'i686'
```

```
HHC02199I Symbol HOSTNUMCPUS  'MP=8'
HHC02199I Symbol MODNAME      'hercules.exe'
HHC02199I Symbol MODPATH      'D:\Hercules\'
HHC02199I Symbol PF01         'SUBST IMMED herc help &0'
HHC02199I Symbol PF11         'IMMED herc devlist TAPE'
HHC02199I Symbol PF10         'SUBST DELAY herc devinit &*'
HHC02199I Symbol CUU          '$(CUU)'
HHC02199I Symbol CCUU         '$(CCUU)'
HHC02199I Symbol DASDPATH     'D:/MVS/DASD'
HHC02199I Symbol READERPATH   'D:/MVS/READER'
HHC02199I Symbol PRINTPATH    'D:/MVS/PRINTER'
HHC02199I Symbol PUNCHPATH    'D:/MVS/PUNCH'
HHC02199I Symbol CSS          '0'
HHC02199I Symbol SUBCHAN      '0008'
```

Figure 102: DEFSYM command (list all symbols)

Example 2:

Define new symbol (and list the symbols afterwards).

```
HHC00013I Herc command: 'defsym TAPEDIR "D:\MVS\TAPE"'
HHC00013I Herc command: 'defsym'
HHC02199I Symbol VERSION      '3.0.7.6879'
HHC02199I Symbol BDATE        'Oct 12 2010'
HHC02199I Symbol BTIME        '18:23:14'
.
several lines not displayed
.
HHC02199I Symbol SUBCHAN      '0008'
HHC02199I Symbol TAPEDIR      'D:\MVS\TAPE'
```

Figure 103: DEFSYM command (define new symbol)

Example 3:

Clear defined symbol (and list the symbols afterwards).

```
HHC00013I Herc command: 'defsym TAPEDIR'
HHC00013I Herc command: 'defsym'
HHC02199I Symbol VERSION      '3.0.7.6879'
HHC02199I Symbol BDATE        'Oct 12 2010'
HHC02199I Symbol BTIME        '18:23:14'
.
several lines not displayed
.
HHC02199I Symbol SUBCHAN      '0008'
HHC02199I Symbol TAPEDIR      ''
```

Figure 104: DEFSYM command (clear defined symbol)

8.50 DELSYM (Delete a symbol)

8.50.1 Function

The DELSYM command deletes a symbol. Please note that the “DELSYM *symbol*” command actually deletes the symbol while the “DEFSYM *symbol*” command (without a given value) clears the symbol, but does not delete it.

8.50.2 Syntax

Descriptive

DELSYM *symbol*

Diagram

```
►►── DELSYM ── symbol ──────────────────────────────────────►◄
```

8.50.3 Parameter

symbol This is the name of the symbol to be deleted.

8.50.4 Examples

Example 1:

Delete symbol TAPEDIR.

```
02:24:08 HHC00013I Herc command: 'delsym TAPEDIR'
```

Figure 105: DELSYM command

8.51 DETACH (Remove device)

8.51.1 Function

The DETACH command removes a device from the current configuration. The effect of this command is immediately visible in the device display of the Hercules Windows GUI and the Hercules “Peripherals” display.

8.51.2 Syntax

Descriptive

DETACH *device*

Diagram

►►── DETACH ── *device* ──────────────────────────────►◄

8.51.3 Parameter

device The device number of the device that is to be removed from the current configuration.

8.51.4 Examples

Example 1:

Detach device 000F from the active configuration.

```
HHC00013I Herc command: 'detach 000f'
HHC01465I 0:000F device detached
```

Figure 106: DETACH command

8.52 DEVINIT (Reinitialize device)

8.52.1 Function

The DEVINIT command can be used to reopen an existing device. The arguments (if any) override the arguments specified in the configuration file for this device. If no arguments (beneath the device number) are given, then the DEVINIT command will reinitialise the device with the same arguments that were previously specified. The device type cannot be changed and must not be specified.

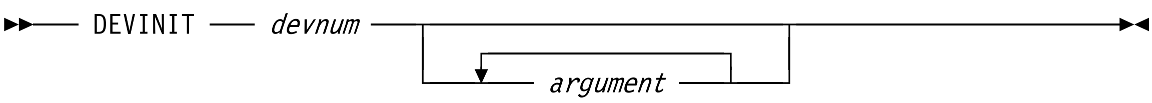
This command can be used to rewind a tape, to mount a new tape or disk image file on an existing device, to load a new card deck into a reader, or to close and reopen a printer or punch device.

8.52.2 Syntax

Descriptive

DEVINIT *devnum* [*argument* [*argument* ...]]

Diagram



8.52.3 Parameter

devnum The device number of the device to be reinitialized.

argument Any arguments that are acceptable for the kind of device being reinitalized.

8.52.4 Examples

Example 1:

Initialize a tape device.

```
HHC00013I Herc command: 'devinit 0480 D:/MVS/TAPE/RPF142.HET compress=1 method=1 level=4'
HHC00221I 0:0480 Tape file 'D:/MVS/TAPE/RPF142.HET', type 'het': format type 'HET file'
HHC00222I 0:0480 Tape file 'D:/MVS/TAPE/RPF142.HET', type 'het': option 'compress' accepted
HHC00222I 0:0480 Tape file 'D:/MVS/TAPE/RPF142.HET', type 'het': option 'method' accepted
HHC00222I 0:0480 Tape file 'D:/MVS/TAPE/RPF142.HET', type 'het': option 'level' accepted
HHC02245I 0:0480 device initialized
```

Figure 107: DEVINIT command

8.53 DEVLIST (List device, device class or all devices)

8.53.1 Function

The DEVLIST command lists devices that are defined in the active Hercules configuration and displays their current status. Depending on the given argument DEVLIST shows information about one device only (when *devnum* is specified) or about all devices of a certain device class (when *devclass* is specified). Without an argument DEVLIST displays information about all defined devices.

8.53.2 Syntax

Descriptive

DEVLIST [*devnum* | *devclass*]

Diagram

```
►►── DEVLIST ──┬──────────────┬──────────────────────────────────►◄
               ├─── devnum ───┤
               └── devclass ──┘
```

8.53.3 Parameter

devnum This specifies the device number of the device for which the status has to be displayed.

devclass This specifies the device class for which DEVLIST has to display the status information.

Valid device classes are:

CON	(Console devices)
CTCA	(Channel-to-channel adapter)
DASD	(Disk devices)
DSP	(Terminals)
LINE	(Communication lines)
PCH	(Card punch devices)
PRT	(Printer devices)
QETH	(QETH devices)

8.53.4 Examples

Example 1:

List all devices of the current configuration.

```
HHC00013I Herc command: 'devlist'
HHC02279I 0:000C 3505 D:/MVS/READER/DUMMY.JCL ascii trunc eof IO[2] open
HHC02279I 0:000D 3525 D:/MVS/PUNCH/PCH00D.TXT ascii crlf IO[2] open
HHC02279I 0:000E 1403 D:/MVS/PRINTER/PRT00E.TXT crlf brwse fcbck IO[2] open
HHC02279I 0:000F 1403 D:/MVS/PRINTER/PRT00F.TXT crlf brwse fcbck IO[2] open
HHC02279I 0:0010 3270 192.168.0.101 IO[173] open
HHC02279I 0:0011 3270 192.168.0.101 IO[61] open
HHC02279I 0:00C0 3270 192.168.0.101 IO[1] open
HHC02279I 0:00C1 3270 192.168.0.101 IO[1] open
HHC02279I 0:00C2 3270 * 192.168.0.101 mask 255.255.255.255 IO[2]
HHC02279I 0:00C3 3270 * 192.168.0.101 mask 255.255.255.255 IO[2]
HHC02279I 0:00C4 3270 * 192.168.0.101 mask 255.255.255.255 IO[2]
HHC02279I 0:00C5 3270 * 192.168.0.101 mask 255.255.255.255 IO[2]
HHC02279I 0:0130 2314 D:/MVS/DASD/SORT00.CCKD [200 cyls] [0 sfs] IO[30] open
HHC02279I 0:0131 2314 D:/MVS/DASD/SORT01.CCKD [200 cyls] [0 sfs] IO[30] open
HHC02279I 0:0132 2314 D:/MVS/DASD/SORT02.CCKD [203 cyls] [0 sfs] IO[30] open
HHC02279I 0:0133 2314 D:/MVS/DASD/SORT03.CCKD [200 cyls] [0 sfs] IO[30] open
HHC02279I 0:0134 2314 D:/MVS/DASD/SORT04.CCKD [200 cyls] [0 sfs] IO[30] open
HHC02279I 0:0135 2314 D:/MVS/DASD/SORT05.CCKD [200 cyls] [0 sfs] IO[30] open
.
several lines not displayed
.
HHC02279I 0:0148 3350 D:/MVS/DASD/MVSRES.CCKD [560 cyls] [0 sfs] IO[92230] open
HHC02279I 0:0149 3350 D:/MVS/DASD/MVSDLB.CCKD [555 cyls] [0 sfs] IO[567] open
HHC02279I 0:0160 3340 D:/MVS/DASD/PAGE00.CCKD [698 cyls] [0 sfs] IO[14302] open
HHC02279I 0:0161 3340 D:/MVS/DASD/PAGE01.CCKD [698 cyls] [0 sfs] IO[441] open
HHC02279I 0:0240 3350 D:/MVS/DASD/PUB000.CCKD [555 cyls] [0 sfs] IO[204] open
HHC02279I 0:0241 3350 D:/MVS/DASD/PUB001.CCKD [555 cyls] [0 sfs] IO[80] open
HHC02279I 0:0242 3350 D:/MVS/DASD/PUB002.CCKD [560 cyls] [0 sfs] IO[25] open
HHC02279I 0:0243 3350 D:/MVS/DASD/PUB003.CCKD [560 cyls] [0 sfs] IO[25] open
HHC02279I 0:0248 3350 D:/MVS/DASD/SYS000.CCKD [560 cyls] [0 sfs] IO[25] open
HHC02279I 0:0249 3350 D:/MVS/DASD/SYS001.CCKD [560 cyls] [0 sfs] IO[18] open
HHC02279I 0:024A 3350 D:/MVS/DASD/SYS002.CCKD [560 cyls] [0 sfs] IO[18] open
HHC02279I 0:024B 3350 D:/MVS/DASD/SYS003.CCKD [560 cyls] [0 sfs] IO[18] open
HHC02279I 0:030E 1403 D:/MVS/LOG/hardcopy.log crlf brwse fcbck IO[145] open
HHC02279I 0:0344 3350 D:/MVS/DASD/SPOOL0.CCKD [560 cyls] [0 sfs] IO[41] open
HHC02279I 0:0345 3350 D:/MVS/DASD/SPOOL1.CCKD [560 cyls] [0 sfs] IO[150] open
HHC02279I 0:0348 3350 D:/MVS/DASD/TST000.CCKD [560 cyls] [0 sfs] IO[22] open
HHC02279I 0:0349 3350 D:/MVS/DASD/TST001.CCKD [560 cyls] [0 sfs] IO[22] open
HHC02279I 0:034A 3350 D:/MVS/DASD/TST002.CCKD [560 cyls] [0 sfs] IO[15] open
HHC02279I 0:034B 3350 D:/MVS/DASD/TST003.CCKD [560 cyls] [0 sfs] IO[15] open
HHC02279I 0:0480 3420 * IO[2]
HHC02279I 0:0481 3420 * IO[2]
HHC02279I 0:0E20 3088 CTCI 192.168.0.99/192.168.0.101 () IO[0]
```

Figure 108: DEVLIST command (list all devices)

Example 2:

List all devices of class DSP (display).

```
HHC00013I Herc command: 'devlist dsp'
HHC02279I 0:0010 3270 192.168.0.101 IO[173] open
HHC02279I 0:0011 3270 192.168.0.101 IO[61] open
HHC02279I 0:00C0 3270 192.168.0.101 IO[1] open
HHC02279I 0:00C1 3270 192.168.0.101 IO[1] open
HHC02279I 0:00C2 3270 * 192.168.0.101 mask 255.255.255.255 IO[2]
HHC02279I 0:00C3 3270 * 192.168.0.101 mask 255.255.255.255 IO[2]
HHC02279I 0:00C4 3270 * 192.168.0.101 mask 255.255.255.255 IO[2]
HHC02279I 0:00C5 3270 * 192.168.0.101 mask 255.255.255.255 IO[2]
```

Figure 109: DEVLIST command (specify device class)

Example 3:

List one device (0148) by device number.

```
HHC00013I Herc command: 'devlist 0148'
HHC02279I 0:0148 3350 D:/MVS/DASD/MVSRES.CCKD [560 cyls] [0 sfs] IO[92230] open
```

Figure 110: DEVLIST command (specify device number)

8.54 DEVPRIO (Display or set device threads process priority)

8.54.1 Function

The DEVPRIO command is used to change the priority of the device threads. See section 5.82 for details on process and thread priorities. Given without an argument the DEVPRIO command displays the current device threads process priority.

Caution: DEVPRIO should not have a higher dispatching priority than the TOD clock and timer thread.

8.54.2 Syntax

Descriptive

DEVPRIO [*nn*]

Diagram

```
►►── DEVPRIO ──┬────────┬──────────────────────────────────►◄
               └── nn ──┘
```

8.54.3 Parameter

nn This value specifies the priority for the device threads. For details on the priority values see section 5.82 (“Process and Thread Priorities”).

8.54.4 Examples

Example 1:

Set the device threads process priority to1 0.

```
HHC00013I Herc command: 'devprio 10'
HHC02204I devprio       set to 10
```

Figure 111: DEVPRIO command

8.55 DEVTMAX (Display or set maximum device threads)

8.55.1 Function

The DEVTMAX command displays the current setting of the DEVTMAX system parameter along with the actual device threads statistics. As an additional argument a new value for the maximum device threads may be specified.

8.55.2 Syntax

Descriptive

DEVTMAX [-1 | 0 | 1-*n*]

Diagram

```
►►── DEVTMAX ──┬──────────────┬──────────────────────────────►◄
               ├──── -1 ──────┤
               ├──── 0 ───────┤
               └──── 1-n ─────┘
```

8.55.3 Parameter

0 Specify 0 to create an unlimited number of ‘semi-permanent’ threads on an ‘as-needed’ basis. This is the default. With this option, a thread is created to service an I/O request for a device if one does not already exist. When the I/O is complete the thread enters an idle state waiting for new work. If a new I/O request for the device arrives before the timeout period expires the existing thread will be reused. The timeout value is currently hard coded at 5 minutes.

Note that this option can cause one thread (or possibly more) to be created for each device in your configuration. Specifying 0 means there is no limit to the number of threads that can be created.

-1 Specify -1 to cause ‘one time only’ temporary threads to be created to service each I/O request to a device. Once the I/O request is complete, the thread exits. Subsequent I/Os to the same device will cause another worker thread to be created again.

1-*n* Specify a value from 1 to n to set an upper limit to the number of threads that can be created to service any I/O request to any device. Like the “0” option, each thread once finished servicing an I/O request enters an idle state. If a new request arrives before the timeout period expires, the thread is reused.

If all threads are busy when a new I/O request arrives a new thread is created only if the specified maximum number of threads have not yet been reached. If the specified maximum number of threads already has been reached then the I/O request is placed in a queue and will be serviced by the first available thread (eg. by whichever thread becomes idle first).

This option was created to address a threading issue, possibly related to the Cygwin phtreads implementation on Windows systems. On Windows systems positive DEVTMAX values are currently not honoured and are treated identically as if the value 0 had been specified. The default for non-Windows systems is 0.

8.55.4 Examples

Example 1:

Display the maximum allowed device threads along with the current statistics.

```
HHC00013I Herc command: 'devtmax'
HHC02241I Max device threads: 0, current: 5, most: 5, waiting: 4, max exceeded: 0
```

Figure 112: DEVTMAX command (list maximum allowed device threads)

Example 2:

Set the maximum allowed device threads to 25.

```
HHC00013I Herc command: 'devtmax 25'
```

Figure 113: DEVTMAX command (set maximum allowed device threads)

8.56 DIAG8CMD (Display or set DIAGNOSE 8 command option)

8.56.1 Function

The DIAG8CMD command specifies whether a command issued through Diagnose 8 will be executed by Hercules as a Hercules commands or not. An optional second argument can be specified to request whether the commands issued using the Diagnose 8 interface will be traced at the console. This may be useful for programs that routinely issue console commands using the Diagnose 8 interface. If no argument is given, DIAG8CMD displays the current settings.

Caution: Enabling this feature may have security consequences. When this feature is enabled it is possible for guest operating systems running under Hercules to issue commands directly to the host operating system by means of the Hercules 'sh' (shell) command. This ability may be disabled via the SHCMDOPT statement.

Note: There are some commands that are being prevented from being used by the Diagnose 8 interface. The list of commands that may not be executed by means of Diagnose 8 can be found in "Appendix D. Hercules Command Groups" under the column 'NODIAG8'.

8.56.2 Syntax

Descriptive

```
DIAG8CMD [DISABLE | ENABLE [ECHO | NOECHO]]
```

Diagram

```
►►── DIAG8CMD ──┬─────────────────────────────┬──────────────────────►◄
                ├── DISABLE ──────────────────┤
                └── ENABLE ──┬────────────────┤
                             ├──── ECHO ──────┤
                             └─── NOECHO ─────┘
```

8.56.3 Parameter

DISABLE Commands issued through the Diagnose 8 interface will generate a Specification Exception program interrupt on the issuing CPU. This is the default together with NOECHO.

ENABLED Commands issued through the Diagnose 8 interface will be executed by Hercules as Hercules commands.

ECHO When ECHO is specified, a message is issued as the console is about to issue the command, the command is redisplayed as if it was entered through the console input line, and a final message is issued to indicate the command completed.

NOECHO When NOECHO is specified, no such messages are displayed and the command completes silently, except for the output of the command itself if the Diagnose 8 interface did not request a response buffer. This is the default together with DISABLE.

The value of ECHO or NOECHO has no effect on command output being placed into a response buffer if the Diagnose 8 interface requested one.

8.56.4 Examples

Example 1:

Display the current DIAG8CMD settings.

```
HHC00013I Herc command: 'diag8cmd'
HHC02203I DIAG8CMD      : disable, no echo
```

Figure 114: DIAG8CMD command (display current settings)

Example 2:

Specify that commands issued through the Diagnose 8 interface are executed as Hercules commands. Additionally issue a message, as the console is about to execute the command, then redisplay the command itself and give a final message, indicating the command has completed.

```
HHC00013I Herc command: 'diag8cmd enable echo'
```

Figure 115: DIAG8CMD command (set new DIAG8CMD mode)

8.57 DIR (Display file and directory listing)

8.57.1 Function

The DIR command displays a list of files and subdirectories in the current directory. The DIR command is only available when running Hercules under Microsoft Windows. For Linux and Mac OS X please use the "LS" command.

8.57.2 Syntax

Descriptive

```
DIR
```

Diagram

```
►►── DIR ──────────────────────────────────────────────────────►◄
```

8.57.3 Parameter

None.

8.57.4 Examples

Example 1:

Show the list of files and subdirectories in the current directory.

```
HHC00013I Herc command: 'dir'
Volume in drive D is Hercules
Volume Serial Number is 6CD7-297E

Directory of D:\Hercules

11.07.2010  01:13    <DIR>          .
11.07.2010  01:13    <DIR>          ..
16.12.2007  01:00           282'624 AWSBrowse32.exe
16.12.2007  01:00           503'808 AWSBrowse32D.exe
16.12.2007  01:00           274'432 AWSBrowse32U.exe
16.12.2007  01:00           491'520 AWSBrowse32UD.exe
28.02.2007  06:24           323'072 AWSBrowse64.exe
28.02.2007  06:24           647'168 AWSBrowse64D.exe
28.02.2007  06:24           317'440 AWSBrowse64U.exe
28.02.2007  06:24           636'416 AWSBrowse64UD.exe
```

```
07.02.2007  05:37               934 awshet.reg
10.07.2010  02:45            15'872 cckdcdsk.exe
10.07.2010  02:45            15'872 cckdcomp.exe
10.07.2010  02:45            24'064 cckddiag.exe
10.07.2010  02:45            17'408 cckdswap.exe
08.01.2009  16:43           162'984 CHANGES
10.07.2010  02:45            14'336 conspawn.exe
.
several lines not displayed
.
16.12.2007  01:00           122'880 TT32Test.exe
28.02.2007  06:28           158'208 TT32Test64.exe
28.02.2007  06:28           390'656 TT32Test64D.exe
28.02.2007  06:28           163'840 TT32Test64U.exe
28.02.2007  06:28           398'336 TT32Test64UD.exe
16.12.2007  01:00           290'816 TT32TestD.exe
16.12.2007  01:00           131'072 TT32TestU.exe
16.12.2007  01:00           299'008 TT32TestUD.exe
16.12.2007  01:00           131'072 TunTap32.dll
16.12.2007  01:00           344'064 TunTap32D.dll
16.12.2007  01:00           135'168 TunTap32U.dll
16.12.2007  01:00           344'064 TunTap32UD.dll
28.02.2007  06:27           144'384 TunTap64.dll
28.02.2007  06:27           561'664 TunTap64D.dll
28.02.2007  06:27           147'456 TunTap64U.dll
28.02.2007  06:27           565'760 TunTap64UD.dll
12.06.2010  18:03    <DIR>          util
08.02.2009  20:03            75'264 zlib1.dll
             121 File(s)     25'408'168 bytes
               5 Dir(s)  74'086'850'560 bytes free
```

Figure 116: DIR command

8.58 DS (Display subchannel)

8.58.1 Function

The DS command shows the subchannel information for a given device number.

8.58.2 Syntax

Descriptive

DS *devnum*

Diagram

```
►►── DS ── devnum ──────────────────────────────────────────►◄
```

8.58.3 Parameter

devnum The device number for which the subchannel information has to be displayed.

8.58.4 Examples

Example 1:

Display subchannel information for device 0AE0.

```
HHC00013I Herc command: 'ds 0ae0'
HHC02268I 0:0AE0 D/T3390 Subchannel Number[0028]
HHC02268I     Path Management Control Word (PMCW)
HHC02268I   IntParm:00F3F8A8
HHC02268I     Flags:189D        Dev:0AE0
HHC02268I       LPM:80 PNOM:00 LPUM:80 PIM:80
HHC02268I       MBI:0183        POM:FF PAM:80
HHC02268I    CHPID0:0A    1:00    2:00   3:00
HHC02268I         4:00    5:00    6:00   7:00
HHC02268I      Misc:00000001
HHC02268I     Subchannel Status Word (SCSW)
HHC02268I     Flags:08C0 SCHC:4020 DS:00 SS:00 Count:0001 CCW:7F01CB70
HHC02268I     Device Status     is Normal
HHC02268I     Subchannel Status is Normal
```

Figure 117: DS command

8.58.5 Explanations

The information presented in the output from the Display Subchannel command is taken from the Sub-channel-Information Block (SCHIB). The Subchannel-Information Block shows - beneath other areas - the Path-Management-Control Word (PMCW) and the Subchannel-Status Word (SCSW). The format of the SCHIB is shown on the following page.

Word	
0-6	Path-Management-Control Word
7-9	Subchannel-Status Word
10-11	Model-Dependent Area / Measurement-Block Adress
12	Model-Dependent Area

Figure 118: Subchannel-Information Block

Words 0-6 of the SCHIB contain the Path-Management-Control Word (PMCW). The format of the PMCW is the following:

<table>
<tr><th>Word</th><th colspan="32"></th></tr>
<tr><td>0</td><td colspan="32">Interruption Parameter</td></tr>
<tr><td>1</td><td colspan="2">00</td><td colspan="3">ISC</td><td colspan="3">000</td><td>E</td><td colspan="2">LM</td><td colspan="2">MM</td><td>D</td><td>T</td><td>V</td><td colspan="16">Device Number</td></tr>
<tr><td>2</td><td colspan="8">LPM</td><td colspan="8">PNOM</td><td colspan="8">LPUM</td><td colspan="8">PIM</td></tr>
<tr><td>3</td><td colspan="16">MBI</td><td colspan="8">POM</td><td colspan="8">PAM</td></tr>
<tr><td>4</td><td colspan="8">CHPID-0</td><td colspan="8">CHPID-1</td><td colspan="8">CHPID-2</td><td colspan="8">CHPID-3</td></tr>
<tr><td>5</td><td colspan="8">CHPID-4</td><td colspan="8">CHPID-5</td><td colspan="8">CHPID-6</td><td colspan="8">CHPID-7</td></tr>
<tr><td>6</td><td colspan="8">0 0 0 0 0 0 0 0</td><td colspan="8">0 0 0 0 0 0 0 0</td><td colspan="8">0 0 0 0 0 0 0 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>F</td><td>X</td><td>S</td></tr>
</table>

Figure 119: Path-Management-Control Word

Legend:

ISC	Interruption-Subclass Code
E	Enabled
LM	Limit Mode

MM Measurement-Mode Enable

D Multipath Mode

T Timing Facility

V Device Number Valid

LPM Logical-Path Mask

PNOM Path-Not-Operational Mask

LPUM Last-Path-Used Mask

PIM Path-Installed Mask

MBI Measurement-Block Index

POM Path-Operational Mask

PAM Path-Available Mask

CHPIDs Channel-Path Identifiers

F Measurement Block Format Control

X Extended Measurement Word Enable

S Concurrent Sense

The Subchannel-Status Word (SCSW) provides information about the status of a subchannel and its associated devices. The format of the SCSW is as follows:

<table>
<tr><td>Word</td><td colspan="16"></td></tr>
<tr><td>0</td><td>Key</td><td>S</td><td>L</td><td>CC</td><td>F</td><td>P</td><td>I</td><td>A</td><td>U</td><td>Z</td><td>E</td><td>N</td><td>0</td><td>FC</td><td>AC</td><td>SC</td></tr>
<tr><td>1</td><td colspan="16">CCW Address</td></tr>
<tr><td>2</td><td colspan="4">Device Status</td><td colspan="8">Subchannel Status</td><td colspan="4">Count</td></tr>
</table>

Figure 120: Subchannel-Status Word

Legend:

Key Subchannel Key

S Suspend Control

L ESW Format

CC Deferred Condition Code

F Format

P Prefetch

I Initial-Status Interruption Control

A Address-Limit-Checking Control

U Suppress-Suspend Interruption

Z Zero Condition Code

E Extended Control

N	Path Not Operational
0	Reserved
FC	Function Control
AC	Activity Control
SC	Status Control
Count	Residual Count

Details about the control blocks, that have been described here only briefly, can be found in IBM's "z/Architecture Principles of Operation" manual.

8.59 ECPSVM (ECPS:VM commands)

8.59.1 Function

This command specifies, whether ECPS:VM (Extended Control Program Support : Virtual Machine) support is to be enabled and – if it is enabled – to which level. Issuing command ECPSVM HELP displays a list of all available commands.

The purpose of ECPS:VM is to provide to the VM/370 operating system a set of shortcut facilities to perform hypervisor functions (CP Assists) and virtual machine simulation (VM Assists).

Although this feature does not affect VM operating system products operating in XA, ESA or z/Architecture mode, it will affect VM/370 and VM/SP products running under VM/XA, VM/ESA or z/VM. Running VM/370 and VM/SP products under VM/XA, VM/ESA or z/VM should be done with ECPS:VM disabled. ECPS:VM should not be enabled in AP or MP environments. ECPS:VM has no effect on non-VM operating systems. It is however recommended to disable ECPS:VM when running native non-VM operating systems.

If a specific level is given this value will be reported to the operating system when it issues a Store ECPS:VM level but it does not otherwise alter the ECPS:VM facility operations. Please note that this is a partial implementation.

Some of the arguments of the ECPSVM command can be abbreviated as shown in the syntax section below.

8.59.2 Syntax

Descriptive

`ECPSVM [Help | STats | DISable | ENable | DEBUG | NOdebug | Level [`*nn*`]]`

Diagram

```
►►── ECPSVM ──┬──────────────────────────┬────────────────────────►◄
              ├──── Help ────────────────┤
              ├──── STats ───────────────┤
              ├──── DISable ─────────────┤
              ├──── ENable ──────────────┤
              ├──── DEBUG ───────────────┤
              ├──── NOdebug ─────────────┤
              └──── Level ───┬───────────┤
                             └─── nn ────┘
```

8.59.3 Parameter

Help Display help with all available ECPSVM options.

STats Show statistical counters.

DISable Disable all ECPS:VM features. This is the default if ECPSVM is not coded.

ENable Enable all ECPS:VM features.

DEBUG Debug ECPS:VM features.

NOdebug Turn debug modus off for ECPS:VM features.

Level Set/show ECPS:VM level.

nn The value *nn* specifies the support level that is reported to the operating system.

8.59.4 Examples

Example 1:

Enable the ECPS:VM features.

```
HHC00013I Herc command: 'ecpsvm enable'
HHC01719I ECPS:VM Command processor invoked
HHC01707I ECPS:VM VM ASSIST feature SVC Enabled
HHC01707I ECPS:VM VM ASSIST feature SSM Enabled
HHC01707I ECPS:VM VM ASSIST feature LPSW Enabled
HHC01707I ECPS:VM VM ASSIST feature STNSM Enabled
HHC01707I ECPS:VM VM ASSIST feature STOSM Enabled
HHC01707I ECPS:VM VM ASSIST feature SIO Enabled
HHC01707I ECPS:VM VM ASSIST feature VTIMER Enabled
HHC01707I ECPS:VM VM ASSIST feature STCTL Enabled
HHC01707I ECPS:VM VM ASSIST feature LCTL Enabled
HHC01707I ECPS:VM VM ASSIST feature DIAG Enabled
HHC01707I ECPS:VM VM ASSIST feature IUCV Enabled
HHC01708I All ECPS:VM VM ASSIST features Enabled
HHC01707I ECPS:VM CP ASSIST feature FREE Enabled


HHC01707I ECPS:VM CP ASSIST feature FRET Enabled
HHC01707I ECPS:VM CP ASSIST feature LCKPG Enabled
HHC01707I ECPS:VM CP ASSIST feature ULKPG Enabled
HHC01707I ECPS:VM CP ASSIST feature SCNRU Enabled
HHC01707I ECPS:VM CP ASSIST feature SCNVU Enabled
HHC01707I ECPS:VM CP ASSIST feature DISP0 Enabled
HHC01707I ECPS:VM CP ASSIST feature DISP1 Enabled
HHC01707I ECPS:VM CP ASSIST feature DISP2 Enabled
HHC01707I ECPS:VM CP ASSIST feature DNCCW Enabled
HHC01707I ECPS:VM CP ASSIST feature DFCCW Enabled
HHC01707I ECPS:VM CP ASSIST feature FCCWS Enabled
```

```
HHC01707I ECPS:VM CP ASSIST feature CCWGN Enabled
HHC01707I ECPS:VM CP ASSIST feature UXCCW Enabled
HHC01707I ECPS:VM CP ASSIST feature TRBRG Enabled
HHC01707I ECPS:VM CP ASSIST feature TRLOK Enabled
HHC01707I ECPS:VM CP ASSIST feature VIST Enabled
HHC01707I ECPS:VM CP ASSIST feature VIPT Enabled
HHC01707I ECPS:VM CP ASSIST feature STEVL Enabled
HHC01707I ECPS:VM CP ASSIST feature FREEX Enabled
HHC01707I ECPS:VM CP ASSIST feature FRETX Enabled
HHC01707I ECPS:VM CP ASSIST feature PMASS Enabled
HHC01707I ECPS:VM CP ASSIST feature LCSPG Enabled
HHC01708I All ECPS:VM CP ASSIST features Enabled
HHC01722I ECPS:VM Command processor complete
```

Figure 121: ECPSVM ENABLE command

Example 2:

Show ECPS:VM statistical counters.

```
HHC00013I Herc command: 'ecpsvm stats'
HHC01719I ECPS:VM Command processor invoked
HHC01702I +-----------+----------+----------+-------+
HHC01706I | VM ASSIST | Calls    | Hits     | Ratio |
HHC01702I +-----------+----------+----------+-------+
HHC01701I | Total     |        0 |        0 |  100% |
HHC01702I +-----------+----------+----------+-------+
HHC01704I 11 entry/entries not shown and never invoked
HHC01702I +-----------+----------+----------+-------+
HHC01706I | CP ASSIST | Calls    | Hits     | Ratio |
HHC01702I +-----------+----------+----------+-------+
HHC01701I | Total     |        0 |        0 |  100% |
HHC01702I +-----------+----------+----------+-------+
HHC01704I 23 entry/entries not shown and never invoked
HHC01722I ECPS:VM Command processor complete
```

Figure 122: ECPSVM STATS command

Example 3:

Show the ECPS:VM level.

```
HHC00013I Herc command: 'ecpsvm level'
HHC01719I ECPS:VM Command processor invoked
HHC01712I Current reported ECPS:VM level is 20
HHC01713I But ECPS:VM is currently disabled
HHC01722I ECPS:VM Command processor complete
```

Figure 123: ECPSVM LEVEL command

Example 4:

Debug the ECPS:VM features.

```
HHC00013I Herc command: 'ecpsvm debug'
HHC01719I ECPS:VM Command processor invoked
HHC01707I ECPS:VM VM ASSIST feature SVC Debug On
HHC01707I ECPS:VM VM ASSIST feature SSM Debug On
HHC01707I ECPS:VM VM ASSIST feature LPSW Debug On
HHC01707I ECPS:VM VM ASSIST feature STNSM Debug On
HHC01707I ECPS:VM VM ASSIST feature STOSM Debug On
HHC01707I ECPS:VM VM ASSIST feature SIO Debug On
HHC01707I ECPS:VM VM ASSIST feature VTIMER Debug On
HHC01707I ECPS:VM VM ASSIST feature STCTL Debug On
HHC01707I ECPS:VM VM ASSIST feature LCTL Debug On
HHC01707I ECPS:VM VM ASSIST feature DIAG Debug On
HHC01707I ECPS:VM VM ASSIST feature IUCV Debug On
HHC01708I All ECPS:VM VM ASSIST features Debug On
HHC01707I ECPS:VM CP ASSIST feature FREE Debug On
HHC01707I ECPS:VM CP ASSIST feature FRET Debug On
.
several lines not displayed
.
HHC01707I ECPS:VM CP ASSIST feature FCCWS Debug On
HHC01707I ECPS:VM CP ASSIST feature CCWGN Debug On
HHC01707I ECPS:VM CP ASSIST feature UXCCW Debug On
HHC01707I ECPS:VM CP ASSIST feature TRBRG Debug On
HHC01707I ECPS:VM CP ASSIST feature TRLOK Debug On
HHC01707I ECPS:VM CP ASSIST feature VIST Debug On
HHC01707I ECPS:VM CP ASSIST feature VIPT Debug On
HHC01707I ECPS:VM CP ASSIST feature STEVL Debug On
HHC01707I ECPS:VM CP ASSIST feature FREEX Debug On
HHC01707I ECPS:VM CP ASSIST feature FRETX Debug On
HHC01707I ECPS:VM CP ASSIST feature PMASS Debug On
HHC01707I ECPS:VM CP ASSIST feature LCSPG Debug On
HHC01708I All ECPS:VM CP ASSIST features Debug On
HHC01709I ECPS:VM global debug On
HHC01722I ECPS:VM Command processor complete
```

Figure 124: ECPSVM DEBUG command

8.60 ENGINES (Set processor engines type)

8.60.1 Function

The ENGINES command specifies the type of engine for each installed processor. The default engine type is CP. The number of installed processor engines is determined by the MAXCPU system parameter.

If the ENGINES command specifies more than MAXCPU engines, the excess engines are ignored. If fewer than MAXCPU engines are specified, the remaining engines are set to type CP (the default). See the MAXCPU system parameter or console command for details regarding the compile time variable MAX_CPU_ENGINES.

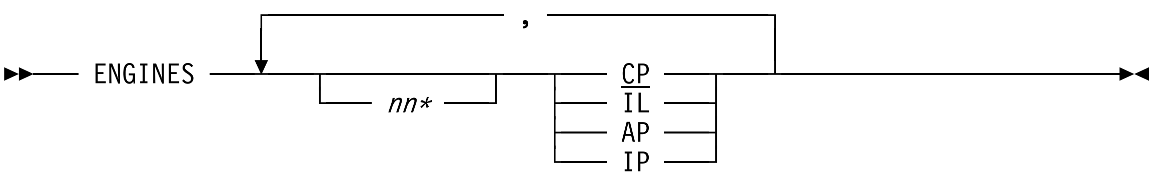
For detailed explanations on the interrelationship between ENGINES, MAXCPU and NUMCPU please see “Appendix B. Configuration of Emulated CPUs”.

8.60.2 Syntax

Descriptive

`ENGINES [nn*] {CP | IL | AP | IP} [, ... ]`

Diagram



8.60.3 Parameter

nn*	This is an optional repeat count.
CP	Specifies a processor engine of type CP. This is the default.
IL	Specifies a processor engine of type IL.
AP	Specifies a processor engine of type AP.
IP	Specifies a processor engine of type IP.

8.60.4 Examples

Example 1:

Specify two engines of type CP, one engine of type IP and one engine of type AP.

```
HHC00013I Herc command: 'engines 2*cp,ip,ap'
HHC00827I Processor CP00: engine 00 type 0 set: 'CP'
HHC00827I Processor CP01: engine 01 type 0 set: 'CP'
HHC00827I Processor IP02: engine 02 type 5 set: 'IP'
HHC00827I Processor AP03: engine 03 type 2 set: 'AP'
```

Figure 125: ENGINES command

8.61 EXEC (Execute a REXX script)

8.61.1 Function

The EXEC console command executes a Rexx script. EXEC invokes Rexx within the Hercules process to execute the Rexx script and commands will be executed synchronously within the Hercules process.

The EXEC command is only available if Hercules is built with Rexx support (see the Hercules “Installation Guide” for details on how to activate Rexx support and the Hercules “General Information” manual for more details about the Rexx support). Chapter 14 (“REXX Support”) in this manual contains additional information about using Rexx with Hercules.

8.61.2 Syntax

Descriptive

EXEC [COMMAND | SUBROUTINE] *rexx* [argument [argument ...]]

Diagram

```
►►── EXEC ──┬──────────────┬── rexx ──┬──────────────────┬──►◄
            ├── COMMAND ───┤          │ ┌──────────────┐ │
            └── SUBROUTINE ┘          └─▼── argument ──┴─┘
```

8.61.3 Parameter

COMMAND Specifies command style for passing arguments to a Rexx script. ‘COMMAND’ may be abbreviated as ‘COM’. If not specified then the argument passing style is determined by the Rexx mode settings (form the REXX system parameter or console command).

SUBROUTINE Specifies subroutine style for passing arguments to a Rexx script. ‘SUBROUTINE’ may be abbreviated as ‘SUB’. If not specified then the argument passing style is determined by the Rexx mode settings (form the REXX system parameter or console command).

rexx This is the name (and optionally the path) of a Rexx script.

If the name of the Rexx to be executed is given without a pathname, the script is searched within the path that has been defined with the REXX command or the default path if no path has been set for the Rexx environment (see the REXX console command for more details).

If a path is specified in the EXEC command, then the Rexx will not be searched in the defined path of the Rexx environment.

argument These are the arguments (separated by spaces) to be passed to the Rexx script.

8.61.4 Examples

Example 1:

Execute Rexx script “testrexx.rexx” with arguments ‘arg1’, ‘arg2’ and ‘arg3’.

```
HHC00013I Herc command: 'exec testrexx arg1 arg2 arg3’
testrexx  started
testrexx  version . . . .  : REXX-Regina_3.6(MT) 5.00 31 Dec 2011
testrexx  source  . . . .  : WIN64 COMMAND d:\mvs\conf\testrexx.REX
testrexx  hostenv . . . .  : HERCULES
testrexx  date  . . . . .  : 6 May 2012
testrexx  time  . . . . .  : 02:39:19
testrexx  arguments . . .  : arg1 arg2 arg3
testrexx  Hercules version : 3.08.0
testrexx  RC environment . : D:\MVS\CONF\Hercules_Test.rc
HHC02208I Uptime 04:02:33
testrexx  ended
```

Figure 126: EXEC command

The executed REXX script from example 1 is the following:

```
/* REXX */
parse arg parms

parms = space(parms)
argc  = words(parms)

parse version ver
parse source src

env = address()

parse var src . . cmd
who = filespec("n",cmd)
parse var who who "." .

say who " started"
say who " version . . . .  :" ver
say who " source  . . . .  :" src
say who " hostenv . . . .  :" env


say who " date  . . . . .  :" date()
say who " time  . . . . .  :" time()
```

```
if parms = ""
then do
     say who " arguments . . .  : no arguments given"
     ret = 0
     end
else do
     say who " arguments . . .  :" parms
     ret = parms
     end

Say who " Hercules version :" value('version',,'SYSTEM')
Say who " RC environment . :" value('HERCULES_RC',,'ENVIRONMENT')

address hercules 'uptime'

say who " ended"

exit ret
```

8.62 EXIT (Terminate the emulator)

8.62.1 Function

The EXIT command (see also the QUIT command) initiates the Hercules shutdown. It terminates all threads, detaches all channels and devices and releases the configuration. Finally it terminates the emulator. If the guest OS has enabled “Signal Shutdown” then a signal shutdown request is sent to the guest OS and termination will begin after the guest OS has shutdown.

The EXIT command acts different depending on how Hercules was built. If Hercules was not compiled with the option 'OPTION_SHUTDOWN_CONFIRMATION' then the command acts as described above.

If Hercules however was built with option 'OPTION_SHUTDOWN_CONFIRMATION' then the following special processing for terminating the emulator takes place.

If EXIT is entered and the command level is not set to 'developer', 'debug' or 'all' then EXIT will first check that all online CPUs are stopped. If any CPU is not in the stopped state message HHC00069I is displayed indicating the number of CPUs still running.

Message HHC02266A follows that prompts for confirmation by entering a second EXIT command within a certain time period (default 10 seconds) to start termination of the emulator. If the time period has expired then the process starts over. This is to prevent an inadvertent shutdown of Hercules while still a guest OS is running.

The time period for the second EXIT command can be set to another value by using the 'QUITMOUT' console command or system parameter.

If all processors are stopped or the command level is set to "developer", "debug" or "all" then quit stops Hercules immediately. 'EXIT FORCE' will also terminate the emulator immediately without any further checks.

8.62.2 Syntax

Descriptive

```
EXIT [FORCE]
```

Diagram

```
►►── EXIT ──┬─────────┬──────────────────────────────►◄
            └─ FORCE ─┘
```

8.62.3 Parameter

FORCE Terminate the emulator immediately.

8.62.4 Examples

Example 1:

Initiate the Hercules shutdown.

```
HHC00013I Herc command: 'exit'
HHC00069I Guest is not quiesced; there are 8 CPUs active
HHC02266A Reenter command 'exit' again within 10 seconds to execute
HHC00013I Herc command: 'exit'
HHC01420I Begin Hercules shutdown
HHC01421I Releasing configuration
.
several lines not displayed
.
HHC01422I Configuration release complete
HHC01423I Calling termination routines
.
several lines not displayed
.
HHC02103I Logger: logger thread terminating
HHC01412I Hercules terminated
```

Figure 127: EXIT command

Example 2:

Initiate an immediate Hercules shutdown.

```
HHC00013I Herc command: 'exit force'
HHC01420I Begin Hercules shutdown
HHC01421I Releasing configuration
.
several lines not displayed
.
HHC01422I Configuration release complete
HHC01423I Calling termination routines
.
several lines not displayed
.
HHC02103I Logger: logger thread terminating
HHC01412I Hercules terminated
```

Figure 128: EXIT FORCE command

8.63 EXT (Generate external interrupt)

8.63.1 Function

The EXT command generates an external interrupt (the virtual INTERRUPT key is pressed).

8.63.2 Syntax

Descriptive

```
EXT
```

Diagram

```
►►── EXT ──────────────────────────────────────────────────────────►◄
```

8.63.3 Parameter

None.

8.63.4 Examples

Example 1:

Generate an external interrupt.

```
HHC00013I Herc command: 'ext'
HHC02228I Key 'interrupt' pressed
```

Figure 129: EXT command

8.64 F{+/-} (Mark frames usable or unusable)

8.64.1 Function

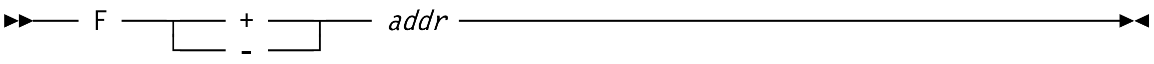
The F{+/-} command is used to mark storage frames as usable or unusable.

8.64.2 Syntax

Descriptive

`F{+ | -} `*`addr`*

Diagram



8.64.3 Parameter

+	Mark storage frame as usable.
-	Mark storage frame as unusable.
addr	The address of the storage frame that has to be marked usable or unusable.

8.64.4 Examples

Example 1:

Mark frame at address x'01000000' as unusable.

```
HHC00013I Herc command: 'f- 01000000'
HHC02204I frame 01000000 set to unusable
```

Figure 130: F{-} command

8.65 FCB (Display current FCB or load new FCB image)

8.65.1 Function

The FCB command is used to load a new FCB image or set the LPI (lines per inch) and LPP (lines per page) settings for a given 1403 or 3211 printer. Without arguments the FCB command displays the current settings for the printer.

To change settings, the printer must be stopped first ("STOP devaddr") and then started again ("START devaddr").

8.65.2 Syntax

Descriptive

FCB *devicenum* [LPI=*lpi*] [LPP=*lpp*]

  [FCB=*l1:c1,l2:c2,l3:c3, ... ,l10:c10,l11:c11,l12:c12*]

Diagram

```
►►── FCB ── devicenum ─┬──────────────┬─┬──────────────┬────────►
                       └── LPI=lpi ───┘ └── LPP=lpi ───┘
►─┬──────────────────────────────────────────────────────────┬──►◄
  └── FCB=l1:c1,l2:c2,l3:c3, ... ,l10:c10,l11:c11,l12:c12pi ─┘
```

8.65.3 Parameter

devicenum This is the device number of the printer for which the current settings have to be listed or changed.

LPI= Specifies the number of lines per inch. The value of *lpi* must be 6 or 8.

LPP= Specifies the number of lines per page. The value of *lpp* must be numeric but is not otherwise checked. Any number of lines per page is allowed.

FCB= FCB= specifies an new FCB image to use for the printer. The argument must be given in the form *l1:c1, ...,;l12:c12* where 'l' and 'c' are both numeric. The value of 'l' is the line number and the value of 'c' is the assigned channel. There is a maximum of 12 'l:c' pairs allowed.

8.65.4 Examples

Example 1:

Display the current FCB for the printer with device address 000E.

```
HHC00013I Herc command: 'fcb 000e'
HHC02210I 0:000E lpi=6 index=0 lpp=66 fcb=1:1,7:2,13:3,19:4,...,43:8,49:10,55:11,61:12,63:9
```

Figure 131: FCB command (display current FCB)

Example 2:

Stop the printer with device address 000E. Set lines per inch to 6, lines per page to 66 and load a new FCB image for the printer. Then start the printer again.

```
HHC00013I Herc command: 'stop 000e'
HHC02214I 0:000E device stopped
HHC00013I Herc command: 'fcb 000e lpi=6 lpp=66 fcb=1:1,6:2,12:3,18:4,...,48:9,54:10,60:11,66:12'
HHC02210I 0:000E lpi=6 index=0 lpp=66 fcb=1:1,6:2,12:3,18:4,...,48:9,54:10,60:11,66:12
HHC00013I Herc command: 'start 000e'
HHC02214I 0:000E device started
```

Figure 132: FCB command (load new FCB image)

8.66 FPC (Display or alter floating point control register)

8.66.1 Function

The FPC command displays or alters the actual content of the floating point control register. Enter FPC by itself to display the register value without altering it.

8.66.2 Syntax

Descriptive

FPC [*xxxxxxxx*]

Diagram

```
►►── FPC ──┬──────────────┬──────────────────────────────────►◄
           └── xxxxxxxx ──┘
```

8.66.3 Parameter

xxxxxxxx	This is the register value in hexadecimal (1-8 hex digits).

8.66.4 Examples

Example 1:

Display the floating point control register.

```
HHC00013I Herc command: 'fpc'
HHC02276I Floating point control register: 00000000
```

Figure 133: FPC command (display value)

Example 2:

Alter the floating point control register.

```
HHC00013I Herc command: 'fpc 00000103'
HHC02276I Floating point control register: 00000103
```

Figure 134: FPC command (alter value)

8.67 FPR (Display or alter floating point registers)

8.67.1 Function

The FPR command displays or alters the actual contents of the floating point registers (basic format). If the AFP bit is set in Control Register 0 then additional floating point registers are displayed (extended format). Enter FPR by itself to display the register values without altering them.

8.67.2 Syntax

Descriptive

FPR [*nn*=xxxxxxxxxxxxxxxx]

Diagram

```
►►── FPR ──┬──────────────────────┬──────────────────────►◄
           └── nn=xxxxxxxxxxxxxxxx ─┘
```

8.67.3 Parameter

nn	This specifies the register number (0,2,4,6 or 0-15, depending on the Control Register 0 AFP bit) to be altered.
xxxxxxxxxxxxxxxx	This is the register value in hexadecimal (1-16 hex digits for 64-bit registers).

8.67.4 Examples

Example 1:

Display the floating point registers.

```
HHC00013I Herc command: 'fpr'
HHC02270I Floating point registers
HHC02270I CP00: FPR0=0000000000000000 FPR2=0000000000000000
HHC02270I CP00: FPR4=0000000000000000 FPR6=0000000000000000
```

Figure 135: FPR command (display value)

Example 2:

Alter the floating point register 04.

```
HHC00013I Herc command: 'fpr 04=000000000000ffff'
HHC02270I Floating point registers
HHC02270I CP00: FPR0=0000000000000000 FPR2=0000000000000000
HHC02270I CP00: FPR4=000000000000FFFF FPR6=0000000000000000
```

Figure 136: FPR command (alter value)

8.68 G (Turn off instruction stepping and start all CPUs)

8.68.1 Function

The G command turns off the instruction stepping and starts all the CPUs. This command has also to be used to continue normal processing after stopping the instruction stepping with the S- command.

8.68.2 Syntax

<u>**Descriptive**</u>

```
G
```

<u>**Diagram**</u>

```
►►── G ──────────────────────────────────────────────────────►◄
```

8.68.3 Parameter

None.

8.68.4 Examples

Example 1:

Stop the instruction stepping and start the CPU.

```
HHC00013I Herc command: 'g'
```

Figure 137: G command

8.69 GPR (Display or alter general purpose registers)

8.69.1 Function

The GPR command displays or alters the actual content of the general purpose registers. Enter GPR by itself to display the register values without altering them.

8.69.2 Syntax

Descriptive

GPR [*nn*=*xxxxxxxx* | *nn*=*xxxxxxxxxxxxxxxx*]

Diagram

```
►►── GPR ──┬──────────────────────────┬──────────────────────────────►◄
           ├───── nn=xxxxxxxx ────────┤
           └── nn=xxxxxxxxxxxxxxxx ───┘
```

8.69.3 Parameter

nn This specifies the optional register number (00-15) to be altered.

xxxxxxxx This is the register value in hexadecimal (1-8 hex digits for 32-bit registers, 1-16 hex digits for 64-bit registers).

8.69.4 Examples

Example 1:

Display the general purpose registers.

```
HHC00013I Herc command: 'gpr'
HHC02269I General purpose registers
HHC02269I CP00: GR00=00000000 GR01=00000000 GR02=0252E12C GR03=02501EB0
HHC02269I CP00: GR04=02502400 GR05=00000C00 GR06=01379E76 GR07=00000000
HHC02269I CP00: GR08=0248EF58 GR09=00000000 GR10=0252EF7F GR11=0252DF80
HHC02269I CP00: GR12=813799C8 GR13=0252DF80 GR14=81379D22 GR15=00000C00
```

Figure 138: GPR command (display general purpose registers)

Example 2:

Alter general purpose register number 15.

```
HHC00013I Herc command: 'gpr 15=ffffffff'
HHC02269I General purpose registers
HHC02269I CP00: GR00=00000000 GR01=00000000 GR02=0252E12C GR03=02501EB0
HHC02269I CP00: GR04=02502400 GR05=00000C00 GR06=01379E76 GR07=00000000
HHC02269I CP00: GR08=0248EF58 GR09=00000000 GR10=0252EF7F GR11=0252DF80
HHC02269I CP00: GR12=813799C8 GR13=0252DF80 GR14=81379D22 GR15=00000C00
```

Figure 139: GPR command (alter general purpose register)

8.70 HAO (Hercules Automatic Operator)

8.70.1 Function

The Hercules Automatic operator (HAO) feature is a facility that allows to automatically issue console commands in response to certain messages being issued. To use the HAO facility it is necessary to define a rule, consisting of a target and an associated command.

The Hercules Automatic Operator is only for those messages issued by Hercules to its console. It cannot be used for messages issued from the guest operating system.

The current implementation limits the total number of defined rules to 64. There is currently no way to define a command whose arguments varies based on actual message text. All of the defined rules are checked for a match each time Hercules issues a message, there is no way at this time to stop the processing of subsequent rules.

8.70.2 Syntax

Descriptive

HAO *command* [*operands*]

where *command* can be:

TGT *target*

CMD *consolecmd*

DEL *nn*

CLEAR

LIST [*nn*]

Diagram

```
►►── HAO ──┤ command ├──────────────────────────────────────────►◄

   where command can be:

├── TGT ── target ─────────────────────────────────────────────┤

├── CMD ── consolecmd ─────────────────────────────────────────┤

├── DEL ── nn ─────────────────────────────────────────────────┤
```

```
├── CLEAR ──────────────────────────────────────────────┤

├── LIST ──┬──────────┬─────────────────────────────────┤
           └── nn ────┘
```

8.70.3 Parameter

TGT	This is the keyword to define a new rule (pattern).
target	Specifies the rule (pattern) to react on. The target is a regular expression pattern which is matched against the text of the messages that Hercules issues.
CMD	This is the keyword to specify a command for a previously defined rule.
consolecmd	Specifies the command to be executed if a target rule matches. The associated command must be a valid Hercules console command.
LIST	List all rules/commands or list only rule/command at index *nn*.
DEL	Delete rule and command at index *nn*.
nn	Number of the index to be listed or deleted.
CLEAR	Specifies to delete all defined rules. This stops the Automatic Operator.

8.70.4 Examples

Example 1:

Define target rule: Check for message HHC01600E ("Unknown Hercules command").

```
HHC00013I Herc command: 'hao tgt hhc01600e'
HHC00077I The 'target' was placed at index 0
```

Figure 140: HAO command (define target rule).

Example 2:

Define command for target rule: Issue "?" command (List all valid commands).

```
HHC00013I Herc command: 'hao cmd help'
HHC00077I The 'command' was placed at index 0
```

Figure 141: HAO command (define command)

Example 3:

List all defined rules.

```
HHC00013I Herc command: 'hao list'
HHC00087I The defined Hercules Automatic Operator rule(s) are:
HHC00088I Index 00: target 'hhc01600e' -> command 'help'
HHC00082I 1 rule(s) displayed
```

Figure 142: HAO command (list defined rules)

Example 4:

Delete rule at index 0.

```
HHC00013I Herc command: 'hao del 0'
HHC00086I Rule at index 0 successfully deleted
```

Figure 143: HAO command (delete rule)

Example 5:

Delete all rules (stop Hercules Automatic Operator).

```
HHC00013I Herc command: 'hao clear'
HHC00080I All HAO rules are cleared
```

Figure 144: HAO command (delete all rules)

Example 6:

Sample of automatic execution of a command (triggered via the use of an unknown console command, as defined in examples 1 and 2).

```
HHC00013I Herc command: 'offload'
HHC01600E Unknown herc command 'offload', enter 'help' for a list of valid commands
HHC00007I Previous message from function ProcessPanelCommand() at cmdtab.c[286]
HHC00081I Match at index 00, executing command 'help'
HHC00013I Herc command: 'help'
HHC01602I Command      Description
HHC01602I -------      ----------------------------------------------
HHC01602I help         list all commands / command specific help
HHC01602I ?            alias for help
HHC01602I cmdlevel     Display/Set current command group
HHC01602I cmdlvl       Alias for cmdlevel
.
.
.
```

Figure 145: HAO fired command

8.71 HELP (List all commands / command specific help)

8.71.1 Function

The “help” command without any options will display a sorted list of all commands matching the current command level with a short description. It is possible to specify a partial command name followed by an asterisk (“*”) to get a list of all commands matching that partial command name. For example ‘help msg*’ will list all commands beginning with ‘msg’ and matching the current command level.

If the list is enterd with a full command name it will display a long form of help information associated with that command if the command is available for the current command level. The list provided by the command without options shows with an asterisk in front of the description if there is additional information available.

The displayed help text may be limited to explaining the general format of the command and its various required or optional parameters and is not meant to replace the appropriate manual.

8.71.2 Syntax

Descriptive

HELP [*command* | *cmd**]

Diagram

```
►►── HELP ──┬────────────────┬──────────────────────────────►◄
            ├─── command ────┤
            └─── cmd* ───────┘
```

8.71.3 Parameter

command The Hercules console command to which additional information is desired.

cmd* Provides a list of all commands beginning with the partial command name ‘cmd’ and matching the current command level.

8.71.4 Examples

Example 1:

List all valid console commands matching the current command level.

```
HHC00013I Herc command: 'help'
HHC01603I
HHC01602I Command      Description
```

```
HHC01602I -------      ----------------------------------------------
HHC01602I !message    *SCP priority messsage
HHC01602I #            Silent comment
HHC01602I *            Loud comment
HHC01602I .reply      *SCP command
HHC01602I ?            alias for help
HHC01602I aea          Display AEA tables
HHC01602I aia          Display AIA fields
HHC01602I alrf         Command deprecated: Use "archlvl"  instead
HHC01602I ar           Display access registers
HHC01602I archlvl     *Set Architecture Level
HHC01602I archmode     Alias for archlvl
HHC01602I asn_and_l    Command deprecated: Use "archlvl" instead
HHC01602I attach      *Configure device
HHC01602I auto_scsi   *Command deprecated - Use "SCSIMOUNT"
HHC01602I autoinit    *Display/Set automatic create empty tape file switch
HHC01602I automount   *Display/Update allowable tape automount directories
HHC01602I b           *Set breakpoint
HHC01602I b+           Set breakpoint
HHC01602I b-          *Delete breakpoint
.
several lines not displayed
.
HHC01602I s{+/-}dev    Turn CCW stepping on/off
HHC01602I t           *Instruction trace
HHC01602I t+          *Instruction trace on
HHC01602I t-           Turn off instruction tracing
HHC01602I t?          *Instruction trace query
HHC01602I timerint    *Display or set timers update interval
HHC01602I tlb          Display TLB tables
HHC01602I toddrag      Display or set TOD clock drag factor
HHC01602I todprio      Set/Display todprio parameter
HHC01602I traceopt    *Instruction trace display options
HHC01602I tt32        *Control/query CTCI-WIN functionality
HHC01602I tzoffset     Set tzoffset parameter
HHC01602I t{+/-}CKD    Turn CKD_KEY tracing on/off
HHC01602I t{+/-}dev    Turn CCW tracing on/off
HHC01602I u            Disassemble storage
HHC01602I uptime       Display how long Hercules has been running
HHC01602I v           *Display or alter virtual storage
HHC01602I version      Display version information
HHC01602I xpndsize    *Define/Display xpndsize parameter
HHC01602I yroffset     Set yroffset parameter
HHC01603I
HHC01610I  (*)  More help available.
```

Figure 146: HELP command

Example 2:

Display a list of commands beginning with ‘cpu’ and matching the current command level.

```
HHC00013I Herc command: 'help cpu*'
HHC01603I
HHC01602I Command      Description
HHC01602I -------      -----------------------------------------------
HHC01602I cpu         *Define target cpu for panel display and commands
HHC01602I cpuidfmt     Set format 0/1 STIDP generation
HHC01602I cpumodel     Set CPU model number
HHC01602I cpuprio      Set/Display cpuprio parameter
HHC01602I cpuserial    Set CPU serial number
HHC01602I cpuverid     Set CPU verion number
HHC01603I
HHC01610I  (*)  More help available.
```

Figure 147: HELP CPU* command

Example 3:

Display additional help text for the MAINSIZE command.

```
HHC00013I Herc command: 'help mainsize'
HHC01603I
HHC01602I Command      Description
HHC01602I -------      -----------------------------------------------
HHC01602I mainsize    *Define/Display mainsize parameter
HHC01603I
HHC01603I Format: mainsize [ mmmm | nnnS [ lOCK | unlOCK ] ]
HHC01603I         mmmm    - define main storage size mmmm Megabytes
HHC01603I
HHC01603I         nnnS    - define main storage size nnn S where S is the multiplier
HHC01603I                   B = no multiplier
HHC01603I                   K = 2**10 (kilo/kibi)
HHC01603I                   M = 2**20 (mega/mebi)
HHC01603I                   G = 2**30 (giga/gibi)
HHC01603I                   T = 2**40 (tera/tebi)
HHC01603I                   P = 2**50 (peta/pebi)
HHC01603I                   E = 2**60 (exa/exbi)
HHC01603I
HHC01603I         lOCK    - attempt to lock storage (pages lock by host OS)
HHC01603I         unlOCK  - leave storage unlocked (pagable by host OS)
HHC01603I
HHC01603I       (none)    - display current mainsize value
HHC01603I
HHC01603I  Note: Multipliers 'T', 'P', and 'E' are not available on 32bit machines
```

Figure 148: HELP MAINSIZE command

8.72 HERC (Send Hercules command)

8.72.1 Function

The HERC command sends a command in any CMDTGT mode to the Hercules Emulator. See also CMDTGT, SCP and PSCP commands.

8.72.2 Syntax

Descriptive

HERC [*cmd*]

Diagram

```
►►── HERC ──┬───────────┬──────────────────────────────────────►◄
            └── cmd ────┘
```

8.72.3 Parameter

cmd	This is the command to be sent to the Hercules Emulator.

8.72.4 Examples

Example 1:

Send a Hercules command in SCP or PSCP command target mode..

```
HHC00013I Herc command: 'herc psw'
HHC02278I Program status word: 040C6000 815E43EA
HHC02300I sm=04 pk=0 cmwp=C as=ar cc=2 pm=0 am=31 ia=15E43D8
```

Figure 149: HERC command

8.73 HERCLOGO (Read new Hercules logo file)

8.73.1 Function

The HERCLOGO command loads a new logo file for 3270 terminal sessions. The logo file defines a welcome screen that is presented when a TN3270 terminal connects to a Hercules 3270 device. For details on how to code the logo file see chapter 10. If no filename is specified then the built-in logo is used instead.

8.73.2 Syntax

Descriptive

HERCLOGO [*filename*]

Diagram

```
►►── HERCLOGO ──┬──────────────┬─────────────────────────────────►◄
                └── filename ──┘
```

8.73.3 Parameter

filename The name (and optionally path) of a logo text file. If no path is specified the logo file is first searched in the current working directory and second in the directory where the Hercules executable resides. If no filename is specified the built-in logo is used instead.

8.73.4 Examples

Example 1:

Load a new logo file.

```
HHC00013I Herc command: 'herclogo d:\hercules\conf\herclogo.txt'
```

Figure 150: HERCLOGO command

8.74 HERCPRIO (Display or set Hercules process priority)

8.74.1 Function

The HERCPRIO console command is used to change the process priority for Hercules. See section 5.82 for details on process and thread priorities. Given without an argument the HERCPRIO command displays the current Hercules process priority.

Caution: HERCPRIO should not have a higher dispatching priority than the TOD clock and timer thread.

8.74.2 Syntax

Descriptive

HERCPRIO [*nn*]

Diagram

```
►►── HERCPRIO ──┬────────┬──────────────────────────────────────►◄
                └── nn ──┘
```

8.74.3 Parameter

nn This value specifies the process priority for Hercules. For details on the priority values see section 5.82 (“Process and Thread Priorities”).

8.74.4 Examples

Example 1:

Set the Hercules process priority to 0.

```
HHC00013I Herc command: 'hercprio 0'
HHC02204I hercprio       set to 0
```

Figure 151: HERCPRIO command

8.75 HST (History of commands)

8.75.1 Function

The HST command displays a list of the last commands entered from the command line. Usually every issued command is automatically added to the command recall history list. To prevent a command from appearing in the command recall history list it must be prefixed with a hyphen ('-') to make it a silent non-echoing command.

8.75.2 Syntax

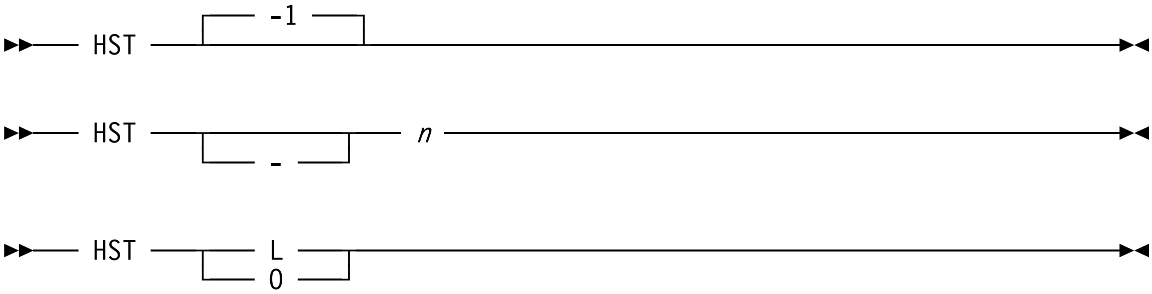
Descriptive

HST [-1]

HST [-]*n*

HST {L | 0}

Diagram



8.75.3 Parameter

-1 HST -1 is the same as HST without argument and retrieves the last entered command.

[-]*n* If n is a positive number then HST retrieves the n-th command from the list. If n is a negative number then HST retrieves the n-th last command from the list.

L or 0 HST L or HST 0 (number "0") displays a list of the last ten commands entered on the command line.

8.75.4 Examples

Example 1:

Display a list of the last ten commands entered.

```
HHC00013I Herc command: 'hst 0'
HHC02273I Index 30: 'aia'
HHC02273I Index 31: 'ar'
HHC02273I Index 32: 'clocks'
HHC02273I Index 33: 'cr'
HHC02273I Index 34: 'maxrates 60'
HHC02273I Index 35: 'help ds'
HHC02273I Index 36: 'gpr'
HHC02273I Index 37: 'http stop'
HHC02273I Index 38: 'http port 8080 noauth'
HHC02273I Index 39: 'http start'
```

Figure 152: HST command (display command recall history list)

Example 2:

Prevent a command from being added to the command recall history list and display a list of the last ten entered commands.

```
HHC00013I Herc command: 'psw'
HHC02278I Program status word: 0000000000000000 0000000000000000
HHC02300I sm=00 pk=0 cmwp=0 as=pri cc=0 pm=0 am=24 ia=0

HHC00013I Herc command: 'hst 0'
HHC02273I Index 30: 'aia'
HHC02273I Index 31: 'ar'
HHC02273I Index 32: 'clocks'
HHC02273I Index 33: 'cr'
HHC02273I Index 34: 'maxrates 60'
HHC02273I Index 35: 'help ds'
HHC02273I Index 36: 'gpr'
HHC02273I Index 37: 'http stop'
HHC02273I Index 38: 'http port 8080 noauth'
HHC02273I Index 39: 'http start'
```

Figure 153: HST command (issue silent non-echoing command)

8.76 HTTP (Start, stop, modify and display HTTP server)

8.76.1 Function

The HTTP console command configures the HTTP server. Depending on the given arguments the port on which the HTTP server will listen and the authorization (if any) or the location of the HTTP server files can be specified. Additional arguments allow to start or to stop the HTTP server. Given without an argument the HTTP command displays the current state of the HTTP server.

8.76.2 Syntax

Descriptive

HTTP [START | STOP | ROOT *path* | PORT *port* {NOAUTH | AUTH *userid password*}]

Diagram

```
►►── HTTP ──┬──────────────────────────────────────────┬──────────►◄
            ├── START ─────────────────────────────────┤
            ├── STOP ──────────────────────────────────┤
            ├── ROOT ── path ──────────────────────────┤
            └── PORT ── port ──┬───────── NOAUTH ──────┤
                               └── AUTH userid password ┘
```

8.76.3 Parameter

START Start the HTTP server (if it is stopped).

STOP Stop the HTTP server (if it is started).

ROOT Keyword to specify the root directory of the HTTP server files. The HTTP root can only be set if the HTTP server is in the stopped state.

path The full path of the root directory where the HTTP server files reside. If this parameter is not specified the default value for Win32 builds of Hercules is the directory where the Hercules executables themselves reside. For non-Win32 builds it is the directory specified as the default package installation directory when the Hercules executables were built. This can vary depending on how the Hercules package was built, it is commonly "/usr/local/share/hercules".

PORT Keyword to specify the port on which the HTTP server will listen (including optional authorization information). The HTTP port and authorization information can only be set if the HTTP server is in the stopped state.

port The port number must be either 80 or within the range of 1024 to 65535 inclusive.

NOAUTH NOAUTH indicates that no userid and password are required to access the HTTP server.

AUTH AUTH indicates that a userid and a password are required to access the HTTP server. The userid and password have to be coded after the AUTH parameter.

userid The userid can be any valid string.

password The password can be any valid string.

8.76.4 Examples

Example 1:

Display the HTTP server status.

```
HHC00013I Herc command: 'http'
HHC01809I HTTP server is waiting for requests
HHC01811I HTTP server root directory D:/HERCULES/HTML/
HHC01808I HTTP server port is port=80 noauth
```

Figure 154: HTTP command (display HTTP server status)

Example 2:

Stop the HTTP server.

```
HHC00013I Herc command: 'http stop'
HHC01805I HTTP server signaled to stop
HHC00101I Thread id 0000097C, prio  0, name 'HTTP server' ended
```

Figure 155: HTTP command (stop HTTP server)

Example 3:

Set the HTTP server root directory to “D:\Hercules\html”.

```
HHC00013I Herc command: 'http root D:/Hercules/html'
HHC02204I httproot       set to D:/Hercules/html/
```

Figure 156: HTTP command (set HTTP server root directory)

Example 4:

Set the HTTP server port to 8081 with required authorization for access. The userid should be set to “hercuser” and the password to “hercpswd”.

```
HHC00013I Herc command: 'http port 8081 auth hercuser hercpswd'
HHC02204I port           set to port=8081 auth userid<hercuser> password<hercpswd>
```

Figure 157: HTTP command (set HTTP server port and authorization)

Example 5:

Start the HTTP server.

```
HHC00013I Herc command: 'http start'
HHC01807I HTTP server signaled to start
HHC00100I Thread id 00001344, prio  0, name 'HTTP server' started
HHC01802I Using HTTPROOT directory 'D:/Hercules/html/'
HHC01803I Waiting for HTTP requests on port 8081
```

Figure 158: HTTP command (start HTTP server)

8.77 I (Generate I/O attention interrupt for device)

8.77.1 Function

The I command generates an I/O attention interrupt for a certain device.

8.77.2 Syntax

Descriptive

I *device*

Diagram

►►── I ── *device* ─────────────────────────────────────►◄

8.77.3 Parameter

device The number of the device for which an I/O attention interrupt will be generated.

8.77.4 Examples

Example 1:

Generate an I/O attention interrupt for device 0148.

```
HHC00013I Herc command: 'i 000f'
HHC02230I 0:000F attention request raised
```

Figure 159: I command

8.78 ICOUNT (Display individual instruction counts)

8.78.1 Function

The ICOUNT command displays statistic information about how often every single machine instruction has been executed so far. By default the list sorted by the instruction code. An additional SORT option displays the instructions sorted by the usage count and percentage of usage

The command is only available if Hercules is built with the “OPTION_INSTRUCTION_COUNTING” option. This option is not activated by default because ICOUNT is a debugging command.

8.78.2 Syntax

Descriptive

```
ICOUNT [SORT | CLEAR]
```

Diagram

```
►►── ICOUNT ──┬──────────┬────────────────────────────────────►◄
              ├── SORT ──┤
              └── CLEAR ─┘
```

8.78.3 Parameter

SORT Displays the instructions sorted by the usage count and percentage of usage.

CLEAR Clears the counters and start counting from zero.

8.78.4 Examples

Example 1:

Display the individual instruction counts.

```
HHC00013I Herc command: 'icount'
HHC02292I Instruction count display:
HHC02292I Inst '04'   count        32688
HHC02292I Inst '05'   count      3695113
HHC02292I Inst '06'   count      1211081
HHC02292I Inst '07'   count      5523916
HHC02292I Inst '08'   count        30274
HHC02292I Inst '09'   count        46739
HHC02292I Inst '0A'   count        95360
```

```
HHC02292I Inst '0E'   count        20764
HHC02292I Inst '0F'   count          955
HHC02292I Inst '10'   count       113522
HHC02292I Inst '11'   count        38753
HHC02292I Inst '12'   count      5743006
HHC02292I Inst '13'   count       189520
HHC02292I Inst '14'   count       625578
HHC02292I Inst '15'   count       504659
HHC02292I Inst '16'   count       148266
HHC02292I Inst '17'   count       114403
HHC02292I Inst '18'   count     12153723
HHC02292I Inst '19'   count      4082623
HHC02292I Inst '1A'   count       586980
HHC02292I Inst '1B'   count      2519610
HHC02292I Inst '1C'   count         4505
HHC02292I Inst '1D'   count        25093
HHC02292I Inst '1E'   count      1237759
HHC02292I Inst '1F'   count      6595895
.
several lines not displayed
.
HHC02292I Inst 'B6'   count           78
HHC02292I Inst 'B7'   count        46856
HHC02292I Inst 'BA'   count      1518619
HHC02292I Inst 'BB'   count       584272
HHC02292I Inst 'BD'   count       464448
HHC02292I Inst 'BE'   count       538116
HHC02292I Inst 'BF'   count       877819
HHC02292I Inst 'D1'   count       234238
HHC02292I Inst 'D2'   count      2827438
HHC02292I Inst 'D3'   count        64695
HHC02292I Inst 'D4'   count         8923
HHC02292I Inst 'D5'   count       945073
HHC02292I Inst 'D6'   count       104993
HHC02292I Inst 'D7'   count       722277
HHC02292I Inst 'DC'   count         1641
HHC02292I Inst 'DD'   count         4296
HHC02292I Inst 'DE'   count           80
HHC02292I Inst 'F1'   count           43
HHC02292I Inst 'F2'   count          411
HHC02292I Inst 'F3'   count         1741
HHC02292I Inst 'F8'   count            2
```

Figure 160: ICOUNT command

Example 2:

Display the individual instruction counts sorted by usage count and percentage of usage.

```
HHC00013I Herc command: 'icount sort'
HHC02292I Sorted instruction count display:
HHC02292I Inst '47'   count     45073338 (19%)
HHC02292I Inst '58'   count     30958976 (13%)
HHC02292I Inst '91'   count     17490895 ( 7%)
HHC02292I Inst '50'   count     17148150 ( 7%)
HHC02292I Inst '18'   count     12193746 ( 5%)
HHC02292I Inst '41'   count      7487721 ( 3%)
HHC02292I Inst '1F'   count      6621520 ( 2%)
HHC02292I Inst '48'   count      6394646 ( 2%)
HHC02292I Inst '12'   count      5782677 ( 2%)
HHC02292I Inst '54'   count      5649330 ( 2%)
HHC02292I Inst '07'   count      5526616 ( 2%)
HHC02292I Inst '5E'   count      4976470 ( 2%)
HHC02292I Inst '19'   count      4096278 ( 1%)
HHC02292I Inst '96'   count      3834527 ( 1%)
HHC02292I Inst '95'   count      3765941 ( 1%)
HHC02292I Inst '94'   count      3755916 ( 1%)
.
several lines not displayed
.
HHC02292I Inst 'B20D' count         4941 ( 0%)
HHC02292I Inst '1C'   count         4591 ( 0%)
HHC02292I Inst 'B209' count         4442 ( 0%)
HHC02292I Inst 'DD'   count         4296 ( 0%)
HHC02292I Inst '5C'   count         3566 ( 0%)
HHC02292I Inst '57'   count         3391 ( 0%)
HHC02292I Inst '8D'   count         3127 ( 0%)
HHC02292I Inst 'F3'   count         1741 ( 0%)
HHC02292I Inst 'DC'   count         1641 ( 0%)
HHC02292I Inst '4E'   count         1339 ( 0%)
HHC02292I Inst '97'   count         1046 ( 0%)
HHC02292I Inst '0F'   count          955 ( 0%)
HHC02292I Inst '93'   count          699 ( 0%)
HHC02292I Inst '4F'   count          435 ( 0%)
HHC02292I Inst 'F2'   count          411 ( 0%)
HHC02292I Inst 'B206' count          179 ( 0%)
HHC02292I Inst 'B6'   count          114 ( 0%)
HHC02292I Inst 'DE'   count           80 ( 0%)
HHC02292I Inst 'F1'   count           43 ( 0%)
HHC02292I Inst 'B208' count           30 ( 0%)
HHC02292I Inst 'B202' count           15 ( 0%)
HHC02292I Inst 'B203' count            6 ( 0%)
HHC02292I Inst '28'   count            3 ( 0%)
HHC02292I Inst 'B200' count            2 ( 0%)
HHC02292I Inst 'F8'   count            2 ( 0%)
HHC02292I Inst '35'   count            1 ( 0%)
```

Figure 161: ICOUNT command (sorted)

8.79 IODELAY (Display or set I/O delay value)

8.79.1 Function

The IODELAY command displays the current setting or sets the amount of time in microseconds to wait after an I/O interrupt is ready to be set pending. The purpose of this parameter is to bypass a bug in the Linux/390 and z/Linux ‘dasd.c’ device driver. The problem is more apt to happen under Hercules than on a real machine, as Hercules may present an I/O interrupt sooner than a real machine.

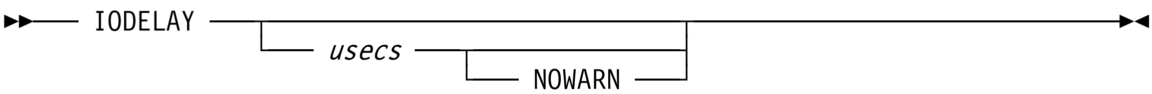
NOTE: OSTAILOR LINUX no longer sets IODELAY to 800 since the problem described above is no longer present in recent versions of the Linux kernel.

8.79.2 Syntax

<u>Descriptive</u>

`IODELAY [`*`usecs`*` [NOWARN]]`

<u>Diagram</u>



8.79.3 Parameter

usecs	Amount of time in microseconds to wait after an I/O interrupt is ready to be set pending.
NOWARN	If the IODELAY value is non-zero a warning message will be issued unless `NOWARN` is specified.

8.79.4 Examples

Example 1:

Display the current IODELAY value.

```
HHC00013I Herc command: 'iodelay'
HHC02203I iodelay       : 0
```

Figure 162: IODELAY command (display value)

Example 2:

Change the IODELAY value to 50 microseconds.

```
HHC00013I Herc command: 'iodelay 50'
HHC02204I iodelay        set to 50
```

Figure 163: IODELAY command (set value)

8.80 IPENDING (Display pending interrupts)

8.80.1 Function

The IPENDING command displays the pending interrupts for each CPU and some lock information.

8.80.2 Syntax

Descriptive

```
IPENDING
```

Diagram

```
►►── IPENDING ──────────────────────────────────────────────►◄
```

8.80.3 Parameter

None

8.80.4 Examples

Example 1:

Display pending interrupts.

```
HHC00013I Herc command: 'ipending'
HHC00850I Processor CP00: CPUint=00000001 (State:00000081)&(Mask:CF00EC4B)
HHC00851I Processor CP00: interrupt not pending
HHC00852I Processor CP00: I/O interrupt not pending
HHC00853I Processor CP00: clock comparator not pending
HHC00854I Processor CP00: CPU timer not pending
HHC00855I Processor CP00: interval timer pending
HHC00856I Processor CP00: ECPS vtimer not pending
HHC00857I Processor CP00: external call not pending
HHC00858I Processor CP00: emergency signal not pending
HHC00859I Processor CP00: machine check interrupt not pending
HHC00860I Processor CP00: service signal not pending
HHC00861I Processor CP00: mainlock held: no
HHC00862I Processor CP00: intlock held: no
HHC00863I Processor CP00: waiting for intlock: no
HHC00864I Processor CP00: lock not held
HHC00865I Processor CP00: connected to channelset 0000
HHC00866I Processor CP00: state STARTED
```

```
HHC00867I Processor CP00: instcount 111019557
HHC00868I Processor CP00: siocount 5181
HHC00869I Processor CP00: psw 070E000000000000
HHC00850I Processor CP01: CPUint=80000000 (State:80000001)&(Mask:8000000A)
HHC00851I Processor CP01: interrupt pending
HHC00852I Processor CP01: I/O interrupt not pending
HHC00853I Processor CP01: clock comparator not pending
HHC00854I Processor CP01: CPU timer not pending
HHC00855I Processor CP01: interval timer not pending
HHC00856I Processor CP01: ECPS vtimer not pending
HHC00857I Processor CP01: external call not pending
HHC00858I Processor CP01: emergency signal not pending
HHC00859I Processor CP01: machine check interrupt not pending
HHC00860I Processor CP01: service signal not pending
HHC00861I Processor CP01: mainlock held: no
HHC00862I Processor CP01: intlock held: no
HHC00863I Processor CP01: waiting for intlock: no
HHC00864I Processor CP01: lock not held
HHC00865I Processor CP01: connected to channelset 0001
HHC00866I Processor CP01: state STOPPED
HHC00867I Processor CP01: instcount 0
HHC00868I Processor CP01: siocount 0
HHC00869I Processor CP01: psw 0000000000000000
HHC00870I config mask 00000003 started mask 00000001 waiting mask 00000001
HHC00871I syncbc mask 00000000
HHC00872I signaling facility not busy
HHC00873I TOD lock not held
HHC00874I mainlock not held; owner ffff
HHC00875I intlock not held; owner ffff
HHC00881I I/O interrupt queue: (NULL)
```

Figure 164: IPENDING command

8.81 IPL (IPL Normal from device xxxx)

8.81.1 Function

This console command performs the Initial Program Load (IPL) Normal manual function from the given device. The IPL command may also be used to perform a load from CD-ROM or server. For example if a standard SUSE S/390 Linux distribution CD is loaded and mounted on /cdrom, this CD-ROM may then be IPL'ed using *"IPL /cdrom/suse.ins"*.

If the first operand is a device name (devnum is a 1- to 4-digit hexadecimal number), a CCW-type IPL is initiated from the indicated device number and SCLP disk I/O is disabled.

If the first operand is a filename a list-directed IPL is performed from the specified ".ins" file and SCLP disk I/O is enabled for the directory path where the ".ins" file is located.

An optional 'LOADPARM' keyword followed by a 1-8 character string can be used to set the LOADPARM prior to IPL.

An optional 'PARM' keyword followed by a string can also be passed to the IPL command processor. The string will be loaded into the low-order 32 bits of the general purpose registers (4 characters per register for up to 64 bytes). The PARM option behaves similarly to the VM IPL command.

An optional 'CLEAR' keyword will initiate a Load Clear manual control function, prior to starting an IPL.

8.81.2 Syntax

Descriptive

```
IPL {devnum | filename} [LOADPARM iplparm | PARM parmstring] [CLEAR]
```

Diagram

```
►►── IPL ──┬── devnum ───┬──┬───────────────────────┬─────────────────►
           └── filename ─┘  ├── LOADPARM iplparm ───┤
                            └── PARM parmstring ────┘

►──┬───────────┬──────────────────────────────────────────────────────►◄
   └── CLEAR ──┘
```

8.81.3 Parameter

devnum	The device address from which the emulator will IPL the system.
filename	The name of a ".ins" file to be loaded.
LOADPARM	This is the optional LOADPARM keyword.
iplparm	This is an 1-8 byte character string used to set the LOADPARM prior to the IPL.

The parameter is operating system dependent, consult the relevant operating system documentation for details.

PARM This is the optional PARM keyword.

parmstring The character string (max. 64 bytes) to be passed to the IPL command processor. The character string will be loaded into the low-order 32 bits of the GPRs (4 characters per register).

CLEAR Initiate a Load Clear manual control function, prior to starting an IPL.

8.81.4 Examples

Example 1:

IPL normal from device 0148.

```
HHC00013I Herc command: 'ipl 0148'
HHC01315I 0:00C2 CH: ccw 03000000 20000001=>040C0000 0001CEFA 00000000 00000000 ................
HHC01312I 0:00C2 CH: stat 0200, count 0001 ''
HHC01313I 0:00C2 CH: sense 40000000 00000000 00000000 00000000 00000000 00000000
HHC01314I 0:00C2 CH: sense INTREQ
HHC01315I 0:00C3 CH: ccw 03000000 20000001=>040C0000 0001CEFA 00000000 00000000 ................
HHC01312I 0:00C3 CH: stat 0200, count 0001 ''
HHC01313I 0:00C3 CH: sense 40000000 00000000 00000000 00000000 00000000 00000000
HHC01314I 0:00C3 CH: sense INTREQ
HHC01315I 0:00C4 CH: ccw 03000000 20000001=>040C0000 0001CEFA 00000000 00000000 ................
HHC01312I 0:00C4 CH: stat 0200, count 0001 ''
HHC01313I 0:00C4 CH: sense 40000000 00000000 00000000 00000000 00000000 00000000
HHC01314I 0:00C4 CH: sense INTREQ
HHC01315I 0:00C5 CH: ccw 03000000 20000001=>040C0000 0001CEFA 00000000 00000000 ................
HHC01312I 0:00C5 CH: stat 0200, count 0001 ''
HHC01313I 0:00C5 CH: sense 40000000 00000000 00000000 00000000 00000000 00000000
HHC01314I 0:00C5 CH: sense INTREQ
HHC01315I 0:0480 CH: ccw 03000000 20000001=>040C0000 0001CEFA 00000000 00000000 ................
HHC01312I 0:0480 CH: stat 0200, count 0001 ''
HHC01313I 0:0480 CH: sense 40220000 00C00300 00000000 00800100 010000FF FF000000
HHC01314I 0:0480 CH: sense INTREQ EOC WRI
HHC01315I 0:0481 CH: ccw 03000000 20000001=>040C0000 0001CEFA 00000000 00000000 ................
HHC01312I 0:0481 CH: stat 0200, count 0001 ''
HHC01313I 0:0481 CH: sense 40220000 00C00300 00000000 00800100 010000FF FF000000
HHC01314I 0:0481 CH: sense INTREQ EOC WRI
.
.
.
```

Figure 165: IPL command

8.82 IPLC (IPL Clear from device xxxx)

8.82.1 Function

The IPLC console command has been deprecated. Please use the IPL command with the CLEAR option instead. See the IPL command for details.

8.82.2 Syntax

See IPL console command.

8.82.3 Parameter

See IPL console command.

8.82.4 Examples

See IPL console command.

8.83 K (Display CCKD internal trace)

8.83.1 Function

The K command displays the internal CCKD trace entries. Because the default value for the CCKD trace entries is set to zero, the trace must first be enabled by setting a trace entry value. This can be done either with the system parameter or the console command "CCKD TRACE=n", where n specifies the number of available trace entries and can be in the range of 1 to 200000.

Please note that the K command can produce a lot of messages depending on the available trace entries.

8.83.2 Syntax

Descriptive

K

Diagram

```
►►── K ──────────────────────────────────────────────────────►◄
```

8.83.3 Parameter

None.

8.83.4 Examples

Example 1:

Display the current CCKD trace entries.

```
HHC00013I Herc command: 'k'
HHC00399I CCKD file: internal cckd trace
HHC00398I 02:07:32.759838 0:0148> start i/o file[0] bufcur 2588 cache[207]
HHC00398I 02:07:32.759842 0:0148> end i/o bufcur 2588 cache[207] waiters 0
HHC00398I 02:07:32.759848 0:0148> start i/o file[0] bufcur 2588 cache[207]
HHC00398I 02:07:32.759852 0:0148> read  trk   2589 (synchronous)
HHC00398I 02:07:32.759854 0:0148> 0 rdtrk     2589
HHC00398I 02:07:32.759857 0:0148> 0 rdtrk[208] 2589 syncio cache miss
HHC00398I 02:07:32.759868 0:0148> read  trk   2589 (asynchronous)
HHC00398I 02:07:32.759872 0:0148> 0 rdtrk     2589
HHC00398I 02:07:32.759875 0:0148> 0 rdtrk[208] 2589 cache miss
HHC00398I 02:07:32.759888 0:0148> 0 rdtrk[208] 2589 buf 0000000007F67CD0 len 19456
HHC00398I 02:07:32.759892 0:0148> trk[2589] read_trkimg
```

```
HHC00398I 02:07:32.759894 0:0148> file[0] l2[10,29] trk[2589] read_l2ent 0x4d08
HHC00398I 02:07:32.759898 0:0148> file[0] read_l2 10 active 0 10 2
HHC00398I 02:07:32.759901 0:0148> file[0] l2[10,29] trk[2589] read_l2ent 0x14c1f2f 6290 6290
HHC00398I 02:07:32.759904 0:0148> file[0] fd[16] read, off 0x14c1f2f len 6290
HHC00398I 02:07:32.759918 0:0148> 0 rdtrk[208] 2589 complete buf 0000000007F67CD0:0100560009
HHC00398I 02:07:32.759922 0:0148> uncompress comp 1 len 6290 maxlen 19456 trk 2589
HHC00398I 02:07:32.760082 0:0148> uncompress zlib newlen 13065 rc 0
HHC00398I 02:07:32.760085 0:0148> validating trk 2589 len 13065 0000560009 0056000900000008
HHC00398I 02:07:32.760091 0:0148> read  trk   2589 uncompressed len 13065
HHC00398I 02:07:32.760096 0:0148> end i/o bufcur 2589 cache[208] waiters 0
HHC00398I 02:07:32.760104 0:0148> start i/o file[0] bufcur 2589 cache[208]
HHC00398I 02:07:32.760112 0:0148> end i/o bufcur 2589 cache[208] waiters 0
HHC00398I 02:07:32.760118 0:0148> start i/o file[0] bufcur 2589 cache[208]
HHC00398I 02:07:32.760124 0:0148> end i/o bufcur 2589 cache[208] waiters 0
.
several lines not displayed
.
HHC00398I 02:07:32.760712 0:0148> end i/o bufcur 2589 cache[208] waiters 0
HHC00398I 02:07:32.760718 0:0148> start i/o file[0] bufcur 2589 cache[208]
HHC00398I 02:07:32.760722 0:0148> end i/o bufcur 2589 cache[208] waiters 0
HHC00398I 02:07:32.760728 0:0148> start i/o file[0] bufcur 2589 cache[208]
HHC00398I 02:07:32.760732 0:0148> read  trk   2590 (synchronous)
HHC00398I 02:07:32.760734 0:0148> 0 rdtrk     2590
HHC00398I 02:07:32.760737 0:0148> 0 rdtrk[209] 2590 syncio cache miss
HHC00398I 02:07:32.760748 0:0148> read  trk   2590 (asynchronous)
HHC00398I 02:07:32.760751 0:0148> 0 rdtrk     2590
HHC00398I 02:07:32.760754 0:0148> 0 rdtrk[209] 2590 cache miss
HHC00398I 02:07:32.760769 0:0148> 0 rdtrk[209] 2590 buf 0000000007F6C8E0 len 19456
HHC00398I 02:07:32.760773 0:0148> trk[2590] read_trkimg
HHC00398I 02:07:32.760775 0:0148> file[0] l2[10,30] trk[2590] read_l2ent 0x4d08
HHC00398I 02:07:32.760779 0:0148> file[0] read_l2 10 active 0 10 2
HHC00398I 02:07:32.760782 0:0148> file[0] l2[10,30] trk[2590] read_l2ent 0x14c37c1 4716 4716
HHC00398I 02:07:32.760785 0:0148> file[0] fd[16] read, off 0x14c37c1 len 4716
HHC00398I 02:07:32.760798 0:0148> 0 rdtrk[209] 2590 complete buf 0000000007F6C8E0:010056000a
HHC00398I 02:07:32.760803 0:0148> uncompress comp 1 len 4716 maxlen 19456 trk 2590
HHC00398I 02:07:32.760927 0:0148> uncompress zlib newlen 9813 rc 0
HHC00398I 02:07:32.760929 0:0148> validating trk 2590 len 9813 000056000a 0056000a00000008
HHC00398I 02:07:32.760935 0:0148> read  trk   2590 uncompressed len 9813
HHC00398I 02:07:32.760940 0:0148> end i/o bufcur 2590 cache[209] waiters 0
HHC00398I 02:07:32.760947 0:0148> start i/o file[0] bufcur 2590 cache[209]
HHC00398I 02:07:32.760954 0:0148> end i/o bufcur 2590 cache[209] waiters 0
HHC00398I 02:07:32.760960 0:0148> start i/o file[0] bufcur 2590 cache[209]
HHC00398I 02:07:32.760966 0:0148> end i/o bufcur 2590 cache[209] waiters 0
.
.
.
```

Figure 166: K command

8.84 KD (Clear held messages)

8.84.1 Function

The KD command is an alias for 'MSGHLD CLEAR' and is used to release all held messages without having to wait for the timeout period to expire.

8.84.2 Syntax

Descriptive

```
KD
```

Diagram

```
►►── KD ──────────────────────────────────────────────────►◄
```

8.84.3 Parameter

None.

8.84.4 Examples

Example 1:

Release all held messages.

```
HHC00013I Herc command: 'kd'
HHC02226I Held messages cleared
```

Figure 167: KD command

8.85 LDMOD (Load a module)

8.85.1 Function

The LDMOD command instructs the Hercules Dynamic Loader to load an additional module. The default search order is within the Hercules directory and in the default DLL search path.

8.85.2 Syntax

Descriptive

LMOD *module* [*module* [*module* ...]]

Diagram

```
            ┌──────────────┐
            ▼              │
►►── LDMOD ───── module ───┴──────────────────────────────►◄
```

8.85.3 Parameter

module A list of modules that are to be loaded.

8.85.4 Examples

Example 1:

Load module “S37X” (S/370 Extension).

```
HHC00013I Herc command: 'ldmod s37x'
HHC01526I HDL: loading module 's37x'...
HHC01527I HDL: module 's37x' loaded
```

Figure 168: LDMOD command

8.86 LEGACYSENSEID (Display or set SENSE ID CCW (x'E4') feature)

8.86.1 Function

The LEGACYSENSEID command specifies whether the SENSE ID CCW (X'E4') will be honoured for the devices that originally did not support that feature. This includes (but may not be limited to) 3410 and 3420 tape drives, 2311 and 2314 direct access storage devices and 2703 communication controllers. Because those legacy devices didn't originally support this command, and for compatibility reasons, the default is OFF or DISABLE.

Specify ON or ENABLE, if your guest operating system needs the Sense ID support to dynamically detect those devices. Note that most current operating systems will not detect those devices even though Sense ID is enabled because those devices never supported the Sense ID in the first place. This mainly applies to custom built or modified versions of guest operating systems that are aware of this specific Hercules capability.

Given without an argument the LEGACYSENSEID command displays the current setting.

8.86.2 Syntax

Descriptive

LEGACYSENSEID {OFF | DISABLE | ON | ENABLE}

Diagram

```
►►── LEGACYSENSEID ──┬──── OFF ────┬──────────────────────────────►◄
                     ├── DISABLE ──┤
                     ├──── ON ─────┤
                     └── ENABLE ───┘
```

8.86.3 Parameter

OFF Specify OFF or DISABLE if your guest operating system does not need the Sense ID support to dynamically detect devices that originally did not support that feature. This is the default.

DISABLE This is the same os ‘OFF’.

ON Specify ON or ENABLE if your guest operating system needs the Sense ID support to dynamically detect devices that originally did not support that feature.

ENABLE This is the same as ‘ON’.

8.86.4 Examples

Example 1:

Specify that the SENSE ID CCW will be honoured for devices that originally did not support that feature.

```
HHC00013I Herc command: 'legacysenseid enable'
HHC02204I legacysenseid  set to enabled
```

Figure 169: LEGACYSENSEID command

8.87 LOADCORE (Load a core image from a file)

8.87.1 Function

The LOADCORE command allows you to load a binary data file into real storage. This function is used mainly for emulator debugging purposes. A certain core snapshot that was saved can be restored at any later time to reproduce some tests with identical real storage values. The file *filename* is presumed to be a pure binary image file previously created via the SAVECORE command. Please note that you must stop all CPUs (see STOP / STOPALL commands) before loading a core image from a file.

8.87.2 Syntax

Descriptive

LOADCORE *filename* [*address* | 0]

Diagram

Diagram

```
                              ┌───── 0 ─────┐
►►── LOADCORE ── filename ────┼─────────────┼─────────────────────────────►◄
                              └── address ──┘
```

8.87.3 Parameter

filename This argument specifies the file name (and optionally the path) of the file from which the core image will be loaded.

address This specifies the start address of the real storage to where the saved file has to be loaded to. A value of zero, the default, means the beginning of real storage.

8.87.4 Examples

Example 1:

Load a core image file to address zero of real storage.

```
HHC00013I Herc command: 'stopall'
HHC00013I Herc command: 'loadcore d:\core02.bin 0'
HHC02250I Loading file 'd:\core02.bin' to location 0
HHC02249I Operation complete
```

Figure 170: LOADCORE command

8.88 LOADPARM (Set IPL parameter)

8.88.1 Function

The LOADPARM command displays or changes the eight-character IPL parameter which is used by all MVS based operating systems (MVS 3.8J, MVS/SP, MVS/XA, MVS/ESA, OS/390, z/OS) to select the system parameter.

8.88.2 Syntax

Descriptive

LOADPARM [*ipl_parameter*]

Diagram

```
►►── LOADPARM ──┬─────────────────┬──────────────────────────►◄
                └── ipl_parameter ┘
```

8.88.3 Parameter

ipl_parameter The system parameter used for the IPL of the intended operating system. The parameters are operating system dependent, the manuals of the operating system must be consulted for details.

8.88.4 Examples

Example 1:

Display the current LOADPARM setting.

```
HHC00013I Herc command: 'loadparm'
HHC02203I loadparm      : 014800M1
```

Figure 171: LOADPARM command (display IPL parameter)

Example 2:

Change the current LOADPARM setting.

```
HHC00013I Herc command: 'loadparm 014800t1'
HHC02204I loadparm       set to 014800T1
```

Figure 172: LOADPARM command (set IPL parameter)

8.89 LOADTEXT (Load a text deck file)

8.89.1 Function

The LOADTEXT command is essentially identical to the LOADCORE command (see section 8.87 for details) except that it loads a text deck file with “TXT” and “END” 80 byte records (i.e. an object deck).

8.89.2 Syntax

Descriptive

`LOADTEXT` *`filename`* `[`*`address`*`]`

Diagram

```
►►── LOADTEXT ── filename ─┬─────────────┬──────────────────────►◄
                           └── address ──┘
```

8.89.3 Parameter

filename This argument specifies the file name (and optionally the path) of the file from which the text deck will be loaded.

address This specifies the start address of the real storage to where the saved text deck file will be loaded.

8.89.4 Examples

See LOADCORE command for similar examples.

8.90 LOG (Direct logger output)

8.90.1 Function

The LOG command lets you redirect the log output to another destination.

8.90.2 Syntax

Descriptive

LOG [*newfile* | OFF]

Diagram

```
►►── LOG ──┬── newfile ──┬──────────────────────────────────────────────►◄
           └──── OFF ────┘
```

8.90.3 Parameter

newfile The file name and optionally the path to which the log output has to be written to.

OFF Stops the output going to the logfile.

8.90.4 Examples

Example 1:

Redirect the log output to a new logfile destination.

```
HHC00013I Herc command: 'log d:/mvs/log/newlog.txt'
```

Figure 173: LOG command (redirect output to a new logfile)

Example 2:

Stop the output going to the logfile.

```
HHC00013I Herc command: 'log off'
HHC02101I Logger: log closed
```

Figure 174: LOG command (stop output to logfile)

8.91 LOGOPT (Display or set logging options)

8.91.1 Function

The LOGOPT command lets you display or change the logging options. "TIMESTAMP" inserts a time stamp in front of each log message, whereas "NOTIMESTAMP" displays log messages with no time stamps. Entering the command without any argument displays the current logging options.

8.91.2 Syntax

Descriptive

```
LOGOPT [TIMESTAMP | NOTIMESTAMP | TIME | NOTIME]
```

Diagram

```
►►── LOGOPT ──┬─────────────────┬──────────────────────────────────►◄
              ├─── TIMESTAMP ───┤
              ├── NOTIMESTAMP ──┤
              ├───── TIME ──────┤
              └──── NOTIME ─────┘
```

8.91.3 Parameter

TIMESTAMP Insert a time stamp in front of each log message. This is the same as using TIME.

TIME Insert a time stamp in front of each log message. This is the same as using TIME-STAMP.

NOTIMESTAMP Display log messages without a timestamp. This is the same as using NOTIME.

NOTIME Display log messages without a timestamp. This is the same as using NOTIME-STAMP.

8.91.4 Examples

Example 1:

Display log messages without a timestamp in front of each message.

```
HHC00013I Herc command: 'logopt notimestamp'
HHC02204I logopts        set to NOTIMESTAMP
```

Figure 175: LOGOPT command

8.92 LPARNAME (Display or set LPAR name)

8.92.1 Function

The LPARNAME command is used to display the current LPAR name or to define the LPAR name returned by the DIAG x'204'.

8.92.2 Syntax

Descriptive

LPARNAME [*lparname*]

Diagram

```
►►── LPARNAME ──┬──────────────┬──────────────────────────────────►◄
                └── lparname ──┘
```

8.92.3 Parameter

lparname This specifies the new LPAR name. The value must be maximum 8 alphanumeric characters. If this parameter is omitted the current LPAR name is displayed.

8.92.4 Examples

Example 1:

Display the current LPAR name.

```
HHC00013I Herc command: 'lparname'
HHC02203I lparname      : HERCULES
```

Figure 176: LPARNAME command (display LPAR name)

8.93 LPARNUM (Display or set LPAR identification number)

8.93.1 Function

The LPARNUM console command displays or set the LPAR identification number. It specifies the one- or two-digit hexadecimal LPAR identification number stored by the STIDP instruction, or BASIC. If a one-digit number from 1 to F (hexadecimal) is specified, then STIDP stores a format-0 CPU ID, unless a subsequent “CPUIDFMT 1” statement is specified. If zero or a two-digit hexadecimal number, except 10 (hexadecimal), is specified, then STIDP stores a format-1 CPU ID. For LPARNUM 10 the current CPUIDFMT is not changed. If LPARNUM is BASIC, then the STIDP instruction stores a basic-mode CPU ID. The default is LPARNUM 1 with a format-0 CPU ID.

8.93.2 Syntax

Descriptive

LPARNUM [BASIC | *n* | *nn*]

Diagram

```
►►── LPARNUM ───┬───────────┬──────────────────────────────────────►◄
                ├── BASIC ──┤
                ├── 1...F ──┤
                ├──── 0 ────┤
                └─── nn ────┘
```

8.93.3 Parameter

BASIC Specifies that STIDP stores a basic-mode CPU ID.

1 … F Specifies the one-digit hexadecimal LPAR identification number. The STIDP instruction stores a format-0 CPU ID, unless a subsequent “CPUIDFMT 1” statement is specified.

0 Specifies 0 as LPAR identification number. The STIDP instruction stores a format-1 CPU ID.

nn Specifies the two-digit hexadecimal LPAR identification number (except 10 hexadecimal). For LPARNUM 10 the current CPUIDFMT is not changed. The STIDP instruction stores a format-1 CPU ID.

8.93.4 Examples

Example 1:

Display the current LPARNUM.

```
HHC00013I Herc command: 'lparnum'
HHC02203I lparnum       : 00
```

Figure 177: LPARNUM command (display LPARNUM).

Example 2:

Set LPARNUM to 21.

```
HHC00013I Herc command: 'lparnum 21'
HHC02204I lparnum        set to 21
```

Figure 178: LPARNUM command (set LPARNUM).

Example 3:

Set LPARNUM BASIC.

```
HHC00013I Herc command: 'lparnum BASIC'
HHC02204I lparnum        set to BASIC
```

Figure 179: LPARNUM command (set LPARNUM BASIC).

8.94 LS (Display file and directory listing)

8.94.1 Function

The LS command displays a list of files and subdirectories in the current directory. The LS command is only available when running Hercules under Linux and Mac OS X. For Microsoft Windows please use the "DIR" command.

8.94.2 Syntax

Descriptive

LS

Diagram

```
►►── LS ──────────────────────────────────────────────────────────►◄
```

8.94.3 Parameter

None.

8.94.4 Examples

Example 1:

Show the list of files and subdirectories in the current directory.

```
HHC00013I Herc command: 'ls -l'
total 868
drwxr-xr-x 3 hercules hercgrp   4096 2010-11-30 15:49 backups
-rw-r--r-- 1 hercules hercgrp   2350 2008-02-18 22:48 condcode
drwxr-xr-x 2 hercules hercgrp   4096 2010-11-30 16:29 dasd
drwxr-xr-x 2 hercules hercgrp   4096 2007-02-23 21:12 docs
-rw-r--r-- 1 hercules hercgrp   1150 2008-02-18 22:48 ibcdasdi.00c
-rw-r--r-- 1 hercules hercgrp    786 2008-02-18 22:48 ibcdasdi.cnf
drwxr-xr-x 2 hercules hercgrp   4096 2010-11-30 17:06 jcl
-rw-r--r-- 1 hercules hercgrp   1758 2010-12-01 18:43 mvs.cnf
-rwxr-xr-x 1 hercules hercgrp   1229 2008-02-13 20:43 mvsdasd
-rw-r----- 1 hercules hercgrp  11032 2010-12-03 15:49 mvslog.txt
-rw-r----- 1 hercules hercgrp      0 2010-12-03 15:40 pch00d.txt
-rw-r----- 1 hercules hercgrp 330245 2010-11-30 15:12 pch013.txt
-rw-r----- 1 hercules hercgrp      0 2010-12-03 15:40 prt00e.txt
-rw-r----- 1 hercules hercgrp  38121 2010-12-03 15:43 prt00f.txt
```

```
-rwxr-xr-x 1 hercules hercgrp 449463 2006-01-16 21:34 rpfuguid.pdf
-rw-r--r-- 1 hercules hercgrp    826 2008-02-18 22:50 smp.cnf
drwxr-xr-x 2 hercules hercgrp   4096 2008-01-14 00:59 source
-rw-r--r-- 1 hercules hercgrp    898 2008-02-18 22:48 sysgen.cnf
drwxr-xr-x 2 hercules hercgrp   4096 2010-11-30 16:22 tape
```

Figure 180: LS command

8.95 LSDEP (List module dependencies)

8.95.1 Function

This function lists the Hercules module dependencies.

8.95.2 Syntax

Descriptive

```
LSDEP
```

Diagram

```
►►── LSDEP ──────────────────────────────────────────────────►◄
```

8.95.3 Parameter

None.

8.95.4 Examples

Example 1:

List the module dependencies.

```
HHC00013I Herc command: 'lsdep'
HHC01535I HDL: dependency 'HERCULES' version '3.0.7.6835' size 11
HHC01535I HDL: dependency 'REGS' version '3.03' size 47760
HHC01535I HDL: dependency 'DEVBLK' version '3.05' size 3360
HHC01535I HDL: dependency 'SYSBLK' version '4.00' size 49824
HHC01535I HDL: dependency 'WEBBLK' version '2.17' size 40
```

Figure 181: LSDEP command

8.96 LSMOD (List dynamic modules)

8.96.1 Function

The LSMOD command lists all the loaded Hercules dynamic modules and shows details for each module.

8.96.2 Syntax

Descriptive

LSMOD

Diagram

```
►►── LSMOD ──────────────────────────────────────────────────────────►◄
```

8.96.3 Parameter

None.

8.96.4 Examples

Example 1:

List the dynamic modules.

```
HHC00013I Herc command: 'lsmod'
HHC01531I HDL: dll type = load, name = s37x, flags = (unloadable, not forced)
HHC01531I HDL: dll type = load, name = hdt3088, flags = (unloadable, not forced)
HHC01532I HDL:  symbol = debug_tt32_tracing, loadcount = 1, owner = hdt3088
HHC01532I HDL:  symbol = debug_tt32_stats, loadcount = 1, owner = hdt3088
HHC01533I HDL:  devtype(s) = CTCI-WIN VMNET CTCT CTCI 3088 LCS
HHC01531I HDL: dll type = load, name = hdt3420, flags = (unloadable, not forced)
HHC01533I HDL:  devtype(s) = 9348 9347 8809 3590 3490 3480 3430 3422 3420 3411 3410
HHC01531I HDL: dll type = load, name = hdt3270, flags = (unloadable, not forced)
HHC01533I HDL:  devtype(s) = SYSG 3287 3270 3215 1052
HHC01531I HDL: dll type = load, name = hdt1403, flags = (unloadable, not forced)
HHC01533I HDL:  devtype(s) = 3211 1403
HHC01531I HDL: dll type = load, name = hdt3525, flags = (unloadable, not forced)
HHC01533I HDL:  devtype(s) = 3525
HHC01531I HDL: dll type = load, name = hdt3505, flags = (unloadable, not forced)
HHC01533I HDL:  devtype(s) = 3505 2501 1442
HHC01531I HDL: dll type = load, name = dyncrypt, flags = (unloadable, not forced)
HHC01532I HDL:  symbol = z900_perform_cryptographic_key_management_operation, loadcount = 1, ...
```

```
HHC01532I HDL:  symbol = z900_perform_cryptographic_computation, loadcount = 1, owner = dyncrypt
HHC01532I HDL:  symbol = z900_compute_message_authentication_code, loadcount = 1, owner = ...
HHC01532I HDL:  symbol = z900_compute_last_message_digest, loadcount = 1, owner = dyncrypt
HHC01532I HDL:  symbol = z900_compute_intermediate_message_digest, loadcount = 1, owner = ...
HHC01532I HDL:  symbol = z900_cipher_message_with_output_feedback, loadcount = 1, owner = ...
HHC01532I HDL:  symbol = z900_cipher_message_with_counter, loadcount = 1, owner = dyncrypt
HHC01532I HDL:  symbol = z900_cipher_message_with_cipher_feedback, loadcount = 1, owner = ...
HHC01532I HDL:  symbol = z900_cipher_message_with_chaining, loadcount = 1, owner = dyncrypt
HHC01532I HDL:  symbol = z900_cipher_message, loadcount = 1, owner = dyncrypt
HHC01532I HDL:  symbol = s390_perform_cryptographic_key_management_operation, loadcount = 1, ...
HHC01532I HDL:  symbol = s390_perform_cryptographic_computation, loadcount = 1, owner = dyncrypt
HHC01532I HDL:  symbol = s390_compute_message_authentication_code, loadcount = 1, owner = ...
HHC01532I HDL:  symbol = s390_compute_last_message_digest, loadcount = 1, owner = dyncrypt
HHC01532I HDL:  symbol = s390_compute_intermediate_message_digest, loadcount = 1, owner = ...
HHC01532I HDL:  symbol = s390_cipher_message_with_output_feedback, loadcount = 1, owner = ...
HHC01532I HDL:  symbol = s390_cipher_message_with_counter, loadcount = 1, owner = dyncrypt
HHC01532I HDL:  symbol = s390_cipher_message_with_cipher_feedback, loadcount = 1, owner = ...
HHC01532I HDL:  symbol = s390_cipher_message_with_chaining, loadcount = 1, owner = dyncrypt
HHC01532I HDL:  symbol = s390_cipher_message, loadcount = 1, owner = dyncrypt
HHC01531I HDL: dll type = load, name = hdteq, flags = (unloadable, not forced)
HHC01532I HDL:  symbol = hdl_device_type_equates, loadcount = 1, owner = hdteq
HHC01531I HDL: dll type = main, name = *Hercules, flags = (not unloadable, not forced)
HHC01532I HDL:  symbol = panel_display, loadcount = 1, owner = *Hercules
HHC01532I HDL:  symbol = panel_command, loadcount = 1, owner = *Hercules
HHC01532I HDL:  symbol = parse_args, loadcount = 0, owner = *Hercules
HHC01533I HDL:  devtype(s) = 9336 9335 9332 9313 3370 3310 0671 9345 3390 3380 3375 3350 3340...
```

Figure 182: LSMOD command

8.97 MAINSIZE (Display or set main storage size)

8.97.1 Function

The MAINSIZE command is used to specify the size of the main storage. Given without an argument the MAINSIZE command displays the current size of the main storage. Storage is allocated in megabytes, unless a specific unit is specified. The actual upper limit of the main storage is determined by the host system's architecture, operating system, and on some systems the amount of physical memory and paging space you have available. The practical limit depends on the maximum amount of storage that can be obtained by “malloc” (usually around 1 GB on 32-bit platforms; on 64-bit platforms the value should only be limited by available paging space).

When increasing the main size Hercules attempts to allocate first the new storage. If the new allocation is successful then the previously allocated memory will be freed. This is to prevent a situation where the old memory is freed first, then the new allocation fails and a reallocation of the memory in the previous size also fails because of storage fragmentation and therefore leaving Hercules without memory.

When decreasing the main storage the memory will stay allocated in the previous size but the storage size will appear as decreased. Subsequent increases will not reallocate memory unless they go over the already allocated amount.

An additional optional argument determines the locking state of the allocated memory (page lock by host operating system). The LOCKED option indicates that the memory is to be locked into storage while UNLOCKED (the default) indicates that the memory is not locked into the storage.

Please note that Hercules preserves the last locking state of MAINSIZE. Once storage is locked, any subsequent change to the main storage size will honor the existing lock state of memory unless the lock state is specified again on the MAINSIZE command.

Caution: Do not lock main storage unless sufficient real memory is available to back up the request. Failure to do so may require the host system to be rebooted.

8.97.2 Syntax

Descriptive

MAINSIZE [*msize*[B | K | M | G | T | P |E] [UNLOCK | LOCK]]

Diagram

```
►►── MAINSIZE ──┬──────────────────────────────────────────────────┬──────────►◄
                └── msize ──┬── M ──┬──┬── UNLOCK ──┬──┘
                            ├── B ──┤  └── LOCK ────┘
                            ├── K ──┤
                            ├── G ──┤
                            ├── T ──┤
                            ├── P ──┤
                            └── E ──┘
```

8.97.3 Parameter

msize The value of *msize* must be a valid decimal number. The actual upper limit is determined by the host system's architecture, the operating system and on some systems the amount of physical memory and paging space that is available.

For storage sizes less than 16M, sizes not on a 4K boundary are rounded up to the next 4K boundary. Otherwise, storage sizes not on a 1M boundary are rounded up to the next 1M boundary.

The minimum size is 4K for architecture levels ALS0 and ALS1 (S/370 and ESA/390), and 8K for architecture level ALS2 (ESAME) and higher. A maximum of 64M may be specified for architecture level ALS0 (S/370), 2048M (2G) for ALS1 (ESA/390) and 16E for architecture level ALS2 (ESAME) and higher.

The default on startup is 2M.

B ‘B’ determines that the number given is specified in bytes (no multiplier).

K ‘K’ determines that the number given is specified in kilobytes (multiplier 2**10).

M ‘M’ determines that the number given is specified in megabytes (multiplier 2**20). This is the default if no unit is appended.

G ‘G’ determines that the number given is specified in gigabytes (multiplier 2**30).

T ‘T’ determines that the number given is specified in terabytes (multiplier 2**40). On 32-bit machines the unit terabytes is not available.

P ‘P’ determines that the number given is specified in petabytes (multiplier 2**50). On 32-bit machines the unit petabytes is not available.

E ‘E’ determines that the number given is specified in exabytes (multiplier 2**60). On 32-bit machines the unit exabytes is not available.

LOCK Attempt to lock the storage (pages locked by the host operating system).

UNLOCK Leave the store unlocked (no pages locked by the host operating system). This is the default.

Notes:

The actual upper limit is determined by the host system’s architecture and operating system and the amount of physical memory and available paging space. The total of MAINSIZE and XPNDSIZE on host systems with a 32-bit architecture will be limited to 4G; host systems with a 64-bit architecture will be limited to less than 16E.

Using minimum storage sizes, storage sizes less than or not on a 64K boundary for architecture level ALS0 (S/370) or not on a 1M boundary for architecture level ALS1 (ESA/390) and higher, it may be possible to generate error conditions not covered by the “Principles of Operations”.

Use of storage sizes greater than supported by the guest operating system may generate incorrect results or error conditions within the guest operating system.

8.97.4 Overview Storage Allocation Units

Unit	Multiplier	Name (Symbol)	IEC Name (IEC Symbol)	Restrictions
B	None	Byte (B)	Byte (B)	
K	2**10	Kilobyte (kB)	Kibibyte (KiB)	
M	2**20	Megabyte (MB)	Mebibyte (MiB)	
G	2**30	Gigabyte (GB)	Gibibyte (GiB)	
T	2**40	Terabyte (TB)	Tebibyte (TiB)	Not on 32-bit machines
P	2**50	Petabyte (PB)	Pebibyte (PiB)	Not on 32-bit machines
E	2*60	Exabyte (EB)	Exbibyte (EiB)	Not on 32-bit machines

Table 20: Storage Allocation Units

8.97.5 Examples

Example 1:

Display the current size of the main storage.

```
HHC00013I Herc command: 'mainsize'
HHC17003I MAIN     storage is 001 GBytes 'mainsize'; storage is not locked
```

Figure 183: MAINSIZE command (display size of main storage)

Example 2:

Set the size of the main storage to 512 MB. Do not lock the memory into the storage.

```
HHC00013I Herc command: 'mainsize 512'
HHC17003I MAIN     storage is 512 MBytes 'mainsize'; storage is not locked
```

Figure 184: MAINSIZE command (Set size of unlocked main storage)

Example 3:

Set the size of the main storage to 256 MB. Lock the memory into the storage.

```
HHC00013I Herc command: 'mainsize 256 lock'
HHC01428I Locking main storage
HHC17003I MAIN     storage is 256 MBytes 'mainsize'; storage is locked
```

Figure 185: MAINSIZE command (Set size of locked main storage)

8.98 MANUFACTURER (Display or set STSI manufacturer code)

8.98.1 Function

The MANUFACTURER console command displays or sets the manufacturer name returned by the STSI instruction. If no argument is given, the current manufacturer name is displayed.

8.98.2 Syntax

Descriptive

Descriptive

MANUFACTURER [*name*]

Diagram

```
►►── MANUFACTURER ──┬──────────┬──────────────────────────────────►◄
                    └── name ──┘
```

8.98.3 Parameter

name Any name with a maximum length of four characters.

8.98.4 Examples

Example 1:

Set the STSI manufacturer name to ‘HRC’.

```
HHC00013I Herc command: 'manufacturer HRC'
HHC02204I manufacturer   set to HRC
```

Figure 186: MANUFACTURER command (set STSI manufacturer code)

8.99 MAXCPU (Display or set maximum number of CPUs)

8.99.1 Function

The MAXCPU command displays or sets the maximum number of installed processor engines. The combination of MAXCPU and NUMCPU controls the behaviour of how many CPU engines will be configured online upon startup and how many can be configured online later. The NUMCPU statement specifies the number of engines which will be configured online at startup time. All processors are CP engines unless otherwise specified by the ENGINES statement.

MAX_CPU_ENGINES is a compile-time variable which sets an upper limit on the value of MAXCPU. The value of MAX_CPU_ENGINES is displayed in the build information message on the Hercules control panel at startup time. To change the value of MAX_CPU_ENGINES you must rebuild Hercules. For Unix builds, specify "./configure --enable-multi-cpu=*nn*" before performing make. For Windows builds, specify "SET MAX_CPU_ENGINES=*nn*" before performing nmake.

MAX_CPU_ENGINES may be up to 128 on 64-bit Linux platforms. On Windows, and on all 32-bit platforms, the maximum value is 64. For performance reasons, values above 32 are not recommended for 32-bit platforms. If MAX_CPU_ENGINES is set to 1 then multiprocessing is disabled. See also the NUMCPU statement for a discussion of the performance implications of MAX_CPU_ENGINES.

The value of MAXCPU cannot exceed the value of MAX_CPU_ENGINES. If MAXCPU is not specified in the configuration file, then its initial value is equal to NUMCPU. If MAXCPU and NUMCPU are both omitted, then MAXCPU is set to 1.

Given without an argument the MAXCPU statement displays the current maximum number of installed processor engines.

For detailed explanations on the interrelationship between MAXCPU, ENGINES and NUMCPU please see “Appendix B. Configuration of Emulated CPUs”.

8.99.2 Syntax

Descriptive

`MAXCPU [nn]`

Diagram

```
►►── MAXCPU ──┬────────┬──────────────────────────────►◄
              └── nn ──┘
```

8.99.3 Parameter

nn Specifies the maximum number of installed processor engines. The value of MAXCPU cannot exceed the value of MAX_CPU_ENGINES.

8.99.4 Examples

Example 1:

Display the number of installed processor engines.

```
HHC00013I Herc command: 'maxcpu'
HHC02203I maxcpu        : 2
```

Figure 187: MAXCPU command (display current number of installed processors)

Example 2:

Set the maximum number of installed processor engines to 8.

```
HHC00013I Herc command: 'maxcpu 8'
HHC02204I maxcpu         set to 8
```

Figure 188: MAXCPU command (set number of installed processors)

8.100 MAXRATES (Display highest MIPS/SIO rate or set new reporting interval)

8.100.1 Function

The MAXRATES command shows the maximum observed MIPS (million instructions per second) and SIO (Start I/O) rates reached during the current reporting interval (up to the time the command is issued). It lets you also set the MAXRATES reporting interval to a new value.

If there is no MAXRATES reporting interval set in the configuration file then the default value for reporting the rates is 1440 minutes (1 day). When the interval is expired a MAXRATES command is automatically issued. The current rates will also be displayed during shutdown of Hercules.

8.100.2 Syntax

Descriptive

MAXRATES [*interval* | MIDNIGHT]

Diagram

```
►►── MAXRATES ──┬──────────────┬──────────────────────────►◄
                ├── interval ──┤
                └── MIDNIGHT ──┘
```

8.100.3 Parameter

interval This is the new interval time in minutes. The default MAXRATES interval that is active after startup of Hercules (if not otherwise specified in the configuration file), is 1440 minutes (1 day). Changes to the MAXRATES interval that are other than "MIDNIGHT" will set the current interval start time to the present; this includes a value of "1440".

MIDNIGHT Sets the interval to 1440 minutes (1 day) and the start time for the interval timer to midnight of the current day. This will cause the MAXRATE statistics to be date aligned.

8.100.4 Examples

Example 1:

Change the MAXRATES interval from the default value (or the last manually set interval value) to 60 minutes.

```
HHC00013I Herc command: 'maxrates 60'
HHC02204I maxrates       set to 60 minutes
```

Figure 189: MAXRATES command (set the interval time)

Example 2:

Display the maximum observed MIPS and I/O rates during the last interval.

```
HHC00013I Herc command: 'maxrates'
HHC02272I Highest observed MIPS and IO/s rates:
HHC02272I From Thu Apr 19 22:58:34 2012 to Fri Apr 20 22:58:34 2012
HHC02272I MIPS: 105.333858
HHC02272I IO/s: 3068
HHC02272I From Fri Apr 20 22:58:34 2012 to Fri Apr 20 23:06:38 2012
HHC02272I MIPS: 5.678100
HHC02272I IO/s: 54
HHC02272I Current interval is 1440 minutes
```

Figure 190: MAXRATES command (display maximum rates)

8.101 MEMLOCK (Lock Hercules memory)

8.101.1 Function

The MEMLOCK console command is used to lock Hercules memory in storage. It defines the locking state of the allocated Hercules memory (page lock by host operating system). Given without argument MEMLOCK displays the current setting. If Hercules memory is locked in storage it cannot be paged out by the host operating system. This may result in some performance improvements.

This console command is available if Hercules is built with option “_HAVE_MLOCKALL”. Currently MEMLOCK is only supported under Linux host operating systems.

8.101.2 Syntax

Descriptive

MEMLOCK [ON | <u>OFF</u>]

Diagram

```
►►── MEMLOCK ──┬──────────┬──────────────────────────────►◄
               ├── ON ────┤
               └── OFF ───┘
```

8.101.3 Parameter

ON ON indicates that the memory is to be locked into storage.

OFF OFF indicates that the memory is not to be locked into the storage.

8.101.4 Examples

Example 1:

Do not lock Hercules memory in storage.

```
HHC00013I Herc command: 'memlock off'
```

Figure 191: MEMLOCK command

8.102 MESSAGE (Display message on console like VM)

8.102.1 Function

The MESSAGE command is actually a VM CP command. It is used to transmit message text to the virtual console of other active users.

However, some programmers also use this command as a simplified way to display messages on the own virtual machine console by using "MESSAGE * <message text>". One program that uses this is the IBM "System z" port of Solaris which is only capable of running under z/VM.

So basically, all the MESSAGE command does is to display a message on the Hercules console with VM like headers. Therefore the MESSAGE command is not really intended to be used directly from the Hercules console as shown in the example below, although it does not hurt doing so.

The MESSAGE command is the same as MSG. See also the MSG and MSGNOH commands in this manual.

8.102.2 Syntax

Descriptive

MESSAGE *parms*

Diagram

```
►►── MESSAGE ── parms ──────────────────────────────────►◄
```

8.102.3 Parameter

parms Specifies the required and optional parameters as described in IBM's VM "CP Commands and Utilities Reference" manual.

8.102.4 Examples

Example 1:

Display a message on the own virtual machine console.

```
HHC00013I Herc command: 'message * transmitted message to VM console'
20:50:33  * MSG FROM HERCULES: transmitted message to VM console
```

Figure 192: MESSAGE command

8.103 MODEL (Display or set STSI model code)

8.103.1 Function

The MODEL console command displays or sets the model names returned by the STSI instruction. If no argument is given the current STSI model names are displayed.

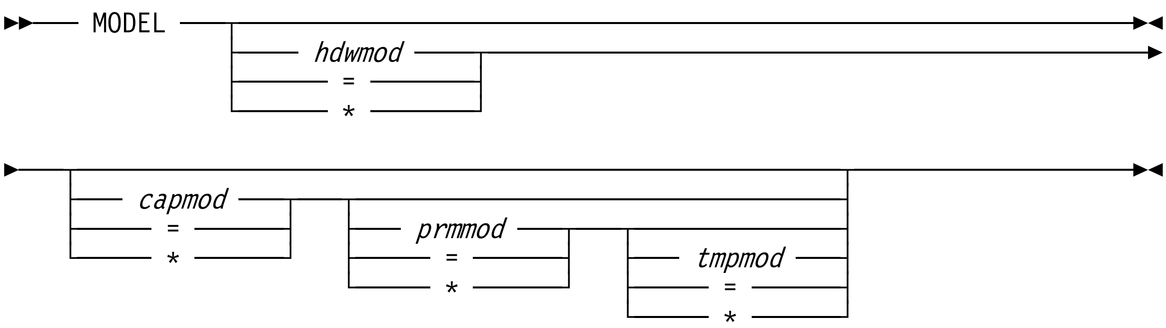
The optional operands specify the hardware model name, the capacity model name, the permanent capacity model name and the temporary capacity model name respectively.

8.103.2 Syntax

Descriptive

MODEL [*hdwmod* | = | * [*capmod* | = | * [*prmmod* | = | * [*tmpmod* | = | *]]]]

Diagram



8.103.3 Parameter

hdwmodel This specifies the hardware model name. This can be any name with a maximum length of 16 characters. Valid characters are A-Z and 0-9. Specifying a “=” resets the hardware model to “EMULATOR”; specifying an “*” leaves the current hardware model name intact. If the MODEL system parameter is not specified in the configuration file then the default name is “EMULATOR”.

capmodel This optional parameter specifies the capacity model name. The *capmodel* can be any name with a maximum length of 16 characters. Valid characters are A-Z and 0-9. Specifying a “=” copies the current hardware model name to the capacity model; specifying an “*” leaves the current capacity model name intact. The default capacity model name is “EMULATOR”.

prmmodel This specifies the permanent capacity model name. The *prmmodel* can be any name with a maximum length of 16 characters. Valid characters are A-Z and 0-9. Specifying a “=” copies the current capacity model name to the permanent capacity model; specifying an “*” leaves the current permanent capacity model name intact.

The default permanent capacity model name is a null string.

tmpmodel This specifies the temporary capacity model name. The *tmpmodel* can be any name with a maximum length of 16 characters. Valid characters are A-Z and 0-9. Specifying a "=" copies the current permanent capacity model name to the temporary capacity model; specifying an "*" leaves the current temporary capacity model name intact. The default temporary capacity model name is a null string.

8.103.4 Examples

Example 1:

Set the STSI model name (hardware model and capacity model) to "EMULATOR".

```
HHC00013I Herc command: 'model emulator emulator'
HHC02204I hdw model      set to EMULATOR
HHC02204I cap model      set to EMULATOR
HHC02204I prm model      set to
HHC02204I tmp model      set to
```

Figure 193: MODEL command (set STSI model name, variant 1)

Example 2:

The same could also be specified using the following simplified command:

```
HHC00013I Herc command: 'model emulator ='
HHC02204I hdw model      set to EMULATOR
HHC02204I cap model      set to EMULATOR
HHC02204I prm model      set to
HHC02204I tmp model      set to
```

Figure 194: MODEL command (set STSI model name, variant 2)

Example 3:

Set the STSI model name, the permanent capacity model name and the temporary model name to "HERCULES", but leave the capacity model name intact.

```
HHC00013I Herc command: 'model hercules * hercules ='
HHC02204I hdw model      set to HERCULES
HHC02204I cap model      set to EMULATOR
HHC02204I prm model      set to HERCULES
HHC02204I tmp model      set to HERCULES
```

Figure 195: MODEL command (set STSI model name, variant 1)

8.104 MODPATH (Display or set dynamic load module path)

8.104.1 Function

The MODPATH command displays or sets the dynamic load module path. If a MODPATH is defined then this path is searched before the default path is searched. When a relative path is specified it is interpreted as a relative path within the default search path. If an absolute path is coded it is interpreted as such.

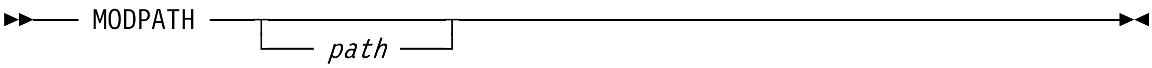
The default MODPATH is “hercules” which means modules are loaded from the directory “hercules” within the default LD_LIBRARY_PATH.

8.104.2 Syntax

Descriptive

MODPATH [*path*]

Diagram



8.104.3 Parameter

path The path where dynamic load modules are loaded from.

8.104.4 Examples

Example 1:

Display the current dynamic load module path.

```
HHC00013I Herc command: 'modpath'
HHC01508I HDL: loadable module directory is 'D:/Hercules/'
```

Figure 196: MODPATH command (display dynamic load module path)

Example 2:

Set the current dynamic load module path to “D:/Hercules/Modules”.

```
HHC00013I Herc command: 'modpath D:/Hercules/Modules'
HHC01508I HDL: loadable module directory is 'D:/Hercules/Modules'
```

Figure 197: MODPATH command (set dynamic load module path)

8.105 MOUNTED_TAPE_REINIT (Control tape initialization)

8.105.1 Function

The MOUNTED_TAPE_REINIT command controls whether reinitialization of tape drive devices via the DEVINIT command in order to mount a new tape should be allowed if there is already a tape mounted on the drive. Without a parameter given the current setting is displayed.

Specifying ALLOW indicates that new tapes may be mounted via "DEVINIT nnnn new-tape-filename" irrespective of whether or not there is already a tape mounted on the drive.

Specifying DISALLOW prevents new tapes from being mounted if one is already mounted. Before the new tape can be mounted the existing one has first to be unmounted (via the "DEVINIT nnnn *" command). Otherwise the DEVINIT attempt to mount the new tape is rejected.

This option is meant as a safety mechanism to protect against accidentally unmounting a tape from the wrong device as a result of a simple typing error and thereby eventually cancelling a potentially important tape job.

Please note that for SCSI tape drives the "DEVINIT nnnn *" command has no effect. The tape must be unmounted manually since it is a real physical device and not emulated via a disk file like '.AWS' or '.HET' tapes.

8.105.2 Syntax

Descriptive

```
MOUNTED_TAPE_REINIT [ENABLE | ALLOW | DISABLE | DISALLOW]
```

Diagram

```
►►── MOUNTED_TAPE_REINIT ──┬─────────────────┬──────────────────────►◄
                           ├── ENABLE ───────┤
                           ├── ALLOW ────────┤
                           ├── DISABLE ──────┤
                           └── DISALLOW ─────┘
```

8.105.3 Parameter

ENABLE Indicates that new tapes may be mounted irrespective of whether or not there is already a tape mounted on the drive. This is the default.

ALLOW This is the same as ENABLE.

DISABLE Prevents new tapes from being mounted if one is already mounted on the drive. Before the new tape can be mounted the currently mounted tape must first to be unmounted. Instead of DISABLE, the argument DISALLOW that has been used in earlier versions of Hercules can also be used.

DISALLOW This is the same as DISABLE.

8.105.4 Examples

Example 1:

Display current tape mount settings.

```
HHC00013I Herc command: 'mounted_tape_reinit'
HHC02203I mounted_tape_reinit: enabled
```

Figure 198: MOUNTED_TAPE_REINIT command (display settings).

Example 2:

Disallow tape mount reinitialization.

```
HHC00013I Herc command: 'mounted_tape_reinit disable'
HHC02204I mounted_tape_reinit set to disabled
```

Figure 199: MOUNTED_TAPE_REINIT command (disable tape mount reinitialization)

8.106 MSG (Display message on console like VM)

8.106.1 Function

The MSG command is actually a VM CP command. It is used to transmit message text to the virtual console of other active users.

However, some programmers also use this command as a simplified way to display messages on the own virtual machine console by using "MSG * <message text>". One program that uses this is the IBM "System z" port of Solaris which is only capable of running under z/VM.

So basically, all the MSG command does is to display a message on the Hercules console with VM like headers. Therefore the MSG command is not really intended to be used directly from the Hercules console as shown in the example below, although it does not hurt doing so.

The MSG command is the same as MESSAGE. See also the MESSAGE and MSGNOH commands in this manual.

8.106.2 Syntax

Descriptive

MSG *parms*

Diagram

```
►►── MSG ── parms ──────────────────────────────────────────►◄
```

8.106.3 Parameter

parms This specifies the required and optional parameters as described in IBM's VM "CP Commands and Utilities Reference" manual.

8.106.4 Examples

Example 1:

Display a message on the own virtual machine console.

```
18:24:09 msg * transmitted message to VM console
18:24:09  18:24:09  * MSG FROM HERCULES: transmitted message to VM console
```

Figure 200: MSG command

8.107 MSGHLD (Display or set timeout of held messages)

8.107.1 Function

The MSGHLD console command is used to display or set the timeout value of held messages. It is also possible to release all held messages without having to wait for the timeout period to expire.

8.107.2 Syntax

Descriptive

MSGHLD {*nnn* | INFO | CLEAR}

Diagram

```
►►── MSGHLD ───┬── nnn ──┬─────────────────────────────────────►◄
               ├── INFO ─┤
               └── CLEAR ┘
```

8.107.3 Parameter

nnn This value specifies the new timeout value of held messages in seconds.

INFO Display the current message held time setting.

CLEAR Release the held messages immediately. The command 'KD' can be used as a shortcut for 'MSGHLD CLEAR'.

8.107.4 Examples

Example 1:

Set the new timeout interval for held messages to 30 seconds.

```
HHC00013I Herc command: 'msghld 30'
HHC02204I message hold time set to 30 seconds
```

Figure 201: MSGHLD command (set new interval)

Example 2:

Release all held messages.

```
HHC00013I Herc command: 'msghld clear'
HHC02226I Held messages cleared
```

Figure 202: MSGHLD command (release messages)

8.108 MSGLEVEL (Display or set the current message display output)

8.108.1 Function

The MSGLEVEL command displays the current setting of the message level or allows setting a new message level. It decides how many and what kind of messages are written to the Hercules console (and to the log). Given without an argument MSGLEVEL displays the current message level settings.

The message level is set per default to 'terse' which turns the verbose message level off. To display the additional messages during configuration file processing Hercules can be started with the "-v" option which sets the verbose message level. As an alternative, the MSGLEVEL system parameter can be set to activate the verbose message level. In this case however, MSGLEVEL must be coded as one of the first statements in the configuration file to take effect at an early stage during configuration file processing.

In addition to the 'terse' level the following options are set by default (if not otherwise overwritten through the MSGLEVEL system parameter or console command): 'nodebug', 'tape', 'dasd', 'comm', 'ur', 'scsi', 'ctca', 'graf', 'thread' and 'channel'.

Certain levels (on, off, text, time) can only be set if Hercules is built with one of the following build options: OPTION_MSGCLR or OPTION_MSGHLD.

8.108.2 Syntax

Descriptive

```
MSGLEVEL [option option ...]

   where option can be:

ON | OFF | TEXT | TIME | NODEBUG |
[+ | -] DEBUG   |
[+ | -] TAPE    |
[+ | -] DASD    |
[+ | -] COMM    |
[+ | -] UR      |
[+ | -] SCSI    |
[+ | -] CTCA    |
[+ | -] GRAF    |
[+ | -] THREAD  |
[+ | -] CHANNEL |
[+ | -] VERBOSE |
[+ | -] TERSE
```

Diagram

```
►►── MSGLEVEL ──┬─────────────────────┬──────────────────────────►◄
                │  ┌───────────────┐  │
                └──▼──┤ option ├───┴──┘

   where option can be:

├──┬──────────────── ON ──────┬──────────────────────────────────────┤
   ├──────────────── OFF ─────┤
   ├──────────────── TEXT ────┤
   ├──────────────── TIME ────┤
   ├──────────────── NODEBUG ─┤
   ├───────┬──────┬─ DEBUG ───┤
   ├── + ──┤      ├─ TAPE ────┤
   └── - ──┘      ├─ DASD ────┤
                  ├─ COMM ────┤
                  ├─ UR ──────┤
                  ├─ SCSI ────┤
                  ├─ CTCA ────┤
                  ├─ GRAF ────┤
                  ├─ THREAD ──┤
                  ├─ CHANNEL ─┤
                  ├─ VERBOSE ─┤
                  └─ TERSE ───┘
```

8.108.3 Parameter

ON	ON displays the messages in the default kind with message number followed by the message text.
OFF	No messages are displayed.
TEXT	Displays only the text part of the message (without message numbers).
TIME	Prefix the messages with a timestamp.
NODEBUG	The messages are not issued in debug mode (not prefixed additionally with the name of the source member and the line number that issues the message).
DEBUG	The messages are prefixed additionally with the name of the source member and the line number that issues the message.
TAPE	Display tape related messages.
DASD	Display DASD related messages.
COMM	Display communications related messages.
UR	Display unit record related messages.

SCSI Display SCSI related messages.

CTCA Display CTCA and LCS related messages.

GRAF Display graphics (3270) related messages.

THREAD Display threading related messages.

CHANNEL Display channel related messages.

VERBOSE Displays additional messages during configuration file processing.

TERSE This turns the verbose message level off. This is the default unless VERBOSE is specified either through the MSGLEVEL system parameter or panel command or if Hercules is started with the “-v” option.

8.108.4 Examples

Example 1:

Display the current message level.

```
HHC00013I Herc command: 'msglevel'
HHC17012I MSGLEVEL = terse nodebug tape dasd ur comm ctca graf scsi
```

Figure 203: MSGLEVEL command (display message level)

Example 2:

Set the message level to supress UR and SCSI related messages.

```
HHC00013I Herc command: 'msglvl -ur -scsi'
HHC17012I MSGLEVEL = terse nodebug tape dasd comm ctca graf
```

Figure 204: MSGLEVEL command (suppress certain messages)

Example 3:

Set the message level to DEBUG.

```
HHC00013I Herc command: 'msglevel debug'
cmdtab.c     350 21:01:56 HHC90000D DBG: Panel_Exit CommandLockCounter 0
logmsg.c     261 HHC00007I Previous message from function ProcessPanelCommand() at cmdtab.c[350]
cmdtab.c     361 21:01:56 HHC90000D DBG: RC = 0
logmsg.c     261 HHC00007I Previous message from function ProcessPanelCommand() at cmdtab.c[361]
cmdtab.c     602 21:02:01 HHC02101I Logger: log closed
```

Figure 205: MSGLEVEL command (set message level DEBUG)

8.109 MSGLVL (Display or set the current message display output)

8.109.1 Function

MSGLVL is an alias for MSGLEVEL. The MSGLVL command displays the current setting of the message level or allows it to set a new message level. See MSGLEVEL for details.

8.109.2 Syntax

See MSGLEVEL command.

8.109.3 Parameter

See MSGLEVEL command.

8.109.4 Examples

See MSGLEVEL command.

8.110 MSGNOH (Display message on console like VM, but without header)

8.110.1 Function

The MSGNOH command is actually a VM CP command. It is used to transmit message text to the virtual console of other active users without the standard MESSAGE command header. However, some programmers also use this command as a simplified way to display messages on the own virtual machine console by using "MSGNOH * <message text>".

So basically, all the MSGNOH command does is to display a message on the Hercules console (like the MESSAGE and MSG commands) but without VM like headers. Therefore the MSGNOH command is not really intended to be used directly from the Hercules console as shown in the example below, although it does not hurt doing so.

See also the MESSAGE and MSG commands in this manual.

8.110.2 Syntax

Descriptive

MSGNOH *parms*

Diagram

```
►►── MSGNOH ── parms ──────────────────────────────────────────►◄
```

8.110.3 Parameter

parms Specifies the required and optional parameters as described in IBM's VM "CP Commands and Utilities Reference" manual.

8.110.4 Examples

Example 1:

Display a message on the own virtual machine console.

```
HHC00013I Herc command: 'msgnoh * transmitted message to VM console'
transmitted message to VM console
```

Figure 206: MSGNOH command

8.111 MT (Control magnetic tape operation)

8.111.1 Function

The MT command performs various actions on magnetic tapes. All operations can be used on a valid tape device. The device must not have any I/O operation in process or pending.

Please note that the WTM operation will truncate the tape after the last WTM written if it is not at the logical end of the tape when the WTM is issued. No warnings are given.

8.111.2 Syntax

Descriptive

MT *device operation*

where operation can be:

```
REW
ASF [nnnn | 1]
FSF [nnnn | 1]
BSF [nnnn | 1]
FSR [nnnn | 1]
BSR [nnnn | 1]
WTM [nnnn | 1]
DSE
DVOL1
```

Diagram

```
►►── MT ── device ──┤ operation ├──────────────────────────────────►◄

where operation can be:

├── REW ────────────────────────────────────────────────────────────┤

                ┌──── 1 ───┐
├── ASF ────────┼──────────┴────────────────────────────────────────┤
                └─── nnnn ─┘
```

```
├── FSF ─┬─── 1 ───┬──────────────────────────────────────────────────┤
         └── nnnn ─┘

├── BSF ─┬─── 1 ───┬──────────────────────────────────────────────────┤
         └── nnnn ─┘

├── FSR ─┬─── 1 ───┬──────────────────────────────────────────────────┤
         └── nnnn ─┘

├── BSR ─┬─── 1 ───┬──────────────────────────────────────────────────┤
         └── nnnn ─┘

├── WTM ─┬─── 1 ───┬──────────────────────────────────────────────────┤
         └── nnnn ─┘

├── DSE ──────────────────────────────────────────────────────────────┤

├── DVOL1 ────────────────────────────────────────────────────────────┤
```

8.111.3 Parameter

device	This specifies the device address of the tape to which the operation will be performed.
REW	Rewind the tape.
ASF	Position the tape at file *nnnn*.
FSF	Forward space *nnnn* files.
BSF	Backward space *nnnn* files.
FSR	Forward space *nnnn* records.
BSR	Backward space *nnnn* records.
WTM	Write *nnnn* tapemarks.
DSE	Data secure erase.
DVOL1	This must be a number in the range of 1-9999. The default is one if *nnnn* is not specified.

nnnn This must be a number in the range of 1-9999. The default is one if *nnnn* is not specified.

8.111.4 Examples

Example 1:

Rewind a tape.

```
HHC00013I Herc command: 'mt 0480 rew'
HHC02800I 0:0480 rew complete
HHC02802I 0:0480 Current file number 1
HHC02803I 0:0480 Current block number 0
```

Figure 207: MT command (REW operation)

Example 2:

Position the tape at file number 5.

```
HHC00013I Herc command: 'mt 0480 asf 5'
HHC02800I 0:0480 asf complete
HHC02802I 0:0480 Current file number 5
HHC02803I 0:0480 Current block number 775
```

Figure 208: MT command (ASF operation)

Example 3:

Forward space 2 files.

```
HHC00013I Herc command: 'mt 0480 fsf 2'
HHC02800I 0:0480 fsf complete
HHC02802I 0:0480 Current file number 7
HHC02803I 0:0480 Current block number 2471
```

Figure 209: MT command (FSF operation)

Example 4:

Forward space 3 records.

```
HHC00013I Herc command: 'mt 0480 fsr 3'
HHC02800I 0:0480 fsr complete
HHC02802I 0:0480 Current file number 7
HHC02803I 0:0480 Current block number 2474
```

Figure 210: MT command (FSR operation)

Example 5:

Write 3 tapemarks.

```
HHC00013I Herc command: 'mt 0480 wtm 3'
HHC02800I 0:0480 wtm complete
HHC02802I 0:0480 Current file number 10
HHC02803I 0:0480 Current block number 2477
```

Figure 211: MT command (WTM operation)

Example 6:

Erase a tape.

```
HHC00013I Herc command: 'mt 0480 dse'
HHC02800I 0:0480 dse complete
HHC02802I 0:0480 Current file number 1
HHC02803I 0:0480 Current block number 0
```

Figure 212: MT command (DSE operation)

8.112 NUMCPU (Display or set number of emulated CPUs)

8.112.1 Function

The NUMCPU commands displays or sets the number of emulated processor engines which will be configured online at startup time. The combination of NUMCPU and MAXCPU controls the behaviour of how many CPU engines will be configured online upon startup and how many can be configured online later.

NUMCPU cannot exceed the value of MAXCPU. If NUMCPU is less than MAXCPU then the remaining engines can be configured online later. The default NUMCPU value is 1. All processors are CP engines unless otherwise specified by the ENGINES system parameter.

Multiprocessor emulation works best if your host system actually has more than one physical CPU, but you can still emulate multiple CPUs nevertheless even on a uniprocessor system (and you might even achieve a small performance benefit when you do).

There is little point, however, in specifying NUMCPU greater than 1 unless your guest operating system (running under Hercules) is actually able to support multiple CPUs. If you do not actually need multiprocessor emulation, then setting MAX_CPU_ENGINES to 1 at compile time might even produce a slight performance advantage too.

Given without an argument the NUMCPU command displays the current number of emulated CPUs.

For detailed explanations on the interrelationship between NUMCPU, MAXCPU and ENGINES please see "Appendix B. Configuration of Emulated CPUs".

8.112.2 Syntax

Descriptive

NUMCPU [*nn*]

Diagram

```
►►── NUMCPU ──┬────────┬──────────────────────────────────►◄
              └── nn ──┘
```

8.112.3 Parameter

nn The number of emulated CPUs. NUMCPU must be less than or equal MAXCPU. If NUMCPU is larger than MAXCPU then an error message is issued, if it is less than MAXCPU then the remaining engines can be configured online later.

8.112.4 Examples

Example 1:

Display the current number of emulated CPUs.

```
HHC00013I Herc command: 'numcpu'
HHC02203I numcpu        : 2
```

Figure 213: NUMCPU command (display current number of emulated CPUs)

Example 2:

Set the number of emulated CPUs to 4 (increase from 2 currently).

```
HHC00013I Herc command: 'numcpu 4'
HHC00100I Thread id 0000080C, prio  0, name 'Processor CP02' started
HHC00811I Processor CP02: architecture mode 'S/370'
HHC00100I Thread id 00000CB4, prio  0, name 'Processor CP03' started
HHC00811I Processor CP03: architecture mode 'S/370'
HHC02204I numcpu         set to 4
```

Figure 214: NUMCPU command (increase emulated CPUs)

Example 3:

Set the number of emulated CPUs to (reduce from 4 currently).

```
HHC00013I Herc command: 'numcpu 2'
HHC00101I Thread id 0000080C, prio  0, name 'Processor CP02' ended
HHC00101I Thread id 00000CB4, prio  0, name 'Processor CP03' ended
HHC02204I numcpu         set to 2
```

Figure 215: NUMCPU command (reduce emulated CPUs)

8.113 NUMVEC (Display or set number of vector facilities)

8.113.1 Function

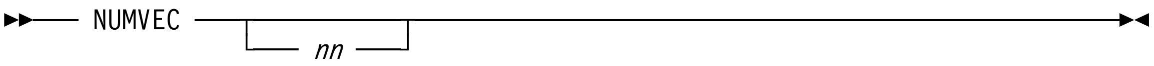
The NUMVEC command displays or sets the number of emulated vector facilities. The vector facility is only available in ESA/390 mode by default. Given without an argument the NUMVEC command displays the current number of vector facilities.

8.113.2 Syntax

Descriptive

NUMVEC [*nn*]

Diagram



8.113.3 Parameter

nn This is the number of desired vector facilities.

8.113.4 Examples

Example 1:

Display the current number of vector facilities.

```
HHC00013I Herc command: 'numvec'
HHC02203I numvec        : 1
```

Figure 216: NUMVEC command (display current number of vector facilities)

Example 2:

Set the number of vector facilities to 2.

```
HHC00013I Herc command: 'numvec 2'
HHC02204I numvec         set to 2
```

Figure 217: NUMVEC command (set number of vector facilities)

8.114 OSTAILOR (Tailor trace information for specific operating system)

8.114.1 Function

The OSTAILOR command lets you display or (re-)specify the intended operating system. The effect of this parameter is to reduce control panel message traffic by selectively suppressing trace messages for program checks which are considered normal in the specified environment.

The argument *QUIET* suppresses all exception messages, whereas the argument *NULL* suppresses none of them. The other options do suppress some messages and do not supress other messages depending on the specified operating system.

Prefix values with a plus (“+”) to combine them with existing values or with a minus (“-“) to exclude them from existing values. See also the PGMTRACE console command which allows you to further fine tune the tracing of program interrupt exceptions.

8.114.2 Syntax

Descriptive

```
OSTAILOR [[+ | -] z/OS | OS/390 | VM | z/VSE | VSE | LINUX | OPENSOLARIS |
                  QUIET | NULL]
```

Diagram

```
►►── OSTAILOR ──┬──────┬──┬───── z/OS ──────┬──────────────────────►◄
                ├─ + ──┤  ├──── OS/390 ─────┤
                └─ - ──┘  ├────── VM ───────┤
                          ├────── VSE ──────┤
                          ├───── z/VSE ─────┤
                          ├───── LINUX ─────┤
                          ├── OPENSOLARIS ──┤
                          ├───── QUIET ─────┤
                          └───── NULL ──────┘
```

8.114.3 Parameter

+ Specifies to combine the value withexisting values.

- Specifies to exclude the value from existing values.

z/OS Code z/OS if you intend to run z/OS.

OS/390 Code OS/390 if you intend to run MVS/370, MVS/XA, MVS/ESA, OS/390.

VM Code VM if you intend to run VM/370, VM/ESA or z/VM.

VSE Code VSE if you intend to run VSE/370 or VSE/ESA.

z/VSE Code z/VSE if you intend to run z/VSE.

LINUX Code Linux if you intend to run Linux/390 or Linux for z/Series.

OpenSolaris Code OpenSolaris if you intend to run OpenSolaris for z/Series.

QUIET QUIET discards all exception messages.

NULL NULL allows all exception messages to be logged.

8.114.4 Examples

Example 1:

Display the currently specified intended operating system.

```
HHC00013I Herc command: 'ostailor'
HHC02203I ostailor      : z/OS
```

Figure 218: OSTAILOR command (display intended operating system)

Example 2:

Change the specified intended operating system to Linux.

```
HHC00013I Herc command: 'ostailor linux'

HHC00013I Herc command: 'ostailor'
HHC02203I ostailor      : LINUX
```

Figure 219: OSTAILOR command (specify intended operating system)

Example 3:

Set the specified intended operating system to VM and combine it with z/OS and VSE, then display the currently specified value(s).

```
HHC00013I Herc command: 'ostailor vm'
HHC00013I Herc command: 'ostailor +z/OS'
HHC00013I Herc command: 'ostailor +vse'

HHC00013I Herc command: 'ostailor'
HHC02203I ostailor      : Custom(0x7B7673FFF7DE7FB4)
```

Figure 220: OSTAILOR command (combine intended operating systems)

Example 4:

Display the currently specified intended operating system(s) and exclude z/OS from the existing values. Then display again the currently specified value(s).

```
HHC00013I Herc command: 'ostailor'
HHC02203I ostailor      : Custom(0x7B7673FFF7DE7FB4)

HHC00013I Herc command: 'ostailor -z/OS'

HHC00013I Herc command: 'ostailor'
HHC02203I ostailor      : Custom(0xFFFFFFFFFFFFFFFC)
```

Figure 221: OSTAILOR command (combine intended operating systems)

8.115 PANRATE (Display or set console refresh rate)

8.115.1 Function

The PANRATE command shows the current setting of the console (panel) refresh rate and allows a new value to be set.

8.115.2 Syntax

Descriptive

PANRATE [SLOW | FAST | *rate*]

Diagram

```
►►── PANRATE ──┬──────────┬──────────────────────────────────►◄
               ├── SLOW ──┤
               ├── FAST ──┤
               └── rate ──┘
```

8.115.3 Parameter

SLOW SLOW is a synonym for a panel refresh rate of 500 milliseconds.

FAST FAST is a synonym for a panel refresh rate of 50 milliseconds.

rate Any value between 1 (10) and 5000 milliseconds. A value less than the Linux system clock tick interval (10 on Intel platforms, 1 on Alpha platforms) or a value of more than 5000 will be rejected.

8.115.4 Examples

Example 1:

Display current panel refresh rate.

```
HHC00013I Herc command: 'panrate'
HHC02203I panrate       : 1000
```

Figure 222: PANRATE command (list current panel refresh rate)

Example 2:

Set panel refresh rate to 500 milliseconds.

```
HHC00013I Herc command: 'panrate 500'
HHC02204I panrate        set to 500
```

Figure 223: PANRATE command (set new panel refresh rate)

8.116 PANTITLE (Display or set console window title)

8.116.1 Function

The PANTITLE command displays or sets an optional console window title-bar string to be used in place of the default title supplied by the windowing system. This option allows one to distinguish between different Hercules sessions when running more than one instance of Hercules on the same machine. Given without an argument the PANTITLE command displays the current console window title.

The PANTITLE option takes effect only when the Hercules console is displayed on an ‘xterm’ terminal (commonly used on Unix systems) or in a Windows command prompt window. Note that this option has no effect when Hercules is run under the control of the Hercules Windows GUI since Hercules's console window is hidden in favour of using the GUI's window instead.

The default console title is a string consisting of the following information:

“LPARNAME – SYSTYPE * *SYSNAME* * *SYSPLEX* – System Status: *colour*”

SYSTYPE, SYSNAME and SYSPLEX are populated by the system call SCLP Control Program Identification. If any of these values is blank, then that field is not presented in the console title. The system status colour has following meanings:

RED One or more CPUs are in a disabled wait state.

AMBER One or more CPUs are not running.

GREEN Everything is working correctly.

8.116.2 Syntax

Descriptive

```
PANTITLE [text | "text text text" | ""
```

Diagram

```
►►── PANTITLE ──┬─────────────────────────┬──────────────────────────►◄
                ├───────── text ──────────┤
                ├── "text text text" ─────┤
                └────────── "" ───────────┘
```

8.116.3 Parameter

text Specifies the optional console window title-bar string to be used. If the value contains any blanks it must be enclosed within double-quotes ("). An empty string ("") will remove the default console title.

8.116.4 Examples

Example 1:

Set the console window title to “Hercules Emulator HMC”.

```
HHC00013I Herc command: 'pantitle "Hercules Emulator HMC"'
HHC02204I pantitle       set to Hercules Emulator HMC
```

Figure 224: PANTITLE command (set a new console window title)

Example 2:

Remove the current window console title.

```
HHC00013I Herc command: 'pantitle ""'
HHC02204I pantitle       set to (none)
```

Figure 225: PANTITLE command (remove console window title)

8.117 PGMPRDOS (Set LPP license setting)

8.117.1 Function

The PGMPRDOS command specifies whether or not Hercules will run licensed program product (LPP) ESA or z/Architecture operating systems.

8.117.2 Syntax

Descriptive

```
PGMPRDOS {RESTRICTED | LICENSED}
```

Diagram

```
►►── PGMPRDOS ───┬── RESTRICTED ──┬──────────────────────────►◄
                 └── LICENSED ────┘
```

8.117.3 Parameter

RESTRICTED When PGMPRDOS is set to RESTRICTED, Hercules will stop all CPUs when a licensed program product operating systems is detected. RESTRICTED is the default.

LICENSED Setting PGMPRDOS to LICENSED will allow you to run licensed program product operating systems normally. This parameter has no effect on Linux/390, Linux for z/Series, or any 370-mode operating system. If you are running Hercules under the Windows GUI a pop up window appears during startup which must be acknowledged before the startup continuous.

8.117.4 Examples

Example 1:

Allow licensed program product operating systems to run normally.

```
HHC00013I Herc command: 'pgmprdos licensed'
```

Figure 226: PGMPRDOS command

8.118 PGMTRACE (Trace program interrupts)

8.118.1 Function

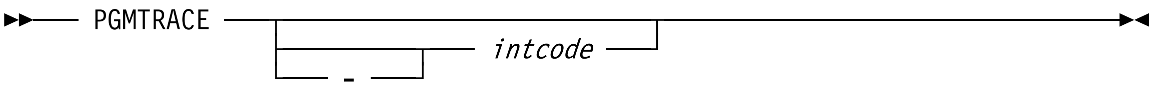
The PGMTRACE command without arguments displays the actual trace program interrupt options. Given with an argument (interruption code) the command changes the current options. Precede the interruption code with a minus sign to stop tracing of that particular interruption code. See also the OSTAILOR console command which allows you to further fine tune the tracing of program interrupt exceptions.

8.118.2 Syntax

Descriptive

`PGMTRACE [[-]`*`intcode`*`]`

Diagram



8.118.3 Parameter

intcode Specifies the interruption code for which trace information is to be written to the Hercules log or for which the trace information is to be stopped. *intcode* has to be a valid program interruption code in the range from 0x01 to 0x40.

- A minus sign, directly preceding the *intcode* parameter, stops tracing for that particular interruption code.

8.118.4 Examples

Example 1:

Display the current program interruption trace settings.

```
HHC00013I Herc command: 'pgmtrace'
HHC02281I * = Tracing suppressed; otherwise tracing enabled
HHC02281I 0000000000000001111111111111111222222222222222233333333333333334
HHC02281I 123456789ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF0
HHC02281I  *             **    *     *              **   **  *           *
```

Figure 227: PGMTRACE command (display settings)

Example 2:

Change the current program interruption trace settings.

```
HHC00013I Herc command: 'pgmtrace 04'
HHC00013I Herc command: 'pgmtrace'
HHC02281I * = Tracing suppressed; otherwise tracing enabled
HHC02281I 0000000000000001111111111111111222222222222222233333333333333334
HHC02281I 123456789ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF0
HHC02281I       *        **    *     *              **   **  *   *  *    *
```

Figure 228: PGMTRACE command (change settings)

8.119 PLANT (Display or set STSI plant code)

8.119.1 Function

The PLANT console command displays or sets the plant name returned by the STSI instruction. If no argument is given, the current plant name is displayed.

8.119.2 Syntax

Descriptive

PLANT [*name*]

Diagram

```
►►── PLANT ──┬──────────┬──────────────────────────────►◄
             └── name ──┘
```

8.119.3 Parameter

name Any name with a maximum length of four characters.

8.119.4 Examples

Example 1:

Set the STSI plant name to ‘ZZ’.

```
HHC00013I Herc command: 'plant ZZ'
HHC02204I plant          set to ZZ
```

Figure 229: PLANT command (set STSI plant name)

8.120 PR (Display prefix register)

8.120.1 Function

The PR command displays the contents of the prefix registers.

8.120.2 Syntax

Descriptive

```
PR
```

Diagram

```
►►—— PR ——————————————————————————————————————————►◄
```

8.120.3 Parameter

None.

8.120.4 Examples

Example 1:

Display the prefix register.

```
HHC00013I Herc command: 'pr'
HHC02277I Prefix register: 7FA03000
```

Figure 230: PR command

8.121 PSCP (Send system control program priority message)

8.121.1 Function

The PSCP command sends a system control program (i.e. guest operating system) priority command in any CMDTGT mode.

8.121.2 Syntax

Descriptive

`PSCP [cmd]`

Diagram

```
►►── PSCP ──┬──────────┬──────────────────────────────────►◄
            └── cmd ───┘
```

8.121.3 Parameter

cmd This is the command to be sent as a system control program (i.e. guest operating system) priority command.

8.121.4 Examples

For a similar sample please refer to the SCP command.

8.122 PSW (Display or alter program status word)

8.122.1 Function

The PSW command displays or alters the actual content of the Program Status Word (PSW).

8.122.2 Syntax

Descriptive

PSW [*operand=value* [*operand=value* ...]]

where operand can be:

SM=*xx*

PK=*nn*

CMWP=*x*

AS=[PRI | SEC | HOME]

CC=*n*

PM=*x*

IA=*xxxxxxxx*

AM=[24 | 31 | 64]

Diagram

```
PSW
        operand=value
```

where operand can be:

```
SM=xx
PK=nn
CMWP=x
```

```
├── AS= ──┬── PRI ──────────────────────────────────────────────────┤
          ├── SEC ──┤
          └── HOME ─┘

├── CC=n ───────────────────────────────────────────────────────────┤

├── PM=x ───────────────────────────────────────────────────────────┤

├── IA=xxxxxxxx ────────────────────────────────────────────────────┤

├── AM= ──┬── 24 ───────────────────────────────────────────────────┤
          ├── 31 ──┤
          └── 64 ──┘
```

8.122.3 Parameter

SM=*xx*	Modifies the PSW system mask (xx is 2 hexadecimal digits).
PK=*nn*	Modifies the PSW protection key (nn is decimal 0 to 15).
CMWP=*x*	Modifies the EC/M/W/P bits of the PSW (x is 1 hexadecimal digit).
AS=*aspace*	This modifies the PSW address space control bits. The address space aspace must be ‘PRI’, ‘SEC’ or ‘HOME’.
CC=*n*	Modifies the PSW condition code (n is decimal 0 to 3).
PM=*x*	Modifies the PSW program mask (x is one hexadecimal digit).
IA=*xxxxxxxx*	Modifies the PSW instruction address (*xxxxxxxx* is one to 16 hexadecimal digits).
AM=*amode*	Modifies the addressing mode bits of the PSW. The addressing mode *amode* must be 24, 31 or 64 bit.

8.122.4 Examples

Example 1:

Display the Program Status Word.

```
HHC00013I Herc command: 'psw'
HHC02278I Program status word: 070E000000000000
HHC02300I sm=07 pk=0 cmwp=E as=pri cc=0 pm=0 am=24 ia=0
```

Figure 231: PSW command (display PSW).

Example 2:

Change the PSW condition code to zero.

```
HHC00013I Herc command: 'psw cc=0'
HHC02278I Program status word: 070E000000000000
HHC02300I sm=07 pk=0 cmwp=E as=pri cc=0 pm=0 am=24 ia=0
```

Figure 232: PSW command (modify condition code)

Example 3:

Change the PSW condition code to zero and the addressing mode bits to AM=31.

```
HHC00013I Herc command: 'psw cc=0 am=31'
HHC02278I Program status word: 070E000080000000
HHC02300I sm=07 pk=0 cmwp=E as=pri cc=0 pm=0 am=31 ia=0
```

Figure 233: PSW command (modify condition code and addressing mode)

8.123 PTP (Enable / disable PTP debugging)

8.123.1 Function

The PTP command enables or disables debug tracing for the device group identified by *devnum* or for all device group(s) if *devnum* is not specified or specified as ‘ALL’.

8.123.2 Syntax

Descriptive

PTP DEBUG {ON | OFF} [*devnum* | ALL] [*mask*]

Diagram

```
                                       ┌──── ALL ────┐
►►── PTP ── DEBUG ──┬── ON ──┬─────────┼─────────────┼───┬────────────┬──►◄
                    └── OFF ─┘         └── devnum ───┘   └─── mask ───┘
```

8.123.3 Parameter

ON Enables the PTP debug tracing.

OFF Disables the PTP debug packet tracing.

devnum Specifies the PTP device group(s) for which debug tracing has to be enabled or disabled.

ALL Enables or disables the debug tracing for all PTP device groups. 'ALL' is the default if no device group is specified.

mask Specifies the type of debug output to be produced. *mask* is a value from 1 to 255. The following debug output can be selected:

- Xxx
- Xxx

8.123.4 Examples

Example 1:

Enable the debug tracing for all PTP devices. Please note that not all hex columns can be displayed in the figure below. Missing columns have been marked with “∫∫”.

```
HHC00013I Herc command: 'ptp debug on all'
HHC02204I PTP debug      set to on ALL
.
.
.
```

Figure 234: PTP command (enable debug tracing)

Example 2:

Disable the PTP debug tracing.

```
HHC00013I Herc command: 'ptp debug off'
HHC02204I PTP debug      set to off
```

Figure 235: PTP command (disable debug tracing)

8.124 PTT (Display or set internal trace)

8.124.1 Function

The PTT command sets or displays the internal trace options. When specified with no operands, the PTT command displays the trace options and the contents of the internal trace table.

When specified with operands, the PTT command sets the trace options and/or specifies which events are to be traced. If the last operand is numeric, it sets the size of the trace table and activates the trace.

The following events can be traced:

- Internal logger events
- Internal timer events
- Internal threading events
- Instruction information events
- Instruction error events
- Program interrupt events
- Interlocked instruction type events
- SIE instruction events
- SIGP instruction events
- I/O instruction events
- LCS timing events
- QETH timing events

The following options can be set:

- Lock table before updating
- Timestamp table entries
- Wraparound trace table
- Automatic display timeout
- Table size

8.124.2 Syntax

Descriptive

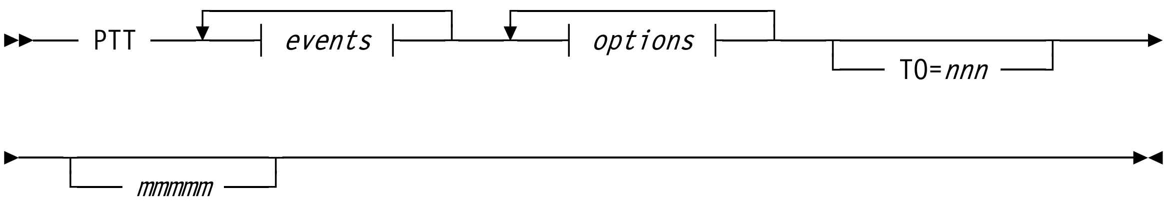
```
PTT [events] [options] [TO=nnn] [nnnnnn]

   where events can be:

[NO]LOG
[NO]TMR
[NO]THR
```

[NO]INF
[NO]ERR
[NO]PGM
[NO]CSF
[NO]SIE
[NO]SIG
[NO]IO
[NO]LCS
[NO]QETH

and where options can be:

[NO]LOCK
[NO]TOD
[NO]WRAP

Diagram



where events can be:

```
├──┬──────────┬── PGM ─────────────────────────────────────────────┤
   └── NO ────┘
├──┬──────────┬── CSF ─────────────────────────────────────────────┤
   └── NO ────┘
├──┬──────────┬── SIE ─────────────────────────────────────────────┤
   └── NO ────┘
├──┬──────────┬── SIG ─────────────────────────────────────────────┤
   └── NO ────┘
├──┬──────────┬── IO ──────────────────────────────────────────────┤
   └── NO ────┘
├──┬──────────┬── LCS ─────────────────────────────────────────────┤
   └── NO ────┘
├──┬──────────┬── QETH ────────────────────────────────────────────┤
   └── NO ────┘

  and where options can be:

├──┬──────────┬── LOCK ────────────────────────────────────────────┤
   └── NO ────┘
├──┬──────────┬── TOD ─────────────────────────────────────────────┤
   └── NO ────┘
├──┬──────────┬── WRAP ────────────────────────────────────────────┤
   └── NO ────┘
```

8.124.3 Parameter

NO*xxxxxxxx*	Deselects the specified trace entry type or deselects the specified option.
LOG	Selects the internal logger trace entries.
TMR	Selects the internal timer trace entries.
THR	Selects the internal threading trace entries.
INF	Selects the instruction information trace entries.
ERR	Selects the instruction error trace entries.
PGM	Selects the program interrupt trace entries.
CSF	Select the interlocked instruction type trace entries.
SIE	Selects the SIE instruction trace entries.

SIGP Selects the SIGP instruction trace entries.

IO Selects the I/O instruction trace entries.

LCS Selects the LCS timing trace entries.

QETH Selects the QETH timing trace entries.

LOCK Lock table before updating

TOD Timestamp table entries

WRAP Specifies that the trace table has be overwritten when it is full. As soon as the end of the table is reached, the next entry to be written to the table goes to the start and overlays the trace entry that was formerly there.

TO=*nnn* Timeout in seconds before the automatic issuance of the PTT command is to occur, to print (display) the internal trace table.

mmmmm Number of trace entries to be kept.

8.124.4 Examples

Example 1:

Display the current defaults, then set the pthread trace options and start the trace.

```
HHC00013I Herc command: 'ptt'
HHC90012I Pttrace: lock tod wrap to 0 0
HHC00013I Herc command: 'ptt prog inter signal io threads logger nowrap 10000'
```

Figure 236: PTT command (display trace options / set options and start trace)

Example 2:

Display the pthread trace entries.

```
HHC00013I Herc command: 'ptt'
cmdtab.c:342       05:23:33.429599 00001084 unlock       00820738 00000000 0
dyngui.c:145       05:23:33.429601 00001084 lock before  002334f8 00000000
dyngui.c:145       05:23:33.429602 00001084 lock after   002334f8 00000000 0
dyngui.c:175       05:23:33.429603 00001084 unlock       002334f8 00000000 0
dyngui.c:1818      05:23:33.429604 00001084 lock before  009473d8 00000000
dyngui.c:1818      05:23:33.429605 00001084 lock after   009473d8 00000000 0
dyngui.c:1821      05:23:33.429620 00001084 unlock       009473d8 00000000 0
dyngui.c:1818      05:23:33.429621 00001084 lock before  009473d8 00000000
dyngui.c:1818      05:23:33.429622 00001084 lock after   009473d8 00000000 0
dyngui.c:1821      05:23:33.429634 00001084 unlock       009473d8 00000000 0
logger.c:493       05:23:33.429906 00001118 unlock       0036a774 00000000 0
logger.c:504       05:23:33.429907 00001118 lock before  0036a774 00000000
logger.c:504       05:23:33.429909 00001118 lock after   0036a774 00000000 0
```

```
logger.c:505       05:23:33.429919 00001118 broadcast    00000000 0036a77c 0
logger.c:506       05:23:33.429921 00001118 unlock       0036a774 00000000 0
logger.c:121       05:23:33.429935 00000CE4 wait after   0036a774 0036a77c 0
logger.c:156       05:23:33.429936 00000CE4 unlock       0036a774 00000000 0
hao.c:589          05:23:33.429943 00000CE4 lock before  00830f18 00000000
hao.c:589          05:23:33.429944 00000CE4 lock after   00830f18 00000000 0
hao.c:610          05:23:33.429945 00000CE4 unlock       00830f18 00000000 0
logger.c:115       05:23:33.429946 00000CE4 lock before  0036a774 00000000
logger.c:115       05:23:33.429947 00000CE4 lock after   0036a774 00000000 0
logger.c:121       05:23:33.429948 00000CE4 wait before  0036a774 0036a77c
cpu.c:1744         05:23:33.435955 00000B34 wait after   00233b54 03426d94 0
cpu.c:1759         05:23:33.435957 00000B34 unlock       00233b54 00000000 0
cpu.c:1586         05:23:33.435959 00000B34 lock before  00233b54 00000000
cpu.c:1586         05:23:33.435960 00000B34 lock after   00233b54 00000000 0
external.c:54      05:23:33.435966 00000B34 *EXTINT      00001004 00000004 00000000
external.c:127     05:23:33.435968 00000B34 unlock       00233b54 00000000 0
control.c:6474     05:23:33.435981 00000B34 lock before  00233b54 00000000
control.c:6474     05:23:33.435982 00000B34 lock after   00233b54 00000000 0
control.c:6497     05:23:33.435986 00000B34 unlock       00233b54 00000000 0
control.c:4570     05:23:33.436008 00000B34 lock before  00233b54 00000000
control.c:4570     05:23:33.436009 00000B34 lock after   00233b54 00000000 0
control.c:4581     05:23:33.436011 00000B34 unlock       00233b54 00000000 0
control.c:4570     05:23:33.436028 00000B34 lock before  00233b54 00000000
control.c:4570     05:23:33.436029 00000B34 lock after   00233b54 00000000 0
control.c:4581     05:23:33.436031 00000B34 unlock       00233b54 00000000 0
control.c:4570     05:23:33.436069 00000B34 lock before  00233b54 00000000
control.c:4570     05:23:33.436070 00000B34 lock after   00233b54 00000000 0
control.c:4581     05:23:33.436072 00000B34 unlock       00233b54 00000000 0
control.c:4632     05:23:33.436287 00000B34 lock before  00233b54 00000000
control.c:4632     05:23:33.436288 00000B34 lock after   00233b54 00000000 0
control.c:4642     05:23:33.436290 00000B34 unlock       00233b54 00000000 0
cpu.c:1586         05:23:33.436291 00000B34 lock before  00233b54 00000000
cpu.c:1586         05:23:33.436292 00000B34 lock after   00233b54 00000000 0
cpu.c:1744         05:23:33.436294 00000B34 wait before  00233b54 03426d94
cpu.c:1744         05:23:33.446986 00000B34 wait after   00233b54 03426d94 0
```

Figure 237: PTT command (display trace entries)

Example 3:

Set the pthread trace options, start the trace and issue automatic display of the trace entries after 2 minutes.

```
HHC90012I Pttrace: prog inter signal io threads logger lock tod nowrap to 0 10000
HHC00013I Herc command: 'ptt prog inter signal io threads logger nowrap to=120 10000'
HHC00100I Thread id 00001610, prio  0, name 'PTT timeout timer' started
pttrace.c:270      05:24:09.648613 00001084 lock before  0036aca0 00000000
pttrace.c:270      05:24:09.648615 00001084 lock after   0036aca0 00000000 0
logger.c:493       05:24:09.648652 00001118 unlock       0036a774 00000000 0
logger.c:504       05:24:09.648654 00001118 lock before  0036a774 00000000
logger.c:504       05:24:09.648655 00001118 lock after   0036a774 00000000 0
logger.c:505       05:24:09.648667 00001118 broadcast    00000000 0036a77c 0
```

```
logger.c:506       05:24:09.648668 00001118 unlock       0036a774 00000000 0
logger.c:121       05:24:09.648747 00000CE4 wait after   0036a774 0036a77c 0
logger.c:156       05:24:09.648749 00000CE4 unlock       0036a774 00000000 0
hao.c:589          05:24:09.648758 00000CE4 lock before  00830f18 00000000
hao.c:589          05:24:09.648759 00000CE4 lock after   00830f18 00000000 0
hao.c:610          05:24:09.648761 00000CE4 unlock       00830f18 00000000 0
logger.c:115       05:24:09.648763 00000CE4 lock before  0036a774 00000000
logger.c:115       05:24:09.648764 00000CE4 lock after   0036a774 00000000 0
logger.c:121       05:24:09.648765 00000CE4 wait before  0036a774 0036a77c
pttrace.c:272      05:24:09.648844 00001084 create       00001610 00000000 0
pttrace.c:275      05:24:09.648845 00001084 unlock       0036aca0 00000000 0
cmdtab.c:342       05:24:09.648848 00001084 unlock       00820738 00000000 0
dyngui.c:145       05:24:09.648851 00001084 lock before  002334f8 00000000
dyngui.c:145       05:24:09.648852 00001084 lock after   002334f8 00000000 0
dyngui.c:175       05:24:09.648854 00001084 unlock       002334f8 00000000 0
dyngui.c:1818      05:24:09.648856 00001084 lock before  009473d8 00000000
dyngui.c:1818      05:24:09.648857 00001084 lock after   009473d8 00000000 0
dyngui.c:1821      05:24:09.648892 00001084 unlock       009473d8 00000000 0
dyngui.c:1818      05:24:09.648893 00001084 lock before  009473d8 00000000
dyngui.c:1818      05:24:09.648894 00001084 lock after   009473d8 00000000 0
dyngui.c:1821      05:24:09.648929 00001084 unlock       009473d8 00000000 0
.
.
.
control.c:4642     05:24:11.878330 00000B50 unlock       00233b54 00000000 0
cpu.c:1586         05:24:11.878333 00000B50 lock before  00233b54 00000000
cpu.c:1586         05:24:11.878334 00000B50 lock after   00233b54 00000000 0
cpu.c:1744         05:24:11.878336 00000B50 wait before  00233b54 030a6d94
cpu.c:1744         05:24:11.885597 00000FE8 wait after   00233b54 031a6d94 0
cpu.c:1759         05:24:11.885599 00000FE8 unlock       00233b54 00000000 0
cpu.c:1586         05:24:11.885601 00000FE8 lock before  00233b54 00000000
cpu.c:1586         05:24:11.885602 00000FE8 lock after   00233b54 00000000 0
external.c:54      05:24:11.885607 00000FE8 *EXTINT      00001004 00000002 00000000
external.c:127     05:24:11.885609 00000FE8 unlock       00233b54 00000000 0
control.c:6474     05:24:11.885622 00000FE8 lock before  00233b54 00000000
control.c:6474     05:24:11.885623 00000FE8 lock after   00233b54 00000000 0
control.c:6497     05:24:11.885627 00000FE8 unlock       00233b54 00000000 0
HHC00101I Thread id 00001610, prio  0, name 'PTT timeout timer' ended
```

Figure 238: PTT command (start trace and issue automatic display)

8.125 PWD (Print working directory)

8.125.1 Function

The PWD command prints the current working directory.

8.125.2 Syntax

Descriptive

```
PWD
```

Diagram

```
►►—— PWD ——————————————————————————————————————————————————►◄
```

8.125.3 Parameter

None.

8.125.4 Examples

Example 1:

Print the current working directory.

```
HHC00013I Herc command: 'pwd'
HHC02204I working directory set to d:\hercules
```

Figure 239: PWD command

8.126 QCPUID (Display CPU ID)

8.126.1 Function

The QCPUID command displays the CPU ID and STSI results presented to the SCP.

8.126.2 Syntax

Descriptive

```
QCPUID
```

Diagram

```
►►─── QCPUID ──────────────────────────────────────────────►◄
```

8.126.3 Parameter

None.

8.126.4 Examples

Example 1:

Display the default CPUID.

```
HHC00013I Herc command: 'qcpuid'
HHC17004I CPUID  = 1900196370600000
HHC17005I CPC SI = 7060.7060-H70.HRC.ZZ.0000000000001963
```

Figure 240: QCPUID command

8.127 QD (Query device information)

8.127.1 Function

The QD command queries one or all devices or device classes depending on the given argument. The device number can be either a single device number or a multiple device number specification. The device class is either CHAN, CON, CTCA, DASD, DSP, FCP, LINE, OSA, PCH, PRT, RDR or TAPE. If no argument is given all devices are queried.

8.127.2 Syntax

Descriptive

QD [*devnum* | *devclass*]

Diagram

```
►►── QD ──┬──────────────┬──────────────────────────────────►◄
          ├── devnum ────┤
          └── devclass ──┘
```

8.127.3 Parameter

devnum The device number or multiple device number specification of the device(s) to be queried.

Devclass Specifies the device class to be queried. The device class is either CHAN, CON, CTCA, DASD, DSP, FCP, LINE, OSA, PCH, PRT, RDR or TAPE

8.127.4 Examples

Example 1:

Query DASD device with address 0AE0.

```
HHC00013I Herc command: 'qd 0148'
HHC02280I 0:0148 SNSID 00  FF383002 335000
HHC02280I 0:0148 RDC   00  38300233 50000000 00002000 022B001E
HHC02280I              10  80004B36 00B90000 00000000 022B0096
HHC02280I              20  00000000 00000000 00000002 4B360001
HHC02280I              30  00000000 00000000 00FF0000 00000000
HHC02280I 0:0148 RCD   00  C4010100 4040F3F3 F5F0F0F0 F0C8D9C3 |D...  3350000HRC|
HHC02280I              10  E9E9F0F0 F0F0F0F0 F0F0F0F0 F0F10300 |ZZ000000000001..|
```

```
HHC02280I               20  C4000000 4040F3F3 F5F0F0F0 F0C8D9C3 |D...  3350000HRC|
HHC02280I               30  E9E9F0F0 F0F0F0F0 F0F0F0F0 F0F10300 |ZZ000000000001..|
HHC02280I               40  D4020000 4040F3F8 F3F0F0F0 F2C8D9C3 |M...  3830002HRC|
HHC02280I               50  E9E9F0F0 F0F0F0F0 F0F0F0F0 F0F10300 |ZZ000000000001..|
HHC02280I               60  F0000001 4040F3F8 F3F04040 40C8D9C3 |0...  3830   HRC|
HHC02280I               70  E9E9F0F0 F0F0F0F0 F0F0F0F0 F0F10300 |ZZ000000000001..|
HHC02280I               80  00000000 00000000 00000000 00000000 |................|


HHC02280I               90  00000000 00000000 00000000 00000000 |................|
HHC02280I               A0  00000000 00000000 00000000 00000000 |................|
HHC02280I               B0  00000000 00000000 00000000 00000000 |................|
HHC02280I               C0  00000000 00000000 00000000 00000000 |................|
HHC02280I               D0  00000000 00000000 00000000 00000000 |................|
HHC02280I               E0  80000002 00001E00 01408048 48480200 |......... ......|
HHC02280I               F0  00808048 00000000 00000000 00000000 |................|
HHC02280I 0:0148 SNSS  00  00481F00 00000000 00000000 00000000
HHC02280I               10  00000000 00000000 00000000 00000000
HHC02280I               20  00000000 00000140
```

Figure 241: QD command

8.127.5 Explanations for device class DASD

The Sense ID (SNSID) is describing the type and the model number of the subsystem and logical volume of the channel and has the following format:

Bytes	**Description**
0	---
1-2	Subsystem Type
3	Subsystem Model and Architecture
4-5	Device Type
6	Device Model
7	Reserved
8-11	Command Interface Word (CIW) for Read Configuration Data
12-15	Command Interface Word (CIW) for Set-Interface-Identifier
16-19	Command Interface Word (CIW) for Read-Node-Identifier

Table 21: Sense ID

The Read Device Characteristics (RDC) defines the characteristics of the logical volume. The format is the following:

Bytes	Description
0-1	Subsystem type
2	Subsystem model number and architecture
3-4	Device type
5	Device model
6-9	Subsystem and device facilities
10	Device class code
11	Device type code
12-13	Number of primary cylinders
14-15	Tracks per cylinder
16	Number of sectors
17-19	Track length
20-21	Length of HA and R0
22	Track capacity calculation formula
23-27	Track capacity calculation factors F1 through F5
28-29	First alternate cylinder address
30-31	Number of alternate tracks
32-33	First diagnostic cylinder address
34-35	Number of diagnostic tracks
36-37	First device support cylinder address
38-39	number of device support tracks
40	MDR record ID
41	OBR record ID
42	Control unit type code
43	Read trackset parameter length
44-45	Maximum record zero length
46	Reserved
47	Track set size
48	Track capacity calculation factor F6
49-50	RPS sector calculation factors
51-53	Reserved
54	Device and control unit features
55	Reserved
56	Real control unit type code
57	Real device type code
58-63	Reserved

Table 22: Read Device Characteristics

The Read Configuration Data (RCD) contains configuration data. It shows how the internal disk subsystem is configured. The RCD has the following format:

Bytes	**Name**	**Description**
000-031	NED1	Volume NED (logical volume within the subsystem)
032-063	NED2	NED for the RAID array
064-095	NED3	NED for the logical subsystem
096-127	NED4	Token NED (establishes a relationship among all configu-ration data records in a control unit in the subsystem)
128-223		Reserved
224-255	QNEQ	General NEQ (describes paths and addresses)

Table 23: Read Configuration Data

Legend:

NED Node Element Descriptor

NEQ Node Element Qualifier

8.128 QPFKEYS (Display the current PF key settings)

8.128.1 Function

The QPFKEYS command displays the current PF key settings.

8.128.2 Syntax

Descriptive

```
QPFKEYS
```

Diagram

```
►►── QPFKEYS ──────────────────────────────────────────────►◄
```

8.128.3 Parameter

None.

8.128.4 Examples

Example 1:

Display the current PF key settings.

```
HHC00013I Herc command: 'qpfkeys'
HHC17199I PF01 'SUBST IMMED herc help &0'
HHC17199I PF02 'SUBST IMMED herc hst &0'
HHC17199I PF03 'UNDEFINED'
HHC17199I PF04 'UNDEFINED'
HHC17199I PF05 'IMMED herc maxrates'
HHC17199I PF06 'IMMED herc uptime'
HHC17199I PF07 'IMMED herc qproc'
HHC17199I PF08 'SUBST IMMED herc qd &0'
HHC17199I PF09 'UNDEFINED'
HHC17199I PF10 'UNDEFINED'
HHC17199I PF11 'SUBST DELAY herc &*'
HHC17199I PF12 'IMMED herc cmdtgt herc'
HHC17199I PF13 'UNDEFINED'
HHC17199I PF14 'UNDEFINED'
HHC17199I PF15 'SUBST IMMED herc sfd &0'
HHC17199I PF16 'SUBST IMMED herc sfc &0'
HHC17199I PF17 'SUBST IMMED herc sfk &0 &1'
```

```
HHC17199I PF18 'SUBST IMMED herc sf+ &0'
HHC17199I PF19 'SUBST IMMED herc sf- &0 &1'
HHC17199I PF20 'SUBST IMMED herc sf= &0 &1'
HHC17199I PF21 'UNDEFINED'
HHC17199I PF22 'UNDEFINED'
HHC17199I PF23 'SUBST DELAY scp &*'
HHC17199I PF24 'IMMED herc cmdtgt scp'
HHC17199I PF25 'UNDEFINED'
HHC17199I PF26 'UNDEFINED'
HHC17199I PF27 'UNDEFINED'
HHC17199I PF28 'UNDEFINED'
HHC17199I PF29 'UNDEFINED'
HHC17199I PF30 'UNDEFINED'
HHC17199I PF31 'UNDEFINED'
HHC17199I PF32 'UNDEFINED'
HHC17199I PF33 'UNDEFINED'
HHC17199I PF34 'UNDEFINED'
HHC17199I PF35 'SUBST DELAY pscp &*'
HHC17199I PF36 'IMMED herc cmdtgt pscp'
HHC17199I PF37 'UNDEFINED'
HHC17199I PF38 'UNDEFINED'
HHC17199I PF39 'UNDEFINED'
HHC17199I PF40 'UNDEFINED'
HHC17199I PF41 'UNDEFINED'
HHC17199I PF42 'UNDEFINED'
HHC17199I PF43 'UNDEFINED'
HHC17199I PF44 'UNDEFINED'
HHC17199I PF45 'UNDEFINED'
HHC17199I PF46 'UNDEFINED'
HHC17199I PF47 'UNDEFINED'
HHC17199I PF48 'UNDEFINED'
```

Figure 242: QPFKEYS command

8.129 QPID (Display process ID of Hercules)

8.129.1 Function

The QPID command displays the process ID of Hercules.

8.129.2 Syntax

Descriptive

```
QPID
```

Diagram

```
►►── QPID ──────────────────────────────────────────────────►◄
```

8.129.3 Parameter

None.

8.129.4 Examples

Example 1:

Display the process ID of Hercules.

```
HHC00013I Herc command: 'qpid'
HHC17013I Process ID = 3544
```

Figure 243: QPID command

8.130 QPORTS (Display TCP/IP ports in use)

8.130.1 Function

The QPORTS command displays the TCP/IP ports in use.

8.130.2 Syntax

Descriptive

```
QPORTS
```

Diagram

```
►►── QPORTS ──────────────────────────────────────────────►◄
```

8.130.3 Parameter

None.

8.130.4 Examples

Example 1:

Display the TCP/IP ports in use.

```
HHC00013I Herc command: 'qports'
HHC17001I Server '        http' is listening on port 8089
HHC17001I Server ' shared_dasd' is listening on port 3999
HHC17001I Server '     console' is listening on port 3278
```

Figure 244: QPORTS command

8.131 QPROC (Display processors type and utilization)

8.131.1 Function

The QPROC command displays the processors type and their utilization.

8.131.2 Syntax

Descriptive

QPROC

Diagram

```
►►── QPROC ──────────────────────────────────────────────►◄
```

8.131.3 Parameter

None.

8.131.4 Examples

Example 1:

Display the processors type and utilitzation.

```
HHC00013I Herc command: 'qproc'
HHC17007I NumCPU = 08, NumVEC = 00, ReservedCPU = 00, MaxCPU = 08
HHC17008I Avgproc  028% 08; MIPS[39.71]; SIOS[998]
HHC17009I PROC CP00 - 067%; MIPS[10.11]; SIOS[358] - Host Kernel(00:00:00.202) User(00:00:39.203)
HHC17009I PROC CP01 - 060%; MIPS[ 8.84]; SIOS[329] - Host Kernel(00:00:00.202) User(00:00:19.640)
HHC17009I PROC CP02 - 046%; MIPS[12.77]; SIOS[255] - Host Kernel(00:00:00.124) User(00:00:15.974)
HHC17009I PROC CP03 - 032%; MIPS[ 4.63]; SIOS[151] - Host Kernel(00:00:00.156) User(00:00:12.792)
HHC17009I PROC CP04 - 009%; MIPS[ 1.09]; SIOS[ 51] - Host Kernel(00:00:00.078) User(00:00:10.670)
HHC17009I PROC CP05 - 005%; MIPS[ 0.65]; SIOS[ 13] - Host Kernel(00:00:00.000) User(00:00:08.470)
HHC17009I PROC CP06 - 008%; MIPS[ 1.22]; SIOS[ 41] - Host Kernel(00:00:00.078) User(00:00:06.754)
HHC17009I PROC CP07 - 002%; MIPS[ 0.36]; SIOS[  8] - Host Kernel(00:00:00.031) User(00:00:06.598)
HHC17010I  - Started        : Stopping        * Stopped
```

Figure 245: QPROC command

8.132 QSTOR (Display main and expanded storage values)

8.132.1 Function

The QSTOR command displays the current main and expanded storage values.

8.132.2 Syntax

Descriptive

QSTOR

Diagram

```
►►── QSTOR ──────────────────────────────────────────────────►◄
```

8.132.3 Parameter

None.

8.132.4 Examples

Example 1:

Display the current main and expanded storage values.

```
HHC00013I Herc command: 'qstor'
HHC17003I MAIN     storage is 006 GBytes 'mainsize'
HHC17003I EXPANDED storage is 000 MBytes 'xpndsize'
```

Figure 246: QSTOR command

8.133 QUIET (Toggle automatic refresh of console display data)

8.133.1 Function

The QUIET command either disables automatic screen refreshing if it is currently enabled or enables it if it is currently disabled. When disabled you will not be able to see the response of any entered command nor any messages issued by the system nor be able to scroll the display, etc. Basically all screen updating is disabled. Entering 'quiet' again re-enables screen updating.

Please note that the QUIET command is disabled in the dyngui and therefore is not available when using the Hercules WinGUI.

8.133.2 Syntax

Descriptive

```
QUIET
```

Diagram

```
►►── QUIET ──────────────────────────────────────────────────────►◄
```

8.133.3 Parameter

None.

8.133.4 Examples

Example 1:

Change between automatic refresh of console display data.

```
HHC00013I Herc command: 'quiet'
HHC02203I automatic refresh: disabled
HHC00013I Herc command: 'quiet'
HHC02203I automatic refresh: enabled
```

Figure 247: QUIET command

8.134 QUIT (Terminate the emulator)

8.134.1 Function

The QUIT command (see also the EXIT command) initiates the Hercules shutdown. It terminates all threads, detaches all channels and devices and releases the configuration. Finally it terminates the emulator. If the guest OS has enabled “Signal Shutdown” then a signal shutdown request is sent to the guest OS and termination will begin after the guest OS has shutdown.

The QUIT command acts different depending on how Hercules was built. If Hercules was not compiled with the option 'OPTION_SHUTDOWN_CONFIRMATION' then the command acts as described above.

If Hercules however was built with option 'OPTION_SHUTDOWN_CONFIRMATION' then the following special processing for terminating the emulator takes place.

If QUIT is entered and the command level is not set to 'developer', 'debug' or 'all' then QUIT will first check that all online CPUs are stopped. If any CPU is not in the stopped state message HHC00069I is displayed indicating the number of CPUs still running.

Message HHC02266A follows that prompts for confirmation by entering a second QUIT command within a certain time period (default 10 seconds) to start termination of the emulator. If the time period has expired then the process starts over. This is to prevent an inadvertent shutdown of Hercules while still a guest OS is running.

The time period for the second QUIT command can be set to another value by using the 'QUITMOUT' console command or system parameter.

If all processors are stopped or the command level is set to 'developer', 'debug' or 'all' then quit stops Hercules immediately. 'QUIT FORCE' will also terminate the emulator immediately without any further checks.

8.134.2 Syntax

Descriptive

```
QUIT [FORCE]
```

Diagram

```
►►── QUIT ──┬───────────┬──────────────────────────────────────►◄
            └── FORCE ──┘
```

8.134.3 Parameter

FORCE Terminate the emulator immediately.

8.134.4 Examples

Example 1:

Initiate the Hercules shutdown.

```
HHC00013I Herc command: 'quit'
HHC00069I Guest is not quiesced; there are 8 CPUs active
HHC02266A Reenter command 'exit' again within 10 seconds to execute
HHC00013I Herc command: 'exit'
HHC01420I Begin Hercules shutdown
HHC01421I Releasing configuration
.
several lines not displayed
.
HHC01422I Configuration release complete
HHC01423I Calling termination routines
.
several lines not displayed
.
HHC02103I Logger: logger thread terminating
HHC01412I Hercules terminated
```

Figure 248: QUIT command

Example 2:

Initiate an immediate Hercules shutdown.

```
HHC00013I Herc command: 'quit force'
HHC01420I Begin Hercules shutdown
HHC01421I Releasing configuration
.
several lines not displayed
.
HHC01422I Configuration release complete
HHC01423I Calling termination routines
.
several lines not displayed
.
HHC02103I Logger: logger thread terminating
HHC01412I Hercules terminated
```

Figure 249: QUIT FORCE command

8.135 QUITMOUT (Display or set quit timeout value)

8.135.1 Function

The QUITMOUT console command is used to display or set the timeout value for a second QUIT, EXIT or SSD command if Hercules is built with the option “OPTION_SHUTDOWN_CONFIRMATION”. If Hercules is built without this option, then the QUITMOUT console command is not available. If QUITMOUT is given without argument it displays the current setting.

8.135.2 Syntax

Descriptive

QUITMOUT [*nn*]

Diagram

```
►►── QUITMOUT ──┬──────┬──────────────────────────────────►◄
                └─ nn ─┘
```

8.135.3 Parameter

nn This specifies the timeout value where *nn* must be in the range of 2 to 60 seconds. If the timeout value is 0 then no second QUIT, EXIT or SSD is necessary.

8.135.4 Examples

Example 1:

Display the current QUITMOUT value.

```
HHC01603I quitmout
HHC17100I Timeout value for 'quit' and 'ssd' is '10' seconds
```

Figure 250: QUITMOUT command

8.136 R (Display or alter real storage)

8.136.1 Function

The R command allows you to display or alter real storage. Up to 64K of real storage can be displayed, up to 32 bytes of real storage can be altered.

8.136.2 Syntax

Descriptive

R {*addr* | *addr.length* | *addr-addr* | *addr=value*}

Diagram

```
►►── R ──┬────── addr ───────┬──────────────────────────────────────────►◄
         ├── addr.length ────┤
         ├──── addr-addr ────┤
         └─── addr=value ────┘
```

8.136.3 Parameter

addr Specifies the address of the real storage that is to be displayed. If *addr* is given without a length or without a second address for the end of the storage area then 64 bytes of real storage are displayed.

addr.length Specifies the address of the real storage area that is to be displayed with starting address and length (from address *addr* with the length of *length*). The *length* value must be given in hexadecimal. The maximum length that can be specified is 64K.

addr-addr Specifies an address range with start and end address (from begin address to end address) of the real storage area that is to be displayed.

addr=value Specifies the address of the real storage area that is to be altered. *value* is a hex-string of up to 32 pairs of hex digits (32 bytes) which will be written to the real storage address given by the *addr* parameter. After altering the storage, 16 bytes of real storage starting at *addr* are displayed.

8.136.4 Examples

Example 1:

Display 256 bytes of real storage starting from location x’00000000’.

```
HHC00013I Herc command: 'r 00000000.ff'
HHC02290I R:00000000:K:06=040C0000 0001CEFA 00000000 00000000  ................
HHC02290I R:00000010:K:06=00009720 00000000 070E0000 00000000  ..p.............
HHC02290I R:00000020:K:06=070C0000 0003C160 070C1000 00077352  ......A-........
HHC02290I R:00000030:K:06=00000000 00000000 070E0000 00000000  ................
HHC02290I R:00000040:K:06=00088618 0C000000 00006318 00009720  ..f...........p.
HHC02290I R:00000050:K:06=FFCA28D9 00000000 040C0000 0001D09A  ...R..........}.
HHC02290I R:00000060:K:06=040C0000 0001D37A 000C0000 0001D9B4  ......L:......R.
HHC02290I R:00000070:K:06=00080000 0001E500 040C0000 0001E972  ......V.......Z.
HHC02290I R:00000080:K:06=00000000 00001004 00020074 00040011  ................
HHC02290I R:00000090:K:06=00091800 00000000 00000000 00000000  ................
HHC02290I R:000000A0:K:06=00000000 00000000 10000000 00FD41C8  ...............H
HHC02290I R:000000B0:K:06=FFFFFFFF 00000000 00000149 00000000  ................
HHC02290I R:000000C0:K:06=00000000 00000000 00000000 00000000  ................
HHC02290I R:000000D0:K:06=00000000 00000000 00000000 00000000  ................
HHC02290I R:000000E0:K:06=00000000 00000000 00000000 00000000  ................
HHC02290I R:000000F0:K:06=00000000 00000000 00000000 00000000  ................
```

Figure 251: R command (display real storage)

Example 2:

Alter 4 bytes of real storage at location x’00000000’ to x’FFFFFFFF’.

```
HHC00013I Herc command: 'r 00000000=FFFFFFFF'
HHC02290I R:00000000:K:06=FFFFFFFF 0001CEFA 00000000 00000000  ................
```

Figure 252: R command (alter real storage)

8.137 RESTART (Generate restart interrupt)

8.137.1 Function

The RESTART command generates a restart interrupt (the virtual RESTART key is pressed).

8.137.2 Syntax

Descriptive

```
RESTART
```

Diagram

```
►►── RESTART ──────────────────────────────────────────────►◄
```

8.137.3 Parameter

None.

8.137.4 Examples

Example 1:

Generate restart interrupt.

```
HHC00013I Herc command: 'restart'
HHC02228I Key 'restart' pressed
```

Figure 253: RESTART command

8.138 RESUME (Resume Hercules)

8.138.1 Function

This command resumes a Hercules session that had been previously suspended with the SUSPEND command (see section 8.171 for details). The data necessary to resume the session is read from a packed (zipped) file called "hercules.srf.gz" located in the current configuration directory.

In order for an instance to be resumed HERCULES must be started with a configuration file describing the configuration at suspend time. For example, MAINSIZE and XPNDSIZE must match and all devices present at suspend time must be present at resume time.

Disk devices must be at the same state as they were at suspend time. They can however be a different file type. For example a disk could be a CCKD disk at suspend time then a CKD disk could be created using DASDCOPY and HERCULES resumed using the CKD disk instead.

HERCULES must also be configured similarly as at suspend time. For example if 4 emulated CPUs were active at suspend time then the session cannot be resumed on a HERCULES with a maximum of two CPUs. Similarly you will not be able to resume a session in z/Architecture mode for a HERCULES that was built without z/Architecture.

After entering the RESUME command on the Hercules console the suspend file is re-imported and the CPUs are put in the STARTED state again resuming guest program operations at the same point and in the same state as the suspend file was created.

There are some caveats when resuming guest operating system processing:

- As seen by the guest operating system, the TOD clock will appear to jump a large value. Some guests may not cope very well with this. For example some guests may be dismayed because certain interrupts will occur way past its due time. Also for S/370 an interval timer interrupt may be lost if the guest is interrupted for more than half the Interval Timer wrap time (around 8 hours).
- Although some effort has been put in order to make this as transparent as possible (that is, it should appear to the guest operating system that the STOP key was pressed for a large amount of time), some state information may be missed.
- Some guest operating systems will fare better if the suspend state is prepared first. For MVS, as an example, it seems to help when a QUIESCE command and a SYSTEM RESTART manual operation are issued prior to suspend the system.

8.138.2 Syntax

Descriptive

```
RESUME
```

Diagram

```
►►── RESUME ─────────────────────────────────────────────►◄
```

8.138.3 Parameter

None.

8.138.4 Examples

Example 1:

Resume a Hercules session.

```
00:52:48 HHC00013I Herc command: 'resume'
00:52:48 HHC00811I Processor CP00: architecture mode 'S/370'
00:52:48 HHC00101I Thread id 00000D28, prio -15, name 'Processor CP00' ended
00:52:48 HHC00101I Thread id 000014DC, prio  0, name 'Processor CP01' ended
00:52:48 HHC00101I Thread id 00000CE8, prio  0, name 'Processor CP02' ended
00:52:48 HHC00101I Thread id 0000155C, prio  0, name 'Processor CP03' ended
00:52:48 HHC00101I Thread id 00001694, prio -20, name 'Timer' ended
00:52:48 HHC02007I SR: resuming suspended file 'Fri Nov 19 00:51:50 2010' created
```

Figure 254: RESUME command

8.139 REXX (Display or set REXX interpreter settings)

8.139.1 Function

The REXX console command is used to manage the Rexx interpreter settings. It allows to specify the paths where the Rexx executables can be found and what extensions for the executables are to be used. If Hercules is built with support for both Rexx environments (Regina Rexx and Open Object Rexx) then the environments can be dynamically enabled or disabled. The message prefixes to be used for Rexx messages (issued through the “say” command) and error messages (issued through the Rexx interpreter) can be set separately.

If the command is given without any arguments it displays the current Rexx interpreter settings.

8.139.2 Syntax

Descriptive

REXX [*option*]

where *option* can be:

ENABLE | START [REGINA | OOREXX]

DISABLE | STOP

PATHS | REXXPATHS {*path* [*delimiter path ...*] | RESET}

SYSPATH {ON | OFF | RESET}

EXTENSIONS | SUFFIXES {*suffix* [*delimiter suffix ...*] | RESET}

RESOLVER {ON | OFF | RESET}

MSGLEVEL {0 | 1 | RESET}

MSGPREFIX {*messageprefix* | OFF | RESET}

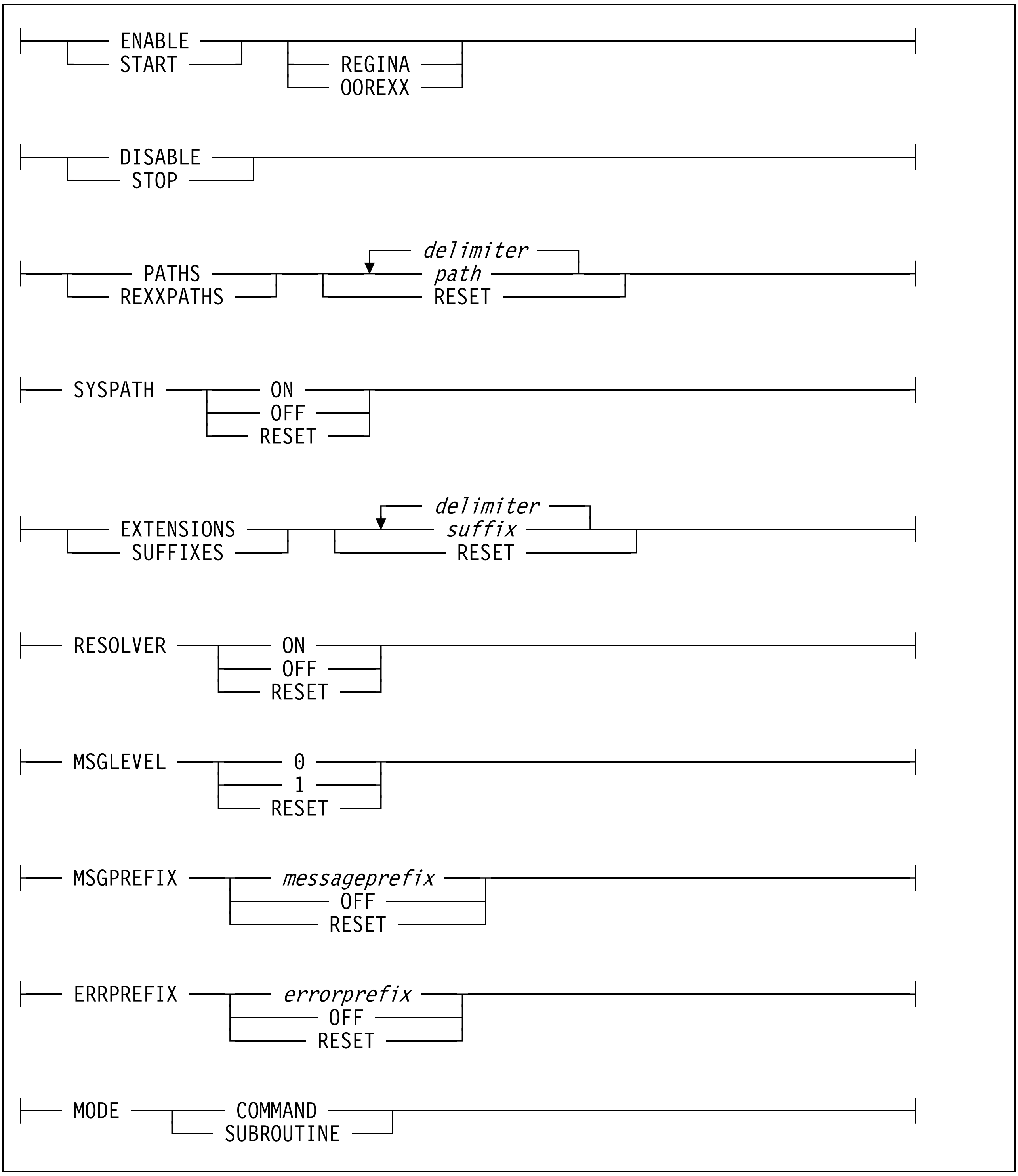
ERRPREFIX {*errorprefix* | OFF | RESET}

MODE {COMMAND | SUBROUTINE}

Diagram

```
►►── REXX ──┬──────────────┬──────────────────────────────────────►◄
            └──┤ option ├──┘
```

where *option* can be:



8.139.3 Parameter

ENABLE Enables the Rexx environment that is specified as argument (Regina or ooRexx). This option is only available if Hercules is built with support for both Regina and ooRexx and cannot be used in a single Rexx environment.

If no environment is given as argument then the default Rexx environment is started.

The default can be specified with the HREXX_PACKAGE environment variable (for details see chapter 14 “REXX Support”). If this variable is not set the ooRexx is the default package used.

ENABLE can be abbreviated as ‘ENA’.

START This is the same as ENABLE. START can be abbreviated as ‘STA’.

DISABLE Disables the currently active Rexx environment. This option is only available if Hercules is built with support for both Regina and ooRexx and cannot be used in a single Rexx environment.

DISABLE can be abbreviated as ‘DIS’.

STOP This is the same as DISABLE. STOP can be abbreviated as ‘STO’.

REGINA Given as argument to the ‘START’ or ‘ENABLE’ options, REGINA specifies that the Regina Rexx environment has to be started.

OOREXX Given as argument to the ‘START’ or ‘ENABLE’ option, OOREXX specifies that the Open Object Rexx (ooRexx) environment has to be started.

PATHS This is the keyword for specifying the search path(s) for the Rexx scripts. PATHS can be abbreviated as ‘PATH’.

REXXPATHS This is the same as PATHS. REXXPATHS can be abbreviated as ‘REXXP’.

path A path (or a list of paths, separated by a delimiter) in which Rexx executables will be searched. If the path is not specified when activating the Rexx environment then the default used is the current path taken from the environment variable ‘PATH’.

SYSPATH Keyword for specifying if the search for the Rexx executables should be extended to the system paths used for executables (when set to on ‘ON’) or if it should be limited to the defined PATHS / REXXPATHS (when set to ‘OFF’).

EXTENSIONS This is the keyword for specifying the filename extension(s) to be used to search the Rexx executables. EXTENSIONS can be abbreviated as ‘EXT’.

If a given script name is in the format *filename.extension* then it is used as is without any further processing

SUFFIXES This is the same as EXTENSIONS. SUFFIXES can be abbreviated as ‘SUF’.

suffix A filename extension (or a list of filename extensions, separated by a delimiter) that identifies a Rexx executable. A filename extension must be specified in the format “.*ext*”.

If there are no extensions specified the defaults used are ‘.REXX’, ‘.rexx’, ‘.REX’, ‘.rex’, ‘.CMD’, ‘.cmd’, ‘.RX’ and ‘.rx’.

delimiter This is the delimiter used for separating multiple paths or multiple extensions. For Linux and Mac OS-X systems this is the colon (“:”), for Windows systems it is the semicolon (“;”).

RESOLVER	Keyword to define who will resolve the script name. When set to "ON" then the Hercules Rexx interface will resolve the script name and issue appropriate messages in case the process fails. When set to "OFF" then the script name will be passed as is to the Rexx interpreter. For examples of the different behaviour of the EXEC command depending on the RESOLVER settings please see the next section.
MSGLEVEL	This is the keyword for specifying the message level to be used. MSGLEVEL can be abbreviated as 'MSGL'.
0	Disables the display of the HHC17503I and HHC17504I messages when a script has finished.
1	Enables the display of the HHC17503I and HHC17504I messages when a script has finished: HHC17503I REXX(*package name*) Exec/Script '*script name*' RetRC(0) HHC17504I REXX(*package name*) Exec/Script '*script name*' RetValue'0'
MSGPREFIX	This is the keyword used to set the prefix for standard messages (issued through 'say'). MSGPREFIX can be abbreviated as 'MSGP'.
msgprefix	Specifies the Rexx standard message prefix to be used. *msgprefix* can be any string up to 9 characters. Embedded blanks are not allowed.
ERRPREFIX	This is the keyword used to set the prefix for error messages. ERRPREFIX can be abbreviated as 'ERRP'.
errorprefix	Specifies the Rexx error message prefix to be used. *errorprefix* can be any string up to 9 characters. Embedded blanks are not allowed.
MODE	This is the keyword used to specify the argument passing style to a Rexx script.
COMMAND	Specifies command style for passing arguments to a Rexx script. 'COMMAND' may be abbreviated as 'COM'.
SUBROUTINE	Specifies subroutine style for passing arguments to a Rexx script. 'SUBROUTINE' may be abbreviated as 'SUB'.
ON	Activates the specified option.
OFF	Deactivates the specified option.
RESET	Given as an argument to one of the options of the Rexx command this will reset the corresponding value to the default settings.

8.139.4 Examples

Example 1:

Display the current Rexx interpreter settings (defaults).

```
HHC00013I Herc command: 'rexx'
HHC17500I REXX(Regina) Rexx Path ( 0) -
HHC17500I REXX(Regina) Sys Path  (18) - (ON)
HHC17500I REXX(Regina) Extensions( 9) - ;.REXX;.rexx;.REX;.rex;.CMD;.cmd;.RX;.rx
HHC17500I REXX(Regina) Resolver       - (ON)
HHC17500I REXX(Regina) Msg Level      - 0
HHC17500I REXX(Regina) Msg Prefix     - (OFF)
HHC17500I REXX(Regina) Err Prefix     - (OFF)
HHC17500I REXX(Regina) Mode           - (Command)
HHC17500I REXX(Regina) REXX-Regina_3.6(MT) 5.00 31 Dec 2011
HHC17500I REXX(Regina) WIN64 FUNCTION Instore
```

Figure 255: REXX command (display interpreter settings)

Example 2:

Set the path for Rexx executables to “d:\mvs\conf” and “d:\mvs\rexx”.

```
HHC00013I Herc command: 'rexx paths d:\mvs\conf;d:\mvs\rexx'
HHC17500I REXX(Regina) Rexx Path ( 2) -d:\mvs\conf;d:\mvs\rexx
HHC17500I REXX(Regina) Sys Path  (18) - (ON)
HHC17500I REXX(Regina) Extensions( 9) - ;.REXX;.rexx;.REX;.rex;.CMD;.cmd;.RX;.rx
HHC17500I REXX(Regina) Resolver       - (ON)
HHC17500I REXX(Regina) Msg Level      - 0
HHC17500I REXX(Regina) Msg Prefix     - (OFF)
HHC17500I REXX(Regina) Err Prefix     - (OFF)
HHC17500I REXX(Regina) Mode           - (Command)
HHC17500I REXX(Regina) REXX-Regina_3.6(MT) 5.00 31 Dec 2011
HHC17500I REXX(Regina) WIN64 FUNCTION Instore
```

Figure 256: REXX command (set PATH)

Example 3:

Set the extensions for Rexx executables to “*.rexx” and “*.rex”.

```
HHC00013I Herc command: 'rexx suffixes .rexx;.rex'
HHC17500I REXX(Regina) Rexx Path ( 2) -d:\mvs\conf;d:\mvs\rexx
HHC17500I REXX(Regina) Sys Path  (18) - (ON)
HHC17500I REXX(Regina) Extensions( 3) - ;.rexx;.rex
HHC17500I REXX(Regina) Resolver       - (ON)
HHC17500I REXX(Regina) Msg Level      - 0
HHC17500I REXX(Regina) Msg Prefix     - (OFF)
HHC17500I REXX(Regina) Err Prefix     - (OFF)
HHC17500I REXX(Regina) Mode           - (Command)
```

```
HHC17500I REXX(Regina) REXX-Regina_3.6(MT) 5.00 31 Dec 2011
HHC17500I REXX(Regina) WIN64 FUNCTION Instore
```

Figure 257: REXX command (set EXTENSION)

Example 4:

Set the prefix for Rexx error messages to “RXERR”.

```
HHC00013I Herc command: 'rexx errpref RXERR’
HHC17500I REXX(Regina) Rexx Path ( 2) -d:\mvs\conf;d:\mvs\rexx
HHC17500I REXX(Regina) Sys Path  (18) - (ON)
HHC17500I REXX(Regina) Extensions( 3) - ;.rexx;.rex
HHC17500I REXX(Regina) Resolver       - (ON)
HHC17500I REXX(Regina) Msg Level      - 0
HHC17500I REXX(Regina) Msg Prefix     - (OFF)
HHC17500I REXX(Regina) Err Prefix     - RXERR
HHC17500I REXX(Regina) Mode           - (Command)
HHC17500I REXX(Regina) REXX-Regina_3.6(MT) 5.00 31 Dec 2011
HHC17500I REXX(Regina) WIN64 FUNCTION Instore
```

Figure 258: REXX command (set ERRPREF)

Example 5:

Disable the error message prefixing.

```
HHC00013I Herc command: 'rexx errpref reset’
HHC17500I REXX(Regina) Rexx Path ( 2) -d:\mvs\conf;d:\mvs\rexx
HHC17500I REXX(Regina) Sys Path  (18) - (ON)
HHC17500I REXX(Regina) Extensions( 3) - ;.rexx;.rex
HHC17500I REXX(Regina) Resolver       - (ON)
HHC17500I REXX(Regina) Msg Level      - 0
HHC17500I REXX(Regina) Msg Prefix     - (OFF)
HHC17500I REXX(Regina) Err Prefix     - (OFF)
HHC17500I REXX(Regina) Mode           - (Command)
HHC17500I REXX(Regina) REXX-Regina_3.6(MT) 5.00 31 Dec 2011
HHC17500I REXX(Regina) WIN64 FUNCTION Instore
```

Figure 259: REXX command (disable ERRPREF)

Example 6:

Messages issued when a Rexx script could not be found and RESOLVER is set to ‘ON’.

```
HHC00013I Herc command: 'rexx resolver on’
HHC17500I REXX(Regina) Rexx Path ( 2) -d:\mvs\conf;d:\mvs\rexx
HHC17500I REXX(Regina) Sys Path  (18) - (ON)
HHC17500I REXX(Regina) Extensions( 3) - ;.rexx;.rex
```

```
HHC17500I REXX(Regina) Resolver       - (ON)
HHC17500I REXX(Regina) Msg Level      - 0
HHC17500I REXX(Regina) Msg Prefix     - (OFF)
HHC17500I REXX(Regina) Err Prefix     - (OFF)
HHC17500I REXX(Regina) Mode           - (Command)
HHC17500I REXX(Regina) REXX-Regina_3.6(MT) 5.00 31 Dec 2011
HHC17500I REXX(Regina) WIN64 FUNCTION Instore

HHC01603I exec samplerexx
HHC17506I REXX(Regina) Exec/script 'samplerexx' not found
```

Figure 260: REXX command (RESOLVER ON)

Example 7:

Messages issued when a Rexx script could not be found and RESOLVER is set to 'OFF'.

```
HHC00013I Herc command: 'rexx resolver off'
HHC17500I REXX(Regina) Rexx Path ( 2) -d:\mvs\conf;d:\mvs\rexx
HHC17500I REXX(Regina) Sys Path  (18) - (ON)
HHC17500I REXX(Regina) Extensions( 3) - ;.rexx;.rex
HHC17500I REXX(Regina) Resolver       - (OFF)
HHC17500I REXX(Regina) Msg Level      - 0
HHC17500I REXX(Regina) Msg Prefix     - (OFF)
HHC17500I REXX(Regina) Err Prefix     - (OFF)
HHC17500I REXX(Regina) Mode           - (Command)
HHC17500I REXX(Regina) REXX-Regina_3.6(MT) 5.00 31 Dec 2011
HHC17500I REXX(Regina) WIN64 FUNCTION Instore


Messages issued when using Regina Rexx:

HHC01603I exec samplerexx
HHC17506I REXX(Regina) Exec/script 'samplerexx' not found


Messages issued when using ooRexx:

HHC01603I exec samplerexx
REX0003E: Error 3: Failure during initialization
REX0534E: Error 3.901: Failure during initialization: Program "samplerexx" was not found
HHC17502E REXX(ooRexx) ooRexx RC(-3)
```

Figure 261: REXX command (RESOLVER OFF)

8.140 RMMOD (Delete a module)

8.140.1 Function

The RMMOD command deletes (unloads) modules.

8.140.2 Syntax

Descriptive

RMMOD *module* [*module* [*module* ...]]

Diagram

8.140.3 Parameter

module The names of the modules that have to be unloaded. If a device is still bound to a module then a message is issued and the module is not deleted.

8.140.4 Examples

Example 1:

Delete module “S37X” (S/370 Extension).

```
HHC00013I Herc command: 'rmmod s37x'
HHC01528I HDL: unloading module 's37x'...
HHC01529I HDL: module 's37x' unloaded
```

Figure 262: RMMOD command

8.141 S (Instruction stepping)

8.141.1 Function

This command sets the instruction stepping and breaking range. The instruction stepping is totally separate from the instruction tracing range (See T command).

A range can be specified. If there is no range then the range includes all addresses. “S 0” eliminates the range and all addresses will be stepped. With or without a given range, the S command displays whether instruction stepping is on or off and the range if any.

The S command by itself does not activate instruction stepping. Use the S+ command to activate instruction stepping.

8.141.2 Syntax

The S command has the following syntax:

Descriptive

S [*addr-addr* | *addr:addr* | *addr.length* | 0]

Diagram

```
►►── S ──┬───────────────────┬──────────────────────────────►◄
         ├──── addr-addr ────┤
         ├──── addr:addr ────┤
         ├─── addr.length ───┤
         └──────── 0 ────────┘
```

8.141.3 Parameter

addr-addr	Specifies an address range with start and end address (from begin address to end address).
addr:addr	Specifies an address range with start and end address (from begin address to end address).
addr.length	Specifies an address range with start and length (from begin address with the specified length).
0	No range is specified or an existing range is reset. Instruction stepping is set for all addresses.

8.141.4 Examples

Example 1:

Set instruction stepping range with begin and end address and query the range afterwards.

```
HHC00013I Herc command: 's 100000-101000'
HHC02229I Instruction stepping off range 100000-101000
HHC00013I Herc command: 's?'
HHC02229I Instruction stepping off range 100000-101000
```

Figure 263: S command (address range)

Example 2:

Set instruction stepping range with start address and length and query the range afterwards.

```
HHC00013I Herc command: 's 100000.1000'
HHC02229I Instruction stepping off range 100000.1000
HHC00013I Herc command: 's?'
HHC02229I Instruction stepping off range 100000-101000
```

Figure 264: S command (address with length)

Example 3:

Set instruction stepping for all addresses and query the range afterwards.

```
HHC00013I Herc command: 's 0'
HHC02229I Instruction stepping off
HHC00013I Herc command: 's?'
HHC02229I Instruction stepping off
```

Figure 265: S command (all addresses)

8.142 S+ (Instruction stepping on)

8.142.1 Function

This command turns on the instruction stepping. After turning instruction stepping on, each ENTER from the console allows Hercules to execute exactly one instruction and lists detailed trace information.

The trace information includes the executing CPU, the PSW, the executed instruction in hexadecimal and in disassembled form as well as register hexadecimal displays of all involved register types.

A range can be specified as for the “S” command, otherwise the existing range is used. If there is no range (or range was specified as 0) then the range includes all addresses.

When an instruction within the range is about to be executed, the CPU is temporarily stopped and the next instruction is displayed. You may then examine registers and/or storage etc. before you press Enter to execute the instruction and stop at the next instruction. To turn off instruction stepping and continue normal execution, enter the “G” command.

8.142.2 Syntax

Descriptive

S+ [*addr-addr* | *addr:addr* | *addr.length* | 0]

Diagram

```
►►── S+ ──┬──────────────────┬──────────────────────────────►◄
          ├─── addr-addr ────┤
          ├─── addr:addr ────┤
          ├── addr.length ───┤
          └──────── 0 ───────┘
```

8.142.3 Parameter

addr-addr Specifies an address range with start and end address (from begin address to end address).

addr:addr Specifies an address range with start and end address (from begin address to end address).

addr.length Specifies an address range with start and length (from begin address with the specified length).

0 No range is specified or an existing range is reset. Instruction stepping is set for all addresses.

8.142.4 Examples

Example 1:

Turn on instruction stepping.

```
HHC00013I Herc command: 's+'
HHC02229I Instruction stepping on
HHC00811I Processor CP00: architecture mode 'S/370'
HHC02267I CP00: PSW=0000014800000080 INST=05F0         BALR  15,0       branch_and_link_register
HHC02267I CP00: GR00=00000000 GR01=00000000 GR02=00000000 GR03=00000000
HHC02267I CP00: GR04=00000000 GR05=00000000 GR06=00000000 GR07=00000000
HHC02267I CP00: GR08=00000000 GR09=00000000 GR10=00000000 GR11=00000000
HHC02267I CP00: GR12=00000000 GR13=00000000 GR14=00000000 GR15=00000000
HHC00834I Processor CP00: running state selected
HHC02267I CP00: PSW=0000014800000082 INST=980E0010     LM    0,14,16(0)            load_multiple
HHC02267I CP00: R:00000010:K:06=00000000 00000000 00000000 00000000  ................
HHC02267I CP00: GR00=00000000 GR01=00000000 GR02=00000000 GR03=00000000
HHC02267I CP00: GR04=00000000 GR05=00000000 GR06=00000000 GR07=00000000
HHC02267I CP00: GR08=00000000 GR09=00000000 GR10=00000000 GR11=00000000
HHC02267I CP00: GR12=00000000 GR13=00000000 GR14=00000000 GR15=40000082
HHC00834I Processor CP00: running state selected
HHC02267I CP00: PSW=0000014880000086 INST=41A0F05A     LA    10,90(0,15)            load_address
HHC02267I CP00: R:000000DC:K:06=41800100 41A0F08A 1B984700 F06E5890  ......0..q..0>..
HHC02267I CP00: GR00=00000000 GR01=00000000 GR02=00000000 GR03=00000000
HHC02267I CP00: GR04=00000000 GR05=00000000 GR06=00000000 GR07=00000000
HHC02267I CP00: GR08=00000000 GR09=00000000 GR10=00000000 GR11=00000000
HHC02267I CP00: GR12=00000000 GR13=00000000 GR14=00000000 GR15=40000082
HHC00834I Processor CP00: running state selected
HHC02267I CP00: PSW=000001488000008A INST=D200FC7D0008 MVC   3197(1,15),8(0)      move_character
HHC02267I CP00: R:00000CFF:K:06=F1 E5D5C9D7 D458FFFF 76B3BD00 AAAAFF 1VNIPM.....]....
HHC02267I CP00: R:00000008:K:06=F1000000 00000000 00000000 00000000  1...............
HHC02267I CP00: GR00=00000000 GR01=00000000 GR02=00000000 GR03=00000000
HHC02267I CP00: GR04=00000000 GR05=00000000 GR06=00000000 GR07=00000000
HHC02267I CP00: GR08=00000000 GR09=00000000 GR10=000000DC GR11=00000000
HHC02267I CP00: GR12=00000000 GR13=00000000 GR14=00000000 GR15=40000082
HHC00834I Processor CP00: running state selected
HHC02267I CP00: PSW=00000148C0000090 INST=95000009     CLI   9(0),0    compare_logical_immediate
HHC02267I CP00: R:00000009:K:06=000000 00000000 00000000 00000000 00 ................
HHC02267I CP00: GR00=00000000 GR01=00000000 GR02=00000000 GR03=00000000
HHC02267I CP00: GR04=00000000 GR05=00000000 GR06=00000000 GR07=00000000
HHC02267I CP00: GR08=00000000 GR09=00000000 GR10=000000DC GR11=00000000
HHC02267I CP00: GR12=00000000 GR13=00000000 GR14=00000000 GR15=40000082
HHC00834I Processor CP00: running state selected
.
.
.
```

Figure 266: S+ command

8.143 S- (Instruction stepping off)

8.143.1 Function

This command turns off the instruction stepping. To continue normal processing after stopping the instruction stepping use the “G” command.

8.143.2 Syntax

Descriptive

```
S-
```

Diagram

```
►►—— S- ——————————————————————————————————————————►◄
```

8.143.3 Parameter

None.

8.143.4 Examples

Example 1:

Turn off instruction stepping.

```
HHC00013I Herc command: 's-'
HHC02229I Instruction stepping off range 40130-441c6
```

Figure 267: S- command

8.144 S? (Instruction stepping query)

8.144.1 Function

The S? command displays whether instruction stepping is on or off and the active range if any.

8.144.2 Syntax

Descriptive

```
S?
```

Diagram

```
▶▶── S? ──────────────────────────────────────────────────▶◀
```

8.144.3 Parameter

None.

8.144.4 Examples

Example 1:

Query instruction stepping.

```
HHC00013I Herc command: 's?'
HHC02229I Instruction stepping off range 100000-101000
```

Figure 268: S? command

8.145 S{+/-} dev (Turn CCW stepping on or off)

8.145.1 Function

This command turns on (s+ dev) or turns off (s- dev) the CCW stepping function. After CCW stepping is turned on every execution of a CCW has to be confirmed by hitting ENTER on the Hercules console and produces detailed trace output.

8.145.2 Syntax

Descriptive

S{+ | -} *devaddr*

Diagram

```
►►── S ──┬── + ──┬── devaddr ──────────────────────────────────►◄
         └── - ──┘
```

8.145.3 Parameter

devaddr This is the address of the device for which CCW stepping will be turned on or off.

+ The plus sign turns on the CCW stepping for the given device. The plus sign must immediately follow the S command (without an intervening blank).

- The minus sign turns off the CCW stepping for the given device. The minus sign must immediately follow the S command (without an intervening blank).

8.145.4 Examples

Example 1:

Turn on CCW stepping.

```
HHC00013I Herc command: 's+0148'
HHC02204I CCW step for 0:0148 set to on
HHC00396I 0:0148 start i/o file[1] bufcur -1 cache[-1]
HHC01315I 0:0148 CHAN: ccw 02000000 60000018=>02000000 60000018 00000000 00000000 ....-.........
HHC00431I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': seeking to cyl 0 head 0
HHC00396I 0:0148 read  trk   0 (asynchronous)
HHC00396I 0:0148 0 rdtrk     0
HHC00396I 0:0148 0 rdtrk[0] 0 cache miss
HHC00396I 0:0148 0 rdtrk[0] 0 buf 07AB5C88 len 19456
```

```
HHC00396I 0:0148 trk[0] read_trkimg
HHC00396I 0:0148 file[1] l2[0,0] trk[0] read_l2ent 0xd08
HHC00396I 0:0148 file[1] read_l2 0 active -1 -1 -1
HHC00396I 0:0148 l2[1,0] cache[0] miss
HHC00396I 0:0148 file[1] fd[18] read, off 0x0000000000000d08 len 2048
HHC00396I 0:0148 file[1] cache[0] l2[0] read offset 0x00000d08
HHC00396I 0:0148 file[0] l2[0,0] trk[0] read_l2ent 0x508
HHC00396I 0:0148 file[0] read_l2 0 active 1 0 0
HHC00396I 0:0148 l2[0,0] cache[1] miss
HHC00396I 0:0148 file[0] fd[17] read, off 0x0000000000000508 len 2048
HHC00396I 0:0148 file[0] cache[1] l2[0] read offset 0x00000508
HHC00396I 0:0148 file[0] l2[0,0] trk[0] read_l2ent 0x116f079 2343 2343
HHC00396I 0:0148 file[0] fd[17] read, off 0x000000000116f079 len 2343
HHC00396I 0:0148 0 rdtrk[0] 0 complete buf 07AB5C88:0100000000
HHC00396I 0:0148 uncompress comp 1 len 2343 maxlen 19456 trk 0
HHC00396I 0:0148 newbuf malloc 023C1CD0 len 19456
HHC00396I 0:0148 uncompress zlib newlen 6817 rc 0
HHC00396I 0:0148 validating trk 0 len 6817 0000000000 0000000000000008
HHC00396I 0:0148 read  trk   0 uncompressed len 6817
HHC00434I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': read count orientation 'index'
HHC00435I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': cyl 0 head 0 record 0 kl 0 dl 8 of 0
HHC00435I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': cyl 0 head 0 record 1 kl 4 dl 24 of 0
HHC00437I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': read data 24 bytes
HHC01312I 0:0148 CHAN: stat 0C00, count 0000 '=>00000000 00000000 06003A98 60000060 ...........q
HHC01315I 0:0148 CHAN: ccw 06003A98 60000060=>00000000 00000000 00000000 00000000 ..............
HHC00434I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': read count orientation 'data'
HHC00435I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': cyl 0 head 0 record 2 kl 4 dl 144 of 0
HHC00437I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': read data 144 bytes
HHC01312I 0:0148 CHAN: stat 0C00, count 0000 '=>07003AB8 40000006 31003ABE 40000005 .... .......
HHC01315I 0:0148 CHAN: ccw 08003A98 00000000=>07003AB8 40000006 31003ABE 40000005 .... ....... .
HHC01315I 0:0148 CHAN: ccw 07003AB8 40000006=>00000000 00000000 00000400 00000000 ..............
HHC00431I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': seeking to cyl 0 head 0
HHC01312I 0:0148 CHAN: stat 0C00, count 0000 ''
HHC01315I 0:0148 CHAN: ccw 31003ABE 40000005=>00000000 04000000 00000000 00000000 ..............
HHC00434I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': read count orientation 'index'
HHC00435I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': cyl 0 head 0 record 0 kl 0 dl 8 of 0
HHC01312I 0:0148 CHAN: stat 0C00, count 0000 ''
.
.
.
```

Figure 269: S+ dev command

Example 2:

Turn off CCW stepping.

```
HHC00013I Herc command: 's-0148'
HHC02204I CCW step for 0:0148 set to off
```

Figure 270: S- dev command

8.146 SAVECORE (Save a core image to a file)

8.146.1 Function

The SAVECORE command allows you to save a portion of real storage in binary form to a file. This command is used mainly for emulator debugging purposes. A certain core snapshot can be saved and then be restored at any later time to reproduce some tests with identical real storage values.

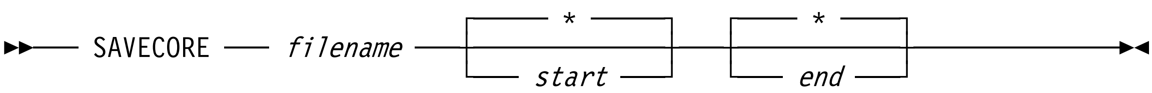
Please note that you have to stop all CPUs (see STOP/STOPALL commands) before saving the core image to a file.

8.146.2 Syntax

Descriptive

```
SAVECORE filename [start | * ] [end | * ]
```

Diagram



8.146.3 Parameter

filename This argument specifies the file name (and optionally the path) of the file where the core image will be written to.

***start* or *** *Start* specifies the start address of the real storage to be saved to the file *filename*. The asterisk (“*”), which is the default, means the first byte of the first modified page as determined by the storage-key changed bit.

***end* or *** *End* specifies the end address of the real storage to be saved to the file *filename*. The asterisk (“*”), which is the default, means the last byte of the last modified page as determined by the storage-key changed bit.

8.146.4 Examples

Example 1:

Save the first 4096 bytes of real storage to file D:\core01.bin.

```
HHC00013I Herc command: 'stopall'
HHC00013I Herc command: 'savecore d:\core01.bin 00000000 00001000'
HHC02248I Saving locations 00000000-00001000 to file 'd:\core01.bin'
HHC02249I Operation complete
```

Figure 271: SAVECORE command (save specific area)

Example 2:

Save all real storage to file D:\core02.bin.

```
HHC00013I Herc command: 'stopall'
HHC00013I Herc command: 'savecore d:\core02.bin * *'
HHC02248I Saving locations 00000000-00FFFFFF to file 'd:\core02.bin'
HHC02249I Operation complete
```

Figure 272: SAVECORE command (save from begin to end)

8.147 SCLPROOT (Set or display SCLP base directory)

8.147.1 Function

The SCLPROOT command sets or displays the SCLP base directory. If a directory is given then SCLP disk I/O for the specified directory path is enabled. NONE disables SCLP disk I/O.

A subsequent list-directed IPL resets the path to the location of the .ins file, and a subsequent CCW-type IPL disables SCLP disk I/O. If no operand is specified, SCLPROOT displays the current setting.

8.147.2 Syntax

Descriptive

SCLPROOT [NONE | *directory*]

Diagram

```
►►── SCLPROOT ──┬─────────────────┬──────────────────────────────►◄
                ├──── NONE ───────┤
                └── directory ────┘
```

8.147.3 Parameter

NONE Disables SCLP disk I/O.

directory Specifies the directory from which SCLP disk I/O is allowed. A subsequent IPL of an ".ins" file or a subsequent CCW-type IPL will override this.

8.147.4 Examples

Example 1:

Display SCLP base directory.

```
HHC00013I Herc command: 'sclproot'
HHC02204I SCLP disk I/O  set to disabled
```

Figure 273: SCLPROOT command (display SCLP base directory)

Example 2:

Set SCLP base directory.

```
HHC00013I Herc command: 'sclproot d:\iso'
HHC00013I Herc command: 'sclproot'
HHC02204I sclproot       set to d:/iso/
```

Figure 274: SCLPROOT command (set SCLP base directory)

8.148 SCP (Send system control program command)

8.148.1 Function

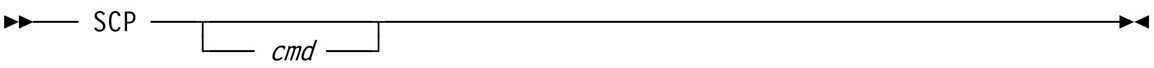
The SCP command sends a command to the system control program (the guest operating system running under Hercules) in any CMDTGT mode.

8.148.2 Syntax

Descriptive

SCP [*cmd*]

Diagram



8.148.3 Parameter

cmd This is the command to be sent to the system control program (guest operating system).

8.148.4 Examples

Example 1:

Send a reply (“r 00,0148”) to the system control program running under Hercules when in HERC or PSCP command target mode.

```
*   IGGN504A SPECIFY UNIT FOR CATALOG.MVS.MASTER ON MVSRES
scp r 00,0148
HHCCP041I SYSCONS interface active
 IEE600I REPLY TO 00 IS;0148
```

Figure 275: SCP command

8.149 SCPECHO (Display or set echo to console and history of SCP replies)

8.149.1 Function

SCPECHO switches the SPC ('.') and priority SCP ('!') replies and responses to the Hercules console on or off or displays the current setting if no argument is given. This is to manage passwords being displayed and journaled.

The default for SCPECHO is off if it is not explicitly switched on in the Hercules configuration file.

8.149.2 Syntax

Descriptive

```
SCPECHO [OFF | ON]
```

Diagram

```
►►── SCPECHO ──┬─────────────┬──────────────────────────────────────────────►◄
               ├─── OFF ─────┤
               └─── ON ──────┘
```

8.149.3 Parameter

OFF Do not route the SPC and priority SCP replies and responses to the Hercules console.

ON Route the SPC and priority SCP replies and responses to the Hercules console.

8.149.4 Examples

Example 1:

Switch echo to Hercules console for SCP and priority SCP commands on and off.

```
HHC00013I Herc command: 'scpecho on'
HHC02204I SCP, PSCP echo set to on

HHC00013I Herc command: 'scpecho off'
HHC02204I SCP, PSCP echo set to off
```

Figure 276: SCPECHO command

8.150 SCPIMPLY (Display or set option to pass non-Hercules commands to the SCP)

8.150.1 Function

SCPIMPLY allows it to switch on or off the passing of all non-Hercules commands (commands unknown to Hercules) to the SPC if the SCP has enabled receipt of SCP commands. Given without an argument the command displays the current value.

The default for SCPIMPLY is off if it is not explicitly switched on in the Hercules configuration file.

8.150.2 Syntax

Descriptive

SCPIMPLY [OFF | ON]

Diagram

```
►►── SCPIMPLY ──┬───────────┬──────────────────────────────────────►◄
                ├── OFF ────┤
                └── ON ─────┘
```

8.150.3 Parameter

OFF Do not route the SPC and priority SCP replies and responses to the Hercules console. This is the default, if SCPECHO is not coded in the configuration file.

ON Route the SPC and priority SCP replies and responses to the Hercules console.

8.150.4 Examples

Example 1:

Switch passing of non-Hercules commands to the SCP on and off.

```
HHC00013I Herc command: 'scpimply'
HHC02204I Value 'scpimply' set to 'off'

HHC00013I Herc command: 'scpimply on'
HHC02204I Value 'scpimply' set to 'on'
```

Figure 277: SCPIMPLY command

8.151 SCRIPT (Run a sequence of commands contained in a file)

8.151.1 Function

The SCRIPT command sequentially executes the commands contained within the file *filename*. The script file itself may also contain script commands but the system ensures that no more than 10 levels of scripts are invoked at any one time to avoid recursion loops. Enter the command without any arguments to list all currently running scripts.

8.151.2 Syntax

Descriptive

SCRIPT [*filename* [*filename* ...]]

Diagram

8.151.3 Parameter

filename The name (and optionally the path) of the script file to be executed.

8.151.4 Examples

Example 1:

Execute script file “Script1.rc”.

```
HHC00013I Herc command: 'script D:\MVS\Conf\Script1.rc'
HHC02260I Script 5: begin processing file 'D:/MVS/Conf/Script1.rc'
```

Figure 278: SCRIPT command (Execute script file)

Example 2:

List all currently running scripts.

```
HHC00013I Herc command: 'script'
HHC02315I Script id:2, tid:00000D2C, level:1, name:D:\MVS\Conf\Script1.rc
HHC02315I Script id:3, tid:00000A4C, level:1, name:D:\MVS\Conf\Script2.rc
HHC02315I Script id:4, tid:000017E8, level:1, name:D:\MVS\Conf\Script3.rc
HHC02315I Script id:5, tid:00000E6C, level:1, name:D:\MVS\Conf\Script4.rc
HHC02315I Script id:6, tid:00000B50, level:1, name:D:\MVS\Conf\Script5.rc
```

Figure 279: SCRIPT command (list currently running scripts)

8.152 SCSIMOUNT (Automatic SCSI tape mounts)

8.152.1 Function

The SCSIMOUNT command displays or modifies the automatic SCSI tape mount option. When entered without any operands it displays the current value and any pending tape mount requests. Entering a value between 1 and 99 enables the option and specifies how often (in seconds) to query SCSI tape drives to automatically detect when a tape has been mounted. When a tape is mounted an unsolicited device attention interrupt will be presented to the guest operating system.

Notes: Enabling this option may negatively impact Hercules performance depending on how the host operating system (Windows, Linux, etc.) processes SCSI attached tape drive status queries.

8.152.2 Syntax

Descriptive

SCSIMOUNT [NO | YES | *secs*]

Diagram

```
►►── SCSIMOUNT ──┬── NO ───┬──────────────────────────────────►◄
                 ├── YES ──┤
                 └── secs ─┘
```

8.152.3 Parameter

NO No indicates that the SCSIMOUNT option is disabled and forcing all SCSI tape mounts to be done manually via an appropriate DEVINIT command.

YES Yes enables the option and causes periodic queries of the SCSI tape drive in a five second interval to automatically detect when a tape is mounted. YES is equivalent with ‘SCSIMOUNT 5’.

secs A value from 1 to 99 seconds inclusive enables the option and causes periodic queries of the SCSI tape drive every *secs* seconds to automatically detect when a tape is mounted.

8.152.4 Examples

Example 1:

Enable the SCSI tape mount option and display the current settings.

```
HHC00013I Herc command: 'scsimount yes'
HHC02204I scsimount      set to yes
HHC00013I Herc command: 'scsimount'
HHC02203I scsimount     : 5
```

Figure 280: SCSIMOUNT command (enable SCSI tape mount option)

Example 2:

Enable the SCSIMOUNT option and query the SCSI tape drive every 15 seconds.

```
HHC00013I Herc command: 'scsimount 15'
HHC02204I scsimount set to 15
```

Figure 281: SCSIMOUNT command (enable SCSI tape mount option and set interval)

8.153 SF+ (Create a new shadow file)

8.153.1 Function

With the SF+ command a new shadow file for a certain dasd device (or for all dasd devices that have shadow files) is created.

8.153.2 Syntax

Descriptive

SF+ {*device* | *}

Diagram

```
►►── SF+ ──┬── device ──┬──────────────────────────────────────────────────►◄
           └──── * ─────┘
```

8.153.3 Parameter

device This specifies the device for which a new shadow file will be created.

***** Specifying an asterisk it is possible to create a (new) shadow file for all dasd devices that have a shadow file defined in their configuration file definition statements.

8.153.4 Examples

Example 1:

Create a new shadow file for device 0148.

```
HHC00013I Herc command: 'sf+ 0148'
HHC00320I 0:0148 CCKD file[1] 'D:/MVS/SHADOW/MVSRES_1.CCKD': shadow file succesfully added
HHC00333I 0:0148           size free  nbr st   reads  writes l2reads    hits switches
HHC00335I 0:0148 --------------------------------------------------------------------
HHC00336I 0:0148 [*]   21623269   0%    0          0       0       0       0        0
HHC00338I 0:0148 D:/MVS/DASD/MVSRES.CCKD
HHC00339I 0:0148 [0]   21621981   0%    0 rd       0       0       0
HHC00340I 0:0148 D:/MVS/SHADOW/MVSRES_*.CCKD
HHC00341I 0:0148 [1]       1288   0%    0 rw       0       0       0
```

Figure 282: SF+ command

8.154 SF- (Delete a shadow file)

8.154.1 Function

The SF- command removes a shadow file. Depending on the MERGE / NOMERGE / FORCE parameter the changes are incorporated in the base file or discarded.

If MERGE is specified or defaulted then the contents of the current file is merged into the previous file, the current file is removed and the previous file becomes the current file.

If NOMERGE is specified then the contents of the current file is discarded and the previous file becomes the current file. However if the previous file is read-only then a new shadow file is created ("re-added") and that becomes the current file.

The FORCE option is required when doing a merge to a base file and the base file is read-only because the 'ro' option was specified on the device configuration statement.

8.154.2 Syntax

Descriptive

SF- {*device* | *} [<u>MERGE</u> | NOMERGE | FORCE]

Diagram

```
                                   ┌──── MERGE ────┐
►►── SF- ──┬── device ──┬──────────┼───────────────┼──────────────────────────►◄
           └──── * ─────┘          ├─── NOMERGE ───┤
                                   └──── FORCE ────┘
```

8.154.3 Parameter

device	This specifies the device for which a shadow file should be deleted.
*****	Specifying an asterisk it is possible to remove a shadow file for all dasd devices that have shadow files.
MERGE	The MERGE parameter (which is the default) specifies that all changes or updates that have been made to the shadow file will be committed.
NOMERGE	The NOMERGE parameter specifies that all changes or updates that have been made to the shadow file will be discarded instead of being committed.
FORCE	The FORCE parameter is used when doing a merge to a base file that is read-only because the 'ro' option was specified on the device configuration statement.

8.154.4 Examples

Example 1:

Remove a shadow file for device 0148 with backwards merge (i.e. *commit* all of the changes/updates).

```
HHC00013I Herc command: 'sf- 0148'
HHC00325I 0:0148 CCKD file[1] 'D:/MVS/SHADOW/MVSRES_1.CCKD': shadow file successfully merged
HHC00333I 0:0148           size free  nbr st   reads  writes l2reads    hits switches
HHC00334I 0:0148                                                 readaheads   misses
HHC00335I 0:0148 --------------------------------------------------------------------
HHC00336I 0:0148 [*]   21627652   0%    9        881      21      14     712     1520
HHC00337I 0:0148                                                          74       23
HHC00338I 0:0148 D:/MVS/DASD/MVSRES.CCKD
HHC00339I 0:0148 [0]   21627652   0%    9 rw     881      21      14
HHC00340I 0:0148 D:/MVS/SHADOW/MVSRES_*.CCKD
```

Figure 283: SF- command

Example 2:

Remove a shadow file for device 0148 without backwards merge (i.e. *discard* all of the changes/updates).

```
HHC00013I Herc command: 'sf- 0148 nomerge'
HHC00325I 0:0148 CCKD file[1] 'D:/MVS/SHADOW/MVSRES_1.CCKD': shadow file successfully removed
HHC00333I 0:0148           size free  nbr st   reads  writes l2reads    hits switches
HHC00334I 0:0148                                                 readaheads   misses
HHC00335I 0:0148 --------------------------------------------------------------------
HHC00336I 0:0148 [*]   21627652   0%    9        883      20      14     715     1520
HHC00337I 0:0148                                                          79       25
HHC00338I 0:0148 D:/MVS/DASD/MVSRES.CCKD
HHC00339I 0:0148 [0]   21627652   0%    9 rw     883      20      14
HHC00340I 0:0148 D:/MVS/SHADOW/MVSRES_*.CCKD
```

Figure 284: SF- NOMERGE command

8.155 SFC (Compress a shadow file)

8.155.1 Function

The SFC command compresses a shadow file.

8.155.2 Syntax

Descriptive

SFC {*device* | *}

Diagram

```
►►── SFC ──┬── device ──┬──────────────────────────────────────────────►◄
           └──── * ─────┘
```

8.155.3 Parameter

device This specifies the device for which the shadow file will be compressed.

***** Specifying an asterisk it is possible to compress the shadow files for all DASD devices that have a shadow file.

8.155.4 Examples

Example 1:

Compress the shadow file for dasd device 0148.

```
HHC00013I Herc command: 'sfc 0148'
HHC00358I 0:0148 CCKD file 'D:/MVS/DASD/MVSRES.CCKD': file already compressed
HHC00333I 0:0148           size free  nbr st   reads  writes l2reads    hits switches
HHC00334I 0:0148                                                 readaheads   misses
HHC00335I 0:0148 ---------------------------------------------------------------------
HHC00336I 0:0148 [*]   21631751   0%    9        885     425      15     715     1522
HHC00337I 0:0148                                                          79       25
HHC00338I 0:0148 D:/MVS/DASD/MVSRES.CCKD
HHC00339I 0:0148 [0]   21627652   0%    9 rd     884      20      15
HHC00340I 0:0148 D:/MVS/SHADOW/MVSRES_*.CCKD
HHC00341I 0:0148 [1]       4099   0%    0 rw       1     405       0
```

Figure 285: SFC command

8.156 SFD (Display shadow file statistics)

8.156.1 Function

The SFD command displays statistical information about the specified shadow file(s).

8.156.2 Syntax

Descriptive

SFD {*device* | *}

Diagram

```
►►── SFD ──┬── device ──┬──────────────────────────────────────►◄
           └──── * ─────┘
```

8.156.3 Parameter

device This specifies the device for which shadow file statistics will be written to the log file.

***** Specifying an asterisk it is possible to create statistical output for all dasd devices that have shadow files.

8.156.4 Examples

Example 1:

Display statistical information for the shadow file of dasd device 0148.

```
HHC00013I Herc command: 'sfd 0148'
HHC00333I 0:0148           size free  nbr st   reads  writes l2reads    hits switches
HHC00334I 0:0148                                                   readaheads   misses
HHC00335I 0:0148 --------------------------------------------------------------------
HHC00336I 0:0148 [*]   21632514   0%   10        885     424      15     715     1522
HHC00337I 0:0148                                                          79       25
HHC00338I 0:0148 D:/MVS/DASD/MVSRES.CCKD
HHC00339I 0:0148 [0]   21627652   0%    9 rd     884      20      15
HHC00340I 0:0148 D:/MVS/SHADOW/MVSRES_*.CCKD
HHC00341I 0:0148 [1]       4862  15%    1 rw       1     404       0
```

Figure 286: SFD command

8.157 SFK (Perform a chkdsk on the active shadow file)

8.157.1 Function

The SFK command performs a chkdsk on the active shadow file(s).

8.157.2 Syntax

Descriptive

SFK {*device* | *} [*n*]

Diagram

```
►►── SFK ──┬── device ──┬──┬────────┬──────────────────────────────►◄
           └──── * ─────┘  └── n ───┘
```

8.157.3 Parameter

device This specifies the device for which active shadow file a chkdsk is to be performed.

***** Specifiying an asterisk it is possible to perform a chkdsk for the active shadow files of all dasd devices.

n The optional check level to be performed (default is 2):

-1 devhdr, cdevhdr, l1 table

0 devhdr, cdevhdr, l1 table, l2 tables

1 devhdr, cdevhdr, l1 table, l2 tables, free spaces

2 devhdr, cdevhdr, l1 table, l2 tables, free spaces, trkhdrs

3 devhdr, cdevhdr, l1 table, l2 tables, free spaces, trkimgs

4 devhdr, cdevhdr, build everything else from recovery

Use the check level '4' only after making a backup and be prepared to wait a long time!

8.157.4 Examples

Example 1:

Perform chkdsk level '2' for the shadow file of dasd device 0148.

```
HHC00013I Herc command: 'sfk 0148 2'
HHC00333I 0:0148           size free  nbr st   reads  writes l2reads    hits switches
HHC00334I 0:0148                                                  readaheads   misses
HHC00335I 0:0148 --------------------------------------------------------------------
HHC00336I 0:0148 [*]   21632514   0%   10        886     426      16     715     1523
HHC00337I 0:0148                                                          79       25
HHC00338I 0:0148 D:/MVS/DASD/MVSRES.CCKD
HHC00339I 0:0148 [0]   21627652   0%    9 rd     884      20      15
HHC00340I 0:0148 D:/MVS/SHADOW/MVSRES_*.CCKD
HHC00341I 0:0148 [1]       4862  15%    1 rw       2     406       1
```

Figure 287: SFK command (chkdsk level 2)

Example 2:

Perform chkdsk level '4' for the shadow file of dasd device 0148.

```
HHC00013I Herc command: 'sfk 0148 4'
HHC00372I 0:0148 CCKD file 'D:/MVS/DASD/MVSRES.CCKD': trk[10320] recovered offset 0x1003 len 763
HHC00373I 0:0148 CCKD file 'D:/MVS/DASD/MVSRES.CCKD': 1 trk images recovered
HHC00377I 0:0148 CCKD file 'D:/MVS/DASD/MVSRES.CCKD': free space rebuilt
HHC00333I 0:0148           size free  nbr st   reads  writes l2reads    hits switches
HHC00334I 0:0148                                                  readaheads   misses
HHC00335I 0:0148 --------------------------------------------------------------------
HHC00336I 0:0148 [*]   21632514   0%   10        886     426      16     715     1523
HHC00337I 0:0148                                                          79       25
HHC00338I 0:0148 D:/MVS/DASD/MVSRES.CCKD
HHC00339I 0:0148 [0]   21627652   0%    9 rd     884      20      15
HHC00340I 0:0148 D:/MVS/SHADOW/MVSRES_*.CCKD
HHC00341I 0:0148 [1]       4862  15%    1 rw       2     406       1
```

Figure 288: SFK command (chkdsk level 4)

8.158 SH (Shell command)

8.158.1 Function

The SH command passes the given command and the parameters (if any) as-is to the shell for process-sing. The results of the passed command are displayed on the Hercules console.

The special ‘STARTGUI’ command must be used if the command being started either directly or indirectly starts a Windows graphical user interface (non command-line) program (e.g. Notepad). If STARTGUI is not used in such cases then Hercules will hang until the graphical user interface program is closed. Note that starting a batch file (command line program) that itself starts a graphical user interface program still requires using STARTGUI.

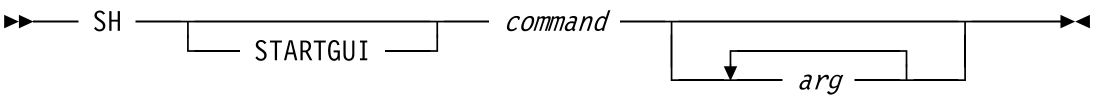
If “foo.bat” contains “start notepad.exe” then issuing “sh foo.bat” will hang Hercules until Notepad is being closed. The same applies if “sh start foo.bat” is used. The correct way is using ‘STARTGUI’ for invoking foo.bat is “sh startgui foo.bat”.

8.158.2 Syntax

Descriptive

SH [STARTGUI] *command* [*arg* [*arg* ...]]

Diagram



8.158.3 Parameter

STARTGUI STARTGUI must be used if the command passed to the shell either directly or in-directly starts a Windows graphical user interface (non command-line) program.

command The command that is to be passed to the shell.

arg These are the parameters passed to the command.

8.158.4 Examples

Example 1:

Display the current working directory through the shell command 'DIR'.

```
HHC00013I Herc command: 'sh dir'
 Volume in drive D is Hercules
 Volume Serial Number is E21F-FE83

 Directory of D:\hercules

13.10.2010  21:23    <DIR>          .
13.10.2010  21:23    <DIR>          ..
16.12.2007  01:00           282'624 AWSBrowse32.exe
16.12.2007  01:00           503'808 AWSBrowse32D.exe
16.12.2007  01:00           274'432 AWSBrowse32U.exe
16.12.2007  01:00           491'520 AWSBrowse32UD.exe
28.02.2007  06:24           323'072 AWSBrowse64.exe
28.02.2007  06:24           647'168 AWSBrowse64D.exe
28.02.2007  06:24           317'440 AWSBrowse64U.exe
28.02.2007  06:24           636'416 AWSBrowse64UD.exe
16.12.2007  01:00               934 awshet.reg
12.10.2010  18:24            14'336 cckdcdsk.exe
12.10.2010  18:24            13'824 cckdcomp.exe
12.10.2010  18:24            73'728 cckddiag.exe
12.10.2010  18:24            15'360 cckdswap.exe
.
several lines not displayed
.
16.12.2007  01:00           131'072 TunTap32.dll
16.12.2007  01:00           344'064 TunTap32D.dll
16.12.2007  01:00           135'168 TunTap32U.dll
16.12.2007  01:00           344'064 TunTap32UD.dll
28.02.2007  06:27           144'384 TunTap64.dll
28.02.2007  06:27           561'664 TunTap64D.dll
28.02.2007  06:27           147'456 TunTap64U.dll
28.02.2007  06:27           565'760 TunTap64UD.dll
13.06.2010  22:04    <DIR>          util
12.10.2010  18:24            19'968 vmfplc2.exe
13.10.2010  21:23             8'157 x.log
20.07.2005  11:48            59'904 zlib1.dll
             127 File(s)     24'988'911 bytes
               5 Dir(s)  67'712'364'544 bytes free
```

Figure 289: SH command

8.159 SHCMDOPT (Display or set shell command option)

8.159.1 Function

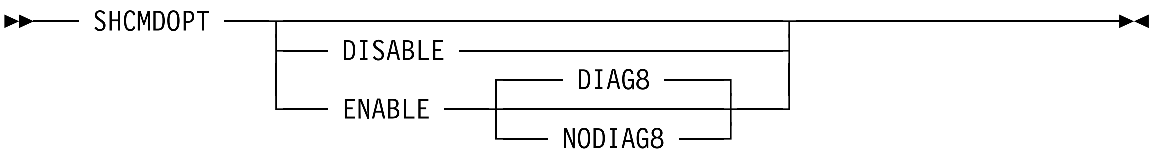
The SHCMDOPT console command specifies the behaviour of the shell (sh) command. It defines if shell commands (sh) are globally enabled or disabled either directly via the Hercules hardware console or programmatically via the DIAG8CMD interface. Given without an argument SHCMDOPT displays the current setting.

8.159.2 Syntax

Descriptive

`SHCMDOPT [DISABLE | ENABLE [DIAG8 | NODIAG8]]`

Diagram



8.159.3 Parameter

DISABLE When set to DISABLE, shell commands (sh) are globally disabled and will result in an error if entered either directly via the Hercules hardware console or programmatically via the DIAG8CMD interface.

ENABLE When set to ENABLE, shell commands (sh) are globally enabled either directly via the Hercules hardware console or programmatically via the DIAG8CMD interface. This is the default.

DIAG8 When DIAG8 is specified (which is the default) the programmatic execution of shell commands (sh) via the DIAG8CMD interface is enabled. This is the default.

NODIAG8 When NODIAG8 is specified only the programmatic execution of shell commands via the DIAG8CMD interface is disabled; shell commands (sh) entered directly via the Hercules hardware console will still work.

NOTE: *"entered directly via the Hercules hardware console"* includes commands entered via the HTTP server facility or entered via "run command" (.rc) scripts.

8.159.4 Examples

Example 1:

Display the current shell command option setting.

```
HHC00013I Herc command: 'shcmdopt'
HHC02203I shcmdopt      : Enabled
```

Figure 290: SHCMDOPT command (display current shell command options)

Example 2:

Disable the the programmatic execution of shell commands via the DIAG8CMD interface, while still allowing shell commands (sh) entered directly via the Hercules hardware console.

```
HHC00013I Herc command: 'shcmdopt enable nodiag8'
HHC02204I shcmdopt       set to Enabled NoDiag8
```

Figure 291: SHCMDOPT command (change shell command options)

8.160 SHOWDVOL1 (Display or set enable showing of DASD volsers in device list)

8.160.1 Function

SHOWDVOL1 indicates whether to show the DASD VOL1 labels (volser) in the device list display. 'YES' shows the volser in addition to the usual filename, whereas 'NO' shows the device list in a traditional filename only format. The 'ONLY' option shows only the volser; the filename is not shown at all. Enter the command without any operands to display the current setings on the console. Note: This console command is only available if Hercules is built with “OPTION_SHOWDVOL1”.

8.160.2 Syntax

Descriptive

```
SHOWDVOL1 [NO | YES | ONLY]
```

Diagram

```
►►── SHOWDVOL1 ──┬──────────────┬──────────────────────────────────►◄
                 ├─── NO ───────┤
                 ├─── YES ──────┤
                 └─── ONLY ─────┘
```

8.160.3 Parameter

NO NO shows the device list in the traditional filename only format. This is the default.

YES YES shows the volser in addition to the usual filename.

ONLY ONLY shows only the volser, the filename is not shown at all.

8.160.4 Examples

Example 1:

Show the DASD VOL1 labels (volser) in the device list.

```
HHC00013I Herc command: 'showdvol1 yes'
HHC02204I showdvol1      set to YES
```

Figure 292: SHOWDVOL command

8.161 SHRD (Display or set shared device server trace)

8.161.1 Function

The SHRD command has two functions. Depending on the given arguments it initializes the shared device server trace table and starts the trace or it displays the contents of the current trace table.

8.161.2 Syntax

Descriptive

SHRD TRACE[*=n*]

Diagram

```
►►── SHRD ── TRACE ──┬────────┬──────────────────────────►◄
                     └── =n ──┘
```

8.161.3 Parameter

TRACE Keyword TRACE, when specified without a decimal number, displays the current contents of the trace table.

n This specifies the trace table size (decimal). If the TRACE argument is given with a number then a trace table with a size of *n* elements initialized.

8.161.4 Examples

Example 1:

Initialize the shared device server trace table with a size of 2500 trace entries.

```
HHC00013I Herc command: 'shrd trace=2500'
```

Figure 293: SHRD command (initialize shared device server trace table)

Example 2:

Print the Shared Device Server trace table.

```
HHC00013I Herc command: 'shrd trace'
HHC00743I Shared: 1290125714.658483 0148:start cur 5966 cache 150
HHC00743I Shared: 1290125714.658486 0148:client_request e2 00 148 1
HHC00743I Shared: 1290125714.658490 0148:client_send e2 00 148 1 0
HHC00743I Shared: 1290125714.658543 0000:recvData    00 00 148 1 0
HHC00743I Shared: 1290125714.658548 0148:client_recv 00 00 148 1 0
HHC00743I Shared: 1290125714.658551 0148:client_response 00 00 148 1 0
HHC00743I Shared: 1290125714.658559 0148:ckd_read trk 5940
HHC00743I Shared: 1290125714.658566 0148:ckd_read trk 5940 cache hit 205
HHC00743I Shared: 1290125714.658576 0148:end cur 5940 cache 205
HHC00743I Shared: 1290125714.658579 0148:client_request e3 00 148 1
HHC00743I Shared: 1290125714.658582 0148:client_send e3 00 148 1 0
HHC00743I Shared: 1290125714.658645 0000:recvData    00 00 148 1 0
HHC00743I Shared: 1290125714.658649 0148:client_recv 00 00 148 1 0
HHC00743I Shared: 1290125714.658652 0148:client_response 00 00 148 1 0
HHC00743I Shared: 1290125714.659060 0148:start cur 5940 cache 205
HHC00743I Shared: 1290125714.659063 0148:client_request e2 00 148 1
HHC00743I Shared: 1290125714.659067 0148:client_send e2 00 148 1 0
HHC00743I Shared: 1290125714.659121 0000:recvData    00 00 148 1 0
HHC00743I Shared: 1290125714.659126 0148:client_recv 00 00 148 1 0
HHC00743I Shared: 1290125714.659128 0148:client_response 00 00 148 1 0
HHC00743I Shared: 1290125714.659134 0148:ckd_read trk 2550
HHC00743I Shared: 1290125714.659138 0148:ckd_read trk 2550 cache hit 107
HHC00743I Shared: 1290125714.659149 0148:end cur 2550 cache 107
HHC00743I Shared: 1290125714.659152 0148:client_request e3 00 148 1
HHC00743I Shared: 1290125714.659155 0148:client_send e3 00 148 1 0
HHC00743I Shared: 1290125714.659209 0000:recvData    00 00 148 1 0
HHC00743I Shared: 1290125714.659213 0148:client_recv 00 00 148 1 0
HHC00743I Shared: 1290125714.659216 0148:client_response 00 00 148 1 0
HHC00743I Shared: 1290125714.659654 0148:start cur 2550 cache 107
HHC00743I Shared: 1290125714.659657 0148:client_request e2 00 148 1
HHC00743I Shared: 1290125714.659660 0148:client_send e2 00 148 1 0
HHC00743I Shared: 1290125714.659720 0000:recvData    00 00 148 1 0
HHC00743I Shared: 1290125714.659724 0148:client_recv 00 00 148 1 0
HHC00743I Shared: 1290125714.659727 0148:client_response 00 00 148 1 0
HHC00743I Shared: 1290125714.659733 0148:ckd_read trk 2553
HHC00743I Shared: 1290125714.659738 0148:ckd_read trk 2553 cache miss -1
HHC00743I Shared: 1290125714.659744 0148:client_send e8 00 148 1 4
HHC00743I Shared: 1290125714.659845 0000:recvData    10 15 148 1 6541
HHC00743I Shared: 1290125714.660025 0000:recvData    00 00 148 1 11957 (uncompressed)
HHC00743I Shared: 1290125714.660030 0148:client_recv 00 00 148 1 11957
HHC00743I Shared: 1290125714.660045 0148:end cur 2553 cache 172
HHC00743I Shared: 1290125714.660049 0148:client_request e3 00 148 1
HHC00743I Shared: 1290125714.660052 0148:client_send e3 00 148 1 0
HHC00743I Shared: 1290125714.660111 0000:recvData    00 00 148 1 0
.
.
.
```

Figure 294: SHRD command (display shared device server trace table)

8.162 SHRDPORT (Set shared device server port)

8.162.1 Function

The SHRDPORT command defines the port number (in decimal) on which the shared device server will listen and starts or stops the shared device server. The shared device server will allow other Hercules instances to access devices on this instance. Currently only DASD devices may be shared.

8.162.2 Syntax

Descriptive

SHRDPORT [*port* | START | STOP]

Diagram

```
►►── SHRDPORT ──┬── port ──┬─────────────────────────────►◄
                ├── START ──┤
                └── STOP ───┘
```

8.162.3 Parameter

port This is the port number for the shared device server. Any valid port number can be specified. Specifying a port number starts the shared device server. It is not necessary to issue an explicit start command.

START Starts the shared device server if it is in the stopped state.

STOP Stops the shared device server if it is in the started state.

8.162.4 Examples

Example 1:

Start the shared device server.

```
HHC00013I Herc command: 'shrdport start'
HHC00100I Thread id 000003E8, prio  0, name 'Shared device server 0.1' started
HHC00737I Shared: waiting for shared device requests on port 3000
```

Figure 295: SHRDPORT command (start the shared device server)

Example 2:

Set the port number for the shared device server to 3999. To change the port number the shared device server has first to be stopped.

```
20:05:26 HHC00013I Herc command: 'shrdport stop'
20:05:37 HHC00013I Herc command: 'shrdport 3999'
20:05:37 HHC00100I Thread id 00000BD0, prio  0, name 'Shared device server 0.1' started
20:05:37 HHC00737I Shared: waiting for shared device requests on port 3999
```

Figure 296: SHRDPORT command (set port number)

8.163 SIZEOF (Display size of structures)

8.163.1 Function

The SIZEOF command shows the size of various Hercules internal structures.

8.163.2 Syntax

Descriptive

SIZEOF

Diagram

```
►►── SIZEOF ─────────────────────────────────────────────────►◄
```

8.163.3 Parameter

None.

8.163.4 Examples

Example 1:

Display the size of internal structures.

```
HHC00013I Herc command: 'sizeof'
HHC02257I (unsigned short) ..      2
HHC02257I (void *) .........      4
HHC02257I (unsigned int) ....      4
HHC02257I (long) ...........      4
HHC02257I (long long) .......      8
.
several lines not displayed
.
HHC02257I REGS (copy len) ...   1584
HHC02257I PSW ..............     40
HHC02257I DEVBLK ...........   3096
HHC02257I TLB entry .........     36
HHC02257I TLB table ......... 36864
HHC02257I CPU_BITMAP ........      4
HHC02257I STFL_BYTESIZE .....     10
```

Figure 297: SIZEOF command

8.164 SRVPRIO (Display or set server threads process priority)

8.164.1 Function

The SRVPRIO command is used to change the priority of the server threads. See section 5.82 for details on process and thread priorities. Given without an argument the SRVPRIO command displays the current server threads process priority.

Caution: SRVPRIO should not have a higher dispatching priority than the TOD clock and timer thread.

8.164.2 Syntax

Descriptive

SRVPRIO [*nn*]

Diagram

```
►►── SRVPRIO ──┬──────────┬──────────────────────────────────────►◄
               └── nn ────┘
```

8.164.3 Parameter

nn This value specifies the process priority for the server threads. For details on the priority values see section 5.82 (“Process and Thread Priorities”).

8.164.4 Examples

Example 1:

Set the server threads process priority to 0.

```
HHC00013I Herc command: 'srvprio 0'
HHC02204I srvprio        set to 0
```

Figure 298: SRVPRIO command

8.165 SSD (Signal shutdown)

8.165.1 Function

The SSD (signal shutdown) command signals an imminent hypervisor shutdown to the guest. Guests who support this are supposed to perform a shutdown upon receiving this request.

An implicit ssd command is given on a hercules QUIT or EXIT command if the guest supports SSD. In that case Hercules shutdown will be delayed until the guest has shutdown.

The SSD command acts different depending on how Hercules was built. If Hercules was not built with option 'OPTION_SHUTDOWN_CONFIRMATION' (#UNDEF in file feature.h) then the command acts as described above.

If Hercules however was built with option “OPTION_SHUTDOWN_CONFIRMATION” then a special processing for SSD takes place.

If SSD is entered and the command level is not set to 'developer', 'debug' or 'all' then SSD first issues message HHC02266A that prompts for confirmation by entering a second SSD command within a certain time period (default 10 seconds) to signal the hypervisor shutdown. If the time period has expired then the process starts over. This is to prevent an inadvertent shutdown of a running guest OS.

The time period for the second SSD command can be set to another value by using the 'QUITMOUT' console command or configuration statement.

If the command level is set to 'developer', 'debug' or 'all' then SSD immediately signals the hypervisor shutdown. 'SSD FORCE' also performs an immediate signal shutdown without any further checks.

8.165.2 Syntax

Descriptive

SSD [FORCE]

Diagram

```
►►── SSD ──┬───────────┬──────────────────────────►◄
           └── FORCE ──┘
```

8.165.3 Parameter

FORCE Perform the signal shutdown immediately.

8.165.4 Examples

Example 1:

Signal an imminent hypervisor shutdown to the guest operating system.

```
HHC00013I Herc command: 'ssd'
HHC02266A Confirm command by entering 'ssd' again within 10 seconds
HHC00013I Herc command: 'ssd'
```

Figure 299: SSD command

Example 2:

Signal an immediate imminent hypervisor shutdown to the guest operating system.

```
HHC00013I Herc command: 'ssd force'
```

Figure 300: SSD FORCE command

8.166 START (Start CPU or printer / punch device)

8.166.1 Function

The START command by itself (without argument) simply starts a stopped CPU, whereas START *devicenum* presses the (virtual) start button on printer or punch device *devicenum*.

8.166.2 Syntax

Descriptive

START [*devicenum*]

Diagram

```
►►── START ──┬──────────────┬──────────────────────────────►◄
             └─ devicenum ──┘
```

8.166.3 Parameter

devicenum The device number of the printer or punch device which should be started.

8.166.4 Examples

Example 1:

Start the printer device 000E.

```
HHC00013I Herc command: 'start 000e'
HHC02212I 0:000E device started
```

Figure 301: START command (start printer)

Example 2:

Start CPU 1.

```
HHC00013I Herc command: 'cpu 1'
HHC00013I Herc command: 'start'
HHC00834I Processor CP01: running state selected
```

Figure 302: START command (start CPU)

8.167 STARTALL (Start all CPUs)

8.167.1 Function

The STARTALL command starts all CPUs.

8.167.2 Syntax

Descriptive

STARTALL

Diagram

```
►►── STARTALL ──────────────────────────────────────────►◄
```

8.167.3 Parameter

None.

8.167.4 Examples

Example 1:

Start all CPU’s.

```
HHC00013I Herc command: 'startall'
```

Figure 303: STARTALL command

8.168 STOP (Stop CPU or printer / punch device)

8.168.1 Function

The STOP command by itself (without argument) simply stops a started CPU, whereas STOP *devicenum* presses the (virtual) stop button on printer or punch device *devicenum*

8.168.2 Syntax

Descriptive

STOP [*devicenum*]

Diagram

```
►►── STOP ──┬──────────────┬──────────────────────────►◄
            └── devicenum ─┘
```

8.168.3 Parameter

devicenum The device number of the printer or punch device which should be stopped-

8.168.4 Examples

Example 1:

Stop the printer device 000E.

```
HHC00013I Herc command: 'stop 000e'
HHC02214I 0:000E device stopped
```

Figure 304: STOP command (stop printer)

Example 2:

Stop CPU 1.

```
HHC00013I Herc command: 'cpu 1'
HHC00013I Herc command: 'stop'
HHC00834I Processor CP01: manual state selected
```

Figure 305: STOP command (stop CPU)

8.169 STOPALL (Stop all CPUs)

8.169.1 Function

The STOPALL command stops all CPUs

8.169.2 Syntax

Descriptive

```
STOPALL
```

Diagram

```
►►—— STOPALL ——————————————————————————————————————►◄
```

8.169.3 Parameter

None.

8.169.4 Examples

Example 1:

Stop all CPU's.

```
HHC00013I Herc command: 'stopall'
```

Figure 306: STOPALL command

8.170 STORE (Store CPU status at absolute zero)

8.170.1 Function

The STORE command stores the CPU status of the active CPU at absolute zero.

8.170.2 Syntax

Descriptive

```
STORE
```

Diagram

```
►►—— STORE ————————————————————————————————————————————►◄
```

8.170.3 Parameter

None.

8.170.4 Examples

Example 1:

Store status for active CPU.

```
HHC00013I Herc command: 'store'
HHC00817I Processor CP00: store status completed
```

Figure 307: STORE command

8.171 SUSPEND (Suspend Hercules)

8.171.1 Function

This command lets you suspend the current Hercules session and shutdown the host machine. Subsequently the suspended session can be resumed (see “RESUME” command). The data necessary to be saved for a later restart is saved in a packed (zipped) file called “hercules.srf.gz” which is located in the current configuration path.

After entering the SUSPEND command the CPUs are put in a stopped state and the contents of the main storage, CPU states, I/O device states and internal Hercules states are dumped onto the “Hercules.srf.gz” file. After the SUSPEND command has finished writing to the suspend file, it schedules an immediate shutdown of Hercules.

Currently device state is only fully saved for CKD disks. Each device class (e.g. TAPE, RDR, PUN, CTC) will need code to save and restore their state. Some states may not be possible to restore (e.g. active TCP/IP connections at the time of suspend). Currently the vector facility state is not saved, neither is the ECPSVM state.

The created suspend file is designed to be HERCULES release independent and to be host architecture independent. For example it is possible to take a suspend file created on HERCULES 3.05.0 on an Intel machine and resume on a Sun machine running HERCULES 3.07.0.

Please note that there are some caveats with suspend / resume processing which can affect the guest operating system running under Hercules. These caveats are described in the chapter about RESUME processing (8.138, RESUME).

There are some caveats when suspending and resuming guest operating system processing:

- As seen by the guest operating system, the TOD clock will appear to jump a large value. Some guests may not cope very well with this. For example some guests may be dismayed because certain interrupts will occur way past its due time. Also for S/370 an interval timer interrupt may be lost if the guest is interrupted for more than half the Interval Timer wrap time (around 8 hours).
- Although some effort has been put in order to make this as transparent as possible (that is, it should appear to the guest operating system that the STOP key was pressed for a large amount of time), some state information may be missed.
- Some guest operating systems will fare better if the suspend state is prepared first. For MVS, as an example, it seems to help when a QUIESCE command and a SYSTEM RESTART manual operation are issued prior to suspend the system.

8.171.2 Syntax

Descriptive

```
SUSPEND
```

Diagram

```
►►── SUSPEND ──────────────────────────────────────────────►◄
```

8.171.3 Parameter

None.

8.171.4 Examples

Example 1:

Suspend the Hercules session.

```
HHC00013I Herc command: 'suspend'
HHC01420I Begin Hercules shutdown
HHC02272I Highest observed MIPS and IO/s rates
HHC02272I   from Sun Sep 26 00:00:00 2010
HHC02272I     to Sun Sep 26 03:50:21 2010
HHC02272I   MIPS: 0.00  IO/s: 0
HHC01421I Releasing configuration
HHC00101I Thread id 000001F0, prio  0, name 'Processor CP00' ended
HHC00101I Thread id 000017A4, prio  0, name 'Processor CP01' ended
HHC01465I 0:000C device detached
HHC01465I 0:000D device detached
HHC01465I 0:000E device detached
.
several lines not displayed.
.
HHC01465I 0:0480 device detached
HHC01465I 0:0481 device detached
HHC01465I 0:0E20 device detached
HHC01422I Configuration release complete
HHC01423I Calling termination routines
HHC01500I HDL: begin shutdown sequence
HHC01501I HDL: calling 'panel_cleanup'
HHC01502I HDL: calling 'panel_cleanup' complete
HHC02103I Logger: logger thread terminating
```

Figure 308: SUSPEND command

8.172 SYMPTOM (Instruction trace display option)

8.172.1 Function

The SYMPTOM command determines how the registers are displayed during instruction tracing and stepping. SYMPTOM is an alias for the TRACEOPT console command. Please see TRACEOPT for details.

8.172.2 Syntax

See TRACEOPT console command.

8.172.3 Parameter

See TRACEOPT console command.

8.172.4 Examples

See TRACEOPT console command.

8.173 SYNCIO (Display SYNCIO device statistics)

8.173.1 Function

This command shows the SYNCIO device statistics for all DASD devices.

8.173.2 Syntax

Descriptive

SYNCIO

Diagram

```
►►── SYNCIO ──────────────────────────────────────────────────►◄
```

8.173.3 Parameter

None.

8.173.4 Examples

Example 1:

Display SYNCIO device statistics.

```
HHC00013I Herc command: 'syncio'
HHC02239I 0:0130 synchronous:           2 asynchronous:            1
HHC02239I 0:0131 synchronous:          48 asynchronous:           11
HHC02239I 0:0132 synchronous:           2 asynchronous:            1
HHC02239I 0:0133 synchronous:           2 asynchronous:            1
.
several lines not displayed
.
HHC02239I 0:034A synchronous:           7 asynchronous:            1
HHC02239I 0:034B synchronous:          13 asynchronous:            8
HHC02239I 0:0480 synchronous:        2326 asynchronous:          629
HHC02239I 0:0481 synchronous:          44 asynchronous:           19
HHC02240I Total synchronous:        57957 asynchronous:         9267   86%
```

Figure 309: SYNCIO command

8.174 SYSCLEAR (SYSTEM CLEAR RESET manual operation)

8.174.1 Function

The SYSCLEAR command issues a SYSTEM CLEAR RESET manual operation. This is the same as the "SYSRESET CLEAR" command. It clears the main storage to x'00'. Additionally the general purpose registers, control registers, etc. are reset to their initial value. At this point the system is essentially in the same state as it was just after having been started.

Please note that all CPUs must be stopped prior to issuing a system clear reset or the function will be rejected with a message.

8.174.2 Syntax

Descriptive

SYSCLEAR

Diagram

```
►►── SYSCLEAR ──────────────────────────────────────────────►◄
```

8.174.3 Parameter

None.

8.174.4 Examples

Example 1:

CLEAR RESET the system.

```
HHC00013I Herc command: 'stopall'
HHC00013I Herc command: 'sysclear'
HHC02311I sysclear completed
```

Figure 310: SYSCLEAR command

8.175 SYSEPOCH (Set base date for TOD clock)

8.175.1 Function

The SYSEPOCH command specifies the base date for the TOD clock. Use the default value (1900) for all systems except OS/360. Use 1960 for OS/360. Values other than these were formerly used to offset the TOD clock by a number of years to move the date before the year 2000 for non-Y2K-compliant operating systems. This use is deprecated and support will be removed in a future Hercules release after which only values of 1900 or 1960 will be accepted.

8.175.2 Syntax

Descriptive

SYSEPOCH {1900 | 1960 | *year* [*+years* | *-years*]}

Diagram

```
►►── SYSEPOCH ──┬── 1900 ─────────────────┬──────────────────────────────►◄
                ├── 1960 ─────────────────┤
                └── year ──┬──────────────┤
                           ├── +years ────┤
                           └── -years ────┘
```

8.175.3 Parameter

1900	Year 1900 is one of the two valid values for SYSEPOCH. 1900 is the default.
1960	Year 1960 is the second of the two valid values for SYSEPOCH.
year	This is the base date for the TOD clock. The only supported values for SYSEPOCH are currently 1900 and 1960. Any other value will produce a warning message showing the equivalent values to specify in the SYSEPOCH statement. 1900 is the default.
+ *years*	Specifies an optional positive year offset. It will be treated as though it had been specified using the YROFFSET statement.
- *years*	Specifies an optional negative year offset. It will be treated as though it had been specified using the YROFFSET statement.

8.175.4 Examples

Example 1:

Specify year 1900 as the base date for the TOD clock and use an offset of 28 years.

```
HHC00013I Herc command: 'sysepoch 1900 +28'
```

Figure 311: SYSEPOCH command (with offset)

Example 2:

Specify year 1960 as the base date for the TOD clock without an offset.

```
HHC00013I Herc command: 'sysepoch 1960'
```

Figure 312: SYSEPOCH command (without offset)

8.176 SYSRESET (SYSTEM RESET manual operation)

8.176.1 Function

The SYSRESET command issues a SYSTEM RESET manual operation. Without any arguments or with the “NORMAL” argument only a CPU and I/O subsystem reset are performed. When the “CLEAR” argument is given then this command is identical function to the “SYSCLEAR” command.

Please note that all CPUs must be stopped prior to issuing a system reset or the function will be rejected with a message.

8.176.2 Syntax

Descriptive

```
SYSRESET [NORMAL | CLEAR]
```

Diagram

```
►►── SYSRESET ──┬────────────┬──────────────────────────────►◄
                ├── NORMAL ──┤
                └── CLEAR ───┘
```

8.176.3 Parameter

NORMAL Performs only a CPU and I/O subsystem reset. This is the same as “SYSRESET” without any argument given.

CLEAR This clears the main storage to x'00'. Additionally the general purpose registers, control registers, etc. are reset to their initial value. At this point the system is essentially in the same state as it was just after having been started. “SYSRESET CLEAR” is identical in functionality to the “SYSCLEAR” command.

8.176.4 Examples

Example 1:

RESET CLEAR the system.

```
HHC00013I Herc command: 'stopall'
HHC00013I Herc command: 'sysreset clear'
HHC02311I sysreset clear completed
```

Figure 313: SYSRESET CLEAR command

8.177 T (Instruction trace)

8.177.1 Function

This command sets the instruction tracing range. Instruction tracing is totally separate from the instruction stepping and breaking range. With or without a range the “T” command displays whether tracing is on or off and the range, if any.

The “T” command by itself does not activate instruction tracing. Use the “T+” command to activate instruction tracing. The command “T 0” eliminates the range (all addresses will be traced).

8.177.2 Syntax

Descriptive

T [*addr-addr* | *addr:addr* | *addr.length* | 0]

Diagram

```
►►── T ──┬──────────────────┬──────────────────────────────────►◄
         ├─── addr-addr ────┤
         ├─── addr:addr ────┤
         ├── addr.length ───┤
         └─────── 0 ────────┘
```

8.177.3 Parameter

addr-addr	Specifies an address range with start and end address (from begin address to end address).
addr:addr	Specifies an address range with start and end address (from begin address to end address).
addr.length	Specifies an address range with start and length (from begin address with the specified length).
0	No range is specified or an existing range is reset. The tracing is active for all addresses.

8.177.4 Examples

Example 1:

Set instruction tracing range with begin and end address and query the range afterwards.

```
HHC00013I Herc command: 't 100000-101000'
HHC02229I Instruction tracing off range 100000-101000
HHC00013I Herc command: 't?'
HHC02229I Instruction tracing off range 100000-101000
```

Figure 314: T command (address range)

Example 2:

Set instruction tracing range with start address and length and query the range afterwards.

```
HHC00013I Herc command: 't 1000000.1000'
HHC02229I Instruction tracing off range 1000000.1000
HHC00013I Herc command: 't?'
HHC02229I Instruction tracing off range 1000000-1001000
```

Figure 315: T command (address with length)

Example 3:

Set instruction tracing for all addresses and query the range afterwards.

```
HHC00013I Herc command: 't 0'
HHC02229I Instruction tracing off
HHC00013I Herc command: 't?'
HHC02229I Instruction tracing off
```

Figure 316: T command (all addresses)

8.178 T+ (Instruction trace on)

8.178.1 Function

This command turns on instruction tracing. A range can be specified as for the “T” command. Instruction tracing is totally separate from the instruction stepping and breaking range.

After turning instruction tracing on, every instruction is listed with detailed trace information. The trace information includes the executing CPU, the PSW, the executed instruction in hexadecimal and in disassembled form as well as register hexadecimal displays of all involved register types.

This function is similar to the instruction stepping. The difference between these two function is that with instruction stepping the execution of every instruction has to be initiated by hitting ENTER on the Hercules console, whereas instruction tracing – once started – traces the instructions without interruption.

Please note that this function will create an enormous amount of output in the Hercules log. Running it on a relatively current machine (3 GHz HT Processor) for 10 seconds creates more than 10 MB of trace data.

8.178.2 Syntax

Descriptive

T+ [*addr-addr* | *addr:addr* | *addr.length* | 0]

Diagram

```
►►── T+ ──┬───────────────────┬──────────────────────────────────►◄
          ├──── addr-addr ────┤
          ├──── addr:addr ────┤
          ├─── addr.length ───┤
          └──────── 0 ────────┘
```

8.178.3 Parameter

addr-addr Specifies an address range with start and end address (from begin address to end address).

addr:addr Specifies an address range with start and end address (from begin address to end address).

addr.length Specifies an address range with start and length (from begin address with the specified length).

0 No range is specified or an existing range is reset. The tracing is active for all addresses.

8.178.4 Examples

Example 1:

Turn instruction tracing on.

```
HHC00013I Herc command: 't+'
HHC02229I Instruction tracing on
HHC02267I CP00: PSW=0000014800000080 INST=05F0         BALR  15,0       branch_and_link_register
HHC02267I CP00: GR00=00000000 GR01=00000000 GR02=00000000 GR03=00000000
HHC02267I CP00: GR04=00000000 GR05=00000000 GR06=00000000 GR07=00000000
HHC02267I CP00: GR08=00000000 GR09=00000000 GR10=00000000 GR11=00000000
HHC02267I CP00: GR12=00000000 GR13=00000000 GR14=00000000 GR15=00000000
HHC02267I CP00: PSW=0000014800000082 INST=980E0010     LM    0,14,16(0)            load_multiple
HHC02267I CP00: R:00000010:K:06=00000000 00000000 00000000 00000000  ................
HHC02267I CP00: GR00=00000000 GR01=00000000 GR02=00000000 GR03=00000000
HHC02267I CP00: GR04=00000000 GR05=00000000 GR06=00000000 GR07=00000000
HHC02267I CP00: GR08=00000000 GR09=00000000 GR10=00000000 GR11=00000000
HHC02267I CP00: GR12=00000000 GR13=00000000 GR14=00000000 GR15=40000082
HHC02267I CP00: PSW=0000014880000086 INST=41A0F05A     LA    10,90(0,15)            load_address
HHC02267I CP00: R:000000DC:K:06=41800100 41A0F08A 1B984700 F06E5890  ......0..q..0>..
HHC02267I CP00: GR00=00000000 GR01=00000000 GR02=00000000 GR03=00000000
HHC02267I CP00: GR04=00000000 GR05=00000000 GR06=00000000 GR07=00000000
HHC02267I CP00: GR08=00000000 GR09=00000000 GR10=00000000 GR11=00000000
HHC02267I CP00: GR12=00000000 GR13=00000000 GR14=00000000 GR15=40000082
HHC02267I CP00: PSW=000001488000008A INST=D200FC7D0008 MVC   3197(1,15),8(0)      move_character
HHC02267I CP00: R:00000CFF:K:06=F1 E5D5C9D7 D458FFFF 76B3BD00 AAAAFF 1VNIPM.....]....
HHC02267I CP00: R:00000008:K:06=F1000000 00000000 00000000 00000000  1...............
HHC02267I CP00: GR00=00000000 GR01=00000000 GR02=00000000 GR03=00000000
HHC02267I CP00: GR04=00000000 GR05=00000000 GR06=00000000 GR07=00000000
HHC02267I CP00: GR08=00000000 GR09=00000000 GR10=000000DC GR11=00000000
HHC02267I CP00: GR12=00000000 GR13=00000000 GR14=00000000 GR15=40000082
HHC02267I CP00: PSW=00000148C0000090 INST=95000009     CLI   9(0),0    compare_logical_immediate
HHC02267I CP00: R:00000009:K:06=000000 00000000 00000000 00000000 00 ................
HHC02267I CP00: GR00=00000000 GR01=00000000 GR02=00000000 GR03=00000000
HHC02267I CP00: GR04=00000000 GR05=00000000 GR06=00000000 GR07=00000000
HHC02267I CP00: GR08=00000000 GR09=00000000 GR10=000000DC GR11=00000000
HHC02267I CP00: GR12=00000000 GR13=00000000 GR14=00000000 GR15=40000082
HHC02267I CP00: PSW=0000014880000094 INST=4780F050     BC    8,80(0,15)       branch_on_condition
HHC02267I CP00: R:000000D2:K:06=6800 FD2E2820 28402860 41800100 41A0 ....... .-......
HHC02267I CP00: GR00=00000000 GR01=00000000 GR02=00000000 GR03=00000000
HHC02267I CP00: GR04=00000000 GR05=00000000 GR06=00000000 GR07=00000000
HHC02267I CP00: GR08=00000000 GR09=00000000 GR10=000000DC GR11=00000000
HHC02267I CP00: GR12=00000000 GR13=00000000 GR14=00000000 GR15=40000082
.
.
.
```

Figure 317: T+ command

8.179 T- (Instruction trace off)

8.179.1 Function

This command turns off the instruction tracing function.

8.179.2 Syntax

Descriptive

```
T-
```

Diagram

```
►►── T- ──────────────────────────────────────────────────────────►◄
```

8.179.3 Parameter

None.

8.179.4 Examples

Example 1:

Turn off instruction tracing.

```
HHC00013I Herc command: 't-'
HHC02229I Instruction tracing off range 1000000-1001000
```

Figure 318: T- command

8.180 T? (Instruction trace query)

8.180.1 Function

The “T?” command displays whether instruction tracing is on or off and the range if any.

8.180.2 Syntax

Descriptive

```
T?
```

Diagram

```
►►── T? ──────────────────────────────────────────────────────►◄
```

8.180.3 Parameter

None.

8.180.4 Examples

Example 1:

Query instruction tracing.

```
HHC00013I Herc command: 't?'
HHC02229I Instruction tracing off range 1000000-1001000
```

Figure 319: T? command

8.181 T{+/-} CKD (Turn CKD_KEY tracing on or off)

8.181.1 Function

The “T{+/-}CKD” command turns the CKD key tracing on (T+CKD) or turns it off (T-CKD). Please note that the command has to be issued without any intervening blanks.

8.181.2 Syntax

Descriptive

```
T{+ | -}CKD
```

Diagram

```
►►── T ──┬── + ──┬── CKD ──────────────────────────────►◄
         └── - ──┘
```

8.181.3 Parameter

T+CKD The T+CKD turns on the CKD key trace. The plus sign and the CKD argument must immediately follow the T command (without an intervening blank).

T-CKD The T-CKD turns off the CKD key trace. The minus sign and the CKD argument must immediately follow the T command (without an intervening blank).

8.181.4 Examples

Example 1:

Turning the CKD key trace on.

```
HHC00013I Herc command: 't+ckd'
HHC02204I CKD key trace  set to on
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'SYS1.NUCLEUS                    '
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEANUC01'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEAVNIPM'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEAVNPA2'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEAVNP01'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEAVNPB2'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'SYS1.LOGREC                     '
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'SYS1.SVCLIB                     '
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEAVNP11'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'SYSCATLG'
```

```
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'Z9999994.VSAMDSPC'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEAVNP1A'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEAVNP03'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEAVNP12'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'SYS1.PARMLIB                  '
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEASYS00'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'SYS1.LINKLIB                  '
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'LNKLST00'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'SYS1.CMDLIB                   '
HHC00423I 0:0240 CKD file 'D:/MVS/DASD/PUB000.CCKD': search key 'SYS2.LINKLIB                  '
HHC00423I 0:0241 CKD file 'D:/MVS/DASD/PUB001.CCKD': search key 'SYS2.CMDLIB                   '
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEAVNP10'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEAOPT00'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEAIPS00'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'ILRPREAD'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEAVNP04'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'ILROPS00'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEAVNPA8'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEAVNP05'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'SYS1.LPALIB                   '
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEAPAK00'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEABLD00'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'ALLOC   '
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'E       '
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'HEWL    '
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEWL    '
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IEWLF440'
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IFOX01  '
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IFOX11  '
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'IFOX62  '
HHC00423I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': search key 'LINK    '
.
.
.
```

Figure 320: T+CKD command

Example 2:

Turning the CKD key trace off.

```
HHC00013I Herc command: 't-ckd'
HHC02204I CKD key trace  set to off
```

Figure 321: T-CKD command

8.182 T{+/-} dev (Turn CCW tracing on or off)

8.182.1 Function

This command turns the CCW tracing for a certain device on (T+dev) or turns it off (T-dev). It is possible to trace several devices concurrently. To enable the tracing of more than one device the T+ command must be issued for each device. The traces can be turned off independently of each other.

This function is similar to the CCW stepping described in section 8.145. The difference between these two function is that with CCW stepping the execution of every CCW has to be initiated by hitting ENTER on the Hercules console, whereas CCW tracing – once started – traces the instructions without interruption.

Notes:

- The tracing function will create an enormous amount of output in the Hercules log.
- The command has to be issued without any intervening blanks.

8.182.2 Syntax

Descriptive

T{+ | -}*devaddr*

Diagram

```
►►── T ──┬── + ──┬── devaddr ──────────────────────────────────►◄
         └── - ──┘
```

8.182.3 Parameter

+	The plus sign turns on the CCW tracing for the given device. The plus sign must immediately follow the T command (without an intervening blank).
-	The minus sign turns off the CCW tracing for the given device. The minus sign must immediately follow the T command (without an intervening blank).
devaddr	This specifies the address of the device for which the CCW tracing is to be turned on or off.

8.182.4 Examples

Example 1:

Turn on the CCW tracing for device 0148.

```
HHC00013I Herc command: 't+0148'
HHC02204I CCW trace for 0:0148 set to on
HHC00396I 0:0148 start i/o file[1] bufcur -1 cache[-1]
HHC01315I 0:0148 CHAN: ccw 02000000 60000018=>02000000 60000018 00000000 00000000 ....-.........
HHC00431I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': seeking to cyl 0 head 0
HHC00396I 0:0148 read  trk   0 (asynchronous)
HHC00396I 0:0148 0 rdtrk     0
HHC00396I 0:0148 0 rdtrk[0] 0 cache miss
HHC00396I 0:0148 0 rdtrk[0] 0 buf 07B8F9E0 len 19456
HHC00396I 0:0148 trk[0] read_trkimg
HHC00396I 0:0148 file[1] l2[0,0] trk[0] read_l2ent 0xd08
HHC00396I 0:0148 file[1] read_l2 0 active -1 -1 -1
HHC00396I 0:0148 l2[1,0] cache[0] miss
HHC00396I 0:0148 file[1] fd[18] read, off 0x0000000000000d08 len 2048
HHC00396I 0:0148 file[1] cache[0] l2[0] read offset 0x00000d08
HHC00396I 0:0148 file[0] l2[0,0] trk[0] read_l2ent 0x508
HHC00396I 0:0148 file[0] read_l2 0 active 1 0 0
HHC00396I 0:0148 l2[0,0] cache[1] miss
HHC00396I 0:0148 file[0] fd[17] read, off 0x0000000000000508 len 2048
HHC00396I 0:0148 file[0] cache[1] l2[0] read offset 0x00000508
HHC00396I 0:0148 file[0] l2[0,0] trk[0] read_l2ent 0x116f079 2343 2343
HHC00396I 0:0148 file[0] fd[17] read, off 0x000000000116f079 len 2343
HHC00396I 0:0148 0 rdtrk[0] 0 complete buf 07B8F9E0:0100000000
HHC00396I 0:0148 uncompress comp 1 len 2343 maxlen 19456 trk 0
HHC00396I 0:0148 newbuf malloc 07B9CED8 len 19456
HHC00396I 0:0148 uncompress zlib newlen 6817 rc 0
HHC00396I 0:0148 validating trk 0 len 6817 0000000000 0000000000000008
HHC00396I 0:0148 read  trk   0 uncompressed len 6817
HHC00434I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': read count orientation 'index'
HHC00435I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': cyl 0 head 0 record 0 kl 0 dl 8 of 0
HHC00435I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': cyl 0 head 0 record 1 kl 4 dl 24 of 0
HHC00437I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': read data 24 bytes
HHC01312I 0:0148 CHAN: stat 0C00, count 0000 '=>00000000 00000000 06003A98 60000060 ...........q
HHC01315I 0:0148 CHAN: ccw 06003A98 60000060=>00000000 00000000 00000000 00000000 ..............
HHC00434I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': read count orientation 'data'
HHC00435I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': cyl 0 head 0 record 2 kl 4 dl 144 of 0
HHC00437I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': read data 144 bytes
HHC01312I 0:0148 CHAN: stat 0C00, count 0000 '=>07003AB8 40000006 31003ABE 40000005 .... .......
HHC01315I 0:0148 CHAN: ccw 08003A98 00000000=>07003AB8 40000006 31003ABE 40000005 .... ....... .
HHC01315I 0:0148 CHAN: ccw 07003AB8 40000006=>00000000 00000000 00000400 00000000 ..............
HHC00431I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': seeking to cyl 0 head 0
HHC01312I 0:0148 CHAN: stat 0C00, count 0000 ''
HHC01315I 0:0148 CHAN: ccw 31003ABE 40000005=>00000000 04000000 00000000 00000000 ..............
HHC00434I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': read count orientation 'index'
HHC00435I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': cyl 0 head 0 record 0 kl 0 dl 8 of 0
HHC01312I 0:0148 CHAN: stat 0C00, count 0000 ''
HHC01315I 0:0148 CHAN: ccw 08003AA0 00000000=>31003ABE 40000005 08003AA0 00000000 .... .........
HHC01315I 0:0148 CHAN: ccw 31003ABE 40000005=>00000000 04000000 00000000 00000000 ..............
HHC00434I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': read count orientation 'count'
```

```
HHC00435I 0:0148 CKD file 'D:/MVS/DASD/MVSRES.CCKD': cyl 0 head 0 record 1 kl 4 dl 24 of 0
HHC01312I 0:0148 CHAN: stat 0C00, count 0000 ''
HHC01315I 0:0148 CHAN: ccw 08003AA0 00000000=>31003ABE 40000005 08003AA0 00000000 .... .........
HHC01315I 0:0148 CHAN: ccw 31003ABE 40000005=>00000000 04000000 00000000 00000000 .............
.
.
.
```

Figure 322: T+ dev command

Example 2:

Turn off the CCW tracing for device 0148.

```
HHC00013I Herc command: 't-0148'
HHC02204I CCW trace for 0:0148 set to off
```

Figure 323: T- dev command

Example 3:

Turn on the CCW tracing for several devices.

```
HHC00013I Herc command: 't+0344'
HHC02204I CCW trace for 0:0344 set to on
HHC00013I Herc command: 't+0345'
HHC02204I CCW trace for 0:0345 set to on
```

Figure 324: T+ dev command (for several devices)

8.183 TIMERINT (Display or set timers update interval)

8.183.1 Function

The TIMERINT parameter displays or sets the internal timer update interval in microseconds. This parameter specifies how frequently Hercules's internal timers-update thread updates the TOD clock, CPU Timer and other architectural related clock and timer values.

The default interval is 50 microseconds which attempts to strike a reasonable balance between clock accuracy and overall host performance.

8.183.2 Syntax

Descriptive

TIMERINT [*interval*]

Diagram

```
►►── TIMERINT ──┬──────────────┬──────────────────────────►◄
                └── interval ──┘
```

8.183.3 Parameter

interval Specifies the timer update interval in microseconds. The minimum allowed value for the interval is 1 microsecond and the maximum is 1'000'000 microseconds (one second.

CAUTION: While a lower TIMERINT value may help increase the accuracy of the guest's TOD clock and CPU Timer values, it could also have severe negative impact on the overall performance of the host operating system. This is especially true when a low value is coupled with a high HERCPRIO and TODPRIO priority setting.

Exercise extreme caution when choosing your desired TIMERINT in relationship to your chosen HERCPRIO and TODPRIO priority settings.

8.183.4 Examples

Example 1:

Display current timers update interval.

```
HHC00013I Herc command: 'timerint'
HHC02203I timerint      : 50
```

Figure 325: TIMERINT command (list current value)

Example 2:

Set timers update interval to 100 microseconds.

```
HHC00013I Herc command: 'timerint 100'
HHC02204I timerint       set to 100
```

Figure 326: TIMERINT command (set new value)

8.184 TLB (Display TLB tables)

8.184.1 Function

The TLB command shows the actual contents of the TLB (Translation Lookaside Buffer) tables.

8.184.2 Syntax

Descriptive

TLB

Diagram

```
►►── TLB ──────────────────────────────────────────────────────────►◄
```

8.184.3 Parameter

None.

8.184.4 Examples

Example 1:

Display the TLB tables.

```
HHC00013I Herc command: 'tlb'
HHC02284I tlbID 0x000E3B mainstor 0000000003A81000
HHC02284I   ix              asd            vaddr              pte   id c p r w ky     main
HHC02284I *000 000000000FFD7C00 0000000000000000 0000000000000001 0E3B 0 0 1 0 00 00000000
HHC02284I *001 000000000FFD7C00 0000000000000800 0000000000000001 0E3B 0 0 1 0 00 00000800
HHC02284I *002 000000000FFD7C00 0000000000001000 0000000000000011 0E3B 0 0 0 0 00 FC580000
HHC02284I *003 000000000FFD7C00 0000000000001800 0000000000000011 0E3B 0 0 1 1 00 00001800
HHC02284I  004 000000000F927C00 0000000000002000 0000000000000021 0E31 0 0 1 0 00 00002000
HHC02284I  005 000000000F927C00 0000000000002800 0000000000000021 0E31 0 0 1 0 00 00002800
HHC02284I  006 000000000F927C00 0000000000003000 0000000000000031 0E31 0 0 1 0 00 00003000
HHC02284I  007 000000000F927C00 0000000000003800 0000000000000031 0E31 0 0 1 0 00 00003800
HHC02284I *008 000000000FFD7C00 0000000000004000 0000000000000041 0E3B 0 0 1 0 00 00004000
HHC02284I *009 000000000FFD7C00 0000000000004800 0000000000000041 0E3B 0 0 0 0 00 FC583800
.
several lines not displayed
.
HHC02284I *3F6 000000000FFD7C00 0000000000FFB000 000000000000FFB1 0E3B 0 0 1 1 00 00FFB000
HHC02284I *3F7 000000000FFD7C00 0000000000FFB800 000000000000FFB1 0E3B 0 0 1 1 00 00FFB800
```

```
HHC02284I *3F8 000000000FFD7C00 0000000000FFC000 000000000000FFC1 0E3B 0 0 0 0 00 FD57B000
HHC02284I *3F9 000000000FFD7C00 0000000000FFC800 000000000000FFC1 0E3B 0 0 1 1 00 00FFC800
HHC02284I *3FA 000000000FFD7C00 0000000000FFD000 000000000000FFD1 0E3B 0 0 0 0 00 FD57C000
HHC02284I *3FB 000000000FFD7C00 0000000000FFD800 000000000000FFD1 0E3B 0 0 1 1 00 00FFD800
HHC02284I *3FC 000000000FFD7C00 0000000000FFE000 000000000000FFE1 0E3B 0 0 1 1 00 00FFE000
HHC02284I *3FD 000000000FFD7C00 0000000000FFE800 000000000000FFE1 0E3B 0 0 1 0 00 00FFE800
HHC02284I *3FE 000000000FFD7C00 0000000000FFF000 000000000000FFF1 0E3B 0 0 0 0 00 FD57E000
HHC02284I *3FF 000000000FFD7C00 0000000000FFF800 000000000000FFF1 0E3B 0 0 1 0 00 00FFF800
HHC02284I 138 tlbID matches
```

Figure 327: TLB command

8.185 TODDRAG (Display or set TOD clock drag factor)

8.185.1 Function

The TODDRAG command displays the current setting of the TOD clock drag factor or sets a new value. This parameter can be used to slow down or speed up the TOD clock by a factor of *nn.* A significant slow-down can improve the performance of some operating systems which consume large amounts of CPU time processing timer interrupts. A drag factor of 2.0 slows down the clock by 50%, a drag factor of 0.5 doubles the speed of the clock, a drag factor of 1.01 slows down the clock by 1% and 0.99 speeds up the clock by 1%.

8.185.2 Syntax

Descriptive

```
TODDRAG [factor]
```

Diagram

```
►►── TODDRAG ──┬──────────┬──────────────────────────────►◄
               └─ factor ─┘
```

8.185.3 Parameter

factor The factor to which the TOD clock has to be slowed down or sped up.

8.185.4 Examples

Example 1:

Display current TOD clock drag factor.

```
HHC00013I Herc command: 'toddrag'
HHC02203I toddrag       : 1.000000
```

Figure 328: TODDRAG command (display TOD clock factor)

Example 2:

Speed up the clock by 10%.

```
HHC00013I Herc command: 'toddrag 0.9'
HHC02204I toddrag        set to 0.9
```

Figure 329: TODDRAG command (set TOD clock factor)

8.186 TODPRIO (Display or set timer thread process priority)

8.186.1 Function

The TODPRIO command is used to change the process priority of the TOD clock and the timer threads. See section 5.82 for details on process and thread priorities. Given without an argument the TODPRIO command displays the current Hercules process priority.

Caution: TODPRIO should be given a dispatching priority equal to or higher than any other thread (CPUPRIO, DEVPRIO, HERCPRIO and SRVPRIO) within Hercules.

8.186.2 Syntax

Descriptive

TODPRIO [*nn*]

Diagram

```
►►── TODPRIO ──┬──────────┬──────────────────────────────────────►◄
               └── nn ────┘
```

8.186.3 Parameter

nn This value specifies the priority of the TOD clock and the timer thread. For details on the priority values see section 5.82 (“Process and Thread Priorities”).

8.186.4 Examples

Example 1:

Set the process priority of the TOD clock and the timer threads to 0.

```
HHC00013I Herc command: 'todprio -20'
HHC02204I todprio        set to -20
```

Figure 330: TODPRIO command

8.187 TRACEOPT (Instruction trace display options)

8.187.1 Function

The TRACEOPT command determines how the registers are displayed during instruction tracing and stepping. Entering the command without any argument simply displays the current trace mode.

8.187.2 Syntax

Descriptive

```
TRACEOPT [TRADITIONAL | REGSFIRST | NOREGS]
```

Diagram

```
►►── TRACEOPT ──┬───────────────────┬──────────────────────────────►◄
                ├── TRADITIONAL ────┤
                ├──── REGSFIRST ────┤
                └────── NOREGS ─────┘
```

8.187.3 Parameter

TRADITIONAL Displays the registers following the instruction about to be executed such that pressing enter (to execute the displayed instruction) then shows the next instruction to be executed followed by the updated registers display.

REGSFIRST Displays the current register contents followed by the instruction about to be executed such that pressing enter (to execute the displayed instruction) then shows the updated registers followed by the next instruction to be executed.

NOREGS Suppresses the register display altogether and shows just the instruction to be executed.

8.187.4 Examples

Example 1:

Display the instruction trace in traditional mode (instruction followed by registers).

```
HHC00013I Herc command: 'traceopt traditional'
HHC02203I Hercules inst trace displayed: traditional mode
HHC02267I CP00: PSW=040C10000001D0C4 INST=58E00230     L     14,560(0,0)            load
HHC02267I CP00: V:00000230:K:06=00000000 00000000 00000000 00FD4000  .............. .
HHC02267I CP00: GR00=00000000 GR01=00000000 GR02=00000000 GR03=00000000
```

```
HHC02267I CP00: GR04=00000000 GR05=00000000 GR06=00000000 GR07=00FFDE28
HHC02267I CP00: GR08=00000000 GR09=4001D0A4 GR10=00000000 GR11=00000000
HHC02267I CP00: GR12=00000000 GR13=00000000 GR14=00000000 GR15=00000000
HHC02267I CP00: CR00=C080EC40 CR01=0FFD7C00 CR02=FFFFFFFF CR03=00000000
HHC02267I CP00: CR04=00000000 CR05=00000000 CR06=00000000 CR07=00000000
HHC02267I CP00: CR08=00000000 CR09=00000000 CR10=00000000 CR11=00000000
HHC02267I CP00: CR12=00000000 CR13=00000000 CR14=EFC00000 CR15=00FEBB10
HHC02267I CP00: PSW=040C10000001D0C8 INST=50E070BC     ST    14,188(0,7)            store
HHC02267I CP00: V:00FFDEE4:K:06=00FFDE28 00000000 4001D0A4 00000000  ........ .}u....
HHC02267I CP00: GR00=00000000 GR01=00000000 GR02=00000000 GR03=00000000
HHC02267I CP00: GR04=00000000 GR05=00000000 GR06=00000000 GR07=00FFDE28
HHC02267I CP00: GR08=00000000 GR09=4001D0A4 GR10=00000000 GR11=00000000
HHC02267I CP00: GR12=00000000 GR13=00000000 GR14=00000000 GR15=00000000
HHC02267I CP00: CR00=C080EC40 CR01=0FFD7C00 CR02=FFFFFFFF CR03=00000000
HHC02267I CP00: CR04=00000000 CR05=00000000 CR06=00000000 CR07=00000000
HHC02267I CP00: CR08=00000000 CR09=00000000 CR10=00000000 CR11=00000000
HHC02267I CP00: CR12=00000000 CR13=00000000 CR14=EFC00000 CR15=00FEBB10
```

Figure 331: TRACEOPT command (TRADITIONAL)

Example 2:

Display the instruction trace in regsfirst mode (registers followed by instruction).

```
HHC00013I Herc command: 'traceopt regsfirst'
HHC02203I Hercules inst trace displayed: regsfirst mode
HHC02267I CP00: PSW=040C00000001D09A
HHC02267I CP00: GR00=00000000 GR01=00000000 GR02=00000000 GR03=00000000
HHC02267I CP00: GR04=00000000 GR05=00000000 GR06=00000000 GR07=00000000
HHC02267I CP00: GR08=00000000 GR09=00000000 GR10=00000000 GR11=00000000
HHC02267I CP00: GR12=00000000 GR13=00000000 GR14=00000000 GR15=00000000
HHC02267I CP00: CR00=C080EC40 CR01=0FFD7C00 CR02=FFFFFFFF CR03=00000000
HHC02267I CP00: CR04=00000000 CR05=00000000 CR06=00000000 CR07=00000000
HHC02267I CP00: CR08=00000000 CR09=00000000 CR10=00000000 CR11=00000000
HHC02267I CP00: CR12=00000000 CR13=00000000 CR14=EFC00000 CR15=00FEBB10
INST=50700230     ST    7,560(0,0)            store
HHC02267I CP00: V:00000230:K:06=00000000 00000000 00000000 00FD4000  .............. .
HHC02267I CP00: PSW=040C00000001D09E
HHC02267I CP00: GR00=00000000 GR01=00000000 GR02=00000000 GR03=00000000
HHC02267I CP00: GR04=00000000 GR05=00000000 GR06=00000000 GR07=00000000
HHC02267I CP00: GR08=00000000 GR09=00000000 GR10=00000000 GR11=00000000
HHC02267I CP00: GR12=00000000 GR13=00000000 GR14=00000000 GR15=00000000
HHC02267I CP00: CR00=C080EC40 CR01=0FFD7C00 CR02=FFFFFFFF CR03=00000000
HHC02267I CP00: CR04=00000000 CR05=00000000 CR06=00000000 CR07=00000000
HHC02267I CP00: CR08=00000000 CR09=00000000 CR10=00000000 CR11=00000000
HHC02267I CP00: CR12=00000000 CR13=00000000 CR14=EFC00000 CR15=00FEBB10
INST=50900234     ST    9,564(0,0)            store
HHC02267I CP00: V:00000234:K:06=00000000 00000000 00FD4000 070E0000  .......... .....
HHC02267I CP00: PSW=040C00000001D0A2
```

Figure 332: TRACEOPT command (REGSFIRST)

Example 3:

Display the instruction trace in noregs mode (instruction only).

```
HHC00013I Herc command: 'traceopt noregs'
HHC02203I Hercules inst trace displayed: noregs mode
HHC02267I CP00: V:00FFDEC8:K:06=00000000 00000000 00000000 00000000  ................
HHC02267I CP00: PSW=040C10000001D0C4 INST=58E00230     L     14,560(0,0)            load
HHC02267I CP00: V:00000230:K:06=00000000 00000000 00000000 00FD4000  .............. .
HHC02267I CP00: PSW=040C10000001D0C8 INST=50E070BC     ST    14,188(0,7)            store
HHC02267I CP00: V:00FFDEE4:K:06=00FFDE28 00000000 4001D0A4 00000000  ........ .}u....
HHC02267I CP00: PSW=040C10000001D0CC INST=58E00234     L     14,564(0,0)            load
HHC02267I CP00: V:00000234:K:06=00000000 00000000 00FD4000 070E0000  .......... .....
HHC02267I CP00: PSW=040C10000001D0D0 INST=50E070C4     ST    14,196(0,7)            store
HHC02267I CP00: V:00FFDEEC:K:06=4001D0A4 00000000 00000000 00000000   .}u............
HHC02267I CP00: PSW=040C10000001D0D4 INST=58D00010     L     13,16(0,0)             load
```

Figure 333: TRACEOPT command (NOREGS)

8.188 TT32 (Control / query CTCI-WIN functionality)

8.188.1 Function

The TT32 command controls or queries the CTCI-WIN functionality for a given device address. It allows you to enable or disable global CTCI-WIN debug tracing, or displays TunTap32 statistics for the specified CTC device.

8.188.2 Syntax

Descriptive

TT32 {DEBUG | NODEBUG | STATS *devnum*}

Diagram

```
►►── TT32 ──┬──────── DEBUG ─────────┬──────────────────────────────────────►◄
            ├─────── NODEBUG ────────┤
            └─── STATS ─── devnum ───┘
```

8.188.3 Parameter

DEBUG Enable global CTCI-WIN debug tracing.

NODEBUG Disable global CTCI-WIN debug tracing.

STATS Display TunTap32 statistics for CTC device.

devnum Device number of the device for which the TunTap32 statistics have to displayed.

8.188.4 Examples

Example 1:

Enable global CTCI-WIN debug tracing.

```
HHC00013I Herc command: 'tt32 debug'
HHC02204I TT32 debug set to enabled
```

Figure 334: TT32 DEBUG command

Example 2:

Disable global CTCI-WIN debug tracing.

```
HHC00013I Herc command: 'tt32 nodebug'
HHC02204I TT32 debug set to disabled
```

Figure 335: TT32 NODEBUG command

Example 3:

Display TunTap32 statistics for device 0E20.

```
HHC00013I Herc command: 'tt32 stats 0e20'
HHC04101I D:\Hercules\TunTap32.dll Statistics:

          Size of Kernel Hold Buffer:       1024K
          Size of DLL I/O Buffer:             64K
          Maximum DLL I/O Bytes Received:      4K

                     0  Total Write Calls
                     0  Total Write I/Os
                     0  Packets To All Zeroes MAC Written
                     0  Total Packets Written
                     0  Total Bytes Written

                   346  Total Read Calls
                  1770  Total Read I/Os
                     0  Internally Handled ARP Packets
                     0  Packets From Ourself
                   913  Total Ignored Packets
                     0  Packets To All Zeroes MAC Read
                   950  Total Packets Read
                524372  Total Bytes Read
```

Figure 336: TT32 STATS command

8.189 TZOFFSET (Set TOD clock offset from GMT)

8.189.1 Function

The TZOFFSET command is used to define the offset of the TOD clock from the current system time. For GMT use the default value (0000). This is also the correct setting if your system time (the time of the operating system on which Hercules is running) is set to local time rather than GMT. For time zones west of Greenwich specify a negative value (example: -0500 for US Eastern Standard Time, -0800 for US Pacific Standard Time). For time zones east of Greenwich, specify a positive value (example: +0100 for Central European Time, +0930 for South Australian Time).

8.189.2 Syntax

Descriptive

TZOFFSET {+*hhmm* | -*hhmm*}

Diagram

```
►►── TZOFFSET ──┬── +hhmm ──┬──────────────────────────────────────►◄
                └── -hhmm ──┘
```

8.189.3 Parameter

+hhmm Use a positive time in hours and minutes for time zones east of Greenwich.

-hhmm Use a negative time in hours and minutes for time zones west of Greenwich.

8.189.4 Examples

Example 1:

Set the offset of the TOD clock from the current system time to Central European Time.

```
HHC00013I Herc command: 'tzoffset +0100'
HHC02204I tzoffset       set to +0100
```

Figure 337: TZOFFSET command

8.190 U (Disassemble storage)

8.190.1 Function

The U command lets you disassemble storage. The argument specifies the storage address to be disassembled. The length of the storage area can be given as an optional argument. The listed storage area is formatted as one line per instruction. Every instruction is shown as a hexadecimal string and as readable assembler statement. Storage areas that cannot be disassembled (data areas) are marked with question marks. Up to 64K bytes can be disassembled.

8.190.2 Syntax

Descriptive

U [R | V | P | H] {*addr.length* | *addr-addr*}

Diagram

```
►►── U ──┬───────┬──┬─ addr.length ─┬──►◄
         ├── R ──┤  └─ addr-addr ───┘
         ├── V ──┤
         ├── P ──┤
         └── H ──┘
```

8.190.3 Parameter

R	The optional 'R' argument will force real address translation mode instead of current PSW mode.
V	The optional 'V' argument will force virtual address translation mode instead of current PSW mode.
P	The optional 'P' argument will force primary space address translation mode instead of current PSW mode.
H	The optional 'H' argument will force home space address translation mode instead of current PSW mode.
addr.length	Specifies the address of the storage area that is to be disassembled with starting address and length (from address *addr* with the length of *length*). The *length* value must be given in hexadecimal. The maximum length that can be specified is 64K.
addr-addr	Specifies an address range with start and end address (from begin address to end address) of the storage area that is to be disassembled. The range specified must not exceed 64K.

8.190.4 Examples

Example 1:

Disassemble 4096 bytes beginning at address 00040130.

```
HHC02289I u 00040130.1000
HHC02289I P00040130: 90DF02EC     STM   13,15,748(0)          store_multiple
HHC02289I P00040134: 4166000C     LA    6,12(6,0)             load_address
HHC02289I P00040138: 58606160     L     6,352(0,6)            load
HHC02289I P0004013C: 58300010     L     3,16(0,0)             load
HHC02289I P00040140: 58430184     L     4,388(3,0)            load
HHC02289I P00040144: 55C06058     CL    12,88(0,6)            compare_logical
HHC02289I P00040148: 478060E0     BC    8,224(0,6)            branch_on_condition
HHC02289I P0004014C: 8840000A     SRL   4,10(0)               shift_right_single_logical
HHC02289I P00040150: 47F060E4     BC    15,228(0,6)           branch_on_condition
HHC02289I P00040154: 88400005     SRL   4,5(0)                shift_right_single_logical
HHC02289I P00040158: 9620820C     OI    524(8),32             or_immediate
HHC02289I P0004015C: 55C06058     CL    12,88(0,6)            compare_logical
HHC02289I P00040160: 478060F8     BC    8,248(0,6)            branch_on_condition
HHC02289I P00040164: 58506228     L     5,552(0,6)            load
HHC02289I P00040168: 47F060FC     BC    15,252(0,6)           branch_on_condition
HHC02289I P0004016C: 5850622C     L     5,556(0,6)            load
HHC02289I P00040170: 58AB0000     L     10,0(11,0)            load
HHC02289I P00040174: 12AA         LTR   10,10                 load_and_test_register
HHC02289I P00040176: 4770611A     BC    7,282(0,6)            branch_on_condition
HHC02289I P0004017A: 94DF820C     NI    524(8),223            and_immediate
HHC02289I P0004017E: 9102022A     TM    554(0),2              test_under_mask
HHC02289I P00040182: 0717         BCR   1,7                   branch_on_condition_register
HHC02289I P00040184: 980402B8     LM    0,4,696(0)            load_multiple
HHC02289I P00040188: 98DF02EC     LM    13,15,748(0)          load_multiple
HHC02289I P0004018C: 07F7         BCR   15,7                  branch_on_condition_register
HHC02289I P0004018E: 465060FC     BCT   5,252(0,6)            branch_on_count
HHC02289I P00040192: 55A06230     CL    10,560(0,6)           compare_logical
HHC02289I P00040196: 4740620E     BC    4,526(0,6)            branch_on_condition
HHC02289I P0004019A: 48E301DC     LH    14,476(3,0)           load_halfword
HHC02289I P0004019E: 54A06234     N     10,564(0,6)           and
HHC02289I P000401A2: 15AE         CLR   10,14                 compare_logical_register
HHC02289I P000401A4: 4720620E     BC    2,526(0,6)            branch_on_condition
HHC02289I P000401A8: B60082B0     STCTL 0,0,688(8)            store_control
HHC02289I P000401AC: B6000308     STCTL 0,0,776(0)            store_control
HHC02289I P000401B0: D40303086166 NC    776(4,0),358(6)       and_character
HHC02289I P000401B6: D6030308616A OC    776(4,0),362(6)       or_character
HHC02289I P000401BC: B7000308     LCTL  0,0,776(0)            load_control
HHC02289I P000401C0: AD010308     STOSM 776(0),1              store_then_or_system_mask
HHC02289I P000401C4: ACFE0308     STNSM 776(0),254            store_then_and_system_mask
HHC02289I P000401C8: B6000308     STCTL 0,0,776(0)            store_control
HHC02289I P000401CC: D60082B00308 OC    688(1,8),776(0)       or_character
.
remaining lines not displayed
.
```

Figure 338: U command

8.191 UPTIME (Display Hercules Emulator uptime)

8.191.1 Function

The UPTIME console command displays how long the Hercules Emulator has been running.

8.191.2 Syntax

Descriptive

```
UPTIME
```

Diagram

```
►►—— UPTIME ——————————————————————————————————————————————►◄
```

8.191.3 Parameter

None.

8.191.4 Examples

Example 1:

Display how long the Hercules Emulator has been running so far (uptime < 1 day).

```
HHC00013I Herc command: 'uptime'
HHC02208I Uptime 06:33:12
```

Figure 339: UPTIME command (uptime < 1 day).

Example 2:

Display how long the Hercules Emulator has been running so far (uptime > 1 week).

```
HHC00013I Herc command: 'uptime'
HHC02208I Uptime 3 weeks, 5 days, 16:48:15.
```

Figure 340: UPTIME command (uptime > 1 week).

8.192 V (Display or alter virtual storage)

8.192.1 Function

The V command displays or alters virtual storage. Up to 64K of virtual storage can be displayed, up to 32 bytes of virtual storage can be altered.

8.192.2 Syntax

Descriptive

V [P | S | H] {*addr* | *addr.length* | *addr-addr* | *addr=value*}

Diagram

```
►►── V ──┬─────┬──┬── addr ────────┬──────────────────►◄
         ├─ P ─┤  ├─ addr.length ──┤
         ├─ S ─┤  ├─ addr-addr ────┤
         └─ H ─┘  └─ addr=value ───┘
```

8.192.3 Parameter

P The optional 'P' argument will force primary space address translation mode instead of current PSW mode.

S The optional 'P' argument will force secondary space address translation mode instead of current PSW mode.

H The optional 'H' argument will force home space address translation mode instead of current PSW mode.

addr Specifies the address of the virtual storage that is to be displayed. If *addr* is given without a length or without a second address for the end of the storage area then 64 bytes of virtual storage are displayed.

addr.length Specifies the address of the virtual storage area that is to be displayed with starting address and length (from address *addr* with the length of *length*). The *length* value must be given in hexadecimal. The maximum length that can be specified is 64K.

addr-addr Specifies an address range with start and end address (from begin address to end address) of the virtual storage area that is to be displayed.

addr=value Specifies the address of the virtual storage area that is to be altered. *value* is a hex-string of up to 32 pairs of digits (32 bytes) which will be written to the virtual storage address given by the *addr* parameter. After altering the storage, 16 bytes of real storage starting at *addr* are displayed.

8.192.4 Examples

Example 1:

Display 256 (x'FF') bytes of virtual storage starting from address x'00010000'.

```
HHC00013I Herc command: 'v 00010000.ff'
HHC02291I V:00010000 (primary) R:00010000
HHC02291I V:00010000:K:04=4770A1C8 5820901C 47F0A1F0 91802003  ..~H.....0~0j...
HHC02291I V:00010010:K:04=47E0A1EC 58202004 47F0A1F0 96803200  .\~......0~0o...
HHC02291I V:00010020:K:04=91803200 4770A1FE 12224770 A1DC5850  j.....~.....~..&
HHC02291I V:00010030:K:04=30201FCC 195C4770 A01C50C0 30245820  .....*....&{....
HHC02291I V:00010040:K:04=30089140 320047E0 A28441B0 202C58D0  ..j ...\sd.....}
HHC02291I V:00010050:K:04=02FC98CD D19005ED 58C031F4 12CC4780  ..q.J....{.4....
HHC02291I V:00010060:K:04=A24E9120 300047E0 A2424160 203047F0  s+j....\s..-...0
HHC02291I V:00010070:K:04=A2464160 20485870 31F445E0 A2E458C0  s..-.....4.\sU.{
HHC02291I V:00010080:K:04=31F812CC 4780A262 187C4160 203045E0  .8....s..@.-...\
HHC02291I V:00010090:K:04=A2E458C0 31FC12CC 4780A276 187C4160  sU.{......s..@.-
HHC02291I V:000100A0:K:04=204045E0 A2E441B0 202C58D0 02FC98CD  . .\sU.....}..q.
HHC02291I V:000100B0:K:04=D19C05ED 91803001 4770A2DA 1FCC59C0  J...j.....s....{
HHC02291I V:000100C0:K:04=30504780 A2A691C0 32004770 A2A659C0  .&..swj{....sw.{
HHC02291I V:000100D0:K:04=200C4780 A2DA58C0 305412CC 4770A2DA  ....s..{......s.
HHC02291I V:000100E0:K:04=1F114180 0001BA18 30544770 A2DA5810  ............s...
HHC02291I V:000100F0:K:04=305858F0 001058F0 F2645800 F0005000  ...0...02...0.&.
```

Figure 341: V command (display virtual storage with length)

Example 2:

Change virtual storage at address x'00010000' to x'FFFFFFFF'.

```
HHC00013I Herc command: 'v 00010000=FFFFFFFF'
HHC02291I V:00010000 (primary) R:00010000
HHC02291I V:00010000:K:06=FFFFFFFF 5820901C 47F0A1F0 91802003  .........0~0j...
```

Figure 342: V command (alter virtual storage)

Example 3:

Display virtual storage from address x'00010000' to adress x'00010100'.

```
HHC00013I Herc command: 'v 00010000-00010100'
HHC02291I V:00010000 (primary) R:00010000
HHC02291I V:00010000:K:06=FFFFFFFF 5820901C 47F0A1F0 91802003  .........0~0j...
HHC02291I V:00010010:K:06=47E0A1EC 58202004 47F0A1F0 96803200  .\~......0~0o...
HHC02291I V:00010020:K:06=91803200 4770A1FE 12224770 A1DC5850  j.....~.....~..&
HHC02291I V:00010030:K:06=30201FCC 195C4770 A01C50C0 30245820  .....*....&{....
HHC02291I V:00010040:K:06=30089140 320047E0 A28441B0 202C58D0  ..j ...\sd.....}
HHC02291I V:00010050:K:06=02FC98CD D19005ED 58C031F4 12CC4780  ..q.J....{.4....
HHC02291I V:00010060:K:06=A24E9120 300047E0 A2424160 203047F0  s+j....\s..-...0
HHC02291I V:00010070:K:06=A2464160 20485870 31F445E0 A2E458C0  s..-.....4.\sU.{
```

```
HHC02291I V:00010080:K:06=31F812CC 4780A262 187C4160 203045E0  .8....s..@.-...\
HHC02291I V:00010090:K:06=A2E458C0 31FC12CC 4780A276 187C4160  sU.{......s..@.-
HHC02291I V:000100A0:K:06=204045E0 A2E441B0 202C58D0 02FC98CD  . .\sU.....}..q.
HHC02291I V:000100B0:K:06=D19C05ED 91803001 4770A2DA 1FCC59C0  J...j.....s....{
HHC02291I V:000100C0:K:06=30504780 A2A691C0 32004770 A2A659C0  .&..swj{....sw.{
HHC02291I V:000100D0:K:06=200C4780 A2DA58C0 305412CC 4770A2DA  ....s..{......s.
HHC02291I V:000100E0:K:06=1F114180 0001BA18 30544770 A2DA5810  ............s...
HHC02291I V:000100F0:K:06=305858F0 001058F0 F2645800 F0005000  ...0...02...0.&.
HHC02291I V:00010100:K:06=1004BA01 F0004770 A2CE982E 31C0947F  ....0...s.q..{m"
```

Figure 343: V command (display virtual storage with range)

Example 4:

Display 256 (x’FF’) bytes of virtual storage starting from address x’00010000’. Force Secondary translation instead of current PSW mode.

```
HHC00013I Herc command: ‘v s 00010000.ff’
HHC02291I V:00010000 (secondary) R:00010000
HHC02291I V:00010000:K:04=4770A1C8 5820901C 47F0A1F0 91802003  ..~H.....0~0j...
HHC02291I V:00010010:K:04=47E0A1EC 58202004 47F0A1F0 96803200  .\~......0~0o...
HHC02291I V:00010020:K:04=91803200 4770A1FE 12224770 A1DC5850  j.....~.....~..&
HHC02291I V:00010030:K:04=30201FCC 195C4770 A01C50C0 30245820  .....*....&{....
HHC02291I V:00010040:K:04=30089140 320047E0 A28441B0 202C58D0  ..j ...\sd.....}
HHC02291I V:00010050:K:04=02FC98CD D19005ED 58C031F4 12CC4780  ..q.J....{.4....
HHC02291I V:00010060:K:04=A24E9120 300047E0 A2424160 203047F0  s+j....\s..-...0
HHC02291I V:00010070:K:04=A2464160 20485870 31F445E0 A2E458C0  s..-.....4.\sU.{
HHC02291I V:00010080:K:04=31F812CC 4780A262 187C4160 203045E0  .8....s..@.-...\
HHC02291I V:00010090:K:04=A2E458C0 31FC12CC 4780A276 187C4160  sU.{......s..@.-
HHC02291I V:000100A0:K:04=204045E0 A2E441B0 202C58D0 02FC98CD  . .\sU.....}..q.
HHC02291I V:000100B0:K:04=D19C05ED 91803001 4770A2DA 1FCC59C0  J...j.....s....{
HHC02291I V:000100C0:K:04=30504780 A2A691C0 32004770 A2A659C0  .&..swj{....sw.{
HHC02291I V:000100D0:K:04=200C4780 A2DA58C0 305412CC 4770A2DA  ....s..{......s.
HHC02291I V:000100E0:K:04=1F114180 0001BA18 30544770 A2DA5810  ............s...
HHC02291I V:000100F0:K:04=305858F0 001058F0 F2645800 F0005000  ...0...02...0.&.
```

Figure 344: V command (display virtual storage with ‘S’ option)

8.193 VERSION (Display version information)

8.193.1 Function

The version command is used to display various version and build information about the Hercules emulator.

8.193.2 Syntax

Descriptive

```
VERSION
```

Diagram

```
►►── VERSION ──────────────────────────────────────────────►◄
```

8.193.3 Parameter

None.

8.193.4 Examples

Example 1:

Display the Hercules version information.

```
HHC00013I Herc command: 'version'
HHC01413I Hercules version 3.07-svn-6583
HHC01414I (c) Copyright 1999-2010 by Roger Bowler, Jan Jaeger, and others
HHC01415I Built on Sep 24 2010 at 23:46:56
HHC01416I Build information:
HHC01417I Windows (MSVC) build for i386
HHC01417I Modes: S/370 ESA/390 z/Arch
.
several lines not displayed
.
HHC01417I Regular Expressions support
HHC01417I Automatic Operator support
HHC01417I Machine dependent assists: cmpxchg1 cmpxchg4 cmpxchg8 fetch_dw store_dw
HHC01417I Running on GOOFY Windows-6.1.7600. 7 Ultimate Edition, 32-bit i686 MP=8
```

Figure 345: VERSION command

8.194 XPNDSIZE (Display or set expanded storage size)

8.194.1 Function

The XPNDSIZE command is used to specify the size of the main storage in megabytes. Given without an argument the XPNDSIZE command displays the current size of the expanded storage. Storage is allocated in megabytes, unless a specific unit is specified. The actual upper limit of the expanded storage is determined by the host system's architecture, operating system, and on some systems the amount of physical memory and paging space you have available. The lower limit is 0.

The practical limit depends on the maximum amount of storage that can be obtained by “malloc” (usually around 1 GB on 32-bit platforms; on 64-bit platforms the value should only be limited by available paging space).

When increasing the expanded size Hercules attempts to allocate first the new storage. If the new allocation is successful then the previously allocated memory will be freed. This is to prevent a situation where the old memory is freed first, then the new allocation fails and a reallocation of the memory in the previous size also fails because of storage fragmentation and therefore leaving Hercules without memory.

When decreasing the expanded storage the memory will stay allocated in the previous size but the storage size will appear as decreased. Subsequent increases will not reallocate memory unless they go over the already allocated amount.

An additional optional argument determines the locking state of the allocated memory (page lock by host operating system). The LOCKED option indicates that the memory is to be locked into storage while UNLOCKED (the default) indicates that the memory is not locked into the storage.

Please note that Hercules preserves the last locking state of XPNDSIZE. Once storage is locked, any subsequent change to the expanded storage size will honor the existing lock state of memory unless the lock state is specified again on the XPNDSIZE command.

Caution: Do not lock expanded storage unless sufficient real memory is available to back up the request. Failure to do so may require the host system to be rebooted.

8.194.2 Syntax

Descriptive

XPNDSIZE [*xsize*[M | G | T] [UNLOCK | LOCK]]

Diagram

```
►►── XPNDSIZE ──┬──────────── ──────────────────────────────────┬──────────►◄
                └── xsize ──┬── M ──┬──┬── UNLOCK ──┬───────────┘
                            ├── G ──┤  └── LOCK ────┘
                            └── T ──┘
```

8.194.3 Parameter

size The value of *xsize* must be a valid decimal number. The actual upper limit is determined by the host system's architecture, the operating system and on some systems the amount of physical memory and paging space that is available.

Storage sizes not on a 1M boundary are rounded up to the next 1M boundary. The lower limit and default is 0.

M 'M' determines that the number given is specified in megabytes (multiplier 2**20). This is the default if no unit is appended.

G 'G' determines that the number given is specified in gigabytes (multiplier 2**30).

T 'T' determines that the number given is specified in terabytes (multiplier 2**40). On 32-bit machines the unit terabytes is not available.

LOCK Attempt to lock the storage (pages locked by the host operating system).

UNLOCK Leave the store unlocked (no pages locked by the host operating system). This is the default.

Notes:

The actual upper limit is determined by the host system's architecture and operating system, the guest operating system and the amount of physical memory and available paging space.

The total of MAINSIZE and XPNDSIZE on host systems with a 32-bit architecture will be limited to less than 4G; host systems with a 64-bit architecture will be limited to less than 16E.

Use of storage sizes greater than supported by the guest operating system may generate incorrect results or error conditions within the guest operating system.

8.194.4 Overview Storage Allocation Units

Unit	Multiplier	Name (Symbol)	IEC Name (IEC Symbol)	Restrictions
M	2**20	Megabyte (MB)	Mebibyte (MiB)	
G	2**30	Gigabyte (GB)	Gibibyte (GiB)	
T	2**40	Terabyte (TB)	Tebibyte (TiB)	Not on 32-bit machines

Table 24: Storage Allocation Units

8.194.5 Examples

Example 1:

Display the current size of the expanded storage.

```
HHC00013I Herc command: 'xpndsize'
HHC17003I EXPANDED storage is 256 MBytes 'xpndsize'; storage is not locked
```

Figure 346: XPNDSIZE command (display size of expanded storage)

Example 2:

Set the size of the expanded storage to 128 MB. Do not lock the memory into the storage.

```
HHC00013I Herc command: 'xpndsize 128'
HHC17003I EXPANDED storage is 128 MBytes 'xpndsize'; storage is not locked
```

Figure 347: XPNDSIZE command (set size of unlocked expanded storage)

Example 3:

Set the size of the expanded storage to 512 MB. Lock the memory into the storage.

```
HHC00013I Herc command: 'xpndsize 512 lock'
HHC01428I Locking expanded storage
HHC17003I EXPANDED storage is 512 MBytes 'xpndsize'; storage is locked
```

Figure 348: XPNDSIZE command (set size of locked expanded storage)

8.195 YROFFSET (Set TOD clock offset from actual date)

8.195.1 Function

The YROFFSET command specifies the number of years the TOD clock is offset from the actual date. Positive numbers will move the clock forward in time while negative numbers will move it backward. A common value for non-Y2K-compliant operating systems is YROFFSET -28 which has the advantage that the day of the week and the presence or absence of February 29 is the same as the current year.

8.195.2 Syntax

Descriptive

YROFFSET {*+years* | *-years*}

Diagram

```
►►── YROFFSET ──┬── +years ──┬──────────────────────────────────────────►◄
                └── -years ──┘
```

8.195.3 Parameter

+years Specifies the number of years the TOD clock is offset positive from the actual date. This value may not be specified as greater than +/-142 years, the total range of the TOD clock. Specifying a value that causes the computed TOD clock year to be more than 142 years later than SYSEPOCH will produce unexpected results.

-years Specifies the number of years the TOD clock is offset positive from the actual date. This value may not be specified as greater than +/-142 years, the total range of the TOD clock. Specifying a value that causes the computed TOD clock year to be earlier than the value of SYSEPOCH will produce unexpected results.

8.195.4 Examples

Example 1:

Specify 28 years to offset the TOD clock from the actual date.

```
HHC00013I Herc command: 'yroffset -28'
HHC02204I yroffset       set to -28
```

Figure 349: YROFFSET command

9. Shared Device Support

9.1 Basics

Shared Device Support (see also "General Information" manual) allows multiple Hercules instances to share devices. The device will be local to one Hercules instance and remote to all other Hercules instances. The local instance is the server for that device and the remote instances are the clients. It is possible that each instance acts as both a client and a server. If a local instance declares a device as remote on another instance and on this remote instance the device is defined again as remote on a third instance, then the original instance will have to hop through this second instance to get to the real device.

It is not necessary to IPL an operating system on the device server. Any number of Hercules instances can act as a server in a "HERCPLEX".

When "SHRDPORT" is specified in the Hercules configuration the thread "shared_server" is started at the end of Hercules initialization. If Shared Device Support is requested on a device statement then the Hercules instances cannot initialize these devices until the server is started on each system. In this case the device trying to access a server gets the 'connecting' bit set on in the DEVBLK and the device still needs to initialize. After the shared server is started a thread is attached for each device that is connecting to complete the connection (the device init handler).

9.2 Caching

Cached records (i.e. CKD tracks or FBA blocks) are kept independently on both the client and server sides. Whenever the client issues a START request to initiate a channel program, the server will return a list of records to purge from the clients cache. These will have been updated by other clients since the last START request. If the list is too large the server will indicate that the client should purge all records for the device.

9.3 Compression

Data that would normally be transferred uncompressed between the client and the host can optionally be compressed by specifying the "COMP=n" keyword on the device configuration statement (see below) or on the attach command. The value n of the "COMP=n" keyword is the zlib compression parameter which must be a number between 1 and 9. A value closer to 1 means less compression but less processor time to perform the compression. A value closer to 9 means the data is compressed more but also more processor time is required to compress the data.

If the server is on localhost then you should not specify compression. Otherwise you are just stealing processor time from Hercules to facilitate compression/decompression. If the server is on a local network then a low value such as 1, 2 or 3 is recommended. There is a tradeoff curve, attempting to trade CPU cycles for network traffic to derive an optimal throughput.

If the devices on the server are compressed devices (i.e. CCKD or CFBA) then the records (track images or block groups) may be transferred compressed regardless of the "COMP=n" settings. This depends on whether the client supports the compression type (zlib or bzip2) of the record on the server and whether the record is actually compressed in the server cache.

An example may help to explain this: Suppose on the client that you execute one or more channel programs to read a record on a CKD track, update a record on the same track, and then read another (or the same) record on the track. For the first read the server will read the track image and pass it to the client as it was originally compressed in the file. To update a portion of the track image the server must uncompress the track image so data in it can be updated. When the client next reads from the track image the track image is uncompressed.

Specifying "COMP=n" means that uncompressed data sent to the client will be compressed. If the data to be sent to the client is already compressed then the data is sent as is unless the client has indicated that it does not support that compression algorithm.

9.4 Usage of Shared Devices

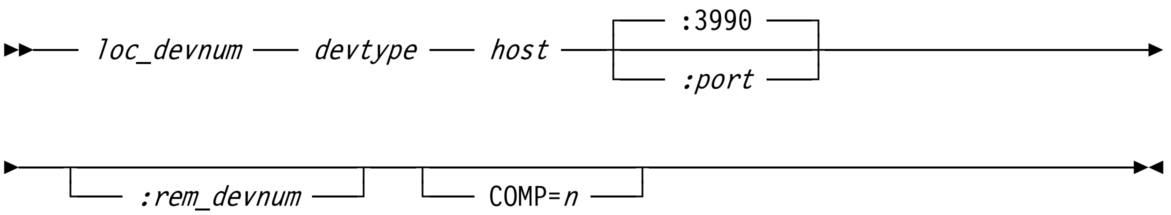
To use a device on a remote Hercules instance, instead of specifying a file name on the device statement, an IP address or a DNS name is specified.

9.4.1 Syntax

Descriptive

loc_devnum devtype host[*:port* | :3990] [*:rem_devnum*] [COMP=*n*]

Diagram



9.4.2 Parameter

loc_devnum This specifies the device address on the local Hercules instance.

***devtype* x** This is the device type.

host This specifies the host name or the IP address of the system where the Shared Device Server is running.

port The port number on which the Shared Device Server is listening. If the port number is omitted then the default port (3990) is used.

rem_devnum This is the device address of the device on the remote Hercules instance. If the remote device address is omitted then the default is the current device number on the local system.

COMP=*n* This keyword requests that the data has to be transferred compressed between the client and the server. The argument *n* specifies the compression level (1-9). A value closer to 1 means less compression but also less processor time to perform the compression. A value closer to 9 means the data is compressed more but also more processor time is required to compress the data.

9.4.3 Examples

Example 1:

There is a shared device server on the local host listening on port 3990 and we want to use its 0100 device as our 0100 device. The device statement will look like this:

```
0100 3390 localhost:3990:0100
```

Because the default port number is actually 3990 and the default remote device address is the same as the local device number the above statement (providing we do not have actually a file named 'localhost') could be shortened as follows:

```
0100 3390 localhost
```

Example 2:

Device sharing can also be split between multiple instances. For example, suppose the following device definitions and system parameter for instance A:

```
SHRDPORT 3990
0100 3390 localhost:3991
0101 3390 mvscat.dasd
```

And for instance B we have the following device definitions and system parameter:

```
SHRDPORT 3991
0100 3390 mvsres.dasd
0101 3390 localhost
```

In this case each instance acts as both a client and a server. Both instances of Hercules are running on the same machine.

The above examples may be clearer if we specify also all the default values. To show this the same configuration is used but in this case the Hercules instances are running on separate physical machines:

Hercules instance A (machine IP 192.168.200.1):

```
SHRDPORT 3990
0100 3390 192.168.200.2:3991:0100   # Remote on 192.168.200.2 (mvsres.dasd)
0101 3390 mvscat.dasd               # Local on  192.168.200.1 (mvscat.dasd)
```

Hercules instance B (machine IP 192.168.200.2):

```
SHRDPORT 3991
0100 3390 mvsres.dasd               # Local  on 192.168.200.2 (mvsres.dasd)
0101 3390 192.168.200.1:3990:0101   # Remote on 192.168.200.1 (mvscat.dasd)
```

10. Hercules 3270 Logo

10.1 Function

The Hercules 3270 logo is the initial welcome screen that is presented when a 3270 terminal connects to a Hercules 3270 device. While in previous releases of Hercules this logo screen was hardcoded it can now be customized.

The customized logo is stored in a plain text file which contains positioning orders, attributes and supports variable substitutions. Each line in the text file represents either an order statement or a plain text line. If no logo file is specified then Hercules uses a built-in default logo. Upon startup Hercules will first look in the current directory for a file called "herclogo.txt". The name of the logo file can also be specified as a startup option by using the "-b" flag. The file may also be specified using the "LOGOFILE" system parameter or can be changed during Hercules operation using the "HERCLOGO" console command.

10.2 Order Commands

Order commands are used to set the current buffer position, set the highlight and/or protected attributes, to force a skip to a new line and to specify text alignment.

10.2.1 Set Buffer Address

Descriptive

`@SBA` *x,y*

Diagram

►►── @SBA ── *x,y* ──────────────────────────────►◄

The Set Buffer Address order sets the current buffer position to row x and column y. Row and column both begin counting with zero.

10.2.2 Set Field

Descriptive

`@SF {H | P | HP}`

Diagram

```
►►── @SF ──┬── H ──┬──────────────────────────────────────────────►◄
           ├── P ──┤
           └── HP ─┘
```

The Set Field order sets the highlight ("H") and/or protected ("P") attribute.

10.2.3 New Line

Descriptive

```
@NL
```

Diagram

```
►►── @NL ─────────────────────────────────────────────────────────►◄
```

The New Line order forces a skip to a new line.

10.2.4 Align

Descriptive

```
@ALIGN {NONE | LEFT | RIGHT | CENTER}
```

Diagram

```
►►── @ALIGN ──┬── NONE ───┬───────────────────────────────────────►◄
              ├── LEFT ───┤
              ├── RIGHT ──┤
              └── CENTER ─┘
```

The Align order specifies the text alignment relative to the left and right borders of the terminal. When the alignment is other than "NONE" a new line is automatically inserted after each line of text. If the alignment is "NONE" then the text will be written without skipping to the next line.

10.3 Variables

It is possible to imbed substitution variables in the outgoing text. Substitution is indicated by enclosing the variable name between "$(" and ")". The Hercules version as an example can be specified by the following string: $(VERSION)

The following variables are defined. It is also possible to specify environment variable names.

- $(VERSION)
  The Hercules version.
- $(HOSTNAME)
  The host name, on which Hercules is running.
- $(HOSTOS)
  The host operating system.
- $(HOSTOSREL)
  The release of the host operating system.
- $(HOSTOSVER)
  The version of the host operating system.
- $(HOSTARCH)
  The host architecture.
- $(HOSTNUMCPUS)
  The number of host CPUs. UP (Uniprocessor for one CPU), or MP=n (Multiprocessor for more than one CPUs).
- $(LPARNAME)
  The LPAR name specified in the configuration file.
- $(CSS)
  The logical channel subsystem set or channel set for the terminal.
- $(SUBCHAN)
  The subchannel number for the terminal.
- $(CCUU), $(ccuu), $(CUU), $(cuu)
  Various forms of the device number of the terminal.

10.4 Sample

The file "herclogo.txt" is provided in the Hercules distribution as a sample template. This sample reflects the contents of the built-in logo.

```
@ALIGN NONE
@SBA 0,0
@SF P
Hercules Version  :
@SF HP
$(VERSION)
@NL
@SF P
Host name         :
```

```
@SF HP
$(HOSTNAME)
@NL
@SF P
Host OS           :
@SF HP
$(HOSTOS)-$(HOSTOSREL) $(HOSTOSVER)
@NL
@SF P
Host Architecture :
@SF HP
$(HOSTARCH)
@NL
@SF P
Processors        :
@SF HP
$(HOSTNUMCPUS)
@NL
@SF P
Chanl Subsys      :
@SF HP
$(CSS)
@NL
@SF P
Device number     :
@SF HP
$(CCUU)
@NL
@SF P
Subchannel        :
@SF HP
$(SUBCHAN)
@SF P
@ALIGN LEFT


           HHH          HHH   The S/370, ESA/390 and z/Architecture
           HHH          HHH                 Emulator
           HHH          HHH
           HHH          HHH  EEEE RRR   CCC U  U L    EEEE  SSS
           HHHHHHHHHHHHHHHH  E    R  R C    U  U L    E    S
           HHHHHHHHHHHHHHHH  EEE  RRR  C    U  U L    EEE   SS
           HHHHHHHHHHHHHHHH  E    R R  C    U  U L    E       S
           HHH          HHH  EEEE R  R  CCC  UU  LLLL EEEE SSS
           HHH          HHH
           HHH          HHH
           HHH          HHH     I can’t believe it’s not a MAINFRAME

           Copyright (c) 1999-2010 Roger Bowler, Jan Jaeger, and others
```

Figure 350: Logo File

This above sample file results in the following Hercules Logo screen:

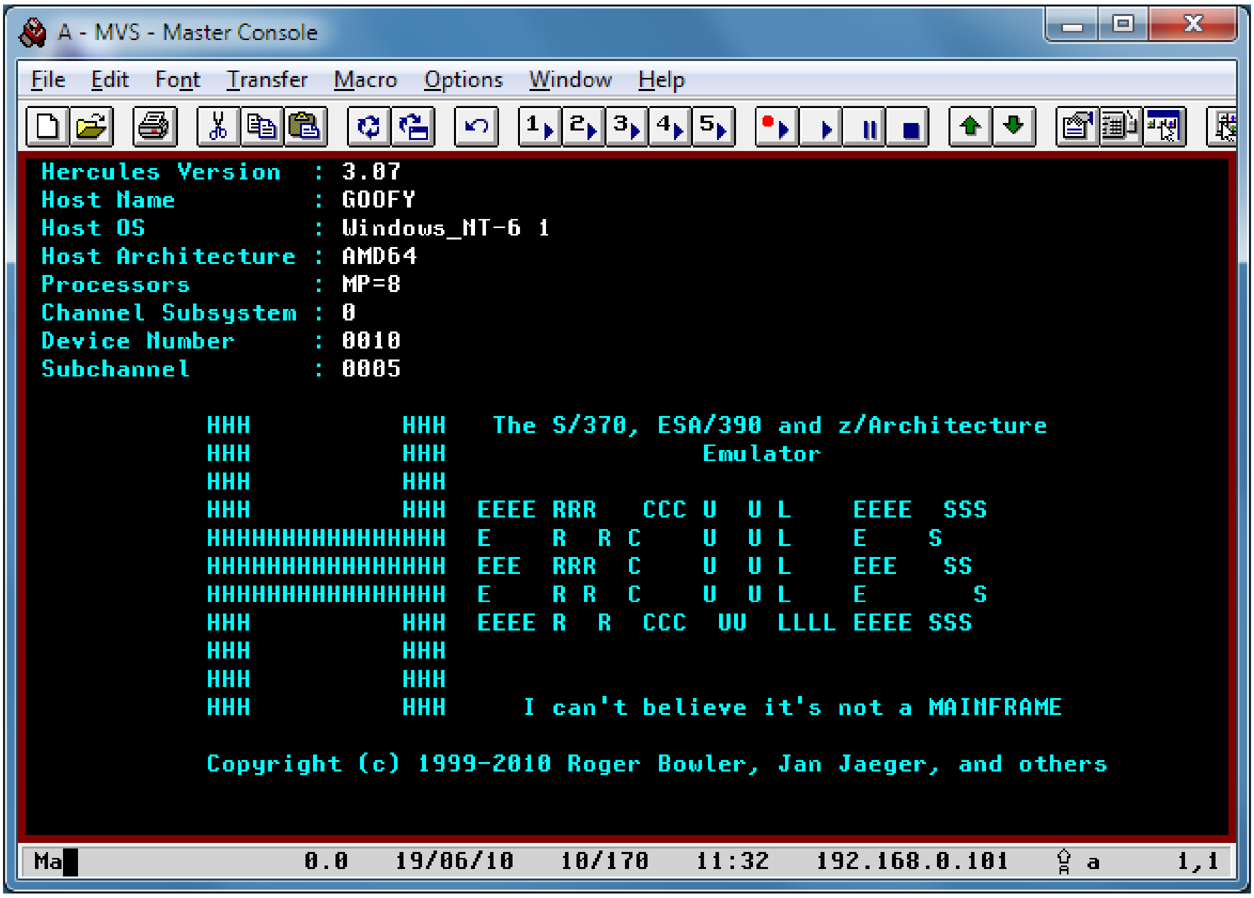


Figure 351: Logo Screen

11. Starting the Hercules Emulator

11.1 Starting Hercules in Native Mode

Hercules can be manually started from a Windows Command Prompt or can be started by calling a batch file (the preferred way). In both cases the syntax is the same.

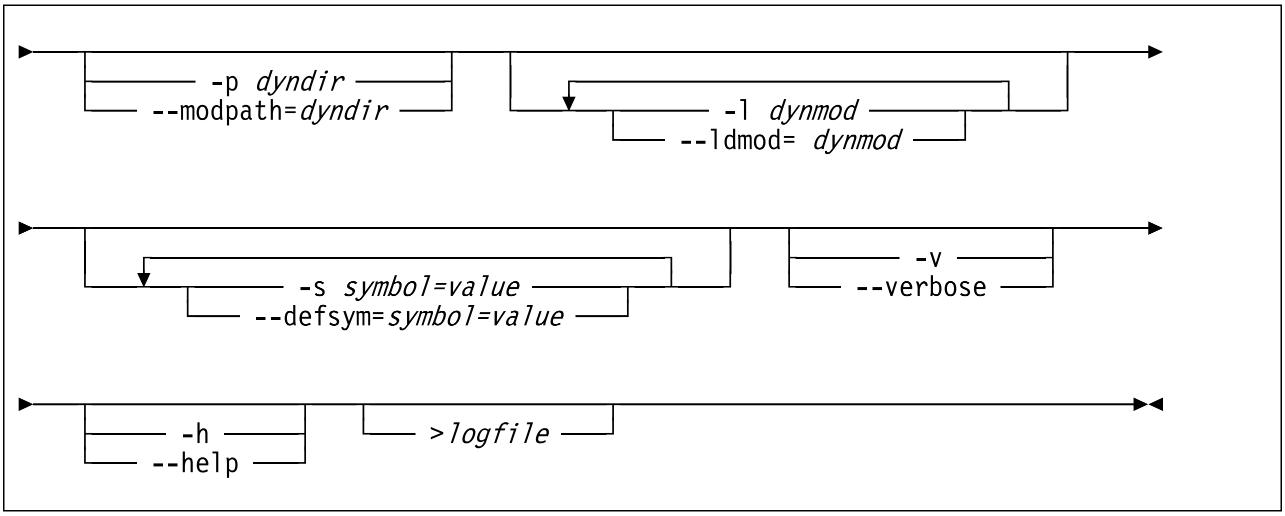
11.1.1 Syntax

Descriptive

```
HERCULES [{-f configfile          | --config=configfile}]
         [{-r rcfile              | --rcfile=rcfile}]
         [{-b logofile            | --herclogo=logofile}]
         [{-d                     | --daemon}]
         [{-p dyndir              | --modpath=dyndir}]
         [{-l dynmod [...]        | --ldmod=dynmod [...]}]
         [{-s symbol=value [...]  | --defsym=symbol=value [...]}]
         [{-v                     | --verbose}]
         [{-h                     | --help}]
         [>logfile]
```

Diagram

```
►►── HERCULES ──┬───────────────────────────┬──────────────────────►
                ├──────── -f configfile ────┤
                └─── --config=configfile ───┘

►──┬────────────────────────┬─────────────────────────────────────►
   ├────── -r rcfile ───────┤
   └─── --rcfile=rcfile ────┘

►──┬─────────────────────────────┬───┬──────────────┬─────────────►
   ├──────── -b logofile ────────┤   ├──── -d ──────┤
   └─── --herclogo=logofile ─────┘   └── --daemon ──┘
```



11.1.2 Parameter

-f *configfile*
--config=*configfile*

This is the name of the configuration file. If the filename is not specified the default is HERCULES.CNF. The name of the default configuration file may be overridden by the 'HERCULES_CNF' environment variable.

-r *rcfile*
--rcfile=*rcfile*

This specifies the name of the Hercules run-commands file. The run-commands file automatically executes panelcommands upon startup.

-b *logofile*
--herclogo=*logofile*

Specifies the name of a customized logo text file in the current Hercules directory (for details on how to customize a logofile see chapter 10).

-d
--daemon

Specifies that Hercules is to be run in “daemon” mode, wherein it runs invisibly with no attached console.

-p *dyndir*
--modpath=*dyndir*

This is the directory from which dynamic modules are to be loaded. The general search order is the following:

1. -p from startup argument.
2. HERCULES_LIB environment variable.
3. MODULESDIR compile time definition
4. The path where Hercules was found and if this is not resolved, then "hercules" is used as pathname.

-l *dynmod*
--ldmod=*dynmod*

This is the name of an additional dynamic load module to be loaded at startup. More than one additional module may be specified, although each must be preceded with the “-l” option.

-s sym=*val*
--defsym=*symbol=value*

Specifies a symbol definition where *sym* is the name of the symbol and *val* is the value that is assigned to the symbol. This option has the same effect as a DEFSYM statement in the configuration file.

If there is a DEFSYM statement in the configuration file that has the same name for the symbol as a symbol defined in the startup command then the configuration DEFSYM statement takes precedence.

-v
--verbose

Specifies that Hercules is to be started with “MSGLEVEL VERBOSE” which displays additional messages during configuration file processing.

-h
--help

Display help information regarding the syntax of the command-line arguments.

>logfile

This is an optional log file which will receive a copy of all messages displayed on the control panel.

11.1.3 Examples

Example 1:

Start Hercules using the configuration file “D:\MVS\CONF\MVS38J.CONF” and write all messages to the log file “D:\MVS\LOGS\HERCULES.LOG”.

```
HERCULES -f D:\MVS\CONF\MVS38J.CONF >D:\MVS\LOGS\HERCULES.LOG
```

Example 2:

Start Hercules using the configuration file “D:\S390\CONF\zLINUX.CNF” and with a logofile called “zLINUX_Logo.txt”. Display additional messages during configuration file processing.

```
HERCULES -f D:\S390\CONF\zLINUX.CNF -b zLINUX_Logo.txt -v
```

Example 3:

Start Hercules using the configuration file “D:\S390\CONF\zLINUX.CNF” and define a symbol named "DASDPATH" that contains the path of the DASD files ("D:\MVS\DASD").

```
HERCULES -f D:\MVS\CONF\MVS38J.CONF -s DASDPATH=D:\MVS\DASD
```

Example 4:

Start Hercules using the configuration file “D:\MVS\CONF\MVS38J.CONF” and automatically execute the panel commands contained in the run-commands file “D:\MVS\CONF\MVS38.RC”

```
HERCULES -f D:\MVS\CONF\MVS38J.CONF -r D:\MVS\CONF\MVS38J.RC
```

11.2 Starting Hercules with the Windows GUI

The Windows GUI can be started by just clicking on the GUI icon. Then all necessary settings required to start Hercules can be specified using the GUI itself.

Alternatively the Windows GUI may also be manually started from a Windows Command Prompt or can be started by calling a batch file (which is the preferred way of these two options). In both cases the syntax is the same.

11.2.1 Syntax

Descriptive

HERCGUI [-f *configfile*]

Diagram

```
►►── HERCGUI ──┬─────────────────┬──────────────────────────►◄
               └── -f configfile ─┘
```

11.2.2 Parameter

-f *configfile* This is the name of a configuration file. If the HercGUI is started with the name of a configuration file then the HercGUI will automatically "power on" Hercules once the HercGUI itself is started. This is called the "Auto-Power-On" feature. When using this feature in conjunction with a Hercules ".RC" (run commands) file it is possible to totally automate Hercules startup and IPL.

11.2.3 Examples

Example 1:

Start Hercules using the HercGUI with the "Auto-Power-On" feature using the configuration file "D:\MVS\CONF\MVS38J.CONF".

```
HERCGUI -f D:\MVS\CONF\MVS38J.CONF
```

Example 2:

Start the HercGUI without a configuration file.

```
HERCGUI
```

11.3 Starting Hercules with the Hercules Studio

The Hercules Studio can be started by navigating to the 'Applications -> System Tools -> Hercules Studio' menu entry and selecting it from there. Then all necessary settings required to start Hercules can be specified using the Hercules Studio itself.

Alternatively the Hercules Studio may also be manually started from a command shell or can be started by calling a script file. In both cases the syntax is the same.

11.3.1 Syntax

Descriptive

HerculesStudio [-f *configfile*] [-r *rcfile*]

Diagram

```
►►── HerculesStudio ──┬────────────────────┬──────────────────────►
                      └── -f configfile ───┘

►──┬───────────────┬──────────────────────────────────────────────►◄
   └── -r rcfile ──┘
```

11.3.2 Parameter

-f *configfile* This is the name of a configuration file. If the Hercules Studio is started with the name of a configuration file then it will automatically “power on” Hercules once the Hercules Studio itself is started. This is called the “Auto-Power-On” feature. When using this feature in conjunction with a Hercules “.RC” (run-commands) file it is possible to totally automate Hercules startup and IPL.

-r *rcfile* This is the name of an optional RC (run-commands) file. This file contains Hercules console commands that are issued exactly as if the commands were entered from the Hercules system console.

11.3.3 Examples

Example 1:

Start Hercules Studio using the configuration file “mvs38j.cnf” and the run-commands file "autoipl.rc".

```
HerculesStudio -f mvs38j.cnf –r autoipl.rc
```

12. The Run-Commands File

12.1 Function

Hercules supports the ability to have console commands automatically executed at startup via the 'run-commands' file. If the run-commands file is found to exist when Hercules starts then each line of the file is read and interpreted as a console command, exactly as if the command were entered from the HMC system console.

The default filename for the run-commands file is "hercules.rc". The default name may be overridden by setting the "HERCULES_RC" environment variable.

Except for the 'pause' command (see below), each command read from the run-commands file is logged to the console preceded by a “>”' (greater-than sign) character. Thus you can easily distinguish between console commands entered from the keyboard from those entered via the run-commands file.

Comment lines starting with '#' are treated as "silent comments" and are thus not logged to the console whereas comment lines starting with '*' are treated as "loud comments" and *will* be logged.

12.2 Run-Commands File Statements

There is a number of different types of commands that can be specified within a run-commands file. These commands are the following:

- “*herccmd*” (any valid Hercules console command)
- PAUSE (delay the processing of the next command)
- # (issue silent comment)
- * (issue loud comment)

12.2.1 Hercules Console Commands

Descriptive

herccmd

Diagram

▸▸── *herccmd* ──────────────────────────────▸◂

In the run-commands file any valid Hercules console command (“*herccmd*”) can be specified. This includes the ‘sh’ (shell) command. Each command that is read from the run-commands file is logged to the console preceded by a ‘>’' (“greater-than” sign) character. This is to distinguish between console commands entered manually from the keyboard from thosecommands entered via the run-commands file.

Examples:

```
MAXRATES MIDNIGHT
IPL 0148
```

12.2.2 PAUSE

Descriptive

PAUSE {*nnn* | *nnn.nnn*}

Diagram

```
►►── PAUSE ──┬── nnn ─────┬──────────────────────────────►◄
             └── nnn.nnn ─┘
```

A pause command is supported in order to introduce a brief delay before reading and processing the next line in the file. The value *nnn* can be any number from 1 to 999 and specifies the number of seconds to delay before reading the next line. Beginning with Hercules version 4.0 pause can also be specified for sub-second delays (e.g. "PAUSE 0.125" to delay the next command for 125 milliseconds).

Examples:

```
PAUSE 2
PAUSE 0.375
```

12.2.3 # (Silent Comment)

Descriptive

[*comment*]

Diagram

```
►►── # ── comment ──────────────────────────────────►◄
```

Lines starting with '#' are treated as "silent comments". They are intended as comments just for the run-commands file itself and are thus not logged to the console. That means they are are not visible when Hercules is started and the run-commands file is executed.

Examples:

```
# ====================================================
# This comment block will not be logged to the console
# ====================================================
```

12.2.4 * (Loud Comment)

Descriptive

`* [`*`comment`*`]`

Diagram

```
►►── * ── comment ──────────────────────────────────────────►◄
```

Lines starting with '*' are treated as "loud comments". They will be logged to the console and thus are visible when the run-commands file is executed.

Examples:

```
* ================================================
* This comment block will be logged to the console
* ================================================
```

12.3 Automating Hercules Startup

Creative use of the run-commands file can completely automate Hercules startup. The next figure shows a sample run-commands file that does the following:

- "pause 1" – Wait one second before processing the first Hercules command. (1)
- "cmdlevel all" – Set the current command group to ALL. This allows that all Hercules commands can be issued. (2)
- "help" – Display all valid commands for the chosen command group that has been set before with CMDLEVEL. (3)
- "maxrates midnight" – Set the MAXRATES reporting interval to midnight to have the MIPS and I/O statistics issued date aligned. (4)
- "sh startgui "D:\MVS\TERMINALS.BAT" – Execute the named batch file in the shell to start a couple of tn3270 sessions. See details below. (5)
- "pause 3" – Wait three seconds until all tn3270 sessions have been successfully connected. (6)

- “ipl 0148” – Finally IPL MVS 3.8J from device address 148. (7)

```
# ===============================================================================
# Hercules RUN-COMMANDS (RC) File
# ===============================================================================

# -------------------------------------------------------------------------------
# Wait 1 second before issuing Hercules commands                             (1)
# -------------------------------------------------------------------------------
pause 1

# -------------------------------------------------------------------------------
# Set command level to 'ALL'                                                  (2)
# -------------------------------------------------------------------------------
cmdlevel all

# -------------------------------------------------------------------------------
# Display all valid Hercules panel commands                                   (3)
# -------------------------------------------------------------------------------
help

# -------------------------------------------------------------------------------
# Display and reset maxrates values                                           (4)
# -------------------------------------------------------------------------------
maxrates midnight

# -------------------------------------------------------------------------------
# Start tn3270 sessions                                                       (5)
# -------------------------------------------------------------------------------
sh startgui "D:\MVS\TERMINALS.BAT"

# -------------------------------------------------------------------------------
# Wait 3 second before IPL’ing MVS                                            (6)
# -------------------------------------------------------------------------------
pause 3

# -------------------------------------------------------------------------------
# IPL MVS 3.8J                                                                (7)
# -------------------------------------------------------------------------------
ipl 0148

# ===============================================================================
# (RC EOF)
# ===============================================================================
```

Figure 352: Run-commands (.rc) file

With the run-commands file shown above the Hercules startup is fully automated. After starting Hercules there is no manual intervention required until MVS is up and running.

The next figure shows the batch file that is used in the run-commands file (step 5) to start the tn3270 sessions.

```
@ECHO OFF

REM *******************************************************************************************
REM Start master console and wait for connection to Hercules
REM *******************************************************************************************
START /D "C:\Program Files (x86)\TomBrennanSoftware\VistaTN3270" VistaTN3270.EXE MVS38J_MST.SES
CHOICE /t 1 /d y

REM ******************************************************************************************
REM Start operator console and wait for connection to Hercules
REM *******************************************************************************************
START /D "C:\Program Files (x86)\TomBrennanSoftware\VistaTN3270" VistaTN3270.EXE MVS38J_OPR.SES
CHOICE /t 1 /d y

REM *******************************************************************************************
REM Start TSO terminals and wait for connection to Hercules
REM *******************************************************************************************
START /D "C:\Program Files (x86)\TomBrennanSoftware\VistaTN3270" VistaTN3270.EXE MVS38J_TSO.SES
CHOICE /t 1 /d y
START /D "C:\Program Files (x86)\TomBrennanSoftware\VistaTN3270" VistaTN3270.EXE MVS38J_TSO.SES

EXIT
```

Figure 353: Batch file to start tn3270 sessions

The CHOICE command used in the batch file is to issue a delay before trying to connect the next tn3270 session.

The “/t” parameter specifies the number of seconds to pause before a default choice is made. Option “/d” specifies the default choice that is issued after the timeout has expired. The character given as argument must be in the set of choices specified by the “/c” option (defaults are ‘Y’ and ‘N’).

13. The “Hercules Automatic Operator” (HAO) Facility

13.1 HAO Introduction

The Hercules Automatic Operator (HAO) feature is a facility which can automatically issue panel commands in response to specific messages appearing on the Hercules console.

To use the Hercules Automatic Operator facility, you first define a rule consisting of a target and the associated command. The target is a regular expression pattern used to match against the text of the various messages that Hercules issues as it runs. Whenever a match is found, the rule “fires” and ist associated command is automatically issued.

The Hercules Automatic Operator facility only operates on messages issued to the Hercules console. These messages may originate from Hercules itself or from the guest operating system via the SCP SYSCONS interface or via the integrated console printer-keyboard (3215-C or 1052-C). HAO cannot intercept messages issued by the guest operating system to its own terminals.

13.2 Defining HAO Rules

To define a HAO rule, enter the command

```
HAO TGT target
```

to define the rule’s target match pattern, followed by the command

```
HAO CMD command
```

to define the rule’s associated panel command.

The target is a regular expression as defined by your host platform. When running on Linux, Hercules uses POSIX Extended Regular Expression syntax. On a Windows platform, regular expression support is provided by Perl Compatible Regular Expression (PCRE). The HAO facility can only be used if regular expression support was included in Hercules at build time.

The associated *command* is whatever valid Hercules panel command you wish to issue in response to a message being issued that matches the given *target* pattern.

13.3 Deleting HAO Rules

To delete a fully or partially defined HAO rule, first use the the following command to get a list of all of the defined (or partially defined) rules

```
HAO LIST [nnn]
```

Where *nnn* is the (optional) number of an existing rule. This gives you the list of all rules with the specified identifier or lists the rule with identifier ‘*nnn*’. Then use the next command to delete the specific rule identified by the identifier ‘*nnn*’

```
HAO DEL nnn
```

To every rule there is a number assigned as the rule is defined. The rules then are subsequently identified by their numeric value.

It is also possible to delete all defined or partially defined rules by issuing the following command

```
HAO CLEAR
```

13.4 Substituting Substrings

The command may contain special variables ($1, $2, $3, etc.) which will be replaced by the values of "capturing groups" in the match pattern. A capturing group is a part of the regular expression enclosed in parenthesis which is matched with text in the target message. In this way commands may be constructed which contain substrings extracted from the message which triggered the command.

The following special variables are recognized:

- $1 to $9 – the text which matched the first to nineth capturing group in the target regular expression.
- $` - the text preceeding the regular expression match.
- $' – the text following the regular expression match.
- $$ - replaced by a single dollar sign.

Note that the substitution of a $*n* variable does not occur if there are fewer than *n* capturing groups in the regular expression.

13.5 Limitations

The current implementation limits the total number of defined rules to 64. This limit may be raised by increasing the value of the HAO_MAXRULE constant in module "hao.c" and rebuilding Hercules.

All defined rules are checked for a match each time Hercules issues a message. There is no way to specify "stop processing subsequent rules". If a message is issued that matches two or more rules, each associated command is then issued in sequence.

13.6 Examples

Example 1:

Issue the command "i 0700" in response to the message:

```
HHC01090I 0:0700 COMM: client 127.0.0.1 devtype 3270: connection reset
```

The following HAO commands define the desired rule:

```
HAO TGT HHC01090I 0:([0-9A-F]{3,4})
HAO CMD i $1
```

Example 2:

Use the dot matrix display of a 3480 tape unit to implement an automatic tape library (please note that the first HAO command has been split over two lines to fit on the page):

```
HAO TGT HHC00224I 0:([0-9A-F]{4}) Tape file *, type aws:
        display (?:".{8}" / )?"M([A-Z0-9]{1,6})\s*S"
HAO CMD devinit $1 D:\MVSTAPES\$2.AWSTAPE
```

14. REXX Support

14.1 Prerequisites

Support for Rexx (both ooRexx and Regina Rexx) is built into the downloadable version of Hercules. The only thing needed is to install one (or both) of the two supported Rexx packages (see Appendix H. Links).

If the Rexx packages are installed with the standard installation procedures (using the provided installers) then the needed environment variables are created automatically during the installation and no manual action is required. For more information on how to build Rexx support into Hercules please refer to the “Installation Guide”.

14.2 Using Rexx

Rexx can be used in various ways within Hercules:

- Invoking Rexx scripts explicitely through the EXEC console command.
- Invoking Rexx in the run-commands file.
- Invoking Rexx in the configuration file.

14.2.1 Explicitely invoking Rexx

The EXEC console command may be used to explicitely invoke a Rexx script. The required argument is the name (and optionally path) of the Rexx script. Optional arguments will be passed to the Rexx script.

Example 1:

```
EXEC d:\rexx\script.rex arg1 arg2
```

In the above example the Rexx script named “script.rex” located on drive d: in directory “rexx” will be called with “arg1” and “arg2” as arguments.

14.2.2 Implicitely invoking Rexx

Rexx will be invoked implicitely if existing script files (e.g. a configuration file or a run-commands file) start with “/*”.

Example:

```
/* REXX */
SELECT
     WHEN SYSTEM = 'S/370' THEN DO
          ADDRESS HERCULES 'ARCHLVL S/370'
          ADDRESS HERCULES 'ENGINES 1*CP'
          ADDRESS HERCULES 'MAINSIZE 16'
          ADDRESS HERCULES 'XPNDSIZE 0'
          END
```

```
    WHEN SYSTEM = 'ESA/390' THEN DO
         ADDRESS HERCULES 'ARCHLVL ESA/390'
         ADDRESS HERCULES 'MAINSIZE 2048'
         ADDRESS HERCULES 'ENGINES 2*CP'
         ADDRESS HERCULES 'XPNDSIZE 2048'
         END
    WHEN SYSTEM = 'z/ARCH' THEN DO
         ADDRESS HERCULES 'ARCHLVL z/ARCH'
         ADDRESS HERCULES 'MAINSIZE 4096'
         ADDRESS HERCULES 'ENGINES 4*CP,2*AP'
         ADDRESS HERCULES 'XPNDSIZE 0'
         END
    OTHERWISE DO
         SAY 'Invalid architecture mode specified'
         RETURN 16
END
```

The above extract from a configuration file shows the different settings of system parameters depending on the value of a variable.

14.3 Command Environment

Hercules commands can be issued from Rexx through the Hercules command environment. The Hercules command environment is the default environment, therefore it is not necessary to specify it, when using Hercules commands within a Rexx script.

Example 1:

Executing the Hercules ‘ARCHLVL’ command from a Rexx script specifying the Hercules command environment.

```
ADDRESS HERCULES 'ARCHLVL S/370'
```

Example 2:

Executing the Hercules ‘ARCHLVL’ command from a Rexx script without specifying the Hercules command environment.

```
'ARCHLVL S/370'
```

14.4 The Rexx Builtin Function “value()”

Hercules symbols and environment variables can be retrieved with “value()” or “getenv()”. The builtin function “getenv()” is obsolete and has been replaced by “value()” and should no longer be used.

14.4.1 Function

The Rexx builtin function “value()” is used to retrieve the value of existing variables. On optional parameter can be used to searchwithin a specific variable pool. To retrieve the values of Hercules symbols the “SYSTEM” pool must be specified, to retrieve the values of Hercules (and other) environment variables the “ENVIRONMENT” pool must be specified. See the Regina Rexx documentation for more details on “value()”.

14.4.2 Syntax

Descriptive

```
VALUE (symbol [,[value] ,[pool]])
```

Diagram

```
►►── VALUE ── ( ── symbol ──────────────────────────────────────────►

►──┬────────────────────────────────────────────────────┬── ) ──────►◄
   └── , ──┬────────────┬── , ──┬───────────┬──┘
           └── value ───┘       └── pool ───┘
```

14.4.3 Parameter

symbol This names an existing variable. If *symbol* does not name an existing variable, the default value is returned, and the NOVALUE condition is not raised.

If *symbol* is not a valid symbol name and the function is used to access an normal Rexx variable, an error occurs.

value If the optional second parameter *value* is specified, the variable will be set to that value, after the old value has been extracted.

pool The optional parameter *pool* might be specified to select a particular variable pool to search for *symbol*. The contents and format of *pool* is implementation dependent.

The default is to search in the variables at the current procedural level in Rexx. Which *pool*s that are available is implementation dependent, but typically one can set variables in application programs or in the operating system.

Specify “SYSTEM” to retrieve the value of a Hercules symbol or “ENVIRONMENT” to retrieve the value of a Hercules (or another) environment variable.

14.4.4 Examples

Example 1:

Retrieve the value of the Hercules symbol “version”.

```
say 'Hercules version = ' VALUE('version',,'SYSTEM')
```

Example 2:

Retrieve the value of the Hercules environment variable “HERCULES_RC”.

```
say 'Hercules RC = ' value('HERCULES_RC',,'ENVIRONMENT')
```

14.5 Error Handling

The error handling differentiates between the following types of errors:

- Command Errors
- Command Failures

An invalid command (‘command not found’) is treated as a ‘command error’. Command errors may be handled with “SIGNAL ON ERROR”.

Failures in an otherwise valid command are treated as ‘command failures’. Command failures may be handled with “SIGNAL ON FAILURE”.

Command return codes < 0 are interpreted as ERROR, return codes > 0 are interpreted as FAILURE. Hercules will abort the startup process if the configuration file Rexx script returns with a non-zero return code.

15. Submitting Jobs via the Socket Reader

15.1 Socket Reader Basics

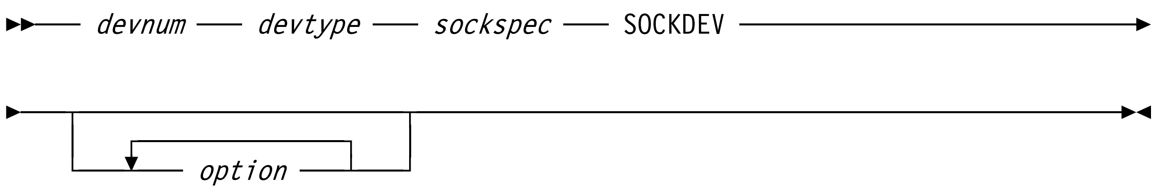
The “sockdev” option allows you to submit cards directly to a Hercules card reader from outside of Hercules. The card reader must be defined with the “sockdev” keyword and either a TCP/IP port number or the name of a Unix Domain Socket. Then whenever you want to submit a card deck to that particular card reader, you use an external program to connect to the socket and transmit the cards to the reader. Socket readers were implemented by Fish, based upon code originally contributed by Malcolm Beattie.

Socket readers are defined in the Hercules configuration like this:

Descriptive

devnum devtype sockspec SOCKDEV [*option* [*option* ...]]

Diagram



Parameters *devnum*, *devtype* and the keyword SOCKDEV are the same as described in chapter 6.5 (“Card Reader Devices”).

The socket specification *sockspec* can take any of the following formats:

ipaddr:port The reader listens on the specific IP address and port number. *ipaddr* must be The IP address of an interface on the local system.

For example, “127.0.0.1:1234” is used to accept only jobs submitted locally via the loopback interface.

hostname:port Similar to the previous example, where *hostname* must resolve to an IP address belonging to the local system.

Example: “localhost:1234”.

port The reader listens on the specified port number and accepts jobs submitted from any IP address defined on the local system.

Example: “1234”.

path/name The reader listens on the specified Unix Domain Socket.

Example: “/tmp/hercrdr.00C”

Examples:

```
000A   2501   127.0.0.1:2501 SOCKDEV ASCII TRUNC EOF
000C   3505   localhost:1234 SOCKDEV ASCII TRUNC EOF
0012   3505   3505 SOCKDEV ASCII TRUNC EOF
0014   2501   /tmp/hercrdr.014 SOCKDEV ASCII TRUNC EOF
```

15.2 Submitting Jobs from Windows

The "HercRdr" program is distributed as part of Fish's GUI package and allows you to send jobs to a socket reader via TCP/IP. Simply enter "HercRdr" from the command line (i.e. from a "Command Prompt" window if you are using Windows NT / 2K / XP) to submit your file(s).

The following screen print shows the help information from the HercRdr program that is displayed whenever the program is called without parameters:

```
D:\HERCULES>hercrdr
Submits card file(s) to a Hercules card reader bound to a given socket:port.

Format:

  HERCRDR [-nnn] [host:port] file [file...]

Where:

  -nnn        timeout value in seconds (1-999; default is 3)
  host:port   sock_spec of target reader (if not specified,
              value of HERCRDR environ var is used, if set)
  file        file(s) to be submitted

Examples:

  HERCRDR localhost:1234 fileone.txt filetwo.txt
  set HERCRDR=localhost:1234
  HERCRDR file3.txt file4.txt
  HERCRDR override:5678 filefive.txt
  HERCRDR 192.168.0.1:5678 666.txt 777.txt 888.txt 999.txt

Returns:

 -1    unclassified error
  0    file(s) successfully submitted
  1    no route to host (bad sock_spec or connection refused)
  2    timeout value exceeded while trying to connect
  3    transmission error (e.g. connection prematurely closed)
  4    file not found (or other file error)
```

Figure 354: HercRdr Help Screen

15.2.1 How to submit jobs directly from SPF/PC

If you are lucky enough to have a copy of the excellent editor SPF/PC Version4 or SPF/Pro (produced by CTC but unfortunately no longer available), then you can submit jobs directly from your edit session. The SUB command can be implemented by means of a REXX macro such as the following one, provided by Volker Bandke:

```
/* +---------------------------- REXX ----------------------------+ */
/*                                                                  */
/*      Name: D:\APPS\SPFPRO\REXX\USER\SUB.SPF                      */
/*                                                                  */
/*      Type: SPF edit macro                                        */
/*                                                                  */
/*      Desc: submit JCL to MVS 3.8                                 */
/*                                                                  */
/*      Creation date: 24 Aug 1999, creation time: 18:49:40         */
/*                                                                  */
/*      Author: (c) Volker Bandke                                   */
/*                                                                  */
/* +--------------------------------------------------------------+ */
'isredit macro (p1 p2 p3 p4 p5 p6 p7 p8 p9)'
"ISREDIT (member) = MEMBER"
"ISPEXEC CONTROL ERRORS CANCEL"
parse upper var member file '.' ext
do
'ISREDIT REPLACE' $$$$$$$$.SPF '.ZF .ZL'
ADDRESS "CMD" "HERCRDR 192.168.1.102:3505 $$$$$$$$.SPF"
zedsmsg = 'File submitted'
zedlmsg = 'The member '||member||' has been submitted to MVS'
'ispexec setmsg msg(isrz000)'
ADDRESS "CMD" "DELETE $$$$$$$$.SPF"
end
EXIT 0
```

Figure 355: SUBmit REXX for SPF/PC

15.3 Submitting Jobs from Unix

Submitting jobs from Unix can be done in two different ways, described in detail in the next sections:

- Using a Perl script
- Using the netcat program

15.3.1 Using a Perl script

Malcolm Beattie has provided a simple Perl script which can submit jobs using either TCP/IP or Unix Domain Sockets. The script is invoked using one of the following command formats:

```
hercsub 192.168.1.102:3505 dummy.jcl
hercsub /tmp/hercrdr.00C dummy.jcl
```

Here is the Hercsub script:

```
============== hercsub ================
#!/usr/bin/perl
use Socket;
use IO::Socket::UNIX;
use IO::Socket::INET;

if (@ARGV < 1) {
  print STDERR "Usage: hercsub socket_spec [job]\n";
  exit 2;
}

my $spec = shift @ARGV;
my $sock;

if ($spec =~ m{^/}) {
  $sock = IO::Socket::UNIX->new(Peer => $spec);
} else {
  $sock = IO::Socket::INET->new(PeerAddr => $spec);
}

die "Failed to connect to socket $spec: $!\n" unless defined($sock);

while (<>) {
  print $sock $_;
}
========== end of hercsub =============
```

Figure 356: The Hercsub Perl Script

15.3.2 Using the netcat program

The netcat (nc) program can also be used to submit files to a Hercules reader via TCP/IP. Install netcat (which is useful for many other things as well) and use the following syntax:

```
nc –w1 localhost 1234 < dummy.jcl
```

For more information on the netcat program, type *man nc* from the Unix shell.

Appendix A: Supported DASD Device Types

The following tables show the supported DASD device types and models with their sizes. The symbol “[*]” in the size column means that any size can be specified, else the size defaults to the first listed model.

CKD Devices

Devicetype-Model	Cylinders	Alternate Cylinders
IBM 2311	[*]	
IBM 2311-1	200	2
IBM 2314	[*]	
IBM 2314	200	3
IBM 3330	[*]	
IBM 3330-1	404	7
IBM 3330-2	808	7
IBM 3330-11	808	7
IBM 3340	[*]	
IBM 3340-1	348	1
IBM 3340-35	348	1
IBM 3340-2	696	2
IBM 3340-70	696	2
IBM 3350	[*]	
IBM 3350-1	555	5
IBM 3375	[*]	
IBM 3375-1	959	1
IBM 3380	[*]	
IBM 3380-1	885	1
IBM 3380-A	885	1
IBM 3380-B	885	1

Devicetype-Model	Cylinders	Alternate Cylinders
IBM 3380-D	885	1
IBM 3380-J	885	1
IBM 3380-2	1770	2
IBM 3380-E	1770	2
IBM 3380-3	2665	3
IBM 3380-K	2665	3
EMC 3380 K+	3339	3
EMC 3380 K++	3993	3
IBM 3390	[*]	1
IBM 3390-1	1113	1
IBM 3390-2	2226	2
IBM 3390-3	3339	1
IBM 3390-9	10017	3
IBM 3390-27	32760	3
IBM 3390-54	65520	3
IBM 9345	[*]	
IBM 9345-1	1440	0
IBM 9345-2	2156	0

Table 25: Supported CKD DASD Devices

FBA Devices

Devicetype-Model	Blocks
IBM 3310	[*]
IBM 3310-1	125664
IBM 3370	[*]
IBM 3370-A1	558000
IBM 3370-B1	558000
IBM 3370-A2	712752
IBM 3370-B2	712752
IBM 9313	[*]
IBM 9313-1	246240
IBM 9332	[*]
IBM 9332-200	360036
IBM 9332-400	360036
IBM 9336-600	554800
IBM 9335	[*]
IBM 9335-1	804714
IBM 9336	[*]
IBM 9336-10	920115
IBM 9336-20	1672881
IBM 9336-25	1672881
IBM 0671-08	513072
IBM 0671	574560
IBM 0671-04	624456

Table 26: Supported FBA DASD Devices

Appendix B. Configuration of Emulated CPUs

This appendix describes the cooperation of the system parameter related to the emulated CPUs in more detail. The system parameter that define the emulated CPUs are:

- ENGINES (Processor engine type)
- MAXCPU (Maximum number of CPUs)
- NUMCPU (Number of emulated CPUs)

B.1 General Explanations and Rules

The ENGINES parameter specifies the type of engine for each emulated processor (valid types are CP, AP, IP or IL). MAXCPU specifies the maximum number of installed processor engines, whereas NUMCPU defines the number of emulated processor engines that are configured online at IML time.

The ENGINES statement specifies any mixture of CPU types up to MAXCPU, the maximum number of installed processors. NUMCPU configures the processor engines in the order that they are specified on the ENGINES statement.

NUMCPU must be less than or equal to MAXCPU. If NUMCPU is larger than MAXCPU then an error message is issued during the processing of the Hercules configuration file. If it is less than MAXCPU then the remaining engines can be configured online later by the operating system.

B.2 Examples

Some examples show the cooperation of these statements and the resulting configurations. In the tables the following abbreviations and colours are used:

CPUs and status

CP	CPU of type CP specified
IP	CPU of type IP specified
AP	CPU of type AP specified
(D)	CPU got its type by default
O	CPU is taken online
F	CPU is taken offline

Colours

xx	Green - Resulting configuration as expected

xx	Yellow - Resulting configuration may or may be not as expected (defaults or ignored engines)

xx	Red - Resulting configuration is in error. The Hercules configuration file must be corrected.

Example 1:

```
ENGINES CP,CP,CP,AP,IP,IP,CP,CP
MAXCPU 8
NUMCPU 8
```

All CPUs are configured online with the type defined.

CPU and State	CPU 1	CPU 2	CPU 3	CPU 4	CPU 5	CPU 6	CPU 7	CPU 8
MAXCPU	1	2	3	4	5	6	7	8
NUMCPU	1	2	3	4	5	6	7	8
ENGINES	CP	CP	CP	AP	IP	IP	CP	CP
CPU type	CP	CP	CP	AP	IP	IP	CP	CP
Status	O	O	O	O	O	O	O	O

Table 27: Correct CPU configuration (example 1)

Example 2:

```
ENGINES CP,CP,CP,AP,IP,IP,CP,CP
MAXCPU 8
NUMCPU 4
```

All CPUs are defined with the specified type and the first four CPUs are taken online. The rest of the CPUs are taken offline.

CPU and State	CPU 1	CPU 2	CPU 3	CPU 4	CPU 5	CPU 6	CPU 7	CPU 8
MAXCPU	1	2	3	4	5	6	7	8
NUMCPU	1	2	3	4	---	---	---	---
ENGINES	CP	CP	CP	AP	IP	IP	CP	CP
CPU type	CP	CP	CP	AP	IP	IP	CP	CP
Status	O	O	O	O	F	F	F	F

Table 28: Correct CPU configuration (example 2)

Example 3:

```
ENGINES CP,CP,CP,AP,IP,IP,CP,CP
MAXCPU 6
NUMCPU 4
```

All CPUs up to the number of MAXCPU are defined with their specified type. Four CPUs, as specified in NUMCPU, are taken online, the other two CPUs are taken offline. Please note that the two excess CPUs that are specified in the ENGINES are ignored, because MAXCPU is set only to six engines.

CPU and State	CPU 1	CPU 2	CPU 3	CPU 4	CPU 5	CPU 6	CPU 7	CPU 8
MAXCPU	1	2	3	4	5	6	---	---
NUMCPU	1	2	3	4	---	---	---	---
ENGINES	CP	CP	CP	AP	IP	IP	CP	CP
CPU type	CP	CP	CP	AP	IP	IP	---	---
Status	O	O	O	O	F	F	---	---

Table 29: Correct CPU configuration (example 3)

Example 4:

```
ENGINES CP,CP,CP,AP,IP,IP
MAXCPU 8
NUMCPU 4
```

The first four engines are defined with their specified type and are taken online because of the NUMCPU system parameter. The next two CPUs are also defined with their specified type, but are taken offline, because of the NUMCPU value. MAXCPU is set to eight CPUs but ENGINES specifies only the type of six of the engines, so the remaining two CPUs are defined per default as type CP and are taken offline as specified in NUMCPU.

CPU and State	CPU 1	CPU 2	CPU 3	CPU 4	CPU 5	CPU 6	CPU 7	CPU 8
MAXCPU	1	2	3	4	5	6	7	8
NUMCPU	1	2	3	4	---	---	---	---
ENGINES	CP	CP	CP	AP	IP	IP	---	---
CPU type	CP	CP	CP	AP	IP	IP	CP(D)	CP(D)
Status	O	O	O	O	F	F	F	F

Table 30: Correct CPU configuration (example 4)

Example 5:

```
ENGINES CP,CP,AP,IP
MAXCPU 6
NUMCPU 6
```

The first four engines are defined with their specified type and are taken online. The next two CPUs are defined per default as type CP, because the number in MAXCPU exceeds the number of CPUs in the ENGINES statement. These additional two CPUs are also taken online through the NUMCPU statement.

CPU and State	**CPU 1**	**CPU 2**	**CPU 3**	**CPU 4**	**CPU 5**	**CPU 6**	**CPU 7**	**CPU 8**
MAXCPU	1	2	3	4	5	6	---	---
NUMCPU	1	2	3	4	5	6	---	---
ENGINES	CP	CP	AP	IP	---	---	---	---
CPU type	CP	CP	AP	IP	CP(D)	CP(D)	---	---
Status	O	O	O	O	O	O	---	---

Table 31: Correct CPU configuration (example 5)

Example 6:

```
ENGINES CP,CP,AP,IP
MAXCPU 6
NUMCPU 8
```

This configuration leads to an error. While the first four CPUs would be taken online with their specified types and the next two CPUs would be of type CP per default and be taken offline, the excess engines from NUMCPU compared against MAXCPU lead to an error message and a failing configuration.

CPU and State	**CPU 1**	**CPU 2**	**CPU 3**	**CPU 4**	**CPU 5**	**CPU 6**	**CPU 7**	**CPU 8**
MAXCPU	1	2	3	4	5	6	---	---
NUMCPU	1	2	3	4	5	6	7	8
ENGINES	CP	CP	AP	IP	---	---	---	---
CPU type	CP	CP	AP	IP	CP(D)	CP(D)	CP(D)	CP(D)
Status	O	O	O	O	O	O	F	F

Table 32: Incorrect CPU configuration (example 6)

Appendix C. Architecture Facilities

The following table shows which architecture facilities are enabled or disabled by default, depending on the specified architecture mode.

Facility	**S/370**	**ESA/390**	**ESAME**	**z/ARCH**
ACC_EX_FS_INDIC	Disabled	Disabled	Disabled	Enabled
ASN_LX_REUSE	Disabled	Enabled	Enabled	Enabled
CMPSC_ENH	Disabled	Disabled	Disabled	Enabled
CONDITIONAL_SSKE	Disabled	Disabled	Enabled	Enabled
CONFIG_TOPOLOGY	Disabled	Disabled	Enabled	Enabled
CSSF	Disabled	Disabled	Enabled	Enabled
CSSF2	Disabled	Disabled	Enabled	Enabled
DECIMAL_FLOAT	Disabled	Disabled	Enabled	Enabled
DETECT_PGMINTLOOP	Enabled	Enabled	Enabled	Enabled
DFP_HPERF	Disabled	Disabled	Enabled	Enabled
ENH_MONITOR	Disabled	Disabled	Disabled	Enabled
ENHANCED_DAT	Disabled	Disabled	Enabled	Enabled
ESAME_ACTIVE	Disabled	Disabled	Enabled	Enabled
ESAME_INSTALLED	Disabled	Disabled	Enabled	Enabled
ETF2_ENHANCEMENT	Disabled	Disabled	Enabled	Enabled
ETF3_ENHANCEMENT	Disabled	Disabled	Enabled	Enabled
EXECUTE_EXTN	Disabled	Disabled	Enabled	Enabled
EXTENDED_IMMED	Disabled	Disabled	Enabled	Enabled
EXTRACT_CPU_TIME	Disabled	Disabled	Enabled	Enabled
FAST_BCR_SERIAL	Disabled	Disabled	Disabled	Enabled
FPS_ENHANCEMENT	Disabled	Disabled	Enabled	Enabled
GEN_INST_EXTN	Disabled	Disabled	Enabled	Enabled
HFP_MULT_ADD_SUB	Disabled	Disabled	Enabled	Enabled

Facility	S/370	ESA/390	ESAME	z/ARCH
HFP_UNNORM_EXT	Disabled	Disabled	Enabled	Enabled
HOST_RESOURCE_ACCESS	Disabled	Disabled	Disabled	Disabled
IDTE_INSTALLED	Disabled	Disabled	Enabled	Enabled
IDTE_SC_REGTAB	Disabled	Disabled	Disabled	Disabled
IDTE_SC_SEGTAB	Disabled	Disabled	Disabled	Disabled
INTERVAL_TIMER	Enabled	Enabled	Enabled	Enabled
IPTE_RANGE	Disabled	Disabled	Disabled	Enabled
LOAD_PROGRAM_PARAM	Disabled	Disabled	Disabled	Enabled
LOGICAL_PARTITION	Enabled	Enabled	Enabled	Enabled
LONG_DISPL_HPERF	Disabled	Disabled	Enabled	Enabled
LONG_DISPL_INST	Disabled	Disabled	Enabled	Enabled
MOVE_INVERSE	Enabled	Enabled	Enabled	Enabled
MSA_EXTENSION_1	Disabled	Disabled	Disabled	Enabled
MSA_EXTENSION_2	Disabled	Disabled	Disabled	Enabled
MSA_EXTENSION_3	Disabled	Disabled	Disabled	Enabled
MSA_EXTENSION_4	Disabled	Disabled	Disabled	Enabled
MSG_SECURITY	Disabled	Disabled	Enabled	Enabled
MVCOS	Disabled	Disabled	Enabled	Enabled
N3	Disabled	Enabled	Enabled	Enabled
PARSING_ENHANCE	Disabled	Disabled	Enabled	Enabled
PROBSTATE_DIAGF08	Disabled	Disabled	Disabled	Disabled
QDIO_ASSIST	Disabled	Disabled	Disabled	Disabled
QDIO_TDD	Disabled	Disabled	Disabled	Disabled
QDIO_THININT	Disabled	Disabled	Disabled	Enabled
QEBSM	Disabled	Disabled	Disabled	Enabled
RES_REF_BITS_MUL	Disabled	Disabled	Disabled	Enabled

Facility	S/370	ESA/390	ESAME	z/ARCH
SENSE_RUN_STATUS	Disabled	Disabled	Enabled	Enabled
SIGP_SETARCH_S370	Disabled	Disabled	Disabled	Disabled
STFL_EXTENDED	Disabled	Disabled	Enabled	Enabled
STORE_CLOCK_FAST	Disabled	Disabled	Enabled	Enabled
SVS	Disabled	Disabled	Disabled	Enabled
TOD_CLOCK_STEER	Disabled	Disabled	Enabled	Enabled
TRAN_FAC_3	Disabled	Disabled	Enabled	Enabled
TRAN_FAC2	Disabled	Disabled	Enabled	Enabled
VIRTUAL_MACHINE	Disabled	Disabled	Disabled	Disabled

Table 33: Architecture Facilities

Appendix D. Hercules Command Groups

The following table shows the affiliation of each console command to the various command groups. The command groups are selected with the CMDLEVEL / CMDLVL system parameter in the configuration file or the console commands of the same name.

Command	**OPER** (0x01)	**MAINT** (0x02)	**PROG** (0x04)	**NONE** (0x08)	**CONFIG** (0x10)	**DEVEL** (0x20)	**DEBUG** (0x40)	**ALL** (0x7F)	**NODIAG8** (0x80)
!message	X	X	X		X	X	X	X	
#	X	X	X	X	X	X	X	X	
$locate							X	X	
$test							X	X	
$zapcmd							X	X	
*	X	X	X	X	X	X	X	X	
.reply	X	X	X		X	X	X	X	
?	X	X	X	X	X	X	X	X	
abs		X	X		X	X	X	X	
aea		X	X		X	X	X	X	
aia		X	X		X	X	X	X	
ar		X	X		X	X	X	X	
archlvl		X	X		X	X	X	X	X
attach	X	X	X		X	X	X	X	
autoinit		X	X		X	X	X	X	
automount		X	X		X	X	X	X	
b		X	X		X	X	X	X	
b+		X	X		X	X	X	X	
b-		X	X		X	X	X	X	
cache					X			X	
cachestats		X	X		X	X	X	X	
capping					X			X	X

Command	**OPER** (0x01)	**MAINT** (0x02)	**PROG** (0x04)	**NONE** (0x08)	**CONFIG** (0x10)	**DEVEL** (0x20)	**DEBUG** (0x40)	**ALL** (0x7F)	**NODIAG8** (0x80)
cckd					X			X	
cd	X	X	X		X	X	X	X	X
cf	X	X	X		X	X	X	X	X
cfall	X	X	X		X	X	X	X	X
clocks		X	X		X	X	X	X	
cmdlevel	X	X	X	X	X	X	X	X	
cmdlvl	X	X	X	X	X	X	X	X	
cmdsep	X	X	X	X	X	X	X	X	
cmdtgt	X	X	X		X	X	X	X	
cnslport					X			X	X
codepage		X	X		X	X	X	X	
conkpalv		X	X		X	X	X	X	
cp_updt		X	X		X	X	X	X	
cpu	X	X	X		X	X	X	X	
cpuidfmt					X			X	X
cpumodel					X			X	X
cpuprio					X			X	X
cpuserial					X			X	X
cpuverid					X			X	
cr		X	X		X	X	X	X	
cscript		X	X		X	X	X	X	
ctc		X	X		X	X	X	X	
define	X	X	X		X	X	X	X	
defstore					X			X	
defsym		X	X		X	X	X	X	
delsym		X	X		X	X	X	X	

Command	**OPER** (0x01)	**MAINT** (0x02)	**PROG** (0x04)	**NONE** (0x08)	**CONFIG** (0x10)	**DEVEL** (0x20)	**DEBUG** (0x40)	**ALL** (0x7F)	**NODIAG8** (0x80)
detach	X	X	X		X	X	X	X	
devinit	X	X	X		X	X	X	X	
devlist	X	X	X		X	X	X	X	
devprio					X			X	X
devtmax					X			X	
diag8cmd					X			X	X
dir	X	X	X		X	X	X	X	X
ds		X	X		X	X	X	X	
ecpsvm		X	X		X	X	X	X	
engines					X			X	X
exec	X	X	X	X	X	X	X	X	
exit	X	X	X	X	X	X	X	X	X
ext	X	X	X		X	X	X	X	X
fcb	X	X	X		X	X	X	X	
fpc		X	X		X	X	X	X	
fpr		X	X		X	X	X	X	
f{+/-} addr		X	X		X	X	X	X	
g		X	X		X	X	X	X	
gpr		X	X		X	X	X	X	
hao			X			X		X	
help	X	X	X	X	X	X	X	X	
herc	X	X	X		X	X	X	X	
herclogo		X	X		X	X	X	X	
hercprio					X			X	X
hst	X	X	X	X	X	X	X	X	
http					X			X	

Command	**OPER** (0x01)	**MAINT** (0x02)	**PROG** (0x04)	**NONE** (0x08)	**CONFIG** (0x10)	**DEVEL** (0x20)	**DEBUG** (0x40)	**ALL** (0x7F)	**NODIAG8** (0x80)
i	X	X	X		X	X	X	X	X
iodelay		X	X		X	X	X	X	
ipending		X	X		X	X	X	X	
ipl	X	X	X		X	X	X	X	X
iplc	X	X	X		X	X	X	X	X
k		X	X		X	X	X	X	
kd	X	X	X		X	X	X	X	
ldmod		X	X		X	X	X	X	
legacysenseid					X			X	
loadcore		X	X		X	X	X	X	
loadparm	X	X	X		X	X	X	X	
loadtext		X	X		X	X	X	X	
log	X	X	X		X	X	X	X	
logopt	X	X	X		X	X	X	X	
lparname					X			X	X
lparnum					X			X	X
ls	X	X	X		X	X	X	X	X
lsdep		X	X		X	X	X	X	
lsmod		X	X		X	X	X	X	
mainsize					X			X	X
manufacturer					X			X	X
maxcpu		X	X		X	X	X	X	
maxrates	X	X	X		X	X	X	X	
memlock					X			X	
message	X	X	X	X	X	X	X	X	
model					X			X	X

Command	**OPER** (0x01)	**MAINT** (0x02)	**PROG** (0x04)	**NONE** (0x08)	**CONFIG** (0x10)	**DEVEL** (0x20)	**DEBUG** (0x40)	**ALL** (0x7F)	**NODIAG8** (0x80)
modpath					X			X	
mounted_tape		X	X		X	X	X	X	
msg	X	X	X	X	X	X	X	X	
msghld	X	X	X		X	X	X	X	
msglevel	X	X	X	X	X	X	X	X	
msglvl	X	X	X	X	X	X	X	X	
msgnoh	X	X	X	X	X	X	X	X	
mt	X	X	X		X	X	X	X	
numcpu		X	X		X	X	X	X	
numvec		X	X		X	X	X	X	
ostailor		X	X		X	X	X	X	
panrate	X	X	X		X	X	X	X	
pantitle	X	X	X		X	X	X	X	
pgmprdos					X			X	X
pgmtrace		X	X		X	X	X	X	
plant					X			X	X
pr		X	X		X	X	X	X	
pscp	X	X	X		X	X	X	X	
psw		X	X		X	X	X	X	
ptp		X	X		X	X	X	X	
ptt		X	X		X	X	X	X	
pwd	X	X	X		X	X	X	X	X
qcpuid	X	X	X		X	X	X	X	
qd		X	X		X	X	X	X	
qpfkeys	X	X	X		X	X	X	X	
qpid	X	X	X		X	X	X	X	

Command	**OPER** (0x01)	**MAINT** (0x02)	**PROG** (0x04)	**NONE** (0x08)	**CONFIG** (0x10)	**DEVEL** (0x20)	**DEBUG** (0x40)	**ALL** (0x7F)	**NODIAG8** (0x80)
qports	X	X	X		X	X	X	X	
qproc	X	X	X		X	X	X	X	
qstor	X	X	X		X	X	X	X	
quiet		X	X		X	X	X	X	
quit	X	X	X	X	X	X	X	X	X
quitmout		X	X		X	X	X	X	
r		X	X		X	X	X	X	
restart	X	X	X		X	X	X	X	X
resume		X	X		X	X	X	X	
rexx					X			X	
rmmod		X	X		X	X	X	X	
s		X	X		X	X	X	X	
s+		X	X		X	X	X	X	
s-		X	X		X	X	X	X	
s?		X	X		X	X	X	X	
savecore		X	X		X	X	X	X	
sclproot					X			X	
scp	X	X	X		X	X	X	X	
scpecho	X	X	X		X	X	X	X	
scpimply	X	X	X		X	X	X	X	
script		X	X		X	X	X	X	
scsimount		X	X		X	X	X	X	
sf+		X	X		X	X	X	X	
sf-		X	X		X	X	X	X	
sfc		X	X		X	X	X	X	
sfd		X	X		X	X	X	X	

Command	**OPER** (0x01)	**MAINT** (0x02)	**PROG** (0x04)	**NONE** (0x08)	**CONFIG** (0x10)	**DEVEL** (0x20)	**DEBUG** (0x40)	**ALL** (0x7F)	**NODIAG8** (0x80)
sfk		X	X		X	X	X	X	
sh		X	X		X	X	X	X	X
shcmdopt					X			X	X
showdvol1	X	X	X		X	X	X	X	
shrd		X	X		X	X	X	X	
shrdport					X			X	X
sizeof		X			X	X	X	X	
srvprio					X			X	X
ssd	X	X	X		X	X	X	X	
start	X	X	X		X	X	X	X	
startall	X	X	X		X	X	X	X	X
stop	X	X	X		X	X	X	X	
stopall	X	X	X		X	X	X	X	X
store	X	X	X		X	X	X	X	
suspend		X	X		X	X	X	X	
symptom		X	X		X	X	X	X	
syncio		X	X		X	X	X	X	
sysclear	X	X	X		X	X	X	X	X
sysepoch					X			X	X
sysreset	X	X	X		X	X	X	X	X
s{+/-} dev		X	X		X	X	X	X	
t		X	X		X	X	X	X	
t+		X	X		X	X	X	X	
t-		X	X		X	X	X	X	
t?		X	X		X	X	X	X	
timerint		X	X		X	X	X	X	

Command	**OPER** (0x01)	**MAINT** (0x02)	**PROG** (0x04)	**NONE** (0x08)	**CONFIG** (0x10)	**DEVEL** (0x20)	**DEBUG** (0x40)	**ALL** (0x7F)	**NODIAG8** (0x80)
tlb		X	X		X	X	X	X	
toddrag		X	X		X	X	X	X	
todprio					X			X	X
traceopt		X	X		X	X	X	X	
tzoffset					X			X	X
tt32		X	X		X	X	X	X	
t{+/-} dev		X	X		X	X	X	X	
t{+/-} CKD		X	X		X	X	X	X	
u		X	X		X	X	X	X	
uptime	X	X	X	X	X	X	X	X	
v		X	X		X	X	X	X	
version	X	X	X	X	X	X	X	X	
xpndsize					X			X	X
yroffset					X			X	X

Table 34: Console commands related to command groups

Appendix E. Build Options for System Parameters and Console Commands

There are a number of system parameters and console commands that depend on the build options set at compile time. They are only available when the corresponding build options have been set when Hercules has been built.

Most of these build option are enabled by default. That means the corresponding system parameters and console commands are available without any further action. A few build options however are disabled by default. If the corresponding system parameters and console commands are required the build options must be set and Hercules must be rebuilt.

Instead of changing the Hercules source files to activate specific build options, the HQA feature (Hercules Build Configurations Quality Assurance) has been introduced. The HQA features allows changing build options by using an environment variable or by using the new configuration or makefile options. For more details about the HQA feature consult the "Installation Guide".

The following table gives an overview about build options affecting system parameters and console commands.

Hercules build option	**Default**	**System parameter / console command**
_FEATURE_ASN_AND_LX_REUSE	X	alrf / asn_and_lx_reuse [1]
_FEATURE_CPU_RECONFIG	X	cf / cfall
_FEATURE_ECPSVM	X	ecpsvm
_FEATURE_SYSTEM_CONSOLE	X	!message / .reply / scpecho / scpimply / ssd
_FW_REF	X	f{+/-}addr / s{+/-}dev / sf-dev / sf+dev / sfc / sfd / sfk / t{+/-}dev
_HAVE_MLOCKALL		memlock / memfree
MSVC		dir [2]
ENABLE_OBJECT_REXX	X	rexx / exec
ENABLE_REGINA_REXX	X	rexx / exec
OPTION_CAPPING	X	capping
OPTION_CKD_KEY_TRACING	X	t{+/-}ckd
OPTION_CMDTGT	X	cmdtgt / herc / pscp / scp
OPTION_CONFIG_SYMBOLS	X	qpfkeys / defsym / delsym
OPTION_DYNAMIC_LOAD	X	modpath / ldmod / lsdep / lsmod / rmmod
OPTION_HAO	X	hao
OPTION_HTTP_SERVER	X	http / httpport / httproot [3]

OPTION_INSTRUCTION_COUNTING		icount
OPTION_IODELAY_KLUDGE	X	iodelay
OPTION_LPP_RESTRICT	X	pgmprdos
OPTION_MIPS_COUNTING	X	maxrates
OPTION_MSGHLD		kd / msghld
OPTION_PTTRACE	X	ptt
OPTION_SCSI_TAPE	X	auto_scsi_mount / scsimount [4]
OPTION_SET_STSI_INFO	X	manufacturer / model / plant
OPTION_SHARED_DEVICES	X	shrdport
OPTION_SHUTDOWN_CONFIRMATION		quitmout
OPTION_TAPE_AUTOMOUNT	X	automount
OPTION_W32_CTCI	X	tt32
OPTION_SHOWDVOL1		showdvol1
PANEL_REFRESH_RATE	X	panrate

Table 35: Build options for system parameters and console commands

Notes:

[1] "alrf" and "asn_and_lx_reuse" have been deprecated and replaced by "archlevel enable asn_lx_reuse".

[2] "_MSVC_" is a build option that is automatically set when Hercules is built on a Windows platform. On Linux and Mac platforms the option is not set and the "ls" console command is available instead.

[3] "httpport" and "httproot" have been deprecated and replaced by "http".

[4] "auto_scsi_mount" has been deprecated and replaced by "scsimount".

Appendix F. Environment Variables

The following table lists the Hercules environment variables. The table is sorted by the usage (configuration, build, ...) and the name of the environment variable.

Usage	Environment Variable	Description	Platform
Hercules Build	BZIP2_DIR	Path to the BZIP2 package	Windows
Hercules Build	PCRE_DIR	Path to the PCRE package	Windows
Hercules Build	ZLIB_DIR	Path to the ZLIB package	Windows
Hercules Configuration	HERCULES_CNF	Default name of configuration file	All
Hercules Configuration	HERCULES_CP	Hercules Codepage	All
Hercules Configuration	HERCULES_RC	Default name of run-commands file	All
Hercules Configuration	HERCULES_LIB	Dynamic load modules path	All
Hercules Configuration	HREXX_PACKAGE	Name of REXX package	All
Hercules Configuration	LD_LIBRARY_PATH	Dynamic load modules path	All
Hercules Configuration	REXX_DIR	Path to the REXX package	All

Table 36: Environment Variables

Appendix G. Syntax

This book uses two kinds of describing the syntax of configuration statements, console commands and utilities. These are:

- Syntax descriptions
- Syntax diagrams

G.1 Reading Syntax Descriptions

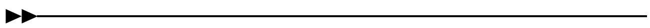
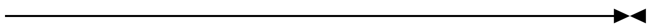
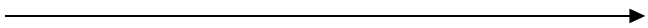
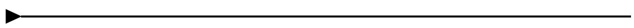
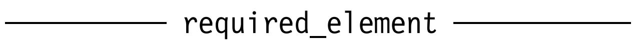
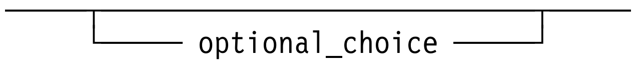
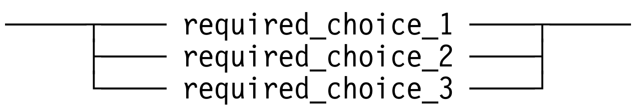
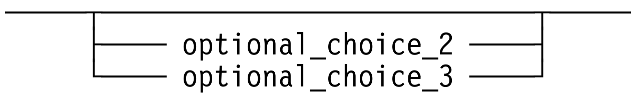
All syntax descriptions in this book (configuration statements, console commands and utilites) use a common structure as described in the following table.

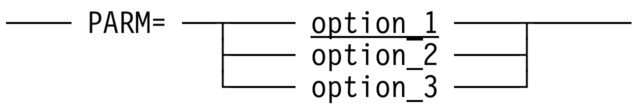
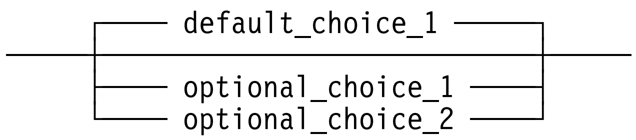
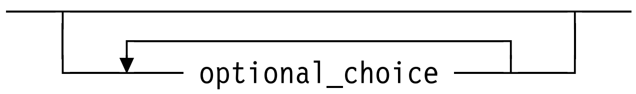
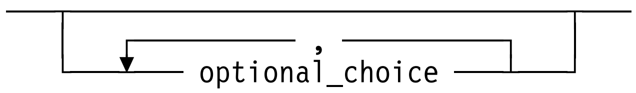
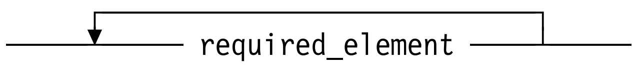
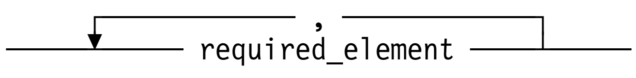
Syntax Element	**Description**
`KEYWORDS`	Keywords are denoted with upper case letters. Obey the spelling. In the actual statements or commands they can be coded in upper case or lower case letters.
`variables`	All user defined values are denoted with lower case italic letters. In the actual statements or commands they can be coded in upper case or lower case letters.
`{ }`	Signifies that all or some portion of the code elements between the braces are required elements. Note that the braces are not part of the statements and must be not coded.
`[ ]`	Signifies that all or some portion of the code elements between the square brackets can optionally appear but are not required elements. Note that the square brackets are not part of the statements and must be not coded.
`|`	The OR symbol signifies that you may use only one of the code elements or values from the possible choices. Note that the OR symbol is not part of the statements and must be not coded.
`xxx,...`	Signifies that there can be more than one value in a comma delimited list. Note that the dots are not part of the statements and must be not coded.
`xxx ...`	Signifies that there can be more than one value in a blank space delimitted list. Note that the dots are not part of the statements and must be not coded.

Table 37: Reading Syntax Descriptions

G.2 Reading Syntax Diagrams

All syntax diagrams in this book (configuration statements, console commands and utilities) use a common structure as described in the following table.

Symbol	Description
	This symbol indicates the beginning of a syntax diagram.
	This symbol indicates the end of a syntax diagram.
	This symbol indicates that the syntax diagram is continued on the next line.
	This symbol indicates that the syntax diagram is a continuation from the previous line.
	A required element (keyword or variable) appears on the main path of the horizontal line. You must specify this element.
	An optional element (keyword or variable) appears below the main path of the horizontal line. You may or may not specify this element.
	A required choice (keyword or variable) appears vertically stacked in the main path of the horizontal line. You must choose one of the available options.
	An optional choice (keyword or variable) appears vertically stacked below the main path of the horizontal line. You may choose one of the available options.

Symbol	Description
	A keyword with options. Only one of the available options may be specified. The underscored option is the default if the whole keyword statement is not coded.
	An optional choice (keyword or variable) with default appears vertically stacked with the default value above the main path of the horizontal line and the remaining optional elements below the main path of the horizontal line. Only one of the available options may be specified. If none of these elements is explicitly specified, the default above the main line is taken.
 	An arrow returning to the left of an element below the main path of the horizontal line indicates an optional repeatable item. A character within the arrow path means that repeated items have to be separated by that character. If there is no character within the arrow path then the items are separated by a blank.
 	An arrow returning to the left of an element on the main path of the horizontal line indicates an required repeatable item. A character within the arrow path means that repeated items have to be separated by that character. If there is no character within the arrow path then the items are separated by a blank.
	This symbol is a reference to a syntax segment, which is described separately below the main syntax diagram. Complex syntax diagrams are occasionally broken into separated simpler segments.

Symbol	Description
├── SEGMENT= ──┬── value_1 ──┬──┤ └── value_2 ──┘	This symbol indicates a syntax segment which is referenced from a main syntax diagram that is shown above the syntax segment.
KEYWORDS	Keywords are denoted with upper case letters. Obey the spelling. Lower case letters are optional and can be omitted (for example DISable). In the actual statements or commands the keywords can be coded in upper case or lower case letters.
variables	All user defined values are denoted with lower case italic letters. They represent user supplied names or values. In the actual statements or commands they can be coded in upper case or lower case letters.

Table 38: Reading Syntax Diagrams

G.3 Sample Syntax Description

The following figure shows a sample of a complex syntax description. This is not an example of an existing system parameter or panel command. It is used mainly to demonstrate the “look and feel” of syntax descriptions.

Syntax

CMDNAME *required_argument* [*optional_argument*]

{*required_choice_1* | *required_choice_2* | *required_choice_3*}

[*optional_choice_1* | *optional_choice_2*]

REQUIRED_KEYWORD=*variable*

[OPTIONAL_KEYWORD=*variable*]

[DEFAULT_KEYWORD | KEYWORD_1 | KEYWORD_2]

[KEYWORD=*default_choice* | KEYWORD={*choice_1* | *choice_2*}]

repeatable_item_n {*repeatable_item_n* ...}

repeatable_item_n {,*repeatable_item_n* ,...}

fragment_name

(*variable_1*, *variable_2*)

fragment_name

[<u>DEFAULT_KEYWORD</u> | KEYWORD_1 | KEYWORD_2 | KEYWORD_3 | KEYWORD_4]

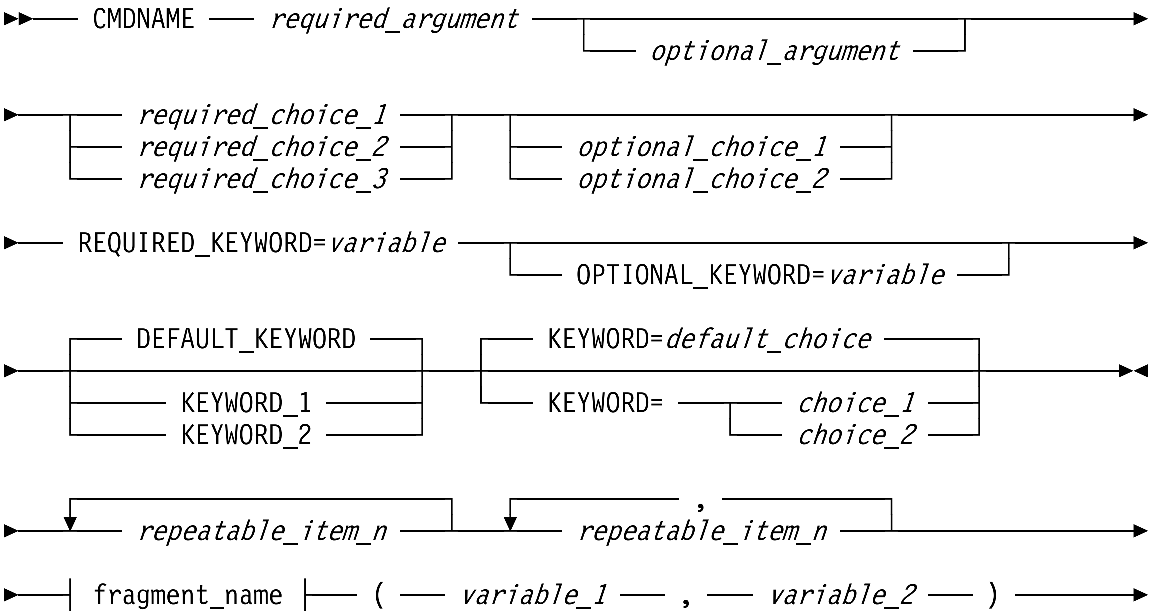
KEYWORD={*choice_1* | *choice_2* | *choice_3* | *choice_4*}

Figure 357: Sample Syntax Description

G.4 Sample Syntax Diagram

The next figure shows a sample of a complex syntax diagram. It shows the same example as in the syntax description in the previous section. Like in the example before it is not based on an existing system parameter or panel command. It is used mainly to demonstrate the “look and feel” of syntax descriptions.

Syntax



fragment_name

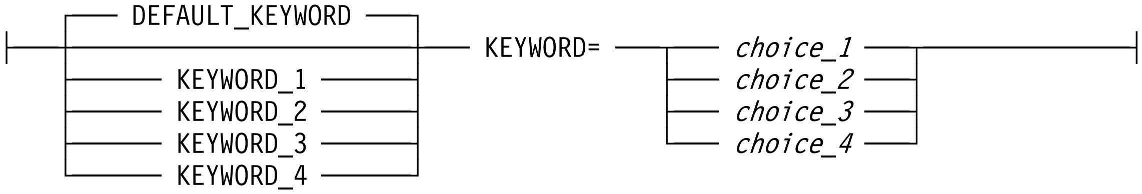


Figure 358: Sample Syntax Diagram

Appendix H. Links

- **The Hercules System/370, ESA/390, and z/Architecture Emulator**

  http://www.hercules-390.eu

- **Hercules source code repositories**

  https://github.com/rbowler/spinhawk (release 3.xx development stream)

  https://github.com/rbowler/sandhawk (release 4.xx development stream)

  https://github.com/hercules-390/hyperion (cutting-edge developer sandbox)

- **Hercules Developer Snapshots (Dave Wade)**

  http://www.smrcc.org.uk/members/g4ugm/snapshots/

- **Hercules PDF Documentation (Peter Glanzmann)**

  http://hercdoc.glanzmann.org

- **The MVS Tur(n)key System, Version 3 (Volker Bandke)**

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

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

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

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

  http://www.softdevlabs.com/Hercules/CTCI-WIN-index.html

- **Hercules Studio (Jacob Dekel)**

  http://www.mvsdasd.org/hercstudio

- **Hebe – Hercules Image Manager (Robin Atwood)**

  http://kde-apps.org/content/show.php/Hebe?content=126738

- **WinPcap, Politecnico di Torino**

  http://www.winpcap.org

- **Vista tn3270, Tom Brennan Software**

  http://www.tombrennansoftware.com

- **X3270, Paul Mattes**

  http://x3270.bgp.nu

- **AWSBROWSE ("Fish", David B. Trout)**

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

- **XMIT Manager**

  www.cbttape.org

- **CBT MVS Utilities Tape (CBTTAPE)**

  www.cbttape.org

- **Microsoft Visual C++ 2008 Express**

  http://www.microsoft.com/express/download/

- **ZLIB**

  http://www.zlib.net

  http://www.softdevlabs.com/Hercules/ZLIB1-1.2.3-bin-lib-inc-vc2008-x86-x64.zip

- **BZIP2**

  http://www.bzip.org

  http://www.softdevlabs.com/Hercules/BZIP2-1.0.5-bin-lib-inc-vc2008-x86-x64.zip

- **PCRE**

  http://www.pcre.org

  http://www.softdevlabs.com/Hercules/PCRE-6.4.1-bin-lib-inc-vc2008-x86-x64.zip

- **Regina REXX**

  http://regina-rexx.sourceforge.net/

- **Open Object Rexx (ooRexx)**

  http://www.oorexx.org/

Hercules Emulator

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

User Reference Guide

Version 4 Release 00

HEUR040000-00