Tuxera ROM-DOS™ user’s guide

Copyright © 1999-2005 by Tuxera US Inc.  
Portions copyright © GpvNO 2005

All Rights Reserved.

Tuxera US Inc. assumes no liability for the use or misuse of this software. Liability for any warranties implied or stated is limited to the original purchaser only and to the recording medium (disk) only, not the information encoded on it.

U.S. Government Restricted Rights. Use, duplication, reproduction, or transfer of this commercial product and accompanying documentation is restricted in accordance with FAR 12.212 and DFARS 227.7202 and by a license agreement.

THE SOFTWARE DESCRIBED HEREIN, TOGETHER WITH THIS DOCUMENT, ARE FURNISHED UNDER A SEPARATE SOFTWARE OEM LICENSE AGREEMENT AND MAY BE USED OR COPIED ONLY IN ACCORDANCE WITH THE TERMS AND CONDITIONS OF THAT AGREEMENT.

Tuxera and ROM-DOS™ are registered trademarks of Tuxera US Inc. FlashFX® is a trademark of Tuxera US Inc.

All other product names are trademarks of their respective holders.

Global Headquarters  
Westendintie 1  
02160 Espoo, Finland

Tuxera USA  
22118 20th Avenue SE, STE 135  
Bothell, WA 98021 USA

info@tuxera.com  
sales@tuxera.com  
www.tuxera.com
## CONTENTS

### CHAPTER 1 | ROM-DOS INTRODUCTION ................................................................. 7
- About ROM-DOS ........................................................................................................ 7
- Conventions used in this manual ............................................................................ 7
- Terminology used in this manual ........................................................................... 7
- Disks and disk drives .............................................................................................. 8
- Recommended Texts .................................................................................................. 9

### CHAPTER 2 | ROM-DOS INSTALLATION ............................................................... 10
- Installing and Running ROM-DOS ........................................................................... 10
  - Development system installation procedure ......................................................... 10
  - Installed files .......................................................................................................... 10
  - ROM-DOS Considerations ....................................................................................... 11

### CHAPTER 3 | ROM-DOS BASICS ......................................................................... 13
- Files, Directories, and Command Line Entries ....................................................... 13
  - Naming Files ........................................................................................................... 13
  - Tree-Structured Directory System ....................................................................... 14
  - Using Wildcard Characters without Long Filename Support ............................... 15
  - Using Wildcard Characters with Long Filename Support ...................................... 16
  - System Prompt ......................................................................................................... 17
  - Command Line ........................................................................................................ 17
  - Redirecting Input and Output .................................................................................. 18
- Using Batch Files ....................................................................................................... 19
  - Batch Filenames ...................................................................................................... 19
  - Creating a Batch File .............................................................................................. 19
  - Batch File Command Line Parameters .................................................................... 19
  - Batch File Commands ............................................................................................ 20
- ROM-DOS Command Summary ................................................................................ 21

### CHAPTER 4 | CONFIGURATION ........................................................................... 25
- Basic Configuration .................................................................................................... 25
- Using Multiple-User Configurations ........................................................................ 25
- Extending Menu Items to AUTOEXEC.BAT ............................................................. 27
- Bypassing CONFIG.SYS and AUTOEXEC.BAT Commands ................................. 28
- Stepping Through CONFIG.SYS and AUTOEXEC.BAT Commands ..................... 28
- Environment Variables ............................................................................................. 29
- Configuring ROM-DOS for International Use ........................................................ 29
  - Changing Country Conventions .......................................................................... 29
  - Displaying Different Code Pages ......................................................................... 31
  - Printing Different Code Pages .............................................................................. 31
  - Changing the Keyboard Layout ............................................................................. 31
  - Configuring Your System: an Example ................................................................. 33
CHAPTER 5 | ROM-DOS INTERNAL COMMANDS ................................................................. 34
  Internal Command Descriptions ........................................................................... 34

CHAPTER 6 | ROM-DOS UTILITY DESCRIPTIONS ......................................................... 73
  ROM-DOS Utilities ................................................................................................. 73
    Command Descriptions ......................................................................................... 73
  Mini-Command Interpreter .................................................................................... 122
    External Commands .............................................................................................. 122
    Internal Commands .............................................................................................. 122
    Configuring the Mini-Command Interpreter ......................................................... 123

CHAPTER 7 | ROM-DOS SERVER AND CLIENT APPLICATIONS ..................................... 125
  Serial File Server .................................................................................................. 125
    Server Program .................................................................................................... 125
    Client Program .................................................................................................... 126
  Remote Disk Program ............................................................................................. 127
    Server Program .................................................................................................... 127
    Client Program .................................................................................................... 127
    Using the Remote Disk .......................................................................................... 128
    Unloading the Server Remotely ............................................................................ 129

CHAPTER 8 | ROM-DOS KEYBOARD LAYOUTS ............................................................. 130
  Keyboard Layouts .................................................................................................. 130

ROM-DOS GLOSSARY ............................................................................................. 134

CHAPTER 9 | SOCKETS INTRODUCTION ..................................................................... 137
  About SOCKETS ..................................................................................................... 137
    System Requirements ........................................................................................... 137

CHAPTER 10 | SOCKETS INSTALLATION ................................................................. 139
  Installing and Running SOCKETS ........................................................................ 139
    Development System Procedure ........................................................................ 139
    Environment Variables ......................................................................................... 140
    File Selection ....................................................................................................... 141
    Building a custom SOCKETS kernel .................................................................... 141
    Configuration ........................................................................................................ 141
    Transfer ................................................................................................................ 141
    Testing .................................................................................................................. 141
    Custom Applications ............................................................................................. 142

CHAPTER 11 | SOCKETS CONFIGURATION ............................................................... 143
  Packet Driver ......................................................................................................... 143
  Serial Operation .................................................................................................... 143
Hardware considerations ................................................................. 143
PPP Functionality ........................................................................... 144
Modem Operation ............................................................................ 144
SOCKETS Configuration Files: SOCKET.CFG, HOSTS .................. 145
Configuring for IPv6 operation ....................................................... 145
SOCKET.CFG Samples ................................................................... 146
SOCKETP, SOCKETM and SOCKETS Overview ....................... 147
PING ............................................................................................... 147

CHAPTER 12 | SOCKETS CONFIGURATION COMMANDS ................................. 150
Commands ....................................................................................... 150
  Overview ..................................................................................... 150
  Notations and Conventions .......................................................... 150
  Command Reference .................................................................... 150
SOCKETS command line options .................................................... 174
Modem Configuration File Syntax ................................................... 175
  Delayed LAN configuration .......................................................... 180
  Multi Destination Drivers .............................................................. 180
Configuration Considerations .......................................................... 183

CHAPTER 13 | SOCKETS CONFIGURATION EXAMPLES ................................. 184
Example 1: Ethernet Connection – SOCKETS Serving a Web Page ................ 184
Example 2: Serial Connection – SOCKETS Dial-up to ISP .................. 185
Example 3: Single Dial-in Connection .............................................. 186
Example 4: Single Dial-in Connection with ASY Interface ............... 187
Example 5: Direct Serial Connection with SOCKETS as a Server .... 187
Example 6: Direct Serial Connection with SOCKETS as a Client ....... 190
Example 7: SOCKETS Machine Using Call Back Verification ........... 191
Example 8: SOCKETS Machine with CBV and Logging-in .............. 193
Example 9: Dial-up SLIP Connection with SOCKETS as an IP Router . 195
Example 10: Multiple Dial-in Connections ...................................... 196

CHAPTER 14 | SOCKETS UTILITY DESCRIPTIONS .......................................... 199
SOCKETS Utilities .......................................................................... 199

CHAPTER 15 | SOCKETS SERVER AND CLIENT APPLICATIONS ....................... 227
HTTP Server ................................................................................... 227
  Overview ..................................................................................... 227
  Server ......................................................................................... 227
  Remote Console Server ............................................................... 228
  Extension CGI .............................................................................. 229
  Passive Mode .............................................................................. 230
  Server Memory .......................................................................... 230
  Spawning CGI ............................................................................ 230
  Authentication ............................................................................ 230
<table>
<thead>
<tr>
<th>Link</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTPD Program</td>
<td>231</td>
</tr>
<tr>
<td>Format of “SOCKET.UPW”</td>
<td>233</td>
</tr>
<tr>
<td>Format of “htaccess”</td>
<td>234</td>
</tr>
<tr>
<td>FTP Server</td>
<td>235</td>
</tr>
<tr>
<td>FTPD Program</td>
<td>235</td>
</tr>
<tr>
<td>FTP Server Commands</td>
<td>236</td>
</tr>
<tr>
<td>Combined HTTP, RC and FTP Server</td>
<td>238</td>
</tr>
<tr>
<td>HTTPFTPD Program</td>
<td>238</td>
</tr>
<tr>
<td>Combined HTTP, FTP, RC and Echo Server</td>
<td>238</td>
</tr>
<tr>
<td>**APPENDIX A</td>
<td>PACKET DRIVERS**</td>
</tr>
<tr>
<td>Overview</td>
<td>239</td>
</tr>
<tr>
<td>Packet Driver Installation</td>
<td>239</td>
</tr>
<tr>
<td>Using a Memory Manager with a Packet Driver</td>
<td>240</td>
</tr>
<tr>
<td>Packet Driver over ODI Driver Installation</td>
<td>240</td>
</tr>
<tr>
<td>Using a Memory Manager with an ODI Driver</td>
<td>241</td>
</tr>
<tr>
<td>Packet Driver over NDIS2 Driver Installation</td>
<td>241</td>
</tr>
<tr>
<td>Using a Memory Manager with an NDIS Driver</td>
<td>243</td>
</tr>
<tr>
<td>**APPENDIX B</td>
<td>NETWORK MANAGEMENT AND TROUBLESHOOTING**</td>
</tr>
<tr>
<td>Network Management</td>
<td>244</td>
</tr>
<tr>
<td>Configuration Case Studies</td>
<td>244</td>
</tr>
<tr>
<td>System Timer Interrupt Use</td>
<td>245</td>
</tr>
<tr>
<td>Advanced Network Configuration</td>
<td>246</td>
</tr>
<tr>
<td>Tuning TCP/IP</td>
<td>246</td>
</tr>
<tr>
<td>TCP Retry Strategy</td>
<td>246</td>
</tr>
<tr>
<td>Keep-alive</td>
<td>247</td>
</tr>
<tr>
<td>Troubleshooting</td>
<td>247</td>
</tr>
<tr>
<td>Problems with LICENSE.DAT File</td>
<td>247</td>
</tr>
<tr>
<td>XPING</td>
<td>247</td>
</tr>
<tr>
<td>Utility Programs</td>
<td>248</td>
</tr>
<tr>
<td>PTEST, Packet Driver Test Utility</td>
<td>248</td>
</tr>
<tr>
<td>SETHOST, IP Address Maintenance Utility</td>
<td>248</td>
</tr>
<tr>
<td>ISTAT, IP and Memory Statistics Utility</td>
<td>248</td>
</tr>
<tr>
<td>**SOCKETS</td>
<td>GLOSSARY**</td>
</tr>
<tr>
<td>INDEX</td>
<td>255</td>
</tr>
</tbody>
</table>
CHAPTER 1 | ROM-DOS INTRODUCTION

ABOUT ROM-DOS
ROM-DOS is a disk operating system that can be loaded in Read Only Memory (ROM) and can run entirely from within ROM and also from a hard or floppy disk, such as in a desktop system. ROM-DOS is functionally equivalent to other brands of DOS and can run programs that are executable under a standard DOS (which executes from RAM).

Conventions used in this manual
This manual uses several notation conventions to denote specific actions and types of information.

- Key combinations, where two or more keys must be pressed simultaneously, are shown with a plus sign. For example, Shift+F1.

- Command line entries and displayed messages are shown in this font. For example,
  
  `DEL MYLETTER.DOC`

- Command line entries indicated by italicized lowercase letters represent information that you supply. For example,
  
  `DEL filename`
  
  indicates a need to enter the name of the file to be deleted.

- Command line entries enclosed within square brackets represent optional information. For example,
  
  `ECHO [message]`
  
  indicates the message portion may be omitted from the ECHO command.

- Command line entries that include mutually-exclusive options separate those options with a vertical bar (|). In the following example, anything more than the BREAK command must be either the ON or OFF.
  
  `BREAK [ON|OFF]`

- You must press the Enter key for ROM-DOS to accept your command line data. The command line entries shown in this manual do not show the Enter key.

Terminology used in this manual
Computer files are stored in the computer’s internal memory (RAM and ROM) or on magnetic media, typically disks. Regardless of where the files reside, all information is stored in bytes. A byte can store a single character of data. The following terms describe the size of memory, files, and disk space:
KB (kilobyte) – One kilobyte equals 1024 bytes, although the number is often rounded to one thousand bytes.

MB (megabyte) – One megabyte equals 1,048,576 bytes but is usually thought of as one million bytes.

GB (gigabyte) – One gigabyte equals 1,073,741,824 bytes but usually is thought of as one billion bytes.

TB (terabyte) – One terabyte equals 1,099,511,627,776 bytes but usually is thought of as one trillion bytes.

Random Access Memory (RAM)
RAM can be written to, read from, erased, and rewritten. RAM is your computer’s electronic workspace during operation. RAM is also called volatile memory. Its’ storage ability is temporary, only holding information while the power is on. When the power is turned off or interrupted, everything stored in RAM is lost.

Within limits, you can change the amount of RAM in a system. Typically, embedded systems have 512KB, 640KB, or 1MB of RAM. Greater amounts of RAM are also possible. In desktop systems, 16MB to 128MB are common.

Read Only Memory (ROM)
ROM is more permanent than RAM. Data is programmed into a ROM device before the device is installed in the computer. Information stored in ROM remains intact whether the system power is on or off.

Disks and disk drives
Computer disks are classified into two basic groups; rotating media such as floppy disks and hard disks, and memory disks such as those formatted in RAM, ROM and flash memory.

Floppy Disks
A floppy disk is a disk-shaped piece of magnetic material much like audio recording tape. The information is stored in concentric tracks that are subdivided into sectors. Typical storage space on the 3.5-inch disk is 1.44MB.

Hard Drives
Hard drives work similar to floppy disks but are fixed in the computer chassis and have a much higher storage capacity. ROM-DOS 6.22 is capable of utilizing hard drives of up to 8GB and ROM-DOS 7.1 is capable of utilizing hard drives of up to 2TB.
RAM Disks
Portions of RAM can be made to behave like disk drives, complete with tree-structured directories. When a disk drive is created in RAM, it can be read from and written to in the same manner as the physical disk media. However, when system power is interrupted, all information on the drive is lost, unless the drive is formatted on static RAM.

ROM Disks
Portions of ROM can be made to behave like disk drives, complete with tree-structured directories. ROM drives differ from RAM drives in that they are written to only once. Thereafter, they can only be read from, much like a write-protected floppy disk. In addition, ROM disks are non-volatile, in that the information is not lost when power is lost.

Flash Memory Disks
Both PCMCIA cards (PC cards) and on-board (resident) flash arrays can serve as disk drives when used with flash file system software such as Datalight’s FlashFX for on-board flash memory. Unlike ROM disks, flash memory disks support both the reading and writing of data.

RECOMMENDED TEXTS
The following are all excellent texts on the topic of DOS and its internal structures. The knowledge in them applies to Datalight’s ROM-DOS as well as most other DOS compatibles. These all come highly recommended by Datalight, Inc.

1. Microsoft MS-DOS Programmer’s Reference, Microsoft Press, 1993
   ISBN 1-55615-546-8

   ISBN 1-56529-150-6

3. DOS Internals by Geoff Chappell, Addison-Wesley, 1994
   ISBN 0-201-60835-9

   ISBN 0-201-63287-X

   ISBN 0-201-62485-0
CHAPTER 2 | ROM-DOS INSTALLATION

INSTALLING AND RUNNING ROM-DOS
This chapter provides an overview of the process for installing ROM-DOS on a development system and running on a target platform. To create a version of ROM-DOS that runs on any system, follow these steps (described in detail in subsequent chapters of this manual).

- Select the hardware platform.
- Select the BIOS.
- BUILD (configure) ROM-DOS for that system.
- Create a ROM disk (required only for diskless systems).
- Program PROMs and install them (diskless systems) or sys the boot drive, and reboot.

Development system installation procedure
The development system is used to install the ROM-DOS files from the distribution CD-ROM and then configure and build ROM-DOS to your target system specifications as described in the following chapters.

To install the ROM-DOS SDK software on your Windows development system:

1. Place the CD-ROM in CD drive.
2. Choose Start/Run and type setup
3. Follow the onscreen instructions presented by the setup program to install the required components.

We strongly recommend you install all Datalight software under its own directory branch. We suggest you accept the default branch name (DL), which corresponds with the examples in this manual.

You should also install the enclosed Datalight Software Developer’s Tool Kit (SDTK) in the same manner. When the installation is complete, modify your path statement to include the C:\DL\DEVTOOLS\BIN directory. This path allows the ROM-DOS BUILD utility to locate all necessary tools.

Installed files
When installation of the ROM-DOS SDK and the SDTK software is complete, ROM-DOS and the Tool Kit software exist on your development system in the following directories/folders.
ROM-DOS Considerations

ROM-DOS runs on most x86 hardware platforms, including the 186, 286, 386, 486 or Pentium, as well as the NEC V-series and the growing number of work-alike processors. Companies such as NEC and Vadem offer all the parts of a PC in a single piece of silicon. ROM-DOS contains no hardware dependencies.

Selecting the BIOS to use depends on the availability of a BIOS for the hardware and the ROM space requirements for the BIOS. When selecting the BIOS, consider not only the availability, but the features of the BIOS. The BIOS should recognize the basic hardware you intend to have available in your system. It is generally the responsibility of the BIOS to support PCMCIA cards (at least SRAM cards) through the BIOS disk Int 13h. The BIOS should also support extended memory on those CPUs that can address RAM beyond 1MB.

Building (or more appropriately, customizing) ROM-DOS for any given system requires a minimum of effort. Building ROM-DOS can be as simple as answering a few prompts that the BUILD program displays and then placing ROM-DOS on disk or in ROM. Chapter 4 is devoted to running the BUILD program.

When building ROM-DOS, consider such issues as which drive letters you want associated with the disk drives. Will drives A: and B: be assigned to the floppy drives, as ordered on a desktop system?
Or will the ROM disk be drive A:, a PCMCIA disk drive B:, followed by floppy and hard disks? ROM-DOS makes no restrictions in this area, so choose what best suits your system. You may need to adjust the file SYSGEN.ASM described under “Ordering Floppy and Hard Disk Drives” in the ROM-DOS Developer’s Guide to order the drive letters.

If a ROM disk is needed, either for booting, or as another disk on the ROM-DOS system, use the ROMDISK.EXE program to create this disk. A ROM disk driver that searches memory for the ROM disk is built into ROM-DOS.

If ROM-DOS will be placed in ROM, the final step is to program the ROMs and place them in the target machine. At this point, the system can be booted using ROM-DOS.

Systems running ROM-DOS directly from a hard or floppy disk do not need to program any ROMs. Simply run FORMAT and/or SYS on the disk and reboot.
FILES, DIRECTORIES, AND COMMAND LINE ENTRIES

Naming Files
A file is a defined set of related information that your computer stores electronically. A file may be stored on a floppy disk (also called a floppy), on a hard drive, on a CD, or may reside in computer memory (RAM or ROM). To maintain control of interaction between various computer files, each must have its own name that both you and the computer can recognize.

Long Filenames
Files used in the ROM-DOS environment, with long filename support enabled, can have up to 260 characters including the file extension and path. Each long filename also has a standard 8.3 filename associated with it. Long filenames can use nearly any character except the following symbols:

Output redirection > Question mark ?
Input redirection < Asterisk *
Backslash \ Double quotes "
Forward slash / Pipe |

When using a long filename or path that contains spaces, the name and/or path must be surrounded by double quotes. For example:

    copy "c:\my directory\my file.doc" "c:\another directory\reference"

Standard 8.3 Filenames
Files used in the ROM-DOS environment, without long filenames support enabled, have two-part names separated by a period. The first part is the filename; the second part is the filename extension. For example, the command interpreter file provided with ROM-DOS is named COMMAND.COM, where COMMAND is the filename and .COM is the filename extension.

Filenames range from one to eight characters in length and consist of any combination of letters, numbers, and the following symbols:

underscore _ Parenthesis {}
Caret ^ exclamation point !
dollar sign $ number sign #
Tilde ~ percent sign %
Ampersand & At sign @
Hyphen - Apostrophe '
Braces {}
Filename Extensions
The second part of the filename is the filename extension. The filename extension has one, two, or three characters and may use the same symbols as the filename. A filename extension is not required, although filename extensions can be helpful in identifying the type of file. Commonly used filename extensions include .DOC for documents, .DAT for data, and .TXT for text files.

You may use any filename extension you choose. However, certain filename extensions have a special meaning to ROM-DOS and should only be used when appropriate. These include:

- .COM – used for executable files
- .EXE – used for executable files
- .BAT – used for batch files

Some application programs, such as word processors, may use or require particular filename extensions for output or input files. It is best to follow the application instructions regarding proper filename extensions for that particular program. For example, a file named LETTER1 may include a filename extension such as LETTER1.DOC or LETTER1.TXT.

It is possible to have several files with the same filename but different extensions. ROM-DOS searches for and accesses the filename extensions in the following order: .COM, .EXE, .BAT, and then all others. For example, you could have an executable file named MYPROG.EXE and a batch file called MYPROG.BAT in the current directory. When you enter MYPROG on the command line, the file MYPROG.EXE is executed. If you want to execute the batch file MYPROG.BAT, you must specify the .BAT extension when entering MYPROG on the command line.

Tree-Structured Directory System
ROM-DOS uses a tree-structured directory system. In this system, each branch of the directory, called a subdirectory, is either attached to the main or root directory or is attached to another subdirectory. The following diagram illustrates the directory system and shows two levels of subdirectories under the root directory.
Naming Subdirectories
You can create any subdirectory structure you choose, giving each subdirectory the name of your choice. The naming of subdirectories is similar to the naming of files. There is an eight-character limit when long filename support is disabled, and you have the same character-choice limitations as for filenames (letters, numbers, and symbols). A subdirectory name can also have an extension. For more information on creating and deleting subdirectories, refer to the MD and RD descriptions later in this manual for more information on creating and removing directories.

Moving around the Directory Tree
When the computer is first turned on, ROM-DOS boots into the root directory. From the root, you can change to any other directory by means of the CD (CHDIR) command. At any given time, ROM-DOS considers you to be in a specific directory, referred to as the current directory. You can make the computer automatically move to a different directory upon system startup by adding the CD command to your AUTOEXEC.BAT file. Refer to the description of the CD command later in this manual for more information on changing the current directory.

Drive Specifications
Since ROM-DOS can store and retrieve information from more than one disk drive, disk drives are given unique names such as A:, B:, C:, and so on. By convention, floppy disk drives are identified as drive A: and drive B:. On systems having only one physical floppy drive, ROM-DOS can treat the one drive as either A: or B:.

The hard drive, if your system has one, is identified as drive C:. Hard drives can be partitioned (divided) into smaller sections with the FDISK utility. Under DOS 6.22, disks exceeding 8GB must be partitioned into two or more areas, with a maximum size of 2GB per partition but will never use more than the first 8GB of the drive. DOS 7.1 does not have these restrictions. A separate drive letter identifies each hard-drive partition. The first partition is drive C:, the next drive D:, and so on. The highest available drive identifier is the letter Z.

To refer to the C: drive, enter the following on the command line

```
c:
```

Note: The drive name may be entered in either uppercase or lowercase.

Using Wildcard Characters without Long Filename Support
To simplify a task performed on a large group of similarly named files, use wildcard characters. Wildcard characters allow you to reference groups of files without entering the complete filename for each file in the group. A wildcard character can substitute all or part of a filename or extension. The two wildcard characters are the asterisk (*) and the question mark (?). The asterisk represents an entire name or a group of characters found within a name beyond the place where the asterisk is in the search mask. The question mark represents a single character. The following table lists some examples of wildcard character usage.
Using Wildcard Characters with Long Filename Support

To simplify a task performed on a large group of similarly named files, use wildcard characters. Wildcard characters allow you to reference groups of files without entering the complete filename for each file in the group. A wildcard character can substitute all or part of a filename or extension. The two wildcard characters are the asterisk (*) and the question mark (?). The asterisk represents an entire name or a group of characters found within a name up to the next character in the search mask. The question mark represents a single character. The following table lists some examples of wildcard character usage:

<table>
<thead>
<tr>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIR C:\TEST*.EXE</td>
<td>Lists all files in the TEST directory having the extension .EXE.</td>
</tr>
<tr>
<td>DIR D.*</td>
<td>Lists all files in the current directory that start with the letter D.</td>
</tr>
<tr>
<td>COPY C:\BAT B:\BAK</td>
<td>Copies all files with a .BAT extension from the C: drive root directory onto the B: drive. The files on the B: drive will have an extension of .BAK. This example backs up a group of files with a single command.</td>
</tr>
<tr>
<td>DIR B:?????*</td>
<td>Lists all the files on the B: drive that have exactly four characters in the filename but have any extension. The question mark substitutes for a single character.</td>
</tr>
<tr>
<td>REN TEST?\BAT TEST?\OLD</td>
<td>Renames all files having TEST for the first four characters in the filename, followed by any single character and the .BAT extension. The files retain the same TEST? filename but gain the file extension OLD. The ? can also be used to match a single specific character in a filename.</td>
</tr>
</tbody>
</table>
System Prompt
After the execution of each command, ROM-DOS displays the system prompt indicating that it is ready for the next instruction. Unless you define the system prompt otherwise with the PROMPT command, the prompt includes only the current disk drive designation followed by a right angle bracket. For example,

A:>

One common choice for prompt line configuration is to include the current path in addition to the drive designation. For example,

A:\MY_FILES>

For more information on configuring the system prompt, refer to the PROMPT command description later in this manual.

Command Line
Your keystrokes appear to the right of the system prompt on the command line. You can use the following keys to edit the contents of the command line:

<table>
<thead>
<tr>
<th>Key</th>
<th>Editing Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>Displays one character at a time from the command line buffer. The right-pointing direction key works in the same way.</td>
</tr>
<tr>
<td>F3</td>
<td>Displays entire contents of command buffer.</td>
</tr>
<tr>
<td>Ins</td>
<td>Allows insertion of one or more characters in the command line.</td>
</tr>
<tr>
<td>Del</td>
<td>Allows deletion of a character from the command line buffer.</td>
</tr>
<tr>
<td>Esc</td>
<td>Cancels the current command line and returns you to a new, empty line.</td>
</tr>
<tr>
<td>Backspace</td>
<td>Deletes to the left. Allows backing up on the command line.</td>
</tr>
</tbody>
</table>

The last command entered on the command line is stored in a command line buffer. You can recall and edit the contents of this buffer as a way of reentering the command to repeat it or make changes to it. For example, if you intended to enter CHKDSK but accidentally entered CHKDSI, a message appears indicating that CHKDSI is a nonexistent command or filename. Rather than reentering the entire string of characters, press F3 to recall the string and use the Backspace key to back up and make the correction.

You may also make corrections to the beginning of long command line entries. For example, suppose you enter the command

CPY DATA1.DAT A:DATA1BAK.DAT

where the command is misspelled (COPY is missing the O). To correct the command line, press F1 to display the first character. Then press Ins followed by O.

You can now enter the rest of the command by pressing F3 once.
Redirecting Input and Output
Certain conventions dictate where each ROM-DOS command receives input data and where it
sends output data. However, by using the right angle bracket (>) and the left angle bracket (<), you
can redirect the input and output.

Input Redirection
The syntax for changing standard input source from keyboard input to file input is

```
< filename
```

When this is added at the end of the command line, ROM-DOS receives its instructions from the file
named `filename`.

**Caution:** If input redirection is used and the input file is incomplete, the system will hang and refuse
to accept information from the keyboard (except for Ctrl+Alt+Del to reboot).

Output Redirection
The syntax for redirecting output to a file is

```
> filename
```

When this is added to the end of the command line, standard output is temporarily directed to the
file named `filename`. If the named file already exists, its contents are replaced with the ROM-DOS
function’s output. Otherwise, a new file is created to hold the output. Output can also be redirected
to a device such as PRN (the printer).

To append output to the receiving file, rather than replace its contents, use `>>`.

```
>> filename
```

This adds the ROM-DOS function’s output to the existing contents of the named file. If the named file
does not exist, ROM-DOS creates the file.

For example, if you want to save a list of the current directory contents to a file, you can redirect the
DIR command’s output to a file by entering:

```
DIR > MYDIR.TXT
```

where `MYDIR.TXT` is the name of the file you want to contain the directory listing.
USING BATCH FILES

A batch file is a standard text file containing a list of commands that can be submitted to ROM-DOS for automatic sequential execution. Using batch files helps you avoid unnecessary retyping of command sequences that are commonly repeated, complex, or difficult to remember.

The ROM-DOS command processor provides full batch file processing, compatible with standard DOS version 6.22, DOS version 7.1, and Windows 98 DOS box long filename support. Batch files can include internal DOS commands, external DOS commands, batch file commands, names of other executable files or programs, or even the names of other batch files.

Batch Filenames

When naming batch files, use the .BAT extension on the filename. This extension tells ROM-DOS to execute the batch file when its name is entered on the command line. The name of the batch file cannot be the name of other internal commands. For example, COPY.BAT is an invalid batch filename.

To execute the batch file, enter the filename on the command line. You need not include the .BAT extension unless a file with the same filename and a .EXE or .COM extension is present in the same directory. Batch file execution begins when you press the Enter key.

Creating a Batch File

You can create a batch file by using any word processor or text editor that saves output as unformatted (ASCII) text. Or you can create them by typing directly from the keyboard into a file. This is done with the command

COPY CON filename.BAT

This tells ROM-DOS to copy the output from the console (keyboard) to the specified file. Once you have entered the above command, you may enter the contents of your batch file.

At the completion of each line, press Enter. As you enter each line, you can make corrections using Backspace and retyping. If you enter an incorrect line or wish to discontinue without saving your work, press Ctrl+C.

When you have finished entering all the lines in your batch file, press Ctrl+Z and then Enter to complete the file and return to the command line prompt.

Batch File Command Line Parameters

A batch file may use parameters placed on the command line. Insert these parameters as arguments for commands or instructions within the batch file. For example, the following batch file, named ARCHIVE, accepts a command line parameter:

PRINT %1
COPY %1 \ARCHIVE\*.*
DEL %1
Execute the ARCHIVE batch file by entering

```
ARCHIVE THISFILE.DAT
```

The %1 parameter takes on the name THISFILE.DAT, and the batch file executes the following to make a copy of THISFILE.DAT in the ARCHIVE subdirectory:

```
PRINT THISFILE.DAT
COPY THISFILE.DAT \ARCHIVE\*.*
DEL THISFILE.DAT
```

**Note**: %1 represents the first parameter in a batch file command. If a command has multiple parameters, they are represented by %2, %3, and so on.

**Batch File Commands**

In addition to the standard ROM-DOS commands, there are other commands specifically for batch files. Refer to CALL, CHOICE, ECHO, FOR, GOTO, IF, PAUSE, REM, and SHIFT later in this manual for more information on these commands.

For example, you may often run a program named MY_INFO1 followed by a program named MY_INFO2, both of which display a screen of information. After running each of these programs, you always clear the screen before proceeding. Your normal keystroke sequence is:

```
MY_INFO1
CLS
MY_INFO2
CLS
```

You could create a batch file named INFO containing the following commands:

```
MY_INFO1
PAUSE
CLS
MY_INFO2
PAUSE
CLS
```

After executing MY_INFO1 (by entering INFO on the command line), the system pauses. When you press a key, the batch file clears the screen and executes MY_INFO2, then pauses again.

Press a key to return to the command line.

**Note**: You can bypass some or all of the commands in your AUTOEXEC.BAT files during system boot. Refer to “Bypassing CONFIG.SYS and AUTOEXEC.BAT Commands” on page 26
### ROM-DOS Command Summary

Following are brief descriptions of all ROM-DOS and SOCKETS commands, including batch file commands.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>CONFIG.SYS command. It directs ROM-DOS to pause for confirmation before processing a command.</td>
<td>Page 34</td>
</tr>
<tr>
<td>@</td>
<td>Used to suppress the display of a single batch-file command line.</td>
<td>Page 35</td>
</tr>
<tr>
<td>;</td>
<td>Identifies nonexecuting lines. The same as the REM command.</td>
<td>Page 36</td>
</tr>
<tr>
<td>ANSI.SYS</td>
<td>A console device driver that allows you to support ANSI codes on the local screen.</td>
<td>Page 73</td>
</tr>
<tr>
<td>ATA.SYS</td>
<td>A PCMCIA ATA disk device driver.</td>
<td>Page 74</td>
</tr>
<tr>
<td>ATTRIB</td>
<td>Displays or modifies the attributes associated with a file.</td>
<td>Page 75</td>
</tr>
<tr>
<td>BACKUP</td>
<td>Backs up a single directory tree to a floppy drive, hard disk, or network drive.</td>
<td>Page 76</td>
</tr>
<tr>
<td>BREAK</td>
<td>Turns on or off the ability to stop program execution at a non-I/O point.</td>
<td>Page 36</td>
</tr>
<tr>
<td>BUFFERS</td>
<td>Sets the number of internal data buffers.</td>
<td>Page 36</td>
</tr>
<tr>
<td>CALL</td>
<td>Batch file command. Invokes execution of a secondary batch file.</td>
<td>Page 37</td>
</tr>
<tr>
<td>CHDIR (also CD)</td>
<td>Changes the current directory.</td>
<td>Page 37</td>
</tr>
<tr>
<td>CHKDSK</td>
<td>Checks the integrity of data on a disk. Displays information.</td>
<td>Page 77</td>
</tr>
<tr>
<td>CHOICE</td>
<td>Allows a user to make a processing choice during the execution of a batch file.</td>
<td>Page 78</td>
</tr>
<tr>
<td>CLS</td>
<td>Clears all information from the monitor's screen.</td>
<td>Page 38</td>
</tr>
<tr>
<td>COMM</td>
<td>ANSI terminal program.</td>
<td>Page 80</td>
</tr>
<tr>
<td>COMMAND</td>
<td>Starts a second DOS command processor.</td>
<td>Page 82</td>
</tr>
<tr>
<td>COPY</td>
<td>Copies files from one storage location to another.</td>
<td>Page 39</td>
</tr>
<tr>
<td>COUNTRY</td>
<td>Designates the country code for displays.</td>
<td>Page 41</td>
</tr>
<tr>
<td>CTTY</td>
<td>Changes the default terminal interacting with ROM-DOS.</td>
<td>Page 42</td>
</tr>
<tr>
<td>DATE</td>
<td>Displays and allows you to change the date from the system's internal calendar.</td>
<td>Page 42</td>
</tr>
<tr>
<td>DEL</td>
<td>Deletes specified files.</td>
<td>Page 43</td>
</tr>
<tr>
<td>DELTREE</td>
<td>Deletes one or more directory trees or individual files.</td>
<td>Page 83</td>
</tr>
<tr>
<td>DEVICE</td>
<td>Installs a device driver into ROM-DOS.</td>
<td>Page 44</td>
</tr>
<tr>
<td>DEVICEHIGH</td>
<td>Loads a device into the upper memory area, if available.</td>
<td>Page 44</td>
</tr>
<tr>
<td>DIR</td>
<td>DIRectory. Lists contents of a specified directory.</td>
<td>Page 45</td>
</tr>
<tr>
<td>DISK2IMG</td>
<td>Creates an image of a drive or disk.</td>
<td>Page 84</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
<td>Location</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>DISKCOPY</td>
<td>Copies the contents of one floppy disk to another of the same type.</td>
<td>Page 85</td>
</tr>
<tr>
<td>DISPLAY</td>
<td>Displays international letters and symbols.</td>
<td>Page 86</td>
</tr>
<tr>
<td>DOS</td>
<td>Installs ROM-DOS into High Memory Area (HMA).</td>
<td>Page 47</td>
</tr>
<tr>
<td>DUMP</td>
<td>Shows contents of memory or a file in hex and ASCII format to the display.</td>
<td>Page 87</td>
</tr>
<tr>
<td>ECHO</td>
<td>Batch file command. Displays (on) or hides (off) commands executed from a batch file.</td>
<td>Page 48</td>
</tr>
<tr>
<td>EGA/EGA3.CPI:</td>
<td>Font data files for use with the international video display driver, DISPLAY.SYS.</td>
<td>Page 87</td>
</tr>
<tr>
<td>EMM386</td>
<td>Enables expanded memory support for capable systems.</td>
<td>Page 88</td>
</tr>
<tr>
<td>ERASE</td>
<td>Erases specified files (same as DEL).</td>
<td>Page 49</td>
</tr>
<tr>
<td>EXE2BIN</td>
<td>Converts a DOS .exe file to a .com file.</td>
<td>Page 90</td>
</tr>
<tr>
<td>EXIT</td>
<td>Used to exit nested running of ROM-DOS within another program.</td>
<td>Page 50</td>
</tr>
<tr>
<td>FCBS</td>
<td>Specifies the number of File Control Blocks (FCBS) open at one time.</td>
<td>Page 50</td>
</tr>
<tr>
<td>FDISK</td>
<td>Initializes and partitions a hard disk for DOS.</td>
<td>Page 90</td>
</tr>
<tr>
<td>FILES</td>
<td>Sets the maximum number of files that can be open at one time on the system.</td>
<td>Page 50</td>
</tr>
<tr>
<td>FIND</td>
<td>Works as a filter to display only lines that contain a specified string.</td>
<td>Page 92</td>
</tr>
<tr>
<td>FOR</td>
<td>Batch file command. Performs one DOS command on a set of files.</td>
<td>Page 51</td>
</tr>
<tr>
<td>FORMAT</td>
<td>Initializes a disk so that ROM-DOS can access files on that disk.</td>
<td>Page 93</td>
</tr>
<tr>
<td>GOTO</td>
<td>Batch file command. Moves control to a specified line in the batch file.</td>
<td>Page 51</td>
</tr>
<tr>
<td>HELP</td>
<td>Lists all available ROM-DOS commands along with brief descriptions.</td>
<td>Page 52</td>
</tr>
<tr>
<td>HIMEM</td>
<td>Manages extended memory and the high memory area on a system that is 286 or greater.</td>
<td>Page 94</td>
</tr>
<tr>
<td>IF</td>
<td>Batch file command. Performs a command based on a specified condition.</td>
<td>Page 52</td>
</tr>
<tr>
<td>INCLUDE</td>
<td>Allows instructions in one configuration block to be included with instructions in another configuration block.</td>
<td>Page 53</td>
</tr>
<tr>
<td>INSTALL</td>
<td>Loads Terminate and Stay Resident (TSR) programs during CONFI.G.SYS processing.</td>
<td>Page 54</td>
</tr>
<tr>
<td>KEYB</td>
<td>Allows altering of the keyboard layout for a different language or nationality.</td>
<td>Page 95</td>
</tr>
<tr>
<td>KEYBOARD/</td>
<td>Keyboard code page data files for use with the international</td>
<td>Page 96</td>
</tr>
<tr>
<td>KEYBRD2 SYS</td>
<td>keyboard driver, KEYB.COM.</td>
<td>Page 96</td>
</tr>
<tr>
<td>LABEL</td>
<td>Creates, changes, or deletes a disk volume label.</td>
<td>Page 97</td>
</tr>
<tr>
<td>LASTDRIVE</td>
<td>Sets the maximum number of drives.</td>
<td>Page 54</td>
</tr>
<tr>
<td>LFNFOR</td>
<td>The LFNFOR command enables/disables long file names when processing FOR commands.</td>
<td>Page 54</td>
</tr>
<tr>
<td>LOADHIGH</td>
<td>Loads a program into the upper memory area, if available.</td>
<td>Page 55</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
<td>Location</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>MEM</td>
<td>Displays the used and free memory in your system.</td>
<td>Page 97</td>
</tr>
<tr>
<td>MENUCOLOR</td>
<td>Allows setting of text and background colors for the startup menu.</td>
<td>Page 55</td>
</tr>
<tr>
<td>MENUDEFAULT</td>
<td>Sets the default menu-item choice and time-out value for making a selection.</td>
<td>Page 56</td>
</tr>
<tr>
<td>MENUITEM</td>
<td>Specifies an item to be placed on the startup menu display during system boot.</td>
<td>Page 57</td>
</tr>
<tr>
<td>Mkdir (also MD)</td>
<td>Creates a new subdirectory.</td>
<td>Page 58</td>
</tr>
<tr>
<td>MODE</td>
<td>Modifies the operation of the printer, serial port, and active video display.</td>
<td>Page 98</td>
</tr>
<tr>
<td>MORE</td>
<td>Displays a text file one screen at a time.</td>
<td>Page 99</td>
</tr>
<tr>
<td>MOVE</td>
<td>Moves files and renames files and directories.</td>
<td>Page 100</td>
</tr>
<tr>
<td>MSCDEX</td>
<td>Enables the use of CD-ROM drives.</td>
<td>Page 101</td>
</tr>
<tr>
<td>NED</td>
<td>A DOS text editor.</td>
<td>Page 102</td>
</tr>
<tr>
<td>NEWFILE</td>
<td>Allows continuation of CONFIG.SYS processing from a new file.</td>
<td>Page 59</td>
</tr>
<tr>
<td>NUMLOCK</td>
<td>Sets the NUMLOCK key to on or off when your computer starts.</td>
<td>Page 60</td>
</tr>
<tr>
<td>PATH</td>
<td>Displays current command search path(s). A new path line can be specified.</td>
<td>Page 60</td>
</tr>
<tr>
<td>PAUSE</td>
<td>Batch file command. Causes execution to halt until a key is pressed.</td>
<td>Page 61</td>
</tr>
<tr>
<td>POWER</td>
<td>Conserves power on the system that employs an APM BIOS.</td>
<td>Page 106</td>
</tr>
<tr>
<td>PRINT</td>
<td>Prints a list of up to ten files.</td>
<td></td>
</tr>
<tr>
<td>PROMPT</td>
<td>Resets the appearance of the system prompt line.</td>
<td>Page 62</td>
</tr>
<tr>
<td>PROTO</td>
<td>PROTO creates function prototypes in C language files.</td>
<td>Page 107</td>
</tr>
<tr>
<td>REM</td>
<td>Batch file command for identifying non-executing lines.</td>
<td>Page 63</td>
</tr>
<tr>
<td>REMDISK</td>
<td>Remote Disk client.</td>
<td>Page 139</td>
</tr>
<tr>
<td>REMSERV</td>
<td>Remote Disk server.</td>
<td>Page 139</td>
</tr>
<tr>
<td>REMQUIT</td>
<td>Terminates Remote Disk server.</td>
<td>Page 140</td>
</tr>
<tr>
<td>REN</td>
<td>Renames files.</td>
<td>Page 64</td>
</tr>
<tr>
<td>RESTORE</td>
<td>Restores files previously saved with BACKUP to the hard disk.</td>
<td>Page 108</td>
</tr>
<tr>
<td>RMDIR (also RD)</td>
<td>Deletes a specified subdirectory.</td>
<td>Page 64</td>
</tr>
<tr>
<td>RSZ</td>
<td>Zmodem file transfer utility.</td>
<td>Page 109</td>
</tr>
<tr>
<td>SET</td>
<td>Sets environment variables and command processor strings.</td>
<td>Page 65</td>
</tr>
<tr>
<td>SHARE</td>
<td>Installs the capabilities for file sharing and file locking on your hard disk.</td>
<td>Page 111</td>
</tr>
<tr>
<td>SHELL</td>
<td>Allows selections of a command interpreter other than COMMAND.COM.</td>
<td>Page 71</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
<td>Location</td>
</tr>
<tr>
<td>-----------</td>
<td>------------------------------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>SHIFT</td>
<td>Batch file command. Shifts replaceable parameters one position to the left.</td>
<td>Page 66</td>
</tr>
<tr>
<td>SMARTDRV</td>
<td>Disk Caching utility for hard &amp; floppy disks, CD-ROM and other devices.</td>
<td>Page 112</td>
</tr>
<tr>
<td>SORT</td>
<td>Utility that sorts text files and displays the output to the standard device.</td>
<td>Page 113</td>
</tr>
<tr>
<td>STACKDEV</td>
<td>Increases the number of stacks available for IRQ handlers and Int13h.</td>
<td>Page 114</td>
</tr>
<tr>
<td>STACKS</td>
<td>Allows for the use of dynamic data stacks to handle interrupts.</td>
<td>Page 67</td>
</tr>
<tr>
<td>SUBMENU</td>
<td>Defines a menu item that represents a secondary menu.</td>
<td>Page 67</td>
</tr>
<tr>
<td>SUBST</td>
<td>Allows one drive to appear as another drive.</td>
<td>Page 115</td>
</tr>
<tr>
<td>SWITCHES</td>
<td>Allows special CONFIG.SYS file options.</td>
<td>Page 69</td>
</tr>
<tr>
<td>SYS</td>
<td>Transfers the hidden system files to a specified drive.</td>
<td>Page 115</td>
</tr>
<tr>
<td>TIME</td>
<td>Displays and allows you to change the current time from the system’s internal clock.</td>
<td>Page 69</td>
</tr>
<tr>
<td>TRANSFER</td>
<td>File transfer utility.</td>
<td>Page 116</td>
</tr>
<tr>
<td>TREE</td>
<td>Displays the path of each directory on a specified drive.</td>
<td>Page 118</td>
</tr>
<tr>
<td>TYPE</td>
<td>Displays the contents of a text file.</td>
<td>Page 71</td>
</tr>
<tr>
<td>UMBLINK</td>
<td>A non-protected mode program that can allow the creation of Upper Memory Blocks using existing RAM areas.</td>
<td>Page 119</td>
</tr>
<tr>
<td>VDISK</td>
<td>Allows the use of memory as a simulated disk driver.</td>
<td>Page 120</td>
</tr>
<tr>
<td>VER</td>
<td>Displays current version of ROM-DOS.</td>
<td>Page 71</td>
</tr>
<tr>
<td>VERIFY</td>
<td>Displays the current VERIFY state or sets the VERIFY state to on or off.</td>
<td>Page 72</td>
</tr>
<tr>
<td>VOL</td>
<td>Displays the volume label on a disk.</td>
<td>Page 72</td>
</tr>
<tr>
<td>XCOPY</td>
<td>Copies multiple files and, optionally, subdirectories.</td>
<td>Page 121</td>
</tr>
</tbody>
</table>
CHAPTER 4 | CONFIGURATION

BASIC CONFIGURATION
Certain standard settings for your system’s operation can be stored in a file named CONFIG.SYS. You may create or edit your own CONFIG.SYS file using a word processor or the COPY CON command. (See “Creating a Batch File” on page 18.)

You must place the CONFIG.SYS file in the root directory of the drive that is used for system startup or boot. If a CONFIG.SYS file is not found, the following default values are used for the following commands:

- BREAK = OFF
- BUFFERS = 15
- COUNTRY = 001
- FCBS = 4
- FILES = 8
- NUMLOCK = ON
- SHELL = COMMAND.COM /P /E:128
- STACKS = 0,0

Example
A typical CONFIG.SYS file might look like this.

- BREAK = ON
- FILES = 15
- BUFFERS = 15
- DEVICE = C:\ROMDOS\HIMEM.SYS

USING MULTIPLE-USER CONFIGURATIONS
Your CONFIG.SYS file can be used to define multiple system configurations. This is handy when several people share a computer and require different working environments. It is also useful for booting your own computer using different device drivers, paths, or settings, depending on the intended computer tasks.

To define multiple configurations within the CONFIG.SYS file, you first need to define a startup menu. Each menu item represents a different system configuration option. Then, for each item on the menu, define a configuration block. Each configuration block contains the specific commands to be implemented as the system completes booting.

The menu-item definition and all configuration blocks are marked with a block header. A block header is a descriptive label enclosed in square brackets ([ ]). The start of the menu items must be marked with the block header [MENU]. Each configuration block may have a unique label of your choice. This label can be up to 70 characters long and can contain most printable characters, including spaces, backslashes (\), forward slashes (/), commas (,), semicolons (;), and equal signs (=). Square brackets ([ ]) cannot be used in block names.
The menu block (or submenu block) may contain only the following commands (a full description for each command can be found in the Command Descriptions section in Chapter 5):

- MENUITEM
- MENUDEFAULT
- MENUCOLOR
- SUBMENU
- NUMLOCK

**Note:** Although NUMLOCK may be used outside of a menu/submenu block, it is typically used to enable the keypad for menu-choice selections in the menu block.

A sample menu block might look as follows:

```
[MENU]
menuitem=Research, Research and Development
menuitem=WP, Word Processing
menuitem=Games, Games
menucolor=8,5 menudefault=WP, 10
```

When the system boots, the following menu displays:

**ROM-DOS 6.22 Startup Menu**

1. Research and Development
2. Word Processing
3. Games

Enter a choice: 1

Each menu item has its own configuration block. Items that are common to all menu choices can be placed in a Configuration Block labeled [COMMON]. All instructions in the common block are carried out along with the specific instructions for any menu item. The [COMMON] block can also be placed at the end of your CONFIG.SYS file so that applications can append commands into this area as the application installs. You may have as many common blocks as you want. The instructions found in the common block(s) are processed in the order they are listed in the CONFIG.SYS file.

When the CONFIG.SYS file is processed by ROM-DOS, it first displays the startup menu that was defined in the [MENU] configuration block, and then waits for your response. The choice made from the menu determines the configuration block whose commands are to be executed. After the menu selection, processing starts with any instructions in CONFIG.SYS prior to the menu block. Then, instructions in the selected configuration block (including instructions added in via an INCLUDE statement) and all common blocks are processed in the order they are listed in CONFIG.SYS. ROM-
DOS ignores the instructions in any nonselected configuration blocks or submenus.

To continue the above example, the configuration blocks might appear as follows:

```
[COMMON]
device=c:\romdos\himem.sys dos=high
break=on [RESEARCH]
files=20 buffers=50
device=vdisk.sys 128 /e [WP]
files=10 buffers=10 lastdrive=m
device=c:\network\loadnet.sys [GAMES]
include=wp device=mouse.sys [COMMON]
```

If choice number 3 is made, selecting [GAMES], the instructions in the [COMMON] configuration block are processed first, followed by the instructions in the [GAMES] configuration block. The [GAMES] section makes use of the INCLUDE command. All of the instructions provided for the WP menu choice also apply to [GAMES]. If any instructions are in the final [COMMON] configuration block, they are processed last.

### EXTENDING MENU ITEMS TO AUTOEXEC.BAT

The defined name of the menu item you have chosen becomes the value of the environment variable CONFIG. For example, if you choose number 3, GAMES, from the preceding menu, the variable CONFIG is set to GAMES. The CONFIG environment variable can then be used in your AUTOEXEC.BAT file to further customize the startup sequence. This environment variable is referenced by `%CONFIG%` in your AUTOEXEC.BAT file.

An example of an AUTOEXEC.BAT file that continues the customization process from the preceding MENU may look like this.

```
prompt $p$g
set temp=c:\mystuff\temp c:\virus\scanit.com
rem Go to section that matches menu rem choice made in CONFIG.SYS
goto %config%
:RESEARCH
path c:\bin;c:\ROMDOS;c:\ROMDOS\utils;c:\BORLANDC cd \ROMDOS
rem Skip other sections and move to end goto end
:WP
path c:\bin;c:\ROMDOS;c:\wp wp
rem Skip next section and move to end goto end
:GAMES
path c:\bin;c:\ROMDOS;c:\gamedir cd \gamedir
gamelist.bat goto end
:end
```
BYPASSING CONFIG.SYS AND AUTOEXEC.BAT COMMANDS

ROM-DOS offers the capability to bypass some or all of the commands in your AUTOEXEC.BAT and CONFIG.SYS files during the boot process. This feature may be useful in tracking system problems that may be related to one or more commands in either of these two files.

To bypass the instructions in both your AUTOEXEC.BAT and CONFIG.SYS files, follow these steps:

1. Turn on, or restart your computer if it is already on, and wait for the following message.
   Starting ROM-DOS...

2. As the above message is being displayed, press the F5 key or hold down the SHIFT key to display the following message.
   ROM-DOS is bypassing your CONFIG.SYS and AUTOEXEC.BAT files.

Your system then continues the boot process using the basic default configurations. You may notice a difference in the way your system behaves. For instance, installable device drivers and memory device drivers are not loaded, and system prompts and paths have default values. If the command interpreter COMMAND.COM is not in the root directory, ROM-DOS may not be able to locate it.

STEPPING THROUGH CONFIG.SYS AND AUTOEXEC.BAT COMMANDS

If you suspect that one or more commands in either the CONFIG.SYS or AUTOEXEC.BAT files are causing problems in your system, you can choose to process or bypass each command as follows:

1. Turn on, or restart your computer if it is already on, and wait for the following message.
   Starting ROM-DOS...

2. As the above message is being displayed, press the F8 key to display the first command in the CONFIG.SYS file.

3. For each command, ROM-DOS displays a [Y,N]? prompt. To process the instruction, press Y. To bypass the instruction, press N. ROM-DOS then moves to the next command in the CONFIG.SYS file. To bypass the confirmation prompt for the remaining instructions in the CONFIG.SYS file and skip the AUTOEXEC.BAT file, you can press the F5 key at any [Y,N]? prompt.

   Note: You can prompt for a single CONFIG.SYS command by using the question mark (?) command prior to the equals sign (=) in the command line. For a complete description of each CONFIG.SYS command, refer to the internal Command Descriptions starting on page 35.

When ROM-DOS finishes all of the commands in the CONFIG.SYS file, it prompts with:

   Process AUTOEXEC.BAT [Y, N]?

Press Y to selectively process the commands in AUTOEXEC.BAT, otherwise press N to bypass all commands in the AUTOEXEC.BAT file.
ENVIRONMENT VARIABLES
A block of system memory is reserved for the definition of certain strings, called environment variables, to be used in the command processor environment. These include the settings you may establish with the PATH and PROMPT commands as well as COMSPEC, which is automatically defined by ROM-DOS at system startup. Environment variables may be defined using the SET command that is explained in the Command Descriptions section in chapter 6.

CONFIGURING ROM-DOS FOR INTERNATIONAL USE
You can configure ROM-DOS to conform to local conventions for date, time, and currency formats by giving the COUNTRY= command. You can also use the COUNTRY command to select an international character set (known as a code page) that determines the sort order. Any code page can also be shown on EGA and VGA displays by loading DISPLAY.SYS. You can remap a keyboard to provide support for various languages and layouts by running KEYB.COM.

ROM-DOS uses a country code to identify the country conventions that are to be used. In most cases, the country code is the same as the international long distance telephone dialing code.

A code page is a set of 256 symbols, including letters, digits, punctuation, and graphic characters. The first 128 symbols in a code page are the standard ASCII characters and are identical in all code pages. The last 128 symbols vary depending on the code page. These symbols include the graphic, line-drawing characters, plus many international letters, currency symbols, and other assorted symbols. A number, such as 437 identifies each code page.

A computer display has one hardware code page built into it. Typically, this is code page 437, which is the standard US code page. CGA and monochrome monitors can only display the hardware code page in text mode. EGA and VGA monitors display the hardware code page unless you load special software (like DISPLAY.SYS). Each country supports a default code page and an alternate code page. The following table lists the valid combinations.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Australia</td>
<td>061</td>
<td>437</td>
<td>850</td>
</tr>
<tr>
<td>Belgium</td>
<td>032</td>
<td>850</td>
<td>437</td>
</tr>
<tr>
<td>Brazil</td>
<td>055</td>
<td>850</td>
<td>437</td>
</tr>
<tr>
<td>Canadian-French</td>
<td>002</td>
<td>863</td>
<td>850</td>
</tr>
<tr>
<td>Czech Republic</td>
<td>042</td>
<td>852</td>
<td>850</td>
</tr>
<tr>
<td>Denmark</td>
<td>045</td>
<td>850</td>
<td>865</td>
</tr>
<tr>
<td>Finland</td>
<td>358</td>
<td>850</td>
<td>437</td>
</tr>
<tr>
<td>France</td>
<td>033</td>
<td>850</td>
<td>437</td>
</tr>
<tr>
<td>Germany</td>
<td>049</td>
<td>850</td>
<td>437</td>
</tr>
<tr>
<td>------------------------------</td>
<td>------</td>
<td>-----------</td>
<td>-----------------------</td>
</tr>
<tr>
<td>Hungary</td>
<td>036</td>
<td>852</td>
<td>850</td>
</tr>
<tr>
<td>Italy</td>
<td>039</td>
<td>850</td>
<td>437</td>
</tr>
<tr>
<td>Japan</td>
<td>081</td>
<td>932</td>
<td>---</td>
</tr>
<tr>
<td>Latin America</td>
<td>003</td>
<td>850</td>
<td>437</td>
</tr>
<tr>
<td>Netherlands</td>
<td>031</td>
<td>850</td>
<td>437</td>
</tr>
<tr>
<td>Norway</td>
<td>047</td>
<td>850</td>
<td>865</td>
</tr>
<tr>
<td>Poland</td>
<td>048</td>
<td>852</td>
<td>850</td>
</tr>
<tr>
<td>Portugal</td>
<td>351</td>
<td>850</td>
<td>860</td>
</tr>
<tr>
<td>Russia</td>
<td>007</td>
<td>437</td>
<td>866</td>
</tr>
<tr>
<td>Spain</td>
<td>034</td>
<td>850</td>
<td>437</td>
</tr>
<tr>
<td>Sweden</td>
<td>046</td>
<td>437</td>
<td>850</td>
</tr>
<tr>
<td>Switzerland</td>
<td>041</td>
<td>850</td>
<td>437</td>
</tr>
<tr>
<td>United Kingdom</td>
<td>044</td>
<td>437</td>
<td>850</td>
</tr>
<tr>
<td>United States</td>
<td>001</td>
<td>437</td>
<td>850</td>
</tr>
<tr>
<td>Former states of Yugoslavia</td>
<td>038</td>
<td>852</td>
<td>850</td>
</tr>
</tbody>
</table>

**Changing Country Conventions**

The command to instruct ROM-DOS to use German conventions, for example, is

```
COUNTRY=049
```

The COUNTRY= command requires COUNTRY.SYS to be present in the root directory of the boot drive. Setting a country code affects

- Date and time formats
- The symbol used to denote currency

If you only specify a country code, ROM-DOS uses the default code page for that country. You can choose the alternate code page by including it in the COUNTRY= command. This command tells ROM-DOS to use German conventions for things such as date and time but using code page 437 instead of 850, the default code page.

```
COUNTRY=049, 437
```

Setting a system code page affects

- The sort order for alphabetizing
- The rules for converting international letters to uppercase
The individual application programs determine whether they make use of these conventions. For example, DOS uses the date format for displaying directories and for showing and getting the current date and time. Some programs may choose to ignore the country information and continue to display dates in a specific format.

**Displaying Different Code Pages**

To display a code page other than the hardware code page, you must load DISPLAY.SYS in CONFIG.SYS. The following command sets the display to show code page 850, assuming both the DISPLAY.SYS driver and the EGA.CPI font file are located in the C:\DOS directory:

```
DEVICE=C:\DOS\DISPLAY.SYS 850 C:\DOS\EGA.CPI
```

The available font files are named EGA.CPI and EGA3.CPI. They are both used for EGA and VGA systems.

If you have an EGA or VGA system, the character fonts are immediately switched to the requested code page. Some characters may look different after you load DISPLAY.SYS because ROM-DOS uses its own font for all 256 characters. For example, your hardware font might use a square-like zero character, but ROM-DOS might use a round zero character. The differences are minor.

**Printing Different Code Pages**

At this time, ROM-DOS does not support printing code pages other than those stored in the printer hardware.

**Changing the Keyboard Layout**

To alter the keyboard layout, issue the KEYB command from within DOS. You can do this by running KEYB in AUTOEXEC.BAT or directly from a DOS prompt. Use this following command to switch to a German keyboard layout, for example:

```
KEYB GR
```

Most countries have two valid code pages. If you do not specify a code page, the default code page is used. The following table lists the valid combinations.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Belgium</td>
<td>be</td>
<td>850</td>
<td>437</td>
</tr>
<tr>
<td>Canadian-French</td>
<td>cf</td>
<td>863</td>
<td>850</td>
</tr>
<tr>
<td>Czech Republic</td>
<td>cz</td>
<td>850</td>
<td>852</td>
</tr>
<tr>
<td>Denmark</td>
<td>dk</td>
<td>850</td>
<td>865</td>
</tr>
<tr>
<td>France</td>
<td>fr</td>
<td>437</td>
<td>850</td>
</tr>
<tr>
<td>Germany</td>
<td>gr</td>
<td>437</td>
<td>850</td>
</tr>
</tbody>
</table>
---|---|---|---
Italy | it | 437 | 850
Latin America | la | 850 | 437
Netherlands | nl | 437 | 850
Norway | no | 850 | 865
Poland | pl | 850 | 852
Portugal | po | 850 | ---
Russia | ru | 437 | 866
Spain | sp | 850 | 437
Sweden | sv | 437 | 850
Swiss French | sf | 850 | 437
Swiss German | sg | 850 | 437
United Kingdom | uk | 437 | 850
United States | us | 437 | 850
Former states of Yugoslavia | yu | 850 | 852

After you have loaded KEYB, your keyboard layout reflects the country you chose. You can switch back to the US keyboard layout at any time by pressing Ctrl+Alt+F1 (Alt+Left-Shift from Russian and Czech Republic keyboards). You can return to the modified keyboard layout by pressing Ctrl+Alt+F2 (Alt+Right-Shift from Russian and Czech Republic keyboards). You can also switch to a completely different layout by running KEYB again and specifying another country identifier.

Appendix G includes diagrams of the different keyboard layouts. The diagrams show native-language keyboards that tend to have different layouts from US keyboards. KEYB does its best to map the available hardware keys to the desired layout. Some symbols may not be available when using a US keyboard and a non-US layout. In the diagrams, symbols appearing in the lower right corner of a key are activated by pressing the AltGr key along with the desired key. On keyboards without a right AltGr, pressing Ctrl+Alt represents the AltGr key.

Note: The AltGr key is not found on a standard US keyboard.

Some keys are prefix keys that don’t generate any symbol by themselves but modify the following keystroke. For example, on most European keyboards, the apostrophe key (’) causes the next letter to be accented. To produce an apostrophe alone, press the apostrophe key followed by the space bar. Other keys that may behave as prefixes, depending on the current keyboard layout, are the backward apostrophe (’), tilde (“”), and caret (“”).
Some keys represent symbols that are not available in all code pages. For example, the German keyboard can produce a capital A with a caret above it. In the default German code page (850), that symbol is represented by the code 182. However, in the alternate German code page (437) there is no such symbol. If you are using the German layout and code page 437, and you try to produce a capital A with a caret above it, you get a caret character followed by an uppercase A (^A).

Note that the keyboard code page could be set to not match the display code page. This can lead to confusion, as the keyboard may produce characters that appear on screen as other symbols.

Continuing the above example, if you are using the German layout with keyboard code page 850, but your display code page is 437, and you produce an uppercase A with a caret above it, the screen displays a box drawing character.

**Configuring Your System: an Example**

To completely configure your system, you need to include commands in your CONFIG.SYS and AUTOEXEC.BAT files. The following sample files set up a computer to use German conventions; to use code page 850 for sorting, uppercase conversions, and the display; and to switch the keyboard layout to German. COUNTRY.SYS is assumed to be in the root directory of the boot drive, and DISPLAY.SYS, EGA.CPI, KEYB.COM, and KEYBOARD.SYS are assumed to be in the C:\DOS directory.

**CONFIG.SYS**

```
BUFFERS=20
FILES=20
COUNTRY=049
DEVICE=C:\DOS\DISPLAY.SYS 850 C:\DOS\EGA.CPI
```

**AUTOEXEC.BAT**

```
@ECHO OFF
PROMPT $P$G
PATH C:\DOS;C:\UTILITY
KEYB GR
```
CHAPTER 5 | ROM-DOS INTERNAL COMMANDS

Internal Command Descriptions

The following pages provide a complete description of each ROM-DOS internal command, including batch file commands. Each entry includes a description of the command’s purpose, command entry syntax, remarks, and examples as appropriate. Internal commands are part of the command processor program COMMAND.COM. These functions are only available while COMMAND.COM is running.

For on-line help information and syntax descriptions, use the /? option with any command. For example:

\texttt{DIR /?}

\textbf{Note:} The file COMMAND.HLP must be available in the root directory on the boot drive to access help information for internal commands.

\textbf{?}

The question mark (?) command directs ROM-DOS to pause and ask for confirmation before processing the command. Place it on a command line in the CONFIG.SYS file following the actual command.

\textbf{Syntax}

\[ [\text{command}]? = \text{command}\_arguments \]

\textbf{Remarks}

The \textit{command} can be any of the following standard CONFIG.SYS commands.

- \texttt{BREAK=}
- \texttt{BUFFERS=}
- \texttt{DEVICE=}
- \texttt{FCBS=}
- \texttt{DOS=}
- \texttt{INSTALL=}
- \texttt{FILES=}
- \texttt{LASTDRIVE=}
- \texttt{STACKS=}
- \texttt{SWITCHES=}

\texttt{command\_arguments} can be any of the available options defined for the command. Refer to the individual command description for complete instructions.

The question mark (?) should be placed just before the equal sign (=) in the command line.

\textbf{Example}

\texttt{DEVICE?=VDISK.SYS 64 /E}
Causes ROM-DOS to pause and ask for confirmation before installing the VDISK. If Yes (Y) is answered, the installation will continue. If No (N) is answered, the device will not be loaded and processing moves on to the next CONFIG.SYS command line.

@  

Internal Command

The @ sign command prevents a single command in a batch file from being echoed to the screen as the batch file is being run. Place the @ sign in front of the command whose display is to be suppressed.

Syntax

@ [batch file command]

Remarks

The batch file command argument can be any executable line in your batch file.

Examples

@COPY FILE1.BAT FILE1.SAV

Executes the COPY instruction, but the instructions are not echoed to the screen as the batch file runs.

@ECHO OFF

The ECHO OFF command differs from the @ sign in that it causes all subsequent commands not to be displayed on the screen. To prevent the ECHO OFF command from displaying itself, place the @ sign in front of the command.

;  

Internal Command

The semicolon (;) command has two purposes: to allow comments in a batch or CONFIG.SYS file, and to temporarily disable a command without physically deleting the command from the file. Refer also to the REM command.

Syntax

;[any text here]

Remarks

Use the (;) command to functionally remove a command from the CONFIG.SYS file without actually deleting it from the CONFIG.SYS file.

Examples

;C:\BIN\VDISK.SYS 64 /E

Prevents the VDISK command from executing until the (;) command is removed.
BREAK

The BREAK command expands the list of operations that can be stopped by pressing Ctrl+C or Ctrl+Break. Alternatively, returns to the default setting of a limited number of break-able operations.

Syntax

```
BREAK [ON|OFF]
```

Remarks

In the normal default condition, the BREAK switch is off. In the off mode, the stop commands, Ctrl+C and Ctrl+Break, affect activities that read from or write to the keyboard, the screen, or the printer. ROM-DOS does not look for these stop commands during any other activities.

With the BREAK switch set to ON, ROM-DOS looks for Ctrl+C and Ctrl+Break during activities such as disk reads and writes.

Examples

```
BREAK ON
Expands the BREAK list.

BREAK OFF
Returns to limited BREAK list.

BREAK
Displays the current BREAK setting.
```

BUFFERS

ROM-DOS has internal buffers to temporarily hold data read from the disk. Increasing the number of internal buffers speeds system performance.

Syntax

```
BUFFERS = number
```

Remarks

Each buffer used by ROM-DOS requires 512 bytes of RAM. The BUFFERS command increases or decreases the amount of RAM used by the operating system.

The minimum number of buffers is two, and the maximum number is 40. When the number is less than two, the number of BUFFERS is set to two. When the number is larger than 40, then BUFFERS is set to 40. The default number of buffers is calculated using a scale. The ratio is 15 to 640KB. Consequently, a system having 640KB of conventional memory will have 15 buffers. If the calculated number is less than two, then two buffers is used.

Examples

```
BUFFERS = 10
This command tells ROM-DOS to create ten buffers. These ten buffers use 5120 bytes of RAM.
```
CALL

Batch File, Internal Command

The CALL command invokes execution of a secondary batch file without exiting the primary batch file. When the secondary batch file is done executing, control is returned to the primary batch file.

Syntax
CALL batchfile [batchfile arguments]

Remarks
Parameters for the secondary batch file may also be included, if appropriate.

Examples
CALL BATCH2
Executes the batch file BATCH2.BAT.

CALL MYBATCH FILEX FILEZ
Executes the batch file MYBATCH.BAT. The arguments passed to MYBATCH.BAT are
%1 = FILEX
%2 = FILEZ

CHDIR (CHange DIRectory)

Internal Command

The CHDIR command changes the current directory.

Syntax
CHDIR [drive:][path]subdir
CD [drive:][path]subdir

Remarks
Subdir is the name of the new current subdirectory. You may use CD in place of CHDIR.
The new directory that is to become the current directory must already exist. Refer to MKDIR for information on the creation of subdirectories.
A series of two periods (..) may be used to indicate a move back to the next-higher or parent directory.
Specifying only the backslash (\) for the subdir argument moves you to the root directory of the current drive.

Examples
CHDIR \TOOLS
Moves you into the subdirectory named TOOLS, whose parent directory is the root of the current drive.
CD A:

CHAPTER 5 | ROM-DOS INTERNAL COMMANDS
CLS (CLear Screen)

The CLS command clears the monitor to display a blank screen.

Syntax

CLS

Remarks

CLS clears the screen, and then redispies the DOS prompt and cursor in the upper left-hand corner. There are no additional options for CLS.
The COPY command copies a file or set of files to a specified destination: another disk, another subdirectory on the current disk, or on a completely different drive. COPY may also be used to alter the filename within the current directory. In addition, this command can be used to direct communication between files and devices (for example, file contents to a printer or keyboard input to a file).

Syntax

This command has several possible formats. The essential structure of each command is

COPY source target option

The source is the copy from filename or device, and the target is the copy to filename or device. Following are various configurations of the COPY command format.

COPY [drive:] [path] filename [/option] [drive:] [path] filename [/option]

Where the first filename indicates the source file(s) to be copied, and the second filename indicates the target area on which to copy.

COPY [drive:] [path] filename [/option] + [drive:] [path] filename [/option] [drive:] [path] filename [/option]

As shown, several source filenames may be listed to be copied into the target filename that is listed last. The source files are concatenated into the target file.

COPY [drive:] [path] filename [/option] device

The target device is a console or printer (PRN).

COPY device [drive:] [path] filename [/option]

The device is the source such as a keyboard or console, the output of which is directed to the target filename.

Options

The /A and /B options represent ASCII and binary, respectively, and act as switches that allow each of these file types to be copied. When /A or /B is used, it applies to the preceding filename. The option remains in effect for any filenames following in the command line until superseded by another /A or /B.

/A and /B options are only needed when combining ASCII and binary files.

/A treats the file as an ASCII file (text file). When used with the source file, everything is copied up to, but not including, the first CTRL+Z end-of-file marker. When /A is used on the target file, a Ctrl+Z is added as the last character in the file.

/B treats the file as a binary file. When /B is used with the source file, the entire file is copied regardless of any Ctrl+Z characters. When /B is used with the target filename, no Ctrl+Z end-of-file marker is added.
/H copies hidden files.

/V is not implemented in ROM-DOS for code size reasons. It is included to maintain command-line compatibility.

/Y copies the current file(s) over the existing file(s) of the same name(s) without confirmation. This option overrides the setting made by the COPYCMD environment variable.

/-Y confirms the copy of one file over the existing file of the same name. This option overrides the setting made by the COPYCMD environment variable.

Set the COPYCMD environment variable with the SET command. This allows you to set confirmation on or off for the COPY command. If you always want to be prompted for confirmation when a file will copy over an existing file, set COPYCMD= /-Y. To automatically overwrite without confirmation during a copy instruction, set COPYCMD= /Y.

/Y. For proper usage, refer to the SET command.

Remarks

When no filename is specified for the target, the new copy is given the same name as the source filename.

When no drive or path is specified for the source, the current drive and directory are assumed.

When no drive or path is specified for the target, the current drive and directory are assumed.

If a drive name only is specified without a path, the current directory for that drive is assumed.

Both source and target filenames may include wildcard characters (*) and (?) to specify a set of several files.

Examples

COPY LETTER.TXT A:
Copies the file LETTER.TXT (in your current drive and path) to the current directory on the disk in drive A:

COPY *.DOC A:
Copies all files in the current directory with an extension of .DOC to the default directory of drive A:

COPY DATAORIG.DOC DATABACK.DOC
Creates a backup copy, DATABACK.DOC, from the file DATAORIG.DOC. The new file is located in the current directory.

COPY JAN.DAT + FEB.DAT + MAR.DAT QTR1.DAT
Copies the files JAN.DAT, FEB.DAT, and MAR.DAT in sequence into the single file, QTR1.DAT.

COPY CON NEWFILE.TXT
Sets up your console (keyboard) to input directly to NEWFILE.TXT. Ctrl+Z followed by Enter closes the file and returns to normal command line operation.
COUNTRY

ROM-DOS supports multiple country formats for time, date, and currency, and other basic country-specific information. A country is identified by a three-digit, international telephone country code.

Syntax

COUNTRY = countrynumber [codepage]

Remarks

The file COUNTRY.SYS must be present in the same directory as your CONFIG.SYS file.

If you do not specify a code page, ROM-DOS uses the default code page for the chosen country. If a code page is specified, it must be either the default or alternate code page for the chosen country.

This command affects the ROM-DOS DATE and TIME commands. Applications that use DOS functions to determine the date, time or currency format, or request that DOS provide character sort order or uppercase information, are affected as well.

Refer to the table provided on page 31 for a list of the currently supported countries.

Examples

COUNTRY= 049
COUNTRY = 049, 437

The next time you start ROM-DOS with either of these COUNTRY commands, the DATE and TIME will be displayed as follows:

DATE
Current date is Wed 20.06.1998
Enter new date (dd.mm.yyyy):

TIME
Current time is 16:39:54,45
Enter new time:

The first COUNTRY command above uses codes page 850, by default, for sorting and case conversion. The second COUNTRY command example uses the specified code page 437 instead.
CTTY (CHANGE TELETYPICAL)  

The CTTY command directs input and output to a different device other than your computer’s standard keyboard and monitor.

Syntax

CTTY device

Remarks

Use CTTY for any situation requiring interaction with an alternate console.

The CTTY command only affects communication with ROM-DOS and with programs that work through ROM-DOS for input and output. For example, BASIC uses standard keyboard input regardless of previous CTTY command usage.

Examples

CTTY COM2
Sets the device on COM2 as the input/output device.

CTTY CON
Returns control to the standard keyboard.

DATE

The DATE command displays the current date (month, day, year) as known to ROM-DOS and also allows you to change it.

Syntax

DATE [mm-dd-yy]
DATE [mm-dd-yyyy]

Remarks

The date set by this command is used, among other things, for date stamping your file revision dates. This information is displayed when you execute a directory listing of your files.

You may want to include the DATE command in your AUTOEXEC.BAT file, so you can set the date during boot. If your computer has an internal, battery-operated clock, you won’t need to do so.

The format of the date command is also dependent on the country specified in CONFIG.SYS. The date is displayed according to local standards for the specified country.

Refer also to the TIME command.

Examples

When entering this command without specifying mm-dd-yyyy, the current date as known to
ROM-DOS is displayed, and you are prompted to enter a new date.

Current date is Sat 6-10-1989
Enter new date (mm-dd-yyyy):

If you do not want to change the date, press Enter. Otherwise, key in the current date and press Enter.

Alternatively, you may skip the display and prompting by entering the current date on the command line. To enter June 10, 1999 (assuming US country support), enter the DATE command as follows:

DATE 6-10-1999
Valid entries for months, days, and years are

mm = 1-12 dd = 1-31 yyyy = 1980-2099

ROM-DOS calculates the day of the week; do not include it in your entry.

**DEL (DELeete)**

The DEL command deletes a specified file or set of files.

**Syntax**

DEL [drive:] [path] filename [/P]

**Remarks**

The DEL command and the ERASE command are functionally identical.

When no drive is specified, the default is assumed. When no path is specified, the default path is assumed.

Global filename characters ? and * (wildcards) can be used in the filename. This should be done with caution as it is possible to delete multiple files unintentionally.

When the filename *.* is used to delete all files in the specified subdirectory, a verification message is displayed:

Are you sure (Y/N) ?

Enter Y to delete all files in the specified subdirectory.

DEL deletes files within a subdirectory, not the subdirectory itself. For subdirectory removal, refer to the RMDIR command.

**Caution:** No ROM-DOS command can undelete a file. Although utilities exist that can attempt an undelete, certain factors can cause the deleted file to be partially or totally lost. The DEL command should be treated as a permanent deletion.
Options

The /P option causes DEL to pause and prompt you before each file is deleted. This option is most useful when deleting files with wildcards.

Examples

DEL A:*.* /P

Deletes all files on the A: drive with a .DOC extension. Before each file is deleted, you are prompted to determine if that file should be deleted. A sample prompt is shown below:

C:\> DEL MYLETTER.DOC MYLETTER.DOC, Delete (Y/N) ?

Deletes the file MYLETTER.DOC from the current default subdirectory.

DEL *.DOC

Deletes all files in the current subdirectory with a .DOC file extension.

DEVICE

CONFIG.SYS Command

The DEL command deletes a specified file or set of files.

Syntax

DEVICE = [drive] [path] driver name [arguments]

Remarks

A device driver allows ROM-DOS to access hardware that is not common in all PCs.

The full drive path and filename of the device must be specified. The arguments are different depending on the device driver.

Example

DEVICE=C:\BIN\VDISK.SYS 120 /e

Installs the ROM-DOS RAM disk driver, VDISK.SYS, via the DEVICE command and dedicates 120KB of extended memory to the RAM disk.

DEVICEHIGH

CONFIG.SYS Command

The DEVICEHIGH command loads an installable device driver into the upper memory area, if available.

Syntax

DEVICEHIGH = [drive] [path] driver name [arguments]
Remarks
A device driver allows ROM-DOS to access hardware that is not common in all PCs. A device can be loaded into the upper memory areas if they are available and there is enough free upper memory to accommodate the driver’s needs. To make high memory available, the EMM386.EXE and HIMEM.SYS utilities must be loaded. If these utilities are not loaded or there is not enough upper memory available, the device is loaded into conventional memory.

The full drive path and filename of the device must be specified. The arguments differ depending on the device driver.

Example
DEVICEHIGH=C:\BIN\MYDEVICE.SYS /20 /M
Installs a driver MYDEVICE with its command line arguments as specified. The device is loaded into upper memory, if available.

DIR (DIRectory)

The DIR command displays a list of the files that are in a specific directory.

Syntax
DIR [drive:][path][filename][/option]

Remarks
Use the DIR command to list all the files in a directory or to show the directory entries of specific files. The standard directory display format includes columns for filenames, filename extensions, file sizes, and the dates and times the files were created.

Options
The /A option causes the DIR command to display only the files that match the specified filename and have the given attribute. The following list shows the legal attribute descriptions.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Archive-ready for archiving</td>
</tr>
<tr>
<td>D</td>
<td>Directories</td>
</tr>
<tr>
<td>H</td>
<td>Hidden files</td>
</tr>
<tr>
<td>R</td>
<td>Read only files</td>
</tr>
<tr>
<td>S</td>
<td>System files</td>
</tr>
<tr>
<td>X</td>
<td>Show attributes</td>
</tr>
<tr>
<td>-</td>
<td>The minus sign can be used to negate listed attributes. For example, to select all files that do not have the archive bit set, use /A-A option.</td>
</tr>
</tbody>
</table>
The /B or bare option causes the filenames to be displayed without volume label, date, time, or size information.

The /L option causes the filenames to be displayed in lowercase.

The /P option selects page mode, which makes ROM-DOS pause the display each time the screen is full. Press any key to go onto the next page of entries.

The /O option causes the filenames to be displayed in sorted order. The sort order can contain one or more of the following attributes.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>By date and time, newest first</td>
</tr>
<tr>
<td>E</td>
<td>Alphabetic order by extension</td>
</tr>
<tr>
<td>G</td>
<td>Directories grouped before files</td>
</tr>
<tr>
<td>N</td>
<td>Alphabetic order by name</td>
</tr>
<tr>
<td>S</td>
<td>By size, smallest first</td>
</tr>
<tr>
<td>-</td>
<td>The minus sign can precede the sort option to reverse the sort order. For example, to sort all files in the directory in reverse alphabetic order, use /O-N option.</td>
</tr>
</tbody>
</table>

The /S option causes the display to include files in subdirectories also. The /V option enables verbose mode.

The /W option displays the list in a wide format without date, time, or size.

The /4 option displays the date with a 4 digit year. The default behavior is to display the last 2 digits of the year.

The DIRCMD environment variable can be used to set the default preferences for the DIR command. The SET command assigns the values to an environment variable. Refer to the SET command for proper usage. For example, to always have the /P option set for DIR, use the statement SET DIRCMD=/P. You can override the default settings in DIRCMD by using the minus sign (-) preceding the option. To cancel the paging for a single use of the DIR command, enter DIR /-P.

The DIRSIZE environment variable is useful for nonstandard screen sizes. As with the DIRCMD variable, the SET command assigns the values. The syntax is

```
SET DIRSIZE rows[,columns]
```

The values for rows and columns only have an affect when you use the /P or /W options with the DIR command. The /P (paging) option uses rows to display the correct number of lines before pausing. The /W (wide display) uses the columns argument to display the correct number of columns across the width of the screen.
Examples

```
DIR
Lists the directory entries of all files in the current drive and directory.

DIR B:\MEMOS
Lists all files in the subdirectory MEMOS on drive B:.

DIR A:*\.RPT/P
Lists the directory entries for all files in drive A: with the extension RPT, displayed one screen at a time.

DIR /ON
Lists all files in the current directory, sorted by filename order.

DIR /AH
Lists all hidden files in the current directory.

DIR *.DOC /B
Lists all files in the current directory with a .DOC extension without file sizes or volume labels.
```

DOS

```
ROM-DOS can be loaded into an upper portion of memory referred to as the High Memory Area (HMA), freeing more of conventional (lower 640KB) DOS memory for use by applications.

Syntax

```
DOS=[HIGH | LOW
```

Remarks

The DOS=HIGH command frees up more of the standard DOS memory for use by applications.

This command only works on 286 and higher CPUs with extended memory and Datalight’s HIMEM.SYS High Memory Manager, or equivalent, installed. It will not work on standard XT class PCs. Setting DOS=HIGH is ignored when ROM-DOS is in ROM.

Example

```
DEVICE=HIMEM.SYS
DOS=HIGH
```

Loads the high memory area device driver and then loads ROM-DOS into the HMA for increased conventional memory. The high-memory-area device driver must be loaded first, before DOS=HIGH.
ECHO

The ECHO command controls whether ROM-DOS commands and other messages are displayed during batch file execution. ECHO also allows you to create your own messages for display.

Syntax

   ECHO [ON|OFF]
   ECHO message
   ECHO

Remarks

   The ON option is the default ECHO setting. It causes commands in a batch file to be displayed as ROM-DOS executes them. Typing ECHO OFF turns off such display, after which the ON option switches it back on again.

   The ECHO command alone, entered without the ON or OFF option, displays the current ECHO setting.

   The message option is a string of characters, such as a warning or a reminder, that you want ROM-DOS to display. Although your message displays whether ECHO is on or off, the message display is useful only when ECHO is off.

   To create a message, enter ECHO followed by your message. If your message is more than one line long, the ECHO command must begin each line of the message.

   The @ symbol can be used to suppress the echoing of a single command when ECHO is on. Place the @ symbol first on the command line. Refer also to the description of the @ command for additional information.

Examples

   ECHO This batch file moves files
   ECHO to another directory.

   Loads the high memory area device driver and then loads ROM-DOS into the HMA for a batch file message with more than one line.

   ECHO OFF

   Sets the ECHO to off.
The ERASE command deletes a specified file or set of files.

**Syntax**

```
ERASE [drive:]path filename [/P]
```

**Remarks**

The DEL command and the ERASE command are functionally identical.

When no drive is specified, the default drive is assumed. When no path is specified, the default path is assumed.

Global filename characters ? and * can be used in the `filename`. This should be done with caution as it is possible to delete multiple files unintentionally.

When the `filename` `*.*` is used to delete all files in the specified subdirectory, a verification message is displayed.

*Are you sure (Y/N)?*

Enter Y to erase (delete) all the files in the specified subdirectory.

**Caution:** ROM-DOS has no command to unerase a file. Although utilities exist that can attempt an unerase, certain factors can cause the erased file to be partially or totally lost. The ERASE command should be treated as a permanent erase.

ERASE deletes files within a subdirectory, not the subdirectory itself. For subdirectory removal, refer to the RMDIR command.

**Examples**

```
ERASE MYLETTER.DOC  
MYLETTER.DOC, Delete (Y/N)?
```

Erases the file `MYLETTER.DOC` from the current default subdirectory.

```
ERASE *.DOC  
```

Erases all files in the current subdirectory with a .DOC file extension.

```
ERASE A:*.*DOC /P  
```

Erases all files on the A: drive with a .DOC extension. Before each file is erased, you are prompted to determine whether that file should be erased.
**EXIT**  
**Internal Command**

The EXIT command exits a secondary nested ROM-DOS operation and returns control of the system to the primary program.

**Syntax**

```
EXIT
```

**Remarks**

The EXIT command has no affect if a secondary COMMAND.COM command processor has not been loaded since the primary COMMAND.COM is always loaded in a permanent mode. A secondary COMMAND.COM is affected if it is loaded without the /P permanent option.

---

**FCBS**  
**Internal Command**

The FCBS command allows you to specify the number of File Control Blocks (FCBs) open at one time.

**Syntax**

```
FCBS = number [,minimum number]
```

**Remarks**

Number specifies the maximum number of FCBs open at any given time. The default for this value is 4. The value for number must be in the range from 1 to 255. The minimum number specifies the minimum number of FCBs to be open at all times. The minimum number argument has the same default and range value as the number argument.

**Example**

```
FCBS = 8, 4
```

Sets the maximum number of FCBs to 8 and leaves at least 4 open at all times.

---

**FILES**  
**CONFIG.SYS Command**

The FILES command specifies the maximum number of files that may be open at one time.

**Syntax**

```
FILES = number
```

**Remarks**

The number of files includes the standard files, stdin, stdout, stderr, stdprn, and stdaux. The minimum value is 8, and the maximum is 255. All other values are ignored.

**Example**

```
FILES = 10
```

Specifies the maximum number of open files to ten.
FOR

The FOR command allows repeated execution of a ROM-DOS command applied to a set of files.

Syntax

FOR %%variable IN (set) DO command %%variable

Remarks

During execution, this command attaches the variable as an identifier to each file in the set of files described; it then applies the command to each of these identified files. The set may be an exact list of completefilenames or a global file specification using wildcard characters.

The FOR subcommand can be used directly on the command line and within a batch file. To use on the command line, substitute a single percent (%) symbol for the double percent signs (%%).

Examples

FOR %%N IN (Q1.TXT Q2.TXT) DO PRINT %%N
Prints only the files Q1.TXT and Q2.TXT.

FOR %%N IN (*.TXT) DO PRINT %%N
Prints all files in the current default directory with a .TXT extension.

GOTO

The GOTO subcommand transfers control to another line of the batch file.

Syntax

GOTO label

Remarks

The label is another line in the batch file consisting of a string up to eight characters long. The label may be an environment variable.

If the specified label is not found, then the batch file terminates with the error message Label not found.

Examples

GOTO MESSAGE
This command moves the control of execution within the batch file to a line that says.

:MESSAGE

Note: A batch file label must be preceded by a colon (:).
HELP

Internal Command

The HELP command provides on-line help of each ROM-DOS command.

Syntax

HELP command

Remarks

The command argument refers to any ROM-DOS internal command or external utility.

The COMMAND.HLP file must be available (i.e., in the path) to use this command. If it is not available, an error message occurs, indicating that COMMAND.HLP cannot be found.

HELP serves as a memory aid. For complete information about ROM-DOS commands, always consult this manual.

HELP for each command can also be displayed by entering a /? following the command name.

All available batch file commands are also listed by HELP.

Examples

    HELP DIR
    DIR /?N

Both commands list the help of the DIR command.

IF

Batch File Command

The IF subcommand allows conditional execution of commands.

Syntax

    IF [NOT] condition command

Remarks

The condition may be any one of the following:

ERRORLEVEL number

string1 == string2

EXIST [drive:] [path]filename

If the condition is true, then the command is executed. Otherwise the command is bypassed, and the next command in the batch file is executed. The [NOT] option tests the opposite of any condition.

The ERRORLEVEL number is true if the last program to execute had an exit code equal or greater than number. Using the [NOT] option with this condition tests if the exit code is less than the number argument.
The condition `string1 == string2` is only true when `string1` and `string2` are identical. The strings must match exactly; uppercase/lowercase mismatches are not allowed. Applying the [NOT] option creates a condition that is true only when the strings are not identical.

The EXIST `filename` is true if the specified `filename` is found. Wildcard characters are allowed in the `filename`. The [NOT] EXIST condition is true when the `filename` cannot be found.

**Examples**

```plaintext
IF ERRORLEVEL 15 GOTO EXIT
Will GOTO the :EXIT label if the ERRORLEVEL was equal to or greater than 15.

IF %1 == CONFIG.SYS PRINT %1
Prints the file stored as the %1 parameter only if its exact name is CONFIG.SYS.

IF NOT EXIST OLD COPY CONFIG.SYS OLD
Copies CONFIG.SYS to OLD if a file named OLD does not exist.
```

**INCLUDE**

**CONFIG.SYS Command**

The INCLUDE command includes the contents of one configuration block into another. The instructions from the originating instruction block, as well as the included block, are carried out. This command can only be used within a CONFIG.SYS configuration block.

**Syntax**

```plaintext
INCLUDE = blockname
```

**Remarks**

This command is useful for sets of instructions common to several system configurations. The commands are defined once in a single configuration block and then inserted into other configuration blocks via the INSERT command. For additional details, refer to ‘Using Multiple-User Configurations’ on page 24.

**Examples**

```plaintext

[MISC]
device=mouse.sys
device=c:\netword\loadnet.sys

[WORDPROC]
files=20
buffers=10
set path=c:\bin;c:\wp;c:\dict
INCLUDE=MISC
```

```
When you choose WORDPROC from a CONFIG.SYS menu, the instructions in the configuration block labeled [WORDPROC] are carried out. The instructions in the INCLUDED block labeled [MISC] are also implemented as part of the [WORDPROC] block of instructions.

**INSTALL**

The INSTALL command loads Terminate and Stay Resident (TSR) programs during CONFIG.SYS processing.

**Syntax**

```
INSTALL = [drive:]path TSR_Program TSR_Arguments
```

**Remarks**

The TSR program is loaded the same as if loaded from AUTOEXEC.BAT, except that an environment is not created. The lack of an environment may cause some programs to execute incorrectly. These programs must be loaded from the AUTOEXEC.BAT file.

**Examples**

```
INSTALL = C:\BIN\MOUSE.COM
```

Loads a mouse driver from CONFIG.SYS using INSTALL. Command line arguments can be included.

**LASTDRIVE**

The LASTDRIVE command sets the maximum number of drives.

**Syntax**

```
LASTDRIVE = letters
```

**Remarks**

`letter` may be any character between A and Z and is the last drive letter that ROM-DOS can access. The default value for `letter` is E.

The minimum number for LASTDRIVE is the number of drives in your computer. If `letter` is less than number of drives in your computer, then the LASTDRIVE command is ignored.

LASTDRIVE is often used to cause ROM-DOS to make more space for nonstandard drives that are not in your system. These drives may be CD-ROM drives, flash disk drives, or network drives.

**Examples**

```
LASTDRIVE = M
```

Causes ROM-DOS to allocate space for eight drives. If the computer has five drives installed, there is room for three additional nonstandard drives.
LFNFOR

Batch File Command

The LFNFOR command enables/disables long file names when processing FOR commands.

Syntax

LFNFOR [ON | OFF]

LOADHIGH

CONFIG.SYS, Internal Command

The LOADHIGH command loads an executable or TSR program into the upper memory area, if available. LOADHIGH can be run as a batch file command or from the DOS command line.

Syntax

LOADHIGH = [drive:]\path\executable [arguments]
-or-
LH = [drive:]\path\executable [arguments]

Remarks

An executable or TSR program can be loaded into the upper memory areas when they are available and have enough free upper memory to accommodate the program’s needs. To make high memory available, the EMM386.EXE and HIMEM.SYS utilities must be loaded. If these utilities are not loaded or there is not enough upper memory available, the program loads into conventional memory.

The full drive path and filename of the device must be specified. The arguments are different depending on the device driver.

Examples

LOADHIGH=C:\\apps\checkit.exe /E

Installs an executable named CHECKIT with its command line arguments as specified. The program loads into upper memory, if available.

MENUCOLOR

CONFIG.SYS Command

The MENUCOLOR command allows you to set the text and background colors for the startup menu. This command can only be used in a menu block within your CONFIG.SYS file.

Syntax

MENUCOLOR = text_color [,background_color]
Remarks

The `text_color` argument selects the display color for the screen text. The color numbers 0 to 15 can be selected from the list below for the text color.

The `background_color` argument is optional. If a value is not entered, the default color 0 (Black) is used. Be sure to specify different colors for background and text, and separate the numbers with a comma. For best results, choose contrasting colors. Only the color numbers 0 to 7 can be used as the background color designation.

For systems whose BIOS does not directly support a video display, such as Datalight's miniBIOS, the standard CONFIG.SYS menu commands, which rely on BIOS screen support, are unusable. To use these commands, the color number sequence of 0 for `text_color` and the default background color (black), or 0,0 for text and background colors can be selected. These numbers represent a color choice of black text with a black background, which is an unusable choice for screen viewing. Using the black/black combination in the MENUCOLOR command line signifies to ROM-DOS to display the startup menu in TTY mode without using BIOS screen/cursor positioning or color changing commands.

**Color values:**

- 0 – Black  
- 3 – Cyan  
- 6 – Brown  
- 9 – Bright Blue  
- 12 – Bright Red  
- 15 – Bright White  
- 1 – Blue  
- 4 – Red  
- 7 – White  
- 10 – Bright Green  
- 13 – Bright Magenta  
- 2 – Green  
- 5 – Magenta  
- 8 – Gray  
- 11 – Bright Cyan  
- 14 – Bright Yellow

**Examples**

`MENUCOLOR=14, 1`

Displays the menu text in bright yellow on a blue background.

`MENUCOLOR=5`

Displays the menu text in magenta with a default background of black.

---

**MENUDEFAULT**

`CONFIG.SYS Command`

The MENUDEFAULT command allows you to set the default menu-item choice and a time-out value for making a menu selection. This command can only be used within a menu configuration block in the CONFIG.SYS file. Refer also to ‘Using Multiple-User Configurations’ on page 25.

**Syntax**

```
MENUDEFAULT = blockname[.timeout]
```
The `MENUITEM` command allows you to specify an item on the startup menu. This command can only be used within a menu configuration block in the `CONFIG.SYS` file. Refer also to 'Using Multiple-User Configurations' on page 25.

**Syntax**

```
MENUITEM = blockname [,menu_text]
```

**Remarks**

The `blockname` argument specifies the default menu item. The value for `blockname` must match a configuration block name defined elsewhere in your `CONFIG.SYS` file.

The optional time-out argument represents the number of seconds ROM-DOS waits for a user input selection before initializing your system with the default configuration. The time-out period can be set to a value between 0 and 90. If you select 0, the default menu item is automatically implemented without a wait. If you do not enter a time-out value, ROM-DOS will not continue until the Enter key is pressed.

If your system BIOS does not support a video display directly, such as Datalight’s miniBIOS, please refer to the `MENUCOLOR` command for special instructions.

**Examples**

```
[MENU]
menuitem=Word_Proc, Word Processing
menuitem=Network, Network
menuitem=Research, Research and Development
menucolor=15,1
menudefault=Word_Proc,20
```

Makes the `Word_Proc` configuration block the default menu item. If you fail to make a selection within 20 seconds, the `Word_Proc` block is processed.
be up to 70 characters long and can contain any characters. If this argument is left off, the blockname is used for the startup menu display.

If your system BIOS does not support a video display directly, such as Datalight’s miniBIOS, please refer to the MENUCOLOR command for special instructions.

Examples

[MENU]
menuitem=Word_Proc, Word Processing
menuitem=Network, Network
menuitem=Research, Research and Development
menudefault=Word_Proc,20

Defines three menu items: Word_Proc, Network, and Research and Development. Each of these has descriptive text and a set of commands defined later in the CONFIG.SYS file. At boot time, these menu items are displayed in the startup menu as follows:

ROM-DOS 6.22 STARTUP MENU
1. Word Processing
2. Network
3. Research and Development
Enter a choice: 1

MKDIR (Make Directory)

The MKDIR command creates a new subdirectory.

Syntax

MKDIR [drive:]\path\subdir

MD [drive:]\path\subdir

where subdir is the name of the new subdirectory to be created. Note that MD may be used instead of MKDIR

Remarks

If no drive or path is specified, the new subdirectory is created within (one level below) the current default directory.

If drive and/or path is specified, everything specified must exist or the command displays an error message.

Examples

MKDIR TEMPDIR1
Creates a new subdirectory named TEMPDIR1 within the current default directory.

MD C:\UTIL\TOOLS
Creates a new subdirectory named TOOLS within UTIL, assuming the subdirectory exists.
NEWFILE

The NEWFILE command allows you to continue CONFIG.SYS file processing from a new file. The file can be located in another directory or even on a different drive.

Syntax

\texttt{NEWFILE=filename[,device]}

Remarks

The NEWFILE command is especially useful when the CONFIG.SYS file is located on an inaccessible drive or in ROM. Additional device drivers or instructions can be added easily to the new file and is processed along with the main CONFIG.SYS file upon starting the system.

When the NEWFILE= instruction is processed, control passes from the present file (the one containing the NEWFILE instruction) to the file specified in the command. Any commands placed after the NEWFILE instruction in the original file are not processed.

If the specified filename cannot be located, CONFIG.SYS processing returns to the original CONFIG.SYS file and the next instruction is processed. The next instruction can even be a second NEWFILE= statement, allowing flexibility for systems which may have a variety of drives installed or not installed at boot time. The first successful NEWFILE statement is processed, transferring control to the specified filename. The remaining instructions in the original CONFIG.SYS file, including NEWFILE= statements, are not processed.

NEWFILE commands can be nested. That is, your original CONFIG.SYS can call a second set of instructions via the NEWFILE command. The second file can, in turn, call a third file by using the NEWFILE command, and so on. Be sure that each filename in the successive steps has a unique name, otherwise, you will create an infinite loop as control is passed back to the same file repeatedly.

When NEWFILE is used, it is also created as an environment variable, retaining the value assigned by the NEWFILE= statement. This can be used in an AUTOEXEC.BAT file for further boot-up decision-making.

The presence of [\textit{device}] on the NEWFILE command line will tell the ROM-DOS kernel to load this device driver before trying to process this NEWFILE statement. The format of this parameter is exactly the same as the parameter for a \texttt{DEVICE=} statement. See page 47 for a description of the \texttt{DEVICE=} statement.

Examples

\texttt{NEWFILE=C:\BIN\NEWCFG\SYS}

Causes instructions in the file NEWCFG.SYS, located in the C:\BIN directory, to be executed as part of the CONFIG.SYS file.

\texttt{NEWFILE=C:\BIN\NEWCFG\SYS,ROMDRIVE\SYS C000}

Causes the device driver, ROMDRIVE.SYS, to be executed prior to processing the instructions in the file NEWCFG.SYS as part of the CONFIG.SYS file.
Also, the environment will contain the following entry:

NEWFILE=C:\BIN\NEWCFG.SYS

Verify this by running the SET command with no parameters.

**NUMLOCK**

The **NUMLOCK** command sets the NumLock key on the keyboard to **ON** or **OFF** when your computer starts.

**Syntax**

```
NUMLOCK=[on|off]
```

**Remarks**

Selecting **ON** designates that the NumLock key is set to on when DOS boots. Selecting **OFF** designates that the NumLock is off when DOS boots. In either case, you still have the ability to manually turn the NumLock key on and off after booting with the **NUMLOCK** command.

**Examples**

```
NUMLOCK=on
```

Sets the NumLock key to on when the system boots.

**PATH**

The **PATH** command sets the search path for command files that are not in the current directory.

**Syntax**

```
PATH [drive:][path][;[drive:][path]] ...
```

**Remarks**

Without a specified search path, ROM-DOS looks for an external command file (one with a .BAT, .COM, or .EXE extension) only in the current directory. The **PATH** command tells ROM-DOS which other directories to search after searching the current directory.

To append one or more additional directories to the **PATH**, use `%PATH%` for the current path. For example, to add C:\DOS to the current path, enter:

```
PATH %PATH%;C:\DOS
```

at the command prompt.

Enter **PATH** without parameters to display the current path. To cancel previously-set command paths, enter:
PATH =
or
PATH =;

Examples
If your application programs reside on a fixed disk, the PATH command enables you to start any of them from any drive or directory. To access utilities, a word processor, and a spreadsheet in subdirectories C:\UTIL, C:\WP, and C:\123, set the path command as follows:
PATH C:\UTIL;C:\WP;C:\123

PAUSE

The PAUSE command suspends the execution of a batch file and resumes operation when any key is pressed.

Syntax
PAUSE [message]

Remarks
A batch program may require that you perform some action such as changing disks or choosing to continue or terminate the operation. When the command processor encounters PAUSE, it suspends execution and displays the message
Strike a key when ready...

After you perform the appropriate action, or make a decision, striking any key other than the combinations Ctrl+C or Ctrl+Break resumes the batch job.

If you press Ctrl+C or Ctrl+Break at this point, ROM-DOS displays
Terminate batch job (Y/N)?
Responding Y ends the batch job. Therefore, strategic placement of the PAUSE command allows you to divide the batch file into sections so you can end it at some intermediate point.

The message option allows you to display a reminder on the screen during the pause. Your message precedes the “Strike a key” message. Note, however, that your message appears only if ECHO is on.

Examples
PAUSE Place blank disk in drive A:
Prompts the user to insert a disk and suspends operation until a key has been hit.
**PROMPT**

The PROMPT command changes the ROM-DOS command prompt.

**Syntax**

```plaintext
PROMPT [text] [$character] [$character...]
```

**Remarks**

The prompt that ROM-DOS normally displays is the letter of the current drive followed by a right angle bracket (>) (the greater-than symbol). By using the PROMPT command, you can change this prompt to include any combination of a message, the current directory, the date, the time, and some other features.

<table>
<thead>
<tr>
<th>Code</th>
<th>Corresponding Prompt</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T</td>
<td>Current time</td>
</tr>
<tr>
<td>$D</td>
<td>Current date</td>
</tr>
<tr>
<td>$P</td>
<td>Current drive and path</td>
</tr>
<tr>
<td>$V</td>
<td>ROM-DOS version number</td>
</tr>
<tr>
<td>$N</td>
<td>Current drive</td>
</tr>
<tr>
<td>$G</td>
<td>The &gt; character</td>
</tr>
<tr>
<td>$L</td>
<td>The &lt; character</td>
</tr>
<tr>
<td>$B</td>
<td>The</td>
</tr>
<tr>
<td>$Q</td>
<td>The = sign</td>
</tr>
<tr>
<td>$H</td>
<td>A backspace (which erases the previous character)</td>
</tr>
<tr>
<td>$E</td>
<td>ASCII code for Escape (X'1B')</td>
</tr>
<tr>
<td>$_</td>
<td>Start a new line (carriage return)</td>
</tr>
<tr>
<td>$$</td>
<td>The $ character</td>
</tr>
</tbody>
</table>

**Examples**

To show this prompt

```
Current directory is drive:\path;
Ready for <command>
```

enter

```
PROMPT Current directory is $P;$_Ready for $Lcommand$G
```

To show the date, time, and current directory on separate lines followed by the greater-than character and a space, enter:
PROMPT $D$_$T$_$P$G<space>

where <space> refers to pressing the spacebar once. The resulting prompt is
Mon 6-26-1989
10:17:45.99
A:\>

REM (REMark)

The REM command has two purposes: to allow comments in a batch or CONFIG.SYS file, and to temporarily disable a command without physically deleting the command from the file. See also the (;) command.

Syntax
REM [message]

Remarks
The REM command provides information but has no affect on the execution of the batch file.
The comment may consist of any set of characters. You may also create a blank line by omitting the message portion of the line.
REM can also be used to temporarily disable a command in a batch file or CONFIG.SYS without having to delete the line from the file.

Examples
REM This batch file created by
REM Jane Doe
These lines may be added at any point in a batch file as user information only.
DEVICE=HIMEM.SYS
DOS=HIGH
REM DEVICE=TESTDEV.SYS /P
Temporarily removes the DEVICE=TESTDEV.SYS statement from these CONFIG.SYS instructions. This statement is not processed again until REM is removed.
**REN (REName)**

The REN command changes the name of a file.

**Syntax**

```
REN [drive:]path]filename1 filename2
```

**Remarks**

REN renames files within a directory; it does not move a file to a different drive or directory as part of the command.

The wildcard characters * and ? may be used to rename more than one file at a time.

ROM-DOS does not allow you to give a file a name that matches the name of an existing file in the same directory.

**Examples**

```
REN B:NOTES.DOC REPORT.DOC
```

Renames the file NOTES.DOC in drive B: to REPORT.DOC.

```
REN *.DOC *.TXT
```

Assigns the extension .TXT to all files with the current extension .DOC.

---

**RMDIR (ReMove DIRectory)**

The RMDIR command removes (deletes) a specified empty subdirectory.

**Syntax**

```
RMDIR [drive:]path]subdir
RD [drive:]path]subdir
```

*subdir* is the name of the subdirectory being deleted. Note that RD may also be used.

**Remarks**

If no drive or path is specified, RMDIR looks for the specified subdir within (one level below) the current default directory. If a drive or path is specified, everything specified must exist or ROM-DOS displays an error message.

RMDIR does not remove a subdirectory unless it is empty. An error message is displayed when you attempt to remove a subdirectory that still contains files or other subdirectories.

**Examples**

```
RD TOOLSC
```

Removes the TOOLS subdirectory from the current directory, assuming TOOLS is an empty directory.
SET

The SET command sets, displays, or removes environment variables.

Syntax

```
SET [variable = [string]]
```

Remarks

Use the environment variables to control the behavior of programs and batch files and also the behavior of ROM-DOS. Use this command in the AUTOEXEC.BAT and CONFIG.SYS files and on the DOS command line. The environment variables that can be defined with the set command include, but are not limited to, PATH, COMSPEC, PROMPT, and user-defined variables.

Using SET `variable =` with no argument string clears the current environment string for the named variable.

Examples

```
SET PROMPT = $p$g
```

Sets the prompt, although the prompt can also be set with the PROMPT command.

```
SET PROMPT =
```

Clears any previously set prompt settings and returns the prompt to its default state.

SHELL

The SHELL command allows you to specify a command interpreter other than the default COMMAND.COM or to load COMMAND.COM with non-default arguments (parameters). ROM-DOS boots this new program, with arguments, instead of that specified internally.

Syntax

```
SHELL = cmd_interpreter arguments
```

Remarks

The SHELL command is most often used to start the initial copy of COMMAND with special parameters. One parameter provides a larger environment than the default 128 bytes.

The `cmd_interpreter` can be any executable program. The full path, including drive letter, should be specified if the program is not in the root directory of the default drive.

Arguments are optional and program-specific and vary depending on the `cmd_interpreter` being executed by the SHELL command.
Examples

SHELL=C:\COMMAND.COM /E:512 /P

Boots the standard Command Processor but sets the environment space to 512 bytes (up from the default 128). The /P parameter tells COMMAND that it is permanent (cannot terminate).

SHELL = C:\TEMP\MYPROG.EXE

Boots a program named MYPROG.EXE, located in the directory TEMP, instead of the standard Command Processor.

SHIFT

Batch File, Internal Command

The SHIFT command moves each replaceable parameter for a batch file one position to the left. Execution of the SHIFT command allows use of more replaceable parameters in a batch file beyond the standard set of %0 through %9.

Syntax

SHIFT

Remarks

This command moves the string or value stored for each replaceable parameter one position to the left. Upon execution of SHIFT, the %0 argument assumes the value of the %1 argument, the %1 argument then assumes the value of the %2 argument, and so on.

Example

The following batch file reads in a list of files (provided as arguments on the command line) and displays each one to the screen. After displaying each one, the SHIFT command copies the next file in the argument list into the %1 slot, verifying the existence of the file, and continues.

Command line argument:
TYPEIT autoexec.bat config.sys net.bat

TYPEIT.BAT batch file:
:repeat
if EXIST %1 goto doit
goto end

:doit
type %1
pause shift
goto repeat

:end
@echo All Done
STACKS

The STACKS command enables the dynamic use of data stacks to handle hardware and software interrupts that use large amounts of stack space. You may use this command only in your CONFIG.SYS file. Use the STACKS command if the system crashes or encounters a stack overflow during the boot or at runtime.

STACKS uses more RAM for the DOS stacks, which you can calculate with the formula (number-of-stack*stack-size). The maximum extra DOS stack size is 32KB (64*512).

**Syntax**

```
STACKS = n,s
```

**Remarks**

- `n` specifies the number of stacks. Valid values for `n` are 0 and integers in the range 8 through 64.
- `s` specifies the size (in bytes) of each stack. Valid values for `s` are 0 and integers in the range 32 through 512.

SUBMENUS

The SUBMENU command defines a menu item that represents a secondary menu when selected. This command may only be used within a menu configuration block in the CONFIG.SYS file.

**Syntax**

```
SUBMENU = blockname[,menu_text]
```

**Remarks**

The `blockname` argument defines the name of the secondary menu block of commands. The block menu must be defined elsewhere in the CONFIG.SYS file, otherwise, ROM-DOS leaves this item off of the startup menu. The label can be up to 70 characters long and can contain most printable characters, including spaces, backslashes (\), forward slashes (/), commas (,), semicolons (;), and equal signs (=). Square brackets ([ ]) cannot be used in blocknames.

The optional `menu_text` argument specifies the text that ROM-DOS displays for this menu item on the startup menu. If this argument is left out, ROM-DOS displays the `blockname` as the text. The `menu_text` can be up to 70 characters long and may contain any character.

The submenu can be defined with any user-provided descriptive label. It need not have the [MENU] label.
In the preceding example, a submenu is defined as one of the startup menu choices. When you select Research and Development from the first menu, a secondary menu is displayed, offering the choices of Project 1 and Project 2. The actual commands for Project 1 and Project 2 are defined in the configuration blocks labeled PROJ1 and PROJ2.
SWITCHES

The SWITCHES command allows special CONFIG.SYS file options.

Syntax

```
SWITCHES=[/k][/n][/f]
```

Remarks

The `/k` argument makes an enhanced keyboard behave like a conventional keyboard.

The `/n` argument prevents the use of the F5 and F8 function keys to bypass the startup commands.

The `/f` argument instructs ROM-DOS to skip the delay after displaying the “Starting ROM-DOS...” message at boot time. The delay allows you time to use the F5 and F8 options to alter the processing of the CONFIG.SYS and AUTOEXEC.BAT files as described in page 26.

Example

```
switches = /n
```

Prevents you from using the F5 and F8 keys at boot time.

TIME

The TIME command displays the current time as shown on the system's internal clock. Allows resetting of the clock.

Syntax

```
TIME [hh:mm:ss] [pm|am]
```

Remarks

The time set by this command is used, among other things, for time stamping your file revision dates. This information is displayed when you execute a directory listing of your files.

You may want to include the TIME command in your AUTOEXEC.BAT file to set the date at boot time. If your computer has an internal, battery-operated clock, you won’t need to do so.

The format of the time command is also dependent on the country specified in CONFIG.SYS. The time is displayed according to local standards for the specified country. Refer also to the DATE command.

If you just want to check the time maintained by ROM-DOS, enter the TIME command alone. ROM-DOS displays something like this

```
Current time is 3:00:02.48p
Enter new time: _
```
After which you press Enter to return to an empty command line.

When you want to change the time, you can include the desired time on the prompt line after the word TIME. Or you may enter the command with no option (as you do to check the time) and enter the new time before pressing Enter.

ROM-DOS displays the time according to the 24-hour clock with the a or p indicator to show AM or PM. The AM / PM indicator can be entered as a or p or as am or pm. The time may be entered in a 24-hour format or a 12-hour format with the AM or PM designators.

The allowed options for hours and minutes are

\[ hh = 0-24 \quad mm = 0-59 \quad indicator = a, p, am, or pm \]

ROM-DOS displays time to hundredths of seconds. When entering time, however, you needn’t enter seconds or hundredths; ROM-DOS assumes a value of zero if they are not specified.

You may skip the display and prompting by typing the current time after the word TIME on the command line.

TIME 23:24

ROM-DOS accepts your entry as the current time.

**Example**

To set the time to 11:15 p.m., enter

TIME 23:15 or TIME 11:15 p

To set the time to 9:26 a.m., enter

TIME 9:26 or TIME 9:26 am

---

**TRUENAME**

*Internal Command*

The TRUENAME command displays the actual path of virtual drives created with SUBST, network drives, normal drives, and the contents of environment variables.

**Syntax**

TRUENAME [path |% environmentvariable%]

**Remarks**

TRUENAME will display the full path to a network drive or a drive created with SUBST.

If TRUENAME is not given a path or environment variable it will display the full path of the current drive up to the current directory.

Long directory or file names will be truncated to 8.3 format.

Network drives will be expanded to show the drive letter in //SERVER/VOLUME: format.
Option

The %environmentvariable% option will cause TRUENAME to display the contents of the environment variable passed to truename.

Examples

TRUENAME C:
Displays the full true path of drive C:

TRUENAME G: (where G: is a drive created with SUBST)
Displays the full true path of the actual drive and path that was SUBST’d

TYPE

The TYPE command displays the contents of a text file on the screen.

Syntax

TYPE [drive:]path]filename [/P]

Remarks

The drive and/or path arguments must refer to a valid drive and directory combination. The filename may not contain any wildcards.

If a file containing formatting codes or other nonalphanumeric characters is displayed with TYPE, unintelligible characters are displayed. This does not harm the system.

The /P causes a pause after every screen full of text.

Examples

TYPE A:AUTOEXEC.BAT
Displays the AUTOEXEC.BAT file on drive A.

VER

The VER command displays the version number of the ROM-DOS in use and allows revision of this version number.

Syntax

VER [n.nn] [/R]

Remarks

If a new version number is specified, two digits after the decimal are required. Note that this
The VERIFY command displays or modifies the VERIFY state.

Syntax

VERIFY [ON | OFF]

Remarks

The VERIFY command does not perform any data verification (same as the COPY /V option). It is included to provide batch file compatibility.

Examples

VER 5.0

Changes the record of the current DOS version in use to DOS 5.0. Any programs that are executed, following this command, recognize that DOS 5.0 is running.

VOL

The VOL command displays the volume label on a specified disk.

Syntax

VOL [drive:]

Remarks

If you do not specify a drive, the current drive is assumed. VOL does not allow the setting of volume labels. Refer to the LABEL command on page 105 for instructions on setting the volume labels.

Examples

VOL

Causes ROM-DOS to display the volume label on the current drive.

VOL C:

Causes ROM-DOS to display the volume label on the C: drive.
CHAPTER 6 | ROM-DOS UTILITY DESCRIPTIONS

ROM-DOS UTILITIES

Command Descriptions
The following pages provide a complete description of each ROM-DOS external command. Each entry includes a description of the command's purpose, command entry syntax, remarks, and examples as appropriate. External commands are stand alone utility programs and device drivers.

For on-line help information and syntax descriptions, use the /? option with any command. For example:

XCOPY /?

ANSI

ANSI.SYS is a console device driver that allows you to support ANSI codes on the local display.

Syntax

Device=ANSI.SYS [options]

Remarks

ANSI.SYS supports standard ANSI escape sequences.

ANSI.SYS writes directly the screen when using text video mode.

Options

The /K option forces use of the extended keyboard BIOS calls which sense F11 and F12. The /X option lets you redefine the extended keys independently.

The /S option disables the keyboard redefinition feature.

The /Tnn option indicates that the video mode nn is a text mode. By default, modes 0, 1, 2, 3 and 7 are text modes.

Examples

DEVICE=ANSI.SYS
This example loads ANSI.SYS with default settings.

DEVICE=ANSI.SYS /T54 /S
Load ANSI.SYS with mode 54h as a video text mode and disable keyboard redefinition.
ATA

ATA.SYS is a PCMCIA ATA disk device driver.

Syntax

DEVICE = ATA.SYS [/Axxxx] [/Ixxxx]

Remarks

ATA.SYS requires an Intel 82365 or compatible controller. The PCMCIA controller card itself
is always mapped to addresses 3E0h and 3E1h. These are fixed addresses and can not be
changed. This driver supports only 5-volt ATA cards. Up to two drives can be supported with
the driver. Use the Datalight FORMAT command to format an ATA disk, if it is not already
formatted.

ATA Cards tested:
Integral 1841PA 17MB ATA card 660KB/s write, 690KB/s read
Toshiba TH6SS160201AA 20MB ATA card 900KB/s write, 1200KB/s read
SanDisk SDCF B 4MB CompactFlash card 300KB/s write, 1000KB/s read
SanDisk SDP3B 2MB ATA card 275KB/s write, 1000KB/s read

Controllers tested:
Vadem VG-465
Intel 82365SL rev A

Options

The /Axxxx option sets the memory window that the ATA card gets mapped to for use. This is
a 4KB window. The default is C000. Replace xxxx with the correct memory segment window
for your installation.

The /Ixxxx options sets the I/O address where the ATA card is mapped to by the controller.
The default I/O address is 240h. The memory usage for this is 16 bytes starting at the given
address.

Examples

DEVICE = ATA.SYS
Load ATA.SYS with the default memory segment and I/O address settings.

DEVICE=ATA.SYS /AD000 /I290
Load ATA.SYS using memory segment address D000h and I/O address 290h.
ATRIB

The ATTRIB command either displays or modifies the attributes of a file or directory.

Syntax

ATTRIB [+R | -R] [+A | -A] [+S | -S] [+H | -H] [(drive:]path)filename [/S] [-C]

Remarks

The file attributes define the characteristics of a file. They determine if a file may be deleted or modified, or if it is archived. Use the ATTRIB command to manage these file attributes.

Wildcard characters may be used in the ATTRIB filename.

The ATTRIB command modifies file attributes if modify commands are given to ATTRIB. The modify commands are

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+/-</td>
<td>Add(+) or remove(-) attribute inserted before each option.</td>
</tr>
<tr>
<td>A</td>
<td>Archive attribute</td>
</tr>
<tr>
<td>-C</td>
<td>Clear all attributes</td>
</tr>
<tr>
<td>H</td>
<td>Hidden file attribute</td>
</tr>
<tr>
<td>R</td>
<td>Read Only attribute</td>
</tr>
<tr>
<td>S</td>
<td>System file attribute</td>
</tr>
<tr>
<td>/S</td>
<td>Recurse into subdirectories</td>
</tr>
</tbody>
</table>

If ATTRIB finds no modify commands, then it displays the files in the specified directory along with the filenames and their current attributes.

Examples

ATTRIB +r myfile.dat
Adds the Read Only attribute to the file myfile.dat.

ATTRIB -a -r *.dat
Removes the Read Only attribute and the Archive attribute of all files with the .DAT extension.

ATTRIB *.dat
Displays the attributes of all files with the .DAT extension.
BACKUP

The BACKUP command backs up a single directory tree to a floppy drive, hard disk, or network drive.

**Syntax**

BACKUP `srcpath` `dstdrive` [/[A] [/[H] [/[L[:<logname>]]] [/[M]] [/[S]] [/[Y]] [/?]]

**Remarks**

The complement program, RESTORE, restores a backup set to hard disk. Backup creates one or more backup volumes in the form of DL970507.001, where the year, month, and day compose the name, and the volume number is the file extension.

The `<srcpath>` (source path) can be any legal DOS path, with an optional file mask, such as `D:\SOURCE\*.C`.

The `<dstdrive>` (destination drive) can be any legal DOS drive.

**Note**

At the time this manual was updated, this utility did not contain long filename support.

**Options**

The A option appends the data to an existing backup file. The /H option backs up hidden files as well as normal files.

The /L option writes a .LOG file in ASCII text form. If no log filename is specified, BACKUP creates a .LOG file in the current directory.

The /M option backs up only files that do not have the archive bit set. BACKUP automatically clears the archive bit for each file it backs up. Use Datalight ATTRIB to view/set the archive bit manually.

The /S option backs up the entire tree, not just the one directory.

The /Y option allows BACKUP to operate in batch files with no user input (affirmative is given for all prompts). However, if the backup set requires multiple floppies, BACKUP prompts for all floppies after the first one.

**Note**: BACKUP operates much faster while using any disk cache program.

**Examples**

The following command backs up all files in the `C:\DEV\ROMDOS` directory, including subdirectories, to the B: drive.

```
BACKUP C:\DEV\ROMDOS B: /S
```
The **CHKDSK (CHecK DiSK)** command checks the disk directories and File Allocation Table (FAT) and displays a disk and memory report.

**Syntax**

```
CHKDSK [drive:]path[filename][/C][/F][/V][/ ]
```

**Remarks**

CHKDSK examines a disk and determines whether it has any errors in the File Allocation Table (FAT) and optionally fixes errors.

**Options**

The `/C` option allows CHKDSK to correct errors without user confirmation. This option can be used along with the `/F` option for corrections to be made without user confirmation.

The `/F` option causes CHKDSK to fix FAT or directory errors on the disk if any are found. If `/F` is not specified, then CHKDSK acts as if fixing the disk, but no corrections are written to the disk.

If errors are detected, you are prompted with a message similar to the following:

```
15 lost allocation units found in 5 chains.
Convert lost chains to files?
```

If you answer `Y` for Yes, each lost chain is written to a file in the root directory of the current drive. Each file will have the name `FILEnnnn.CHK`. `nnnn` will be a sequential number. The first chain will be in `FILE000.CHK`. These files can be verified to see if they contain valuable information, and then deleted if desired. If you answer `N` for No to the above prompt, CHKDSK still makes the corrections, however, the lost chains are not saved to the disk.

The `/V` option causes CHKDSK to display each path and file as it is processed. The `/` option causes CHKDSK to hide the program signon message.

If a filename is used, then CHKDSK displays all files matching that specification that have noncontiguous data areas on the disk. Files that are stored in noncontiguous areas, especially .EXE files, have slower disk access times. If CHKDSK reports a large number of files with this problem, you should use a utility program that optimizes the files and frees space on your disk.

After checking the disk, CHKDSK displays any error messages followed by a report on the state of the disk that was checked. An example of the report is shown below.

```
Volume ROM-DOS created 11-09-2001 1:00a
Volume Serial Number is 190E-4AA2
362496 bytes total disk space
     0 bytes in 1 hidden files
     6144 bytes in 2 user files
356352 bytes available on disk
```
655360 bytes total memory
595360 bytes free

CHKDSK does not wait for a disk to be inserted before the checking is initiated, nor does it repair any errors.

Examples

CHKDSK a:
Checks the integrity of drive A:. The report is printed to the console.

CHKDSK d: >drive_d.rpt
Checks the integrity of hard drive disk D. The report is saved in a file named DRIVE_D.RPT.

CHOICE

The CHOICE command allows a user to select between different options during the processing of a batch file.

Syntax

CHOICE [/C[:keys] [/N] [/S] [/T[:c,nn]] [Prompt Text]

Remarks

/C[:keys] specifies the allowed keys for the user prompt. The colon is optional for the command syntax. The default selection for the keys is YN. More than two key choices may be entered.

/N prevents display of the user prompt. The selected keys are still valid, but they are not displayed in the prompt message.

/S selects case-sensitivity regarding the prompt. With the /S switch, you must enter the response in the exact case used with the /C option.

/T[:c,nn] sets a time delay. CHOICE pauses for the specified number of seconds waiting for a response. If none is given, the default key choice is used. The argument c is the default-key choice character. The nn argument specifies the number of seconds to pause. Valid number selections are from 0 to 99. A 0 setting produces no pause.

Prompt Text for the prompt is optional. You can display different output by using the text field, or not, and by using or removing the /N switch along with the text.

Examples

When the following CHOICE command is used.

CHOICE /c:yn
you will see:
[Y,N,C]?

If the text argument is added

CHOICE /c:ync Please select Yes, No, or Continue

the display will be:

Please select Yes, No, or Continue [Y,N,C]?

If the key choice prompt is left off

CHOICE /n Continue reading file

only the following appears (note key choice of YN is default)

Continue reading file

A complete batch file example follows. Each prompt selection returns an errorlevel that can be trapped. The errorlevel corresponds to the order of the key choice. For example, with /c:teo, t returns 1, e returns 2, and o returns 3.

@echo off
c:ls
echo.
echo T Run TIME-IN Program
echo E Run Employee Update
echo O Run TIME-OUT Program
echo.
c:\dos\choice /teo /t:t,5 Please select option
if errorlevel 3 goto timeout
if errorlevel 2 goto update
if errorlevel 1 goto timein
:timeout
tmiout.exe
goto end
:update
update.exe
goto end
:timein
tmin.exe
goto end
:end
The COMM communications program provides the ability to communicate with a remote ROM-DOS system. COMM supports Xmodem file transfer, autodialing, Zmodem, and terminal emulation and time zones.

Command Line Options

All command line options must be separated by a space.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/B#</td>
<td>Sets the baud rate to # on startup. The available baud rates are 300, 1200, 2400, 4800, 9600, 19200, 38400, 57600, and 115200.</td>
</tr>
<tr>
<td>/COM#</td>
<td>Sets the communications port to COM1 or COM2; both are supported.</td>
</tr>
<tr>
<td>/8N1</td>
<td>Sets the serial port to 8 data bits, no parity, one stop bit.</td>
</tr>
<tr>
<td>/7E1</td>
<td>Sets the serial port to 7 data bits, even parity, one stop bit.</td>
</tr>
<tr>
<td>/8N2</td>
<td>Sets the serial port to 8 data bits, no parity, two stop bits.</td>
</tr>
<tr>
<td>/8O1</td>
<td>Sets the serial port to 8 data bits, odd parity, one stop bit.</td>
</tr>
<tr>
<td>/8E1</td>
<td>Sets the serial port to 8 data bits, even parity, one stop bit.</td>
</tr>
</tbody>
</table>

You can reverse the placement of the parity and data bits on the command line. For example, /8N1 is equivalent to /N81.

Environment Variables

The environment variable COMM is supported, which may set certain communications parameters. The switches are identical to the command line options, for example:

```
SET COMM= /COM2 /B2400 /7E1
```

Causes COMM to start using COM2, at 2400 baud, with 7 data bits, even parity, and 1 stop bit. If new options are specified on the command line, they override the environment variable settings. Invalid options are ignored.

An additional environment variable for time zones is also supported. The TZ variable allows you to compensate for time zone differences when sending and receiving files and when the time stamp on a received file is critical. This variable only affects transfers done with the Z-modem protocol. X-modem transfers are not affected. By default, if no time zone is set, the Datalight utilities COMM and RSZ assume Greenwich Mean Time (GMT).

Setting the TZ variable is not necessary when using all programs. If you are transferring files between two Datalight utilities, the file time stamp is not affected. Some utilities, such as Windows Hyper-terminal, automatically convert the time stamp on the file to GMT and assumes that the receiving program will adjust the time to the correct local time upon receipt.
The syntax for the TZ environment variable is:

```
SET TZ= <abbreviation> +|- value
```

Abbreviation represents any three-letter abbreviation for the chosen time zone. The variable serves as a reminder to the user. For example, if setting the time zone for Pacific Standard Time, the variable could be set as PST; and for Eastern Standard Time as EST. The abbreviation is only a placeholder in the syntax for the TZ variable. There are no incorrect abbreviation choices as long as only three letters are used.

Value represents the number of hours this time zone varies from GMT. For example, the west Coast of the United States is –8 hours relative to GMT. This value may have to be adjusted to compensate for daylight savings time. There should be no spaces between the abbreviation, plus or minus sign, and the value. Some examples are:

```
SET TZ=PST–8
SET TZ=CMT–3
SET TZ=GMT+2
```

If an incorrect format for the time zone is entered, the default of GMT is used.

On a desktop PC running Windows programs, such as Hyper Terminal, determining the time zone is part of the setup of the operating system. However, if COMM is run from a DOS box, the TZ variable still will need to be set on the target machine.

**COMM Commands**

Enter most commands by pressing an Alt+key combination. Some commands take effect immediately (such as changing the baud rate), while others require further information (such as a filename) before continuing.

If you do not want to execute a command, or want to stop a command while it is running (such as a file transfer), press the Esc key to return to terminal mode.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alt+B</td>
<td>Sets the baud rate. This command toggles between all the available baud rates. Continue to press Alt+B until the desired baud rate appears on the status line at the bottom of the display.</td>
</tr>
<tr>
<td>Alt+C</td>
<td>Clears the display.</td>
</tr>
<tr>
<td>Alt+D</td>
<td>Autodial allows entry of number to be dialed. Press Enter to redial the previous number.</td>
</tr>
<tr>
<td>Alt+E</td>
<td>Toggle echo (duplex). Press Alt+E to toggle the duplex between full (echo off) and half (echo on).</td>
</tr>
<tr>
<td>Alt+H</td>
<td>Hangs up if the modem is capable of hanging up with an ATH0 command.</td>
</tr>
<tr>
<td>Alt+P</td>
<td>Toggles through the available parameters allowing settings to be made. Esc cancels the command.</td>
</tr>
<tr>
<td>Alt+T</td>
<td>When enabled, pressing the Enter key generates a CR/LF instead of just a CR.</td>
</tr>
<tr>
<td>Alt+X</td>
<td>Exits the program. This command does not drop the carrier, so use this command if you need to do MS-DOS operations while online. You can run COMM again without losing the carrier and continue with telecommunications.</td>
</tr>
<tr>
<td>Option</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>PgUp</td>
<td>Sends a file to a remote computer, giving you the option of either Xmodem or ASCII file transfer protocols. Esc cancels at any time during the transfer.</td>
</tr>
<tr>
<td>PgDn</td>
<td>Receives a file using the Xmodem or ASCII file transfer protocol. Press Esc to cancel at any time during the transfer.</td>
</tr>
</tbody>
</table>

**Terminal Emulation**

Currently, COMM supports a subset of the ANSI escape codes that compose the only terminal emulation available. These escape codes, A, B, C, D, H, J, and K, should meet most needs since the emulation includes such features as cursor positioning and erase to the end of the line and/or page.

**File Transfer Recovery with Zmodem Protocol**

The Zmodem file transfer protocol has the ability to resume a set of file transfers at the point of interruption, such as in the case of a call hang-up or disconnected cable. In the event of a failed Zmodem upload or download, run COMM as follows to resume the file transfer:

1. Press the PgDn key to initiate a receive-file operation
2. Select Zmodem in the file transfer option list.
3. Select Y to enable the ZMODEM crash recovery option.

Auto downloads do not use crash recovery.

**COMMAND**

The COMMAND command starts a new command processor.

**Syntax**

```
COMMAND[/e:number][/k filename][/p][/c string][/msg]
```

**Remarks**

Use this command to start a new instance of the ROM-DOS command processor, the program that contains the internal ROM-DOS commands.

Starting a new instance of the command processor also produces a new environment.

The size of the environment is 256 bytes by default but can be changed using the /e switch. The environment size may be larger than the first copy if this is a secondary copy of COMMAND.

Command and its arguments can also be used in a SHELL= statement in your CONFIG.SYS file. Refer to the description of SHELL for more details.
Options
The /e:number switch sets the environment size. Number represents the size of the environment in bytes. Number must be in the range from 160 to 32,768. All other values are ignored and replaced with the default value of 256. ROM-DOS rounds up the value entered to the nearest multiple of 16.

The /k filename option tells the command processor to run the specified filename and then display the ROM-DOS command prompt. It is not recommended that this option be used in a CONFIG.SYS SHELL= statement.

The /p switch causes COMMAND not to exit but to remain permanent. Use the /p switch only when COMMAND is used in a CONFIG.SYS SHELL statement.

The /c string switch causes COMMAND to execute the command in string and then terminate. The string command can be any internal or external command.

The /msg option causes all error messages to be stored in memory. This option is recommended only for floppy disk-based systems. ROM-DOS keeps many of its error messages in the nonresident portion of COMMAND.COM rather than using valuable memory to store them. If an error message is needed, and you have loaded ROM-DOS from a floppy disk, the message will only be available if the boot disk is still in the drive. By using the /msg option, the message is available in memory at all times. The /p option must be used along with the /msg option.

Examples
COMMAND /C DIR C:
Causes a new copy of COMMAND to be executed. It performs a DIR command on the C: drive and then exits back to the previous Command Processor.

SHELL=C:\COMMAND.COM /P /E:256
Loads a permanent copy of COMMAND with an environment size of 256 bytes.

DELTREE
The DELTREE command deletes one or more directory trees or individual files.

Syntax
DELTREE [/Y] [[ ] [drive:]path [[drive:]path[...]]

Remarks
/Y prevents DELTREE from prompting before deleting.
/I hides the program signon message.
[drive:]path indicates the name(s) of the file(s) or directory tree(s) to delete. Wildcards are allowed.
Examples

Datalight DELTREE deletes one or more directory trees. For example, to delete all files and directories in the tree C: \TEMP, enter

DELTREE C: \TEMP.

You can also use DELTREE to delete individual files, one at a time, using wildcards. For example, to delete selected files in the current directory, enter

DELTREE *. *

DELTREE then prompts you for each file it finds, allowing you to choose whether to delete them.

Caution: Be careful when using wildcards with DELTREE. DELTREE deletes all specified files and subdirectories regardless of their attributes. Multiple files and/or subdirectories may be specified.

DISK2IMG

The DISK2IMG command creates a binary image of a drive or disk.

Syntax

DISK2IMG Drive: DestinationPath

Remarks

The drive specifies the source disk.

The DestinationPath is the name and path for the binary image output.

Options

None.

Examples

DISK2IMG A: C: \ FLOPPY. IMG

This command creates a file FLOPPY. IMG containing a binary image of the A: drive.
DISKCOMP

The DISKCOMP command compares contents of two floppy diskettes.

**Syntax**

DISKCOMP drive1: drive2: [/option]

**Remarks**

The drive parameters specify the diskettes to compare.

No matter which drive letters are used, the user will be prompted to insert the diskettes.

The disks that may be compared are 720KB, 1.2MB, and 1.44MB disks. Both the source disk and the target disk must be of the same type.

If any problem occurs during the compare process, DISKCOMP indicates the side, track, and sector where the problem occurred.

The source or the target disk may not be a RAM or virtual disk. DISKCOMP compares only floppy disks, not hard disks.

**Options**

The /1 option compares only the first side of the diskettes.

The /8 option compares only the first eight sectors of each track.

**Examples**

DISKCOMP A: B:

Compares the contents of the diskettes in drive A: onto drive B:

DISKCOMP A: A:

Reads the contents of the diskette in drive A:, then prompts for the user to swap diskettes and compares those contents with the new target diskette.

DISKCOPY

The DISKCOPY command copies the entire contents of one floppy disk to another.

**Syntax**

DISKCOPY drive1: drive2: [/option]

**Remarks**

The first drive specifies the source disk, and the second drive specifies the target disk.
The disks that may be copied are 720KB, 1.2MB, and 1.44MB disks. Both the source disk and the target disk must be of the same type. If the target disk is unformatted, or has a different format, then it is reformatted before copying.

If any problem occurs during the copy process, DISKCOPY indicates the side, track, and sector where the problem occurred.

The source or the target disk may not be a RAM or virtual disk. DISKCOPY copies only floppy disks, not hard disks.

Options

The /1 option copies only the first side of the disk. The /V option verifies the copy after completion.

The /M option forces multi-pass copy using memory only.

Examples

DISKCOPY A: B:
Duplicates the contents of the disk in drive A: onto drive B:. Drive B: must support the same type of disk as drive A:.

DISKCOPY A: A:
Duplicates the contents of the disk in drive A:. You are prompted for swapping the source and target disk as needed to perform the entire copy.

DISPLAY

DISPLAY is a device driver that allows you to view international letters and symbols (code pages) on EGA and VGA displays.

Syntax

device = [drive:][path]DISPLAY.SYS codepage[fontfilename]

Remarks

DISPLAY immediately reconfigures your video adapter to display characters from the selected code page instead of those characters built into the hardware.

The codepage argument specifies the code page you wish to display. ROM-DOS supports code pages 437, 850, 852, 860, 863, 865, and 866.

The fontfilename argument, when included, tells ROM-DOS where to find the EGA.CPI or EGA3.CPI font file. EGA.CPI is the default file choice. If EGA.CPI is in the same directory as CONFIG.SYS, this can be omitted.

Currently, font information for all countries other than Russia and Czechoslovakia can be found in EGA.CPI. Russia’s and Czechoslovakia’s information is contained in EGA3.CPI.
For more information on DISPLAY, see the section on ‘Configuring ROM-DOS for International Use’ on page 30.

Examples

device=DISPLAY.SYS 850
device=C:\DOS\DISPLAY.SYS 850 C:\DOS\EGA.CPI

These examples configure the video adapter to display code page 850. The second example is used if the DISPLAY.SYS and EGA.CPI files are in the C:\DOS directory instead of the same directory as CONFIG.SYS.

DUMP

DUMP is a utility that dumps out the contents of memory or the contents of a file.

Syntax

dump filename [offset] [length]
dump /address [length]

Remarks

filename The name of the file to dump.
offset The offset into the file in hexadecimal to start dumping from.
length The length of bytes in hexadecimal to dump.
address The hexadecimal address in memory to start dumping from.

Examples

Dump myfile.txt 0 20
This will dump the first 20 hex bytes of MYFILE.TXT

EGA.CPI/EGA3.CPI

EGA.CPI and EGA3.CPI are font data files for use with the international video display driver, DISPLAY.SYS. These files contain alternate font sets to display in place of the hardware code page built into the EGA or VGA display monitor.

Syntax

Refer to the instructions for DISPLAY.SYS for usage.

Remarks

Currently, font information for all countries other than Russia and the Czechoslovakia can be found in the file EGA.CPI. Russia’s and Czechoslovakia’s information is contained in EGA3.CPI.
EMM386

The EMM386 device driver enables expanded memory support for systems capable of supporting expanded memory, such as the 386 and higher CPUs. HIMEM.SYS, or another extended memory specifications (XMS) manager, must be installed prior to EMM386.

Syntax (in CONFIG.SYS)

```
device=[drive:] [path] HIMEM.SYS

device=[drive:] [path] EMM386.EXE {I=xxxx-yyyy} {X=xxxx-yyyy}

[FRAME=seg[,memK]]

[ROM=xxxx-yyyy] [D=xxx] [NOMOVEXBDA]
```

**Remarks**

The HIMEM.SYS driver must be loaded for the EMM386 utility to function properly. Datalight’s EMM386 supports Expanded Memory Services (EMS), Upper Memory Blocks (UMB), Virtual DMA Services (VDS), Virtual Control Program Interface (VCPI), and ROM shadowing.

MS-DOS checks for the presence of an XMS UMB provider after it loads any device. If such a device is found, it allocates all available UMBs, builds MCB chains within them, and records the fact that an upper memory chain is available in its SDA. However, ROM-DOS does not function in this manner. The Datalight EMM386 driver builds its own MCB chains within its allocated UMBs and sets up the previously-mentioned data structures with the help of the DOS kernel. Consequently, XMS UMBs that are made available by programs such as MS-DOS’ EMM386.EXE are not recognized by ROM-DOS.

The `I=xxxx-yyyy` option tells EMM386 to include UMBs in the range specified by `xxxx-yyyy`. The `X=xxxx-yyyy` option tells EMM386 to exclude or not to make UMBs in a specified range. EMM386, by default, attempts to make UMBs in the range from C800- F800, excluding areas already occupied by ROM or RAM. Up to eight ranges can be specified. The range can be specified as an exact range, for example, D000 to DFFF or as the starting and ending marks for the range D000 to E000.

The `FRAME=` option defines the starting segment for four 16K pages that can be mapped in and out at will. The optional `memK` argument specifies how much memory to reserve for EMS. The default behavior is to allow all XMS memory to be used as EMS but also to share it, if not in use, with other processes such as Windows. The EMS memory is simulated using extended memory. The segment must be on an even 16K boundary. For example,

```
DEVICE = EMM386.EXE FRAME=E000,512
```

The above CONFIG.SYS statement allocates 512KB of extended memory to be used as EMS memory, with four Windows into that memory at E000:0. By default, EMM386 uses memory as requested for EMS up to the maximum in the system.
The ROM= option specifies a region of ROM to be supported by shadow RAM. This can speed up system performance if the BIOS resides in ROM and does not do its own ROM shadowing. As with all memory ranges, the range must begin and end on 4KB boundaries.

The D= option specifies the amount of RAM in kilobytes (specified as a base-10 number) that is to be reserved for a VDS buffer. The default value is zero. Values for the D= option must be between 16KB and 256KB and are rounded up to the nearest 4KB.

The NOMOVEXBDA option tells EMM386 not to move the Extended Bios Data Area from conventional memory to upper memory.

The MAX=memk option specifies the maximum amount of EMS memory in KB.

The LOW argument prevents the relocation of EMM386 into the first UMB with sufficient size to hold it. EMM386 remains in conventional memory, using approximately 20KB, but reserving UMB areas for use by other drivers.

The PS2 argument forces EMM386 to use the PS2-style (port92h) A20 line control.

The RAM argument is included for compatibility and has no effect on the function or setup for EMM386.

Datalight's EMM386 contains auto detection of BIOS extensions. EMM386 automatically searches for UMBs in the range of C800-F800, similar to MS EMM386. It supports VCPI, VDS, and all Int 67H functions, shared XMS/EMS, and works with MS Windows. Datalight’s EMM386. EXE is significantly smaller than other extended memory managers.

Datalight's EMM386 does not support reallocation of EMS pages. Only four map-able pages are supported. EMS handle 0 is not supported. LiM 3.2 contexts obtained through Int 67h function 47h share eight storage slots with EMS handle names.

**Examples**

DEVICE = EMM386.EXE I=C800-F000
Maps RAM from extended memory into the address C800:0 to just under F000:0 and defines an Upper Memory Block region there. The range could also have been specified as C800-EFFF. When no FRAME= option is supplied, neither EMS nor VCPI services are provided by EMM386.

DEVICE = EMM386.EXE I=C800-D7FF I=F000-F7FF FRAME=E000, 1024
Maps in RAM from extended memory into C800-D7FF and into F000-F7FF and define Upper Memory Block Regions there. Also, EMS support is allowed with four 16KB pages starting at E000:0 with 1024KB (1MG) worth of 16KB pages.

DEVICE = EMM386.EXE ROM=F000-FFFF
Specifies an address range for EMM386 to use for shadow RAM. The ROM represented by the address range is copied into RAM. The RAM area is remapped into the ROM address space and write protected. In this example, the BIOS that normally occupies the 64KB block at F000:0h is copied into RAM and run from there. This may speed up your system if it does not already make use of shadow RAM.
DEVICE = EMM386.EXE D=20
Sets aside 20KB of memory for Direct Memory Access (DMA). Another client, such as a network TSR, can then utilize this memory buffer for disk I/O.

EXE2BIN

EXE2BIN attempts to convert an executable file into a com file.

Syntax
EXE2BIN infile outfile

Remarks
The following must be true for the conversion to be successful: CS=DS=SS (Tiny Model) and there cannot be any segment fixups. Segment fixup addresses are contained in the .exe header so that variable sin the code will be set to contain the segment address which can only be known when DOS loads the file. COM files do not have a mechanism to do fixups.

Examples
EXE2BIN test.exe test.com
Convert the .exe file test.exe into test.com.

FDISK

The FDISK command initializes a hard disk for ROM-DOS to use. FDISK allows you to specify the number of logical drives that are available using a single or multiple physical hard disks. FDISK can be run from its menu or by command line arguments. FDISK supports drives that require LBA support.

Syntax

Options
The drive number option specifies the BIOS drive number to use with other command line arguments. For hard drives, this number is 80h, 81h, and so on, depending on the number of hard drives and partitions in the system.

The /3 option enables FAT32 mode

The /B option creates a FAT16 partition and forces the update of the master boot record code.

The /C option causes FDISK to proceed without confirmation messages.

The /I# argument creates a Super-boot partition with an identification (ID) number of #. 98h is
the default value for #.

The /N option prevents writing of the master boot record code for the Super-boot partitions.

The /R option refreshes the master boot record code only

The /S# option sets the partition size in MB. The default setting creates a partition as large as possible.

The /V argument displays the current partition information then exits.

Remarks

The FDISK command, when entered with no command line arguments, prompts with the command menu:

V) View partition(s)
R) Raw display of partition sectors
C) Create DOS partition(s)
P) Create Super-boot partition
D) Delete a partition
A) Delete all partitions
T) Toggle boot status
M) Write Master Boot Code
Q) Quit without saving
S) Save changes (and reboot)

Enter the letter that corresponds to your choice. If you make a mistake, press Esc to return to the previous menu.

Initialization of a hard disk is usually performed when it is first setup for use. Do this by using choice C (or possibly N). Choice C displays a description of available disk space and a recommendation for using that available space. Follow these recommendations to initialize a hard disk with as many 2GB partitions as possible.

If choice C indicates that the hard disk has partitions already, then they may be deleted if needed. The only reason for deleting partitions is to build smaller-sized partitions or fewer larger-sized partitions.

The M choice writes the code for the master boot record. By default, FDISK only writes the disk partition information. If FDISK finds that the Boot Code is invalid, it automatically writes the boot record. If the existing Boot Code appears valid, it is not rewritten unless you select the M option.

After completing all desired modifications and/or additions, select S to save changes. Once FDISK has modified the disk, the following message appears.

Press any key and ROM-DOS will reboot

After rebooting, each newly initialized drive must be formatted prior to use. Format only those drives that are new or have been resized. Formatting destroys any existing data.
If no changes were made with FDISK, or you wish to abandon the changes made, select Q to quit.

**Examples**

FDISK 81 /R /
Refreshes the master boot record code on drive 81h without confirmation.

FDISK 80 /I98 /S5 /C
Creates a 5MB Super-boot partition with an ID of 98h and without confirmation.

FDISK 80 /
Creates a partition that is as large as possible.

---

**FIND**

The FIND command displays lines, within a disk file, that contain a specified string of characters.

**Syntax**


**Options**

The /C option displays only the count of lines found with the specified `string`.

The /N option displays the line number of the line found containing the `string`.

The /V option displays the lines that do not contain the `string`.

The /I option ignores the case of characters

The `string` argument specifies the string of characters to search for. Double quotes are required around the string.

The `filename` argument specifies the file or group of files to search in. The complete drive and path can be specified. Wildcard characters can be used in the `filename`.

**Examples**

FIND printf junk.c
Displays each line in the file JUNK.C that contains the string printf.

dir | FIND DIR
Displays each line in a directory listing that contains a DIR. The command first executes a DOS DIR command with the output piped into the FIND command. The FIND command then displays each line that contains the DIR `string`.

FIND /C ROM-DOS MANUAL.TXT
.....MANUAL.TXT: 105
Displays a count of the lines in the file MANUAL.TXT that contain the string ROM-DOS.
FORMAT

The FORMAT command initializes a disk so ROM-DOS can access files on that disk. A disk must be formatted before ROM-DOS can use it.

Syntax

```plaintext
```

Remarks

FORMAT initializes the disk and directory of the specified drive. The size of the formatted disk is the largest possible size that the specified drive supports, unless a different size is specified via a command line option.

Options

The `drive:` parameter specifies the drive letter to be formatted.

The `/4` switch causes the floppy disk to be formatted as a 360KB disk even if the drive is a 1.44MB, 2.88MB, or 1.2MB drive.

The `/7` switch causes the floppy disk to be formatted as a 720KB disk even if the drive is a 1.44MB or 2.88MB drive.

The `/C` switch causes FORMAT to format one disk without operator input. The disk is assumed to be in the specified drive, and FORMAT exits immediately when the format is complete. This switch is useful in batch files or programs that require a formatted disk without user input.

The `/F:size` option specifies the size of the floppy disk to be formatted. Available size values are 360, 720, 1.2, 1.44, and 2.88, and are entered as `/F:size`. For example, `/F:1.2`.

The `/H` switch causes the system files not to be hidden or write-protected. This can be used along with the `/S` option.

The `/Q` option causes FORMAT to do a quick format. A quick format reinitializes the disk, deleting each file and subdirectory from the disk. A quick format can only be performed on a previously fully formatted disk.

The `/S` switch causes FORMAT to copy the ROM-DOS system files, ROM-DOS.SYS and COMMAND.COM, onto the disk. The file ROM-DOS.SYS is renamed and stored on the disk as files IBMBIO.COM and IBMDOS.COM, which are stored as hidden files, unless the `/H` option is used.

The `/V:LABEL` switch causes FORMAT to place a volume label on the disk. If the volume label is not provided on the command line, you are prompted for the volume label once the format is complete.

The `/I` switch causes FORMAT to run without display of the sign-on message.
HIMEM

The HIMEM.SYS device driver manages extended memory and the High Memory Area (HMA) in a 286, 386 or greater, or PS/2 systems. HIMEM prevents programs from simultaneously using the same area of memory for two different purposes. HIMEM supports the Extended Memory Specification (XMS) 2.0. HIMEM is installed as a device driver in CONFIG.SYS.

Syntax

```
DEVICE = [drive:]path HIMEM.SYS [/machine:n] [/A20+] [/PS2]
[/CONTROLA20:OFF] [/BIOS] [/QUIET]
```

Remarks

The HIMEM driver can be used to allow ROM-DOS to run in High Memory.

HIMEM supports a default of 32 handles.

HIMEM should not be used with older versions of Datalight’s VDISK. Current versions of VDISK use XMS memory if it is available.

HIMEM recognizes PS/2-style A20 line control and determines whether to use the PS/2 A20 control or the AT A20 control method automatically by calling Int 15h, function C0h (get system configuration). This automatic detection can be overridden with the /Machine:n, /A20, A20+, or /PS2 command line switches in the event that the auto detection on a given system fails.

/Machine:1 and /A20 both designate the PC AT A20 control method. These switches instruct HIMEM not to wait for the A20 line to settle.

/Machine:2 and /PS2 both designate the PS/2 control method.

/A20+ is similar to /A20 but instructs HIMEM to wait for the A20 line to settle.

/Machine:3 designates support for the Phoenix Cascade BIOS A20 control methods.

Alternately, /CONTROLA20:OFF instructs HIMEM to not detect the control method for the A20 line and assumes the A20 line is always on.

The /BIOS switch forces the use of BIOS Int15h, Function 87h, for data transfers to and from XMS memory.

The /QUIET switch forces HIMEM to remove the sign-on message when loading.

Error Conditions

No Extended Memory—An extended memory error condition can occur if the BIOS (via Int 15H, function 88H) notifies HIMEM that there is no extended memory. In this situation, HIMEM displays an appropriate error message and does not install.

Failure to Control the A20 Line—When HIMEM installs, it attempts to control the A20 line, which controls access to the HMA. HIMEM first attempts control via the AT method using the
8259 keyboard control). If that fails, HIMEM then attempts control via the PS/2 method (using I/O port 60H). If both methods fail, HIMEM assumes it can’t control the A20 line and displays the message

A20 Control (OFF)

If either of these errors occur, try using the /A20, /A20+, or /PS2 in the HIMEM command line.

Note also that some older programs assume that the machine is a 1MB 8086 and so require that the A20 line to be disabled (OFF) while they run. Current programs typically do not require that the A20 line be disabled.

Examples

Device = C:\DOS\HIMEM.SYS
Installs the XMS device driver. Once this driver is installed, accessing the HMA and Extended Memory (XMS) memory areas are legal. The Extended Memory area can contain up to 2GB of memory. Typical systems have 4, 8, or 16MB of XMS memory installed.

Device = C:\DOS\HIMEM.SYS /machine:1
Forces the use of the AT-style A20 line control.

The HIMEM driver fails to load when either the machine does not have memory above the 1MB boundary or the BIOS does not provide support for it. It also fails to load when another XMS manager has been previously installed.

KEYB

The KEYB command allows you to alter the keyboard layout for a different language or nationality.

Syntax

KEYB
KEYB countryid
KEYB countryid[codepage][keyboard filename]

Remarks

KEYB is a terminate and stay resident program (TSR). Running KEYB with no arguments shows the current settings of a resident copy of KEYB, if there is one.

The countryid argument is a two-letter code that specifies which country, region, or language is to become current.

When no codepage is included, KEYB uses the default code page for the countryid. You can specify either the default or the alternate code page for any countryid.

The keyboard filename argument tells KEYB where to find its data file (KEYBOARD.SYS or KEYBRD2.SYS). When no keyboard filename is given, KEYB first looks for KEYBOARD.SYS in
the current directory, then in the directory containing KEYB.COM.

Currently, keyboard data for all countries except Russia and Czechoslovakia are found in KEYBOARD.SYS. The file KEYBRD2.SYS contains the data for Russia and Czechoslovakia.

Refer to the table on page 31 for a list of the supported countryid codes, along with their default and alternate code pages.

If a copy of KEYB has already been run, it is reconfigured to the new specifications. While KEYB is active, you can switch back to a U.S. layout at any time by pressing Ctrl+Alt+F1 (Alt+Left-Shift for Russian and Czech Republic keyboards). You can toggle back to the alternate layout by pressing Ctrl+Alt+F2 (Alt+Right-Shift for Russian and Czech Republic keyboards).

Examples

KEYB GR
KEYB GR,437
KEYB GR,C:\TOOLS\KEYBOARD.SYS

Each of these commands establishes a German keyboard layout. The first and third use code page 850, while the second uses code page 437. In the third case, the KEYBOARD.SYS file is located in the C:\TOOLS directory.

KEYBOARD & KEYBRD2

KEYBOARD.SYS and KEYBRD2.SYS are keyboard code page data files for use with the international keyboard driver, KEYB.COM.

Syntax

Refer to KEYB.COM for usage instructions.

Remarks

Currently, keyboard data for all countries except Russia and Czechoslovakia are found in KEYBOARD.SYS. The file KEYBRD2.SYS contains the data for Russia and Czechoslovakia.
LABEL

The LABEL command sets or deletes a disk volume label.

Syntax

    LABEL [drive:] [volume string]

Remarks

The volume string may be up to 11 characters in length. LABEL only uses the first 11 characters of a volume label. The characters that are acceptable in a volume string are the same as those for a filename.

LABEL prompts as follows. Enter the new label and press enter to modify the existing label.

    Volume in drive C is x x x x x x x x x
    Volume label (11 characters, ENTER for none)?
    If a volume label has previously been assigned to a disk, and you do not enter a new volume label, the following message is printed.

    Delete current volume label (Y/N)?
    If the disk did not have a volume label prior to running the LABEL command, the above message will not appear.

Examples

    LABEL a:
    LABEL displays the volume label of drive A; if one exists, and allows it to be modified or deleted.

MEM

The MEM command displays the used and free memory in your system.

Syntax

    MEM [/B | /C | /R]

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/B</td>
<td>Displays each BIOS extension and its size.</td>
</tr>
<tr>
<td>/C</td>
<td>Classifies the memory usage.</td>
</tr>
<tr>
<td>/R</td>
<td>Does raw dump of the MCB chain.</td>
</tr>
</tbody>
</table>
MEM displays a list of the DOS memory contents, what free space is available, and how much memory is in conventional memory, upper memory, the HMA and extended memory. This program is useful to fine tune the system to have as much free memory as possible for applications.

Options

The /B option displays BIOS extensions in the range from C000:0 to F800:0.
The /C option shows program, TSR, and device driver sizes.
The /R option shows a low-level DOS listing of MCBs (Memory Control Blocks).

MODE

The MODE command modifies the operation of the printer, serial port, and active video display.

Syntax

MODE LPT#[:]=COM#[:]
MODE COM#:baud[,parity[,databits[,stopbits[,P]]]]
MODE video mode
MODE display lines

Remarks

The first syntax above redirects line printer output to the serial port.
The second syntax above changes the operation of the specified communications port. The options that can be modified are listed below. Invalid values for any of the options are flagged with an error message.

<table>
<thead>
<tr>
<th>Baud</th>
<th>110, 150, 300, 600, 1200, 2400, 4800, 9600</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parity</td>
<td>N - None, O - Odd, E - Even</td>
</tr>
<tr>
<td>Databits</td>
<td>Either 7 or 8</td>
</tr>
<tr>
<td>Stopbits</td>
<td>Either 1 or 2 stop bits</td>
</tr>
<tr>
<td>P</td>
<td>Printer Port</td>
</tr>
</tbody>
</table>

Using the P option as the last argument causes output to be sent repeatedly to the printer port until successfully received. Without the P, output is sent only once, causing a critical error if unsuccessful.

The third syntax changes the active video mode for the display terminal. The valid choices for this version of the MODE command are as follows:

40 — Indicates 40 characters per line.
80 — Indicates 80 characters per line.
bw40 — For a color graphics adapter with color disabled and 40 characters per line.
bw80 — For a color graphics adapter with color disabled and 80 characters per line.
co40 — Indicates a color monitor with color enabled and 40 characters per line.
co80 — Indicates a color monitor with color enabled and 80 characters per line.
mono — For a monochrome display. Assumes 80 characters per line.

The final syntax sets the number of display lines. Valid values included L25, L43, and L50.

**Note:** A serial port should be initialized before an LPT device is redirected to it.

**Examples**

```markdown
MODE COM1:9600,n,8,1
```
Modifies the settings for the COM1 device to a baud rate of 9600, no parity, eight data bits, and one stop bit.

```markdown
MODE LPT2:=COM2
```
Redirects the output from LPT2 to the COM2 serial port. All following output to LPT2 actually goes to the COM2 device.

```markdown
MODE mono
```
Indicates a monochrome display adapter.

**MORE**

The MORE command displays a text file one screen at a time.

**Syntax**

```markdown
MORE [filename]
or
command | MORE
```

**Remarks**

The input to MORE may come from a file, or it may be piped in from another filter or any other DOS command. If the filename is present, then the file is viewed; otherwise MORE reads from the Standard Input.

Once a screen has been viewed, a line is displayed on the bottom of the screen indicating the percent of the file that has been viewed. At this point, there are several options for the next lines of text to be viewed.

- **B** Display the previous full page
- **<enter>** Display just one more line.
- **T** Display starting at the top of the file.
- **Spacebar** Display the next full page of text.
- **Q** Exit MORE
Examples

DIR | MORE
Displays a directory one screen at a time.

MORE READ.ME
Displays the file READ.ME one page at a time.

MOVE

The MOVE command moves files and renames files and directories.

Syntax

To move one or more files:
MOVE [/Y | /-Y] [drive:]path[filename][,...] destination

To rename a directory:
MOVE [/Y | /-Y] [drive:]pathdirname1 dirname2

Remarks

[drive:]pathfilename1 specifies the location and name of the file or files you want to move.

destination specifies the new location of the file. Destination can consist of a drive letter and colon, a directory name, or a combination. If you are moving only one file, you can also include a filename if you want to rename the file when you move it.

For renaming a directory:

[drive:]pathdirname1 specifies the directory you want to rename.

dirname2 specifies the new name of the directory.

Options

/Y suppresses prompting to confirm creation of a directory or overwriting of the destination.

/Y causes prompting to confirm creation of a directory or overwriting of the destination. The /Y option may be present in the COPYCMD environment variable. This may be overridden with /-Y on the command line.
MSCDEX

The Datalight CD-ROM driver enables CD-ROM drives.

Syntax

MSCDEX [options]

Remarks

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/D:&lt;Name&gt;</td>
<td>CD-ROM device name (default MSC002)</td>
</tr>
<tr>
<td>/E</td>
<td>Use extended memory for CD-ROM buffers</td>
</tr>
<tr>
<td>/H</td>
<td>Displays the help screen</td>
</tr>
<tr>
<td>/K</td>
<td>Allow Kanji with the CD-ROM volumes</td>
</tr>
<tr>
<td>/L:&lt;Letter&gt;</td>
<td>Specifies the drive letter for CD-ROM drive</td>
</tr>
<tr>
<td>/M:&lt;Number&gt;</td>
<td>Specifies the number of sector buffers (default 4)</td>
</tr>
<tr>
<td>/V</td>
<td>Directs MSCDEX to display memory statics when it starts</td>
</tr>
</tbody>
</table>

A low-level driver from the CD-ROM drive manufacturer interfaces to the actual CD-ROM hardware while the Datalight CD-ROM driver provides the interface between ROM-DOS and that hardware driver. This interface is called the Microsoft CD-ROM extensions (or MSCDEX).

For each hardware driver loaded, use the /D: option to specify the name of that driver. The default name is MSC002.

The /L: option specifies the drive letter to use for the CD-ROM drive. The default driver letter is the next available. To increase the number of available drive letters, use the LASTDRIVE command in CONFIG.SYS.

The /M: option specifies the number of buffers to use to speed up CD-ROM drive access.

The /X option puts those buffers in extended memory rather than conventional memory.

Examples

MSCDEX /D:MSC003
Installs the CD-ROM driver using MSCD003 as the device name.

MSCDEX /L:G
Installs the CD-ROM driver using G as drive letter.

MSCDEX /M:6 /X
Installs the CD-ROM driver using six buffers allocated in XMS.
The NED editor is a menu-based text editor available for use with ROM-DOS. This editor is similar to other desktop editors but has special functions designed for use in editing C-source and assembly code.

Starting the Editor

To start the editor, enter

```
NED [filename] [filename]
```

NED may be initiated with or without filename arguments. Wildcard file specifications are allowed.

Up to ten files can be entered on the command line. If NED is run without arguments, it loads all files accessed during the last editing session, returning you to the exact position in the file. You can switch between the open files.

You can also enter

```
NED @errfile
```

where errfile is the name of your compiler error output file. NED loads all files that had errors and allows you to move between errors.

Once NED is running, you may load files into memory by using the File/Open menu command. File/Reload replaces the current file with a new file or reloads a new copy of the same file.

File/Reload confirms before replacing an unsaved file.

Basic Editor Operation

NED uses the standard Windows interface for cut, copy, and paste operations. Del and Shift+Del both move the selected block to the clipboard. There is no true undo command, but Ctrl+V or Shift+Ins may be used to paste the clipboard contents to the current cursor position. Table 1 lists the all the default shortcut keys.

If a search string is all lowercase, NED treats it as a case-insensitive search. If a search string contains any uppercase letters, it is case sensitive. The replacement string is inserted exactly as entered. Repeating a Search command repeats the last Forward or Backward Search operation, not the last Replace operation.

There is one bookmark for all files. Once the bookmark is set, going to the bookmark returns you to the file and position where you set it.

The Indent and Remove-indent (referred to as Undent in the Options/Do Command) commands work on tabs. Indent inserts a tab at the beginning of the current line, or if a block is active, at the beginning of each line in the block. Remove-indent removes the first tab from the current line or from each line in the block. If there are no tabs, Remove-indent has no effect.

Toggle case inverts the case of the current character if no block is active. If a block is active, Toggle case sets the entire block to uppercase if the first character was lowercase, and to lowercase if the first character was uppercase.
Tabs are currently set to 3 for .C, .H, .CPP, .HPP, and .T files. They are set to 8 for all other files.

File/Print prints the current block if there is one, otherwise it prints the current file. NED prompts for a device to print to, which may be a filename. Tabs are expanded to spaces.

The Options/Do Command is intended primarily for debugging. This command allows you to execute any editor command by choosing it from a menu list.

The macro commands (Record Macro/Play Macro) allow you to define a sequence of keystrokes that can be repeated over and over. Select Record Macro (ALT=), enter the keystrokes, then press ALT= again. The macro sequence can be played by selecting Play Macro or by pressing ALT. Keyboard bindings are saved in NED.CFG in the same directory as NED.EXE. NED.CFG also contains the list of active files and positions.

If you record and play a recursive macro, it plays continuously.

If you press an invalid key on a menu, NED operates as if you pressed enter.

If you run out of memory, such as when you have more than 300KB of files open, NED returns to DOS.

**Remote Editing**

NED operates as a full-screen editor, even through a serial port using ANSI Escape codes. Any communication program capable of emulating an ANSI terminal will work with NED in remote mode.

NED automatically detects if the console is redirected through a serial port, either via CTTY, or when a serial console. NED does not support ANSI key codes, so the use of PC function keys and standard PC cursor keys is supported through control keys. To use the special control keys, copy the NEDREMOT.CFG to the name NED.CFG in the same directory that NED.EXE is run from on the target system. This NED configuration file was created using the “Map a key” function (under the Options menu), and can be modified in the same manner.

Always use the Esc key to get to the menus. Use Ctrl+K to enable/disable blocking mode when selecting text. The remote key mapping is provided in the following list.

- Ctrl+A: Left arrow
- Ctrl+B: Find backward
- Ctrl+C: Copy to clipboard
- Ctrl+D: Go to mark
- Ctrl+E: Delete to end of line
- Ctrl+F: Find forward
- Ctrl+G: Go to line number
- Ctrl+H: Delete previous character (same as Backspace)
### Default Hot Keys

Many of the editor commands can be accessed directly by pressing key combinations. For example, press Alt+X to exit the editor and save any open files. The following table lists the default hot keys. You can redefine the commands and keys using the Bind HotKey command available on the Options Menu.

<table>
<thead>
<tr>
<th>Key Combinations</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ctrl+I</td>
<td>Insert tab (same as Tab)</td>
</tr>
<tr>
<td>Ctrl+J</td>
<td>Page down</td>
</tr>
<tr>
<td>Ctrl+K</td>
<td>Toggle block mode (for cutting to clipboard)</td>
</tr>
<tr>
<td>Ctrl+L</td>
<td>Delete the entire line</td>
</tr>
<tr>
<td>Ctrl+M</td>
<td>Insert return (same as Enter)</td>
</tr>
<tr>
<td>Ctrl+N</td>
<td>Toggle insert/overwrite mode</td>
</tr>
<tr>
<td>Ctrl+O</td>
<td>Open a file</td>
</tr>
<tr>
<td>Ctrl+P</td>
<td>Toggle through previous 3 positions</td>
</tr>
<tr>
<td>Ctrl+Q</td>
<td>Home</td>
</tr>
<tr>
<td>Ctrl+R</td>
<td>Search/Replace</td>
</tr>
<tr>
<td>Ctrl+S</td>
<td>Right arrow</td>
</tr>
<tr>
<td>Ctrl+T</td>
<td>Top of document</td>
</tr>
<tr>
<td>Ctrl+U</td>
<td>Page up</td>
</tr>
<tr>
<td>Ctrl+V</td>
<td>Insert clipboard at cursor</td>
</tr>
<tr>
<td>Ctrl+W</td>
<td>Up arrow</td>
</tr>
<tr>
<td>Ctrl+X</td>
<td>Delete to clipboard</td>
</tr>
<tr>
<td>Ctrl+Y</td>
<td>End of document</td>
</tr>
<tr>
<td>Ctrl+Z</td>
<td>Down arrow</td>
</tr>
<tr>
<td>Ctrl+[</td>
<td>Menu/Cancel operation (same as Esc)</td>
</tr>
<tr>
<td>Ctrl+]</td>
<td>Brace match</td>
</tr>
<tr>
<td>Ctrl+\</td>
<td>Do a command (opens a menu with all NED commands)</td>
</tr>
</tbody>
</table>

### Troubleshooting Remote NED

If nothing appears on the terminal screen, check the baud rate of the terminal program, check the serial cable (it should normally be a null-modem cable), and check that the terminal program is set to emulate ANSI escape codes.

In some cases, it is possible for the remote auto-detect to fail. In this case, run the program NEDREMOT prior to running NED. NEDREMOT sets a word at 40:E8h to inform NED to operate remotely.
<table>
<thead>
<tr>
<th>Key</th>
<th>Function</th>
<th>Key</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alt+Q</td>
<td>Quit without saving</td>
<td>F1</td>
<td>Help</td>
</tr>
<tr>
<td>Alt+X</td>
<td>Exit, saving as needed</td>
<td>F7</td>
<td>Load file into current buffer</td>
</tr>
<tr>
<td>Ctrl+A</td>
<td>Search again</td>
<td>F9</td>
<td>Save file</td>
</tr>
<tr>
<td>Ctrl+B</td>
<td>Search backward</td>
<td>F10</td>
<td>Exit asking for save as needed</td>
</tr>
<tr>
<td>Ctrl+C</td>
<td>Copy the block to clipboard</td>
<td>Left Arrow</td>
<td>Left one character</td>
</tr>
<tr>
<td>Ctrl+D</td>
<td>Find the mark</td>
<td>Right Arrow</td>
<td>Right one character</td>
</tr>
<tr>
<td>Ctrl+E</td>
<td>Erase to end-of-line</td>
<td>Up Arrow</td>
<td>Up one line</td>
</tr>
<tr>
<td>Ctrl+F</td>
<td>Search forward</td>
<td>Down Arrow</td>
<td>Down one line</td>
</tr>
<tr>
<td>Ctrl+G</td>
<td>Go to a line number</td>
<td>Home</td>
<td>Beginning of line</td>
</tr>
<tr>
<td>Ctrl+H</td>
<td>Indent the block</td>
<td>End</td>
<td>End of line</td>
</tr>
<tr>
<td>Ctrl+K</td>
<td>Toggle block mode</td>
<td>Page Up</td>
<td>Up one screen</td>
</tr>
<tr>
<td>Ctrl+L</td>
<td>Delete line to the clipboard</td>
<td>Page Down</td>
<td>Down one screen</td>
</tr>
<tr>
<td>Ctrl+M</td>
<td>Set the mark</td>
<td>Center (E)</td>
<td>Center the cursor onscreen</td>
</tr>
<tr>
<td>Ctrl+N</td>
<td>Read a file into a new buffer</td>
<td>Ctrl+Left Arrow</td>
<td>Left one word</td>
</tr>
<tr>
<td>Ctrl+P</td>
<td>Move to the previous position</td>
<td>Ctrl+Right Arrow</td>
<td>Right one word</td>
</tr>
<tr>
<td>Ctrl+Q</td>
<td>Quote the next character</td>
<td>Ctrl+Up Arrow</td>
<td>Up one C function</td>
</tr>
<tr>
<td>Ctrl+R</td>
<td>Replace text</td>
<td>Ctrl+Down Arrow</td>
<td>Down one C function</td>
</tr>
<tr>
<td>Ctrl+S</td>
<td>Switch to the next buffer</td>
<td>Ctrl+Home</td>
<td>Scroll toward beginning of file</td>
</tr>
<tr>
<td>Ctrl+T</td>
<td>Toggle the case of character(s)</td>
<td>Ctrl+End</td>
<td>Scroll toward end of file</td>
</tr>
<tr>
<td>Ctrl+U</td>
<td>Remove indent from the block</td>
<td>Ctrl+Page Up</td>
<td>Beginning of file</td>
</tr>
<tr>
<td>Ctrl+V</td>
<td>Insert the clipboard</td>
<td>Ctrl+Page Down</td>
<td>End of file</td>
</tr>
<tr>
<td>Ctrl+W</td>
<td>Delete word to the clipboard</td>
<td>Ins</td>
<td>Toggle Insert/Overwrite mode</td>
</tr>
<tr>
<td>Ctrl+X</td>
<td>Delete block to the clipboard</td>
<td>Del</td>
<td>Delete character</td>
</tr>
<tr>
<td>Ctrl+Z</td>
<td>Cancel the selected block</td>
<td>Backspace</td>
<td>Delete character backward</td>
</tr>
<tr>
<td>Alt =</td>
<td>Start/end recording macro</td>
<td>Ctrl+Ins</td>
<td>Copy block to clipboard</td>
</tr>
<tr>
<td>Alt -</td>
<td>Playback macro</td>
<td>Ctrl+ Back-Space</td>
<td>Delete word backward</td>
</tr>
<tr>
<td>Alt+F7</td>
<td>Previous error</td>
<td>Shift+Ins</td>
<td>Insert the clipboard</td>
</tr>
<tr>
<td>Alt+F8</td>
<td>Next error</td>
<td>Shift+Del</td>
<td>Delete block to clipboard</td>
</tr>
</tbody>
</table>
POWER

POWER.EXE conserves power on a system that has APM (Advanced Power Management) by shutting down various subsystems (screen, disk drives, etc.) that are not being used. POWER.EXE can be used in an INSTALL command in CONFIG.SYS, in AUTOEXEC.BAT or on the command line.

Syntax

From command line or AUTOEXEC.BAT, enter

```
[path]power[options]
```

From CONFIG.SYS, enter

```
Install=[path]power[options]
```

Remarks

The system BIOS must support the following APM functions for POWER.EXE to work.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>APM Installed</td>
</tr>
<tr>
<td>01</td>
<td>Connect</td>
</tr>
<tr>
<td>04</td>
<td>Disconnect</td>
</tr>
<tr>
<td>05</td>
<td>CPU idle</td>
</tr>
<tr>
<td>06</td>
<td>CPU busy</td>
</tr>
<tr>
<td>07</td>
<td>Set Power State</td>
</tr>
<tr>
<td>08</td>
<td>Get PM Event</td>
</tr>
</tbody>
</table>

For each device, you can specify the length of time that the device must be inactive before it is turned off. If you specify a length of zero for a device, POWER.EXE disables power management for that device.

The (#) argument for each command option defines the number of seconds the device can remain inactive before it is powered down.

Options

The /C# option sets the inactive time for the COM ports. The default value is two seconds.

The /D# option sets the inactive time for the disks. The default value is 30 seconds. Currently, all disk drives are treated as a single device.

The /P# option sets the inactive time for printers. The default time is two seconds.

The /S# option sets the inactive time for the screen. The default time is nine seconds.
The /H option displays a user help screen.

The /K options sets the keyboard time.

The /ADV:MIN argument provides the minimum (most responsive) power reduction.

The /ADV:REG argument provides the standard power reduction.

The /ADV:MAX argument provides the maximum (least responsive) power reduction.

The /STD argument provides the standard power reduction.

The /OFF argument turns power management off.

**Examples**

```
POWER /S20 /C0 /P0
```

Runs POWER, turning off the screen after 20 seconds of inactivity, never turning off the COM and printer ports, and uses the default inactivity period for the disk drives (30 seconds) and keyboard.

---

**PROTO**

PROTO creates function prototypes in C language files.

**Syntax**

```
PROTO filename [filenames]
```

**Remarks**

The start of the function must begin with the first character in a new line. A simple space can cause the function prototype for this function to be overlooked. Which can be useful under certain circumstances. In order to have proto create function prototypes for any assembly files there must be a multi-line comment (see below) with a c-style function description.

**C language file:**

```
unsigned MultiplyTwoBytes( unsigned char ucValue1, unsigned char ucValue2) {
    ... 
}
```

**Assembly Language (ASM) file:**

```
comment ~
unsigned MultiplyTwoBytes( unsigned char ucValue1, unsigned char ucValue2) {} 
~
Public MultiplyTwoBytes
MultiplyTwoBytes proc
    ...
MultiplyTwoBytes endp
```
The prototypes are only displayed to the screen so the user can redirect them where they want.

Proto takes every option on the command line and assumes that it is a filename. Therefore you can specify multiple filenames on the command line. Wildcard characters are allowed in the filenames.

**Note:** At the time this manual was updated, this utility did not contain long filename support.

**Examples**

```
PROTO hello.c hello.asm >hello.h
Create a prototype listing in the output file hello.h from the files hello.c and hello.asm.

PROTO test*.c >test.h
Create a prototype listing in the output file test.h from all of the files matching the name mask of test*.c in the current directory.
```

### RESTORE

The RESTORE command is the complement program to BACKUP. It restores files previously saved with BACKUP to the hard disk.

**Syntax**

```
RESTORE srcdrive [dstpath] [options]
```

**Remarks**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/D</td>
<td>Show directory of files in backup volume(s).</td>
</tr>
<tr>
<td>/P</td>
<td>Prompt only if destination file already exists and is a hidden, system, or read-only file, or is marked as changed. Do not prompt on unchanged existing files.</td>
</tr>
<tr>
<td>/S</td>
<td>Restore all subdirectories.</td>
</tr>
<tr>
<td>/Y</td>
<td>Answer Yes to all prompts.</td>
</tr>
</tbody>
</table>

The srcdrive is the drive to restore from and may include an optional file mask. For example, RESTORE A*:*.C restores only those files ending with .C from the backup set in drive A:.

The dstpath must be a DOS destination path. If no destination path is given, RESTORE assumes the current directory. The destination may contain wildcards (which override wildcards placed in the srcdrive option).
Note: At the time this manual was updated, this utility did not contain long filename support.

Options

The /D option just displays a directory of what files are in the backup set, similar to Dir. This is useful if you are not sure what files or dates/times are in the backup set.

The /S option restores from subdirectories as well.

The /Y option is useful when running RESTORE from a batch file. Any entire set from another hard drive or network drive can be restored without user input.

Examples

RESTORE A: /D
Displays files in the backup.

RESTORE A: D:TEMP\fx*.fh /S
Completes restore, with subdirectories, of all files whose filenames begin with fx and have a .fh file extension.

RESTORE B:*.c /S
Restores only files with a .C file extension to the current directory.

RSZ

RSZ.EXE is a Zmodem file transfer utility used to transfer files over a serial port to another machine running the Zmodem file transfer protocol. RSZ.EXE can be used in place of the COMM program and can be started from within a DOS batch file to send and receive files. In addition, RSZ.EXE does not require that the system have a video display as does the COMM program. RSZ.EXE uses approximately 24KB of RAM.

Syntax

RSZ /Pn [\Inn] /Bn /Hn /Fn [/Q] [/V] /[R][S file1[ file2 ...]]

Some examples of RSZ usage include:

RSZ /R
RSZ /P3 /B115200 /H2 /C /F0 /S a.b lmnop z.*

RSZ Program Options

All command line options must be separated by a space.
### Option Description

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/P</td>
<td>Port number: n = 1, 2, 3, or 4 for COM1 to COM4 (default setting is 1 for COM1).</td>
</tr>
<tr>
<td>/B</td>
<td>Baud rate: n = 50,110,300, 115200. Always N81 when changed. If this option is not specified, the program uses the current port parameters.</td>
</tr>
<tr>
<td>/Inn</td>
<td>IRQ Numbers — valid IRQ selections are 3 – 15. Default value for COM2 and COM4 is IRQ3; the default value for COM1 and COM3 is IRQ4.</td>
</tr>
<tr>
<td>/H</td>
<td>Handshaking options. Both sides must use the same value: 0 = none (default), 1 = software, 2 = hardware.</td>
</tr>
<tr>
<td>/F</td>
<td>File management options (all files are binary): 0 = skip, 1 = resume, 2 = make duplicate, 3 = replace (default)</td>
</tr>
<tr>
<td>/R</td>
<td>Receive files specified by the sender using Zmodem protocol.</td>
</tr>
<tr>
<td>/S</td>
<td>Send the specified files using Zmodem protocol. Wildcards are allowed.</td>
</tr>
<tr>
<td>/V</td>
<td>Verbose switch forces the status display to ON when the console is redirected to a second COM port, and allows the Esc key to break out of a transfer. Esc will not work when RSZ is using the same COM port as the remote console.</td>
</tr>
<tr>
<td>/Q</td>
<td>Do not display/send status information while transferring files. This option is automatically selected when running CTTY.</td>
</tr>
</tbody>
</table>

### File Transfer Recovery

For Zmodem file transfer recovery to work after a failed transfer, you must use RSZ with the /F1 option on the receiver command line. This option selects the resume file management option.

### Time Zones

An additional environment variable for time zones is also supported. The TZ variable allows you to compensate for time zone differences when sending and receiving files and when the time stamp on a received file is critical. This variable only affects transfers done with the Z-modem protocol. X-modem transfers are not affected. By default, if no time zone is set, the Datelights utilities COMM and RSZ assume Greenwich Mean Time (GMT).

Setting the TZ variable is not necessary when using all programs. If you are transferring files between two Datelights utilities, the file time stamp is not affected. Some utilities, such as Windows HyperTerminal, automatically convert the time stamp on the file to GMT and assumes that the receiving program will adjust the time to the correct local time upon receipt.

The syntax for the TZ environment variable is:

```
SET TZ= <abbreviation> +|- value
```

Abbreviation represents any three-letter abbreviation for the chosen time zone. The variable serves as a reminder to the user. For example, if setting the time zone for Pacific Standard Time, the variable could be set as PST and for Eastern Standard Time as EST. The abbreviation is only a placeholder in the syntax for the TZ variable. There are no incorrect abbreviation choices as long as only three letters are used.
Value represents the number of hours this time zone varies from GMT. For example, the west coast of the United States and Canada is –8 hours relative to GMT. This value may have to be adjusted to compensate for daylight savings time. There should be no spaces between the abbreviation, plus or minus sign, and the value. Some examples are:

```
SET TZ=PST –8
SET TZ=CMT –3
SET TZ=GMT+2
```

If an incorrect format for the time zone is entered, the default of GMT is used.

On a desktop PC running Windows programs, such as HyperTerminal, determining the time zone is part of the setup of the operating system. However, if COMM is run from a DOS box, the TZ variable still will need to be set on the target machine.

---

**SHARE**

The SHARE command installs the capabilities for file sharing and file locking on your hard disk.

**Syntax**

```
SHARE [/L:nn][/u]
Or from CONFIG.SYS
INSTALL=[drive:][path]SHARE.EXE [/options]
```

**Remarks**

The SHARE utility is most commonly used in a network or multitasking environment where file sharing is necessary. When SHARE is loaded, DOS utilizes the SHARE utility to validate read and write requests from application programs and users.

**Note**

At the time this manual was updated, this utility did not contain long filename support.

SHARE should not be used under DOS 7.1. With the addition of FAT32 support to DOS 7.1, the internal DOS share tables were overwritten.

**Options**

The /L:nn option specifies the maximum number of files that can be locked at one time. The default number is 20.

The /U option unloads the share utility and frees the memory. SHARE does not unload if other TSRs have been loaded on top of it. The other TSRs must be unloaded first before trying to unload SHARE.
The SMARTDRV command provides disk and diskette cacheing. This is a memory resident program that speeds up your system’s performance by reducing the number of times that an application has to physically access the disk. Datalight’s SMARTDRV is fully compatible with Microsoft’s documented SMARTDRV API (application programming interface).

**Syntax**


**Remarks**

To cache MSCDEX performance, you must load that driver first.

**Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>drive [+</td>
<td>−]</td>
</tr>
<tr>
<td>/X</td>
<td>Disable write-behind caching for all drives.</td>
</tr>
<tr>
<td>/U</td>
<td>Don’t load the CD-ROM caching module.</td>
</tr>
<tr>
<td>/C</td>
<td>Writes all information currently in the write-cache to the disk.</td>
</tr>
<tr>
<td>/R</td>
<td>Clears the cache and restarts the SMARTDRV program.</td>
</tr>
<tr>
<td>/F</td>
<td>Writes the cached data before the command prompt returns (default).</td>
</tr>
<tr>
<td>/N</td>
<td>Doesn’t write the cached data before the command prompt returns.</td>
</tr>
<tr>
<td>/V</td>
<td>Displays the Verbose status messages when loading the cache.</td>
</tr>
<tr>
<td>/Q</td>
<td>Does not display any messages while loading, aka Quiet mode (default).</td>
</tr>
</tbody>
</table>

**Examples**

SHARE

Loads the SHARE program from the command line.

INSTALL=C:\UTILS\SHARE.EXE /I:30

Installs SHARE from the CONFIG.SYS file and changes the maximum number of locked files to 30.

SHARE /U

Unloads SHARE and frees the used memory.
### Option Description

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/S</td>
<td>Displays information about the current status of the cache.</td>
</tr>
<tr>
<td>InitCacheSize</td>
<td>Specifies the amount of XMS memory (KB) for the cache.</td>
</tr>
<tr>
<td>/E: ElementSize</td>
<td>Specifies how many bytes of information to move at one time.</td>
</tr>
<tr>
<td>/B: BufferSize</td>
<td>Specifies the size of the read-ahead buffer.</td>
</tr>
</tbody>
</table>

### Example

The first example enables a write-cached drive C, a read-only cached drives D and Q – the latter being a CD-ROM drive. Note that drive A is also read-cached, even though it was not specified on the command line. The second example shows the results of a status report.

SMARTDRV C+ D Q /V 2048 /E:2048 /B:4096
then
SMARTDRV /S

### SORT

The SORT command sorts a text file and displays the output to the standard device.

**Syntax**

SORT [/R] [/+n] [[drive1]:[path1]filename1] [> [drive2]:[path2]filename2]

or

command | SORT [/R] [/+n] [> [drive2]:[path2]filename2]

**Remarks**

SORT normally starts its comparisons at the first character in a line.

The /+n option is zero justified, the first character in file is at position zero. To sort on the third character in line you would use /+2.

The input to SORT may come from a file, or it may be piped in from another filter or any other DOS command.

**Options**

The /+n option causes SORT to begin its alphabetical sorting starting at the nth position in the string.

The /r option causes SORT to sort in the reverse alphabetical order.
The STACKDEV.SYS command is used to increase the number of stacks available for IRQ handlers and Int13h. The standard STACKS= command for ROM-DOS increases stack space for the DOS stacks only.

Syntax

```
DEVICE=STACKDEV.SYS
```

Remarks

Under various conditions, a stack overflow error occurs while booting ROM-DOS, usually during the CONFIG.SYS or AUTOEXEC.BAT file processing. For example, some other program may service an interrupt and not provide its own stack for that purpose. If, inside its interrupt handler, the program consumes a lot of stack space by allocating many automatic variables, calling other interrupt handlers, or by deeply nesting sub-routines, then the interrupt handler may overflow the stack of the program that it interrupted. This situation usually results in a system crash.

The STACKDEV.SYS device driver handles stack support similar to MS-DOS' internal Stacks support. STACKDEV increases the stack space available for IRQ handlers and Int13h by switching to a local stack each time an interrupt occurs. STACKDEV does not allocate the stacks from a pool, instead, each interrupt on the list is given exactly one stack with a fixed stack size of 128 bytes.

STACKDEV differs from the internal STACKS= support with ROM-DOS in that the STACKS= command is used to allocate stack space for ROM-DOS' own internal stacks and does not add additional stack support for IRQs.

Examples

```
DEVICE = STACKDEV.SYS
```

Loads the STACKDEV.SYS device driver during CONFIG.SYS processing. This driver should be loaded early in the file so that it can handle the IRQ stack needs of other device drivers or programs.
**SUBST**

The SUBST command allows one drive to appear as another drive. This is useful for creating a consistent drive letter when the drive and/or path may change.

**Syntax**

- `SUBST [drive1: [drive2:]:path]`
- `SUBST drive1: /D`

**Options**

- The `/D` switch disables the substituted drive letter (un-installs it).
- The `drive1` argument must be currently unassigned and non-substituted.
- The `drive2` and path arguments must be a valid drive and/or directory.

**Remarks**

SUBST without any options displays the currently substituted drives. The path may be any legal DOS path, including network drives. DOS by default will not reserve room for extra drive letters. The LASTDRIVE command will be required to make available extra drive letters.

**Examples**

- `SUBST e: b:\subdir`
- `SUBSTitute drive E: to use B:\SUBDIR`
- `SUBST`
  - Displays all SUBSTituted drives.
- `SUBST e: /D`
  - Drive E: is no longer SUBSTituted.

---

**SYS**

The SYS command copies the ROM-DOS system files ROM-DOS.SYS and COMMAND.COM from the disk in the default drive to the disk in the specified drive. The file ROM-DOS.SYS is renamed and stored on the disk as files IBMBIO.COM and IBMDOS.COM, which are stored as hidden files.

**Syntax**

- `SYS drive: [/options]`

**Remarks**

- Use the SYS command to transfer the ROM-DOS system files to a floppy disk or hard disk. The disk can be a formatted blank disk or can contain files; it is not necessary for the system files
to be the first files on the disk. The only requirement is that there is enough contiguous free space on the disk for the new system files to be placed. If the disk already contains system files, installing the new system files deletes the existing files.

The command processor, COMMAND.COM, is also transferred to the disk and does not need to be copied into the same contiguous space as the system files.

You can run SYS three different ways. The first is to boot and run your system with ROM-DOS. When you run the SYS command this way, SYS copies the ROM-DOS system files and COMMAND.COM from the root directory of the default/current disk drive.

The second method is to run SYS from the root directory of a disk drive that has been previously prepared with the SYS command, but isn’t booted and running. For example, you can run SYS from a bootable floppy disk to copy the files to the hard disk without actually booting from the floppy disk itself.

The third method uses the file ROM-DOS.SYS, the equivalent of the hidden system files IBMBIO.COM and IBMDOS.COM. ROM-DOS.SYS should be present in the same directory with COMMAND.COM and SYS.COM. These three files can be placed in the root directory or subdirectory on a floppy disk (that need not be booted or bootable), or in a subdirectory on the hard drive. Run the SYS command from the directory where the files reside to transfer the system files to the destination drive.

Options

The /C option prevents confirmation before transferring system files.
The /H option does not hide newly transferred system files.
The /I option prevents display of the sign-on message.

Examples

SYS B:
Copies the ROM-DOS system files to drive B:

TRANSFER

TRANSFER is a file-exchange utility that allows embedded systems to upload and download files over a serial link using the XMODEM protocol. The program running on the host system may be either the COMM terminal program, another of Datalight’s serial communications utilities, or another instance of TRANSFER. Refer to ‘COMM’ for information regarding the COMM program.

In systems where the BIOS supplies a serial port console, TRANSFER uses BIOS Int 10h function 0Eh and Int 16h functions 0 and 1 to transfer files. In systems having a BIOS that does not supply a remote
console, TRANSFER may also be used to transfer files over the same serial link as that used for the CTTY console. A specific serial port (COM1, COM2, and so on) can also be specified for TRANSFER to use for file transfers.

To move a file between systems, run TRANSFER on the target system. Either TRANSFER or COMM may be run on the host PC. If COMM is running on the host PC, press the PgUp key on the PC for COMM to send a file to the target system. COMM prompts for the filename and the protocol for TRANSFER; specify the Xmodem protocol. If you are using TRANSFER on the host PC, select the COM port, the baud rate, and specify either send or receive.

Run TRANSFER as follows:

```
A>TRANSFER [options] filename
```

The `filename` parameter specifies the file to be uploaded or downloaded. A path and drive letter may precede the filename. Wildcards are not allowed in the `filename` parameter.

The `options` for TRANSFER are listed in the following table.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/S</td>
<td>Sends a file.</td>
</tr>
<tr>
<td>/R</td>
<td>Receives a file.</td>
</tr>
<tr>
<td>/B#</td>
<td>Sets the baud rate. The rate number may be 300, 1200, 2400, 4800, 9600, 19200, 38400, 57600, or 115200. The default rate is 9600. Specifying the baud rate causes TRANSFER to use a serial port rather than the default console. The default port is COM1; it can be changed using the /COM option.</td>
</tr>
<tr>
<td>/BC</td>
<td>Uses the BIOS console to transfer files (instead of CTTY).</td>
</tr>
<tr>
<td>/COM#</td>
<td>The /COM# tells TRANSFER not to disable use of the console, and allows the user to select the COM port. The COM number(#) may be either 1, 2, 3 or 4. This option (as well as /B#) causes TRANSFER to use interrupt driven serial I/O. An INS8250/16450/16550 or compatible UART is assumed at standard PC addresses. If the baud rate option is not used along with the /COM option, the default baud rate is 9600.</td>
</tr>
<tr>
<td>/Q</td>
<td>Prevents the display of the output to the screen.</td>
</tr>
<tr>
<td>/IRQ</td>
<td>Allows changing of the COM port IRQ value from the default setting. The default values are IRQ4 for COM1 and COM3; IRQ3 for COM2 and COM4.</td>
</tr>
</tbody>
</table>

When transferring files over the console, TRANSFER uses DOS calls, by default, to allow operation with CTTY. On some smaller systems it may be preferable to use a BIOS console interface to achieve higher throughput. The /BC option tells TRANSFER to use the BIOS’ console interface rather than that of DOS.
TRANSFER Program Examples

The following example receives a file via the console. The data of the file is placed on drive B: in a file named FILE.DAT.

A:>transfer /r B:file.dat

This example sends the file JUNK.ABC over COM4 at 1200 baud, using IRQ 11.

A:>transfer /s /B1200 /COM4 /IRQ11 junk.abc

The following two examples show the use of TRANSFER on both the host PC and the target system. The file ED.EXE is sent from the host PC to the target system. The file received on the target system is named VI.EXE.

Target system command:
A:>TRANSFER /r B:VI.EXE

Host PC system command:
A:>TRANSFER /s C:\BIN\ED.EXE

TREE

The TREE command displays each subdirectory and, optionally, the files within them for a specified drive.

Syntax

TREE [drive:]path [/F] [/A] [/]

Remarks

The TREE command displays the full path of each subdirectory on a specified disk.

drive: specifies the drive that TREE displays the subdirectories from. This argument must be specified. The path argument can refer to any currently existing subdirectory name.

Option

The /F option causes TREE to display the files in each subdirectory.
The /A option will force the use of text characters instead of ASCII graphical characters.
The / option hides the program signon message

Examples

TREE C:
Displays all subdirectories on drive C:

TREE A: /F
Displays all subdirectories on drive A: along with the files within each subdirectory.
UMBLINK builds upper memory blocks (UMBs) which have a distinct MCB chain and may or may not be linked into the DOS MCB chain. UMBLINK can be used on some systems where EMM386 cannot be used.

**Syntax**

```
DEVICE=UMBLINK.EXE [x=mmmm-nnnn]
```

**Remarks**

UMBLINK allows for the conversion of various RAM memory regions into DOS UMBs into which device drivers and TSRs may be loaded. UMBLINK is a simple device driver that, once installed, does nothing but filter the Int21h, Function 5803h calls to link and unlink the UMB arena from the normal DOS MCB chain. UMBLINK must appear before any DEVICEHIGH lines in your CONFIG.SYS file.

Unlike EMM386.EXE, UMBLINK.EXE is not a protected mode control program. It finds certain kinds of memory that already exists in the adapter space and links that memory into the DOS upper memory chain. It then becomes resident and, like Datalight’s EMM386, becomes the UMB LINK handler for ROM-DOS. The memory types that UMBLINK can recognize are S-ICE.EXE UMBs (eliminating the need for Numega’s UMB.SYS), XMS UMBs, and pre-existing RAM that is already either physically or logically addressable within the adapter space at the time the program is run.

**Option**

The x=mmmm-nnnn argument specified a hex segment range of existing RAM to be excluded from the UMB conversion process. Both mmmm and nnnn must be in the range C000:0h to F000:0h.

**Examples**

```
DEVICE=UMBLINK.EXE X=C000 – C800
```

Load UMBLINK.EXE and exclude the range from C000:0h to C800:0h from the UMB creation process.
VDISK

VDISK is a device driver that allows you to use RAM as a disk.

**Syntax**

```
device = VDISK [size [secs[dirs]]] [/E]
```

**Remarks**

VDISK partitions some of your computer’s memory as a disk. This disk is called a RAM disk or virtual disk and is much faster than either a floppy or hard disk. The RAM disk can use either standard DOS program memory or extended memory (above 1MB) for the disk. Any data on the VDISK is lost when the system power is turned off.

The size argument specifies the size of the VDISK in kilobytes. The default is 64KB. The memory selected is allocated from the DOS memory pool, decreasing the amount of memory available for programs unless the extended memory switch is used.

The secs argument specifies the sector size in bytes. The default is 512 bytes per sector. This value must be 128, 256, 512, or 1024. All other values are not valid, and the default of 512 bytes is used.

The dirs argument specifies the number of root directory entries. The default value is 64 directory entries. There may be any number of root directory entries between 2 and 1024. If an odd number is given, it is rounded up to the nearest multiple of 16 to fill the entire sector.

The /E argument causes VDISK to use extended memory (memory above the 1MB boundary) instead of DOS program memory for the disk.

The VDISK driver increases the resident size of DOS.

**Note**: Interrupts are turned off during the transfer of data from extended memory to conventional memory.

**Examples**

```
device = VDISK.SYS
```

Builds a 64KB RAM disk in DOS memory.

```
device = C:\DOS\VDISK.SYS 220 /E
```

Builds a 220KB RAM disk in extended memory. The VDISK device driver is loaded from the C: drive and the \DOS directory. VDISK assumes the default 512 byte sector size and 64 directory entries.

```
device = VDISK.SYS 45 128 18
```

Builds a 45KB RAM disk in DOS memory. There are 128 byte sectors and 18 root directory entries.
XCOPY

The XCOPY command copies multiple files and, optionally, subdirectories from one disk to another.

Syntax

```
```

Remarks

Use the XCOPY command to copy multiple files and subdirectories, if they exist.

The source and destination parameters are complete drive-path and file-specification descriptions. If you do not specify a path, XCOPY assumes the default path. If either filename is not specified, then *.* is assumed.

The ATTRIB command may be used to modify the archive bit for the various XCOPY options that check the archive status of files. Refer to the ATTRIB command on page 83 description for instructions.

Option

The /A option copies only source files that have the archive bit set in them. The archive is not reset.

The /D: mm-dd-yy option copies only those files with a date later than that specified. The /E option creates subdirectories on the target even if they are empty.

The /M option copies only those source files that have the archive bit set. Once the source file is copied, the archive bit is reset.

The /P option prompts before each file is copied. The prompt appears as follows; enter Y to copy the file:

```
C:\COMMAND.COM (Y/N)?
```

The /S option copies files in subdirectories of the source directory.

The /V option verifies each write to the disk.

The /Q option does not display the file names while copying.

The /F option displays full source and destination names while copying.

The /K option copies attributes. Normal xcopy will reset read-only attributes. The /H option copies hidden and system files also.

The /W option waits before starting to copy files and prompts with the following message:

```
Press any key to begin copying file(s)
```
Examples

XCOPY \bin\*.exe a: /a

Copies all files in the BIN subdirectory to the A: drive that have an .EXE extension and that have the archive bit set.

MINI-COMMAND INTERPRETER

Some ROM-DOS applications run on standard PCs and require full command interpreter functionality. Some ROM-DOS applications do not require a command interpreter; for instance, when the application is loaded at boot time and runs until the power is turned off.

The mini-command interpreter (MINICMD.COM) suits applications that require a limited and possibly tailored command interpreter. MINICMD.COM contains only the basic functions of COMMAND.COM and is only 4KB as opposed to the 34KB of COMMAND.COM. MINICMD.COM supports program launching, limited batch file execution, echo control, and environment management. MINICMD.COM can also be used to load applications specified in the AUTOEXEC.BAT file at boot time.

/P (permanent) is the only option to MINICMD.COM. The permanent option causes MINICMD.COM to remain loaded at all times and is used by ROM-DOS when initially loading MINICMD.COM. The permanent option causes MINICMD.COM to ignore the EXIT command. The environment size (/E) option is not implemented. The initial environment size can be specified when building ROM-DOS using the _env_para variable in the SYSGEN.ASM file.

External Commands

MINICMD.COM executes applications and batch files in the same manner as COMMAND.COM. MINICMD.COM searches in the local directory for the program or batch file and then uses the directories specified in the PATH environment variable. The search precedence is .COM file, .EXE file, then .BAT file.

Internal Commands

MINICMD.COM supports several internal commands. Each internal command can be enabled or disabled by means of a #define statement in the file MINICMD.H. A description of each internal command is given in the following table.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>CD</td>
<td>Displays or changes the current working directory.</td>
<td>CD \database\new Changes the current working directory to DATABASE-NEW.</td>
</tr>
<tr>
<td>COPY</td>
<td>Copies one file to another. This is only a single file copy. Wild cards are not supported. You must provide the full path (if not the current directory) and filename for both the source and the destination file.</td>
<td>COPY FILE1.DAT \BIN\BACKUP.DAT Copies one file to the \BIN directory with the new name of BACKUP.DAT.</td>
</tr>
<tr>
<td>DEL</td>
<td>Deletes a single file. Wild cards are not supported.</td>
<td>DEL file1.dat Deletes the file named FILE1.DAT.</td>
</tr>
</tbody>
</table>
### Command Table

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DIR</strong></td>
<td>Displays a list of the files in a specified or the current directory.</td>
<td><code>DIR \ROMDO\DEVSRC</code> Displays the contents of the <code>\DEVSRC</code> directory.</td>
</tr>
<tr>
<td>**ECHO [ ON</td>
<td>OFF</td>
<td>Text ]**</td>
</tr>
<tr>
<td><strong>EXIT</strong></td>
<td>Terminates the mini-command interpreter. If the /P option was specified to mini-command interpreter during spawning, then the EXIT command has no effect.</td>
<td><strong>EXIT</strong></td>
</tr>
<tr>
<td><strong>GOTO</strong></td>
<td>Transfers control to another line in the batch file.</td>
<td><strong>GOTO MESSAGE</strong> Moves the control of execution within the batch file to the line labeled MESSAGE.</td>
</tr>
<tr>
<td><strong>IF</strong></td>
<td>Allows conditional execution of commands.</td>
<td>Refer to the ROM-DOS User’s Guide.</td>
</tr>
<tr>
<td><strong>PAUSE</strong></td>
<td>Causes the batch file processing to stop and wait for the user to press a key.</td>
<td><strong>PAUSE</strong></td>
</tr>
<tr>
<td><strong>REM</strong></td>
<td>Causes the subsequent command-line characters to be ignored; usually used to place comments in a batch file.</td>
<td><strong>REM This is a comment.</strong></td>
</tr>
<tr>
<td><strong>SET [ARG= [Text]]</strong></td>
<td>Sets, displays, or removes environment variables. SET without arguments displays the current environment variable settings.</td>
<td><strong>SET Path=f:\bin</strong> Initializes the PATH environment variable to F:\BIN.</td>
</tr>
<tr>
<td><strong>TYPE</strong></td>
<td>Displays a text file on the console.</td>
<td><strong>TYPE File1.dat</strong> Displays the file <code>FILE1.DAT</code> on the console.</td>
</tr>
<tr>
<td><strong>VER</strong></td>
<td>Displays the version number of MINICMD.COM</td>
<td><strong>VER</strong></td>
</tr>
<tr>
<td><strong>[DRIVE]</strong></td>
<td>Specifies a new current drive with a single letter followed by a colon.</td>
<td><strong>A:</strong> Sets the current drive to the A drive.</td>
</tr>
</tbody>
</table>

### Configuring the Mini-Command Interpreter

MINICMD.COM is configurable and commands can be added to or removed from it. After modifications, MINICMD.COM must be rebuilt using the Borland C compiler and assembler provided in the Datalight SDTK.

The command_table structure contained in CMDMAIN.C lists the internal commands and the associated C-function called when that command is entered on the command line. Removing an entry from the command_table structure makes that command unavailable.
A new internal command can be added by adding the command name, along with the name of the C-function into the command_table structure. Each new C-function is placed into a separate file so it can be added or removed easily.

The C-prototype for functions implementing a mini-COMMAND is:

```c
void do_cmd(char *args);
```

When an internal command is entered, the function in the command_table is called with the text options following the command on the command line. For example:

```
C:\>DIR \bin\abc
```

The function to implement the DIR command is called with the args option equal to `\bin\abc`.

SERIAL FILE SERVER

With these utilities, the client machine can access some or all of the drives of the server machine through a null modem cable and any serial port. Each of the selected drives on the Server will be mapped to available drive letters on the client machine, in a similar fashion to network drives.

The Serial File Server shares its drives and services the request of the client. The client, or Serial File Link, is able to access and use the remote drives. The serial ports on both systems must be connected via a null modem cable. The Serial File Server utilities work across a standard 3-pin serial cable, similar to other Datalight serial I/O utilities (REMDISK/REMSERV, COMM and TRANSFER). The cable does not require the CTS/RTS DTS/DTR pins.

Server Program

The host or server machine runs the program SERSERV.EXE, which can make any or all of its drives available to the client. The syntax of SERSERV.EXE is

```
```

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/R</td>
<td>Tells the Server not to map removable drives.</td>
</tr>
<tr>
<td>/N</td>
<td>Tells the Server not to map network drives.</td>
</tr>
<tr>
<td>/O&lt;list&gt;</td>
<td>The program will only map drive letters from this &lt;list&gt;.</td>
</tr>
<tr>
<td>/D&lt;list&gt;</td>
<td>The program will not map any drive letters from this &lt;list&gt;.</td>
</tr>
<tr>
<td>/C#</td>
<td>Selects the communication port. Available ports are 1 through 4. COM1 is the default port.</td>
</tr>
<tr>
<td>/I#</td>
<td>Set the IRQ for the communications port. Valid settings are 3 – 15. Default is IRQ3 for COM2 and COM4, and IRQ4 for COM1 and COM3.</td>
</tr>
<tr>
<td>/B#</td>
<td>Selects the baud rate for transmission. Available baud rates are 300, 1200, 2400, 4800, 9600, 19200, 38400, 57600, and 115000. The default baud rate is 115000.</td>
</tr>
<tr>
<td>/W#</td>
<td>Sets the time-out in the range of 2 to 60 seconds. The default is 10 seconds.</td>
</tr>
</tbody>
</table>

To display a help screen for SERSERV from the DOS prompt, enter

```
SERSERV /?
```

Example: To connect to a normal server with COM1 at 115200 baud, and map all drives except for logical drives (which are never mapped), enter

```
SERSERV
```

The server program can be terminated at any time by pressing the Q key. SERLINK can then no longer access any of the servers drives until the SERSERV program is run again.
### Option Description

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/U</td>
<td>Unloads SERLINK from memory, thereby disabling the drive letters and freeing the memory occupied by SERLINK.</td>
</tr>
<tr>
<td>/R</td>
<td>Tells the Server not to map removable drives.</td>
</tr>
<tr>
<td>/N</td>
<td>Tells the Server not to map network drives.</td>
</tr>
<tr>
<td>/O&lt;list&gt;</td>
<td>The program will only map drive letters from this &lt;list&gt;.</td>
</tr>
<tr>
<td>/D&lt;list&gt;</td>
<td>The program will not map any drive letters from this &lt;list&gt;.</td>
</tr>
<tr>
<td>/C#</td>
<td>Selects the communication port. Available ports are 1 through 4. COM1 is the default port.</td>
</tr>
<tr>
<td>/I#</td>
<td>Set the IRQ for the communications port. Valid settings are 3 – 15. Default is IRQ3 for COM2 and COM4, and IRQ4 for COM1 and COM3.</td>
</tr>
<tr>
<td>/B#</td>
<td>Selects the baud rate for transmission. Available baud rates are 300, 1200, 2400, 4800, 9600, 19200, 38400, 57600, and 115000. The default baud rate is 115000.</td>
</tr>
<tr>
<td>/W#</td>
<td>Sets the time-out in the range of 2 to 60 seconds. The default is 10 seconds.</td>
</tr>
</tbody>
</table>

Note: This option should not be used when the client has files open, since they will be closed when quitting.

### Client Program

The program SERLINK runs on the client system and creates new drive letters for the client. SERLINK uses the next available system drive letters, in order. For example, if the last assigned drive was D:, the first drive SERLINK creates is drive E:. This drive acts like any other network drive, except that it requires the serial port. The syntax for loading SERLINK is:

```
```

The syntax for unloading SERLINK is:

```
SERLINK.EXE /U
```

Example: To connect to a normal server with COM1 at 115200 baud, and map all drives except for logical drives (which are never mapped), enter

```
SERLINK
```

Example: To install SERLINK from the DOS prompt or from a batch file (such as AUTOEXEC.BAT) at 9600 baud, on COM2, and not map any removable drives, enter

```
SERLINK /B9600 /C2 /R
```
REMOTE DISK PROGRAM

The remote disk program allows you to access a disk drive on a remote system via a serial cable and standard PC-style (8250 UART) serial port. Remote disk allows added flexibility for diskless systems and systems tight on available space.

In a remote disk setup one system, called the server, shares its drives. The other system, called the client, accesses and uses the remote drives. The serial ports on both systems must be connected via a null modem cable. Remdisk/Remserv works across a standard 3-pin serial cable, similar to other Datalight serial I/O utilities (COMM and TRANSFER). The cable does not require the CTS/RTS DTS/DTR pins.

Server Program

The server system runs the program REMSERV.EXE that can make a single drive on the server system available to the client. The available drive can be changed at any time by quitting the REMSERV program and then running the program again with a new drive letter. The syntax of REMSERV.EXE is

```
REMSERV.EXE d: [/Bnnn] [+] [-] [/COMn] [/Tnnn] [/S] [/H]
```

where d: represents the letter of the drive the server makes available to the client.

### Option Description

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/Bnnn</td>
<td>Selects the baud rate for transmission. Available baud rates are 300, 1200, 2400, 4800, 9600, 19200, 38400, 57600, and 115000. The default baud rate is 115000.</td>
</tr>
<tr>
<td>+/-</td>
<td>The plus sign (+) specifies packet-style transmission and is recommended for any baud rate over 19200. The default setting is to include + for packet transmission. Use the minus sign (-) to specify polling operation under Windows95.</td>
</tr>
<tr>
<td>COMn</td>
<td>Selects the communication port. Available ports are 1 and 2. COM1 is the default port.</td>
</tr>
<tr>
<td>/IRQn</td>
<td>Set the IRQ for the communications port. Valid settings are 3 – 15. Default is IRQ3 for COM 2 and COM4, and IRQ4 for COM1 and COM3.</td>
</tr>
<tr>
<td>/Tnnn</td>
<td>Sets the time-out in the range of 2 to 3,640 seconds.</td>
</tr>
<tr>
<td>/S</td>
<td>Instructs REMSERV to run without any display output.</td>
</tr>
<tr>
<td>/H</td>
<td>Selects hardware handshaking for flow control. To select drive B: as the available server drive at 115000 baud, packet transmission, using COM1, enter <code>REMSERV B:</code>. To set drive C: as the server disk at 9600 baud, without packet-style transmission, on COM2, enter: <code>REMSERV C: /B9600 /COM2</code></td>
</tr>
</tbody>
</table>

The server program can be terminated at any time by pressing the Esc key. The client can then no longer access the server’s drive until the REMSERV program is run again.

Client Program

The program REMDISK runs on the client system and creates a new drive letter for the client. REMDISK uses the next available system drive letter. For example, if the last assigned drive was D;
REMDISK creates drive E:. This drive acts like any other drive, except that it requires the serial port. REMDISK.EXE can be loaded by a DEVICE= command in the CONFIG.SYS file can be entered at the DOS prompt. The syntax for REMDISK is:

```
REMDISK [/U] [/H] [/Bnnnn] [+|-] [/Tnnn] [/COMn]
```

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/U</td>
<td>Unloads REMDISK from memory, thereby disabling the drive letter and freeing the memory occupied by REMDISK. This option can only be used when REMDISK is installed from the DOS command line. A remote disk installed via CONFIG.SYS cannot be unloaded.</td>
</tr>
<tr>
<td>/H</td>
<td>Selects hardware handshaking for flow control.</td>
</tr>
<tr>
<td>/Bnnnn</td>
<td>Selects the baud rate for transmission. Available baud rates are 300, 1200, 2400, 4800, 9600, 19200, 38400, 57600, and 115000. The default baud rate is 115000.</td>
</tr>
<tr>
<td>+/-</td>
<td>The plus sign (+) specifies packet-style transmission and is recommended for any baud rate over 19200. The default setting is to include + for packet transmission. Use the minus sign (-) to specify polling operation under Windows95.</td>
</tr>
<tr>
<td>/Tnnn</td>
<td>Sets the time-out in the range of 2 to 3,640 seconds.</td>
</tr>
<tr>
<td>/IRQn</td>
<td>Set the IRQ for the communications port. Valid settings are 3 – 15. Default is IRQ3 for COM 2 and COM4, and IRQ4 for COM1 and COM3.</td>
</tr>
<tr>
<td>COMn</td>
<td>Selects the communication port. Available ports are 1 and 2. COM1 is the default port.</td>
</tr>
</tbody>
</table>

To install the REMDISK program from CONFIG.SYS at 19200 baud, on COM1, using packet-style transmission, insert the following line in CONFIG.SYS and then reboot the system (remember to include the full path to find REMDISK.EXE if not located in the root directory).

```
DEVICE=REMDISK.EXE /B19200 +
```

To display a help screen for REMDISK from the DOS prompt, enter

```
REMDISK /?
```

To install REMDISK from the DOS prompt or from a batch file (such as AUTOEXEC.BAT) at 9600 baud, without packet-style transmission, on COM2, enter

```
REMDISK /B9600 /COM2
```

To unload the REMDISK installed from the batch file or the DOS prompt, enter

```
REMDISK /U
```

**Using the Remote Disk**

To use the remote disk, both REMDISK and REMSERV must be running on their respective systems and must use the same baud rate and packet or nonpacket style transmission. After starting both programs, you can access the new drive on the client system. You can change the default directory to this new drive, copy files to and from the remote drive, and also run utilities such as CHKDSK on
the drive. The remote drive on the server system can be used as any other drive on the client system.

**Unloading the Server Remotely**
To unload the remote server, REMSERV, use the utility REMQUIT on the client, REMDISK. REMQUIT takes no command line parameters.
KEYBOARD LAYOUTS
The following keyboard charts represent several countries supported by ROM-DOS. If you need a keyboard layout not displayed in this Appendix, please contact Tuxera.

**Canada**

![Canada keyboard layout](image)

**Denmark**

![Denmark keyboard layout](image)
CHAPTER 8 | ROM-DOS KEYBOARD LAYOUTS

**Sweden**

**United Kingdom**

**United States**
ROM-DOS GLOSSARY

ANSI (American National Standards Institute)
A group that defines U.S. standards for the information processing industry. ANSI participates in defining network protocol standards.

API (Application Program Interface)
An API is a specification of the methods an application programmer can use to access services provided by a software module. In the case of a network, the API specifies the interface to the network software. A common way of interfacing a terminal emulator to networking software in a PC is to use Interrupt 14h. This is the PC BIOS entry point for serial port support, but when used for networking purposes, the original entry point is reused to provide a similar, but much expanded function. In addition to the native character at a time transfer, block transfers are also offered to increase throughput.

Baud
Literally, the number of times per second the signal can change on a transmission line. Commonly, the transmission line uses only two signal states making the baud rate equal to the number of bits per second that can be transferred. The underlying transmission technique may use some of the bandwidth, so it may not be the case that users experience data transfers at the line’s specified bit rate.

BIOS
Basic Input Output System – software that interfaces directly with the hardware.

BIOS extension
A short program that the BIOS recognizes and executes as the BIOS initializes the system.

Boot
Booting is restarting and reloading DOS. A PC can be booted by turning it off and then turning it on or by pressing the Ctrl, Alt, and Del keys simultaneously.

Bootable disk
A system disk that contains the files necessary to start and run the computer.

BOOTP (Bootstrap Protocol)
A protocol a host uses to obtain startup information, including its IP address, from a server.

Built-in device
Built-in device is an input/output device which is part of the DOS kernel.

DOS
Disk Operating System – an operating system that relies on disks for file storage.
**DOS kernel**
The DOS kernel is the part of DOS that handles a standard DOS call (Int 21h). It handles opening, reading/writing of files, loads programs, and manages memory.

**FAT**
File Allocation Table – a data table which allows DOS to keep track of file location on the disk so that they can be accessed by programs running on DOS.

**Flow control**
Control of the rate at which hosts or routers inject packets into a network or Internet, usually to avoid congestion.

**LAN (Local Area Network)**
Any physical network technology designed to span short distances (up to a few thousand meters). Usually, LANs operate at tens of megabits per second through several gigabits per second.

**Memory disk**
A disk that uses either ROM or RAM for the disk media. The memory disk has a FAT, directories, and file data.

**MTU (maximum transmission unit)**
The largest amount of data that can be transferred across a given physical network.

**PCMCIA**
Personal Computer Memory Card Interface Association. PCMCIA is a group that defined the standard for a credit-card size card that may act as a memory RAMDISK, ROMDISK, or FLASHDISK. These cards are commonly referred to as PC cards.

**POST**
Power On Self Test – a test performed by the BIOS that checks the computer hardware for problems before fully initializing the computer.

**RAM disk**
A disk drive that uses RAM for the media in place of the usual rotating disk drive.

**ROM-DOS**
Datalight operating system that can be placed in and execute from within a ROM.

**ROM disk**
A disk drive that uses ROM for the media in place of the usual rotating disk drive.

**ROM scan**
The scanning of the ROM area for BIOS extensions performed by the BIOS at initialization time.
ROM
Read Only Memory. This is memory that is not changeable once placed in a computer.

Shell
The Shell is the command interpreter, usually COMMAND.COM. The shell takes text commands and calls the DOS kernel to implement them.

System disk
See Bootable disk.
CHAPTER 9 | SOCKETS INTRODUCTION

ABOUT SOCKETS

System Requirements
SOCKETS requires an IBM compatible 186 or higher system with a minimum:

- 512KB of RAM
- ROM-DOS 6.22 or compatible

SOCKETS network operation requires one or more of the following, depending on the network configuration:

- Any network interface supporting the packet driver specification class 1, 3, 6 or 11.
- PC compatible asynchronous serial ports (COM ports).
- Any network interface supporting the ODI specification

Datalight SOCKETS is an Internet protocol software extension to ROM-DOS that provides a powerful data communication facility whereby embedded systems and users of embedded systems can communicate with other computers (including PCs and mainframes) and their printers. Datalight SOCKETS also provides the facilities to run custom-written applications which allows you to:

- Run applications on a TCP/IP host system from a remote embedded system.
- Transfer data between an embedded system and TCP/IP hosts.
- Run network aware applications on an embedded system.
- Print to an embedded system from TCP/IP hosts and vice versa.

Datalight SOCKETS consists of:

- A TSR kernel:

Connecting to a physical Ethernet or Token Ring network using a network interface with associated Packet Driver and/or to a point-to-point serial network using standard serial communication ports with or without modem dial in/out.

Implementing standard Internet protocols ARP, PPP, LCP, IPCP, IPv6CP, PAP, CHAP MD5, IP, IPv6, ICMP, ICMPv6, IGMP, RIP, UDP, TCP, BOOTP, DHCP and DNS.

Providing IP routing support for IPv4.

Providing two Application Programming Interfaces (APIs) Providing a Socket Print client

Providing a Socket Print Server and LPD Server
Optionally keeping MIB II status and statistical information.

- C libraries and source code to access the APIs including a TCP/IP Sockets library implementing the BSD Sockets abstraction. The libraries also support 32 bit applications using a DOS extender.
- A Sockets kernel build program.
- A SOCKETS configuration program.
- Utility programs to test the network and display the status of the kernel.
- Mail programs in source and binary format.
- Resident servers for FTP, HTTP and Remote Console including a CGI API for serving dynamic web-pages and a Remote Console Java applet to emulate a DOS console of the embedded system on a Java capable browser. Remote Console clients for both DOS and Windows. WebDos, a methodology to enable browser based access to an embedded system, including WebForms for easy browser accessed application development and WebDos Commander for managing the embedded file system.
- An FTP client and DOS and Windows versions of HTTP file GET and PUT utilities.
- Print clients for SOCKETS printing and LPD printing (LPR).
- A resident FTP API to implement FTP client/server functionality in user written programs.
- A resident RFC compliant NETBIOS API allowing file sharing using third party and freeware redirectors and file services.
CHAPTER 10 | SOCKETS INSTALLATION

INSTALLING AND RUNNING SOCKETS

Upon receiving SOCKETS, electronic or CD, SDK or demo, the first step involved is installing to a Windows 9X or WIN NT based development system. The install process will create, by default, a directory of C:\DL\SOCKETS.

```
<root>
    + --- <DL>
        + --- SOCKETS
            +--- BSDSOCK
            +--- BUILDLIB
            +--- CAPI
            +--- CLIENTS
            +--- CONFIGS
            +--- EXAMPLES
            +--- INCLUDE
            +--- LIB
            +--- SERVER
            +--- UTILS

Default Directory Structure
```

The files installed cover:

- Kernel applications (the core of TCP/IP communication)
- Configuration examples
- Utilities
- Optional applications
- API with programming examples

The SDK does not include drivers for Network Interface controllers; hereafter referred to as a NIC. Please refer to Chapter 3 for information on obtaining and configuring network drivers as well as serial configuration details.

The remainder of this chapter describes how to proceed after SOCKETS has been installed to the development system.

Development System Procedure

Insert the ROM-DOS / SOCKETS SDK into your CD-ROM and run “D:\SETUP” where “d” is the appropriate drive letter for your CD. The install process will create a DL\SOCKETS directory. Within the directory DL\SOCKETS will exist the pre-built kernel applications SOCKETM.EXE and SOCKETP.EXE and the Sockets custom kernel build utility, SBUILD. Beyond that, there are subdirectories for:

- UTILS - contains tools for troubleshooting and configuring SOCKETS.
- CONFIGS - contains example configuration files for SOCKETS connections.
CHAPTER 10 | SOCKETS INSTALLATION

- EXAMPLES - contains programming examples.
- CAPI - contains the source code of the Basic API, CAPI.
- BSDSOCK - contains the source code of the Advanced Sockets API.
- CLIENTS - contains the client applications.
- SERVER - contains the SOCKETS server providing HTTP and FTP services.
- INCLUDE - contains the header files to compile user programs and custom kernels.
- LIB - contains the libraries of the APIs to link user programs.
- BUILDLIB - contains the kernel libraries to build custom kernels.

Environment Variables
SOCKETS uses environment variables to determine the location of necessary configuration files. They can be set, using the DOS SET command, within the autoexec.bat file at startup or within a batch file prior to SOCKETS being loaded. They are:

- SOCKETS
- HTTP_DIR
- HOSTNAME
- FTPDIR

Examples

```
Set SOCKETS=C:\NETWORK
Set HTTP_DIR=C:\HTML
Set HOSTNAME=FTPDEMO
Set FTPDIR=C:\FTP
```

Remarks
The environment variable SOCKETS is used to indicate to the SOCKETS kernel where configuration files, license file for the demo version, and access rights files are located. For example “Set SOCKETS=C:\NETWORK” would cause SOCKETS to look for configuration files within the directory of C:\NETWORK.

The environment variable HTTP_DIR is used by the SOCKETS server and indicates the location of html files if the files are stored in a separate directory from the server executable. To continue with the previous example if the server were in the directory of C:\NETWORK and the html files were in C:\HTML then the environment variable would read “SET HTTP_DIR=C:\HTML”.

The environment variable HOSTNAME is used by SOCKETS to indicate the banner name displayed during an FTP session.

The environment variable FTPDIR is used by SOCKETS to indicate the location of the temporary file created by the SOCKETS server during an FTP session. Please refer to Chapter 7 for more details.
File Selection
What you wish to do with the SOCKETS TCP stack will dictate which files are to be transferred to the target hardware. At a minimum your target hardware will require the TCP stack executable:

- SOCKETM.EXE for serial / ppp connection
- SOCKETP.EXE for an Ethernet connection
- SOCKET.CFG for both serial and ethernet use
- MODEM.MCF for a serial / ppp connection

If you wish to use the Datalight Web Server with SOCKETS you will need the HTTPD.EXE, HTTPFTPD.EXE or SUPERD.EXE file located within DL\SOCKETS\SERVER. You will also need an index.htm file that contains the home page of the web server. One has been provided within the same directory.

Full explanations of the files are available in "Chapter 7, ROM-DOS Server" on page 124.

Building a custom SOCKETS kernel
Building a kernel containing only the features required by the application is performed using the SBUILD utility. This process is described in detail in the Tutorials chapter of the Developer’s Guide.

Configuration
The configuration of SOCKETS is determined by what you would like to do with the TCP/IP stack. Some examples are provided within “Chapter 13, SOCKETS Configuration Examples” of the types of configurations easily accomplished with SOCKETS. For quick testing of SOCKETS please reference the sample configuration files supplied within the \CONFIGS directory. Their explanations are found in “Chapter 13, SOCKETS Configuration Examples”.

To make your own configuration, run the supplied file sconfig.exe to configure SOCKETS for your target hardware. SCONFIG.EXE is a text-based tool that will prompt for various settings necessary to customize SOCKETS. At completion sconfig.exe creates the file SOCKET.CFG that must accompany SOCKETS to the target hardware. Running SCONFIG may take place on either your target hardware or your development system. If you wish to configure the stack after transferring files to your target hardware, please include the file SCONFIG.EXE in your list of files to be transferred to your target hardware.

Transfer
Transfer the selected files onto the target system into a \DL\SOCKETS\ directory. As Datalight ROM-DOS has various methods of serial port transfers available, please refer to the ROM-DOS manual for specific details.

Testing
If you are using an Ethernet TCP stack you must first load a packet driver specific to your NIC. The packet driver provides a software interrupt to allow communication between the NIC and SOCKETS (default is 0x60). If you do not have a packet driver or are unsure of its use please contact Datalight Technical Support. Although Datalight does not manufacture packet drivers our support team is knowledgeable about packet driver implementation.
Type SOCKETM.EXE or SOCKETP.EXE to launch SOCKETS. The various options for launching
SOCKETS are:

/n=normal_sockets
/i=capi_interrupt
/d=dos_compatible_sockets
/m=memory_size
/p=print_delay
/s=stack_size
/v=interrupt_vector
/q
/0
/u

Once SOCKETS has been loaded correctly, the most basic test to ensure that everything is working
correctly is to “PING” a known server or gateway that the SOCKETS machine is connected to. If the
ping is returned everything is working, if the ping fails please refer to the troubleshooting section of
the SOCKETS manual.

Custom Applications
The key to designing an application to work with SOCKETS is the TCP/IP Basic and Advanced
APIs. These APIs provide the interface between your application and SOCKETS. The TCP/IP Basic
and Advanced APIs must be linked into your application at compile time. At the time of this manual
both APIs have been designed to work with the Borland C, Microsoft C 1.52 and GNU-C, DJGPP
compilers. The latter compiles 32 bit code for DOS and can be used with a DPMI DOS extender.
Other ports of the TCP/IP Basic and Advanced APIs will be posted upon completion. The SOCKETS
manual details each of the functions within the APIs and can be used as a reference if you need to
make changes in order to compile either API into your application. Please contact your sales rep or
technical support for an updated compiler compatibility list.
CHAPTER 11 | SOCKETS CONFIGURATION

Configuring SOCKETS consists of setting up environment variables, loading required drivers, creating configuration files and running the appropriate SOCKETS kernel with the correct command line parameters.

The SOCKETS environment variables are discussed in Chapter 10, SOCKETS Installation.

PACKET DRIVER
When SOCKETS is required to interface to a Local Area Network or any special network, one or more Packet Drivers must first be loaded. A Packet Driver presents a standard interface to underlying network hardware and is normally supplied by the vendor of the NIC used to connect to the physical network.

The following packet-driver classes are supported:

- Class 1 DIX Ethernet
- Class 3 Token Ring
- Class 6 SLIP
- Class 11 IEEE Ethernet

Most Ethernet packet drivers support both classes 1 and 11. As the default, Class 1 should normally be used.

Each Packet Driver requires a software interrupt vector through which it is accessed. For a single Packet Driver, Interrupt 0x60 or 0x69 is normally used. Avoid using Interrupt 0x61 and 0x7f as those are the defaults used by the SOCKETS APIs. Interupt 0x62 is the default used by the SOCKETS FTP API and 0x63 that of the SOCKETS HTTP Extension CGI API.

For specific information on loading packet drivers, refer to the documentation of the specific driver you are using.

Example:

```
rtspkt 0x60
```

When only one Packet Driver is used, the SOCKETS kernel will search for the interrupt vector when the interface command in the configuration file specifies that a Packet Driver must be used. The interrupt can also be explicitly specified and must be so specified when more than one Packet Driver are used.

SERIAL OPERATION

Hardware considerations
For asynchronous serial operation SOCKETS supports the standard PC COM ports without an external
driver. Up to six COM ports with or without interrupt sharing may be used. SOCKETS checks for and uses FIFO buffered 16550 UARTs for faster throughput. It is strongly recommended that buffered UARTs be used for high-speed applications.

The interface command in the configuration file specifies the I/O address, hardware interrupt and speed to be used and parameter commands can be used to specify character data format and flow control. Flow control may be performed by means of modem control signals (hardware flow control) or XON/XOFF control characters (software flow control). Software flow control is only possible when PPP is used. It will not function for SLIP or CSLIP.

**PPP Functionality**

PPP can be set up for “client” and/or “server” operation. “Client” operation normally refers to a dial-out operation and “server” to a dial-in operation.

For “server” operation, log-in is controlled by a list of Username/Password pairs, each coupled to a remote IP address, which is assigned to the peer if requested to do so during the IPCP negotiation. As an option, a Username/Password pair can be flagged to provide Callback Verification (CBV). In this case, the call is terminated as soon as the PPP negotiations have been successfully completed. Then a reverse call is made by the server; using the number in the modem command file if a modem is involved. During the first subsequent set of PPP negotiations, the CBV flag is ignored to prevent another callback. When the PPP session terminates, the CBV flag is again enabled.

Another PPP feature is the ability to specify that a PPP session should start immediately when SOCKETM is loaded, or to delay that until traffic is generated. It is also possible to specify two sets of PPP parameters per interface. One set of parameters is for outgoing connections; the other set is for incoming connections.

**MODEM OPERATION**

When using serial operations, modem dial-in and dial-out, may optionally be used. Simple modem scripting commands contained in modem configuration files may be used to establish modem connections. Commands in the modem file can specify that a dial-up connection will always be dialed whenever DCD is not detected, or specify that dial on demand is disabled. For robust operation, it is recommended that the DCD modem signal be supported by the hardware and that the modem must drop a connection when the DTR modem signal is lowered. In order to work properly, the modem must support a command set like the AT command set. If the hardware does not comply to this requirement, a modem may still be used. In this case the interface will be specified as not having a modem and a user program must perform the modem connect/disconnect functions. An API function, IfaceIOCTL(), is provided to make this possible. IfaceIOCTL() allows a user program to:

- Initiate a modem connection (dial)
- Disconnect a modem connection
- Enable or disable dial on demand
- Enable or disable a serial port to allow a user program to access it without conflict
- Read modem and connection status.
A sample program, IOCTL.EXE, shows the use of this function and is provided in source and binary format in the SOCKETS Software Development Kit.

SOCKETS CONFIGURATION FILES: SOCKET.CFG, HOSTS
SOCKETS uses two files in the \DL\SOCKETS directory (default), or any other directory specified by the SOCKETS environment variable. These files are SOCKET.CFG, the default start-up file, and HOSTS, the host names file. If not found, SOCKETS uses the default SOCKET.CFG in the \DL\SOCKETS directory. For a single interface LAN configuration a configuration file need not exist. If not found, SOCKETS will default to an interface called if0 with DHCP support for IPv4 and address autoconfiguration for IPv6.

SOCKET.CFG is a text file containing configuration commands. Empty lines and lines starting with # are ignored. Commands are used to specify protocol parameters like the IP address of the stack, interface parameters like Packet Driver or Asynchronous Serial lines, routes and various other parameters. Here is a simple example:

```
ip address demo
Set the IP address of this host to 192.6.1.1. interface pdr if0 dix 1500 5
Use Packet Driver, naming the interface ‘if0’, MTU=1500, Receive buffers = 5
route add default if0 router
Route all traffic to unknown destinations via ‘if0’ using ‘router’ as a gateway
tcp mss 1460
TCP Maximum Segment Size = 1460. tcp window 2920
TCP Maximum window = 2920. start prntserv
Start printer server on PRN using default port of 10.
```

HOSTS is an optional file containing mappings of IP addresses in dotted decimal notation to names.

Sample HOSTS file:

```
192.6.1.1 demo
192.6.1.2 router
192.6.1.3 server
```

CONFIGURING FOR IPV6 OPERATION
Since IPv6 is autoconfiguring, no new configuration commands are used. In an IPv6 only kernel the ‘ip address host’ command is ignored except that host is stored as the local host name which only has local significance. A link-local address is automatically configured using the MAC address as a basis. Other addresses are auto-configured from the MAC address and prefixes advertised by routers.

The ‘route’ command is not implemented for an IPv6 only kernel. Default router lists are configured from Route Advertisements sent by routers.
SCONFIG is a SOCKETS configuration utility. It supports only the barest configurations: Ethernet over a packet driver and PPP over a serial modem. Just answer the multiple choice questions, and then fill in a few data fields and SCONFIG writes simple SOCKET.CFG and MODEM.MCF files for you.

**SOCKET.CFG SAMPLES**

The following configuration file contains the minimum possible commands for a valid configuration file: just one. This is to specify that the interface should use a Packet Driver, the interrupt vector which must be searched for. It should use DIX encapsulation, have an MTU of 1500 and have a maximum of 5 receive buffers. Since no IP address is specified, BOOTP will be used and the required operating parameters will be retrieved from a BOOTP server, which must be available on the network. For an IPv6 only kernel, this single line file is also sufficient.

**SOCKET.CFG:**

```
interface pdr if0 dix 1500 5
```

The following is a more typical example specifying a static IP address, a Packet Driver interface, a default route, the TCP MSS and WINDOW, and the SOCKET PRINT server to be started.

**SOCKET.CFG:**

```
# Sample configuration file
ip address 192.6.1.1
interface pdr if0 dix 1500 5
route add default if0 192.6.1.2
tcp mss 1460
tcp window 2920
start prntserv
```

The next example is a configuration file for running PPP over a direct serial link (no modem). It uses COM1 at the default I/O address of 0x3f8 using interrupt level 4 at 9600 baud with no flow control. The local IP address in this case will be obtained from the PPP peer. No compression of PPP headers and no TCP/IP header compression will be negotiated. No authentication will be done.

**SOCKET.CFG:**

```
if ace asy p0 ppp 1500 30 0x3f8 4 9600
route add default p0
par p0 ipcp local address 0.0.0.0
par p0 lcp start
par p0 ipcp open
```

The following example is a more typical configuration file for running PPP over a direct serial link with a modem. It uses COM1 at the default I/O address of 0x3f8 using interrupt level 4 at 28800 baud with Xon/Xoff flow control. The local IP address in this case will be obtained from the PPP peer. Compression of PPP headers and TCP/IP header compression will be negotiated. PAP authentication
will be negotiated and “PppUser” as a username and “PppPassword” as a password will be used for authentication. Modem commands will be retrieved from the MODEM.MCF file.

SOCKET.CFG:

```plaintext
iface asy p0 ppp 1500 30 0x3f8 4 28800 modem.mcf
param p0 288000 x x
route add default p0
par p0 lcp local accm 0x0a0000
par p0 lcp local acfc on
par p0 lcp local pfc on
par p0 lcp local magic on
par p0 pap user PppUser PppPassword
par p0 ipcp local compress tcp 16 1
par p0 ipcp local address 0.0.0.0
par p0 ipcp start
par p0 ipcp open
```

**SOCKETP, SOCKETM AND SOCKETS OVERVIEW**

Two pre-built versions of the SOCKETS kernel are available: SOCKETM.EXE provides support for serial connections as well as Packet Driver connections. SOCKETP.EXE only provides support for Packet Drivers and has a smaller memory footprint than SOCKETM.EXE. A Sockets kernel may also be custom-built using the SBUILD utility. By default this kernel is named SOCKETS. SOCKETS, SOCKETM or SOCKETP runs as a Terminate and Stay Resident (TSR) program.

Running SOCKETS without any parameters will be sufficient in simple configurations and will use the default %SOCKETS%\SOCKET.CFG and %SOCKETS%\HOSTS configuration files. Parameters are used to specify another configuration file and to tailor the facilities like memory and stack usage, number of simultaneous connections and the API interrupt vectors to use. The /U parameter may be used to unload an already resident SOCKETS kernel.

**PING**

The PING.EXE program gives a quick method to test your SOCKETS installation. The source code is also supplied as an example.

**Syntax**

PING ip_address

Where the ip_address may be a numeric or symbolic address. Refer to the Sockets Utilities description for advanced PING options that can be specified.
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARPSTAT</td>
<td>Displays information about the current ARP table and allows entries to be removed.</td>
<td>Page 200</td>
</tr>
<tr>
<td>CAPIDiAG</td>
<td>A diagnostic program for the SOCKETS TCP/IP Basic API, the stack and the network.</td>
<td>Page 200</td>
</tr>
<tr>
<td>DHCPSTAT</td>
<td>Displays status about the DHCP settings and allows a DHCP lease to be renewed.</td>
<td>Page 201</td>
</tr>
<tr>
<td>FTP</td>
<td>File Transfer Protocol (FTP) client application.</td>
<td>Page 202</td>
</tr>
<tr>
<td>FTPD</td>
<td>A server providing FTP services.</td>
<td>Page 202</td>
</tr>
<tr>
<td>FTPAPI</td>
<td>A TSR that provides FTP services through an application programming interface.</td>
<td>Page 205</td>
</tr>
<tr>
<td>GETMAIL</td>
<td>A POP3 mail client.</td>
<td>Page 205</td>
</tr>
<tr>
<td>HTTPD</td>
<td>A server providing HTTP services.</td>
<td>Page 245</td>
</tr>
<tr>
<td>HTTPFTPD</td>
<td>A server providing both HTTP and FTP services.</td>
<td>Page 255</td>
</tr>
<tr>
<td>HTTPGET</td>
<td>An application that can retrieve files from an HTTP server.</td>
<td>Page 206</td>
</tr>
<tr>
<td>HTTPPUT</td>
<td>An application that can put files to an HTTP server.</td>
<td>Page 207</td>
</tr>
<tr>
<td>IFSTAT</td>
<td>Displays information about configured SOCKETS interfaces.</td>
<td>Page 208</td>
</tr>
<tr>
<td>IOCTL</td>
<td>A diagnostic utility for testing the SOCKETS TCP/IP Basic API IOCTL functions.</td>
<td>Page 209</td>
</tr>
<tr>
<td>IOCTLH</td>
<td>A diagnostic utility for testing the SOCKETS TCP/IP Basic API IOCTL Hotswap functions for wireless or removable Ethernet cards.</td>
<td>Page 210</td>
</tr>
<tr>
<td>IPSTAT</td>
<td>Displays information about SOCKETS memory usage and IP interface.</td>
<td>Page 210</td>
</tr>
<tr>
<td>LPR</td>
<td>A printer client for UNIX style print servers.</td>
<td>Page 211</td>
</tr>
<tr>
<td>MAKEMAIL</td>
<td>Prepares text and data for sending through SENDMAIL.</td>
<td>Page 212</td>
</tr>
<tr>
<td>MEMSTAT</td>
<td>A utility that reports memory usage statistics.</td>
<td>Page 214</td>
</tr>
<tr>
<td>NETBIOS</td>
<td>A TSR that provides a NETBIOS compatible network interface.</td>
<td>Page 214</td>
</tr>
<tr>
<td>PDTEST</td>
<td>A diagnostic program that tests loaded packet drivers.</td>
<td>Page 216</td>
</tr>
<tr>
<td>PING</td>
<td>A utility to send ICMP echo requests and report on replies.</td>
<td>Page 217</td>
</tr>
<tr>
<td>RC.JAR</td>
<td>A Java remote console applet for connecting to hosts running HTTPD.</td>
<td>Page 218</td>
</tr>
<tr>
<td>RC_DK.JAR</td>
<td>A version of RC.JAR with Danish keyboard support.</td>
<td>Page 218</td>
</tr>
<tr>
<td>RCCLI</td>
<td>A DOS remote console client for connecting to hosts running HTTPD.</td>
<td>Page 218</td>
</tr>
<tr>
<td>ROUTE</td>
<td>A utility to display and manage routes.</td>
<td>Page 219</td>
</tr>
<tr>
<td>SCONFIG</td>
<td>A SOCKETS configuration utility.</td>
<td>Page 158</td>
</tr>
<tr>
<td>SENDMAIL</td>
<td>delivers e-mail messages packaged by MAKEMAIL to an Internet mail server.</td>
<td>Page 220</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
<td>Location</td>
</tr>
<tr>
<td>-----------</td>
<td>------------------------------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>SETHOST</td>
<td>Sets an environment variable to the name of the SOCKETS client based upon the client’s MAC address.</td>
<td>Page 220</td>
</tr>
<tr>
<td>SNTPCLI</td>
<td>Used for getting the system network time information.</td>
<td>Page 222</td>
</tr>
<tr>
<td>SOCKDIAG</td>
<td>A utility used to diagnose the SOCKETS TCP/IP Advanced interface, the stack and the network.</td>
<td>Page 223</td>
</tr>
<tr>
<td>SOCKSTAT</td>
<td>A utility to report the status of TCP and UDP sockets.</td>
<td>Page 224</td>
</tr>
<tr>
<td>TCP</td>
<td>A utility used to examine and change the TCP parameters.</td>
<td>Page 226</td>
</tr>
<tr>
<td>XPING</td>
<td>A utility to send ICMP messages continuously to a specified host and report on replies.</td>
<td>Page 227</td>
</tr>
</tbody>
</table>
Overview
This section describes the SOCKETS configuration commands available when starting and running SOCKETS. To execute any of the commands described in the following sections, insert the appropriate entries in the SOCKET.CFG or another configuration file.

Notations and Conventions
In the command summary, use is made of the hostid notation, which denotes a host, router, gateway, or network. hostid may be specified either by a symbolic name listed in the HOSTS file, or a numeric IP address with decimal notation; for example, 192.10.240.1.

The following conventions apply to command syntax.

- italics indicate that the term is a parameter to be specified by the user.
- [ ] Square brackets indicate that the enclosed item is optional.
- / A forward slash is used as a leading character for an optional switch in some commands. Both the forward slash and the switch that follows it form part of the command syntax. The equal sign before extra parameters is optional in most cases.
- | The vertical bar indicates that there is a choice between two or more selections, but that only one of the options indicated may be specified.
- bold type indicates a reserved key word as part of the command syntax and is to be typed exactly as indicated.
- # Commands proceeded by a hash sign (#) are ignored. They are used for comments in configuration files.

Command Reference

arp

arp adds a new entry to the Address Resolution Cache. arp can also specify the type of ARP used to determine IP address conflicts.

Syntax

- `arp add hostid ether | ieee hw_addr`
- `arp init {DHCP | GRAT | BOTH}`

Remarks

‘arp add’ includes a new entry in the Address Resolution Cache. Do not add entries with duplicate IP or hardware addresses as this will cause malfunctioning of the network.
‘arp init’ specifies the type of ARP used to determine IP address conflicts. By default both DHCP and Gratuitous ARPs are sent to determine if the SOCKETS IP address conflicts with any other IP address.

The DHCP and Gratuitous ARP types are slightly different methods of determining IP address conflicts, but each causes a different set of events to occur. In both cases, if a machine is currently using the IP address specified in the ARP packet, that machine will respond to the ARP request which signals an IP address conflict.

DHCP ARP was designed to be used in conjunction with an IP address received from a DHCP server. A machine could lose its lease an IP address, but continue to use that IP address. As a result, the DHCP ARP is used to make sure that another machine is not currently using the specified IP address. Fortunately, the DHCP ARP can also be used with static IP addresses. A difficulty arises when a Proxy Server receives a DHCP ARP packet, the server responds on behalf of the IP address.

A Proxy Server responds to ARP requests for a set of machines that may not always be available for network traffic. If a machine using that IP address is being served by a Proxy Server, even if the DHCP ARP came from a machine being served by a Proxy Server, it responds to the DHCP ARP packet which signals an IP address conflict.

The Gratuitious ARP can also be used to determine IP address conflicts, but it has one major side effect. When a machine receives a Gratuitous ARP packet, and has the requested IP address in its ARP table, that machine will update its ARP table to match the hardware address contained in the Gratuitous ARP packet. If another machine is currently using the requested IP address, then that machine can no longer communicate with the network until it sends out another Gratuitous ARP.

By specifying BOTH, a DHCP ARP is broadcast to the network, and if no other machine responds, then a Gratuitous ARP is broadcast to the network. The followup Gratuitous ARP causes APR tables with the requested IP address to be updated with the MAC address with the Gratuitous ARP.

**Options**

**hostid**

The IP address of a remote host that is to be added to the ARP cache. This value may be a symbolic name from the HOSTS file or a decimal (dotted) address.

**hw_addr**

Used with add it denotes the hardware (node) address of the remote host whose IP address is given in hostid. This must be a six-digit hexadecimal address separated by colons for Ethernet.

**DHCP**

Used to do only DHCP ARP to determine IP address conflicts.

**GRAT**

Used to do only Gratuitous ARP to determine IP address conflicts.
Both

Used to do DHCP and Gratuitous ARP to determine IP address conflicts. This is the default action.

Example Commands

```
arp add unix_host ether 00:00:65:0D:E6:04
arp add 127.0.1.3 ether 00:00:65:0D:E6:04
arp init grat
```

domain

If a host name is not a decimal (dotted) address and it is not found in the HOSTS file and at least one Domain Name Server has been defined, an attempt is made to obtain the address from the defined DNS server(s). The number of times any server is polled (retries), in addition to the time to wait for a response, can also be specified. A suffix may be specified and is attached to all names not containing any dots.

All of the following sub-commands can be issued without the optional parameters to obtain information on the current status.

Syntax

```
domain server [host_name]
domain retry [retry_count]
domain time [wait_time]
domain suffix [domain]
```

Remarks

- **domain server** adds a DNS address or lists the current servers if host_name not specified.

- **domain retry** specifies the retry count for polling each server. domain retry lists the retry count if retry_count not specified.

- **domain time** specifies the time (milliseconds) to wait for a response before attempting retry. domain time lists the time (milliseconds) to wait if wait_time not specified.

- **domain suffix** specifies the domain suffix to add to all simple names; names that contains no dots. domain suffix lists the domain suffix if domain is not specified.

Example Commands

```
domain retry 3
domain server 196.2.1.1
domain suffix myorg.co.za
domain time 2000
```
iface

iface is a synonym for the interface command.

interface

interface informs SOCKETS of the hardware or software communications interface(s) to be used at the network interface level. At least one network interface is required, and two or more are used in gateway (router) applications.

The class, or mode, of each interface defines the encapsulation used for packaging the data frame into the transport frame. Some types of interface support only one class.

When SOCKETS is defined with multiple interfaces, you first declare an IP address which is associated with the immediately following interface. The defined net mask is then used to add a route through the interface to the connected network. Using the same IP address would result in multiple routes to the same network. The default route is set on the first interface with an IP address with a zero net mask (for example, IP address 19.63.10.11/0).

Each interface command uses the IP address from the last supplied IP address command.

Syntax (general)

interface type name class other parameters

Syntax

interface pdr name dix mtu numbuf [intvec [irq]]
interface asy name [slip | cslip | ppp] mtu buflim ioaddr iovec speed [modemfile]
interface mdd name [slip | cslip] mtu buflim ioaddr iovec speed [modemfile]
interface aslink name [slip | cslip] mtu buflim ioaddr iovec speed [modemfile]

Option

type

Type defines the type of hardware or software interface.
interface supports the following software interfaces.

<table>
<thead>
<tr>
<th>Interface</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>asy</td>
<td>Standard PC asynchronous interface (RS232 port)</td>
</tr>
<tr>
<td>pdr</td>
<td>packet driver interface</td>
</tr>
</tbody>
</table>

name

Name defines the name by which the interface is known on the local host. name is a symbolic name known only to the local host on which it is used.
name may be arbitrarily assigned. Each interface command on the same host must have a unique name assigned. This name is used by other commands referring to this interface, e.g. the param command.

class
class specifies how IP datagrams are to be encapsulated in the link level protocol of the interface. Some interfaces offer a choice between classes while others use a fixed class. The following classes are available and are listed with their associated types.

<table>
<thead>
<tr>
<th>Type</th>
<th>Class (defined in the following list)</th>
</tr>
</thead>
<tbody>
<tr>
<td>pdr</td>
<td>dix, ieee, token, driver, slip</td>
</tr>
<tr>
<td>asy</td>
<td>slip, cslip, ppp</td>
</tr>
<tr>
<td>mdd</td>
<td>slip, cslip</td>
</tr>
<tr>
<td>aslink</td>
<td>slip, cslip</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dix</td>
<td>The DEC/Intel/Xerox Ethernet interface also known as Blue Book Ethernet or Ethernet II.</td>
</tr>
<tr>
<td>token</td>
<td>IBM Token Ring. Source routing is supported for multiple rings.</td>
</tr>
<tr>
<td>ieee</td>
<td>IEEE 802.3 Ethernet with SNAP headers.</td>
</tr>
<tr>
<td>driver</td>
<td>Use the default class for the packet driver.</td>
</tr>
<tr>
<td>slip</td>
<td>Serial Link Internet Protocol (SLIP) for point-to-point asynchronous links. This mode is compatible with UNIX SLIP</td>
</tr>
<tr>
<td>cslip</td>
<td>Compressed Serial Link Internet Protocol (SLIP) for faster reaction over point-to-point synchronous links.</td>
</tr>
<tr>
<td>ppp</td>
<td>Point-to-point protocol over asynchronous links.</td>
</tr>
</tbody>
</table>

mtu
mtu specifies the Maximum Transmission Unit size, in bytes. Datagrams larger than this limit are fragmented into smaller pieces at the IP layer. The maximum value of mtu for the various interfaces is:

Ethernet - 1500

For serial links a standard value for mtu is 576. (576 is the maximum according to specifications, but may be increased on reliable connections as long as both sides use the same value.)

numbuf
numbuf specifies how many incoming datagrams may be queued on the receive queue at one time. If this limit is exceeded, further received datagrams are discarded. This mechanism is used to prevent fast interfaces from filling up memory when data cannot be handled fast enough.
buflim

*buflim* specifies the maximum number of outgoing datagrams or packets to queue before starting to discard datagrams. This mechanism is used to prevent the memory from filling up when a serial link goes down.

bufsize

*bufsize* specifies the size of the ring buffer in bytes to be allocated to the receiver in raw mode.

intvec

*intvec* specifies the software interrupt number (vector) in hexadecimal to use for resident packet drivers.

ioaddr

*ioaddr* is the I/O base address in hexadecimal of a serial port or the hardware controller and must correspond with the jumper or switch settings used during the setup of the controller board. The standard values for serial ports are:

- COM1 03F8h
- COM2 02F8h
- COM3 03E8h
- COM4 02E8h

iovec

*iovec* is the hardware interrupt vector used by the serial port or controller and must correspond with the jumper or switch settings used during setup of the controller. The standard values for serial ports are:

- COM1 4
- COM2 3
- COM3 4
- COM4 3

irq

*irq* is the hardware interrupt vector used by the network interface controller. This is only used for faster response in SOCKETS.

modemfile

A file containing the modem commands and scripts.

speed

*speed* specifies the transmission speed for serial interface devices (baud rate). Before using a serial connection you have to set flow control with the par command.

**Example Commands**

```
interface pdr if0 dix 1500 5 0x60
interface asy ser0 cslip 576 15 0x3f8 4 9600
interface asy p0 ppp 576 5 0x3f8 4 9600 pppmod.mcf
```
**IP**

**IP** displays or sets the values of the options selected when defining the IP (Internet Protocol) host address of the next interface to be defined.

**Syntax**

```
IP address [hostid [/net_bits]]
IP status
IP ttl [number]
```

**Remarks**

**IP address** sets the IP host address of the next interface to be defined. A route is automatically added to each interface for the default or specified net mask for it’s address. To make an automatic route the default, specify the net bits as zero. When specified without the optional parameters, IP address displays the current value(s) of the local host IP address(es). To assign different IP addresses to different interfaces on the same host, an IP address statement must precede each interface definition. The last IP address given is used in case of missing IP address statements. Three invalid addresses with special meaning may be used: 0.0.0.0 is used to specify that the actual address should be obtained by BOOTP in the case of a LAN connection or by PPP negotiation in the case of a serial connection. 0.0.0.1 is used to specify that DHCP should be used to obtain an address. 0.0.0.2 is used to specify that the IP address will be assigned by a user-written program and that the address is inactive until the user-written program is run.

**IP status** displays Internet Protocol (IP) statistics, such as total packet counts and error counters of various types. It also displays statistics on the Internet Control Message Protocol (ICMP). This includes the number of ICMP messages of each type sent or received.

**IP ttl** sets the default time-to-live value which is placed in each outgoing IP datagram. The ttl value limits the number of gateway hops the datagram is allowed to take in order to kill datagrams that got stuck in loops.

**Option**

**hostid**

`hostid` specifies the IP host address to assign to the next interface to be defined. This may be a symbolic name from the HOSTS file, or a dotted decimal address.

**/net_bits**

A net mask can be specified for the host. In the **IP address** command an optional `/net_bits` can be used to indicate the number of bits in the network ID. The net mask is used to determine whether an incoming datagram is a broadcast and also for sending UDP broadcasts.

Net masks are more easily represented in binary or hexadecimal format. For example, the IP address 128.11.5/24 corresponds to a net mask of 255.255.255.0 (FFFFFFFF00h), 25 bits to 255.255.255.128 (FFFFFFFF80h) and 26 bits to 255.255.255.192 (FFFFFFC0h).
The default net mask used corresponds to the class of address used if not explicitly specified.

<table>
<thead>
<tr>
<th>Net Bits</th>
<th>Net Mask</th>
<th>Class</th>
<th>IP address range</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>255.0.0.0</td>
<td>A</td>
<td>0.x.x.x to 127.x.x.x</td>
</tr>
<tr>
<td>16</td>
<td>255.255.0.0</td>
<td>B</td>
<td>128.x.x.x to 191.x.x.x</td>
</tr>
<tr>
<td>24</td>
<td>255.255.255.0</td>
<td>C and higher</td>
<td>192.x.x.x or higher</td>
</tr>
</tbody>
</table>

If you want to subdivide your network, you can divide it by two for every net bit added. The following table provides information on converting between net bits and net mask. The number of net bits to add when changing a 0 in the net mask to:

<table>
<thead>
<tr>
<th>Net Bits</th>
<th>Net Mask</th>
<th>Net Bits</th>
<th>Net Mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>128</td>
<td>5</td>
<td>248</td>
</tr>
<tr>
<td>2</td>
<td>192</td>
<td>6</td>
<td>252</td>
</tr>
<tr>
<td>3</td>
<td>224</td>
<td>7</td>
<td>254</td>
</tr>
<tr>
<td>4</td>
<td>240</td>
<td>8</td>
<td>255</td>
</tr>
</tbody>
</table>

When number is omitted, IP ttl displays the current value of the time to live parameter.

**par**

par invokes a device-specific control routine. par operates differently for each interface type and even interface mode.

**Syntax**

```
par ifname [arg1...argn]
```

**Option**

- **ifname**

  *ifname* defines the name used in the interface command for the device to be controlled.

- **arg1...argn**

  These parameters depend on the type of interface in use.
Example
To change the baud rate of a serial interface and specify software flow control, use:

```
par s10 19200 x x
```

par, Configuring the PPP Interface

The PPP interface is optionally configured by using the par command. Various parameters at the LCP, IPCP and PAP levels can be specified.

Setting PPP Options, Local and Remote LCP/IPCP

When a local option is specified, the value of the option is used in the initial Configuration Request to the peer. Options not specified, are not be requested. For each option, the ‘allow off’ parameter disallows the peer to include that option in its response. By default, all options are allowed in the response, even if the option is not included in the request.

When a remote option is specified, the value of the option is used in the initial response to the configuration request from the peer. If an option is disallowed, it does not allow the remote to specify that option in its request. By default all options are allowed.

Local and remote options are specified by:

```
par iface lcp|lcpin local|remote option [parameters ...] [allow [on|off]]
```

The par command options are as follows:

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>par iface lcp</td>
<td>lcpin localremote acfm bitmap</td>
</tr>
<tr>
<td>par iface lcp</td>
<td>lcpin localremote authent [pap</td>
</tr>
<tr>
<td>par iface lcp</td>
<td>lcpin localremote acfc [onoff] [onoff]</td>
</tr>
<tr>
<td>par iface lcp</td>
<td>lcpin localremote pfc [onoff] [onoff]</td>
</tr>
<tr>
<td>par iface lcp</td>
<td>lcpin localremote magic [onoff]</td>
</tr>
<tr>
<td>par iface lcp</td>
<td>lcpin localremote mr</td>
</tr>
</tbody>
</table>
### Syntax and Description

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
</table>
| `par iface lcp|lcpin [open|start|listen]` | Set the LCP connection mode.  
  start is used to initiate a connection as soon as the stack has been loaded. If a modem is used, this will also initiate an immediate dial-out.  
  open is used to delay a connection until data needs to be sent. If a modem is used, this provides a dial-on-demand facility.  
  listen is used when the peer must initiate the connection.  
  Note that two commands can be given to specify both open and listen in which case a connection can be initiated from either side. |
| `par iface ip|ipcin local|remote adress [ip_address>|allow [on|off]]` | Set the IP address option. If set to 0.0.0.0, the peer must supply it. |
| `par iface ip|ipcin local|remote compress [tcp slots [flag]] | allow [on|off]]` | Set the TCP/IP header compression. slots should be equal to the maximum number of concurrent TCP connections. Low values preserve memory. 4 - 16 are good values. flag = 0 to not compress the slot number; and =1 to compress. The default is 1. |
| `par iface ipv6cp|ipv6cpin local|remote iid [iid|allow [on|off]]` | Set the IPv6 Interface ID |

### Setting PPP Options, Retry Counters

The following retry counters can be set:

```
par <iface> lcp|lcpin retry <configure>|<failure>|<terminate> <count>
par <iface> ip|ipcin|ipv6cp|ipv6cpin retry <configure>|<failure>|<terminate> <count>
```

- **iface** is the name assigned to the interface.
- **configure** is the number of configuration requests (default 20).
- **failure** is the number of bad configuration requests allowed from peer (10).
- **terminate** is the number of termination requests before shutdown (2).
- **count** is the number of retries.

### Setting PPP Options, Timeout Values - in milliseconds

```
par iface lcp|lcpinip|ipv6cpinip timeout milliseconds
```

### Setting PPP Options, Authentication - username/password

For PAP or CHAP authentication on PPP connections:

```
par iface ap user username password
```
Setting PPP Options, Open - a specified layer

```
par iface lcp lcpin ipcp ipcpin ipv6 lcp ipv6 lcpin open
```

Start – an lcp connection

```
param iface lcp lcpin start
```

When `lcp` is specified, a PPP connection is immediately initiated.

Listen – for an lcp connection

```
param iface lcp lcpin listen
```

When `lcpin` is specified, a PPP connection is delayed until an incoming call is detected.

**par, Alternative Routing Control Sub-commands**

The Alternative Routing Control Sub-commands set up and check the SOCKETS alternative route mechanism. More than one route can be specified to a target host or network. The first route that has an associated interface in the up state is used.

An interface is in the up state when it is defined by the interface command. It enters the query state when it does not receive valid input within a specified up-time period after data expecting a response is sent. At this stage three (catering for links with a high data loss) ICMP echo requests (ping) are sent to a query IP address. It enters the down state by a SOCKETS command or when it does not receive valid input within the specified up-time period after entering the query state. If an up time has never been specified or a value of 0 is specified, the interface will stay in the up state whether valid input is received or not.

An interface enters the up state by a SOCKETS command or when valid input is received on that interface while it is in the down or query states. An ICMP echo request is sent on an interface in the down state every downtime period. If a downtime has never been specified or a value of 0 is specified, the ICMP echo request is not sent. Up time and downtime is specified in seconds.

**Syntax**

```
par ifname [ uptime | downtime ] time
par ifname query hostname
```

**Options**

- **Ifname**
  
  *Ifname is the interface name of an asy interface.*

- **Uptime**

  *Uptime is the time to allow for no response on a defined connection.*

- **Downtime**

  *Downtime is the period of time to retry a defined connection.*
Example Alternative Routing

Two SLIP interfaces are used to get to the target network 192.6.1.0. The first interface, named if0 should preferably be used, but if it stops receiving for a period of 20 seconds, it should try to ping 192.6.1.2 and if no response is received within another 20 seconds, if1 should take over, but if0 should be tried every five seconds. Interface if1 should disconnect after 80 seconds of no traffic.

The SOCKET.CFG file should contain the following:

```
interface asy if0 slip ...
par if0 uptime 20 par if0 downtime 5
par if0 query 192.6.1.2
interface asy if1 slip ...
par if1 uptime 80 par if1 downtime 5
par if1 query 192.6.1.2
route add 192.6.1.0 if0
route add 192.6.1.0 if1
```

In the case of both if0 and if1 failing, both will retry every five seconds until one comes up. The return paths should also be maintained in a similar way with SOCKETS or by using RIP.

par, COM Port Speed and Flow Control Sub-command

This par sub-command allows the baud rate, flow control and data parameters to be set and is only to be used on asy type interfaces.

Syntax

```
par ifname speed outflow inflow bits
```

Options

ifname

Ifname is the interface name of an asy interface.

speed

Speed sets the baud rate for a serial link. The standard speeds are: 150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400 and 57600.

outflow

Outflow sets the output flow control for a serial link. When not supplied it defaults to none.

inflow

Inflow sets the input flow control for a serial link. When not supplied it defaults to none. The following list shows outflow and inflow selections.

no flow control

- Xon Xmit on/Xmit off (Ctrl-Q, Ctrl-S)
- DCD Data Carrier Detect modem signal
- CTS Clear To Send modem signal
DSR Data Set Ready modem signal
DTR Data Terminal Ready modem signal
RTS Request To Send modem signal
ixxx inverse of modem signal xxx

Invalid selections are ignored or replaced by more logical selections. The usage of Xon/Xoff is not recommended for most applications. The ixxx support is for non-standard equipment that uses a reversed signal on the required pin.

bits

bits sets the number of data bits per character, number of stop bits, and parity for a serial link. It is a three-character string consisting of dsp from the following table. When not supplied, bits defaults to value of 81n.

<table>
<thead>
<tr>
<th>Values for d</th>
<th>5, 6, 7, or 8 data bits</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Values for s</th>
<th>1 or 2 stop bits</th>
</tr>
</thead>
</table>

| Values for p | o = odd parity  
|-------------|-----------------|
|             | e = even parity 
|             | n = no parity 
|             | m = mark 
|             | s = space |

Example

```
par asyl 57600 cts rts 81n
par s1p0 4800 xon xon 72e
```

par, RIP Advertising Sub-command for Interfaces

When the RIP advertise command has been used, this par sub-command makes allowance to disable and re-enable RIP advertising on a specific interface.

Syntax

```
par ifname [ ripadv | noripadv ]
```

Options

Ifname

`ifname` is the interface name of an asy interface.

Ripadv

Ripadv indicates to use route advertising on the defined interface.

Noripadv

Noripadv indicates to not use route advertising on the defined interface.

Examples

```
par if0 noripadv
par if1 ripadv
```
printer

The **printer** redirector operates by intercepting Interrupt 17 and passing printer output to a local or remote print server. A print session is started when output is first sent to a specific port, and is stopped after a user-specified period of no output. When a printer port is redirected, the IP address and TCP port number of the destination print server, in addition to the timeout period, is specified. A printer port can both be used as a print server port and as a redirected port.

**Syntax**

```
printer printer_port timeout IP_address [TCP_port]
```

**Options**

- **printer_port**
  - Printer port 0, 1, 2 or 3 to redirect. Port 0 corresponds with PRN, 1 with LPT2 and so on. Serial printers can be defined with the SRPRINT.EXE TSR that forms part of the SOCKETS package.

- **timeout**
  - Timeout period in seconds for closing the TCP connection.

- **IP_address**
  - IP address of print server host (could be the local host).

- **TCP_port**
  - TCP port of print server (Default is 10).

**Comments**

To ensure proper queuing of remote and local print sessions to a local printer, the printer port must be redirected to the local IP address and TCP port of the print server using that port.

**Examples**

The IP address of print_host in the following example is defined in the HOSTS file.

```
printer 0 15 print_host
```

The following example uses printer port 1 (LPT2) for both local and remote printing. (If the printer command is placed before start prntserv, SOCKETS will not work correctly.)

```
ip address this_host
.
.
start prntserv 1010 1
printer 1 15 this_host 1010
```
RIP

The Routing Information Protocol (RIP) allows a SOCKETS gateway to advertise routes and allows SOCKETS to recognize advertised routes from SOCKETS and other gateways. These two facilities can be individually selected.

Syntax

```
RIP advertise [time [1|2] [ self ]]
RIP use [time ]
```

Options

time

time is the number of elapsed seconds before the advertisement is repeated when used in the advertise sub-command.

```
time is the period during which routes are added or amended as a result of RIP, and are valid for time seconds. Such added or amended routes are dropped if another RIP advertisement is not received within time
```

self

```
self advertises SOCKETS connection to the network.
```

RIP Advertise Sub-command

RIP advertise causes an immediate advertisement of all relevant routes and repeats it every time seconds. The default value for time is 30 seconds. RIP version 2 advertisements are sent by default. To send only version 1 advertisements, add a 1 on the command line.

When RIP advertise is selected, all interfaces advertise all routes except those routes making use of that specific interface (split horizon) and routes marked Private. (To prevent certain interfaces from using RIP, see the parameter command.) A route which is dropped as a result of a RIP update or which becomes unavailable as a result of its associated interface going into the down state, is immediately advertised as being infinite (metric = 16) and is not advertised until it becomes available again. The advertisement is sent as a sub-net broadcast using the net mask and IP address of the interface.

Example

```
RIP advertise 30 self
```

RIP use, Update RIP Sub-command

RIP use causes a RIP request for routes and a continuous update of routes according to RIP advertisements. Routes added or amended as a result of RIP are valid for time seconds and are dropped if another RIP advertisement is not received within that time. The default value of time is 240 seconds.

When RIP use is selected, routes are updated according to received RIP advertisements. Routes added or amended as a result of RIP, have a timeout associated with them. If another RIP advertisement is not received during that time, the route is dropped. A route is also
dropped if an advertisement of infinity (metric = 16) is received. To prevent dropping a route, it must be marked as Static.

On RIP and static routes: — The metric of a route marked static is never updated by a RIP advertisement. Instead a duplicate route is added before the static route. If the duplicate route is dropped as a result of a timeout or RIP, the static route is used again.

On RIP and mdd or x25 interfaces: — On mdd or x25 interfaces, a RIP route is only sent when the last datagram was not a RIP route. This is done to allow an interface to time out and be disconnected when not used.

Also, the route on the receiving end still times-out and becomes unavailable. This means that inactivity timers should be shorter than route timers; as specified in RIP use timer.

When an aslink to mdd association is broken, all non-static routes using the specific mdd are marked as having a hop-count of infinity; the metrics are set to 16.

**route**

*route* creates an entry in the IP routing table for SOCKETS to determine where to send data. The Alternative Routing mechanism allows more than one route to be specified to a particular host or network. Failure of one route causes an automatic switch to the next route.

Refer also to the *ip address* command for specifying the net mask, because a route is automatically added to each interface for the default or specified net mask for that address. When multiple routes are defined to the same address, SOCKETS uses the route with the network size (largest number of bits in the net mask).

**Syntax (general)**

```
route [ add | drop destination ifname [gateid |none [ metric [proxy] [private] [static] ] ] ]
```

**Syntax (specific)**

```
route
route add [ hostid | netid ] ifname [gateid]
route add [ hostid | netid[/mask] ] ifname [gateid]
route add default ifname
route drop [ hostid | netid ]
route drop [ hostid | netid[/mask] ]
route drop default
```
Options

add or drop
   Sub-command to add or drop (remove) a route from the routing table.

default
   All transmissions to IP addresses not otherwise defined in routing commands are sent via
   the network interface specified by ifname.

hostid
   hostid is the IP address of a destination remote host to which data must be sent, or a remote
   host that must be removed from the routing table (dropped).

netid
   netid is the IP address of a destination network to which data must be sent. Any host with
   this IP network address is able to receive the data. Whether a particular host will use the
   data depends on the host portion of the specific IP address in the IP header.

mask
   mask specifies the number of bits in the network portion of the address if sub-netting is
   used. If not used, the network portion of the address is determined according to the class
   (A, B or C) of the address.

ifname
   ifname defines the name used in the interface command for the immediate network on
   which the data for the designated host must be sent. This is the network level interface to
   be used by the local host to reach the remote host.

gateid
   gateid parameter specifies the IP address of a host, on the same physical network as the
   local host, which is used as a gateway or router to a different network. The gateway or
   router host specified in gateid must be directly reachable on the same physical network as
   the local host defining this gateway. In other words, this must be the nearest gateway to this
   local host.

metric
   When using RIP or Proxy ARP a value from 0 to 16 for metric must be specified indicating the
   distance or cost of that route. A metric of 16 indicates that the route is down.

proxy, private and static
   To support the Routing Information Protocol (RIP) the route command utilizes the proxy,
   private and static key words. These words can be used in any order following metric.

Proxy ARP should be used with care and not in conjunction with RIP. When more than one
host responds to an ARP request, it can cause confusion and even lead to system crashes.
This is possible in situations where more than one gateway implements Proxy ARP to a
common destination. See also ROM-DOS Developers’ Guide, “About TCP/IP” for more
information.
When “RIP advertising” is selected, all interfaces advertise all routes except those routes making use of that specific interface (split horizon) and routes marked private. A route which is dropped as a result of a RIP update or which becomes unavailable as a result if its associated interface going into the down state, is immediately advertised as being infinite (metric = 16) and is not advertised until it becomes available again. In order for an interface to be used for advertising, a route without a gateway using that interface must be available. The advertisement is sent as a sub-net broadcast using the net mask of the host and the IP address of the interface.

When “RIP using” is selected, routes are updated according to received RIP advertisements. Routes added or amended as a result of RIP, have a timeout associated with them. If another RIP advertisement is not received during that time, the route is dropped. A route is also dropped if an advertisement of infinity (metric = 16) is received. To prevent dropping a route, it must be marked as static. The metric of a route marked static is never updated by a RIP advertisement. Instead a duplicate route is added before the static route. If the duplicate route is dropped as a result of a timeout or RIP, the static route is used again.

**Examples**

```
route add default ipx0
route add unix_net eth0
route add unix_host ipx1 unix_gate
route add unix_net2 eth0 /eth 1
route add unix_ net ipx0 unix_ gate
route add subnet/26 eth0 sub_gw
route drop unix_net
```

route can specify a Proxy ARP on a route, as follows:

```
route add net interface gateway metric [proxy]
```

When using Proxy ARP, gateway and metric must be specified. If no gateway is used, none can be specified. For example:

```
route add 192.6.1.0 ifx25 none 5 proxy
```

---

**start**

**start** starts a SOCKETS Print server (prntserv) or LPD and makes it available for access by remote hosts.

**Syntax**

```
start prntserv [port [printer_number]] [/m=mask] [/s=status] [/r=report] [/d=print_delay]
start lpd [printer_number] /n=printer_name [/m=mask] [/s=status] [/r=report] [/d=print_delay] [/f]
```
Options

port

Port is the TCP port to use (the default is 10).

printer_number

printer_number is a value of 0, 1, or 2 to correspond with LPT1(PR1), LPT2 and LPT3. Any number up to 3 may be defined by the serial printer TSR (SRPRINT) as a printer on a COM port. The default printer number is 0.

Note: Both LPD and prntserv may use the same printer.

printer_name

The name of the printer queue.

mask

Used to activate status reporting and cope with non-standard printer status reporting.

status

Used to activate status reporting and cope with non-standard printer status reporting.

report

Used to activate status reporting and cope with non-standard printer status reporting.

print_delay

A decimal number specifying the number of times a printer will be tested for busy status before the busy status is accepted. This solves the problem that a single character is sent to the printer every timer tick (55 milliseconds), resulting in extremely slow printing by the print server. print_delay should be the lowest value still permitting fast printing. The default value is 40 which works fine on a 33 Mhz 386 CPU. On a slower CPU the value can be less and on a faster CPU it may need to be more.

/f

Send a form-feed (new page) to the printer at the end of the print job.

Printer Status Reporting

Optional status reporting has been implemented to give print client implementations more control over print jobs. The following print server enhancements have been enabled:

1. Optional status reporting to Print Client. The status reporting includes messages like:
   Timeout
   Last character has been sent to printer
   I/O Error
   Printer selected
2. Customization of printer status checking.

3. Print queue flushing.

Raw printer status bytes as defined for BIOS INT 17 services can be sent from the server to the client on the established connection when:

1. The first data is being printed.

2. The status changes while attempting to print.

3. The last character in the stream (the character before the FIN) has been sent to the printer. The unused bit Bit1 in the status is used to indicate this event.

The status byte is defined as follows:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>80h</td>
<td>Printer ready</td>
</tr>
<tr>
<td>6</td>
<td>40h</td>
<td>Acknowledged</td>
</tr>
<tr>
<td>5</td>
<td>20h</td>
<td>Out of paper</td>
</tr>
<tr>
<td>4</td>
<td>10h</td>
<td>Printer selected</td>
</tr>
<tr>
<td>3</td>
<td>08h</td>
<td>I/O Error</td>
</tr>
<tr>
<td>2</td>
<td>04h</td>
<td>Unused</td>
</tr>
<tr>
<td>1</td>
<td>02h</td>
<td>Last character has been sent to printer</td>
</tr>
<tr>
<td>0</td>
<td>01h</td>
<td>Timeout</td>
</tr>
</tbody>
</table>

The status bits are returned by the Int 17 BIOS call (with the exception of Bit1) and are not always consistent, but depend on the BIOS of the particular server.

To print successfully, three conditions must be met: 1) the printer must be selected, 2) be ready, and 3) contain paper.

Sum the hexadecimal values for these conditions and use that as the /s and /m parameters for the start prntserv commands. The default values used are /mA8 for the mask and /s80 for the status which implies that data is sent to the printer when it is not busy and out-of-paper, or I/O Error status is not set. Printer selected is ignored as this status bit is often not reported correctly. To include the printer selected bit, specify /mB8 and /s90.

**Example**

```
start prntserv 10 0
```
Notes on SOCKETS printing

The socket print server accepts TCP connections and routes output to one of four system printers on a parallel port or serial port. Up to four print servers can be started on different TCP ports, connecting to different printers. The server accepts multiple calls, but prints strictly in order of received calls. This means that a connection must be closed to allow another client to print, thereby providing print queuing. There is no indication of queuing status, so a non-ready printer will not hang SOCKETS. Other operations are not affected.

The print server is instructed to send status bytes in the start prntserv command by specifying an optional argument /r=report where report is a hexadecimal value specifying the status bits which are checked for changes to initiate a report. (A report is always sent at the beginning of a job.) The default value is 0 that means that the print server will not report any status changes. A logical value to use is /r=3F which causes any changes, except Printer ready and Acknowledged, to be reported. (The Acknowledged bit is not useful in this application because it toggles at each character; too fast for most networks to carry all the generated traffic.) Note that the full printer status byte is transmitted without filtering any bits.

Notes on printing

Provision is made for non-standard status responses from printers. The mask specifies a hexadecimal value to perform a logical AND operation with the status indication from the printer and status is the hexadecimal value of the result which causes the printer to print. These options are only used locally at the server and the result is not passed on to the client. Change these options only when error conditions (for example, printer switched off or printer off-line) causes output to be lost. The default values used are 0xA8 for the mask and 0x80 for the ready status which implies that data is sent to the printer when it is not busy and Out of paper or I/O Error status is not set. Note that Printer selected is ignored as this status bit is often not reported correctly. To include the Printer selected bit, specify /m=B8 and /s=90.

tcp

tcp commands display or set various TCP operating parameters.

Syntax)

tcp ackonpush 1
 tcp irtt [time]
tcp lport [port_number]
tcp mss [size]
tcp retry [number]
tcp rtt [time]
tcp smss [size]
tcp timemax [time]
tcp window [size]
Options

time

_time is the new time value in seconds, or milliseconds if “ms” is appended to the number, as in 2000ms.

port_number

_port_number is the local port starting number.

size

For tcp mss, _size is the maximum segment size in bytes sent on all outgoing TCP connect requests (SYN segments). size tells the remote host the size of the largest segment that may be received by this host. When changing the MSS value, any existing connections remain unchanged.

For tcp smss, _size is the send maximum segment size in bytes sent on all outgoing TCP connect requests. This limits the size of the largest segment that may be sent by this host. When changing the SMSS value, any existing connections remain unchanged.

For tcp window, _size is the size of the receive window in bytes for any new TCP connections. Existing connections are unaffected.

number

_number is the number of retries attempted without receiving an acknowledgement from the remote host before the connection is broken. If the value exceeds 255, it implies an infinite number of retries; such a connection does not time-out. The default value for _number is 6.

tcp ackonpush Sub-command

tcp ackonpush disabled “Delayed ACK”. Normally TCP acknowledgment should occur as RFC1122 states:

“4.2.3.2 When to Send an ACK Segment

A host that is receiving a stream of TCP data segments can increase efficiency in both the Internet and the hosts by sending fewer than one ACK (acknowledgment) segment per data segment received; this is known as “delayed ACK” [TCP:5].

A TCP SHOULD implement a delayed ACK, but an ACK should not be excessively delayed; in particular, the delay MUST be less than 0.5 seconds, and in a stream of full-sized segments there SHOULD be an ACK for at least every second segment.”

The purpose of using “delayed ACK” is to decrease the amount of traffic generated on a TCP connection. In certain applications a delayed ACK can lead to a slower response. The user can now disable the “delayed ACK” functionality by using this command in the SOCKETS configuration file.

ACKONPUSH can be abbreviated to ACK.

Example

tcp ackonpush 1

tcp ack 1

tcp irtt Sub-command

tcp irtt displays or sets the initial round-trip-time estimate. When specified without an argument,
the command displays the current values of TCP parameters including the initial round-trip-time in milliseconds.

time is the initial round-trip-time (IRT) estimate and is used for new TCP connections until the actual value can be measured and adapted to. By increasing this value when operating over slow communication links, unnecessary retransmissions that otherwise occur before the smoothed estimate value approaches the correct value are minimized. The system default is 5000 milliseconds.

To affect incoming connections, tcp irtt should be executed before the servers are started.

Example

tcp irtt 120

Sample Output

TCP: IRTT 5 ms Retry 6 MSS 1460 SMSS 1460 Window 2920

tcp lport Sub-command

tcp lport specifies the local port starting number. When specified without a number the current value of the next free local port number is displayed.

Example

tcp lport 2004

Sample output

lport = 2004

tcp mss Sub-command

tcp mss displays or sets the TCP maximum segment size in bytes. When size is not specified, the current values of the TCP parameters, including the maximum segment size, are displayed. It is recommended to reduce the MSS and SMSS on bad network connections. The SOCKETS queuing overhead is 12 bytes per WriteSocket request i.e. you will need 52 bytes for each request. It will queue 2*MSS bytes for you if it can't send right away because of window constraints by the peer or the Nagle heuristic. That means that if your MSS is set to the default of 1460, 2920 bytes will be queued, consuming 3796 bytes in your case. If you have 5 connections, you need 18980 bytes just to buffer your outgoing data. Use IPSTAT to determine how much memory you have available for everything. (You need memory for each connection, incoming data and all other TCP functions).

Example

tcp mss 1460

tcp retry Sub-command

tcp retry displays or sets the retry count before a connection is broken. When specified without the number parameter, tcp retry displays the current values of TCP parameters, including the retry count. Refer also to “TCP Retry Strategy” on page 265.
Chapter 12 | Sockets Configuration Commands

**tcp rtt Sub-command**

tcp rtt replaces the automatically computed round-trip time (RTT) for the specified connection with the time in milliseconds. SOCKETS calculates the RTT as a smooth average of past measured RTTs, starting with the IRTT on a new connection. To get the current RTT in use for a connection n, use the tcp status n command that gives the smoothed average RTT indicated by SRTT. Because tcp rtt provides a manual override of the normal back-off retransmission timing mechanisms, it may be used to speed up recovery from a series of lost packets.

**Example**

tcp rtt 4 100

**tcp smss Sub-command**

tcp mss displays or sets the TCP send maximum segment size in bytes. When size is not specified, the current values of the TCP parameters, including the SMSS, are displayed. A small SMSS causes the remote to reduce its segment size. tcp mss can reduce the MSS and SMSS on bad network connections with high loss rates or where large packets get lost.

**Example**

tcp smss 512

**tcp window sub-command**

tcp window displays or sets the default and maximum receive window size. When specified without the size parameter the current TCP parameters, including the current window size, are displayed.

**Example**

tcp window 2920

**tcp timemax Sub-command**

tcp timemax sets the maximum duration of a tcp retry. If a value greater than 255 seconds is specified, connections never timeout. This is very useful in wireless applications where nodes roam in and out of service.

**Example**

tcp timemax 2000ms
SOCKETS COMMAND LINE OPTIONS

The command line options for SOCKETS, SOCKETM and SOCKETP are identical.

Syntax

Sockets [/options...] [config_file [arguments_for_config_file...]]

Options

/?
/a

/n=normal_sockets
/i=capi_interrupt
/d=dos_compatible_sockets
/h
/m=memory_size
/p
/s=stack_size
/v=api_interrupt
/q
/u

config_file

The configuration file to use.

If config_file is omitted, SOCKETS searches the following paths:

%SOCKETS%\DOS\SOCKET.CFG
%SOCKETS%\SOCKET.CFG SOCKET.CFG

If the SOCKETS environment variable is not set, %SOCKETS% above defaults to

\DL\SOCKETS

arguments_for_config_file

Can be used to replace %1 to %9 in the configuration file. It is often used to set the IP address
or other variable parameters in the configuration file. It can also be used to simplify many
functions.

/?
/h

Displays help about command line options.

/0

Instrucets SOCKETS to not attempt to save and restore the extended register set of an 80386
processor. This is the only 386-specific code in SOCKETS/DOS, and disabling it makes
SOCKETS/DOS fully 80186 compatible.

/a

Instrucets SOCKETS to enable the reception of Multicast packets by the Packet Driver. Use this
when using multicast functions.
/a2
Instructs SOCKETS to ignore the reception of locally looped-back Multicast packets by the Packet Driver. This is required to run IPv6 Sockets on a Windows system using NDIS3PKT.

/n=normal_sockets
The number of normal sockets to reserve for use by the Compatibility API. If that API is not needed or no normal socket function is used, set this to 0. This value defaults to 8 i.e. /n=8.

/d=dos_compatible_sockets
The number of DOS compatible sockets to reserve for use by the Compatibility API. If that API is not needed or no DOS compatible function is used, set this to 0. Note that the TCP/IP SOCKETS API (BSD) does not use any DOS compatible sockets. This value defaults to 12 i.e. /n=12.

/i=capi_interrupt
The compatibility API uses interrupt 61hex i.e. /i=61 To change it to a different value, specify the value in hexadecimal eg. /i=62.

/v=api_interrupt
The SOCKETS API normally uses interrupt 7Fh i.e. /v=7f To change it to a different value, specify the value in hexadecimal eg. /v=7e.

/m=memory_size
Specifies the working memory or heap size for SOCKETS and has a default value of 16384 (16KB) for SOCKETP and 22528 (22KB) for SOCKETM. Increase this value if IPSTAT shows Memory allocation failures during operation.

/p
Sets the minimum size for allocation requests from packet drivers to 64 bytes. Some packet drivers request a specific amount of memory, but proceed to write a minimum of 64 bytes to the allocated memory space.

/s=stack_size
Specifies the stack size used by SOCKETS and has a default value of 2048 (2k). Increase this value if problems are experienced when using nested Async Notification handlers.

/q
Load SOCKETS without displaying any startup messages or diagnostics.

/u
Run SOCKETS with the /u switch to unload it from memory. If another program has taken over the interrupts SOCKETS has hooked, SOCKETS just disables itself when the /u switch is used.

MODEM CONFIGURATION FILE SYNTAX
A modem configuration file is a text file containing lines defining modem “scripts” defining the operation of a modem to make and receive calls. A modem configuration file can be given any name which is referenced in the interface command.

A modem configuration file consists of separate commands lines, each starting with one of the following characters in the first column:

i initialisation string/script (for the modem)

m time modern pacing - send a script character to modem every time milliseconds. Default is 55 ms. Use m 0 for no pacing.

n telephone number to dial (one number per line)
retry_count (when a connection or script failed)

exchange_id (XID for user identification)

dial command/script (talk to modem till connected)

answer prompt script (answer incoming call)

connect string/script (initiate outgoing call)

hangup string/script (end the call)

parameters (for debugging/watching and dial control)

timeout value - Lower DTR for time milliseconds to disconnect modem. Default is 10,000 milliseconds.

command_character (default is @ in this syntax)

what all good programmers do

parameters comprise of an optional string specifying whether dial communications must be displayed and how it must be controlled. Use x to display transmit data and r for received data. The display is only active while DCD is low. Use d to always dial when DCD is or becomes low. Use n for no dial on demand. Note that d and n are mutually exclusive. To specify that no DCD signal is available use a c.

The initialisation strings, answer prompt, connect string, dial commands and hangup string consist of modem and login commands or prompts and special functions to cause delays and wait for DCD or strings. The simple scripts are one line strings where commands start with the command_character (default is @) followed by a script command, followed by a time in milliseconds (and a receive search string in the case of @r). The following script commands can be used:

@t time

@w time wait for the full time specified

@d time wait for DCD (modem to get connected)

@f time wait for ^F XID ^M and find mdd with the matching XID

@a time wait for IP Address (anywhere in data stream) and use it as your own for this interface

@r time string@r terminate time if followed by a string starting with digits

string send string to modem/remote (default in all scripts)

@n insert telephone number (used in dial string)

@x insert XID (used in connect string)

@@ send command_character (@) to modem/remote
Note:

- The time is given in milliseconds and is terminated by the first character that is not a decimal digit. Maximum time is one hour (3 600 000ms).

- The script is aborted when a conditional wait times out, without making the connection.

- Receive strings (@r) are terminated with a command_character (@).

- Send strings are terminated with the next command or the end of a line. A carriage return character (CR) is not automatically added.

- To put control characters in the string use ^n where n is A for 0x01, B for 0x02 and so on. A CR is ^M and a LF is ^J. Use ^SPACE to send the ^ character to the modem.

- Do NOT include spaces as separation characters since all characters are interpreted.

- The @’s are strictly interpreted from left to right - do not confuse them with terminating and initiating @’s. After @r, a terminating @ must follow. An initiating @ for the next command MUST be given - it is not automatically assumed since a send string is implied by default.

Modem dial and answer process

Here is a brief discussion of the modem dial and answer processes:

The initialisation script line is executed when Sockets starts up, when a script fails to complete after re-tries and after a modem connection has been broken.

The dial script line is executed when data is to be sent and dial-on-demand is active, when dial-when-not-connected is active (specified by ‘d’ in the (p)arameter line) or when the user program requests a dial using IfaceIOCTL().

The connect script line is executed when (d)ial completes successfully.

The answer script line is executed when a DCD signal is available and DCD is raised without a dial operation in progress or when a DCD signal is not available and data is received with the modem in an idle state.

The hangup script line is executed whenever the modem connection must be terminated. If no hangup line is given, DTR is used to terminate the modem connection.

The modem is connected when the connect or answer script completes successfully or when dial completes in the absence of a connect line or the answer script would have been executed, but had not been specified.

Example

@w1000 wait for 1 second
@d30000 wait for DCD, retry dial after 30 seconds.
@@ send @ to modem/remote
atdt@n^M^J dial the next number in the list
@r2000login@ wait for the string "login"

An example or two is the fastest way to understand the explanations above. For a simple
connection where no passwords or identity checks are needed, try the following:

# Modem definitions for Zoltrix Hayes compatible modem
# This example is for 'asy' interfaces making outgoing
# connections with no logon sequence.
#
# initialisation string
# (Warning: consult the manual for your own modem)
# send "a", wait half a second, send "a", wait half a second,
# send "ats0=0" - do not use auto answer
# send "&c1" - use state of carrier for DCD
# send "&d3" - disconnect when DTR low
# send "&k3" - enable RTS/CTS flow control
# send "&q5" - select error correction
# send "<CR><LF>", wait 100 milliseconds
i a@w500a@aw500ats0=0&c1&d3&k3&q5^M^J@w100
#
# dial command - send "<CR><LF>", wait 2 seconds
# send "atdt<number><CR><LF>"
# wait 30 seconds for DCD
d ^M^J@w2000atdt@n^M^J@d30000
#
# two numbers to dial in rotation
n 790-1234
n 790-1235 #
# parameters: r=show modem receive, x=show modem xmit
p r
#
# number of retries
r 5

A more typical login sequence (where a user ID and password is required) will use the connect
script as in this example:

# modem definitions for US Robotics Hayes compatible modem
# send "a", wait a tenth of a second
# send another "a", wait a tenth of a second,
# send "at&c1" - use state of carrier for DCD
# send "&d3" - disconnect when DTR low
# send "&i0&r2&h1" - enable RTS/CTS flow control
# send "&m4" - select error correction  
# send "s0=0" - do not use auto answer  
# send "<CR><LF>", wait 100 milliseconds  
1 a@w100a@w100at&c1&d3&i0&r2&h1&m4s0=0^M^J@w100  
# number to dial  
n 03456789  
# number of retries  
r 4  
# dial command - send "<CR><LF>", wait 1 second  
# send "atdt<number><CR><LF>"  
# wait 40 seconds for DCD  
d ^M^J@w1000atdt@n^M^J@d40000  
#  
# connect script - wait 7 seconds for login prompt  
# send user ID, wait 2 seconds for password prompt  
# send password, wait 2 second for acknowledgement  
c @r7000login@myuserid^M@r2000sword@mypassw^M @r2000Welcome@  
# parameters: r=show modem receive, x=show modem xmit  
p rx  

Note that the '@m's in the above connect string are not commands since the @ is interpreted as the end of a @r 'wait for receive' string. Sometimes you would have two @'s like this example: Should you want to put a small delay before replying to a prompt it would contain @@ as follows:  
c @r7000login@@w100myuserid^M  
A typical dial script line when DCD is not available, would be:  
d ATDT@n^M^J@w20000CONNECT@  
The modem will be connected when CONNECT is received within 20 seconds after sending the dial string and no connect is specified.  

A typical answer script line when DCD is not available, would be:  
a @r10000RING@  
The modem will be connected if RING is received within 10 seconds after receiving input. It is recommended to always specify an answer script line when DCD is not available since any noise on the line while the modem is idle, will be regarded as an incoming connection which will make it impossible to dial without disconnecting first. If no incoming call is to be accepted, use an answer string which will fail quickly e.g:  
a @w100Wait 100 milli-seconds for this unlikely string!@  
In this case it would be prudent to disable modem auto-answering in the init string as well.  

For receiving incoming calls, the answer command/script is used. You can use it to do a single user ID/password controlled login, just a user ID login, or unconditional acceptance.
The more general case is to use mdd interfaces with an XID (exchange ID) to validate multiple users. Examples for using Multi Destination Drivers (MDD) are given below. If your SOCKETS workstation accepts only incoming calls, you do not need to enter dial commands.

**Retry Strategy on Time-out**

For outgoing calls, when a wait for DCD (@d) times out without receiving DCD, SOCKETS will drop the call (by lowering DTR). If the number of retries has not been exhausted, SOCKETS will retry the dial command with the next phone number rotating through the list of numbers. If a connection with the remote modem is successful (received a DCD) and there is a timeout on the wait for string command in the connect script, the connect script is re-started for the number of retries specified. If the number of retries has been exhausted, SOCKETS will retry the dial and connect commands only after being prompted again by receiving traffic for this destination host or net.

For incoming calls, when there is a time out on the wait for XID or string commands, the answer- prompt script is re-started for the number of retries specified. If the number of retries has been exhausted, SOCKETS will drop the call by running the hangup command or by lowering DTR.

**Delayed LAN configuration**

This feature allows an ethernet connection to be initialized by the application when the connection is required. This allows the SOCKETS configuration parameters to be read from a device, such as EEPROM. To do this, SOCKETS is configured with an “inactive” IP address of 0.0.0.2.

The SOCKETS configuration file will look something like the following:

```plaintext
# assign "inactive" IP address
ip address 0.0.0.2
# configure the interface
iface pdr if0 dix 1500 10
# display the assigned IP address
ip address
# display the default route route
# display the Initial Round Trip Time value
tcp irtt
```

A utility, DCONFIG.EXE, is provided with source code as an example of how an application can configure the ethernet connection. DCONFIG.EXE and the corresponding source file, DCONFIG.C, can be found in the UTILS folder of the SOCKETS directory tree.

**Multi Destination Drivers**

When you have more than one destination to or from dial-up links using SLIP, and are using more than one modem, you need a mechanism to link the logical interface (with IP address and routing info) to the physical interface (COM port with modem definitions). Using an asy type interface permanently links the logical and physical interface with one IP address. This
works OK for dialling out to multiple destinations only where you are the end user and nobody routes through you.

Two interface types - **mdd** and **aslink** - have been defined to enable dial-up operation to other networks using SLIP.

- **mdd** defines a logical interface with associated IP address, routes, MTU, buffer limit and a modem configuration file but without a physical interface. For each dial-up destination a **mdd** interface is created, and a route to each destination specified.

- **aslink** interface defines an uncommitted asynchronous interface with an associated COM port as specified by the I/O address, and interrupt vector.

When initial traffic is to be sent to a destination through a **mdd** interface, SOCKETS scans all **aslink** interfaces for the first available one and assigns it to the **mdd** interface. This association is broken when the dial connection is broken or when the dial attempt fails. A dial attempt fails when the retry count specified in the modem file expires.

Incoming calls can be received on any **aslink** interface. This interface must then be associated with the correct **mdd** interface (and IP address with route) for the calling host. An exchange identifier (XID) defined by both the calling host and the called host achieve this.

The protocol for exchanging the XID is implemented partly by SOCKETS and partly by the user defined ‘connect’ and ‘answer’ scripts in the modem configuration files. When an incoming call is received by an **aslink** interface, as seen when DCD is raised, the ‘answer’ script is executed. This script must contain a command to wait for the XID and match it to a local **mdd** containing the same XID and an acknowledgement to the sender that the exchange has been successful. The XID must be preceded by an ACK (^F) and followed by a CR (^M). The ‘connect’ script of the calling host is used for this purpose. The following ‘answer’ and ‘connect’ scripts may be used:

**a** @w100^M^JYour ID?@f1000ccc

This means: Wait for 100 milliseconds (after receiving the DCD) send a CR/LF to write on a new line and prompt the user with ‘Your ID?’ Wait 1 second to receive a valid XID matching with an mdd interface and then send a few ‘c’ to acknowledge ‘connected’ for the caller.

**c** @r500ID?@^F@x^M@r500c@

This means: Wait half a second to receive the ‘ID?’ prompt, send the required ACK (^F), XID (@x) and CR (^M). Wait another half second for acknowledgement from the remote with a ‘c’.

It can occur that a request for making an outward connection and an attempt by an incoming connection clash at the same **mdd** interface. One example is when a connection is broken and both sides redial each other to restore the connection. A mechanism must be used to allow only one of the attempts to be successful. Therefore a **mdd** interface will drop an **aslink** trying to make an outgoing connection as soon as it senses another **aslink** trying to connect to it with an incoming connection.
In summary the modem commands used by the mdd and the aslink in their respective modem files for the following functions are:

- For initiating the modem: Always use the ‘i-’ command in the aslink modem file.

- For making a call: Use the telephone numbers from the n-command in the mdd modem file, select any available aslink, and use its d-command. When the modems are connected, use the c-command in the mdd modem file to logon with its XID (x-command).

- For answering a call: Use the a-command in the aslink modem file to query the user for his XID (logon sequence), find the mdd modem file with the matching XID (x-command).

More examples are available in the HAYES.MOD file. See the command descriptions in the Modem Configuration File Syntax section above.

**Example**

An **mdd** modem configuration file for both incoming and outgoing connections:

```
n 790-1234
n 790-1235
x id-abc
c @r20000ID?0^F@x^M@r500c@ r 5
```

An **aslink** modem configuration file for both incoming and outgoing connections:

```
i a@w500a@w500ats0=1&c1&d3&k3&q5^M^J@w100
a ^M^JYour ID?@f5000@w200ccc
d ^M^J@w2000atdt@n^M^J@d30000
r 5
```

**Example**

An **aslink** modem configuration file for incoming connections:

```
i a@w500a@w500ats0=1&c1&d3&k3&q5^M^J@w100
a ^M^JYour ID?@f5000@w200ccc
r 5
```

An **mdd** modem configuration file for incoming connections:

```
x id-abc
```

**Example**

An **mdd** modem configuration file for outgoing connections:

```
n 790-1234
n 790-1235
x id-abc
c @r20000ID?0^F@x^M@r500c@ r 5
```
An aslink modem configuration file for outgoing connections:

```
i a@w500a@w500ats0=0&c1&d3&k3&q5^M^J@w100
```

```
d ^M^J@w2000atdt@n^M^J@d30000
```

```
r 5
```

**CONFIGURATION CONSIDERATIONS**

**MTU** (Maximum Transmission Unit)

The MTU is the maximum size in bytes of a data packet (including all IP and TCP header information.) Information is sent across the Internet in packets, which are reassembled into a whole when they reach their destination. The size of these packets is dependent on the MTU of the machines along the route the packets travel.

The MTU can be specified by each individual computer. When you send out a request for information, the computer you are requesting information from will read your request and hopefully send out packets of the size that you request. If the destination machine has a smaller MTU, the packets will be broken into pieces, or fragmented.

**MSS** (Maximum Segment Size)

MSS is the maximum size in bytes of the data portion of a TCP/IP packet. This value is normally the MTU setting minus 40. MSS is used by SOCKETS to determine the WriteSocket queue size. The SOCKETS queuing overhead is 12 bytes per WriteSocket request so you will need 52 bytes for each request. SOCKETS will queue 2*MSS bytes if it can't send right away because of window constraints by the peer or the Nagle Algorithm. This means that if your MSS is set to the ethernet default of 1460, 2920 bytes will be queued, consuming 3796 bytes with overhead per connection. If you have five connections, you need 18980 bytes just to buffer your outgoing data. Use IPSTAT to determine how much memory you have available. (You need memory for each connection, incoming data, and all other features of the TCP/IP stack).

**Buffers**

Buffers are used to store data packets that are sent and received. The size of each buffer is determined by the MTU setting. Additionally, SOCKETS adds an additional 12 bytes for each WriteSocket request. The maximum amount of memory required for buffers is determined by the following formula

\[
\text{Maximum Buffer Memory Use} = \text{MTU} \times \text{Buffers}
\]

A buffer limit of 30 is excessive if you have 1500 byte buffers (1500 * 30 = 45000). By default, SOCKETS allocates 20000 bytes, which is used for a lot of other processes as well, so you are bound to encounter out-of-memory situations. If you set MTU to 1500, then 5 would be a recommended setting for the number of buffers.
The following examples within this chapter refer to specific files, such as "EXAMPLE1.CFG" and "MODEM.MC2". These example files can be found in the \SOCKETS\CONFIGS directory created during the install process.

- Ethernet connection with SOCKETS acting as a server to a Win9X or NT system. The DL web server can be launched and a standard desktop browser will “surf” to the SOCKETS system.
- Dial-Up serial connection to an ISP.
- Single Dial-In Connection.
- Single Dial-In connection with ASY interface.
- Direct serial connection with SOCKETS as a server.
- Direct serial connection with SOCKETS as a client.
- Dial-up slip connection with SOCKETS as an IP router.
- Multiple dial-in connections.

Example 1: Ethernet Connection – SOCKETS Serving a Web Page
SOCKETP is loaded from the command line as follows:

```
SOCKETP.EXE EXAMPLE1.CFG
```

To demonstrate the web interface, next launch HTTPD.EXE. If the index file is not within the current directory please remember to set the environment variable HTTP_DIR to the location of the index files or nothing will be displayed to the desktop browser.

```
SET HTTP_DIR=C:\DL\SOCKETS\SERVER
```

```
HTTPD.EXE /R
```

The configuration file EXAMPLE1.CFG contains:

<table>
<thead>
<tr>
<th>Ip address 196.6.111/24</th>
<th>Sets an IP address of 196.6.111 with a 24-bit subnet mask to the following interface.</th>
</tr>
</thead>
<tbody>
<tr>
<td>iface pdr if0 dix 1500 5 0x60</td>
<td>Creates a standard Ethernet connection, type pdr (packet driver), interface name if0, type dix, MTU 1500, Buffer Limit 5, Interrupt vector of the packet driver at 0x60.</td>
</tr>
<tr>
<td>route add default if0 196.0.0.1</td>
<td>Set the default route for packets travelling to and from if0 to 196.0.0.1</td>
</tr>
<tr>
<td>domain server 196.0.0.1</td>
<td>Set the domain name server to 196.0.0.1 This server is used to convert names to IP addresses.</td>
</tr>
</tbody>
</table>
Example 2: Serial Connection – SOCKETS Dial-up to ISP
SOCKETM is loaded from the command line as follows:

```
SOCKETM.EXE EXAMPLE2.CFG
```

The configuration file EXAMPLE2.CFG contains:

```
interface asy p0 ppp 576 10 0x3f8 4 19200 modem.mc2
route add default p0
ip ttl 64
tcp mss 1460
tcp window 2920
par p0 ipcp local compress tcp 16 1
par p0 ipcp local address 0.0.0.0
par p0 lcp local acm 0
par p0 lcp local acfc on
par p0 lcp local pfc on
par p0 lcp local magic on
par p0 pap user test example
par p0 ipcp open
par p0 lcp open
```

These commands set up a serial connection to an ISP. The `interface asy` command creates an asynchronous PPP connection on COM1 IRQ4 named p0 with an MTU of 576 and buffer limit of 10. The `route add` command routes all packets through interface p0. The `ip ttl` command sets the time to live for packets at 64 “hops.” The `tcp mss` command sets the Maximum Segment Size to 1460 bytes. The `tcp window` command sets the Windows Size to 2920 bytes. The `par p0 ipcp local compress tcp 16 1` command enables header compression, where 16 are the maximum number of concurrent TCP/IP connections. The `par p0 ipcp local address 0.0.0.0` command sets the server-assigned IP address. The `par p0 lcp local acm 0` command sets the asynchronous control character map to all bits zero. The `par p0 lcp local acfc on` command sets the address control field compression on or off. The `par p0 lcp local pfc on` command sets the protocol field compression on or off. The `par p0 lcp local magic on` command sets the magic number option on or off. The `par p0 pap user test example` command sets the user name to “test” and password to “example.” The `par p0 ipcp open` and `par p0 lcp open` commands open specified layers. The `lcp = link control protocol` command should be used when an immediate dial operation is required. The “par if0 lcp start” command should be used when dial-on-demand is desired. The “par if0 lcp open” command should be used when connect-on-received-call is desired. The “par if0 lcp listen” command should be used. The “par if0 lcp ” command should be used. The start command implies the open command and the open and listen commands may both be used. Dial-on-demand can also be disabled by the “p n” modem configuration file parameter and by an API call. Dial-on-demand can be enabled by an API call.
Example 3: Single Dial-in Connection
Allow only single users to dial in and issue each with an IP address on the fly. Setupasy interfaces, each with its own IP address, COM port, modem and telephone number. In the answer script, a password may be asked (no need for XID: does it in the script), and on success an IP address is sent in dotted decimal form. (Should you want to use different passwords, an mdd/aslink setup as described in Example 2 may be used, but then every user will have his own IP address fixed to the XID.)

PPP Parameters:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lppin</td>
<td>IP control protocol.</td>
</tr>
<tr>
<td>Lcp</td>
<td>Link control protocol.</td>
</tr>
</tbody>
</table>

The commands lcp, pap, ap and ipcp can be specified for incoming parameters. These correspond with the lcp, pap, ap and ipcp commands. The reason for having a different set of PPP parameters is to allow the SOCKETS implementation to act as both a “PPP server” and a “PPP client” without having to re-configure it.

The configuration file EXAMPLE3.CFG contains interface configurations for all modem ports as well as other connections (like Ethernet) that might be used. An asy interface could be:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>i A@w100atdt^M^J@w100</td>
<td>Initialise modem</td>
</tr>
<tr>
<td>d ^M^J@w2000ATDT^M^J@d40000</td>
<td>Dialling string</td>
</tr>
<tr>
<td>n 123-4567</td>
<td>Number to dial</td>
</tr>
<tr>
<td>r 3</td>
<td>Number of retries</td>
</tr>
<tr>
<td>p rx</td>
<td>Enable debugging output to screen</td>
</tr>
</tbody>
</table>

Example 3: Single Dial-in Connection

MODEM.MC2

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>i A@w100atdt^M^J@w100</td>
<td>Initialise modem</td>
</tr>
<tr>
<td>d ^M^J@w2000ATDT^M^J@d40000</td>
<td>Dialling string</td>
</tr>
<tr>
<td>n 123-4567</td>
<td>Number to dial</td>
</tr>
<tr>
<td>r 3</td>
<td>Number of retries</td>
</tr>
<tr>
<td>p rx</td>
<td>Enable debugging output to screen</td>
</tr>
</tbody>
</table>

The modem definition file is MODEM.MC3:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>i a@w100atzt^M^J@ w500ats0=1^M^J@ w100</td>
<td>Initialise modem (consult the manual for your own modem initialisation string)</td>
</tr>
<tr>
<td>x 192 6 3 2</td>
<td>Specify IP address (in the XID variable to make the answer scripts more uniform).</td>
</tr>
</tbody>
</table>
a @w1000^M^JPassword? @
r9000Sockets@ @w200^M^J-
JYour address: @x ^M^J@
r1000GOTIP@

Answer script to prompt remote for logon:
Wait 1 second, send "<CR><LF>" to get cursor on a new line, send "Password? " and wait 9 seconds to receive "Sockets" somewhere in the data stream # (the connection will break if failed), wait 200 milliseconds and send the IP address with @x on a new line. Wait 9 seconds to receive "GOTIP" as confirmation.

p rx Parameters for debugging (show modem data): r=receive and/or x=xmit

Example 4: Single Dial-in Connection with ASY Interface
A user who dials into the system in Example 3 which supplies the IP address for the user, has to use an asy interface to his modem. The user must specify the @a address command in the connect script in the modem file. The @a time command waits for time milliseconds to see the dotted decimal form of an IP address. It assigns this address to the asy interface. The user should have a default route to this asy interface to be able to see the whole world through it, or just a specific route for what he wants to see. The configuration file SOCKET4.CFG should contain at least the asy interface configuration similar to the previous examples.

PPP Parameters:

lpcp = ip control protocol.
Lcp = link control protocol.

The commands lcpin, papin, apin and ipcpin can be specified for incoming parameters. These correspond with the lcp, pap, ap and ipcp commands. The reason for having a different set of PPP parameters is to allow the SOCKETS implementation to act as both a “PPP server” and a “PPP client” without having to re-configure it.

The modem definition file is MODEM.MC4:

Example 5: Direct Serial Connection with SOCKETS as a Server
This section explains how to configure SOCKETS as a server to listen on the designated serial port and wait for a valid connection and use standard dial up networking to connect.
The Client Connection
There are two methods of connection available - direct serial connection or dial in on a standard phone line. The Client here can be any Win95, Win98, or WinNT system.

For a direct serial connection to a Windows environment earlier than Windows 2000 and XP, a modem driver must be installed. Such a driver does not ship standard with SOCKETS but can be found through Internet searches. The client must then create a dial up networking connection with the phone number of CLIEN1, no username or password.

Configuring SOCKETS
Configuration of the SOCKETS software is handled within two files, namely EXAMPLE5.CFG and MODEM.MC5. SOCKETS can be loaded by means of a batch file that contains the following commands:

```
SOCKETM EXAMPLE5.CFG
```

These commands allow SOCKETS to allocate more memory, than allowed by the default values, when initiating a TCP/IP connection over a serial link. SOCKETS processes the EXAMPLE5.CFG file followed by the MODEM.MC5 file.

SOCKETS Configuration File Details
PPP Parameters:

- **Ipcp = ip control protocol.**
- **Lcp = link control protocol.**

The commands **lcpin, papin, apin** and **ipcpin** can be specified for incoming parameters. These correspond with the **lcp, pap, ap** and **ipcp** commands. The reason for having a different set of PPP parameters is to allow the SOCKETS implementation to act as both a “PPP server” and a “PPP client” without having to re-configure it.

EXAMPLE5.CFG contains:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iface asy if0 ppp 576 5 0x1f8 3 19200 modem.mc5</td>
<td>Creates an asynchronous connection on com 2, IRQ 3 with a Baud rate of 19200, MTU of 576, and a Buffer Limit of 5. Access the modem.mc5 file for specific initialization strings.</td>
</tr>
<tr>
<td>par if0 ipcpin local compress tcp 16 1</td>
<td>Enables header compression, where 16 are the maximum number of concurrent TCP/IP connections. 1 turns on compression, 0 turns off compression.</td>
</tr>
<tr>
<td>par if0 ipcpin local address 196.10.229.2</td>
<td></td>
</tr>
<tr>
<td>par if0 ipcpin local address 196.10.229.4</td>
<td></td>
</tr>
<tr>
<td>par if0 ipcp local compress tcp 16 1</td>
<td>Enables header compression, where 16 are the maximum number of concurrent TCP/IP connections. 1 turns on compression, 0 turns off compression.</td>
</tr>
<tr>
<td>par if0 ipcp local address 196.10.229.2</td>
<td></td>
</tr>
</tbody>
</table>

User SocketsUser SocketsPassword 196.10.229.18

For user connecting with name “SocketsUser” and Password “SocketsPassword” assign the ip address of 196.10.229.18.
par if0 lcp local accm 0  |  Asynch control character map, set all bits to zero.
par if0 lcp local acfc on  |  Address control field compression on or off.
par if0 lcp local pfc on  |  Protocol field compression on or off.
par if0 lcp local magic on  |  Magic number option on or off. The magic number is used to detect loop back links by creating a number, sending a configure request, then comparing the number received. If it is the same, and then there is a possible loop back, repeat test.
par if0 lcp local authen pap  |  Set local authentication to pap
par if0 lcp local accm 0  |  Asynch control character map set all bits to zero.
par if0 lcp local acfc on  |  Address control field compression on or off.
par if0 lcp local pfc on  |  Protocol field compression on or off.
par if0 lcp local magic on  |  Magic number option on or off. The magic number is used to detect loop back links by creating a number, sending a configure request, then comparing the number received. If it is the same, then there is a possible loop back, repeat test.
par if0 ipcp open  |  Open specified layer. ipcp = ip control protocol.
par if0 lcp listen  |  Lcp = link control protocol.
par if0 lcpin listen  |  When an immediate dial operation is required the “par if0 lcp start” command should be used. When dial-on-demand is desired the “par if0 lcp open” command should be used and when connect-on-received-call is desired, the “par if0 lcp listen” command should be used. The start command implies the open command and the open and listen commands may both be used. Dial-on-demand can also be disabled by the “p n” modem configuration file parameter and by an API call. Dial-on-demand can be enabled by an API call.
route add default if0  |  Route all packets through interface if0
ip address route  |  Prints out the systems ip address and route. Useful for debugging only.

**MODEM.MC5**

Initialize Modem. For a direct serial connection no initialization string is necessary.

a @r20000CLIENT1@CLIENT-SERVER  |  When Sockets receives the string “CLIENT” send the response “ClientServer”
r 3  |  Number of Retries
p nx  |  Debugging information parameters: r=show modem receive, x=show modem xmit
d=always dial, n=no dial-on-demand
Example 6: Direct Serial Connection with SOCKETS as a Client

Setting the Server
The NT server must have Remote Access Services enabled, commonly referred to as RAS, as well as the “Direct Serial cable between two PCs” modem driver installed on the proper COM port. The following example also connects using DHCP. If this feature is to be used then DHCP services must also be setup on the NT server. Other options would include the use of static IP addressing or BOOTP. Please review the cabling requirements from the WinNT documentation or help files. Standard cables often will not work with direct connections due to the cabling demands of WinNT.

Configuring the Software
Configuration of the SOCKETS software is handled within two files, namely EXAMPLE6.CFG and MODEM.MC4. SOCKETS can be loaded by means of a batch file that contains the following commands:

```bash
SOCKETM EXAMPLE6.CFG
```

These commands allow SOCKETS to allocate more memory than allowed by the default values when initiating a TCP/IP connection over a serial link. SOCKETS processes the EXAMPLE6.CFG file followed by the MODEM.MC6 file.

Configuration File Details
PPP Parameters:

- **lpcp = ip control protocol.**
- **Lcp = link control protocol.**

The commands `lcpin, papin, apin` and `ipcpin` can be specified for incoming parameters. These correspond with the `lcp, pap, ap` and `ipcp` commands. The reason for having a different set of PPP parameters is to allow the SOCKETS implementation to act as both a “PPP server” and a “PPP client” without having to re-configure it.

**EXAMPLE6.CFG:**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>socket</td>
<td>Creates an asynchronous connection on com 2, IRQ 3 with a baud rate of 19200 and access the modem.cfg file for specific initialization strings.</td>
</tr>
<tr>
<td>par if0 lpcp local compress tcp 16</td>
<td>Enables header compression, where 16 are the maximum number of concurrent TCP/IP connections. 1 turns on compression, 0 turns off compression.</td>
</tr>
<tr>
<td>par if0 lpcp local address 0.0.0.0</td>
<td>The ip address 0.0.0.0 indicates that the server must assign the client a valid ip address.</td>
</tr>
<tr>
<td>par if0 lcp local accm 0</td>
<td>Asynch control character map, set all bits to zero.</td>
</tr>
<tr>
<td>par if0 lcp local acfc on</td>
<td>Address control field compression on or off.</td>
</tr>
<tr>
<td>par if0 lcp local pfc on</td>
<td>Protocol field compression on or off.</td>
</tr>
</tbody>
</table>
**Example 7: SOCKETS Machine Using Call Back Verification**

**Scenario:**

If the NT machine logs into the SOCKETS machine with SocketsUser1 / SocketsPassword1, the NT machine is assigned IP address 196.10.229.18 and the SOCKETS machine is 196.10.229.2.

If the NT machine logs into the SOCKETS machine with “SocketsUser / SocketsPassword,” the SOCKETS machine will break the connection and dial back to 08036501 logging in as “NtUser” with password “NtPassword.” The NT machine assigns the IP addresses of both machines.

The SOCKETS machine never initiates a modem call unless it has been called first. Once SOCKETM is running, the IOCTL program can be used to initiate a dial operation or to enable dial on demand.

**SOCKETS Configuration File Details**

**PPP Parameters:**

\[ \text{lp} \text{cpp} = \text{ip control protocol.} \]
\[ \text{lcp} = \text{link control protocol.} \]

The commands \textit{lcpin}, \textit{papin}, \textit{apin} and \textit{ipcppin} can be specified for incoming parameters. These correspond with the \textit{lcp}, \textit{pap}, \textit{ap} and \textit{ipcpp} commands. The reason for having a different set of PPP parameters is to allow the SOCKETS implementation to act as both a “PPP server” and a “PPP client” without having to re-configure it.
EXAMPLE7.CFG:

iface asy p ppp 1500 30 0x3f8 4 9600 modem.mc7
Creates an asynchronous connection designated interface "p" type ppp on COM1 IRQ4 with an MTU of 1500, Buffer limit of 30, baud rate 9600 and modem information contained within the modem.mc7 file.

route add default p
Routes all packets through interface "p".

user SocketsUser Sockets-Password 196.10.229.18 cbv
If user is SocketsUser with password SocketsPassword assign IP address of 196.10.229.18 then disconnect session and call back.

user SocketsUser1 Sockets-Password1 196.10.229.18
If user is SocketsUser1 with password SocketsPassword1 assign IP address of 196.10.229.18 and continue session.

par p ipcpin local compress tcp 16 1
Enables header compression, where 16 are the maximum number of concurrent TCP/IP connections. 1 turns on compression, 0 turns off compression.

par p ipcpin local address 196.10.229.2
Assign local IP address 196.10.229.2.

par p ipcp local compress tcp 16 1
Enables header compression, where 16 are the maximum number of concurrent TCP/IP connections. 1 turns on compression, 0 turns off compression.

par p ipcp local address 0.0.0.0
The ip address 0.0.0.0 indicates that the server must assign the client a valid ip address.

par p lcpin local accm 0
Sets the Asynch Control Character Map to 0.

par p lcpin local acfc on
Sets local Address and Control Field Compression on.

par p lcpin local pfc on
Sets local Protocol Field Compression on.

par p lcpin local magic on
Magic number option on or off. The magic number is used to detect loop back links by creating a number, sending a configure request, then comparing the number received. If it is the same, then there is a possible loop back: repeat test.

par p lcpin local authen pap
Sets the local Authentication protocol to pap.

par p lcp local accm 0
Sets the local Asynch Control Character Map to 0.

par p lcp local acfc on
Sets the local Address and Control Field Compression to on.

par p lcp local pfc on
Sets the local Protocol Field Compression to on.

par p lcp local magic on
Magic number option on or off. The magic number is used to detect loop back links by creating a number, sending a configure request, then comparing the number received. If it is the same, then there is a possible loop back, repeat test.

par p pap user NtUser NtPassword
Allow login for user NtUser with password NtPassword. PAP corresponds with the previous set authentication protocol.

par p ipcp open
Set incoming dial-in method to open.

par p ipcpin open
Set incoming dial-in method to open.

par p lcp listen
Set dial-out method to listen.

par p lcpin listen
Set dial-out method to listen.
Example 8: SOCKETS Machine with CBV and Logging-in
PPP Parameters:

\[
\text{Ipcp} = \text{ip control protocol.}
\]

\[
\text{Lcp} = \text{link control protocol.}
\]

The commands \text{lcpin}, \text{papin}, \text{apin} and \text{ipcpin} can be specified for incoming parameters. These correspond with the \text{lcp}, \text{pap}, \text{ap} and \text{ipcp} commands. The reason for having a different set of PPP parameters is to allow the SOCKETS implementation to act as both a “PPP server” and a “PPP client” without having to re-configure it.

**EXAMPLE8.CFG**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\text{iface asy p ppp 1500 30 0x18 4 9600 modem.mc8}</td>
<td>Creates an asynchronous connection designated interface “p” type ppp on COM1 IR04 with an MTU of 1500, Buffer limit of 30, baud rate 9600 and modem information contained within the modem.mc8 file.</td>
</tr>
<tr>
<td>\text{route add default p}</td>
<td>Routes all packets through interface “p”.</td>
</tr>
<tr>
<td>\text{user NtUser NtPassword 196.10.229.208}</td>
<td>If user is NtUser with password NtPassword assign IP address of 196.10.229.208 then disconnect session.</td>
</tr>
<tr>
<td>\text{par p ipcp local compress tcp 16 1}</td>
<td>Enables header compression, where 16 are the maximum number of concurrent TCP/IP connections. 1 turns on compression, 0 turns off compression.</td>
</tr>
<tr>
<td>\text{par p ipcp local address 0.0.0.0}</td>
<td>The ip address 0.0.0.0 indicates that the server must assign the client a valid IP address.</td>
</tr>
<tr>
<td>\text{par p lcp local accm 0}</td>
<td>Sets the local Asynch Control Character Map to 0.</td>
</tr>
<tr>
<td>\text{par p lcp local acfc on}</td>
<td>Sets the local Address and Control Field Compression to on.</td>
</tr>
</tbody>
</table>
Initialize the modem; please refer to the manufacturer documentation for the specific modem initialization string.

Number to dial

Dial string.

Debugging information. Parameters: r=show modem receive, x=show modem xmit d=always dial, n=no dial-on-demand
When the “NT look-alike” side starts up, it dials 08034131. The “SOCKETS” side answers and after the successful login, drops the connection and dials 08036501. During PPP negotiation, the “NT look-alike” side is assigned IP address 196.10.229.202 and the “SOCKETS” side, IP address 196.10.229.108.

**Example 9: Dial-up SLIP Connection with SOCKETS as an IP Router**
You want to connect your LAN to another network via a modem using a dial-up SLIP link. The SOCKETS PC will act as an IP router (or gateway) to the other network. This example is symmetrical: You can use the same setup on both sides to enable any side to initiate the call. (The IP addresses will have to be unique.)

Use an *asy* interface with CSLIP to get better throughput. Your network is 192.6.1.0 and your address 192.6.1.111. The *asy* interface will also be 192.6.1.111 linking the 192.6.2.0 network (or the rest of the world) on the other side. The modems establish a connection with no logon required.

**PPP Parameters:**

- *lpccp* = ip control protocol.
- *Lcp* = link control protocol.

The commands *lpccp, papin, apin* and *lpccpin* can be specified for incoming parameters. These correspond with the *lcp, pap, ap* and *lpccp* commands. The reason for having a different set of PPP parameters is to allow the SOCKETS implementation to act as both a “PPP server” and a “PPP client” without having to re-configure it.

The configuration file EXAMPLE9.CFG contains:

- **Ip address 196.6.1.111/24**
  Sets an IP address of 196.6.1.111 with a 24-bit subnet mask to the following interface.

- **Iface pdr if0 dix 1500 10 0x60**
  Creates a standard Ethernet connection, type pdr (packet driver), interface name if0, type dix, MTU 1500, Buffer Limit 10, Interrupt vector of the packet driver at 0x60.

- **IPAddress=192.6.1.111/0**
  Sets an IP address of 192.6.1.111 with a 0-bit subnet mask to the following interface.

- **Iface asy si0 cslip 576 10 3f8 4 19200 asy-io.mod**
  Creates a standard modem connection on COM1. The connection is asynchronous, interface name si0, MTU 576, Buffer Limit 4, baud rate 19200, and reference the file asy-io.mod for specific modem information.

- **; Make this the default route:**
  For interface si0 set the serial baud rate to 19200, output flow to CTS, input flow to RTS.

- **CTS = Clear to Send modem signal. RTS = Request To Send modem signal.**
  *Use only for asy type interfaces*

- **par si0 19200 CTS RTS**
  Enable Routing Information Protocol on interface si0.

- **rip advertise 30**
  Set RIP advertise time to 30. Time is the number of elapsed seconds before the advertisement is repeated.

- **rip use 200**
  Set RIP use time to 200. Time is the period during which routes are added or amended as a result of RIP, and are valid for time seconds.
The start up file SOCKETS.STU will display the current status with:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip address</td>
<td>Display the IP address</td>
</tr>
<tr>
<td>route</td>
<td>Display the routes</td>
</tr>
<tr>
<td>ifstat</td>
<td>Display the interfaces</td>
</tr>
</tbody>
</table>

The modem definition file is ASY-IO.MOD:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>l</td>
<td>Initialise modem (Caution: consult the manual for your own modem)</td>
</tr>
<tr>
<td>d</td>
<td>Dial command: send &quot;&lt;CR&gt;&lt;LF&gt;&quot;, wait 2 seconds, send &quot;atdt&lt;number&gt;&lt;CR&gt;&lt;LF&gt;&quot;, wait 40 seconds for DCD</td>
</tr>
<tr>
<td>n</td>
<td>The number to dial</td>
</tr>
<tr>
<td>r</td>
<td>Number of retries</td>
</tr>
<tr>
<td>p r</td>
<td>Parameters for debugging (show modem data): r=receive and/or x=xmit</td>
</tr>
</tbody>
</table>

Password protected logon facilities can be added with answer and connect scripts. (See next examples or ASY-IOL MOD on disk)

---

**Example 10: Multiple Dial-in Connections**

You have a number of modems for dial-in of various users or other networks each with a fixed IP address. When somebody dials into your system, SOCKETS will need to know his IP address to be able to set up a return route to him. The network size for each remote host must also be known. (You have to understand sub-networks for this example.)

The way to do this is to set up a mdd interface for each IP address (using an IP address on that network). Each mdd interface has a modem definition file containing a unique exchange identifier (XID). Each modem has an aslink interface with a modem file containing an answer script. When the remote dials in, he has to specify his XID in his connect script (like a login). The aslink then links to the mdd with the same XID and your remote user is connected using the routes set for his mdd interface.

**PPP Parameters:**

- ipcp = ip control protocol.
- Lcp = link control protocol.

The commands **lcpin**, **papin**, **apin** and **ipcpin** can be specified for incoming parameters. These correspond with the **lcp**, **pap**, **ap** and **ipcp** commands. The reason for having a different set of PPP parameters is to allow the SOCKETS implementation to act as both a “PPP server” and a “PPP client” without having to re-configure it.
The configuration file EXAMPL10.CFG should contain:

<table>
<thead>
<tr>
<th>IPAddress=197.55.2.9</th>
<th>Set the IP address of the following interface to 197.55.2.9</th>
</tr>
</thead>
<tbody>
<tr>
<td>iface pdr if0 dix 1500 10 0x60</td>
<td>Creates a standard Ethernet connection, type pdr (packet driver), interface name if0, type dix, MTU 1500, Buffer Limit 10, Interrupt vector of the packet driver at 0x60.</td>
</tr>
<tr>
<td>; A Multi Destination Driver</td>
<td></td>
</tr>
<tr>
<td>; connecting to the smallest subnet allowing 2 hosts</td>
<td></td>
</tr>
<tr>
<td>IPAddress=193.101.51.1/30</td>
<td>Set the IP address of the following interface to 193.101.51.1 with a 30-bit subnet mask.</td>
</tr>
<tr>
<td>iface mdd mdd0 slip 576 10 mdd10.mod</td>
<td>Create a multiple destination interface, type mdd, interface name mdd0, class slip, MTU 576, Buffer Limit 10, access the file mdd10.mod for specific interface commands.</td>
</tr>
<tr>
<td>iface asy as10 cslip 576 10 0x3F8 4 19200 aslink.mod</td>
<td>Create an aslink connection on COM1.</td>
</tr>
</tbody>
</table>

All the modems (aslink interfaces) may use the ASLINK.MOD file:

| i a@w100at"^M"J@w500ats0=1^K M^J@w100 | Initialise modem (Warning: consult the manual for your specific modem initialisation string) |
| a @w500"^M"XID?@f5000@w20 0ccc | Answer script to prompt for XID |
| p r | Parameters for debugging (show modem data): r=receive and/or x=xmit |

All the mdd interfaces will use a specific MDDn.MOD file with its own unique XID. The XID links to the IP address in the configuration of the mdd interface. MDD0.MOD will be:

| x id0 | Set the XID variable |
| p r | Parameters for debugging (show modem data): r=receive and/or x=xmit |

The users dialling into the system above may use asy interfaces with a modem file containing the following commands (see file ASY-OX.MOD):

<p>| i a@w100at&quot;^M&quot;J@w500ats0=1^K M^J@w100 | Initialise modem (Warning: consult the manual for your own modem) |
| d ^M^J@d2000atdt@r^n&quot;^M&quot;J@d400 00 | Dial command: send &quot;&lt;CR&gt;&lt;LF&gt;&quot; wait 2 seconds, send &quot;atdt&lt;number&gt;&lt;CR&gt;&lt;LF&gt;&quot; wait 40 seconds for DCD. |</p>
<table>
<thead>
<tr>
<th>n 790-1234</th>
<th>The number to dial</th>
</tr>
</thead>
<tbody>
<tr>
<td>r 5</td>
<td>Number of retries</td>
</tr>
<tr>
<td>x id0</td>
<td>Set the XID variable</td>
</tr>
<tr>
<td>c @1000XID@^F@x^M@r100c</td>
<td>Connect script to login at the remote: wait 1 second after DCD to receive XID prompt, send the “^F XID ^M sequence, receive a “c” as confirmation.</td>
</tr>
<tr>
<td>p rx</td>
<td>Parameters for debugging (show modem data): r=receive and/or x=xmit</td>
</tr>
</tbody>
</table>
SOCKETS UTILITIES
SOCKETS utilities make use of command-line parameters and/or configuration files. Please be careful to note the name and location of the configuration file used by the application you are working with. All SOCKETS applications require that the kernel be loaded before the application is run in order to function properly.

ARPSTAT

ARPSTAT is a demonstration program to illustrate the use of the ArpApi function. It displays the status of the ARP cache and can also be used to remove ARP entries. It is hard coded to only remove Ethernet entries.

Syntax
ARPSTAT [ip address1] [ip address2] [...] [ip addressN]

Remarks
ip addressN
The IP address of an ARP entry to be removed. If this is 0.0.0.0 then all entries will be removed.

Examples
ARPSTAT 198.162.1.1
Removes the ARP entry associated with the IP address of 198.162.1.1 and displays the status of the remaining entries.

CAPIDIAG

This is a diagnostic program for the SOCKETS TCP/IP Basic interface. It tests the API as well as the stack and even the network when external servers are specified. It also measures the performance of the network for TCP and UDP traffic. The external server(s) need to run both TCP and UDP echo servers on TCP port 7 and UDP port 7. A Sockets machine can be used as a server by running SUPERD. The source code for CAPIDIAG is supplied as an advanced example of using the Basic API (CAPI).

Syntax
CAPIDIAG [/b=bulk_test_seconds] [/d] [/f=fail_action] [/n] [/t=timeout_seconds] [/v=verbosity level] [server1] [server2] [...] [serverN]
CHAPTER 14 | SOCKETS UTILITY DESCRIPTIONS

DHCPSTAT

DHCPSTAT displays the DHCP information for the machine.

Syntax

```
DHCPSTAT [r | v]
```

Options

The `r` option is for forcing a renewal of the DHCP lease.

The `v` option displays the SOCKETS version information.

Examples

```
DHCPSTAT
```

This will display all the DHCP information, such as IP address and lease time.

Options

The `/b` option allows the testing time for each traffic test to be set.

The `/d` option specifies debugging of the server.

The `/f` option specifies the fail action. Valid values for this option are:

- 0 print error and continue
- 1 print error and wait for Esc.
- 2 print error and exit

The `/t` option sets the number of timeout seconds.

The `/n` option specifies that the local loop back traffic tests should be skipped.

The `/v` option sets the level of verbosity. Valid values for this option are:

- 0 only print errors
- 1 print names of all tests
- 2 print additional information on errors
- 4 print summary of events
- 8 print kernel configuration
- 16 print buffer for bulk read back failure
- 32 print NET_ADDR
- 64 print Read values

Examples

```
CAPIDIAG /f=2 server1 server2
```

This will run the tests to a local built-in echo server and to server1 and server2. It will exit the program on a failure.
FTP

FTP is a file transmitting and retrieving client that runs in interactive or batch mode.

Syntax

FTP server [options]

Options

/n
/v
/p=Port
/f=ScriptFile [ScriptParameters]

Remarks

Server
The name or ip address of a server to connect to.

/n
Suppress progress indicator.

/v
Verbose output for troubleshooting.

/p=Port
Connect to a server port other than the standard FTP port number of 21.

/f=ScriptFile
A file containing commands for the client to send to the server upon connection. Simple parameter substitution is performed, with the first element of ScriptParameters accessible as “%1,” etc.

ScriptParameters
Parameters to pass into the ScriptFile.

Return Codes

0 Success
1 Parameter error
2 SOCKETS not loaded
3 User aborted
4 Transfer aborted
5 Error writing local file
6 Error reading local file
Other Server returned error response code; to find that error code, add 390 to the response code returned by FTP. The result will always be greater than or equal to 400 in this case.
Examples

FTP /n FTP.cdrom.com /f=getfile.scr /f/simtelnet/msdos DIRS.TXT
(The file GETFILE.SCR):
user anonymous
pass root@
cd %1
binary
get %2
quit

FTP Commands

The commands entered at the FTP client can be interpreted and translated to standard FTP
commands to be sent to the server. The FTP server might recognise more, or less, commands
than the standard list of commands as specified in RFC 959. The site command is always server
dependent. Some of the standard commands are implemented differently in various servers.

Useful things to note are:

1. The put and get commands allow multiple file transfers by usage of wild card characters.
   When getting files with paths or long names, no translation of foreign file names are
done. Specify a valid DOS local_file name.

2. A short directory list (NLST) is obtained by ls and the long list with dir.

3. Some of the commands can be abbreviated.

4. Some commands are aliases added for user comfort like bye, exit and quit; get and
   mget; and put and mput.

5. The optional [local_file] parameter will, when specified, cause the output of that
   command to be logged to a file. By specifying the file as PRN you can get immediate
   printouts.

6. On some servers you might specify the optional [remote_file] parameter as PRN or the
   printer output device to do remote printing. (See also the site nopath command for the
   SOCKETS FTP server.)

7. The F3 key and spacebar can be used to recall the last command word by word.

Below is a list of commands recognised by the SOCKETS FTP client (some FTP servers might
not offer all the facilities):

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>abort</td>
<td>Cancel an incomplete transfer</td>
</tr>
<tr>
<td>append</td>
<td>&quot;Put&quot; a file at the server but append it if the file exists</td>
</tr>
<tr>
<td>ascii</td>
<td>Synonym for type a</td>
</tr>
<tr>
<td>binary</td>
<td>Synonym for type i</td>
</tr>
<tr>
<td>bye</td>
<td>Synonym for quit</td>
</tr>
<tr>
<td>cd directory</td>
<td>Synonym for cwd</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------------------------------------------------------</td>
</tr>
<tr>
<td><code>cwd directory</code></td>
<td>Change server directory</td>
</tr>
<tr>
<td><code>dele file</code></td>
<td>Delete a server file</td>
</tr>
<tr>
<td><code>dir [file l directory [local_file]]</code></td>
<td>Synonym for <code>list</code></td>
</tr>
<tr>
<td><code>exit</code></td>
<td>Synonym for <code>quit</code></td>
</tr>
<tr>
<td><code>get remote_file(s) [local_file]</code></td>
<td>Transfer a file from the server in the current mode (type)</td>
</tr>
<tr>
<td><code>image</code></td>
<td>Synonym for <code>type i</code></td>
</tr>
<tr>
<td><code>ls [file l directory [local_file]]</code></td>
<td>Synonym for <code>nlst</code></td>
</tr>
<tr>
<td><code>lcd directory</code></td>
<td>Perform a local change directory</td>
</tr>
<tr>
<td><code>ldir [file l directory]</code></td>
<td>Give a local directory listing</td>
</tr>
<tr>
<td><code>list [file l directory [local_file]]</code></td>
<td>Give a long directory listing</td>
</tr>
<tr>
<td><code>mget remote_file(s) [local_file]</code></td>
<td>Synonym for <code>get</code></td>
</tr>
<tr>
<td><code>mkdir remote_directory</code></td>
<td>Create a server directory</td>
</tr>
<tr>
<td><code>mput local_file(s) [remote_file]</code></td>
<td>Synonym for <code>put</code></td>
</tr>
<tr>
<td><code>nlst [file l directory [local_file]]</code></td>
<td>Give a short names-only directory listing</td>
</tr>
<tr>
<td><code>pass [password]</code></td>
<td>Password for username</td>
</tr>
<tr>
<td>`pasv [on</td>
<td>off]`</td>
</tr>
<tr>
<td><code>put local_file(s) [remote_file]</code></td>
<td>Transfer a file to the server in the current mode (type)</td>
</tr>
<tr>
<td><code>Pwd</code></td>
<td>Print working directory at server</td>
</tr>
<tr>
<td><code>quit</code></td>
<td>Terminate FTP session</td>
</tr>
<tr>
<td><code>quote remote_command [args ...]</code></td>
<td>Send a command to the server without any interpretation</td>
</tr>
<tr>
<td><code>rmdir remote_directory</code></td>
<td>Remove (delete) a server directory</td>
</tr>
<tr>
<td><code>rnfr existing_filename</code></td>
<td>Rename a file, command 1 of 2</td>
</tr>
<tr>
<td><code>rnto new_filename</code></td>
<td>Rename a file, command 2 of 2</td>
</tr>
<tr>
<td><code>site sub-command</code></td>
<td>Send server specific commands</td>
</tr>
<tr>
<td><code>size file</code></td>
<td>Report the file size in bytes as a 213 message</td>
</tr>
<tr>
<td><code>shell</code></td>
<td>Shell to DOS for IFTP.EXE</td>
</tr>
<tr>
<td><code>stat</code></td>
<td>Report the status of a transfer or active connections</td>
</tr>
<tr>
<td><code>System</code></td>
<td>Return operating system information from the server</td>
</tr>
<tr>
<td><code>type [i I a]</code></td>
<td>Report or select the file transfer mode: image (binary) or ASCII</td>
</tr>
<tr>
<td><code>user [username]</code></td>
<td>Username to logon</td>
</tr>
<tr>
<td><code>verbose [ on I off ]</code></td>
<td>Verbose mode reports more of the FTP negotiations</td>
</tr>
</tbody>
</table>
FTPAPI

FTPAPI is both a server and client for the FTP protocol that loads as a TSR. It should be called directly from your application. The documentation for this interface is found in FTPAPI.H. A complete “server and multiple client” sample program is provided as FTPTEST.C and FTPTEST.EXE.

Syntax

\texttt{FTPAPI [options] [Port]}

Options

\texttt{/m=MemorySize}

\texttt{/c}

\texttt{/s}

\texttt{/u}

\texttt{/v=disable passive mode}

Remarks

\texttt{/m=MemorySize}

The number of bytes of memory available to the server. This value defaults to 32768.

\texttt{/c}

Running \texttt{FTPAPI} again with this option disables the TSR’s listen socket.

\texttt{/s}

Running \texttt{FTPAPI} again with this option displays the TSR’s status.

\texttt{/u}

Running \texttt{FTPAPI} again with this option unloads the TSR.

\texttt{[Port]}

\texttt{FTPAPI} listens on the listed port. This parameter defaults to the standard FTP port number of 21.e.

Configuration File

\texttt{FTPAPI} uses the standard SOCKETS password file, SOCKET.UPW, for validating logins. See \texttt{FTPD} for more information.

Examples

\texttt{FTPAPI /m=40000 42}

GETMAIL

GETMAIL retrieves all of the messages from a POP3 (Post Office Protocol version 3) Internet mail server. Each message is stored as an individual file on the local machine. GETMAIL also creates a log file to indicate successful downloads or errors.
Syntax

GETMAIL server user password

Options

Server
The IP address or DNS name of the Internet mail server from which to download messages. The messages that are downloaded are named by sequential file number and are placed in the current working directory.

User
The username for server identification purposes.

Password
The secret for account authentication on the server.

Logging Format, Code String Timestamp

Weekday Month Day Time Year

Timestamp
Weekday Month Day Time Year.

Code
Three digit integer. 000 means perfect success, 100-199 mean usage error and 200-299 means TCP/IP error from server.

String
Human-readable explanation of the error code.

Examples

GETMAIL 10.0.0.1
GETMAIL 10.0.0.1 guest secret

HTTPGET, WHTTPGET AND W9HTTPGET

HTTPGET is a DOS HTTP (web) client that can retrieve the contents of a URL to a local file. WHTTPGET is a Windows NT, 2000, XP console mode program with the same functionality as HTTPGET. W9HTTPGET is the Windows 9x counterpart and is not enabled for IPv6 operation.

Syntax


Options

-v
Display extra output for troubleshooting.
HTTPPUT is a DOS HTTP (web) client that can upload the contents a local file to a URL. WHTTPPUT is a Windows NT, 2000, XP console mode program with the same functionality as HTTPPUT. W9HTTPPUT is the Windows 9x counterpart and is not enabled for IPv6 operation. The target server must support the PUT method like HTTPD does.

Syntax


Options

- **-v**
  Display extra output for troubleshooting.

Examples

HTTPGET http://www.datalight.com/images/logohead.gif

WHTTPGET -v http://www.datalight.com/images/logohead.gif logo.gif

**Remarks**

**Port**
Use to specify a remote port other than the default of 80 to connect to.

**Server**
Use to specify a server name if the URL doesn’t contain one.

**URL**
The URL to retrieve from. The notation http:// is optional and may be left out.

**Username:Password**
The username and password to use if the request must be authenticated. Rather than keeping the filename from the URL, the contents may be saved to localfile.

**Examples**

HTTPGET http://www.datalight.com/images/logohead.gif

WHTTPGET -v http://www.datalight.com/images/logohead.gif logo.gif
Username:Password
The username and password to use if the request must be authenticated.

localfile
Rather than keeping the filename from the URL, the contents may be saved to localfile.

Examples
HTTPPUT -a=peter:baskaron www.datalight.com/images/logohead.gif
WHTTPGET -v http://www.datalight.com/images/logohead.gif logo.gif

IFSTAT

IFSTAT displays the status of the Interface, Modem, Serial, PPP, and it displays the version information for SOCKETS.

Syntax
IFSTAT [/r=c|msec] [f][i][m][p][s][t][v]

Options
/r=c – repeat the display on significant changes until a key is pressed.
/r=msec – repeat the display every msec milliseconds until a key is pressed.
 f - show full MIB II statistics. Sockets must be built to gather MIB II statistics.
 i - show the Interface status.
 m - show the Modem status.
 p - show the PPP status.
 s - show the Serial status.
 t - show the current time.
 v - show the version information.

Examples
IFSTAT v
This will display the SOCKETS version information.

Notes
The timestamp and repeat options are intended to provide better diagnostics for modem and PPP connections.

The changes that will cause a new printout depend on the options that are selected as follows:
IOCTL

IOCTL is a diagnostic utility for the SOCKETS IOCTL functions using the TCP/IP Basic API.

Syntax

```
IOCTL interface name
```

Remarks

```
interface name   The name of the interface to be tested.
```

Examples

```
IOCTL if0
Tests the IOCTL functions through the interface "if0" based on the commands given during the diagnostics.
```
IOCTLH

IOCTLH is a diagnostic utility for the SOCKETS IOCTL functions using the TCP/IP Basic API. This version supports the hot-swappable functionality.

Syntax

IOCTLH interface name

Remarks

interface name The name of the interface to be tested.

Examples

IOCTLH if0
Tests the IOCTL functions through the interface “if0” based on the commands given during the diagnostics.

IPSTAT

The IPSTAT utility returns statistics on IP and memory. Use IPSTAT to check for error conditions and memory problems.

Syntax

ipstat [/2] [/a] [/c] [/m] [/s] [message]

/2 show MIB-2 counters
/a show memory allocation
/c clear memory allocation list
/m show memory statistics
/s show number of sockets
message show message

Examples

IPSTAT %TIME%
The following will be displayed (The values may differ):
Datalight Sockets IPSTAT v6.22 (Revision 4.20.1560)
Copyright (c) 1989-2005 Datalight, Inc.
Portions copyright (c) GPvNO 2004-2005.
LPR

LPR is a printer client for UNIX-style printer servers. An LPD server can be configured for SOCKETS.

Syntax

LPR /s=Server /p=Printer /u=Agent Filename

Options

/r
/q
/l=Port
/h=LocalHostName
/c=JobClass
/j=JobName
/n
/t

Sockets build 1565
17:30:04.49
Kernel supports IPv4
IPv4 address 192.168.0.53 IPv4 Subnet mask: 255.255.255.0 IPv4 Routing table:

<table>
<thead>
<tr>
<th>Destination</th>
<th>Class</th>
<th>Interface</th>
<th>Gateway</th>
<th>Metric</th>
<th>RIP time/flags</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td></td>
<td>if0</td>
<td>192.168.0.200</td>
<td>1</td>
<td>Static</td>
</tr>
<tr>
<td>Default</td>
<td></td>
<td>if0</td>
<td>192.168.0.200</td>
<td>1</td>
<td>Static</td>
</tr>
<tr>
<td>192.168.0.0</td>
<td>Net C</td>
<td>if0</td>
<td>1</td>
<td>Static</td>
<td></td>
</tr>
<tr>
<td>192.168.0.0</td>
<td>Net C</td>
<td>if0</td>
<td>1</td>
<td>Static</td>
<td></td>
</tr>
</tbody>
</table>

IP stats at 14DF:02C4:

Total packets 36
Smaller than minimum size 0
IP header length too small 0
Wrong IP version 0
IP header checksum errors 0
Unsupported protocol 0
Remarks

/q
Query Mode. Can be followed by agent names or job numbers to filter output.

/r
Remove Mode. May be followed by job numbers to specify jobs to remove.

/s=Server
Hostname of the print server.

/p=Printer
Name of the printer device on the server to be used for output.

/u=Agentr
User name on the server. Used for identification.

Filename
Local file name to spool to the server.

/l=Portr
Connect to the specified port on the server rather than the standard port number of 515.

/h=LocalHostName
Name of the local host for job identification purposes.

/c=JobClass
Name of the job class for job identification and scheduling purposes.

/j=JobName
Name of the job, for identification purposes; defaults to the local file name if not specified.

/n
Run without user interaction.

/t
Text filter. Strips all unprintable characters before printing.

Examples
LPR /n /s=10.0.0.1 /p=prn0 /u=Tester output.dat
LPR /n /s=10.0.0.1 /p=prn0 /u=Tester /q
LPR /n /s=10.0.0.1 /p=prn0 /u=Tester /r

MAKEMAIL

MAKEMAIL packages the body text and any attachments for delivery using the SENDMAIL application.

Syntax
MAKEMAIL –nScriptFile –tToAddress –fFromAddress –sSubject –bBodyTextFile - oOutputFile- aAttachment
Options

ScriptFile
The name of a file that contains all of the information needed to create the mail file. Each new
command line argument in the script file must be on a separate line. Multiple ToAddress and
Attachment lines may be specified.

ToAddress
The e-mail address of the recipient(s) of this mail. Additional recipients are specified by
repeated use of the –t parameter. If the ToAddress is a name that can be resolved by either
the DNS server or host file then the @servername is not necessary.

FromAddress
Used to identify the sender of the message.

Subject
The subject line of the e-mail message.

BodyTextFile
The local file containing the body text of the e-mail message to deliver.

OutputFile
The local file name in which to store the prepared file for delivery by SENDMAIL. This file is
overwritten if it already exists!

Attachment
The name of a local file to be binary attached to this e-mail message. Multiple attachments are
created by repeated use of the –a parameter. Files are attached as MIME parts, encoded with
the application/x-uuencode content type.

Examples

MAKEMAIL -tfred@yahoo.com -fmary@yahoo.com –sStatus -bmessage.txt – omail.dat
MAKEMAIL -tfred -tbarney -fwilma –sDinner –bmenu.txt –omail.dat

Example Script File

MAIL.SCR:

-tfred@yahoo.com
-tbarney@yahoo.com
-fwilma@yahoo.com
-sStatus
-bmessage.txt
-aFile1.bin
-aFile2.bin
-omail.dat
MEMSTAT displays memory usage within SOCKETS. MEMSTAT.EXE is provided with source code, MEMSTAT.C, as an example of how to access SOCKETS memory usage information from within an application.

Syntax

MEMSTAT [F]

Remarks

Without an argument, MEMSTAT displays memory usage as follows (the actual values may differ):

MEMSTAT version 1.03

<table>
<thead>
<tr>
<th>Information</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sockets build</td>
<td>1565</td>
</tr>
<tr>
<td>Memory available</td>
<td>16032</td>
</tr>
<tr>
<td>Largest available block</td>
<td>15260</td>
</tr>
<tr>
<td>Memory allocation failures</td>
<td>0</td>
</tr>
<tr>
<td>Memory free errors</td>
<td>0</td>
</tr>
<tr>
<td>Minimum stack observed</td>
<td>946</td>
</tr>
</tbody>
</table>

With an argument, the following information is added (the actual values may differ):

Free chain in segment 14DF:

<table>
<thead>
<tr>
<th>Address Range</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>0A98-0A9F</td>
<td>4</td>
</tr>
<tr>
<td>0F3C-4ADB</td>
<td>15260</td>
</tr>
<tr>
<td>4B9C-4D33</td>
<td>404</td>
</tr>
<tr>
<td>4D5C-4D7F</td>
<td>32</td>
</tr>
<tr>
<td>53C8-54BF</td>
<td>244</td>
</tr>
<tr>
<td>5500-551B</td>
<td>24</td>
</tr>
<tr>
<td>5C7C-5CAB</td>
<td>44</td>
</tr>
<tr>
<td>5D00-5D17</td>
<td>28</td>
</tr>
<tr>
<td>Total</td>
<td>16032</td>
</tr>
</tbody>
</table>

NETBIOS

NETBIOS is a TSR that provides a NETBIOS-compatible network interface.

Syntax

NETBIOS [options]
Options
/a=NameCount  
/b=BroadcastFile  
/c=NCBs  
/l=LanAdapterNumber  
/n=HostsFile  
/s=SessionCount  
/u

Remarks
/a=NameCount  
Maximum number of names to cache.
/b=BroadcastFile  
Local file containing IP addresses not on the local network segment that should be considered part of the “broadcast” group.
/c=NCBs  
Number of NCBs to allocate.
/l=LanAdapterNumber  
For the NETBIOS API, the adapter number of this interface.
/n=HostsFile  
Local file name containing name-to-IP mappings.
/s=SessionCount  
Number of simultaneous sessions to allow.
/u  
Running NETBIOS again with this option unloads the TSR.

Configuration File
NETBIOS uses a file to statically map names to IP addresses. This is the HostsFile. The file contains space-separated parameters in the format:

name IP

NETBIOS uses a second file to list machines that should be added to the local “broadcast” group. This is the BroadcastFile. The file contains one parameter per line in the format:

IP

Configuration File Parameters
name  
The local alias for this entry.
IP  
The remote IP address of this entry.
Example

```
NETBIOS /n=lmhosts
```

Note

Filesharing between a sockets system and other systems can be accomplished by using NETBIOS with third-party or freeware redirector and/or server software.

PDTEST

The PDTEST utility is a diagnostic program that tests loaded packet drivers. PDTEST does not perform a full test since it does not transmit traffic through the drivers, but just checks their status.

It reports such things as the interrupt vector of the packet driver, its class, the MAC address, and information on the status of the driver. PDTEST is useful for checking that the packet driver was actually loaded which interrupt was used, and the class of packet driver. Classes supported by SOCKETS are marked with an asterisk (*) in the following list of recognized classes:

- Class 1* DIX Ethernet_II
- Class 3* 802.5 Token Ring
- Class 5 Appletalk
- Class 6* SLIP
- Class 9 AX.25 Amateur Radio
- Class 11* 802.3 with 802.2 headers IEEE
- Class 12 FDDI with 802.2 headers
- Class 13 Internet X.25
- Class 14 Northern Telecom LANSTAR encapsulating DIX
- Class 16 Point to Point Protocol for serial lines
- Class 17 802.5 Token Ring w/expanded RIFs

Most Ethernet packet drivers support both classes 1 and 11. Class 1 is the default class and should normally be used.

Syntax

```
pdtest
```

Remarks

Packet driver found at 0x60

Version 9, class 1, type 57, number 0, functionality 6 Name: MAC/DIS converter

High performance driver
PING sends ICMP echo requests to an IP host and reports on the reception of subsequent echo replies. Source code is provided as an example of the Basic API (CAPI) and in order for customers to build the functionality into their own applications.

Syntax
PING [options] ip-address [scope-id]

Remarks
ip address This may be a numeric address or a name address.

Options
/d=time
Delay for time milli-seconds between pings.
/i=incr
Increment length of each packet by incr from len to max.
/l=len
Length of packet.
/m=max
Maximum length of packet. Used with /i and /r options.
/n=number
Number of pings.
/q
Quiet, only print stats.
/r
Random length from len to max.
/t
Ping until terminated by s, t, q, x or Esc
Other keys print current statistics..
/x
XPING-like report on a single line.
/w=time
Wait for response for time milli-seconds.
**ip-address**
The IPv4 or IPv6 address to be pinged.

**scope-id**
Used to specify an interface to be used in the case of an IPv6 link-local addresses. It specifies the number of the interface as defined in the configuration file (e.g. SOCKET.CFG) starting at 1.

**Examples**

PING 10.0.0.1 20
This will ping the address of 10.0.01 every 20 clock ticks.

ping /x /t /d=500 host25
This will give the same functionality as XPING host25.

---

**RC.JAR**

**RC_JAR** is a Java remote console applet for connecting to hosts running **HTTPD**. It shows the text-mode contents of the **HTTPD** machine’s display memory, and allows fully interactive keyboard input. This is an alternative to the **RCCLI** applet.

**Syntax**

The link to RC.JAR should be embedded in an HTML page served by the HTTPD server. An example is available in the CGI/ subdirectory of the HTTP applications.

**Remarks**

RC_JAR has been compiled for use with the Java Development Kit version 1.4.1. The browsers Mozilla 1x and newer, as well as Microsoft’s Internet Explorer 4.0 and newer are supported. In order to use with a Netscape Browser a security certificate must be compiled into RC.JAR.

---

**RC_DK.JAR**

**RC_DK_JAR** is a Java remote console applet specifically for Danish keyboards. Please see **RC.JAR** for more information.

---

**RCCLI, WRCCLI AND W9RCCLI**

**RCCLI** is a DOS remote console client for connecting to hosts running HTTPD, and is an alternative to the **RC.JAR** applet. **WRCCLI** is a Windows NT, 2000,XP console mode client with the same functionality while **W9RCCLI** is the Windows 9x counterpart without support for IPv6. It shows the text-mode contents of the HTTPD machine’s display memory, and allows fully interactive keyboard input. To exit the remote console client, press Ctrl-Alt-X.
ROUTE

ROUTE is used to display or modify routes. The source code, ROUTE.C, is provided as an example of manipulating routing information.

Syntax

route [print]
route default
route drop <destination>[/bits] [/m=mask] [/l]

Options

/x - proxy
/p - private
/s - static
/b - add at bottom
/l - list new routes

Examples

route

<table>
<thead>
<tr>
<th>Destination</th>
<th>Bits</th>
<th>Interface</th>
<th>Gateway</th>
<th>Metric Proto</th>
<th>Flags</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>0</td>
<td>if0</td>
<td>192.168.0.200</td>
<td>Netmgmt</td>
<td>s</td>
<td></td>
</tr>
<tr>
<td>192.168.0.0</td>
<td>24</td>
<td>if0</td>
<td>none</td>
<td>Netmgmt</td>
<td>s</td>
<td></td>
</tr>
</tbody>
</table>
SENDMAIL

SENDMAIL delivers e-mail messages packaged by the MAKEMAIL application to an Internet mail server. SENDMAIL also creates a local log file to indicate successful send or failures.

Syntax
SENDMAIL server file

Options
Server
The IP address or DNS name of the Internet mail server to receive the message.

File
The file, created by the MAKEMAIL utility, to deliver.

Logging Format
Timestamp, Code String

Timestamp
Weekday Month Day Time Year.

Code
Three digit integer. 000 means perfect success, 100-199 mean usage error and 200-299 means TCP/IP error from server.

String
Human–readable explanation of the error code.

Examples
SENDMAIL mail.datalight.com mail.dat

SETHOST

The SETHOST utility sets an environment variable (default HOSTNAME) to the name contained in a map file (default MACHOST.MF) according to the hardware address (MAC or Ethernet) found by searching for a packet driver tsr. Network management is simplified when using %HOSTNAME% in the SOCKET.CFG files to set IP addresses.

SETHOST can be used in either of two modes:

• To update the MACHOST.MF file or
• To set the HOSTNAME environment variable.

Syntax
sethost [/f=n | /n=hostid] [/c=] [/m=map_file] [/v=variable]
Options

without a hostid set the variable.

/f=n
Use nth environment block (required for some systems - try /n=1 first.).

/n=hostid
This is the IP address of the local PC in symbolic form as in HOSTS or in decimal form to add to the mapfile.

/c=
Preserve the case of the environment variable.

/m=map_file
Use map_file instead of default MACHOST.MF.

/v=variable
Use variable instead of the default name: SETHOST.

Note: The equal signs are required in this case since SETHOST supports applications where the n= is optional. A slash without an equal sign indicates the setting of a host id.

Examples

To modify or add an entry in MACHOST.MF:

sethost /n=ws_name

To set the variable HOSTNAME:

sethost

Installing SETHOST

Installing SETHOST requires a working installation of SOCKETS. Also, all workstations run SOCKETS from a server disk.

Edit the HOSTS file to add all the workstations’ names and IP addresses. We recommend that all the workstation IP addresses be preceded with an asterisk to make them hidden to other users looking into the list of hosts. For example, *198.147.35.120 admin03 Copy the SETHOST.EXE program and an empty file MACHOST.MF to your server, in a directory such as X:\SOCKETS, for example.

At each workstation, log in as supervisor (or have write access to X:\SOCKETS) and execute the DOS commands:

x:
CD \SOCKETS\UTILS
SETHOST /N=WS_NAME
SETHOST
SET
The SETHOST /N=WS_NAME command creates an entry in the MACHOST.MF file with the name of the workstation and its MAC (Ethernet card) address. This is the important “once-only” command. The next two commands are to verify that the ws_name is stored in the environment variable HOSTNAME.

In the AUTOEXEC.BAT or any batch file executed after login and before running SOCKETS, put the following commands:

```
x:
CD \SOCKETS\UTILS
SETHOST
```

The IP addresses are now linked to the MAC addresses of the network cards. If a network adapter or host system is changed, update the MACHOST.MF file with

```
sethost /ws_name
```

The MACHOST.MF file keeps the mapping from the MAC addresses to the workstation names and the HOSTS file maps the workstation name to its IP address.

The variable name HOSTNAME and filename MACHOST.MF are the defaults for SETHOST. EXE but can be user-specified. Run SETHOST /? to display the available options.

---

### SNTPCLI

SNTPCLI is used for getting system network time information from a system network time protocol (SNTP) server.

**Syntax**

SNTPCLI server [minutes]

**Remarks**

server The server providing the time service.

**Options**

minutes An optional parameter which provides an adjustment to the server time in minutes.

**Example**

SNTPCLI 182.109.2.80

Retrieves the time information from the server specified.

**Time Zones**

An additional environment variable for time zones is also supported. The TZ variable allows SNTPCLI to compensate for time zone differences on the SNTP server when setting the local system time. By default, if no time zone is set, SNTPCLI assume Greenwich Mean Time (GMT).
SOCKDIAG is a utility used to diagnose the SOCKETS Advanced TCP/IP interface. It tests the API as well as the stack and even the network when external servers are specified. It also measures the performance of the network for TCP and UDP traffic. The external server(s) need to run both TCP and UDP echo servers on TCP port 7 and UDP port 7. A Sockets machine can be used as a server by running SUPERD. The source code for SOCKDIAG is supplied as an advanced example of using the Sockets API and can be compiled without any change as a Windows program using Winsock to demonstrate the portability of the code.

Syntax

```
```

Options

- `/b` - set the number of seconds for each traffic test to run.
- `/d` - specify server debugging.
- `/f` - set the action to take on a failure. Valid values are:
  - 0 print error and continue
  - 1 print error and wait for Esc.
  - 2 print error and exit

Setting the TZ variable is not necessary when using all programs. The syntax for the TZ environment variable is:

```
SET TZ= <abbreviation> +|- value
```

Abbreviation represents any three-letter abbreviation for the chosen time zone. The variable serves as a reminder to the user. For example, if setting the time zone for Pacific Standard Time, the variable could be set as PST and for Eastern Standard Time as EST. The abbreviation is only a placeholder in the syntax for the TZ variable. There are no incorrect abbreviation choices as long as only three letters are used.

Value represents the number of hours this time zone varies from GMT. For example, the west coast of the United States and Canada is –8 hours relative to GMT. This value may have to be adjusted to compensate for daylight savings time. There should be no spaces between the abbreviation, plus or minus sign, and the value. Some examples are:

```
SET TZ=PST –8
SET TZ=CMT –3
SET TZ=GMT+2
```

If an incorrect format for the time zone is entered, the default of GMT is used.
/l - set the address of the local echo port.
/n - specify that the local loopback traffic tests should be skipped
/r - set the address of the remote echo port.
/s - specific server debugging.
/t - set the number of seconds before a timeout occurs.
/v option set the level of verbosity. Valid values are:
   0  only print errors
   1  print the names of all tests
   2  print additional information on errors
   4  print getaddrinfo() values

Example

SOCKDIAG  /n  server1
This will skip the local loopback traffic tests, but test server1.

SOCKSTAT

SOCKSTAT is used to display the status of all sockets (as used for the Basic and Advanced APIs) and also all the UDP listening ports. Note that it is likely that there will be more TCP and UDP connections than sockets in the system. Built-in servers like LPD and most of the server and client applications like FTPD and FTP use the Proprietary API which does not make use of sockets.

Syntax

SOCKSTAT

SRPRINT

SRPRINT is a small TSR allowing a serial printer to be used via BIOS INT17, thus allowing the Sockets built-in Socket Printing and LPD to support serial printers. In addition it allows the SocketPrinting server to act as a Serial Port server (conversion of uni- or bi-directional serial streams to TCP streams).

Syntax

SRPRINT [options]
Options

/b=baud-rate
/c=comm-port 1 or 2
/d=data-bits 7 or 8
/e=receive-flow-control rts, dtr, xon, itrms, idtr
/f=send-flow-control cts, dcd, dsr, xon, icts, idcd or idsr
/n=printer-number 0, 1, 2 or 3
/l list current parameters
/p=parity none, odd, even, mark or space
/r reset flow control
/s=stop-bits 1 or 2
/t=timeout timeout in seconds
/u unload
/z=buffer-size must be more than 40

Example

srprint /c=1 /f=xon /n=2 /b=19200
This will direct printer port 2 to COM1, use xon/xoff flow control and set the baud rate to 19200

srprint /c=1 /l
This will list the current settings and status of the serial printer (actual values may differ):

Serial printer rev 2.00 parameters:

Comm port: 1  Send buffer count: 0
Printer number: 2  Recv buffer count: 0
Baud rate: 19200  Errors:
Data bits: 8  Flow: 0
Parity: None  Overrun: 0
Stop bits: 1  Parity: 0
Send flow control: Xon/Xoff  Framing: 0
Recv flow control: n/a  Break: 0
Buffer size: 932 Status: Idle.
TCP

TCP.EXE is a utility used to examine and change the TCP parameters and connections.

Syntax

TCP close n
TCP irtt [n]
TCP mss [n]
TCP reset n
TCP retry [n]
TCP status [/columns [/v=mode [refresh time]]]
TCP status n [e]
TCP window [n]

Remarks

n Connection number in decimal or a TCB address in hexadecimal.
close Close connection specified by n.
irtt Set, using n, or display the Initial Round Trip Time.
mss Set, using n, or display the Maximum Segment Size.
reset Reset connection n.
retry Set, using n, or display the retry count.
status Display summary status of all connections.
status n Display detailed status for connection n. Add e for extended status.
window Set, using n, or display window size.

Options

columns The number of columns to display status with refresh.
mode 0 – direct video, 1 – VT100/ANSI sequences, 2 – Standard output.
refresh time The time in seconds after which an automatic refresh occurs.

The following keys are active while refreshing the display:

Esc Exit.
d Enter number of times to read status before displaying it.
f Toggle freeze/unfreeze.
n Enter number of newest entries to highlight.
s Switch to different sort mode: I – Internal (unsorted), H – Host, T – Time
1,2,3,4 Select the number of columnsAny otherRefresh display.
Examples

TCP irtt
Displays the current irtt value.

TCP s /2 1
Displays the status of all connections in two columns, refreshing every second, until Esc is pressed.

XPING

XPING starts a continuous string of pings until stopped by a keystroke. XPING has been superseded by PING. Source code is provided as an example of the Proprietary API.

Syntax

XPING ip address [interval]

Remarks

ip address This may be a numeric address or a name address.

Options

interval The time to wait between pings in clock ticks (each about 55 milliseconds long).

Example

XPING 10.0.0.1 20
This will ping the address of 10.0.0.1 every 20 clock ticks.
HTTP SERVER

Overview
The SOCKETS HTTP server, HTTPD.EXE, is a small, fast, reliable and extendable web server that can run as either an application or TSR. Apart from the minimum required file download capability, the following additional capabilities are provided:

1. Remote Console Server- ability to gain terminal-type access to the server system, using a standard browser, without the need to install any software on the browser computer

2. Authentication – Both system wide and directory wise

3. CGI Extendibility – The ability to extend the server to create dynamic web pages, perform specialized tasks, etc.

4. A Server Side Includes (SSI) interface is provided using the CGI interface, enabling a user to create web pages using HTML templates with variable names, which is substituted in-time with specific values. WebForms, a more comprehensive SSI interface, is also provided allowing WebDos applications to be easily developed. WebDos commander and WTCP are based on WebForms.

5. Ability to run as a background process

6. Flexibility to control physical parameters such as memory usage and number of connections

Server
The HTTP server is used to send static web pages existing as files on the server or dynamically generated web pages to a remote client (browser). Dynamic pages can be generated in two ways:

1. Extension CGI. By calling an external CGI handler, the server provides an API to external handlers. A Server Side Includes (SSI) interface and WebForms are provided as well, which makes it very easy to create powerful interactive web pages.

2. Spawning CGI. The basic mechanism used by CGI is that arbitrary programs can be spawned from the web server with input as received from the remote browser and output that can be sent to the browser.

The Remote Console Server accepts input from a remote client that is fed to the keyboard buffer for use by an arbitrary program using it. It also monitors the screen display buffer area and sends screen information to the remote client.

The SOCKETS password file controls authentication. Authentication is user specific and may also differ from directory to directory. It may also be put off for either some or all users. See the section on authentication. It is also possible to embed a single backdoor user account into the executable
server program file. For security reasons, the documentation for this feature must be requested from Datalight.

The HTTP server can support multiple simultaneous sessions. The GET, PUT and POST request methods are implemented as well as the following MIME types:

  text/html, text/plain, image/gif, image/jpeg, image/jpeg and application/octet-stream.

The MIME type is determined by the file extension.

**Remote Console Server**

**Initialization**

When the DOS client RCCLI.EXE or the Windows client WRCCLI.EXE or the Windows 9x client W9RCCLI.EXE is used, a connection is simply established to the Remote Console Server on TCP port 81.

When a browser is used as the client it retrieves a page containing a reference to the Remote Console applet that is automatically downloaded if not yet in the cache. The applet then connects to the Remote Console Server (RCS) on TCP port 81. An example download page is supplied as REMCON.HTM.

Almost any application e.g. COMMAND.COM or even a full-screen text editor like NED.EXE can be run on the server. The remote keyboard and display control the application as if they were locally attached.

On the remote side, the client acts as a simple terminal emulator that displays what it receives from the server and sends what is entered from the keyboard to the server.

It is not required to have a real display adapter on the embedded system server, only to have display buffer memory and a BIOS and application software that writes to the display buffer memory.

When a new connection is made, all the screen data, as well as the cursor position, is sent to the client. Subsequently the RCS keeps a watch on the video memory and cursor position and whenever a change is detected, the RCS sends the changed data to the client.

Keyboard data received from the client is passed to the keyboard buffer making it available as keyboard input for use by any application executing on the server.

The remote console applet is a Java 1.4.1 applet, supplied as RC.JAR, and will function on any Java 1.4.1 compliant browser. For Internet Explorer it may be required to download, install and select Sun Java for applet operations. Please note that a security certificate has not been compiled into RC.JAR so it is not compliant with versions of the Netscape browser that require a security certificate to run Java applets. For additional information about RC.JAR or RCCLI.EXE, please see the Utility Description Chapter.
Extension CGI

The SOCKETS HTTP servers (HTTPD/HTTPFTPD) provide a facility to call functions in other modules which may be TSR or transient programs. These functions are referred to as “HTTPD extensions.” For more information please see the “ROM-DOS Developer’s Guide” section “CGI Application API”.

Extension CGI Examples

A number of examples are included to demonstrate the usage of both the Spawning and Extension API. Source code is included, as well as a make file and a BorlandC 5 IDE file.

Put all .htm and .exe files in the %HTTP_DIR% (default is \dl\sockets\server) directory and start HTTPD, HTTPFTPD or SUPERD. The CGI programs may be pre-loaded or WebDos Commander may be loaded and used to invoke the CGI programs. Examples may be accessed through index.htm.

The only Spawning CGI example is form.exe and it is accessed through form.htm. Please note that COMMAND.COM must not be the currently executing program on the server as it uses all available free memory. WEBDOSC.EXE is a good program to have executing on the server.

The examples may operate in one of two modes:

As a TSR (resident) program: this is the default behavior. Unloading of the TSR can be performed by the user from the remote browser session or by unloading the server. De-registration is possible by loading the program again. This routine may be repeated.

As a transient program: use the ‘/t’ command line switch to activate. This option will immediately spawn ‘command.com’. From this prompt other cgi programs may be loaded. The program exits when ‘command.com’ is exited by typing ‘exit’ at the prompt.

These programs are:

WEBDOSC

WebDos Commander presents a “Commander” like interface via a remote browser allowing a user to navigate the directory, copy, view, delete and execute files and create and delete folders (directories). WEBDOSC is best run as a transient program and not a TSR since that makes it possible to execute other programs which is not possible when COMMAND.COM is executing. Access it from the browser as webdosc.htm.

WTCP

WTCP dynamically displays the status of TCP connections on the server machine. Access it from the browser as wtcp.htm or from WebDOS Commander as wtcp.exe. WTCP can be unloaded using the browser interface by pressing the “Exit” button.

CGIDEMO

This program contains three Extension API functions:

1. A simple program that accepts data from a user and echoes it back nicely formatted. Access echoform.htm from the browser.
2. A page visit counter. Access num.htm from the browser.
3. Does the same as the form.exe spawn example. Get caform.htm from the browser.

SSI
A simple SSI implementation that demonstrates the SSI interfaces. Template.htm is filled by some variables. Get ssi.htm from the browser.

FFUR (Form-base File Upload Receiver)
It handles the upload of a file using POST commands. Access ffur.htm.

Passive Mode
The server may be run in passive mode by specifying a '/p' command line switch. When passive, the server will record network events but only handle them once it is triggered by a CGI user.

Server Memory
The server’s memory usage may be controlled in two ways:

1. By specifying the amount of memory when going TSR.
2. By specifying the maximum number of connections the server will allow.

Option 1 is the recommended option. Use Option 2 if you have ‘heavy’ web pages – usually the type where pages consist of frames and many images, etc. Connections are generally reset when more connections are attempted than the defined maximum. The client then must retry to establish the lost connections, leading to a more distributed load on the server.

Spawning CGI
An external program, indicated by the requested URL, is spawned. All relevant information is passed as environment variables. The CGI program gets all input (e.g. posted data) from standard in and sends all response through standard out. Spawning CGI is discouraged in favor of Extension CGI. For more information please see the “ROM-DOS Developer’s Guide” section “CGI Application API”.

Authentication
Default authentication matches the capabilities of the FTP server as documented in the section “FTP Server” on page 252. A file called “SOCKET.UPW” should exist in the SOCKETS (environment variable) directory.

The default permission file controls remote console access. Each listed user has a single-letter privilege code set if he has privilege to use the Remote Console. The code should be missing if that user does not have Remote Console privilege.

An additional authentication feature is implemented - htaccess. This feature provides a per-directory permission override mechanism. It is enabled using ‘t’ as command line switch. If htaccess is enabled, the default mechanism may be skipped (but no default users or remote console access will be available).
A file called HTACCESS (typically hidden) contains authentication overrides to enable partial anonymous access or additional password security to subdirectories, etc. If this feature is activated, the server code will look for HTACCESS files in each directory starting from the requested path and continuing upward in the directory structure (assuming the root directory to be at the top) until an HTACCESS file is found. If no file is found, then the default settings are used. An anonymous access entry is available for the developer to specify that some subdirectory is authorized for any user, although its parent directory is password-protected. CGI scripts can also be controlled via the HTACCESS mechanism.

**HTTPD Program**

The syntax for HTTPD is:

```
HTTPD [options] [<http_port>] [<rc_port>]
```

Any combination of these switches may be used. They should be separated by at least one space.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>/?</code></td>
<td>display help screen</td>
</tr>
<tr>
<td><code>/r</code></td>
<td>run server in TSR mode</td>
</tr>
<tr>
<td><code>/s</code></td>
<td>display server status</td>
</tr>
<tr>
<td><code>/t</code></td>
<td>enable htaccess directory level authentication</td>
</tr>
<tr>
<td><code>/u</code></td>
<td>unload if resident</td>
</tr>
<tr>
<td><code>/c</code></td>
<td>close listen</td>
</tr>
<tr>
<td><code>/d</code></td>
<td>do not start remote console</td>
</tr>
<tr>
<td><code>/g</code></td>
<td>allow old type (spawning) CGI</td>
</tr>
<tr>
<td><code>/p</code></td>
<td>Passive mode</td>
</tr>
<tr>
<td><code>/i=InterruptNumber</code></td>
<td>Interrupt number for cgi API</td>
</tr>
<tr>
<td><code>/m=MemorySize</code></td>
<td>set memory size</td>
</tr>
<tr>
<td><code>/n=MaximumConnections</code></td>
<td>number of simultaneous connections</td>
</tr>
<tr>
<td><code>/a=ScreenX&gt;, &lt;ScreenY</code></td>
<td>set screen aspect</td>
</tr>
<tr>
<td><code>/v=ScreenBufferSegment</code></td>
<td>set video buffer address (hex)</td>
</tr>
<tr>
<td><code>/k</code></td>
<td>Abort all active connections without confirmation when unloading.</td>
</tr>
<tr>
<td><code>/l=Pathname[,CacheEntries[,Timeout[,[TimeFormat[,Format][]]]]]</code></td>
<td>Log login/out events to a file.</td>
</tr>
</tbody>
</table>
Remarks

_ScreenX, ScreenY_
The width and height of the screen area to serve for the remote console session. These values default to 80 and 25, respectively.

_ScreenBufferSegment, ScreenBufferOffset_
Together, a pointer to the top-left corner of the display memory to serve for the remote console session. These values default to B000 and 0000 respectively, for monochrome display adapters and to B800 and 0000 respectively, for color display adapters.

_MemorySize_
The maximum amount of memory available to the server. The default value is 32K. The value of m can range from 8192 to 63472.

_MaximumConnections_
The maximum number of simultaneous connections allowed by the server.

_InterruptNumber_
The interrupt number to access the CGI API.

_Pathname_
The pathname to log login/logout events to.

_CacheEntries_
The number of cache entries for keeping track of authenticated HTTP (Web) sessions. An HTTP session is not uniquely defined as is the case of an FTP or Remote Console session where specific login and logout events occur. In the case of HTTP each request must be authenticated by means of an Authentication Header. This is normally done automatically by the client (Browser) by presenting a single login dialog and applying the values entered to all requests sent to the specific remote host and port. There is never an explicit logout event which is sent to the server. To simulate a logout event, HTTPD keeps the username/password combinations of a number of “sessions” in a cache with a timeout period. When a new request for an existing entry is received, the timeout period is reset to the maximum value. When the timeout expires, the session is deemed to be logged out. If too many sessions for the specified number of cache entries are required, the oldest session is “logged out” and the entry is re-used for the new session. A subsequent request with the username/password of the logged out session, will be handled as a new session. The default number of entries is 20.

_Timeout_
The timeout period for cache entries in seconds. The default is 900 seconds (15 minutes).

_TimeFormat_
The format in which the time field is logged, can be specified using the specifiers used by the strftime() function of C. The default format is "%m-%d-%y/%I:%M%p". If it is required to use a ‘;’ in any of the formats, a different non-digit character should be used to terminate the Timeout parameter e.g. a ‘;’. The same character should be used to terminate the TimeFormat parameter if a Format parameter is also given.
Format
The format of a log event entry can be specified as in the C printf() function. The default format is “%-3s %-10.10s %-10.10s %s %-20.20s”. All arguments are string variables. The first argument is a three character string with the first character specifying the Service as follows:
F - FTP, H - HTTP, R - Remote Console
The second character specifies the type of event as follows:
I – login, O – logout, S – startup.
The third character specifies the status of the event as follows:
S – Success, F - Failure
The second argument is the Username, the third the Password, the fourth the Time (as specified by the TimeFormat) and the last the IP address. Whenever a parameter is not applicable, “---” is used.

http_port
HTTP port to listen on. This parameter defaults to the standard HTTP port number of 80.

rc_port
Remote Console port to listen on. This parameter defaults to 81.

The “root” directory for web content is the current directory when HTTPD is started. This can be changed by setting an environment variable HTTP_DIR e.g.
SET HTTP_DIR=D:\SERVER\WEB

Format of “SOCKET.UPW”
This is the same file used for the FTP server’s (FTPD.EXE) permissions. This file consists of lines where each line contains a user’s information. A line starting with a # is considered a comment and is ignored. Each line consists of four fields:

<Username> <Password> <Working Directory> <Permissions> [# comment]

Username: The name of this user. If it is *, it will be used when the client does not specify a username.
Password: This user’s password. If it is *, no password is required
Working Directory: The user will only have access to this directory and its subdirectories. If it is ‘/’, this user has access to the whole system. HTTP_DIR can be referred to as ‘\’. If a relative path is specified, it is appended to HTTP_DIR.
Permissions: IMPORTANT when a user is granted both FTP and HTTP permissions, the FTP permissions must appear first, otherwise they will be ignored. Operations allowed. May contain any combination of the following tokens:
e - User may ‘get’ files
p - User may ‘post’ data (Only allowed for CGI)
u - User may ‘put’ files
g - User may use cgi
m - User may use Remote Console

Fields should be separated by single spaces. If any field is missing the entry is ignored. A comment may follow the last field (permissions) of the line.

**Note:** If a default user is supplied, it should always appear first in the list of users. Only users below the default user will be considered.

**Format of “htaccess”**
Any directory may contain this file, and serve as overrides to the general permissions for the containing directory and all its subs until another htaccess is found. This file consists of lines where each line contains a user’s information. A line starting with a # is considered a comment and is ignored. Each line consists of three fields:

<Username> <Password> <Permissions> [# comment]

**username:** The name of this user. If it is *, it will be used when the client didn’t specify a username.

**Password:** This user’s password. If it is *, no password is required.

**Permissions:** Operations allowed. may contain any combination of following tokens:

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>e</td>
<td>User may ‘get’ files</td>
</tr>
<tr>
<td>p</td>
<td>User may ‘post’ files</td>
</tr>
<tr>
<td>u</td>
<td>User may ‘put’ files</td>
</tr>
<tr>
<td>g</td>
<td>User may use cgi</td>
</tr>
</tbody>
</table>

Fields should be separated by single spaces. If any field is missing the entry is ignored. A comment may follow the last field (permissions) of the line.

**Note:** If a default user is supplied, it should always appear first in the list of users. Only users below the default user will be considered.
FTP SERVER

FTPD is a file server that can run either as an application or as a TSR. The name of the server as displayed in the banner is determined by the HOSTNAME environment variable. If the environment variable is not set, the name “Socket” is used. The user password file, SOCKET.UPW, in the SOCKETS directory (indicated by the SOCKETS environment variable) controls access.

A temporary file is created when a directory listing is requested. This file is created in the current directory, but can be created in any directory as specified in the FTPDIR environment variable.

FTPD Program

The syntax for FTPD is:

FTPD [options] [<ftp_port>]

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/? /h</td>
<td>display help screen</td>
</tr>
<tr>
<td>/r</td>
<td>run server in TSR mode</td>
</tr>
<tr>
<td>/s</td>
<td>display server status</td>
</tr>
<tr>
<td>/u</td>
<td>unload if resident</td>
</tr>
<tr>
<td>/c</td>
<td>close listen</td>
</tr>
<tr>
<td>/c</td>
<td>close listen</td>
</tr>
<tr>
<td>/m=MemorySize</td>
<td>set memory size</td>
</tr>
<tr>
<td>/n=MaximumConnections</td>
<td>number of simultaneous connections&gt;</td>
</tr>
<tr>
<td>/k</td>
<td>Abort all active connections without confirmation when unloading.</td>
</tr>
</tbody>
</table>

Remarks

MemorySize
The number of bytes of memory available to the server. This value defaults to 32768.

MaximumConnections
The maximum number of simultaneous connections allowed by the server.

ftp_port
FTPD will listen on the listed port. This parameter defaults to the standard FTP port number of 21.

Configuration File

FTPD uses the standard SOCKET.UPW file for validating logins. The file is composed of text lines, each representing a login name, password, and the configuration to use for a session opened with those credentials. Space characters separate the parameters in the file, which are in the following format:

name password directory rights
The location of the username/password file to be used by the server is specified by the environment variable SOCKETS as follows:

```
%SOCKETS%\SOCKET.UPW
```

If the variable SOCKETS is not specified, the following file is used:

```
\DL\SOCKETS\SOCKET.UPW
```

**Configuration File Parameters**

- **name**
  - The login name of this record.

- **password**
  - The password to authenticate a user trying to login as this name.

- **directory**
  - The starting directory for this user.

- **rights**
  - Up to four characters specifying which permissions this user is granted:
    - r means that this user has read access.
    - w means that this user has write access.
    - c means that this user has permission to make new directories.
    - d means that this user has permission to change to a directory other than his starting location and subdirectories from the starting location.

**Example Socket.upw**

```
Admin admin c:\ drwc
Guest * c:\guest dr
```

**Example Command Line**

```
FTPD /m=40000 /rc
```

**FTP SERVER COMMANDS**

The following commands are recognised by the SOCKETS FTP server:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>abor</td>
<td>cancel an incomplete transfer</td>
</tr>
<tr>
<td>acct</td>
<td>specify account – ignored by server</td>
</tr>
<tr>
<td>appe</td>
<td>&quot;put&quot; a file at the server but append it if the file exists</td>
</tr>
<tr>
<td>cdup</td>
<td>change directory up</td>
</tr>
<tr>
<td>cwd directory</td>
<td>change server directory</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>dele file</td>
<td>delete a server file</td>
</tr>
<tr>
<td>epasv</td>
<td>request extended connection information from the server in order to make a connection to the server for transfer of data</td>
</tr>
<tr>
<td>eprt</td>
<td>send extended port information to the server requesting the server to make a data transfer connection</td>
</tr>
<tr>
<td>feat</td>
<td>list special features supported</td>
</tr>
<tr>
<td>help</td>
<td>request a list of implemented commands</td>
</tr>
<tr>
<td>list [file</td>
<td>directory]</td>
</tr>
<tr>
<td>mdtm [file</td>
<td>directory]</td>
</tr>
<tr>
<td>mkd remote_directory</td>
<td>create a server directory</td>
</tr>
<tr>
<td>nlist [file</td>
<td>directory]</td>
</tr>
<tr>
<td>noop</td>
<td>no operation – acknowledged by server</td>
</tr>
<tr>
<td>pass [password]</td>
<td>password for username</td>
</tr>
<tr>
<td>pasv</td>
<td>request connection information from the server in order to make a connection to the server in order to transfer data</td>
</tr>
<tr>
<td>port</td>
<td>send port information to the server requesting the server to make a data transfer connection</td>
</tr>
<tr>
<td>retr remote_file</td>
<td>transfer a file from the server in the current mode</td>
</tr>
<tr>
<td>stor local_file</td>
<td>transfer a file to the server in the current mode</td>
</tr>
<tr>
<td>pwd</td>
<td>print working directory</td>
</tr>
<tr>
<td>quit</td>
<td>terminate FTP session</td>
</tr>
<tr>
<td>rmd remote_directory</td>
<td>remove (delete) directory</td>
</tr>
<tr>
<td>rnfr existing_filename</td>
<td>rename a file, command 1 of 2</td>
</tr>
<tr>
<td>rnto new_filename</td>
<td>rename a file, command 2 of 2</td>
</tr>
<tr>
<td>size file</td>
<td>report the file size in bytes as a message prefixed with 213</td>
</tr>
<tr>
<td>stat</td>
<td>report the status of a transfer or active connections</td>
</tr>
<tr>
<td>syst</td>
<td>return operating system information from the server</td>
</tr>
<tr>
<td>type [i</td>
<td>a]</td>
</tr>
<tr>
<td>user [username]</td>
<td>username to logon</td>
</tr>
</tbody>
</table>
COMBINED HTTP, RC AND FTP SERVER
HTTPFTPD is a combined HTTP, RC and FTP server that can run either as an application or as a TSR. By default, it processes normal HTTP requests on port 80 and normal FTP requests on port 21 and Remote Console on port 81. Refer to HTTPD and FTPD descriptions for information on using HTTPFTPD.

HTTPFTPD Program
The syntax for FTTPD is:
HTTPFTPD [options] [<http_port> [<ftp_port> [<rc_port>]]]

For options refer to the description of HTTPD and FTPD.

http_port
    HTTP port to listen on. This parameter defaults to the standard HTTP port number of 80.

ftp_port
    FTP port to listen on. This parameter defaults to the standard FTP port number of 21

rc_port
    Remote Console port to listen on. This parameter defaults to 81.

Example Command Linesw
HTTPFTPD /m=40000 /r
HTTPFTPD /a=80,25 /v=a000:0000 /r
HTTPFTPD /l=%http_dir%\netacces.txt,10,120;%x,%X;%s,%s,%s,%s,%s

The last example will generate a Comma Seperated Variable log file in the Web root directory of the web server.

COMBINED HTTP, FTP, RC AND ECHO SERVER
SUPERD is a combined HTTP, RC and FTP server as well as TCP and UDP Echo servers that can run either as an application or as a TSR. By default, it processes normal HTTP requests on port 80, normal FTP requests on port 21 and Remote Console on port 81. Echoservers for both TCP and UDP are at ports 7. Refer to the HTTPFTPD description for information on using SUPERD.

SUPERD is useful for running in a target system when using CAPIDIAG or SOCKDIAG to test a network.
OVERVIEW
SOCKETS provides support for a wide range of LAN adapters supporting the Packet driver specifications.

The packet driver specification is adhered to by many vendors of LAN adapters and is also freely available as public domain software from the Internet e.g. from www.crynwr.com. SOCKETS only supports the packet driver specification, but public domain converters or “shims” are available to convert from NDIS or ODI to packet driver. Some “shims” may be obtained from Datalight Inc. but are not distributed with SOCKETS. SOCKETS may use up to eight packet drivers.

Using the same network board, SOCKETS can operate simultaneously with other network operating systems such as Novell, Microsoft, Banyan and others.

**Note:** Software configuration of network hardware, such as Plug and Play settings or packet drivers, must be completed before SOCKETS is loaded. Failure to properly load and configure the appropriate software results in error messages from SOCKETS. To help you isolate which device is failing; the interface command is referenced in those error messages.

For development using Microsoft Windows, it is recommended to purchase an NDIS3 Virtual Packet Driver from www.danlan.com. Versions are available for both Windows 9X and Windows NT/2000. Use a different IP address for SOCKETS than that used by Windows.

**Note:** SOCKETS can be run in a Console Window (DOS box) using Windows NT/2000, but problems may be experienced scheduling it unless constantly calling an API function. XPING can be used for this purpose to run TSR servers.

The supplied utility PDTEST.EXE can be used to test or diagnose your packet driver before attempting to start SOCKETS. For further information on PDTEST, refer to Appendix ?? “Managing the Network and Troubleshooting.”

**Packet Driver Installation**
After you have installed SOCKETS, copy your NIC (Network Interface Controller) packet driver, from the NIC Driver disk. The various manufacturers supply their own packet drivers that may differ from what is documented here. Always consult the software and documentation supplied with your network controller first.

**Note:** Many PCI packet drivers require no command line parameters to load and, by default, set up an interrupt vector of 0x60.
APPENDIX A  |  PACKET DRIVERS

Using a Memory Manager with a Packet Driver
If a board using mapped memory (for example, SMC) is used with an upper memory block manager (EMM386), the shared memory must be excluded:

```
DEVICE=C:\DOS\EMM386.EXE X=D000-D3FF
```

PACKET DRIVER OVER ODI DRIVER INSTALLATION
If you already have Novell using ODI installed, just modify AUTOEXEC.BAT and the existing NET.CFG, otherwise make a \ODI directory on your hard drive and copy the following files to it:

<table>
<thead>
<tr>
<th>Filename</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSL.COM</td>
<td>Novell WSGEN floppy disk</td>
</tr>
<tr>
<td>NETX.EXE or VLM.EXE</td>
<td>Novell WSGEN floppy disk</td>
</tr>
<tr>
<td>IPXODI.COM</td>
<td>Novell WSGEN floppy disk</td>
</tr>
<tr>
<td>ODIPKT.COM</td>
<td>NIC Driver disk *</td>
</tr>
<tr>
<td>NIC ODI Driver</td>
<td>NIC Driver disk</td>
</tr>
</tbody>
</table>

* Note: ODIPKT.COM may not be provided by your vendor. In that case, there are many solutions available on the Web – just do a search for “odipkt.com”.

Examples of ODI Drivers:
SMC8000.COM, SMCPPLUS.COM, NE2000.COM, 3C5X9.COM or 3C503.COM
If you do not have a NET.CFG file, create an ASCII text file in the \ODI directory called NET.CFG that should look similar to the following:

```
Link Support
    buffers 8 1600
Protocol IPX
    Bind ODI_driver
Link Driver ODI_driver
    int irq
    port io port
    frame ETHERNET_802.3
    frame ETHERNET_II
```

The important parts to check are the buffers in the Link Support section and the order of the Ethernet Frame (also called Envelope Type) lines. Each Frame line specifies a logical board, starting from number 0. ODIPKT should link to the Ethernet_II board, which, in this example, would be board number 1. The file must contain at least the following:

```
Link Support
    buffers 8 1600
Link Driver ODI_driver
    frame ETHERNET_II
```
Example for SMC:

- Link Support
  - buffers 8 1600
- Link Driver SMCPLUS
  - Port #1 280
  - mem #1 00000000 2000/10
  - Int #1 3
  - frame ETHERNET_802.3
  - frame ETHERNET_II

Add the following lines to your AUTOEXEC.BAT file:

```
CD\ODI
LSL
rem Your NIC ODI driver
ODIPKT 1 96
IPXODI
NETX
F:
```

The syntax for ODIPKT.COM is:

```
ODIPKT logical_board interrupt_vector
```

The interrupt vector must be specified in decimal; for example 0x60 = 96.

logical_board is the index of the Frame type entry starting at 0. The normal frame type to use with SOCKETS on Ethernet is ETHERNET_II which is the second entry (or logical board 1) in the preceding example.

Example

```
CD\ODI
LSL
SMCPLUS
ODIPKT 1 96
```

Using a Memory Manager with an ODI Driver

If a board using mapped memory (for example, SMC) is used with an upper memory block manager (for example, EMM386), the shared memory must be excluded as follows:

```
DEVICE = C:\DOS\EMM386.EXE X=0000-D3FF
```

Packet Driver Over NDIS2 Driver Installation

After you have installed SOCKETS, make a \LANMAN directory on your hard disk and copy the following files to it:
Filename

PRO.MSG
PROH.MSG
PROTMAN.DOS
DIS_PKT.DOS
NETBIND.COM
NDIS 2 Driver for NIC

Add the following lines to your CONFIG.SYS file in the following order and not separated by any other command:

DEVICE=C:\LANMAN\PROTMAN.DOS
DEVICE=C:\LANMAN\<Your NIC's NDIS driver>
DEVICE=C:\LANMAN\DIS_PKT.SYS

Example

DEVICE=C:\LANMAN\PROTMAN.DOS
DEVICE=C:\LANMAN\SMC8000.DOS
DEVICE=C:\LANMAN\DIS_PKT.DOS

Add the following line to your AUTOEXEC.BAT file

C:\LANMAN\NETBIND

Create an ASCII text file called PROTOCOL.INI in the \LANMAN directory to pass parameters to the various drivers:

[PROTOCOL MANAGER]
DRIVERNAME=PROTMAN$

[PKTDRV]
DRIVERNAME=PKTDRV$
BINDINGS=<label_name>
INTVEC=0x60

[label_name]
DRIVERNAME= Your NIC’s NDIS driver name
IRQ=irq
RAMADDRESS= ram_base_addr
IOBASE= io_base_addr

Example

[PROTOCOL MANAGER]
DRIVERNAME=PROTMAN$

[PKTDRV]
DRIVERNAME=PKTDRV$
BINDINGS=WDMAC
INTVEC=0x60
[WDMAC]
  DRIVERNAME=WDMAC$
  IRQ=3
  RAMADDRESS=0xD000
  IOPBASE=0x280

**Using a Memory Manager with an NDIS Driver**
If a card using mapped memory (e.g. SMC) is used with an upper memory block manager (e.g. EMM386), the shared memory must be excluded as follows:

```
DEVICE = C:\DOS\EMM386.EXE X=0D000-D3FF
```

For more information on the NDIS drivers consult your network operating system documentation.
APPENDIX B | NETWORK MANAGEMENT AND TROUBLESHOOTING

This chapter describes solutions to common LAN problems, both configuration and performance.

NETWORK MANAGEMENT
The Network Manager is expected to:

- Setup each SOCKETS application according to user requirements.
- Monitor the status of various connections and trace traffic as it traverses the network, in order to resolve problems.
- Modify the software configuration to align it with changes in the physical environment on which it runs.

CONFIGURATION CASE STUDIES

Managing Host Names on a File Server-Based LAN
The SETHOST.EXE program manages the host names on a file server based LAN. The purpose of running SETHOST.EXE is to keep all the IP addresses in a single file on the server and allow all workstations to run the same software setup and yet maintain a unique IP address at each workstation. An alternative for non-file server-based networks is to use BOOTP or DHCP where a suitable server is available.

Installing SETHOST
Edit the HOSTS file to add all the workstation names and IP addresses. We recommend that all the workstation IP addresses be preceded with an asterisk to make them hidden to other users looking into the list of hosts. For example,

```
*198.147.35.120 admin03
```

Copy the SETHOST.EXE program and an empty file MACHOST.MF to your server in a directory that is named, for this example, X:\SOCKETS.

At each workstation, log in as supervisor (or have write access to X:\SOCKETS) and execute the DOS commands:

```
x:
  cd \SOCKETS\DOS
  SETHOST /N=WS_NAME
  SETHOST
  SET
```
The SETHOST /N=WS_NAME command creates an entry in the MACHOST.MF file with the name of the workstation and its MAC (Ethernet board) address. This is the important “once-only” command. The next two commands are to verify that the ws_name is stored in the environment variable HOSTNAME.

In the AUTOEXEC.BAT, or any batch file executed after login and before running SOCKETS, put the following commands:

```
x:  
CD \SOCKETS\DOS  
SETHOST
```

In SOCKETS, set the IP address in SOCKET.CFG as follows:

```
ip address \HOSTNAME\
```

The IP addresses are now linked to the MAC addresses of the network cards. If you change a network board or swap a PC, must update the MACHOST.MF file with

```
sethost /ws_name
```

The MACHOST.MF file keeps the mapping from the MAC addresses to the workstation names and the HOSTS is used to map the workstation name to its IP address.

The variable name HOSTNAME and filename MACHOST.MF are the defaults for SETHOST.EXE but they can be user specified. Execute sethost /? to see the available options. See also “.

**SYSTEM TIMER INTERRUPT USE**

The timer interrupt (INT 1CH) and the hardware interrupt vectors specified in the interface command(s) or found automatically in Packet Drivers by SOCKETS, are hooked. Whenever such an interrupt occurs or when the API is called, the SOCKETS scheduler is called. The scheduler looks for queued incoming packets (queued at interrupt time by the Packet Driver or serial port driver) as well as the timer queue for timeouts, such as not receiving a TCP acknowledgment in time.

Packets are generally sent when a packet is received or the API is invoked, unless an ARP resolution is in progress, the TCP connection is not yet established, the Nagle heuristic is in operation or the offered window does not allow it. UDP packets are generally sent immediately when received by the API unless an ARP resolution is in progress.

Whenever SOCKETS is executing, it sets a BUSY flag and when the API is called during this time, which is only possible if it is called from an interrupt service routine, the API call fails with ERR_RE_ENTRY. Testing the BUSY flag before executing an API call can circumvent this error. The address of the BUSY flag can be obtained by the GET_BUSY_FLAG low-level API call.

The timer interrupt is assumed to be the standard PC timer interrupt with a period of approximately 55 ms. The exact period of the timer interrupt at INT 1CH is not important, but modifying the period
will affect the scheduling of the kernel. Too long a period may cause erratic behaviour and too short a period may cause excessive overhead. The important value for timing purposes is the BIOS timer variable at absolute location 046CH. The DWORD at 046CH MUST increment at a rate of 65536 increments per hour. (About 18 per second).

ADVANCED NETWORK CONFIGURATION

SOCKETS offers additional networking capabilities for large networks. Using the Routing Information Protocol (RIP), SOCKETS can be configured to be aware of multiple IP routers/gateways.

BOOTP servers are detected when SOCKETS is started without specifying an IP address or an address of 0.0.0.0. To trace the BOOTP negotiations, use a trace all iodt command in your .CFG file before defining any interfaces.

DHCP servers are detected and used when SOCKETS is started with an IP address of 0.0.0.1.

For file server linked networks the SOCKETS utility SETHOST can be used to centralize management of IP addresses. SETHOST maintains a file mapping of the Ethernet (MAC) address of each machine to an IP address.

SOCKETS’ Alternative Routing feature allows more than one route to be specified to a particular host or network. Failure of one route causes an automatic switch to the next route. The failed route is tested periodically and used again when it becomes available.

TUNING TCP/IP

Tuning a computer is a trade-off between the speed of operation and the amount of memory it uses. Performance depends on the number of TCP connections for the computer; more sessions require more memory.

TCP Retry Strategy

If there is a delay in your network connections, SOCKETS employs an intelligent retry strategy. A retry is attempted after a retry interval that gets longer as a function of the number of the retry. The first retry is attempted after the current RTT (Round Trip Time) plus one PC clock tick (18 milliseconds) has elapsed without a response. SOCKETS calculates the RTT as a smooth average of past measured RTTs, starting with the IRTT on a new connection.

For the first five intervals, the time doubles, giving interval lengths of 2, 4, 8, 16 and 25 times the RTT. Then, the square of the interval number is used starting with 5-squared, giving 25 times RTT. Consequently, increasing the number of retries can cause the total elapsed time to become quite long. More than 255 retries results in an infinite number of retries, causing connections to never time out.

When a SOCKETS station starts up, it sends a broadcast ARP request with its IP address to check for duplicate IP addresses. Any SOCKETS client in a retry mode picks up this ARP. Then it retries immediately without waiting for the next scheduled retry time.
To get the current RTT in use for a connection n, use the tcp status n command that gives the smoothed average RTT indicated by SRTT.

Keep-alive
All SOCKETS servers test established connections after a minute of no-traffic by sending an empty packet with a decreased sequence number and waiting for the acknowledgement. If the server does not receive an acknowledgement it retries using the retry strategy described above with the smoothed Round Trip Time (RTT). When all the retries have failed, the connection has timed out and the server resets that connection. This prevents servers from hanging in a connected state when the remote client has stopped responding.

TROUBLESHOOTING

Problems with LICENSE.DAT File
For SOCKETS demo only (socketmd.exe and socketpd.exe).

The SOCKET environment variable is required in the demonstration version to indicate the directory with the LICENSE.DAT file, which contains the license information.

The LICENSE.DAT file contains your license information in demonstration versions. When this error message appears, the most frequent problem is that the file is in the wrong directory.

SOCKETS looks in the SOCKETS directory (as given in the SOCKET environment variable) for the LICENSE.DAT file and then (to accommodate simple new installations) in the \SOCKETS directory of the current drive. If the file contents have been damaged, it will not run. The information in the LICENSE.DAT file is displayed when you start. The number next to SL indicates the number of concurrent users allowed.

XPING
The XPING utility is the most basic test to see if connections are working. Always first try to ping a host that does not seem to respond. Failure to get a response to a ping can usually be traced to one of the following reasons:

- Network cable not plugged in.
- Incomplete bind of the drivers on the local machine. (Carefully check all of the diagnostic messages while booting).
- Inadequate routing information. It may also mean an invalid return route somewhere.
• Lost packages along the route. Sending many ping requests and get only some back; could relate to network hardware problems.

• Remote host is not responding (not switched on, software not loaded, not connected, and so on).

A ping response displays the time taken to get a response. Note that the timing depends on the clock ticks. In an 80x86 PC these ticks are approximately 55ms, so limit the accuracy of the reported times to 55ms.

UTILITY PROGRAMS
The utilities described in the following sections will help you to configure and test your SOCKETS installation.

PDTEST, Packet Driver Test Utility
The PDTEST utility is a diagnostic program that tests loaded packet drivers. See PDTEST in "Chapter 6, ROM-DOS Utility Descriptions" for more information.

SETHOST, IP Address Maintenance Utility
The SETHOST utility sets an environment variable (default HOSTNAME) to the name contained in a map file (default MACHOST.MF) according to the hardware address (MAC or Ethernet) found by searching for a packet driver tsr. See SETHOST in "Chapter 6, ROM-DOS Utility Descriptions" for more information.

IPSTAT, IP and Memory Statistics Utility
The IPSTAT utility returns statistics on IP and memory. See IPSTAT in "Chapter 6, ROM-DOS Utility Descriptions" for more information.
Address Mask (also referred to as NetMask)

A bit mask used to select bits from an IP address for subnet addressing. The mask is 32 bits long, and selects the network portion of the IP address and one or more bits of the local portion.

ANSI (American National Standards Institute)

A group that defines U.S. standards for the information processing industry. ANSI participates in defining network protocol standards.

API (Application Program Interface)

An API is a specification of the methods an application programmer can use to access services provided by a software module. In the case of a network, the API specifies the interface to the network software. In TCP/IP, the idea of a “Socket” as the endpoint of a connection is used. A “socket” then refers to an abstraction to define the endpoint of a connection as far as the API is concerned. A socket can be created, opened, read, written, closed, and deleted in much the same way a file is handled in DOS. The difference is that two sockets must exist, normally on two hosts, before a connection can be made. A read operation on one side must always have a matching write operation on the other side.

A common way of interfacing a terminal emulator to networking software in a PC is to use Interrupt 14h. This is the PC BIOS entry point for serial port support, but when used for networking purposes, the original entry point is reused to provide a similar, but much expanded function. In addition to the native character at a time transfer, block transfers are also offered to increase throughput.

ARP (Address Resolution Protocol)

The TCP/IP protocol used to dynamically bind a high-level IP Address to a low-level physical hardware address. ARP is used across a single physical network and is limited to networks that support hardware broadcast.

Baud

Literally, the number of times per second the signal can change on a transmission line. Commonly, the transmission line uses only two signal states making the baud rate equal to the number of bits per second that can be transferred. The underlying transmission technique may use some of the bandwidth, so it may not be the case that users experience data transfers at the line’s specified bit rate.

BIOS

Basic Input Output System – software that interfaces directly with the hardware.

BIOS extension

A short program that the BIOS recognizes and executes as the BIOS initializes the system.

Boot

Booting is restarting and reloading DOS. A PC can be booted by turning it off and then turning it on or by pressing the Ctrl, Alt, and Del keys simultaneously.
Bootable disk
A system disk that contains the files necessary to start and run the computer.

BOOTP (Bootstrap Protocol)
A protocol a host uses to obtain startup information, including its IP address, from a server.

Broadcast
A packet delivery system that delivers a copy of a given packet to all hosts that attach to it is said to broadcast the packet. Broadcast may be implemented with hardware or software.

Built-in device
Built-in device is an input/output device which is part of the DOS kernel.

CSLIP (Compressed Serial Line Internet Protocol)
CSLIP is an enhancement of SLIP by implementing Van Jacobson header compression. CSLIP uses more memory than SLIP but provides better throughput and faster response times, especially on small packets.

Datagram
The basic unit of information passed across a TCP/IP connection. An IP datagram is to an Internet as a hardware packet is to a physical network. It contains a source and destination address along with data.

DHCP (Dynamic Host Configuration Protocol)
A protocol that a host uses to obtain all necessary configuration information including IP address.

DNS (Domain Name Server)
The on-line distributed database system used to map human-readable machine names into IP addresses. DNS servers throughout the connected Internet implement a hierarchical namespace that allows sites freedom in assigning machine names and addresses. DNS also supports separate mappings between main destinations and IP addresses.

Domain
A part of the DNS naming hierarchy. Syntactically, a domain name consists of a sequence of names separated by periods.

DOS
Disk Operating System – an operating system that relies on disks for file storage.

DOS kernel
The DOS kernel is the part of DOS that handles a standard DOS call (Int 21h). It handles opening, reading/writing of files, loads programs, and manages memory.

FAT
File Allocation Table – a data table which allows DOS to keep track of file location on the disk so that they can be accessed by programs running on DOS.
**Flow control**
Control of the rate at which hosts or routers inject packets into a network or Internet, usually to avoid congestion.

**FTP (File Transfer Protocol)**
The TCP/IP standard, high-level protocol for transferring files from one machine to another. FTP uses TCP.

**Gateway**
Originally, researchers used the term IP gateway for dedicated computers that route packets; vendors have adopted the term IP router. Gateway now refers to an application program that interconnects two services.

**ICMP (Internet Control Message Protocol)**
An integral part of the Internet Protocol that handles error and control messages. Specifically, router and hosts use ICMP to send reports of problems about datagrams back to the original source that sent the datagram. ICMP also includes an echo request/reply used to test whether a destination is reachable and responding.

**IPCP (IP Control Protocol)**
A PPP protocol responsible for configuring the IP protocol parameters on both ends of the point-to-point link.

**IPCPIN**
A PPP protocol monitoring incoming requests responsible for configuring the IP protocol parameters on both ends of the point-to-point link. This was implemented to allow one instance of SOCKETS to act as both a client and server.

**IP (Internet Protocol)**
The TCP/IP standard protocol that defines the IP datagram as the unit of information passed across an Internet and provides the basis for connectionless, best-effort packet delivery service. IP includes the ICMP control and error message protocol as an integral part. The entire protocol suite is often referred to as TCP/IP because TCP and IP are the two fundamental protocols.

**LAN (Local Area Network)**
Any physical network technology designed to span short distances (up to a few thousand meters). Usually, LANs operate at tens of megabits per second through several gigabits per second.

**LCP (Link Control Protocol)**
A PPP protocol responsible for establishing, configuring, and testing the data link connection.

**LCPIN**
A PPP protocol monitoring incoming requests which is responsible for establishing, configuring, and testing the data link connection.
Memory disk
A disk that uses either ROM or RAM for the disk media. The memory disk has a FAT, directories, and file data.

MIME (Multipurpose Internet Mail Extensions)
A standard used to encode data such as images as printable ASCII text for transmission through e-mail.

Modem
A modem (modulator/demodulator) converts digital computer signals into analog signals as used in telephone equipment. The data is sent across the telephone lines and converted back to digital signals by another modem at the destination node.

Using dial-up modems, a remote client can gain access to a network through the telephone line. Remote client users can gain access to the network resources just as if they were physically connected to the LAN. SOCKETS supports the PPP, SLIP and CSLIP protocols.

MSS (maximum segment size)
The largest segment allowed for communicating across a TCP/IP connection.

MTU (maximum transmission unit)
The largest amount of data that can be transferred across a given physical network.

Multicast
A technique that allows copies of a single packet to be passes to a selected subset of all possible destinations.

Nagle Algorithm
This algorithm states that under some circumstances, there will be a waiting period of 200 ms before data is sent over a connection. The following are the specific rules used by the Nagle Algorithm in deciding when to send data:

- If a packet is equal or larger than the segment size (or MTU), and the TCP window is not full, send an MTU size buffer immediately
- If the interface is idle, or the TCP_NODELAY flag is set, and the TCP window is not full, send the buffer immediately.
- If there is less than ½ of the TCP window in outstanding data, send the buffer immediately.
- If sending less than a segment size buffer, and if more than ½ the TCP window is outstanding, and TCP_NODELAY is not set, wait up to 200 msec for more data before sending the buffer.

For more information please see RFC-896, “Congestion Control in IP/TCP”

Nagle Heuristic
See Nagle Algorithm.
Packet
Used loosely to refer to any small block of data sent across a packet switching network.

Packet Driver
Local area network software that divides data into packets for sending on the network, and reassembles the data into its original form when it arrives at its destination.

PCMCIA
Personal Computer Memory Card Interface Association. PCMCIA is a group that defined the standard for a credit-card size card that may act as a memory RAMDISK, ROMDISK, or FLASHDISK. These cards are commonly referred to as PC cards.

POST
Power On Self Test – a test performed by the BIOS that checks the computer hardware for problems before fully initializing the computer.

PPP (Point-to-Point Protocol)
A protocol for framing IP when sending across a serial line.

RAM disk
A disk drive that uses RAM for the media in place of the usual rotating disk drive.

RFC (Request for Comment)
The name of a series of notes that contain surveys, measurements, ideas, techniques, and observations, as well as proposed and accepted TCP/IP protocol standards.

RIP (Routing Information Protocol)
A protocol used to propagate routing information inside an autonomous system.

ROM-DOS
DataLight operating system that can be placed in and execute from within a ROM.

ROM disk
A disk drive that uses ROM for the media in place of the usual rotating disk drive.

ROM scan
The scanning of the ROM area for BIOS extensions performed by the BIOS at initialization time.

ROM
Read Only Memory. This is memory that is not changeable once placed in a computer.

Router
A special purpose, dedicated computer that attaches to two or more networks and forwards packets from one to the other.
RTT (round-trip time)
A measure of delay between two hosts.

Shell
The Shell is the command interpreter, usually COMMAND.COM. The shell takes text commands and calls the DOS kernel to implement them.

SLIP (Serial Line Internet Protocol)
A framing protocol used to send IP across a serial line. SLIP is popular when sending IP over dialup phone lines.

SMTP (Simple Mail Transfer Protocol)
The TCP/IP standard protocol for transferring electronic mail messages from one machine to another.

System disk
See Bootable disk.

TCP (Transmission Control Protocol)
The TCP/IP standard transport level protocol that provides the reliable, full duplex, stream service on which many application protocols depend.

TTL (time-to-live)
A technique used in best-effort delivery systems to avoid endlessly looping packets.

UDP (User Datagram Protocol)
The TCP/IP standard protocol that allows an application program on one machine to send a datagram to an application program on another.

WWW (World Wide Web)
The large-scale information service that allows a user to browse information. WWW offers a hypermedia system that can store information as text, graphics, audio, etc.
INDEX

% symbol
  how to use in a batch file, 18

; sign
  using in a batch file, 34
  using in CONFIG.SYS, 34

? symbol
  using to pause CONFIG.SYS commands, 33
  using to suppress command echo, 34

A20 line control
  handling by HIMEM.SYS, 94

Address Resolution Cache, 150

Advertised routes
  using the rip advertise command to advertise, 164
  using the rip use command to update routes, 164

Alternate routing connections
  setting control with the par command, 141

ANSI.SYS, 73

APM BIOS
  how it fits in the power management scheme, 106

Archive files
  changing with ATTRIB, 75

ARP Command, 150

ARPSTAT utility, 199

ATA.SYS, 74

ATTRIB command
  using to manage file attributes, 75

AUTOEXEC.BAT
  bypassing commands in, 28
  CONFIG environment variable, 28
  troubleshooting commands in, 28
  using to extend menu items, 27

AUTOEXEC.BAT file
  needed with ODI drivers, 240

BACKUP command
  using to save files, 76

Batch file commands
  :., 35
  @, 35
  disabling with a, 35
  conditional execution with IF, 52
  disabling with REM, 63
  displaying batch file commands/messages, 48
  displaying while executing, 48
  executing a secondary batch file, 37
  pausing processing of with PAUSE, 61
  repeating the execution of with FOR, 51
  SHIFT, 66
  shifting parameters, 66
  transferring control with GOTO, 51

Batch file execution
  from within another batch file, 37

Batch file messages
  creating with REM, 63

Batch files
  bypassing commands in, 38
  clearing the display, 38
  commands that can be used in, 20
  disabling the user prompt, 77
  how to create and use, 19
  parameters, 20
  preventing echo with @, 35
  selecting available user options, 78
  specifying allowable input keys, 78
  comment text using, 35, 66

Baud rate for a serial link
  setting with the par command, 161

BIOS
  needed for advanced power management, 106

Boot time problems
  stepping through AUTOEXEC.BAT, 28
  stepping through CONFIG.SYS, 28

Bootable disk
  how to create using SYS, 115

BREAK command
  using to expand Ctrl+C operations, 36

Buffers, 183

BUFFERS command
  using to set buffer memory, 36

Bypassing AUTOEXEC.BAT commands
  how to, 28

Bypassing CONFIG.SYS commands
  how to, 28

Cache
  using SMARTDRV to enhance disk speed, 112

CALL command
  using to start a batch file, 37

CAPDIAG Utility, 199

CD command
  using to switch directories, 37

CD-ROM drives
  using MSCDEX.EXE to support, 101

Changing the command line
  using specific keys, 17

CHDIR command
  using to switch directories, 37
INDEX

CHKDSK command
using to fix disk errors, 77

CHOICE command
using to set time delay, 78
using to set users prompt/keys, 78

Clock settings
changing with the TIME command, 69

CLS command
using to clear the display, 38

Code page
loading different with DISPLAY.SYS, 29, 86, 87
Code page numbers
for different countries, 29

Code pages
using for different keyboards, 29

Colors
setting text and background with MENUCOLOR, 55

COM port speed/flow control
setting with the par command, 160

COM ports
changing configuration with MODE, 98

Combined HTTP and FTP Server, 238

COMMAND command
using to start the command processor, 82

Command interpreter
adding/removing commands, 123
loading other than COMMAND.COM, 66
using small version with ROM-DOS, 122

Command line
batch file parameters, 20
how to edit the contents of, 17

Command line functions
adding/removing from the command interpreter, 123

Command line prompt
using PROMPT to configure, 61

Command processing
bypassing commands with ?, 33

Command processor
description of, 82
how to start, 82

COMMAND.COM
loading a different command interpreter than, 63
transferring to floppy disk, 15
Commands summary of, 21

CONFIG.SYS
bypassing commands in, 28
CONFIG environment variable, 27
setting the processing level, 25
settings for multiple users, 25
troubleshooting commands in, 28
using the COUNTRY command, 41
CONFIG.SYS commands, 35
changing how CONFIG.SYS executes, 68
disabling with a, 35
configuring ROM-DOS for international use, 41
dynamic use of data stacks, 67
executing from a secondary file, 58
install programs to high memory, 55
loading ROM-DOS in high memory, 47
loading TSR programs, 54
pausing during command processing with ?, 33
setting number of open files, 50
setting the display colors, 56
setting the highest driver letter, 54
setting the number of data buffers, 36
specifying the number of FCBs, 50
the processing of, 25
use of the SUBMENU command, 67
using INCLUDE for multiple-user setups, 53
using to install device drivers, 44, 45, 55
using to set for multiple users, 56, 57
using to set the command interpreter, 66
using to set the NumLock key, 60

CONFIG.SYS comment text using, 35

Configuration block
using for multiple configurations, 25

Configuration blocks
example of using in CONFIG.SYS, 25
including for multiple configurations, 53
using for multiple configurations, 53

Configuring ROM-DOS
for multiple users, 25, 56, 57
with CONFIG.SYS, 25

Connection maintenance
how servers determine client non-response, 247
using XPING to verify working connections, 247

Connection retry interval
setting the Round Trip Time (RTT), 246

Connection termination
setting the retry count with tcp retry, 171

Connection timeouts
how servers determine client non-response, 247
how Sockets adjusts the retry attempts, 246

COPY command
using to create new disk files, 39

COUNTRY command
using in CONFIG.SYS, 42
using to select character set, 29, 30, 42

COUNTRY identifier
for Belgium, 29
for Canadian-French, 29
for Czech Republic, 29
for Denmark, 29
for France, 29
Directories and subdirectories
- changing from one to another, 15
- creating with MKDIR, 58
- definition of the current directory, 15
- deleting with MKDIR, 64
- naming conventions, 15

Directories/subdirectories
- using TREE to list, 70, 118
- using XCOPY to create, 121

DIRECTORIES environment variable
- using to set DIR command preferences, 45

Disk
- checking with CHKDSK, 77
- creating a volume label, 97
- creating multiple with FDISK, 90
- displaying a volume label, 72
- formatting floppy with FORMAT, 93
- formatting/initializing with FDISK, 90
- setting the maximum number of, 54

Disk drives
- used in computers
  - description of, 8
  - naming conventions, 15
- using SUBST to map drive letters, 115

Disk files
- checking free disk space, 77
- definition of, 13
- displaying directory contents, 44, 70, 118
- displaying text using MORE, 99
- displaying the contents of, 70
- locating text within using FIND, 92
- locking by loading SHARE.EXE, 111
- making hidden with ATTRIB, 75
- making read-only with ATTRIB, 75
- managing with directories, 14
- recovering with RESTORE, 108
- saving with BACKUP, 76
- sharing by loading SHARE.EXE, 111
- sorting text within using SORT, 113
- specifying the number of open with FILES, 50
- using COPY to create, 39
- using DEL to delete, 43
- using DELTREE to delete directories, 83
- using DISKCOPY to create, 85, 86
- using ERASE to delete, 49
- using in place of keyboard input, 18
- using MOVE to relocate or rename, 100
- using REN to rename, 64
- using to receive system output, 19
INDEX

using XCOPY to create, 121
Disk partitions
creating with FDISK, 90, 92
Disk speed
using SMARTDRV to enhance disk speed, 112
DISKCOMP
comparing diskettes with, 85
DISKCOPY command
using to create new disk files, 85, 86
Display colors
setting text and background, 56
DISPLAY.SYS
using to display a different code page, 30
using to display code pages, 86, 87
using to display international characters, 86, 87
Displaying file lists
using DIR to list, 45
Displaying subdirectories
using TREE to list, 70, 118
Domain Command, 152
Domain Name Server, 152
DOS command
using to load ROM-DOS, 47
DOS references, 3
Driver (network) specifications
a descriptions of, 239
Drivers
ODI driver and AUTOEXEC.BAT file, 240
ODI driver and NET.CFG file, 240
ODI driver installation, 240
ODI driver ODIPKT.COM, 241
packet driver installation, 239
Duplicating directories/files
using XCOPY to create new, 121
Duplicating files
using COPY to create new, 40
using DISKCOPY to create new, 85, 86
Echo batch file commands
preventing with @, 34
ECHO command
using to display batch file messages/commands, 48
EGA.CPI, 86
EMM386.EXE
using to support expanded memory, 88
Environment, 61
Environment variables
CONFIG.SYS, 27
DIRCMD, 45
DIRSIZE, 45
PATH, 60
setting PATH and PROMPT, 29
setting with SET, 65
Environment Variables, 140
ERASE command
using to erase disk files, 49
Erasing files
using DEL to delete, 44
using DELTREE to delete directories, 83
using ERASE to delete, 49
Euro, 29
EXE2BIN
using to creat .COM files, 90
EXIT command
using to terminate a nested ROM-DOS session, 49
Expanded memory
using EMM386.EXE to support, 88
Extended memory
using HIMEM.SYS to support, 93
Extension CGI, 229
F5 key
using to bypass CONFIG/AUTOEXEC commands, 28
F8 key
using to step AUTOEXEC.BAT commands, 28
using to step CONFIG.SYS commands, 28
FCBS command
specifying number of file control blocks, 50
FDISK command
using to format a hard disk, 90
File control blocks
specifying the number of with FCBS, 50
File names
how to create using wildcard characters, 16
File server
using SETHOST.EXE to manage files on, 244
Files
displaying the contents of, 70
fixing errors with CHKDSK, 77
locating text within using FIND, 91
locking by loading SHARE.EXE, 111
making hidden with ATTRIB, 75
making read-only with ATTRIB, 75
naming conventions explained, 13
recovering with RESTORE, 108
saving with BACKUP, 75
sharing by loading SHARE.EXE, 111
sorting text within using SORT, 113
using COPY to create, 39
using DEL to delete, 43
using DELTREE to delete, 83
using DIR to list directory contents, 45
using DISKCOPY to create entire disk, 85, 86
using ERASE to delete, 49
using in place of keyboard input to the system, 19
using MOVE to relocate or rename, 100
INDEX

using REN to change the name of, 64
using to receive display/printer data, 19
using XCOPY to create, 121
FILES command
specifying number of file control blocks, 50
FIND command
using to search within disk files, 92
Finnish keyboard layout, 131
Flash memory
using as a disk drive, 3
flow control, 177
Flow control for a serial link
setting with the par command, 161
FOR command
using to repeat batch file commands, 51
FORMAT command
using to format a floppy disk, 93
French keyboard layout, 131
French-Canadian keyboard layout, 131
FTP
commands, 236
FTP Client, 201
FTP Combined Server, 236
FTP Server, 235
FTPAPI Client/Server, 204
FTP, 235
gateway application example, 194
Gateways
using RIP to set up multiple, 246
German keyboard layout, 131
GETMAIL Application, 204
GOTO command
using to transfer control in batch files, 51
HELP command
using to obtain ROM-DOS help, 52
Help information on ROM-DOS
displaying with the HELP command, 52
Hidden files
making so with ATTRIB, 75
High memory
using EMM386.EXE to support, 88
using HIMEM.SYS to support, 94
High memory area (HMA)
using the DOS command to load ROM-DOS, 47
HIMEM.SYS
using to support extended memory, 94
Host names
using SETHOST.EXE to manage on a file server, 244
Hosts, 145
Htaccess, 235
HTTP Combined Server, 238
HTTP Server, 227
HTTPD Program, 231
HTTPGET, 205, 206
IF command
conditional execution of batch file commands, 52
iface command
using to define a PPP interface, 158
Iface Command, 153
IFSTAT Utility, 207
INCLUDE command
using to include configuration blocks, 53
INSTALL command
using to load TSR programs, 54
Installing drivers
ODI driver and the AUTOEXEC.BAT file, 240, 241
ODI driver and the NET.CFG file, 240
ODI driver on the target system, 240
packet driver on the target system, 239
Installing Sockets, 139
Interface Command, 153
International use
configuring ROM-DOS for, 29, 42, 95
IOCTL Utility, 208
IOCTLH Utility, 209
IP Address Maintenance Utility, 248
IP and Memory Statistics Utility, 248
IP Command, 156
IP Routing Table, 165
Ipstat, 248
IPSTAT Utility, 209
IRTT (Initial Round Trip Time)
setting with the tcp –rtt command, 171
Italian keyboard layout, 132
KEYB
using to select a different keyboard/country, 31
KEYB command
using to alter keyboard layouts, 95
Keyboard input
how to redirect from a file, 18
Keyboard layout
Canada, 130
Denmark, 130
Finland, 131
France, 131
Germany, 131
Italy, 132
loading different countries with KEYB, 31
Norway, 132
setting for different countries, 31
Spain, 132
Sweden, 133
United Kingdom, 133
United States, 133
Keyboard layouts
altering with KEYB, 95
KEYBOARD.SYS, 96
Keyboards code pages
using for different keyboards, 29
KEYBRD2.SYS, 96
LABEL command
using to create/delete a disk label, 97
LASTDRIVE command
using to set the highest drive letter, 54
LFN, 13
License.dat file, 247
LOADHIGH command
using to install programs in high memory, 55
Loading programs
installing in high memory, 55
Local and remote options
setting for point-to-point protocol, 158
Local port starting number
setting with the tcp lport command, 172
Long Filenames, 13
Wildcards, 15
LPD, Starting, 167
LPR Printer Client, 210
Mail Retrieval application, 204
Mail Message Creation, 211
Mail Sending Utility, 219
MAKEMAIL utility, 211
Maximum Segment Size, 183
Maximum Transmission Unit, 182
MD command
using to create a new directory/subdirectory, 58
mdd
(Multi Destination Drivers), 180
MEM command
using to display the amount of memory, 97
Memory configuration
using expanded memory, 88
Memory disk
creating a RAM disk with VDISK, 120
Memory manager
using with ODI drivers, 241
using with packet drivers, 240
Menu blocks
using for multiple configurations, 26, 53
Menu configuration block
using to define menu items, 67
Menucolor
example of using in CONFIG.SYS, 26
MENUCOLOR command
using to set text/background colors, 56
Menudefault
example of using in CONFIG.SYS, 26
MENUDEFAULT command
using to set default menu, 56
Menuitem
example of using in CONFIG.SYS, 26
MENITEM command
using to specify startup menu, 57
Mini-command interpreter
how to configure, 123
using with ROM-DOS, 122
MKDIR command
using to create a new directory/subdirectory, 58
MODE command
using to change a COM port/printer/display, 98
modem
examples, 194
pool, 180
Modem Configuration Examples, 176
Modem Configuration File, 175
Modem Operation, 144
Modem Retry Strategy, 180
MORE command
using when displaying text files, 99
MOVE command
using to move/rename new files/directories, 100
Moving files
using MOVE to relocate/rename, 100
MSCDEX.EXE
using to support CD-ROM drives, 101
MSS, 183, 252
mss: max segment size, 145
MTU, 135, 183,
Multi Destination Drivers, 180
Naming conventions
for batch files, 19
for disk drives, 15
for disk files, 11
using wildcard characters, 14
Naming files
using REN to rename existing, 64
NEDREMT.EXE editor program
using to remotely create/alter text files, 103
NET.CFG file
needed with ODI drivers, 240
NETBIOS Utility, 213
Network Management, 244
NEWFILE command
using to switch from CONFIG.SYS, 58
Norwegian keyboard layout, 132
NUMLOCK command
using to set the NumLock key, 60
ODI drivers
entries in AUTOEXEC.BAT, 240
file names of typical ODI drivers, 240
how to install on the target system, 240
need a ODI.CFG file, 240
running ODIPKT.COM, 241
using a memory manager with, 241
Open Data-Link Interface (ODI)
a description of, 239
Packet Driver, 143
NDIS, 241
Packet driver test utility, 248
Packet drivers
 a description of, 239
 file names of typical packet drivers, 239
 how to install on the target system, 239
 using a memory manager with, 240
par command
 using to define PPP local/remote options, 158
 using to define PPP retry counters, 159
 using to define PPP timeout values, 160
 using to define PPP username/password, 160
Par Command, 158
Parameters
 for batch files, 20
Partitions for disk drives
 creating with FDISK, 90
PATH Command
 using to set the environment variable, 60
PAUSE command
 halting execution of batch file commands, 61
Pdtest, 248
PDTEST Utility, 215
PING, XPING, XPING
Point-to-Point Protocol (PPP)
 configuration of the interface, 158
Point-to-point protocol option
 open a specified layer, 160
 setting local and remote LCP/IPCP, 154
 setting retry counters, 155
 setting timeout values, 156
 setting username/password, 156
Point-to-Point Protocol parameters/options
 using with Sockets for DOS, 154
Port speed and flow control
 setting with the par command, 157
POWER.EXE
 using to implement power management, 106
PPP Functionality, 144
Print Server, Starting, 167
Printer Client, LPR, 210
Printer Command, 163
Printer ports
 changing configuration with MODE, 98
Printer Redirector, 163
Processing AUTOEXEC.BAT commands
 how to, 28
Processing CONFIG.SYS commands
 how to, 28
Programming
 through the use of batch files, 19
PROMPT command
 using to alter the command line prompt, 62
PROTO
 creating C language function prototypes with, 107
RAM disk
 creating a memory-based disk, 120
Random Access Memory (RAM)
 definition of, 2
RC_JAR Remote Console Utility, 217
RC_DK_JAR Remote Console for Danish Keyboards, 232
RCCLI Client, 217
RD command
 using to delete a directory/subdirectory, 64
Read Only Memory (ROM)
 definition of, 2
Redirecting input/output
 using the CTTY command to, 42
REM command
 inserting comments in batch files, 63
Remote Console Client, 228
Remote Console Client, RCCLI, 217
Remote Console for Danish Keyboards, 217
Remote Console Utility, 216
REN command
 using to rename disk files, 64
RESTORE command
 using to recover saved files, 108
Retry counter options
 setting for point-to-point protocol, 158
 setting with the tcp retry command, 171
Retry Strategy, Modems, 180
rip advertise command
 setting/disabling with the par command, 158
 using to advertise routes, 160
RIP Command, 164
rip use command
 using to create rip requests for route updates, 164
RMDIR command
 using to delete a directory/subdirectory, 64
RMDIR/RD command
INDEX

using to delete an empty directory, 64
ROM disk
creating with ROMDISK.EXE, 7
ROM-DOS
changing the displayed version number, 71
configuring for international use, 29, 42, 95
configuring for multiple users, 25, 56, 57
configuring with CONFIGSYS, 25
displaying the version number of, 71
transferring system files, 115
using a command interpreter with, 122
ROM-DOS memory contents
displaying with MEM, 97
Round Trip Time (RTT)
how Sockets adjusts the retry attempts, 246
Route Command, 165
Routing control for alternate connections
setting with the par command, 160
Routing Information Protocol (RIP), 164
using to setup multiple IP routers/gateways, 246
Routing Table, entry, 165
RTT (Round Trip Time) for a connection
replacing the auto-set RTT value, 171
Sconfig Utility, 146
Searching files
using FIND to examine contents, 92
Segment size (maximum)
setting with the tcp mss command, 172
Send segment size (maximum)
setting with the tcp smss command, 173
SENDMAIL Utility, 219
Server connections
how servers determine client non-response, 247
using XPING to verify working
connections, 247
SET command
using to set, remove, display environment variable, 65
Sethost, 248
SETHOST Utility, 219
SETHOSTEXE program
managing host names on a file server, 244
SHARE command
using to allow file sharing, 111
SHELL command
using to start a command interpreter, 65
SHIF T command
using to shift batch file parameters, 66
SMARTDRV command
using to enhance disk speed, 112
SNTPCLI Utility, 221
SOCKDIAG Utility, 222
Socket.cfg, 145
Socket.UPW, 233
SocketM, 147
Command Line Options, 174
SocketP, 147
Command Line Options, 174
Sockets
Combined HTTP and FTP Server, 238
Configuration, 141, 143
Configuration Files, 141
Custom Applications, 142
Environment Variables, 140
Extension CGI, 229
file selection, 141
FTP Server, 235
FTPD, 235
Htaccess, 230
HTTP Server, 238
HTTPD program, 238
Installing, 139
Modem Operation, 144
Packet Driver, 143
Password Permissions file, 233
PPP Functionality, 144
Remote Console Client, 228
Sconfig, 146
Serial Operation, 143
Socket.UPW, 233
Test Programs, 248
troubleshooting, 247
Utilities, 248
Sockets Configuration, 150
Sockets Configuration Examples, 184
Dial-up SLIP Connection with Sockets as an IP Router, 195
Direct Serial Connection with Sockets as a Client, 190
Direct Serial Connection with Sockets as a Server, 187
Multiple Dial-in Connections, 196
Single Dial-in Connection, 186
Single Dial-in Connection with ASY Interface, 187
Sockets Dial-up to ISP, 185
Sockets Machine Using Call Back Verification, 191
Sockets Machine with CBV and Logging-in, 193
Sockets Serving as a Web Page, 184
Sockets Diagnostic Utilities
PDTEST, 215
SOCKDIAG, 217, 222
Sockets Diagnostic Utility
CAPIDIAG, 199
DHCPSSTAT, 200
IFSTAT, 207
IOCTL, 208
IOCTLH, 209
IPSTAT, 209
Sockets Examples, 184
Sockets Printer Redirection, 163
Sockets system requirements, 137
SORT command
   using to sort contents of text files, 113
Spanish keyboard layout, 132
STACKDEV.SYS, 114
STACKS command
   using to set dynamic data stacks, 67
Start Command, 167
Start LPD, 167
Start Print Server, 167
SUBMENU commands
   using to define menu items, 67
SUBST command
   using to substitute disk drive letters, 115
SUPERBOOT
   creating partitions with FDISK, 90
Swedish keyboard layout, 133
SWITCHES command
   using to control CONFIG.SYS execution, 69
SYS (System) command
   using to copy system files to a disk, 115
System files
   changing with ATTRIB, 75
System prompt
   definition and contents of, 17
tcp
   window, 145
TCP Command, 170
tcp irtt command
   using to set IRTT (Initial Round Trip Time), 172
tcp lport command
   using to set local port starting number, 172
tcp mss command
   using to set maximum segment size, 173
TCP Operating Parameters, 170
tcp retry command
   using to set/display the retry count, 172
tcp rtt command
   using to replace the RTT (Round Trip Time), 173
tcp smss command
   using to set maximum send segment size, 173
tcp timemax command
   using to set the maximum tcp timeout, 173
tcp timeout
   setting the maximum with the tcp timemax command, 173
TCP Utility, 225
tcp window command
   using to set the maximum receive window size, 173
TCP, Changing Parameters, 225
Terminate and stay resident programs
   loading with INSTALL, 54
Terminating a TCP connection
   using tcp retry to set retry count, 172
TIME command
   using to set the internal clock, 69
Time settings
   changing with the TIME command, 69
Timeout values
   setting for point-to-point protocol, 160
Timer Interrupt, 245
TREE command
   using to list directories/subdirectories, 70, 118
Troubleshooting
   using XPING to verify working connections, 247
TSR, 54
TSR programs
   loading with INSTALL, 54
TYPE command
   using to display the contents of text files, 70
U.K. keyboard layout, 133
U.S. keyboard layout, 133
UMBLINK
   gaining upper memory blocks with, 118
UMBLINK.EXE, 118
   Username/password
   setting for point-to-point protocol, 160
VDISK.SYS
   using to create a RAM disk, 120
VER command
   using to display ROM-DOS version number, 71
VERIFY
   changing the verify state, 71
VOL command
   using to display a disk label, 72
Volume labels
   how to create and delete, 97
   how to display, 72
Wildcard characters
   using to create file names, 16
Window size (receive)
   setting the maximum with the tcp window command, 173
window size: TCP, 145
XCOPY command
   using to create new directories/files, 121
XPING
   using to verify working connections, 228
XPING Utility, 216, 226