Networks • Communications

DECnet–DOS

Supplemental Information

AA–JH72A–TV

digital

DECnet–DOS

Supplemental Information

Order No. AA–JH72A–TV

March 1987

This manual describes changes in updating DECnet–DOS and DECnet–Rainbow V1.0 and V1.1 and DECnet–VAXmate V1.0 to DECnet–DOS, DECnet–Rainbow, and DECnet–VAXmate V1.2. Changes for V1.2 include bug fixes, enhancements, and newly supported features.

Supersession/Update Information:	This is a new manual.
Operating System and Version:	MS–DOS V2.11 PC DOS V2.10 PC DOS V3.10 PC DOS V3.20 VAXmate MS–DOS V3.10
Software Version:	DECnet–DOS V1.2 DECnet–Rainbow V1.2 DECnet–VAXmate V1.2

digital

AA-JH72A-TV
First Printing, March 1987

This manual was produced by Networks and Communications Publications.

Contents

Preface

1 DECnet–DOS Changes

1.1 Overview 1–1
1.2 Highlight of Changes from Previous Versions of DECnet–DOS 1–1
1.3 Changes to the DECnet–DOS Installation Procedure (DIP) 1–2
1.3.1 Newly Supported Features 1–2
1.3.2 Enhancements 1–2
1.4 Changes to DECnet–DOS SETHOST 1–3
1.4.1 Bug Fixes 1–3
1.4.2 Newly Supported Features 1–4
1.4.2.1 Using 132–Column Support 1–4
1.4.2.2 Using National and Multinational Character Sets 1–5
1.4.3 New Switches 1–5
1.4.3.1 /KEYBOARD 1–5
1.4.3.2 /SERVICE 1–6
1.4.3.3 /SETUP 1–6
1.4.3.4 /SWITCH = *n* 1–7
1.4.4 Enhancements 1–8
1.5 Changes to the DECnet–DOS Mail Utility 1–11
1.6 Changes to the DECnet–DOS Network Device Utility (NDU) 1–11
1.6.1 Bug Fixes 1–11
1.7 Changes to DECnet–DOS Transparent File Access (TFA) 1–13
1.7.1 Bug Fixes 1–13
1.8 Changes to DECnet–DOS DTR/DTS 1–14
1.9 Changes to the DECnet–DOS Spawner 1–14
1.9.1 Bug Fixes 1–14
1.9.2 Newly Supported Features 1–14
1.9.3 Enhancements 1–17
1.10 Changes to the DECnet–DOS File Access Listener (FAL) 1–17
1.11 Changes to the DECnet–DOS Network File Transfer Utility (NFT) 1–17
1.11.1 Enhancements 1–17

1.12 Changes to the DECnet–DOS Programming Interface Library (DNETLIB) 1–18
1.12.1 Bug Fixes 1–18
1.12.2 Enhancements 1–18
1.12.3 Compiling DNETLIB Sources 1–18
1.12.4 Newly Supported Features 1–19
1.13 Changes to the DECnet–DOS Data Link Layer (DLL) 1–19
1.13.1 Enhancements 1–19
1.13.1.1 MICOM Driver 1–19
1.13.1.2 3COM Driver 1–19
1.13.1.3 DEPCA Driver 1–19
1.14 Changes to DECnet–DOS DECnet Network Process (DNP) 1–20
1.14.1 Bug Fixes 1–20
1.14.2 Newly Supported Features 1–20
1.14.3 Enhancements 1–20
1.15 Changes to the DECnet–DOS Scheduler 1–20

2 DECnet-Rainbow Changes

2.1 Overview 2–1
2.2 Highlight of Changes from Previous Versions of DECnet–Rainbow 2–1
2.3 Changes to the DECnet–Rainbow Installation Procedure (DIP) 2–1
2.3.1 Enhancements 2–2
2.4 Changes to DECnet–Rainbow SETHOST 2–2
2.4.1 Bug Fixes 2–2
2.4.2 Newly Supported Features 2–2
2.4.3 Enhancements 2–3
2.5 Changes to the DECnet–Rainbow Mail Utility 2–4
2.6 Changes to the DECnet–Rainbow Network Device Utility (NDU) 2–4
2.6.1 Bug Fixes 2–4
2.7 Changes to DECnet–Rainbow Transparent File Access (TFA) 2–6
2.7.1 Bug Fixes 2–6
2.8 Changes to DECnet–Rainbow DTR/DTS 2–7
2.9 Changes to the DECnet–Rainbow Spawner 2–7
2.9.1 Bug Fixes 2–7
2.9.2 Newly Supported Features 2–8
2.9.3 Enhancements 2–10
2.10 Changes to the DECnet–Rainbow File Access Listener (FAL) 2–10
2.11 Changes to the DECnet–Rainbow Network File Transfer Utility (NFT) 2–10
2.11.1 Enhancements 2–10
2.12 Changes to the DECnet–Rainbow Programming Interface Library (DNETLIB) 2–11
2.12.1 Bug Fixes 2–11
2.12.2 Enhancements 2–11
2.12.3 Compiling DNET Library Sources 2–12

2.12.4 Newly Supported Features 2–12
2.13 Changes to DECnet–Rainbow DECnet Network Process (DNP) ... 2–12
2.13.1 Bug Fixes 2–12
2.13.2 Enhancements 2–13

3 DECnet–VAXmate Changes

3.1 Overview 3–1
3.2 Highlight of Changes from Version 1.0 of DECnet–VAXmate 3–1
3.3 Changes to the DECnet–VAXmate Installation Procedure (DIP) 3–2
3.3.1 Enhancements 3–2
3.3.2 Using DIP in a VAXmate/VMS Services Environment 3–3
3.4 Changes to DECnet–VAXmate SETHOST 3–3
3.4.1 Bug Fixes 3–3
3.4.2 Newly Supported Features 3–4
3.4.3 New Switches 3–5
3.4.3.1 /SERVICE 3–5
3.4.3.2 /SETUP 3–5
3.4.3.3 /SWITCH = *n* 3–6
3.4.4 Enhancements 3–6
3.5 Changes to the DECnet–VAXmate Mail Utility 3–9
3.6 Changes to the DECnet–VAXmate Network Device Utility (NDU) 3–9
3.6.1 Bug Fixes 3–9
3.7 Changes to DECnet–VAXmate Transparent File Access (TFA) 3–11
3.7.1 Bug Fixes 3–12
3.8 Changes to DECnet–VAXmate DTR/DTS 3–12
3.9 Changes to the DECnet–VAXmate Spawner 3–12
3.9.1 Enhancements 3–15
3.10 Changes to the DECnet–VAXmate File Access Listener (FAL) 3–15
3.11 Changes to DECnet–VAXmate Network File Transfer (NFT) 3–15
3.11.1 Enhancements 3–15
3.12 Changes to the DECnet–VAXmate Programming Interface Library (DNETLIB) 3–16
3.12.1 Bug Fixes 3–16
3.12.2 Enhancements 3–16
3.12.3 Compiling DNETLIB Sources 3–17
3.12.4 Newly Supported Features 3–17
3.13 Changes to the DECnet–VAXmate DECnet Newtork Process (DNP) 3–17

A Keyboard Illustrations

B Sample C Programs

B.1 Sample Client Program B–1

B.2 Sample Server Program B–5
B.3 Sample Transparent Task-to-Task Program B–10

Figures

A–1 The Rainbow 100 System Keyboard A–2
A–2 The VAXmate (LK250) Keyboard A–3
A–3 The IBM PC and PC/XT Keyboard A–6
A–4 The IBM PC AT Keyboard A–9
A–5 The IBM Enhanced Personal Computer Keyboard A–12

Tables

A–1 Key Mappings (IBM PC/XT Keyboard to DEC VT102-Type Keyboard) A–4
A–2 Keypad Mappings (IBM PC/XT Keypad Keys to DEC VT102-Type Keypad Keys) A–5
A–3 Key Mappings (IBM PC AT Keyboard to DEC VT102-Type Keyboard) A–7
A–4 Keypad Mappings (IBM PC AT Keypad Keys to DEC VT102-Type Keypad Keys) A–8
A–5 Key Mappings (IBM Enhanced PC Keyboard to DEC VT102-Type Keyboard) A–10
A–6 Keypad Mappings (IBM Enhanced PC Keypad Keys to DEC VT102-Type Keypad Keys) A–11

Preface

This document includes updated information for version 1.2 of the DECnet–DOS, DECnet–Rainbow, and DECnet–VAXmate software products. In the context of this document, the terms DECnet–DOS, DECnet–Rainbow, and DECnet–VAXmate are used to refer to the DECnet–DOS software, the DECnet–Rainbow software, and the DECnet–VAXmate software.

This document describes changes that have been made since DECnet–DOS V1.1, DECnet–Rainbow V1.1, and DECnet–VAXmate V1.0 (there is no V1.1 for DECnet–VAXmate). The changes include bug fixes, enhancements, and newly-supported features.

You should use this information in conjunction with the material presented in the DECnet–DOS documentation set. Specifically, you should refer to the *DECnet–DOS Release Notes*. You should also look at the README.TXT file which is included with the other files for this kit.

Intended Audience

This document is intended for users of VAXmate workstations, Rainbow personal computer, and IBM PC, PC/XT, and Personal Computer AT personal computers. Some of the information presented in this document is very technical. If you do not understand the material, you should seek assistance from the person who is responsible for configuring your network.

Structure of This Document

This document consists of 3 chapters and 2 appendixes:

Chapter 1	Describes changes to DECnet–DOS.
Chapter 2	Describes changes to DECnet–Rainbow.
Chapter 3	Describes changes to DECnet–VAXmate.
Appendix A	Contains keyboard illustrations for VAXmate workstations and the Rainbow, IBM PC/XT, and IBM PC AT personal computers. It also includes an illustration of the IBM Enhanced PC keyboard.
Appendix B	Contains sample C programs. These programs are also available as files on the diskettes that accompany this document.

Graphic Conventions Used in This Document

The following graphic conventions are used in this document:

Convention	Meaning
`Monospaced type`	Monospaced type indicates examples of system output or user input. System output is in black; user input is in `red`.
italics	Lowercase italics in commands and examples indicate that either the system supplies or you should supply a value.
key	Indicates that you should press the specified key. CTRL/*x* indicates that you should hold down the CONTROL key while you press the *x* key, where *x* is a letter. Note that unless otherwise specified, you should end every command line by pressing RET.

1 DECnet–DOS Changes

1.1 Overview

This chapter describes changes from DECnet–DOS Version 1.0 and Version 1.1 to DECnet–DOS Version 1.2. (Note that although DECnet–DOS is a set of software products that includes DECnet–DOS, DECnet–Rainbow, and DECnet–VAXmate, this chapter describes the changes that apply specifically to DECnet–DOS for the IBM PC, PC/XT, and Personal Computer AT.)

The following list highlights the DECnet–DOS changes. The changes are then broken down by components in later sections.

1.2 Highlight of Changes from Previous Versions of DECnet–DOS

- Bug fixes.
- Support for the new Enhanced IBM PC/XT and PC AT. (The IBM PC/XT Model 286 is not supported.)
- Support for PC DOS Version 3.20.
- Performance improvements to Network File Transfer (NFT).
- Functional improvements to the Job Spawner and the DECnet Test Receive utility (DTR).
- SETHOST support for DEC national character sets (7-bit) and DEC multinational character sets (8-bit).
- SETHOST local IBM-style print screen support (for IBM PCs).

- SETHOST 132-column support (for IBM PCs).
- Support for Digital's new Ethernet communications board for personal computers, the DEPCA. There is also support for the DEPCA Data Link layer (DLL).
- Changes to the DECnet–DOS Programming Interface Library sources (DNETLIB) that provide enhanced performance for programming applications that use this library.

The following sections describe changes that have been made to the individual components.

1.3 Changes to the DECnet–DOS Installation Procedure (DIP)

The DECnet–DOS Installation Procedure (DIP) is an automated procedure that lets you easily install DECnet–DOS by responding to a set of questions and making selections from a series of menus. Refer to the *DECnet–DOS Installation Guide* for instructions on how to use DIP, while noting the changes and enhancements that are described here.

1.3.1 Newly Supported Features

Version 1.2 of DIP includes support for the following:

- The PC DOS Version 3.20 operating system.
- The new Enhanced IBM PC/XT and Personal Computer AT.
- The DEPCA communications board and the DEPCA Data Link layer.

1.3.2 Enhancements

During installation, DIP now performs the following operations:

- Checks for the proper version of the DIP.DAT/DIP.SAV database and displays an error if it is an incompatible version.
- Provides enhanced performance for copying files from the kit.
- Checks the communication type of the DECnet–DOS parameter file (DECPARM.DAT, if one exists). If the new configuration is for a different communication type, DIP deletes the existing DECPARM.DAT. It creates a new DECPARM.DAT when the network is reloaded and NCP is rerun.
- Runs the new configuration and verification utility automatically upon system reboot.
- Adds the STACKS parameter to the system startup file, CONFIG.SYS (if you are running DOS Version 3.20 and the parameter is not already there). Currently this parameter is set to the default values defined by DOS. In subsequent releases the default values may change.

- Copies all character translation tables and keyboard screen template files automatically from the kit to the specified DECnet database path (if you select SETHOST.EXE to be installed).
- Copies the FIXNVD.EXE file automatically from the kit if you select the Network Device utility (NDU) to be installed. (Refer to Section 1.6 for information about using FIXNVD.EXE with NDU.)
- Adds the command line switch /IRQ = 3 in the AUTOEXEC.BAT file when invoking the DLL for either the 3COM driver or the MICOM driver. (Refer to Section 1.13 for more information about this switch.)
- Displays the current EXECUTOR node name and address if DECnet is already installed and is running on your system. DIP gives you the option of changing the node name and address at the same time it displays the information.
- Creates and/or modifies the Spawner database file (DECSPAWN.DAT).
- Creates backup copies of the CONFIG.SYS and AUTOEXEC.BAT files before editing them. The backup files have the extension .BAK.

  For example:

  CONFIG.SYS will have a backup file named CONFIG.BAK

  AUTOEXEC.BAT will have a backup file named AUTOEXEC.BAK

  If you select the Job Spawner, FAL, or DTR during the installation procedure, DIP will also create a backup copy of the DECSPAWN.DAT file.

  For example:

  DECSPAWN.DAT will have a backup file named DECSPAWN.BAK

1.4 Changes to DECnet–DOS SETHOST

1.4.1 Bug Fixes

SETHOST has implemented bug fixes for the following problems:

- The "partial escape sequence seen" error observed when running ALL–IN–1 Version 2.0 software with the WPS–PLUS editor over a CTERM link has been fixed.
- Connections to reverse LAT service now work (Version 1.2 and later of the Terminal Server). (Reverse LAT service is a service offered by a terminal server for a host connected to it by an asynchronous terminal line.) SETHOST connections to LAT services behave as if you were connected to a terminal server. However, SETHOST does not support direct connections to terminal servers.
- In previous versions, using the HOLD SCREEN key while connected over a CTERM link would cause the system to hang. This has been fixed so that HOLD SCREEN now works properly with CTERM.

- In previous versions, if you were logging a terminal session to the disk and the disk became full, the system would hang. This has been fixed so that instead of hanging the system when the disk fills up, SETHOST now closes the file and displays an error message.

1.4.2 Newly Supported Features

Version 1.2 of SETHOST includes support for the following:

- The IBM Enhanced PC keyboard for the IBM PC/XT and IBM Personal Computer AT
- Local IBM-style print screen (for IBM PCs)
- 132-column support (for IBM PCs with color adapters only)
- The DEC LK250 keyboard
- Four new switches (see Section 1.4.3)
- DEC national character sets (7-bit) and DEC multinational character sets (8-bit). Note that full use of 8-bit character sets is limited due to hardware and operating system constraints.

1.4.2.1 Using 132-Column Support — To use 132-column support on IBM PCs, you must have a color adapter (monochrome adapters can only support the use of 80 columns). If you have a color monitor with color disabled, you need to enter the Setup menu and enable color. You must also set the terminal characteristics to 132-column mode. You can set this mode either by selecting it from the Setup menu or by using a SET TERMINAL command (such as the VMS command, SET TERM/WIDTH = 132). Once this mode is set, you can switch easily back and forth between screens by using the ALT/F9 key combination.

1.4.2.2 Using National and Multinational Character Sets — The following is a detailed list of the character sets that are available with SETHOST:

- DEC Multinational STD 1 Character Set
- DEC Multinational STD 2 Character Set
- United States/Canada Character Set
- Dutch Character Set
- Finnish Character Set
- French Canadian Character Set
- French Character Set
- German Character Set
- Italian Character Set
- Norwegian/Danish Character Set
- Spanish Character Set
- Swedish Character Set
- Swiss Character Set
- United Kingdom Character Set

You can select or change character sets from the General setup menu, which is now accessible with the new /SETUP switch. (Refer to the following section which describes the four new switches for SETHOST.)

1.4.3 New Switches

Four new switches have been added to SETHOST. These switches allow you to:

- tell SETHOST which keyboard layout to use.
- display the LAT service name table (similar to the SHOW SERVICE command on a terminal server).
- change the setup for the terminal emulator without having to make a network connection first.
- change the switch character that SETHOST uses for accessing the SETHOST menu.

The switches are:

1. /KEYBOARD
2. /SERVICE
3. /SETUP
4. /SWITCH = *n*

1.4.3.1 /KEYBOARD — The /KEYBOARD switch allows you to specify to SETHOST the type of keyboard you are using. When you use this switch (or when you first start SETHOST on an IBM PC), SETHOST displays the keyboard menu. You can then specify the keyboard you want to use by making a selection from the menu.

The following is an example of the SETHOST keyboard menu:

```
Please indicate the type of keyboard you are using with your system.
If you wish to change keyboards at a later date,
use SETHOST /KEYBOARD to display this menu again.

        Enter the number that is the closest match to your keyboard:

        1) PC or PC/XT
        2) PC AT
        3) Enhanced PC
        4) Digital VAXmate (LK250)

        The current default is keyboard number 3.

        Keyboard number?   RET
        Defaulting to keyboard number 3.
```

In the previous example, the default keyboard selection was 3. If you do not select a number from the menu, SETHOST tries to determine the type of keyboard you are using and defaults to that number. SETHOST can also use information that was saved from your previous keyboard selection. The information is stored in the NVTDEF.DAT file in the DECnet database directory.

See on-line help (SETHOST /HELP) for further information.

1.4.3.2 /SERVICE — The /SERVICE switch displays the personal computer LAT driver's service name table. This table lists the services that are available for use by LAT. The services listed are those that have been defined using NCP or those that are advertised on the local area network. If you are using LAT, you can only connect to a service whose name appears in the service name table.

The default size of the service name table is 10. You can change the size of the table by using the LAT switch, /D:*n*. (The DECnet Installation Procedure places the LAT command in the AUTOEXEC.BAT file without any switches. To add the /D switch and change the value of *n*, edit the AUTOEXEC.BAT file.) The value of *n* is added to the default value of 10. For example, if you specify LAT /D:4, the list displayed by SETHOST /SERVICE can include up to 14 service names.

Once you have seen the list of services that are available, you can connect to any of the services by issuing the SETHOST *service-name* command. (You must be using LAT protocol for the connection.)

1.4.3.3 /SETUP — The /SETUP switch allows you to invoke the setup menus without having to make a network connection. You can change settings in any of the setup directories. Use the cursor arrow keys to move between boxes, then press RET or enter to make a selection in each box. When you have completed all of your selections, use the Save Parameters option in the Action menu to save the new defaults.

1.4.3.4 /SWITCH = *n* — The /SWITCH = *n* switch lets you specify a character to use for accessing the SETHOST menu during an active session. By default, the CTRL/4 RET key combination displays this menu. In previous versions, the CTRL/\ RET key combination would display this menu. (The CTRL/\ RET combination was difficult to use on European keyboards, so the default was changed to CTRL/4 RET.) Both combinations will display the menu.

NOTE

> **This is a two-step sequence**. In order for CTRL/4 RET to work properly, you must first press CTRL and 4 simultaneously, then release the two keys and press RET.

With /SWITCH, you can replace the CTRL/4 default with a different character. The variable *n* can be an ASCII decimal code character from 1 to 28, where 1 is equivalent to CTRL/A, 2 is equivalent to CTRL/B, 3 is equivalent to CTRL/C, and so on up to 26. Character 27 is equivalent to the ESC character (not recommended, but it can be used) and 28 is equivalent to CTRL/\.

For example:

SETHOST /SWITCH = 2 would change CTRL/4 RET to CTRL/B RET.

Using SETHOST /SWITCH with no argument resets the default to CTRL/4 RET.

1.4.4 Enhancements

SETHOST Version 1.2 includes the following enhancements:

- There is a new keyboard handler. It takes over the keyboard interrupt only for the keys it needs. All other keys are handled or processed by the standard MS–DOS operating systems. This allows some software that was not compatible with DECnet–DOS Version 1.1 to run with DECnet–DOS Version 1.2 (for example, Borland International's SideKick).
- New single screen keyboard templates are available on line as files in the DECnet database directory (such as the \DECNET directory). The files are:
  - ATTPLUS.HLP (for the AT&T Personal Computer 6300 PLUS)
  - IBMAT.HLP (for the IBM Personal Computer AT)
  - IBMEPC.HLP (for the IBM Enhanced Personal Computer)
  - IBMXT.HLP (for the IBM PC/XT)
  - ZENAT.HLP (for the ZENITH AT compatible)
  - ZENXT.HLP (for the ZENITH XT compatible)

  NOTE

  Keyboard templates are provided for reference purposes only. Their presence in the on-line help files does not imply support for any PCs other than those listed in the Software Product Description (SPD).

  The on-line help text also provides pointers to these files (when you use the /HELP switch). Illustrations of the keyboard templates for the IBM PC/XT, PC AT and the IBM Enhanced Personal Computer are included in Appendix A of this document. Illustrations of the Rainbow and the VAXmate keyboards are also included in Appendix A.
- There is now a way to configure SETHOST for international keyboards. To do this, follow these steps:
  1. Install the DOS KEYB program to select the keyboard you want. (The KEYB program is a DOS program that replaces the resident keyboard program. For example, if you wanted to use a French keyboard, you would install the KEYBFR program.)
  2. Tell SETHOST which character set you wish to use (follow steps 3 through 6).
  3. Type SETHOST/SETUP to get into the Setup menu.
  4. Use the arrow keys and the RET or ENTER key to get to the General menu.
  5. Select the character set you want to use.
  6. Save this as the new default by using the Save Setup Parameters option on the Action menu.

- SETHOST provides translation only from the ROM character set to a DEC character set and back again. On the IBM PC, this means that the IBM international key sequences will work to generate an IBM character code. This code will then be translated by SETHOST into a DEC character.

  For example:

  - Two-key compose sequences. (To display é, you must first press ´and then press e.)
  - CTRL/ALT/F1 to toggle back to a US English keyboard layout (press all 3 keys at the same time). CTRL/ALT/F2 to toggle back to the country keyboard layout.
  - ALT/*nnn*, where *nnn* is a 3-digit decimal number. This combination can be used to generate any IBM ROM character.

- SETHOST also has a DEC COMPOSE key. However, the operating system (DOS) sees the keys first and may choose to preempt some of the keys for the IBM style of international support. If this happens, there are two ways around it:

  1. Use the IBM method

     or

  2. Use CTRL/ALT/F1 to switch temporarily back to a US English keyboard, then use the COMPOSE key.

- DEC's multinational STD 2 character set is for use in Norway and other countries that use character set ROMs that are different from the US character set ROMs.

- The character set files that SETHOST uses for translating are stored in the DECnet database directory (for example, C:\DECNET\*.CHR). These are ASCII text files, and they contain two 256-character tables. The first table is the DECIBM table. The DEC character looks into this table to find the character it needs, then returns the IBM ROM character (which is then displayed on the screen). The second table is the IBMDEC table. The IBM ROM character looks into this table to find the character that it needs, then returns the DEC character (which is then displayed on the screen).

  Since SETHOST uses the IBM ROM character set and not a graphic character set for displaying characters, you are limited in the use of certain characters. When you select a character from the character set file, SETHOST tries to determine the closest match to the character you want to display.

- In the General setup menu, leaving or moving out of the character set menu causes the character set file that you selected to be read to the disk. You can make your selection while you are still in this menu, but it will not take effect until you move out of the menu. Once you have left the menu, the file you selected will replace the internal tables that SETHOST uses. If you hear a beep, then that file was not found.

SETHOST's default table is for DEC multinational character sets with a US keyboard (this also works for 7-bit US keyboards). SETHOST will not beep if the DECM1.CHR or US.CHR files cannot be found; it will just use the default tables.

- Since the VT100 terminal does not have an escape sequence to tell the remote system that SETHOST can use 8-bit characters, you must tell the remote system and the SETHOST terminal emulator to use 8-bit characters (if you select the DEC multinational character set). For example, on a remote VMS system, you would add SET TERM/EIGHT to your LOGIN.COM file.

- For the 7-bit national character sets, some keys will display different characters depending on whether the keyboard is in typewriter mode or in data processing mode. You can switch from one mode to the other using the General setup menu. This allows you to get to characters that would normally be available only with DEC multinational character sets.

  The keys that are affected by this are #,], [, @, \, }, {, |, ~, `, and ^. What these keys change to depends on the character set you select.

- Two files, NVTDEF.DAT and NVTVID.DAT, are created when using SETHOST. These files contain information about the version of SETHOST that you are using. The NVTDEF.DAT file is created when you use the /KEYBOARD switch to make a keyboard selection, or when you use the /SAVE switch. The NVTVID.DAT file is created when you save information using the Save Parameters option in the Setup menu. In previous versions of SETHOST, the file names were SETHOST.DEF and VT102.DAT. These older files are still intact if you want to use them, but you should use the new files.

  When you start SETHOST, it checks for the proper version number in NVTVID.DAT. If SETHOST determines that this file contains the wrong version, it will beep and ignore the information in the file. You should use the Save Parameters option to replace the NVTVID.DAT file with an updated copy that contains the correct information.

- If you are using ALL–IN–1 or WPS–PLUS software on a VT100 terminal, DEC multinational characters are not displayed. In order to get WPS–PLUS software to use the multinational character set (MCS) and to properly display the MCS characters in the on-line help screens, you need to define the following symbols on your VMS system:

  DEFINE KOA$TERMINAL_*xxxx* "OUTPUT_SETHOST"
  DEFINE KOA$TERMINAL_MCS_SETHOST "Y"

  where *xxxx* is the terminal name (such as RTA1, VTA32, and so on).

1.5 Changes to the DECnet–DOS Mail Utility

Mail no longer uses the access control information that is saved by NCP.

Mail now adds the DECnet node address of the mail sender to the Subject field, just before the "Reply to" address. In Version 1.1, this address was not included. Now when other users receive mail from your DECnet–DOS node, your node's address appears in this field. For example:

```
Subj: Finance Meeting   "From 55.125, Reply to GRAVY::NELSON"
```

1.6 Changes to the DECnet–DOS Network Device Utility (NDU)

After you issue an NDU DELETE command, NDU now prompts you for confirmation in the MS–DOS style:

```
Are you sure (Y/N)?
```

1.6.1 Bug Fixes

In DECnet–DOS Version 1.1, the virtual disk facility was shipped with the following problems:

1. A bug caused all DECnet–DOS Version 1.1 32-megabyte disks created as the first NDU function after a system boot (under MS–DOS or PC DOS Version 2 systems) to have an error in their headers. This error does not affect the performance or integrity of the virtual disk facility.
2. 32-megabyte disks created under MS–DOS or PC DOS Version 2 systems are not usable from MS–DOS or PC DOS Version 3 systems. (The opposite of this is also true: 32-megabyte disks created on MS–DOS or PC DOS Version 3 systems are not usable from MS–DOS or PC DOS Version 2 systems.)

DECnet–DOS Version 1.2 solves both of these problems with the following:

- All virtual disks created by DECnet–DOS Version 1.0 software remain usable in all configurations.
- All 1.2-megabyte, 10-megabyte, and 20-megabyte virtual disks created by DECnet–DOS Version 1.1 remain usable in all configurations.
- All 32-megabyte virtual disks created by DECnet–DOS Version 1.1 under MS–DOS or PC DOS Version 3 systems will remain usable on any Version 3 system. These disks are not usable from MS–DOS or PC DOS Version 2 systems.

- 32-megabyte virtual disks created by DECnet–DOS Version 1.1 under MS–DOS or PC DOS Version 2 systems are not usable with DECnet–DOS Version 1.2 **until they are repaired using the FIXNVD utility**. The FIXNVD utility is automatically installed by the DECnet–DOS Installation Procedure, DIP. The utility's file name is FIXNVD.EXE.

  If you attempt to open a Version 1.1 virtual disk with Version 1.2, NDU generates the following error message:

  ```
  Cannot OPEN this disk created with an older release of DECnet.
   Run FIXNVD first.
  ```

  To use the FIXNVD utility, first make a copy of your virtual disk file. For example:

  ```
  $ COPY V11DISK1.FIL V11DISKBU.DSK RET
  ```

  Now you can run FIXNVD on the new file. This is the syntax for using FIXNVD:

  FIXNVD *node*[*/user/password/account*] *backup-disk-name*

  Example 1:

  ```
  $ COPY VDISK.DSK DISK.007 RET
  ```

  (on VMS node VMSNOD)

  ```
  C> FIXNVD VMSNOD DISK.007 RET
  ```

  (on your PC)

  Example 2:

  ```
  $ COPY VDISK.DSK DISK$07:[SMITH.DISKS]PAYROLL.DAT RET
  ```

  (on VMS node REMOTE)

  ```
  C> FIXNVD REMOTE/SMITH/HARP DISK$07:[SMITH.DISKS]PAYROLL.DAT RET
  ```

  (on your PC)

FIXNVD will not write to a file unless it is a proper candidate for repair.

NOTE

> FIXNVD will make the files and directories allocated in excess of 33,439,744 bytes inaccessible! In fixing the bug and making Version 2 and Version 3 disks compatible, FIXNVD must shrink the maximum size of 32-megabyte disks. Any files or directories written when the disk volume was filled in excess of 33,439,744 bytes will become inaccessible. Run CHKDSK to determine if this has happened:
>
> CHKDSK D:
>
> If the disk volume is filled, reinstall DECnet–DOS Version 1.1 (or just NDU.EXE and NDDRV.SYS from DECnet–DOS Version 1.1) and restore the backed-up virtual disk. Copy the files from the backed-up disk to two new virtual disks, then reinstall DECnet–DOS Version 1.2.

1.7 Changes to DECnet–DOS Transparent File Access (TFA)

Transparent file access functions let you open and close a remote file, create or delete a remote file, read from or write records to a remote file, submit a batch job, or search a directory for a specific file or files. Since these functions are performed transparently, you do not need to use the information in this section unless you are performing network programming tasks.

You can also use TFA to type and copy files to and from remote nodes. Use the following format:

TYPE \\f\node\\filespec
TYPE \\f\node\\remotefile > *localfile*
COPY *localfile*\\f\node\\remotefile

You should note that the COPY command only works if the remote file already exists on the remote node. If the file does not exist on the remote node, the COPY operation will fail.

1.7.1 Bug Fixes

TFA has implemented bug fixes for the following problems:

- In previous versions of TFA, a file OPEN operation which specified both read and write access would only allow writing. Now it only allows reading. TFA cannot both read and write a file during a single OPEN/CLOSE session.
- The Transparent Network Test utility (TNT) previously reported a false error if it was run right after TFA was installed. An actual error condition did not exist. This has been fixed so that a false error is no longer reported.

- Read access to a file larger than 64K bytes with null record attributes (such as a large RUNOFF output file) would appear to succeed, but no data would be returned. This has been fixed. Data is now returned, indicating whether or not the operation succeeded.
- Issuing a file CLOSE with a handle of –1 indicates that the logical link in use for directory functions be closed. TFA did not check to insure that it was open first. This caused writing to random locations in memory. This has been fixed.
- If a FIND FIRST FILE function was performed following a FIND FIRST FILE without an intervening CLOSE command (using a handle of –1), an error would occur. The second FIND FIRST FILE function now closes the link automatically.

1.8 Changes to DECnet–DOS DTR/DTS

In DECnet–DOS Version 1.1, the DECnet Test Receive utility (DTR) would automatically log information to the file DTR.LOG in the DECnet database directory whenever it was run by the Spawner. DTR no longer does this automatically. In DECnet–DOS Version 1.2, if you want to log information to the DTR.LOG file you must specify the /LOG switch. For example:

```
C:\DTR /LOG RET
```

1.9 Changes to the DECnet–DOS Spawner

1.9.1 Bug Fixes

There was a previous parsing restriction for DECSPAWN.DAT (the parameter file for the Spawner utility) that would only allow a single space between the object name or number and the parameter associated with it. This restriction has been removed. There may be multiple spaces and/or tabs between the object name or number and the parameter.

1.9.2 Newly Supported Features

- The Spawner can now support logging, using the /LOG switch. When you use this switch, the Spawner creates a file called SPAWNER.LOG in the DECnet database directory (for example, \DECNET\SPAWNER.LOG). All network accesses are recorded into this file. The log information will include the name of the connecting node, the object number, and the time of the request. The information in the log file will not include any actions that occurred as the result of the request. (Those actions or results would appear on the screen at the time of the request.)

  Format for using the /LOG switch:

  SPAWNER /LOG

The following is a sample SPAWNER.LOG file:

```
SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 15:41:52
Connect request from node BOCA for object #17 on Wed Sep 10 1986 at
15:42:02
Executing: fal -use 2... on Wed Sep 10 1986 at 15:42:03
SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 15:42:08
SPAWNER exiting... on Wed Sep 10 1986 at 15:42:14
```

(In this example, object #17 is FAL, a request to run the File Access Listener.)

- The Spawner can now execute batch files. For example, you can create a BATCH.BAT file that contains the following information:

  ```
  ECHO Executing Batch File---Echoing arguments passed:
  Echo %1 %2 %3 %4
  ```

  In this sample file, %1, %2, %3, and %4 represent the different arguments you can use (such as ARG1, ARG2, and so on).

  Then, define the following in your DECSPAWN.DAT file:

  ```
  #129        C:\DECNET\BATCH.BAT        ARG1      ARG2
  ```

  When you run the Spawner, it detects a connect request for object number 129 (# 129) and accepts the request. It then spawns the batch file (BATCH.BAT) with the command line arguments specified in DECSPAWN.DAT (in this case, the arguments are ARG1 and ARG2). The Spawner also passes a socket number to the batch file. The socket appears as –USE *n*, where *n* is the socket number. The batch file can use this socket, or it can ignore it.

  After the batch file completes processing and control is returned to the Spawner, the Spawner closes the socket whose number it had previously passed to the batch file. (This is the default. If you do not close the socket from the batch file, the Spawner will automatically close the one that was opened at the start of the batch operation.)

  The output looks like this:

  ```
  press '!' to abort
  SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 15:49:33

  Connect request from node BOCA for object #129 on Wed Sep 10 1986 at
  15:49:36

  C:\SPAWNER\T>echo Executing Batch File---Echoing arguments passed:
  Executing Batch File---Echoing arguments passed:

  C:\SPAWNER\T>Echo ARG1 ARG2 -USE 2
  ARG1 ARG2 -USE 2

  SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 15:49:40

  SPAWNER exiting... on Wed Sep 10 1986 at 15:49:54
  ```

- The Spawner can also pass command line arguments to the servers. For example, you can now run FAL and include any of the valid FAL qualifiers. The following is a sample DECSPAWN.DAT file which includes command line arguments for FAL:

  ```
  #17          FAL          /E          /B
  ```

  FAL can now be spawned with the arguments /ERROR (/E) and /BINARY (/B). The output looks like this:

  ```
  press '!' to abort
  SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 16:01:23

  Connect request from node BOCA for object #17 on Wed Sep 10 1986 at
  16:01:32

  Executing: fal -use 3... on Wed Sep 10 1986 at 16:01:32
                  FAL - File Access Listener - Version 1.2

  Network Driver Version: 1.2
  Current working directory:C:\SPAWNER\T
  All files will be sent as binary.
  Existing files will NOT be overwritten.
  No access checking will be done (world has read/write privileges).

  FAL running...
  DIRECTORY access from BOCA:: for LOCAL""::*.*

  SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 16:01:37

  SPAWNER exiting... on Wed Sep 10 1986 at 16:01:40
  ```

  The following is a sample DECSPAWN.DAT file requesting that DTR be run using the /LOG switch:

  ```
  #63     DTR     /LOG
  ```

  Now when you run the Spawner, you will see output that looks like this:

  ```
  press '!' to abort
  SPAWNER (Version 1.2) listening... on Wed Sep 24 1986 at 10:22:10

  Connect request from node MSDOS for object #63 on Wed Sep 24 1986 at
  10:22:15

  Executing: dtr -use 2... on Wed Sep 24 1986 at 10:22:15

  [Appending messages to log file: C:\DECNET\DTR.LOG]

  DTR --I-- V1.2 (19-Sep-86) started on Wed Sep 24 1986 at 10:22:15
  DTR --I-- Terminated on Wed Sep 24 1986 at 10:22:16
  SPAWNER (Version 1.2) listening... on Wed Sep 24 1986 at 10:22:16

  SPAWNER exiting... on Wed Sep 24 1986 at 10:22:25
  ```

1.9.3 Enhancements

The on-line help text has been updated to include the new functionality.

1.10 Changes to the DECnet–DOS File Access Listener (FAL)

FAL has implemented the following bug fixes for use with RSTS systems:

- In previous versions of DECnet–DOS FAL, wildcard access for DELETE operations from RSTS systems did not work. This has been corrected so that wildcards can be used in file specifications to delete more than one file at a time.
- In previous versions of FAL, DELETE requests from RSTS systems did not include any confirmation. All of the files in the request were deleted, with no chance of changing your mind and saving any of the files. This has been corrected so that any DELETE requests from an RSTS system will first ask for confirmation before a file is deleted.

1.11 Changes to the DECnet–DOS Network File Transfer Utility (NFT)

1.11.1 Enhancements

- In DECnet–DOS Version 1.1, the Network File Transfer utility (NFT) displayed the following message after successful COPY and APPEND operations:

  [*n* records]

  where *n* represented the number of records copied or appended.

  This has been changed so that NFT now reports the number of bytes transferred and the transfer rate. (The transfer rate is computed by using the time interval from a successful OPEN of the remote file through a CLOSE of the remote file. The rate does not include process startup time on the remote system.) The report appears in the following format:

  [*n* bytes at *m* bytes/second]

  where *n* is the number of bytes and *m* bytes/second is the rate at which the bytes were transferred.
- NFT's performance has been improved significantly.
- NFT now prompts you for confirmation of a DELETE request for any of these file specifications:

  "*.*", "*.*;*", or "*.*.*".

  The prompt is in the MS–DOS style:

  ```
  Are you sure (Y/N)?
  ```

1.12 Changes to the DECnet–DOS Programming Interface Library (DNETLIB)

1.12.1 Bug Fixes

The Programming Interface Library (DNETLIB) has implemented bug fixes to correct the following problems:

- The *dnet_conn*( ) function has been fixed to properly check error returns on all socket calls.
- The bug in *dnet_getalias*( ) has been fixed to properly check for –1 before attempting to close the DECALIAS.DAT database file.

1.12.2 Enhancements

DNETLIB Version 1.2 includes the following enhancements:

- Socket calls such as *send*( ), *recv*( ), *swrite*( ), and *sread*( ) have been modified for improved performance.
- The function *dnet_getacc*( ) now performs case insensitive comparisons when checking incoming access control information.
- The Break Source utility (BREAKSRC) has been changed to interpret DNETLIB.SRC as a compressed binary file instead of an ASCII file.
- The sources for 3 sample C programs are now included in the DNETLIB.SRC file. The source file names are:
  - LOOP.C (a sample client program)
  - MIRROR.C (a sample server program)
  - TTTXAMPL.C (a sample transparent task-to-task program)

  Appendix B contains copies of these sample programs.

1.12.3 Compiling DNETLIB Sources

When certain C compilers are allocating storage for data structures whose members are larger than a ''char'', they may ordinarily begin this storage on ''int'' boundaries.

If DNET library sources are compiled in this way, then calls to the DECnet Network Process (DNP) may fail with errors (for example, ''Protocol Family not supported''). These errors are due to the fact that the data structures which are being passed down to DNP have some bad bytes. This results in data not being in places where DNP expects them, or in data structures that are not of the supported byte sizes.

To avoid these data errors, if your C compiler has a switch which causes data structures to be packed more tightly, then you should use that compiler and the accompanying switch for compiling the DNET library sources.

1.12.4 Newly Supported Features

Support has been added to the DNET source library to allow building of medium-model and small-model programs. (A medium-model program is a program that contains multiple code segments and one data segment. A small-model program is a program that contains one code segment and one data segment.)

1.13 Changes to the DECnet–DOS Data Link Layer (DLL)

The DECnet–DOS DLL that is used with the MICOM driver has implemented a bug fix for the following:

- The MICOM driver now responds with the proper device type for MOP system ID messages.

1.13.1 Enhancements

1.13.1.1 MICOM Driver — An initialization switch has been added to allow changes to the interrupt request line (IRQ) for the DLL that is used with the MICOM driver. The switch is:

/IRQ = *n*

where *n* is a line number from 1 to 7. (DECnet–DOS only supports 7 lines within the PC bus structure.)

For example:

```
DLL C:\DECNET /IRQ = 2
```

If you do not use the switch to change this setting, the default value is 3.

1.13.1.2 3COM Driver — An initialization switch has also been added to allow changes to the interrupt request line (IRQ) for the DLL used with the 3COM driver. The switch is the same as the one used for the DLL MICOM driver:

/IRQ = *n*

where *n* is a line number from 1 to 7. (DECnet–DOS only supports 7 lines within the PC bus structure.)

For example:

```
DLL C:\DECNET /IRQ = 4
```

If you do not use the switch to change this setting, the default value is 3.

1.13.1.3 DEPCA Driver — The DEPCA is Digital's new Ethernet communications board for personal computers. DECnet–DOS Version 1.2 now includes Data Link layer support for the DEPCA. (To install the DEPCA board, refer to the installation guide that was included with it.)

The DEPCA Data Link layer is self-configuring. There are no switches needed to specify the IRQ setting.

1.14 Changes to DECnet–DOS DECnet Network Process (DNP)

1.14.1 Bug Fixes

A bug which formerly prevented asynchronous accept calls from working properly has been fixed.

In the Assembly Language SETSOCKOPT and GETSOCKOPT programming calls for DNP, the size of the *sockopt_dn* data structure is 12 bytes (not 14 bytes as previously mentioned in the *DECnet–DOS Programmer's Reference Manual*).

The *op_optlen* data structure listed in the Data Structure Type Summary for the SETSOCKOPT and GETSOCKOPT calls should read *sop_optlen*.

The *snd_how* data structure is documented under the SHUTDOWN call, and should not be listed in the Data Structure Type Summary for the SETSOCKOPT and GETSOCKOPT calls.

1.14.2 Newly Supported Features

The DECnet–DOS Network Process (DNP) Version 1.2 now supports the new Enhanced PC/XT.

In previous versions of DECnet–DOS, the size of messages used for non-transparent task-to-task programming was limited to 2K (or 2048 bytes). DNP now supports messages of up to 4K (or 4096 bytes).

1.14.3 Enhancements

Changes have been made to make DNP more powerful during times when buffers are in short supply.

1.15 Changes to the DECnet–DOS Scheduler

The Scheduler takes control of the real-time clock for all operations whenever you start the DECnet–DOS network. To preserve compatibility with applications that are not well-behaved or with systems that do not have a CMOS clock, a switch has been added (/S) to the Scheduler startup. This switch lets you force the Scheduler to use the system clock instead of the real-time clock.

To use the /S switch, you can enter it at the DOS prompt or you can include it in the batch (or command) file that you use to start the network.

Example 1 (from the DOS prompt)**:**

```
C> SCH/S (RET)
```

or

```
C> SCH /S (RET)
```

Example 2 (a sample AUTOEXEC.BAT batch file)**:**

```
echo off
set COMSPEC=C:\COMMAND.COM
PATH C:\DECNET\:C:\TASKS:C:\DOS

REM *** DECnet-DOS Ethernet Configuration ***
SCH /S
DLL C:\DECNET\
LAT
DNP C:\DECNET\
TTT
TFA
```

2
DECnet–Rainbow Changes

2.1 Overview

This chapter describes the changes that have been made from DECnet–Rainbow Version 1.0 and 1.1 to DECnet–Rainbow Version 1.2. The changes are highlighted in the following list, then broken down by components in later sections.

2.2 Highlight of Changes from Previous Versions of DECnet–Rainbow

- Bug fixes.
- Performance improvements to Network File Transfer (NFT).
- Functional improvements to the Job Spawner and the DECnet Test Receive utility (DTR).
- Changes to the DECnet–Rainbow Programming Interface Library sources (DNETLIB) that provide enhanced performance for programming applications that use this library.

The following sections describe changes that have been made to the individual components.

2.3 Changes to the DECnet–Rainbow Installation Procedure (DIP)

The DECnet–Rainbow Installation Procedure (DIP) is an automated procedure that lets you easily install DECnet–Rainbow by responding to a set of questions and making selections from a series of menus. Refer to the *DECnet–Rainbow Installation Guide* for instructions on how to use DIP, while noting the changes and enhancements that are described here.

2.3.1 Enhancements

During installation, DIP now performs the following operations:

- Checks for the proper version of the DIP.DAT/DIP.SAV database and displays an error if it is an incompatible version.
- Provides enhanced performance for copying files from the kit.
- Runs the new configuration and verification utility automatically upon system reboot.
- Copies the FIXNVD.EXE file automatically from the kit if you have selected the Network Device utility (NDU) to be installed. (Refer to Section 2.6 for information about using FIXNVD.EXE with NDU.)
- Displays the current EXECUTOR node name and address if DECnet is already installed and is running on your system. DIP gives you the option of changing the node name and address at the same time it displays the information.
- Creates and/or modifies the Spawner Database File (DECSPAWN.DAT).
- Creates backup copies of the CONFIG.SYS and AUTOEXEC.BAT files before editing them. The backup files have the extension .BAK. For example:

  CONFIG.SYS will have a backup file named CONFIG.BAK

  AUTOEXEC.BAT will have a backup file named AUTOEXEC.BAK

  If you select the Job Spawner, FAL, or DTR during the installation procedure, DIP will also create a backup copy of the DECSPAWN.DAT file. For example:

  DECSPAWN.DAT will have a backup file named DECSPAWN.BAK

2.4 Changes to DECnet–Rainbow SETHOST

2.4.1 Bug Fixes

The "partial escape sequence seen" error observed when running ALL–IN–1 Version 2.0 software over a CTERM link has been fixed.

2.4.2 Newly Supported Features

DECnet–Rainbow now includes support for 8-bit multinational character sets over a CTERM link. (You should note that full use of 8-bit character sets is limited due to hardware and operating system constraints.)

DECnet–Rainbow also supports 132 columns as a terminal characteristic. You can set 132-column mode either by using the Rainbow's Setup menu or by issuing a terminal command. For example, on VMS systems you would use the command SET TERM/WIDTH = 132.

2.4.3 Enhancements

- A new switch has been added to SETHOST, /SWITCH = *n*.

  The /SWITCH = *n* switch lets you specify a character to use for accessing the SETHOST menu during an active session. By default, the CTRL/4 RET key combination displays this menu. In previous versions, the CTRL/\ RET key combination would display this menu. (The CTRL/\ RET combination was difficult to use with European keyboards, so the default was changed to CTRL/4 RET.) Both combinations will display the menu.

  NOTE

  This is a two-step sequence. In order for CTRL/4 RET to work properly, you must first press CTRL and 4 simultaneously, then release the two keys and press RET.

  With /SWITCH, you can replace the CTRL/4 default with a different character. The variable *n* can be an ASCII decimal code character from 1 to 28 (where 1 is equivalent to CTRL/A, 2 is equivalent to CTRL/B, and so on).

  For example:

  SETHOST /SWITCH = 2 would change CTRL/4 RET to CTRL/B RET.

  Using SETHOST /SWITCH with no argument resets the default of CTRL/4 RET.

  NOTE

  When you use on-line HELP for SETHOST, you will see other switches listed (/KEYBOARD, /SERVICE, and /SETUP). These other switches have no effect on DECnet–Rainbow SETHOST.

- Two files, NVTDEF.DAT and NVTVID.DAT, are created when using SETHOST. These files contain information about the version of SETHOST that you are using. The NVTDEF.DAT file is created when you use the /SAVE switch. The NVTVID.DAT file is created when you save information using the Save Parameters option in the Setup menu. In previous versions of SETHOST, the files were named SETHOST.DEF and VT102.DAT. The older versions of the files are still intact if you want to use them, but you should use the new files.

  When you start SETHOST, it checks for the proper version number in NVTVID.DAT. If SETHOST determines that this file contains the wrong version, it will beep and ignore the information in the file. You should use the Save Parameters option to replace the NVTVID.DAT file with an updated copy that contains the correct information.

- If you are using ALL–IN–1 or WPS–PLUS software on a VT100 terminal, DEC multinational characters are not displayed. In order to get WPS–PLUS software to use the multinational character set (MCS) and to properly display the MCS characters in the on-line help screens, you need to define the following symbols on your VMS system:

  DEFINE KOA$TERMINAL_*xxxx* "OUTPUT_SETHOST"
  DEFINE KOA$TERMINAL_MCS_SETHOST "Y"

  where *xxxx* is the terminal name (such as RTA1, VTA32, and so on).

2.5 Changes to the DECnet–Rainbow Mail Utility

Mail no longer uses the access control information that is saved by NCP.

Mail now adds the DECnet node address of the mail sender to the Subject field, just before the "Reply to" address. In Version 1.1, this address was not included. Now when other users receive mail from your DECnet–Rainbow node, your node's address appears in this field. For example:

```
Subj: Finance Meeting   "From 55.125, Reply to GRAVY::NELSON"
```

2.6 Changes to the DECnet–Rainbow Network Device Utility (NDU)

After you issue an NDU DELETE command, NDU now prompts you for confirmation in the MS–DOS style:

```
Are you sure (Y/N)?
```

2.6.1 Bug Fixes

In DECnet–Rainbow Version 1.1, the virtual disk facility was shipped with the following problems:

1. A bug caused all DECnet–Rainbow Version 1.1 32-megabyte disks created as the first NDU function after a system boot (under MS–DOS or PC DOS Version 2 systems) to have an error in their headers. This error does not affect the performance or integrity of the virtual disk facility.
2. 32-megabyte disks created under MS–DOS or PC DOS Version 2 systems are not usable from MS–DOS or PC DOS Version 3 systems. (The opposite of this is also true: 32-megabyte disks created on MS–DOS or PC DOS Version 3 systems are not usable from MS–DOS or PC DOS Version 2 systems.)

DECnet–Rainbow Version 1.2 solves both of these problems with the following:

- All virtual disks created by DECnet–Rainbow Version 1.0 remain usable in all configurations.
- All 1.2-, 10-, and 20-megabyte virtual disks created by DECnet–Rainbow Version 1.1 remain usable in all configurations.

- All 32-megabyte virtual disks created by DECnet–Rainbow Version 1.1 under MS–DOS or PC DOS Version 3 systems will remain usable on any Version 3 system. These disks are not usable from MS–DOS or PC DOS Version 2 systems.
- 32-megabyte virtual disks created by DECnet–Rainbow Version 1.1 under MS–DOS or PC DOS Version 2 systems are not usable with DECnet–Rainbow Version 1.2 **until they are repaired using the FIXNVD utility**. The FIXNVD utility is automatically installed by the DECnet–Rainbow Installation Procedure, DIP. The utility's file name is FIXNVD.EXE.

  If you attempt to open a Version 1.1 virtual disk with Version 1.2, NDU generates the following error message:

  ```
  Cannot OPEN this disk created with an older release of DECnet.
   Run FIXNVD first.
  ```

  To use the FIXNVD utility, first make a copy of your virtual disk file. For example:

  ```
  $ COPY V11DISK1.FIL V11DISKBU.DSK (RET)
  ```

  Now you can run FIXNVD on the new file. This is the syntax for using FIXNVD:

  FIXNVD *node*[*/user/password/account*] *backup-disk-name*

  Example 1:

  ```
  $ COPY VDISK.DSK DISK.007 (RET)
  ```

  (on VMS node VMSNOD)

  ```
  C> FIXNVD VMSNOD DISK.007 (RET)
  ```

  (on your PC)

  Example 2:

  ```
  $ COPY VDISK.DSK DISK$07:[SMITH.DISKS]PAYROLL.DAT (RET)
  ```

  (on VMS node REMOTE)

  ```
  C> FIXNVD REMOTE/SMITH/HARP DISK$07:[SMITH.DISKS]PAYROLL.DAT (RET)
  ```

  (on your PC)

FIXNVD will not write to a file unless it is a proper candidate for repair.

NOTE

FIXNVD will make the files and directories allocated in excess of 33,439,744 bytes inaccessible! In fixing the bug and making Version 2 and Version 3 disks compatible, FIXNVD must shrink the maximum size of 32-megabyte disks. Any files or directories written when the disk volume was filled in excess of 33,439,744 bytes will become inaccessible. Run CHKDSK to determine if this has happened:

CHKDSK D:

If the disk volume is filled, reinstall DECnet–Rainbow Version 1.1 (or just NDU.EXE and NDDRV.SYS from DECnet–Rainbow Version 1.1) and restore the backed-up virtual disk. Copy the files from the backed-up disk to two new virtual disks, then reinstall DECnet–Rainbow Version 1.2.

2.7 Changes to DECnet–Rainbow Transparent File Access (TFA)

Transparent file access functions let you open and close a remote file, create or delete a remote file, read from or write records to a remote file, submit a batch job, or search a directory for a specific file or files. Since these functions are performed transparently, you do not need to use the information in this section unless you are performing network programming tasks.

You can also use TFA to type and copy files to and from remote nodes. Use the following format:

TYPE \\f\node\\filespec
TYPE \\f\node\\remotefile > *localfile*
COPY *localfile*\\\\f\node\\remotefile

You should note that the COPY command only works if the remote file already exists on the remote node. If the file does not exist on the remote node, the COPY operation will fail.

2.7.1 Bug Fixes

TFA has implemented bug fixes for the following problems:

- In previous versions of TFA, a file OPEN operation which specified both read and write access would only allow writing. Now it only allows reading. TFA cannot both read and write a file during a single OPEN/CLOSE session.
- The Transparent Network Test utility (TNT) previously reported a false error if it was run right after TFA was installed. An error condition did not exist. This has been fixed so that a false error is no longer reported.

- Read access to a file larger than 64K bytes with null record attributes (such as a large runoff output file) would appear to succeed, but no data would be returned. This has been fixed. Data is now returned, indicating whether or not the operation succeeded.
- Issuing a file CLOSE with a handle of –1 indicates that the logical link in use for directory functions be closed. TFA did not check to insure that it was open first. This caused writing to random locations in memory. This has been fixed.
- If a FIND FIRST FILE function is performed following a FIND FIRST FILE without an intervening CLOSE command (using a handle of –1), an error would occur. The second FIND FIRST FILE function now closes the link automatically.

2.8 Changes to DECnet–Rainbow DTR/DTS

In DECnet–Rainbow Version 1.1, the DECnet Test Receive utility (DTR) would automatically log information to the file DTR.LOG in the DECnet database directory whenever it was run by the Spawner. DTR no longer does this automatically. In DECnet–Rainbow Version 1.2, if you want to log information to the DTR.LOG file you must specify the /LOG switch. For example:

```
C:\DTR /LOG RET
```

2.9 Changes to the DECnet–Rainbow Spawner

2.9.1 Bug Fixes

There was a previous parsing restriction for DECSPAWN.DAT (the parameter file for the Spawner utility) that would only allow for a single space between the object name or number and the parameter associated with it. This restriction has been removed. There may be multiple spaces and/or tabs between the object name or number and the parameter.

2.9.2 Newly Supported Features

- The Spawner can now support logging, using the /LOG switch. When you use this switch, the Spawner creates a file called SPAWNER.LOG in the DECnet database directory (for example, \DECNET\SPAWNER.LOG). All file accesses are recorded into this file. The log information will include the name of the connecting node, the object number, and the time of the request. The information will not include any actions that occurred as the result of the request.

  For example:

  SPAWNER /LOG

  The following is a sample SPAWNER.LOG file:

  ```
  SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 15:41:52
  Connect request from node BOCA for object #17 on Wed Sep 10 1986 at
  15:42:02
  Executing: fal -use 2... on Wed Sep 10 1986 at 15:42:03
  SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 15:42:08
  SPAWNER exiting... on Wed Sep 10 1986 at 15:42:14
  ```

  (In this example, object #17 is FAL, a request to run the File Access Listener.)

- The Spawner can now execute batch files. For example, you can create a BATCH.BAT file that contains the following information:

  ECHO Executing Batch File---Echoing arguments passed:

  Echo %1 %2 %3 %4

  In this sample file, %1, %2, %3, and %4 represent the different arguments you can use (such as ARG1, ARG2, and so on).

  Then, define the following in your DECSPAWN.DAT file:

  ```
  #129          C:\DECNET\BATCH.BAT          ARG1     ARG2
  ```

  When you run the Spawner, it detects a connect request for object number 129 (# 129) and accepts the request. It then spawns the batch file (BATCH.BAT) with the command line arguments specified in DECSPAWN.DAT (in this case, the arguments are ARG1 and ARG2). The Spawner also passes a socket number to the batch file. The socket appears as –USE *n*, where *n* is the socket number. The batch file can use this socket, or it can ignore it.

After the batch file completes processing and control is returned to the Spawner, the Spawner closes the socket whose number it had previously passed to the batch file. (This is the default. If you do not close the socket from the batch file, the Spawner will automatically close the one that was opened at the start of the batch operation.)

The output looks like this:

```
press '!' to abort
SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 15:49:33

Connect request from node BOCA for object #129 on Wed Sep 10 1986 at
15:49:36

C:\SPAWNER\T> echo Executing Batch File---Echoing arguments passed:
Executing Batch File---Echoing arguments passed:

C:\SPAWNER\T>Echo ARG1 ARG2 -USE 2
ARG1 ARG2 -USE 2

SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 15:49:40

SPAWNER exiting... on Wed Sep 10 1986 at 15:49:54
```

- The Spawner can also pass command line arguments to the servers. For example, you can now run FAL and include any of the valid FAL qualifiers. The following is a sample DECSPAWN.DAT file which includes command line arguments for FAL:

```
#17          FAL          /E          /B
```

FAL can now be spawned with the arguments /ERROR (/E) and /BINARY (/B). The output looks like this:

```
press '!' to abort
SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 16:01:23

Connect request from node BOCA for object #17 on Wed Sep 10 1986 at
16:01:32

Executing: fal -use 3... on Wed Sep 10 1986 at 16:01:32
                FAL - File Access Listener - Version 1.2

Network Driver Version: 1.2
Current working directory:C:\SPAWNER\T
All files will be sent as binary.
Existing files will NOT be overwritten.
No access checking will be done (world has read/write privileges).

FAL running...
DIRECTORY access from BOCA:: for LOCAL""::*.*

SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 16:01:37

SPAWNER exiting... on Wed Sep 10 1986 at 16:01:40
```

The following is a sample DECSPAWN.DAT file, requesting that DTR be run using the /LOG switch:

```
#63      DTR      /LOG
```

Now when you run the Spawner, you will see output that looks like this:

```
press '!' to abort SPAWNER (Version 1.2) listening... on Wed Sep 24
1986 at 10:22:10

Connect request from node MSDOS for object #63 on Wed Sep 24 1986 at
10:22:15

Executing: dtr -use 2... on Wed Sep 24 1986 at 10:22:15

[Appending messages to log file: C:\DECNET\DTR.LOG]

DTR --I-- V1.2 (19-Sep-86) started on Wed Sep 24 1986 at 10:22:15
DTR --I-- Terminated on Wed Sep 24 1986 at 10:22:16
SPAWNER (Version 1.2) listening... on Wed Sep 24 1986 at 10:22:16

SPAWNER exiting... on Wed Sep 24 1986 at 10:22:25
```

2.9.3 Enhancements

The on-line help text has been updated to include the new functionality.

2.10 Changes to the DECnet–Rainbow File Access Listener (FAL)

FAL has implemented the following bug fixes for use with RSTS systems:

- In previous versions of DECnet–Rainbow FAL, wildcard access for DELETE operations from RSTS systems did not work. This has been corrected so that wildcards can be used in file specifications to delete more than one file at a time.
- In previous versions of FAL, DELETE requests from RSTS systems did not include any confirmation. All of the files in the request were deleted, with no chance of changing your mind and saving any of the files. This has been corrected so that any DELETE requests from an RSTS system will first ask for confirmation before a file is deleted.

2.11 Changes to the DECnet–Rainbow Network File Transfer Utility (NFT)

2.11.1 Enhancements

- In DECnet–Rainbow Version 1.1, the Network File Transfer utility (NFT) displayed the following message after successful COPY and APPEND operations:

  [*n* records]

  where *n* represented the number of records copied or appended.

After the batch file completes processing and control is returned to the Spawner, the Spawner closes the socket whose number it had previously passed to the batch file. (This is the default. If you do not close the socket from the batch file, the Spawner will automatically close the one that was opened at the start of the batch operation.)

The output looks like this:

```
press '!' to abort
SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 15:49:33

Connect request from node BOCA for object #129 on Wed Sep 10 1986 at
15:49:36

C:\SPAWNER\T> echo Executing Batch File---Echoing arguments passed:
Executing Batch File---Echoing arguments passed:

C:\SPAWNER\T>Echo ARG1 ARG2 -USE 2
ARG1 ARG2 -USE 2

SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 15:49:40

SPAWNER exiting... on Wed Sep 10 1986 at 15:49:54
```

- The Spawner can also pass command line arguments to the servers. For example, you can now run FAL and include any of the valid FAL qualifiers. The following is a sample DECSPAWN.DAT file which includes command line arguments for FAL:

```
#17          FAL          /E          /B
```

FAL can now be spawned with the arguments /ERROR (/E) and /BINARY (/B). The output looks like this:

```
press '!' to abort
SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 16:01:23

Connect request from node BOCA for object #17 on Wed Sep 10 1986 at
16:01:32

Executing: fal -use 3... on Wed Sep 10 1986 at 16:01:32
               FAL - File Access Listener - Version 1.2

Network Driver Version: 1.2
Current working directory:C:\SPAWNER\T
All files will be sent as binary.
Existing files will NOT be overwritten.
No access checking will be done (world has read/write privileges).

FAL running...
DIRECTORY access from BOCA:: for LOCAL""::*.*

SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 16:01:37

SPAWNER exiting... on Wed Sep 10 1986 at 16:01:40
```

The following is a sample DECSPAWN.DAT file, requesting that DTR be run using the /LOG switch:

```
#63     DTR     /LOG
```

Now when you run the Spawner, you will see output that looks like this:

```
press '!' to abort SPAWNER (Version 1.2) listening... on Wed Sep 24
1986 at 10:22:10

Connect request from node MSDOS for object #63 on Wed Sep 24 1986 at
10:22:15

Executing: dtr -use 2... on Wed Sep 24 1986 at 10:22:15

[Appending messages to log file: C:\DECNET\DTR.LOG]

DTR --I-- V1.2 (19-Sep-86) started on Wed Sep 24 1986 at 10:22:15
DTR --I-- Terminated on Wed Sep 24 1986 at 10:22:16
SPAWNER (Version 1.2) listening... on Wed Sep 24 1986 at 10:22:16

SPAWNER exiting... on Wed Sep 24 1986 at 10:22:25
```

2.9.3 Enhancements

The on-line help text has been updated to include the new functionality.

2.10 Changes to the DECnet–Rainbow File Access Listener (FAL)

FAL has implemented the following bug fixes for use with RSTS systems:

- In previous versions of DECnet–Rainbow FAL, wildcard access for DELETE operations from RSTS systems did not work. This has been corrected so that wildcards can be used in file specifications to delete more than one file at a time.
- In previous versions of FAL, DELETE requests from RSTS systems did not include any confirmation. All of the files in the request were deleted, with no chance of changing your mind and saving any of the files. This has been corrected so that any DELETE requests from an RSTS system will first ask for confirmation before a file is deleted.

2.11 Changes to the DECnet–Rainbow Network File Transfer Utility (NFT)

2.11.1 Enhancements

- In DECnet–Rainbow Version 1.1, the Network File Transfer utility (NFT) displayed the following message after successful COPY and APPEND operations:

  [*n* records]

  where *n* represented the number of records copied or appended.

This has been changed so that NFT now reports the number of bytes transferred and the transfer rate. (The transfer rate is computed by using the time interval from a successful OPEN of the remote file through a CLOSE of the remote file. The rate does not include process startup time on the remote system.) The report appears in the following format:

[*n* bytes at *m* bytes/second]

where *n* is the number of bytes and *m* bytes/second is the rate at which the bytes were transferred.

- NFT's performance has improved significantly.
- NFT now prompts you for confirmation of a DELETE request for any of these file specifications:

  "*.*", "*.*;*", or "*.*.*".

  The prompt is in the MS–DOS style:

  ```
  Are you sure (Y/N)?
  ```

2.12 Changes to the DECnet–Rainbow Programming Interface Library (DNETLIB)

2.12.1 Bug Fixes

The Programming Interface Library (DNETLIB) has implemented bug fixes to correct the following problems:

- The *dnet_conn*( ) function has been fixed to properly check error returns on all socket calls.
- The bug in *dnet_getalias*( ) has been fixed to properly check for –1 before attempting to close the DECALIAS.DAT database file.

2.12.2 Enhancements

DNETLIB Version 1.2 includes the following enhancements:

- Socket calls such as *send*( ), *recv*( ), *swrite*( ), and *sread*( ) have been modified for improved performance.
- The function *dnet_getacc*( ) now performs case insensitive comparisons when checking incoming access control information.
- The Break Source utility (BREAKSRC) has been changed to interpret DNETLIB.SRC as a compressed binary file instead of an ASCII file.

- The sources for 3 sample C programs are now included in the DENETLIB.SRC file. The source file names are:

  - LOOP.C (a sample client program)
  - MIRROR.C (a sample server program)
  - TTTXAMPL.C (a sample transparent task-to-task program)

  Appendix B contains copies of these sample programs.

2.12.3 Compiling DNET Library Sources

When certain C compilers are allocating storage for data structures whose members are larger than a "char", they may ordinarily begin this storage on "int" boundaries.

If DNET library sources are compiled in this way, then calls to the DECnet Network Process (DNP) may fail with errors (for example, "Protocol Family not supported"). These errors are due to the fact that the data structures which are being passed down to DNP have some bad bytes. This results in data not being in places where DNP expects them, or in data structures that are not of the supported byte sizes.

To avoid these data errors, if your C compiler has a switch which causes data structures to be packed more tightly, then you should use that compiler and the accompanying switch for compiling the DNET library sources.

2.12.4 Newly Supported Features

Support has been added to the DNET source library to allow building of medium-model and small-model programs. (A medium-model program is a program that contains multiple code segments and one data segment. A small-model program is a program that contains one code segment and one data segment.)

2.13 Changes to DECnet-Rainbow DECnet Network Process (DNP)

2.13.1 Bug Fixes

A bug which formerly prevented asynchronous accept calls from working properly has been fixed.

In the Assembly Language SETSOCKOPT and GETSOCKOPT programming calls for DNP, the size of the *sockopt_dn* data structure is 12 bytes (not 14 bytes as previously mentioned in the *DECnet-DOS Programmer's Reference Manual*).

The *op_optlen* data structure listed in the Data Structure Type Summary for the SETSOCKOPT and GETSOCKOPT calls should read *sop_optlen*.

The *snd_how* data structure is documented under the SHUTDOWN call, and should not be listed in the Data Structure Type Summary for the SETSOCKOPT and GETSOCKOPT calls.

2.13.2 Enhancements

Changes have been made to make DNP more powerful during times when buffers are in short supply.

In previous versions of DECnet–Rainbow, the size of messages used for non-transparent task-to-task programming was limited to 2K (or 2048 bytes). DNP now supports messages of up to 4K (or 4096 bytes).

3 DECnet–VAXmate Changes

3.1 Overview

This chapter describes the changes that have been made from DECnet–VAXmate Version 1.0 to DECnet–VAXmate Version 1.2 (there is no Version 1.1 for this product). The changes are highlighted in the following list, then broken down by components in later sections.

3.2 Highlight of Changes from Version 1.0 of DECnet–VAXmate

- Bug fixes.
- Performance improvements to Network File Transfer (NFT).
- Functional improvements to the Job Spawner and the DECnet Test Receive utility (DTR).
- SETHOST local IBM-style print screen support.
- SETHOST support for DEC national character sets (7-bit) and DEC multinational character sets (8-bit).
- Changes to the DECnet–VAXmate Programming Interface Library sources (DNETLIB) that provide enhanced performance for programming applications that use this library.

The following sections describe changes that have been made to the individual components.

3.3 Changes to the DECnet–VAXmate Installation Procedure (DIP)

The DECnet–VAXmate Installation Procedure (DIP) is an automated procedure that lets you easily install DECnet–VAXmate by responding to a set of questions and making selections from a series of menus. Refer to the *DECnet–VAXmate Installation Guide* for instructions on how to use DIP, while noting the changes and enhancements that are described here.

3.3.1 Enhancements

During installation, DIP now performs the following operations:

- Checks for the proper version of the DIP.DAT/DIP.SAV database and displays an error if it is an incompatible version.
- Provides enhanced performance for copying files from the kit.
- Checks the communication type of the DECnet–VAXmate Parameter File (DECPARM.DAT, if one exists). If the new configuration is for a different communications type, DIP deletes the existing DECPARM.DAT. It creates a new DECPARM.DAT when the network is reloaded and NCP is rerun.
- Runs the new configuration and verification utility automatically upon system reboot.
- Copies all character translation tables and keyboard screen template files automatically from the kit to the specified DECnet database path (if you select SETHOST.EXE to be installed).
- Copies the FIXNVD.EXE file automatically from the kit if you select the Network Device utility (NDU) to be installed. (Refer to Section 3.6 for information about using FIXNVD.EXE with NDU.)
- Displays the current EXECUTOR node name and address if DECnet software is already installed and is running on your system. DIP gives you the option of changing the node name and address at the same time it displays the information.
- Creates and/or modifies the Spawner database file (DECSPAWN.DAT).
- Creates backup copies of the CONFIG.SYS and AUTOEXEC.BAT files before editing them. The backup files have the extension .BAK. For example:

  CONFIG.SYS will have a backup file named CONFIG.BAK

  AUTOEXEC.BAT will have a backup file named AUTOEXEC.BAK

  If you select the Job Spawner, FAL, or DTR during the installation procedure, DIP will also create a backup copy of the DECSPAWN.DAT file. For example:

  DECSPAWN.DAT will have a backup file named DECSPAWN.BAK

3.3.2 Using DIP in a VAXmate/VMS Services Environment

If you are installing DECnet–VAXmate into a VAXmate/VMS Services environment, you should note the following information:

1. DIP assumes that an AUTOEXEC.BAT file already exists on the boot drive (as preconfigured by the VAXmate/VMS Services installation procedure).
2. DIP assumes that the NET START RDR command exists in this AUTOEXEC.BAT file. This is the command that starts up the components that are necessary for running MS–NET software. DIP uses this command as a flag to properly edit the AUTOEXEC.BAT file for a DECnet–VAXmate/MS–NET installation.

3.4 Changes to DECnet–VAXmate SETHOST

3.4.1 Bug Fixes

SETHOST has implemented bug fixes for the following problems:

- In the previous version, using the HOLD SCREEN key while connected over a CTERM link would cause the system to hang. This has been fixed so the HOLD SCREEN now works properly with CTERM.
- In the previous version, if you were logging a session to the disk and the disk became full, the system would hang. This has been fixed so that instead of hanging the system when the disk fills up, SETHOST now closes the file and displays an error message.

3.4.2 Newly Supported Features

SETHOST Version 1.2 includes support for the following:

- The DEC LK250 keyboard.
- Three new switches (see Section 3.4.3)
- DEC national character sets (7-bit) and DEC multinational character sets (8-bit). The following is a detailed list of the character sets that are available with SETHOST:
  - DEC Multinational STD 1 Character Set
  - DEC Multinational STD 2 Character Set
  - United States/Canada Character Set
  - Dutch Character Set
  - Finnish Character Set
  - French Canadian Character Set
  - French Character Set
  - German Character Set
  - Italian Character Set
  - Norwegian/Danish Character Set
  - Spanish Character Set
  - Swedish Character Set
  - Swiss Character Set
  - United Kingdom Character Set

  (You should note that full use of 8-bit character sets is limited due to hardware and operating system constraints.)

  You can select or change character sets from the General setup menu, which is now accessible with the new /SETUP switch. (Refer to the following section which describes the three new switches for SETHOST.)

- Use of 132 columns. To use 132-column support, you must have color enabled on your monitor (the default is monochrome). To change from the default, use the Setup menu to enable color. Then, you can set the terminal characteristic to 132 columns by issuing a terminal command (for example, on VMS systems you would use the command SET TERM/WIDTH = 132). You can easily switch back and forth between screens by using the ALT/F9 key combination.
- The underline character (_) is now visible. Use the Setup menu to change your terminal mode to COLOR ENABLED if you want to display this character. (As an alternative, you can turn up the intensity on your VAXmate workstation monitor to display the underline character.)

3.4.3 New Switches

Three new switches have been added to SETHOST. These switches allow you to:

- display the LAT service name table (similar to the SHOW SERVICES command on a terminal server).
- change the setup for the terminal emulator without having to make a network connection first.
- change the switch character that SETHOST uses for accessing the SETHOST menu.

The switches are:

1. /SERVICE
2. /SETUP
3. /SWITCH = *n*

3.4.3.1 /SERVICE — The /SERVICE switch displays the personal computer LAT driver's service name table. This table lists the services that are available for use by LAT. The services listed are those that have been defined using NCP or those that are advertised on the local area network. You can only connect to a service whose name appears in the service name table.

The default size of the service name table is 10. You can change the size of the table by using the LAT switch, /D:*n*. (The DECnet–VAXmate Installation Procedure places the LAT command in the AUTOEXEC.BAT file without any switches. To add the /D switch and change the value of *n*, edit the AUTOEXEC.BAT file.) The value of *n* is added to the default value of 10. For example, if you specify LAT /D:4, the list displayed by SETHOST /SERVICE can include up to 14 service names.

Once you have seen the list of services that are available, you can connect to any of the services by issuing the SETHOST *service-name* command. (You must be using LAT protocol for the connection.)

3.4.3.2 /SETUP — The /SETUP switch allows you to invoke the setup menus without having to make a network connection. You can change settings in any of the setup directories. Use the cursor arrow keys to move between boxes, then press RET or ENTER to make a selection in each box. When you have completed all of your selections, use the Save Parameters option in the Action menu to save the new defaults.

3.4.3.3 /SWITCH = *n* — The /SWITCH = *n* switch lets you specify a character to use for accessing the SETHOST menu during an active session. By default, the CTRL/4 RET key combination displays this menu. In the previous version of DECnet–VAXmate, the CTRL/\ RET key combination would display this menu. (The CTRL/\ RET combination was difficult to use on European keyboards, so the default was changed to CTRL/4 RET.) Both combinations will display the menu.

NOTE

> **This is a two-step sequence.** In order for CTRL/4 RET to work properly, you must first press CTRL and 4 simultaneously, then release the two keys and press RET.

With /SWITCH = *n*, you can replace the CTRL/4 default with a different character. The variable *n* can be an ASCII decimal code character from 1 to 28 (where 1 is equivalent to CTRL/A, 2 is equivalent to CTRL/B, and so on).

For example:

SETHOST /SWITCH = 2 would change CTRL/4 RET to CTRL/B RET.

Using SETHOST /SWITCH with no argument resets the default of CTRL/4 RET.

NOTE

> When you use on-line help for SETHOST, you will notice that a fourth switch (/KEYBOARD) is also listed. This switch allows users to change keyboards from the default VAXmate (LK250) keyboard. However, it is not necessary to change keyboards when using DECnet–VAXmate because you should already be using the VAXmate keyboard.

3.4.4 Enhancements

SETHOST Version 1.2 includes the following enhancements:

- There is a new keyboard handler. It takes over the keyboard interrupt only for the keys it needs. All other keys are handled or processed by the standard VAXmate MS–DOS operating system. This allows some software that was not compatible with DECnet–VAXmate Version 1.0 to run with DECnet–VAXmate Version 1.2.
- There is now a way to configure SETHOST for international keyboards. To do this, follow these steps:

  1. Install the DOS KEYB program to select the keyboard you want. (The KEYB program is a DOS program that replaces the resident keyboard program. For example, if you wanted to use a French keyboard, you would install the KEYB FR7FR.KEY program.)
  2. Tell SETHOST which character set you wish to use (follow steps 3 through 6).

3. Type SETHOST/SETUP to get into the Setup menu.
4. Use the arrow keys and the RET or ENTER key to get to the General menu.
5. Select the character set you want to use.
6. Save this as the new default by using the Save Setup Parameters option on the Action menu.

- SETHOST provides translation only from the ROM character set to a DEC character set and back again. On the VAXmate workstation, this means that the VAXmate international key sequences will work to generate a VAXmate IBM-compatible character code. This code will then be translated by SETHOST into a DEC character.

  For example:

  - Two-key compose sequences. (To display é, you must first press ´and then press e.)
  - CTRL/ALT/F1 to toggle back to a US English keyboard layout. (Press all 3 keys at the same time.) CTRL/ALT/F2 to toggle back to the country keyboard layout.
  - ALT/*nnn*, where *nnn* is a 3-digit decimal number. This combination can be used to generate any IBM ROM character.

- SETHOST also has a DEC COMPOSE key. However, the operating system (DOS) sees the keys first and may choose to preempt some of the keys for the VAXmate style of international support. If this happens, there are two ways around it:

  1. Use the VAXmate method

     or

  2. Use CTRL/ALT/F1 to switch temporarily back to a US English keyboard, then use the COMPOSE key.

- DEC's multinational STD 2 character set is for use in Norway and other countries that use character set ROMs that are different from the US character set ROMs.

- The character set files that SETHOST uses for translating are stored in the DECnet database directory (for example, C:\DECNET\*.CHR). These are ASCII text files, and they contain two 256-character tables. The first table is the DECVM table. The DEC character looks into this table to find the character it needs, then returns the VAXmate ROM character (which is then displayed on the screen). The second table is the VMDEC table. The VAXmate ROM character looks into this table to find the character that it needs, then returns the DEC character (which is then displayed on the screen).

Since SETHOST uses the VAXmate ROM character set and not a graphic character set for displaying characters, you are limited in the use of certain characters. When you select a character from the character set file, SETHOST tries to determine the closest match to the character you want to display.

- In the General setup menu, leaving or moving out of the character set menu causes the character set file that you selected to be read to the disk. You can make your selection while you are still in this menu, but it will not take effect until you move out of the menu. Once you have left the menu, the file you selected will replace the internal tables that SETHOST uses. If you hear a beep, then that file was not found.

  SETHOST's default table is for DEC multinational character sets with a US keyboard (this also works for 7-bit US keyboards). SETHOST will not beep if the DECM1.CHR or US.CHR files cannot be found; it will just use the default tables.

- Since the VT100 terminal does not have an escape sequence to tell the remote system that SETHOST can use 8-bit characters, you must tell the remote system and the SETHOST terminal emulator to use 8-bit characters (if you select the DEC multinational character set). For example, on a remote VMS system, you would add SET TERM/EIGHT to your LOGIN.COM file.

- For the 7-bit national character sets, some keys will display different characters depending on whether the keyboard is in typewriter mode or in data processing mode. You can switch from one mode to the other using the General setup menu. This allows you to get to characters that would normally be available only with DEC multinational character sets.

  The keys that are affected by this are #,], [, @, \, }, {, |, ~, ', and ^. What these keys change to depends on the character set you select.

- Two files, NVTDEF.DAT and NVTVID.DAT, are created when using SETHOST. These files contain information about the version of SETHOST you are using. The NVTDEF.DAT file is created when you make a keyboard selection or when you use the /SAVE switch. The NVTVID.DAT file is created when you save information using the Save Parameters option in the Setup menu. In the previous version of SETHOST, the file names were SETHOST.DEF and VT102.DAT. The older version of SETHOST will continue to use these files.

  When you start SETHOST, it checks for the proper version number in NVTVID.DAT. If SETHOST determines that this file contains the wrong version, it will beep and ignore the information in the file. You should use the Save Parameters option to replace the NVTVID.DAT file with an updated version that contains the correct information.

- If you are using ALL–IN–1 or WPS–PLUS software on a VT100 terminal, DEC multinational characters are not displayed. In order to get WPS–PLUS software to use the multinational character set (MCS) and to properly display the MCS characters in the on-line help screens, you need to define the following symbols on your VMS system:

  DEFINE KOA$TERMINAL_*xxxx* "OUTPUT_SETHOST"
  DEFINE KOA$TERMINAL_MCS_SETHOST "Y"

  where *xxxx* is the terminal name (such as RTA1, VTA32, and so on).

3.5 Changes to the DECnet–VAXmate Mail Utility

Mail no longer uses the access control information that is saved by NCP.

Mail now adds the DECnet node address of the mail sender to the Subject field, just before the "Reply to" address. In Version 1.0, this address was not included. Now when other users receive mail from your DECnet–VAXmate node, your node's address appears in this field. For example:

```
Subj: Finance Meeting   "From 55.125, Reply to GRAVY::NELSON"
```

3.6 Changes to the DECnet–VAXmate Network Device Utility (NDU)

3.6.1 Bug Fixes

In DECnet–VAXmate Version 1.0, the virtual disk facility was shipped with the following problems:

1. A bug caused all DECnet–VAXmate Version 1.0 32-megabyte disks created as the first NDU function after a system boot (under MS–DOS or PC DOS Version 2 systems) to have an error in their headers. This error does not affect the performance or integrity of the virtual disk facility.
2. 32-megabyte disks created under MS–DOS or PC DOS Version 2 systems are not usable from MS–DOS or PC DOS Version 3 systems. (The opposite of this is also true: 32-megabyte disks created on MS–DOS or PC DOS Version 3 systems are not usable from MS–DOS or PC DOS Version 2 systems.)

DECnet–VAXmate Version 1.2 solves both of these problems with the following:

- All virtual disks created by DECnet–VAXmate Version 1.0 remain usable in all configurations.
- All 1.2-, 10-, and 20-megabyte virtual disks created by DECnet–VAXmate Version 1.0 remain usable in all configurations.

- All 32-megabyte virtual disks created by DECnet–VAXmate Version 1.0 under MS–DOS or PC DOS Version 3 systems will remain usable on any Version 3 system. These disks are not usable from MS–DOS or PC DOS Version 2 systems.
- 32-megabyte virtual disks created by DECnet–VAXmate Version 1.0 under MS–DOS or PC DOS Version 2 systems are not usable with DECnet–VAXmate Version 1.2 **until they are repaired using the FIXNVD utility**. The FIXNVD utility is automatically installed by the DECnet–VAXmate Installation Procedure, DIP. The utility's file name is FIXNVD.EXE.

  If you attempt to open a Version 1.0 virtual disk with Version 1.2, NDU generates the following error message:

  ```
  Cannot OPEN this disk created with an older release of DECnet.
   Run FIXNVD first.
  ```

  To use the FIXNVD utility, first make a copy of your virtual disk file. For example:

  ```
  $ COPY V10DISK1.FIL V10DISKBU.DSK RET
  ```

  Now you can run FIXNVD on the new file. This is the syntax for using FIXNVD:

  FIXNVD *node*[*/user/password/account*] *backup-disk-name*

  Example 1:

  ```
  $ COPY VDISK.DSK DISK.007 RET
  ```

  (on VMS node VMSNOD)

  ```
  C> FIXNVD VMSNOD DISK.007 RET
  ```

  (on your PC)

  Example 2:

  ```
  $ COPY VDISK.DSK DISK$07:[SMITH.DISKS]PAYROLL.DAT RET
  ```

  (on VMS node REMOTE)

  ```
  C> FIXNVD REMOTE/SMITH/HARP DISK$07:[SMITH.DISKS]PAYROLL.DAT RET
  ```

  (on your PC)

FIXNVD will not write to a file unless it is a proper candidate for repair.

NOTE

> FIXNVD will make the files and directories allocated in excess of 33,439,744 bytes inaccessible! In fixing the bug and making Version 2 and Version 3 disks compatible, FIXNVD must shrink the maximum size of 32-megabyte disks. Any files or directories written when the disk volume was filled in excess of 33,439,744 bytes will become inaccessible. Run CHKDSK to determine if this has happened:
>
> CHKDSK D:
>
> If the disk volume is filled, reinstall DECnet–VAXmate Version 1.0 (or just NDU.EXE and NDDRV.SYS from DECnet–VAXmate Version 1.0) and restore the backed-up virtual disk. Copy the files from the backed-up disk to two new virtual disks, then reinstall DECnet–VAXmate Version 1.2.

- All 32-megabyte virtual disks created by DECnet–VAXmate Version 1.2 will be usable in all configurations running DECnet–VAXmate Version 1.2 or later.
- After you issue an NDU DELETE command, NDU now prompts you for confirmation in the MS–DOS style:

  ```
  Are you sure (Y/N)?
  ```

3.7 Changes to DECnet–VAXmate Transparent File Access (TFA)

Transparent file access functions let you open and close a remote file, create or delete a remote file, read from or write records to a remote file, submit a batch job, or search a directory for a specific file or files. Since these functions are performed transparently, you do not need to use the information in this section unless you are performing network programming tasks.

You can also use TFA to type and copy files to and from remote nodes. Use the following format:

TYPE	\\f\node\\filespec
TYPE	\\f\node\\remotefile > *localfile*
COPY	*localfile*\\f\node\\remotefile

You should note that the COPY command only works if the remote file already exists on the remote node. If the file does not exist on the remote node, the COPY operation will fail.

3.7.1 Bug Fixes

TFA has implemented bug fixes for the following problems:

- In the previous version of TFA, a file OPEN operation which specified both read and write access would only allow writing. Now it only allows reading. TFA cannot both read and write a file during a single OPEN/CLOSE session.
- The Transparent Network Test utility (TNT) previously reported a false error if it was run right after TFA was installed. An actual error condition did not exist. This has been fixed so that a false error is no longer reported.
- Read access to a file larger than 64K bytes with null record attributes (such as a large runoff output file) would appear to succeed, but no data would be returned. This has been fixed. Data is now returned, indicating whether or not the operation succeeded.
- Issuing a file close with a handle of –1 indicates that the logical link in use for directory functions be closed. TFA did not check to insure that it was open first. This caused writing to random locations in memory. This has been fixed. TFA will not try to perform directory functions if the logical link is closed.
- If a FIND FIRST FILE function was performed following a FIND FIRST FILE without an intervening CLOSE command (using a handle of –1), an error would occur. The second FIND FIRST FILE function now closes the link automatically.

3.8 Changes to DECnet–VAXmate DTR/DTS

In DECnet–VAXmate Version 1.0, the DECnet Test Receive utility (DTR) would automatically log information to the file DTR.LOG in the DECnet database directory whenever it was run by the Spawner. DTR no longer does this automatically. In DECnet–VAXmate Version 1.2, if you want to log information to the DTR.LOG file you must specify the /LOG switch. For example:

```
C:\DTR /LOG (RET)
```

3.9 Changes to the DECnet–VAXmate Spawner

- There was a previous parsing restriction for DECSPAWN.DAT (the parameter file for the Spawner utility) that would only allow for a single space between the object name or number and the parameter associated with it. This restriction has been removed. There may be multiple spaces and/or tabs between the object name or number and the parameter.

- The Spawner can now support logging, using the /LOG switch. When you use this switch, the Spawner creates a file called SPAWNER.LOG in the DECnet database directory (for example, \DECNET\SPAWNER.LOG). All file accesses are recorded into this file. The log information will include the name of the connecting node, the object number, and the time of the request. The information will not include any actions that occurred as the result of the request.

  For example:

  SPAWNER /LOG

  The following is a sample SPAWNER.LOG file:

```
SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 15:41:52
Connect request from node BOCA for object #17 on Wed Sep 10 1986 at
15:42:02
Executing: fal -use 2... on Wed Sep 10 1986 at 15:42:03
SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 15:42:08
SPAWNER exiting... on Wed Sep 10 1986 at 15:42:14
```

  (In this example, object #17 is FAL, a request to run the File Access Listener.)

- The Spawner can now execute batch files. For example, you can create a BATCH.BAT file that contains the following information:

  ECHO Executing Batch File---Echoing arguments passed:

  Echo %1 %2 %3 %4

  In this sample file, %1, %2, %3, and %4 represent the different arguments you can use (such as ARG1, ARG2, and so on).

  Then, define the following in your DECSPAWN.DAT file:

```
#129          C:\DECNET\BATCH.BAT          ARG1     ARG2
```

  When you run the Spawner, it detects a connect request for object number 129 (# 129) and accepts the request. It then spawns the batch file (BATCH.BAT) with the command line arguments specified in DECSPAWN.DAT (in this case, the arguments are ARG1 and ARG2). The Spawner also passes a socket number to the batch file. The socket appears as –USE *n*, where *n* is the socket number. The batch file can use this socket, or it can ignore it.

After the batch file completes processing and control is returned to the Spawner, the Spawner closes the socket whose number it had previously passed to the batch file. (This is the default. If you do not close the socket from the batch file, the Spawner will automatically close the one that was opened at the start of the batch operation.)

The output looks like this:

```
press '!' to abort
SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 15:49:33

Connect request from node BOCA for object #129 on Wed Sep 10 1986 at
15:49:36

C:\SPAWNERt> echo Executing Batch File---Echoing arguments passed:
Executing Batch File---Echoing arguments passed:

C:\SPAWNERt> Echo ARG1 ARG2 -USE 2
ARG1 ARG2 -USE 2

SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 15:49:40

SPAWNER exiting... on Wed Sep 10 1986 at 15:49:54
```

- The Spawner can also pass command line arguments to the servers. For example, you can now run FAL and include any of the valid FAL qualifiers. The following is a sample DECSPAWN.DAT file which includes command line arguments for FAL:

```
#17          FAL          /E          /B
```

FAL can now be spawned with the arguments /ERROR (/E) and /BINARY (/B). The output looks like this:

```
press '!' to abort
SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 16:01:23

Connect request from node BOCA for object #17 on Wed Sep 10 1986 at
16:01:32

Executing: fal -use 3... on Wed Sep 10 1986 at 16:01:32
                FAL - File Access Listener - Version 1.2

Network Driver Version: 1.2
Current working directory:C:\SPAWNERt
All files will be sent as binary.
Existing files will NOT be overwritten.
No access checking will be done (world has read/write privileges).

FAL running...
DIRECTORY access from BOCA:: for LOCAL""::*.*

SPAWNER (Version 1.2) listening... on Wed Sep 10 1986 at 16:01:37

SPAWNER exiting... on Wed Sep 10 1986 at 16:01:40
```

The following is a sample DECSPAWN.DAT file, requesting that DTR be run using the /LOG switch:

```
#63     DTR     /LOG
```

Now when you run the Spawner, you will see output that looks like this:

```
press '!' to abort
SPAWNER (Version 1.2) listening... on Wed Sep 24 1986 at 10:22:10

Connect request from node MSDOS for object #63 on Wed Sep 24 1986 at
10:22:15

Executing: dtr -use 2... on Wed Sep 24 1986 at 10:22:15

[Appending messages to log file: C:\DECNET\DTR.LOG]

DTR --I-- V1.2 (19-Sep-86) started on Wed Sep 24 1986 at 10:22:15
DTR --I-- Terminated on Wed Sep 24 1986 at 10:22:16
SPAWNER (Version 1.2) listening... on Wed Sep 24 1986 at 10:22:16

SPAWNER exiting... on Wed Sep 24 1986 at 10:22:25
```

3.9.1 Enhancements

The on-line help text has been updated to include the new functionality.

3.10 Changes to the DECnet–VAXmate File Access Listener (FAL)

FAL has implemented the following bug fixes for use with RSTS systems:

- In the previous version of DECnet–VAXmate FAL, wildcard access for DELETE operations from RSTS systems did not work. This has been corrected so that wildcards can be used in file specifications to delete more than one file at a time.
- In the previous version of FAL, DELETE requests from RSTS systems did not include any confirmation. All of the files in the request were deleted, with no chance of changing your mind and saving any of the files. This has been corrected so that any DELETE requests from an RSTS system will first ask for confirmation before a file is deleted.

3.11 Changes to DECnet–VAXmate Network File Transfer (NFT)

3.11.1 Enhancements

- In DECnet–VAXmate Version 1.0, the Network File Transfer utility (NFT) displayed the following message after successful COPY and APPEND operations:

  [*n* records]

  where *n* represented the number of records copied or appended.

This has been changed so that NFT now reports the number of bytes transferred and the transfer rate. (The transfer rate is computed by using the time interval from a successful OPEN of the remote file through a CLOSE of the remote file. The rate does not include process startup time on the remote system.) The report appears in the following format:

[*n* bytes at *m* bytes/second]

where *n* is the number of bytes and *m* bytes/second is the rate at which the bytes were transferred.

- NFT's performance has improved significantly.
- NFT now prompts you for confirmation of a DELETE request for any of the following file specifications:

  "*.*", "*.*;*", or "*.*.*".

  The prompt is in the MS–DOS style:

  ```
  Are you sure (Y/N)?
  ```

3.12 Changes to the DECnet–VAXmate Programming Interface Library (DNETLIB)

3.12.1 Bug Fixes

The Programming Interface Library (DNETLIB) has implemented bug fixes for the following problems:

- The *dnet_conn*( ) function has been fixed to properly check error returns on all socket calls.
- The bug in *dnet_getalias*( ) has been fixed to properly check for –1 before attempting to close the DECALIAS.DAT database file.

3.12.2 Enhancements

DNETLIB Version 1.2 includes the following enhancements:

- Socket calls such as *send*( ), *recv*( ), *swrite*( ), and *sread*( ) have been modified for improved performance.
- The function *dnet_getacc*( ) now performs case insensitive comparisons when checking incoming access control information.
- The Break Source utility (BREAKSRC) has been changed to interpret DNETLIB.SRC as a compressed binary file instead of an ASCII file.

- The sources for 3 sample C programs are now included in the DNETLIB.SRC file. The source file names are:
  - LOOP.C (a sample client program)
  - MIRROR.C (a sample server program)
  - TTTXAMPL.C (a sample transparent task-to-task program)

  Appendix B contains copies of these sample programs.

3.12.3 Compiling DNETLIB Sources

When certain C compilers are allocating storage for data structures whose members are larger than a ''char'', they may ordinarily begin this storage on ''int'' boundaries.

If DNET library sources are compiled in this way, then calls to the DECnet Network Process (DNP) may fail with errors (for example, ''Protocol Family not supported''). These errors are due to the fact that the data structures which are being passed down to DNP have some bad bytes. This results in data not being in places where DNP expects them, or in data structures that are not of the supported byte sizes.

To avoid these data errors, if your C compiler has a switch which causes data structures to be packed more tightly, then you should use that compiler and the accompanying switch for compiling the DNET library sources.

3.12.4 Newly Supported Features

Support has been added to the DNET source library to allow building of medium-model and small-model programs. (A medium-model program is a program that contains multiple code segments and one data segment. A small-model program is a program that contains one code segment and one data segment.)

3.13 Changes to the DECnet–VAXmate DECnet Newtork Process (DNP)

A bug which formerly prevented asynchronous accept calls from working properly has been fixed.

In the Assembly Language SETSOCKOPT and GETSOCKOPT programming calls for DNP, the size of the *sockopt_dn* data structure is 12 bytes (not 14 bytes as previously mentioned in the *DECnet–DOS Programmer's Reference Manual*).

The *op_optlen* data structure listed in the Data Structure Type Summary for the SETSOCKOPT and GETSOCKOPT calls should read *sop_optlen*.

The *snd_how* data structure is documented under the SHUTDOWN call, and should not be listed in the Data Structure Type Summary for the SETSOCKOPT and GETSOCKOPT calls.

In the previous version of DECnet–VAXmate, the size of messages used for non-transparent task-to-task programming was limited to 2K (or 2048 bytes). DNP now supports messages of up to 4K (or 4096 bytes).

A
Keyboard Illustrations

This appendix contains illustrations of the following keyboards:

- Digital's Rainbow 100 System keyboard
- Digital's VAXmate (LK250) keyboard
- IBM's PC and PC/XT keyboard
- IBM's Personal Computer AT keyboard
- IBM's Enhanced Personal Computer keyboard

These illustrations are included for reference purposes.

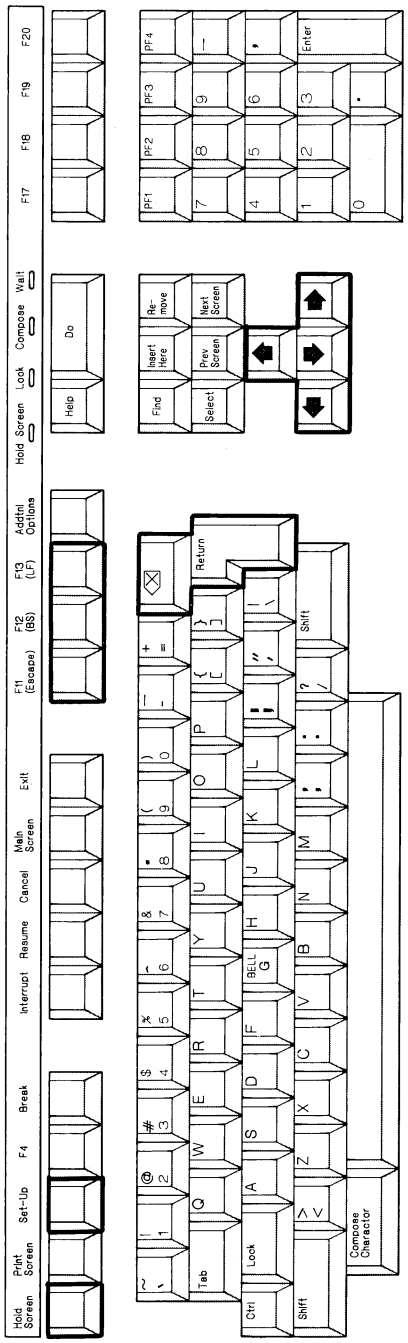

Figure A–1: The Rainbow 100 System Keyboard

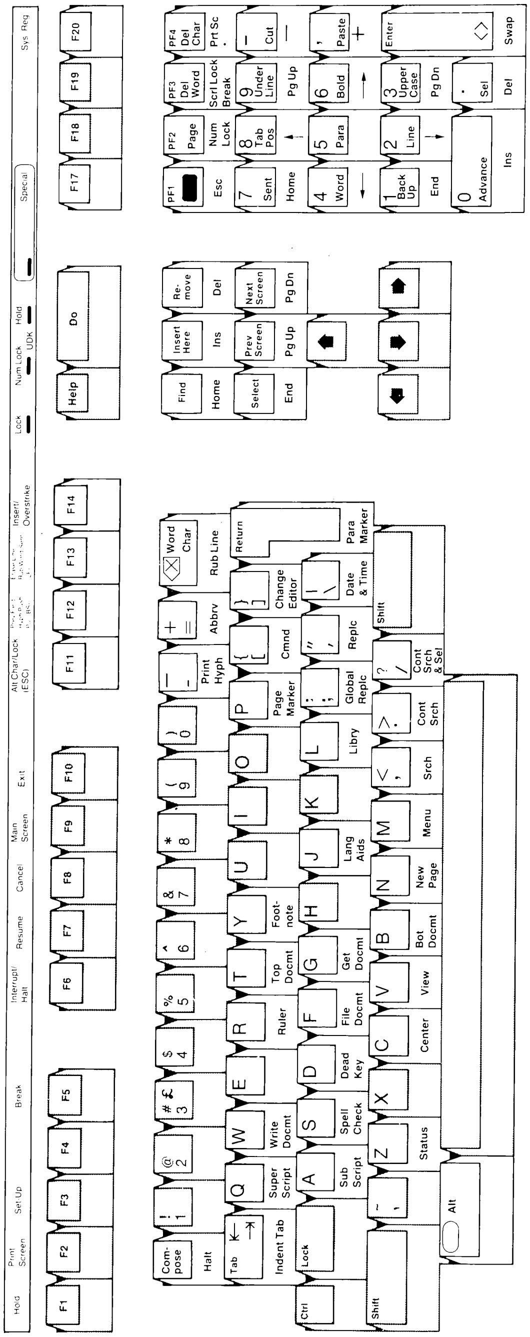

Figure A–2: The VAXmate (LK250) Keyboard

Table A-1: Key Mappings (IBM PC/XT Keyboard to DEC VT102-Type Keyboard)

IBM PC or PC/XT Key (or Key Combination)	DEC VT102-Type Key
F1	Find
F2	Select
F3	Insert Here
F4	Remove
F5	Previous Screen
F6	Next Screen
F7, F8, F9, F10	↑, ↓, ←, → (arrow keys)
ALT/F1	Hold Screen
ALT/F2	Print Screen
ALT/F3	Setup
ALT/F9	Flips screen = alternates between 80 columns and 132 columns (color only)
ALT/F10	Compose
Backspace	Delete
Enter	Return

Table A–2: Keypad Mappings (IBM PC/XT Keypad Keys to DEC VT102–Type Keypad Keys)

IBM PC or PC/XT Keypad Key	DEC VT102–Type Keypad Key	DEC VT102–Type Shifted Keypad Key
Num Lock	PF1	PF2
Scroll Lock	PF3	PF4
Minus –	Minus –	Comma ,
Plus +	Enter	Do (DEC F16)
. Del	. Del	Remove
0 Ins	0	Insert Here
1 End	1	Select
2↓	2	↓
3 Pg Dn	3	Next Screen
4 ←	4	←
5	5	Help
6 →	6	→
7 Home	7	Find
8 ↑	8	↑
9 Pg Up	9	Previous Screen

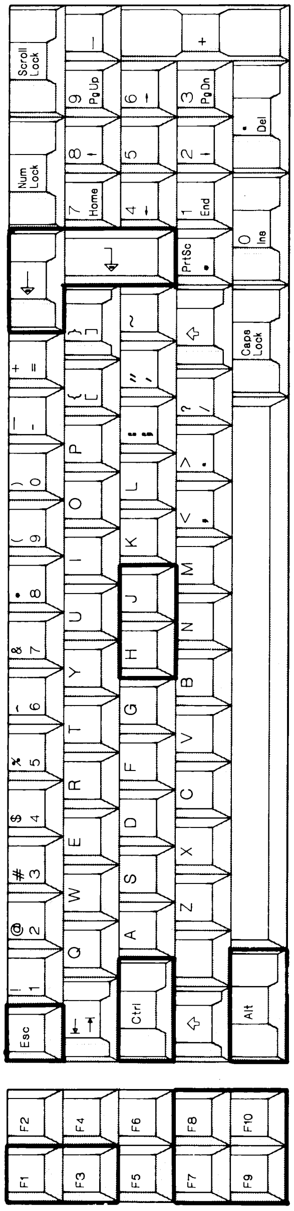

Figure A–3: The IBM PC and PC/XT Keyboard

Table A-3: Key Mappings (IBM PC AT Keyboard to DEC VT102-Type Keyboard)

IBM PC AT Key (or key Combination)	DEC VT102-Type Key
F1	Find
F2	Select
F3	Insert Here
F4	Remove
F5	Previous Screen
F6	Next Screen
F7, F8, F9, F10	↑, ↓, ←, → (arrow keys)
ALT/F1	Hold Screen
ALT/F2	Print Screen
ALT/F3	Setup
ALT/F9	Flips screen = alternates between 80 columns and 132 columns (color only)
ALT/F10	Compose
Backspace	Delete
Enter	Return

Table A-4: Keypad Mappings (IBM PC AT Keypad Keys to DEC VT102-Type Keypad Keys)

IBM PC AT Keypad Key	DEC VT102-Type Keypad Key	DEC VT102-Type Shifted Keypad Key
Esc	PF1	Esc
Num Lock	PF2	
Scroll Lock	PF3	
Sys Req	PF4	
Minus -	Comma ,	
Multiply *	Minus -	
Plus ▢	Enter	Do (DEC F16)
. Del	.	Remove
0 Ins	0	Insert Here
1 End	1	Select
2 ↓	2	↓
3 Pg Dn	3	Next Screen
4 ←	4	←
5	5	Help
6 →	6	→
7 Home	7	Find
8 ↑	8	↑
9 Pg Up	9	Previous Screen

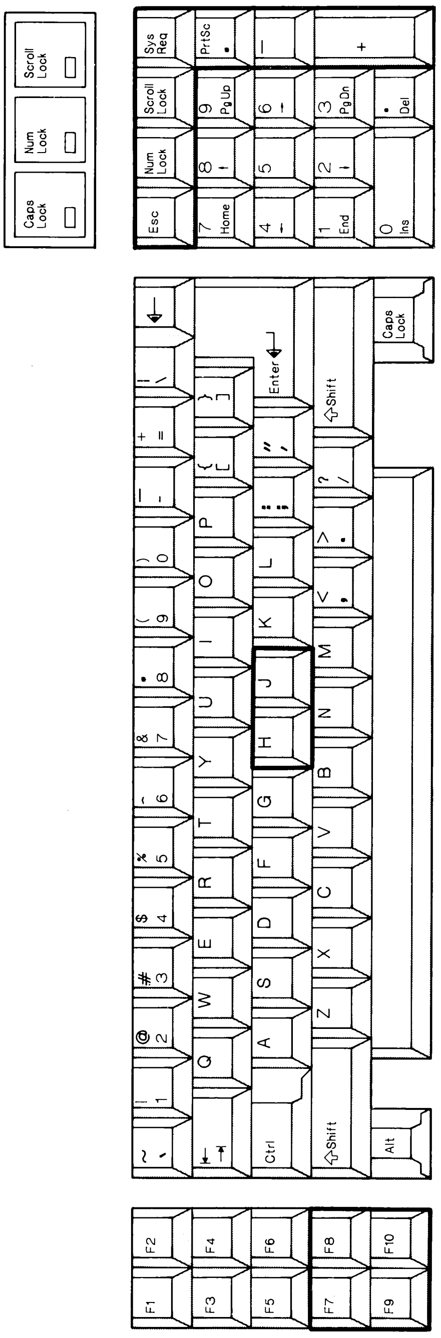

Figure A–4: The IBM PC AT Keyboard

Table A–5: Key Mappings (IBM Enhanced PC Keyboard to DEC VT102–Type Keyboard)

IBM Enhanced PC Keyboard Key	DEC VT102–Type Key	DEC VT102–Type Shifted Key
Backspace	Delete	
Enter	Return	
Print Screen	Print Screen	
Scroll Lock	Do	Help
Pause	Hold Screen	
Insert	Insert Here	
Delete	Remove	
Home	Find	
End	Select	
Page Up	Previous Screen	
Page Down	Next Screen	
↑, ↓, ←, → (arrow keys)	↑, ↓, ←, → (arrow keys)	
F1 through F10	F1 through F10	F11 through F20
F11	F11	F13
F12	F12	F14

Table A–6: Keypad Mappings (IBM Enhanced PC Keypad Keys to DEC VT102–Type Keypad Keys)

IBM Enhanced PC Keypad Keys	DEC VT102–Type Keypad Key	DEC VT102–Type Shifted Keypad Key
Num Lock	PF1	
Slash /	PF2	
Multiply *	PF3	
Minus –	PF4	
Plus +	Comma ,	Minus –
Enter	Enter	
. Del	. Del	
0 through 9	0 through 9	

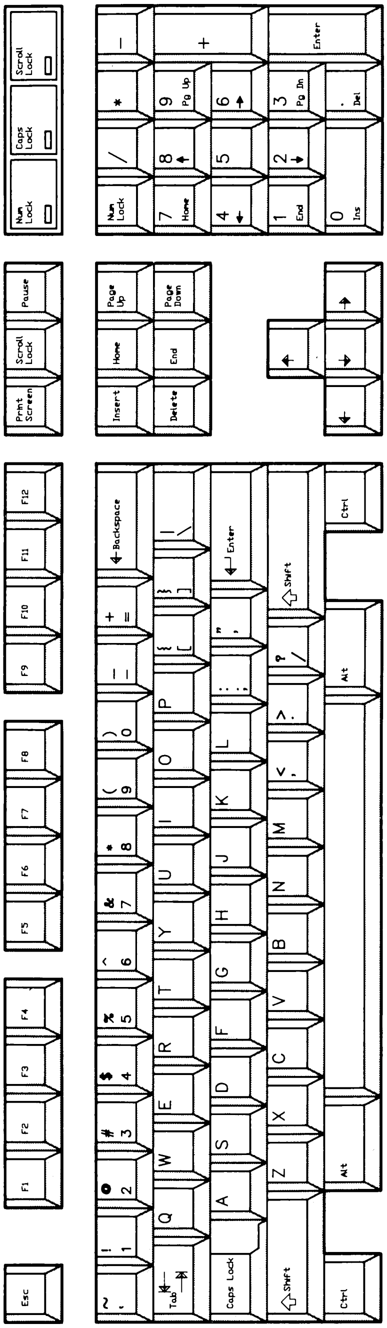

Figure A-5: The IBM Enhanced Personal Computer Keyboard

B Sample C Programs

B.1 Sample Client Program

```
/*
 * Program - LOOP
 *
 *   o DECnet-DOS loop test example task.
 *   o Can be used to attempt to connect to any remote DECnet
 *     object (works well with DECnet MIRROR object, #25).
 */

#include <stdio.h>

/*
 * User defined symbols for conditional compilation.
 */
#include "dnprefix.h"

/*
 * Include some network interface headers.
 */
#include "types.h"  /* Type definitions, abstract data types. */
#include "time.h"   /* Time data structures. */
#include "dn.h"     /* Network data structures and */
                    /* definitions. */
#include "socket.h" /* Socket interface layer definitions. */
#include "sioctl.h" /* Socket I/O control functions. */
#include "errno.h"  /* Global user error definitions returned */
                    /* in 'errno'. */
/*
 * Conditionalize for DECnet-ULTRIX compatibility.
 */
#ifndef MSDOS
#define sclose(s) close(s)
#define sioctl(s,f,a) ioctl(s,f,a)
#endif
```

```
#define SEQUENCED_PACKET 0
#define STREAM           1

/*
 * Version string.
 */
static char        version[] = "X1.16.0";

/*
 * Main line code.
 */
main(argc, argv)
int     argc;
char    *argv[];
{
    /*
     * Local data.
     */
    struct  timeval tmv;
    char    *node, *object;
    u_char  optional_send[16];
    u_char  optional_receive[16];
    u_char  data_buffer[10];
    field32 readfds, writefds;
    int     rec_len;ems;
    int     sock_type;
    int     sock;
    int     loop;
    int     count;
    int     len;
    int     ind0;
    char    bio[1];

    /*
     * Display our current version.
     */
    printf("\tLOOP - Example Client Program - Version %s\n",
            &version[0]);

    /*
     * Make sure there are a valid number of input arguments.
     */
    if (argc < 3)
    {
        printf("Usage: loop <node name or address>\
 <#objnum or objnam>\n");
         exit(1);
    }

    /*
     * Set up optional data to send with connect.
     */
    strcpy(&optional_send[0], "hello");

    /*
     * Attempt to connect to the object on the remote node.
     */
    rec_len = sizeof(optional_receive);
    node = *++argv;
    object = *++argv;
    sock_type = SEQUENCED_PACKET;
    printf("connecting to node \"%s\", object \"%s\"\n",
           node, object);
    if ((sock = dnet_conn(node, object, sock_type,
         &optional_send[0],
                strlen("hello"),
                &optional_receive[0], &rec_len)) < 0)
    {
      nerror("dnet_conn");
      exit(1);
    }
    printf("connect complete with node \"%s\",\
object \"%s\"\n", node, object);

    /*
     * Check for returned optional data.
     */
    if (rec_len)
    {
        puts("optional data received:");
        for (ind0 = 0; ind0 < rec_len; ind0++)
        {
            printf(" %d", optional_receive[ind0]);
        }
        puts("");
    }

    /*
     * Fill a data buffer with dummy data.
     */
    for (loop = 0; loop < sizeof(data_buffer); loop++)
    {
        data_buffer[loop] = loop;
    }

    /*
     * Try to send a dummy data buffer 10 times
     * to target object as long as link is still active.
     */
    loop = 10;
    while (loop--)
    {
        if (dnet_eof(sock) == 1)
        {
            printf("link is down.\n");
            sclose(sock);
            exit(1);
         }

        if ((count = send(sock, &data_buffer[0],
                          sizeof(data_buffer), 0)) < 0)
        {
            perror("write");
            sclose(sock);
            exit(1);
        }
    }
    printf("data successfully sent to %s\n", node);
```

```
/*
 * Now set the socket to nonblocking mode.
 */
bio[0] = 1;
sioctl(sock, FIONBIO, &bio[0]):
/*
 * Clean out the data buffer.
 */
bzero(&data_buffer[0], sizeof(data_buffer));

/*
 * Continue to receive data from target object until
 * disconnected.
 */
while (1)
{
    /*
     * Check if link is still active.
     */
    if (dnet_eof(sock) == 1)
    {
        printf("link is down.\n");
        sclose(sock);
        exit(1);
    }

    /*
     * Now check to see if the socket has data available
     * to read and timeout after 15 seconds.
     */
    readfds = 1 << sock;
    tmv.tv_sec = 15;
    if ((ind0 = select(sock + 1, &readfds, 0, 0, &tmv)) < 0)
    {
            perror("select");
    }
    else
    {
        if (ind0 == 0)
        {
            printf("receive wait timed out.\n");
            sclose(sock);
            exit(1);
        }
    }

    if ((count = recv(sock, &data_buffer[0],
                      sizeof(data_buffer), 0)) < 0)
    {
        if (errno != EWOULDBLOCK)
        {
            perror("read");
            break;
        }
        else
            continue;
    }
    printf("data received (%d bytes):\n", count);
    for (ind0 = 0; ind0 < count; ind0++)
    {
```

```
                printf(" %d", data_buffer[ind0] );
            }
            puts("");
        }

        /*
         * Finish up. Make the socket linger on close to allow
         * things to get cleaned.
         */
        if (setsockopt(sock, SOL_SOCKET, SO_LINGER, 0, 0) < 0)
        {
            perror("setsockopt");
        }

        /*
         * Close the socket and exit program.
         */
        sclose(sock);
        exit(0);
}
```

B.2 Sample Server Program

```
/*
 * Program - MIRROR
 *
 *   o DECnet-DOS, mirror server, DECnet object 25
 *   o Can run as a stand-alone task OR by the Job Spawner.
 *       . Copy MIRROR.EXE into a directory in the DOS path
 *       . Make this entry in the DECSPAWN.DAT file
 *              "#25 mirror"
 *       . Run the Job Spawner, for example: SPAWNER.EXE
 *   o Example use would be to run an NTU LOOP NODE from another
 *     DECnet-DOS node to the node which is running this server
 *     program.
 */

#include <stdio.h>
#include "types.h"
#include "dnmsdos.h"
#include "dn.h"
#include "socket.h"
#include "time.h"
#include "errno.h"
#include "scbdef.h"

#define MAX_BUF_SIZE 2048     /* maximum loop data buffer */

struct sockaddr_dn sockaddr; /* accept connect data structure */
struct optdata_dn opt;       /* optional data buffer */
char buff[MAX_BUF_SIZE];     /* data buffer */
int lsock = -1;              /* incoming connections on */
                             /* listening socket */
int sock = -1;               /* data communications socket */
char mode[1];                /* accept mode */
int spawned = 0;             /* if non-zero, we are child of SPAWNER */
char mir_version[] = "X1.16.0";
```

```
/*
 * Sample DECnet-DOS server task. This task will bind itself
 * as DECnet object number 25, the standard DECnet object
 * reserved for a mirror task. When started, the mirror is the
 * only running task. To terminate, the user may
 * press '!'.
 */
main(argc, argv)
int argc;
char **argv;
{
    extern char *malloc();
    extern char *dnet_ntoa();
    int len;
    int nfds;
    unsigned long read;
    struct timeval tmv;

    /*
     * Check to see if we are child of SPAWNER.
     */
    if (argc > 1)
        spawned = mir_spawned(++argv);


    if (!spawned)
    {
        /*
         * Set up listening socket for incoming connect requests.
         */
        if ((lsock = socket(AF_DECnet, SOCK_SEQPACKET, 0)) < 0)
           mir_exit("socket failed", errno);

        /*
         * Bind task to DECnet object 25.
         */
        bzero(&sockaddr, sizeof(sockaddr));
        sockaddr.sdn_family = AF_DECnet;
        sockaddr.sdn_objnum = 25;
        if (bind(lsock, &sockaddr, sizeof(sockaddr)) < 0)
            mir_exit("bind failed", errno);

        /*
         * Set up listening socket to listen for incoming connect
         * requests.  Allow for up to 5 pending incoming
         * connect requests.
         */
        if (listen(lsock, 1) < 0)
            mir_exit("listen failed", errno);

        /*
         * Display 'how to exit' message.
         */
        printf("\npress '!' to abort");
    }

    /*
     * Set up peer information for display if run by SPAWNER.
     */
    if (spawned)
    {
        len = sizeof(sockaddr);
        if (getpeername(sock, &sockaddr, &len) < 0)
            mir_exit("get peer information failed", errno);
    }

    /*
     * If not spawned, listen for incoming connect requests until
     * there is keyboard input.
     */
    for (;;)
    {
        if (!spawned)
        {
            /*
             * Poll listening socket for incoming connect request.
             */
            printf("\nMIRROR (Version %s) listening...",
                    mir_version);

            for (;;)
            {
                if (mir_keyboard_input())
                    mir_exit(NULL, 0);

                bzero(&tmv, sizeof(tmv));
                read = 1 << lsock;
                nfds = lsock + 1;
                if (select(nfds, &read, 0, 0, &tmv) > 0)
                {
                    if (read & (1 << lsock))
                        break;
                }
            }

            /*
             * Issue a deferred accept on the connect request - send
             * some optional data along with it.
             */
            mode[0] = ACC_DEFER;
            if (setsockopt(lsock, DNPROTO_NSP, DSO_ACCEPTMODE,
                           &mode[0], sizeof(mode)) < 0)
            {
                 mir_exit("set accept mode", 1);
            }

            len = sizeof(sockaddr);
            if ((sock = accept(lsock, &sockaddr, &len)) < 0)
                mir_exit("accept failed", errno);
        }

        if (spawned)
            printf("\n\t\tMIRROR - Version %s", mir_version);

        /*
         * Set up outgoing optional data - maximum mirror
         * data buffer size.
         */
        bzero(&opt, sizeof(opt));
        opt.opt_optl = sizeof(unsigned short);
```

```
*(unsigned short *)&opt.opt_data[0] = MAX_BUF_SIZE;
if (setsockopt(sock, DNPROTO_NSP, DSO_CONDATA, &opt,
    sizeof(opt)) < 0)
{
    mir_exit("set socket option - optional data",
             errno);
}

if (setsockopt(sock, DNPROTO_NSP, DSO_CONACCEPT,
    0, 0) < 0)
{
    mir_exit("set connect accept", 1);
}

/*
 * Display peer information.
 */
printf("\n\nLoop connect request from node: %s",
       dnet_ntoa(&sockaddr.sdn_add));

if (sockaddr.sdn_objnum == 0)
    printf("\nRequesting object name: %s",
           &sockaddr.sdn_objname[0]);
else
    printf("\nRequesting object number: %d",
           sockaddr.sdn_objnum);
printf("\n");

/*
 * Loop data while link is still active and other end is
 * still sending data.
 */
while(!dnet_eof(sock))
{
    len = MAX_BUF_SIZE;
    len = sread(sock, buff, &len);
    if (len == 0)
    {
        if (dnet_eof(sock))
            break;
    }
    else
    {
       if (len < 0)
           mir_exit("sread", 1);
    }

    if (buff[0] != 0)
    {
        buff[0] = -1;
        len = 1;
    }
    else
        buff[0] = 1;

    if (swrite(sock, buff, len) < 0)
        mir_exit("swrite", 1);
}
```

```
            /*
             * If child of SPAWNER, we are finished.
             */
            if (spawned)
                mir_exit(NULL, 0);

            /*
             * Finished with current data socket, close it up.
             */
            if (sock != -1)
                sclose(sock);
        }
}
int mir_keyboard_input()
{
    SCB scb;

    scb.AH = SCBC_DCIO;
    scb.DX = 0xff;
    msdos(&scb);
    return (scb.AL == '!');
}
mir_exit(sp, err)
char *sp;
int err;
{
    if (sp != NULL)
    {
        strcpy(buff, "\nmirror - ");
        strcat(buff, sp);
        perror(buff);
    }

    if (lsock != -1)
        sclose(lsock);

    if (sock != -1)
        sclose(sock);

    exit(err);
}
int mir_spawned(cp)
char **cp;
{
    int status = 0;

    upper(*cp);
    if (strcmp(*cp, "-USE") == 0)
    {
        status++;
        sock = atoi(*++cp);
    }
    return(status);
}
```

B.3 Sample Transparent Task-to-Task Program

```
/*
 * Program - TTTXAMPL
 *
 * Sample client program written in C that shows Transparent
 * Task-to-Task using DECnet-DOS.
 *
 * When running this program, the command line argument should
 * look like a network task specification. See the following
 * examples as well as examples cited in the documentation:
 * For example:
 *
 *      \\t\pcdos\\#100(to connect by object number)
 *      \\t\pcdos\smith\xxxxx\\TIMESRV(to connect by object name)
 *
 * After getting a handle (for example, by connecting to a
 * remote object), an attempt is made to write/send some data to
 * the object and then close the handle.
 *
 * o    All C include files and external functions are
 *      distributed with the DECnet-DOS kit in the file
 *      DNETLIB.SRC.
 *
 * o    When attempts to run this program fail, run the utility
 *      TNT.EXE shipped with the DECnet-DOS kit to examine
 *      DECnet errors.
 *
 */
#include        <stdio.h>
#include        "types.h"
#include        "scbdef.h"
#include        "errno.h"

static char buf[100];
static char err_msg[] = "(run TNT.EXE to examine network error)";
static char version[] = "X1.16.0";

/*
 * Function(s) included in DNETLIB.SRC
 */
extern int hopen();
extern int hwrite(), hclose();

main(argc, argv)
int argc;
char *argv[];
{
    int i;
    int j;
    int len;
    int h = 0;

    /*
     * Display version banner.
     */
  printf("\tTTTXAMPL - Transparent Task-to-Task Example - Version %s\n",
           version);
```

```
if (argc < 2)
{
    printf("Usage: tttxampl <TTT_network_task_string>\n");
    printf("\n example:\n");
    printf("\t tttxampl \\\\t\\pygmy\\\\<#object_number>\n");
    printf("\t or\n");
    printf("\t tttxampl \\\\t\\pygmy\\\\<object_name>\n");
    exit(1);
}

/*
 * Fill a dummy data buffer.
 */
for (i = 0; i < sizeof(buf); i++)
     buf[i] = i;

/*
 * Open file (access remote network object).
 */
h = hopen(argv[1], SCBC_HOPEN);
if (h == ERROR)
{
    perror("\nopen");
    printf("\n %s", err_msg);
    exit(1);
}
printf("\nopen succeeded  handle: %u (connected to object)",
        h);

/*
 * Write to file (send data to remote object).
 */
if (hwrite(h, &buf[0], sizeof(buf)) != sizeof(buf))
{
    perror("\nwrite");
    printf("\n %s", err_msg);
}
else
    printf("\nwrite succeeded (sent data to object)");

/*
 * Read from file handle (receive data from object, if any).
 */
len = hread(h, &buf[0], sizeof(buf));
if (len < 0)
{
    perror("\nread");
    printf("\n %s", err_msg);
}
else
{
printf("\nread %u byte(s) (received from object)\n",
            len);
    for (i = j = 0; i < len; i++, j++)
    {
        if (j > 9)
        {
            printf("\n");
            j = 0;
        }
```

```
                printf(" %4u", buf[i]);
            }
        }

        /*
         * Close file handle (disconnect link).
         */
        hclose(h);

        printf("\nfinished.");
        exit(0);
}
```

Index

D

DECnet–DOS changes, 1–1
  Data Link layer
    DEPCA driver, 1–19
    MICOM driver, 1–19
    3COM driver, 1–19
  Data Link layer (DLL), 1–19
  DECnet Network Process (DNP), 1–20
  DLL switch, 1–3
  File Access Listener (FAL), 1–17
  FIXNVD.EXE, 1–3, 1–12 to 1–13
  highlights, 1–1
  Installation Procedure (DIP), 1–2
  Mail utility, 1–11
  Network Device utility (NDU), 1–11
  Network File Transfer utility (NFT), 1–17
  Programming Interface Libarary (DNETLIB), 1–18
    compiling sources, 1–18
  Scheduler, 1–20
  SETHOST, 1–3
    enhancements, 1–8
      character set code, 1–9
      character set files, 1–9
      COMPOSE key, 1–9
      configuring international keyboards, 1–8
      new keyboard handler, 1–8
      new keyboard templates, 1–8
      NVTDEF.DAT file, 1–10

DECnet–DOS changes (Cont.)
      NVTVID.DAT file, 1–10
      using 7-bit characters, 1–10
      using 8-bit characters, 1–10
      using ALL–IN–1 software, 1–10
      using WPS–PLUS software, 1–10
    national and multinational character sets, 1–5
    new switches, 1–5
      /KEYBOARD, 1–5
      /SERVICE, 1–6
      /SETUP, 1–6
      /SWITCH, 1–7
    132-column support, 1–4
  Spawner, 1–14
    batch file support, 1–15
    command line argument support, 1–16
    logging support, 1–14
  Test Receive utility (DTR/DTS), 1–14
  Transparent File Access (TFA), 1–13
DECnet–Rainbow changes, 2–1
  DECnet Network Process (DNP), 2–12
  File Access Listener (FAL), 2–10
  FIXNVD.EXE, 2–2, 2–5 to 2–6
  highlights, 2–1
  Installation Procedure (DIP), 2–1
  Mail utility, 2–4
  Network Device utility (NDU), 2–4
  Network File Transfer utility (NFT), 2–10
  Programming Interface Library (DNETLIB), 2–11

DECnet–Rainbow changes (Cont.)
  compiling sources, 2–12
 SETHOST, 2–2
  enhancements, 2–3
   NVTDEF.DAT file, 2–3
   NVTVID.DAT file, 2–3
   using ALL–IN–1 software, 2–4
   using WPS–PLUS software, 2–4
  new switch
   /SWITCH, 2–3
  132-column support, 2–2
  8-bit multinational character support, 2–2
 Spawner, 2–7
  batch file support, 2–8
  command line argument support, 2–9
  logging support, 2–8
 Test Receive utility (DTR/DTS), 2–7
 Transparent File Access (TFA), 2–6
DECnet–VAXmate changes, 3–1
 DECnet Network Process (DNP), 3–17
 File Access Listener (FAL), 3–15
 FIXNVD.EXE, 3–2, 3–10 to 3–11
 highlights, 3–1
 Installation Procedure (DIP), 3–2
 Mail utility, 3–9
 Network Device utility (NDU), 3–9
 Network File Transfer (NFT), 3–15
 Programming Interface Library (DNETLIB), 3–16
  compiling sources, 3–17
 SETHOST, 3–3
  enhancements, 3–6
   character set code translation, 3–7
   character set files, 3–7, 3–8
   COMPOSE key, 3–7
   configuring internationnal keyboards, 3–6
   new keyboard handler, 3–6
   NVTDEF.DAT file, 3–8
   NVTVID.DAT file, 3–8
   using 7-bit characters, 3–8
   using 8-bit characters, 3–8
   using ALL–IN–1 software, 3–9
   using WPS–PLUS software, 3–9
  national and multinational character sets, 3–4
  new switches, 3–5
   /SERVICE, 3–5

DECnet–VAXmate changes (Cont.)
   /SETUP, 3–5
   /SWITCH, 3–6
  using 132-column support, 3–4
 Spawner, 3–12
  batch file support, 3–13
  command line argument support, 3–14
  logging support, 3–13
 Test Receive utility (DTR/DTS), 3–12
 Transparent File Access (TFA), 3–11

HOW TO ORDER ADDITIONAL DOCUMENTATION

DIRECT TELEPHONE ORDERS

In Continental USA and Puerto Rico call 800–258–1710	In Canada call 800–267–6146	In New Hampshire Alaska or Hawaii call 603–884–6660

ELECTRONIC ORDERS (U.S. ONLY)

Dial 800–DEC–DEMO with any VT100 or VT200
compatible terminal and a 1200 baud modem.
If you need assistance, call 800–DEC–INFO.

DIRECT MAIL ORDERS (U.S. and Puerto Rico*)

DIGITAL EQUIPMENT CORPORATION
P.O. Box CS2008
Nashua, New Hampshire 03061

DIRECT MAIL ORDERS (Canada)

DIGITAL EQUIPMENT OF CANADA LTD.
940 Belfast Road
Ottawa, Ontario, Canada K1G 4C2
Attn: A&SG Business Manager

INTERNATIONAL

DIGITAL
EQUIPMENT CORPORATION
A&SG Business Manager
c/o Digital's local subsidiary
or approved distributor

Internal orders should be placed through the Software Distribution Center (SDC),
Digital Equipment Corporation, Northboro, Massachusetts 01532

*Any prepaid order from Puerto Rico must be placed
with the Local Digital Subsidiary:
809–754–7575

DECnet–DOS
Supplemental Information
AA–JH72A–TV

READER'S COMMENTS

What do you think of this manual? Your comments and suggestions will help us to improve the quality and usefulness of our publications.

Please rate this manual:

	Poor				Excellent
Accuracy	1	2	3	4	5
Readability	1	2	3	4	5
Examples	1	2	3	4	5
Organization	1	2	3	4	5
Completeness	1	2	3	4	5

Did you find errors in this manual? If so, please specify the error(s) and page number(s).

General comments:

Suggestions for improvement:

Name ______________ Date ______________
Title ______________ Department ______________
Company ______________ Street ______________
City ______________ State/Country ______________ Zip Code ______________

DO NOT CUT FOLD HERE AND TAPE

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY LABEL
FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

digital

Networks and
Communications Publications
550 King Street
Littleton, MA 01460–1289

DO NOT CUT FOLD HERE

CUT ALONG DOTTED LINE