MicroVMS Release Notes, Version 4.6

Order Number: AA-KN07A-TN

June 1987

This document describes Version 4.6 of the MicroVMS operating system. It lists and discusses system changes, new features, corrected problems, and restrictions in the use of the system. It also describes changes and corrections to the MicroVMS documentation set.

Revision/Update Information:	This is a new manual, which adds to and corrects information contained in the *MicroVMS Release Notes, Version 4.5*, Order Number AA-JG66A-TN, published September, 1986.
Operating System and Version:	MicroVMS Version 4.6
Software Version:	MicroVMS Version 4.6

digital equipment corporation
maynard, massachusetts

June 1987

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by Digital Equipment Corporation or its affiliated companies.

The postpaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DEC
DEC/CMS
DEC/MMS
DECnet
DECsystem-10
DECSYSTEM-20
DECUS
DECwriter
DIBOL
EduSystem
IAS
MASSBUS
PDP
PDT
RSTS
RSX
UNIBUS
VAX
VAXcluster
VMS
VT
digital

ZK4413

HOW TO ORDER ADDITIONAL DOCUMENTATION
DIRECT MAIL ORDERS

USA & PUERTO RICO*

Digital Equipment Corporation
P.O. Box CS2008
Nashua, New Hampshire
03061

CANADA

Digital Equipment
of Canada Ltd.
100 Herzberg Road
Kanata, Ontario K2K 2A6
Attn: Direct Order Desk

INTERNATIONAL

Digital Equipment Corporation
PSG Business Manager
c/o Digital's local subsidiary
or approved distributor

In Continental USA and Puerto Rico call 800-258-1710.

In New Hampshire, Alaska, and Hawaii call 603-884-6660.

In Canada call 800-267-6215.

*Any prepaid order from Puerto Rico must be placed with the local Digital subsidiary (809-754-7575).

Internal orders should be placed through the Software Distribution Center (SDC), Digital Equipment Corporation, Westminster, Massachusetts 01473.

This document was prepared using an in-house documentation production system. All page composition and make-up was performed by TeX, the typesetting system developed by Donald E. Knuth at Stanford University. TeX is a trademark of the American Mathematical Society.

Contents

Chapter 3 Problems, Restrictions, and Notes

Part II LAT/VMS Management Guide

Chapter 4 Functions of LAT/VMS Software on a VMS System

Chapter 5 VAX/VMS Service Node Management

Chapter 6 Setting Up Remote Printers

Chapter 7 LAT Port Driver QIO INTERFACE

Chapter 8 LATCP Command Descriptions

Examples

Figures

Tables

Preface

MicroVMS Release Notes, Version 4.6 describes Version 4.6 of the MicroVMS operating system. It lists and discusses changes to the system, new features, corrected problems, and restrictions in its use. It also describes changes and corrections to the documentation set. Part II of the *MicroVMS Release Notes, Version 4.6* contains the *LAT/VMS Management Guide*. This guide provides complete documentation of the new LAT/VMS software.

This book contains information of interest to general users, system managers, application programmers, and system programmers.

This book has two parts. Part I contains the following chapters:

- Chapter 1: Upgrading to MicroVMS Version 4.6 (guidelines for performing the Version 4.6 upgrade)
- Chapter 2: New and Changed Features
- Chapter 3: Problems, Restrictions, and Notes

Part II contains the *LAT/VMS Management Guide*. This guide describes and explains the tasks involved in managing the Local Area Transport (LAT) software on Version 4.6 systems. It includes the following chapters and appendices:

- Chapter 4: Functions of LAT/VMS Software on a VMS System
- Chapter 5: VAX/VMS Service Node Management
- Chapter 6: Setting Up Remote Printers
- Chapter 7: LAT Port Driver QIO Interface
- Chapter 8: LATCP Command Descriptions
- Appendix A: ASCII Characters for Node and Service Names
- Appendix B: Qualifiers for DCL Printer Setup Commands
- Appendix C: LATCP Error Messages

The following documents provide information that is relevant to Version 4.6 and the Version 4.6 upgrade procedure:

- *MicroVMS Installation and Operations Guide: VAXstation I,II and MicroVAX I,II*

- *MicroVMS VAXstation 2000/MicroVAX 2000 Installation Guide*
- *MicroVMS Release Notes, Version 4.5*
- *MicroVMS User's Manual*
- *MicroVMS Operating System, Version 4.6 Software Product Description, SPD 28.05.09*
- *System Software Ordering Table* (SPD 28.98.xx)

MicroVMS users can purchase any or all manuals of the full VAX/VMS documentation set.

The following conventions are observed in this manual:

Convention	Meaning
RET	A symbol with a one- to six-character abbreviation indicates that you press a key on the terminal, for example, RET.
$ SHOW TIME 11-NOV-1987 11:55:22	Command examples show all output lines or prompting characters that the system prints or displays in black letters. All user-entered commands are shown in red letters.
$ TYPE MYFILE.DAT . . .	Vertical series of periods, or ellipsis, means either that not all the data that the system would display in response to the particular command is shown or that not all the data a user would enter is shown.
file-spec, . . .	Horizontal ellipsis indicates that additional parameters, values, or information can be entered.
[logical-name]	Square brackets indicate that the enclosed item is optional. (Square brackets are not, however, optional in the syntax of a directory name in a file specification or in the syntax of a substring specification in an assignment statement.)
quotation marks (") apostrophe (')	The term quotation marks is used to refer to double quotation marks ("). The term apostrophe (') is used to refer to a single quotation mark.

Part I MicroVMS Release Notes, Version 4.6

Part I contains release notes for MicroVMS Version 4.6. It includes instructions for upgrading to Version 4.6, descriptions of new and changed features, and a summary of problems, restrictions, and notes.

Chapter 1

Upgrading to MicroVMS Version 4.6

1.1 Performing the Upgrade

MicroVMS Version 4.6 is an upgrade. You install an upgrade by performing either the MicroVMS upgrade procedure or the installation procedure. The upgrade procedure does the following:

- Makes room for the upgrade by purging and deleting some system files, but leaves some system files, and all user files, intact
- Creates a parallel system directory structure
- Restores the MicroVMS base system files from the distribution media to the system disk
- Merges old and new files
- Cleans up files and structures used only during the upgrade

Since Version 4.6 is a functional release, you must reinstall MicroVMS options and layered products.

The installation procedure does the following:

- Initializes the target disk (the disk on which Version 4.6 will be installed), erasing its entire contents
- Creates a system directory structure
- Restores the MicroVMS Base system files from the distribution media to the system disk

If you are upgrading on a MicroVAX I, MicroVAX II, VAXstation I, or VAXstation II, refer to the following book for complete instructions on performing the upgrade procedure and installation procedure: *MicroVMS Installation and Operations Guide: VAXstation I,II and MicroVAX I,II.*

If you are upgrading on a VAXstation 2000 or MicroVAX 2000, refer to the following book for complete instructions on performing the upgrade procedure and installation procedure: *MicroVMS VAXstation 2000/MicroVAX 2000 Installation Guide.*

These books also contain complete instructions for installing DECnet.

NOTE: The distribution media are labeled MicroVMS Version 4.6. During the upgrade, system prompts specify Version 4.6. Your documentation might refer to previous versions of MicroVMS.

1.2 Post-Upgrade Procedures—Mandatory Update

After you have upgraded to MicroVMS Version 4.6, but before you have installed MicroVMS options, you must install the mandatory update.

The mandatory update has one of the following labels:

- MicroVMS V4.6 BINRX50 Mandatory Update
- MicroVMS V4.6 BINTK50 Mandatory Update

Use the following procedure to install the mandatory update:

1. Log in to the System Manager's account (SYSTEM).
2. Ready the mandatory update media on the device drive that you will be using.
3. Enter the following command:

   ```
   $ @SYS$UPDATE:VMSINSTAL VMSMUP046 device-name
   ```

   Enter the device name in the form *ddcu.*

The procedure prompts you for certain information (for example, whether you have inserted the mandatory update and are ready to proceed). Upon completion, the procedure shuts down the system, after which you must reboot.

Chapter 2

New and Changed Features

This chapter discusses new features of the operating system for Version 4.6. It also describes features that have changed since the release of Version 4.5.

The material in this chapter is organized as follows:

Section 2.1—General User Information
Section 2.2—System Manager Information
Section 2.3—Application Programmer Information
Section 2.4—System Programmer Information

2.1 General User Information

The following section describes the new features of MicroVMS Version 4.6 of interest to the general user. It also discusses changes to the operating system since Version 4.5.

2.1.1 SET HOST/DTE/DIAL Command—Modem Support

The SET HOST/DTE/DIAL DCL command now supports the following modems:

- DF03
- DF112
- DMCL (any modem that uses DIGITAL Modem Control Language)

2.1.2 Support for New VT300 Series Terminals

Version 4.6 provides support for the new VT300 series terminals. The terminal device type is VT300_SERIES, and the terminal characteristic is DEC_CRT3. The DEC_CRT3 characteristic indicates the following:

- ISO LATIN-1 character set resident
- The ability to display a twenty-fifth status line
- The ability to report explicit state information about itself

When entering the SET TERMINAL command, you can specify VT300_SERIES as a terminal type for the /DEVICE_TYPE qualifier. You can also supply a value of 3 to the /DEC_CRT qualifier, which sets the DEC_CRT3 terminal characteristic.

2.2 System Manager Information

The following section describes the new features of MicroVMS Version 4.6 of interest to the system manager. It also discusses changes to the operating system since Version 4.5.

2.2.1 AUTOGEN Enhancements

The following sections describe Version 4.6 AUTOGEN enhancements.

2.2.1.1 OLDSITE Parameter-Passing Mechanism Becoming Obsolete

The MODPARAMS parameter-passing mechanism supersedes the OLDSITE mechanism. The transition to the MODPARAMS mechanism involves two steps. First, Version 4.6 eliminates SYS$SYSTEM:OLDSITE1.DAT. Second, the next major release will eliminate SYS$SYSTEM:OLDSITE2.DAT, OLDSITE3.DAT, AND OLDSITE4.DAT.

The parameter values created by the files OLDSITEn.DAT can be found in the file SYS$SYSTEM:PARAMS.DAT. These parameter values are indicated by comment lines specifying which OLDFILEn.DAT the parameters come from. DIGITAL recommends that you review those parameters in the most recent version of SYS$SYSTEM:PARAMS.DAT.

If, when you review SYS$SYSTEM:PARAMS.DAT, you find parameters that were transferred from OLDSITEn.DAT files that you feel are necessary for your site, include the records for these parameters from PARAMS.DAT in SYS$SYSTEM:MODPARAMS.DAT. If MODPARAMS.DAT does not exist, it may be created with any text editor.

After updating MODPARAMS.DAT to reflect the parameters transferred from the OLDSITEn.DAT files, invoke AUTOGEN as follows:

```
@SYS$UPDATE:AUTOGEN GETDATA REBOOT INITIAL
```

For more information about the MODPARAMS parameter-passing mechanism, refer to Section 11.4 of the *VAX/VMS System Manager's Reference Manual.*

2.2.1.2 New WSMAX Check

Since a value for WSMAX that is too high can disable pool expansion, AUTOGEN now displays a warning if it discovers a user-supplied value for WSMAX that appears to be too high.

2.2.1.3 Current Parameter Values Saved In SYS$SYSTEM:VAXVMSSYS.OLD

AUTOGEN now saves current system parameters in SYS$SYSTEM:VAXVMSSYS.OLD before updating these parameters in SYS$SYSTEM:VAXVMSSYS.PAR.

2.2.1.4 Specifying an Alternate Startup Command Procedure in MODPARAMS.DAT

If you use a startup command procedure other than SYS$SYSTEM:STARTUP.COM, you can now assign the name of your procedure to the symbol STARTUP in MODPARAMS.DAT. After invoking AUTOGEN, your procedure becomes the default startup command procedure.

For example, to specify MY_STARTUP.COM as the new default startup command procedure, make the following entry in MODPARAMS.DAT:

```
STARTUP = "SYS$MANAGER:MY_STARTUP.COM"
```

2.2.1.5 QUORUM Value Now Calculated

For Version 4.6, AUTOGEN calculates a value for the QUORUM parameter by selecting the higher of the following two values: the initial quorum, or the current cluster quorum.

2.2.2 Page and Swap File Handling—Improvements

Version 4.6 improves the way AUTOGEN handles page and swap files. AUTOGEN now understands and manipulates secondary files. Generally, if secondary page or swap files exist, AUTOGEN's file manipulation involves secondary files but excludes primary files; AUTOGEN assumes that primary files are on a cluster common system disk. The following list describes how AUTOGEN handles different types of input:

1. If AUTOGEN does not receive user-supplied information from MODPARAMS.DAT, it performs default page and swap file size calculations. If no secondary files exist, AUTOGEN applies any changes to the primary files. If secondary files exist, AUTOGEN applies changes evenly across all secondary page or swap files, but does not modify primary files.

2. AUTOGEN can receive general user-supplied size information from MODPARAMS.DAT. This information consists of records of the form *PAGEFILE* = *n* or *SWAPFILE* = *n*. If *n* is zero, the corresponding section is skipped. If *n* is not zero, and no secondary files exist, AUTOGEN applies the value to primary files. If *n* is not zero, and secondary files exist, AUTOGEN applies any change evenly across all secondary files, but does not modify primary files.

3. For Version 4.6, you can specify the individual sizes of all existing page and swap files (including secondary files), as well as the location and size of new files that you want AUTOGEN to create. To do this, define symbols in MODPARAMS.DAT using the following format:

   ```
   {PAGE/SWAP}FILEn_{NAME/SIZE}
   ```

   In this format, *n* is an integer that specifies the page or swap file. Refer to the primary page and swap files by specifying a value of 1 for *n*; refer to subsequent files by specifying increasingly higher integer values for *n*. For example, to refer to a secondary page or swap file, you could specify a value of 2 for *n*. Braces ({}) indicate that you must choose between the options delimited by a backslash (/). For example, specify PAGE or SWAP, NAME or SIZE.

   For existing files, you typically define _SIZE symbols only; AUTOGEN already has the name and location. For example, to direct AUTOGEN to set the primary page file size to 10000 blocks, you would use the following symbol definition:

   ```
   PAGEFILE1_SIZE = 10000
   ```

   To direct AUTOGEN to create a new secondary swap file named PAGED$:[PAGESWAP]SWAPFILE.SYS that holds 30000 blocks, you would use the following symbol definitions:

   ```
   SWAPFILE2_NAME = "PAGED$:[PAGESWAP]SWAPFILE.SYS"
   SWAPFILE2_SIZE = 30000
   ```

   Note that you must manually edit SYS$MANAGER:SYSTARTUP.COM to include a SYSGEN command that installs the newly created secondary file.

You cannot specify both general and explicit information as described in numbers 2 and 3 above. AUTOGEN issues a warning if conflicting symbol definitions exist in MODPARAMS.DAT.

If the creation or extension of a file would cause the target disk to become more than 95% full, AUTOGEN issues a warning, and does not perform the operation.

For more information about AUTOGEN, refer to Section 11.1 of the *VAX/VMS System Manager's Reference Manual.*

2.2.3 MTHRTL—System-Wide Logical Name

SYS$SYSTEM:STARTUP.COM now defines the logical name MTHRTL as a system-wide logical name. The change was necessary for systems that needed UVMTHRTL; for consistency, the change was applied to all systems.

2.2.4 LAT/VMS—New Features

Version 4.6 includes new Local Area Transport (LAT) software.

The new LAT/VMS software supports asynchronous printers connected to LAT terminal servers. The software consists of new LTDRIVER and LATCP software, as well as a new software component, LATSYM—the LAT print symbiont.

Prior to Version 4.6, printer support was provided through the LATplus software; this software was included as part of the terminal server distribution kit. Version 4.6 includes an enhanced version of LAT as part of the operating system. In addition to having the LATplus features, LAT/VMS has a QIO interface that allows application programs to make host-initiated requests for connections to remote devices. If you originally installed LATplus as part of the terminal server kit, do not reinstall after performing the Version 4.6 upgrade.

Part II of this book contains complete documentation of the new LAT software, including instructions for modifying SYS$MANAGER:SYSTARTUP.COM and SYS$MANAGER:LTLOAD.COM.

The following new features are provided:

- A QIO Interface for LTDRIVER functions.
- LATCP user interface is new.
- Terminal characteristics can be passed to the host.
- The break character can be passed to the host.

2.2.4.1 QIO Interface

LAT/VMS 4.6 supports two QIO function code modifiers that allow application programs to request and terminate connections to remote devices on servers. These remote devices are mapped to an application device on the host. Refer to Chapter 4 of the *LAT/VMS Management Guide* for a complete description of these function code modifiers.

A print queue can be set up to automatically access the remote device located on the terminal server. Similar print queues can be set up on other service nodes, allowing the printer to be shared by users on many service nodes. Requests for connections to the port are queued on the server and are serviced on a first-in/first-out basis when the port becomes available.

2.2.4.2 New LATCP User Interface

The LAT/VMS software implements a new LATCP interface. With the new features, LATCP does the following:

- Adheres to DCL syntax standards
- Allows up to eight services per node
- Supports a SHOW PORTS command
- Allows you to change a single characteristic for a node without affecting the other node characteristics
- Allows a fixed service rating value to be specified

Refer to the *LAT/VMS Management Guide* for a complete description of the LATCP commands. The Software Notes section discusses the LATCP SHOW PORTS display.

2.2.4.3 Passing Terminal Characteristics

Terminal servers can now pass information about terminals to the host. Speed, parity, and character size are passed to the service node on session start-up. The server also passes information on parity, framing, and overrun errors to the service node if an error occurs at the terminal.

Terminal server users can pass the break character to a service node without the server interpreting BREAK as a request to go into local mode.

2.2.4.4 Software Notes

This section discusses some of the software features and restrictions, and supplements information in the *LAT/VMS Management Guide*.

Rating Algorithm

The service rating algorithm has changed to incorporate both the idle cpu time and the percentage of active users as follows:

```
                                       ijoblim - ijobcnt
(100 * % cpu_idle_time  +  155 * ------------------- ) * cpu_type_factor
                                            ijoblim
```

A penalty factor for small memory configurations is subtracted from the rating value as calculated in the algorithm shown.

Therefore, it is very important that the interactive job limit be set to a realistic value for your system. This can be done in the system's SYSTARTUP.COM file with the following command.

```
$ STARTUP$INTERACTIVE_LOGINS :== nn
```

In this example, *nn* is a number that takes into account your processor type and user load. This command is typically the last command in the SYSTARTUP.COM file.

SHOW PORTS Display

The LATCP program now has a display that shows information about interactive and applications ports. The display shows the server name and port name for active connections.

2.2.4.5 Problems with LAT-11 Software Layered Product

This section discusses problems and limitations when using the LAT-11 software that runs on a PDP-11 and that has been configured to serve terminals.

Bad Message Received Error

This LAT-11 software registers a Bad Message Received error for a service node that sends out a configuration message without any services. This can happen if an incorrect sequence is used when starting up the service node. Use the following sequence in the LTLOAD.COM file and when starting up the service node manually:

1. LCP SET NODE command to set the node name and characteristics
2. LCP CREATE SERVICE and LCP SET SERVICE commands to set up the services for the node
3. LCP START NODE command to start LAT service on the node

This LAT-11 software also registers a Bad Message Received error if a service node offers more than two services. Do not enable more than two services on any node accessed by LAT-11.

Cluster Nodes Not Accessible by LAT-11

The LAT-11 software requires that the first service name offered by a VAX/VMS node be the node name. For a VAXcluster, the cluster service name must be the second service offered by the VAX/VMS node. All other DIGITAL terminal server products require only a cluster service name. Enabling the node name as a service is only necessary when using LAT-11. Add the following command line to your LTLOAD.COM file before the command line that creates the cluster wide service name:

```
$ LCP CREATE SERVICE /IDENT /NOLOG
```

The sample LTLOAD.COM file that follows shows you where to enter this command line.

Example 2-1 Sample LTLOAD.COM File for Use with LAT-11

```
$ ! This command procedure starts up the LAT protocol and is compatible
$ ! with LAT-11.
$
$ RUN SYS$SYSTEM:SYSGEN
CONNECT LTAO/NOADAPTER
$
$! Invoke LATCP
$
$LCP := $LATCP
!
! The following commands set up LAT service with the default name
! SYS$NODE and default ident SYS$ANNOUNCE.  The first LAT service name
! defaults to the node name SYS$NODE.  YOU MUST SPECIFY a cluster wide
! service name as the first parameter in the command line.  Use the
! remaining parameters to specify values for other node characteristics,
! such as group codes.
!
$LCP SET NODE /IDENT  'P2' 'P3' 'P4' /NOLOG
$LCP CREATE SERVICE /ID                ! Fix for lat-11 bug
$LCP CREATE SERVICE 'P1' /IDENT /NOLOG ! Provide cluster service name as P1
$LCP START NODE
$!
```

New Features Not Supported

The LAT-11 software does not support the new functions provided by the VAX/VMS Version 4.6 LAT software. You cannot connect a remote printer to the LAT-11 software. However, all LAT functions previously supported by LAT-11 still work with VMS Version 4.6.

2.2.4.6 Miscellaneous LAT/VMS Problems

This section discusses problems affecting the LAT/VMS software and suggested solutions for those problems.

Delay in Process Disconnect

If virtual terminals are enabled on your system and you enter a terminal server DISCONNECT command, the process does not immediately go away; it goes away when the timeout period expires. This is normal and should be expected.

LATCP STOP NODE Command

The STOP NODE command deletes the current node characteristics. Avoid using the following sequence of commands:

```
LCP> STOP NODE
LCP> START NODE
```

Instead of using these commands, invoke the LTLOAD.COM file as previously shown, or set up the characteristics for your node manually with the SET NODE and SET SERVICE commands.

Note that if you issue a LATCP STOP NODE command, all LAT terminal users are disconnected from the node, and a process rundown is initiated.

LATCP /QUEUE Qualifier

When using the /QUEUE and /NOQUEUE qualifiers in the SET PORT command, exit LATCP and rerun LATCP before entering the command (for example, using the "LCP" foreign command will accomplish this).

LTLOAD.COM and VMS Version 4.6 Installation

The installation of VMS Version 4.6 overwrites any existing SYS$MANAGER:LTLOAD.COM file. This occurs because the LATCP command syntax has been completely redone since prior versions of VMS. Note that this situation also applies to a LTLOAD.COM file based on LATplus/VMS, in spite of its compatible syntax. If you want to continue using your existing LTLOAD.COM file, save it and restore it later.

LATCP and the DELETE PORT command

The DELETE PORT command does not correctly shut down a session when this is used on a port with an active session. Use the DELETE PORT command only for inactive application ports.

Solicit Connection QIO

Do not enter the solicit connection QIO if LATCP has not yet started the LAT protocol. The QIO request may not complete and will not return an error.

LAT PASSALL Session

When using a host-initiated connection with the UCB set to the PASSALL characteristic, the terminal server's input flow control for the port is disabled. This is normal behavior.

2.2.5 Network Control Program—SHOW CIRCUIT Command Changes

Prior to Version 4.6, the Network Control Program's SHOW CIRCUIT ... SUMMARY command displayed circuit information about all adjacent nodes (routing and non-routing). For Version 4.6, the SUMMARY parameter displays circuit information about adjacent routing nodes only. However, the STATUS parameter continues to display circuit information about all adjacent nodes. For more information about the SHOW CIRCUIT command, refer to the *VAX/VMS Network Control Program Reference Manual.*

2.3 Application Programmer Information

The following section describes the new features of MicroVMS Version 4.6 of interest to the application programmer. It also discusses changes to the operating system since Version 4.5.

2.3.1 Debugger—New Features

2.3.1.1 Predefined Breakpoints

If any portion of your program is written in VAX™ Ada® , then the following two breakpoints are automatically established when you invoke the debugger (the output of a SHOW BREAK command is shown):

```
Breakpoint on ADA event "DEPENDENTS_EXCEPTION" for any value
Breakpoint on ADA event "EXCEPTION_TERMINATED" for any value
```

These breakpoints are equivalent to issuing the following commands:

```
DBG> SET BREAK/EVENT=DEPENDENT_EXCEPTION
DBG> SET BREAK/EVENT=EXCEPTION_TERMINATED
```

Ada programmers find these breakpoints convenient for debugging tasking programs.

2.3.1.2 CALL from an Exception Breakpoint

Prior to Version 4.6, you could not issue the CALL command directly after an exception breakpoint was triggered. This restriction has been removed.

Some related restrictions still apply. If a routine is called with the CALL command just after an exception breakpoint was triggered, no breakpoints, tracepoints, or watchpoints set within that routine are triggered. However, they are triggered if the CALL command is given at another time.

™ VAX is a trademark of Digital Equipment Corporation.

® Ada is a registered trademark of the U.S. Government (Ada Joint Program Office).

2.3.1.3 STEP from an Exception Breakpoint

Prior to Version 4.6 you could not issue the STEP command directly after an exception breakpoint was triggered. This restriction has been removed.

Issuing a STEP command at an exception breakpoint causes you to step to the start of whatever exception handler gets control. If you have not declared any exception handlers, the exception is resignalled and the debugger prompt is displayed—that is, the STEP command has no effect.

2.3.1.4 Non-static Watchpoints

You can now set watchpoints on variables that are dynamically allocated, such as those on the stack or in registers. These are called nonstatic watchpoints.

You can set a watchpoint on a nonstatic variable only when its defining routine is active. If you try to set a watchpoint on a nonstatic variable when its defining routine is not active, the debugger issues a warning, as in the following example:

```
DBG> SET WATCH Y
%DEBUG-W-SYMNOTACT, nonstatic variable 'Y' is not active
```

To implement nonstatic watchpoints the debugger must trace every instruction, slowing down the execution of the program being debugged. When you set a nonstatic watchpoint, the debugger determines whether the watched location is statically or nonstatically allocated. If the location is nonstatically allocated, the debugger issues an informational message that you are setting a nonstatic watchpoint, so that you will be aware of the slower performance.

A nonstatic watchpoint is automatically canceled when execution returns from its defining routine, and an informational message is issued to that effect.

2.3.1.5 Debugger—Changes for VAXstations

Prior to Version 4.6, if the debugger was invoked on a VAXstation, it created a separate emulated terminal window for debugger input and output. Thus, terminal I/O performed by the program was logically and physically separated from debugger I/O. This behavior is especially useful for debugging screen mode applications.

To eliminate a potential problem, the method the Version 4.6 debugger uses to control the separate window has been changed. Previously, the debugger controlled the separate window with UIS$xxx calls. The debugger now uses the new OSC sequences to communicate control functions to the terminal emulator. As a result, the debugger's behavior is slightly different for Version 4.6. The debugger still creates a separate window, but only if both of the following conditions are met:

- You must be running VWS V3.0 (or higher).
- You must have the following system logical name defined:

  ```
  $ DEFINE/SYSTEM/EXEC UIS$VT_ENABLE_OSC_STRINGS TRUE
  ```

2.3.2 VAX PASCAL Run-Time Library—Changes

The following subsections describe Version 4.6 changes to the PASCAL Run-Time Library.

2.3.2.1 Changes to DEC and UDEC

- The default number of significant digits for the DEC and UDEC built-in routines has been changed from eight to ten.

  For example, consider the following code segment:

  ```
  WRITELN('<',DEC(12345),'>');
  ```

  Prior to Version 4.6, this code generated the following:

  ```
  <     00012345>
  ```

  For Version 4.6, this code generates:

  ```
  <  0000012345>
  ```

  To duplicate previous behavior, specify the number of significant digits in calls to the DEC or UDEC built-in routines.

- The default length parameter has been changed from 12 to 11 characters.

For more information about changes to DEC and UDEC, see the release notes for VAX PASCAL Version 3.5.

2.3.2.2 Enhanced KEY Attribute

For Version 4.6, the VAX PASCAL Run-Time Library includes support for the enhanced VAX PASCAL KEY attribute. The KEY attribute now accepts the following additional keywords:

[NO]CHANGES
[NO]DUPLICATES
ASCENDING
DESCENDING

2.3.2.3 New Default RECORD_LENGTH for TEXT files

For Version 4.6, the default record length for TEXT files increases from 133 to 255 characters. To duplicate previous behavior, you must specify the following in your OPEN statement:

```
RECORD_LENGTH := 133
```

2.3.2.4 Use of EXTEND, REWRITE, and TRUNCATE on Shared Sequential Files

Prior to Version 4.6, the VAX PASCAL Run-Time Library enforced solitary access to sequential files for the EXTEND, REWRITE, or TRUNCATE built-in routines. Version 4.6 supports shared access to sequential files for these built-ins.

2.3.3 PL/I Run-Time Library Now Supports VAX PL/I Version 3.0

The Version 4.6 PL/I Run-Time Library supports VAX PL/I Version 3.0. The Run-Time Library contains several minor changes that support the PL/I SUBSCRIPTRANGE, STRINGRANGE, and STORAGE conditions; these changes are incompatible with previous versions of VAX PL/I. Specifically, the PL/I Run-Time Library has the following new primary condition values:

- PLI$_SUBRG (replaces PLI$_SUBRANGEn)
- PLI$_STRRANGE (replaces PLI$_SUBSTRn)
- PLI$_STORAGE (replaces LIB$GET_VM)

For each new primary condition value, the built-in ONCODE function returns the old primary status value.

DIGITAL recommends that Version 4.6 customers upgrade to VAX PL/I Version 3.0. If you choose to run a previous version VAX PL/I, you should recode where appropriate. For example, you might make the following code change:

```
ON VAXCONDITION(PLI$_SUBSTR2) BEGIN;
     .
     .
     .
    END;
```

Change to:

```
DCL PLI$_STRRANGE GLOBALREF FIXED BIN(31) VALUE;
ON VAXCONDITION(PLI$_STRRANGE) BEGIN;
    IF ONCODE() ^= PLI$_SUBSTR2
    THEN
        CALL RESIGNAL();
    ELSE
     .
     .
     .
    END;
```

2.3.4 VAX Ada Run-Time Library—Enhancement for Unhandled Exceptions

VMS Version 4.6 improves the way the Ada Run-Time Library deals with unhandled Ada exceptions and VAX conditions. Exceptions and conditions are considered to be unhandled if they propagate as far as they can go—to the level of a task or a main program—and a VMS or VAX Ada Run-Time Library catch-all handler gains control. Catch-all handlers are located in frames enclosing the main program and library packages, each task body, and each accept body.

Beginning with Version 4.6, new catch-all handler messages are produced, and changes to program execution behavior have been made, as follows:

- If an unhandled VAX condition with a severity of success, information, warning, or error (any severity except severe) reaches an Ada Run-Time Library catch-all handler, the handler displays the condition message and continues program execution. This behavior is consistent with the behavior of VMS catch-all handlers.
- If an Ada exception or a VAX condition with a severity of severe reaches an Ada Run-Time Library catch-all handler, the handler displays the exception or condition message, and then the task, main program, or rendezvous becomes completed. (Note, however, that when an exception or severe condition leaves an accept body, the message is not displayed because the exception or condition will propagate to both of the tasks involved in the rendezvous.)
- The Ada Run-Time Library catch-all handlers display a warning when an unhandled exception may have to wait for dependent tasks to terminate.

The new catch-all handler messages are directed to both SYS$OUTPUT and SYS$ERROR.

Also beginning with Version 4.6, the point in the VAX Ada exception-handling sequence at which waiting for dependent tasks takes place has changed. Prior to Version 4.6, waiting for dependent tasks took place during the search for an applicable exception handler; with Version 4.6, waiting has been deferred until an applicable handler has been found. A more detailed explanation of this change follows.

In VAX Ada, a general condition handler is automatically established for all stack frames that have exception handlers, and a run-time table of active exception parts is maintained for each frame. The general condition handler determines which Ada exception handler in the frame eventually gains control (if any). Any subsequent Ada exception propagation takes place in two phases. During the first phase, the general condition handler determines which Ada exception handler should gain control; each frame on the stack is searched for this handler. When the applicable handler is found, the general condition handler requests a stack unwind, and the second phase begins. During the second phase, each frame is removed from the stack. Prior to

VMS Version 4.6, waiting for the termination of tasks dependent on some Ada frame took place during the first phase (search for a handler). Now, waiting for dependent tasks takes place during the second phase (unwind). After the unwind, the handler for the exception executes.

The Version 4.6 exception-handling improvements will have the following effects:

- Programs written entirely in Ada will not be visibly affected by the change in the point at which waiting for dependent tasks takes place. Such programs will be affected only by the new catch-all handler error messages and by continuation of the main program in cases of nonsevere unhandled exceptions.
- Software (such as the CLI) that signals a condition in order to print a message, expecting continuation at the point of the signal, is now supported—provided that the program does not handle the condition (exception) before the condition gets to the Ada Run-Time Library catch-all handlers.
- Software that signals a nonsevere condition value with a call to the VMS Run-Time Library routine LIB$SIGNAL, but that does not want the continuation that LIB$SIGNAL usually leads to, must call the Run-Time Library routine LIB$STOP instead, or use an Ada raise statement (if the signaling software is written in Ada).
- A task no longer terminates silently because of an unhandled exception—the exception message is now displayed. In addition, the exception message will appear before waiting begins for dependent tasks (because such waiting may cause a deadlock). This will make Ada programs more robust because an unexpected exception in a production program will now generate a message.

  If you do not want your software to produce task termination messages, you may want to have exception handlers in those task bodies to which you expect unhandled exceptions to propagate. For example, if you expect that the predefined exception END_ERROR will cause task termination messages in one of your tasks, you could have the following code, or its equivalent (the action need not be a null statement), in the exception part of the affected task body:

  ```
  when END_ERROR =>  null;
  ```

  The handler absorbs the unhandled exception and prevents it from propagating further. The use of a handler in this situation also allows you to see that the termination resulting from this exception is to be expected.
- The change in the point at which waiting for dependent tasks takes place may affect mixed-language programs. Prior to Version 4.6, an Ada exception that propagated to non-Ada code would cause execution to wait until all dependent Ada tasks terminated; a handler in the non-Ada code could not execute until the tasks terminated. With Version 4.6, the exception will be propagated, and

dependent tasks will continue to execute; a handler in the non-Ada code may execute concurrently with the dependent tasks.

If some software beyond your control is adversely affected by the messages resulting from unhandled exceptions, you can hide the messages by defining the logical names SYS$OUTPUT, SYS$ERROR, and ADA$OUTPUT. Define SYS$OUTPUT and SYS$ERROR to be where you want the messages to go, and define ADA$OUTPUT to be where you want Ada output (from package TEXT_IO) to go.

NOTE: You should redirect error-message output only as a temporary measure until you have modified your program as previously described. If you redirect SYS$OUTPUT, be careful to ensure that you do not miss other error messages that might occur; DIGITAL advises that you capture the output directed to SYS$OUTPUT and compare it with output containing the messages you would otherwise expect.

For information about new features of the debugger that affect VAX Ada programs, see Section 2.3.1.

2.3.5 VAX C Run-Time Library—Changes

2.3.5.1 Function Restrictions Removed

Prior to Version 4.6, **printf** functions could not format more than 512 characters in a single call. For Version 4.6, **printf** functions accept formatted output of unlimited length. However, an individual field in the resulting string cannot be longer than 512 characters.

2.3.5.2 File Sharing Now Supported

Prior to Version 4.6, the VAX C Run-Time Library did not support file sharing. The Version 4.6 VAX C Run-Time Library supports file sharing when you use record mode to access files; you must use the ctx=rec file attribute with all file open functions. Specify the shr=xxx file attributes as appropriate.

2.3.5.3 Stream I/O Facilities

Version 4.6 improves stream I/O facilities in the VAX C Run-Time Library. You can now specify the mbc=nnn file attribute when opening stream files. The value for this attribute specifies the number of blocks to allocate for I/O buffer. Reads and writes are performed using this block size.

For more information about changes to the VAX C Run-Time Library, see current documentation for VAX C Version 2.3.

2.4 System Programmer Information

The following section describes the new features of MicroVMS Version 4.6 of interest to the system programmer. It also discusses changes to the operating system since Version 4.5.

2.4.1 System Services—New Item Codes

The following item codes have been added to $GETSYI.

C Arguments

SYI$_XCPU

When SYI$_XCPU is specified, $GETSYI returns the extended CPU processor type of the node. $GETSYI returns this information only for the local node.

The general processor type value should be obtained first by using the SYI$_CPU item code. For some of the general processor types, there is extended processor type information provided by the item code, SYI$_XCPU. For other general processor types, the value returned by the SYI$_XCPU item code is currently undefined.

Since the processor type is a longword decimal number, the **buffer length** field in the item descriptor should specify 4 (bytes).

The $PRDEF macro defines the symbols for the extended processor types. The current extended processor types available and their symbols are as follows:

VAX Processor Type Symbol	Extended Processor Type	Extended Processor Symbol
PR$_SID_TYPUV	MicroVAX II VAXstation II	PR$_XSID_UV_UV2
	MicroVAX 2000 VAXstation 2000	PR$_XSID_UV_410
PR$_SID_TYP8NN	VAX 8500	PRS$_XSID_N8500
	VAX 8550	PRS$_XSID_N8550
	VAX 8700	PRS$_XSID_N8700
	VAX 8800	PRS$_XSID_N8800

SYI$_XSID

When SYI$_XSID is specified, $GETSYI returns processor-specific information. For the MicroVAX II, this information is the contents of the system type register of the VAX node. The system type register contains the full extended information used in determining the extended system type codes. For other processors, the data returned by SYI$_XSID are currently undefined.

Since the value of this register is a longword hexadecimal number, the **buffer length** field in the item descriptor should specify 4 (bytes).

2.4.2 System Dump Analyzer—New Command

The Version 4.6 System Dump Analyzer has a new command, SHOW CALL_FRAME. This section describes the new command in detail. The information in this section updates the *VAX/VMS System Dump Analyzer Reference Manual.*

SHOW CALL_FRAME

Displays the locations and contents of the longwords representing a procedure call frame.

Format

SHOW CALL_FRAME *address*

Command Parameter

address

An expression representing the starting address of the procedure call frame you want to display.

Qualifiers

/NEXT_FP

Displays the procedure call frame starting at the address stored in the FP longword of the last call frame displayed by this command.

Description

Whenever a procedure is called using CALLG or CALLS instructions, information is stored on the stack of the calling routine in the form of a procedure call frame. This call frame contains the following longwords:

- A condition handler address
- A longword containing the stack pointer alignment bits, the register save mask for registers R0 - R11, and the saved PSW of the caller
- The saved AP value of the calling routine
- The saved FP value of the calling routine
- The saved PC value (return address) of the calling routine

- One longword for each saved register (R0 - R11) of the caller, specified by the register save mask

The SHOW CALL_FRAME command displays the call frame information by interpreting a specified address expression as the beginning address of the call frame. If no address expression or options are specified, the default address expression for SHOW CALL_FRAME is the longword contained in the current process FP register.

The following example shows the display produced by the SHOW CALL_FRAME command. The display consists of the following sections:

Instruction Type	The display indicates which type of instruction, either a CALLG or CALLS instruction, generated the procedure call frame.
Call Frame Address	SDA lists all the virtual addresses that are part of the call frame. The call frame addresses are listed in a column that increases in increments of 4 bytes (one longword).
Call Frame Contents	SDA lists the contents of the call frame longwords in a column next to the call frame addresses.
Symbols	SDA attempts to display the contents of the longwords in the call frame with the exception of the "Mask-PSW" longword, which is not symbolized.
Longword Description	SDA provides a meaningful description of the contents of each longword in the context of a procedure call frame.
Stack Alignment	SDA provides a message describing the number of bytes by which the stack pointer was adjusted prior to storing the call frame information.
Argument List	For CALLS cases, the argument list is displayed by virtual address and contents in two columns below the stack alignment field.

All valid procedure call frames have a 0 in bit 28 of the second longword of the call frame. If the call frame specified has a 1 in bit 28 of the second longword of the call frame, the call frame is invalid and the SDA display shows:

```
Invalid Call Frame:   Bit 28 is Set in "Mask-PSW" Longword
```

All valid procedure call frames begin on a longword boundary. If the specified address expression does not begin on a longword boundary, the call frame is invalid and the SDA display shows:

```
Invalid Call Frame:   Start Address Not On Longword Boundary
```

Example

```
SDA> SHOW CALL_FRAME 7FFE7D94


           Call Frame Information
           ----------------------
                Call Frame Generated by CALLS Instruction
           Condition Handler      7FFE7D94  00000000
           SP Align Bits = 00     7FFE7D98  20FC0000
              Saved  AP           7FFE7D9C  7FFED024
              Saved  FP           7FFE7DA0  7FFE7DE4    CLT$GL_KSTKBAS+005E4
              Return PC           7FFE7DA4  801A0CEE    SYSTEM_PRIMITIVES+005E4
                   R2             7FFE7DA8  7FFE7DD0    CTL$GL_KSTKBAS+005D0
                   R3             7FFE7DAC  7FFCCFF8
                   R4             7FFE7DB0  80443D90
                   R5             7FFE7DB4  7FFCD000
                   R6             7FFE7DB8  7FFE6400    MMG$IMGHDRBUF
                   R7             7FFE7DBC  00000003
           Align Stack by 0 Bytes =>
           Argument List          7FFE7DC0  00000003
                                  7FFE7DC4  7FFE7DD0    CTL$GL_KSTKBAS+005D0
                                  7FFE7DC8  00000000
                                  7FFE7DC8  00000000
```

2.4.3 Ethernet/802 Device Drivers

The following sections describe changes to the Ethernet/802 device drivers.

2.4.3.1 802 Response Packets

The Ethernet/802 device drivers now allow a response packet to be transmitted on channels that have the 802 packet format enabled. This is accomplished using the WRITE function code and the IO$M_RESPONSE modifier. Use of this modifier is validated for those 802 channels that have Class I service enabled; the control field value for channels with Class I service enabled must be either XID or TEST in order to send a response packet.

2.4.3.2 802 User Supplied Services

Prior to Version 4.6, the Ethernet/802 device drivers responded to XID and TEST command packets. For Version 4.6, all XID and TEST packets (command or response) for channels with User Supplied service are not responded to by the Ethernet/802 device drivers, but are instead passed to the application through READ requests.

Ethernet/802 device drivers still respond to XID and TEST command packets for channels with Class I service enabled.

2.4.3.3 Protocol Type Validation

The protocol type (NMA$C_PCLI_PTY) parameter is now validated on the SETMODE QIO. The validation happens as follows: the Ethernet/802 device driver takes the low order word of the longword parameter and swaps the two bytes. This new word value may not be less than 1501 (O5DD hexadecimal). If the value is less than 1501, SS$_BADPARAM status is returned in the IOSB.

Chapter 3

Problems, Restrictions, and Notes

This chapter discusses problems corrected in Version 4.6 of the MicroVMS operating system. It also describes any restrictions that apply to the use of the Version 4.6 operating system and contains other information concerning the release.

For ease of reference, the material in this chapter is organized as follows:

Section 3.1—General User Information
Section 3.2—System Manager Information
Section 3.3—Application Programmer Information
Section 3.4—System Programmer Information

3.1 General User Information

This section describes problems resolved in MicroVMS Version 4.6, lists known restrictions, and contains other information about the release of interest to the general user.

3.1.1 Command Procedures Restriction

For Version 4.6, all commands, full-line comments, and labels in command procedures must be preceded by a dollar sign ($). Although users have always been instructed to place a dollar sign before commands and labels, pre-Version 4.6 command procedures that omitted dollar signs in front of labels did not necessarily stop executing. Version 4.6, however, now treats missing dollar signs as illegal syntax; a command procedure that omits a dollar sign before a command or label will stop executing. (Continue to omit dollar signs from the beginning of data lines.)

3.1.2 SET HOST/DTE Problems Corrected

Version 4.6 makes the following corrections to the SET HOST/DTE command:

- SET HOST/DTE/LOG no longer inserts an extra line-feed character in log file records.
- You can now use SET HOST/DTE/LOG with systems that use null characters as filler after carriage returns or line feeds. Previous releases incorrectly interpreted the end-of-line for these systems.
- Prior to Version 4.6, the break signal time was variable. For Version 4.6, the break signal lasts one-half second. As a result, all systems now interpret the break signal correctly.
- When a hang-up terminates a SET HOST/DTE modem line connection, SET HOST/DTE terminates with the following message:

  ```
  %SYSTEM-F-HANGUP, data set hang-up.
  ```

For more information about the SET HOST/DTE, command see the *VAX/VMS DCL Dictionary.*

3.1.3 SET TERMINAL/PASTHRU/PERMANENT Now Works Correctly

Prior to Version 4.6, if you attempted to set a terminal to permanent PASTHRU mode using the command, SET TERMINAL/PASTHRU/PERMANENT, the terminal did not remain in PASTHRU mode.

Version 4.6 corrects this problem. If you set a terminal to permanent PASTHRU mode, PASTHRU mode continues until the system is rebooted, or until this characteristic is changed.

3.1.4 DECalc Version 2.2 Problem

Running DECalc V2.2 under Version 4.6 produces the following problem: accessing HELP aborts your image. This problem will be fixed in the next release of DECalc.

Until the next release of DECalc, you can rename the appropriate help files in SYS$HELP. Attempts to access HELP will then produce a "file not found" error message, but will not abort the image.

3.1.5 SET QUEUE, START/QUEUE, and INITIALIZE/QUEUE Problem Corrected

Prior to Version 4.6, DCL commands SET QUEUE, START/QUEUE, AND INITIALIZE/QUEUE incorrectly erased queue and job attributes set previously with the /BLOCK_LIMIT and /PAGES qualifiers. Version 4.6 corrects this problem.

3.2 System Manager Information

This section describes problems resolved in MicroVMS Version 4.6, lists known restrictions, and contains other information about the release of interest to the system manager.

3.2.1 ASYNCH DECNET Lines Problem—Corrected

Prior to Version 4.6, MicroVMS attempted to switch terminal lines set for ASYNCH DECNET back to terminal mode after power failures. Power failures resulted in a useless terminal line or a system crash.

The power fail logic in the terminal driver now recognizes lines set for ASYNCH DECNET and leaves those lines intact.

3.2.2 MTHRTL Installation

Version 4.6 installs the MTHRTL library differently than previous releases. Version 4.6 installs MTHRTL using special code in SYS$SYSTEM:STARTUP.COM that determines the appropriate library for your system. However, if you did not run AUTOGEN at the end of your Version 4.6 installation, you might receive messages stating that MTHRTL is already installed. Run AUTOGEN to eliminate this problem.

3.2.3 Future Release Will Enforce Modem Signal Requirements

A future release of the operating system will enforce modem signal requirements, described in Section 8.2.3 of the *VAX/VMS I/O User's Reference Manual: Part I,* before allowing a login. System managers should ensure that their host system modems are properly wired and meet the requirements.

3.2.4 EDTSECINI Editor Support

The EDTSECINI editor provides a TPU-based EDT keypad emulation editor. This editor is supplied in both source and compiled format in Version 4.6.

However, for the next major release, the EVE editor will support the EDT keypad; DIGITAL will no longer supply either the EDTSECINI.TPU or EDTSECINI.TPU$SECTION files. If you plan to continue using the EDTSECINI editor, DIGITAL recommends you save a copy before performing the upgrade for the next major release.

3.3 Application Programmer Information

This section describes problems resolved in MicroVMS Version 4.6, lists known restrictions, and contains other information about the release of interest to the application programmer.

3.3.1 VAX Ada Run-Time Library—Corrections

Version 4.6 includes the following corrections to the VAX Ada Run-Time Library that affect VAX Ada programs:

- Programs can now open files on multiple terminals using package TEXT_IO. In previous versions, this often resulted in access violations in the Ada Run-Time Library.
- Programs that attempt to recover from an error opening a file, where the FORM parameter to the OPEN or CREATE procedures is specified, now perform correctly. Previous versions produced unexpected errors or caused the program to hang.
- Prior to Version 4.6, tasking programs that opened files on other DECnet network nodes, or that opened indexed organization files, sometimes received unexpected error messages when the file was used soon after being opened; the Ada Run-Time Library did not wait correctly for the record stream connect to complete. Version 4.6 corrects this problem.

  Note that in multitasking applications, the call to the VAX Record Management Services (RMS) to open or create a file is always a synchronous operation, and all tasking suspends until the RMS operation completes.

You do not have to relink VAX Ada programs to take advantage of these corrections.

Section 3.5 includes a list of fixes for the VAX Ada Run-Time Library.

3.3.2 VAX PASCAL Run-Time Library—Corrections

The following subsections describe problems with the VAX PASCAL Run-Time Library corrected for Version 4.6.

3.3.2.1 Invalid String Truncation by DEC and UDEC

Current VAX PASCAL documentation states that when the length specified for the DEC and UDEC built-in routines is too short to hold the converted value, the Run-Time Library truncates the resulting string on the left. Prior to Version 4.6, however, the Run-Time Library filled the string with asterisks, instead of truncating the string. Version 4.6 corrects this problem. The Run-Time Library now truncates the string as described in VAX PASCAL documentation.

3.3.2.2 KEY Checks Problem Corrected

When opening an existing file, the VAX PASCAL OPEN procedure verifies that all KEY attributes specified by the user program are consistent in data type and size with the actual ISAM keys found in the file. Prior to Version 4.6, if you skipped a key position with the KEY attribute, the Run-Time Library did not verify subsequent keys. For example, if a record contained KEY attributes for keys 0, 1, 3, 4, and 5, the Run-Time Library verified keys 0 and 1, but did not verify keys 3, 4, and 5. Version 4.6 corrects this problem; the Run-Time Library checks all keys for consistency.

3.3.3 VAX BASIC Run-Time Library—Corrections

The following subsections describe problems with the VAX BASIC Run-Time Library corrected for Version 4.6.

3.3.3.1 Problem With RMS Bits RAB$V_WAT and RAB$V_TMO Corrected

Prior to Version 4.6, RMS bits RAB$V_WAT and RAB$V_TMO were cleared in the RAB$L_ROP after each FIND or GET was executed. This problem has been corrected for Version 4.6. These bits now stay set when applications use a USEROPEN to set them.

3.3.3.2 Run-Time Dimensioned Arrays Problem Corrected

Releases prior to Version 4.6 had a problem with the deallocation of run-time dimensioned arrays. The problem caused a loss of virtual memory, which could produce quota exceeded error messages. Version 4.6 corrects this problem.

3.3.4 VAX SCAN Run-Time Library—Correction to ENDFILE Built-In Function

Previous versions of the VAX SCAN Run-Time Library had the following problem: after you opened a file for input, read to the end of the file, and closed the file, the ENDFILE built-in function incorrectly returned the value TRUE.

Version 4.6 corrects this problem. The ENDFILE built-in function always returns FALSE if the file is not open.

3.3.5 DECtalk DTK$ Facility Corrections

The DTK$ facility consists of routines that control the functions of the DECtalk device. Version 4.6 includes the following corrections to the RTL DTK$ facility:

- Prior to Version 4.6, the phone status processing algorithm incorrectly processed status messages by interpreting them as keypad input. Version 4.6 corrects this problem. The Version 4.6 DTK$ facility includes a redesigned algorithm that uses two queues to process status messages and keypad input separately.
- Simulation of the AUTOSTOP mode for the DECtalk I device now works correctly.
- Prior to Version 4.6, if you specified the TIMEOUT argument when using a DECtalk III device, DTK$DIAL_PHONE ignored the argument. Version 4.6 corrects this problem.
- Prior to Version 4.6, key codes of the form DTK$K_TRM_xxxx were incorrect, causing unexpected results. These keypad constants have been corrected in Version 4.6 and now function as documented.

For more information on the RTL DTK$ facility, refer to the *VAX/VMS Run-Time Library Routines Reference Manual.*

3.3.6 Debugger—Restrictions

The following subsections describe restrictions that apply to the Version 4.6 debugger.

3.3.6.1 SET SCOPE Command

Before issuing a SET SCOPE command, be sure that the module that contains the elements named in the path name has already been set, either dynamically by the debugger or by means of a SET MODULE command. Use the SHOW MODULE command to determine whether a module is set (that is, whether its symbols have been loaded into the run-time symbol table).

3.3.6.2 SET IMAGE Command

When you issue a SET IMAGE command and specify a list of images, only the last image in the list is set. For example:

```
DBG> SET IMAGE A,B,C
```

In this example, only image C is set. To set images A, B, and C, issue separate SET IMAGE commands for each image.

3.3.7 $REWIND and $PUT Problems in Versions 4.4 Through 4.6

Version 4.4 introduced general support for write sharing of sequential files. One aspect of that support was the ability to have multiple writers appending to the file, known as shared append.

While this feature has a number of useful attributes, it has had several problems with $REWIND and $PUT operations when RAB$V_TPT has been set. These two problems have interacted to cause a set of undesirable and changing behaviors.

In Version 4.4 neither of these operations worked correctly. $REWIND would change the current record position to the start of the file, but would also reset the shared append state. $PUT with RAB$V_TPT set would reposition to end of file if the shared append state were set but correctly truncate otherwise. This meant that sequences like the following (in FORTRAN) appeared to work, but all WRITE statements after the REWIND would not be correctly interlocked against other accessors:

```
REWIND (UNIT=1)
WRITE (1,10) MY_DATA
```

Version 4.5 corrected the $REWIND problem. This had the following two effects on the previous code fragment:

- WRITE statements were correctly interlocked because the shared append state was still set.
- The REWIND appeared not to work because the WRITE statement was incorrectly repositioning to the current end of file.

Version 4.6 corrects the problem with $PUT when RAB$V_TPT is set. The REWIND now appears to work correctly because the WRITE statement is not repositioning to the end of file; $PUT operations against files accessed for shared-append only reposition to the actual end of file if the current record position is the end of the file. This repositioning is required in order to take into account any records that might have been written by other accessors between operations.

3.4 System Programmer Information

This section describes problems resolved in MicroVMS Version 4.6, lists known restrictions, and contains other information about the release of interest to the system programmer.

3.4.1 Terminal Port Drivers Must Be Recompiled

Because of a change in the terminal driver CLASS_UNIT_INIT macro, you should recompile all terminal port drivers that are not part of the VAX/VMS or MicroVMS product.

The CLASS_UNIT_INIT change prevents the accidental switching of class vector tables in the UCB after a power failure; this keeps intact those terminal lines that have been switched to an alternate terminal class driver.

3.4.2 DECnet-VAX Nontransparent Connections—$DASSGN System Service

Version 4.6 corrects a problem with the $DASSGN system service on DECnet-VAX nontransparent connections.

The *VAX/VMS Networking Manual* states that issuing the $DASSGN system service on a nontransparent DECnet connection deassigns the channel and terminates the logical link immediately. This operation is equivalent to using $CANCEL followed by $QIO IO$_DEACCESS!IO$M_ABORT.

Prior to Version 4.6, however, $DASSGN on DECnet-VAX nontransparent connections did not work this way. Instead, DECnet issued the $QIO IO$_DEACCESS system service. As a result, remote tasks could not tell whether the logical link was terminated because the local task ran down normally, or because the local task aborted.

For Version 4.6, when a local task terminates a logical link by issuing the $DASSGN system service, the remote task receives an ABORT status in the mailbox.

3.4.3 IO$M_RESET Modifier in User-Written Drivers

Prior to Version 4.2, the IO$M_RESET function modifier had the same value as the IO$M_INHERLOG modifier, which inhibits error logging. Because the IO$M_RESET bit was used only by the DR11-W driver, Version 4.2 changed the value of IO$M_RESET to decouple it from the IO$M_INHERLOG bit. Release notes for Version 4.2 documented this change. It is important to note that the change also affects user-written drivers that use the IO$M_RESET modifier.

To avoid possible problems, you should concurrently reassemble the following:

- All user-written drivers that use the IO$M_RESET modifier
- All programs that perform QIOs to user-written drivers that use the IO$M_RESET modifier

Part II LAT/VMS Management Guide

Part II contains the LAT/VMS Management Guide.

This guide makes reference to LAT software running on VMS systems. Note that new LAT software is also part of the MicroVMS Version 4.6 operating system.

Chapter 4

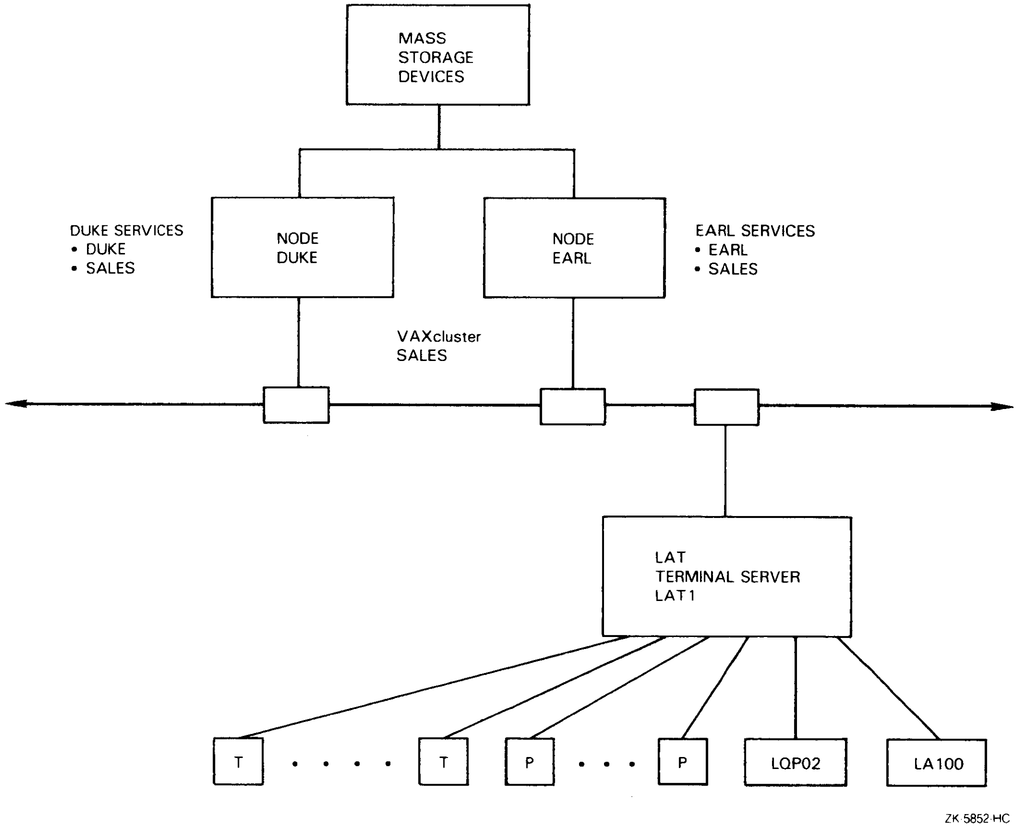
Functions of LAT/VMS Software on a VMS System

LAT/VMS functions in an Ethernet local area network (LAN) environment. The LAT/VMS software is part of your VMS operating system. LAT/VMS permits your VMS system and a server to exchange data. In this manual, remote printers are used as examples to describe how application programs access remote devices on a terminal server(s). LAT/VMS also has the features required for your VMS system to function as a node that accepts connections requested by server users.

4.1 Local Area Transport (LAT) Protocol

LAT is an Ethernet-based protocol which makes use of unique Ethernet features to provide an efficient means of logically connecting servers to one or more LAT nodes on the same Ethernet. LAT does not use DECnet to transport messages, but it can coexist on a network with DECnet. A typical LAT network is shown in Figure 4–1.

Figure 4–1 A Typical LAT Network



Servers and DIGITAL operating systems implement the LAT protocol for communications between servers and host systems. The LAT/VMS software implements LAT on your VMS system.

4.2 LAT Print Symbiont

The LAT/VMS software includes the LATSYM print symbiont. LATSYM differs from the VMS print symbiont, as it allows the LAT port driver to dynamically create and terminate connections to remote printers on servers. LATSYM uses the same I/O function code modifiers that are provided for application programs (see Chapter 7).

NOTE: You cannot use the standard VMS print symbiont for LAT connections. See Chapter 6 for the configurations required to set up your service node for remote printers.

4.3 LAT Definitions

A LAT node runs software that implements the LAT protocol. There are two types of LAT nodes: a VAX/VMS system that offers resources to terminal server users and servers that offer resources over the LAN.

4.3.1 Service Nodes

A service node is a LAT node that offers services to server users. In this manual, service node refers to a VAX/VMS computer system running the LAT software. Terminal servers that function as a service node also allow users to connect to services offered over the LAN (see Section 4.4.6).

4.3.2 Groups

Groups are used to partition a LAT network among LAT/VMS nodes and servers. A server can establish a logical connection with the LAT/VMS nodes that share a common group with the server. Nodes and servers that are partitioned in this fashion must be on the same Ethernet. See Chapter 5 for information about setting up groups for your service node.

4.3.3 Services

A service is a resource offered by service nodes on the LAN. You can have up to eight services defined for your node. Each service offers all of the resources of your node even though each service has a different service name. Services are announced by all service nodes on the LAN. Servers accept service announcements from service nodes that share a common group with the server.

4.3.4 Sessions

Sessions are logical connections from devices at terminal server ports to services. Each terminal on a server appears as if it were connected directly to a service it is using. During a session, the actions of the terminal server are transparent to users. LAT server software enables a user to maintain several sessions at once, allowing the user to move quickly between sessions. Refer to the server *Management Guide* or *Operations Guide.*

4.4 Server/Service Node Communications

Service node and server communications involve a number of concepts.

4.4.1 Virtual Circuits

The exchange of data on a LAT network relies on the use of a virtual circuit to create sessions between a server and a service node. A virtual circuit is an independent, logical path that is used for exchanging data between two nodes. LAT virtual circuits are multiplexed over a single physical circuit. The physical circuit hardware includes the server's network interface, the network hardware, and the service node's network interface.

To establish server/service node communications, the server determines whether a virtual circuit exists to the service node offering a service or requesting a remote printer connection. If no virtual circuit between the nodes currently exists, the server establishes one.

Once started, a virtual circuit between two nodes lasts as long as sessions are active. During this period, this single virtual circuit channels all the data exchanged by all sessions between the two nodes. Thus, all concurrent connections between a pair of LAT nodes—whether requested by a terminal user on a server or by an application program on a service node—share one virtual circuit. A LAT/VMS node can have concurrent virtual circuits to as many as 255 nodes.

Virtual circuit messages are sent automatically by the server at regular intervals. The interval between transmissions is determined by a circuit timer, which the server maintains. The server holds the requests-to-send data made by server ports by buffering input at a port until the circuit timer expires. The server then transmits all data that is pending from all of its ports in one virtual circuit message. If the server is maintaining multiple virtual circuits, the server builds one message per circuit and sends out multiple virtual circuit messages. Each message contains data for only those ports connected to the node at the other end of the circuit.

4.4.2 Multicast Service Announcements

A service node periodically sends a multicast message that contains information about the node and any services it offers to terminal servers. The multicast message contains a node name, group designations, service names, and service ratings. Using information received in these multicast messages, each server builds up a directory of service node names and service names. However, service nodes do not maintain an equivalent directory. For example, when an application program requires the use of a remote printer, the service node multicasts a request for information about the node offering that printer.

The service announcement includes identification strings for the service node and services offered by the node. An identification string is a short ASCII string that you specify when setting up your service node or creating a service.

4.4.3 Load Balancing and Service Rating

Servers provide a load-balancing feature that allows them to automatically establish connections to a service on the least busy service node that offers the service. Load balancing is especially useful for VAXcluster management, though it is not limited to clusters.

Load balancing depends upon service ratings provided by the service nodes. The service rating value for a service is calculated dynamically by each service node that offers it. The node announces the rating value over the LAN as part of the service announcement message. The service rating is based on the overall level of activity of the node and the processor type. A high level of system activity gives a low service rating and hence inhibits new connections. The use of service ratings allows load balancing among nodes offering the same services. Servers compare the service rating values provided by all service nodes offering a requested service, and connect to the node with the best (highest) rating.

Service names provide a mechanism for distributing user demand on VAXcluster nodes. By creating an identical service name on two or more service nodes, you enable servers to balance the load on the service nodes in the VAXcluster that offer the common service.

4.4.4 Automatic Failover

In addition to load balancing, servers provide automatic failover when multiple service nodes offer a common service. Automatic failover is a failure-recovery function that takes over if a session is disrupted because a service node becomes unavailable. After such a failure, the server automatically searches for other service nodes that offer the same service. The server attempts to connect to the service on an alternative node with the highest service rating. In the case of VAXclusters, automatic failover provides a reliable terminal connection to a reliable service. Automatic failover allows users to log in again and to continue working in the event of a node failure.

4.4.5 Remote Printer Device

A printer connected to a server is called a remote printer. A remote printer must be an asynchronous ASCII character device. A remote printer on a server can be shared by all LAT/VMS service nodes, which makes it possible to optimize the use of remote printers. If a server with a printer is located in an office, users of terminals on that server can conveniently obtain hard-copy listings for their printing tasks.

Application programs on a service node can request a connection to a remote device. A request by an application program for a connection to a server is called a host-initiated request. Host-initiated requests by an application program identify the targeted remote device. The identification is the server name and the port ID or a service offered at that port. Each remote device is mapped to a logical device (applications port) on the service node.

When an application program attempts to access the applications port, the LAT port driver (LTDRIVER) sends a host-initiated request over the LAN to the server. The LAT port driver is contained in the LAT/VMS software. The server then makes the connection. To guarantee that all the service nodes obtain reasonable access to each remote device, a server manager can enable a first-in-first-out (FIFO) queue on a server. A queued or nonqueued request is accepted by the server if the remote port is free. If the remote port is busy and queuing is enabled on the server, then a remote request is queued.

4.4.6 Servers as Service Nodes

In addition to offering printer support, some servers also operate as LAT service nodes. In functioning as a service node, a server advertises services, for example, a remote printer or dial-out modem, to its own port users or to other servers. When one of these services is requested, the server completes the logical connection to the service.

Note that not all servers offer printer support or are capable of functioning as LAT service nodes. Refer to the *Software Product Description* of each server to determine its capabilities.

NOTE: The LAT port driver only accepts connections through one Ethernet adapter on each service node.

4.5 LAT/VMS Files

The files that constitute the LAT/VMS software are installed in the following directories:

SYS$SYSTEM directory

LTDRIVER.EXE

LATCP.EXE

LATSYM.EXE

SYS$MANAGER directory

LTLOAD.COM

SYS$HELP directory

LATCP.HLB

Chapter 5

VAX/VMS Service Node Management

As a LAT service node manager, you are responsible for several broadly defined tasks, which include:

- Setting up the characteristics of your service node (5.4).
- Managing services offered by your service node (5.5).
- Setting up remote devices for your service node (Chapter 6).
- Editing your system startup files (5.7).
- Starting the LAT software (5.8.4).
- Stopping the LAT software (5.8.5).

This chapter discusses setting up characteristics of your service node and managing services offered by your service node.

5.1 Management Overview

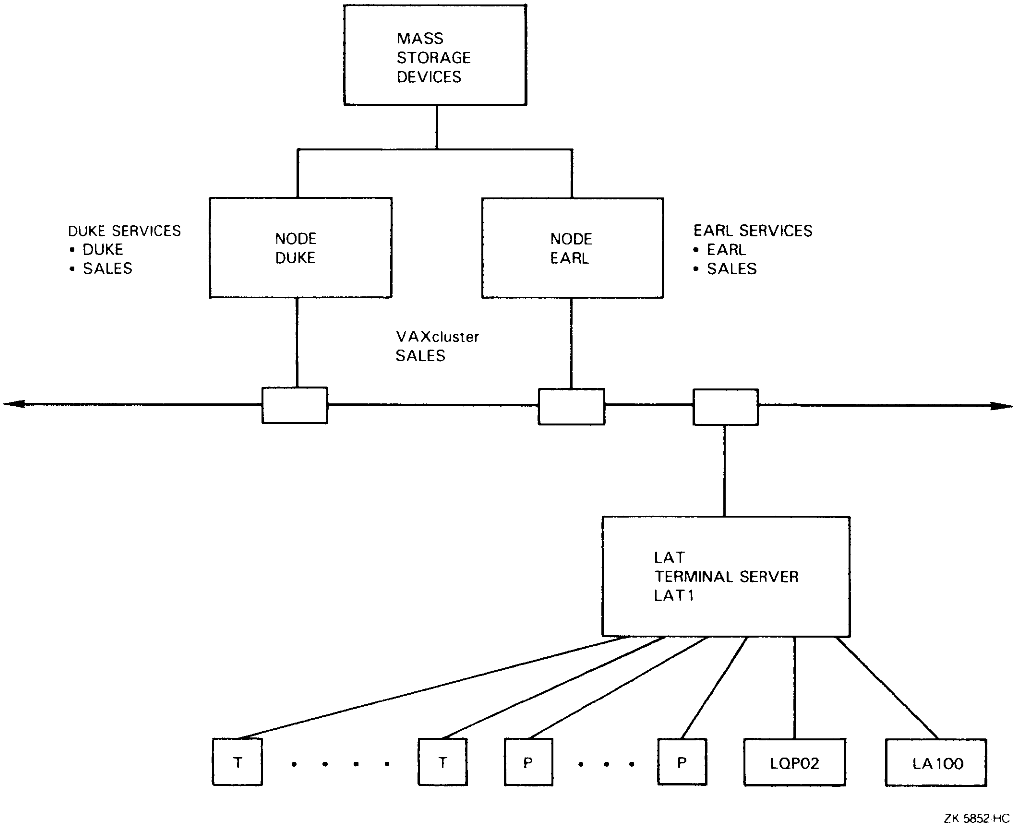
The LTLOAD command file for managing your LAT service node is part of the LAT/VMS software provided in the SYS$MANAGER account. LTLOAD.COM contains LATCP commands. The characteristics set up by the LTLOAD.COM file are your LAT default characteristics. To set up the characteristics to customize your service node and services, edit LTLOAD.COM.

5.2 Example of a LAT Network

A sample LAT network is shown in Figure 5–1. This figure is referred to throughout this chapter.

The nodes DUKE and EARL, which are in a VAXcluster, can make host-initiated requests for any of the remote printers connected to the LAT1 server. These nodes share the same mass storage devices and offer a common cluster LAT service called SALES. The LAT terminal server, LAT1, has interactive terminals represented by the letter T. The printers on this server are represented by the letter P or by a specific printer name. LQP02 and LA100 are printers.

Figure 5-1 Service Nodes, Remote Printers, and Services



5.3 The LTLOAD.COM File

The LTLOAD.COM file contains commands that perform many of the basic LAT management functions needed at system startup time. The LTLOAD.COM file:

- Loads the LAT port driver using a SYSGEN command
- Sets up your LAT service node characteristics
- Creates services for your service node

- Sets up some of the remote printer characteristics
- Starts the LAT port driver

To tailor the service node characteristics of your service node, edit the LTLOAD.COM file.

After the LTLOAD.COM file is altered for your service node, enter a command in your system startup file to automatically execute the LTLOAD.COM file upon system startup. Section 5.7 explains how to do this.

Note that if DECnet is going to be started on your node, you must start DECnet before starting the LAT port driver.

The LTLOAD.COM default file that you get with your LAT software is shown in Example 5–1. This file applies default node and service characteristics to your system. To override the defaults, you can pass the characteristics as parameters in the command line. This method is especially useful if you have multiple nodes in a VAXcluster with the same LAT characteristics.

The LTLOAD.COM file accepts the following command line parameters:

- P1 specifies the service name.
- P2—P4 specifies up to three valid qualifiers for the LATCP SET NODE command, such as /ENABLE=(25,26).

You can also override the default parameters by editing the LTLOAD.COM file and specifying the node and service characteristics that you desire.

If you execute LTLOAD.COM without making changes or passing parameters, the default characteristics are assigned to your node.

The CREATE PORT and SET PORT commands in LTLOAD.COM set up a remote printer for your service node. These commands are explained in Chapter 6.

The LTLOAD.COM file, shown in Example 5–2, refers to the service nodes, server, and printers in Figure 5–1. Notice that the P1 and P2—P4 parameters are not specified in this example. The commands in this file are explained in the following sections of this chapter.

Example 5-1 LTLOAD.COM Default File

```
$ ! This command procedure starts up the LAT protocol
$ ! and configures applications ports for remote printer use.
$
$ RUN SYS$SYSTEM:SYSGEN
CONNECT LTA0: /NOADAPTER
$
$ ! Invoke LATCP
$
$ LCP := $LATCP
!
! The following commands set up the LAT service with the default name
! SYS$NODE and default ident SYS$ANNOUNCE.  If you do not want the
! default SYS$NODE node name applied, specify a service name for the first
! parameter in the command line.  You can use up to three parameters
! (P2 -- P4) to specify additional characteristics, such as group
! codes, for your node.
!
$ LCP SET NODE /IDENT  'P2' 'P3' 'P4' /NOLOG
$ LCP CREATE SERVICE 'P1' /IDENT /NOLOG
$ !
$ RUN SYS$SYSTEM:LATCP
!
! Set up the applications ports that will support remote printer
! access.
!
! Create the applications ports.
!
! CREATE PORT LTA1: /NOLOG
! CREATE PORT LTA2: /NOLOG
!
! Maps applications port(s) to a specific port(s) on the server.
!
! SET PORT LTA1: /APPLICATION /NODE=SERVER_1 /PORT=PORT_5
! SET PORT LTA2: /APPLICATION /NODE=SERVER_1 /PORT=LQ_PRINTER
!
! Start the node.
!
START NODE
EXIT
```

Example 5-2 Example of LAT Service Node Startup Command File

```
$ ! This command procedure starts up the LAT protocol
$ ! and configures applications ports for remote printer use.
$
$ RUN SYS$SYSTEM:SYSGEN
CONNECT LTA0: /NOADAPTER
$
$ ! Invoke LATCP
$
$ RUN SYS$SYSTEM:LATCP
!
! Set up the node name DUKE with the announcement "A member of
! the SALES VAXcluster".  Enable groups 1 and 4, and set the
! multicast timer.
!
!
SET NODE DUKE /IDENT="A member of the SALES VAXcluster" /NOLOG
SET NODE DUKE /ENABLE=(1,4) /MULTICAST_TIMER=70 /NOLOG
!
! Create services DUKE and SALES.
!
CREATE SERVICE DUKE /IDENT="DUKE Interactive Service" /NOLOG
CREATE SERVICE SALES /IDENT="SALES Service" /NOLOG
!
! Set up the applications ports that will support remote printer
! access.
!
! Create the applications ports.
!
CREATE PORT LTA321: /NOLOG
CREATE PORT LTA322: /NOLOG
!
! Maps applications port(s) to a specific port(s) on the terminal server.
!
SET PORT LTA321: /APPLICATION /NODE=LAT1 /PORT=PORT_7
SET PORT LTA322: /APPLICATION /NODE=LAT1 /SERVICE=PRINTER
!
! Start the node.
!
START NODE
EXIT
```

5.4 Setting Service Node Characteristics

As the service node manager, you need to set up the following service node characteristics:

- Node name
- Node identification announcement
- LAT network groups
- Multicast timer

5.4.1 Node Name

All LAT service nodes must have a node name that is unique within the LAT network.

If the service node is part of a DECnet network, the LAT service node name should be the same as the DECnet node name. The DECnet node name has to be unique within the same logical Ethernet and must be unique within the entire DECnet network. On DECnet nodes, the LAT node name is given the DECnet node name, SYS$NODE, by default. If the service node is not running DECnet, but will be in the future, then it is recommended that you define SYS$NODE before using the LTLOAD.COM file and LATCP.

The LAT node name can be from 1 to 16 ASCII characters long. Legitimate characters are described in Appendix A.

The following LATCP command assigns the name DUKE to your service node:

```
SET NODE DUKE
```

The node name default is the translation of the SYS$NODE logical name.

5.4.2 Node Identification Announcement

The node identification announcement is a description for your node. The announcement is advertised to server users in multicast messages that your service node processes once the LAT port driver is started. The announcement can be a string of up to 64 ASCII characters in length, that cannot begin with an ampersand (&). Nonprintable characters are translated as spaces. Enclose the string in quotation marks ("").

The following command specifies an identification announcement for node DUKE:

```
SET NODE DUKE /IDENT="A member of the SALES VAXcluster"
```

If you do not specify a node identification announcement, the default is the string for SYS$ANNOUNCE. If a string was not specified for SYS$ANNOUNCE and you do not specify a node identification announcement string, then only your service node name is sent in the multicast messages.

5.4.3 LAT Network Groups

Groups are used to partition the LAT network into logical subdivisions. Groups are set up by the network manager, system managers, and server managers. Controlling groups allows you to restrict message traffic between servers and service nodes. To establish a connection, the server requesting a connection to your LAT service node must share at least one group with your node.

When messages are received by a server from service nodes that are not in any group enabled on the server, these messages are ignored. Groups help manage the size of the server's databases by limiting the number of service nodes for which the server keeps information. Groups are not intended as a security mechanism.

The following LATCP command enables groups 1 and 4 for service node DUKE:

```
SET NODE DUKE /ENABLE=(1,4)
```

Group 0 is enabled by default for all service nodes and servers. If you do not want group 0 enabled, you must specifically disable it using the /DISABLE qualifier.

5.4.4 Multicast Timer

The multicast timer determines the time between the multicast messages sent by your service node. The minimum value is 10 seconds; the maximum is 255 seconds.

The default value of 60 seconds is usually adequate. If you assign a large value to the multicast timer, network overhead is minimized because multicast messages are sent less frequently. However, terminal server users have to wait longer for services to become available after the server is rebooted, or after recovering from a network problem. In addition, the service rating used in load balancing by the server (see Chapter 4) is provided in these messages. Infrequent multicasts by the service node can affect load balancing by the server.

If you assign a small value to the multicast timer, more network resources are consumed since the server must adjust its LAT database and process multicast messages more frequently.

Note that whenever you change a characteristic for your node, the multicast messages are immediately sent out over the network to announce that change.

The following LATCP command sets the multicast timer for node DUKE to 70 seconds:

```
SET NODE DUKE /MULTICAST_TIMER=70
```

This command causes multicast messages for node DUKE to be sent out on the network every 70 seconds.

5.5 Managing Services

You can create up to eight services on your service node. All the functions and features offered by your VMS system are included for each service.

Use the CREATE SERVICE command to specify the following characteristics for your service node:

- Service names
- Service announcements
- Service ratings

5.5.1 Service Names

Service names can be up to 16 ASCII characters long. Legitimate characters are described in Appendix A. Users generally request access to the resources of a service node by using a service name rather than the LAT node name.

Several service nodes can share one service name. A shared service name is especially useful in VAXclusters. It allows the cluster to be known by a cluster name instead of individual node names.

The name of each service and the rating of the service node (see Chapter 4) are contained in the multicast messages sent by your node. The service name is displayed to server users when they enter a SHOW SERVICES command at the server's local prompt.

The following LATCP command assigns the service name SALES to node DUKE, assuming that you issue the command on node DUKE:

```
CREATE SERVICE SALES
```

If you do not specify a service name in the command line, the default service name is assigned. The default is the translation of the SYS$NODE logical name. You can create only one service with the default service name. You must specify a unique service name for each service that you create.

5.5.2 Service Announcements

The service announcement is a description for your service. The announcement is advertised to server users in the multicast messages. The announcement is a string of up to 64 ASCII characters long that cannot begin with an ampersand (&). Nonprintable characters are translated as spaces. Enclose the string in quotation marks ("").

The following LATCP command specifies an announcement string for the service SALES:

```
CREATE SERVICE SALES /IDENT="SALES Service"
```

The default for the service announcement is the value for SYS$ANNOUNCE.

5.5.3 Service Ratings

At every multicast message interval, the LAT port driver software generates a dynamic service rating for services that it offers on the LAT network. Servers use this service rating for load balancing. Dynamic service ratings vary from 255 (highly available to users) to 0 (not available to users).

Normally, these dynamically generated service ratings are adequate and allow efficient load balancing on the LAT network. However, you have the option of overriding the dynamic values by assigning a static service rating. If the dynamic rating on any alternate node drops below the specified static rating, a static service rating value can be used to direct server users temporarily away from or toward a particular node. Note that load balancing does not take place when static ratings are used on all nodes.

To specify a rating of 195 for the service SALES, use the following command:

```
CREATE SERVICE SALES /STATIC_RATING=195
```

This command disables the dynamic service rating generated by the LAT port driver.

5.6 Advertising Services

All services that you create for your service node are advertised in the multicast messages processed by your node. Advertisement of the services begins when the LAT port driver has been started using the LATCP START NODE command:

```
START NODE
```

If new services are added while the driver is running, the multicast messages are sent out immediately over the network to announce the new services.

5.7 Editing Your System Startup Files

To configure your service node automatically upon system startup, change your startup procedure to invoke the LTLOAD.COM command file. This command file is located in the SYS$MANAGER directory. This section describes the procedures for invoking the LTLOAD.COM file on individual service nodes and on VAXcluster nodes.

5.7.1 Invoking the LTLOAD.COM File on Individual Nodes

To invoke the LTLOAD.COM file, put a command line in your startup procedure. Where you put the command line depends on whether your node is running DECnet. If DECnet is present, insert this command after invoking STARTNET.COM. If DECnet is not present, execute the LTLOAD.COM file when the node is ready to accept interactive users. The command is:

```
$ @SYS$MANAGER:LTLOAD.COM
```

You can also create a command file using the commands necessary to set up and start print queues for remote printers. To configure the remote printers automatically upon system startup, change your startup procedure to include this file, as described in Chapter 6.

5.7.2 Invoking the LTLOAD.COM File on VAXcluster Nodes

VAXclusters can have LAT nodes that have identical LAT configurations. In this case, use a common LTLOAD.COM file and pass the cluster service name as a command line parameter when LTLOAD.COM is invoked. Edit the COMMON_STARTUP.COM file, and include a command line similar to the one in the following example after DECnet has been started:

```
$ @SYS$COMMON:[SYSMGR]LTLOAD.COM SALES "/ENABLE=(24,25)"
```

This command passes the cluster service name, SALES, to all nodes in the cluster and enables groups 24 and 25.

VAXclusters can also have LAT nodes that have different LAT configurations, such as the applications ports. In this case, edit a node-specific SYSTARTUP.COM file to invoke LTLOAD.COM after DECnet has been started, as shown in Example 5-3.

Example 5-3 Invoking LTLOAD.COM on a VAXcluster Node

```
$ ! This is a site specific startup command procedure for node DUKE
$ ! on the SALES VAXcluster.
$ !
$ ! Invoke the common system startup procedure
$ !
$   @SYS$COMMON:[SYSMGR]COMMON_STARTUP.COM
$ !
$ ! Start up DECnet
$ !
$   @SYS$MANAGER:STARTNET.COM
$ !
$ ! Define the Login Announcement
$ !
$   DEFINE/SYS/EXEC SYS$ANNOUNCE -
    "DUKE-A Member of the Sales VAXcluster"
$ !
$ ! Start LAT
$ !
$   @SYS$MANAGER:LTLOAD.COM
$ !
```

5.8 Interactive Use of LATCP

To modify any of the node or service characteristics, enter LATCP commands interactively. Only one user at a time can run LATCP. (To invoke LATCP, you must have CMKRNL privileges.)

Edit the LTLOAD.COM file with the service node characteristics that you want to remain in effect when the system is shut down and restarted. Your service node characteristics in the LTLOAD.COM file are initiated upon system startup.

The service node characteristics specified interactively with the LATCP SET command remain in effect until your service node is shut down.

NOTE: Before you use the LATCP commands interactively, you must load and start the LAT port driver either manually or by executing LTLOAD.COM. If DECnet is going to be started on your node, you must start DECnet before starting the LAT port driver.

5.8.1 Invoking LATCP

Before invoking LATCP, you need CMKRNL privileges. To invoke LATCP, define a symbol for LCP, as follows:

```
$ LCP :== $LATCP
```

After defining the foreign command LCP, you can enter LATCP commands at the DCL prompt by starting each command with the expression LCP. For example:

```
$ LCP SET NODE DUKE
```

To use the default SYS$NODE logical name, enter a LATCP command at the DCL prompt, as follows:

```
$ LCP SET NODE
```

Alternatively, you can define a symbol, such as LCP, then enter that symbol to invoke LATCP, as follows:

```
$ LCP
```

In this case, the system issues the LCP> prompt, and you can enter multiple LATCP commands without typing LCP before each command. For example:

```
LCP> SET NODE DUKE
LCP> START NODE
```

5.8.2 Using HELP for LATCP Commands

With the LATCP HELP command, you can get general information on all of the LATCP commands. In addition, you can get syntax information for each LATCP command. To get syntax information for a LATCP command, specify HELP and the command name, as follows:

```
LCP> HELP SET NODE
```

This command displays the format and qualifiers for the SET NODE command.

5.8.3 Exiting from LATCP

To exit from the LATCP utility and return to the DCL command level, enter the following command:

```
LCP> EXIT
```

Pressing CTRL/Z has the same effect as the EXIT command.

5.8.4 Loading and Starting the LAT Port Driver

The following procedure is recommended for loading the LAT port driver interactively. Load the driver by using the following SYSGEN command (you must have CMKRNL privileges):

```
SYSGEN> CONNECT LTAO: /NOADAPTER
```

To return to DCL, use the following command:

```
SYSGEN> EXIT
```

To start the LAT port driver, use the following command:

```
LCP> START NODE
```

This command starts the LAT port driver on your service node.

NOTE: Set up your node and service characteristics before starting the LAT port driver.

5.8.5 Stopping the LAT Port Driver

Before stopping the LAT port driver, it is suggested that you use the DCL REPLY command to notify system users. See the STOP NODE command in Chapter 8 for an example. To stop the LAT port driver, use the following command:

```
LCP> STOP NODE
```

This command shuts down the LAT port driver and disconnects all sessions to your node.

5.8.6 Displaying Your Service Node Characteristics

To display the characteristics for your service node, use the following command:

```
LCP> SHOW CHARACTERISTICS
```

This command displays characteristics and service information for your service node. See the SHOW CHARACTERISTICS command in Chapter 8 for an example of the display.

5.8.7 Changing a Service Name

To change a service name, first eliminate the service using the DELETE SERVICE command. Then enter a CREATE SERVICE command to assign a new service name. The following is an example:

```
LCP> DELETE SERVICE SALES
LCP> CREATE SERVICE ACCOUNTING
```

If you do not specify a service name with the DELETE SERVICE command, the service name, which is the translation of the SYS$NODE logical name, is deleted.

5.8.8 Changing a Service Announcement

You can change the service announcement string by using the SET SERVICE command. For example, the following command changes the announcement string for the service SALES to "A Member of the Sales Cluster":

```
LCP> SET SERVICE SALES /IDENT="A Member of the Sales Cluster"
```

5.8.9 Changing a Service Rating

You can change a static service rating to a dynamic service rating by using the following command:

```
LCP> SET SERVICE SALES /NOSTATIC_RATING
```

The service ratings placed in subsequent multicast messages are calculated dynamically on the basis of your service node's activity.

Chapter 6

Setting Up Remote Printers

This chapter discusses the LATCP commands required to associate remote printers with your service node, and the DCL commands necessary to set up a remote printer as a spooled device and to set up print queues. This chapter also discusses troubleshooting for remote printer queues.

At system startup, the LATCP commands necessary to set up the remote printers on your service node will be effective if you entered them previously in the LTLOAD.COM procedure. You then need to create a command procedure for the DCL commands.

An example of the LTLOAD.COM file is shown in Chapter 5. An example of a command procedure for setting up and starting print queues on an individual node is shown in Example 6–1; an example for nodes on a VAXcluster is shown in Example 6–2.

To set up a remote printer for your service node, you need to do the following:

1. Create applications ports on the service node (6.1).
2. Map applications ports to server ports (6.2).
3. Set up Printer Characteristics (6.3).
4. Define the form for the remote printer (6.3.2).
5. Set up the remote printer as a spooled device (6.3.3).
6. Initialize and start the queue(s) (6.3.4).
7. When appropriate, set up remote printing on VAXclusters (6.4).

NOTE: This chapter does not discuss all of the qualifiers used in the examples. Refer to the *VAX/VMS DCL Dictionary* for information on these qualifiers.

6.1 Creating Applications Ports on Service Nodes

The logical device for an application program on your service node is called an applications port. The LATSYM print symbiont on your service node uses an applications port to access a remote printer. Use LATCP commands to create an applications port by specifying a port name in the form LTA*n*:, where *n* is a number from 1 through 9999. The following is an example of creating an applications port:

```
CREATE PORT LTA321: /APPLICATION
```

This command creates the applications port LTA321:

6.2 Mapping Applications Ports to Server Ports and Services

You need to logically associate (map) an applications port with a remote printer on a server. To do this, use the SET PORT command to specify the applications port name and the server name, plus one or both of the following:

- Server port name
- Remote service name

The service name on the server is associated with one or more specific ports on that server.

NOTE: Obtain the server port name, server name, and remote service name from the server manager.

The following example shows the applications port LTA321: (created in the previous example) being mapped to a remote printer by using the SET PORT command. The name of the applications port (LTA321:), the server (LAT1), and the server port (LN03) are specified:

```
SET PORT LTA321: /APPLICATION /NODE=LAT1 /PORT=LN03
```

The /APPLICATION qualifier specifies that the LTA321: port on the service node functions as an applications port.

The next example shows the applications port LTA322: being mapped to a set of remote printers on a server using the SET PORT command. The names of the applications port, the server, and the remote service are specified.

```
SET PORT LTA322: /APPLICATION /NODE=LAT1 /SERVICE=PRINTER
```

The service PRINTER represents an available remote printer.

Use a service name when the server has more than one printer and you don't care which printer the server selects. This gives you a better chance of getting a printer when you need it.

NOTE: You need the server manager to assign a service name to a bank of printers on the server. Not all servers support remote services.

6.3 Setting Up Printer Characteristics for non-Clustered Nodes

You can create a command procedure that configures your remote printers, using DCL commands. Once you configure your remote printers in the command procedure, call the command procedure in your SYS$MANAGER:SYSTARTUP.COM. This ensures automatic configuration for your remote printers on system startup.

Using the command procedure allows you to maintain remote print queues separately from other queues on a node. This separation is useful because queues for local applications devices usually are started before the LTLOAD.COM file is executed, while remote print queues must be started afterward. Separation also reduces the possibility of unintentionally interfering with the other applications devices and queues on your node when you are setting up applications ports and queues for remote printers.

Example 6–1 is a sample of a command procedure for setting up remote printer characteristics and for starting the queues for those printers.

Example 6-1 Command Procedure for Configuring Remote Printers

```
$ ! This command procedure sets up the local characteristics and
$ ! print queues for remote printers.  The remote printers should have
$ ! been mapped to the applications ports by the LTLOAD.COM command
$ ! procedure.  NOTE: The queue manager must be running before executing
$ ! this file.
$ !
$ ! Set up local characteristics for the remote printers.
$
$   SET TERMINAL LTA321: /PERM /DEVICE=LN03 /WIDTH=255 /PAGE=60 -
                 /LOWERCASE /NOBROAD
$   SET TERMINAL LTA322: /PERM /DEVICE=LA210 /WIDTH=255 /PAGE=66 -
                 /NOBROAD
$
$ ! Set the remote printers spooled.
$
$   SET DEVICE LTA321: /SPOOLED=(LN03$PRINT,SYS$SYSDEVICE:)
$   SET DEVICE LTA322: /SPOOLED=(LA210$PRINT,SYS$SYSDEVICE:)
$
$ ! Define a form to use with the remote printers.  Be sure to use a
$ ! form number that has not already been used.
$
$   DEFINE/FORM LN_FORM 10 /WIDTH=80 /STOCK=DEFAULT /TRUNCATE
$
$ ! Initialize the remote printer queues.
$ ! The following assumes that the queue manager has been started.
$
$   INITIALIZE/QUEUE /START /PROCESSOR=LATSYM /FORM_MOUNTED=LN_FORM -
              /RETAIN=ERROR /DEFAULT=(NOBURST,FLAG=ONE,NOTRAILER) -
              /RECORD_BLOCKING LN03$PRINT /ON=LTA321:
$   INITIALIZE/QUEUE /START /PROCESSOR=LATSYM /RETAIN=ERROR -
              /DEFAULT=(NOBURST,FLAG=ONE,NOTRAILER) /RECORD_BLOCKING -
               LA210$PRINT /ON=LTA322:
$
```

6.3.1 Setting Up Terminal Characteristics for Remote Printers

Once you map an applications port on your service node to a remote printer, (see Section 6.2), use the DCL SET TERMINAL command to specify terminal characteristics for the remote printer.

Refer to Appendix B for a list of supported printers and the associated terminal characteristics that you must specify for each printer. When you specify a device type, the characteristics for that device are automatically assigned. In the following example, the SET TERMINAL command sets up the automatic line width and page length for the LA210 printer.

```
$ SET TERMINAL LTA322: /PERMANENT /DEVICE=LA210 /NOBROADCAST
```

Alternatively, you can override the automatic line width and page length values for a device by entering different width and page values. For example, a laser printer can load special font files if you override the printer's automatic page width using the following command:

```
$ SET TERMINAL LTA322: /PERMANENT /DEVICE=LN03 /WIDTH=255 -
  /PAGE=66 /NOBROADCAST
```

This command sets the page width for the LN03 printer to 255.

Do not use the SET TERMINAL/WIDTH command to specify paper width, instead use the DEFINE/FORM command described in Section 6.3.2.

NOTE: The /PERMANENT and /NOBROADCAST qualifiers must be specified for remote printers.

6.3.2 Defining a Form for a Remote Printer

The DEFINE/FORM command defines a form name and number, as well as the physical paper stock. Do not issue DEFINE/FORM unless the queue manager is running. Define a new form if no appropriate form exists on your system. To look at the form names and form numbers currently defined on your service node, use the SHOW QUEUE/FORM command.

All DEFINE/FORM commands should resemble the following example, which defines a form named LN_FORM and numbered 10:

```
$ DEFINE/FORM LN_FORM 10  /WIDTH=80 /STOCK=DEFAULT /TRUNCATE
```

When a print job is submitted without a /FORM qualifier, the symbiont uses the default stock for the job. This is important, because the stock of a job must match the stock of the queue, or the job remains pending on the queue and does not print. The /STOCK=DEFAULT qualifier gives the system-wide default for paper stock to the form. Then, by assigning this form to a queue using INITIALIZE/QUEUE (see Section 6.3.4), you allow jobs without a /FORM qualifier to print, because both job and queue require the default stock. Note that if a user specifies a form other than the queue's form, the job still can print if the two forms use an identical stock. In this case, however, the flag page can be incorrectly formatted.

By specifying the default stock for the form that the queue uses, you reduce the likelihood of such problems. Other methods for avoiding problems with incompatible stock designations are discussed in the section on printing jobs (Section 6.5) and on troubleshooting (Section 6.6).

NOTE: The /STOCK=DEFAULT parameter works only when a system-wide default exists. If printers were not used on your system before, a default stock may not exist for the system. If your system lacks this default, add the following command to the procedure for setting up remote printers or at the top of the file you use for defining print forms:

```
$ DEFINE/FORM DEFAULT 0 /WIDTH=132 /STOCK=DEFAULT
```

6.3.3 Setting Up a Remote Printer as a Spooled Device

Use the SET DEVICE/SPOOLED command to set up a remote printer as a spooled device. For example, the following command sets the applications port LTA322: as a spooled device associated with the queue LA210$PRINT:

```
$ SET DEVICE LTA322: /SPOOLED=(LA210$PRINT,SYS$SYSDEVICE:)
```

6.3.4 Initializing Queues for Remote Printers

The INITIALIZE/QUEUE command in the following example initializes the queue for a remote printer.

```
$ INITIALIZE/QUEUE /START /PROCESSOR=LATSYM /FORM_MOUNTED=DEFAULT -
   /RETAIN=ERROR /DEFAULT=(NOBURST,FLAG=ONE,NOTRAILER) -
   /RECORD_BLOCKING LA210$PRINT /ON=LTA322:
```

Do not issue the INITIALIZE/QUEUE command unless the queue manager is running. For information on the queue manager, refer to the *VAX/VMS System Manager's Reference Manual.*

Among other things, the qualifiers used with the INITIALIZE/QUEUE command start and set up remote printers, as follows:

- /START starts the queue.
- /PROCESSOR=LATSYM specifies the LAT print symbiont, which is required for remote printers.
- /FORM_MOUNTED=DEFAULT assigns a default form to the queue. Defining a default form is discussed in Section 6.3.2.

  Note that it is not necessary to include the /FORM_MOUNTED qualifier if the queue uses the system-wide default form.
- LA210$PRINT names the queue.
- /ON=LTA322: specifies an applications port associated with the remote printer. The applications port must have been previously mapped to the printer with the SET PORT command (see Section 6.2).
- /RETAIN=ERROR ensures that error status messages are generated.

An error status at a remote device cannot be passed back directly to the queue manager. Thus, when a connection with a server cannot be made, no error status is sent to the user. However, specifying the /RETAIN=ERROR qualifier in the INITIALIZE/QUEUE command ensures that you can get status for such an error. When this qualifier is in effect, a job that fails because of a problem on the network is labeled with an error status message, such as "Checkpointed:". See Section 6.6 on troubleshooting for additional information about checkpointed errors.

6.4 Setting Up Remote Printing on VAXclusters

On a VAXcluster, it is recommended that you configure applications ports on at least two nodes, so that a redundant path to the device is available in the event of a failure of a cluster node. To configure a remote-printer applications port on a cluster node, include LATCP CREATE PORT and SET PORT commands for that port in the node's LTLOAD.COM file.

6.4.1 Guidelines for Command Procedures on VAXclusters

On VAXclusters, where management of remote printers can become complex, the following order of events must be adhered to, although additional events can occur between the listed events:

- DECnet is started.
- LAT is started, applications ports for remote printers are established when you invoke the LTLOAD.COM file.

  You can have node-specific LTLOAD.COM files for nodes on the cluster with different LAT characteristics. You can alternatively have a cluster common LTLOAD.COM file for nodes that have identical LAT characteristics. Note that not all nodes require applications ports defined for remote devices.
- Queues to remote printers are set up and started (queue manager must be running).

6.4.2 Queues in a VAXcluster Environment

Queues are used in several ways in a VAXcluster environment. On one node, you can set up a device-specific queue that points to one remote printer on a server. A device-specific queue always points to a particular device. The remote printer must be mapped to an applications port on the node. To initialize the device-specific queue on one node, define the device-specific queue in the startup command procedure for that node.

You can also set up a generic queue on one node to handle requests for several remote printers. On this same node, you need to first create a device-specific queue for each printer. The generic queue points to the printers via each printer's device-specific queue. In this case, there is one path to each remote printer. Each printer is mapped to an applications port on this node. To initialize a generic queue on one node in the cluster, define the device-specific queues and the generic queue in the startup command procedure for that node.

Finally, you can set up a generic queue that provides a redundant path via several nodes to several remote printers. The generic queue points to the remote printers via each printer's device-specific queue. Each printer must have a device-specific queue on each participating node. Notice that each printer must be mapped to the same applications port on each participating node. Define the generic queue and the device-specific queue in the startup command procedure for each node using the generic queue.

Example 6–2 consists of a generic queue for one printer and a generic queue for two printers. See Section 6.3 for an explanation of the other commands used in this example.

Refer to the *VAX/VMS Guide to VAXclusters* and the *VAX/VMS System Manager's Reference Manual* for detailed information about generic output queues.

Example 6-2 Command Procedure for Configuring Remote Printers Using a Generic Queue

```
$ ! This is an example of a cluster command procedure which sets up
$ ! characteristics and queues for remote printers.
$ !
$ ! This file assumes that two nodes in the cluster access the remote
$ ! devices, and that only those nodes call this file.
$ !
$ ! Compute the name of the executing node.
$ !
$ NODE = F$GETSYI("NODENAME")
$ !
$ DUKE_START = "/NOSTART"
$ EARL_START = "/NOSTART"
$ !
$ ! Redefine one of the previous symbols.
$ !
$ 'NODE'_START = "/START"
$ !
$ ! Set up local characteristics for the remote printers.
$ !
$ ! This procedure assumes that the remote printers have been mapped
$ ! to the same applications port on each node that accesses them.
$ !
$ SET TERMINAL LTA401: /PERM /DEVICE=LA38 /WIDTH=80 /PAGE=60 -
        /LOWERCASE /NOBROADCAST
$ SET TERMINAL LTA402: /PERM /DEVICE=LA120 /NOBROADCAST
$ !
$ ! Set the remote printers spooled.
$ !
$ SET DEVICE LTA401: /SPOOLED=('NODE'$38PRINT,SYS$SYSDEVICE)
$ SET DEVICE LTA402: /SPOOLED=('NODE'$120PRINT,SYS$SYSDEVICE)
$ !
$ ! Define a form to use with the remote printers.  Be sure to use a
$ ! form number that has not already been used.
$ !
$ DEFINE/FORM LA_FORM 10 /WIDTH=80 /STOCK=DEFAULT /TRUNCATE
$ !
$ ! Initialize the remote printer queues.
$ ! The following assumes that the queue manager has been started.
$ !
$ INITIALIZE/QUEUE /PROCESSOR=LATSYM /FORM_MOUNTED=LA_FORM /RETAIN=ERROR -
    /DEFAULT=(NOBURST,FLAG=ONE,NOTRAILER) /RECORD_BLOCKING -
    /ON=DUKE::LTA401: 'DUKE_START' DUKE$38PRINT
```

(Continued on next page)

Example 6-2 (Cont.) Command Procedure for Configuring Remote Printers Using a Generic Queue

```
$ !
$ INITIALIZE/QUEUE /PROCESSOR=LATSYM /RETAIN=ERROR -
    /DEFAULT=(NOBURST,FLAG=ONE,NOTRAILER) /RECORD_BLOCKING -
    /ON=DUKE::LTA402: 'DUKE_START' DUKE$120PRINT
$ !
$ INITIALIZE/QUEUE /PROCESSOR=LATSYM /FORM_MOUNTED=LA_FORM /RETAIN=ERROR -
    /DEFAULT=(NOBURST,FLAG=ONE,NOTRAILER) /RECORD_BLOCKING -
    /ON=EARL::LTA401: 'EARL_START' EARL$38PRINT
$ !
$ INITIALIZE/QUEUE /PROCESSOR=LATSYM /RETAIN=ERROR -
    /DEFAULT=(NOBURST,FLAG=ONE,NOTRAILER) /RECORD_BLOCKING -
    /ON=EARL::LTA402: 'EARL_START' EARL$120PRINT
$ !
$ ! Initialize the cluster wide generic queues.
$ !
$ ! A generic queue with one printer.
$ !
$ INITIALIZE/QUEUE /START /GENERIC=(DUKE$38PRINT, EARL$38PRINT) -
    SYS$38PRINT
$ !
$ ! A generic queue with two printers.
$ !
$ INITIALIZE/QUEUE /START /GENERIC=(DUKE$38PRINT, DUKE$120PRINT, -
    EARL$38PRINT, EARL$120PRINT) TERMINAL$120PRINT
```

6.5 Printing Jobs

After you set up remote printers, a user on your service node can issue a DCL command to print a file. The command in the following example prints the file LATDOC.MEM on the remote LA120 printer:

```
$ PRINT LATDOC.MEM /QUEUE=SYS$120PRINT
```

To avoid problems with flag pages or with queues whose form does not use the default stock, assign a system-wide global symbol for each queue. This symbol must be assigned in the system-wide login command procedure, which defaults to SYS$MANAGER:SYLOGIN.COM. Select a symbol, such as 38PRINT, to represent a PRINT command specifying a given queue and its form type. For example:

```
$ 38PRINT :== PRINT /QUEUE=SYS$38PRINT /FORM=LA_FORM
```

Instruct users to specify this symbol when they send printing jobs to remote queues. The symbol provides the user with an easy and accurate way to print the files, as in the following example:

```
$ 38PRINT LATDOC.MEM
```

6.6 Troubleshooting Problems with a Remote Printer Queue

Errors can occur on a print queue for a number of reasons. Several causes of errors and suggested responses to some of these are discussed in this section.

The effect of errors upon the queue varies according to the type of error. An error can cause an individual job to remain in a pending state. However, some errors cause the entire queue to pause or stop. When the queue stops, all jobs on the queue remain in the pending state until the queue is restarted. Before you restart a queue, determine and correct the condition that caused it to stop.

Three general types of errors are discussed in this section:

- **Checkpointed errors**—Errors that stop the queue and checkpoint the current job (holding it for resubmission when the queue restarts).

  These errors are caused by some problem between the service node and the server, such as a network problem or an incorrect service name assigned to the remote device.

- **Suspended printing errors**—Errors that stall the queue suspend printing jobs temporarily (permitting partially printed jobs to be left on the queue and completed later).

  These errors are caused by a problem in the server and/or the printer while a job is printing, such as the printer running out of paper.

- **Pending errors**—Errors that force an individual job to remain in a pending state but leave the queue functioning.

  These errors are caused by a print command that specifies a printer form whose paper stock is incompatible with the stock specified for the queue.

  NOTE: Jobs that remain pending for a long time have not necessarily experienced an error. Use the SHOW QUEUE/FULL command to determine whether a delay in printing was requested by the user.

These three types of errors and suggested methods of responding to them are discussed in greater detail in the remainder of this section.

6.6.1 Checkpointed Errors

Checkpointed errors are caused by the failure to create a connection.

Two types of errors can prevent a connection, and one type can terminate an existing connection:

- Incorrect names assigned for the LTA*n*: port

  The node name does not match the name of the server, the applications port name does not match the name of a server port that allows remote access, or the service name does not match the name of a service offered by the server. Furthermore, if both a service name and a port name are specified, the service must be offered on the port. Check with the server manager or network manager to verify that the correct names are specified on the service node.

- Host-initiated request never received

  For example, in some cases, the server is on a different Ethernet or on an Ethernet segment that is currently disconnected. In other cases, the server is inactive when the request is sent.

- Connection abnormally terminated because of a network or server problem

  For example, a job is printing when a network problem causes the circuit to go down.

To ensure that checkpointed jobs are identified by an error status message, use the /RETAIN=ERROR qualifier in the INITIALIZE/QUEUE command (see Section 6.3.4). A checkpointed error status message makes it easy to identify a failed job. Refer to Chapter 7 for additional information about error status messages. With the job number you can delete a job, if you wish to submit it to another queue rather than wait until the problem with the stopped queue is resolved. However, if you prefer, you can leave the job on the queue.

After a checkpointed error occurs, the queue stops or pauses. Identify and remove the cause of the error before restarting the queue. Once the underlying problem is resolved, reset and restart the queue, as follows:

```
$ STOP/QUEUE/RESET queue-name
$ START/QUEUE queue-name
```

You can resubmit the checkpointed job with the following DCL command:

```
$ SET QUEUE queue-name/ENTRY=entry-number/NOHOLD
```

Refer to the *VAX/VMS DCL Dictionary* for more details on the START/QUEUE command.

6.6.2 Suspended Printing Errors

Suspended printing errors occur if the job fails to print completely after it was accepted by a server. These errors can result from a problem on the network, server, or printer. These printing errors cause the queue to stall, but the job is only temporarily suspended. Once the problem is corrected, the job resumes printing.

6.6.3 Pending Errors

A pending error can occur if the stock associated with a job differs from the stock associated with a queue. Any print job whose stock is incompatible with the stock of the queue remains in a pending state. However, no error message is generated; the job continues to be labeled simply as "Pending" until it is deleted or until the stock assigned to the queue is changed. A pending error affects only the individual job. The queue does not stop, so error-free jobs can continue to print. To resolve a pending error, the stock of the queue must be changed to match the job or the job must be deleted and submitted to a queue with compatible stock.

Methods for reducing the occurrence of pending errors include:

- Assigning the default stock when defining the form to be assigned to a queue (see Section 6.3.2).

  When a queue uses the default stock, users need not specify a form in print commands, thereby reducing a likely source of error.

- Creating a system-wide global symbol for printing to a queue with a nondefault stock (see Section 6.5).

  When users enter this symbol, they avoid pending errors.

Note that the success of a singular preventative method depends on users avoiding print commands that specify forms requiring unavailable stock. Therefore, occasionally a user who specifies a form can experience a problem with a print job. In this case, the following are possible solutions:

- You or the user can delete a job requiring an incompatible stock and either resubmit the job to the same queue (while specifying a form using the suitable stock) or submit the job to a different queue whose stock is what the user specified in the original PRINT/FORM command.

- If an alternative queue is not available and printing a job with a different stock is essential, change the stock of the printer and/or the queue. Changing the stock involves the following steps:
  - Stopping the queue
  - Changing the physical stock in the printer, if desired
  - Changing the stock specification of the queue to match the new stock
  - Restarting the queue

Chapter 7

LAT Port Driver QIO INTERFACE

7.1 VAX/VMS Application Programs and LAT Devices

The LAT software allows application programs to use remote devices on a server(s). A remote device, such as a printer, can be shared over the LAT network. A typical LAT network is shown in Chapter 4. Before a remote device can be accessed by an application program, the remote device needs to be mapped to an applications port on your VMS system. See Chapter 6 for a discussion about mapping. Once the remote device is mapped, the application program can establish and terminate a connection to that device. The connection is made through the applications port on your node that is associated with the remote device.

This chapter discusses the QIO interface to the LAT port driver and the function code modifiers that you use to establish and terminate connections to remote devices. You must use these QIO functions to establish a connection to a remote device from an application program. DIGITAL does not support any other methods of connection.

7.2 Using VAX/VMS Function Codes and Modifiers for LAT Devices

The VAX/VMS I/O terminal function codes are described in the *VAX/VMS I/O Reference Manual.* You cannot use all of those function codes for LAT devices. The VAX/VMS terminal port function code modifiers that you can use for LAT devices are:

- All read function code modifiers.
- All write function code modifiers.

VAX/VMS does not support the following SET MODE or SET CHARACTERISTICS function code modifiers for LAT devices:

- TT$M_MODEM
- TT2$M_SETSPEED

- TT$M_ALTRPAR
- TT$M_BREAK
- TT$M_ALTFRAME
- IO$M_LOOP
- IO$M_UNLOOP

Read and write modem function code modifiers are not supported for LAT devices.

Flow control to the physical device is handled by the terminal server instead of the host. A separate flow control mechanism exists between the server and host. The following terminal characteristics do not apply to LAT terminals:

- TT$M_HOSTSYNC
- TT$M_READSYNCH
- TT$M_TTSYNC

7.3 LAT Port Driver Function Code

The LAT port driver accommodates I/O requests from application programs for connections to remote devices (for example, a graphics printer) on a server. A request for the LAT port driver must include the VMS IO$_TTY_PORT function code. IO$_TTY_PORT allows the VMS terminal device driver to forward a LAT-specific request to the LAT port driver. In addition, the request must include a LAT port function code modifier.

7.3.1 LAT Port Function Code Modifiers

The LAT port function code modifiers are:

- IO$M_LT_CONNECT
- IO$M_LT_DISCON

IO$M_LT_CONNECT is the function code modifier that requests the LAT port driver to make a connection to a remote device on a server. IO$M_LT_DISCON is the function code modifier that requests the LAT port driver to terminate the LAT connection to the remote device.

When an application program issues an IO$M_LT_CONNECT request for a connection to a remote device, one of the following situations occurs:

- The connection is established. This situation occurs if the connection is successful. You can use the device.

- The connection is timed out. This situation occurs if the server is not available, or if an incorrect server name is specified.
- The connection is rejected. This situation occurs if an incorrect port name or service is specified or if the server, service, or remote port is disabled.
- The request is queued at the server. This situation occurs if the remote port is busy when requested. In this case, the QIO is not completed until the connection is established, rejected, or timed out.

  When a connection request is queued at the server, the QIO function does not complete until the request is removed from the queue. The $CANCEL system service does not cancel the queued connection. To cancel the connection request, issue an IO$_TTY_PORT!IO$M_LT_DISCON disconnect QIO. Include an exit handler in your application programs that issues the disconnect QIO on exit. Issuing the disconnect QIO to an already disconnected device does not cause any problems.

The LAT port driver cannot attempt to connect to a remote device under the following circumstances:

- The request is not to an applications port. In this case, the QIO request was probably issued to an interactive port. The LAT port driver rejects the request.
- The requested applications port is busy. In this case, the QIO requests an applications port that is already in use. The LAT port driver rejects the request.

After you issue an IO$_TTY_PORT!IO$M_LT_DISCON (disconnect QIO), the applications port's UCB momentarily goes off line. If you issue a connect QIO for a remote device immediately after a disconnect QIO, it is also possible that the connect QIO may return a SS$_DEVACTIVE status. In this situation, retry the connect QIO.

7.3.2 Hangup Notification

To ensure that the terminal driver notifies application programs that are writing data of an abnormal connection termination, enable a CTRL/Y AST on the channel. To do this, use the IO$_SETMODE function code and IO$M_CTRLYAST function code modifier. Note that VMS does not return an AST parameter to the CTRL/Y AST routine.

When an application program with a pending read request has an abnormal LAT connection termination, the VMS terminal driver returns a SS$_HANGUP hangup notification in the first word of the I/O status block.

7.3.3 I/O Status Block

When an application program makes an I/O request for a connection to a remote device on a server, the LAT port driver puts status information about the request (see Table 7-1) into the first word of the I/O status block.

If the server rejects the request, the LAT port driver returns a numeric LAT rejection reason code in the second word of the I/O status block. This numeric code represents the reason for the rejection. Table 7-2 describes the LAT rejection reason codes.

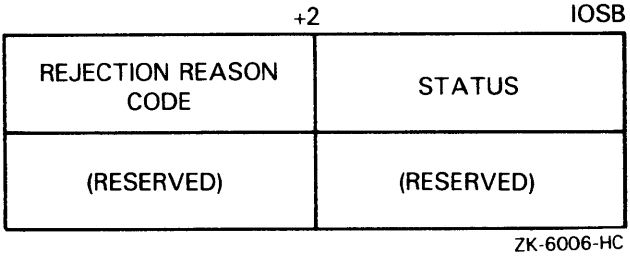
Table 7-1 IO$M_LT_CONNECT Status

Event	Status	Explanation
Connection established	IOSB = SS$_NORMAL	The connection is successful, and the device is ready to use.
Connection timeout	IOSB = SS$_TIMEOUT	The connection timed out. The server is not available or an incorrect server name was specified. The timeout period is 5 seconds.
Connection rejected	IOSB = SS$_ABORT IOSB+2 = LAT reject reason code	The connection cannot be made. The LAT port driver updates the I/O status block.
Request is not to an applications port	SS$_ILLIOFUNC No status in IOSB. QIO is rejected immediately.	The QIO request is not to an applications port. The LAT port driver rejects the request.
Connection already established on port	SS$_DEVACTIVE No status in IOSB. QIO is rejected immediately.	The QIO request is for an applications port already in use. The LAT port driver rejects the request.

NOTE: If a request for a connection is queued on the server, the QIO is not completed until the connection is established, rejected, or timed out.

The LAT port driver puts status information about a connection request into the first word of the I/O status block. An example of how a status block might look after an I/O request is shown in Figure 7-1.

Figure 7-1 First Word of the I/O Status Block



The rejection reason code for abort status at the IOSB+2 refers to the rejection codes shown in Table 7-2. This field is only valid when a QIO request has the abort status.

Table 7-2 LAT Rejection Reason Codes for Abort Status

Value	Reason
0	Unknown
2	System shutdown in progress
5	Insufficient resources at server
6	Port or service in use
7	No such service
8	Service is disabled
9	Service is not offered on the requested port
10	Port name is unknown
13	Immediate access rejected
14	Access denied
15	Corrupted request
16	Requested function is not supported
17	Session cannot be started
18	Queue entry deleted by server
19	Illegal request parameters

7.4 Programming Example

In this example, the program requests a connection to an applications port. The program uses the LAT/VMS port function code and the function code modifiers for the LAT port driver to solicit the connection to the applications port.

Example 7-1 Application Program for Connecting to a Remote Device

```
         .TITLE   LAT APPLICATIONS PORT PROGRAMMING EXAMPLE
         .IDENT   /1.0/
;**************************************************************
;
;                 LAT APPLICATIONS PORT PROGRAM
;
;**************************************************************
         .SBTTL   DECLARATIONS
;
; DEFINE SYMBOLS
;
         $IODEF                                ; I/O FUNCTION CODES
         $QIODEF                               ; QIO DEFINITION CODES
;
; DECLARE EXIT HANDLER CONTROL BLOCK
;
EXIT_HANDLER_BLOCK:
         .LONG    0                            ; SYSTEM USES THIS FOR POINTER
         .LONG    EXIT_HANDLER                 ; ADDRESS OF EXIT HANDLER
         .LONG    1                            ; ARGUMENT COUNT FOR HANDLER
         .LONG    STATUS                       ; DESTINATION OF STATUS CODE
STATUS:           .BLKL    1                   ; STATUS CODE FROM $EXIT
;
; ALLOCATE TERMINAL DESCRIPTOR AND CHANNEL NUMBER STORAGE
;
TT_DESC:
         .ASCID   /SYS$INPUT/                  ; NAME OF TERMINAL
TT_CHAN:
         .BLKW    1                            ; TT CHANNEL NUMBER STORAGE
LT_DESC:
         .ASCID   /LTA700:/                    ; NAME OF LT DEVICE
LT_CHAN:
         .BLKW    1                            ; LT CHANNEL NUMBER STORAGE
;
; APPEND <CR><LF> TO MESSAGE
OUT_MSGLEN = 2
OUT_MSG:
         .ASCII   <CR><LF>
```

(Continued on next page)

Example 7-1 (Cont.) Application Program for Connecting to a Remote Device

```
;
; ALLOCATE INPUT BUFFER
;
IN_BUFLEN = 80
IN_BUF:
        .BLKB   IN_BUFLEN                       ; ALLOCATE CHARACTER BUFFER
IN_IOSB:
        .BLKQ   1                               ; INPUT I/O STATUS BLOCK
SOL_IOSB:
        .BLKQ   1                               ; SOLICT CONNECT QIO I/O STATUS BL
;
; DEFINE CARRIAGE CONTROL SYMBOLS
;
        CR=^X0D                                 ; CARRIAGE RETURN
        LF=^X0A                                 ; LINE FEED
;
; DEFINE OUTPUT MESSAGES
;
; MESSAGES ARE ACCESSED BY INDEXING INTO A TABLE OF LONGWORDS
; WITH EACH MESSAGE DESCRIBED BY A MESSAGE ADDRESS AND LENGTH
;
MSG_TABLE:                                      ; TABLE OF MESSAGE ADD. AND LEN.
        .LONG   01$                             ; FIRST MESSAGE ADDRESS
        .LONG   05$                             ; FIRST MESSAGE LENGTH
        .LONG   10$                             ; MESSAGE ADDRESS
        .LONG   15$                             ; MESSAGE LENGTH
        .LONG   20$                             ; MESSAGE ADDRESS
        .LONG   25$                             ; MESSAGE LENGTH
        .BLKQ   2                               ; BLANK MESSAGE CODES
        .LONG   50$                             ; MESSAGE ADDRESS
        .LONG   55$                             ; MESSAGE LENGTH
        .LONG   60$                             ; MESSAGE ADDRESS
        .LONG   65$                             ; MESSAGE LENGTH
        .LONG   70$                             ; MESSAGE ADDRESS
        .LONG   75$                             ; MESSAGE LENGTH
        .LONG   80$                             ; MESSAGE ADDRESS
        .LONG   85$                             ; MESSAGE LENGTH
        .LONG   90$                             ; MESSAGE ADDRESS
        .LONG   95$                             ; MESSAGE LENGTH
        .LONG   100$                            ; MESSAGE ADDRESS
        .LONG   105$                            ; MESSAGE LENGTH
        .BLKQ   2                               ; BLANK MESSAGE CODES
```

(Continued on next page)

Example 7-1 (Cont.) Application Program for Connecting to a Remote Device

```
         .LONG    130$                        ; MESSAGE ADDRESS
         .LONG    135$                        ; MESSAGE LENGTH
         .LONG    140$                        ; MESSAGE ADDRESS
         .LONG    145$                        ; MESSAGE LENGTH
         .LONG    150$                        ; MESSAGE ADDRESS
         .LONG    155$                        ; MESSAGE LENGTH
         .LONG    160$                        ; MESSAGE ADDRESS
         .LONG    165$                        ; MESSAGE LENGTH
         .LONG    170$                        ; MESSAGE ADDRESS
         .LONG    175$                        ; MESSAGE LENGTH
         .LONG    180$                        ; MESSAGE ADDRESS
         .LONG    185$                        ; MESSAGE LENGTH
         .LONG    190$                        ; MESSAGE ADDRESS
         .LONG    195$                        ; MESSAGE LENGTH
;
; MESSAGES
;
01$:     .ASCII   /REASON UNKNOWN/
05$=.-01$
;
10$:     .ASCII   <CR><LF>/CONNECTION ESTABLISHED/
15$=.-10$
;
20$:     .ASCII   /SYSTEM SHUTDOWN IN PROGRESS/
25$=.-20$
;
50$:     .ASCII   /INSUFFICIENT RESOURCES/
55$=.-50$
;
60$:     .ASCII   /PORT OR SERVICE IN USE/
65$=.-60$
;
70$:     .ASCII   /NO SUCH SERVICE/
75$=.-70$
;
80$:     .ASCII   /SERVICE IS DISABLED/
85$=.-80$
;
90$:     .ASCII   /SERVICE NOT OFFERED BY REQUESTED PORT/
95$=.-90$
;
100$:    .ASCII   /PORT NAME IS UNKNOWN/
105$=.-100$
```

(Continued on next page)

Example 7-1 (Cont.) Application Program for Connecting to a Remote Device

```
;
130$:    .ASCII   /IMMEDIATE ACCESS REJECTED/
135$=.-130$
;
140$:    .ASCII   /ACCESS DENIED/
145$=.-140$
;
150$:    .ASCII   /CORRUPTED REQUEST/
155$=.-150$
;
160$:    .ASCII   /REQUESTED FUNCTION IS NOT SUPPORTED/
165$=.-160
;
170$:    .ASCII   /SESSION CANNOT BE STARTED/
175$=.-170$
;
180$:    .ASCII   /QUEUE ENTRY DELETED BY LOCAL NODE/
185$=.-180$
;
190$:    .ASCII   /ILLEGAL REQUEST PARAMETERS/
195$=.-190$
;
NOTCON:           .ASCII   <CR><LF>/CONNECTION REJECTED - /
NOTCONL=.-NOTCON
;
; STATIC QIO PACKETS FOR MESSAGE OUTPUT USING QIO$_G FORM
;
WRITE_QIO:
         $QIO     FUNC=IO$_WRITEVBLK!IO$M_BREAKTHRU!IO$M_REFRESH,-
                  EFN=1
;
ERROR_QIO:
         $QIO     FUNC=IO$_WRITEVBLK!IO$M_BREAKTHRU!IO$M_REFRESH,-
                  EFN=1
;
;
         .SBTTL   MAIN ROUTINE
;++
;
; FUNCTIONAL DESCRIPTION:
```

(Continued on next page)

Example 7-1 (Cont.) Application Program for Connecting to a Remote Device

```
;
;*****************************************************
;
;               MAIN PROGRAM ROUTINE
;
;*****************************************************
;
;       The following code assigns a channel to the
;       applications port and attempts to create a connection
;       to that port.  The connection status is displayed on
;       the users terminal.  Input from the users terminal
;       is output on the applications port:  ^C input from the
;       user terminates the program.
;
; INPUT PARAMETERS:
;       None.
;
; OUTPUT PARAMETERS:
;       None.
;
;--
START:
        .WORD                           ; ENTRY MASK
;
;       Assign channels
;
        $ASSIGN_S       DEVNAM=TT_DESC,- ; ASSIGN CHANNEL TO USERS
                        CHAN=TT_CHAN    ; TERMINAL
        BLBS    R0, 10$                 ; NO ERROR IF SET
        BRW     ERROR                   ; ELSE, ERROR
10$:    $ASSIGN_S       DEVNAM=LT_DESC,- ; ASSIGN CHANNEL TO LT DEVICE
                        CHAN=LT_CHAN
        BLBS    R0, 20$                 ; NO ERROR IF SET
        BRW     ERROR                   ; ELSE, ERROR
;
;       Enable ^C on user terminal and ^Y on applications port.
;       Post read to user terminal and solicit connection to
;       applications port.
;
20$:    BSBW    ENABLE_CTRLCAST         ; ENABLE CONTROL C AST'S
        BSBW    SOL_CONNECT             ; TRY TO CONNECT TO LT DEVICE
        BSBW    ENABLE_READ             ; QUEUE READ
30$:    NOP                             ;
        BRB     30$                     ; KEEP LOOPING UNTIL ^C
        RET
```

(Continued on next page)

Example 7-1 (Cont.) Application Program for Connecting to a Remote Device

```
;
         .SBTTL   ENABLE_CTRLYAST - Enable CTRLYAST on applications port
;++
; FUNCTIONAL DESCRIPTION:
;
;        Routine to allow hangup notification.  This routine enables
;        CTRLY AST delivery for the applications port.  This routine will
;        be called if an abnormal termination occurs to the remote
;        device.
;
; INPUT PARAMETERS:
;
;        None.
;
; OUTPUT PARAMETERS:
;
;        None
;
;--
ENABLE_CTRLYAST:
         $QIOW_S CHAN=LT_CHAN,-
                  FUNC=#IO$_SETMODE!IO$M_CTRLYAST,-
                  P1=HANGUP,-                   ; AST ROUTINE ADDRESS
                  P3=#3                         ; USER MODE
         BLBS     R0, 10$                       ; NO ERROR IF SET
         BRW      ERROR
10$:     RSB
;
         .SBTTL   HANGUP - AST Routine for Control Y
;++
;
; FUNCTIONAL DESCRIPTION
;
;        AST routine to execute when ^Y status is returned for the
;        applications port.  This status is returned when the
;        connection to the remote device is abnormally terminated.
;
; INPUT PARAMETERS:
;        None
;
; OUTPUT PARAMETERS:
;        None
;
;--
```

(Continued on next page)

Example 7-1 (Cont.) Application Program for Connecting to a Remote Device

```
;
HANGUP:
        .WORD   ^M<>
        MOVZWL  #SS$_HANGUP,R0                  ; INDICATE HANGUP
        BRW     ERROR                           ; AND EXIT
;
        .SBTTL  ENABLE_READ - QUEUE A READ TO THE TERMINAL
;++
;
; FUNCTIONAL DESCRIPTION
;
;       Routine to queue a read to the terminal.  The queued
;       read will not affect writes due to the fact that
;       breakthru has been set for writes.
;
;
; INPUT PARAMETERS:
;       None
;
; OUTPUT PARAMETERS:
;       None
;
;--
;
ENABLE_READ:
        $QIO_S  CHAN=TT_CHAN, -                 ; MUST NOT BE QIOW FORM
                FUNC=#IO$_READVBLK,-
                IOSB=IN_IOSB,-
                ASTADR=READAST,-
                P1=IN_BUF,-
                P2=#IN_BUFLEN
        BLBS    R0,10$                          ; NO ERROR IF SET
        BRW     ERROR
10$:    RSB
;
        .SBTTL  READAST - AST Routine for Read Completion
;++
;
; FUNCTIONAL DESCRIPTION
```

(Continued on next page)

Example 7-1 (Cont.) Application Program for Connecting to a Remote Device

```
;
;        AST routine to execute on read completion.  The data which
;        was input to/from the users terminal is output on the
;        applications port.  Another read is then posted.
;
; INPUT PARAMETERS:
;        None
;
; OUTPUT PARAMETERS:
;        None
;
;--
;
READAST:
         .WORD    ^M<R2,R3,R4,R5>             ; PROCEDURE ENTRY MASK
         BLBS     IN_IOSB,10$                 ; CHECK IOSB FOR SUCCESS
         MOVZWL   IN_IOSB, R0                 ; PUT ERROR STATUS IN R0
         BRW      ERROR                       ; EXIT WITH ERROR
10$:     MOVZWL   IN_IOSB+2,R0                ; GET NUMBER OF CHARACTERS READ
         ADDL2    #OUT_MSGLEN,R0              ; ADD SIZE OF FIXED ACK
         $QIO_S   CHAN=LT_CHAN,-              ; OUTPUT MESSAGE TO LT DEVICE
                  FUNC=#IO$_WRITEVBLK,-
                  P1=OUT_MSG,-
                  P2=R0
         BSBW     ENABLE_READ                 ; QUEUE NEXT READ
         RET
;
         .SBTTL   ENABLE_CTRLCAST - ENABLE CONTROL C AST
;++
; FUNCTIONAL DESCRIPTION:
;
;        Routine to allow CONTROL C recognition on users terminal
;
; INPUT PARAMETERS:
;
;        None.
;
; OUTPUT PARAMETERS:
;
;        None
;
```

(Continued on next page)

Example 7-1 (Cont.) Application Program for Connecting to a Remote Device

```
;--
ENABLE_CTRLCAST:
         $QIOW_S CHAN=TT_CHAN,-
                  FUNC=#IO$_SETMODE!IO$M_CTRLCAST,-
                  P1=CTRLCAST,-                ; AST ROUTINE ADDRESS
                  P3=#3                        ; USER MODE
         BLBS     RO, 10$                      ; NO ERROR IF SET
         BRW      ERROR
10$:     RSB
;
         .SBTTL   CTRLCAST - AST Routine for Control C
;++
;
; FUNCTIONAL DESCRIPTION
;
;        AST routine to execute when ^C is received.  The connection
;        to the applications port is stopped and the program is terminated
;        with normal completion status.
;
; INPUT PARAMETERS:
;        None
;
; OUTPUT PARAMETERS:
;        None
;
;--
;
CTRLCAST:
         .WORD     ^M<>
         $QIO_S  CHAN=LT_CHAN,-          ; DISCONNECT SESSION TO LT DEVICE
                 FUNC=#IO$_TTY_PORT!IO$M_LT_DISCON
ERROR:   $EXIT_S RO                      ; EXIT
         RSB
;
         .SBTTL   SOL_CONNECT - Solicit Connection to Applications Port
;++
;
; FUNCTIONAL DESCRIPTION:
;
;        This routine issues the QIO to the LT driver to solicit
;        the connection to the applications port.
;
```

(Continued on next page)

Example 7-1 (Cont.) Application Program for Connecting to a Remote Device

```
; INPUT PARAMETERS:
;        None
;
; OUTPUT PARAMETERS:
;        None
;
;--
;
SOL_CONNECT:
         $QIO_S   CHAN=LT_CHAN,-
                  FUNC=#IO$_TTY_PORT!IO$M_LT_CONNECT,-
                  ASTADR=SOLAST,-
                  IOSB=SOL_IOSB
         BLBS     R0,10$
         BRW      ERROR
10$:     RSB
;
         .SBTTL   SOLAST - AST Routine for connection solicitation status
;++
;
; FUNCTIONAL DESCRIPTION
;
;        AST routine to execute when connection solicitation is
;        complete.  If status is success, print success message and
;        return.  If status is rejection, print reject message plus
;        reject reason and exit.  If status is otherwise, exit.
;
; INPUT PARAMETERS:
;        None
;
; OUTPUT PARAMETERS:
;        None
;
;--
;
```

(Continued on next page)

Example 7-1 (Cont.) Application Program for Connecting to a Remote Device

```
SOLAST:
          .WORD     ^M<>
          MOVZWL    SOL_IOSB,R0                  ; GET RETURN STATUS
          BLBC      R0,10$                       ; IF CLEAR, ERROR
          MOVL      R0,R1                        ; COPY STATUS CODE FOR INDEX
          JSB       WRITE_STATUS                 ; OUTPUT SUCCESS MESSAGE
          BSBW      ENABLE_CTRLYAST              ; ENABLE CONTROL Y AST'S
          RET
10$:      CMPW      R0,#SS$_ABORT                ; IS THIS A REJECTED CONNECTION?
          BNEQ      ERROR                        ; IF EQ, OUTPUT ERROR MESSAGE
          $QIOW_G             ERROR_QIO          ; OUTPUT ERROR MESSAGE FIRST
          MOVZWL    SOL_IOSB+2,R1                ; SET R1 FOR OFFSET INTO TABLE
          MOVZWL    TT_CHAN,ERROR_QIO+8          ; INSERT CHANNEL INTO QIO PACKET
          JSB       WRITE_STATUS                 ; OUTPUT ERROR REASON
          BRW       ERROR                        ; EXIT
WRITE_STATUS:
          MOVQ      MSG_TABLE[R1],-              ; PUT MESSAGE INTO QIO
                    WRITE_QIO+QIO$_P1
          MOVZWL    TT_CHAN,WRITE_QIO+8          ; INSERT CHANNEL INTO QIO PACKET
          $QIOW_G             WRITE_QIO
          RSB
          .SBTTL    EXIT_HANDLER:
;++
;
; FUNCTIONAL DESCRIPTION:
;
;         Exit handler routine to execute when image exits.  It will
;         cancel any outstanding I/O on these channels.
;
; INPUT PARAMETERS:
;         None
;
; OUTPUT PARAMETERS:
;         None
;
;--
;
EXIT_HANDLER:
          .WORD
          $CANCEL_S           CHAN=TT_CHAN       ; FLUSH ANY OUTPUT
          $CANCEL_S           CHAN=LT_CHAN
          RET
          .END START
```

Chapter 8

LATCP Command Descriptions

This chapter contains an alphabetical reference of all the LAT/VMS LATCP commands. Chapters 5 and 6 provide examples of some of the commands described in this chapter.

The LATCP utility allows you to control and obtain information from the LAT port driver (LTDRIVER). Using LATCP, you can do the following:

- Start and stop the LAT port driver
- Specify configuration characteristics for your service node and its services
- Modify and display configuration characteristics
- Show and zero system counters

Entering Commands

To enter LATCP commands, invoke LATCP at the DCL prompt, as follows:

```
$ RUN SYS$SYSTEM:LATCP
```

Once LATCP is invoked, the system issues the LCP> prompt. You can enter LATCP commands using the following format:

command-keyword [*parameter(s)*] [*/qualifier(s)*]

You can enter multiple qualifiers and their arguments on one command line; qualifiers are separated by slashes (/). Also, you can continue a command to a new line by typing a hyphen and then pressing RET . For example:

```
LCP> SET NODE DUKE /IDENT="SALES VAXCLUSTER" -  RET
_LCP> /MULTICAST_TIMER=50 /ENABLE=(1,2)
```

You can enter LATCP commands in either uppercase or lowercase characters (or a combination of both). Command lines can be up to 132 characters in length.

Commands can be abbreviated to their shortest unique length. For example, the commands SHOW CHARACTERISTICS and SHOW PORTS can be abbreviated as SH CH and SH PO respectively. However, to avoid ambiguity and possibly entering a command by accident, it is recommended that you abbreviate all commands to no fewer than three characters.

Error messages for the LATCP commands are shown in Appendix C.

Review the graphics conventions in the Preface. These conventions are used in the command descriptions in this chapter.

CREATE PORT

Creates an applications port on your service node.

Format

CREATE PORT *port-name*

Command Qualifiers	Defaults
/APPLICATION	/APPLICATION
/[NO]LOG	/LOG

Restrictions

- LTA0: is not a valid parameter.
- An error is returned if the specified port already exists.

Parameters

port-name

Specifies the name of the applications port to be created in the form LTA*n*:, where *n* is a unique number from 1 through 9999.

Description

The CREATE PORT command creates an applications port on your LAT service node. The applications port must be logically associated (mapped) with a remote device on a server. Use the SET PORT command to do this mapping.

Command Qualifiers

/APPLICATION

Specifies that the port being created on your service node functions as an applications port.

/[NO]LOG

Specifies whether characteristics of the ports on your service node are displayed when this command is executed.

Example

```
LCP> CREATE PORT LTA27: /APPLICATION /NOLOG
```

This command creates a port LTA27: to be used as an applications port on your service node. The /NOLOG qualifier in this command specifies that the characteristics for the applications ports on your service node are not displayed.

CREATE SERVICE

Creates a service on your service node.

Format

CREATE SERVICE *service-name*

Command Qualifiers	Defaults
/IDENTIFICATION="id-string"	SYS$ANNOUNCE
/[NO]LOG	/LOG
/[NO]STATIC_RATING=rating	/NOSTATIC_RATING

Restrictions

You cannot create more than one service with the default service name.

Parameters

service-name

Specifies a LAT service name of 1 to 16 ASCII characters. Eligible characters are described in Appendix A. If you do not specify a service name in the command line, the default service name is the translation of the SYS$NODE logical name.

Coordinate the service names throughout the network to avoid duplicating them unintentionally.

Description

This command creates a service offered by your service node. You can assign up to eight service names on your service node. You can later modify the service characteristics with the SET SERVICE command. The service is announced in the multicast messages sent by your service node.

Several service nodes can share one service name. A shared service name is especially useful in VAXclusters. It allows the cluster to be known by a cluster name and also by individual node names.

Command Qualifiers

/IDENTIFICATION="identification-string"

Specifies a description for the service. The string is advertised to servers in the multicast messages sent by your service node. The string can have up to 64 ASCII characters, which cannot begin with an ampersand (&). Nonprintable characters are translated as spaces. Enclose the string in quotation marks (""). The default for this announcement is the translation of the SYS$ANNOUNCE logical name.

/[NO]LOG

Specifies whether the characteristics for your service node are displayed with this command.

/[NO]STATIC_RATING=rating

Disables dynamic rating and specifies a static rating when you omit NO. Disables static rating and starts dynamic rating when you include NO.

Example

```
LCP> CREATE SERVICE SALES /STATIC_RATING=195 /NOLOG
```

The command in this example creates the service "SALES" on your service node and assigns a static rating of 195. The /NOLOG qualifier in this command specifies that the characteristics for your service node are not displayed.

DELETE PORT

Deletes an applications port from your service node.

Format

DELETE PORT *port-name*

Command Qualifiers	**Defaults**
None	None

Restrictions

The port was created previously with the LATCP CREATE PORT command.

Parameters

port-name

Specifies the name of the applications port you want to delete.

Description

The DELETE PORT command stops any active session on the applications port and then deletes this port from your service node.

Example

```
LCP> DELETE PORT LTA27:
```

The command in this example deletes the applications port LTA27:. The port was created previously with the CREATE PORT command.

DELETE SERVICE

Deletes a service from your service node.

Format

DELETE SERVICE *service-name*

Command Qualifiers	Defaults
/[NO]LOG	/LOG

Restrictions

None.

Parameters

service-name

Specifies the name of the service, created previously with CREATE SERVICE, to be deleted. To find service names, use the SHOW CHARACTERISTICS command. The default is the translation of the SYS$NODE logical name.

Description

The DELETE SERVICE command removes the service from your service node. The service is no longer available to server users and is no longer sent in the multicast messages sent by your service node.

Command Qualifiers

/[NO]LOG

Specifies whether the characteristics for your service node are displayed when this command is executed.

Example

```
LCP> DELETE SERVICE SALES
```

The command in this example removes the service SALES from your service node. The service is no longer available to server users.

EXIT

Stops execution of LATCP and returns you to the DCL command level.

Format

EXIT

Command Qualifiers	Defaults
None	None

Restrictions

None.

Parameters

None.

Description

The EXIT command enables you to exit from LATCP and returns you to DCL command level. Pressing CTRL/Z has the same effect as the EXIT command.

Examples

1
```
LCP> EXIT
```

2
```
LCP> CTRL/Z
```

Each of these commands ends the LATCP session and returns control to the DCL command level.

HELP

Lists and explains the LATCP commands.

Format

HELP *command-name*

Command Qualifiers	Defaults
None	None

Restrictions

None.

Parameters

command-name

The name of a LATCP command.

Description

HELP is an on-line reference for LATCP commands. If you do not specify a command name, HELP displays general information on the commands for which help is available. Supplying a command name obtains syntax information on that command. After you get an initial HELP display and then press RET, the HELP stops and your LATCP prompt appears.

Example

```
LCP> HELP SET PORT
```

In this example, the HELP SET PORT command produces a description of the SET PORT command and shows the command format.

SET COUNTERS

Resets your service node counters to zero.

Format

SET COUNTERS/ZERO

Command Qualifiers	Defaults
/ZERO	/ZERO

Restrictions

You cannot use this command to set device counters or server counters to zero.

Parameters

None.

Description

The SET COUNTERS/ZERO command allows you to test the performance of your service node over a period of time. Once the counters for your LAT service node are set to zero, you can observe information that accumulates over the specific period by using the LATCP SHOW COUNTERS/NODE command.

Command Qualifiers

/ZERO

Specifies to reset the service node counters to zero.

Example

```
LCP> SET COUNTERS/ZERO
```

The command in this example resets your service node counters to zero.

SET NODE

Allows you to specify LAT node characteristics.

Format

SET NODE *node-name*

Command Qualifiers	Defaults
/DISABLE=*group-list*	None
/ENABLE=*group-list*	/ENABLE=(0)
/IDENTIFICATION="id-string"	SYS$ANNOUNCE
/[NO]LOG	/LOG
/MULTICAST_TIMER=seconds	/MULTICAST_TIMER=60

Restrictions

None.

Parameters

node-name

Specifies the name you assign to your service node. The node name can be from 1 to 16 ASCII characters in length. Eligible characters are described in Appendix A.

The node name should be the same as the DECnet node name. The DECnet node name must be unique within the same logical Ethernet as well as within the entire DECnet network. On DECnet nodes, the LAT node name is given the DECnet node name, SYS$NODE, by default. If the service node is not running DECnet but will be in the future, it is recommended that you define SYS$NODE.

The default is the translation of the SYS$NODE logical name.

Description

The SET NODE command allows you to specify:

- Node name
- Node identification
- Groups

- Timing of configuration messages

See Chapter 5 for a discussion of these node characteristics. Any characteristics that you omit are not changed from their previous settings.

Because LATCP commands change characteristics dynamically, you can use SET NODE prior to activating the LAT port driver or at any time when the LAT port driver is active.

Command Qualifiers

/DISABLE=group-list

Removes previously enabled groups associated with your service node.

/ENABLE=group-list

Gives your service node access to the listed groups. There are 256 groups, numbered from 0 through 255. When you enter a group list, use commas (,) to separate individual groups.

Group 0 is enabled by default for all service nodes and servers. For additional information about group 0, see your server *Management Guide*.

NOTE: Not all servers support 256 groups.

/IDENTIFICATION="identification-string"

Specifies a description for your service node. The string is advertised to servers in the multicast messages sent by your service node. The string can have up to 64 ASCII characters and cannot begin with an ampersand (&). Nonprintable characters are translated as spaces. Enclose the string in quotation marks ("").

/[NO]LOG

Specifies whether your service node characteristics are displayed when this command is executed. /NOLOG prevents the display.

/MULTICAST_TIMER=seconds

Specifies the time, in seconds, between the multicast messages sent by your service node. The minimum value is 10 seconds; the maximum is 255 seconds. The default value is 60.

Examples

1

```
LCP> SET NODE DUKE /IDENT="SALES VAXCLUSTER" /NOLOG
```

The command in this example specifies that the announcement "SALES VAXCLUSTER" be included in the multicast messages sent from node DUKE. The /NOLOG qualifier in this command specifies that the characteristics of your service node are not displayed.

2

```
LCP> SET NODE DUKE /MULTICAST_TIMER=50 /ENABLE=(1,2)
```

The command in this example causes the node DUKE to send multicast messages every 50 seconds. This command also enables groups 1 and 2 for NODE DUKE.

3

```
LCP> SET NODE DUKE /DISABLE=2
```

The command in this example disables group 2 for node DUKE. Group 2 was enabled previously for the service node.

SET PORT

Logically associates (maps) an applications port with a remote port on a server.

Format

SET PORT *port-name*

Command Qualifiers	Defaults
/APPLICATION	/APPLICATION
/[NO]LOG	/LOG
/NODE=remote-node-name	None
/PORT=remote-port-name	None
/[NO]QUEUED	/QUEUED
/SERVICE=remote-service-name	None

Restrictions

You must get the remote node (server) name, remote port name, and remote service names from the server manager.

Parameters

port-name

Specifies the name of the applications port. The applications port name must be in the form LTA*n*:, where *n* is a unique number from 1 through 9999.

Description

The SET PORT command maps an applications port on your service node to a port on a server. The applications port must have been created previously with the CREATE PORT command.

You must specify the applications port name and the remote node (server) name, plus one or both of the following:

- Server port name

- Remote service name

NOTE: If you want to connect to a specific port for a service, specify the remote service name and the port name.

The service name on the server is associated with one or more specific ports on that server.

Command Qualifiers

/APPLICATION

Specifies that the port on your service node functions as an applications port.

/[NO]LOG

Specifies whether or not to display the characteristics of the ports on your service node when this command is executed.

/NODE=remote-node-name

Specifies the name of the remote node (server) to be logically associated with the applications port on your service node.

/PORT=remote-port-name

Specifies the name of the remote port on a server associated (mapped) with the applications port.

/[NO]QUEUED

Specifies the type of access being requested for the remote port.

There are two types of access requests to a port: queued and nonqueued. The LAT server manager defines the type of access allowed.

If you do not want your remote requests to be queued on the server, specify /NOQUEUED. A queued or nonqueued request is accepted by the server if the remote port is free. If the remote port is busy and queuing is enabled on the server, then a remote request is queued. Do not specify /NOQUEUE if the port is connected to a remote printer that is accessed by LATSYM.

/SERVICE=remote-service-name

Specifies the name of the remote service offered at the server port associated with the applications port.

Examples

1
```
LCP> SET PORT LTA322: /NODE=LAT1 /PORT=PORT_5
```
This example specifies that the applications port LTA322: is associated with the port named PORT_5 on the server named LAT1.

2
```
LCP> SET PORT LTA322: /NODE=LAT1 /SERVICE=PRINTER /QUEUED
```
This command associates the applications port LTA322: with the service PRINTER on server LAT1. The service PRINTER can be associated with one or more ports on LAT1. The /QUEUED qualifier specifies that the server, offering the service PRINTER, queue the remote connection request.

SET SERVICE

Allows you to dynamically change service characteristics.

Format

SET SERVICE *service-name*

Command Qualifiers	Defaults
/IDENTIFICATION="id-string"	SYS$ANNOUNCE
/[NO]LOG	/LOG
/[NO]STATIC_RATING=static-rating	/NOSTATIC_RATING

Restrictions

- You can only specify the service name of a service you created previously with the CREATE SERVICE command.
- Your service node characteristics in the LTLOAD.COM file become operational upon system startup. The characteristics defined with the SET SERVICE command do not stay in effect after a restart of your system.

Parameters

service-name

Specifies the name of the service whose characteristics that you change with this command.

The default service name is the translation of the SYS$NODE logical name.

Description

The SET SERVICE command dynamically changes the characteristics of a service that you created previously with the CREATE SERVICE command.

Command Qualifiers

/IDENTIFICATION="identification-string"

Specifies a new description of the service. The string is announced to server users in the multicast messages sent by your service node. The string can have up to 64 ASCII characters and cannot begin with an ampersand (&). Nonprintable characters are translated as spaces. Enclose the string in quotation marks ("").

/[NO]LOG

Specifies whether or not to display the qualifier values used in this command when this command is executed. /NOLOG prevents the display.

/[NO]STATIC_RATING=rating

Disables dynamic rating and specifies a static rating when you omit NO. Disables static rating and starts dynamic rating when you include NO.

Example

```
LCP> SET SERVICE SALES /IDENT="A MEMBER OF THE SALES CLUSTER"
```

The command in this example specifies a new announcement "A MEMBER OF THE SALES CLUSTER" for the service SALES. This string is announced with the service SALES in the multicast messages sent by your service node.

SHOW CHARACTERISTICS

Displays the characteristics for your service node.

Format

SHOW CHARACTERISTICS

Command Qualifiers	Defaults
None	None

Restrictions

None.

Parameters

None.

Description

This command displays the node and service parameters for your LAT service node.

- Qualifier settings for the SET NODE command
- LAT protocol status
- LAT protocol version
- Service characteristics defined for your service node

Example

```
LCP> SHOW CHARACTERISTICS

LCP Characteristics

Node name = \DUKE\
Node name = \DUKE\ Node Identification = \A MEMBER OF THE SALES CLUSTER\
Groups = (0,64,127)
Multicast timer = 60 seconds
LAT Version = 5.1        LAT Protocol is active

Service Names and Ids:

Service name : \SALES\              rating :  auto
            id : \SALES SERVICE\
```

SHOW COUNTERS

Displays counter information for your LAT service node.

Format

SHOW COUNTERS

Command Qualifiers	Defaults
/DEVICE	/NODE
/NODE	/NODE
/SERVERS	/NODE

Restrictions

None.

Parameters

None.

Description

This command displays counter information tabulated by the LAT port driver.

Command Qualifiers

The default for the qualifiers is the /NODE qualifier.

/DEVICE

Specifies to display the Ethernet device counters. This information is the sum of all Ethernet usage on your node, including LAT and DECnet.

For additional information on device counters, see the Network Control Program (NCP) in the VAX/VMS documentation.

/NODE

Specifies to display LAT counters for your service node (does not include DECnet counters). Table 8–1 gives descriptions of the LAT counters.

Table 8-1 Descriptions of LAT Node Counters

Counter	Meaning
Receive frames	The number of LAT messages successfully received by the node.
Receive errors	The number of received messages with detected problems. The Ethernet controller flagged these messages.
Receive duplicates	The number of duplicated received messages; this can indicate a system slowdown.
Transmit frames	The number of LAT messages successfully transmitted by the node.
Transmit errors	The number of transmitted messages with detected problems. The Ethernet controller flagged these messages.
Last transmit failure code	A hexadecimal number that indicates the reason for the last transmit failure. If a failure code exists, the failure reasons are shown at the end of the counters display.
Retransmissions	The number of LAT messages that the node retransmitted because they were not acknowledged by the server(s).
Circuit timeouts	The number of times a circuit to a server timed out, indicating that a server failed to send a valid message in the required time span.
Protocol errors	The number of LAT messages with an illegal format received by the node. The actual error identification is recorded in the protocol error bit mask.
Protocol bit mask	A hexadecimal number that indicates circuit message errors or slot errors. If you convert this number to binary format, you can find protocol error bit mask definitions listed in Table 8–2
Resource errors	The number of times the service node attempted to create a circuit with a new server but failed because of insufficient resources.

Table 8–1 (Cont.) Descriptions of LAT Node Counters

Counter	Meaning
No transmit buffer	The number of times no buffer was available for transmission.
Unit timeouts	A repeat of a deallocation of resources. This occurs when an attempt is made to deallocate a Unit Control Block (UCB) that was already deallocated. This happens when there is an attempt to stop a LAT circuit on the service node.
Solicitation failures	The number of times a request for a connection to a remote device failed.
Discarded output bytes	The number of data bytes which were discarded because of an overflow of an internal buffer before the data could be output to an LT device.

Table 8–2 lists the protocol error bit mask definitions.

Table 8–2 Protocol Error Bit Mask Definitions

Bit	Meaning
0	Invalid start message received
1	Zero node index received
2	Node circuit index out of range
3	Node circuit sequence invalid
4	Node circuit index no longer valid
5	Circuit was forced to halt
6	Invalid server slot index
7	Invalid node slot index
8	Invalid credit field or too many credits used
9	Repeat create of slot by server
10	Invalid sequence number received in start message
11	Repeat disconnect of slot by server

/SERVERS

Specifies to display LAT counters for all servers known to your service node.

Note that some servers may be listed twice in the SHOW SERVER display. LATCP keeps up to two sets of counters for servers which had an abnormal circuit termination. This allows the server information to be retained and examined by a service specialist.

Examples

The following is an example of a display generated by the SHOW COUNTERS /NODE command:

1

```
LCP>SHOW COUNTERS /NODE

LCP Node Counters

     127597   Receive frames
          0   Receive errors
          3   Receive duplicates
     161885   Transmit frames
          0   Transmit errors
   00000000   Last transmit failure code
         28   Retransmissions
          6   Circuit timeouts
          0   Protocol errors
   00000000   Protocol bit mask
          0   Resource errors
          0   No transmit buffer
          0   Unit timeouts
          0   Solicitation failures
          0   Discarded output bytes
```

The following is an example of a display generated by the SHOW COUNTERS /SERVER command:

2

```
LCP>SHOW COUNTERS /SERVERS

LCP Server Counters for LAT1

       7882   Receive frames
       8743   Transmit frames
          0   Retransmissions
          0   Out of sequence frames
          0   Invalid messages
          0   Invalid slots
```

SHOW PORTS

Displays the characteristics for the ports on your service node.

Format

SHOW PORTS *port-name*

Command Qualifiers	Defaults
/APPLICATION	/APPLICATION
/INTERACTIVE	None

Restrictions

Do not use the /APPLICATION or /INTERACTIVE qualifiers with a specific port name.

Parameters

port-name

Specifies the name of the port for which information is displayed. When you issue the SHOW PORTS command without entering a port name, characteristics for all of the LTA*n*: ports on the service node are displayed.

Description

If the port is an applications port, the display lists the remote node name and remote port and/or remote service name that you specified in the SET PORT command. If the port is an interactive port, it is currently being used by a server user. For all ports with current connections, the server sends the node name and port name to your service node. These are listed in the display.

Command Qualifiers

/APPLICATION

Generates a display for all applications ports.

/INTERACTIVE

Generates a display for all interactive ports.

Example

```
LCP> SHOW PORTS

Local Port Name = LTA62:    <interactive>
   Actual Remote Node Name = LAT10
   Actual Remote Port Name = PORT_7

Local Port Name = LTA322:    <application>
   Specified Remote Node Name = LAT1
   Specified Remote Port Name = LQP02
   Actual Remote Node Name = LAT1
   Actual Remote Port Name = LQP02

Local Port Name = LTA321:    <application>
   Specified Remote Node Name = LAT1
   Specified Remote Service Name = PRINTER
```

The first port displayed in this example is the interactive port LTA62: connected to Port_7 on the LAT10 server.

The second port displayed in this example is the LTA322: applications port. Note that in this display the presence of the actual values indicates an established connection.

The third port displayed in this example is the LTA321: applications port mapped to the PRINTER service on the LAT1 server.

SHOW SERVERS

Displays the characteristics for the servers known to your service node.

Format

SHOW SERVERS

Restrictions

None.

Parameters

None.

Description

Displays the following information about servers known to your service node:

- Ethernet address
- Server status
- Number of active users

Example

```
LCP>SHOW SERVERS

LCP Server Characteristics for LAT1

Ethernet address = AA-00-03-01-0D-BC
Server is active                Active users = 1
```

START NODE

Starts the LAT port driver and sets service node characteristics.

Format

START NODE *node-name*

Command Qualifiers	Defaults
/DISABLE=group-list	None
/ENABLE=group-list	/ENABLE=(0)
/IDENTIFICATION="id-string"	SYS$ANNOUNCE
/[NO]LOG	/LOG
/MULTICAST_TIMER=seconds	/MULTICAST_TIMER=60

Restrictions

None.

Parameters

node-name

Specifies the name you choose for your service node. The node name can be from 1 to 16 ASCII characters long. Eligible characters are described in Appendix A. The default is the translation of the SYS$NODE logical name.

Description

The START NODE command activates the LAT port driver. Before issuing this command, however, you must invoke the System Generation Utility (SYSGEN) to load the LAT port driver and the first LT: template unit control block (UCB) as follows:

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN> CONNECT LTAO: /NOADAPTER
```

Note that use of the SYSGEN CONNECT command requires the CMKRNL privilege.

After the LAT port driver is activated, you can dynamically modify your node characteristics with SET NODE. Also, if DECnet is to be started on your node, start DECnet before starting the LAT port driver.

When you start LAT on your node, LATCP attempts to find an Ethernet controller device on the node. If the controller on your node does not match a controller that LAT recognizes, LAT attempts to translate the logical name, LAT$DEVICE, as the controller name. Use the following command to define an Ethernet controller device before starting LAT:

```
$ DEFINE/SYSTEM/EXEC LAT$DEVICE dev-name:
```

The STOP NODE command clears the current node characteristics. Invoke the LTLOAD.COM file to start LAT, or set the node characteristics manually before starting LAT.

Command Qualifiers

/DISABLE=group-list

Removes previously enabled groups associated with your service node.

/ENABLE=group-list

Gives your service node access to the listed groups. There are 256 groups, numbered from 0 through 255. When you enter a group list, use commas (,) to separate individual groups. The default is that no groups are enabled.

NOTE: Not all servers support 256 groups.

/IDENTIFICATION="identification-string"

Specifies a description for your service node. The string is advertised to servers in the multicast messages sent by your service node. The string can have up to 64 ASCII characters and cannot begin with an ampersand (&). Nonprintable characters are translated as spaces. Enclose the string in quotation marks ("").

/[NO]LOG

Specifies whether to display your service node characteristics when this command is executed. /NOLOG prevents the display.

/MULTICAST_TIMER=seconds

Specifies the time, in seconds, between the multicast messages sent by your service node. The minimum value is 10 seconds; the maximum is 255 seconds. The default value is 60.

Example

```
LCP> START NODE DUKE /ID
```

The command in this example starts node DUKE and assigns default values for the command qualifiers.

STOP NODE

Shuts down the LAT port driver and terminates all of the sessions on LTA*n*: ports.

Format

STOP NODE

Command Qualifiers	Defaults
None	None

Restrictions

None.

Parameters

None.

Description

Use the following recommended steps to stop the LAT port driver with STOP NODE:

1. Since active connections are disconnected without warning, use the DCL REPLY command to issue warnings to LAT users to log off the node's services, before you issue the STOP command.

   If the node is down for a long period, tell the user when the node will be back up.

2. In response to the LCP> prompt, issue the LATCP command SET NODE *node-name* /IDENTIFICATION="*identification-string*". In the identification string, announce the reason for the shutdown to the LAT server(s).

3. Issue the LATCP STOP NODE command.

If you want to stop the LAT port driver and want your service node to have the same characteristics when you restart the driver, do not use the START NODE command. Instead, invoke the SYS$MANAGER:LTLOAD.COM command procedure. Alternatively, after stopping the LAT port driver using STOP NODE, you can set up your node characteristics manually with the SET NODE and SET SERVICE commands.

Example

```
$ REPLY /ALL "LAT SERVICE SHUTTING DOWN IN 5 MINUTES. PLEASE LOG OFF"
```

This command notifies users not connected through a server that the node is temporarily shutting down.

```
LCP> SET NODE DUKE /IDENT="SHUT DOWN FOR SERVICE--BACK UP 2:00 PM."
```

This command notifies users connected through a server that the services offered by node DUKE are temporarily unavailable.

```
LCP> STOP NODE
```

This command shuts down the LAT port driver and disconnects all sessions to your node.

Appendix A

ASCII Characters for Node and Service Names

These are the ASCII characters used to specify node and service names for your service node:

- “$”—dollar sign, ASCII code 36
- “-”—hyphen, ASCII code 45
- “.” or “–”— period or dash, ASCII code 46
- “0 through 9”—numerals, ASCII codes 48–57
- “A through Z”—uppercase letters, ASCII codes 65–90
- “_”—underscore, ASCII code 95
- “a through z”—lowercase letters, ASCII codes 97–122
- Part of the international character set—ASCII codes 192–253

Care must be taken to coordinate the service names throughout the network to avoid duplicating service names unintentionally.

Names can be from 1 through 16 characters in length. Note that spaces are invalid.

Appendix B

Qualifiers for DCL Printer Setup Commands

This appendix describes the terminal and queue characteristics required for setting up printers on VMS systems. The appendix contains two tables:

- Table B–1 shows the formats of standardized SET TERMINAL and INITIALIZE/QUEUE commands. These formats illustrate qualifiers that are required for all printers.
- Table B–2 lists particular printers and indicates any associated values that usually should be specified for a given command qualifier.

In addition, the appendix includes comments about a few of these qualifiers. For further details about the qualifiers for SET TERMINAL and INITIALIZE/QUEUE see Chapter 6 in this guide and the *VAX/VMS DCL Dictionary*.

Table B–2 shows the standardized formats of the SET TERMINAL and INITIALIZE/QUEUE commands. The qualifiers in these two commands specify terminal or queue characteristics that you must set for all printers on your VMS system. Variables are in lowercase italics.

Table B–1 Command Qualifiers Required for Remote Printers

Command	Qualifier
SET TERMINAL LTA*nn*:	/PERMANENT /NOBROADCAST /DEVICE=*type* /WIDTH=*value* /PAGE=*value*
INITIALIZE/QUEUE	/START /PROCESSOR=LATSYM /FORM=*form* /RETAIN=ERROR /DEFAULT=*values* /RECORD_BLOCKING /ON=LTA*nn*:*queue_name*

The following list includes comments on some of the qualifiers shown in Table B-2:

SET TERMINAL Comments:

/DEVICE This qualifier specifies a page width and page length that is appropriate for the device specified in a SET TERMINAL command. However, for some devices you may want to select alternative values. For example, the qualifier /DEVICE=LA210 will set the /WIDTH value to 132 and the /PAGE value to 66. When you use narrow paper in the LA210 device, you can override the default width value with a /WIDTH=80 qualifier, and the printer will be correct for the paper used.

INITIALIZE/QUEUE Comments:

/DEFAULT This qualifier, unless otherwise specified in the list specifies any valid defaults the user wishes for the printer.

/FORM This qualifier usually specifies a user-defined form having the appropriate /PAGE and /WIDTH values to match the printer and terminal characteristics.

Different types of printers vary in the values necessary for terminal and queue characteristics. Table B–2 shows values that must be specified for qualifiers that define nondefault characteristics. Entries listed as "User Preference" means that you can use any valid /DEVICE, /WIDTH, and /PAGE values with the SET TERMINAL command or any valid /DEFAULT and /FORM values with the INITIALIZE/QUEUE command.

Table B-2 Additional Qualifiers Required for Particular Devices

Device	SET TERMINAL Command	INITIALIZE/QUEUE Command
LA12, LA34 LA36, LA38 LA50, LA100 LQP02, LQP03 LN01S, LN03S	User Preference	User Preference
LA120, LA210	/NOTAB	User Preference
LXY12-DA LXY22-DA	/FORM /WIDTH=134 /PAGE=66 /NOWRAP /NOEIGHTBIT	User Preference (NOTE: Do NOT spool these devices.)
LCP01	/INTERACTIVE /FULLDUP /TAB /FORM /SCOPE /LOWERCASE /EIGHTBIT /TTSYNC /NOMODEM /NOECHO /NOWRAP /NOESCAPE	/DEFAULT= (NOFLAG,NOFEED, NOTRAILER,NOBURST) /SCHEDULE= NOSIZE
LG01,LG02	/FORM /WIDTH=134 /PAGE=66 /NOWRAP	/DEFAULT=NOFEED
LVP16 Plotter	/WIDTH=132 /PAGE=0	/DEFAULT= (NOFLAG,NOFEED, NOTRAILER,NOBURST)
DECtalk DTC01, DTC03	/WIDTH=132 /PAGE=0	/DEFAULT= (NOFLAG,NOFEED, NOTRAILER,NOBURST)

Appendix C

LATCP Error Messages

This appendix lists the LATCP error messages and gives an explanation of each one.

```
%LATCP-E-CANT BIND, Can't initialize device.
```

Explanation: Issued when the LAT device cannot be initialized.

```
%LATCP-E-INTERNAL, LATCP internal error
```

Explanation: Issued when LATCP or LAT port driver has insufficient buffer space or another problem.

```
%LATCP-E-IVCMD, Invalid command
```

Explanation: Issued when LATCP reports a general command syntax error.

```
%LATCP-E-IVDEV, Invalid node or device name
```

Explanation: Issued when an invalid LT device or node name is specified on CREATE or SHOW commands, or when an invalid or nonexistent LT device name is specified on a SET command.

```
%LATCP-E-IVQUAL, Value for qualifier "XXX" is invalid as "VVV"
```

Explanation: Issued when the value VVV for the parameter XXX of a SET command is invalid or out of range.

```
%LATCP-E-LOCKED, Data base locked. Try again later.
```

Explanation: Issued when LATCP cannot run because another user is modifying the LATCP database.

```
%LATCP-E-MAXSERV, Maximum number of services exceeded
```

Explanation: Issued on a CREATE SERVICE command when eight other services already exist.

```
%LATCP-E-NONODE, Node name has not been initialized
```

Explanation: Issued when an attempt was made to start the LAT port driver without a proper LAT node name set up.

```
%LATCP-E-NOPORTS, No such port(s)
```

Explanation: Issued when a nonexistent port is specified on a SHOW command.

```
%LATCP-E-NOSUCHSERV, Service name does not exist
```

Explanation: Issued when you attempt to use the SET command to modify a service not previously set up with a CREATE command.

```
%LATCP-E-NOTFROMLAT, Unable to shut down LAT from a LAT terminal
```

Explanation: Issued when you enter a LATCP STOP NODE command from a LAT terminal.

```
%LATCP-E-NOTINITED, LAT terminal port driver controller init not
called
```

Explanation: Issued when the LAT port driver cannot start because you did not properly initialize it.

```
%LATCP-E-NOTLOADED, LAT terminal port driver (LTDRIVER) is not loaded
```

Explanation: Issued when you attempt to start the LAT port driver before loading the driver using the SYSGEN CONNECT command.

```
%LATCP-E-NOTSTARTED, LAT terminal port driver not started
```

Explanation: Issued whenever an attempt to start the LAT port driver is unsuccessful.

```
%LATCP-E-NOTSTOPPED, LAT terminal port driver not stopped
```

Explanation: Issued whenever a STOP NODE command fails to stop the LAT protocol.

```
%LATCP-I-NOSERVERS, No known servers
```

Explanation: Issued when you issue a SHOW COUNTERS/SERVER or SHOW SERVERS command, but there are not any servers in the LATCP database.

```
%LATCP-E-SERVEXISTS, Service name already exists
```

Explanation: Issued when you attempt to CREATE a service already created for this node.

```
%LATCP-I-STARTED, LAT terminal port driver started
```

Explanation: Issued when the LAT port driver is started successfully.

```
%LATCP-I-STOPPED, LAT terminal port driver stopped
```

Explanation: Issued when the LAT port driver is stopped successfully.

Index

MicroVMS Release Notes,
Version 4.6
AA–KN07A–TN

READER'S COMMENTS

Note: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well organized? Please make suggestions for improvement.

____

____

____

____

____

____

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

____

____

____

____

____

____

Please indicate the type of user/reader that you most nearly represent:

- □ Assembly language programmer
- □ Higher-level language programmer
- □ Occasional programmer (experienced)
- □ User with little programming experience
- □ Student programmer
- □ Other (please specify) ______________________________

Name ______________________________ Date ______________

Organization ______________________________

Street ______________________________

City ______________________ State ____________ Zip Code or Country ________

---- Do Not Tear-Fold Here and Tape ----

digital

No Postage
Necessary
if Mailed
in the
United States

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

POSTAGE WILL BE PAID BY ADDRESSEE

DIGITAL EQUIPMENT CORPORATION
The Manager, Office Program
ZK02-1/N20
110 SPIT BROOK ROAD
NASHUA, NH 03062 - 9990

---- Do Not Tear-Fold Here ----

MicroVMS Release Notes,
Version 4.6
AA–KN07A–TN

READER'S COMMENTS

Note: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well organized? Please make suggestions for improvement.

____

____

____

____

____

____

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

____

____

____

____

____

____

Please indicate the type of user/reader that you most nearly represent:

- □ Assembly language programmer
- □ Higher-level language programmer
- □ Occasional programmer (experienced)
- □ User with little programming experience
- □ Student programmer
- □ Other (please specify) ____________________

Name ____________________ Date ____________

Organization ________________________________

Street ____________________________________

City __________________ State ________ Zip Code
or Country ______

---- Do Not Tear-Fold Here and Tape ----

digital

No Postage
Necessary
if Mailed
in the
United States

DIGITAL EQUIPMENT CORPORATION
The Manager, Office Program
ZK02-1/N20
110 SPIT BROOK ROAD
NASHUA, NH 03062 - 9990

---- Do Not Tear-Fold Here ----