Professional™

300 series

PRO/DECnet Tool Kit
Release Notes

Order No. AV-AV71A-TK

Developer's Tool Kit

digital

software

PRO/DECnet Tool Kit
Release Notes

Order No. AV-AV71A-TK

March 1984

This document provides information specific to this release of the PRO/DECnet Tool Kit.

SUPERSESSION/UPDATE INFORMATION:	This is a new manual.
OPERATING SYSTEM AND VERSION:	RSX-11M V4.1 RSX-11M-PLUS V2.1 VAX/VMS V3.4 P/OS V2.0
SOFTWARE VERSION:	PRO/DECnet Tool Kit V1.0

DIGITAL EQUIPMENT CORPORATION
Maynard, Massachusetts 01754

Networks Communications and Publications typeset this manual using Digital's TMS-11 Text Management System.

CONTENTS

PREFACE

The following release notes include updated information for the PRO/DECnet Tool Kit Version 1.0. You should use these notes in conjunction with the material presented in the PRO/DECnet Tool Kit documentation set.

CHAPTER 1
SOFTWARE COMPONENTS — ADDITIONAL INFORMATION

1.1 DAPRES TASK BUILD COMMAND FILE

The following example shows a task build command file and overlay descriptor file which link a task to DAPRES. In this way, the task can perform RMS file access to files located on other nodes in the network. The remote nodes must support remote RMS requests. The task can still handle RMS file access to local files.

This file access does not require any changes to the task code. If PRO/DECnet has not been installed, a task built to use DAPRES will still run, but it will only access files on the local system.

```
;
; DAPTSTBLD.CMD - PROTKB task build command file.
;
; Build a task image for a task called DAPTST. This
; task will be able to access files on remote RMS
; systems.
;
DAPTST.TSK,DAPTST.MAP/-SP=DAPTSTBLD.ODL/MP
CLSTR=DAPRES,RMSRES:RO
//
```

```
;
; DAPTSTBLD.ODL - PROTKB overlay descriptor file.
;
; Describe the overall structure of the task. A
; DAPRES co-tree must be included, using RMSALL.
; The RMSALL symbol references definitions in
; the [1,5]DAPRLX.ODL file.
;
        .ROOT    TST,RMSALL
;
; Root segment for the DAPTST task
;
TST:      .FCTR   DAPTST.OBJ
;
; Link to the system DAPRES overlay descriptor file.
;
@LB:[1,5]DAPRLX.ODL

          .END
```

RMS file access is documented in Chapter 5 of the *PRO/DECnet Tool Kit Programmer's Reference Manual.*

1.2 RMS ERROR CODES FOR PRO/DECnet

The following RMS error codes are not shown in Appendix A of the *PRO/RMS-11 MACRO Programmer's Guide* (Order No. AA-P099A-TK). They are possible errors for RMS PRO/DECnet network operations.

ER$FAL Operation not supported by remote node Octal: 176550
Decimal: -664

The remote node does not support the requested RMS-11 operation. The STV field of the FAB or RAB contains (in its high 4 bits) an error reason code. These codes include:

0 — Incompatible operating systems; the low 12 bits of the STV field contain the type of the remote operating system

1 — Incompatible file systems; the low 12 bits of the STV field contain the type of the remote file system

2 — DAP version number smaller than 5; the low 12 bits of the STV field contain the DAP version number

3 — DAP modification number smaller than 6; the low 12 bits of the STV field contain the DAP modification number

4 — Unsupported file organization

5 — Unsupported record access

6 — Operation not supported by FAL; the low 12 bits of the STV field contain the operation code

7 — Remote I/O buffer too small; the low 12 bits contain the size of the remote I/O buffer

ER$NAE Unmappable network access error Octal: 175630 Decimal: -1128

If this error occurs, write down the name of the operation and file organization and any related information you can obtain such as contents of registers and a task builder map of the task. Then contact your Professional support person.

ER$NET Network link lost Octal: 175570 Decimal: -1160

The STV field of the FAB or RAB contains the network error code.

ER$SUP Operation not supported over network Octal: 174610 Decimal: -1656

The requested operation is not supported over the network.

ER$UIN Field value rejected by FAL Octal: 174510 Decimal: -1720

The FAL (File Access Listener) rejected the value in a control block field; the STV field of the FAB or RAB contains a code showing which field. See your DECnet documentation for the meanings of these codes.

1.3 OPTIMIZING NETWORK THROUGHPUT IN DECnet APPLICATION TASKS

When designing a PRO/DECnet task, it is important to consider the network data throughput. The rate at which tasks can exchange data depends on several factors including:

- The type of transfer (task-to-task or remote file access)
- The buffer alignment (even or odd byte buf alignment (even or odd byte buffer start)
- The number of transmit buffers at the transmit end
- The number of receive buffers at the receive end
- The number of network I/O requests per data packet
- The size of the data blocks being transferred
- Other system activity

This section discusses issues which apply to the task-to-task programming interface. Some of these issues also apply to the remote file access (RMS) interface. However, the ways of optimizing user designed protocols are not covered here.

NOTE

A significant increase in throughput occurs when the transmit buffer starts on an even byte boundary rather than an odd buffer byte boundary. DECnet must move the data from the user task's buffer into network I/O buffers (which always start on an even byte boundary). If the data moves from an even byte user task buffer to an even byte network buffer, the data is moved a word at a time. If the user task buffer starts on an odd byte boundary, the data is moved a byte at a time.

Similar results can occur for receive buffers where the data is moved from the DECnet buffers into the user task buffers. Therefore, you should always start the transmit and receive buffers on an even byte boundary.

The simplest networking program uses synchronous I/O and single buffering. This design is the least efficient method of transferring data. Time and data throughput are wasted with this type of program. An I/O call must be completed before a task can issue another call. As a result, the task is not keeping the logical link filled with transferring data.

You can increase data throughput by converting the synchronous I/O calls to asynchronous I/O calls. Such conversion allows the task to assemble the next packet for transmission while the previous packet is being sent.

You can realize a further gain by using multiple buffers for transmission and reception. By moving from a single to a double buffering level, your task's throughput is increased the most. A somewhat smaller gain is achieved by moving from a double to a triple buffering level. Higher buffering levels generally do not increase your task's throughput because the processor is completely utilized for moving triple buffers. However, paths which suffer from very long delays, such as satellite links, may benefit from higher buffering levels.

These changes allow for as much data in the logical link as possible. For multiple buffer levels, DECnet actually reduces the amount of processor used per data transfer due to optimizations which occur in the DECnet protocols.

Another important throughput increase is related to the task's buffer size and the number of I/O requests per packet of data. When a task uses small buffers and issues a large number of I/O requests per quantity of data, the throughput rate decreases because the processor is busy processing many small data packets. Additional processor time is also lost due to the overhead of I/O packet processing and context switching caused by numerous I/O requests. To optimize throughput, user task buffers should be as large as possible. The amount of available processor time increases because less time is spent processing a large buffer rather than a series of small buffers.

User task buffers can be a maximum of 8128 bytes in length, even though the segment and large buffer sizes in DECnet are smaller. This is due to a concept called segmentation. When DECnet receives for transmission a user task data packet which is larger than the segment size, it breaks the packet into pieces equal to or less than the segment size and transmits them. When the remote DECnet system receives the segments, it places them into the target task's receive buffer, and recreates the original large buffer.

NOTE

> Throughput is significantly increased when DECnet receives a large buffer to be segmented as opposed to several smaller buffers which cause additional I/O packet processing. Large buffers also free up processor time because DECnet uses less DECnet protocol overhead to move the data to the other node.

Large transmit and receive buffers result in significant gains on both ends of the logical link. On the transmit end, less transmit request I/O processing occurs per unit of data. The process of receiving a packet generally involves issuing another receive I/O request to keep the buffering level at the same level. This essentially causes two context switches per receive, one to complete the current receive, and another to issue the new receive. By using large data buffers for reception, a significantly smaller amount of I/O packet processing and context switching occur per data packet.

To summarize, the following techniques can increase a DECnet task's performance:

Design Concept	**Increase Throughput?**	**Increase Processor?**
Starting buffer on an even byte boundary	Yes	Yes
Asynchronous I/O	Yes	No
Multiple buffering levels	Yes	Yes
Decreasing number of I/Os per data	Yes	Yes
Increasing user buffer size	Yes	Yes

Buffering considerations are discussed in Chapters 2 and 3 of the *PRO/DECnet Tool Kit Programmer's Reference Manual.*

1.4 PRO/TOOL KIT AND DCL

To use the PRO/Tool Kit DCL, you must install the PRO/Tool Kit on your Professional. You should also install the diskette labeled NETTK. This diskette contains the network components required for DCL remote file access and the object libraries required for PRO/DECnet application development.

Do not install the diskette labeled PRODCL2. This diskette should only be used if you have installed the end-user PRO/DCL and not the PRO/Tool Kit. PRODCL2 does not include the object libraries specific to PRO/DECnet.

The procedure for installing NETTK is documented in Chapter 1 of the *PRO/DECnet Tool Kit Installation Guide.*

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 WLL BE PAID BY ADDRESSEE

Professional 300 Series Publications
DIGITAL EQUIPMENT CORPORATION
146 MAIN STREET
MAYNARD, MASSACHUSETTS 01754

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