DEC-11-SYDC-D

PDP-11

GETTING DOS ON THE AIR

AUGUST 1971

THIS DOCUMENT IS FOR INFORMATION PURPOSES, AND IS SUBJECT TO CHANGE WITHOUT NOTICE.

IT DESCRIBES HOW TO LOAD THE DOS SOFTWARE ONTO DISK FROM PAPER TAPE AND DECTAPE. IT ALSO CONTAINS USEFUL PROGRAMMING INFORMATION AND CORRECTIONS TO CURRENT DOS DOCUMENTS.

YOUR ATTENTION IS INVITED TO THE SPECIAL NOTE ON THE NEXT PAGE.

pdp11
PROGRAM
LIBRARY

SPECIAL NOTE

SOFTWARE PERFORMANCE REPORT

If you have any problem or discover any inadequacy in your DOS software or its documentation, please report it using the Software Performance Report forms enclosed in your software kit.

Give the Software Performance Report to your DEC Software Specialist. In most cases he will be able to provide an immediate answer to your problem, as he is kept informed of new information as soon as it becomes available. If yours is an original problem, the Software Specialist will ensure that all necessary details, examples, and supporting material are included in the Report, and then he will forward the complete report to DEC's Software Information Service Group in Maynard, Massachusetts for a thorough investigation of your problem. As soon as the investigating programmer has the answer to your problem, it will be sent to you via the Software Specialist.

This procedure is intended to provide fast replies to your Software Performance Reports either by an immediate answer from your Software Specialist or as the result of concentrating our software maintenance effort on well-documented original problems.

Your inputs are most appreciated in our continuing effort to improve our software, and with your help our commitment to good software support will remain apparent.

If you have any questions on this procedure, please contact your Software Specialist.

READER'S COMMENTS CARD

Your attention is invited to the last page of this document. The "Reader's Comments" page, when filled in and mailed, is beneficial to both you and DEC; all comments received are acknowledged and are considered when documenting subsequent manuals.

PREFACE

Chapter 1 explains, using step-by-step instructions, how to build the PDP-11 Disk Operating System (DOS). We suggest that you read this chapter before building your first system. This reading should clarify the purpose for performing certain steps in the building process as well as familiarize you with the entire process.

Chapter 2 contains programming notes and techniques which reveal short-cuts and other helpful information for added programming efficiency.

Chapter 3 is a detailed description of the DOS Monitor DECtape Setup and Update Program (MODS). MODS can be used to copy the SYSLOD.SYS and MONLIB.SYS programs from the delivered DECtape onto a back-up DECtape.

Chapter 4 can be used to update existing system manuals to describe the current versions of the system programs.

This document assumes familiarity with the PDP-11 system, DOS Monitor, Link-11 Linker, and PIP File Utility Package.

DOCUMENTATION CONVENTIONS USED HEREIN

1. All command strings are terminated by typing the RETURN key; this is standard, and not shown in most examples since the key does not print on the teleprinter. Where necessary, the symbol ↵ is used to represent the RETURN key.

2. Monitor prints a period (.) or dollar sign($) to which the command string is typed. System programs print a number sign (#) to which the command string is typed.

   For clarity, these symbols have been underlined when the command strings they precede are typed by the user.

3. The CTRL/C key combination is typed by holding down the CTRL key while typing the C key; ↑ C prints on the teleprinter. CTRL/C ensures that the Monitor will accept the next command, but will not necessarily immediately stop a job in progress.

4. In certain examples, sysdev: is used to represent the system device name of the particular installation's system disk, i.e., DK:, DF:, and DC:.

CONTENTS

CHAPTER 1

BUILDING THE DISK OPERATING SYSTEM

1.1 INTRODUCTION

The PDP-11 Disk Operating System (DOS) software is delivered on either paper tape or DECtape. The system software consists of the following programs, identified as shown below.

Monitor Modules	SYSLOD.SYS	
	MONLIB.SYS	
System Load Modules	PIP .LDA	(DECtape only)
	LINKOB.LDA	
	LINK11.LDA	
System Object Modules	PIP .OBJ	
	LINKOB.OBJ	
	LINK11.OBJ	
	EDIT11.OBJ	
	ODT11R.OBJ	
	LIBR11.OBJ	
	PALOB .OBJ	
	PAL11R.OBJ	
	PALSYM.OBJ	
	MODS .OBJ	(DECtape only)
Source Module	PALSYM.PAL	

(Building and running PDP-11 FORTRAN under the DOS is described in GETTING FORTRAN ON THE AIR, DEC-11-SFDB-D.)

The DOS Monitor is available in three major versions; each is tailored to run on a particular disk: RK11, RF11, or RC11. There are two versions for the RK11 disk: one for high-density (2200 BPI) and one for low-density (1100 BPI). The procedures for building the DOS differ slightly, depending on the disk that is to be used as the systems device. Section 1.4 describes how to load the system programs on the RK11 and RF11 disks. Section 1.5 describes how to load the system programs on the RC11 disk.

Briefly, the suggested sequence of operations in building DOS is:

1. Load the System Loader (SYSLOD.SYS).
2. Load and run the DOS Monitor (MONLIB.SYS).
3. Link the system object modules into load modules.

The newly-built DOS should be backed up by transferring a copy of the load and overlay modules onto paper tape or DECtape (see Section 1.6). MODS can be used to transfer the System Loader and Monitor from DECtape to DECtape (see Section 1.6.2 and Chapter 3).

This chapter contains operating procedures in a step-by-step fashion. It does not explain the purpose of each step as they are described in the DOS Monitor, Link-11, and PIP manuals.

1.2 LOADING THE SYSTEM LOADER AND DOS MONITOR

The System Loader (SYSLOD.SYS) is used to load and start the DOS Monitor (MONLIB.SYS). SYSLOD has been delivered on paper tape; and for systems with DECtape, it has been placed on DECtape for more automatic loading. To load the SYSLOD program into core, proceed to:

Section 1.2.1	for paper tape
Section 1.2.2	for DECtape

1.2.1 Loading SYSLOD from Paper Tape

The SYSLOD program is loaded from paper tape using the Bootstrap and Absolute Loaders:

1. Toggle the Bootstrap Loader into core using the console Switch Register as shown on the PDP-11 Instruction List card.
2. Set the ENABLE/HALT switch to HALT.
3. Place the Absolute Loader paper tape in the proper reader (as specified by the Bootstrap Loader). The special leader code (351) must be over the read sensors.
4. Set the Switch Register to xx7744, where xx is as specified in Step 1 above.
5. Depress the LOAD ADDRess switch.
6. Set the ENABLE/HALT switch to ENABLE.
7. Depress the START switch.

The Absolute Loader is read into core.

8. Place the SYSLOD.SYS tape in the proper reader with blank leader tape over the read sensors.

CAUTION

When placing paper tapes in the reader, ensure that blank leader tape is directly over the

the read sensors. When the name of the program is punched on the leader, blank tape separates the name from meaningful data. Exception:

Absolute Loader tape has special leader code 351 which must be over the read sensors.

9. Set the ENABLE/HALT switch to HALT.
10. Set the Switch Register to xx7500, where xx is as specified in Step 1 above.
11. Depress the LOAD ADDRess switch.
12. Set the ENABLE/HALT switch to ENABLE.
13. Depress the START switch.

The SYSLOD program is read into core and halts at location 30462. Any other halt is an error; start again at Step 8 above. (See Section 1.7 for an explanation of SYSLOD error halts.)

The SYSLOD program is in core. Go to Section 1.2.3.

1.2.2 Loading SYSLOD from DECtape

The SYSLOD program is automatically loaded from DECtape using the BM792-YB ROM Bootstrap Loader. (Users without the BM792-YB must load SYSLOD from paper tape.)

1. Mount the DECtape containing SYSLOD.SYS on DECtape 0.
2. Set the REMOTE and WRITE LOCK switches on DECtape 0.
3. Set the ENABLE/HALT switch to HALT.
4. Set the Switch Register to 173100.
5. Depress the LOAD ADDRess switch.
6. Set the ENABLE/HALT switch to ENABLE.
7. Set the Switch Register to 177344.
8. Depress the START switch.

The SYSLOD program is read into core and halts at location 30462. Any other halt is an error; start again at Step 1 above. (See Section 1.7 for an explanation of SYSLOD error halts.)

The SYSLOD program is in core. Go to Section 1.2.3.

1.2.3 Loading the DOS Monitor

There is one Monitor for the RF11 disk and one for the RC11 disk. However, there are two Monitors for the RK11 disk: a high-density

(2200 BPI) and a low-density (1100 BPI) version. Ensure that the Monitor is the one tailored for your disk.

CAUTION

> Check to ensure that the disk write protection is disabled, i.e., not on. Operation of the write protection switch(s) is explained in the appropriate hardware maintenance manual. For RK systems, the cartridge must be on Unit 0.

Regardless of whether SYSLOD was loaded from paper tape or DECtape, the DOS Monitor (MONLIB.SYS) can be loaded from either paper tape or DECtape. Go to:

Section 1.2.4	for paper tape
Section 1.2.5	for DECtape

1.2.4 Loading MONLIB from Paper Tape

The DOS Monitor is on four paper tapes, labeled:

```
MONLIB.SYS  VØØ4A     TAPE n OF 4
```

where n is the number giving the sequence in which the tapes are to be loaded. The loading procedure follows.

1. Set Switch Register bit 0 down (to 0 position).
2. Set Switch Register bit 15:

   down to replace the Monitor only

   up to clear (zero) the entire disk, which must be done when building on a new (fresh) disk.
3. Place the properly numbered MONLIB.SYS tape in the reader with blank tape directly over the read sensors.
4. Depress the CONTinue switch. The tape is read in. With bit 15 up (to 1 position), the first MONLIB tape reads in about 4 inches and pauses while the disk is being cleared.
5. At the end of the tape, SYSLOD halts at location 30120. Any other halt is an error (see Section 1.7).
6. Repeat Steps 3, 4, and 5 for each MONLIB.SYS tape in the sequence specified by the tape number.
7. Set Switch Register bit 0 up (to 1 position).
8. Depress the CONTinue switch.

The Monitor is booted into core and prints:

```
MONITOR  VØØ4A

$
```

Now log in as explained in Section 1.3.

1.2.5 Loading MONLIB from DECtape

When loading the DOS Monitor from DECtape:

1. Mount the DECtape containing MONLIB.SYS on DECtape 0.
2. Set the REMOTE and WRITE LOCK switches on DECtape 0.
3. Set Switch Register bit 0 up (to 1 position).
4. Set Switch Register bit 15:
   - down to replace the Monitor only,
   - up to clear (zero) the entire disk, which must be done when building on a new (fresh) disk.
5. Depress the CONTinue switch.

With bit 15 up (to 1 position), the DECtape spins and then pauses while the disk is being cleared.

The System Loader loads the Monitor from DECtape to disk and halts at location 30120. Any other halt is an error (see Section 1.7).

If any paper tapes are to be loaded (Monitor patches, etc.), set bit 0 down and follow the procedure in Section 1.2.1, Steps 3 through 8. If no tapes are to be loaded:

6. Depress the CONTinue switch again.

The Monitor is booted into core and prints:

```
MONITOR  VØØ4A

$
```

Now log in as explained in Section 1.3.

1.3 LOGGING IN

Log in under User Identification Code 1,1 as shown below (The user types command strings in response to Monitor's . or $.):

```
$LOG 1,1
DATE:- 25-DEC-75
TIME:- 00:01:56
$
```

Monitor responds by printing a meaningless date and time. Use the DATE and TIME commands to enter the correct date and time. However, if the system does not contain a KW11-L Real-Time Clock, the TIME command is not necessary. For example:

```
$DATE 30-JUL-71
$TIME 16:48
$
```

You are now "on-the-air" with the DOS Monitor. System programs should now be loaded from paper tape or DECtape to the disk.

Section 1.4 for RK11 or RF11 disk
Section 1.5 for RC11 disk

1.4 BUILDING SYSTEM PROGRAMS ON THE RK11 OR RF11 DISK

The DOS system programs can be loaded on the disk from either paper tape or DECtape.

Section 1.4.1 for paper tape
Section 1.4.2 for DECtape

1.4.1 Building from Paper Tapes on the RK11 or RF11 Disk

Briefly, the process is as follows: Run the Link-11 Overlay Builder, LINKOB.LDA, and then the Link-11 load module, LINK11.LDA. Use Link-11 to link PIP.OBJ into a load module on the disk where it can be used to transfer the required system object modules from paper tape onto the disk. The object modules can then be linked into load modules using Link-11.

NOTES

Command strings should be typed in response to the underlined . and $ and # characters exactly as shown below unless otherwise indicated. All command strings are terminated with the RETURN key.

1. Place the LINKOB.LDA tape in the reader and type:

```
$RUN PR:
```

2. Place the LINK11.LDA tape in the reader and type:

   a. <u>8K systems</u>:

   ```
   $GET PR:
   $SAVE LINK.LDA
   $BE
   ```

   b. <u>12K to 28K systems</u>:

   ```
   $RUN PR:
   ```

Link-11 loads into core, starts automatically, and prints:

```
LINK-11  VØØ5A

PASS 1
#
```

3. Place PIP.OBJ TAPE 1 OF 2 in the reader and type:

   a. <u>8K to 16K systems</u>:

   ```
   #PIP.LDA<PR:/CC/TA:2/E
   ```

   b. <u>20K to 28K systems</u>:

   ```
   #PIP.LDA<PR:/CC/TA:2/T:77474/E
   ```

<u>CAUTION</u>

Do not link PIP.OBJ higher than the top of 16K, i.e., with 20K or more, use the command string in b. above.

At end of tape, Link-11 prints:

```
AØØ2 Ø6332Ø
$
```

4. Place PIP.OBJ TAPE 2 OF 2 in the reader and type:

```
$CO
```

Link-11 prints the load map (an example is shown below) and pauses:

```
TRANSFER ADDRESS: Ø224Ø4
LOW LIMIT: Ø21612
HIGH LIMIT: Ø3746Ø

PASS 2

AØØ2 Ø6332Ø
$
```

5. Place PIP.OBJ TAPE 1 OF 2 in the reader and type:

```
$CO
```

Link-11 prints:

```
AØØ2 Ø6332Ø
$
```

6. Place PIP.OBJ TAPE 2 OF 2 in the reader and type:

```
$CO
```

PIP.OBJ is now linked into a load module, PIP.LDA, on the disk. Link-11 prints:

```
LINK-11  VØØ5A

PASS 1
$
```

7. Return to the Monitor by typing the CTRL/C key combination, clear core with the KILL command, and RUN PIP:

```
#↑C
.KI

$RUN PIP


PIP-11  VØØ4A

#
```

Use PIP to transfer the required object modules from paper tape to disk.

8. Place the paper tape (identified as shown in the command strings below) in the reader and type:

```
#LINKOB.OBJ<PR:/FB  }
                    } (Not required for 8K systems)
#LINK11.OBJ<PR:/FB  }

#EDIT11.OBJ<PR:/FB

#LIBR11.OBJ<PR:/FB

#PALOB.OBJ<PR:/FB

#PAL11R.OBJ<PR:/FB

#PALSYM.OBJ<PR:/FB
```

9. Return to Monitor, clear core, and run Link-11:

```
#↑C
.KI
```

a. <u>8K systems, type</u>:

```
$RUN LINK
```

b. <u>12K to 28K systems</u>, place LINK11.LDA tape in reader and type:

```
$RUN PR:
```

Link-11 prints:

```
LINK-11  V005A

PASS 1
#
```

10. Link the required object modules to the top of available core (except PIP.OBJ) using the command strings shown below (use the TOP switch to link elsewhere).

NOTE

The load maps printed by Link-11 are not shown here since the format is similar for all links, and the data varies with the core size of the system and with the TOP switch when used.

```
#LINKOB<LINKOB.OBJ/E

#LINK<LINK11.OBJ/E
```

(Not required for 8K systems)

LINKOB and LINK11 <u>must</u> always be linked to the exact same HIGH LIMIT (see load maps).

```
#EDIT<EDIT11.OBJ/E


#LIBR<LIBR11.OBJ/E


#PALOB<PALOB.OBJ/E


#PAL<PALSYM.OBJ,PAL11R.OBJ/E
```

11. Return to Monitor, clear core, and RUN PIP:

```
#↑C
.KI

$RUN PIP


PIP-11  V004A

#
```

12. Direct PIP to list a brief directory of the files on the disk (an example is shown below):

```
#/BR


DF0:

MONLIB
LINK11.OVR
LINK  .LDA
PIP   .LDA
EDIT11.OBJ
LIBR11.OBJ
PALOB .OBJ
PAL11P.OBJ
PALSYM.OBJ
EDIT  .LDA
LIBR  .LDA
PALOB .LDA
PAL   .LDA
```

13. Delete all object modules from the disk:

```
#*.OBJ/DE
```

If you plan to debug user programs using ODT-11R, PIP the tape labeled ODT11R.OBJ from paper tape to disk:

14. Place ODT11R.OBJ tape in reader and type:

```
#ODT11R.OBJ<PR:/FB
```

15. Return to Monitor,.clear core, and RUN LINKOB and PALOB:

```
#↑C
.KI

$RUN LINKOB         (Not required for 8K systems)

$RUN PALOB
```

16. RUN PIP and get a full directory of the newly-built Disk Operating System (an example is shown below):

```
$RUN PIP


PIP-11  VØØ4A

#/DI


DIRECTORY DFØ: [1,1]

ØØ-XXX-72

MONLIB         5Ø8C ØØ-XXX-7Ø <377>
LINK11.OVR      36C ØØ-XXX-72 <233>
LINK  .LDA      6Ø  ØØ-XXX-72 <2ØØ>
PIP   .LDA      7Ø  ØØ-XXX-72 <233>
ODT11R.OBJ      44  ØØ-XXX-72 <233>
PAL11R.OVR      4ØC ØØ-XXX-72 <233>
EDIT  .LDA      56  ØØ-XXX-72 <233>
LIBR  .LDA      35  ØØ-XXX-72 <233>
PALOB .LDA     1Ø7  ØØ-XXX-72 <233>
PAL   .LDA      65  ØØ-XXX-72 <233>

TOTL BLKS:  1Ø21
TOTL FILES:   1Ø
```

Now run the newly-built DOS by assembling PALSYM.PAL (the source of the PAL-11R symbol table).

17. Return to Monitor, clear core, place the PALSYM.PAL tape in the reader, and assemble the source as shown below:

```
#↑C
.KI

$RUN PAL

PAL11R  V005A

#PALSYM<PR:PALSYM.PAL



 END

A002 063320
$
```

18. Place the PALSYM.PAL tape in reader again, type CTRL/C, and then the COntinue command:

```
$↑C
.CO

000000 ERRORS
```

The symbol table was assembled without errors. There is now a file named PALSYM.OBJ on the disk; delete it using PIP.

19. Return to Monitor, clear core, RUN PIP, and delete PALSYM.OBJ:

```
#↑C
.KI

$RUN PIP


PIP-11  V004A

#PALSYM.OBJ/DE
```

A back-up system should now be created as explained in Section 1.6.

1.4.2 Building from DECtape on the RK11 or RF11 Disk

Briefly, the process is as follows: Use PIP to transfer required object modules from DECtape to disk. Use Link-11 to link the object modules into load modules.

In the following examples an RF11 disk is assumed; with an RK11 disk, use DK0: wherever DF0: appears.

NOTES

Command strings should be typed in response to the underlined . and $ and # characters exactly as shown below unless otherwise indicated. All command strings are terminated with the RETURN key.

1. Mount the object module DECtape on Unit 0 (ensure that the REMOTE and WRITE LOCK switches are set on DT0:), then load and run PIP.LDA from DT0:

a. 8K systems:

```
$GET DTØ:PIP.LDA
$SAVE PIP.LDA
$BE
```

b. 12K to 28K systems:

```
$RUN DTØ:PIP.LDA
```

PIP prints:

```
PIP-11  VØØ4A

#
```

2. Transfer required object modules from DT0: to disk:

a. 12K to 28K systems:

```
#DFØ:<DTØ:PIP.OBJ,LINKOB.OBJ,LINK11.OBJ
```

b. All systems:

```
#DFØ:<DTØ:EDIT11.OBJ,LIBR11.OBJ,PALOB.OBJ

#DFØ:<DTØ:PALSYM.OBJ,PAL11R.OBJ,MODS.OBJ
```

3. Return to Monitor by typing the CTRL/C key combination, clear core with the KILL command, and run LINKOB.LDA and LINK11.LDA:

```
#↑C
.KI
```

a. 8K systems:

```
$RUN DT0:LINKOB.LDA

$GET DT0:LINK11.LDA
$SAVE LINK.LDA
$BE
```

b. 12K to 28K systems:

```
$RUN DT0:LINKOB.LDA

$RUN DT0:LINK11.LDA
```

Link-11 loads into core, starts automatically, and prints:

```
LINK-11  V005A

PASS 1
#
```

4. Link the required object modules to the top of available core (except PIP.OBJ) using the command strings shown below (use the TOP switch to link elsewhere).

NOTE

The load maps printed by Link-11 are not shown here since the format is similar for all links, and the data varies with the core size of the system and with the TOP switch when used.

CAUTION

Do not link PIP.OBJ higher than the top of 16K, i.e., with 20K or more, use the command string in b. below.

a. 12K and 16K systems:

```
#PIP<PIP.OBJ/CC/E
```

b. 20K to 28K systems:

```
#PIP<PIP.OBJ/CC/T:77474/E
```

c. All systems:

```
#LINKOB<LINKOB.OBJ/E    }
                        }  (Not required for 8K systems)
#LINK<LINK11.OBJ/E      }
```

LINKOB and LINK11 must always be linked to the exact same HIGH LIMIT (see load maps).

```
#EDIT<EDIT11.OBJ/E

#LIBR<LIBR11.OBJ/E

#PALOB<PALOB.OBJ/E

#PAL<PALSYM.OBJ,PAL11R.OBJ/E

#MODS<MODS.OBJ/E
```

5. Return to Monitor, clear core, run PIP, and delete all object modules from the disk:

```
#↑C
.KI

$RUN PIP


PIP-11  VØØ4A

#*.OBJ/DE
```

6. Transfer ODT11R.OBJ and PALSYM.PAL from DT0: to disk:

```
#DFØ:<DTØ:ODT11R.OBJ,PALSYM.PAL
```

7. Return to Monitor, clear core, and RUN LINKOB and PALOB:

```
#↑C
.KI

$RUN LINKOB            (Not required for 8K systems)

$RUN PALOB
```

8. RUN PIP and get a full directory of the newly-built Disk Operating System (an example directory is shown below):

```
$RUN PIP


PIP-11  VØØ4A

#/DI


DIRECTORY DF0: [1,1]

ØØ-XXX-72

MONLIB       5Ø8Ø ØØ-XXX-72 <377>
ODT11R.OBJ     53 ØØ-XXX-72 <233>
PALSYM.PAL     85 ØØ-XXX-72 <233>
LINK11.OVR    36Ø ØØ-XXX-72 <233>
PAL11R.OVR    4ØØ ØØ-XXX-72 <233>
PIP   .LDA     7Ø ØØ-XXX-72 <233>
LINKOB.LDA     96 ØØ-XXX-72 <233>
LINK  .LDA     59 ØØ-XXX-72 <233>
EDIT  .LDA     56 ØØ-XXX-72 <233>
LIBR  .LDA     35 ØØ-XXX-72 <233>
PALOB .LDA    1Ø7 ØØ-XXX-72 <233>
PAL   .LDA     65 ØØ-XXX-72 <233>
MODS  .LDA     3Ø ØØ-XXX-72 <233>

TOTL BLKS:  124Ø
TOTL FILES:   13
```

Now run the newly-built DOS by assembling PALSYM.PAL (the source of the PAL-11R symbol table).

9. Return to Monitor, clear core, RUN PAL, and assemble PALSYM.PAL as shown below:

```
#↑C
.KI

$RUN PAL

PAL11R  VØØ5A

#PALSYM<PALSYM.PAL



 END

ØØØØØØ ERRORS
```

The symbol table was assembled without errors. There is now a file named PALSYM.OBJ on the disk; delete it using PIP.

10. Return to Monitor, clear core, RUN PIP, and delete PALSYM.OBJ and PALSYM.PAL from the disk:

```
#↑C
.KI

$RUN PIP


PIP-11  V004A

#PALSYM.OBJ,PALSYM.PAL/DE
```

A back-up system should now be created as explained in Section 1.6.

1.5 BUILDING SYSTEM PROGRAMS ON THE RC11 DISK

Briefly, the Link-11 load module, LINK11.LDA, is run from DECtape and then used to link required system object modules from DECtape to disk. Note that DECtape is required when the RC11 is the only disk in the system.

> NOTES
>
> Command strings should be typed in response to the underlined . and $ and # characters exactly as shown below unless otherwise indicated. All command strings are terminated with the RETURN key.

1. Mount the object module DECtape on Unit 0 (ensure that the REMOTE and WRITE LOCK switches are set on DT0:), then load and run LINKOB.LDA and LINK11.LDA from DT0:

   a. 8K systems:

```
$RUN DTØ:LINKOB.LDA

$GET DTØ:LINK11.LDA
$SAVE LINK.LDA
$BE
```

   b. 12K to 28K systems:

```
$RUN DTØ:LINKOB.LDA

$RUN DTØ:LINK11.LDA
```

Link-11 loads into core, starts automatically, and prints:

```
LINK-11  V005A

PASS 1
#
```

2. Link the required object modules to the top of available core (except PIP.OBJ) using the command strings shown below (use the TOP switch to link elsewhere).

> NOTE
>
> The load maps printed by Link-11 are not shown here since the format is similar for all links, and the data varies with the core size of the system and with the TOP switch when used.

CAUTION

~~Do not link PIP.OBJ higher than the~~
top of 16K, i.e., with 20K or more,
use the command string in b. below.

a. 12K and 16K systems:

```
#PIP.LDA<DT0:PIP.OBJ/CC/E
```

b. 20K to 28K systems:

```
#PIP.LDA<DT0:PIP.OBJ/CC/T:77470/E
```

c. All systems:

```
#LINKOB<DT0:LINKOB.OBJ/E
                                  (Not required for 8K systems)
#LINK<DT0:LINK11.OBJ/E
```

LINKOB and LINK11 must always be linked to the exact same HIGH LIMIT (see load maps).

```
#EDIT<DT0:EDIT11.OBJ/E

#LIBR<DT0:LIBR11.OBJ/E

#PALOB<DT0:PALOB.OBJ/E

#PAL<DT0:PALSYM.OBJ,PAL11R.OBJ/E

#MODS<DT0:MODS.OBJ/E
```

3. Return to Monitor by typing the CTRL/C key combination, clear core with the KILL command, mount a fresh DECtape on DECtape Unit 2, and use PIP to ZEro DT2: (the back-up DECtape):

```
#↑C
.KI
```

a. 8K systems:

```
$GET DT0:PIP.LDA
$SAVE PIP.LDA
$BE
```

b. 12K to 28K systems:

```
$RUN PIP
```

PIP loads into core, starts automatically, and prints:

```
PIP-11  V004A

#DT2:/ZE
```

4. Return to Monitor, clear core, run MODS, and copy SYSLOD.SYS and MONLIB.SYS from the supplied DECtape (DT0:) to the back-up DECtape (DT2:):

```
#↑C
.KI

$RUN MODS

MODS-11  V003A
#DT2:<DT0:
```

5. Return to Monitor, clear core, run PIP, and copy the following load modules onto DT2:

```
#↑C
.KI

$RUN PIP


PIP-11  V004A

#
```

a. <u>8K systems</u>:

```
#DT2:<DT0:LINKOP.LDA,ODT11R.OBJ,PALSYM.PAL

#DT2:<PIP.LDA,LINK.LDA
```

b. <u>12K to 28K systems</u>:

```
#DT2:<DT0:ODT11R.OBJ,PALSYM.PAL

#DT2:<LINKOP.LDA,PIP.LDA,LINK.LDA
```

c. <u>All systems</u>:

```
#DT2:<EDIT.LDA,LIBR.LDA,PALOP.LDA,PAL.LDA,MODS.LDA
```

6. Delete the following load modules from disk:

```
#LINK.LDA,EDIT.LDA,LIBR.LDA,PAL.LDA,MODS.LDA,PIP.LDA/DE
```

7. Return to Monitor, clear core, and run LINKOB.LDA and PALOB.LDA:

```
#↑C
.KI

$RUN LINKOB.LDA     (Not required for 8K systems)

$RUN PALOB.LDA
```

8. Run PIP from DT2:

```
$RUN DT2:PIP


PIP-11  VØØ4A

#
```

a. <u>8K systems</u>, delete PALOB.LDA from disk:

```
#PALOB.LDA/DE
```

b. <u>12K to 28K systems</u>, delete LINKOB.LDA and PALOB.LDA from disk:

```
#LINKOB.LDA,PALOB.LDA/DE
```

9. List a full directory of the disk and back-up DECtape (example directories are shown below):

```
#DCØ:,DT2:/DI


DIRECTORY DCØ: [1,1]

3Ø-JUL-71

MONLIB        284C ØØ-XXX-7Ø <377>
LINK11.OVR     36C 3Ø-JUL-71 <233>
PAL11R.OVR     4ØC 3Ø-JUL-71 <233>

TOTL BLKS:    36Ø
TOTL FILES:     3
```

(directories continue on next page)

```
DIRECTORY DT2: [1,1]

3Ø-JUL-71

SYSLOD.SYS    31C 3Ø-JUL-71 <377>
MONLIB.SYS    84  3Ø-JUL-71 <377>
LINKOB.LDA    24  3Ø-JUL-71 <233>
ODT11R.OBJ    13  3Ø-JUL-71 <233>
PALSYM.PAL    21  3Ø-JUL-71 <233>
PIP   .LDA    17  3Ø-JUL-71 <233>
LINK  .LDA    15  3Ø-JUL-71 <233>
EDIT  .LDA    14  3Ø-JUL-71 <233>
LIBR  .LDA     9  3Ø-JUL-71 <233>
PALOB .LDA    27  3Ø-JUL-71 <233>
PAL   .LDA    17  3Ø-JUL-71 <233>
MODS  .LDA     8  3Ø-JUL-71 <233>

FREE BLKS:   282
FREE FILES:   44
```

The PDP-11 DOS is now ready for use. Run PAL from DT2: and assemble PALSYM.PAL (the source of the PAL-11R symbol table).

10. Return to Monitor, clear core, RUN PAL, and assemble as shown below:

```
#↑C
.KI

$RUN DT2:PAL

PAL11R  VØØ5A

#PALSYM<DT2:PALSYM.PAL




 END

ØØØØØØ ERRORS
```

The symbol table was assembled without errors. There is now a file named PALSYM.OBJ on the disk; delete it using PIP.

11. Return to Monitor, clear core, RUN PIP from DT2:, and delete PALSYM.OBJ:

```
#↑C
.KI

$RUN DT2:PIP

PIP-11  VØØ4A

#PALSYM.OBJ/DE
```

1.6 CREATING A BACK-UP SYSTEM

A copy of the newly-built DOS can be transferred from disk to paper tape or DECtape. This back-up system is then available to facilitate building subsequent systems if necessary, or it can be used to transfer a fresh copy of any system load module into core or onto the disk.

1.6.1 On Paper Tapes

To be backed up with paper tapes, RUN PIP and transfer a copy of any disk file (excluding MONLIB and overlay files, OVR) onto punched paper tape. For examples:

```
$RUN PIP

PIP-11  V004A

#PP:<EDIT.LDA

#PP:<LIBR.LDA

#PP:<PAL.LDA
```

and etc.

Each paper tape punched should be labeled immediately after it is removed from the punch bin.

1.6.2 On DECtape

All files of the newly-built DOS can be transferred to a back-up DECtape. MODS is used to copy SYSLOD and MONLIB from DECtape to DECtape, and PIP is used to copy source, object, load, and overlay files.

1. Dial a DECtape transport to Unit 2.
2. Mount a fresh DECtape on DT2:.
3. Set the REMOTE and WRITE ENABLE switches on DT2:.
4. RUN PIP.
5. ZEro DT2: using PIP.
6. RUN MODS.
7. Copy SYSLOD.SYS and MONLIB.SYS onto DT2: using MODS.
8. RUN PIP.
9. Copy all or any disk file onto DT2: using PIP.

The printout might appear as shown below when transferring all disk files and SYSLOD and MONLIB onto DT2:

```
$RUN PIP


PIP-11  V004A

#DT2:/ZE

#↑C
.KI

$RUN MODS

MODS-11  V003A
#DT2:<DT0:
#↑C
.KI

$RUN PIP


PIP-11  V004A

#DT2:<*.LDA,*.OBJ,*.PAL,*.OVR/CO

#
```

1.7 RECOVERING FROM SYSTEM HALTS

1.7.1 SYSLOD Error Halts

When a system loading error is detected, the SYSLOD program halts at the address shown in the console ADDRESS REGISTER with a display in the DATA register. The halts can be interpreted as follows:

Halt Address	DATA Display	
30222	2	Input data error.
	3	File open error (possibly a wrong tape was mounted).
	4	Load tape format error.
31570 (RK11 high) 31624 (RK11 low) 31620 (RF11) 31600 (RC11)		Read error occurred in trying to read the Monitor into core from disk.

Generally, when an error halt occurs the program must be reloaded. However, in the case of mounting the wrong tape (DATA=3), mount the correct tape (with MONLIB, start over with the first tape) and press the CONTinue switch twice--loading should resume correctly; otherwise, reload SYSLOD.SYS.

1.7.2 Halts While Linking

When the linking process is interrupted by a system error (Snnn), the cause is often due to:

1. The filename already exists on the directory. The user can use PIP to either specify a different filename in the command string or remove the filename from the directory with the UNlock and DElete commands, and then link again.
2. A bad read. The user can use PIP to UNlock and DElete the filename from the directory, and then link again.

Note that Link-11 enters the filename in the directory and then attempts to link the module.

1.7.3 Fatal Error Halts

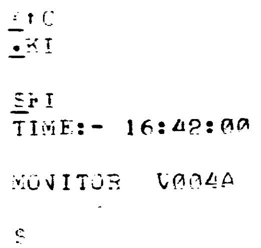
When a fatal error message (Fnnn) is printed the system program is suspended. In most cases the BEgin command will restart the system program. If this fails, type CTRL/C followed by the KILL command and then restart the system program with the RUN command.

When the keyboard is locked, restart the Monitor as explained in Section 1.8.2.

1.8 RESTARTING THE MONITOR

1.8.1 From Keyboard

The user can, at any time, restart the Monitor with the FInish command. For example, when a programming session is concluded with the FI command, the Monitor is automatically restarted for another session. Unless the last character on the teleprinter is a $, the KILL command should precede the FI command. Printout might be:



```
$↑C
.KI

$FI
TIME:- 16:42:ØØ

MONITOR  VØØ4A

$
```

The user should then log in (see Section 1.3) under his UIC and issue the desired Monitor command. This type of restart does not alter file directories, date, or the time clock.

1.8.2 From Disk or DECtape

The BM792-YB ROM Bootstrap Loader can be used at any time to load a fresh copy of the Monitor into core from disk or DECtape. (Users without the BM792-YB must load SYSLOD from paper tape, as explained in Section 1.2.1.)

1. Set the Switch Register to 173100.
2. Set the ENABLE/HALT switch to HALT.
3. Depress the LOAD ADDRess switch.
4. Set the Switch Register to:

177406	for	RK11 systems
177462	for	RF11 systems
177450	for	RC11 systems

5. Set the ENABLE/HALT switch to ENABLE.
6. Depress the START switch.

A fresh copy of the Monitor is booted into core and prints:

```
MONITOR V004A

$
```

Log in with the appropriate UIC. This type of restart does not alter file directories; date and time must, however, be re-entered.

A fresh copy of the Monitor can be booted into core from DECtape as explained from Section 1.2.2 onward.

CHAPTER 2

~~PROGRAMMING NOTES AND CAUTIONS~~

2.1. DOS MONITOR, VØØ4A

2.1.1 Programming Notes

1. The Monitor expects to be able to write on any part of a disk surface, therefore, the write protection must be disabled. In particular, the user should be careful to ensure that the appropriate switches are correctly set at times when there is no resident Monitor in core to produce the proper error diagnosis (i.e., during SYSLOD or a Monitor boot). Otherwise the system will not operate.

2. If it is desired to copy a DECtape containing SYSLOD, it is necessary to use either a block-by-block DECtape copy program or the MODS program (see Chapter 3). The nature of the SYSLOD file does not allow its being copied by PIP, i.e., it is a position-dependent contiguous file.

3. When the attention message AØØ2 is printed by the Monitor because of a DECtape not ready situation, it is safe to dial the DECtape (by rotating the thumb switch) to the proper drive number; <u>unless</u> the faulting request is OPENO, <u>and</u> the message is due to the requested drive being in WRITE LOCK status, <u>and</u> you wish to dial another transport to the drive number currently requested. In this one instance the directory of the DECtape has already been read by the Monitor and the Monitor is unaware that the dialing took place. Therefore, the directory of the latter tape will be garbled.

4. If CTRL/C is typed <u>immediately</u> following the RETURN key, the Monitor will echo the ↑C and hang -- a Monitor restart is required. A very slight hesitation (blink your eyes) after typing the RETURN key is all that is necessary to circumvent this problem.

5. It is legal to type a command to the Monitor at any time. However, if you are giving a command to the Monitor when it detects a situation requiring a diagnostic message (e.g., an OPEN failure in a running program while you are typing CTRL/C TI), a conflict for the swap buffer will result which causes the Monitor to hang. If this happens, a Monitor restart is required.

6. The use of the BEGIN command after a program failure requires some care. As far as possible the Monitor tries to clean up for the user by closing any input files currently open for input and deleting any files not yet completely created. The Monitor employs its own methods for doing this rather than using normal file-structured operations, because these operations would attempt to write to the relevant device data stored in core which must be suspect following the program crash. In particular, no attempt is made to change bit maps as permanently recorded. This can mean that on disks which use several bit maps to cover their surface, some blocks allocated to the files being deleted will not be released for further use, even though the files themselves have been removed. A series of crashes can thus lead to the disk being filled, although other evidence appears to contradict this. The user should therefore consider whether he should chance disk-corruption and use KILL rather than BEGIN and then delete the incomplete files with PIP. It also follows that if the automatic deletion effected by BEGIN could lose irrecoverable data to the user, he should again KILL after a crash.

7. If the system crashes to the extent that rebooting the Monitor from scratch is the only recourse, the user must be aware that when a file is opened for creation, its initial directory entry is made immediately. However, it is only when the file is closed that all the blocks allocated to it are permanently recorded as such. Because of the crash, the file may not be closed; some blocks given to it will still remain free and might be used for some other purpose. Nevertheless, the initial directory entry still exists and the only way to remove

it is by deletion, which includes release of all blocks associated with the file. If the deletion ~~is not done immediately after the crash and before~~ any other operations are carried out, it is possible that disk-corruption can ensue. It can be avoided only by the user doing the necessary clean-up as soon as convenient.

8. The Monitor includes drivers for RF11, RK11, and RC11 disks, and the CR11 card reader, as well as those available in earlier versions. The RF11 and RC11 Drivers are quite straightforward. The RK11 Driver expects all its associated units to be set up for high-density usage, except when the resident system device is the low-density cartridge. The CR11 Driver is suitable for reading only ASCII cards, though these may originate from either 026 or 029 punches; and the user has the option of requesting suppression of columns 73-80 and trailing blanks in preceding columns.

    However, the sources for both these drivers (which are available from DEC's Program Library) contain conditional assembly features to alter their usage if required. The Monitor also contains the necessary links for the driver for MT11 Magnetic Tape which will be available shortly.

9. Another edition of the DOS Monitor Programmer's Handbook (DEC-11-MWDB-D) is available. The user's attention is drawn particularly to the changes listed below:

    a. Change in SPECIAL FUNCTIONS CALL
    b. Extension of MODIFY command
    c. Introduction of new code under EMT 41 for requesting System Device Name
    d. Differentiation between ASCII and Binary in TRAN call
    e. Invalid code error in EMT 42

10. This release of the Monitor has the following discrepancies with the DOS Monitor Programmer's Handbook as published:

    a. No files are deleted on FInish, regardless of their protection codes

b. Macro calling for Monitor operations is not yet possible. The user must instead directly expand the sequences required in basic assembly language.

c. The only valid device accepted by console DUmp command is the line printer (LP:).

11. By its nature a dataset, other than one being updated, can supply input or accept output but not both simultaneously. Therefore, it is not possible to issue READ and WRITE commands to the same dataset at any one time and obtain meaningful results. Because a device, such as paper tape reader or punch, normally forces uni-directional requests or because the device is treated as file-structured, an appropriate OPEN command must be issued and this forces the direction. The Monitor provides the necessary checks in these cases. However, it does not presently protect the user of the Teletype, which is bidirectional and not file-structured. READ and WRITE on the same dataset will be accepted but will then cause invalid operations. The user must exercise caution by making two datasets available or, if the same dataset is used, logically changing the direction by : OPEN, READ, CLOSE, OPEN, WRITE, CLOSE.

12. The following bugs are currently known to exist in this version of DOS Monitor (VØØ4A).

a. When the system disk is RK11, an attempt to access a disk block outside the range of the disk surface (i.e., greater than 4800 on a high-density cartridge or 2400 on a low one) will cause the system to hang. If this should occur, reboot the Monitor.

b. If more than one output file is concurrently opened on an RK11 disk on any unit other than 0, it is highly likely that that disk's directory structure will be corrupted when the files are closed. For the present, users should restrict their usage of disks on these units to one output file at a time.

c. Occassionally it has been found that transferring card files to DECtape under PIP, where the blank supp suppress oftion for cards is exercised, can cause garbage data to be added to the end of the output file. Until this problem is rectified, the user can remove the garbage via the Editor.

2.1.2 Reserved Filename Extensions

The filename extension is a convenient means of storing information about the attributes of a file. DEC reserves the following extensions for use by DEC-supplied PDP-11 system programs. Additional extensions may be pre-empted for our use as the need arises.

TABLE 2-1

Reserved Filename Extensions

Extension		Attribute
ALG	*	ALGOL source file
BAS		BASIC source file
BAK	**	Backup file
BLI	**	BLISS source file
CBL	*	COBOL source file
CMD		Command file
CRF		Input to cross-referencing program
DAT	**	DATA file for FORTRAN job
DDT		Reserved for DDT
FTN	**	FORTRAN source file
FCL	*	FOCAL source list
LBO	**	Library of object modules (other types of libraries may also be implemented)
LDA	**	Load module, Absolute
LDR	**	Load module, Relocatable
LOG	**	Logging file
LSP	*	LISP source file
LST	**	Listing file
MAC		MACRO assembler source file
MAP	**	MAP file
MFD		Master file directory
OBJ	**	Object module
OPR	**	Program generation information
OVR		Overlay
PAL	**	PAL assembler source file
PL1	*	PL/I source file
RNO	**	Input for RUN-OFF program
RPG	*	RPG source file
SNO	*	SNOBOL source file
SPC	**	SPEC format text
SYM		File of symbols
SYS		System management
TMP	**	Temporary scratch file
UFD		User file directory

* Just in case we ever need them.
** Used now or in the very near future

Unstarred items _may_ be used in the near future. No definite plans.

2.1.3 System Use of UIC's

The UIC is composed of two 8-bit fields and is represented as two 3-digit octal numbers whose values are less than or equal to 377. Thus, [2,316] is a legal UIC. The first field is referred to as the _project_ number; the second field is the _programmer_ number.

The following UIC's are reserved for use with DEC software. Installations should not use these UIC's for any purpose other than those purposes stated in DEC documentation.

Project numbers:	Ø-17	and	377
Programmer numbers:	Ø-17	and	377

Thus, the following are among those reserved:

[2,21Ø] [1,1] [Ø,Ø]
[17,377] [4,17] [Ø,15]

2.1.4 Device Names in Radix-5Ø

The Radix-5Ø values for device names which appear on Monitor diagnostic message printouts are as follows:

Device	Radix-5Ø Value (Base 8)
DF	1476Ø
DK	1527Ø
DC	1457Ø
KB	4242Ø
PR	6332Ø
PP	632ØØ
LP	466ØØ
DT	16Ø4Ø
MT	5214Ø
CR	1262Ø
PT	6344Ø

2.2 PAL 11R ASSEMBLER VØØ5A

~~2.2.1 Programming Notes~~

The symbol table capacity of PAL-11R VØØ5A running under the Disk Operating System and utilizing overlays varies as a function of the I/O devices involved in the assembly. This is extremely relevant for 8K of core memory configurations.

At assembly time, 5497 decimal words are always occupied by the following items:

Resident PAL-11R	32Ø9
Permanent Symbol Table	336
PAL-11R's stack	1ØØ*
Absolute loader	98
Unnamed .CSECT entry	4
User Current Location "." symbol	4
Resident DOS (RF11 Disk)	1746**
	5497

In addition to this figure, consider the non-resident requirements of DOS which consist of the buffers and drivers needed for the assembly and $256_{1Ø}$ words for the .OPEN processor.

The Assembler allows for:

two input devices (files)
three output devices (files)
    i.e., binary
          listing
          symbol table
and the command output device.

The command output device and buffer is always required and occupies a total of 56Ø decimal words.

```
 5497
+56Ø
 6Ø57
```

*The Assembler will decrease its stack size in favor of symbol table entries. If the symbol table requirements are still too large, the assembly will be aborted with the FØØ1 (STACK OVERFLOW) DOS error message.

**If the system device is RK11, this figure increases by $144_{1Ø}$ words. If the system device is RC11, this figure decreases by 7 words.

Periodically during every pass of the assembly, the .OPEN processor will be needed; thus, for symbol table purposes it can be considered as a permanent requirement.

```
 6Ø57
 +256
 6313
```

The input file or files is required for the duration of an assembly. The medium chosen for input directly affects symbol table storage space, e.g.,

	Buffer Size per File	Driver Size
RF11 or RC11 Disk	96 decimal words	system device included in resident DOS
RK11 Disk	288 decimal words	
DECtape	288 decimal words	165 decimal words

The user saves 357 decimal words if his input file resides on the RK or RC disk rather than on DECtape. Thus, PIPing the sources from DECtape to disk prior to assembly increases symbol table storage by 89 entries (4 words per symbol).

The output files are only required during their respective assembly passes; that is,

symbol table	-	end of pass 1
listing	-	pass 2
binary	-	pass 2

Choice of output medium has the same benefits here and, additionally, the user can "string out" the non-resident needs by specifying via the PASS command string option that the listing be output during pass 3.

Three examples, one worst case and the other two considerably better, will illustrate the above-mentioned facts:

EXAMPLE #1 (Worst case):

```
#DT1:TEST.OBJ,DT2:TEST.LST,DT2:TEST.SYM<DT3:TEST.PAL
     ↑            ↑            ↑          ↑
    288          288          288*      165+288
```

```
1Ø29+6313 = 7342  leaving room for 211 user symbols
                  or .CSECT names in 8K of core.
```

*The binary file in pass 2 would use the same space required by the symbol table file at the end of pass 1.

EXAMPLE #2:

~~Note that when a device is not specified in the command string,~~ the system device is taken as the default. Assuming the system device is RF or RC disk:

```
#TEST.OBJ,TEST.LST,TEST.SYM<TEST.PAL
  ↑          ↑          ↑       ↑
  96         96         96      96
```

```
288+6313 = 66Ø1   leaving room for 397 user symbols or
                  .CSECT names in 8K of core.
```

The listing and symbol table files can later be PIPed from disk to a hard copy device such as the line printer, teleprinter, paper tape punch, etc.

EXAMPLE #3:

```
#TEST.OBJ,TEST.LST/PASS:3,TEST.SYM<TEST.PAL
    ↑          ↑                ↑        ↑
    96         96               96*      96
```

The PASS:3 switch makes the binary and listing outputs occur during separate passes; thus, they share output buffers and effectively increase symbol table capacity by 96 words or 24 symbols. The total capacity is now 421 symbols.

The user should always be aware of his symbol table needs relative to what his hardware configuration can provide. If he does this, he can segment programs that will exceed the limits, assemble them separately, and link the object modules with Link-11.

2.2.1.1 Running the Overlay Builder (PALOB.LDA) -- The overlay builder creates the file of overlays via logical data set name OVR which has the default file specifier:

```
sysdev:PAL11R.OVR[LOGIN UIC]<233>
```

It is recommended that the creation of the file of overlays be accomplished either while

1. logged in under [1,1]

*The binary file in pass 2 would use the same space required by the symbol table file at the end of pass 1. A UIC specification is required if PALOB.LDA is under a UIC other than the LOGIN UIC.

or

2. with the Monitor ASSIGN command overriding the LOGIN UIC, e.g.:

```
.AS sysdev:PAL11R.OVR[1,1],OVR
```

In either of the above environments, the overlay builder can be executed, e.g.:

```
.RUN sysdev:PALOB.LDA*
```

and will create the file of overlays, PAL11R.OVR, on the system disk with a UIC of [1,1] and then will exit to the Monitor which is indicated by the output of the "$".

Transfer the Assembler from DECtape to the Monitor disk area with PIP.

```
$PAL[1,1]<DTØ:PAL11R.LDA**
```

2.2.1.2 Running the Assembler -- The Assembler can now be executed via the Monitor command RUN:

```
.RUN    PAL
```

it will search for its overlays first under the file specifier

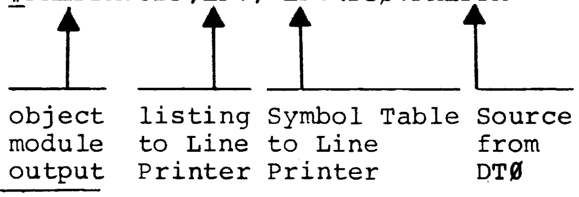
```
sysdev:PAL11R.OVR[LOGIN UIC]
```

and, if not found, then under

```
sysdev:PAL11R.OVR[1,1]
```

A typical command to PAL-11R to assemble the source of PALSYM that exists on DECtape with the file name PALSYM.PAL would be:

```
#PALSYM.OBJ,LP:, LP:<DTØ:PALSYM
```



*A UIC specification is required if PALOB.LDA is under a UIC other than the LOGIN UIC.

**A UIC specification will be required if PAL11R.LDA is under a UIC other than the LOGIN UIC.

If the source was on paper tape, the command would read:

```
#PALSYM.OBJ,LP:,LP:<PR >
```

source from
paper tape
reader

The Monitor informs the user when to load the reader with the tape by typing

```
AØØ2    6332Ø
$
```

on the command output device. AØØ2 denotes device not ready; 6332Ø is PR (paper tape reader) in Radix-5Ø notation.

When the tape is ready, the user types the Monitor command COntinue.

The end of an assembly pass is indicated by END being printed on the command output device.

The end of an assembly is indicated by "#" (the command string request character) being printed on the command output device

2.2.1.3 Overlays on DECtape

Note that DECtape can be utilized as the overlay device. All that is required to accomplish this is:

1. Make the desired file specifier change prior to running PALOB.LDA

   ```
   .AS DTØ:PAL11R.OVR[1,1],OVR
   .RUN DTØ:PALOB.LDA
   ```

2. Make the <u>same</u> file specifier change prior to running PAL

   ```
   .AS DTØ:PAL11R.OVR[1,1],OVR
   .RUN PAL
   ```

2.2.2 Cautions

1. If no extension is specified for an input file, the Assembler will first search for a file using the default

extension "PAL". If this file is not present, the Assembler will then search for a file with a null extension. Both files being absent would result in a fatal Monitor error.

2. Macros for the DOS EMT's are not implemented. The user must type the expanded form.

3. @A(RØ)
   @A(RØ)+
   @A-(RØ)

   are all accepted by the Assembler as valid representations of the indexed deferred addressing mode @A(RØ).

2.3 EDIT-11 TEXT EDITOR VØØ2A

~~2.3.1 Programming Notes~~

1. Use of PT: and KB: -- When using KB: for input, commands to Edit-11 which read text into the buffer require that the user manually supply the end-of-page or end-of-data flags. The procedure is as follows: (↵ denotes the RETURN key):

   a. To end a Read command from KB: the user should type CTRL/C followed by

      END

      followed by the RETURN and LINE FEED keys in that order.

   b. To end an EXit command from KB: the user should type CTRL/C followed by

      END

      followed by the RETURN and LINE FEED keys in that order.

   c. When using PT: for input, the procedure is:

      (1) Put paper tape in reader and move switch to START.

      (2) Start Edit-11 and type command string.

      (3) Edit-11 will open the primary output file, read a few characters, then print an * to signal its readiness for a command.

      (4) Leave the reader on. Edit-11 will read tape when commanded to do so.

      (5) When the end-of-tape is reached during a read operation, signal so by typing CTRL/C followed by

         END PT

         followed by the RETURN and LINE FEED keys in that order.

      (3) When Edit-11 finishes punching leader, it pauses and waits for the punch to be turned off. When you have done so, pressing the LINE FEED key will resume Edit-11 operation.

      (4) From this point on, Edit-11 will pause before and after every output command to allow you to turn the punch on and

off. Pressing the LINE FEED key will cause it to continue.

d. When using PT: for output, the procedure is:

(1) With punch off, start Edit-11 and type command string.

(2) When Edit-11 opens the punch for output, it will begin outputting nulls to the teleprinter. Turn on the punch.

2. Monitor A Errors -- If at any time during the editing process, or during the initialization, an AØØ2 error is received from the Monitor, the user is required to take action to ready a device. Common causes are specifying PR: when there is no tape in the reader or specifying the wrong DECtape drive. Correct the problem, then type:

```
CO
```

followed by the RETURN key. Edit-11 will continue.

3. PT: Versus DF: -- If the user plans to do a lot of paper tape editing, he can often save a lot of time by using the disk to store edited versions of files. Edit from PT: to DF: then do editing from disk to disk. When satisfied, PIP the good file back to paper tape.

4. Use of Subsidiary Output -- As mentioned in the Edit-11 manual, use of a fast output device such as LP: for listing text via the EW command will greatly increase editing turn-around time. Use the L command sparingly and the printer often.

2.3.2 Cautions

1. If Edit-11 encounters an invalid Open because of a file already existing, it does not recover. Rather, an FØ12 Monitor error will be printed on the teleprinter, and it is necessary to restart the Editor with CTRL/C and BEgin.

2. The F and T commands presently provide for outputting only in the primary output device. To provide a Form Feed on the secondary output file, the Form Feed character must be inserted into the page buffer by the user. There is also presently no means of punching trailer on the secondary output device.

2.4 ODT-11R DEBUGGING PROGRAM VØØ2A

2.4.1 Caution

A problem exists in version VØØ2A and previous versions with regard to the saving of breakpoints at ODT's three entry points (OD K; OD R; OD). There are two substantially different ways that ODT can be entered:

1. ODT was used to set breakpoints in a program and the program crashed when it was given control by ODT. In this case, the Monitor will have control and the user will have typed:

   OD K

   or the user will have manually restarted ODT.*

2. Any other entry to ODT (via Monitor commands such as RV, BE, OD; or because a breakpoint occurred).

VØØ2A does not distinguish between these cases. Consequently, when ODT saves breakpoints and restores user instructions to their previous values, it will restore memory correctly in case 1 , but will move an octal ØØØØ3 into locations on which breakpoints have been set in case 2. This will be fixed in a subsequent release of ODT.

Until then, avoid exiting from ODT to the Monitor with breakpoints set in the program if ODT is to be re-entered. For example: when patching and saving prior to debugging:

```
$GE  prog
$OD
ODT11R  VØØ2A
*                          (make patches here)
  .
  .
  .
$SA
$OD K                      (reenter ODT, preserve re-
                           location registers)
BEnnnnnn                   (Set breakpoints here)
*
  .                        (Debug - if crash occurs,
  .                        exit OD K)
  .                        (make additional patches)
  .
  .
$;B                        (clear breakpoints)
*C
$SA                        (save includes ODT)
$OD K
BEnnnnnn                   (reenter ODT)
  .                        (continue using ODT)
  .
```

*ODT's three starting addresses are at ODT's base +172_8(OD), 174_8(OD R), and 176_8(OD K).

2.5 PIP FILE UTILITY PACKAGE, VØØ4A

2.5.1 Programming Notes

1. The /CO feature has a number of restrictions which will be fixed in the next version. Use of the asterisk feature with the /CO switch in a merge operation only obtains the first of the asterisk specified files. No error message is given.

2. There is space for only 15 different UIC's on the RF11 disk system.

2.5.2 Cautions

1. To transfer ASCII cards from the card reader to disk or DECtape, the /FA switch must be used.

2. If the user attempts to transfer a series of files (such as with the * command) in which one has a protection code that does not allow transfer, a fatal Monitor error will result and successive files designated for transfer in that command string will not be transferred.

3. CTRL/C RE does not yet work. Use CTRL/C BE.

4. If a file appears on disk or DECtape with a name not acceptable to the CSI, it must be deleted using the * feature. For example:

   ```
         ABC.  A    (2 spaces in extension)
   ```

   is not acceptable and should be deleted.

   ```
         ABC.*/DE
   ```

   The above works unless the file is locked; in which case it must be deleted using SYSLOD, as the * feature has not yet been implemented for UNLOCK.

5. The * feature is not implemented for the RENAME and UNLOCK switches.

6. When transferring files, the * cannot be specified in the output field of the command string; it will create a file with the extension (space)AB, and no error message is given.

To copy files from one device to another using the *, the * must be in the input field of the com~~mand string. For example:~~

```
dev1:<dev2:*.*
```

or

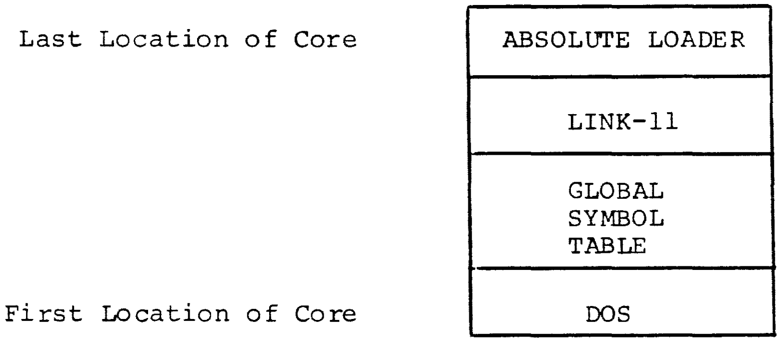
```
dev1:<dev2:*.PAL
```

2.6. LINK-11 LINKER VØØ5A

2.6.1 Programming Notes

The whole Linker resides in core while it is linking. Neither the input files nor the output file are ever "in core" in their entirety. In pass one, the Linker reads the input files one buffer at a time and extracts the global symbols from them. These symbols are kept in core in the global symbol table for the whole duration of the linking and relocating process. In pass two, the input files are read again, one buffer at a time. (The output file will need to be loaded into core subsequently by a system loader.)

The memory map as shown below represents core while Link-11 is in control:



To utilize all possible core space for additional symbol table space, use the delivered load module of Link-11 (LINK11.LDA) to relocate the object module of Link-11 (LINK11.OBJ) as high in core as possible.

1. If a top or bottom was specified once, then at each successive linking-relocating task if a top or bottom is not specified, the one formerly specified will still be in effect (<u>and not the default assumption</u>).

2. No switch can appear alone by itself in an input specification (i.e., ,/U, and ,/E are illegal; ,dev:filname.ext/U, and def:filname.ext/E are legal).

3. There are several ways to link ODT11R.

   a. dev:outfil.ext,mapdev:<dev:test,dev:ODT11R.OBJ/OD/E

      (The OD switch is set)
      allows the Monitor command BE to start the test

program, OD to start ODT (assuming in all cases that the Monitor command GET OUTFIL.EXT precedes).

b. dev:outfil.ext,mapdev:<dev:ODT11R.OBJ.dev:test/E

(no OD switch)
allows BE to begin ODT. OD will have no effect.

c. dev:outfil.ext,mapdev:<dev:ODT11R.OBJ/E

allows ODT to be run by itself with the BE command.

4. When using the /OD switch:

default device	--	as usual, disk
default file name	--	ODT
default extension	--	as usual, OBJ
secondary UIC	--	1,1

If ODT.OBJ is on either the system or user disk area, the input file specification for ODT in the CSI reduces to:

```
/OD
```

For example:

```
<DT1:PROG,/OD
```

when the device for ODT is DT1. /OD, cannot be the first input file specification. However, the following is valid:

```
sysdev:ODT[1,1]/OD,
```

5. a. When creating object modules by PAL-11R or when transferring object modules by PIP, always specify in the output file specification a file extension in order to be able to link it with Link-11, which assumes extension .OBJ when none is specified.

   b. Always specify a file extension for the output file of Link-11; specify the same file extension when issuing the Monitor commands RUN or GET.

6. If a WRITE error occurs, the dev:filnam.ext part of the error message is not printed.

7. The Linker will not link libraries from paper tape.

8. The /OD and /TR switches are mutually exclusive.

9. If the user has a library which contains backward references, put the library file into the command

string to the Linker repeatedly until all references are satisfied. For example, given the four object modules:

```
    PA, PB, PC, PD
```

Object Module PA:

```
   .TITLE PA
   .GLOBL B
   .END
```

Object Module PB:

```
   .TITLE PB
   .GLOBL B,C
 B:1
   .END
```

Object Module PC:

```
   .TITLE PC
   .GLOBL C
 C:1
   .END
```

Object Module PD:

```
   .TITLE PD
   .END
```

where PA references PB,and PB references PC.

If we create three files F1.OBJ, F2.OBJ, and F3.OBJ where:

(a) file F1.OBJ contains object module PA,

(b) file F2.OBJ contains a library in which PB is first and PC is second, and

(c) file F3.OBJ contains object module PD

then the four programs can be linked with the following command strings:

```
    #OUT,MAP<F1,F2/L/U
```

There were no undefined references left, so conclude the linking process by typing the command string:

```
    #F3/E
```

If file F2.OBJ contains a library in which object module PC is first and PB is second, then after the command string:

```
#OUT,MAP F1/F2/L/U
```

the following message will be printed:

```
UNDEFINED REFERENCES
B
```

Now issue the second command string:

```
#F2/L/U/
```

There will be no more undefined references left this time, so the linking process can be concluded by typing the third (final) command string:

```
#F3/E
```

The four linking modes are shown in Table 2-2.

TABLE 2-2

LINK MODES

	Function Performed by Linker		
	Generate	Generate	Report
Format	Load Module	Load Map	Errors
output,<map inputs	yes	yes	yes
output<inputs	yes	no	yes
,map<inputs	no	yes	yes
<inputs	no	no	yes

2.6.2 Cautions

If the user means to type:

```
PP:,LP:<PR:/E
```

but accidentally types:

```
PP:,LP<PR:/E
```

the load map (an ASCII file) will be punched on the paper tape before the load module (a binary file). The Linker will not detect the error since the erroneous string is a legal one, thus the load module cannot be loaded (since there is an ASCII file in front of it).

There is a fair amount of blank tape between the load map and the load module, so either separate them or relink the task.

Files will be linked to the top of available core unless the Top (/T) or Bottom (/B) switch is used to specify a different area in core. PIP, however, must not be linked higher than the top of 16K. Therefore, with systems of 20K or more, the Top switch must be specified in the command string, i.e., /T:77ØØØ.

LINK-11 will malfunction when a blank field appears for the map specification. For example:

```
#LOAD.MOD,<OBJ.MOD/E
```

The lack of an output device specification causes DOS to print an apropriate message and halt.

The following command strings will be serviced properly:

```
#LOAD.MOD<OBJ.MOD/E

#LOAD.MOD,LP:<OBJ.MOD/E
```

In the first case no map is required; in the second case the line printer is specified as the map printing device.

2.7 LIBR-11 LIBRARIAN VØØ2A

2.7.1 Programming Notes

1. Creating a library requires an output library specification and the input files to be placed in the library. The input files must be preceded by a comma, so they are not considered to be input libraries. Listing the newly created library is optional.
2. To update a library, both output and input library specifications are required, along with object modules or input files. Updating includes deletion of object modules, insertion of object modules, and replacing of one or more object modules. Listing the updated library is optional.
3. To list the contents of a library only requires a listing file specification and an input library specification. The listing file specification must be preceded by a comma, so that it is not interpreted as an output library.

CHAPTER 3

~~MONITOR DECTAPE SETUP AND UPDATE PROGRAM~~

3.1 INTRODUCTION

The Monitor within the PDP-11 Disk Operating System (DOS) is basically a series of individual program modules, each of which can satisfy some requirement of a user program or of an operator request from the console keyboard. The appropriate module is normally brought into the computer memory only when needed. Otherwise, it resides within a library (MONLIB) on the system disk and for speed of loading it must be stored as a core image.

It is the function of the System Loader (SYSLOD), generally, to prepare the system disk for DOS usage and, in particular, to establish MONLIB in the required form. It accomplishes the latter by converting the Monitor modules from the load format produced by PAL-11R Assembler and Link-11 Linker processes. If the modules are linked to paper tape, they can be fed directly to SYSLOD in a specified order via the PC11 Reader. SYSLOD, however, is also prepared to accept as input a DECtape file, MONLIB.SYS, which contains the load-format modules already strung together in the required sequence.

The Monitor DECtape Setup and Update Program (MODS) provides a means whereby the user can prepare this DECtape file and later maintain it, should existing modules need to be replaced or new ones be added. Moreover, the ROM Bootstrap, stipulated as a basic element of any DOS-supplied configuration to enable automatic Monitor startup from disk, can also be used with DECtape. MODS, therefore, stores SYSLOD itself on the same tape as MONLIB.SYS as a core image readily accessible by the Bootstrap. The basic format of the tape thus produced uses facilities outside those provided by the standard file structure operations of DOS. It cannot, as a result, be copied by the usual means, PIP-11 (although once prepared it can thereafter be used to store other files in the normal way). Instead, MODS can be used to copy the basic system tape content to another tape.

3.1.1 Operating Configuration

The MODS program is designed to operate under DOS. It can be run on any configuration which will support DOS, provided that DECtape is included. For ease of operation, however, a PC11 Reader is also recommended. The ASR-33 Teletype as the sole means of paper tape input is time-consuming and imposes other restrictions which will be discussed later.

3.1.2 Outline of This Chapter

This Chapter is intended to show how MODS may be used. Section 3.2 gives a general description of the program and Section 3.3 details its operating instructions.

It is assumed that the user is already familiar with the general principles of DOS as described in the Programmer's Handbook (DEC-11-MWDA-D).

3.2 GENERAL DESCRIPTION

3.2.1 Command Input

A command string entered through the console keyboard directs the MODS program to operate in one of two modes and at the same time indicates the relevant devices to be used.

a. Setup -- A new DECtape is prepared to contain SYSLOD and MONLIB.SYS by transferring both from the same source, either DECtape or paper tape(s); or by copying MONLIB.SYS from another DECtape and inserting a new SYSLOD from paper tape.

b. Update -- The new DECtape is a copy of SYSLOD from another DECtape and MONLIB.SYS is also a copy but, in this case, possibly modified by new module versions entered from paper tape.

The command strings to accomplish this are shown in detail in section 3.3. In general, since MODS uses the DOS Monitor Command String Interpreter (CSI) to analyze the user's input, they follow the format prescribed for that routine.

MODS checks the validity of the user's specification and, if any error is encountered, an appropriate message is printed at the console teleprinter and the user is requested to enter a new string. MODS also confirms, where SYSLOD is to be copied from another DECtape, that it is, in fact, available on that tape. If not, a Monitor fatal error message results and the user must begin MODS anew.

3.2.2 Basic Preparation

The first step in the preparation of the new DECtape for either mode of operation is to establish a basic file-structure upon it. Basically this requires the writing of appropriate information in Blocks 100-101 (Master File Directory), Blocks 102-103 (User File Directory), and Block 104 (Master Bit Map). In addition, Blocks 70-77 are cleared in readiness for later usage as storage for individual File Bit Maps.

While providing this basic layout, MODS also reserves two contiguous areas on the new tape and enters appropriate identifying file information in the UFD:

a. Blocks 0-37 will be used to store the core image of SYSLOD (covering memory locations 0-36776) and its Loader called by the ROM Bootstrap.

b. Blocks 320-1077 are set up as a dummy file to force the allocation of blocks to MONLIB.SYS to be at the forward end of the tape only.

At the same time, MODS initiates the filling of its internal buffers with any data to be entered from paper tape.

3.2.3 Transfer of SYSLOD

In either mode, SYSLOD can be copied from another DECtape by merely transferring Blocks 0-37 from that tape to the new tape. This is speedily done by moving alternate sets of four blocks in two tape travel directions.

The paper tape SYSLOD is in the load format output by the Link-11 Linker. It must be preceded by a special resident version of DOS Monitor in similar load format. (The DEC Program Library supplies both routines on the same tape.) This format must be appropriately converted into the core image required. Moreover, in order that this image can be accessible via the ROM Bootstrap, an appropriate loader must be provided.

Therefore, MODS first moves into Block 0 the required loader from a copy stored within itself. It then checks that the paper tape being entered does, indeed, contain the special Monitor version and, if so converts the input into its core image starting at Block 1 on the tape and continuing across contiguous blocks, two at a time. When the end of this routine is encountered, MODS expects the entry of SYSLOD itself and checks for this. If satisfied, the similar core image on the tape is prepared with blocks corresponding to any unused intervening core space first being cleared. MODS also establishes an appropriate link between the two routines within the image.

During this entire transfer process, errors arising either because the tapes entered are not those expected or are not in the correct format or through general read failure are reported to the user by an appropriate message at the printer and a new start must be made.

3.2.4 MONLIB.SYS Entry (Setup Mode)

When stored on the DECtape, MONLIB.SYS is a normal linked file within the DOS file-structure scheme. To copy from one DECtape to another is a relatively straightforward operation, provided that the input tape specified by the user does, in fact, contain the requisite copy. Because of the dummy file mentioned in section 3.2.2, the second half of the transfer between the two tapes will occur with the new tape moving in the reverse direction, finishing approximately back at the Directory blocks.

When MONLIB.SYS is merely a string of the Monitor modules in linker format, copying from paper tape is a simple process. In this case, however, the user must ensure that the first module entered is a copy of the normally permanent resident Monitor and the second module is the transient Monitor section TMON linked with a copy of the READ-WRITE processor. The remaining modules may follow in any order. (Tapes supplied by DEC Program Library are numbered to provide the requisite sequence.)

3.2.5 MONLIB.SYS Entry (Update Mode)

The user may be completely unaware of the sequence in which MONLIB.SYS was originally set up on this input DECtape. To allow him to enter his new modules from paper tape in any order, the Update mode of MODS first stores all of them as a series of temporary files on the system disk and maintains its own in-core directory of these for easy access later.

The following points concerning this process should be noted:

a. MODS uses the loader format to differentiate between modules. Therefore, there is no need for the user to enter them individually; if required, they can all be strung together on the same physical paper tape.

b. Should an error occur at this stage, some of these disk files could remain and might then cause file-structure errors if MODS is asked to start over. To prevent this, MODS first deletes any file on the disk using a name it is about to enter. Users should, therefore, avoid setting up their own files using names allocated to Monitor routines as given in the programming manual with a .TMP extension.

c. It follows from b. that if the user enters more than one copy of the same module, the last one seen will always be the one included in the final version of MONLIB.SYS on the new DECtape.

After all the paper tape modules have been handled in this fashion, MODS begins its transfer to the new DECtape. As the start of each module is encountered on the input tape, the in-core directory is searched for a corresponding new version. If none has been supplied, the original is merely copied across. Otherwise, the original is skipped and the new version is used to replace it. The disk file is then immediately deleted, but its in-core reference is retained. Because the skip of the original precedes its replacement from disk and MODS ignores the latter if the required file is no longer present, the immediate deletion has the effect of removing any other copy of the same module which might occur later within the original DECtape file. Thus the new MONLIB.SYS file may be not only an updated version but also a tidier one.

Finally, at the end of the original DECtape input, MODS makes a last search of its in-core directory and adds any completely new modules to the output file.

3.2.6 General Cleanup

After closing the new file, MODS requests deletion of the dummy file at the end of the tape and reinitializes itself for a further run. The new tape itself can be used later to store other system programs using PIP.

3.2.7 General Comment on Paper Tape Usage

The MODS program has been designed to allow the use of either PC11 Reader, if available, or the paper tape reader of ASR-33 Teletype, in that transfers are carried out using .TRAN requests to the Monitor rather than the more customary .READ. This follows from the fact that the ASR-33 is classified as an ASCII-only device and cannot be used to read the binary data of a Linker output module.

However, another problem still exists. All input from the ASR-33 is monitored for the start of a keyboard command from the operator, as indicated by the entry of CTRL/C. Because the computer cannot distinguish between a character coming from the keyboard and one coming from the paper tape reader, recognition of the appropriate ASCII code (3 or 203) within the binary data is treated in this way. As a result, MODS cannot allow operator intervention if its paper tape input comes from the ASR-33.

3.3 OPERATING INSTRUCTIONS

3.3.1 Loading MODS

The new DECtape to be prepared by MODS must be set up under the standard System Identification Code 1,1. The user should, therefore, ensure that he has logged in under this. He may then request MODS loading by a normal RUN command at the keyboard. When entered, MODS identifies itself and prints # in readiness for the input of its command string. Assuming MODS has been stored on the disk, the resulting teleprinter record might be (user input underlined):

```
MONITOR   V002A
$  LO 1,1
DA: dd-mmm-yy
TI: hh:mm:ss
$RU MODS

MODS-11   V003A
#
```

3.3.2 Command Input

~~As indicated in the General Description, section 3.2.1, the input~~ string now entered by the user can be one of several forms, depending upon requirements. Examples are given below:

NOTE

1. [...] Items enclosed in square brackets can be omitted from the string, in which case DTØ: will be the default assumption.
2. {...} Items enclosed in braces show possible alternatives, one of which must be present.
3. n represents the DECtape unit number.
4. Ta:n indicates the decimal number (n) of physical paper tapes to be entered via PC11 Reader and must be supplied.

a. Set up a new DECtape where MONLIB.SYS and SYSLOD are on the same source:

$$\text{[DTn:<]}\left\{\begin{matrix}\text{DTn:}\\ \text{PR:/TA:n}\\ \text{PT:}\end{matrix}\right\}$$

b. Set up a new DECtape with MONLIB.SYS from another DECtape and SYSLOD on paper tape:

$$\text{[DTn:<] DTn:,}\left\{\begin{matrix}\text{PR:/TA:n}\\ \text{PT:}\end{matrix}\right\}$$

c. Update MONLIB.SYS from another DECtape, new modules being entered from paper tape:

$$\text{[DTn:]/U<DTn:,}\left\{\begin{matrix}\text{PR:/TA:n}\\ \text{PT:}\end{matrix}\right\}$$

Upon completion of the requested operation, MODS prints #.

3.3.3 Error Reporting

Errors in the input string or through read failure in the subsequent transfer operations are diagnosed by plain language messages. In all cases, MODS returns to request a completely new input string by printing # following the message.

File structure errors due to the nonexistence of SYSLOD or

MONLIB.SYS on a specified input DECtape result in a fatal diagnostic message from the DOS Monitor. The user may then request a restart by either a Begin or a REstart command.

3.3.4 Operator Action During Processing

Apart from the special treatment required if PT: forms part of the input specified (see section 3.3.5), the only operator actions likely to arise during processing are as follows:

a. A specified DECtape is not ready -- the DECtape driver initiates the printing of an A002 16040 message if the correct units are not made available or the output unit is write-locked. The program is resumed if the omission is rectified and CO) is entered as a keyboard command.

b. At the end of each physical paper tape being entered via the PC11 Reader (until the count given in the input string runs out), MODS prints an A002 63320 message. The next tape should be placed in the reader and CO) be entered.

NOTE

MODS expects the first paper tape to be already set up in the reader when the command input string is issued. If not, the nonexistent tape at this point counts as one of the entries.

3.3.5 Special Action for PT: Usage

As noted in the General Description under Section 3.2.7, usage of PT: poses a special problem. For this version of MODS (V003A) this can only be solved by the user making the following modifications to the resident Monitor before entering the input command string:

```
#↑C                     ;SUSPEND PROGRAM & ENTER MONITOR
.WA                     ;           LISTENING MODE

$ MO 3374               ;CHANGE MONITOR AS REQUIRED

3374/1011     411       ;USER TYPED 411

$ MO 3424
3424/1767     412       ;USER TYPED 412

$CO                     ;RESTART MODS OPERATION
                        ;NOW ENTER COMMAND STRING
```

After typing in the command string as required, the paper tapes can be fed through the ASR-33 reader one by one (with the reader turned on). No computer pauses occur between tapes. Also, the user should avoid striking the keyboard during this period, as all input is now deemed to originate from the reader. It is also advisable that the user take extra precautions to avoid error action necessitated by failure to set up the DECtapes correctly.

When all paper tapes have been input, the user should halt the computer and manually restore the locations changed above. He should then press CONTinue and enter the following command at the keyboard in order to proceed:

```
#↑C
.EN PT:)
```

followed by the HERE IS key on ASR33 or BREAK key on ASR35.

NOTE

Tapes should be removed from the reader while blank trailer tape is still being read.

CHAPTER 4

DOS SOFTWARE MANUAL UPDATE

This chapter contains errata/addenda sheets for updating your PDP-11 Disk Operating System software manuals. Errata sheets should be used in making pen-and-ink corrections to the appropriate manuals; replacement pages should be placed in the appropriate manuals as updated pages.

Listed below are the current PDP-11 DOS software manuals. Also given are the document number and status of each manual, the date of the current update sheet(s), and the version number of the program which the updated manual describes.

Title & Doc. No.	Status[1]	Update[2]	Program Version
Disk Operating System Monitor Programmer's Handbook DEC-11-MWDB-D	3		VØØ4A
FORTRAN Programmer's Manual DEC-11-KFDA-D			VØØ1A
PAL-11R Assembler Programmer's Manual DEC-11-ASDB-D	1	July '71	VØØ5A
Edit-11 Text Editor Programmer's Manual DEC-11-EEDA-D			VØØ2A
ODT-11R Debugging Program Programmer's Manual DEC-11-OODA-D			VØØ2A
PIP File Utility Package Programmer's Manual DEC-11-PIDA-D	1	July '71	VØØ4A
Link-11 Linker & Libr-11 Librarian Programmer's Manual DEC-11-ZLDA-D	1	July '71	VØØ5A & VØØ2A

[1] Status: 1 = Errata/addenda attached
2 = Revised edition in process
3 = New edition (contains previous updates)

[2] Date of applicable updates

NOTE: Additions to previous update sheets and replacement pages made at this time are indicated by a bar in the left margin of the current sheets.

PAL-11R Assembler, Programmer's Manual

July, 1971

DEC-11-ASDB-D

Page

viii ADD: Appendix D Permanent Symbol Table
Appendix E System Error Messages

8-9 Section 8.12, beneath first paragraph. CHANGE:

```
.IFZ        to:      .IFZ  or  .IFEQ
.IFNZ       to:      .IFNZ or  .IFNE
.IFL        to:      .IFL  or  .IFLT
.IFLE
.IFG        to:      .IFG  or  .IFGT
```

9-1 Section 9.2, third line, CHANGE:

```
PAL11R  VØØ2A    to:     PAL11R  VØØ5A
```

10-1 Error Code L, CHANGE TO:

> Line buffer overflow, i.e., input line is greater than 83 characters. Extra characters on a ...

A-3 Octal Codes 136 and 137, ADD in REMARKS columns:

136	(prints as a circumflex on some devices)
137	(prints as an underline on some devices)

B-4
B-5
B-6
B-8

Under sections B.3.1, B.3.2, B.3.3, B.3.4 and B.3.7, remove the parentheses from around the element to which an arrow points. For example

```
         (SE)→DE
    but  (SE)-(DE)
```

B-9 CHANGE mnemonics as indicated below:

```
.IFZ  or  .IFEQ
.IFNZ or  .IFNE
.IFL  or  .IFLT
.IFG  or  .IFGT
```

B-10 Error Code L, CHANGE TO:

> Line buffer overflow, i.e., input line exceeds 83 characters. All characters beyond 72 are ignored.

C-1 Line 18, CHANGE TO:

> the Monitor command RUN while logged in under UIC 1,1. It will produce...

Line 19, CHANGE TO:

> PAL11R.OVR, on the system device[1] with User...

Line 25, CHANGE TO:

> Presence of...on the system device[1]

C-6   DELETE entire page by defacing.

C-7
through REMOVE FROM MANUAL
C-12

Appendix D dated July 1971. INSERT INTO MANUAL.

Appendix E dated July 1971. INSERT INTO MANUAL.

APPENDIX D

PERMANENT SYMBOL TABLE

A listing of the Permanent Symbol Table follows:

```
        .TITLE PALSYM
;THE PERMANENT SYMBOL TABLE (PST)
;
;       VERSION LEVEL 2
;       PATCH LEVEL A
;
;       COPYRIGHT 1971, DIGITAL EQUIPMENT CORP.
;       MAYNARD, MASSACHUSETTS
;
        ;THE PST IS A SEPERATE OBJECT MODULE WHICH
        ;IS LOADED TO RESIDE IMMEDIATELY BELOW THE
        ;ASSEMBLER IN CORE MEMORY.  ONCE ASSEMBLED, IT
        ;IS FIXED IN SIZE; HOWEVER, IT MAY BE
        ;EDITED, ASSEMBLED, AND LINKED TO SUIT THE
        ;SPECIFIC NEEDS OF CUSTOMERS.
        ;THE PST IS BOUNDED BY THE INTERNAL GLOBAL
        ;SYMBOLS SYMTBB AND SYMTBT, WHERE THE
        ;FORMER IS THE HIGHEST ADDRESS OF THE
        ;PST AND THE LATTER IS THE ADDRESS OF
        ;THE 1ST WORD BELOW THE PST.
        ;THE PST IS ORDERED (TO PERMIT BINARY
        ;SEARCHING) WITH THE SMALLEST SYMBOL (IN
        ;MOD40 PACKED NOTATION) HIGH IN CORE AND
        ;THE LARGEST SYMBOL (IN MOD40 PACKED NOTATION)
        ;LOW IN CORE.
        ;MOD40 CHARACTER REPRESENTATION
        ;A THRU Z  : 1 THRU 26 RESPECTIVELY
        ;       $  :  27
        ;       .  :  28
        ;0 THRU 9  : 30 THRU 39 RESPECTIVELY
        ;MOD40 PACKED NOTATION
        ;1ST PACKED TRIAD = CHAR1*40*40+CHAR2*40+CHAR3
        ;2ND PACKED TRIAD = CHAR4*40*40+CHAR5*40+CHAR6
        ;
        ;EACH ENTRY IS 4 WORDS WITH THE LOWEST WORD
        ;CONTAINING THE 1ST PACKED TRIAD, THE NEXT
        ;LOWEST CONTAINING THE 2ND PACKED TRIAD, THE
        ;NEXT LOWEST CONTAINING THE VALUE (WHICH FOR
        ;ASSEMBLER DIRECTIVES IS AN EXTERNAL GLOBAL
        ;TO BE LINKED TO THE APPROPRIATE PROCESSOR IN
        ;PAL-11R) AND THE HIGHEST WORD CONTAINING
        ;THE FLAGS IN THE LOW BYTE AND THE CONTROL
        ;SECTION ID (WHICH IS ALWAYS 0 FOR PST
        ;ENTRIES) IN THE HIGH BYTE.
        ;
        ;INTERNAL GLOBAL SYMBOLS:
                .GLOBL SYMTBB, SYMTBT
        ;
        ;EXTERNAL GLOBAL SYMBOLS:
                .GLOBL  ASCII, ASECT, BYTE, CSECT, END
                .GLOBL  ENDC, EOT, EVEN, GLOBL, IFDF
                .GLOBL  IFG, IFGE, IFL, IFLE, IFNDF
                .GLOBL  IFNZ, IFZ, LIMIT, RAD50
                .GLOBL  TITLE, WORD
        ;
        ;FLAGS:
          ASMDIR=10          ;BIT 3 BEING ON IN THE FLAG
                             ;BYTE INDICATES THAT THIS PST
                             ;ENTRY IS AN ASSEMBLER DIRECTIVE.
          BYTFLG=1           ;BIT 0 BEING ON IN THE FLAG
                             ;BYTE INDICATES THAT THIS PST
                             ;ENTRY IS BYTE ENABLED.  THIS
                             ;ALLOWS ONE ENTRY TO SATISIFY
                             ;SEARCHES FOR WORD AND BYTE
                             ;INSTRUCTIONS.  E.G. THE ENTRY
                             ;'MOV', BECAUSE BIT 0 IS ON,
                             ;WILL SATISFY SEARCHES FOR
                             ;'MOV' OR 'MOVB'
;
;INSTRUCTION CLASS:
;BITS 4-7 OF THE FLAG BYTE DESIGNATE THE
;TYPE OF INSTRUCTION TO PROVIDE DISPATCH
;INFORMATION TO PAL-11R.
SCLAS0=0                     ;OPERATE GROUP
SCLAS1=20                    ;UNARY GROUP
SCLAS2=40                    ;BINARY GROUP
SCLAS3=60                    ;RTS
SCLAS4=100                   ;BRANCH GROUP
SCLAS5=120                   ;JSR
SCLAS6=140                   ;TRAP GROUP
;
;
;
        .EVEN
SYMTBT: .WORD   0                  ;1ST REGISTER BELOW PST.
;
        .WORD   131247             ; .WORD
        .WORD   070440
        .WORD   WORD
        .WORD   ASMDIR
;
        .WORD   131051             ; .TITLE
        .WORD   077345
        .WORD   TITLE
        .WORD   ASMDIR
;
        .WORD   130721             ; .RAD50
        .WORD   017226
        .WORD   RAD50
        .WORD   ASMDIR
;
        .WORD   130351             ; .LIMIT
        .WORD   051274
        .WORD   LIMIT
        .WORD   ASMDIR
;
        .WORD   130156             ; .IFZ
        .WORD   121200
        .WORD   IFZ
        .WORD   ASMDIR
```

```
;
        .WORD   130156          ; .IFNZ
        .WORD   055620
        .WORD   IFNZ
        .WORD   ASMDIR
;
        .RAD50  /.IF/           ; .IFNE
        .RAD50  /NE/
        .WORD   IFNZ
        .WORD   ASMDIR
;
        .WORD   130156          ; .IFNDF
        .WORD   054046
        .WORD   IFNDF
        .WORD   ASMDIR
;
        .RAD50  /.IF/           ; .IFLT
        .RAD50  /LT/
        .WORD   IFL
        .WORD   ASMDIR
;
        .WORD   130156          ; .IFLE
        .WORD   045710
        .WORD   IFLE
        .WORD   ASMDIR
;
        .WORD   130156          ; .IFL
        .WORD   045400
        .WORD   IFL
        .WORD   ASMDIR
;
        .RAD50  /.IF/           ; .IFGT
        .RAD50  /GT/
        .WORD   IFG
        .WORD   ASMDIR
;
        .WORD   130156          ; .IFGE
        .WORD   026210
        .WORD   IFGE
        .WORD   ASMDIR
;
        .WORD   130156          ; .IFG
        .WORD   025700
        .WORD   IFG
        .WORD   ASMDIR
;
        .RAD50  /.IF/           ; .IFEQ
        .RAD50  /EQ/
        .WORD   IFZ
        .WORD   ASMDIR
;
        .WORD   130156          ; .IFDF
        .WORD   014760
        .WORD   IFDF
        .WORD   ASMDIR
;
        .WORD   130044          ; .GLOBL
        .WORD   057034
        .WORD   GLOBL
        .WORD   ASMDIR
;
        .WORD   127736          ; .EVEN
        .WORD   020560
        .WORD   EVEN
        .WORD   ASMDIR



;
        .WORD   127727          ; .EOT
        .WORD   076400
        .WORD   EOT
        .WORD   ASMDIR
;
        .WORD   127726          ; .ENDC
        .WORD   014570
        .WORD   ENDC
        .WORD   ASMDIR
;
        .WORD   127726          ; .END
        .WORD   014400
        .WORD   END
        .WORD   ASMDIR
;
        .WORD   127613          ; .CSECT
        .WORD   017714
        .WORD   CSECT
        .WORD   ASMDIR
;
        .WORD   127551          ; .BYTE
        .WORD   076710
        .WORD   BYTE
        .WORD   ASMDIR
;
        .WORD   127473          ; .ASECT
        .WORD   017714
        .WORD   ASECT
        .WORD   ASMDIR
;
        .WORD   127473          ; .ASCII
        .WORD   012061
        .WORD   ASCII
        .WORD   ASMDIR
;
;
;
        .WORD   107761          ;WAIT
        .WORD   076400
        .WORD   000001
        .WORD   SCLAS0
;
        .WORD   100014          ;TST
        .WORD   0
        .WORD   005700
        .WORD   SCLAS1+BYTFLG
;
        .WORD   077721          ;TRAP
        .WORD   062000
        .WORD   104400
        .WORD   SCLAS6
```

```
;
                .WORD   Ø75131          ;SWAB
                .WORD   ØØ62ØØ
                .WORD   ØØØ3ØØ
                .WORD   SCLAS1
;
                .WORD   Ø75Ø12          ;SUB
                .WORD   Ø
                .WORD   16ØØØØ
                .WORD   SCLAS2
;
                .WORD   Ø73642          ;SEZ
                .WORD   Ø
                .WORD   ØØØ264
                .WORD   SCLASØ
;
                .WORD   Ø73636          ;SEV
                .WORD   Ø
                .WORD   ØØØ262
                .WORD   SCLASØ
;
                .WORD   Ø73626          ;SEN
                .WORD   Ø
                .WORD   ØØØ27Ø
                .WORD   SCLASØ
;
                .WORD   Ø73613          ;SEC
                .WORD   Ø
                .WORD   ØØØ261
                .WORD   SCLASØ
;
                .WORD   Ø73473          ;SCC
                .WORD   Ø
                .WORD   ØØØ277
                .WORD   SCLASØ
;
                .WORD   Ø73423          ;SBC
                .WORD   Ø
                .WORD   ØØ56ØØ
                .WORD   SCLAS1+BYTFLG
;
                .WORD   Ø71663          ;RTS
                .WORD   Ø
                .WORD   ØØØ2ØØ
                .WORD   SCLAS3
;
                .WORD   Ø71651          ;RTI
                .WORD   Ø
                .WORD   ØØØØØ2
                .WORD   SCLASØ
```

```
;
        .WORD   071352          ;ROR
        .WORD   0
        .WORD   006000
        .WORD   SCLAS1+BYTFLG
;
        .WORD   071344          ;ROL
        .WORD   0
        .WORD   006100
        .WORD   SCLAS1+BYTFLG
;
        .WORD   070533          ;RESET
        .WORD   021140
        .WORD   000005
        .WORD   SCLAS0
;
        .WORD   054750          ;NOP
        .WORD   0
        .WORD   000240
        .WORD   SCLAS0
;
        .WORD   054117          ;NEG
        .WORD   0
        .WORD   005400
        .WORD   SCLAS1+BYTFLG
;
        .WORD   051656          ;MOV
        .WORD   0
        .WORD   010000
        .WORD   SCLAS2+BYTFLG
;
        .WORD   040612          ;JSR
        .WORD   0
        .WORD   004000
        .WORD   SCLAS5
;
        .WORD   040230          ;JMP
        .WORD   0
        .WORD   000100
        .WORD   SCLAS1
;
        .WORD   035254          ;IOT
        .WORD   0
        .WORD   000004
        .WORD   SCLAS0
;
        .WORD   035163          ;INC
        .WORD   0
        .WORD   005200
        .WORD   SCLAS1+BYTFLG
```

```
;
        .WORD   031064          ;HALT
        .WORD   076400
        .WORD   0
        .WORD   SCLAS0
;
        .WORD   020534          ;EMT
        .WORD   0
        .WORD   104000
        .WORD   SCLAS6
;
        .WORD   014713          ;DEC
        .WORD   0
        .WORD   005300
        .WORD   SCLAS1+BYTFLG
;
        .WORD   012445          ;COM
        .WORD   0
        .WORD   005100
        .WORD   SCLAS1+BYTFLG
;
        .WORD   012412          ;CNZ
        .WORD   0
        .WORD   000254
        .WORD   SCLAS0
;
        .WORD   012330          ;CMP
        .WORD   0
        .WORD   020000
        .WORD   SCLAS2+BYTFLG
;
        .WORD   012272          ;CLZ
        .WORD   0
        .WORD   000244
        .WORD   SCLAS0
;
        .WORD   012266          ;CLV
        .WORD   0
        .WORD   000242
        .WORD   SCLAS0
;
        .WORD   012262          ;CLR
        .WORD   0
        .WORD   005000
        .WORD   SCLAS1+BYTFLG
;
        .WORD   012256          ;CLN
        .WORD   0
        .WORD   000250
        .WORD   SCLAS0
;
        .WORD   012243          ;CLC
        .WORD   0
        .WORD   000241
        .WORD   SCLAS0
```

```
;
        .WORD   011473          ;CCC
        .WORD   0
        .WORD   000257
        .WORD   SCLAS0
;
        .WORD   010003          ;BVS
        .WORD   0
        .WORD   102400
        .WORD   SCLAS4
;
        .WORD   007763          ;BVC
        .WORD   0
        .WORD   102000
        .WORD   SCLAS4
;
        .WORD   007520          ;BR
        .WORD   0
        .WORD   000400
        .WORD   SCLAS4
;
        .WORD   007414          ;BPL
        .WORD   0
        .WORD   100000
        .WORD   SCLAS4
;
        .WORD   007265          ;BNE
        .WORD   0
        .WORD   001000
        .WORD   SCLAS4
;
        .WORD   007221          ;BMI
        .WORD   0
        .WORD   100400
        .WORD   SCLAS4
;
        .WORD   007164          ;BLT
        .WORD   0
        .WORD   002400
        .WORD   SCLAS4
;
        .WORD   007157          ;BLOS
        .WORD   073300
        .WORD   101400
        .WORD   SCLAS4
;
        .WORD   007157          ;BLO
        .WORD   0
        .WORD   103400
        .WORD   SCLAS4
```

```
;
        .WORD   007145          ;BLE
        .WORD   0
        .WORD   003400
        .WORD   SCLAS4
;
        .WORD   006774          ;BIT
        .WORD   0
        .WORD   030000
        .WORD   SCLAS2+BYTFLG
;
        .WORD   006773          ;BIS
        .WORD   0
        .WORD   050000
        .WORD   SCLAS2+BYTFLG
;
        .WORD   006753          ;BIC
        .WORD   0
        .WORD   040000
        .WORD   SCLAS2+BYTFLG
;
        .WORD   006711          ;BHIS
        .WORD   073300
        .WORD   103000
        .WORD   SCLAS4
;
        .WORD   006711          ;BHI
        .WORD   0
        .WORD   101000
        .WORD   SCLAS4
;
        .WORD   006654          ;BGT
        .WORD   0
        .WORD   003000
        .WORD   SCLAS4
;
        .WORD   006635          ;BGE
        .WORD   0
        .WORD   002000
        .WORD   SCLAS4
;
        .WORD   006531          ;BEQ
        .WORD   0
        .WORD   001400
        .WORD   SCLAS4
```

```
;
        .WORD   006413          ;BCS
        .WORD   0
        .WORD   103400
        .WORD   SCLAS4
;
        .WORD   006373          ;BCC
        .WORD   0
        .WORD   103000
        .WORD   SCLAS4
;
        .WORD   004512          ;ASR
        .WORD   0
        .WORD   006200
        .WORD   SCLAS1+BYTFLG
;
        .WORD   004504          ;ASL
        .WORD   0
        .WORD   006300
        .WORD   SCLAS1+BYTFLG
;
        .WORD   003344          ;ADD
        .WORD   0
        .WORD   060000
        .WORD   SCLAS2
;
        .WORD   003343          ;ADC
        .WORD   0
        .WORD   005500
        .WORD   SCLAS1+BYTFLG
;
;
;
SYMTBB=.-2                      ;HIGHEST ADDRESS OF PST.

        .END
```

APPENDIX E

SYSTEM ERROR MESSAGES

			Error Code
1.	Command String Semantic Errors		
	1.1	Illegal switch Too many switches Illegal switch value Too many switch values	S2Ø3
	1.2	Too many output file specifications	S2Ø4
	1.3	Too many input file specifications	S2Ø5
	1.4	Input file missing	S2Ø6
2.	Limit Errors		
	2.1	More than 376_8 .CSECT's	S2ØØ
	2.2	More than 177_8 nested conditionals	S2Ø1
3.	I/O Errors		
	3.1	Binary or listing file structured device full or input command string too long	S2Ø2
	3.2	Input .TRAN error on Assembler's overlay file	S2Ø7

PIP, File Utility Package July 1971

DEC-11-PIDA-D

Page	Correction
v	Add: 3.7 Determining Free Blocks, /FR 3-9
1-1	Add to list of PIP features: 10. List filenames only. 11. List only certain file(s) or groups of files. 12. List the number of free blocks on device specified.
2-2	Change: DF: to DFn: Disk, fixed head, unit n Add to device list: DCn: Disk, fixed head, unit n CR: Card reader Last two lines on page should read: When a device is not repeated in a command string, the current device is assumed to be the last device specified on the same side (input or output) of the string.
2-3 and 2-4	Replace with the attached page, dated July 1971.
2-5	Add FREE to the list of action switches in section 2.2.1.
3-1	5th line from the bottom should read: copy three files...
3-5 and 3-6	Replace with the attached page dated July 1971.
3-9	Replace with attached page, dated July 1971.
4-1 through 4-4	Replace entire Chapter 4 with the attached pages, 4-1 through 4-10, dated July 1971.
5-1	Add to the list of error messages: S252 Filename given when none allowed. S254 User tried to zero the system resident disk. I354 DK11 disk cartridge was not zeroed (illegal to CONFIRM:).
A-1	Add: /FR Free #DT1:/FR List the number of free blocks remaining on DECtape unit 1.
A-2	Change: DF: to DRn: Disk, fixed head, unit n Add to device list: DCn: Disk, fixed head, unit n CR: Card reader

When no device is specified, the system disk is assumed as the default I/O device (except for /DI, /BR, /FR switches where the keyboard is the default output device). When no unit number is indicated for the device, unit Ø is assumed.

2.1.2 Filename Specification

When required, the filename specification, filnam, consists of from one to six letters or digits, or an asterisk; all characters in excess of six are ignored. The asterisk can be used to represent all file names.

2.1.3 Filename Extension Specification

When required, the filename extension specification, .ext, consists of a period followed by from one to three letters or digits, or an asterisk, which represents all extensions of a named file.

2.1.4 User Identification Code Specification

When required, the user identification code, [uic], consists of a pair of octal numbers separated by a comma and enclosed by square brackets. (The left and right square brackets are typed using SHIFT/K and SHIFT/M respectively.) The left number specifies the user group and the right number specifies the user within the group.

When a UIC does not appear in a command string, the UIC of the user specified with the DOS LOGIN command is assumed.

2.1.5 Switch Specification

When required, the switch specification, /swl:..., consists of a slash followed by one or more letters, and optionally followed by a value specification of octal or decimal digits separated from the switch name by a colon. Switch names can be of any length; however, only the first two characters are ever used. If a switch is unique with one character, only that character needs to be typed.

2.2 SWITCHES

PIP's non-transfer operations are selected by use of switches, and are entered into the command string by preceding the switch with a slash. If more than one switch is used, each is still preceded by a slash.

A summary of PIP's switch options is listed below:

Switch	Name	Operation
/AL	Allocate	Allocate a contiguous file
/BR	Brief Directory	List only filenames and extensions of a directory
/CO	Contiguous	Create a contiguous output file
/DE	Delete	Delete the file
/DI	Directory	List the directory or a subset of the directory (see Chapter 4).
/EN	Enter	Enter the User Identification Code (UIC) in the Master File Directory (MFD)
/FA	Formatted ASCII	Transfer file in formatted ASCII mode
/FB	Formatted Binary	Transfer file in formatted binary mode
/FR	Free	List the number of free blocks left on the device specified
/PR	Protect	Change protection on the file
/RE	Rename	Rename the file
/UN	Unlock	Unlock the User File Directory (UFD) to recover the file
/ZE	Zero	Zero DECtape, magtape, or disk directory

All switches must be typed as the last part of the file specifier, that is, they cannot appear before the filename on which the switch is to operate. Some options require a numeric value to be associated with the switch. For example:

```
/PR
```

is the switch to change the file protection code. The new protection code would follow and be separated from /PR by a colon. For example, a protection change might be written as:

```
#DTØ:ABC/PR:155
```

which would change the protection code of file ABC on DECtape unit Ø from whatever it was before to code 155.

PIP's switch options can be grouped into two categories: action switches and qualifying switches, as explained below.

Unless specified, file transfers assume linked files and use the corresponding DOS functions to perform the action. The /CO switch is ~~used only to create new contiguous output files. The output field of~~ a PIP command should contain the contiguous specification for a merge operation. The /CO switch is ignored on the input side of the command in a file merge operation (although not in a file copy operation).

When /CO is encountered in the output field of a merge operation, the output file specified is written as a contiguous file. For example:

```
    #DK:ABC/CO<DTØ:A,B
```

This command merges the files A and B on DECtape unit Ø to a single contiguous file, ABC, on the disk. (A fatal error occurs if this command is given where the file DK:ABC already exists.)

When transferring files without combining, PIP recognizes the /CO switch associated with any input files and creates the corresponding output file as contiguous. For example, to copy 3 files (currently linked or contiguous) from disk and make them all contiguous on DTØ:, type:

```
    #DTØ:<ABC.DAT/CO,DEF.DAT/CO,GHI.DAT/CO
```

Contiguous files are denoted in the directory listing by a C following the number of blocks for the file.

3.2 RENAMING FILES

The RENAME switch is provided to allow the user a means of changing the name of a file. The general form of the command is:

```
        dev:FILE.1/RE<dev:FILE.2
    or
        dev:FILE.1<dev:FILE.2/RE
```

which changes the name of FILE2 to FILE1.

As before, if no device is specified, the disk is assumed. For example:

```
        #MAIN.OBJ/RE<TESTX.OBJ
```

changes the name of TESTX.OBJ to MAIN.OBJ.

When renaming, it is mandatory that the same device be used on either side of the < symbol. The following is illegal:

```
#DTØ:FILE1<FILE2
```

because the device for FILE2 was not specified, thus the disk is assumed. The operation would not be performed and an error message would be printed on the user's terminal. The command should be written as:

```
#DTØ:FILE1<DTØ:FILE2
```

3.3 DELETING FILES

The DELETE switch is provided to allow the user a means of deleting one or more files. The simple form of the command is:

```
dev:file/DE
```

For example, the command:

```
#DTØ:MAIN.LDA/DE
```

would delete the file MAIN.LDA from DTØ:.

A number of files can be deleted by specifying a sequence of filenames before the DELETE switch. For example, the command:

```
#MAIN1,DT1:MAIN2.OBJ,DT2:MAIN2.PAL/DE
```

would delete all three files.

The DELETE switch supports the asterisk in the filename or extension fields of file specifications. The asterisk is interpreted as "all", as in *.PAL meaning all files with an extension of PAL. To delete all files with an extension of LDA from a DECtape, type:

```
#DT1:*.LDA/DE
```

To delete all files from the disk, type:

```
#*.*/DE
```

To delete all files from a DECtape, it is recommended that the ZERO switch be used, since that operation is much faster.

should now be transferred to another filename, such as:

```
#DF:FILE.NEW<FILE.OLD
```

which copies the old file and guarantees that it is in the proper state. The old file should then be deleted, such as:

```
#DF:FILE.OLD/DE
```

and the latest file can be renamed, if desired, such as:

```
#DF:FILE.NAM/RE<FILE.NEW
```

A bad link can cause erroneous data to be entered in the link words such that the file may be linked into unknown portions of the disk. Thus, it will be impossible to transfer the file onto a back-up device. Do not attempt to delete the bad file -- it would be disastrous to a portion of the disk. Instead, use Edit-11 and search for the end of the good data, write out the entire last buffer, and close the new output file with the EF command. The bad file should then be renamed to some uncommon name and left as it is until the disk is reinitialized.

3.7 DETERMINING FREE BLOCKS, /FR

The free block switch, /FR, can be used to obtain the total number of free blocks remaining on any file-structured device. For example:

```
#DF:/FR
```

causes PIP to print the number of unused blocks on the RF11 Disk. The system device is assumed when no device is specified. For example:

```
#/FR
```

CHAPTER 4

DIRECTORY HANDLING

A significant number of PIP features center around the manipulation of files within directories and of the directories themselves.

4.1 ENTERING USER IDENTIFICATION INTO DIRECTORY

The User Identification Code (UIC) must appear in the Master File Directory (MFD) before a user is able to create his own files on the disk or DECtape. This code is generally entered into the MFD by using the ENTER option in PIP, although the system building program creates a few fundamental User File Directories (UFDs).

The UIC of the current user of the system is passed to DOS by the LOGIN command. For example:

```
.LOGIN 1Ø,11
```

sets the current UIC to [1Ø,11].

The UIC of the current user is obtained from DOS and written into a spare slot in the MFD. If no spare slots exist in the MFD, PIP prints a fatal error message, and no attempt is made to extend the MFD. The recommended sequence is then:

```
.LOGIN uic
$RUN PIP
#DF:/EN
```

This procedure is not always necessary for DECtapes because the ZERO switch enters the UIC of the current user when the directory is initialized.

If a user other than the one originally zeroing the DECtape wishes to access files on the tape, he must enter his UIC on the DECtape or explicitly reference the UIC field during each reference.

4.2 DIRECTORY LISTINGS

A directory of all or certain selective files on any file-structured device can be printed on the teleprinter or line printer; PIP assumes the teleprinter unless LP: is specified in the command string. The information listed in a directory is some subset of all the information in the User File Directory (UFD) entry for each file.

The full (/DI) and brief (/BR) directory switches can be used in various ways to determine the type and amount of information to be listed. The asterisk feature (see Section 4.3) can be used with either directory switch for added flexibility.

4.2.1 Full Directories, /DI

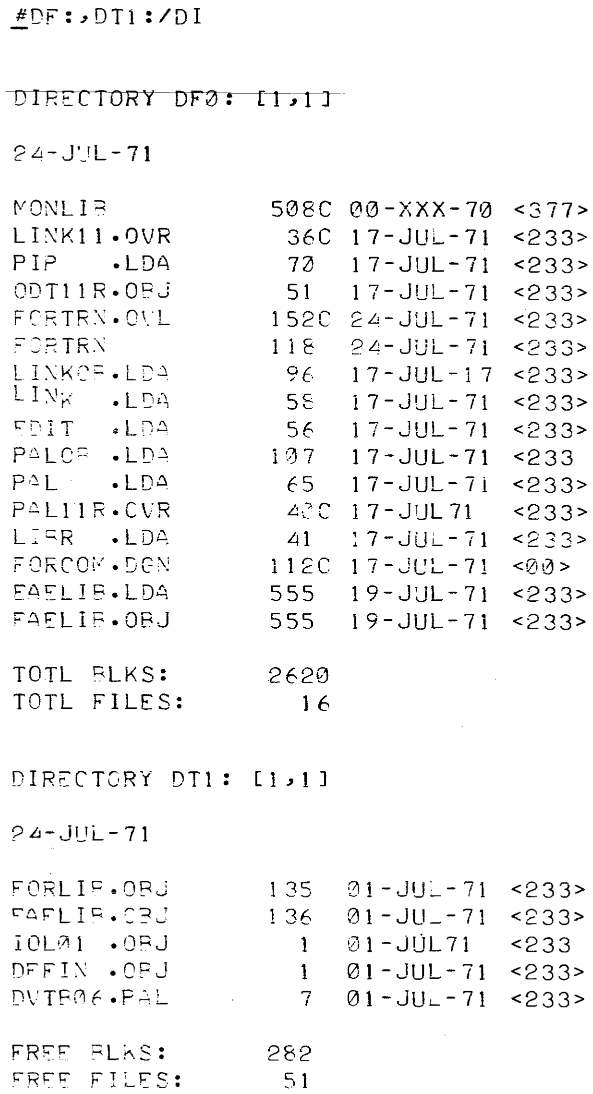
A full directory contains a title line to identify the device and UIC, current (logged in) date, and all filenames with extensions, number of blocks per file (followed by a C if the file is contiguous), file creation date, and file protection code, plus the total number of blocks and files (for that UIC). For example:

```
#/DI

DIRECTORY DFØ: [1,1]

22-JUL-71

MONLIB          508C ØØ-XXX-7Ø  <377>
LINK11.OVR       36C 21-JUL-71  <233>
PIP    .LDA      7Ø   19-MAR-71 <233>
FCL    .PAL      61   21-JUL-71 <233
PAL11R.OVR       4ØC 21-JUL-71  <233>
PAL              65   21-JUL-71 <233>
LINK             59   21-JUL-71 <233>

TOTL BLKS:       839
TOTL FILES:        7
```

Notice that the system device (DFØ: in this case) was assumed and the user was logged in under UIC 1,1.

Whenever a file is transferred onto disk or DECtape, the date in the directory reflects the date on which the transfer took place, not the date on the original file.

The full directories of multiple devices can be obtained:



```
#DF:,DT1:/DI

~~DIRECTORY DFØ: [1,1]~~

24-JUL-71

MONLIB             508C ØØ-XXX-7Ø  <377>
LINK11.OVR          36C 17-JUL-71  <233>
PIP   .LDA          7Ø  17-JUL-71  <233>
ODT11R.OBJ          51  17-JUL-71  <233>
FORTRN.OVL         152C 24-JUL-71  <233>
FORTRN             118  24-JUL-71  <233>
LINKOF.LDA          96  17-JUL-17  <233>
LINK  .LDA          58  17-JUL-71  <233>
EDIT  .LDA          56  17-JUL-71  <233>
PALCR .LDA         1Ø7  17-JUL-71  <233
PAL   .LDA          65  17-JUL-71  <233>
PAL11R.CVR          40C 17-JUL71   <233>
LIBR  .LDA          41  17-JUL-71  <233>
FORCOM.DGN         112C 17-JUL-71  <ØØ>
EAELIB.LDA         555  19-JUL-71  <233>
EAELIB.OBJ         555  19-JUL-71  <233>

TOTL BLKS:         262Ø
TOTL FILES:          16

DIRECTORY DT1: [1,1]

24-JUL-71

FORLIB.OBJ         135  Ø1-JUL-71  <233>
EAELIB.OBJ         136  Ø1-JUL-71  <233>
IOLØ1 .OBJ           1  Ø1-JUL71   <233
DEFIN .OBJ           1  Ø1-JUL-71  <233>
DVTBØ6.PAL           7  Ø1-JUL-71  <233>

FREE BLKS:         282
FREE FILES:         51
```

When a full DECtape directory is obtained, the listing is concluded with the total number of free blocks and files, whereas with a disk the number of used blocks and files are reported.

The full directory listing can be limited to produce a partial list of specified files on one or more devices. The following example illustrates the use of the asterisk to list only those files with the extensions specified on the two devices specified.

```
#DF:*.OVR,*.OBJ,DT1:*.PAL/DI


DFØ:

LINK11.OVR         36C 17-JUL-71  <233>
PAL11R.OVR         4ØC 17-JUL-71  <233>

ODT11R.OBJ         51   17-JUL-71  <233
EAELIB.OBJ        555   19-JUL-71  <233>

DT1:

DVTB06.PAL          7   Ø1-JUL-71  <233>
```

Notice that the UIC, date, and number of blocks and files for that UIC are not reported in partial /DI listings.

Full information about a particular file can be obtained by specifying the filename and extension with the /DI switch. For example:

```
#FTNØØØ.OVL/DI


DKØ:

FTNØØØ.OVL       1ØC 15-JUL-71  <233>
```

This obtains all information about FTNØØØ.OVL if it is in the DKØ: directory. (If the file is not in the directory, there is no listing, but no error message. Likewise, FTNØØØ.OVL/BR allows the user to verify the existence of the file.)

Any UIC directory on any device can be listed by specifying that UIC in the command to PIP. For example, if a user is logged in under UIC 2ØØ,2ØØ, he can give the following command (the results are shown below):

```
#DT1:[1,1]/DI


DIRECTORY DT1:  [1,1]

21-JUL-71

FORCOM.DGN        27C Ø1-JUL-71  <233>
FORTRN.Ø8K        23  Ø1-JUL-71  <233>
OVLØ8K.LDA        27  Ø1-JUL-71  <233>
OVL18K.LDA        26  Ø1-JUL-71  <233>
OVL28K.LDA        23  Ø1-JUL-71  <233>
OVL38K.LDA        26  Ø1-JUL-71  <233>
OVL48K.LDA        28  Ø1-JUL-71  <233>
OVL58K.LDA        28  Ø1-JUL-71  <233>
OVL68K.LDA        25  Ø1-JUL-71  <233>
FORTRN.12K        23  Ø1-JUL-71  <233>
OVLØ  .LDA        27  Ø1-JUL-71  <233>
OVL1  .LDA        26  Ø1-JUL-71  <233>
OVL2  .LDA        23  Ø1-JUL-71  <233>
OVL3  .LDA        26  Ø1-JUL-71  <233>
OVL4  .LDA        28  Ø1-JUL-71  <233>
OVL5  .LDA        28  Ø1-JUL-71  <233>
OVL6  .LDA        25  Ø1-JUL-71-<233>

FREE BLKS:       123
FREE FILES:       39
```

If the user requests information on any particular file or uses the asterisk feature where no such file exists, no error message is given. The directory is considered blank; and a blank line is printed. In the example below there is no *.OVR file on DTØ:

```
#DF:*.OVR,DTØ:*.OVR/DI


DFØ:

LINK11.OVR        36C 21-JUL-71  <233>
PAL11R.OVR        4ØC 21-JUL-71  <233>

DTØ:
```

4.2.2 Brief Directories, /BR

The brief directory switch can be used to list only the device name, filenames, and extensions. The asterisk and specific filenames can be used with the brief directory switch to obtain partial brief directories, as in section 4.2.1. The following PIP example demonstrates the correspondence between /DI and /BR directory listings.

```
#*.OVL/DI


DKØ:

FTNØØØ.OVL       1ØØ 15-JUL-71  <233>
FTNØØ1.OVL        9C 15-JUL-71  <233>
FTNØØ2.OVL        7C 15-JUL-71  <233>
FTNØØ3.OVL        9C 15-JUL-71  <233>
FTNØØ4.OVL       1ØC 15-JUL-71  <233
FTNØØ5.OVL       11C 15-JUL-71  <233>
FTNØØ6.OVL        8C 15-JUL-71  <233>
```

```
#*.OVL/BR


DKØ:

FTNØØØ.OVL
FTNØØ1.OVL
FTNØØ2.OVL
FTNØØ3.OVL
FTNØØ4.OVL
FTNØØ5.OVL
FTNØØ6.OVL
```

An example of the /BR switch with the system device assumed is shown below:

```
#/BR


DF0:

MONLIF
LINK11.OVR
PIP   .LDA
FCL   .PAL
PAL11R.OVR
PAL
LINK
```

Other examples of /BR switch usage are shown below:

```
#FOR.E/BR


DF0:

FOR   .E
```

This is the most efficient way to verify the existence of a particular file on a given device.

```
#DTØ:,DT1:/BR


DTØ:

FORCOM.DGN
FORTRN.Ø8K
OVLØ8K.LDA
OVL18K.LDA
OVL28K.LDA
OVL38K.LDA
OVL48K.LDA
OVL58K.LDA
OVL68K.LDA
FORTRN.12K
OVLØ  .LDA
OVL1  .LDA
OVL2  .LDA
OVL3  .LDA
OVL4  .LDA
OVL5  .LDA
OVL6  .LDA

DT1:

FORLIB.OBJ
F4FLIB.OBJ
IOLØ1 .OBJ
DEFIN .OBJ
DVTBØ6.PAL
```

4.2.3 Free Blocks, /FR

Since a full directory listing of a system disk provides the user only with a summary of the number of blocks used by his current UIC, the /FR switch is available for determining the number of free blocks remaining. For example:

```
#/FR



DFØ:
FREE BLKS:  5513
```

This switch can be used with any device, but is generally only necessary with disk, as the information is not provided automatically with a directory listing. Further examples of the /FR switch follow:

```
#/FR


DKØ:
FREE BLKS:  4113




#DF:,DTØ:/FR


DFØ:
FREE BLKS:  3243

DTØ:
FREE BLKS:   282
```

4.3 ASTERISK IN DIRECTORY LISTINGS

The asterisk (*) feature for filenames and extensions works for directory listings, both normal and brief form. For example, the command

```
#*.PAL/BR
```

gives a brief directory listing for all files on the user's disk area with an extension of PAL. The asterisk can appear in the filename field, the extension field, or both. In addition, groups of file specifiers can be arranged to provide sequential listings, as in

```
#DTØ:*.MAC,DF:FORTN.*/DI
```

4.4 DECTAPE AND RK11 DISK INITIALIZATION

4.4.1 DECtape

In order to initialize a new DECtape with the basic file structure information required by the DOS Monitor, the ZERO switch is provided.

This switch works only on DECtape and performs no action for other devices.

The function creates the permanent bit maps, the file bit maps, the MFD, and a UFD for the user currently running this program.

This switch may be used either by itself, as in:

```
#DTØ:/ZE
```

or in combination with other switches, as in:

```
#DTØ:ONE/ZE<DT1:A,B,C
```

which zeroes DTØ: first, then creates file ONE on DTØ: by concatenating files A, B, and C from DT1:. When used in combination with other switches, ZERO is performed before any other implied actions.

4.4.2 RK11 Disk

The Zero switch can be used to initialize the RK11 disk. As with DECtape, the following four actions occur during initialization, in the order shown below:

1. The entire disk cartridge is zeroed;
2. The first and second MFD blocks are written;
3. The UIC of the current user (and only this UIC) is entered in the MFD; then
4. Bit map blocks are written.

The RK11 disk is initialized by typing:

```
#RKn:/ZE
```

where n is the octal number of the appropriate cartridge (or omitted if RKØ is meant). PIP responds by printing:

```
CONFIRM:
```

to which type either H (for high-density) or L (for low-density) followed by the RETURN key. Should the user decide not to initialize the disk, he types the SPACE bar before typing the RETURN key.

High-density cartridges are marked 22ØØ BPI, and low-density cartridges are marked 11ØØ BPI.

3.5 BUILDING AND RUNNING THE SEGMENTED LINKER

3.5.1 Running the Overlay Editor (LINKOB.LDA)

The overlay builder creates the file of overlays via logical dataset name OVR which has the default file specifier:

```
     sysdev:LINK11.OVR[login uic]<233>
```

where sysdev: represents the system device, i.e., RK:, RF:, or RC:.

It is recommended that the creation of the file of overlays be accomplished either while

1. logged in under [1,1], or
2. with the Monitor ASSIGN command overriding the logged-in UIC. For example:

```
          $AS sysdev:LINK11.OVR[1,1],OVR
```

In either case, the overlay builder can be executed with:

```
          $RUN DTØ:LINKOB.LDA
```

and will create the file of overlays (LINK11.OVR) on disk with a UIC of [1,1]. The overlay builder will then exit to the Monitor, which will print $.

The Linker should then be transferred from DECtape to the Monitor disk area using PIP:

```
          $RUN PIP
          PIP    VØØ2A

          #LINK 1,1<DTØ:LINK11.LDA
```

3.5.2 Running the Linker

The Linker is executed with the Monitor command RUN:

```
          $RUN LINK
```

The Linker will search for its overlays first under the file specifier:

```
          sysdev:LINK11.OVR[login uic]
```

and if not found, then under:

```
          sysdev:LINK11.OVR[1,1]
```

3.5.3 Overlays on DECtape

The DECtape can be used as the overlay device. All that is required to accomplish this is:

1. Make the desired file specifier change prior to running LINKOB.LDA. That is:

   ```
   $AS DTØ:LINK11.OVR[1,1],OVR
   $RUN DTØ:LINKOB.LDA
   ```

2. Make the same file specifier change prior to running LINK. That is:

   ```
   $AS DTØ:LINK11.OVR[1,1],OVR
   $RUN LINK
   ```

Link-11 Linker and Libr-11 Librarian July 1971

Programmer's Manual

~~DEC-11-ZLDA-D~~

Page — Change

1-3 Under section 1.3, make the following additions:

Two Absolute load modules for 8K systems
Two Object modules for relinking
Seven ASCII source tapes

3-1 At top of page, insert:

See section 3.5 for instructions on assembling, building and running the Linker.

3-1 and 3-8 Change version number as shown in terminal output to:

```
LINK-11    VØØ5A
```

3-9 and 3-10 Append insert page included.

HOW TO OBTAIN SOFTWARE INFORMATION

Announcements for new and revised software, as well as programming notes, software problems, and documentation corrections are published by Software Information Service in the following newsletters.

Digital Software News for the PDP-8 & PDP-12
Digital Software News for the PDP-11
Digital Software News for the PDP-9/15 Family

These newsletters contain information applicable to software available from Digital's Program Library, Articles in Digital Software News update the cumulative Software Performance Summary which is contained in each basic kit of system software for new computers. To assure that the monthly Digital Software News is sent to the appropriate software contact at your installation, please check with the Software Specialist or Sales Engineer at your nearest Digital office.

Questions or problems concerning Digital's Software should be reported to the Software Specialist. In cases where no Software Specialist is available, please send a Software Performance Report form with details of the problem to:

Software Information Service
Digital Equipment Corporation
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754

These forms which are provided in the software kit should be fully filled out and accompanied by teletype output as well as listings or tapes of the user program to facilitate a complete investigation. An answer will be sent to the individual and appropriate topics of general interest will be printed in the newsletter.

Orders for new and revised software and manuals, additional Software Performance Report forms, and software price lists should be directed to the nearest Digital Field office or representative. U.S.A. customers may order directly from the Program Library in Maynard. When ordering, include the code number and a brief description of the software requested.

Digital Equipment Computer Users Society (DECUS) maintains a user library and publishes a catalog of programs as well as the DECUSCOPE magazine for its members and non-members who request it. For further information please write to:

DECUS
Digital Equipment Corporation
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754

PDP-11 Getting DOS On The Air
DEC-11-SYDC-D
August 1971

READER'S COMMENTS

Digital Equipment Corporation maintains a continuous effort to improve the quality and usefulness of its publications. To do this effectively we need user feedback -- your critical evaluation of this manual.

Please comment on this manual's completeness, accuracy, organization, usability and readability.

____

____

____

____

Did you find errors in this manual? If so, specify by page.

____

____

____

____

____

How can this manual be improved?

____

____

____

____

____

Other comments?

____

____

____

____

____

Please state your position. ______________________ Date: __________

Name: ______________________ Organization: ______________________

Street: ______________________ Department: ______________________

City: ______________ State: ______________ Zip or Country __________

- - - - - - - - - - - - - - - - - - - Fold Here -

- - - - - - - - - - - - - - Do Not Tear - Fold Here and Staple - - - - - - - - - - - - - -

digital

Digital Equipment Corporation
Software Information Services
146 Main Street, Bldg. 3-5
Maynard, Massachusetts 01754