PDP-8 SYSTEMS
MODULE III
OS/8 – DEC/X8

Student Guide

digital™

Student Guide

List of Contents

SECTION 1

OS/8 OPERATING
SYSTEM

OS/8 INTRODUCTION

The OS/8 Operating System is a sophisticated operating system designed for the PDP-8 family of computers. This system permits use of a wide range of peripherals and all available core up to 32K. OS/8 offers a versatile keyboard monitor that supervises a comprehensive library of system programs. These features make OS/8 a significant improvement in small computer operating systems.

OS/8 OVERVIEW

HARDWARE CONFIGURATIONS

The OS/8 system can operate with the following devices as the system device.

TC01/TC08 DECtape
LINCtape (PDP-12)
TD8E DECtape
DF32/RF08 disk
RK8E disk
RK8 disk
RX01 diskette

The term system device refers to the device on which the OS/8 system resides and which it utilizes for system functions. Thus, DECtape unit 0 is the system device for a DECtape-based system. A nonsystem device is any peripheral not specifically used for system functions, such as LPT:, PTR:, DTA2:, etc.

A typical medium-sized system might contain a PDP-8/E with at least 8K words of core memory, TD8E DECtape and control, and an RK8E disk pack and control. A disk system offers the additional convenience of easy and fast access to files and large amounts of storage.

Up to 15 of the following devices can be included in a single OS/8 system:

o   As many as 8 DECtape units (TC01/TU55), TC08/TU56. or TD8E/TU56)

o   TA8E/TU60 cassette units

o   Tm8E/TU10 magnetic tape units

o   High-speed paper tape reader/punch

o   High-speed paper tape reader/punch

o   Up to four RK8E disks

o   Up to four RK8 disks

o   Up to four RS08 disks

o   Up to four DF32 disks

o   Card reader (optical mark or punched cards)

o   Line printer

o   PDP-12 LINCtape

OS/8 OVERVIEW

o   PDP-12 scope

o   Any other device for which it is possible to write a
        device handler in one or two pages of core

SYSTEM SOFTWARE COMPONENTS

The main software components of the OS/8 system include

        Keyboard Monitor
        The Concise Command Language
        Command Decoder   •
        Library of system programs
        Device handlers
        User Service Routine (USR)

The Keyboard Monitor provides communication between you and the
OS/8 executive routines by accepting commands from the console
terminal.   The commands enable you to create logical names for
devices, run system and user programs, and save programs.

The Concise Command Language (CCL) provides an extended set of
Monitor Commands.

The Command Decoder allows you to communicate with a system
library program by accepting a command string from the keyboard
indicating input/output files. Following your keyboard command to
run a system library program, the Command Decoder prints an
asterisk and then accepts the command line containing device and
file specifications.

The library of system programs allow the user to create, load, and
run programs.   Extensive file maintenance programs are also
included.

Device handlers are subroutines designed to transfer data to and
from peripheral devices.   OS/8 is able to interface with as many
as 15 different peripherals at a time.   During system generation,
device handlers become an integral part of the system; both system
and user programs have access to any available device. (The BUILD
program allows quick and easy alteration of any available device.)

The User Service Routine (USR) controls the directory operations
for the OS/8 system.   A program can use the USR by means of
standard subroutine calls such as those used to activate device
handler subroutines.   Some of the functions performed by the USR
are loading device handlers, searching file directories, creating
and closing output files, calling the Command Decoder, and
chaining of programs.

OS/8 OVERVIEW

When OS/8 is operating, the Command Decoder, Keyboard Monitor, and USER are swapped into core from the system device as required. When their operation has been completed, the previous contents of core are restored.

The memory-resident portion of OS/8 is extremely small (256 words), allowing for a maximum use of memory by user programs.

OS/8 SYSTEM CONVENTIONS

OS/8 observes the following conventions in the names of devices, files, and units of storage.

PERMANENT DEVICE NAMES

During configuration, the OS/8 BUILD program assigns permanent names to the devices in a system. You can change these names by reconfiguring the system, but you must keep in mind that some CCL commands and system programs operate on the assumption that certain names are present. The DIRECT command, for example, uses the name TTY: as a default device for listings, and the CREF program assumes LPT: as a default output device. Therefore, it is good practice to keep the folowing names always present on the system.

    SYS:
    DSK:
    TTY:
    LPT:

TABLE 1-1 lists all the device names used by OS/8.

FILE NAMES AND EXTENSIONS

File names may contain up to six alphanumeric characters, followed optionally by a period and an extension of two alphanumeric characters. The extension usually identifies the file by type. For example, a .PA extension after a file name indicates that the file contains a PAL8 source program.

In most cases, you will want to conform to the standard extensions established for OS/8. If you omit the extension on an output file specification, some system programs append assumed extensions. PAL8, for example, will add .PA to an output file. If you specify a file for input and omit the extension, some system programs will look for a file with an assumed extension. For example, if you specify a program called PUMP as input to PAL8, PAL8 looks for PUMP.PA. If it fails to find it, it looks for the file name and no extension.

TABLE 1-2 lists the file extensions used by OS/8.

OS/8 SYSTEM CONVENTIONS

TABLE 1-1

Permanent Device Names

Permanent Name	I/O Device
SYS	System device (disk if the system has a large disk -- RK8 or RFØ8; otherwise DTAØ)
DTAn	DECtape n, where n is an integer in the range Ø to 7, inclusive
LTAn	When using BUILD, LINCtapes may be called LTA rather than DTA. n is an integrer in the range Ø to 7 inclusive.
DSK	Default storage device for all files.
TTY	Terminal keyboard and printer
PTP	Paper tape punch
PTR	Paper tape reader
CDR	Card reader
LPT	Line printer
CSAn	Cassette drive n, where n is an integer in the range Ø to 7, inclusive
MTAn	Magnetic tape drive n, where n is an integer in the range Ø to 7 inclusive
DF	DF32 disk
RF	RFØ8 disk
RKAn	RKØ1 or RKØ5 disk unit n, where n is an integer in the range Ø to 3
TV	VR12 scope (PDP-12 only)
BAT	Pseudo device which reads from BATCH input stream (see BATCH section in Chapter 2)
RXAn	Diskette n (floppy), where n is an integer in the range of Ø-7 inclusive
RKBn	DECpack n, where n is an integer in the range Ø-1

OS/8 SYSTEM CONVENTIONS

TABLE 1-2
OS/8 File Name Extensions

.BA	BASIC source file
.BI	Batch input file
.BK	Backup ASCII file
.BN	Absolute binary file
.DA	Data file
.DC	Documentation file
.DI	Directory listing
.FT	FORTRAN language source file
.HL	Help file
.LD	F4 load mode
.LS	Assembly listing output
.MA	Macro source file
.MP	File containing a loading map
.PA	PAL8 source file
.RA	RALF assembly language file
.RB	Relocatable binary source file
.RL	Relocatable binary file
.SB	8K SABR source file
.SV	Core image file or SAVE file; appended to a file name by the R, RUN, SAVE, and GET Keyboard Monitor commands
.SY	System head
.TE	TECO macro file
.TM	Temporary file generated by FORTRAN or SABR
.TX	Text files

OS/8 KEYBOARD COMMANDS

The OS/8 Monitor and the Concise Command Language program --
CCL.SV -- provide you with more than fifty different keyboard
commands.

o   The Monitor provides the following commands, which you
        may abbreviate to the first two letters.

                ASSIGN
                DEASSIGN
                GET
                SAVE
                ODT
                RUN
                R
                START
                DATE

        The Monitor displays a dot to indicate that it is ready
        to accept a command.

        To execute a command that you have typed, press RETURN or
        ALTMODE.

        Any error that you make in the use of these commands
        causes the Monitor to print an error message, display the
        dot, and wait for you to try again.

o   The Concise Command Language program (CCL) provides an
        extended set of Monitor commands. Some of these commands
        allow you to call a system program indirectly, perform an
        operation, and return automatically to the Monitor. This
        method is simpler to use than the standard calling
        sequence for a program. For example, the following
        two-line sequence causes PAL8 to assemble a source
        program called SCOOP-PA and send a binary and listing
        file to DSK, the default device.

                .R PAL8
                *SCOOP.BN,SCOOP.LS<SCOOP.PA

        You can obtain the same results faster by using the CCL
        PAL command with the -LS option.

                .PAL SCOOP.PA-LS

        Other CCL commands perform special functions not
        available through OS/8 utility programs.

        You can write your own CCL commands and add them to the
        CCL program. For instructions, see the OS/8 Software
        Support Manual.

OS/8 KEYBOARD COMMANDS

You enter a CCL command the same say you enter a Keyboard Monitor
Command -- in response to the terminal dot. Normally, you
terminate the command line with the RETURN key. Depending on the
characteristics of the command you are using, control may return
to the Monitor when the operation is completed or may remain
within another OS/8 program. To remain under program control when
control would normally return to you, terminate the CCL command
with an ALTMODE. (Note that this termination procedure is the
opposite of the way most OS/8 programs work.)

CCL provides the following commands, which you may abbreviate to
the letters printed as capitals:

BACKspace	DUplicate	PRInt	UC
BAsic	EDIT	PUNch	UNLoad
BOot	DOF	REName	VERsion
CCL	EXEcute	RES	ZERO
COMPare	HELp	REWind	
COMPile	LIst	SET	
COPy	LOad	SKIP	
CREate	MAKe	SQuish	
CREF	MAP	SUbmit	
DAte	MEMory	TEco	
DEassign	MUNG	TYpe	
DELete	ODT	UA	
DIRect	PAL	UB	

A special CCL command -- called CCL - deactivates the entire
Concise Command Language Program and all the commands that run
under it. To reactivate the program, you must run it with the R
command.

EXAMPLE

.CCL   - DEACTIVATES THE CCL PROGRAM
.
.R CCL   - ACTIVATES THE CCL PROGRAM
.

COMMAND FORMAT

The general format of the command line is
    command output:file<input:file/option
where
    command   is a legal OS/8 command

    output: is the name of the device you specify to receive
               output

    file   is the name and extension of an output file

OS/8 KEYBOARD COMMANDS

    input: is the name of the device you specify for input

    file   is the name and extension of an input file

    /option   is a command qualifier

Some commands permit multiple file and device specifications; refer to the descriptions of the commands you want to use for details.

OS/8 command options let you choose the way you want to execute a command. File options let you optimize the storage on an output device.

EXAMPLE

.ZERO RXA1:/Y

This command clears the directory of RXA1 and then copies a bootable system to RCA1 (due to the "Y" option).

WARNING:

> Use ZERO only on devices that contain user programs and data files. If you zero the system device, you will destroy the system programs. ZERO will not clear the directory of SYS until it has printed a message to ask if you are sure you want to proceed. If you do, type Y; if you do not, type any other character.

USING WILDCARDS

Wildcards, which certain OS/8 commands accept, make it possible for you to refer to a group of related files with a single file specification. OS/8 provides two wildcards:

*   The asterisk (*), which replaces an entire filename or extension

o   the question mark (?), which replaces any single character

OS/8 KEYBOARD COMMANDS

The following commands permit both the asterisk and question mark
as wildcards in input specifications.

    COPY
    DELETE
    DIRECT
    LIST
    RENAME
    TYPE
Here are some examples of the various ways you can abbreviate
input specifications with wildcards.

    .DEL TEST1.*   deletes all files on DSK with the name TEST1
                      and any extension

    .DIR *.BN   displays a directory of all files on DSK with
                      a .BN extension and any name

    .DIR TES??.*   displays a directory of all files with names
                      beginning TES and any extension

    .LIST ???.??   lists the contents of all DSK files with names
                      of three characters or less

You may use the asterisk wildcard in an output file name.   The
question mark, however, is illegal.   If you omit the output file
name altogether, the system assumes *.* -- that is, all files with
any extension.

For example, this command

    .COPY RXA1:*.BK,SYS:*.PA

copies all files from SYS with a PA extension to RXA1, adding the
extension BK.

                        NOTE:

        A filename may not contain embedded
        asterisks.   For example, TE*T.* is an
        illegal specification and will produce
        the following error message:

        ILLEGAL *

OS/8 KEYBOARD COMMANDS

WARNING

Use wildcards in COPY and DELETE commands with extreme caution to avoid destroying irreplacable files. Always observe the following fail-safe measures.

    o   Keep a backup copy of the system diskette and all other important files.

    o   Use the Q option with COPY and DELETE. The system pauses to make sure you have specified the file you intended. If you wish to go through with the operation, type Y in response to the query. If not, type any other character.

EXAMPLE

.COPY RXA1:*.*<RXA0:*.*/Q

FILES COPIED:

SET.SV?Y	(TYPE "Y" TO COPY FILE)
RESORC.SV?N	(TYPE "ANY CHAR" for NO)

OS/8 KEYBOARD COMMANDS

BOOT

The BOOT command makes it possible for you to bootstrap onto another device or onto another PDP 8 system. The format is:

    .BOOT/DV
    .BO /DV
    .R BOOT/DV

Where "DV" is a mnemonic that specifies the device to boot from.

If you type BOOT with no argument, boot prints a slash (/) to indicate that you must enter a mnemonic.

For example, this command

    .BOOT/RX

Bootstraps the RX8E floppy disk.

If you wish to halt before doing the bootstrap, type the command, a mnemonic, and a period. For example:

    .BOOT/DK.

The period causes the computer to halt, giving you time to mount a new device. To continue the operation, press the continue switch on the console. This form of the command is useful when only one disk or dectape drive exists on the system.

OS/8 Boot Mnemonics are as follows:

    CA   TA8E Cassette
    DK   Default System Disk
    DL   Link Tape
    DM   RFØ8 or DF32
    DT   Default System Tape
    LT   Link Tape
    PT   Paper Tape Reader
    RE   RK8E Disk
    RF   RFØ8 or DF32
    RK   RK8 Disk
    RX   RX8E Floppy Disk
    TC   TD8E DEC Tape
    TY   TCØ8 Unit 4 (Type Set Only)
    VE   Prints Current Version of BOOT.SV
    ZE   Zeros out Field Ø

CCL

The "CCL" command disables the concise command language program on

OS/8 KEYBOARD COMMANDS

the system device. The format is:

    .CCL

The command accepts no arguments

The "CCL" command totally deactivates the CCL feature of OS/8 so
that the system will not accept any CCL command.

If you wish to use the CCL feature again, you must reactivate it
with the R command. To do this, type

    .R CCL

COPY

The COPY command transfers files from one device to another. The
format is:

    .COPY OUTPUT:FILE<INPUT:FILE

COPY is a CCL command that runs the FOTP program. For complete
information on file transfer, reference the FOTP program in this
module.

DELETE

The DELETE command removes files from the directory of the device
you specify. The format is:

    .DELETE INPUT:FILE

Enter the file that you want to delete in the command line. You
may specify only one device in command line. Example:

    .DELETE RXA1:TEST.SV

You may use the wildcard asterisk (*) to specify file names and
extensions and the question mark (?) to indicate single
charactors. Wild cards enable you to remove an entire group of
related files with a single delete command. For example:

    .DELETE *.SV

Removes all files with a SV extension from the system device.
Use wildcards with extreme caution to avoid deleting irreplacable
files.

OS/8 KEYBOARD COMMANDS

DIRECT

The DIRECT command produces listings of OS/8 device directories.
The format is:

    .DIRECT OUTPUT:FILE<INPUT:FILE

Direct prints a directory of all the files on all the devices that
you specify in the command line. Direct makes the following
assumptions about input and output specifications:

-    If you omit the input device, direct assumes "DSK".

-    If you omit input file names, direct assumes *.*, that
     is, all files with any extension.

-    If you omit the output device, direct prints the
     directory on the terminal.

-    Direct automatically adds a "DI" extension on an output
     file.

For example, this command

    .DIR RXAØ:DILIST<RXAØ:

Creates the file DILIST.DI on RXAØ. This file can be printed on
any useable output device. For example:

    .R PIP
    *LQP:,RXAØ:DILIST.DI

These commands will print the directory file on the LQP printer.

OPTIONS
    /B   Include starting block numbers
    /C   List only files with current date
    /E   Include empty files
    /F   Fast mode
    /M   List empties only
    /O   List only files with other than todays date
    /R   List remainder of files after first one.
    /U   Treat each input specification separately
    /V   List files not of form specified
    /W   Give version number
    =N   Use N columns

OS/8 KEYBOARD COMMANDS

GET

The GET command loads a memory-image file (*.SV) into memory. The
format is:

    GET INPUT:FILE.SV

If you omit the extension, get looks for a file with the name you
specify and an .SV extension. You must specify the device; get
does not assume DSK.

For example, to load into memory a file called TTYTST.SV on RXA0,
Type

    .GET RXA0:TTYTST

During execution, get loads the file and its core control block
into memory, then transfers the CCB to a special area on the
system device for reference and maintenance. Get also places the
job status word into location 7746 of field 0 to indicate what
parts of memory the file uses and how. It loads the block number
of the first block of the file into location 7747.

To run a program that you have loaded into memory with GET, use
the start of execute command.

MEMORY

The MEMORY command finds the highest field available in hardware
or limits the fields available in software.

The format is

    MEMORY

or

    MEMORY n

where

    n   is an octal number from 0 to 7 representing the number of
          fields (each containing 4K words of memory) in software.

For example, this command line

    .MEMORY 3

limits the amount of memory available in the system to 16K words.

OS/8 KEYBOARD COMMANDS

The following list shows all the values of n and their meaning:

    Ø   all available memory fields
    1   8K words of memory
    2   12K words
    3   16K words
    4   2ØK words
    5   24K words
    6   28K words
    7   32K words

To find the amount of memory currently being used by OS/8, type
the command with no argument. The following output indicates that
a MEMORY 4 command, entered previously, has restricted a 32K
system to only 2ØK words of available memory.

    .MEMORY
    2ØK/32K MEMORY

If the system is using all available memory, the Monitor prints
the total amount. For example:

    .MEMORY
    32K MEMORY

The MEMORY command causes the execution of CCL.SV.

ODT

The ODT command enables you to debug the program currently in
memory, control its execution, and make alterations by typing ODT
instructions at the terminal.

The format is

    ODT

Once you have entered the command with the RETURN key, you may
examine and modify any memory location of the program currently in
memory or use the breakpoint feature to control program execution.

When using ODT to debug a program, you must call I/O devices by
their permanent names. As long as ODT is in control of the
system, all user-defined names are invalid.

OS/8 KEYBOARD COMMANDS

R

The R command loads and starts a memory-image file from the system device. The format is

    R file.SV

R writes the block number of the first block in the file in location 7747 in field 0.

Since the R command loads files from the system device only, you may not specify an input device other then DSK in the command line. If you omit the file extension, R assumes SV.

For example, this command

    .R TEST

looks for a program called TEST.SV on the system device and loads and executes it.

The R command differs from the RUN command in that it does not send the Core Control Block to the system device. To save a program that does not have its Core Control Block in the usual place on SYS, you must include all the optional arguments in the SAVE command.

RUN

The RUN command loads a memory image (SV) file into memory, transfers its Core Control Block to the system device, and begins execution at the starting address of the program. It places the block number of the first block in the file into location 7747 of field 0.

The format is

    RUN input:file

If you enter a file name without an extension, RUN assumes SV. You must specify a device; RUN does not assume DSK.

For example, the following RUN command GETs and STARTs PROG.SV on RXA1.

    .RUN RXA2:PROG.SV

OS/8 KEYBOARD COMMANDS

SAVE

The SAVE command makes an executable binary file of the program currently in memory, assigns it a name, and stores it on a device. If you do not specify the locations in memory that you want to save, the SAVE command automatically looks for the information on the current Core Control Block.

The format is

    SAVE device:file fnnnn-fmmmm,fpppp;fssss=cccc

where

fnnn	is a 5-digit octal number representing the field (f) and starting address of a continuous portion of memory that you want to save
fmmmm	is the final address (in the same field) of that part of memory you intend to save
fpppp	is a 5-digit octal number representing the address of one location in memory. A single address causes SAVE to save the entire page on which the location occurs
;fssss	is a 5-digit octal number representing the starting address of the program you want to save
=cccc	is a 4-digit octal number representing the contents of the Job Status Word

If you omit the extension of the file name, SAVE appends SV. If you omit the other arguments, SAVE finds the locations it requires in the current Core Control Block.

START

The START command begins execution of the memory image program currently in memory at the address you specify in the command line. If you omit the address, START uses the starting location in the current Core Control Block.

OS/8 KEYBOARD COMMANDS

The format is

    START fnnnn

where

    fnnnn   is a 5-digit octal number representing a field
                     (f) and the location in memory (nnnn) you want
                     to use as a starting address

For example, this command

    .START 10555

starts executing the program currently in memory at location 555
in field 1.

This command

    .START

starts the program at the address contained in the current Core
Control Block.

The Monitor runs the START command.

ZERO

The ZERO command clears the directory of the device you specify,
creating an empty file directory. The format is

    ZERO device

For example, the following example clears the directory of RXA1.

    .ZERO RXA1:

Use ZERO only on devices that contain user programs and data
files. If you zero the system device, you will destroy the system
programs. ZERO will not clear the directory of SYS until it has
printed a message to ask if you are sure you want to proceed. If
you do, type Y; if you do not, type any other character.

OS/8 SYSTEM PROGRAMS

Now that you have seen the OS/8 Keyboard Commands and examples of how to use them, we will discuss some of the OS/8 system programs that will assist you in

- COPYING

- UPDATING

- MAINTAINING

DIAGNOSTIC MEDIA.

The system utility programs that we will discuss in this module are:

FOTP   File-Oriented Transfer Program

PIP   Peripheral Interchange Program

RXCOPY   RX Floppy Disk Program

FOTP

FOTP transfers files from one device to another, deletes files from a device, and renames files.

FOTP

FOTP copies files in the image mode, that is, it copies the file word for word, character for character, without making any changes in the file. Thus you may use FOTP to copy core image and binary files as well as ASCII files, without specifying options to identify the type of file.

CALLING FOTP

To call FOTP from the system device, type:

    .R FOTP

In response to the keyboard monitor DOT. (You can also call FOTP indirectly with several CCL commands.) The command decoder prints a asterisk (*) in the left margin and waits to receive a line of I/O files and options. FOTP accepts one output specification and up to five input specifications. The command line may be terminated with a carriage return (FOTP retains control) or with an escape (control returns to the keyboard monitor).

OS/8 SYSTEM PROGRAMS

INPUT SPECIFICATIONS

FOTP input specifications consist of a device, a file name, and a file name extension.

Within the input specification, FOTP allows you to use a wild card construction. This means that the file name or the extension may be replaced totally with an asterisk or partially with a question mark to designate certain file names or extensions. For example:

    TEST1.*   All files with the name TEST1 and any extension

    *.BN   All files with a BN extension and any file name

    .   All files

You can include as many as five input specifications in a single command line. If all the files are on the same device, the input device need be specified only once. For example:

    .R FOTP
    RXA1:.*<RXA1:*.BN,*.SV,*.RL

Refers to files on RXA1 that have .BN, .SV, or .RL extensions with any file name.

OUTPUT SPECIFICATIONS

FOTP output specifications consist of a device, a file name, and a file extension. Output specifications are optional. You can use the wild card asterisk in output specifications, but use of the question mark is illegal.

If no output device is specified but a file name is given, then "DSK:" is assumed. If no file name is specified, then *.* is assumed. Thus the following output specifications

    *SWING<RXA1:SWING.SV

    SWING.<RXA1:SWING.SV

    *RXA0:<RXA1:SWING.SV

ARE EQUIVALENT:

    DSK:SWING

    DSK:SWING.*

    RXA0:*.*

OS/8 SYSTEM PROGRAMS

USING FOTP

FOTP performs file transfers in a different manner than other OS/*
transfer programs, the following is a detailed description of the
way FOTP works. One of the main uses of FOTP is to copy files
from one device to another. The following examples show how FOTP
examines each aspect of a command to determine what operation will
take place.

EXAMPLE 1:

To copy the file TTYTST.SV from RXA1 to RKAØ, changing its name to
OUTTST, type the following:

> .R FOTP
> *RKAØ:OTTST.SV<RXA1:TTYTST.SV

1.   If FOTP does not find the file TTYTST.SV on RXA1, the
      following message appears and no transfer occurs:

      *No files of the form TTYTST.SV

2.   FOTP examines RKAØ to determine whether it already
      contains a file OUTTST.SV. If OUTTST.SV exists on RKAØ,
      FOTP deletes it before beginning the transfer. This
      process is called "PREDELETION".

3.   The /N option specifies that no predeletion is desired.
      Thus the command:

      *RKAØ:OUTTST.SV<RXA1:TTYTST.SV/N

      Begins to copy TTYTST.SV to RKAØ wiwthout deleting the
      old OUTTST.SV. FOTP does this by opening a tentative
      file names OUTTST.SV on RKAØ. When the command completes
      the transfer operation, it closes the tentative file.
      Closing this tentative file makes it a permanent file and
      deletes any old files of the same name.

      This process is called "POSTDELETION".

4.   FOTP assigns thhe creation date of TTYTST.SV to
      OUTTST.SV. This is an advantage over PIP, which would
      assign the current date to the new file. If you always
      transfer files with FOTP, you preserve the original
      creation date of the file.

OS/8 SYSTEM PROGRAMS

5. Use the /T option of FOTP to assign the current date to a
    file. For example:

    *RXAØ:OUTTST.SV<RXA1:TTYTST.SV/T

EXAMPLE 2

To copy all files from RKAØ to RKA1, type:

    RKA1:.*<RKAØ:*.*

Note that the *.* specification has a different meaning when it
appears on the left side of the < than is does when it appears on
the right. When used on the output (left) side, *.* means that
the output file name is the same as the input file name. When
used on the input (right) side, *.* means transfer or consider all
files on this device.

EXAMPLE 3

One feature of FOTP allows you to use the same command line to
transfer multiple files from one device to another. For example,
to transfer five binary files from DTA2 to DTA1, type:

    *DTA1:<DTA2:TEST1.BN,TEST2.BN,TEST3.BN,TEST4.BN,TEST5.BN

PIP would have required five commands to perform the same
operation.

PIP PERIPHERAL INTERCHANGE PROGRAM

CALLING AND USING PIP

TO CALL PIP FROM THE SYSTEM DEVICE, TYPE:

    .R PIP

The command decoder then prints an asterisk in the left margin of
the console and waits to receive a line of I/O files and options.

Since PIP performs file transfers for all file types (ASCII, Image
or Binary), there are no assumed extensions PIP assigns to file
names for either input or output files. You must specify all
extensions.

OS/8 SYSTEM PROGRAMS

Following completion of a PIP operation, the command decoder again
prints an asterisk in the left margin and waits for another PIP
I/O specification line. You can return to the keyboard monitor by
typing CTRL/C or by terminating the specification line with the
escape (ALTMODE) key.

PIP OPTIONS

Table 1-3 details the options allowed on a PIP I/O specification
line. Generally, you indicate /A, /B, or /I for each transfer; if
you have specified none of these, the system proceeds as if you
had typed /A.

OS/8 SYSTEM PROGRAMS

OPTION	MEANING
/A	Transfer files in ASCII mode.
/B	Transfer files in binary mode (used for absolute and relocatable binary files).
/C	Eliminate trailing blanks. Valid for ASCII only.
/D	Delete the old copy of the output file before doing any data transfer.
/E	List directories in extended form
/F	List directories in short form.
/G	Ignore any errors that occur during a file transfer and continue copying.
/I	Transfer files in image mode.
/S	Move all files from the input device to the output device, eliminating any embedded empty files.
/Y	Copy the OS/8 system area from the input device to the output device. *also makes bootable copy*
/Z	Zero directory of output device before file transfer.

Table 1-3

PIP Options

OS/8 SYSTEM PROGRAMS

RXCOPY

You can use the RXCOPY program to copy or transfer the entire contents and system head of one RX floppy to another RX floppy disk. Use this program only with RX permanent device names or a user-defined name that you have assigned to an RX device. Specifying file names in the I/O specification line results in an error message.

To load and run RXCOPY, type:

```
.R RXCOPY
*OUTPUT DEV:<INPUT DEV:/OPTIONS
•
```

EXAMPLE:

```
.R RXCOPY
*RXA1:<SYS:
```

When you have loaded RXCOPY and entered the I/O specification line at the keyboard, the program copies the input device to the output device on a sector-by-sector basis. When the operation is complete, the monitor dot appears on the screen, and the specified output device becomes an exact duplicate of the input device.

OPTION	MEANING
/P	Pause and wait for response before execution
/N	Copy with no check
/M	Check both devices for identical contents
/R	Read every block on specified device and list any bad tracks and sectors

&ABSLDR
ABSLDR.SV

@CALLING COMMANDS:
.LOAD DEV:BINFILE.BN,...
.LOAD BINFILE.BN,... /FROM DSK

@SWITCHES:
/8 PROG DOESN'T USE BELOW 02000
/9 PROG DOESN'T USE BELOW 12000
/G GO
/I CORE IMAGE FILE
/P PROG DOESN'T DESTROY EXTENDED BATCH RESIDENT
/R RESET
/S MULTIPLE BINARIES/FILE
/N FORCE LOADING TO FIELD N (N IS AN OCTAL DIGIT)
/N FORCE LOADING TO FIELD N (N IS AN OCTAL DIGIT)
=FNNNN SET STARTING ADDRESS

```
&ASSIGN
&DATE
&DEASSIGN
&GET
&MEMORY
&R
&RUN
&SAVE
&START
&SQUISH
&UA
&ZERO
KEYBOARD MONITOR AN OTHER COMMANDS

@CALLING COMMANDS:
.ASSIGN DEV NAME /ASSIGN NAME TO DEVICE
.DAY DD-MON-YY /ENTER DATE INTO SYSTEM
.DEASSIGN /DEASSIGN LOGICAL DEVICE NAMES
.GET DEV FILE.EX /LOAD CORE IMAGE
.MEMORY N /SPECIFY HIGHEST MEMORY FIELD AVAILABLE
.R FILE /EXECUTE FILE.SV FROM SYS
.RUN DEV FILE.EX /EXECUTE FILE.EX FROM THE DEVICE
.SAVE DEV FILE.EX /SAVE CORE IMAGE
.SQUISH DEV: /COMPRESS FILE STORAGE ON DEVICE
.START FNNNN /START EXECUTION
.UA COMMAND /SAVE COMMAND(.UA<CR> EXECUTES IT)
.ZERO DEV: /ZERO DEVICE'S DIRECTORY
```

&BASIC
BASIC.SV

@CALLING COMMANDS:
.BASIC
.R BASIC

@INTERNAL COMMANDS:
BYE EXIT FROM BASIC
.IST LIST CURRENT PROGRAM'S STATEMENTS
NAME RENAME CURRENT PROGRAM
NEW PREPARE FOR A NEW PROGRAM
OLD RETRIEVE AN OLD PROGRAM
RUN RUN CURRENT PROGRAM
SAVE SAVE CURRENT PROGRAM
SCRATCH DELETE CURRENT PROGRAM

.HELP BCOMP PRINTS BASIC COMPILER ERROR MESSAGES
.HELP BRTS PRINTS BASIC RUN-TIME ERROR MESSAGES

&BOOT
BOOT.SV

@CALLING COMMANDS:
.BOOT/DV

@SWITCHES:
/CA TA8E CASSETTE CAPS-8
/DK ANY DISK
/DL LINCTAPE DIAL
/DM ANY DISK DISK MONITOR
/DT ANY TAPE
/LT LINCTAPE
/PT PT8E (LOADS BINLDR)
/RE RK8E DISK
/RF RF08,DF32 DISKS
/RK RK8 DISK
/RX RX8E FLOPPY DISK
/TD TD8E DECTAPE
/TY TYPESET (UNIT 4)
/VE VERSION #
/TC TC08 DECTAPE ALL SYSTEMS
/ZE ZEROES CORE (FIELD 0)

DEVICES ARE UNIT 0 IF NOT SPECIFIED

DV. HALT AFTER LOADING BOOTSTRAP

&BUILD
BUILD.SV

@INTERNAL COMMANDS:
$ALTER GRP,LOC
$ALTER GRP,LOC=VALUE
$BOOT
$BUILD
$CTL ACTNAM
$CTL ACTNAM=VALUE
$CORE N
$DCB ACTNAM
$DCB ACTNAM=VALUE
$DSK ACTNAM
$DSK GRP:NAME
$EXAMINE GRP,LOC
$INSERT GRP
$INSERT GRP:NAME,...
$LOAD DEV:FILENM.BN
$LOAD ACTNAM
$NAME ACTNAM=NEWHAM
$PRINT
$QLIST
$REPLACE ACTNAM,...=GRP:NEWNAM,,,
$SIZE ACTNAM
$SIZE ACTNAM=VALUE
$SYS GRP
$SYS GRP:NAME,...
$UNLOAD GRP
$UNLOAD GRP:NAME,...

@ERRORS:
?BAD ARG NO DEVICE NAME IN LOAD COMMAND
?BAD INPUT INPUT NOT A VALID BINARY FILE
?BAD LOAD BINARY HANDLER NOT IN CORRECT FORMAT
?BAD ORIGIN ORIGIN IN BINARY FILE NOT IN RANGE 200-577
?CORE NOT ENOUGH MEMORY AVAILABLE
?DSK DSK IS NOT FILE STRUCTURED
?HANDLERS MORE THAN 15 HANDLERS ARE ACTIVE
I/O ERROR ERROR DURING LOAD
?NAME MISSING NAME
NO ROOM TOO MANY DEVICE HANDLERS LOADED
NAME NOT FOUND DEVICE OR FILE NAME NOT FOUND
?PLAT TOO MANY PLATTERS SPECIFIED FOR DEVICE
?SYNTAX BAD SYNTAX
?SYS HANDLER IS NOT A SYSTEM HANDLER OR TWO SYSTEM HANDLERS
ARE ACTIVE OR HANDLER CORESIDENT WITH NON-ACTIVE SYS
SYS ERROR I/O ERROR OCCURRED WITH SYSTEM HANDLER. PRESS CONTINUE
TO RETRY
SYS NOT FOUND NO ACTIVE HANDLER BY NAME OF SYS DURING BOOTSTRAP

&
&CCL
OS/8 MONITOR COMMANDS

CMD PROG EXPL

ASSIGN KBM ASSIGNS LOGICAL NAME
BACKSP CAMP BACKSPACES DEV
BASIC BASIC ENTERS BASIC SYSTEM
BOOT BOOT BOOTSTRAP TO DEV
CCL CCL DISABLES CCL
COMPARE SRCCOM COMPARES FILES
COMPILE PAL8 COMPILES PROG
F4/FORT
BASIC
RALF
SABR
COPY FOTP COPIES FILES
CREATE EDIT OPENS FILE FOR EDITING
CREF PAL8 ASSEMBLES AND CHAINS TO CREF
CREF CREF'S LISTING
DATE KBM/CCL SPECIFIES DATE
DEAS CCL DEASSIGNS LOGICAL DEVICES
DELETE FOTP DELETES FILES
DIRECT DIRECT PRINTS DIRECTORIES
DUPLIC RXCOPY COPIES RX DISKS
EDIT EDIT EDITS FILE
EOF CAMP WRITES END-OF-FILE
EXECUTE PAL8 COMPILES AND EXECUTES
F4/FORT
BASIC
RALF
SABR
ABSLDR LOADS AND EXECUTES
LOAD(ER)
GET KBM GETS CORE-IMAGE
HELP HELP LIST'S HELP FILE
LIST FOTP LISTS FILES
LOAD ABSLDR LOADS FILES
LOAD(ER)
MAKE TECO MAKES NEW FILE FOR EDITING
MAP BITMAP PRINTS BITMAP
MEMORY CCL SPECIFIES MACHINE CORE SIZE
MUNG TECO MUNGS FILE WITH TECO MACRO
ODT KBM RUNS OCTAL DEBUGGER
PAL PAL8 RUNS PAL8
PRINT LPTSPL RUNS `LPTSPL' IF PRESENT
PUNCH FOTP PUNCHES DATA
R KBM RUNS PROGRAM FROM SYS:
RENAME FOTP RENAMES FILES
RESORC RESORC PRINTS RESOURCE OF SYSTEMS
REWIND CAMP REWINDS DEV

RUN KBM RUNS PROGRAM
SAVE KBM SAVES CORE IMAGE
SET SET ALTERS PARAMETERS
SKI CAMP SKIPS RECORDS
SQUISH PIP SQUISHES DEV
START KBM STARTS PROG
SUBMIT BATCH STARTS BATCH JOB
TECO TECO EDITS FILE
TYPE FOTP TYPES FILES
UA CCL REMEMBERS COMMAND
UB CCL
UC CCL
UNLOAD CAMP UNLOADS DEV
VERSION CCL TYPES VERSION #
ZERC PIP ZEROES DEV

@SWITCHES:
-L OUTPUT TO LPT:
-S OUTPUT TO TV:
-T OUTPUT TO TTY:
-P OUTPUT TO PTP:
-D OUTPUT TO DUMP:
-N OUTPUT TO NULL:
-LS PRODUCE LISTING
-NM NO BINARY YET
-MP PRODUCE MAP
-EXT SET DEFAULT EXTENSION

@FEATURES:
/X PASS SWITCH OPTION X TO PROGRAM
(XYZ) PASS SWITCH OPTIONS TO PROGRAM
[N] MAX OUTPUT SIZE
=NNN PASS OCTAL NUMBER TO PROGRAM
#NNN TAKE INTERNAL OCTAL FORM OF FILENAME
@FILE REPLACE IN CMD LINE BY FILE'S CONTENTS
$ COMPLEMENT DEFAULT ALTMODE SWITCH

&DIRECT
DIRECT.SV

@CALLING COMMANDS
.DIR DEV:LISTFILE.DI<DEV:FILETYPE /* IS WILD NAME OR EXTENSION
.DIR FILETYPE /? IS WILD CHARACTER

@SWITCHES:
/B INCLUDE STARTING BLOCK NUMBER (OCTAL)
/C LIST ONLY FILES WITH CURRENT DATE
/E INCLUDE EMPTIES
/F FAST MODE
/I PRINT ADDITIONAL INFO WORDS
/L USUAL MODE
/M LIST EMPTIES ONLY
/O LIST ONLY FILES WITH OTHER THAN TODAY'S DATE
/R LIST REMAINDER OF FILES AFTER FIRST ONE (BUT USE /C,/O)
/U TREAT EACH INPUT SPECIFICATION SEPARATELY
/V LIST FILES NOT OF FORM SPECIFIED
/W GIVE VERSION NUMBER
=N USE N COLUMNS

&EDIT
&CREATE
EDIT.SV

@CALLING COMMANDS:
.EDIT DEV:OUTFILE.PA<DEV:INFILE.PA
.CREATE OUTFILE.PA

@SWITCHES:
/A RETURN TO EDITOR ON CLOSE
/B CONVERT 2 OR MORE SPACES TO TAB
/D PREDELETE

@ERRORS
?0 INPUT ERROR
?1 OUTPUT ERROR
?2 CLOSE ERROR
?3 OPEN ERROR
?4 COULDN'T LAOD DEVICE HANDLER

@INTERNAL COMMANDS:
A APPEND TEXT
B LIST # OF CORE LOCATIONS LEFT
C CHANGE TEXT
D DELETE TEXT
E OUTPUT BUFFER, TRANSFER REST OF DATA, AND CLOSE
F AND J, SEARCH FOR NEXT OCCURRENCE OF SAME STRING
G GET AND LIST TAGGED LINE
I INSERT
J INTER-BUFFER STRING SEARCH
K KILL BUFFER
L LIST TEXT
M MOVE TEXT
N WRITE BUFFER, KILL AND READ NEXT PAGE
P WRITE TEXT BUFFER TO OUTPUT
Q IMMEDIATE END OF FILE
R READ TEXT FROM INPUT DEVICE
S CHARACTER SEARCH
T PUNCH TRAILER TAPE
V PRINT ON LP08
Y INPUT TEXT PAGE, NO OUTPUT
PRINT VERSION NO.

```
&FOTP
&LIST
&COPY
&RENAME
&TYPE
&DELETE
FOTP.SV
```

@CALLING COMMANDS:
.COPY DEV:OUTFILE.EX<DEV:INFILE.EX /* IS WILD NAME OR EXTENSION
.REN DEV:NEWFILE.EX<DEV:OLDFILE.EX /? IS WILD CHARACTER
.DEL DEV:FILE.EX
.LIST DEV:FILE.EX /= .COPY LPT:<DEV:FILE.EX
.TYPE DEV:FILE.EX /= .COPY TTY:<DEV:FILE.EX

@SWITCHES:
/C MATCH ONLY FILES WITH CURRENT DATE
/D DON'T TRANSFER (I.E. AT MOST ONLY DELETE
/F REQUEST NEW DEVICE IF OUT OF ROOM
/L TYPE LOG OF INPUT FILENAME MATCHES (*)
/N NO PRE-DELETE
/O MATCH ONLY FILES WITH OTHER THAN TODAY'S DATE
/Q QUERY USER ABOUT FILE BEFORE OPERATION (*)
/R RENAME
/T USE TODAY'S DATE
/U TREAT EACH INPUT SPECIFICATION SEPARATELY
/V MATCH FILES NOT OF FORM SPECIFIED
/W PRINT VERSION #

NOTES:

(*) /D CAUSES LOG OF OUTPUT FILES (IF /L ALSO)

IF INDEV: EQUALS OUTDEV:, THEN /N IS FORCES.

IF NO INPUT FILE, *.* IS FORCED EXCEPT FOR /D
IF OUTPUT DEVICE SPECIFIED, BUT NO FILE, *.* IS ASSUMED.

↻P ABORT OPERATION, FIX OUTPUT DIRECTORY
↻C FIX OUTPT DIRECTORY, RETURN TO OS/8
↻ SUPPRESS TYPEOUT

&ODT
ODT

@CALLING COMMANDS
.ODT

@INTERNAL COMMANDS:
NNNNN/ OPEN LOC
/ REOPEN LAST OPENED LOC
NN<CR> DEPOSIT NN IN OPEN LOC, CLOSE LOC
NN<LF> DEPOSIT NN IN OPEN LOC, CLOSE LOC, OPEN AND DISPLAY NEXT
LOC
NN;... DEPOSIT NN IN OPEN LOC, CLOSE AND OPEN NEXT LOC
<CR> CLOSE PREVIOUSLY OPENED LOC
<LF> CLOSE LOC, OPEN NEXT LOC
N+ OPEN CUR LOC+N
N- OPEN CUR LOC-N
⊙ CLOSE LOC, OPEN LOC ADDRESSED BY CONTENTS
_ CLOSE LOC, OPEN POINTED TO BY CONTENTS
ÑNG GO
NNB ESTABLISH BREAKPOINT
B REMOVE BREAKPOINT
A OPEN AC
L OPEN LINK
C CONTINUE FROM BREAKPOINT
NNC CONTINUE, ITERATE NN TIMES
M OPEN SEARCH MASK
<LF> OPEN LOWER SEARCH LIMIT
<LF> OPEN UPPER SEARCH LIMIT
NNW SEARCH CORE FOR NN MASKED BETWEEN LIMITS
D OPEN DATA FIELD (ØØ1Ø=FIELD 1)
F OPEN FIELD FOR ⊙, _, W (ØØ1Ø=FIELD 1)
⊙O SUPPRESS PRINTING

```
& PIP
PIP.SV

@SWITCHES:
/A ASCII MODE
/B BINARY MODE
/C ELIM TRAILING BLANKS
/D DELETE OUTPUT FILE BEFORE TRANSFER
/G IGNORE ERRORS
/I IMAGE MODE
/O OKAY TO COMPRESS OR ZERO
/S SQUISH
/T CONVERT TABS TO SPACES, ETC.
/V VERSION #
/Y COPY SYSTEM HEAD
/Z ZERO OUTPUT DIRECTORY BEFORE TRANSFER
=N # OF ADDITIONAL INFO WORDS (/Z OR /S)
=N SIZE TO CLOSE OUTPUT FILE (/I)
```

&SET
SET.SV

@CALLING COMMANDS:
.SET DEV PARAMETER(S)
.SET DEV NO PARAMETER(S)

@PARAMETERS:
READONLY DECLARE DEVICE TO BE READ ONLY
FILES DECLARE DEVICE TO BE FILE STRUCTURED
DVC CHANGE DEVICE CODES
VERSION X CHANGE VERSION
LOCATION N[=M] EXAMINE OR CHANGE LOCATIONS
LV8E DECLARE LINE PRINTER TO BE AN LV8E
LA8A DECLARE LINE PRINTER TO BE LA180 ON DKC8-AA
LA78 SAME AS .SET LPT NO LA8A
WIDTH N SET WIDTH OF LINE PRINTER OR TTY
LC DECLARE LINE PRINTER OR TTY TO HAVE LOWER CASE
ECHO RESTORE TTY CHARACTER ECHOING
PAGE RESTORE TTY ⊖S AND ⊖Q FACILITIES
TAB IN TTY PRINT TABS (DON'T SIMULATE WITH SPACES)
FILL IN TTY APPEND FILL CHARACTERS AFTER TABS
FLAG IN TTY FLAG LOWER CASE CHARACTERS
SCOPE ERASE CHARACTER ON TTY RUBOUTS
ESC PRINT ESC(ASCII 033) WITHOUT CONVERTING IT TO $ SIGN
ARROW PRINT CONTROL CHARACTERS WITH UP ARROW (E.G. ⊖C, ⊖S)
HEIGHT [M] SET TTY TTY SCREEN HEIGHT
PAUSE [N] SET TTY PAUSE TIME
COL N SET DIRECT TO USE N COLUMNS (.SET TTY COL 2)
CODE N CHANGE TTY IOTS OR CARD READER CODES
PARITY EVEN/ODD SET MAGTAPE PARITY
OS8 DECLARE SYS TO BE OS/8
OS78 DECLARE SYS TO BE OS/8
INIT XXXXX CAUSE SYS TO EXECUTE XXXXX ON BOOTSTRAPPING

SECTION 2

DEC/X8 SOFTWARE SYSTEM

INTRODUCTION

DEC/X8 is a modular software system. This implies that the total
system may have many unique software configurations dependent upon
the type and placement of the various software modules. A
software system structured in this manner can be likened to a
modular hardware system in which predefined hardware modules may
be inserted into a predefined hardware mainframe (within certain
constraints) to produce a desired effect.

OVERVIEW

DEC/X8 consists of three major sections. The first and primary
section is the DEC/X8 monitor. The monitor is the software
mainframe and true "work-horse" of the exerciser. It controls
interrupt servicing deferred service queuing, and user-exerciser
communications.

The second section is the DEC/X8 builder. The builder is used
only during the exerciser building phase and provides the means
with which the user "inserts" software modules (jobs) into the
DEC/X8 mainframe and saves the fully configured exerciser in
suitable form for future use. In 4K systems, the builder is
necessarily restricted to paper tape or PMK-Ø2 cassette
input/output. However, in systems with 8K or more memory and at
least one standard mass storage device, the builder becomes
virtually device independent by interfacing directly with the 8K
programming system, OS/8. Both monitor and builder reside on the
same binary tape or file.

The third and final section of DEC/X8 consists of all available
DEC/X8 software modules (jobs). Each of these modules is designed
to exercise a specific function and/or device associated with the
PDP 8 hardware. Each module resides on its own binary tape (or a
file on a floppy disk), has its own descriptive document, and is
included in product code "AXZZZ". The list of modules available
from the DEC software distribution center (SDC) may be found in
the current edition of the "PDP-8 Software Price List". In
addition, all object programs associated with DEC/X8 are available
on OS/8 formatted media and may be ordered from the software
distribution center.

DEC/X8 SOFTWARE MAINFRAME
16K EXAMPLE


```
               ........................
               .                      .
USER<-------->.        AXQAB-E        .     USER-EXERCISER COMMUNICATIONS
VT100         .                       .     SERVICE INTERRUPT JOBS
               .        MONITOR        .     SERVICE BACKGROUND JOBS
               .                      .
               ........................
               .                      .
               .                      .     USED ONLY IN THE BUILDING
               .        BUILDER        .     PHASE TO BUILD AND SAVE
               .                      .     THE EXERCISER.
               ........................
```

EXERCISER MODULES (JOBS)

```
               ........................
               .   J01    4 PAGE      .
               .   J02    4 PAGE      .
               .   J03    2 PAGE      .
               .   J04    2 PAGE      .
    M          ........................
    E          .   J11    4 PAGE      .
    M          .   J12    4 PAGE      .
    O          .   J13    2 PAGE      .
    R          .   J14    2 PAGE      .
    Y          ........................
               .   J21    4 PAGE      .
               .   J22    4 PAGE      .
               .   J23    2 PAGE      .
               .   J24    2 PAGE      .
               ........................
               .   J31    4 PAGE      .
               .   J32    4 PAGE      .
               .   J33    2 PAGE      .
               .   J34    2 PAGE      .
               ........................
```

DEC/X8 BUILDING PROCEDURE

Three steps must be accomplished to successfully create a DEC/X Exerciser.

    -    Planning
    -    Building
    -    Saving

Planning Phase

The purpose of the following steps is to describe the planning which is necessary to build an operational DEC/8X Exerciser. The maximum configuration is still restricted to 32K even if more memory is available. This restriction allows "custom" exercisers to be configured on 32K systems even though they will be used on 128K systems. Part of the KT8-A support expands this 32K exerciser up to 128K at load time.

1. Complete the DEC/X8 hardware inventory worksheet.

    -    Process type
    -    Available memory
    -    Internal processor options
    -    Hardware you wish to test
    -    Required module (job)
    -    Number of pages

    To determine what software modules are required and their page size reference the "DEC/X8 Software Module Index" (AXQAF).

2. Complete the DEC/X8 module work sheet. Write in the module name to the right of each job number following the rules for page size as follows:

    JX1 ---- 4 page   JX3 ----- 2 page
    JX2   JX4

    Using the "DEC/X8 Software Module Index" fill in the priority of each module beginning with priority 00 (i.e. S,B,C,....X would be pri 00,01,02, etc.).

3. Complete the DEC/X8 priority work sheet (using the data from the module work sheet). Priority 00 first then 01,02, etc. to determine if a module requires parameters (for initialization). It will be necessary to reference module microfiche AX?????.

4. We are now ready to "build" a DEC/X8 exerciser.

SAMPLE BUILDING DIALOGUE

.R DX8MBE   —THE OS/8 MEDIA IS BOOTED AND THE
                                             DECX/8 MONITOR BUILDER IS CALLED
                                             AND IDENTIFIES ITSELF

REVISION E

AXQAB-E

REFER TO THE DEC/X8 USERS GUIDE "PREFACE" FOR A
DESCRIPTION OF THE CHANGES FROM REV A TO REV E.

HSR? N   —SYSTEM HAS NO PC04
HSP? N

EXT MEM? (00-37)03   —FIELDS 00-03 PRESENT FOR 16K

PDP-8? N   —NOT A "STRAIGHT-8"

PDP-8E? Y   —8E OR 8A

KT? N   —NO KT8-A

VT100? Y   —CONSOLE TERMINAL

EAE? N   —NO EAE ON SYSTEM

PF? N   —NO PF (POWER FAIL OPTION)

MP? N   —NO MEMORY PARITY OPTION

TS? N   —TIME SHARE DISABLED

^C

!^B OK? Y   —NOW IN BUILDER MODE

INPUT VIA OS/8 ? Y

OUTPUT VIA OS/8 USING FILE BASED DEVICE? Y

% B   —BEGIN TO BUILD, INSTALLING
                                             MODULES IN PRIORITY ORDER

STUDENT GUIDE

SAMPLE BUILDING DIALOGUE

```
PRI:00%                           -REQUESTING FIRST JOB NUMBER

                                  -ENTER  EACH  JOB  FROM  THE  DECX8
                                   WORK SHEET

PRI:00% J02                       -RX8-E JOB
*FLOPPY                            FILE NAME
DC:0750                           STANDARD DEVICE CODE?  (75)
                                   CARRIAGE RETURN = YES

PRI:01% J01                       -CLOCK JOB.  USED FOR ALL CLOCKS
*TIMERA                            FILE NAME
DC:0130                            DEVICE CODE 13?  (YES)

PRI:02% J03                       -OPERATE JOB
*OPRATE                            FILE NAME

PRI:03% J14                       -OPERATE JOB
*OPRATE                            FILE NAME

PRI:04% J33                       -OPERATE JOB
*OPRATE                            FILE NAME

PRI:05% J34                       -OPERATE JOB
*OPRATE                            FILE NAME

PRI:06% J04                       -MRI08A OB
*MRI08A                            FILE NAME

PRI:07% J11                       -RANMRI JOB
*RANMRI                            FILE NAME

PRI:10% J12                       -RANMRI JOB
*RANMRI                            FILE NAME

PRI:11% J13                       -MRI08A JOB
*MRI08A                            FILE NAME

PRI:12% J21                       -RANMRI JOB
*RANMRI                            FILE NAME

PRI:13% J22                       -RANMRI JOB
*RANMRI                            FILE NAME

PRI:14% J23                       -MRI08A JOB
*MRI08A                            FILE NAME

PRI:15% J24                       -MRI08A JOB
*MRI08A                            FILE NAME
```

SAMPLE BUILDING DIALOGUE

PRI:16% J31   -RANMRI JOB
*RANMRI   FILE NAME

PRI:17% J32   -RANMRI JOB
*RANMRI   FILE NAME

PRI:20% ^C   -NO MORE JOB SLOTS TO ENTER;
                                      RETURN TO MONITOR WITH ^C

!   -MONITOR PROMPT

                                     -WE ARE NOW READY TO INITIALIZE
                                      THE EXERCISER BEFORE WE SAVE IT
                                      ON THE OS/8 MEDIA

!AI   -INITIALIZE "ALL" COMMAND

IJ01   -MONITOR WILL IDENTIFY EACH JOB
TIMERA   BY ITS FILE NAME AND REQUEST
A 100   PARAMETERS.   YOU MUST REFERENCE
B 0000   THE MICOFICHE FOR EACH JOB.
C 0000   EXAMPLE IS FOR THE PDP-8E LINE
D 0   CLOCK.

IJ02   -RX8-E JOB
FLOPPY
A 0   WE ARE SELECTING ALL DEFAULTS
B 0   EXCEPT FOR DRIVE SELECTION.
C 0   WE HAVE SELECTED DRIVE 1 ONLY
D 1 0 1

IJ03   -FOR THE OPERATE TEST TO RUN
OPRATE 2   WE MUST SPECIFY PROCESSOR TYPE.

IJ04   -NO PARAMETERS FOR THIS JOB BUT
MRI08A   MONITOR WILL IDENTIFY IT ANYWAY.

IJ11
RANMRI

IJ12
RANMRI

IJ12
RANMRI

IJ13
MRI08A

STUDENT GUIDE

SAMPLE BUILDING DIALOGUE

```
IJ14                                -OPERATE JOB 2 = PDP-8E/A
OPRATE 2

IJ21
RANMRI

IJ22
RANMRI

IJ23
MRI08A

IJ24
MRI08A

IJ31
RANMRI

IJ32
RANMRI

IJ33
OPRATE 2                            - 2 = PDP-8E/A

IJ34
OPRATE 2                            - 2 = PDP-8E/A

!                                   -WE ARE NOW READY TO SAVE THE
                                    CUSTOM EXERCISER.  WE MUST
                                    RETURN TO THE BUILDER.

! ^B OK? Y                          -RETURN TO BUILDER

%                                   -BUILDER PROMPT

%P                                  -TELLS BUILDER TO SAVE EXERCISER
```

       UPON ".", TYPE "SAVE (DEV) (FILE)" ---

```
.SAVE RXA0:TEST                     -THIS WILL SAVE THE EXERCISER ON
                                    RXA0 AND NAME IT "TEST.SV"

                                    -WE NOW HAVE A RUNABLE DEC/X8
                                    FILE NAMED TEST.SV ON RXA0.
```

2-9

RUNNING DEC/X8

.   -OS/8 HAS BEEN BOOTED

.   -OS/8 KEYBOARD MONITOR WAITING
.   FOR COMMAND INPUT.

.

.R TEST   -THIS COMMAND WILL LOAD AND RUN
                                           THE OS/8 FILE "TEST.SV"

Eventually the processor will halt (PDP-8E MA=3002). At this time
remove all volatile media. i.e disks, floppys etc.

                            NOTE
                If the processor is of the VT78 category
                no "HLT" instructions are performed.
                The exerciser comes up in the DEC/X8
                monitor.

Starting the customized exerciser (from a HALT following LOAD)

1.   Extended load address 0000.
2.   Load address 3000.
3.   Depress start.
4.   The program will HALT with MA = 3002.
5.   Depress continue.

^C   -PRINT ^C AND DISPLAY THE
!   DEC/X8 MONITOR PROMPT.
!S   -THIS COMMAND WILL PRINT JOB
                                           STATUS. NOTE THAT ALL JOBS ARE
                                           IN THE "K" (KILL) STATE.

JOB	MODULE	MAINDEC	FLD	PRI	STATE	CNTR
01	TIMERA	AXDKA-E	00	01	K	0000
02	FLOPPY	AXRXB-C	00	00	K	0000
03	OPRATE	AXKAC-C	00	02	K	0000
04	MRI08A	AXKAA-B	00	06	K	0000
11	RANMRI	AXKAB-B	01	07	K	0000
12	RANMRI	AXKAB-B	01	10	K	0000
13	MRI08A	AXKAA-B	01	11	K	0000
14	OPRATE	AXKAC-C	01	03	K	0000
21	RANMRI	AXKAB-B	02	12	K	0000
22	RANMRI	AXKAB-B	02	13	K	0000
23	MRI08A	AXKAA-B	02	14	K	0000
24	MRI08A	AXKAA-B	02	15	K	0000
31	RANMRI	AXKAB-B	03	16	K	0000
32	RANMRI	AXKAB-B	03	17	K	0000
33	OPRATE	AXKAC-C	03	04	K	0000
34	OPRATE	AXKAC-C	03	05	K	0000

RUNNING DEC/X8

Two commands are available for running jobs.

1. "RJXX" run job JXX results in the job JXX being switched to
   the run state. The job will not be started until ^E or ^R is
   typed to exit the keyboard monitor.

   To run jobs J01,J02,J03 and J04 type the following.

```
!                                    -MUST BE AT KEYBOARD MONITOR
!RJ01                                -SWITCH J01 TO "R" STATE
!RJ02                                -SWITCH J02 TO "R" STATE
!RJ03                                -SWITCH J03 TO "R" STATE
!RJ04                                -SWITCH J04 TO "R" STATE
!^E                                  -EXIT KEYBOARD MONITOR AND
                                      RUN ALL "R" STATUS JOBS.
```

2. "AR" run all jobs is available in systems with 8K or more
   memory and causes all existent jobs not already in the "R"
   state to be switched to the "R" state.

   To run all jobs type the following:

```
!                                    -MONITOR PROMPT
!AR                                  -SWITCHES ALL JOBS TO "R"
!^E                                  -EXIT KEYBOARD MONITOR AND
                                      RUN ALL JOBS.

^C                                   -USER COMMAND TO RETURN TO THE
!                                     DEC/X8 KEYBOARD MONITOR.
!S                                   -ALL JOBS IN RUN STATE
```

DEC/X8

JOB	MODULE	MAINDEC	FLD	PRI	STATE	CNTR
01	TIMERA	AXDKA-E	00	01	R	0000
02	FLOPPY	AXRXB-C	00	00	R	0000
03	OPRATE	AXKAC-C	00	02	R	0000
04	MRI08A	AXKAA-B	00	06	R	0000
11	RANMRI	AXKAB-B	01	07	R	0000
12	RANMRI	AXKAB-B	01	10	R	0000
13	MRI08A	AXKAA-B	01	11	R	0000
14	OPRATE	AXKAC-C	01	03	R	0000
21	RANMRI	AXKAB-B	02	12	R	0000
22	RANMRI	AXKAB-B	02	13	R	0000
23	MRI08A	AXKAA-B	02	14	R	0000
24	MRI08A	AXKAA-B	02	15	R	0000
31	RANMRI	AXKAB-B	03	16	R	0000
32	RANMRI	AXKAB-B	03	17	R	0000
33	OPRATE	AXKAC-C	03	04	R	0000
34	OPRATE	AXKAC-C	03	05	R	0000

RUNNING DEC/X8

Two commands are available for killing jobs.

1. "KJXX" kill job JXX sets the "COMMAND TO KILL" flag in job
   JXX. The job will not be killed until ^E is commanded and the
   job has the chance to reach a point of exit.

   To kill jobs J01,J11 and J33 type the following:

   !KJ01   -SET KILL FLAG FOR JOB J01
   !KJ11   -SET KILL FLAG FOR JOB J11
   !KJ33   -SET KILL FLAG FOR JOB J33
   !^E   -ENTER RUN MODE TO ALLOW JOBS
                                      TO REACH AN EXIT POINT.

2. "AK" kill all jobs is available only in systems with at least
   8K of memory. The "AK" command sets the "COMMAND TO KILL"
   flag in all existent jobs.

   To kill all jobs type the following:

   !AK   -COMMAND TO KILL ALL JOBS
   !^E   -ENTER RUN MODE TO ALLOW JOBS
                                      TO REACH AN EXIT POINT.

Two commands are available for exiting the DEC/X8 keyboard
monitor, turning the interrupt system on, and resuming job
servicing.

1. ^E "EXERCISE" results in the interrupt being turned on and the
   monitor resuming job servicing. Also any jobs which have just
   been put in the "R" state will be started, any jobs which have
   their "COMMAND TO KILL" flag set will be killed.

2. ^R "EXERCISE and force rotation (or scatter)" results in all
   actions specified by ^E, and in addition forces rotation and
   an automatic status report. If the system has only 4K ^R
   functions exactly as ^E.

   IMPORTANT NOTE:

   Rotation and status reports are available only in 8K or
   greater systems.

RUNNING DEC/X8

In the following example we will show how these commands are used
in running the 16K exericizer that we have already built

```
.                                      -BOOT OS/8
.R TEST                                -RUN THE CUSTOM EXERICIZER
^C                                     -DEC/X8 STARTS IN COMMAND MODE
!S                                     -COMMAND TO DEC/X8 FOR STATUS
```

JOB	MODULE	MAINDEC	FLD	PRI	STATE	CNTR
01	TIMERA	AXDKA-E	00	01	K	0000
02	FLOPPY	AXRXB-C	00	00	K	0000
03	OPRATE	AXKAC-C	00	02	K	0000
04	MRI08A	AXKAA-B	00	06	K	0000
11	RANMRI	AXKAB-B	01	07	K	0000
12	RANMRI	AXKAB-B	01	10	K	0000
13	MRI08A	AXKAA-B	01	11	K	0000
14	OPRATE	AXKAC-C	01	03	K	0000
21	RANMRI	AXKAB-B	02	12	K	0000
22	RANMRI	AXKAB-B	02	13	K	0000
23	MRI08A	AXKAA-B	02	14	K	0000
24	MRI08A	AXKAA-B	02	15	K	0000
31	RANMRI	AXKAB-B	03	16	K	0000
32	RANMRI	AXKAB-B	03	17	K.	0000
33	OPRATE	AXKAC-C	03	04	K	0000
34	OPRATE	AXKAC-C	03	05	K	0000

```
!AR                                    -COMMAND TO RUN ALL EXISTENT JOBS

!^R                                    -EXIT KEYBOARD, ENTER "RUN" MODE
```

TIMERA-J01 FLD 0 ET: 0   00   00

STAT ERR IN J02 MOD:FLOPPY FLD:03 CNTR:0000 RPC:0343 CODE:0004
   SA:0000 SB:0120 SC:0020 SD:0103 SE:0025 SF:7767

ROTATE 2227

DEC/X8

JOB	MODULE	MAINDEC	FLD	PRI	STATE	CNTR
01	TIMERA	AXDKA-E	01	01	R	0000
02	FLOPPY	AXRXB-C	01	00	R	0000
03	OPRATE	AXKAC-C	01	02	R	0000
04	MRI08A	AXKAA-B	00	06	R	0000
11	RANMRI	AXKAB-B	02	07	R	0000
12	RANMRI	AXKAB-B	02	10	R	0000
13	MRI08A	AXKAA-B	02	11	R	0000
14	OPRATE	AXKAC-C	01	03	R	0000
21	RANMRI	AXKAB-B	03	12	R	0000

RUNNING DEC/X8

```
22   RANMRI   AXKAB-B   03   13   R   0000
23   MRI08A   AXKAA-B   03   14   R   0000
24   MRI08A   AXKAA-B   02   15   R   0000
31   RANMRI   AXKAB-B   00   16   R   0000
32   RANMRI   AXKAB-B   00   17   R   0000
33   OPRATE   AXKAC-C   00   04   R   0000
34   OPRATE   AXKAC-C   03   05   R   0000
```

STAT ERR IN J02 MOD:FLOPPY FLD:01 CNTR:0000 RPC:0343 CODE:0004
   SA:0000 SB:0120 SC:0020 SD:0076 SE:0010 SF:7766

STAT ERR IN J02 MOD:FLOPPY FLD:01 CNTR:0000 RPC:0343 CODE:0004
   SA:0000 SB:0210 SC:0020 SD:0074 SE:0003 SF:7767

STAT ERR IN J02 MOD:FLOPPY FLD:01 CNTR:0000 RPC:0343 CODE:0004
   SA:0000 SB:0120 SC:0020 SD:0000 SE:0030 SF:7766

```
^C                              -COMMAND TO STOP DEC/X8 RETURN TO
!                                DEC/X8 KEYBOARD MONITOR.
!KJ02                           -KILL THE JOB THAT IS ERRORING
!^R
```

STAT ERR IN J02 MOD:FLOPPY FLD:03 CNTR:0000 RPC:0343 CODE:0004
   SA:0000 SB:0120 SC:0020 SD:0071 SE:0003 SF:7754

ROTATE 3430

DEC/X8

```
JOB  MODULE   MAINDEC   FLD  PRI  STATE  CNTR
01   TIMERA   AXDKA-E   03   01   R      7705
02   FLOPPY   AXRXB-C   00   00   K      0000
03   OPRATE   AXKAC-C   01   02   R      0000
04   MRI08A   AXKAA-B   00   06   R      0000
11   RANMRI   AXKAB-B   00   07   R      0000
12   RANMRI   AXKAB-B   01   10   R      0000
13   MRI08A   AXKAA-B   02   11   R      0000
14   OPRATE   AXKAC-C   01   03   R      0000
21   RANMRI   AXKAB-B   01   12   R      0000
22   RANMRI   AXKAB-B   02   13   R      0000
23   MRI08A   AXKAA-B   03   14   R      0000
24   MRI08A   AXKAA-B   02   15   R      0000
31   RANMRI   AXKAB-B   02   16   R      0000
32   RANMRI   AXKAB-B   03   17   R      0000
33   OPRATE   AXKAC-C   00   04   R      0000
34   OPRATE   AXKAC-C   03   05   R      0000
```

Note the error message at the top of this page. The DEC/X8 software will report errors either by printout or by halting the processor. The user must then analyze any error printouts or processor halts to determine corrective action.

RUNNING DEC/X8

ERROR REPORTING

There are four major classes of errors:

- Monitor Errors
- Builder Errors
- OS/8 Errors
- Job Errors

MONITOR ERRORS

Monitor errors may occur during the building and running phases. With one exception, all monitor errors result in a program halt without an error printout. The exception is the power fail error as detected by the power fail option. In this case, when power is restored, the monitor prints "PF". The exerciser than enters monitor mode and awaits user commands.

BUILDER ERRORS

Builder errors occur only during the building phase and result in a program halt. Their are no printouts for this class of error.

OS/8 ERRORS

OS/8 errors are indicated by an error printout and return to either the OS/8 keyboard monitor or to the DEC/X8 builder.

JOB ERRORS

Job errors are usually reported by error printouts. A program halt without a report will occur if the error condition is considered fatal.

RUNNING DEC/X8

ERROR HALT RECOVERY PROCEDURES

When an error halt occurs, refer to the table below to find the procedure to follow.

ADDRESS RANGE	MEMORY FIELD	EXERCISER PHASE	PROCEDURE
0000-0177	BF	RUN	1
0200-1177	0	BUILD	1
1200-1377	BF	RUN	1
1400-3177	0/1	ANY	1
3200-3577	BF	ANY	1
3600-4577	BF	RUN	2
4600-5577	BF	RUN	2
5600-6577	0/1	BUILD	1
6600-7177	BF	RUN	2
7200-7577	BF	RUN	2
7600-7777	0/1	BUILD	OS/8 DOCUMENTATION

All other areas are undefined crashes

RECOVERY PROCEDURES

1. To find the error cause or definition and error recovery procedure, refer to the "the table of error halts" of the monitor/builder listing. AXQABF0 DEC/X8 Users Guide.

2. Save all pertinent register contents, halt location, AC, ect. Now examine the first location in that job slot. This location always should contain the job number in bits 6-11. Then compute the relative address of the halt as follows:

    RRRR = AAAA + 200 - SSSS

    Where (RRRR) is the relative address, (AAAA) the absolute address of the halt and (SSSS) is the first address of that job slot. Now using the relative address, reference the applicable module listing to determine the cause of the error. For recovery, restart the exerciser at 003000.

RUNNING DEC/X8

JOB ERROR REPORTS

A job error report always has the following header:

"TYPE ERROR" IN JXX MOD: TITLE FLD:BF CNTR:NNNN RPC:NNNN CODE:NNNN

TYPE: "STAT" for a status error. "DATA" for a data error.
JXX: Job Number
MOD: Module Title
FLD: The memory bank in which the job currently resides
CNTR: The contents of the job's pass counter
RPC: The relative module PC at which the error was detected
CODE: This job's error ID code (refer to module fiche for
        definition)

In addition, various status words pertaining to the job in error
may be indicated in either "STAT" or "DATA" reports. These status
words appear below the header as follows:

SA:NNNN   SB:NNNN   SC:NNNN   SD:NNNN

There may be from 0 to 26(10) status words depending on the
module. The symbols SA,SB, ect. appear in alphabetical order and
are defined in the applicable module document.

"DATA" errors also include a set of data words for each set of
failing data. The data sets are reported after the "SX" and are
prefixed by "DX". After the final "DX" report, a tally of the
number of failing data sets is reported.

We will now examine the error printout that occurred while running
the DEC/X8 Exerciser (16K example)

STAT ERR IN J02 MOD:FLOPPY FLD:03 CNTR:0000 RPC:0340 CODE:0004
    SA:0000 SB:0120 SC:0020 SD:0106 SE:0016 SF:7770

RUNNING DEC/X8

ERROR ANALYSIS EXAMPLE

STAT ERR IN J02 MOD:FLOPPY FLD:03 CNTR:0000 RPC:0343 CODE:0004
    SA:0000 SB:0120 SC:0020 SD:0106 SE:0016 SF:7770

It it important to note that a status (STAT) error occurred
without any "DATA" error information. This indicates that the
data was either good or that no transfer occurred.

The failing job is J02 which is the "Floppy" module currently in
memory field 03. The pass counter (CNTR) indicates no complete
passes yet. This is all the information we can be sure of without
referencing the module microfiche.

REFERENCE MICROFICHE AXRXBC0:

The microfiche for this module defines the symbols in the stat
error printout as follows:

RPC :0343   -JMS ERROR   /STATUS ERROR REPORT

CODE:0004   -WRITE FUNCTION

   SA:0000   -STATUS A REGISTER

   SB:0120   -STATUS B REGISTER

   SC:0020   -DRIVE NUMBER AND TYPE OF TRANSFER

   SD:0160   -TRACK

   SE:0016   -SECTOR

   SF:7770   -NUMBER OF SECTORS TO R/W (COMPLEMENT FORM)

The 0020 (Bit 7), in the command register (SC:), indicates drive 1
is to be selected. Note that status a register is clear. Bit 4
of this register should be set to indicate drive ready. This
error was caused by running the exerciser without making the drive
ready.

RUNNING DEC/X8

ON-LINE MEMORY MODIFICATION/EXAMINING

^O is the basic command used for examining or modifying memory. There are two modes of address specification for this command, absolute and relative. The absolute mode should be used whenever the absolute address is know (monitor address). The relative mode should be used in specifying module (or job) addresses.

The absolute form is as follows where "FFAAAA" is an absolute address:

!^OFFAAAA   DDDD   - FF = FIELD, AAAA = ADDRESS
                                         DDDD = CONTENTS OF LOCATION

The relative form requires the following:

!^OJXX   - JXX = JOB NUMBER
AAAA   - AAAA = RELATIVE ADDRESS

The following example shows how to examine the first five locations of the operate module, job J03 and change the relativve location 0206 to a halt instruction (7402).

!^OJ03			—JOB TO EXAMINE
0200			—FIRST ADDRESS OF MODULE
006600	0003		—LINE FEED TO ADVANCE
006601	1720		—LINE FEED
006602	2201		—LINE FEED
006603	2405		—LINE FEED
006604	0000		—LINE FEED
006605	0130	7402	—CARRIAGE RETURN TO CLOSE
006605	7420		—CARRIAGE RETURN TO MONITOR

DEC/X8

WORKSHEETS AND EXAMPLE

DEC/X8 HARDWARE INVENTORY WORK SHEET

PROCESSOR TYPE

PDP 8_____   PDP 8E/A___✓___   VT78_____

MEMORY (NUMBER OF 4K FIELDS)_____

OPTIONS

KT_____ VT100_____ EAE_____ PF_____ MP_____ TS_____

DEVICES TO TEST	REQUIRED MODULE	PAGES	
RX01 x 2	Flopy	4	
RK8EDS	RK8E	4	
	TT8P	4	D
	+DRST	4	I
		2	E
	NR8P	2	
	RANNRT	2	
KL8 x 3	SLUP	4	

DEC/X8 MODULE WORK SHEET

FIELD 0

J01 _TIMERA_ PRI _D_

J02 _RK8E15_ PRI _E_

J03 _CPPRGE_ PRI _2_

J04 _RX_ PRI _2_

FIELD 1

J11 _FLOPPY_ PRI _E_

J12 _FLOPPY_ PRI _E_

J13 _MRTAL_ PRI ____

J14 _____ PRI ____

FIELD 2

J21 _ADRST1_ PRI _2_

J22 _MSLW1#1_ PRI _J_

J23 _PHVT_ PRI _2_

J24 _____ PRI _2_

FIELD 3

J31 _MSLW#2_ PRI _J_

J32 _MSLW#3_ PRI _J_

J33 _CPPRTE_ PRI _2_

J34 _____ PRI _2_

FIELD 4

J41 _____ PRI ____

J42 _____ PRI ____

J43 _____ PRI ____

J44 _____ PRI ____

FIELD 5

J51 _____ PRI ____

J52 _____ PRI ____

J53 _____ PRI ____

J54 _____ PRI ____

FIELD 6

J61 _____ PRI ____

J62 _____ PRI ____

J63 _____ PRI ____

J64 _____ PRI ____

FIELD 7

J71 _____ PRI ____

J72 _____ PRI ____

J73 _____ PRI ____

J74 _____ PRI ____

DEC/X8 PRIORITY WORK SHEET

PRIORITY	JOB	MODULE	JOB PARAMETERS (IF REQUIRED)
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____
____	____	____	A____ B____ C____ D____ E____ F____

DEC/X8 HARDWARE INVENTORY WORK SHEET

EXAMPLE

PROCESSOR TYPE

PDP 8_____   PDP 8E/A _YES_   VT78_____

MEMORY (NUMBER OF 4K FIELDS)__Ø3___

OPTIONS

KT__NO__   VT1ØØ_YES__   EAE__NO__   PF__NO__   MP__NO__   TS__NO__

DEVICES TO TEST	REQUIRED MODULE	PAGES
RXØ1	FLOPPY	4
LINE CLOCK	TIMERA	4
CPU	MR1Ø8A	2
CPU	RANMRI	4
CPU	OPRATE	2

STUDENT GUIDE

DEC/X8 MODULE WORK SHEET

EXAMPLE

FIELD Ø

J01 __FLOPPY__ PRI __1__

J02 __TIMERA__ PRI __Ø__

J03 __OPRATE__ PRI __2__

J04 __MR!Ø8A__ PRI __6__

FIELD 4

J41 _____ PRI_____

J42 _____ PRI_____

J43 _____ PRI_____

J44 _____ PRI_____

FIELD 1

J11 __RANMRI__ PRI __7__

J12 __RANMRI__ PRI __1Ø__

J13 __MRIØ8A__ PRI __11__

J14 __OPRATE__ PRI __3__

FIELD 5

J51 _____ PRI_____

J52 _____ PRI_____

J53 _____ PRI_____

J54 _____ PRI_____

FIELD 2

J21 __RANMRI__ PRI __12__

J22 __RANMRI__ PRI __13__

J23 __MRIØ8A__ PRI __14__

J24 __MRIØ8A__ PRI __15__

FIELD 6

J61 _____ PRI_____

J62 _____ PRI_____

J63 _____ PRI_____

J64 _____ PRI_____

FIELD 3

J31 __RANMRI__ PRI __16__

J32 __RANMRI__ PRI __17__

J33 __OPRATE__ PRI __4__

J34 __OPRATE__ PRI __5__

FIELD 7

J71 _____ PRI_____

J72 _____ PRI_____

J73 _____ PRI_____

J74 _____ PRI_____

DEC/X8 PRIORITY WORK SHEET

EXAMPLE

PRIORITY	JOB	MODULE	JOB PARAMETERS (IF REQUIRED)
00	J01	TIMERA	A 100 B 0000 C 0000 D 0 E F
01	J02	FLOPPY	A 0 B 0 C 0 D 101 E F
02	J03	OPRATE	A 2 B C D E F
03	J14	OPRATE	A 2 B C D E F
04	J33	OPRATE	A 2 B C D E F
05	J34	OPRATE	A 2 B C D E F
06	J04	MRI08A	A B C D E F
07	J11	RANMRI	A B C D E F
10	J12	RANMRI	A B C D E F
11	J13	MRI08A	A B C D E F
12	J21	RANMRI	A B C D E F
13	J22	RANMRI	A B C D E F
14	J23	MRI08A	A B C D E F
15	J24	MRI08A	A B C D E F
16	J31	RANMRI	A B C D E F
17	J32	RANMRI	A B C D E F
			A B C D E F
			A B C D E F
			A B C D E F
			A B C D E F
			A B C D E F
			A B C D E F
			A B C D E F
			A B C D E F

DEC/X8

JOB AND DEVICE CODE INFORMATION

DECX8 JOB MODULE SIZE AND PRIORITY LIST

NAME	SIZE	PRIORITY	NAME	SIZE	PRIORITY
TC01DT	4	A	HSRHSP	2	I
DC08A	4	A	DC02	2	I
TA8ECS	4	B	TYPSET	4	I
RL8A	4	B	CDP8	4	I
RX02	4	B	DP8E	4	I
CARD8E	2	C	LPD8	2	I
VT8E	4	D	ICSX8	2	I
TIMERA	4	D	ADRSTT	4	I
FLOPPY	4	E	MULTTY	2	J
RK8EDS	4	E	MULSLU	4	J
RF08DS	4	E	MSLULP	4	J
DF32	4	E	TTYLUP	2	L
TC12LT	4	F	RANMRI	4	Z
FPP12	4	F	MRI08A	2	Z
FPP8A	4	F	OPRATE	2	Z
TM8EMT	4	F	NOTFUN	2	Z
PLOTER	4	G	EAEALL	4	Z
TC58MT	4	G	EAEDP	2	Z
LQP8	2	H	TD8EDT	4	Z
PRNTER	2	H	VCAD8E	2	Z

PDP-8E DEVICE CODES

00/	PROCESSOR	50/	PLOTTER
01/	HSR	51/	"
02/	HSP	52/	"
03/	CONSOLE KEYBOARD	53/	A/D CONVERTER
04/	CONSOLE PRINTER	54/	" "
05/	VC8-E	55/	FPP
06/	N/A	56/	FPP
07/	N/A	57/	
10/	N/A	60/	DF32/RL8-E
11/	DC02/KG8-EA	61/	DF32/RF08/RL8-E
12/	DC02	62/	DF32/RF08
13/	CLOCK	63/	DF32/CR8-E/RF08
14/	N/A	64/	RF08
15/	N/A	65/	LE-8
16/	N/A	66/	LA180
17/	N/A	67/	CR8-E
20-27/	MEMORY	70/	TC58/TM8-E
30-37/	USER DEFINED	71/	TC58/TM8-E
40/	DP8-E	72/	TC58/TM8-E
41/	DP8-E	73/	TC58/RK8-E
42-47/	USER DEFINED	74/	TC58/RK8-E
		75/	RK8-E/RX01
		76/	TC01/RX01
		77/	TC01

PDP 8

STATUS REGISTERS

STATUS REGISTERS

DF32
STATUS REGISTER

AC BIT	MEANING
Ø	PHOTO SYNC
1	EMA 5
2	EMA 4
3	EMA 3
4	EMA 2
5	EMA 1
6	EXT MEM (EA3)
7	EXT MEM (EA2)
8	EXT MEM (EA1)
9	DATA REQUEST LATE
10	DISK NON-EXISTENT OR WRITE LOCK
11	PARITY ERROR

TRACK ADDRESS= EMA BITS 1-3
DISK SELECT = EMA BITS 5-4

LQP8
STATUS REGISTER

BIT	FUNCTION
Ø	PRINTER READY
1	CHARACTER READY
2	CARRIAGE READY
3	PAPER READY
4	NOT USED
5	LEFT PAPER HI (N/A SPLIT PLATEN)
6	CHECK (ERROR)
7	PAPER OUT
8	BAIL DISENGAGE (N/A SPLIT PLATEN)
9	LIFT READ (N/A ONLY USING BLACK RIBBON)
10	LEFT BLACK
11	INTERRUPT ENABLE

STATUS REGISTERS

LQP IOT'S

6500	SKIP ON DONE
6501	READ DATA
6502	AC TO LQP AND MOVE PAPER
6503	AC TO LQP AND MOVE CARRIAGE
6504	AC TO LQP AND PRINT
6505	READ STATUS AND CLEAR DONE
6506	LOAD STATUS AND SET DONE
6507	RESTORE

RF08
STATUS REGISTER

AC BIT	MEANING
0	PHOTO SYNC (PCA)
1	DATA REQ ENABLE (DRE)
2	WRITE LOCKED (WLS)
3	ERROR INTERRUPT ENABLE (EIE)
4	PHOTOSYNC INTERRUPT ENABLE (PIE)
5	COMPLETION INTERRUPT ENABLE (CIE)
6-8	MEMORY FIELD
9	DATA REQUEST LATE (DRL)
10	NON-EXISTENT DISK (NXD)
11	PARITY ERROR (PER)

RK02
STATUS REGISTER

AC BIT	MEANING
0	ERROR FLAG
1	TRANSFER DONE
2	CONTROL BUSY
3	TIME OUT ERROR
4	PARITY OR TIMING ERROR
5	DATA REQUEST LATE
6	TRACK ADDRESS ERROR
7	BAD SECTOR
8	WRITE LOCKED
9	TRACK CAPACITY EXCEEDED
10	SELECT ERROR
11	BUSY

STATUS REGISTERS

RK8E

COMMAND REGISTER

BIT 0-2
000=READ DATA
001=READ ALL
010=SET WR PRT
011=SEEK ONLY
100=WRITE DATA
101=WRITE ALL
BIT 3 INTERRUPT ON DONE
BIT 4 SET DONE ON SEEK DONE
BIT 5 BLOCK LENGTH
0=256
1=128
BIT 6-8 EXT MEM ADD
BIT 9-10 DRIVE SELECT
BIT 11 MSB CYC ADD

STATUS REGISTER

BIT 0 0=DONE 1=BUSY
BIT 1 0=STOP 1=MOVING
BIT 2 ALWAYS=0
BIT 3 1=SEEK FAIL
BIT 4 1=FILE NOT READY
BIT 5 1=CONTROL BUSY
BIT 6 1=TIMING ERROR
BIT 7 1=WRITE LOCK ERROR
BIT 8 1=PARITY ERROR
BIT 9 1=DATA REQ LATE
BIT 10 1=DRIVE STATUS ERROR
BIT 11 1=CYC ADD ERROR

TA8-E (TU60) CASSETTE

STATUS A REGISTER

AC BIT	FUNCTION
11	ENABLE INTERRUPTS
10	NOT USED
9	NOT USED
8-6	FUNCTION REGISTER

AC6	AC7	AC8	OPERATION
0	0	0	READ
0	0	1	REWIND
0	1	0	WRITE
0	1	1	BACK SPACE TO FILE GAP
1	1	0	WRITE GAP
1	0	1	BACK SPACE TO BLOCK GAP
1	1	0	READ/WRITE CRC CHARACTER
1	1	1	SPACE FORWARD TO FILE GAP

5	TU60 DRIVE SELECT 0=DRIVE A 1=DRIVE B
4	SELECT ENABLE
0-3	NOT USED FOR STATUS

STATUS REGISTERS

STATUS B REGISTER

AC BIT	FUNCTION
11	READY FLAG
10	WRITE LOCK
9	REWINDING
8	DRIVE EMPTY
7	EOF (FILE GAP OR BLANK SPACE)
6	EOT/BOT
5	TIMING ERROR (NO RESPONSE)
4	CRC/BLOCK ERROR

TC08
STATUS REGISTER

AC BIT	MEANING
0	ERROR FLAG (EF)
1	MARK TRACK ERROR (MK-TRK)
2	END OF TAPE (END)
3	SELECT ERROR (SE)
4	PARITY ERROR (PE)
5	TIMING ERROR (TIM)
6-8	MEMORY FIELD (MF)
9-10	NOT USED
11	DECTAPE FLAG (DTF)

TC58
STATUS REGISTER

AC BIT	MEANING
0	ERROR FLAG
1	TAPE REWINDING
2	BEGINNING OF TAPE
3	ILLEGAL COMMAND
4	PARITY ERROR
5	END OF FILE
6	END OF TAPE
7	READ COMPARE ERROR
8	RECORD LENGTH ERROR
9	DATA REQUEST LATE
10	BAD TAPE
11	MAGTAPE FLAG

STATUS REGISTERS

RX01   RX02

COMMAND AND STATUS   COMMAND AND STATUS

```
0 = N/A                                 0 = N/A
1 = N/A                                 1 = N/A
2 = 0                                   2 = 0
3 = 0                                   3 = DENSITY
4 = MAINTENANCE                         4 = MAINTENANCE
5 = 8 BIT TRANSFERS                     5 = 8 BIT TRANSFERS
6 = N/A                                 6 = N/A
7 = UNIT SELECT                         7 = UNIT SELECT
8 = FUNCTION CODE                       8 = FUNCTION CODE
9 =      "       "                      9 =      "       "
10=      "       "                      10=      "       "
11= N/A                                 11= N/A
```

FUNCTION CODES

```
000 = FILL BUFFER
001 = EMPTY BUFFER
010 = WRITE SECTOR
011 = READ SECTOR
100 = SET DENSITY (RX02 ONLY)
101 = READ STATUS A
110 = WRITE WITH DEL DATA MARK
111 = READ STATUS B
```

RX01   RX02

STATUS A   STATUS A

```
0 = N/A                                 0 = N/A
1 = N/A                                 1 = N/A
2 = N/A                                 2 = N/A
3 = N/A                                 3 = N/A
4 = DRIVE READY                         4 = DRIVE READY
5 = DELETED DATA                        5 = DELETED DATA
6 = N/A                                 6 = DRIVE DENSITY
7 = N/A                                 7 = DENSITY ERROR
8 = WRITE PROTECT ERROR                 8 = RX02
9 = INTERRUPT DONE                      9 = INTERRUPT DONE
10= PARITY ERROR                        10= N/A
11= CRC ERROR                           11= CRC ERROR
```