XVM/DOS USERS MANUAL

DEC-XV-ODMAA-A-D

XVM Systems

digital

XVM/DOS USERS MANUAL

DEC-XV-ODMAA-A-D

digital equipment corporation · maynard. massachusetts

First Printing, January 1976

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

The following are trademarks of Digital Equipment Corporation:

DIGITAL	DECsystem-10	MASSBUS
DEC	DECtape	OMNIBUS
PDP	DIBOL	OS/8
DECUS	EDUSYSTEM	PHA
UNIBUS	FLIP CHIP	RSTS
COMPUTER LABS	FOCAL	RSX
COMTEX	INDAC	TYPESET-8
DDT	LAB-8	TYPESET-10
DECCOMM	DECsystem-20	TYPESET-11

CONTENTS

CONTENTS (Cont.)

FIGURES

TABLES

LIST OF ALL XVM MANUALS

The following is a list of all XVM manuals and their DEC numbers, including the latest version available. Within this manual, other XVM manuals are referenced by title only. Refer to this list for the DEC numbers of these referenced manuals.

BOSS XVM USER'S MANUAL	DEC-XV-OBUAA-A-D
CHAIN XVM/EXECUTE XVM UTILITY MANUAL	DEC-XV-UCHNA-A-D
DDT XVM UTILITY MANUAL	DEC-XV-UDDTA-A-D
EDIT/EDITVP/EDITVT XVM UTILITY MANUAL	DEC-XV-UETUA-A-D
8TRAN XVM UTILITY MANUAL	DEC-XV-UTRNA-A-D
FOCAL XVM LANGUAGE MANUAL	DEC-XV-LFLGA-A-D
FORTRAN IV XVM LANGUAGE MANUAL	DEC-XV-LF4MA-A-D
FORTRAN IV XVM OPERATING ENVIRONMENT MANUAL	DEC-XV-LF4EA-A-D
LINKING LOADER XVM UTILITY MANUAL	DEC-XV-ULLUA-A-D
MAC11 XVM ASSEMBLER LANGUAGE MANUAL	DEC-XV-LMLAA-A-D
MACRO XVM ASSEMBLER LANGUAGE MANUAL	DEC-XV-LMALA-A-D
MTDUMP XVM UTILITY MANUAL	DEC-XV-UMTUA-A-D
PATCH XVM UTILITY MANUAL	DEC-XV-UPUMA-A-D
PIP XVM UTILITY MANUAL	DEC-XV-UPPUA-A-D
SGEN XVM UTILITY MANUAL	DEC-XV-USUTA-A-D
SRCCOM XVM UTILITY MANUAL	DEC-XV-USRCA-A-D
UPDATE XVM UTILITY MANUAL	DEC-XV-UUPDA-A-D
VP15A XVM GRAPHICS SOFTWARE MANUAL	DEC-XV-GVPAA-A-D
VT15 XVM GRAPHICS SOFTWARE MANUAL	DEC-XV-GVTAA-A-D
XVM/DOS KEYBOARD COMMAND GUIDE	DEC-XV-ODKBA-A-D
XVM/DOS READER'S GUIDE AND MASTER INDEX	DEC-XV-ODGIA-A-D
XVM/DOS SYSTEM MANUAL	DEC-XV-ODSAA-A-D
XVM/DOS USERS MANUAL	DEC-XV-ODMAA-A-D
XVM/DOS V1A SYSTEM INSTALLATION GUIDE	DEC-XV-ODSIA-A-D
XVM/RSX SYSTEM MANUAL	DEC-XV-IRSMA-A-D
XVM UNICHANNEL SOFTWARE MANUAL	DEC-XV-XUSMA-A-D

PREFACE

This manual is the prime document for the XVM/DOS Monitor Software System and describes its features, concepts, programming, and operating procedures. The first four chapters provide a general description of the XVM/DOS System components, both hardware and software, fundamental system concepts, and file structures.

The remaining six chapters deal with the XVM/DOS system at a more technical level. They are primarily concerned with I/O programming requirements and techniques under the Monitor, runtime keyboard commands, and operating procedures. The information in these chapters is directed primarily to readers who are familiar with either the FORTRAN IV language or the MACRO assembly language, described in the FORTRAN IV XVM Language Manual and the MACRO XVM Assembler Language Manual. FORTRAN users, however, need only be concerned with Chapters 7, 8, and 10, since FORTRAN I/O considerations are specifically covered in the FORTRAN IV XVM Operating Environment Manual.

Detailed information on the internal operations of the XVM/DOS Monitor and its file structure as well as procedures for preparing user-created system software are provided in the XVM/DOS System Manual. Brief descriptions of all system programs with applicable document names are contained in Chapter 2.

A quick reference summary of the command strings, operating procedures and error messages for the Monitor and system programs is provided in the XVM/DOS Keyboard Command Guide.

CHAPTER 1
DISK OPERATING SYSTEM

1.1 INTRODUCTION

The XVM Disk Operating System (XVM/DOS) is an integrated set of software designed to meet the demands of research, engineering, and industrial environments. It includes the software necessary for simplfied programming and efficient operations. XVM/DOS (DOS) brings to the user the advantages of disk resident storage via rapid access to the system's resources. The minimum and the optional hardware supported by DOS is described in Section 1.2, "System Hardware." The Monitor System is an integrated set of commonly-used programs used for the development of user applications. These programs include tools for:

Program Preparation,
Compilation,
Assembly,
Debugging, and
Execution of User Programs.

The user controls the operating system by instructions to the Monitor. The Monitor runs the jobs, supervises data and file manipulation, and interacts with the operator/user in a simple, conversational manner.

Within the system, data on mass storage is handled by macro statements used with the MACRO XVM symbolic assembler language, and by the mass storage language elements incorporated into the FORTRAN IV language.

1.1.1 System Features

Disk Resident System Software	The XVM/DOS System Software resides on either DECdisk, Disk Pack, or Disk Cartridge.
Interactive Operation	An interactive keyboard/program Monitor permits device-independent programming and automatic calling and loading of system and user programs.

I/O Device Handlers	Data and file manipulating I/O device handlers are supplied for standard system peripherals.
Programmed Monitor Commands	Input/Output programming is simplified by the use of a set of system commands which are standardized for system-supported I/O devices.
Wide Addressing Range	Programs can address up to 128K of memory.
Conversational Mode	System Utility Programs interact with the operator/user in a simple, conversational manner.
Dynamic Storage Allocation	The available disk storage is automatically allocated for optimum storage utilization.
Dynamic Buffer Allocation	Input/Output buffers are automatically controlled by the Monitor. It allocates only that space which is required for the system and the user.
Disk File Structure	Allows efficient use of disk capacity and data retrieval for processing via;

- System supported DECdisk, Disk Cartridge and Disk Pack devices, providing both economy and storage capacity
- High data storage capacity (Disk Pack - 83.7 million words, DECdisk - 2.09 million words, Disk Cartridge - 10 million words)
- Random/Sequential File access
- File Protection through unique user directories

- Random Access - formatted as well as unformatted Input/Output (FORTRAN IV)

User-Created System Programs	The user may easily incorporate his own software into the operating system, thereby tailoring the system to his hardware and software needs.
I/O Spooling	Line Printer, card reader, and XY plotter spooling is available on systems with Unichannel-15.
Programming Languages	Several programming languages are offered: FORTRAN IV, FOCAL, MACRO XVM and MAC11 (Unichannel-15 systems)
Bank and Page Modes	Choice of 8K (Bank Mode) or 4K (Page Mode) direct addressability.
Automatic Mode Checking	The system verifies the various mode settings against available hardware.

The system provides for several levels of user file protection. Using unique User Identification Codes, each user can be assured of his file integrity. Files are protected and invisible to other users. The system provides privileged access to all files via a supervisory code maintained by the system owner or manager.

1.2 SYSTEM HARDWARE

The Disk Operating System is defined within the limits of a particular XVM hardware system configuration; i.e., central processor model, minimum and maximum core requirements, necessary features, and types and numbers of peripheral devices.

1.2.1 Minimum Hardware Requirements

For an XVM system[10]:

KP15 Central Processor
24,576 Words of 18 Bit Memory
LA36, LA30C, KSR33, KSR35, or VT05 Console Terminal[2,12]
PC15 High-Speed Paper Tape Reader/Punch
KE15 Extended Arithmetic Element
KW15 Real-Time Clock[11]
XM15-Memory System including;[10]
    Wide Addressing
    Memory Protection and Relocation
    Instruction Prefetch
    Automatic Priority Interrupt
TC15 DECtape Control with 1 TU56 Dual DECtape Transport[4]; or
TC59 Magtape Control with 1 TU10, TU20 or TU30 (7 or 9-track) Magtape Transport
RK15 Cartridge Disk System with RK05 Cartridge Disk Drive and 8,192 Words of 16-Bit Core Memory;[5] or
RP15 Disk Pack Control with one RP02 or RP03[1] Disk Pack Drive; or
RF15 DECdisk Control with two RS09 DECdisk drives.
CR11 Card Reader or CR15 Card Reader or CR03B Card Reader (Required Only to Utilize the BOSS XVM Feature)[9]
LP11 Line Printer or LS11 Line Printer or LV11 Electrostatic Printer; or
LP15 Line Printer (required only to utilize BOSS XVM feature)[9]

For a PDP-15 Computer System:

KP15 Central Processor
24,576 Words of 18-Bit Core Memory
KSR35, KSR33, LA30C, LA36 or VT05 Console Terminal[2]
PC15 High-Speed Paper Tape Reader/Punch
KE15 Extended Arithmetic Element
KW15 Real-Time Clock[11]
KA15 Automatic Priority Interrupt (required only for RK15 systems in certain configurations)[3]
TC15 DECtape control with one TU56 Dual-DECtape Transport[4];;or TC59 Magtape Control with one TU10, TU20 or TU30 (7- or 9-track) Magtape Transport
RF15 DECdisk Control with two RS09 DECdisk drives; or
RP15 Disk Pack Control with all RP02/RP03[1]
Disk Pack Drive; or
RK15 Cartridge Disk System with one RK05 Cartridge Disk Drive and 8,192 Words of 16-Bit Core Memory[5]
CR15 Card Reader or CR11 Card Reader or CR03B Card Reader (required only to utilize the BOSS XVM feature)[9]
LP15 Line Printer or LP11 Line Printer or LS11 Line Printer or LV11 Electrostatic Printer (required only to utilize the BOSS XVM feature)[9]

1.2.2 Optional Hardware

For the XVM:[10]

CR15/CR11/CR03B Card Readers
Core Memory 16-Bit (12K)[6]
Core Memory 18-Bit (ME15, MF15) (32K to 128K)
TC15 4-TU56 Dual DECtape or TC02 8-TU55 DECtape
RF15 8-RS09 Disk
RK15 8-RK05 Disk Cartridge[6]
RP15 8-RP02/RP03 Disk Packs[1]

FP15 Floating Point Processor
VT15 1-VT04 or VT07 Graphics Display
LP11/LP15/LS11/LV11[6] Line Printer
TC59 8-TU10/TU20/TU30 Magtape
XY11/XY311[6] Plotter
VP15A Storage Scope
LA36/LA30C/KSR33/ASR33/KSR35/ASR35/VT05/LK35[2,7,12] Terminal
VW01 Writing Tablet

For the PDP-15:

XM15 Memory Processing Unit including:

- Wide Addressing
- Memory Protection and Relocation
- Instruction Prefetch
- Automatic Priority Interrupt

KA15 Automatic Priority Interrupt
CR03B/CR15/CR11[6] Card Reader
Core Memory 16-Bit (12K)[6]
Core Memory 18-Bit (MM15/MK15/ME15/MF15) (32K to 128K)[8]
TC15 4-TU56 or TC02/8 TU55 DECtape
RF15 8-RS09 Disk
RK15 8-RK05 Disk Cartridge[6]
RP15 8-RP02 RP03 Disk Pack[1]
FP15 Floating Point Processor
VT15 1-VT04 or VT07 Graphics Display
LP15/LP11/LS11/LV11[9] Line Printer
TC59 8-TU10/TU20/TU30 Magtape
XY11/XY311[6] Plotter
VP15A Storage Scope
KSR33/KSR35/ASR35/LA30C/LA36/VT05/LK35[2,7,12] Terminal
VW01 Writing Tablet

NOTES:

1. The RP03 Disk Pack is supported as if it were an RP02 Disk Pack.
2. The LA36 and LA30C DECwriters and the VT05 Video Terminal are supported up to 300 baud.
3. API is required if the user has more than four PDP-11 options which need to interrupt the PDP-15.
4. The older style of DECtape can be used: TC02 Control with 2 TU55 single DECtape transports.
5. The RK15 includes within it an 8,196-word UNICHANNEL-15 peripheral processor. If the spooling software is to be used, the requirement for 16-bit core memory is raised from 8,192 to 12,288 words, if more than two devices are to be spooled.
6. Prerequisite hardware for CR11, LP11, LS11, LV11, XY11, and XY311, plus 16-bit core memory, is the RK15. The RK15 minimally contains a UNICHANNEL-15 with 8K words of 16-bit memory, an RK11E Cartridge Disk Control and 1 RK05 Cartridge Disk Drive. At any given time, only one line printer, one card reader, and one XY plotter in the combined XVM/PDP-11 or PDP-15/PDP-11 system is supported by the software. The LS11 or LV11 printer must be connected to the same vectors/priority as would the LP11.
7. The LK35 Graphic Keyboard is not a substitute for the console terminal; consequently, it also requires an LT15A Single Teletype[1] Control. The paper tape facility on ASR Teletypes is not

[1]Teletype is a registered trademark of the Teletype Corporation.

software supported. The LA30C and LA36 DECwriters and the VT05 Video Terminal can be run up to a baud rate of 300.

8. Memory configuration greater than 32K require the XM-15 options and can utilize ME15 or MF15 memory only.
9. The Card Reader and Line Printer requirements are waivered for those users not desiring BOSS XVM operation.
10. The XM15 hardware option is not compatible with MM-15 memory, KA-15 automatic priority interrupt and KM/KT-15 memory protect and relocate. The XM15 hardware functionally replaces these options.
11. The KW15 real-time clock is recommended for NON-UC15 systems and is required for UC15 systems.
12. The LA36 is supported as an 80 column, upper case only terminal.

1.2.2.1 UNICHANNEL-15 Hardware - The UC15 System, in its standard configuration, consists of the following equipment:

PDP-11 computer

DR15-C Device Interface

Two DR11-C Device Interfaces

Memory Multiplexer

Local memory - up to 12,288 16-bit words of core memory

NOTE

The PDP-11, which functions as the programmable controller, can itself only process 16-bit words but controls peripherals that can process 18-bit words to provide compatibility with the XVM.

The DR15-C and the two DR11-C Device Interfaces provide a communication facility between the XVM and the PDP-11. In the normal mode of operation the XVM interrupts the PDP-11 to send to it commands and data. The PDP-11 processes the commands, accepts the data, and when done, interrupts the XVM to indicate job completion and possible error conditions.

The Memory Multiplexer functions as a memory bus switch to allow either the XVM or the PDP-11 to communicate with the common memory.

The following illustration shows the UC15 hardware configuration.

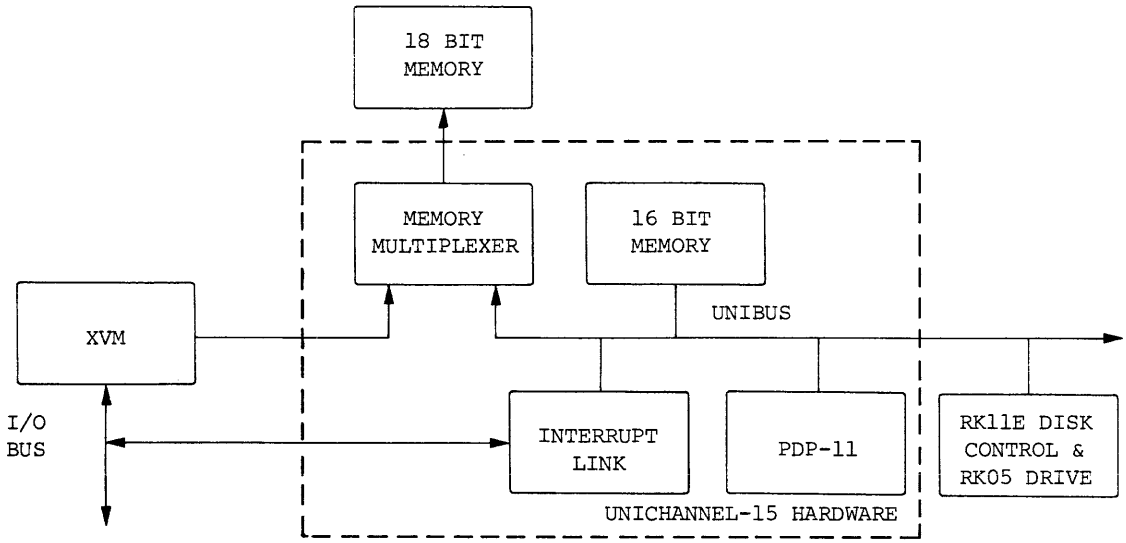


Figure 1-1
Unichannel-15 Hardware

1.2.2.2 CTRL X Feature - The control X feature is available to the user whose system configuration includes a Graphics Processor subsystem. This feature gives the capability of changing from the hard copy output of the teleprinter to the soft copy output of the VT15 Display.

1.2.2.3 Real-Time Clock - The Real-Time Clock runs continuously in order to update an elapsed time register. It can be used to time jobs or to control program execution.

In addition to the above, the user may adapt the system to incorporate many other specialized peripherals, such as X-Y plotter, data acquisition equipment, etc.

1.2.3 The System Device

The system device for DOS may be the RF15 DECdisk, the RK15 Disk Cartridge, the RP02 Disk Pack, or RP03 (supported as an RP02).

The RF15 DECdisk equipment is composed of up to eight fixed-head, rotating disks which are treated by DOS as one contiguous storage area. The DOS Monitor provides for simultaneous use of the DECdisk or Disk Cartridge or Disk Pack as a system device, file device, and scratch device.

1.3 SYSTEM SOFTWARE

The XVM Disk Operating System's service routines perform four primary tasks for all user applications (see Figure 1-2).

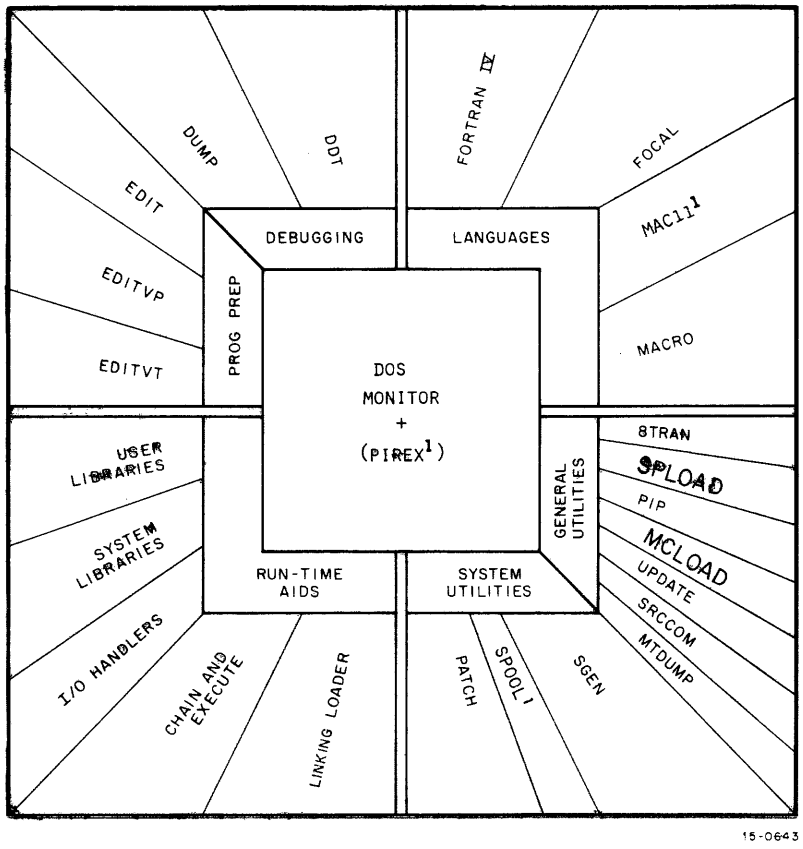


Figure 1-2
XVM Monitor Disk Operating System Software

[1]For UC15 systems only.

1. Program Preparation and Maintenance - There are system programs to aid user program preparation by preparing file text for source programs and programs to aid testing and maintenance of object programs.
2. Language Assembly and Compilation - Programs are available to translate problem-oriented and procedure oriented languages into machine language and to incorporate routines into complete, executable programs.
3. Run-Time Aids - External routines from several libraries are available to the user. The libraries may contain either user-designed routines or those provided by DOS which can be implicitly called.
4. Utilities - DOS provides facilities for efficient storage, flow, and retrieval of system and user data. There are also system programs that provide file verification and data buffer allocation, file data manipulation, etc.

The system software lets the user deal with many complex problems in a simple and straightforward manner. The system will allow the user to perform all of these functions:

1. Write programs in four higher-level languages - FORTRAN IV, FOCAL, MAC11[1] and MACRO.
2. Edit and debug programs prior to run.
3. Load and link programs.
4. Run programs by:
   a. handling I/O,
   b. reading and writing named random-access files on disk storage,
   c. providing run-time device independence,
   d. chaining long programs.
5. Batch processing from paper tape or cards, or a mass storage file.
6. Support full spooling of multiple devices on the UNIBUS.

[1]For UC15 systems only.

Listed in Table 1-1 are individual system programs in the XVM Disk Operating System.

Table 1-1
DOS System Software

System Functions	Program Name	Description
MONITORS	BOSS Monitor	A stand alone batch processing software system. See the BOSS XVM User's Manual.
	DOS Monitor	Allows system parameter changes and device assignments.
	PIREX[1]	A multi-tasking, multi-programming executive that runs in the PDP-11 local memory.
LANGUAGES	FORTRAN IV	Compiler, Object Time System, Science Library.
	FOCAL	An on-line interactive algebraic language.
	MACRO	A symbolic XVM assembler language.
	MAC11[1]	A symbolic PDP-11 assembler language.
PREPARATION AND DEBUGGING	DDT	A Dynamic Debugging Technique for FORTRAN and MACRO programs.
	DUMP	The capability to output specified core locations.
	EDIT	Text Editor providing insertion, deletion, and modification of symbolic text.
	EDITVP & EDITVT	Special versions of EDIT which provide fast soft copy editing on the VT or VP display system.
UTILITIES		
General	PIP	Facilitates the manipulation and transfer of a data file from any input to any output device.
	UPDATE	Binary program retrieval and library update program.
	SRCCOM	Source compare program.

[1]For UC15 system only.

Table 1-1 (Cont)
DOS System Software

System Functions	Program Name	Description
	MTDUMP	Magnetic Tape DUMP program.
	8TRAN	Translates PDP-8 source code into XVM code.
UTILITIES		
System	SGEN	Provides the ability to tailor the system structure to a particular hardware/software configuration.
	PATCH	Makes corrections to systems programs on the systems device, and adds programs to the system.
	MCLOAD[1]	Installs MAC11 paper tapes on the system disk.
	SPLGEN[1]	Creates spooling areas on RK disk cartridges.
	SPLOAD[1]	Installs SPOL11 paper tapes on the system disk.
	SPOOL[1]	Permits automatic temporary storage of low speed data on a high speed storage device.
OPERATING PROGRAMS	Linking Loader	Loads relocatable programs and required routines.
	CHAIN & EXECUTE	Multiple segmentation of large programs and overlays to allow economy of core.

These programs are supervised by the Monitor to form an interactive collection of service programs. This relationship is illustrated in Figure 1-3.

1.3.1 How DOS is Supplied

The media on which the XVM/DOS system is supplied and the steps necessary to build the initial system are described in the XVM/DOS V1A System Installation Guide.

1.4 OPTIONAL SOFTWARE

The software packages, described below, are optionally purchased by the XVM/DOS User.

[1]Unichannel-15 systems only.

1.4.1 DECsystem-10/XVM Communications Link

The DECsystem 10 to XVM Communications Link software provides support for a single hardwired, error correcting, asynchronous data link between a UNICHANNEL system and a DECsystem-10. The communications link is a low speed (2400 baud or less) data link for the users of XVM/DOS the DECsystem-10.

1.4.2 LV11 Electrostatic Printer/Plotter Raster Scan Package

A library of modules allows the user to 'snapshot' a VT-15 display image onto the LV11 Electrostatic Printer/Plotter. An intermediary in this process is a disk file of a standardized graphics format.

A user without a VT15, wishing to use the LV plot capability, may call the VT15 graphics routines without starting the VT15 scope, or he may write data in the format of the intermediary disk file.

CHAPTER 10
OPERATING PROCEDURES

10.1 INTRODUCTION

This chapter provides general operating procedures and considerations to assist novice users in operating in the XVM/DOS Monitor environment. Procedures for system startup and system generation are the subject of Chapter 7 and are not included here. The discussions and examples in this chapter assume a properly configured and running system. It is assumed that the reader is familiar with the keyboard commands, as defined in Chapter 8. Further, the user should be aware of the different types of I/O device handlers and their versions, as described in Chapter 9, and in particular, the tables which list those handlers which can be used with the XVM/DOS System programs.

Specific operating procedures for the system programs (i.e., command strings, options, functions, error messages, and so on) are provided in the various language and utility program manuals listed in the Preface. Once a user gains some understanding of the use of the system programs and their commands from these manuals, he will find that the XVM/DOS Keyboard Command Guide provides most of the information normally required for day-to-day user reference.

Since most procedures involved with keyboard operation are specific to particular programs, very little will be said here about them.

In general, keyboard operations at the Monitor level consist of issuing commands to set up system operation, prior to calling system and user programs. This usually involves at least the LOGIN and LOGOUT commands, and quite often requires the PIP program, as well.

This chapter consists primarily of two large examples of what might be typical XVM/DOS operations. Each example is the result of an actual session at the XVM. On the left side of each page of the examples is the actual teleprinter output. All commands typed by the user are underlined. On the right is a running commentary on all significant teleprinter lines.

10.2 EXAMPLE OF KEYBOARD OPERATIONS

Figure 10-1, Example of XVM/DOS Keyboard Operating Procedures, demonstrates some typical keyboard procedures. The example consists of a series of operations in which a simple FORTRAN program is transferred to the disk from DECtape, edited, compiled, run, and transferred back to DECtape. Figure 10-2 shows the program. The program simply prints the numbers 1 through 10 on the device associated with .DAT slot 4, and then returns control to the Monitor.

10.3 EXAMPLE OF OPERATING PROCEDURES USING COMMAND BATCHING MODE

The example which is contained in Figure 10-3 demcnstrates typical operations using the Monitor's Command Batching Facility. Essentially the same types of operations which can be performed at the keyboard can be accomplished using Command Batching Mode. There are, of course, some operations which, although legal, are more difficult to accomplish, such as editing. To prepare a job for Command Batching Mode, the Programmer creates a file of keyboard commands in the sequence in which they would normally be issued from the teleprinter, interspersed with the special batching commands described in paragraph 8.12 (as applicable). The file can exist on DECtape, magtape or disks, or either on punched cards or paper tape. The programmer then submits the batch file and other required data (DECtapes, Magtapes, etc. for use during the job run) to the computer operator for execution along with appropriate instructions. Users who prepare batching files on punched cards should remember to place an end-of-file card at the end of the card deck. This card is created by multiple punching all punch positions in card column 1 (see 9.3.8.1 d).

Example 10-3 is basically similar to that in Figure 10-1 except that no editing is performed since it is more easily and accurately accomplished by the programmer directly. Similarly, requests for device assignments were also omitted since this information cannot be utilized at runtime in a batching environment.

```
>^C
```

The user types CTRL C to prepare the Monitor for keyboard command input. If no response, the Monitor must be reinitialized using the bootstrap -- see Chapter 7.

```
XVM/DOS Vnxnnn
$DATE
DATE 12/09/75
```

If the Monitor requests a date (indicating it has just been started via the bootstrap) enter it as required. Otherwise check the date (type D↵) to make sure it is current and correct it if it is not.

```
$LOGIN GAR
```

If the user wishes to use disk storage, he must log-in to the Monitor with a UIC of his creation.

```
XVM/DOS Vnxnnn
```

At this point, the user might consider typing other commands such as: VT ON, KEEP ON, PROTECT etc. (refer to Chapter 8).

```
BANK MODE   32K  API ON  UC15 ON  POLLER ON  GAR
$PIP
```

Call PIP

```
PIP XVM Vnxnnn
>L TT_SY

     09-DEC-75
 DIRECTORY LISTING  (GAR)
   1641 FREE BLKS
     12 USER FILES
    235 USER BLKS
EXBD    COM       46  31-OCT-75
EXBD    BIN       11  31-OCT-75
FOCAL   BIN       35  14-NOV-75
FNEW    BIN        2  14-NOV-75
.LIBNF  BIN      111  14-NOV-75
DSK     BAT        2  21-OCT-75
CH3     FCL        1  18-NOV-75
CH1     FCL        1  18-NOV-75
CH2     FCL        1  18-NOV-75
TST     FCL        1  18-NOV-75
>T SY_DT1 FTNTST SRC
```

Request a User File Directory listing to see if there is enough space available for the operations to be performed. (Since there are 1641 blocks indicated, the user can continue. If enough space were not available, however, other files in the UFD would have to be deleted.)

If the user had no previous UFD on the disk, he would create one using the PIP command.

N␣DK↵

Transfer program (FTNTST SRC) from user's DECtape to disk.

Figure 10-1
Example of XVM/DOS Keyboard Operating Procedures

```
>^C

XVM/DOS Vnxnnn

$R EDIT

.DAT     DEVICE  UIC     USE
-15      SYA     GAR     OUTPUT/SCRATCH
-14      SYA     GAR     I/O
-13      SYA     GAR     SECONDARY OUTPUT
-12      LPA     GAR     LISTING
-10      CMA     GAR     SECONDARY INPUT


$EDIT
```

```
EDITOR XVM Vnxnnn
>OPEN FTNTST
EDIT
>P 4
C
        EXAMPLE - SAMPLE FORTRAN TEST PROGRAM
C


>T
>N 2
>P
        EXAMPLE - SAMPLE FORTRAN TEST PROGRAM
>C //C/
C       EXAMPLE - SAMPLE FORTRAN TEST PROGRAM
>CLOSE

EDITOR XVM Vnxnnn
>^C

XVM/DOS Vnxnnn

$R F4

.DAT     DEVICE  UIC     USE
-13      SYA     GAR     OUTPUT
-12      LPA     GAR     LISTING
-11      SYA     GAR     INPUT


$A TT -12
```

Figure 10-1 (Cont.)
Example of XVM/DOS Keyboard Operating Procedures

```
$F4

FPF4M XVM V1A000
>BL_FTNTST


END PASS1
PAGE  001      FTNTST SRC    09-DEC-75    00:20      FPF4M XVM V1A000

0001     C
0002     C        EXAMPLE - SAMPLE FORTRAN TEST PROGRAM
0003     C
0004              DO 1 I=1,10
0005     1        WRITE (4,100) I
0006     100      FORMAT(6X,I3)
0007              STOP 12345
0008              END
PROGRAM SIZE = 00050, NO ERRORS
PROGRAM SIZE = 00050, NO ERRORS
FPF4M XVM V1A000
>^C

XVM/DOS V1A000

$R USER

.DAT     DEVICE   UIC
+1       SYA      GAR
+2       SYA      GAR
+3       SYA      GAR
+4       TTA      GAR
+5       PRA      GAR
                     ^P


$A TT 4

$GLOAD

BLOADER XVM V1A000
>_FTNTST

         1
         2
         3
         4
         5
         6
         7
         8
         9
        10
STOP  012345


XVM/DOS V1A000
```

Figure 10-1 (Cont.)
Example of XVM/DOS Keyboard Operating Procedures

```
$PIP↵

PIP XVM V1A000

>T DT1_SY FTNTST SRC↵

IOPS4 DTA1^R↵



>L TT_SY↵

     09-DEC-75
 DIRECTORY LISTING  (GAR)
   1637 FREE BLKS
     14 USER FILES
    237 USER BLKS
 EXBD    005       46  31-OCT-75
 FTNTST  BIN        1  09-DEC-75
 EXBD    BIN       11  31-OCT-75
 FOCAL   BIN       35  14-NOV-75
 FNEW    BIN        2  14-NOV-75
 .LIBNF  BIN      111  14-NOV-75
 FTNTST  SRC        1  09-DEC-75
 DSK     BAT        2  21-OCT-75
 CH3     ECL        1  18-NOV-75
 CH1     FCL        1  18-NOV-75
 CH2     FCL        1  18-NOV-75
 TST     FCL        1  18-NOV-75

>D SY FTNTST SRC↵

>^C

XVM/DOS V1A000

$LOGOUT↵
```

Figure 10-1 (Cont.)
Example of XVM/DOS Keyboard Operating Procedures

```
0001     C
0002               EXAMPLE - SAMPLE FORTRAN TEST PROGRAM
 >011<
0003     C
0004               DO 1 I=1,10
0005     1         WRITE (4,100) I
0006     100       FORMAT(6X,I3)
0007               STOP 12345
0008               END
```

Figure 10-2
Listing of Sample FORTRAN Program

```
>^C

XVM/DOS Vnxnnn

$B SY <SCR> EXAMPL

XVM/DOS Vnxnnn

$$JOB TEST COMMAND BATCHING MODE
$LOGIN GAR
XVM/DOS Vnxnnn


BANK MODE   32K   API ON   UC15 ON   POLLER ON   GAR

$LOG
      PLEASE MOUNT DECTAPE
      CONTAINING "FTNTST SRC" ON
      DECTAPE UNIT 1 AND SET TO
      WRITE ENABLE.
      THANKS!
$$PAUSE
^R
^R
PIP
PIP XVM Vnxnnn

>T SY_DT1 FTNTST SRC

>
XVM/DOS Vnxnnn

$$JOB
$A TT -12

$F4
FPF4M XVM Vnxnnn
>BL_FTNTST

END PASS1
PAGE  001      FTNTST SRC    09-DEC-75    00:39      FPF4M XVM Vnxnnn

0001     C
0002     C         EXAMPLE - SAMPLE FORTRAN TEST PROGRAM
0003     C
0004               DO 1 I=1,10
0005     1         WRITE (4,100) I
0006     100       FORMAT(6X,I3)
0007               STOP 12345
0008               END
PROGRAM SIZE = 00050, NO ERRORS
PROGRAM SIZE = 00050, NO ERRORS
FPF4M XVM Vnxnnn
>
XVM/DOS Vnxnnn

$$JOB
$A TT 4
```

Figure 10-3
Example of Command Batching Mode

```
$GLOAD
BLOADER XVM Vnxnnn
>_FTNTST

        1
        2
        3
        4
        5
        6
        7
        8
        9
       10
STOP  012345


XVM/DOS Vnxnnn

$$JOB
$PIP
PIP XVM Vnxnnn

>T DT1_SY FTNTST BIN

>D SY FTNTST SRC

>L TT_SY
      09-DEC-75
 DIRECTORY LISTING  (GAR)
   1636 FREE BLKS
     13 USER FILES
    236 USER BLKS
 EXBD    005       46  31-OCT-75
 EXBD    BIN       11  31-OCT-75
 FOCAL   BIN       35  14-NOV-75
 FNEW    BIN        2  14-NOV-75
 .LIBNF  BIN      111  14-NOV-75
 DSK     BAT        2  21-OCT-75
 FTNTST  BIN        1  09-DEC-75
 CH3     FCL        1  18-NOV-75
 CH1     FCL        1  18-NOV-75
 CH2     FCL        1  18-NOV-75
 TST     FCL        1  18-NOV-75

>L TT_DT1
      09-DEC-75
 DIRECTORY LISTING
   1066 FREE BLKS
      2 USER FILES
     10 SYSTEM BLKS
 FTNTST  SRC        2       1
 FTNTST  BIN        1       1
```

Figure 10-3 (Cont.)
Example of Command Batching Mode

```
>
XVM/DOS Vnxnnn

$$JOB
$LOGOUT
XVM/DOS Vnxnnn


BANK MODE   32K  API ON  UC15 ON  POLLER ON  SCR

$$EXIT

XVM/DOS Vnxnnn

$
```

Figure 10-3 (Cont.)
Example of Command Batching Mode

10.4 ERROR DETECTION AND RECOVERY PROCEDURES

All major components of the XVM/DOS Software System contains facilities for error detection and operator notification. This includes the System Programs, I/O Device Handling Routines and the Monitor itself. The operator is notified of the existence of an error condition most generally by a message output to the console teleprinter. There are, however, two situations in which the operator would not receive a message on the teleprinter. The first is the normal occurrence of end-of-file, end-of-medium, parity and checksum error conditions by means of the information fields provided in header word Ø of the affected logical record. Recovery from these errors is not discussed here, since it is the user's program which must recognize these conditions and determine the appropriate corrective action. (Refer to paragraph 6.3.1.1.) The second situation is the occurrence of an undetectable error such as a user's program looping endlessly.

Under these conditions, the user can detect the error condition by observing that the program is not operating as expected. His recourse in this case is to abort the operation by typing CTRL C or CTRL Q.

There are two types of error messages output to the console teleprinter:

- I/O errors which are detected by the I/O device handlers and printed through the Monitor's error message facility. These messages, referred to as "IOPS errors", consist of the mnemonic "IOPS" or "IOPSUC" followed by a number.

- System Program and Monitor detected errors which generally result from syntactically incorrect or illogical command strings or other illegal operating conditions. These messages are generally self-explanatory and consist of an appropriate symbol, word, phrase or sentence.

The degree to which IOPS error recovery is possible can vary with the nature of the error. In some situations, it may be possible to continue the operation from the point where it was interrupted by the error message. For example, the IOPS4 message (I/O Device Not Ready) occurs when an I/O device hardware not ready condition is detected by the handler. To recover, all that is required is that the operator correct the condition (e.g., card reader out of cards) and type CTRL R. In other situations the operator may have to retype a command string correctly, restart the interrupted program (CTRL P) or call in the Monitor (CTRL C) to assign or reassign an I/O device or to reload the program. Some users may wish to save an image of core at the time of the error. In that case, the QDUMP command (see 8.7.2) would be typed before reloading the offending program. If, after typing CTRL C, the Monitor does not start up because a runaway program has destroyed the bootstrap, the bootstrap must be reinitialized or reloaded using procedures described in Chapter 7. In UC15 systems, when an IOPS4 condition arises on UNICHANNEL devices, the operation continues automatically once the condition is corrected, except for the RKØ5 where CTRL R must be typed.

Complete identification, explanation and recovery information for IOPS errors is provided in Appendix D and for IOPSUC errors in Appendix J. Similar information for other error messages is found in the applicable language or utility program manual for the applicable System Program. The XVM/DOS Keyboard Command Guide provides complete command summaries of all XVM/DOS System Program and IOPS error messages.

CHAPTER 2
SYSTEM PROGRAMS

2.1 INTRODUCTION

This chapter gives a brief description of the function of each major program in the system. Each system program will assist the user in performing a particular task in the process of application design and implementation. Considerations for system modification are reserved for the XVM/DOS System Manual and the UNICHANNEL XVM Software Manual.

2.2 CHOICE OF LANGUAGES

User source programs can be implemented at several levels depending on those particular system features required for a given processing environment. DOS supports three levels of automation for object program preparation:

1. Compiler level language - FORTRAN IV
2. Assembly language - MACRO/MAC11[1]
3. Interpretive language - FOCAL

2.2.1 FORTRAN IV Compiler

The FORTRAN IV compiler is a higher-level, procedure-oriented language system that accepts statements written in the FORTRAN IV language and produces a relocatable object program capable of being loaded by the Linking Loader. XVM FORTRAN IV is based on the language of ANSI Standard FORTRAN (X3.9-1966). The system is augmented by the Floating Point Processor, the FORTRAN IV compiler, and an Object Time System.

Object time program capabilities include "floating point" instructions. Subroutines written in either FORTRAN IV or MACRO assembly language can be loaded with and called by FORTRAN IV main programs. Comprehensive source language diagnostics are produced during compilation, and a symbol table is generated for use in on-line debugging with DDT.

---

[1] For UC15 systems only.

The system's Data-Directed Input-Output package permits input or output of ASCII data without reference to a FORMAT statement. The system can also perform memory-to-memory transfers (ENCODE/DECODE) moving data from memory to the I/O Buffer to memory under FORMAT control.

The two FORTRAN IV compilers supported by XVM/DOS are: (1) F4M and (2) FPF4M (Floating Point mode). Each has its own Object Time System and Science Library so that program routines may utilize all system hardware and software features.

A FORTRAN IV program may be compiled and run in several different equipment environments. The FORTRAN programmer need not be too concerned with the details of his environment since the FORTRAN Object-Time System (OTS) will ensure that his source statements generate the appropriate computer instructions. For example, an arithmetic statement such as A=A*B will appear the same in any FORTRAN IV program. In the object program it may be transformed to a subroutine call or a floating point instruction, depending on the hardware configuration on which the program is produced. FORTRAN data-transmission statements automatically call a number of OTS subroutines which serve as an interface between the user program and the Monitor. These routines may also be called from MACRO assembly language programs. Further, programs written in FORTRAN IV can be linked to programs or routines written in the MACRO assembly language.

For more information concerning this higher-level programming language, refer to the FORTRAN IV XVM Language Manual and the FORTRAN IV XVM Operating Environment Manual.

2.2.2 MACRO XVM Assembler

This MACRO Assembler provides users with macro generating and calling facilities within the context of a symbolic assembler. Some of the prominent features of MACRO include:

1. The ability to:
   - (a) define macros,
   - (b) define macros within macros (nesting),
   - (c) redefine macros (in or out of macro definitions),
   - (d) call macros within macro definitions,
   - (e) have macros call themselves (recursion).

2. Conditional assembly based on the computational results of symbols or expressions.
3. Repeat functions.
4. Boolean manipulation.
5. Optional octal, symbolic, and cross-reference listings.
6. Two forms of radix control (octal, decimal) and two text modes (ASCII and 6-bit trimmed ASCII).
7. Global symbols for easy linking of separately assembled or compiled programs.
8. Choice of output format: relocatable, absolute binary (checksummed), or full binary -- capable of being loaded via the hardware READIN switch.
9. Ability to utilize user-designed input/output macros.
10. A Table of Contents option containing the page numbers and text of all assembled .TITLE statements in the program.
11. The ability to combine up to three input files for one assembly.

MACRO permits the programmer to use mnemonic symbols to represent instruction operation codes, locations, and numeric quantities. It is essentially a comprehensive macro instruction generator. This generator permits easy handling of recursive instruction sequences, changing only the arguments.

The assembler facilitates the development of instructions called "macros" which, when used as a source statement, can cause a specific sequence of instructions to be generated in the object program. Refer to the MACRO XVM Assembler Language Manual for a complete description of the language.

The standard object code produced by MACRO is in a relocatable format which is acceptable to the Disk Operating System's Linking Loader utility program. Relocatable programs that are assembled separately and use identical global symbols[1] where applicable, can be combined by the Linking Loader into an executable program.

[1]Symbols which are referenced in one program and defined in another.

An output listing, showing both the programmer's source coding and the binary object program produced by MACRO, is printed if desired.

This listing includes all the symbols used by the programmer with their assigned values. If assembly errors are detected, erroneous lines are marked with specific letter error codes.

2.2.3 MAC11 Assembler[1]

The MAC11 Assembler provides the user of the UNICHANNEL-15 system with the capability of assembling the full repertoire of the PDP-11 instruction set. Besides making the UC15 system self-sufficient it provides the users with operating and calling facilities within the context of a symbolic assembler. Some notable features of MAC11 are:

1. The ability to:
   (a) define macros,
   (b) define macros within macros (nesting),
   (c) redefine macros (in or out of macro definition)
   (d) call macros within macro definitions
   (e) provide alternate exit points from macros (particularly nested macros)
   (f) pass arguments (numeric and non-numeric) and compute the number of arguments passed
   (g) provide a built-in error reporting capability in macros
2. create automatic local labels
3. concatenate strings
4. generate indefinite and definite repeat blocks
5. conditional assembly based on the computational results of symbols or expressions
6. two forms of radix control (octal, decimal) and three text modes (ASCII, ASCIZ and 6 bit ASCII)
7. ability to utilize user designed input/output macros
8. use local and non-local labels
9. provide a table of contents containing the page numbers, text of all assembled .TITLE statements in the program and the line numbers of an output listing.

[1] Only for UC15 systems.

MAC11 permits the programmer to use mnemonic symbols to represent instruction operation codes, locations, and numeric quantities. It is essentially a comprehensive macro instruction generator. This generator permits easy handling of recursive instruction sequences, changing only the arguments.

The assembler facilitates the development of instructions called "macros" which, when used as a source statement, can cause a specific sequence of instructions to be generated in the object program. Refer to the MAC11 XVM Language Manual for a complete description of the language.

The only object code produced by MAC11 is in an absolute format on papertape.

An output listing, showing both the programmer's source coding and the binary object program produced by MAC11, is printed if desired. This listing includes all the symbols used by the programmer with their assigned values. If assembly errors are detected, erroneous lines are marked with specific letter error codes.

2.2.4 FOCAL Interpreter

FOCAL (Formulating On-Line Calculations in Algebraic Language) operates in on-line conversational mode, using natural language and arithmetic terms to establish a simplified environment for the computer-aided solution of business and scientific arithmetic problems. Included in FOCAL are such features as:

1. Linkage to assembly language (MACRO) routines to establish a user library of commonly used functions.
2. Use of COMMON to facilitate chaining in the same manner as FORTRAN IV.

With FOCAL, the user can generate mathematical models, plot curves, solve sets of simultaneous equations in n-dimensional arrays, and do much more. Refer to the FOCAL XVM Language Manual for a complete description of this program.

FOCAL library commands allow the user to save and then call programs by name. These commands result in files consistent with the DOS file format. Such files can be manipulated by other DOS programs, such as

PIP and EDITOR. FOCAL has commands which allow the segmentation (chaining) of FOCAL programs.

The ability to write FOCAL functions in MACRO assembly language and subsequently interface these functions with the FOCAL interpreter is an important feature. These functions are processed in the same way as the normal internal functions which DEC supplies with the interpreter.

2.3 SYSTEM GENERATOR (SGEN)

The System Generator (SGEN) is a standard DOS Utility program used to modify disk resident system files. SGEN, which is available only to the system manager via MIC protection enables him to tailor his system to add to the supplied software in order to develop a resident software system unique to the installation or to his specific needs. The System Manager calls the system generator program via the Monitor command "SGEN". When SGEN is loaded, it initiates an interactive question/answer sequence regarding the following system functions and parameters:

1. The size of the ↑Q area,
2. Hardware options,
3. Type of printer unit used,
4. Required device handler designations (i.e., the standard I/O configuration the user wants for the system programs),
5. Skip-Chain information - Priority Interrupt Skip Chain contents and order,
6. Default assumptions, including: teleprinter characteristics, Bank or Page Mode, use of XVM mode, etc.,
7. System memory size,
8. .DAT slot assignments,
9. Monitor Identification Code to designate the privileged access by the System Manager,
10. Default buffers that are needed at any time during a user program,
11. Default Files Protection Code.

Careful planning is necessary to ensure that the most efficient system will be developed for the user's particular needs. For more information, refer to the SGEN XVM Utility Manual.

2.4 PATCH UTILITY PROGRAM

PATCH is used to: (1) make corrections to the binary version of many system programs on the system device, (2) examine and change any word in any disk or DECtape block, or (3) convert relocatable binary programs into system programs.

Facilities provide for:

1. The selection, examination and alteration of registers within DOS System programs, and any data word block on mass storage, including the system information blocks, SYSBLK and COMBLK;
2. The installation of suitable relocatable programs into a user system as non-relocatable System programs;
3. The loading of absolute programs into a user system as System programs.

The PATCH user must first be logged in under the Monitor Identification Code (MIC) to have access to system files. Binary programs which are not in system program format (e.g., relocatable link-loadable programs, and XCT programs which are executable files built by the System program CHAIN) cannot readily be corrected by using PATCH.

With PATCH the user can:

1. Select a system program to be patched.
2. Select a single block to be patched.
3. Obtain an octal printout of the contents of a particular location in a program.
4. Alter the content of the listed location by simply typing the desired content in octal.
5. Use the READ command to either replace or patch a system program. This enables the user to easily make corrected copy available for instant use without requiring reassembly, regeneration of a system or core patching. This is most useful for handling small updates or new versions of a program.
6. Select and open specific word locations within SYSBLK or COMBLK.
7. Select and examine registers within a system ↑Q area.

8. Automatically convert relocatable binary files into system program format and load the converted file onto the system device, provided disk space has been reserved by SGEN. This feature permits user programs to be called directly from the Monitor. It also enables the program to completely overlay its loader, to make the most effective use of core storage.

For more information concerning this System Utility program, refer to the PATCH XVM Utility Manual.

2.5 CHAIN AND EXECUTE PROGRAMS

The programs CHAIN and EXECUTE allow the user to segment programs in order to construct and run a system of core overlays in an easy and straightforward manner.

CHAIN reserves portions of user core (called COMMON blocks) from one segment to another so that the program segments can communicate. The FORTRAN IV compiler and the MACRO assembler can reserve COMMON blocks for future segmentation. This method of segmentation permits multiple overlays of executable code, constants, variables, arrays, and labeled COMMON blocks.

Both system programs are required for segmentation:

1. CHAIN - processes a version of the Linking Loader Code (Object Program code) allowing the user to build all the various segments (or chains) of his program into an absolute executable (XCT) type file.
2. EXECUTE - a control program which initiates loading of an executable file and transfers control from one chain segment to another. EXECUTE uses handlers with .TRAN and .FSTAT capabilities to improve initial and overlay loading performance. CHAIN organizes subroutines into units called LINKS, which may overlay each other. Several LINKS may overlay a larger LINK without overlaying each other. A LINK is loaded into core when a subroutine within the LINK is called and remains resident until overlayed. A LINK's core image is not recorded or "swapped out" when it is overlayed. The same image is brought into core each time a LINK is loaded. For maximum run-time efficiency, segments must be processed serially. See the CHAIN XVM/EXECUTE XVM UTILITY Manual for detailed instructions.

2.5.1 Advantages/Disadvantages of CHAIN & EXECUTE

Advantages

1. CHAIN
   a. Can build an operable program whose core requirement is larger than that of the run-time machine;
   b. Can be used to create elaborate overlay structures;
   c. Allows the user to request a detailed load map;
2. EXECUTE
   a. Its loader is more core efficient than the Linking Loader since Linking Loader code is processed only once;
   b. It is smaller at load time than the Linking Loader;
   c. The response time for loading and starting a user program is faster with EXECUTE then with the Linking Loader.

Disadvantages

1. One additional step is required to process Object Code in preparation for run-time.
2. DDT (Dynamic Debugging Technique) cannot be used.
3. At run-time, the execution of a segmented file requires a certain amount of processing overhead.

2.6 LINKING LOADER

The Linking Loader loads any FORTRAN IV or MACRO object program which exists in relocatable format. Its tasks include relocation of programs, loading of called external subroutines, retrieval and loading of implied subroutines, and building and relocation of the necessary symbol tables. See the Linking Loader XVM Utility Manual for detailed instructions.

The loader first loads all the named programs included in the keyboard command string. It then additionally loads and links all requested library subprograms. The handlers are loaded from the device handler directory (IOS), the requested library subprograms are loaded from the user library, if one exists, and the system library (found in BNK or PAG). In addition, the loader can type out a core map which specifies the name and address of each program, subprogram, library routine, .GLOBL and common block loaded.

2.7 DYNAMIC DEBUGGING TECHNIQUE (DDT) PROGRAM

DDT provides on-line debugging facilities that enable the user to load and operate his program in a real-time environment while maintaining strict control over each program section. DDT allows the operator to insert and delete breakpoints, examine and change registers, patch programs, and search for specific constants or word formats.

A breakpoint halts operation when the program flow arrives at the designated location. The DDT breakpoint feature allows the insertion and simultaneous use of up to four breakpoints. The search facility allows the operator to specify a search through any part or all of an object program with a printout of all locations whose contents are equal (or unequal) to a specified value. This search feature also works for portions of words, as modified by a mask. With DDT, registers may be examined and modified in either instruction format or octal code, and addresses may be specified in symbolic relative, octal relative, or octal absolute. Patches may be inserted in either MACRO source language or octal. For more information, refer to the DDT XVM Utility Manual.

2.8 DUMP PROGRAM

DUMP gives the user the ability to output, on any device specified, core locations that have been preserved on disk via the CTRL Q (↑Q) Monitor command. It also provides the ability to dump DECtape or disk blocks onto any device. For more information refer to the XVM/DOS Keyboard Command Guide.

2.9 MAGNETIC TAPE DUMP (MTDUMP) UTILITY PROGRAM

The MTDUMP program provides the user who employs magnetic tape as a storage medium with the ability to view and manipulate any named portion (i.e., file) or specified record of a tape.

Some of the features provided by MTDUMP are:

a. Files may be output (dumped) onto any system device in any of four possible formats.

b. Comments may be inserted into a DUMP file

c. Files may be copied onto another tape.

d. Magtape directories can be listed and cleared.

e. Magtapes can be compared.

For more information, refer to the MTDUMP XVM Utility Manual.

2.10 TEXT EDITOR PROGRAMS, EDIT, EDITVP AND EDITVT

The Text Editor provides the ability to read alphanumeric text from paper tape, Disk Cartridge, DECdisk, Disk Pack, DECtape, etc.

The user can then examine and correct the text, writing it back on paper tape, Disk Cartridge, Disk Pack, DECdisk, and DECtape devices. Programmers can also use the Text Editor to create new symbolic programs.

The Editor operates on lines of symbolic text delimited by carriage return (CR) or ALT MODE characters. These lines can be read into a buffer, selectively examined, moved, deleted, or modified, and written out. New text may be substituted, inserted, or appended.

The programs EDITVT and EDITVP are similar to EDIT except that they permit text to be displayed on the VT15 Graphic Display and VP15A (CRT). Refer to the EDIT/EDITVP/EDITVT XVM Utility Manual.

2.11 PERIPHERAL INTERCHANGE PROGRAM (PIP)

With exceptions, PIP can transfer data files from any input device to any output device. It can be used to:

(1) refresh file directories of disk or DECtape,
(2) list file directory contents of disk or DECtape,
(3) delete, insert, segment, or combine files,
(4) perform code conversions,
(5) assign protection codes,
(6) transfer files, or
(7) copy the entire contents of disk and DECtape storage units.

Refer to the PIP XVM Utility Manual.

2.12 LIBRARY UPDATE PROGRAM

The system program gives the user the capability to examine, extract, and update the binary library files on mass storage devices. For more information, refer to the UPDATE XVM Utility Manual.

2.13 SOURCE COMPARE PROGRAM (SRCCOM)

The SRCCOM program compares any two symbolic source programs (ASCII) and indicates their differences. This program is useful for program identification and/or verification, proofing an edited program, comparison of old and new versions of the same program, etc. For more information, refer to the SRCCOM XVM Utility Manual.

2.14 VT15 XVM GRAPHICS SOFTWARE

The GRAPHIC15 (VT15) consists of system I/O device handlers and the graphics run-time system. The run-time system provides display file construction, interaction with light-pen and push-buttons, and storage and retrieval of display images. The run-time system can be LOAD'ed or CHAIN'ed with MACRO or FORTRAN applications programs. Display files can be constructed in memory above 32K. For further information refer to the VT15 XVM Graphics Software Manual.

2.15 PDP-8 TO XVM TRANSLATOR (8TRAN)

This program is used as an aid in translating programs written in the assembly languages of the Digital PDP-8 computer (PAL III, MACRO-8) into MACRO XVM form. The translator does not produce an executable

program, but translates a major portion of the PDP-8 code into equivalent MACRO XVM code and indicates those areas of the program which must be reviewed and processed by the programmer. For more information see the 8TRAN XVM Utility Manual.

2.16 VP15A GRAPHICS SOFTWARE

The VP15A Graphics Software package consists of a group of routines which can be used with either FORTRAN IV or MACRO programs to operate the VP15A Storage Tube Display. Included in the package are an I/O device handler, text, point-plotting, and other routines, all described in the VP15A XVM Graphics Software Manual.

2.17 SPOOLER DISK AREA GENERATION[1] (SPLGEN)

SPLGEN allows the user to dynamically create or alter the RK disk area used by the UC15 spooler on any RK disk unit (0 through 7).

2.18 SPOOLER INSTALLATION PROGRAM[1] (SPLOAD)

SPLOAD allows the user to install, on the system disk, the SPOL11 paper tape produced by MAC11.

2.19 SPOOLER CONTROL PROGRAM[1] (SPOOL)

SPOOL is used to initiate or terminate UNICHANNEL spooling using any RK disk unit which has been previously prepared for spooling by SPLGEN.

2.20 BATCH OPERATING SOFTWARE SYSTEM (BOSS)

BOSS is a dedicated (stand-alone) batch processing software system. BOSS is disk-resident with a set of programs supervised by the BOSS monitor. The monitor tailors the system configuration according to the BOSS command language cards, and then supervises operation of the system or user programs requested by control cards. BOSS has a unique and versatile job control language which can be modified by the user and tailored to his specific needs. For more information, see the BOSS XVM Users Manual.

---

[1] For UC15 systems only.

2.21 MAC11 INSTALLATION PROGRAM (MCLOAD)

MCLOAD allows the user to install, on the system disk, the MAC11 paper tape produced during system installation.

CHAPTER 3
SYSTEM CONCEPTS

3.1 XVM/DOS MONITORING FUNCTIONS[1]

There are three sections to the DOS Monitor: (1) the Resident Monitor, (2) the Nonresident Monitor, and (3) the System Loader. For a UC15 system, there is a fourth section, PIREX, that runs on the PDP-11.

The Resident Monitor remains in core when system or user programs are running, and acts as the interface between the program and the system's facilities. PIREX is always core resident in the UC15 system. It acts as a communication link between the PDP-11 and the XVM in addition to performing other operations like spooling. (For more information refer to the XVM UNICHANNEL Software Manual. During program operation, the Resident Monitor has general control over the system. It functions to:

1. Maintain orderly program flow,
2. Handle teleprinter I/O,
3. Act on Monitor service calls,
4. Validate and transmit I/O calls to device handlers,
5. Announce error diagnostics.

The operator may alter the structure of the Resident Monitor via commands to the Nonresident Monitor. The Nonresident Monitor allows the operator to interrogate and alter many key parts of the system, in order to set up the system for the next program. It functions to

1. Set I/O conditions by assigning physical devices to logical unit numbers,
2. Supply system information,
3. Save or restore core images,
4. Load and Execute system and user programs,
5. Change default system parameters.

Normally, at the end of execution of a particular program, the operator, the Batching Command String, or the program itself returns control to the Nonresident Monitor. At that point, the operator or the

[1] BOSS is described in the BOSS XVM Users Manual.

Batching Command String sets up the system for the next program and calls it in via commands to the Nonresident Monitor.

The System Loader .SYSLD loads

(a) all core-image system programs and all I/O handlers for those system programs,
(b) the Linking Loader, and
(c) EXECUTE

In almost all cases, a change of program involves actions by the System Loader. The System Loader, however, is completely invisible to the user except for .SYSLD errors, such as insufficient core.

The XVM/DOS Software System provides an interface between the system - or user-created program and the external world of I/O devices. This simplifies I/O programming. This interface is comprised of a functionally related group of software called the Input/Output Programming System, or simply IOPS. This is a conceptual term which encompasses

(1) the I/O device handling routines within DOS,
(2) a portion of the Monitor which is used in dispatching I/O commands to them, and
(3) a Monitor routine for printing error messages.

I/O device handlers are provided for all standard devices (see Chapter 9). These handlers relieve the user of the burden of I/O service, file management, overlapping I/O considerations and unwanted device dependence. I/O commands and data modes are standardized and are recognized by DOS device handlers. This facilitates device independence. For example, a non-directoried device handler such as for the paper tape reader will ignore (rather than declare an error) a command to "seek a file" (which is required for directoried devices prior to issuing commands to read). There are other features of the system which contribute to device independence; they are discussed in later paragraphs.

3.1.1 System Communication Table (SCOM)

The System Communication Table (SCOM) is a set of registers that are referenced by the Monitor, I/O device handlers, and other system programs. It acts as a common parameter area for information required by both the System Loader and Monitor. The System Communication Table begins at absolute location 100_8 (Bank 0).

The following list briefly outlines some of the SCOM table functions:

Free Core Limits
Hardware Option Availability
System and User Program Start Addresses
Handlers
Interrupt Levels
Number of Buffers Allocated
Number of Words/Buffer
Number of Entries in Mass Storage Busy Table
User Identification Code
Software Control Switches
Date (MMDDYY)
Time (HHMMSS) and other Clock Information
Default Protection Code for Files

A complete list of the SCOM table functions is given in the XVM/DOS System Manual. User programs may use SCOM locations following the conventions and techniques outlined in the XVM/DOS System Manual.

3.1.2 Monitor/User Interaction

The console teleprinter is the primary user-system interface for DOS program control. This control is implemented by commands to the Monitor, which accepts the three types described below:

1. Commands which perform special services
2. Commands which load system programs
3. Control character commands which provide system control while running user or system programs.

NOTE

In the context of this manual, the term "console keyboard" designates any one of several keyboard/printer/display I/O devices which could be used by the Monitor as the system command console device.

The operator at the keyboard types commands to allocate system resources, load and start system and user programs, terminate program operation, and exchange information with the Monitor. Most of the Monitor's keyboard commands are issued prior to loading programs and are interpreted by the Nonresident Monitor, since it is not resident in core during system or user program execution. During program execution, a small set of keyboard commands is available for general

program control. These commands are interpreted by the teleprinter's I/O device handler (which is part of the resident portion of the Monitor), and are used to control program start and restart, dumping of core, and the reloading of the Nonresident Monitor. Details on the DOS commands which can be issued from the console keyboard are described in Chapter 8.

The keyboard commands are, however, not strictly limited to input from the keyboard. The Monitor can be operated in a Command Batching Mode in which keyboard commands can be issued either from punched cards or paper tape or mass-storage files with minimum operator intervention. Similarly, the Monitor's responses to commands are not strictly limited to a keyboard device's printer or display, but may also be output to other devices including the VT15 Display or a line printer, when available.

3.2 I/O COMMUNICATION

The Monitor, by means of Device Handlers and Priority Interrupt (PI) or optional Automatic Priority Interrupt (API), permits simultaneous operation of I/O devices along with overlapping computations.

A system or user program initiates an I/O function by means of a Monitor command (system macro), which is interpreted within the Monitor as a legitimate I/O call. The I/O call includes a logical I/O device number as one of its arguments. The Monitor establishes the logical/physical I/O device association by means of a special table. When this has been accomplished, the Monitor passes control to the appropriate device handler to initiate the I/O function, after which control is returned to the system or user program. The system or user program retains control until an interrupt (PI or API) occurs; at this time the device handler takes control, in order to perform and/or complete the specified I/O function. The program may continue computation or other processing while waiting for I/O completion. This feature allows the programmer to make optimum use of available time.

3.2.1 Device Independence

In the DOS Monitor environment, the system manager may set up default device associations (correspondence between logical/physical devices) during system generation. Just prior to loading a system or user

program, a user may change these associations via the ASSIGN keyboard command. This capability adds true device independence to DOS.

All device handlers are nonresident in the sense that only those handlers required by a program are loaded into core.

3.2.2 I/O Device Handlers

Users are spared the task of writing system software to handle input/output to all standard system peripherals, since appropriate routines (Device Handlers) are supplied with the DOS Monitor. These routines process file and data level commands to the peripheral device. They generally perform the following functions:

1. Drive I/O devices,
2. Block Data Records for Devices, if necessary
3. Manipulate files,
4. Optimize device timing,
5. Allocate/Deallocate Storage Space on the device,
6. Request/Return core space.

All communication between user programs and I/O device handlers is made via I/O Macros. The FORTRAN user need not be concerned with I/O Macros; they are generated by the FORTRAN Object Time System (OTS) routines which process the standard FORTRAN READ and WRITE statements at run time. Macros are covered in Chapters 5 and 6.

There may be available in the system several handler versions for a particular device (e.g., DKA, DKB, DKC). Each represents different compromises between core use and handler flexibility. Device handlers are covered in depth in Chapter 9.

3.2.3 Device Assignment Table (.DAT)

The DOS Monitor contains a Device Assignment Table (.DAT), with an entry for each device used. Since the contents of the table can be altered by commands to the Monitor, actual I/O devices may be changed without altering the program references (logical device units) to these devices. Refer to Chapter 4 for more information concerning .DAT.

3.3 FILE STRUCTURES

A file structure, as defined for DOS, is a method of recording, linking and cataloging data files. Each peripheral device has an associated file structure which governs the manner in which data are stored.

Card files and paper tape files are always organized as sequential files and both files and records are processed sequentially. A file or record in the middle of the medium can only be accessed after all preceding items have been processed. This is a restriction which is a consequence of the nature of the storage medium. In contrast, DOS provides direct access to files stored on DECtape or disk. The system maintains directories on these devices that point to each file on the device. Hence, such devices are called "directoried" or "file-oriented". Both DECtape and magnetic tape (magtape) permit the user to operate either in a directoried or a non-directoried (sequential) mode. The system maintains a mini-directory for each file which points to each physical block in the file.

The user of the XVM Disk Operating System is not required to preallocate file storage; the operating system provides the file storage space dynamically on demand. Not only is this convenient for the user because he does not have to worry about allocation when he is creating files, but it conserves storage by preventing large portions of storage from being unnecessarily tied up. More information on this topic can be found in Chapter 4.

3.3.1 User File Directories and UIC's

On DECtape, there is only one directory for the whole tape. On disk, there is a central directory, called the Master File Directory (MFD), but each user can have his own User File Directory (UFD). The MFD points to each UFD. Each UFD is named by a unique three character User Identification Code (UIC). The User File Directory Table (.UFDT) is part of the Resident Monitor associated with the Device Assignment Table. It indicates the User Identification Code (UIC) associated with every .DAT slot (i.e., each logical device). Disk I/O to a particular .DAT slot will go to files in the UFD named by the corresponding .UFDT slot.

The Monitor finds User File Directories by seeking associated User Identification Codes (UIC's), which are all listed in the Master File Directory. The UIC is necessary for all directory-oriented I/O to the disk. A programmer may identify himself (LOGIN) to the system with only one UIC at a time, but he may have as many UIC's as he wishes, provided each is unique and none is reserved[1]. Further, programs may simultaneously reference files under several different UIC's.

3.3.2 Monitor Identification Code (MIC)

A three-character Monitor Identification Code (MIC) is established at system generation time to provide privileged access by the System Manager. When the System Manager uses the MIC, all system and user files are open for reference or change. This code acts as a key to temporarily remove file protection. System generation, modification of system programs and listings of the entire disk directory are allowed only under the Monitor Identification Code.

3.4 FILE PROTECTION

DOS offers a simplified form of file protection. Each User File Directory has a protection code (optionally specified in commands to PIP), and each file has a protection code (optionally specified in the .ENTER command or a command to PIP). The protection codes are in effect only when a user tries to reference a file listed under a UIC other than the one currently logged into the system. If a User File Directory is protected, then the protection is provided for any file in the directory. For more information, refer to Chapter 6.

The default protection code for the files is established at system generation time. Users may temporarily change the file default protection code via the PROTECT command to the Monitor. (See Chapter 8.)

3.5 I/O BUFFERS

Two Monitor commands allow any handler or user program to call the Monitor to allocate and deallocate buffers. Each buffer is obtained from a "buffer pool" as needed. For more information, refer to Chapter 6.

---

[1]The LOGIN keyboard command is described in Chapter 8.

3.6 CHOICE OF EXECUTABLE FORM

The object code usually produced by FORTRAN IV and MACRO is relocatable binary which is made absolute by CHAIN, or loaded at run time by the Linking Loader. In addition to relocatable output, the MACRO user may specify non-relocatable types of output code. In a UC15 system, the user can produce absolute binary papertapes from MAC11. No direct loading facility in the form of a loader is provided with it, (the user can write his own simple loader program) although PIREX has execution facilities. See the XVM UNICHANNEL Software Manual for details of PDP-11 program development.

3.6.1 Relocatable Binary

A relocatable object program may be loaded into any part of memory, regardless of which locations are assigned at assembly or compile time. To accomplish this, the address portion of some instructions must have a relocation constant added to it. This relocation constant is added during segmentation, by CHAIN, or at load time by the Linking Loader. It is equal to the difference between the actual memory location into which an instruction is loaded and the location that was assigned to it at assembly time. The language processors use codes to identify storage words as relocatable, absolute, or external.

Relocatable Binary object programs may be loaded into and executed from any part of memory, regardless of the core locations assigned to the instructions by the language translator. The primary advantage of such a system is that it enables the user to easily write and load into core memory many programs (e.g., the main program and several subroutines) with no necessity for prior core mapping. Relocatable Binary is always produced by FORTRAN and is produced by MACRO when there are no .ABS or .FULL pseudo-ops.

3.6.2 Absolute Binary Forms

Absolute Binary object code is produced using the MACRO Assembler's .ABS, .ABSP, .FULL, or .FULLP pseudo-ops as described below. Programs assembled with these pseudo-ops are loaded and executed independently of the DOS System software. Refer to the MACRO XVM Assembler Language Manual for more information. MAC11 produces only an absolute format). format (this format is different from the MACRO XVM absolute format). Refer to the MAC11 XVM Assembler Language Manual.

3.6.2.1 .ABS and .ABSP Binary - The .ABS and .ABSP binary forms are checksummed binary coded programs or instructions assembled as .ABS (Absolute Binary) and assigned to occupy specific or absolute locations in core memory. These programs can only be loaded into or executed from the locations assigned by the language translator. The task of placing multiple routines into core for execution becomes a tedious one. Absolute binary does have the advantage of a smaller loader, thus enabling the user to execute a larger program than is possible using relocatable binary. Ordinarily, the Assembler will precede the output with an Absolute Binary Loader which will load the punched output at object time. The loader is itself loaded via Hardware Read-in Mode.

3.6.2.2 .FULL and .FULLP Binary - The .FULL and .FULLP binary forms are unchecksummed binary consisting solely of 18-bit storage words. Programs assembled in this form are output on paper tape and are loaded via Hardware Read-in Mode.

3.7 LOADER CONTROL

As indicated in the previous section, CHAIN and the Linking Loader make relocatable object programs absolutely addressed. In addition, they join relocatable programs by supplying definitions for global symbols which are referenced in one program and defined in another.

3.7.1 Globals

The global feature is another of the programmer conveniences provided by the MACRO Assembler. It allows the user to provide a symbolic linkage between separately assembled programs including: a main program, subprograms, and general subroutines in system (Bank or Page) Libraries or User Libraries.

In MACRO, the pseudo-op .GLOBL, followed by a list of symbols, is used to define to the Assembler two types of global symbols:

1. Internal Globals - defined in the current program and referenced by other programs.
2. External Symbols - referenced in the current program and defined in another program. Each external symbol will be used by the Linking Loader or CHAIN to store the actual address.

All references to external symbols should be indirect references, because memory banks may have to be crossed.

The FORTRAN equivalents to external and internal .GLOBL's are CALL and SUBROUTINE. The Linking Loader and CHAIN use this information to relocate and then link the programs to each other.

3.7.2 Program Loading

At program load time, the user, via appropriate keyboard commands, can select either Page or Bank Mode program loading and execution. Two versions of the Linking Loader and System Library are provided - one for each mode.

3.7.2.1 XVM Mode - User programs may run in XVM mode which permits indirect addressing of memory beyond 32K.

3.7.2.2 Page Mode Operation - The DOS Monitor, when operating in Page mode, loads and relocates user programs in 4K pages and permits address modification via the index register (Index Addressing). In Page Mode, the loader obtains library routines from the library in the PAG System file directory.

3.7.2.3 Bank Mode Operation - Running DOS in Bank mode permits direct addressing within 8K banks, but does not permit the use of the index register for address modification. All core-image system programs run in Bank Mode. In Bank Mode, the loader obtains library routines from the library in the BNK System file directory.

3.8 ERROR DETECTION

Comprehensive error checking and recovery are provided by the DOS Monitor, the loaders, and the I/O system as follows:

a. DOS Monitor Errors - error conditions related to system devices, illegal device assignment, program name, and command references.

b. IOPS Errors - all Input/Output device and data errors.

c. Linking and System Loader Errors - memory overflow, input data errors, unresolved globals, and illegal .DAT Slot requests.

Detailed lists of errors that occur in the latter two categories can be found in Appendices D and E, respectively.

The automatic core-dump commands QDUMP or CTRL Q will condition the Monitor to dump memory on the "save" or "CTRL Q" area of the system device, in the event of an unrecoverable error.

Terminal errors are not reported to the user's operating programs; there are, however, a few I/O-detected errors that are reported to the user for program use, e.g., parity, checksum, and buffer overflow errors are indicated in special control words of each IOPS data record (see Chapter 6). Error detection and recovery are discussed in Chapter 10.

After error messages are output, the user may optionally restart the program (CTRL P), dump core (CTRL Q), or return control to the Non-resident Monitor (CTRL C). Other options are available to the user when errors occur; these will be discussed in succeeding chapters of this manual.

3.9 INPUT/OUTPUT SPOOLING

The UNICHANNEL-15 configurations allow spooling of tasks using PDP-11 peripherals and memory. The actual spooling program (SPOL11) resides in PDP-11 (local) memory while a special interface program for the spooling function (SPOOL) resides in XVM (common) memory. Both of these spooling program modules are resident on the system disk.

For further information regarding spooling, refer to the XVM UNICHANNEL Software Manual.

The UNICHANNEL-15 system supports full, input spooling from the Card Reader (CR11), and, output spooling to the Line Printer (LP11/LS11/LV11) and plotter (XY11/XY311) utilizing any RK05 disk cartridge. The actual spooling program is resident in the PDP-11 (local) memory when spooling is enabled. It is dynamically connected or disconnected with PIREX at run time by SPOOL.

SPOOL determines if the spooler is running. If so, SPOOL asks "END?". If the reply is yes, a terminate spooling directive is sent to PIREX and the spooler is disabled. If the spooler is not running, SPOOL asks on which RK drive the user wishes to begin spooling. Spooling may be done on any RK unit that has a cartridge that has been initialized with a spooler area by the SPLGEN program. If the cartridge has a spooler area and if there is room in the PDP-11 local memory, the spooler is read from the system disk (DP0, DK, or RK0) and transferred to local PDP-11 memory and started. Note that the questions "RK UNIT #" and "BEGIN?" must be answered in this process.

All questions have default replies displayed. These replies may be selected by entering a carriage return. The options on YES/NO questions are "Y" or "N". The default value for the RK unit is the unit upon which spooling was done previously (or unit 0 if PIREX was just loaded).

Example:

```
XVM/DOS Vnxnnn

$SPOOL

SPOOL XVM Vnxnnn

        RK UNIT # [0] 0

        BEGIN ? (Y) Y

        SPOOLING ENABLED

XVM/DOS Vnxnnn

$SPOOL

SPOOL XVM Vnxnnn

        END ? (Y) Y

        SPOOLING DISABLED

XVM/DOS Vnxnnn

$
```

Appendix I lists the possible error messages and the conditions which cause them.

3.9.1 Output of Last SPOOLed Information

UNICHANNEL users are advised to include the .CLOSE X statement if writing a MACRO program or an ENDFILE X statement if writing a FORTRAN program, where X is the .DAT slot assigned to the Line Printer/Plotter to ensure output of the last few records. In the absence of this statement, the last few records will be output only after the spooler internal buffer containing these records is full.

CHAPTER 4
FILE STRUCTURES

4.1 INTRODUCTION

This chapter will define those concepts and facilities of the XVM/DOS system that are available for storage and retrieval of data from XVM mass-storage hardware/software systems, including DECdisk (RF), Disk Pack (RP) Magtape (MT), DECtape (DT) and Disk Cartridge (RK).

A large part of any programming task is accepting input and producing output. Therefore, it is necessary to understand the Input/Output process to take full advantage of the Disk Operating System's features.

4.2 DEVICE ASSIGNMENTS

As stated in the previous chapter, device assignment is managed through the Monitor's Device Assignment Table (.DAT). The .DAT associates logical device units with physical ones through "slot" numbers, which correspond to the logical device numbers.

Device assignment slots are assignable via the Monitor ASSIGN command at run time. Refer to Section 8.5 of Chapter 8 for more information. Default assignments are defined during system generation (see SGEN XVM Utility Manual. Information in each program indicates which "slots" are required for that program. These slots are called the "active slots". The loaders use the information about active slots to bring in handlers for the devices named by the active slots. At run time, then each slot contains the physical device unit number (if any) and a pointer to the appropriate device handler, which was brought in by the loader. Without device assignment on a .DAT slot, a program's call to that slot will result in an I/O error -- the slot will not point to a handler. Depending on the system generation, as many as sixty-three entries may be in the .DAT for program device assignment.

Each I/O command under DOS references a .DAT slot. These commands pass control to the appropriate device handlers via a Monitor routine that uses the pointers in the .DAT. These device handlers are responsible for transferring data between the program and I/O devices.

They also initiate the physical reading or writing of files and perform opening and closing of files and other functions peculiar to a given hardware device. Each slot then links the device-driving functions within a specified device handler to the program. The handlers all test for functions which they cannot service; for example, trying to rewind a card reader. Some functions are ignored; others are illegal and cause error messages to be output to the console teleprinter. All device handlers operate either with or without the Automatic Priority Interrupt (API) option.

4.3 FILES

A file is a collection of related records treated as a unit. In Inventory Control, for example, one line of an invoice forms an item, a complete invoice forms a record, and the complete set of such records forms a file. The word "file" is used in the general sense to mean any collection of information items similar to one another in purpose, form and content. "File" may also be generally applied to external storage media such as papertape, punched cards, a Magtape, a DECtape, DECdisk platters, Disk Cartridge and a Disk Pack.

4.3.1 Records

In DOS, a record is a set of one or more related data words accessible to the user as one item through .READ and .WRITE MACRO program statements or FORTRAN language statements. (Chapter 6 describes .READ and .WRITE). The smallest addressable logical item within the file is this logical record.

4.3.2 Words

A word is the least addressable physical data unit.

I/O Buffers are internal to, and must be defined by, each program. With the exception of certain block transfers, a buffer contains a single record. Under most of the Data Modes (described below), the first two words in an I/O buffer (i.e., a record) provide system information and cannot be used for data. This Header Word Pair within an I/O buffer is detailed in Chapter 6. On blocked devices (Disk and DECtape), these header word pairs govern the physical structure within the file. Before output, the user must set up the Header Word Pair. On input, the Header Word Pair arrives with Data Mode Information and

certain error indicators. The user is responsible for checking the various header parameters to determine if the data was read without error.

4.3.3 Data Modes

The Monitor allows data transmission to or from a system or user program in several different modes which structure internal data storage. Two modes, IOPS ASCII and IOPS Binary, offer the advantages of device independence. All handlers accept IOPS ASCII data, and most accept IOPS Binary. Three other data modes, Dump, Image Alphanumeric and Image Binary, allow the user to take advantage of device dependent characteristics. For more information, refer to Chapter 6.

4.4 FILE STRUCTURES

As stated in Chapter 3, a file structure is a method of recording, linking, and cataloging data files. File structuring, in general, is applicable to all I/O devices, including cards, paper tape, printers, keyboards, etc. In DOS, however, file structures are associated with Mass Storage devices only. A file structure dictates the file and record access methods. This organizational structuring is important because a file can be effective for a user application only if it is designated to meet specific requirements. A user must consider the following factors:

SIZE - Growth of the file may require a change in file structure or data mode.

ACTIVITY - The need to access many different records within a file (percentage of activity) or to frequently access the same records (an active file) will influence information retrieval efficiency. This requires the user to select the right method for each job task.

VOLATILITY - The number of additions and deletions to a file will affect the efficiency of the structures used.

4.4.1 File and Data Access Techniques

The usual technique for applications using Magtape is sequential access to the data file and sequential access of the data records. This

access is also characteristic of most uni-directional media, such as cards, paper tape, keyboards, printers, and displays.

Another technique, which in DOS is applicable only to the disks, involves random-access (an alternate term is "direct-access") to a file and random-access of data records within the file.

A third and intermediate technique, which is normally used on DECtape and on disk, employs random-access to a file and sequential access to the file's data records. The formal term is "random-sequential file structure", the informal term is "DECtape file structure".

4.4.2 Sequential Access

Sequential access is a storage retrieval technique in which a file and the records within it must be retrieved in the sequence in which they physically occur. Sequential access, when applied to the process of locating the beginning of a file or a data record within the file, means that the time required for such access is dependent on the necessity for waiting while non-desired records are processed in turn.

1. Records can only be retrieved sequentially, beginning with the first record and accessing each subsequent record in turn.
2. Direct retrieval is not possible, since no directory exists to the physical locations of the records.
3. When adding or deleting a record, the system copies all the records preceding that record in the file, then performs the addition or deletion. The system then recopies all the remaining records in the file.

4.4.3 Direct Access

Direct access to a file or to a data record within the file means that the time required for such access is independent of the location of the file or record relative to other files or records.

Direct access should be considered as a valuable access method when retrieving selected records from random files; but it is not suited for sequential retrieval of records. The direct access technique makes it possible to process only the affected record during a file update. This reduces data sorting.

The main advantage of direct access processing is that it requires fewer processing steps. Along with its advantages, however, a direct access application requires special considerations by the system programmer. Backup on a direct access system differs from backup on a tape system because the old record on a direct access device is destroyed when the updated record is written over it.

4.5 MAGNETIC TAPE FILE STRUCTURE

The DOS software provides for industry-compatible magnetic tape (Magtape) as either a directoried or a non-directoried medium. The magnetic tape handlers communicate with a single Tape Control Unit with up to eight magnetic tape transports.

When used as a non-directoried medium, there are a number of major differences between Magtape and other mass storage devices such as DECtape or Disk. Magtape is well suited for handling data records of variable length; such records, however, must be treated in serial fashion. The physical position of any record may be defined only in relation to the preceding record.

When used as a directoried medium, Magtape assumes the external operating characteristics of DECtape (described in 4.6).

Sequential tape files are written one after the other, starting at the physical beginning of the tape. These files are separated from one another by End-of-File marks (hardware-detected) or by an End-of-Record line (software-detected). (Refer to Figure 4-1, Sequential Data Access.) To read the N^{th} file on a tape, the user must first rewind to the beginning of the tape and then skip serially through N-1 files to the desired file. Sequential tape files need not have a filename; only a known position relative to the files on the tape.

Data records within a file are recorded in sequence. Sequential statements direct records to and from memory in the sequence in which they are physically on the device. To access the M^{th} record within the N^{th} file, the user first locates the beginning of the file (skipping N-1 files) and then skips through M-1 records.

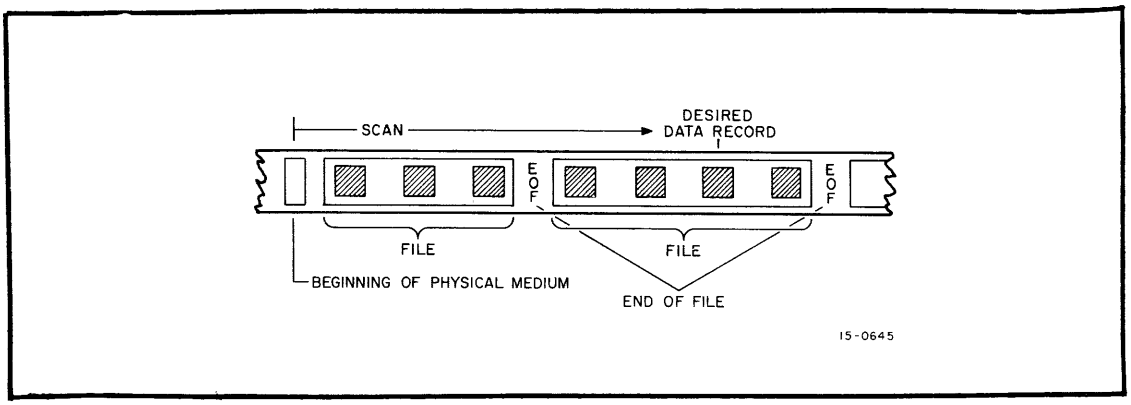


Figure 4-1
Sequential Data Access

This sequential access method does not need a tape-resident directory to point to files. It is, therefore, called "non-directoried" in DOS. The non-directoried access method is generally applicable to unit-record peripheral devices.

4.6 DECTAPE FILE STRUCTURE

If magnetic tape transports could identify absolute positions on a tape, there would be no need to do sequential file access. The transport could search to a known location, and start processing. Then, an extra body of information (a directory) on the tape could point to each file. DECtape has this capability.

In directoried mode, files are given unique names. The name and position of the beginning of a file are recorded in a directory. Directories are at a fixed location (Block 100_8) on the tape. DECtape directories point to the first block of each file and each block points to the next. DECtape directories also maintain bit maps which indicate which tape blocks each file occupies and a map to show all occupied blocks on the tape. The file directory is fixed in size; consequently, the number of files that may be recorded is limited by the amount of storage available on the device (1100_8 blocks) and/or by the number of file name slots in the directory (56 maximum).

File structures which employ a directory allow simpler and, in the long run, faster access to a file (the beginning of a file). See Figure 4-2. This is a distinct advantage over those devices which do not use a directory and must therefore rely on a file's position relative to other files in order to locate it.

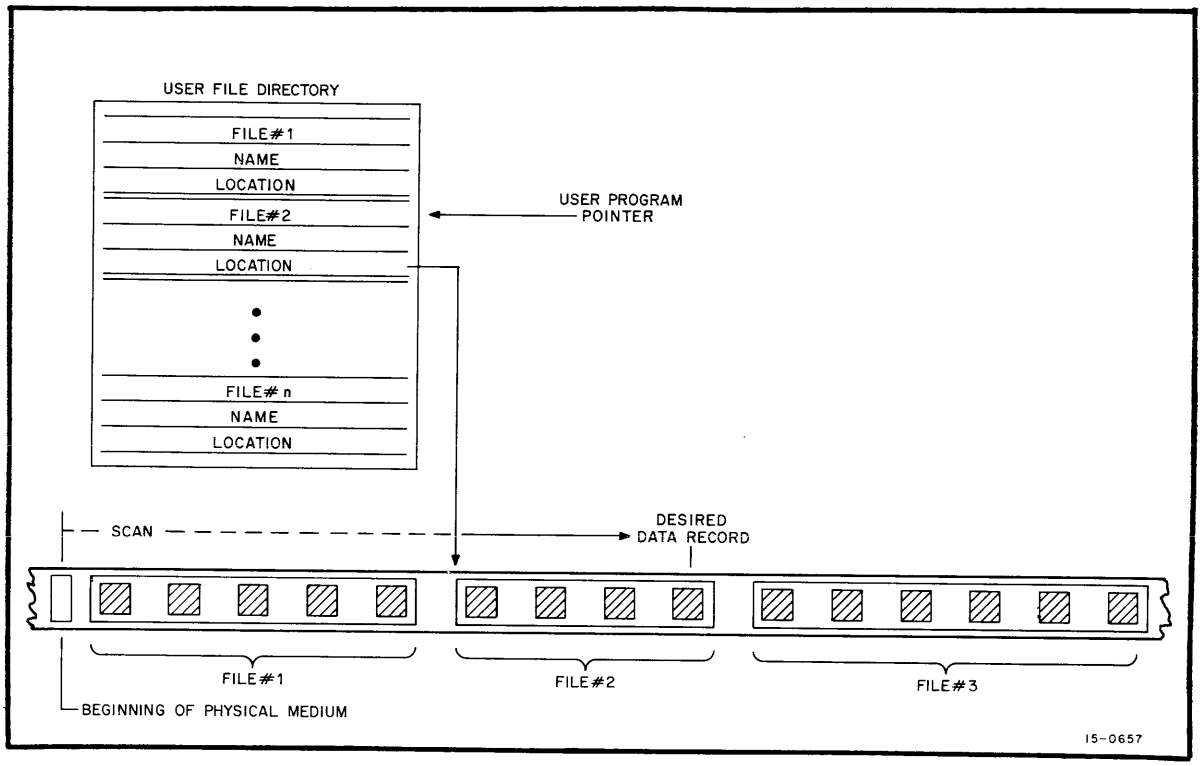


Figure 4-2
Directoried Data Access

Programmers need not use the directory capabilities of DECtape. They can treat DECtape as magnetic tape by issuing magnetic tape I/O commands. Then no directory or identifying information of any kind is recorded or referenced on the tape and operation is similar to that of magnetic tape (described in 4.5). The DECtape handlers with their respective I/O functions allow the use of either of these modes.

4.7 DISK FILE STRUCTURE

The DOS disk file structure is in some ways analogous to DECtape file structure. Ordinarily, each disk user has a directory which points to named files, just as each DECtape has a directory. A single user's disk directory might correspond to a single DECtape directory. The DECtape, however, has only one directory, but the disk has as many directories as users have cared to establish. Whereas DECtape directories may reference only a maximum of 56_{10} files, the number of files associated with any one directory on the disk is limited only by the available disk space. A single disk file's size is also limited by the available space, as is true with DECtape.

The DECtape directory is in a known location...at block 100_8. Since the disk may have a variable number of directories, the Monitor must know how to find each user's directory. It, therefore, maintains a Master File Directory (MFD) at a known location, and the Master File Directory points to each User File Directory (UFD). DOS allows only those users who know a special code, called the Monitor Identification Code (MIC), to have access to the MFD. Figure 4-3, Disk Data Access, illustrates the organization of the disk.

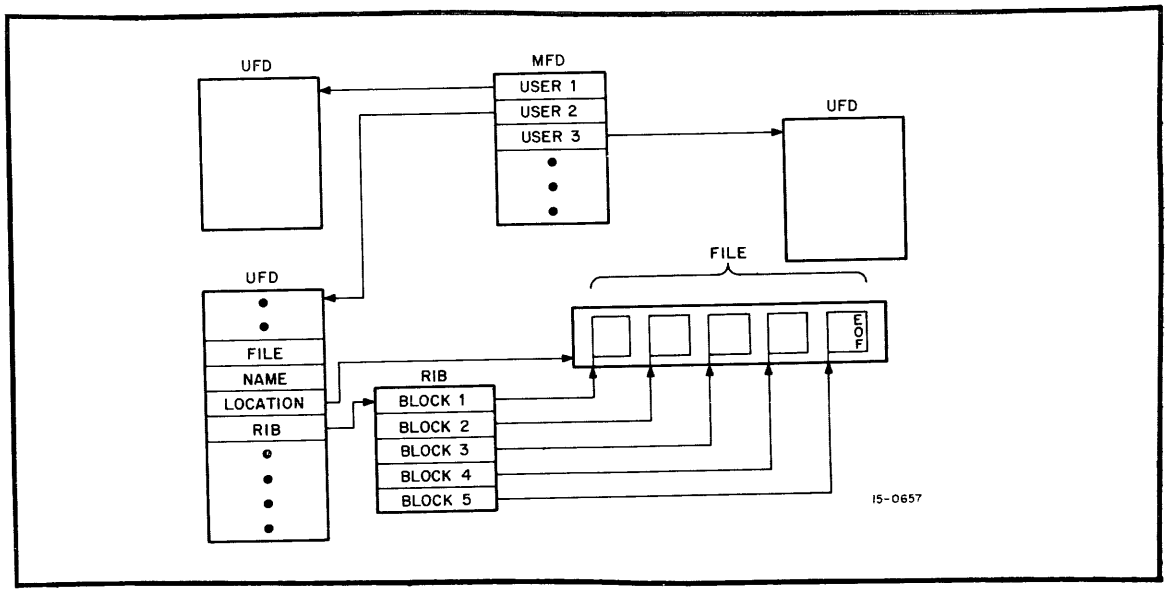


Figure 4-3
Disk Data Access

4.7.1 User Identification Codes (UIC)

The Monitor finds User File Directories by seeking associated User Identification Codes (UIC's) which are all listed in the Master File Directory. The UIC is a three-character code that is necessary for all directory-oriented I/O to the disk. A programmer may establish a new User File Directory by:

1. Logging in his new UIC to the Monitor,
2. Calling PIP,
3. Issuing an "N␣DK↵" command for RF15 DECdisk or "N␣DP↵" for RP15 Disk Pack or "N␣RK↵" for RK05 Cartridge Disk.

This establishes a new User File Directory, or refreshes (wipes clean) an old directory under that UIC. The "enter file" I/O command will also create a UFD, if none exists.

4.7.2 The User File Directory Table (.UFDT)

The Monitor must have a way of knowing which User File Directory to reference when a program issues I/O commands to the disk. It makes the association between disk I/O commands and User File Directories by using a User File Directory Table (.UFDT). There are as many entries in the User File Directory Table as there are slots in the Device Assignment Table. Figure 4-4, Relationship Between the .DAT and the .UFDT, shows how the two compare. Disk I/O to a particular .DAT slot will affect files in the User File Directory names in the corresponding .UFDT slot. Programmers may modify .UFDT slots before loading a program, and programs themselves may alter .UFDT.

.DAT/.UFDT Number	.DAT Contents	.UFDT Contents	Comment
+N ⋮ ⋮ ⋮	nondisk handler ⋮ ⋮ ⋮	UIC_1 ⋮ ⋮ ⋮	This UIC is irrelevant
+2	disk handler	UIC_1	I/O to this slot will go to files in UIC_1
+1	disk pack handler	UIC_2	I/O to this slot will go to files in UIC_2
⋮ ⋮ -15	⋮ ⋮ none assigned	⋮ ⋮ UIC_1	This UIC is irrelevant

The operator has logged in under UIC_1. He has assigned some nondisk handler to .DAT+N, the DECdisk to .DAT+2, and the Disk Pack unit Ø to .DAT+1, and NON (no handler) to .DAT -15. In addition, he has changed the .UFDT assignment for .UFDT+1 to UIC_2. I/O to .DAT+N will not reference the UIC, since UIC's are relevant only to disk I/O. I/O to .DAT+2 will reference UIC_1, while I/O to .DAT+1 will reference UIC_2. The UIC for .DAT -15 is irrelevant, since no handler is assigned.

Figure 4-4
Relationship Between the .DAT and the .UFDT

4.7.3 File Protection

DOS offers a simple form of file protection. Each User File Directory has a protection code and each file has a protection code. The protection codes only protect a programmer's files from other users - not from himself. If a User File Directory is protected, then the protection codes for each file are in effect. If the user File Directory has been specified as unprotected, then no protection is provided for any file in the directory. There are three three possible protection states for files in a protected User File Directory.

File Protection Codes:

1 = Unprotected, with the exception that the file may not be deleted and the number of blocks may not change, if the directory is protected.

2 = Write protected, if directory protected.

3 = Read/Write protected, if directory protected.

User File Directories may have one of two protection states:

File Directory Protection Codes:

0 = Unprotected

1 = Protected

The default protection code for User File Directories is always 1, protected. The default protection code for files is established at system generation time.

4.7.4 Organization of Specific Files on Disk

The disk handlers write out files sequentially, just as the DECtape handlers do. On the close of an output file, the disk handlers also fill out a Retrieval Information Block (RIB). The RIB performs the same functions as the file bitmap on DECtape, and associates the logical sequence of blocks in the file with the physical locations of the blocks on the disk. The disk handlers use the RIB to implement random access and to delete files.

After a user has created a disk file, he can access records sequentially as with DECtape files. He can also read and write physical blocks of that file by referencing relative block numbers. The user

is prohibited, however, from changing the total number of blocks in the file. To do this he must copy the contents of the current file into a new, larger file.

4.7.5 The Disk Handlers

The disk handlers allow as many concurrently open input or output files as there are .DAT slots available to the user, and buffers available to the disk handler. The disk handlers operate under a dynamic buffer allocation scheme. Whenever the Monitor loads a system or user program, it allocates buffer space. This space is called the buffer pool. Whenever a program opens a disk file, the handler obtains a buffer from the buffer pool. The handlers return the buffer when the program closes the file.

The buffers in the buffer pool are available to programs, as well as to disk handlers. Whenever a program is using a buffer, however, it is unavailable to any other program.

CHAPTER 5
DOS SYSTEM MACROS

5.1 INTRODUCTION

This chapter contains detailed reference information concerning XVM/DOS System Macros. I/O Macros (those which have a .DAT slot number as one argument) are discussed in Chapter 6.

5.2 MONITOR-PROCESSED COMMANDS

The MACRO assembler permits the use of higher-level system instructions called "System Macros" which, when used as a source statement, can cause a specific sequence of Monitor operations to occur at run-time.

5.2.1 Summary of DOS Monitor System Macros

Table 5-1, Summary of System Macros, summarizes the non-I/O macros which the Monitor implements. The following paragraphs describe the individual system macros in detail.

Table 5-1
Summary of System Macros

Name	Description
.EXIT	Requests the System Loader to bring in the Non-resident Monitor.
.GET	Overlays core with the contents of a named file.
.GTBUF	Requests a buffer from the buffer pool.
.GVBUF	Returns a buffer to the buffer pool.
.OVRLA	Requests the System Loader to load the named core-image program.
.PUT	Creates a file containing the current core image.
.SYSID	Aids in system program version identification.
.TIMER	Initializes a time interval after which program control passes to a user specified subroutine.

5.2.2 DOS System Macro Expansions

The following standards apply to the descriptions of the System and I/O Macros presented in this chapter and in Chapter 6:

A. Format Conventions

1. Upper case terminology must be used as stated in the FORMAT Description.
2. Lower case terminology indicates user-supplied information, as defined by ARGUMENTS.
3. Names for user-supplied parameters indicate maximum length of field by the number of characters in the name. Thus, "namptr" may be a maximum of six characters long, and "num" may be a maximum of three characters.
4. Brackets ([ ]) indicate optional quantities.
5. A bracket on its side (⌴) indicates a space.
6. Quantities listed vertically and enclosed in braces ( ) indicate the user must choose one from the group.
7. The expansions indicate how one might write an expanded macro directly to the assembler. The expansions may appear different from expansions produced by the assembler. They are, however, functionally equivalent.

B. Sixbit File Representation

Many of the System and I/O Macros have a "namptr" argument which points to a 3-word block of core in the user's program containing the Sixbit representation of a file name and extension. This is simply an abbreviated form of ASCII in which the rightmost 6 bits of the ASCII code are used. This allows 3 characters to be packed in one word of storage. Sixbit packing can be accomplished either by the user program or by the MACRO Assembler's .SIXBT pseudo-op. (Refer to the MACRO XVM Assembler Language Manual. The example following paragraph 6.8.5 demonstrates the use of .SIXBT.

5.2.2.1 .PUT - Creates a file which contains the current core image. The size of the file is equal to the minimum of the ↑Q area size and the current system memory size. The value is kept in .SCOM, and will be stored in the file (as part of .SCOM in the core image) for use by GET and .GET. The file's name and extension will be that found in the locations pointed to by namptr. See also .GET.

FORMAT:

.PUT␣f,namptr

ARGUMENTS:

f = (function) determines the startup location for a .GET with f=4, or a GET Keyboard command to the Nonresident Monitor.

f = 0 Subsequent load will start at the location following the EXPANSION.

f = 1 Subsequent load will start at current CTRL P address.

f = 2 Subsequent load will start at current CTRL T address.

f = 3 Subsequent load will start at current CTRL S address.

namptr = Pointer to the address of a 3-word, .SIXBT representation of the name and extension of the file to be created (unused spaces must contain nulls). The 3-word block must be entirely within the lower 32K words of XVM memory.

EXPANSION:

```
LOC+0  CAL+f
LOC+1  26
LOC+2  namptr
```

5.2.2.2 .GET - Overlays core with the contents of the file whose name is indicated by the Macro argument namptr. The file must have been created by a .PUT System Macro, or a PUT command to the Nonresident Monitor, and it must reside on the device associated with .DAT -14. .GET destroys the current contents of .SCOM, including DATE and TIME. The .GET command tests the image size of the file as stored in the .SCOM in the image. If the image size is larger than either the size of the ↑Q area or the current system memory size, then the following question is asked:

IMAGE TOO LARGE - CONTINUE (Y/N)?

If the reply is "N", control is returned to the Nonresident Monitor.

If the reply is "Y", the minimum of the ↑Q area size and the current system memory size is copied into the ↑Q area and into memory. If the image size is smaller or equal to both the ↑Q area size and the current memory size, the image will be copied into the ↑Q area and into memory. See also the .PUT System Macro, and the PUT and GET commands to the Nonresident Monitor.

FORMAT:

.GET␣f,namptr

ARGUMENTS: f = (function) - determines startup location after overlay.

- f = Ø Start at location after expansion of System Macro that created the file. If file was created via a .PUT command to the Nonresident Monitor, load core from the file, and await a command from the console keyboard.
- f = 1 Start at CTRL P address
- f = 2 Start at CTRL T address
- f = 3 Start at CTRL S address.

  (Refer to the XVM/DOS System Manual for more information.)
- f = 4 Start at the location specified by the .PUT System Macro. If the file was created via a .PUT call to the Nonresident Monitor, load core, and await command from the console keyboard.

namptr = Pointer to the first word of a three-word .SIXBT representation of the filename and extension for the core image to be loaded. (Unused spaces must contain nulls.) The 3-word block must be entirely within the lower 32K of XVM memory.

EXPANSION:

```
LOC+Ø   CAL+f
LOC+1   25
LOC+2   namptr
```

5.2.2.3 .GTBUF - Requests a buffer from the buffer pool. If a buffer is available from the pool, the Monitor will return the address of the first word of buffer in the AC. If no buffer is available, the Resident Monitor will return 777777 in the AC. On return, execution begins at LOC+3. LOC+2 can be used as desired. This Macro gives user programs as well as handlers access to buffers.

FORMAT:

.GTBUF

ARGUMENTS:

none

EXPANSION:

```
LOC+Ø   CAL+Ø
LOC+1   21
LOC+2   Ø
```

5.2.2.4 .GVBUF - Allows a user to return to the buffer pool a buffer obtained via a .GTBUF System Macro.

FORMAT:

.GVBUF

ARGUMENTS:

None, at assembly time, but AT RUN TIME, the program must load LOC+2 with the address of the buffer to be freed. If LOC+2 does not contain the address of an allocated buffer, the AC will contain 777777 on return.

EXPANSION:

```
LOC+Ø   CAL+Ø
LOC+1   22
LOC+2   Ø    (Load with first address of buffer)
```

5.2.2.5 .OVRLA - Requests the System Loader to bring into core and start the core image system program whose name is pointed to by the Macro argument namptr. If there is not enough room in core to load the program requested, the Resident Monitor will return control to LOC+3.

WARNING

All I/O should be completed before issuing an .OVRLA.

FORMAT:

.OVRLA␣namptr

ARGUMENTS:

namptr = (name pointer) first address of the two-word . SIXBT representation of the name and extension of the file containing the program to be brought in. The two-word block must be within the lower 32K of XVM memory.

EXPANSION:

```
LOC+Ø   CAL+Ø
LOC+1   24
LOC+2   namptr
```

5.2.2.6 .EXIT - Requests the System Loader to bring in the Non-resident Monitor. Current contents of core are lost. All unclosed output files are destroyed, all input files are closed.

FORMAT:

.EXIT

ARGUMENTS:

none

EXPANSION:

```
LOC+Ø   CAL+Ø
LOC+1   15
```

5.2.2.7 .TIMER - Allows programs to schedule routines to be called after specific time intervals.

Refer to Chapter 2 of the XVM/DOS System Manual for programming rules for these routines.

FORMAT:

.TIMER␣nnnnnn,addres

ARGUMENTS:

nnnnnn = (time interval) number (in decimal radix) of clock ticks from "now" after which the subroutine is to be called

$$Ø<nnnnnn\ 2^{18}-1 \text{ or } 262143$$

addres = address of the routine to be called after the specified number of ticks. This address must be within the lower 32K of XVM memory.

EXPANSION:

```
LOC+Ø   CAL+Ø
LOC+1   14
LOC+2   addres
LOC+3   -nnnnn
```

5.2.2.8 .SYSID - Aids in system program version identification. The .SYSID macro performs two functions:

1. It defines several parameters. These parameters might be used for conditional assembly control. The following parameters are defined:

```
%XVM = 1/indicates XVM hardware
%VERSN = 1/indicates version 1 (in V1A)
%RELES = 1/indicates release A (in V1A)
```

These parameters are not altered by .SYSID if they are already defined. Remember that these parameters cannot be defined by .SYSID until the macro has been invoked; and that the parameters will only be defined the first time .SYSID is invoked. The definitions of these parameters are not included in the assembly listing.

2. It provides a convenient method for inserting a text string into an assembler statement.

FORMAT: →| .SYSID␣front, back

EXPANSION: front @ XVM␣V1A @ back

CHAPTER 6
PROGRAMMED I/O COMMANDS

6.1 INTRODUCTION

This chapter describes the concepts, commands and methods for incorporating input/output commands in MACRO assembly language programs. FOCAL and FORTRAN users need not, in general, be concerned with the contents of this chapter, since each of these languages has its own I/O capability.

The DOS IOPS is the interface between any program and the external world of I/O devices. I/O device handlers are provided for all standard devices and are described in Chapter 9. These handlers relieve the user of the burden of I/O considerations and unwanted device dependence.

6.2 GENERAL I/O COMMUNICATIONS

Under DOS, all I/O transfers and subsidiary I/O operations are initiated by programmed I/O commands called I/O Macros, as shown in Table 6-1. These I/O Macros are macro instructions which have a .DAT slot as one argument. As with System Macros, they are permanently defined in the MACRO Assembler.

As can be seen from Table 6-1, there are four I/O Macros within the DOS Software System which effect data transfer: .READ, .WRITE, .RTRAN and .TRAN. The first three permit operation using the standard DOS file structures described in Chapter 4 and provide the most device independent approach to I/O programming. The .TRAN Macro functions independently of DOS file structures.

Table 6-1

SYNOPSIS OF DOS I/O MACROS

Macro	Function
.CLEAR[1]	Initializes a directory on a directoried mass storage device. All data on the device is lost and fresh bit maps and directories are written.
.CLOSE[1]	Terminates use of a file. In the case of output files on the disk, fills out the Retrieval Information Block (RIB) for later .RTRAN commands.
.DLETE[1]	Deletes a file from a directory on a mass storage device.
.ENTER	Primes a directoried mass storage device to accept an output file.
.FSTAT	Checks the directory on a mass storage device for the existence of a named file.
.INIT[1]	Initializes the device and device handler.
.MTAPE	Provides special commands for industry compatible magnetic tape.
.RAND	Opens a disk file for random processing via .RTRAN macros.
.READ	Transfers a logical record, or the requested number of words, whichever is smaller, from the device to the user's I/O buffer.
.RENAM[1]	Renames a file in a directory of a mass storage device.
.RTRAN	Allows input and output to access any block in a pre-existent file on the disk (any word if RF).
.SEEK	Checks for a named file in a directory on a mass storage device. If the file is present, prepares it as an input file.
.TRAN	Gives independence from DOS file structure by allowing input and output access to Magtape, or to any block on disk or DECtape located by its physical block number. A file can be opened for sequential or random access and still be .TRANed via the same .DAT slot without destroying the file structure.
.USER	Allows programs to change the .UFDT.
.WAIT	Waits for I/O already started to complete, then continues.
.WAITR	If I/O is complete execution continues. If I/O is not complete control is transferred to the specified address.
.WRITE	Transfers data from user's I/O buffer to the device.

[1]At completion of these operations, the buffer is given back to the buffer pool, if disk or DECtape "A" version I/O handler is used.

6.2.1 .READ/.WRITE/.RTRAN Operations

.READ and .WRITE Macros permit the user to sequentially input and output records of a file consisting of ASCII lines or binary data to any device in the standard set of XVM I/O devices. The .RTRAN Macro provides the user with random access to physical blocks when using the disk devices. Initially, files are created sequentially (.WRITE) and can subsequently be accessed both sequentially (.READ) and randomly (.RTRAN). Data which is to be read and written by these commands must reside entirely within the lower 32K of XVM memory.

6.2.2 .TRAN Operations

The last data transfer macro (.TRAN) functions only on mass storage devices. It provides I/O transfer capability at the device level without regard for established file structures. This type of transfer provides for user-designed file structuring. The .TRAN function permits data transfers directly to and from all XVM memories including memory above the first 32K.

Much of the remainder of this chapter is devoted to I/O programming within the DOS file structures and, specifically, to the methods and considerations involving the creation of sequential files. Little can be said here about the utilization of the .TRAN and .RTRAN Macros since the system places the burden of data structuring, interpretation, checking and packing on the programmer.

6.3 SEQUENTIAL FILE PROCESSING

I/O operations under DOS consist of the transfer of ASCII or binary logical records between buffer areas in the system or user program and I/O devices represented by the device handlers in IOPS. (A logical record is defined as the amount of ASCII or binary data which is transferred to or from a program as the result of the execution of a single .WRITE or .READ I/O Macro.) The size of each logical record depends upon the structure of the data (Data Mode) and the device or set of devices addressed. The format and data structure of each logical record output in IOPS and Image Data Modes is described below. For Dump Mode, however, no particular requirements are placed on the user by the system. Dump mode is provided to permit user-created data structures within the DOS file structure.

6.3.1 Logical Record Format, IOPS and Image Modes

Each logical record to be output in IOPS and Image Modes must be formatted as shown in Figure 6-1. The record, consisting of 2n 18-bit words contains a two word header (called header word pair) followed by 2n-2 words of ASCII or binary data.

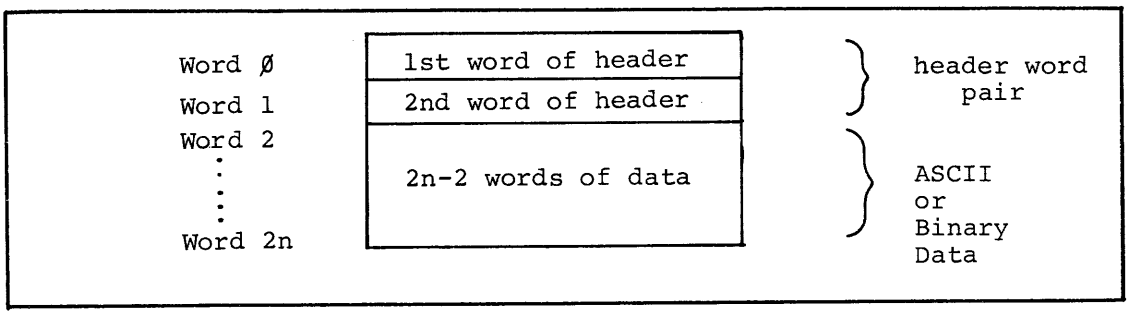


Figure 6-1 IOPS and Image Mode Logical Records Format

The header word pair is a communications link between the user's program and the I/O device handlers in IOPS. It is divided into various fields which contain information about the data which resides in the record. The information in some fields is supplied by the program to assist the device handlers in writing records and subsequently retrieving them. Other fields contain information which is provided by the device handlers for use by the user or system program when verifying records during input. Refer to paragraph 6.4.3 for an explanation of Dump Mode record format.

6.3.1.1 Header Word Pair Format - Figure 6-2 shows the format of the header word pair. Of particular significance is the Word Pair Count; it specifies the number of word pairs of data in each logical record. In addition, for all Data Modes except Dump Mode, it is the prime logical record terminating condition when using mass storage devices. This term should not be confused with the Word Count, which is an argument used in I/O Macros. The Word Count gives the actual size of the I/O buffer which contains the logical records prior to output and after input operations (see 6.7).

6.3.1.2 Using the Header Word Pair Before Output - The program must calculate and then set the appropriate word pair count in bits 1-8 of header word zero, unless they have already been set by a device handler on input (i.e., an input device handler set up the header word pair for each record read). This count overrides the word count passed to IOPS by the .WRITE system Macro. The I/O mode field, bits 14-17, is set by the device handler from the I/O mode argument specified in the .WRITE Macro. The checksum word need not concern the user since checksums are computed by IOPS (when using IOPS Data Modes).

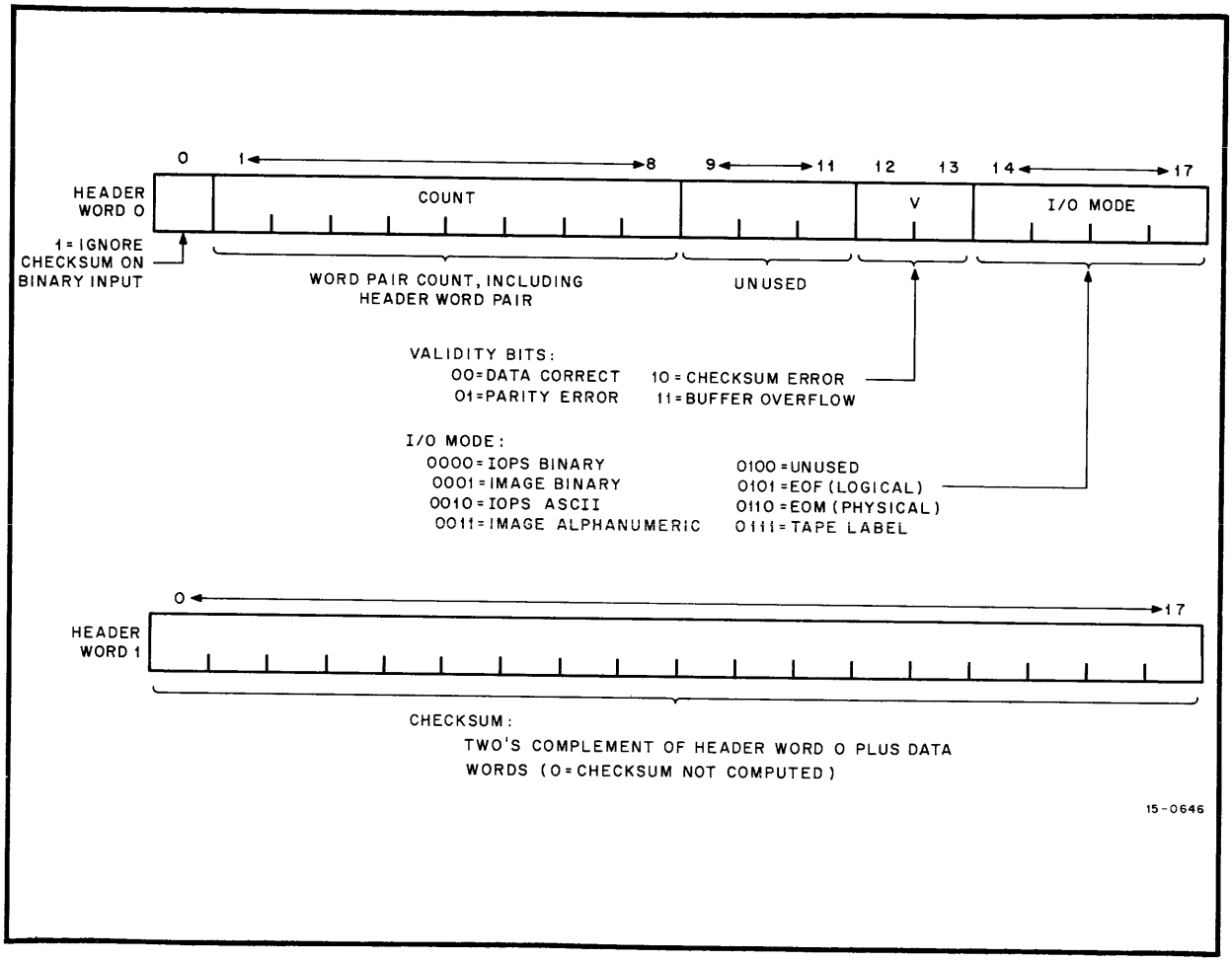


Figure 6-2
Format of Header Word Pair

6.3.1.3 Using the Header Word Pair Before Input - The user need not be concerned with the header word pair since it will be set up by the device handler during input to enable the user to determine the status of his record after input has terminated.

6.3.1.4 Using the Header Word Pair During Input - The Word Count (specified in the .READ Macro) is used by the device handler to determine the maximum number of locations provided in the user's input buffer for the data being read. If the Word Count is exceeded before the end of the record (as specified by the Word Pair Count) has been reached or if a parity or checksum error has occurred, the handler sets the validity bits of header word Ø as required to indicate the error.

6.3.1.5 Using the Header Word Pair After Input - Header word Ø of each logical record input should be examined by the user program to determine whether errors have occurred. Specifically, the validity and I/O Mode bits should be tested. If both checksum and parity errors are detected by a handler, priority is given to a parity error and the checksum error will not be indicated. IOPS ignores checksum errors on binary input if bit Ø of word Ø is set to 1. When examining the I/O Mode Bits (bits 14-17) the occurrence of a buffer overflow condition means that the user program's I/O buffer, as specified by the Word Count in the .READ Macro, is not large enough to contain the record just read. The portion of the record which caused the overflow is lost. In addition, IOPS uses the I/O Mode Bits to indicate that either the physical end-of-medium (EOM) or the logical end-of-file (EOF) has been reached; otherwise, these bits specify the Data Mode of the Record.

6.4 DATA MODES

The device handlers within IOPS allow data transfers via .READ and .WRITE Macros in one of the five Data Modes listed below:

IOPS Binary	(Mode Ø)
IOPS ASCII	(Mode 2)
Image Binary	(Mode 1)
Image Alphanumeric	(Mode 3)
Dump	(Mode 4)

Data Modes permit the user to select the data structuring features of the system which are important to his application. The device independent features of the system can be enhanced through the use of the IOPS Modes, which are standardly used by all DOS System programs (e.g., both FORTRAN and MACRO accept source programs in the form of IOPS

ASCII files and produce object code as IOPS Binary files). Conversely, if specific device dependent features are desired, Image Mode can be used primarily with non-mass storage devices, and Dump Mode can be used primarily with mass storage devices.

6.4.1 IOPS Modes

The IOPS Data Modes, both ASCII and Binary are the standard data structures of the DOS Software System. Using these modes, all ASCII and Binary data input is verified and converted into standard records regardless of its original form on the input device. Before output, programs must format data into standard IOPS records. On output, IOPS calculates checksums and either reconverts the data to the form required by the output device or, in the case of mass storage devices, stores the data in the standard record format.

6.4.1.1 IOPS ASCII - IOPS ASCII is used by DOS System Software as the standard ASCII Data Mode. It accommodates the entire 7-bit ASCII (1968) 128 character set as shown in Appendix A. All alphanumeric data, whatever its original form on input (i.e., 8-bit ASCII, Hollerith, etc.) or final form on output, is converted internally by the non-mass storage device handlers and stored in core and on mass storage devices as "5/7 ASCII". This term refers to the internal packing and storage scheme used for IOPS ASCII in which five 7-bit ASCII characters are packed into two contiguous 18-bit words. Figure 6-3 shows this relationship. ASCII packed

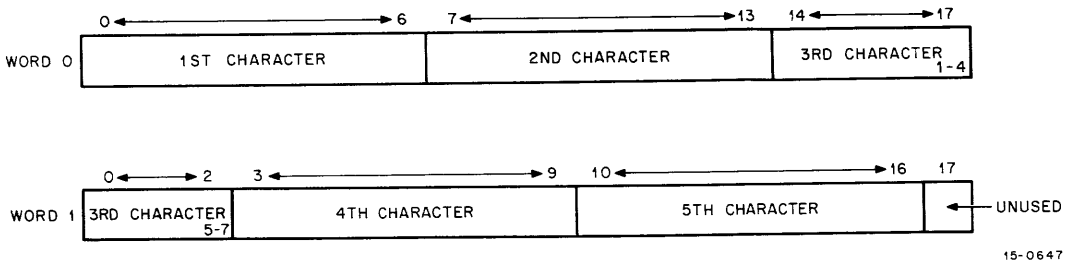


Figure 6-3 5/7 ASCII Packing Scheme

in this form can be stored "as is" on any mass storage device. I/O requests involving 5/7 ASCII should be made using an even word count argument to accommodate the paired data. ASCII data is ordinarily input or output character-by-character via non-mass storage devices such as teleprinter, line printer, paper tape reader and punch. It can also be stored on mass storage devices in 5/7 form (see Figures 6-4 and 6-5). On paper tape an IOPS ASCII character is defined as a 7-bit character with even parity in the eighth (high order) bit, in keeping with USASCII standards. Further, IOPS performs a parity check on input, prior to the 5/7 packing and on output IOPS generates the correct parity.

Non-parity IOPS ASCII occurs in data originating at a Model 33, 35, or 37 Teleprinter, without the parity option. This data always appears with the eighth (high order) bit set to 1. Apart from parity checking, the IOPS routines handle IOPS ASCII and non-parity IOPS ASCII data identically.

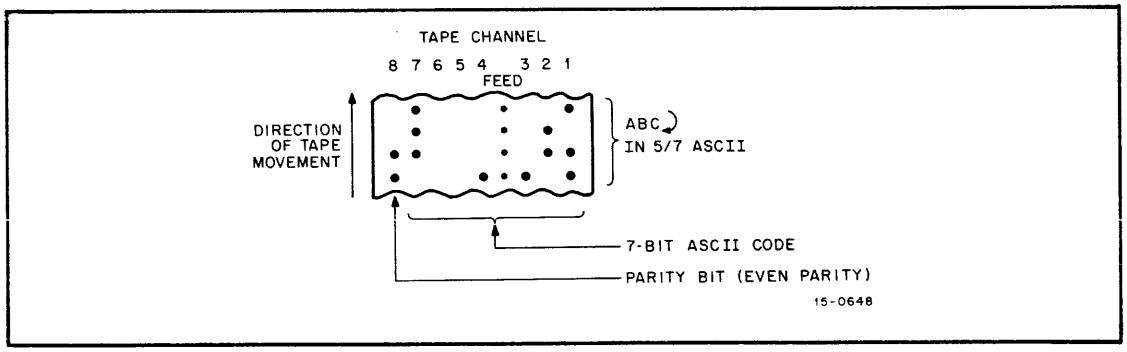


Figure 6-4 IOPS ASCII on Paper Tape

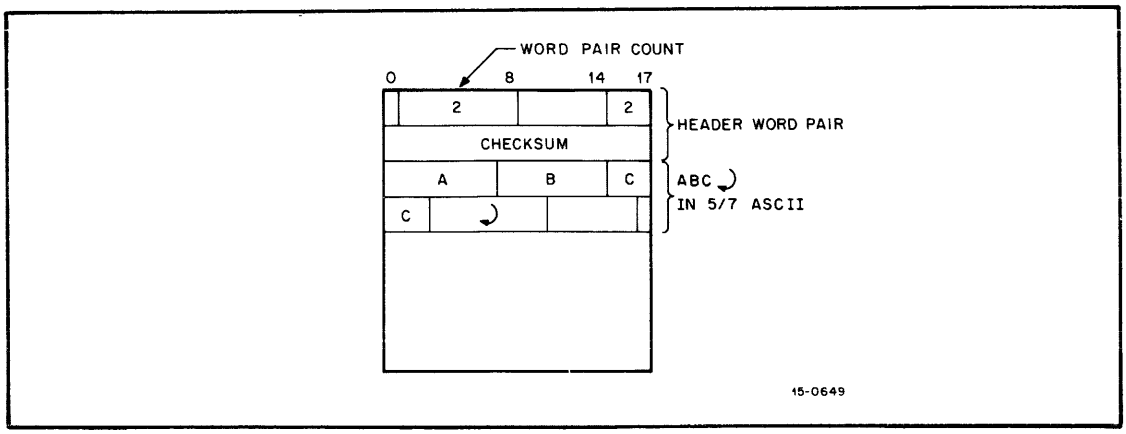


Figure 6-5 IOPS ASCII in I/O Buffers and on Mass Storage Devices

Each logical record is an alphanumeric line followed by a Carriage Return (CR) of ALT MODE. CR (or ALT MODE) is a required line terminator in IOPS ASCII mode to non-mass storage. Unused character positions in the IOPS Word Pair after the CR or ALT MODE are ignored. Control character scanning is performed by some device handlers for editing or control purposes (refer to Chapter 9).

On input, each IOPS ASCII .READ results in a 5/7 packed ASCII line being placed in the program's I/O buffer. If this line is to be interpreted, it must be unpacked by the program. Conversely, on output each .WRITE assumes that the data in the user's I/O buffer is a 5/7 packed ASCII line. Thus, it is up to the program performing IOPS ASCII data transfers to unpack each input line and pack each output line. Appendix B contains assembly listings of generalized packing and unpacking routines which can be incorporated into user programs as required.

6.4.1.2 IOPS Binary _ IOPS Binary data is blocked in an even number of words, with each block preceded by a two-word header (see Figure 6-6). On paper tape (see Figure 6-7), IOPS uses six bits per frame, with the eighth channel always set to 1, and the seventh channel containing the parity bit (odd parity) for channels 1 through 6 and channel 8. The parity feature supplements the checksumming as a data validity provision in paper tape IOPS binary.

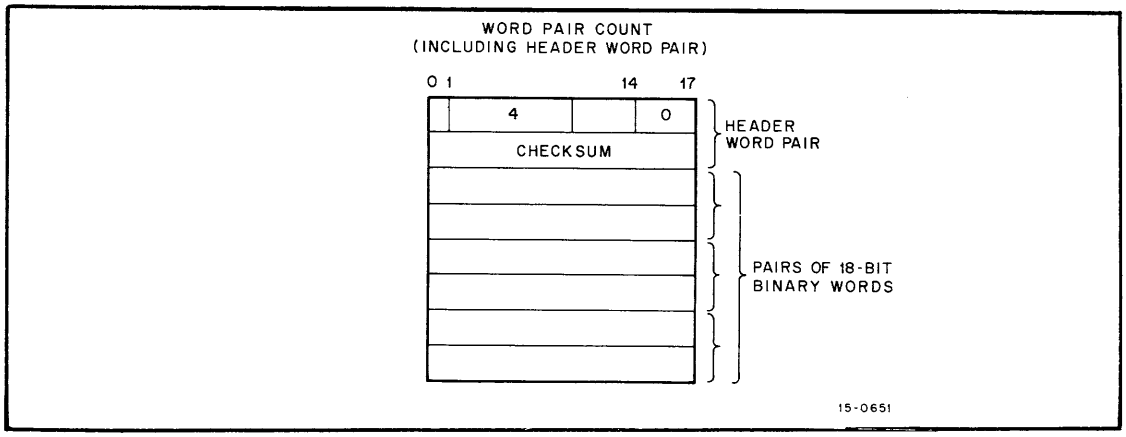


Figure 6-6 IOPS Binary in I/O Buffers and on Mass Storage Devices

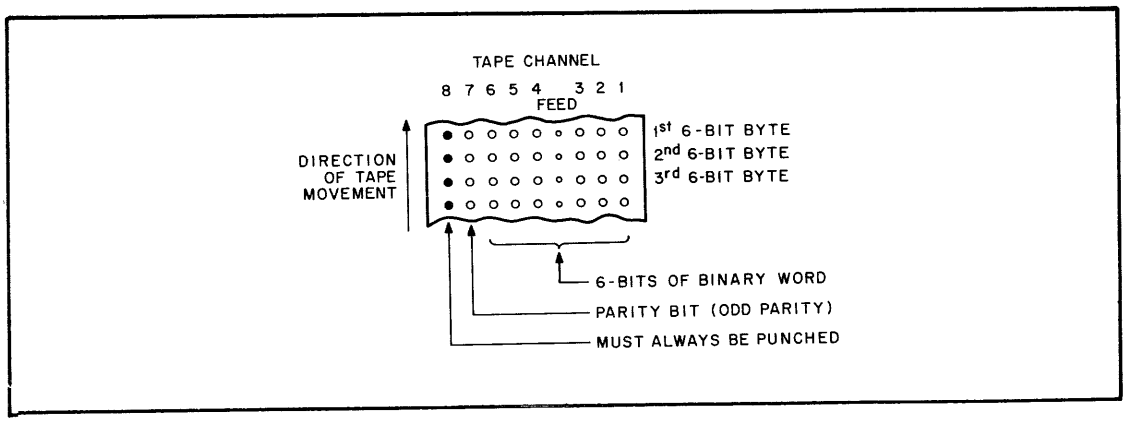


Figure 6-7 IOPS Binary Data on Paper Tape

6.4.2 Image Modes

Image Mode data, both Image Alphanumeric and Image Binary, is read, written and stored (on mass storage devices) in the form established by the source or terminal device. These modes permit the user to take advantage of the data structuring features peculiar to non-mass storage devices. These modes are strictly device dependent and no checking, packing or interpretation of data is performed. For example, when dealing with Alphanumeric data the teleprinter editing features RUBOUT and CTRL U, described in 9.3.1 are ignored, as well as the IOPS ASCII line terminators Carriage RETURN and ALT MODE. With the line printer, however, Carriage RETURN and ALT MODE are accepted as legal line terminators.

Image Alphanumeric Mode results in the transfer of all eight bits of an ASCII character to or from an I/O buffer (see Figures 6-8 and 6-9). Image Binary data is unchecksummed binary and appears on paper tape, in I/O buffers and on mass storage devices as shown in Figures 6-10 and 6-11.

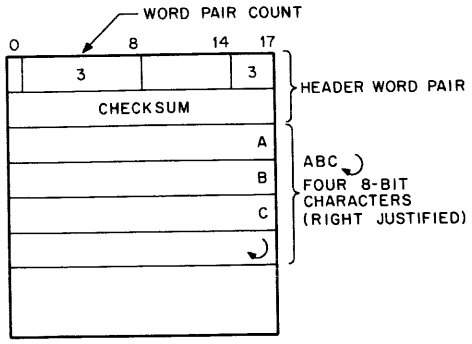


Figure 6-8 Image Alphanumeric Data in I/O Buffers and on Mass Storage Devices

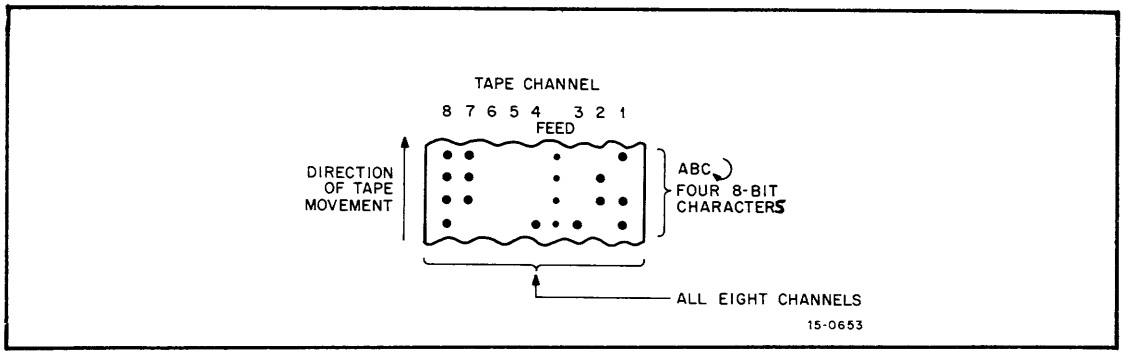


Figure 6-9 Image Alphanumeric Data on Paper Tape

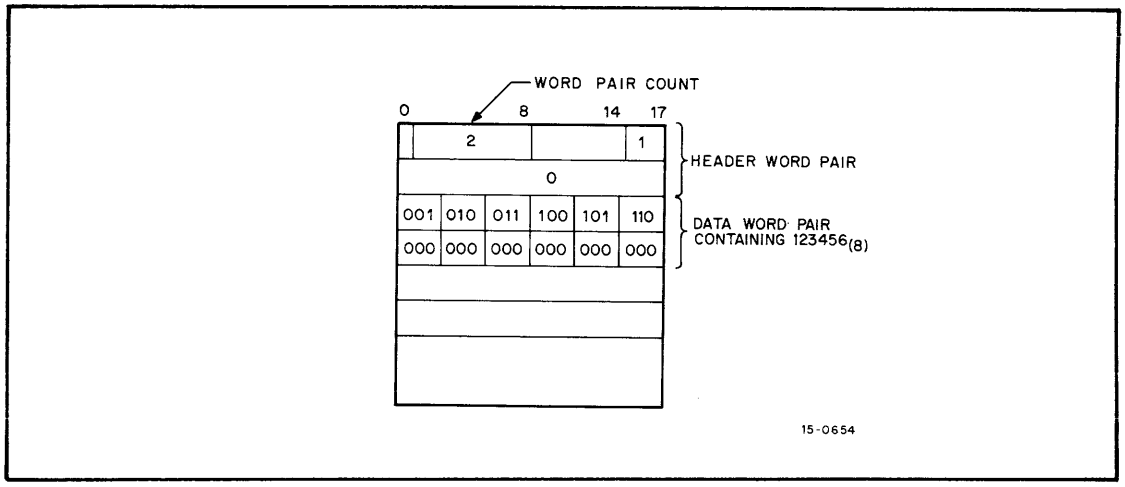


Figure 6-10 Image Binary in I/O Buffers and on Mass Storage Devices

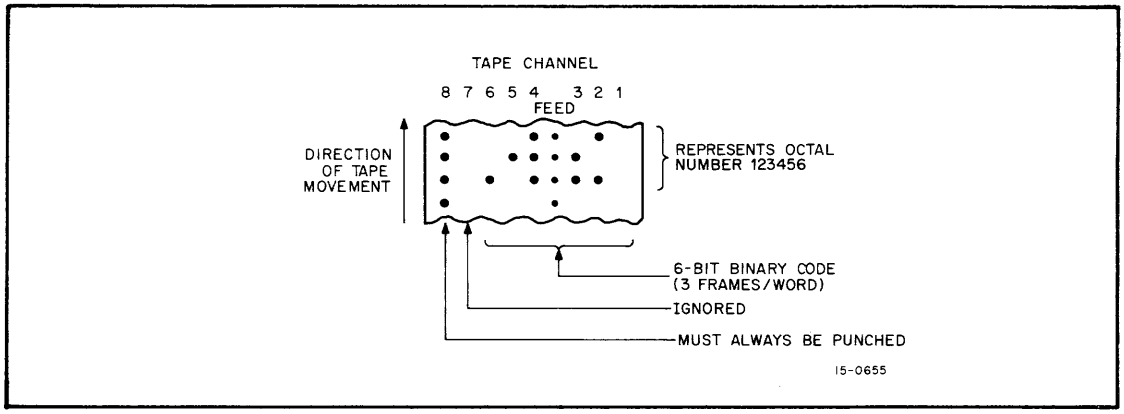


Figure 6-11 Image Binary on Paper Tape

6.4.3 Dump Mode

Dump Mode permits complete freedom in data structuring in the context of a file structured environment[1]. Data input or output in Dump Mode is not interpreted, checked or packed and record length is not limited by the software. Dump Mode is used to output from, or load directly into, any core memory area. All data transferred are treated strictly as 18-bit binary words (i.e., core images). No Header Word Pair is used. Each .READ or .WRITE statement in Dump Mode requires arguments which merely define the core starting address of the area input or output and the number of words (Word Count) to be transferred. Dump Mode is normally used with mass-storage devices although paper tape I/O is possible. With paper tape, data is interpreted on output or input as three six-bit frames per 18-bit word (see Figure 6-11). On input from paper tape, the 7 channel is ignored and the 8 channel must be punched in each data frame. This requirement is automatically met by the paper tape punch handler when output is in dump mode.

6.4.4 Logical Record Terminators

IOPS detects the end of each logical record transferred using a predefined group of terminators or terminating conditions. These logical record terminators vary with the Data Mode in effect, the particular device or set of devices involved, and the transfer direction (input or output). Table 6-2 summarizes the terminating conditions for logical records during input and output for each IOPS Data Mode. Detailed information on terminating conditions for each I/O device handler is provided in Appendix C.

[1]The .TRAN System Macro (paragraph 6.7.13) permits user specified file structuring as well as data structuring. A Dump Mode transfer can be thought of as effectively a file structured .TRAN.

Table 6-2

Logical Record Terminators

Data Mode	Input	Output
IOPS ASCII	Carriage RETURN or ALT MODE Word Pair Count[1] EOM Word Count[2] EOF[1]	Carriage RETURN ALT MODE Word Pair Count[3]
IOPS Binary	Word Pair Count EOM Word Count[2] EOF[1]	Word Pair Count
Image Alpha-numeric and Image Binary	Word Pair Count[1] Word Count EOM EOF[1]	Word Pair Count
Dump	Word Count EOM EOF[1]	Word Count

[1]Mass storage only.

[2]If word count is exceeded before a terminator is encountered, IOPS sets bits 12 and 13 of Header Word Ø to 3 (Buffer Overflow).

[3]If the Word Pair Count is 1 or less, the line is ignored; if greater than 1, ignore the count and accept Carriage RETURN or ALT MODE (non-file oriented devices only). Bulk storage devices require a word Pair Count greater than Ø and less than $2ØØ_8$; otherwise an IOPS 23 error will occur.

6.5 I/O BUFFERS

6.5.1 Space Allocation

Each system or user program which is to perform I/O operations must allocate an I/O Buffer for each I/O device and Unit (i.e., a .DAT slot) that is to be used simultaneously. This can be accomplished by, but is not limited to, one of the methods described below.

Static buffer allocation can be accomplished through the use of the MACRO Assembler's.BLOCK pseudo-op as described in the MACRO Assembler Language Manual. A tag must be used to permit referencing of the buffer in each I/O Macro. For example:

```
        .
        .
        .
        .DEC
INBUF   .BLOCK 52       /CREATES 52(10) WORD BUFFER
                        /CALLED INBUF
OTBUF   .BLOCK 254      /CREATES 254(10) WORD BUFFER
        .OCT            /CALLED OTBUF
        .
        .
        .
```

Dynamic buffer allocation can be performed through the use of the Monitor Macros .GTBUF and .GVBUF described in Chapter 5. Alternatively, the user can create a buffer at execution time from free core by simple calculation, using the information contained in the System Communication Registers, .SCOM+2 and .SCOM+3 (absolute locations 102 and 103 octal). These registers contain the lowest and highest addresses of the registers in unused core, below the system bootstrap respectively.

6.5.2 Size Considerations

When choosing a maximum I/O buffer size to specify in the Word Count arguments for .READ and .WRITE I/O Macros, both the set of possible devices and the Data Mode must be considered. The buffer must reside entirely within the lower 32K of XVM memory. As a general rule, when using IOPS and Image Modes, the maximum logical record size can never exceed the maximum buffer sizes specified in Table 6-3. These are based on physical device limitations. In Dump Mode, however, there are no restrictions on buffer size provided the buffer resides entirely within the lower 32K of XVM memory; a logical record can occupy any number of physical blocks on a mass storage device. Programs which are to communicate with a number of devices using IOPS or Image

Table 6-3

Maximum I/O Buffer Sizes for IOPS and Image Mode Transfers

Device	Max. Buffer Size[1]	Data Modes	Comments
Disk Cartridge (RK)	254 (376_8)	All	IOPS and Image Modes permit more than one logical record (depending on its size) per physical block. Dump Mode records can span an unlimited number of blocks.
DECdisk (DK)	254 (376_8)	All	
Disk Pack (DP)	254 (376_8)	All	
DECtape (DT)	255 (377_8)	All	
Magtape (MT)	255 (377_8)	All	
Teleprinter (TT)	34 (42_8)	IOPS ASCII	Allows for 80 5/7 packed characters.
Plotter (XY) XY11/XY311	34 (42_8)	IOPS ASCII IOPS Binary	
	74	Image Alphanumeric	(Allows for 72 characters)
Paper Tape Reader (PR)	52 (64_8)	All	
Paper Tape Punch (PP)	52 (64_8)	All	
Line Printer LP-15/LP-11 (80 column)	36 (44_8)	IOPS ASCII	Allows for 80 5/7 packed characters
	84	Image Alphanumeric	Allows for 80 characters
Line Printer LP-15/LP-11/ LS-11/LV11 (132 column)	56 (70_8)	IOPS ASCII	Allows for 132 5/7 packed characters.
	136	Image Alphanumeric	Allows for 132 characters
Card Reader (CD) CR03B/CR15/CR11	36 (44_8)	IOPS ASCII	Allows for 80 5/7 packed characters plus a handler supplied Carriage RETURN.
VP15A Storage Tube Display	32 (40_8)	IOPS ASCII	Allows for 75 5/7 characters (only 72 can be displayed)
	74	Image Alphanumeric	Allows for 72 characters

[1] The Octal representation of buffer sizes is shown for values returned after issuing a .INIT, indicating the maximum size which can be used for IOPS Modes. Other sizes shown can be used with Image Modes as specified.

Modes must limit the output buffer size to that of the device with the smallest buffer. Conversely, in setting-up input buffers, the size must be at least as large as that for the device with the largest buffer size.

For example; consider the I/O buffer requirements for a program which must be capable of transferring IOPS ASCII lines to either a teleprinter or the DECdisk. The maximum I/O buffer size for the teleprinter is 34_{10} locations and for the DECdisk is 254_{10} locations. Therefore, the user should choose the smaller size since his program must deal with both. More importantly, if the ASCII records stored on the DECdisk were greater than 34 locations in length, the user could not transfer them back to the teleprinter without truncation or reformatting.

6.6 SPECIFYING I/O DEVICES

As mentioned in preceding chapters, the Monitor maintains a Device Assignment Table (.DAT), which has "slot" numbers that correspond directly to logical device numbers. Each .DAT slot contains a pointer to a device handler and a unit number when applicable (for Disk Pack, DECtape or Magnetic Tape units). All I/O communication in the DOS Monitor environment is accomplished by the logical/physical device associations provided by the Device Assignment Table.

When writing a MACRO program which uses I/O Macros, it is necessary to incorporate the MACRO Assembler's .IODEV pseudo-op somewhere in the program to specify to CHAIN or the Linking Loader which logical device numbers (.DAT slots) are to be used. The .IODEV pseudo-op causes the Assembler to generate a code in the object program which instructs CHAIN or the Linking Loader to load the device handlers associated with the specified .DAT slots.

For example: .IODEV 3,5,6 causes the device handlers assigned to .DAT slots 3, 5, and 6 to be loaded with the program containing the pseudo-op.

6.7 I/O MACRO DESCRIPTIONS

The paragraphs which follow describe the function of the I/O Macros, the information to be provided by the user (arguments), and the assembly language expansion of each[1]. Either the I/O Macros of their

[1]Paragraph 5.2.2 describes conventions and symbology used in presenting the I/O Macros.

assembly language expansions can be incorporated directly in MACRO programs. Typical I/O command sequences are discussed in paragraph 6.8.4. Except for the .TRAN function, all I/O buffers, file name blocks, and addresses must be entirely within the lower 32K of XVM Memory. In general, the system does not check to see that these restrictions have been faithfully applied.

6.7.1 .CLEAR

Initializes all bit maps and directories on the device. Eliminates all references to files in the directory of the device associated with the named .DAT slot. In order to avoid clearing a directory when its files are still in use, the directory is checked for open files. If there are no open files, the directory is cleared; otherwise, an IOPS 1Ø error message (file still active) results. .CLEAR to a system device may only be used under the MIC, because it will destory the system.

FORMAT:

```
.CLEAR␣[-]ds
```

ARGUMENTS:

ds = .DAT slot (octal radix)

EXPANSION: (all values below are octal)

```
LOC+Ø   CAL   [-]ds&777
LOC+1   5
```

6.7.2 .CLOSE

When directed to a .DAT slot used for input: Clears all flags related to that .DAT slot. This deactivates the .DAT slot, and another .INIT will be necessary to use the .DAT slot in the future. Any allocated buffer is returned.

When directed to a .DAT slot used for output: Allows all associated output to finish, and then writes an end-of-file (EOF)[1,2] software indicator in the last header word pair. If the .DAT slot is associated with a directoried device, any earlier file of the same name and extension is deleted from the directory. Operation then continues as for input files (above).

---

[1] ØØ1ØØ5 / 776773 all except non-directoried magtape.

[2] 2005 / 775773 non-directoried magtape.

FORMAT:

.CLOSE␣[-]ds

ARGUMENTS:

ds = .DAT slot (octal radix)

EXPANSION: (all values below are octal)

```
LOC+Ø    CAL␣[-]ds&777
LOC+1    6
```

6.7.3 .DLETE

Deletes a file from the directory of the device associated with the named .DAT slot.

If the specified file cannot be found, the contents of the AC will be Ø on return.

FORMAT:

.DLETE␣[-]ds,namptr

ARGUMENTS:

ds = .DAT slot number (in octal radix)

namptr = Pointer to the address of the first of 3 words containing the .SIXBT representation of the name and extension of the file to be deleted (unused spaces must contain nulls). The 3-word block must reside entirely within the first 32K of XVM memory.

EXPANSION: (all values below are octal)

```
LOC+Ø    CAL+1ØØØ␣[-]ds&777
LOC+1    2
LOC+2    namptr
```

6.7.4 .ENTER

Initializes a directory for a new output file. The file will be placed on the device associated with the .DAT slot named as one of the parameters to the macro. Transfers control to the Monitor error handler to print the appropriate error message if there is no available space. in the file directory. ENTER is ignored by non-directored output devices.

FORMAT:

.ENTER␣[-]ds,namptr,p

ARGUMENT:

ds = .DAT slot (octal radix)

namptr = Pointer to the address of the first of three words containing the .SIXBT representation of the name and extension of the file to be created. (Unused spaces must contain nulls). The 3-word block must reside entirely within the first 32K of XVM memory.

p = Disk file protection code to be assigned. Possible values for p:

1 - file is unprotected
2 - file is WRITE protected
3 - file is READ/WRITE protected

Is p is omitted, the default protection code established at system generation time is used.

EXPANSION: (all values below are octal)

```
LOC+Ø   CAL+1ØØØ*p␣[-]ds&777
LOC+1   4
LOC+2   namptr
```

6.7.5 .FSTAT

Checks the status of a file in a directory. Ignored by non-directoried devices. The namptr parameter points to the name and extension of the file whose status is desired. On return, the AC will contain the first block number of the file, if the directory lists a file with the indicated name. The contents of the AC will be zero on return if the specified file is not on the device. It is recommended that programmers use .FSTAT prior to executing .SEEK if they wish to retain program control when a file is not in the directory.

Bits Ø through 2 of LOC+2 must be set to zero prior to the execution of the CAL. On return, these bits of LOC+2 will contain a code indicating the type of device associated with .DAT slot ds.

Ø = non-directoried device
1 = DECtape file structure
2 = RF DECdisk file structure
3 = RP Disk Pack file structure
4 = MAGtape
5 = RK Disk Cartridge file structure

FORMAT:

.FSTAT␣[-]ds,namptr

ARGUMENTS:

ds = .DAT slot (octal radix)

namptr = Pointer to the address of the first of three words containing the .SIXBT representation of name and extension of the file (unused spaces must contain nulls). The 3-word block must reside entirely within the first 32K of XVM memory.

EXPANSION: (all values below are octal)

```
LOC+Ø   CAL+3ØØØ␣[-]ds&777
LOC+1   2
LOC+2   namptr
```

6.7.6 .INIT

Initializes a device and device handler. Programmers must give a .INIT prior to giving any I/O commands referencing the named .DAT slot. Any .DAT slot initialized via a .INIT must eventually be closed via a .CLOSE.

The handler that services any .INIT will return (in LOC+3 of the expansion) the maximum size of the line buffer allowed for that handler. For example, DTA will return 377 (255_{10}) in LOC+3, and DKA will return 376 (254_{10}) in LOC+3 of the expansion.

FORMAT: .INIT␣[-]ds,dd,restrt

ARGUMENTS:

ds = .DAT slot (octal radix)

dd = Direction of data flow

Ø - file will be an input file
1 - file will be an output file

restrt = Restart address. Although restrt must be included to avoid assembly time errors, it has meaning only for .INIT commands referencing. .DAT slots assigned to the teleprinter. The restart address must reside within the lower 32K of XVM memory.

EXPANSION: (all values below are octal)

```
LOC+Ø   CAL+dd*1ØØØ␣[-]ds&777
LOC+1   1
LOC+2   restrt
LOC+3   Ø (Handler will return maximum buffer size
          in LOC+3.)
```

NOTE

Bits 5, 6 and 8 of LOC+Ø provide added information for the VT15 XVM software, the line printer handler and the disk handlers. See the individual handler descriptions for more information. Bit 7 of LOC+Ø should always be Ø.

6.7.7 .MTAPE

Performs functions unique to industry-standard magnetic tape, i.e., functions for non-directoried magnetic tape. Limited functions are also provided on DECtape and disk (see Chapter 9). See descriptions of appropriate handlers for more information.

FORMAT:

```
.MTAPE␣[-]ds,nn
```

ARGUMENTS:

ds = .DAT slot number (in octal radix)

nn = code number of magnetic tape function or configuration:

- ØØ - rewind to load point
- Ø2 - backspace record
- Ø3 - backspace file
- Ø4 - write end-of-file
- Ø5 - skip record
- Ø6 - skip file
- Ø7 - skip to logical end of tape
- 1Ø - 7-channel, even parity, 2ØØ bpi
- 11 - 7-channel, even parity, 556 bpi
- 12 - 7-channel, even parity, 8ØØ bpi
- 13 - 9-channel, even parity, 8ØØ bpi
- 14 - 7-channel, odd parity, 2ØØ bpi
- 15 - 7-channel, odd parity, 556 bpi
- 16 - 7-channel, odd parity, 8ØØ bpi
- 17 - 9-channel, odd parity, 8ØØ bpi

EXPANSION: (all values below are octal)

```
LOC+Ø   CAL+nn*1ØØØ␣[-]ds&777
LOC+1   7
```

6.7.8 .RAND

Opens a disk file for random access via .RTRAN macros. Returns the number of blocks in the file in LOC+3 of the expansion.

FORMAT:

.RAND␣[-]ds,namptr

ARGUMENTS:

ds = .DAT slot (octal radix)

namptr = Pointer to the first word of a 3-word .SIXBT representation of the filename and extension of the file to be opened (unused spaces must contain nulls). The 3-word block must reside entirely within the lower 32K of XVM memory.

EXPANSION: (all values below are octal)

```
LOC+Ø   CAL+5ØØØ␣[-]ds&777
LOC+1   2
LOC+2   namptr
LOC+3   Ø
```

6.7.9 .READ

Returns the next logical record to the user's I/O buffer. If the record is longer than the user's I/O buffer, the handler will fill the buffer and the rest of the record will be lost. If the record is shorter than the user's I/O buffer, the handler will use only the part of the buffer it needs. The handlers use several indicators to determine the length of record. Appendix C, Input/ Output Data Mode Terminators, shows which handlers use what indicators for each data mode.

Since I/O operations and internal data transfers may proceed asynchronously with computation, a .WAIT command must be used after a .READ command before the user attempts to access the data in his I/O buffer, or to read another line into it.

The user should always check bits 14 through 17 of the first word of the I/O buffer for end-of-medium and end-of-file conditions. In non-Dump modes, the user should also interrogate bits 12 and 13 of the first word of the I/O buffer to ensure that the record was read without error.

FORMAT:

.READ␣[-]ds,m,bufadd,wdc

ARGUMENTS:

ds = .DAT slot (octal radix)

m = Data Mode, as follows:

Ø - Data Mode is IOPS Binary
1 - Data Mode is Image Binary
2 - Data Mode is IOPS ASCII
3 - Data Mode is Image Alphanumeric
4 - Data Mode is Dump Mode

bufadd = address of user's I/O buffer

wdc = word count (decimal radix), including the two-word header. wdc should equal the length of the buffer at bufadd.

EXPANSION: (all values below are octal)

```
LOC+Ø   CAL+m*1ØØØ␣[-]ds&777
LOC+1   1Ø
LOC+2   bufadd
LOC+3   -wdc
```

NOTE

The user must assume that the I/O buffer lies entirely within the lower 32K of XVM memory.

6.7.10 .RENAM

Renames a file (useful only with directoried devices). The old name and extension are pointed to by namptr; the new name must be located at namptr+3. The AC will be zero on return if the file specified at namptr cannot be found.

FORMAT:

```
.RENAM␣[-]ds,namptr
```

ARGUMENTS:

ds = .DAT slot (octal radix)

namptr = pointer to the old name and extension (new name is at namptr+3. (Unused spaces must contain nulls.) The 6-word block must reside entirely within the lower 32K of XVM memory.

EXPANSION: (all values below are octal)

```
LOC+Ø   CAL+2ØØØ␣[-]ds&777
LOC+1   2
LOC+2   namptr
```

6.7.11 .RTRAN

Allows random access to blocks in a file opened via a .RAND I/O macro. Programmers may read or write into any block of a file, but may not

change the length of the file.

FORMAT:

.RTRAN␣[-]ds,d,relblk,bufadd,beg,wdc

ARGUMENTS:

ds = .DAT slot (octal radix)

d = direction:

Ø - direction is input
1 - direction is output

relblk = block number (octal radix) relative to beginning of the file ... first block is block 1, etc.

bufadd = address of I/O buffer in user's core space.

beg = first physical word of physical block to be read or written...ignored for output to disk pack and disk cartridge; must be octal radix, Ø≤beg≤375.

wdc = number of words, starting with beg, to be read or written...ignored for output to disk pack and disk cartridge; must be DECIMAL radix, Ø≤wdc≤(253-beg).

EXPANSION: (all values below are octal)

```
LOC+Ø   CAL+4ØØØ␣[-]ds&777
LOC+1   2
LOC+2   d*4ØØØØØ+relblk
LOC+3   bufadd
LOC+4   beg
        .DEC
LOC+5   -wdc
```

NOTE

The user must assume that the I/O buffer lies entirely within the loader 32K of XVM memory.

6.7.12 .SEEK

Opens a file for input on a directoried device. .SEEK is ignored by non-directoried input devices.

If the file is listed in the device's directory, the handler reads the first block of the file into a buffer (A subsequent .READ will obtain the first logical record in the user's I/O buffer.)

If the file is not listed in the device's directory, the handler will pass control to the error handling routine in the Monitor. If the

user wishes to retain control in the case that the desired file is not in the directory, he should first issue an .FSTAT command.

FORMAT:

.SEEK␣[-]ds,namptr

ARGUMENTS:

ds = .DAT slot (octal radix)

namptr = pointer to the three-word .SIXBT representation of the name and extension of the file to be opened for input. (Unused spaces must be null-filled.) The 3-word block must reside entirely within the lower 32K of XVM memory.

EXPANSION: (all values below are octal)

```
LOC+Ø   CAL␣[-]ds&777
LOC+1   3
LOC+2   namptr
```

6.7.13 .TRAN

Allows device-dependent, non-directoried input and output to any mass storage device. Users address such blocks by their physical locations. .TRAN should be followed by a .WAIT macro, to ensure that the transfer has been completed.

FORMAT:

.TRAN␣[-]ds,d,blknum,bufadd,wdc

ARGUMENTS:

ds = .DAT slot (octal radix)

d = direction of transfer:

Ø - input, forward
1 - output, forward
2 - input, backward (DECtape only)
3 - output, backward (DECtape only)
4 - Input, true 9-track (Magtape only)
5 - output, true 9-track (Magtape only)

NOTE

DECtape blocks must be read in the direction they were written in, in order to obtain meaningful results.

blknum = block number (octal radix) of block at which to start input or output; I/O with word count larger than one block will continue with the next contiguous block; end of tape or disk will cause IOPS error.

bufadd = address of I/O buffer; I/O buffer must be at least as long as wdc. The buffer may reside anywhere in XVM memory, including core above the first 32K.

wdc = number of words to be transferred (decimal radix). The maximum word count permitted is 131,072.

EXPANSION: (all values below are octal):

```
LOG+Ø    CAL+d*1ØØØ␣[-]ds&777
LOC+1    13
LOC+2    blknum
LOC+3    bufadd
LOC+4    -wdc
```

6.7.14 .USER

Allows users to dynamically assign UIC's to desired UFDT slots, thus permitting access to UFD's other than the UFD specified at LOGIN time (refer to paragraph 8.4). Protection for the specified UFD and its files remains in effect. .USER must be issued before the .INIT of the I/O sequence for which UFD change is desired (see example below). .USER will also take effect if the proceeding I/O call to the same .DAT slot, as that given in the .USER, is .CLOSE, .RENAM, .MTAPE (rewind), or .DLETE.

FORMAT:

```
.USER␣[-]ds,uic
```

ARGUMENTS:

ds = .UFDT/.DAT slot number (octal radix)

uic = (UIC) the .SIXBT representation of the three-character User Identification Code.

EXPANSION: (all values below are octal)

```
LOC+Ø    CAL␣[-]ds&777
LOC+1    23
LOC+2    uic
```

For example:

```
.USER 1,ABC
.INIT 1,ØØ
.USER 1,CDE
    .
.SEEK 1,FILE      This .SEEK searches for FILE under the
    .             UFD called ABC.
.CLOSE 1
    .
    .
```

```
.INIT 1,Ø,Ø
.SEEK 1, FILE     This .SEEK searches for FILE under the
     .            UFD called CDE.
     .
```

6.7.15 .WAIT

Obtains and holds control until the user's I/O buffer is available after an I/O operation. Should be used before accessing an I/O buffer after .READ, .WRITE, .TRAN, and .RTRAN commands.

FORMAT:

.WAIT␣[-]ds

ARGUMENT:

ds = .DAT slot (octal radix)

EXPANSION: (all values below are octal)

```
LOC+Ø   CAL␣[-]ds&777
LOC+1   12
```

6.7.16 .WAITR

Returns control to the address specified as an argument to the CAL, if I/O is not complete. If I/O is complete, returns control to the next location after the macro expansion. It is the user's responsibility to return to the .WAITR command, or do another one.

FORMAT:

.WAITR␣[-]ds,waitad

ARGUMENTS:

ds = .DAT slot (octal radix)

waitad = address to which control will be returned, if I/O is incomplete. The address must reside within the lower 32K of XVM memory.

EXPANSION: (all values below are octal)

```
LOC+Ø   CAL+1ØØØ␣[-]ds&777
LOC+1   12
LOC+2   waitad
```

6.7.17 .WRITE

.WRITE transfers a logical record from the user's I/O buffer to the handler's buffer.

The .WRITE command establishes the mode in which data is transferred. The Header Word Pair Count determines the maximum amount of data to be transferred in all modes except Dump, which references the "wdc" argument.

The only limits on data transferred in Dump Mode are the amount of data buffer space available and the capacity of the device accepting the data. On physically blocked devices, such as DECtape and disk, the handler will start from its current position in a block and fill successive blocks until the transfer is complete. If a Dump Mode transfer does not completely fill the last block used, a subsequent Dump Mode transfer will fill that block, before using any other.

.WAIT or .WAITR must be used after a .WRITE command, before the user's I/O buffer is used again, to ensure that the transfer to the device has been completed.

FORMAT:

.WRITE␣[-]ds,m,bufadd,wdc

ARGUMENTS:

ds = .DAT slot (octal radix)

m = data mode for transfer

Ø - IOPS Binary
1 - Image Binary
2 - IOPS ASCII
3 - Image Alphanumeric
4 - Dump

bufadd = address of user's I/O buffer containing data to be transferred.

wdc = word count, number of words to be transferred (decimal), (relevant for Dump Mode transfers only).

EXPANSION: (all values below are octal)

```
LOC+Ø   CAL+m*1ØØØ␣[-]ds&777
LOC+1   11
LOC+2   bufadd
LOC+3   -wdc
```

NOTE

The user must assure that the I/O buffer lies entirely within the lower 32K of XVM memory.

6.8 USING I/O MACROS

The programmer must observe certain conventions when incorporating I/O Macros into a program. In general, consideration must be given to:

- The physical device and its capabilities
- I/O device handler characteristics
- The I/O Macro Syntax
- The I/O Macro Sequence for the desired file access/ structure (see Chapter 4)

6.8.1 Physical Device Capabilities

The considerations involved here are obvious and need little comment. Simply stated, the user must understand the gross differences between various devices. For example, a .READ cannot be issued to a line printer. Similarly, binary data cannot be output to a teleprinter.

6.8.2 Device Handler Characteristics

Many of the standard XVM I/O devices available to DOS users are provided with several handler versions. These versions vary from one to another as to the I/O Macros and Data Modes which are acceptable to them. Some versions permit the full set of I/O Macros and Data Modes to be used, while others incorporate a subset of these features. These limited capability handlers are provided primarily for use where core allocation is a problem, since they are smaller than those with greater capability. The user must be aware of these handler differences, particularly if he wishes to utilize device dependent characteristics. Detailed descriptions of the DOS device handlers are provided in Chapter 9.

6.8.3 I/O Macro Syntax

The order in which I/O Macros are used is important to the success of an I/O transfer. There are basic rules of syntax which must be adhered to in order to avoid run time I/O (IOPS) errors. These rules apply to any I/O sequence directed to the same .DAT Slot.

1. .INIT must always be issued before any other I/O Macro. It initializes the handler associated with the referenced .DAT slot for either input or output. Subsequent .INITs can be used to unconditionally terminate an unwanted I/O operation. Exception: A .INIT will not terminate I/O to the teletype. See the description of the Teletype handler in Chapter 9 for a method to stop I/O in progress.

Table 6-4

Legal I/O Macro Combinations

Commands which may follow:	Commands which may precede															
	Misc.	Directory Maintenance				File Structure Initialize				Data Transfer				Misc.		
	.INIT	.FSTAT	.RENAM	.DLETE	.CLEAR	.SEEK	.ENTER	.RAND	.MTAPE	.READ	.WRITE	.RTRAN	.TRAN	.WAIT	.WAITR	.CLOSE
.INIT	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
.FSTAT	X	X	X	X	X								X	X	X	X
.RENAM	X	X	X	X	?								X	X	X	X
.DLETE	X	X	X	X	?								X	X	X	X
.CLEAR	X	X	X	X	?								X	X	X	X
.SEEK	X	X	X	X	?								X	X	X	X
.ENTER	X	X	X	X	X								X	X	X	X
.RAND	X	X	X	X									X	X	X	X
.MTAPE	X	X	X	X	X				X				X	X	X	X
.READ	X					X			X	X				X	X	X
.WRITE	X						X		X		X			X	X	X
.RTRAN	X							X				X		X	X	X
.TRAN	X	X	X	X	X								X	X	X	X
.WAIT	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
.WAITR	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
.CLOSE	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
.USER	?	?	X	X	?	?	?	?	?	?	?	?	X	?	?	X

Notes: ? = Illogical Combination
X = Legal combination
Blank = Illegal Combination

2. .CLOSE (or .MTAPE Rewind, for Magtape) must always be the last Macro issued in the sequence. It terminates the current I/O operation and use of the associated .DAT slot. Another .INIT must be issued if subsequent transfers are to be made via this .DAT slot.

3. I/O transfers to directoried devices (Disk, DECtape or Magtape) require the use of .SEEK before .READ, .ENTER before .WRITE and .RAND before .RTRAN.

4. Intermixing of .SEEK, .ENTER, .MTAPE, and .RAND Macros in the same I/O sequence (i.e., occurring between an .INIT and a .CLOSE referencing the same .DAT slot) is illegal.

The acceptability of specific combinations of I/O Macros is shown in Table 6-4.

6.8.4 Selecting an I/O Macro Sequence

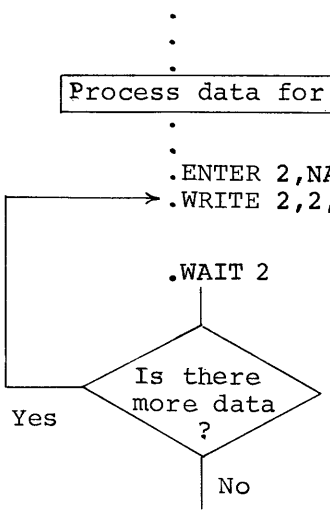
The user has the capability of selecting command sequences which emphasize either device independence or device dependence, as his needs dictate. Much of the device independence of the DOS system is obtained by using I/O Macro sequences which are general enough to be acceptable to a wide range of device handlers. These sequences are based on sequential file access via the .READ and .WRITE I/O Macros, which are recognized by all DOS Device Handlers. Simply by using .INIT - .READ (or .WRITE) - .CLOSE sequences along with the applicable IOPS Data Modes, a user's program has the capability of communicating with all non-mass storage devices (including teleprinter, paper tape reader and punch, line printer, card reader and VP15A CRT). By adding the .FSTAT, .SEEK, and .ENTER Macros, the same program can, in addition, utilize the system's mass storage devices (including Disk, DECtape and Magtape). The examples below further illustrate these sequences.

Example 1 - Typical Output Sequence

```
.IODEV 2              /Use .DAT slot 2
   .
   .
.INIT  2,1,RSTRT      /Initialize .DAT slot 2 for output (1)
   .
   .
```

Optional check for existing file and device type, e.g., .FSTAT, .RENAM, .DLETE, etc.

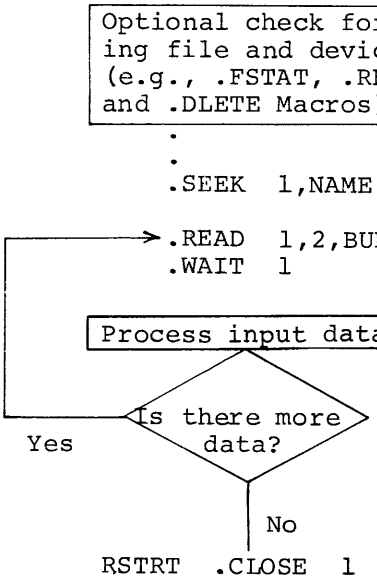
Example (cont'd)




```
                .ENTER 2,NAME
                .WRITE 2,2,BUFF,34      /Output a record to .DAT slot 2
                                        /from BUFF in IOPS ASCII Mode.
                .WAIT 2                 /Wait till buffer is free

                .CLOSE 2
NAME            .SIXBT   "FIL@@@SRC"    /File name in .SIXBT
BUFF            .BLOCK 42               /Spaces are null filled (i.e., @=null)
                                        /I/O buffer
```

Example 2 - Input Sequence




```
                .IODEV 1                /Use .DAT slot 1

                .INIT  1,Ø,RSTRT        /Initialize .DAT slot 1 for input(Ø)

                .SEEK  1,NAME

                .READ  1,2,BUFF,34
                .WAIT  1

RSTRT           .CLOSE  1

NAME            .SIXBT     "FIL@@@SRC"  /File name in .SIXBT.  Spaces are
BUFF            .BLOCK 42               /null filled (i.e., @=null)
                                        /I/O buffer
```

Device dependent I/O programming is based in large part on the use of device-specific I/O Macro sequences and to a lesser degree on the structure (Data Mode) of the data. For example: .RTRAN can only be used with disk handlers; non-mass storage devices do not need .SEEK or .ENTER for operation (they ignore them). Furthermore, the interpretation of data using the Image and Dump Data Modes becomes the user's responsibility, rather than the handler's. The user who does not need device independence can eliminate unnecessary instructions in his program and obtain additional free core for other purposes. For instance, assume that in the previous examples a directoried device was never to be used. The programmer could then economize by eliminating all I/O Macros associated with I/O on directoried devices such as .SEEK, .ENTER, .CLEAR, .DLETE, .FSTAT.

Figure 6-12 shows the relationship between standard I/O Macro sequences and the various I/O devices and file structure/access features of the DOS-15 system. Except for .INIT and .CLOSE, there is no implied order to the macros in any particular I/O sequence shown (the reader should refer to paragraphs 6.7 and 6.8.3.1 for this information).

6.8.5 Programming Example

The following example illustrates the use of I/O Macros in a MACRO Assembly Language program. The program accepts an ASCII line from the teleprinter keyboard, creates a file on the disk (or if the .DAT slot is properly assigned, on any other directoried device), reads the file back from the disk, and prints it on the teleprinter. Before subsequent keyboard inputs, the program prints the following message on the teleprinter:

```
FILE ALREADY PRESENT!!

DO YOU WISH TO KEEP IT ?(Y OR N AND CR)>
```

By typing a Y on the keyboard, the file previously created is saved and a new file is created for the next line input from the keyboard. By typing an N rather than a Y, the next line input from the keyboard is written on the disk using the filename associated with the line previously typed (that line will be deleted).

The name of the file is initially ECHO 001. A new file name is created each time a Y is typed in response to the above message by

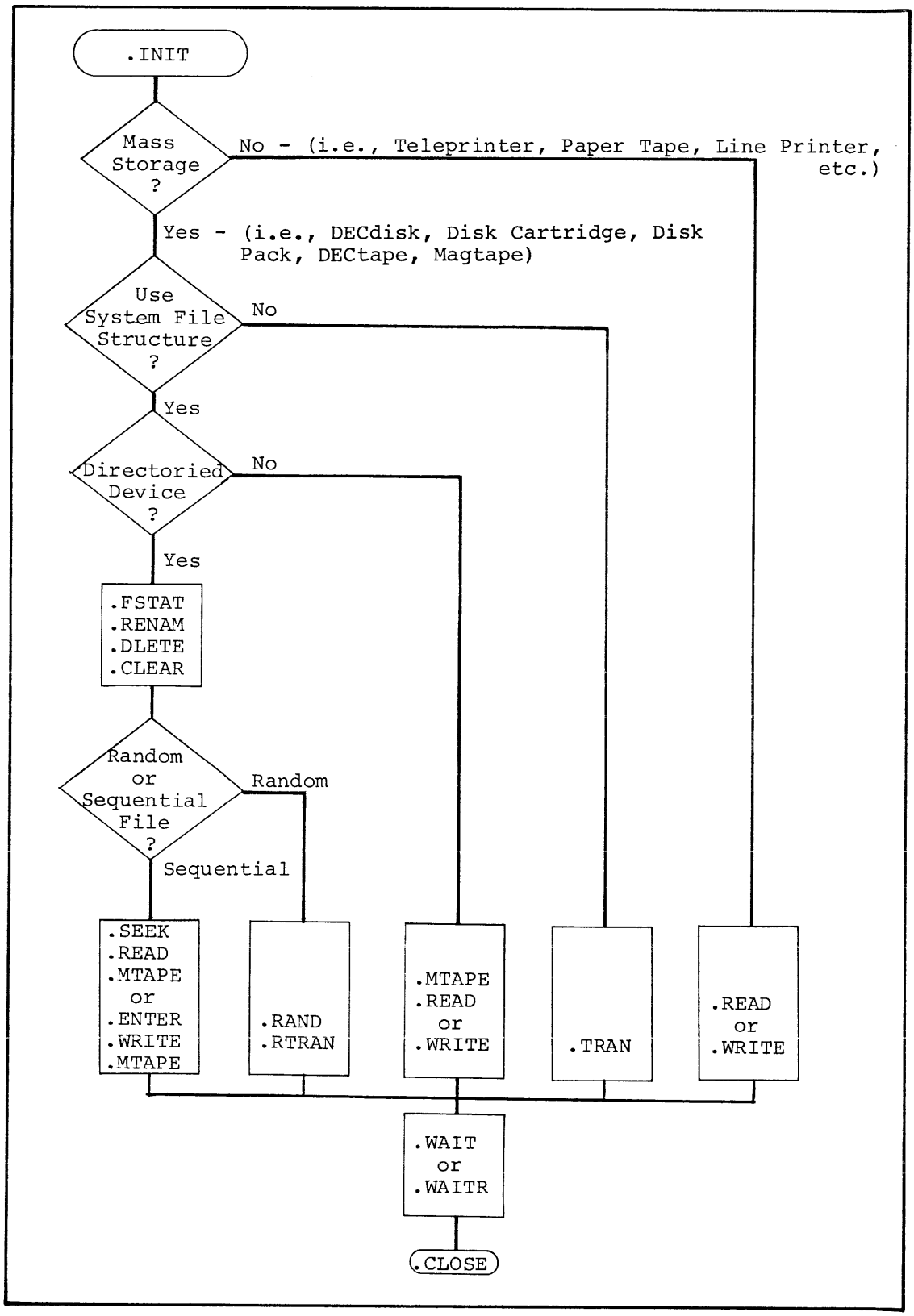


Figure 6-12
I/O Macros Applicable to Specific Devices and Data Access Techniques

incrementing the location in the program called NAME+2 which initially contains the "001" extension of the file name in Sixbit trimmed ASCII (.SIXBT pseudo-op). This produces a series of uniquely named files (one each time a Y is typed) as follows: ECHO 001, ECHO 002, ECHO 003, ECHO 004, etc.

The arguments used by the I/O Macros in this program are given symbolic names by means of MACRO direct assignment statements at the beginning of the program. These names permit the programmer to change the real values of these arguments readily and also facilitate recall. The listing shown below is the source listing and is followed by an assembly listing which shows how Macros are expanded at assembly time. The reader may wish to compare these expansions with the Macro descriptions in the beginning of this chapter.

Programming Example - Source Listing

```
        .TITLE DKECHO
DISK=7
TTI=6
TTO=5
IN=0
OUT=1
IOPS=2
        .IODEV  5,6,7
BEGIN   .INIT   DISK,OUT,RESTRT         /INITIALIZE DISK OUTPUT,
        .INIT   TTI,IN,RESTRT           /TELETYPE INPUT,
        .INIT   TTO,OUT,RESTRT          /AND TELETYPE OUTPUT
START   .FSTAT  DISK,NAME               /IS FILE PRESENT?
        SZA                             /NO, INPUT KEYBOARD
        JMP     UPDATE                  /YES, OUTPUT MSG1 AND MSG2
READKB  .WRITE  TTO,IOPS,MSG3,34        /TYPE A GO AHEAD SYMBOL (>)
        .READ   TTI,IOPS,BUFFER,34      /INPUT IOPS ASCII FROM KEYBOARD
        .WAIT   TTI                     /WAIT UNTIL INPUT COMPLETE
        .EJECT
        LAC     UDSW                    /TEST UPDATE SWITCH
        SZA                             /0 REPLACE INPUT FILE
        JMP     NEWFIL                  /-1=SAVE INPUT; CREATE NEW OUTPT
WRITE   .ENTER  DISK,NAME               /CREATE NEW DISK FILE
        .WRITE  DISK,IOPS,BUFFER,34     /OUTPUT DATA ON DISK
        .WAIT   DISK                    /WAIT UNTIL OUTPUT COMPLETED
        .CLOSE  DISK                    /CLOSE FILE
READDT  .INIT   DISK,IN,RESTRT          /INITIALIZE DISK INPUT
        .SEEK   DISK,NAME               /LOCATE FILE "NAME"
        .READ   DISK,IOPS,BUFFER,34     /READ INTO BUFFER
        .WAIT   DISK                    /WAIT UNTIL READ COMPLETE
        .EJECT
        .WRITE  TTO,IOPS,BUFFER,34      /OUTPUT TO TELETYPE
        .WAIT   TTO                     /WAIT UNTIL OUTPUT COMPLETE
RESTRT  .CLOSE  TTO                     /TERMINATE TELETYPE OUTPUT,
        .CLOSE  TTI                     /TELETYPE INPUT,
        .CLOSE  DISK                    /AND DISK INPUT/OUTPUT
        JMP     BEGIN                   /LOOP FOR UPDATE OPTION
```

Programming Example - Source Listing (Cont.)

```
UPDATE  .WRITE  TTO,IOPS,MSG1,34        /OUTPUT MSG1
        .WAIT   TTO                     /AND MSG2
        .EJECT
        .WRITE  TTO,IOPS,MSG2,34        /ON
        .WAIT   TTO                     /TELETYPE
        .READ   TTI,IOPS,COM,8          /READ RESPONSE
        .WAIT   TTI                     /WAIT UNTIL READ COMPLETE
        LAC     COM+2                   /GET FIRST WORD
        AND     (774000                 /SAVE FIRST SEVEN BITS
        SAD     (544000                 /IS CHAR A Y?
        JMP     YES
        DZM     UDSW                    /NO, SET TO REPLACE INPUT FILE
        JMP     READKB                  /LOOP TO READ KEYBOARD
YES     CLC                             /SET UPDATE SW. TO SAVE
        DAC     UDSW                    /INPUT, CREATE NEW OUTPUT
        JMP     READKB                  /LOOP TO READ KEYBOARD
NEWFIL  LAC     NAME+2                  /CHANGE EXT
        LRS     3                       /LEAST SIGNIFICANT DIGIT OF
                                        /SIXBT VALUE OF LAST CHAR IN EXT
        RTR                             /STRIP OFF HIGH PART OF CODE
        RAR
        LRS     3                       /LEAST SIGNIFICANT DIGIT OF
                                        /SIXBT VALUE OF MIDDLE CHAR IN EXT
        RTR                             /STRIP OFF HIGH PART OF SIXBT CODE
        RAR
        LLS     6                       /PUT BACK IN AC
        AND     (777                    /STRIP OF HIGH ORDER PART OF REMAININ(
                                        /SIXBT CODE FOR LAST EXT CHAR
        TAD     (1                      /INCREMENT TO MAKE NEW EXT
        LRS     6                       /REVERSE PROCESS TO FIX UP EXT IN
                                        /PROPER SIXBT
        RTL
        RAL
        LLS     3
        RTL
        RAL
        LLS     3
        AND     (070707
        XOR     (606060
        DAC     NAME+2
        JMP     WRITE                   /TO CREATE NEW OUTPUT
MSG1    MSG2-MSG1/2+1000                /WPC FOR HEADER WORD 0
        0
        .ASCII "FILE ALREADY "
        .ASCII "PRESENT!!"<15>
        .EJECT
MSG2    MSG3-MSG2/2+1000                /WPC FOR HEADER WORD 0
        0
        .ASCII "DO YOU WISH TO KEEP IT ?"
        .ASCII "(Y OR N AND CR)  >"<175>
MSG3    COM-MSG3/2+1000
        0
        .ASCII ">"<175>
COM     .BLOCK  10
BUFFER  .BLOCK  42
NAME    .SIXBT  "ECHO@@001"
UDSW    0
        .END    BEGIN
```

Programming Example - Assembly Listing

```
PAGE    1      DKECHO 001      DKECHO

    1                                          .TITLE DkECHO
    2                  000007 A     DISK=7
    3                  000006 A     TTI=6
    4                  000005 A     TTO=5
    5                  000000 A     IN=0
    6                  000001 A     OUT=1
    7                  000002 A     IOPS=2
    8                                          .IODEV  5,6,7
    9          00000 R                 BEGIN   .INIT   DISK,OUT,RESTRT          /INITIALIZE DISK OUTPUT,
               00000 R 001007 A *G             CAL*OUT*1000 DISK&777
               00001 R 000001 A *G             1
               00002 R 000074 R *G             RESTRT+0
               00003 R 000000 A *G             0
   10                                          .INIT   TTI,IN,RESTRT          /TELETYPE INPUT,
               00004 R 000006 A *G             CAL*IN*1000 TTI&777
               00005 R 000001 A *G             1
               00006 R 000074 R *G             RESTRT+0
               00007 R 000000 A *G             0
   11                                          .INIT   TTO,OUT,RESTRT         /AND TELETYPE OUTPUT
               00010 R 001005 A *G             CAL*OUT*1000 TTO&777
               00011 R 000001 A *G             1
               00012 R 000074 R *G             RESTRT+0
               00013 R 000000 A *G             0
   12          00014 R                 START   .FSTAT  DISK,NAME              /IS FILE PRESENT?
               00014 R 003007 A *G             CAL*3000 DISK&777
               00015 R 000002 A *G             2
               00016 R 000301 R *G             NAME
   13          00017 R 740200 A                SZA                            /NO, INPUT KEYBOARD
   14          00020 R 600103 R                JMP     UPDATE                 /YES, OUTPUT MSG1 AND MSG2
   15          00021 R                 READKB  .WRITE  TTO,IOPS,MSG3,34       /TYPE A GO AHEAD SYMBOL (>)
               00021 R 002005 A *G             CAL*IOPS*1000 TTO&777
               00022 R 000011 A *G             11
               00023 R 000223 R *G             MSG3
                               *G              .DEC
               00024 R 777736 A *G             -34
                                              .EJECT
```

Programming Example - Assembly Listing (Cont.)

```
PAGE    2     DKECHO 001     DKECHO

  16                                          .READ   TTI,IOPS,BUFFER,34      /INPUT IOPS ASCII FROM KEYBOARD
            00025 R 002006 A *G               CAL*IOPS*1000 TTI&777
            00026 R 000010 A *G               10
            00027 R 000237 R *G               BUFFER
                            *G                .DEC
            00030 R 777736 A *G               -34
  17                                          .WAIT   TTI                     /WAIT UNTIL INPUT COMPLETE
            00031 R 000006 A *G               CAL TTI&777
            00032 R 000012 A *G               12
  18
  19        00033 R 200304 R                  LAC     UDSW                    /TEST UPDATE SWITCH
  20        00034 R 740200 A                  SZA                             /0 REPLACE INPUT FILE
  21        00035 R 600136 R                  JMP     NEWFIL                  /=1=SAVE INPUT; CREATE NEW OUTPT
  22        00036 R                   WRITE   .ENTER  DISK,NAME               /CREATE NEW DISK FILE
            00036 R 000007 A *G               +CAL*1000 DISK&777
            00037 R 000004 A *G               4
            00040 R 000301 R *G               NAME
  23                                          .WRITE  DISK,IOPS,BUFFER,34     /OUTPUT DATA ON DISK
            00041 R 002007 A *G               CAL*IOPS*1000 DISK&777
            00042 R 000011 A *G               11
            00043 R 000237 R *G               BUFFER
                            *G                .DEC
            00044 R 777736 A *G               -34
  24                                          .WAIT   DISK                    /WAIT UNTIL OUTPUT COMPLETED
            00045 R 000007 A *G               CAL DISK&777
            00046 R 000012 A *G               12
  25                                          .CLOSE  DISK                    /CLOSE FILE
            00047 R 000007 A *G               CAL DISK&777
            00050 R 000006 A *G               6
  26        00051 R                   READDT  .INIT   DISK,IN,RESTRT          /INITIALIZE DISK INPUT
            00051 R 000007 A *G               CAL*IN*1000 DISK&777
            00052 R 000001 A *G               1
            00053 R 000074 R *G               RESTRT*0
            00054 R 000000 A *G               0
                                              .EJECT
```

```
PAGE   3     DKECHO 001     DKECHO

27                                          ,SEEK   DISK,NAME              /LOCATE FILE "NAME"
          00055 R 000007 A *G               CAL DISK&777
          00056 R 000003 A *G               3
          00057 R 000301 R *G               NAME
28                                          ,READ   DISK,IOPS,BUFFER,34    /READ INTO BUFFER
          00060 R 002007 A *G               CAL*IOPS*1000 DISK&777
          00061 R 000010 A *G               10
          00062 R 000237 R *G               BUFFER
                          *G                ,DEC
          00063 R 777736 A *G               -34
29                                          ,WAIT   DISK                   /WAIT UNTIL READ COMPLETE
          00064 R 000007 A *G               CAL DISK&777
          00065 R 000012 A *G               12
31                                          ,WRITE  TTO,IOPS,BUFFER,34     /OUTPUT TO TELETYPE
          00066 R 002005 A *G               CAL*IOPS*1000 TTO&777
          00067 R 000011 A *G               11
          00070 R 000237 R *G               BUFFER
                          *G                ,DEC
          00071 R 777736 A *G               -34
32                                          ,WAIT   TTO                    /WAIT UNTIL OUTPUT COMPLETE
          00072 R 000005 A *G               CAL TTO&777
          00073 R 000012 A *G               12
33        00074 R                   RESTRT  ,CLOSE  TTO                    /TERMINATE TELETYPE OUTPUT,
          00074 R 000005 A *G               CAL TTO&777
          00075 R 000006 A *G               6
34                                          ,CLOSE  TTI                    /TELETYPE INPUT,
          00076 R 000006 A *G               CAL TTI&777
          00077 R 000006 A *G               6
35                                          ,CLOSE  DISK                   /AND DISK INPUT/OUTPUT
          00100 R 000007 A *G               CAL DISK&777
          00101 R 000006 A *G               6
36        00102 R 600000 R                  JMP     BEGIN                  /LOOP FOR UPDATE OPTION
                                            ,EJECT
```

```
PAGE    4     DKECHO 001     DKECHO

   37         00103 R                  UPDATE  .WRITE  TTO,IOPS,MSG1,34        /OUTPUT MSG1
              00103 R 002005 A *G              CAL+IOPS*1000 TTO&777
              00104 R 000011 A *G              11
              00105 R 000163 R *G              MSG1
                              *G               ,DEC
              00106 R 777736 A *G              -34
   38                                          .WAIT   TTO                     /AND MSG2
              00107 R 000005 A *G              CAL TTO&777
              00110 R 000012 A *G              12
   40                                          .WRITE  TTO,IOPS,MSG2,34        /ON
              00111 R 002005 A *G              CAL+IOPS*1000 TTO&777
              00112 R 000011 A *G              11
              00113 R 000177 R *G              MSG2
                              *G               ,DEC
              00114 R 777736 A *G              -34
   41                                          .WAIT   TTO                     /TELETYPE
              00115 R 000005 A *G              CAL TTO&777
              00116 R 000012 A *G              12
   42                                          .READ   TTI,IOPS,COM,8          /READ RESPONSE
              00117 R 002006 A *G              CAL+IOPS*1000 TTI&777
              00120 R 000010 A *G              10
              00121 R 000227 R *G              COM
                              *G               ,DEC
              00122 R 777770 A *G              -8
   43                                          .WAIT   TTI                     /WAIT UNTIL READ COMPLETE
              00123 R 000006 A *G              CAL TTI&777
              00124 R 000012 A *G              12
   44         00125 R 200231 R                 LAC     COM+2                   /GET FIRST WORD
   45         00126 R 500305 R                 AND     (774000                 /SAVE FIRST SEVEN BITS
   46         00127 R 540306 R                 SAD     (544000                 /IS CHAR A Y?
   47         00130 R 600133 R                 JMP     YES
   48         00131 R 140304 R                 DZM     UDSW                    /NO, SET TO REPLACE INPUT FILE
   49         00132 R 600021 R                 JMP     READKB                  /LOOP TO READ KEYBOARD
   50         00133 R 750001 A     YES         CLC                             /SET UPDATE SW, TO SAVE
                                               .EJECT
```

Programming Example - Assembly Listing (Cont.)

```
PAGE    5     DKECHO 001     DKECHO

   51          00134 R 040304 R              DAC     UDSW            /INPUT, CREATE NEW OUTPUT
   52          00135 R 600021 R              JMP     READKB          /LOOP TO READ KEYBOARD
   53          00136 R 200303 R     NEWFIL   LAC     NAME+2          /CHANGE EXT
   54          00137 R 640503 A              LRS     3               /LEAST SIGNIFICANT DIGIT OF
   55                                                                /SIXBT VALUE OF LAST CHAR IN EXT
   56          00140 R 742020 A              RTR                     /STRIP OFF HIGH PART OF CODE
   57          00141 R 740020 A              RAR
   58          00142 R 640503 A              LRS     3               /LEAST SIGNIFICANT DIGIT OF
   59                                                                /SIXBT VALUE OF MIDDLE CHAR IN EXT
   60          00143 R 742020 A              RTR                     /STRIP OFF HIGH PART OF SIXBT CODE
   61          00144 R 740020 A              RAR
   62          00145 R 640606 A              LLS     6               /PUT BACK IN AC
   63          00146 R 500307 R              AND     (777            /STRIP OF HIGH ORDER PART OF REMAIN
   64                                                                /SIXBT CODE FOR LAST EXT CHAR
   65          00147 R 340310 R              TAD     (1              /INCREMENT TO MAKE NEW EXT
   66          00150 R 640506 A              LRS     6               /REVERSE PROCESS TO FIX UP EXT IN
   67                                                                /PROPER SIXBT
   68          00151 R 742010 A              RTL
   69          00152 R 740010 A              RAL
   70          00153 R 640603 A              LLS     3
   71          00154 R 742010 A              RTL
   72          00155 R 740010 A              RAL
   73          00156 R 640603 A              LLS     3
   74          00157 R 500311 R              AND     (070707
   75          00160 R 240312 R              XOR     (606060
   76          00161 R 040303 R              DAC     NAME+2
   77          00162 R 600036 R              JMP     WRITE           /TO CREATE NEW OUTPUT
   78          00163 R 006000 A     MSG1     MSG2-MSG1/2*1000        /WPC FOR HEADER WORD 0
   79          00164 R 000000 A              0
   80          00165 R 432231 A              .ASCII "FILE ALREADY "
               00166 R 442500 A
               00167 R 406312 A
               00170 R 242602 A
               00171 R 422624 A
               00172 R 000000 A
   81          00173 R 502450 A              .ASCII "PRESENT!!"<15>
               00174 R 551612 A
               00175 R 472504 A
               00176 R 120432 A
```

Programming Example - Assembly Listing (Cont.)

```
PAGE   6     DKECHO 001     DKECHO

  83        00177 R 012000 A     MSG2     MSG3-MSG2/2*1000                /WPC FOR HEADER WORD 0
  84        00200 R 000000 A              0
  85        00201 R 422364 A              ,ASCII "DO YOU WISH TO KEEP IT ?"
            00202 R 054636 A
            00203 R 525012 A
            00204 R 744646 A
            00205 R 441012 A
            00206 R 447500 A
            00207 R 456130 A
            00210 R 550100 A
            00211 R 446504 A
            00212 R 037400 A
  86        00213 R 242624 A              ,ASCII "(Y OR N AND CR)  >"<175>
            00214 R 047644 A
            00215 R 202344 A
            00216 R 040634 A
            00217 R 421010 A
            00220 R 351122 A
            00221 R 201007 A
            00222 R 676400 A
  87        00223 R 002000 A     MSG3     COM-MSG3/2*1000
  88        00224 R 000000 A              0
  89        00225 R 373720 A              ,ASCII ">"<175>
            00226 R 000000 A
  90        00227 R        A     COM      ,BLOCK  10
  91        00237 R        A     BUFFER   ,BLOCK  42
  92        00301 R 050310 A     NAME     ,SIXBT  "ECHO@@001"
            00302 R 170000 A
            00303 R 606061 A
  93        00304 R 000000 A     UDSW     0
  94                000000 R              ,END    BEGIN
            00305 R 774000 A *L
            00306 R 544000 A *L
            00307 R 000777 A *L
            00310 R 000001 A *L
            00311 R 070707 A *L
            00312 R 606060 A *L
                SIZE=00313     NO ERROR LINES
```

Program Example - Assembly Listing (Cont.)

PAGE 7 DKECHO CROSS REFERENCE

BEGIN	00000	9*	36	94						
BUFFER	00237	16	23	28	31	91*				
COM	00227	42	44	87	90*					
DISK	000007	2*	9	12	22	23	24	25	26	27
		28	29	35						
IN	000000	5*	10	26						
IOPS	000002	7*	15	16	23	28	31	37	40	42
MSG1	00163	37	78*	78						
MSG2	00177	40	78	83*	83					
MSG3	00223	15	83	87*	87					
NAME	00301	12	22	27	53	76	92*			
NEWFIL	00136	21	53*							
OUT	000001	6*	9	11						
READDT	00051	26*								
READKB	00021	15*	49	52						
RESTRT	00074	9	10	11	26	33*				
START	00014	12*								
TTI	000006	3*	10	16	17	34	42	43		
TTO	000005	4*	11	15	31	32	33	37	38	40
		41								
UDSW	00304	19	48	51	93*					
UPDATE	00103	14	37*							
WRITE	00036	22*	77							
YES	00133	47	50*							

6.8.6 File Integrity Considerations

In a system such as DOS, which offers a flexible I/O command repertoire, I/O programming requires care...otherwise, numerous IOPS errors may result. Further, there are I/O sequences which the system allows... which, if carelessly used, can result in the destruction of the user's files or those of others. The paragraphs below describe some important considerations.

a. Extreme care must be exercised when using the .TRAN macro to output to the DECdisk, Disk Cartridge, Disk Pack, DECtape or Magtape. The user must know the disk and DECtape file structures completely, because .TRAN operates completely outside these file structures, and ignores the existence of all directories and bit maps. The entire contents of the disk or DECtape or Magtape, therefore, are vulnerable to the user of .TRAN.

b. Caution should be used when reading a file from the disk sequentially (.READ) from one .DAT slot, while modifying the same file via another .DAT slot using random access (.RTRAN).

c. Generally speaking, output files are not recognized by the system until they are .CLOSED. Under most circumstances, termination of program control and return to the Monitor will cause the Monitor to delete any unclosed output files. Occasionally, a system crash or other unusual phenomenon will cause a disk output file to be truncated. Truncated files are the remains of output files that the system did not get a chance to delete. Directory listings from PIP that contain an asterisk (*) after a file name indicate a truncated file. They take up disk space and should be deleted via commands to PIP.

CHAPTER 7
SYSTEM INITIALIZATION

7.1 INTRODUCTION

This Chapter describes the procedures to be followed when loading, starting, and tailoring the XVM/DOS Software System. Under normal circumstances, these procedures should rarely need to be used. Occasionally, however, a program may enter a runaway condition which could result in the inadvertent destruction of a part of the XVM/DOS software residing on the system device or in core. In addition, it may be necessary to change the XVM/DOS software configuration from time to time to permit the use of a new I/O device or system program. These alterations to the system may be performed only by those who have access to the system's Monitor Identification Code (MIC). Thus the average user need not be concerned with the contents of this chapter (except when loading using the XVM/DOS Bootstrap), (Paragraph 7.3.2).

Chapter 10 contains the operating procedures to be used once the system is loaded and running.

7.2 HOW THE SYSTEM SOFTWARE IS SUPPLIED

The XVM/DOS Monitor and System programs are supplied to users on either DECtape or Magtape, depending upon the particular hardware configuration. The bulk of the software system resides on the DECtape or Magtape medium in a special form which is meaningful only to the DOSSAV program, which can transfer the system to the disk. Thus, these tapes are often called "disk restore tapes" as they may be used only for this purpose. The XVM/DOS software kits and instructions for building the initial system are given in the XVM/DOS V1A System Installation Guide.

7.3 SYSTEM STARTUP PROCEDURES

The following paragraphs describe the procedure for loading and starting the XVM/DOS Software System using the DOSSAV and XVM/DOS Bootstrap programs.

7.3.1 Loading PIREX

This section applies only to UNICHANNEL Systems. XVM users without the UC15 hardware option should proceed to Section 7.3.2. UNICHANNEL systems must load PIREX before any of the PDP-11 peripherals may be used.

It is assumed that the system cartridge is running on unit 0. Halt the PDP-11 by pressing the ENABLE/HALT console switch down. Stop the XVM by pressing the STOP toggle. On both machines, the run light should be off. Set the XVM address switches to 17700 (octal). (White line shows on bottom of 'on' bit.) Place the ABSL11 paper tape in the XVM paper tape reader. Press both the STOP and RESET switches simultaneously and then the READIN switch on the XVM console. The paper tape should read in, and the XVM should halt (RUN light off).

Place 140000 (switches number 14 and 15 are up) in the address switches of the PDP-11. (Note that a bit is 'ON' when the corresponding switch is up.) Press down the LOAD-ADDRESS switch. Move the ENABLE-HALT switch to ENABLE (up). Press down the START switch. At this point the PDP-11 should start (RUN light should come on).

Place the PIREX paper tape in the XVM paper tape reader and then press the CONTINUE toggle on the XVM. The PIREX tape should read in, and the XVM should halt. The PIREX monitor in the PDP-11 should start (RUN light on, and bit 0 of the address lights on). This is the characteristic pattern of the PIREX null job.

7.3.2 Disk Restoration (DOSSAV)

The XVM/DOS Software System is transferred from the Disk Restore DECtapes or Magtapes supplied by Digital Equipment Corporation to the appropriate disk device using the DOSSAV utility program. This program provides users with the ability to save and subsequently restore all occupied blocks[1] on the disk using either DECtape, Magtape, DECdisk, Disk Cartridge, or Disk Packs as the storage medium. Not only is DOSSAV used to install the XVM/DOS System, it can also make additional copies of the system currently on the disk, of a newly tailored system (i.e., after System Generation), or of the contents of Disk Pack or Disk Cartridge unit 1-7 (in the case of multiple Disk Pack, Disk Cartridge systems).

DOSSAV should be used periodically by the system manager to create a "backup" tape of the current system plus user files. These system backups should be performed weekly (approximately, depending on system activity and the ease of replaceability of user files) and the most recent 3 or 4 backup tapes should be saved. If this procedure is followed, a minimum of time and effort would be lost in re-building the system in the event of serious disk file corruption or accidental deletion of files. Note also that DOSSAV may be used to quickly save all files of a "data disk", which contains only data files (no system files).

DOSSAV operations interactively via the console teleprinter and asks the user a series of questions to determine which devices and unit numbers are to be used and, for Magtape, density and track count in- When all necessary information has been obtained, DOSSAV automatically begins the specified operation. If a save or restore operation requires more than one tape, the program stops and outputs a message on the teleprinter to that effect. The user can then mount the next tape and continue. The program performs error checking to detect both hardware and command string errors and output appropriate messages on the teleprinter. RUBOUT and CTRL/U are permitted when answering questions.

In addition, the program permits DECdisk users to restore systems created in a small DECdisk configuration to an environment with a larger DECdisk configuration[2]. The reverse situation, however, is not possible. DOSSAV is a stand-alone program[3] supplied in paper tape.

---

[1] When a SPOOLER area is present, it is not saved by DOSSAV.

[2] This type of operation should only be done with master tapes (or copies of them) since block 1775_8 must not be occupied when performing a restoration to a system with 5 or more platters.

[3] For non UC15 systems only. PIREX has to be running for UC15, RK based systems; refer to Section 7.3.1.

form and is loaded via the high speed paper tape reader using the XVM's Hardware Readin Mode (load address 5772Ø[1], restart address 54ØØØ).

7.3.2.1 Operating Procedures - The following procedures should be used when restoring or saving the XVM/DOS System Software (or other user-created data on Disk Pack or Disk Cartridge units 1-7):

For UC15, RK based systems[2] refer to Section 7.3.1 for the PIREX start up procedures.

a. Place the DOSSAV paper tape in the Paper Tape Reader.
b. Set the console ADDRESS Switches to 57720[1].
c. To restore the disk:[3]
    1. Mount the XVM/DOS Disk Restore DECtape or Magtape for the appropriate hardware configuration (i.e., DECdisk, Disk Pack or Disk Cartridge) on the applicable tape drive.
    2. Set the DECtape drive WRITE ENABLE/LOCK switches to LOCK. (Magtape users should remove the Write Enable ring from the tape reel.)
    3. Set the disk READ/WRITE PROTECT switches to ENABLE.
d. To save the contents of the disk:
    1. Mount a fresh tape on the appropriate tape drive.
    2. Set the DECtape WRITE ENABLE/LOCK switch to ENABLE. (Magtape users should install the Write Enable ring on the tape reel.)
e. Set the tape drive unit number switches as desired.
f. Set all ON LINE/OFF LINE switches of the devices being used to ON LINE.
g. Press the XVM Console Switches STOP and RESET simultaneously, then press the READIN switch. (The DOSSAV tape should pass through the reader.)

---

[1] DOSSAV may be loaded at any of the following octal locations: 17720 37720, 57720, or 77720. A binary loader precedes DOSSAV on the paper tape and this loader moves DOSSAV to bank 2 where DOSSAV actually runs the restart address must be 54000_8.

[2] If PIREX is not running.

[3] Spooling must be disabled during any DOSSAV operations.

7.3.2.2 Commands - Once the paper tape has been read in, DOSSAV will identify itself on the teleprinter and begin to ask the user a series of questions about the devices to be used, as shown in Table 7-1. Each user response must be terminated by a Carriage RETURN. It should also be noted that some of the questions shown are typed out only when applicable (i.e., no questions are asked about Disk Pack unit numbers or Magtape density, or number of channels if these devices are not to be used). Legal input and output device combinations are shown in Table 7-2.

Table 7-1
DOSSAV Commands

Query	User Responses
INPUT DEVICE?	DT = DECtape, MT[4] = Magtape, DK = DECdisk, DP = Disk Pack, RK = Disk Cartridge
UNIT NO?	Legal unit numbers are 0 - 7
SAVE Q AREA[3] (Y OR N)	Y = SAVE THE CTRL/Q AREA N = DO NOT SAVE THE CTRL/Q AREA
TRACK[5] (7 OR 9)?	Magtape track or channel number. (If 9 is specified, density is assumed to be 800 BPI.)
DENSITY[1,2] (2,5,8)?	Magtape recording density: 2=200 Bits Per Inch (BPI), 5=556 BPI, 8=800 BPI.
OUTPUT DEVICE?	(See INPUT DEVICE above.)
UNIT NO?	(See UNIT NO. above.)
TRACK (7 OR 9)?	(See TRACK above.)
DENSITY (2,5,8)?	(See DENSITY above.)
DATE CREATED:	The date that the restore tape was created is typed out by DOSSAV.

[1] All XVM/DOS Magtapes distributed by DEC are 800 BPI, Odd Parity.

[2] This question is only asked when using 7-track magtape.

[3] Asked only when saving disks which have a SYSBLK and a ↑Q Area of non-zero length.

[4] Magtapes with even parity cannot be saved or restored with DOSSAV.

[5] This question is asked only when using magtape.

Table 7-2
Legal DOSSAV I/O Device Combinations

Input Device \ Output Device	DEC-disk (DK)	Disk Pack (DP)	DEC-tape (DT)	Mag-tape[1] (MT)	Disk Cartridge (RK)
DECtape (DT)	OK	OK	Illegal[2]	Illegal	OK
Magtape (MT)	OK	OK	Illegal	Illegal	OK
DECdisk (DK)	Illegal	OK	OK	OK	OK
Disk Pack (DP)	OK	Illegal[2]	OK	OK	Illegal
Disk Cartridge	Illegal	OK	OK	OK	Illegal[2]

Once the last question has been answered, DOSSAV proceeds with the specified operation. If additional tapes are required to complete the restore or save operation, the tape is rewound if the unit number is not equal to 0 and the following message is output:

```
TAPE DONE.  MOUNT ANOTHER
```

At this point, the user should either mount a fresh tape, if a save operation is being performed, or mount the next tape in the sequence established at tape creation, if a restore operation. Then type the new unit number on which the tape is mounted followed by a Carriage RETURN to proceed with the operation. When the requested operation is entirely complete, DOSSAV restarts and identifies itself as before:

```
DOSSAV XVM Vnxnnn    where: nx  is the version and
INPUT DEVICE?               nnn is the edit number
                                for this version.
```

At this point the current restore or save operation is complete. If the XVM/DOS Software was being restored, it is now ready to be started as specified in 7.3.3. If other DOSSAV operations are desired, the user should proceed again as specified in 7.3.2.1.

7.3.2.3 Examples of DOSSAV Commands - The following examples illustrate typical DOSSAV commands when restoring and saving the XVM/DOS Software System. User responses are underlined.

[1]All XVM/DOS Magtapes distributed by DEC are 800 bpi, Odd Parity.
[2]Use PIP Utility program with the (H) switch option for this operation.

Disk Restoration

1. Restore DECdisk system from DECtape units 1 and 2:

```
DOSSAV XVM Vnxnnn
INPUT DEVICE? DT↵
UNIT #? 1↵
OUTPUT DEVICE? DK↵
DATE CREATED:  09-DEC-75
TAPE DONE.  MOUNT ANOTHER
2↵
```

(The user mounted the next tape, on unit #2 then typed a 2↵ to continue.)

```
DOSSAV XVM Vnxnnn
INPUT DEVICE?
```

(Operation complete DOSSAV restarts.)

2. Restore DECdisk system from Magtape unit 0:

```
DOSSAV XVM Vnxnnn
INPUT DEVICE? MT↵
UNIT #? 0↵
TRACK(7 OR 9)? 7↵
DENSITY (2,5,8)? 8↵
OUTPUT DEVICE? DK↵
DATE CREATED:  09-DEC-75
```

(All XVM/DOS System Disk Restore Magtapes are 800 BPI, Odd Parity.)

```
DOSSAV XVM Vnxnnn
INPUT DEVICE?
```

(Operation complete)

3. Restore Disk Pack System from DECtape units 1 and 2:

```
DOSSAV XVM Vnxnnn
INPUT DEVICE? DT↵
UNIT #? 1↵
OUTPUT DEVICE? DP↵
UNIT #? 0↵
DATE CREATED:  09-DEC-75
TAPE DONE.  MOUNT ANOTHER
2↵
```

(The user mounted the next tape on unit number 2, then typed a 2↵ to continue.)

```
DOSSAV XVM Vnxnnn
INPUT DEVICE?
```

(Operation complete)

4. Restore Disk Pack system from Magtape unit 1:

```
DOSSAV XVM Vnxnnn
INPUT DEVICE? MT↵
UNIT #? 1↵
TRACK(7 OR 9)? 9↵
OUTPUT DEVICE? DP↵
UNIT #? 0↵
DATE CREATED:  09-DEC-75

DOSSAV XVM Vnxnnn
INPUT DEVICE?
```

(Operation complete)

5. Restore Disk Cartridge from Magtape unit 0:

```
DOSSAV XVM Vnxnnn
INPUT DEVICE? MT↵
UNIT #? 0↵
TRACK(7 OR 9)? 9↵
OUTPUT DEVICE? RK↵
UNIT #? 0↵
DATE CREATED:  04-DEC-75

DOSSAV XVM Vnxnnn                    (Operation Complete)
INPUT DEVICE?
```

Saving the Contents of the Disk

The single example below should suffice in illustrating this type of operation, since save operations are simply the reverse of restore operations.

Save a Disk pack system on DECtape units 2 and 3:

```
DOSSAV XVM Vnxnnn                    (DOSSAV allows for as many
INPUT DEVICE? DP↵                    DECtapes or Magtapes as are
UNIT #? 0↵                           necessary to contain the entire
OUTPUT DEVICE? DT↵                   contents of the specified disk.)
UNIT #? 2↵
TAPE DONE. MOUNT ANOTHER
3↵

DOSSAV XVM Vnxnnn                    (Operation complete)
INPUT DEVICE?
```

7.3.2.4 Error Messages and Meanings - DOSSAV attempts to detect all keyboard and run-time errors and to recover if possible. The three types of errors which can occur are shown below along with their meanings and recovery procedures when applicable.

a. Command String Errors - These errors occur when a question is answered incorrectly. DOSSAV repeats the question.

Message	Meaning
ILLEGAL DEVICE	Either an illegal device mnemonic (one other than DT, MT, DK, RK or DP) or an illegal combination of devices (DT for input and MT for output) was typed.
BAD TRACK	A track number other than 7 or 9 was typed.
BAD DENSITY	A density other than 2(200 BPI), 5(556 BPI) or 8(800 BPI) was typed.

b. Recoverable Operating Errors - These errors occur when one of the I/O devices is not properly set up. When the condition has been corrected, operation can be resumed by typing the unit number of the I/O device that has been set up correctly, followed by a carriage RETURN for the first three messages. For the xx ERROR IGN message activate the continue switch on the console to continue DOSSAV; otherwise, restart DOSSAV.

Message	Meaning
TAPE NOT READY	The DECtape or Magtape unit is not switched to ON LINE, is not set to WRITE ENABLE, or is not set to the unit number specified in the UNIT NO? question or two or more tapes are on-line and dialed to the same unit number.
DISK NOT READY	The DECdisk is not set to WRITE ENABLE.
DISK PACK NOT READY	The Disk Pack unit is not switched to ON LINE, is not set to WRITE ENABLE or is not set to the unit number specified in the UNIT #? question.
xx ERROR IGN	A parity/checksum error exists in the block currently being transferred; the block number is in the AC, the XVM is faulted. This block will be ignored (not saved or restored) and the operation continued by pressing the "CONT" switch on the XVM console.

xx = RK/DK/DP

c. Unrecoverable Errors - Errors associated with these messages are primarily hardware errors from which DOSSAV cannot recover. After the message is typed, DOSSAV restarts itself.

Message	Meaning
DECTAPE ERROR MAGTAPE ERROR DECDISK ERROR DISK PACK ERROR DISK ERROR	Hardware error detected
ATTEMPT TO RESTORE SYSTEM TO WRONG DISK	The user tried to restore a DECdisk system using a Disk Pack restore tape, for example.
BLK 1775 OCCUPIED. NO 2ND SAT CREATED.	The user did not use a master restore tape (i.e., block 1775 is occupied) when restoring a system created for 4 or fewer platters to a system having 5 or more platters.

7.3.2.5 Restart Procedures - The restart procedures below should be used to terminate the current operation prematurely or to reinitialize DOSSAV if it fails to start up automatically after an error.

a. Press the XVM Console Switches STOP and RESET simultaneously.
b. Set the ADDRESS switches to 54ØØØ.
c. Press the START Console Switch.

DOSSAV should then identify itself as when originally loaded, otherwise it must be reloaded as described in Paragraph 7.3.2.1.

7.3.3 Loading and Starting the Monitor

The DOS Monitor is loaded into core from either the DECdisk or Disk Pack or Disk Cartridge[1] and is automatically started with the XVM/DOS Bootstrap Loader program. Once loaded, the bootstrap remains in the upper 141_8 locations of the core bank into which it was loaded, during all normal system operation. The bootstrap not only aids in initializing the Monitor, but also acts as an integral part of the Monitor as it operates. It is supplied on paper tape in three versions. RFBOOT is for use with a DECdisk system, RKBOOT is for use with the Disk Cartridge systems and RPBOOT is for use with Disk Pack Systems.

Each time that the XVM/DOS System Software is restored using DOSSAV, the bootstrap must be loaded into core. Occasionally, a runaway program or hardware malfunction may cause the destruction of the Monitor as it resides in core, and prevent the user from restarting it by keyboard command. In this circumstance, it is often possible to restart the bootstrap (unless it, too, has been destroyed) and avoid reloading the bootstrap. The paragraphs which follow describe the initial loading and restart procedures for the XVM/DOS Bootstrap Loader.

7.3.3.1 Loading the Bootstrap

a. Select the appropriate version of the bootstrap (either DECdisk, Disk Cartridge or Disk Pack) and place it in the paper tape reader.
b. Set the XVM console ADDRESS switches to one of the two addresses shown below in accordance with the maximum core size of the system.

[1] Refer to Section 7.3.1 for PIREX start up procedures.

Address	Maximum Core Size
57637	24K
77637	32K or more

c. Press the console switches STOP and RESET; then press READIN.

Once the bootstrap is loaded, it starts automatically and loads the Monitor from the disk[1]. When loading is complete, the Monitor gets control and identifies itself on the console teleprinter as follows:

```
XVM/DOS Vnxnnn
ENTER DATE (MM/DD/YY) -
```

where: nx is the version and
nnn is the edit number for this version.

The system is now loaded and operable and is ready to accept the keyboard commands (Chapter 8) in accordance with the operating procedures described in Chapter 10. Users with new systems, that is, systems which have not previously been tailored, should refer to Paragraph 7.4.

7.3.3.2 Bootstrap Restart Procedures - As mentioned above, situations occasionally arise in which the Monitor must be reloaded by the bootstrap. If the bootstrap is intact in core, it may be restarted by the procedures which follow.

a. Set the console ADDRESS switches to one of the addresses shown below in accordance with the maximum core size of the system.

Address	Maximum Core Size
57646	24K
77646	32K or more

b. Press the console switches STOP and RESET; then press START.

c. The Monitor should then identify itself as shown in 7.3.2.1.

d. If the Monitor does not identify itself, the bootstrap has been destroyed and must be reloaded as described in 7.3.2.1.

[1]For UC15, RK based systems, this occurs only if PIREX is running.

7.4 SYSTEM MODIFICATION (TAILORING) PROCEDURES

The software package supplied to each user is a general purpose version of the XVM/DOS Software System. This means that the system contains all of the standard XVM/DOS language programs, utility programs, library routines, and I/O device handling routines supported by Digital Equipment Corporation. Included as a part of this general package is a utility program called the System Generator (or SGEN). SGEN enables the user to tailor the XVM/DOS software to suit the particular hardware configuration and operating requirements of his installation.

Specifically, SGEN provides the user with the ability to: (1) delete system programs or add his own[1]; (2) add and delete I/O device handlers; (3) alter system load-time parameters such as: I/O device assignments, teleprinter model currently being used, number of buffers to be allocated, file and directory protection codes, etc.; (4) change system operating parameters including: the Priority Interrupt Skip Chain, number of positive .DAT slots, the presence of 7- or 9-channel Magtape drives, the Monitor Identification Code (MIC). The amount of disk cartridge utilized for spooling on UNICHANNEL systems can be altered by SPLGEN.[2]

---

[1] Described in the XVM/DOS System Manual and the SGEN XVM Utility Manual.

[2] Described in the XVM/DOS Keyboard Command Guide.

CHAPTER 8
KEYBOARD COMMANDS

8.1 INTRODUCTION

This chapter describes the commands which can be issued from console keyboard to direct the operations of the XVM/DOS system software. In communicating with the Monitor, the keyboard of the console teleprinter is used as the system's control device. The operator at the keyboard types commands to allocate system resources, load and start system and user-created programs, terminate program operation, and exchange information with the Monitor. Most of the Monitor's keyboard commands are issued prior to loading system or user programs and are interpreted by the Nonresident Monitor.

During program execution, a small set of keyboard commands is available for general program control. These commands are interpreted by the teleprinter's I/O device handler (which is part of the Resident Monitor), and are used to control program start and restart, dumping of core, and the reloading of the Nonresident Monitor. Paragraph 8.11 describes commands used during program execution.

The console teleprinter is the communications interface between the user-operator and the Monitor. The interaction between the operator and the Nonresident Monitor is completely conversational. Each command issued causes the Monitor to type out an appropriate reply. Monitor responses may vary from a single character to several pages of information.

In the context of this manual, the term "console keyboard" designates any one of several keyboard/printer/display I/O devices which could be used by the Monitor as the system command console device. (That is, it is associated with .DAT slots -2 and -3.)

The keyboard commands are, however, not strictly limited to input from the keyboard. The Monitor can be operated in a Command Batching Mode (see 8.12) in which keyboard commands can be issued from punched cards, paper tape or from a mass-storage device with minimum operator intervention. Similarly, the Monitor's responses to commands are not strictly limited to a keyboard device's printer or display, but may also be output to other devices including the VT15 display or a line printer, when available.

8.2 KEYBOARD COMMAND FORMATS AND CHARACTERS

8.2.1 Keyboard Command Elements

All keyboard commands, except those using the keyboard CTRL key, consist of at least two elements, a command name and a terminator. Some commands require an additional third element consisting of one or more arguments inserted between the command name and the terminator. Each command name is separated from its argument (or argument string) by one or more spaces. Delimiters between multiple arguments vary, and are specified in the descriptions for the individual commands. Except as otherwise specified, each command string can be terminated by either a Carriage RETURN or an ALT MODE. CTRL commands are formed by simultaneously depressing the keyboard's CTRL key and letter key, and are interpreted by the Resident Monitor. These commands need no terminators, such as Carriage RETURN or ALT MODE. They are interpreted immediately after they have been typed. Once typed, the command is echoed in the form of an up-arrow (↑) followed by the letter which identifies the command. Thus, CTRL C is echoed ↑C.

The symbols defined in Section 5.2.2 are used in illustrating the command formats described in the succeeding paragraphs of this chapter. XVM/DOS system programs accept ASCII characters shown in Appendix A.

8.2.2 Editing Features

The teleprinter's device handler provides two keyboard editing functions which can be used to change the line currently being typed (prior to typing Carriage RETURN or ALT MODE).

RUBOUT — The RUBOUT key permits successive deletion of characters, starting with the last character typed. Each RUBOUT deletes one character and causes a backslash (\) to be echoed in response. RUBOUT does not delete characters past the previous line terminator. Once all characters in a line have been deleted, additional RUBOUTS are ignored. For example, if the command INSTRUCT were mistyped as INSTRUTC, it could be corrected by typing two RUBOUTs followed by CT, as shown below:

Example:

```
$INSTRUTC\\CT
```

CTRL U — Formed by depressing the CTRL key and striking the U key, this command during input eliminates all characters typed up to the last Carriage RETURN or ALT MODE and echoes a @. Thus an irretrievably bollixed input line may be eliminated (before typing Carriage RETURN or ALT MODE) by typing CTRL U. This feature can also be used during output to abort the current line.

8.2.3 When to Issue Keyboard Commands

All keyboard commands, except for the CTRL commands (see 8.11) are only recognized and accepted when the Nonresident Monitor is in core, as evidenced by the appearance of either of the following identifiers on the teleprinter:

```
XVM/DOS Vnxnnn      where:   nx is the version and
                             nnn is the edit number
$                            of this version.

    or

$
```

The operator can obtain the Nonresident Monitor by typing CTRL C, (described in 8.11.1). When the Monitor is initialized using the Bootstrap, or by typing CTRL C, the full identification is typed. At all other times it indicates its readiness to accept keyboard commands by simply typing the dollar sign ($). Once the $ is typed, the Monitor idles until the operator types a command.

8.3 COMMANDS TO REQUEST SYSTEM INFORMATION

The commands in this paragraph provide the user with various lists of system information. The lists include: (1) general system information, (2) keyboard commands, (3) error messages, and (4) .DAT slot assignments. Since these lists tend to be lengthy, users who have either a line printer or a VT15 Display can speed up the output of this information by using the special line printer and VT15 commands (described in paragraphs 8.9 and 8.8, respectively).

8.3.1 SCOM

The SCOM command causes the typeout of XVM/DOS system information. The information includes (1) available I/O device handlers, (2) system default parameters, (3) important core addresses, and (4) the Priority Interrupt Skip Chain order.

Form: S[COM]↵

In this and subsequent descriptions, characters enclosed in brackets are optional.

```
$API ON

BANK MODE   64K  API ON  XVM OFF  UC15 ON  POLLER OFF

$XVM ON

BANK MODE   64K  API ON  XVM ON  UC15 ON  POLLER OFF

$SCOM

SYSTEM INFO - XVM/DOS - 06/25/75

77646 - BOOTSTRAP RESTART ADDR
77636 - 1ST FREE CELL BELOW BOOTSTRAP
4157- ADDR OF .DAT
4215- ADDR OF .UFD
20 - NO. OF POS. .DAT SLOTS
API IS ENABLED
XVM MODE IS ENABLED
BANK MODE OPERATION
7 CHANNEL MAGTAPE ASSUMED BY HANDLERS
 80-CHARACTER LINE PRINTER ASSUMED BY HANDLERS
2 - DEFAULT FILE PROTECTION CODE
4 - CURRENT FILE PROTECTION CODE
03 - DEFAULT BUFFS SETTING
1432- ^Q ADDRESS FOR MANUAL DUMP
I/O HANDLERS AVAILABLE
TTA   TELETYPE: I/O, ASCII MODES, ALL FUNCTIONS
PRA   TAPE READER: INPUT, ALL MODES, ALL FUNCTIONS
PRB   TAPE READER: INPUT, IOPS ASCII MODE, ALL FUNCTIONS
PPA   PUNCH: OUTPUT, ALL MODES, ALL FUNCTIONS
PPB   PUNCH: OUTPUT,ALL MODES LESS IOPS ASCII, ALL FUNCTIONS
PPC   PUNCH: OUTPUT, IOPS BINARY MODE, ALL FUNCTIONS
DTA   DECTAPE: 3 FILES, I/O, ALL MODES, ALL FUNCTIONS
DTC   DECTAPE: 1 FILE, INPUT, IOPS MODES, LIM FUNCTIONS
DTD   DECTAPE: 1 FILE, I/O, ALL MODES, ALL FUNCTIONS
DTE   DECTAPE: 1 FILE, I/O, ALL MODES, NO .MTAPE
DTF   DECTAPE: NON-FILE ORIENTED FOR F4 .OTS
DKA   DECDISK: N FILES, I/O, ALL MODES, ALL FUNCTIONS
DKB   DECDISK: N FILES, I/O, ALL MODES, LIM FUNCTIONS
DKC   DECDISK: N FILES, INPUT, ALL MODES, LIM FUNCTIONS
DPA   DISKPACK: N FILES, I/O, ALL MODES, ALL FUNCTIONS
DPB   DISKPACK: N FILES, I/O, ALL MODES, LIM FUNCTIONS
DPC   DISKPACK: N FILES, INPUT, ALL MODES, LIM FUNCTIONS
RKA   DISKCART: N FILES, I/O, ALL MODES, ALL FUNCTIONS
RKB   DISKCART: N FILES, I/O, ALL MODES, LIM FUNCTIONS
RKC   DISKCART: N FILES, INPUT, ALL MODES, LIM FUNCTIONS
MTA   MAGTAPE: 3 FILES, I/O, ALL MODES, ALL FUNCTIONS
MTC   MAGTAPE: 1 FILE, INPUT, IOPS MODES, ALL FUNCTIONS
MTF   MAGTAPE: NON-FILE ORIENTED FOR F4 .OTS
LPA   LINE PRINTER: OUTPUT, ASCII MODES, ALL FUNCTIONS
CDB   CARD READER: INPUT, IOPS ASCII MODE, ALL FUNCTIONS
VPA   VP DISPLAY: OUTPUT, ASCII AND DUMP MODES, ALL FUNCTIONS
VTA   VT-15: I/O
XYA    PLOTTER: OUTPUT, ASCII & BINARY MODES, ALL FUNCTIONS
LKA   LK-35 KEYBOARD: INPUT, ASCII MODES, ALL FUNCTIONS
SKIP CHAIN ORDER
```

```
SPFAL
DTDF
DSSF
RKSF
DPSJ
MTSF
SPDI
WTSK
SDDF
CRSI
CRSD
LPSF
CLSF
RSF
PSF
KSF
KSF1
TSF
DTEF
DPSE
MPSNE
MPSK
SPE
CRSF
LSSF
XYSF
KSF2
KSF3
KSF4
KSF5

$^C

$
```

8.3.2 INSTRUCT

The INSTRUCT command causes a typeout of either the keyboard commands or system errors, depending upon which form of the command is used. This command utilizes .DAT -12 as the output device.

Form 1: I[NSTRUCT]

```
XVM/DOS Vnxnnn

$I

DOS-15 COMMANDS
        LOG(L): USER COMMENTS TERMINATED BY ALTMODE
        LOGW: USER COMMENTS TERMINATED BY ALTMODE, WAIT FOR ^P TO CONTINUE
        DATE(D): ECHO DATE
        DATE(D) MM/DD/YY: ENTER DATE
        TIME(T): ECHO TIME
        TIME(T) HHMM: ENTER TIME
        PROTECT(P) N: CHANGE DEFAULT PROTECTION CODE TO N
        KEEP(K) ON/OFF: KEEP .DAT SLOTS UNALTERED ON .EXIT
        LOGIN UIC: DEFINE NEW CURRENT UIC
        LOGOUT: SIGN OFF UIC
        DOS15: ENTER DOS15 BATCH MODE
        SCOM(S): SYSTEM INFO
        INSTRUCT(I): LIST OF MONITOR COMMANDS
        INSTRUCT(I) ERRORS: DESCRIPTION OF ERROR CODES
        REQUEST(R), REQUEST(R) PRGNAM: .DAT SLOT USAGE
        REQUEST(R) USER: POSITIVE .DAT SLOT USAGE
        MODE: DISPLAY CURRENT MACHINE OPERATING ENVIRONMENT
        ASSIGN(A) DEVN <UIC> A,B,.../ETC.: .DAT SLOT MODS
        QDUMP(Q): SET TO SAVE CORE (^Q) ON .IOPS ERROR
        HALT(H): SET TO HALT ON .IOPS ERROR
        ^Q: SAVE CORE ON UNIT Q
        GET(G): RESTORE CORE FROM ^Q AREA
        GETP:CORE FROM ^Q AREA AND RESTART WITH ^P
        GETT: RESTORE CORE FROM ^Q AREA AND RESTART WITH ^T
        GETS : RESTORE CORE FROM ^Q AREA AND RESTART WITH ^S
        GET(G) N FILE : RESTORE CORE FROM FILE ON UNIT N AND RESTART
        GETP N FILE : RESTORE CORE FROM FILE ON UNIT N AND RESTART WITH ^P
        GETT N FILE : RESTORE CORE FROM FILE ON UNIT N AND RESTART WITH ^T
        GETS N FILE : RESTORE CORE FROM FILE ON UNIT N AND RESTART WITH ^S
        PUT N FILENM : PUT ^Q AREA INTO FILENM ON UNIT N
        API ON/OFF: CHANGE STATE OF API
        VT ON/OFF: TURN GRAPHIC DISPLAY ON/OFF
        HALF ON/OFF: TURN HALF BUFFER MODE FOR GRAPHIC DISPLAY ON
        TAB ON/OFF: PERFORM SOFTWARE EMULATION OF TAB KEY.
        FILL ON/OFF: PROVIDE FILL CHARACTERS AFTER CR/LF.
        LP ON/OFF: TURN ON/OFF LINE PRINTER FOR OUTPUT OF SOME NRM COMMANDS
        XVM ON/OFF: RUN USER PROGRAMS WITH XM-15 OPTION.
        BANK ON/OFF: CHANGE STATUS OF BANK MODE
        PAGE ON/OFF: CHANGE STATUS OF PAGE MODE
        BUFFS N: CHANGE DEFAULT BUFFER ALLOCATION
        CHANNEL 7/9: SETUP DEFAULT ASSUMPTION FOR MAGTAPE
        ^C: RESTORE DOS-15      ^P: USER RESTART        ^T: RESTART DDT
        ^X: TURN VT ON OR OFF
```

```
DOS-15 PROG LOADING COMMANDS AND PROGNAM FOR REQUEST COMMAND
        LOAD: LINK LOAD AND WAIT FOR ^S GLOAD:'
        GLOAD: LINK LOAD AND GO
        DDT: LINK LOAD WITH SYMBOLS AND GO TO DDT
        DDTNS: LINK LOAD W/O SYMBOLS AND GO TO DDT
        MACRO: MACRO ASSEMBLER
        F4: FORTRAN IV COMPILER
        EDIT: TEXT EDITOR
        PIP: PERIPHERAL INTERCHANGE PROG
        DUMP: BULK STOR DEV DUMP
        UPDATE: LIBR FILE UPDATE
        SRCCOM: SOURCE COMPARE
        EDITVP: STORAGE SCOPE EDITOR
        EDITVT: GRAPHIC DISPLAY (VT) EDITOR
        PATCH: SYSTEM TAPE PATCH ROUTINE
        EXECUTE(E) FILE: LOAD AND RUN FILE XCT
        CHAIN: XCT CHAIN BUILDER
        8TRAN: PDP-8 TO PDP-15 TRANSLATOR
        MTDUMP: MAG-TAPE UTILITY PROGRAM
DOS-15: BATCH
        BATCH(B) DVCUN <UIC> FILNAM]
                DV: ANY VALID XVM/DOS DEVICE WHICH SUPPORTS INPUT
                    PLUS 'SY' MEANING THE CURRENT SYSTEM DEVICE.
                UN: UNIT NUMBER
                UIC: USER IDENTIFICATION CODE
                FILNAM: A FILE NAME WITH THE REQUIRED EXTENSION 'BAT'
        $JOB: CONTROL COMMAND WHICH SEPARATES JOBS
        $DATA: BEGINNING OF DATA
        $END: END OF DATA
        $PAUSE: WAIT FOR ^R ON TTY
        $EXIT: LEAVE BATCH MODE
        ^T: SKIP TO NEXT JOB
        ^C: LEAVE BATCH MODE
        ^R: CONTINUE FROM $PAUSE


XVM/DOS Vnxnnn

$
```

Form 2: I[NSTRUCT]␣ERROR[S]↵

Example:

```
XVM/DOS Vnxnnn

$I ERRORS

DOS-15 - .IOPS:
         0 ILL FUNCTION CAL - CAL ADDR
         1 CAL* ILL - CAL ADDR
         2 .DAT SLOT ERROR - CAL ADDR
         3 ILL INTERRUPT - I/O STATUS REGISTER
         4 DEV NOT READY  - TYPE ^R WHEN READY *
         5 ILL .SETUP CAL - CAL ADDR
         6 ILL HANDLER FUNCTION - CAL ADDR *
         7 ILL DATA MODE - CAL ADDR *
        10 FILE STILL ACTIVE - CAL ADDR **
        11 SEEK/ENTER/INIT NOT EXECUTED - CAL ADDR *
        12 UNRECOVERABLE DEVICE ERROR - STATUS REG B AND UNIT NO.
        13 FILE NOT FOUND - CAL ADDR **
        14 DIRECTORY FULL - CAL ADDR
        15 DEVICE FULL - CAL ADDR **
        16 OUTPUT BUFFER OVERFLOW - CAL ADDR
        17 TOO MANY FILES FOR HANDLER - CAL ADDR *
        20 DISK FAILURE (^R TO RETRY) - DISK STATUS,BLK #,DEVICE/UNIT #,CAL FU
        21 ILL DISK ADDR - BLOCK NO,DEVICE/UNIT NO,CAL FUNCTION,UIC
        22 TWO OUTPUT FILES ON ONE DECTAPE UNIT - CAL ADDR
        23 OUTPUT BUFFER OVERFLOW - CAL ADDR
        24 ILL UNIT NUMBER - CAL ADDR
        25 NEGATIVE OR 0 CHARACTER COUNT (IOPS ASCII WRITE)
           X OR Y INCREMENT TOO LARGE (>2**14) (BINARY WRITE)
        27 ILLEGAL WRITE TYPE
        30 API SOFTWARE LEVEL ERROR - API STATUS REG
        31 NON-EXISTENT MEMORY REF - PC
        32 MEMORY PROTECT VIOLATION - PC
        33 MEMORY PARITY ERROR - PC
        34 POWER FAIL SKIP NOT SETUP - PC
        35 CONTROL CHARACTER TYPED WHEN RECOVERY NOT POSSIBLE
           - CONTROL CHARACTER ROUTINE ADDRESS
        37 LINE OVFLO - CAL ADDR
        40 HEADER LABEL ERROR - CAL ADDR
        41 DIRECTORY FORMAT ERROR - CAL ADDR
        42 ACCESSIBILITY MAP OVFLO - CAL ADDR
        43 DIRECTORY RECORDING ERROR - CAL ADDR
        44 LOGICAL EOT FOUND - CAL ADDR
        45 LONG INPUT RECORD - CAL ADDR
        46 ATTEMPT TO DELETE SYSTEM FILE - CAL ADDR
        47 ILL HORIZONTAL TAB - CAL ADDR
        51 ILLEGAL USER FILE DIRECTORY - CAL ADDR **
        55 NO BUFFERS/TCB AVAILABLE - CAL ADDR *
        61 PARITY ERROR IN DIRECTORY OR FILE BIT MAP - CAL ADDR *
        63 PROTECTED USER FILE DIRECTORY - CAL ADDR *
        64 PROTECTED FILE - CAL ADDR **
        65 UNRECOVERABLE MAGTAPE/TELECOMUNICATIONS ERROR - STATUS WORD
        66 RELATIVE BLOCK IS 0 OR NOT WITHIN FILE SCOPE (.RTRAN) **
        67 I/O BUFFER LOCATED ABOVE 32K BOUNDARY - CAL ADDR
           .RTRAN ARGUMENTS CAUSE DATA BLOCK OVERFLOW - CAL ADDR **
        70 BUFFER SIZE TOO SMALL - CAL ADDR *
        71 EMPTY UIC **
```

```
        72 INPUT PARITY OR WRITE CHECK ERROR (^R TO RETRY)
            - CALADDR,BLOCK NO,DEVICE/UNIT NO,CAL FUNCTION,UIC
        73 NULL FILE NAME GIVEN ON SEEK/ENTER/DLETE/FSTAT/RAND *
        74 FILE STRUCTURE DEGRADATION - ATTEMPT TO CLEAR SUBMAP
           BIT THAT WAS ALREADY OFF **
        75 FILE STRUCTURE DEGRADATION - ILLEGAL SUBMAP WORD1
        76 FILE STRUCTURE DEGRADATION - ILLEGAL BACKWARD POINTER FOR FIRST
           MFD OR UFD BLOCK (^R TO RETRY)
        77 ATTEMPTED USE OF NON-EXISTANT ^Q AREA
          *           DISK ONLY:
                      CAL ADDR,DEVICE AND UNIT NO.,CAL FUNCTION,UIC
          **          DISK ONLY:
                      CAL ADDR,DEVICE AND UNIT NO.,CAL FUNCTION,UIC,FILE NAME
UC-15 ERROR MESSAGES
       FORM: IOPSUC TSK ###
WHERE TSK IS ONE OF THE FOLLOWING:
       EST     STOP I/O TASK
       ESD     SOFTWARE DRIVER
       RKU     DISK CARTRIDGE
       DTU     DECTAPE
       LPU     LINE PRINTER
       CDU     CARD READER
       PLU     X-Y PLOTTER
       ESP     SPOOLER
       EMA     MAC-11
       PRX     POLLER
       LVU     LV-11 ELECTROSTATIC PRINTER/PLOTTER
AND ### IS ONE OF THE FOLLOWING:
       3  ILLEGAL INTERRUPT TO DRIVER
       4  DEVICE NOT READY
       12 DEVICE FAILURE
       15 SPOOLER DISK AREA FULL
       20 SPOOLER DISK FAILURE
       25 XY PLOTTER - VALUE TOO LARGE FOR PLOTTING
       26 SPOOLER TERMINATED UNEXPECTEDLY
       27 XY PLOTTER - INCORRECT MODE
       45 GREATER THAN 80 COLUMNS IN CARD
       55 NO SPOOLER BUFFERS AVAILABLE
       72 ILLEGAL PUNCH COMBINATION
       74 TIMING ERROR - CARD COLUMN LOST - RETRY CARD
       75 HARDWARE BUSY - DRIVER NOT BUSY
       76 HARDWARE ERROR BETWEEN CARDS
       77 UNRECOGNIZED TASK REQUEST
       200 NON-EXISTANT TASK REFERENCED
       300 ILLEGAL API LEVEL GIVEN.
       400 SPOOLER EMPTY - PDP-15 INPUT REQUEST PENDING
           ILLEGAL DIRECTIVE CODE GIVEN
       500 NO FREE CORE IN PDP-11 LOCAL MEMORY
       600 ATL NODE FOR THIS TCN MISSING
       601 FREE CORE MAP FOR LOCAL MEMORY IS BAD
       602 TCB ADDRESS GREATER THAN 28K
       777 REQUEST NODE WAS NOT AVAILABLE
LOADER ERRORS - .LOAD OR .SYSLD
        1 MEMORY OVERFLOW
        2 DATA ERROR
        3 SUBR NOT FOUND
        4 .DAT SLOT ASSIGNMENT ERROR
        5 PROG SEGMENT GREATER THAN 4K (PAGE MODE)
        6 COMMON BLOCK LENGTH GREATER THAN 32K-1 WORDS
        7 RELOCATED VALUE OF SYMBOL GREATER THAN 32K-1
        8 ATTEMPT TO INITIALIZE A LOCATION OUTSIDE THE
          RANGE OF THE CURRENT PROGRAM OR COMMON BLOCK
        9 WARNING -- MORE THAN ONE HANDLER LOADED FOR
          A DEVICE, THE LOADING PROCESS CONTINUES
```

```
OBJECT TIME SYSTEM ERRORS - .OTS
         5 ILL REAL SQUARE ROOT ARG
         6 ILL DOUBLE SQUARE ROOT ARG
         7 ILL INDEX IN COMPUTED GOTO
        10 ILL I/O DEV #
        11 ILL INPUT DATA
        12 ILL FORMAT STATEMENT
        13 ILL REAL LOG ARG
        14 ILL DOUBLE LOG ARG
        15 ZERO RAISED TO ZERO OR NEGATIVE POWER
        16 ATAN2(0.0,0.0)
        17 DATAN2(0.000,0.000)
        21 UNDEFINED FILE
        22 ILLEGAL RECORD SIZE
        23 SIZE DISCREPANCY
        24 TOO MANY RECORDS OR ILLEGAL RECORD #
        25 MODE DISCREPANCY
        26 TOO MANY OPEN FILES
        30 SINGLE INTEGER OVERFLOW
        31 EXTENDED (DOUBLE) INTEGER OVERFLOW
        32 SINGLE FLT. OVERFLOW
        33 DOUBLE FLT. OVERFLOW
        34 SINGLE FLT. UNDERFLOW
        35 DOUBLE FLT. UNDERFLOW
        36 FLT. DIVIDE CHECK
        37 INTEGER DIVIDE CHECK
        40 ILLEGAL CHARACTER COUNT
        41 ARRAY EXCEEDED
        42 BAD INPUT DATA
        50 FPP MEMORY PROTECT/NON-EXISTANT MEMORY VIOLATION
        51 ILLEGAL I/O DIRECTION CHANGE TO DISK

XVM/DOS Vnxnnn

$
```

8.3.3 REQUEST

The REQUEST command causes a typeout of the I/O Device Handlers currently associated with the slots of the Monitor's Device Assignment Table (.DAT). Since this command is closely related to the commands which affect I/O device assignments, it is described in paragraph 8.5.1.

8.3.4 MODE

The MODE command causes the typeout of the following mode specification message. This mode specification message is also output immediately after the date is entered; and after any relevant mode changes.

$$\text{nnnK}\ \begin{Bmatrix}\text{BANK}\\ \text{PAGE}\end{Bmatrix}\ \begin{Bmatrix}\text{API ON}\\ \text{API OFF}\end{Bmatrix}\ \left[\begin{Bmatrix}\text{POLLER ON}\\ \text{POLLER OFF}\end{Bmatrix}\ \begin{Bmatrix}\text{UC15 ON}\\ \text{UC15 OFF}\end{Bmatrix}\right]\left[\begin{Bmatrix}\text{XVM ON}\\ \text{XVM OFF}\end{Bmatrix}\right]\ \text{uic}$$

where:

nnK	always appears and "nnn" is the current memory size (MEMSIZ).
BANK	appears whenever the system is in bank mode.
PAGE	appears whenever the system is in page mode.
{API ON / API OFF}	appears whenever the API hardware is present and enabled (ON) or disabled (OFF).
{UC15 ON / UC15 OFF}	appears whenever the UC15 peripheral processor is present and its use is enabled (ON) or disabled (OFF).
{POLLER ON / POLLER OFF}	appears whenever the UC15 peripheral processor is present and the poller is enabled (ON) or disabled (OFF).
{XVM ON / XVM OFF}	appears whenever the XM15 hardware option is present and wide addressing is enabled (ON) or disabled (OFF).
uic	is the User Identification Code.

8.4 COMMANDS RELATED TO FILE PROTECTION

8.4.1 LOGIN

This command permits the operator to enter his User Identification Code (UIC) into the system in order to do directoried disk I/O. After a LOGIN, the Nonresident Monitor sets the slots of the User File Directory Table (UFDT) to the three-character code entered. All input/output operations to the disk are directed to the UFD associated with the last UIC entered by this command, unless a program has subsequently executed a .USER I/O Macro, or the operator has issued an ASSIGN command (see 8.5.2).

Each LOGIN command issued enters a new UIC into the system and automatically deletes the one entered previously. Each LOGIN is an implicit LOGOUT (see LOGOUT, in paragraph 8.4.4). A UIC must consist of exactly three alphanumeric characters in any combination except "@@@", "???", "PAG", "BNK", "SYS", "IOS" and "CTP".

Form: LOGIN␣uic)

where: uic = User Identification Code

8.4.2 MICLOG

This command permits the Monitor Identification Code (MIC) to be entered into the system. This provides the operator with unrestricted access to all files contained in the various directories on the disk and permits the system programs SPLGEN[1], SGEN and PATCH to be used to modify the system. The MIC of each system, as initially supplied to the user, is "SYS". As with LOGIN, a MICLOG entry is deleted from the system by the LOGOUT or LOGIN commands. The MIC is usually known only by the system owner, and the code is easily changed at system generation.

Form: MICLOG␣mic)

where: mic = Monitor Identification Code

8.4.3 PROTECT

The PROTECT command is used to alter the default value of the file protection code, set when the system was generated (by the SGEN program). The default file protection codes set by this command remain in effect until another PROTECT command is given or until the user issues a LOGIN or LOGOUT (which resets the protection code to the system's default value). Refer to 4.7.3 for a list of these codes.

Form: P[ROTECT]␣n)

where: n = Protection Code

8.4.4 LOGOUT

This command deletes the current UIC or MIC entry from the system. LOGOUT also resets all system parameters affected by keyboard commands to their default status. These parameters include:

[1] UC15 Systems only.

a. .DAT and .UFDT assignments (ASSIGN command)

b. Commands which take an "ON/OFF" argument, such as: KEEP, FILL, TAB, HALF, LP, PAGE/BANK, API, and VT.

c. Commands which take a numeric argument, including: CHANNEL, PROTECT, and BUFFS.

Form: LOGOUT)

8.5 COMMANDS DEALING WITH I/O DEVICE ASSIGNMENTS

8.5.1 REQUEST

This command causes a typeout of the I/O devices currently associated with the slots of the Monitor's Device Assignment Table (.DAT) and the UIC's associated with the User File Directory Table (.UFDT). The command can be issued using various arguments which result in a complete printout of the assignments or selected portions thereof. If REQUEST is issued with no argument, the entire .DAT/.UFDT list of assignments is output. If the argument USER is inserted, only the positive (user) .DAT and associated .UFDT assignments are output. If an argument which is a legal system program name is used (e.g., MACRO, PIP, etc.), only the assignments for that program are output.

Form: R[EQUEST]␣ [USER / or / prog]

"prog" may be any of the following:

MACRO	UPDATE	CHAIN
F4	PIP	EXECUTE
FOCAL	[1]MAC11	8TRAN
PATCH	[2]SPOOL	DDT
DTCOPY	EDIT	DDTNS
GLOAD	EDITVP	BOSS
LOAD	EDITVT	[1]SPLGEN
DUMP	SGEN	[1]SPLOAD
MTDUMP	SRCCOM	

Example 1:

```
$R

.DAT    DEVICE  UIC
-15     SYA     SCR
-14     SYA     SCR
-13     SYA     SCR
-12     LPA     SCR
```

[1]Unichannel-15 systems only.
[2]Only for RK based UC15 systems.

```
-11      SYA      SCR
-10      CMD      SCR
-7       RKL      SYS
-6       SYA      SCR
-5       NON      SCR
-4       SYA      SCR
-3       TTA      SCR
-2       CMD      SCR
-1       SYA      SYS
+1       SYA      SCR
+2       SYA      SCR
+3       SYA      SCR
+4       TTA      SCR
+5       PRA      SCR
+6       LPA      SCR
+7       DPA      SCR
+10      DPA      SCR
+11      DTA      SCR
+12      DTA      SCR
+13      PPA      SCR
+14      NON      SCR
+15      NON      SCR
+16      NON      SCR
+17      NON      SCR
+20      NON      SCR

$
```

Example 2:

```
$R USER

.DAT     DEVICE   UIC
+1       SYA      SCR
+2       SYA      SCR
+3       SYA      SCR
+4       TTA      SCR
+5       PRA      SCR
+6       LPA      SCR
+7       DPA      SCR
+10      DPA      SCR
+11      DTA      SCR
+12      DTA      SCR
+13      PPA      SCR
+14      NON      SCR
+15      NON      SCR
+16      NON      SCR
+17      NON      SCR
+20      NON      SCR

$
```

Example 3:

```
$R MACRO

.DAT     DEVICE   UIC      USE
-14      SYA      SCR      INPUT
-13      SYA      SCR      OUTPUT
-12      LPA      SCR      LISTING
-11      SYA      SCR      INPUT
-10      CMD      SCR      SECONDARY INPUT
-1       SYA      SYS      SYSTEM MACRO FILE

$
```

8.5.2 ASSIGN

This command permits the temporary reassignment of the various slots of the Monitor's Device Assignment Table (.DAT) to I/O device handlers other than those permanently assigned at system generation. In addition, the corresponding slots of the User File Directory Table (.UFDT) can also be reassigned to UIC's other than the UIC which is currently in effect. Unless the KEEP command is issued, the change of assignment is effective only for the current job (i.e., the program about to be run), since the permanent assignments are restored when the Nonresident Monitor regains control (i.e., after the current job has terminated). The KEEP command (described below) can be used to retain assignments from job to job.

Prior to using ASSIGN, the user should be familiar with the various handlers which can be used with the program for which the assignments are to be made. Chapter 9 describes the handlers in the system. A list of the handlers available on any given system can be obtained in the printout obtained with the SCOM command. The following rules should be observed when typing ASSIGN commands:

a. Device handler names consist of three characters which can be abbreviated to two characters if the last character is an "A". Thus, "DKA" becomes "DK". In addition, a number can be typed as a fourth character to specify the device unit number (in octal). The unit number is applicable for devices which can have more than one unit: Disk Pack, Disk Cartridge, DECtape, and Magtape. If the unit number is zero, it need not be specified. Thus, "DTAØ" becomes simply "DT", similarly, "DPBØ" can be typed as "DPB". "DTA1" may be typed as "DT1".

b. .DAT/.UFDT slot numbers (octal) must be within the legal range for the particular system being used. Since the number of negative slots does not change (-15 is the lowest negative slot), the user need only be concerned with the number of positive slots available. This can be determined either from a SCOM or a REQUEST USER command.

c. A series of assignments can be typed on the same line, using a single ASSIGN command, by separating the assignments with a slash (/). The user can then type another device name, UIC, and slot number(s). (See examples below.)

d. Assigning NON instead of a device handler name will assign a null handler to .DAT slots that are not needed. This will save core since no handler will be loaded at run time.

e. The pseudo-device 'SY' (or 'SYA' or 'SYS', all equivalent) may be assigned. The actual handler used will be that of the system resident disk type (i.e., RK, RP, or DK).

f. The pseudo-device 'CM' (or 'CMA' or 'CMD', all equivalent) may be assigned. The actual device used will be the Batch device/command file in Batch mode, the BOSS Run Time File if in BOSS mode, or TTA if in neither.

g. Two pseudo-device handler names are available and may be specifield instead of a device handler name. 'SY' is a pseudo-device meaning 'the system-resident disk type'. In other words, if a system is RP02-based, SY is equivalent to using DP. Unit numbers and uic specifications may be used as with a disk name, but a handler version (A,B,C,etc.) may not be used. 'CM' is a pseudo-device meaning 'the command device'. This pseudo-device is designed for use with the Monitor Command Batch facility (see Section 8.12). Input from the CM pseudo-device is equivalent to the console terminal (TT) device if the system is not in batch mode, or the batch input device or file if the system is in batch mode. Output a .DAT slot assigned to CM to equivalent to the console terminal printer (TT) device.

Form: A[SSIGN]␣ { dev / dev / NON } { <uic> / <uic> } ␣a[,b,c,etc.]↵

where:

uic = legal User Identification Code

dev = Device Handler name and unit (if applicable)

a,b,c,etc. = Legal slot numbers

NON = No device handler

Examples:

1. To assign the teleprinter to .DAT slot -11 and the paper tape reader (version A) to .DAT slot 14, type:

   A␣TT␣-11↵

   A␣PR␣14↵

   or

   A␣TT␣-11 / PR␣14↵

2. To assign UFD "ABC" to .UFDT slot -14, and the Disk Pack and UFD "TRE" to . UFDT slot 1Ø type:

   A␣<ABC>␣-14 / DP␣<TRE>␣1Ø↵

3. To assign the Disk Pack to several .DAT slots, type:

   A␣DP␣1,2,3,15↵

8.5.3 KEEP ON/OFF

This command instructs the Monitor either to retain or reset .DAT/ .UFDT slot assignments after the current program (for which the assignments were made) terminates execution and control returns to the Monitor. "ON" retains assignments and "OFF" allows them to be reset. When a LOGOUT command is issued, the "OFF" parameter is automatically set.

Form: K[EEP]␣{ON / OFF}↵

8.6 CORE ALLOCATION COMMANDS

8.6.1 BUFFS

This command temporarily changes the parameter which specifies the default value for the number of buffers available in the Monitor's buffer pool for disk I/O and for the .GVBUF and .GTBUF Monitor Commands. The default value is restored whenever the Nonresident Monitor returns. This value is set during system generation along with the actual size of the buffers to be allocated (the default values for systems as initially distributed are: Number of Buffers = 3, Buffer Size = 500_8).

The user should exercise care when issuing this command, since the Disk device handlers and the DECtape "A" handler obtain the buffers as required for each opened file, and terminal errors result when an insufficient number of buffers is available. Alternatively, program loading errors occur when the number of buffers allocated results in an insufficient amount of core for program loading. When requesting system programs, the user need not be concerned about buffer availability since each system program has its own default parameter for the maximum number of buffers required (e.g., MACRO has 4, EDIT has 4).

Form: BUFFS␣n↵

where: n = number of buffers (decimal radix) desired.

Table 8-1

SYSTEM PROGRAM LOADING COMMANDS

Command	Program Loaded
F4	FORTRAN IV compiler.
MACRO	MACRO XVM Assembler.
*MAC11	MAC11 Assembler.
EDIT	Symbolic Text Editor.
EDITVP	Symbolic Text Editor for the VP15A Display.
EDITVT	Symbolic Text Editor for the Graphic-15 System.
LOAD	Linking Loader (manual program start)
GLOAD	Linking Loader (load and go program start)
PIP	Peripheral Interchange Program.
DDT	Dynamic Debugging Technique Program
DDTNS	DDT program with no user symbol table loaded (i.e., octal number debugging).
DUMP	Program to create listing of the contents of the QAREA (see 8.7).
CHAIN	Program to create a system of core overlays.
E[XECUTE]	Control program which supervises core residency during execution of a CHAIN-built overlay system.
SRCCOM	Source Compare Program (for comparing two ASCII files).
MTDUMP	Magtape user's utility program.
SGEN[1]	System Generator Program.
PATCH[2]	Mass Storage Patching Program.
*SPLOAD	SPOOLER Installation Program.
*MCLOAD	MAC11 Installation Program.
*SPOOL	SPOOLER Control Program.
*SPLGEN[1]	SPOOLER Disk Area Generator.
UPDATE	Program to create and update library files.
8TRAN	Program to translate PDP-8 assembly code to XVM assembly code.

[1]This program can only be run when the user is logged-in to the system with the MICLOG command.

[2]When this program is used with the system device, the user must be logged-in under the MIC.

*UNICHANNEL-15 system only.

8.6.2 XVM ON/OFF

This command is used to turn XVM mode on or off, which affects the amount of memory available for loading user programs.

Form: XVM␣ON/OFF↵

XVM mode requires the presence of the XM15 hardware option. If this option is not present when the "XVM ON" command is given, the following message is output:

```
XVM NOT AVAILABLE
```

When XVM mode is disabled, the effective top of memory which can be used is the address of the system bootstrap. When XVM mode is enabled, user program COMMON blocks are permitted to reside in memory above the system bootstrap. The setting of this parameter does not effect the "MEMSIZ" parameters (paragraph 8.6.3). Whenever the XVM ON/OFF command is given, a mode specification message is output as described in Section 8.3.4.

8.6.3 MEMSIZ

This command is used to set the current system memory size.

Form: MEMSIZ␣numK↵

where "num" is a decimal number in the range from 24K through 128K in 8K increments. This command attempts to set the system memory size to the value entered. The memory size may not be set lower than the bootstrap address. If "num" is less than the bootstrap address or if the XM15 hardware option is not present, "MEMSIZ" is set to the bootstrap address and the following message is output:

```
nnK NOT AVAILABLE--MEMORY SIZE=mmmk
```

where "nnnk" is the requested memory size and
"mmmK" is the actual (physical) memory size or bootstrap address

The above message is also output whenever the requested memory size exceeds the actual memory size. "MEMSIZ" is then set to the actual memory size.

Whenever the MEMSIZ command is given, a mode specification message is output as described in Section 8.3.4.

8.7 CORE IMAGE SAVE/RESTORE COMMANDS

The commands described in this section provide facilities for saving and restoring the entire image of core. These commands can be used to advantage not only for obtaining "snapshots" of core for debugging purposes, but also for rapid loading of commonly used programs, particularly user-created programs which use many library routines.

These commands work in conjunction with a reserved area on the DECdisk and each Disk Pack or cartridge called the Save Area, or QAREA. The QAREA is a temporary storage area into which a core image may be dumped and from which core images are restored. This is considered a temporary storage area since each time a command to dump core is given, the previous contents of this area are lost. Additional commands are therefore supplied, to permit users to create named files of core images in the QAREA. Such files can be used for permanent storage and can be restored to core via Keyboard commands. The DUMP program can be used to obtain listings of core dumped into the QAREA.

8.7.1 CTRL Q

This command interrupts the program currently running, dumps the contents of core into the QAREA of the system device and returns control to the Nonresident Monitor. The ↑Q function dumps the minimum of the ↑Q area size and the current system memory size. The command is typed by simultaneously depressing the CTRL and Q keys on the keyboard. Upon receipt of the command, the Monitor echoes a ↑Q. (If no echo occurs, no QAREA exists on the device, or the system has crashed.) After the ↑Q is echoed, core is dumped into the QAREA and the Monitor regains control as shown below.

Form: CTRL Q

Response:
```
↑Q
XVM/DOS Vnxnnn
$
```

8.7.2 QDUMP

This command instructs the Monitor to automatically execute a CTRL Q command (see above) when a terminal IOPS error is detected. IOPS errors are listed in Appendix D.

Form: Q[DUMP]

8.7.3 PUT

This command instructs the Nonresident Monitor to create a file on the device associated with .DAT slot -14. The file is a copy of the contents of the ↑Q area on this system device (unit Ø). The size of the file is equal to the minimum of the ↑Q area size and the current system memory size. This value is kept in .SCOM, and will be stored in the file (as part of .SCOM in the core image) for use by GET and .GET. The user must type a file name which may consist of up to six characters and a three character extension.

Form: PUT␣filnam␣ext↵

where:

filnam = File Name (1-6 characters)
ext = Required filename extension (1-3 characters)

8.7.4 GET

This command instructs the Monitor to restore (to core) a core image residing either in the QAREA of the system device, or on the device associated with .DAT slot -14 as a named file (i.e., created by a PUT command). No name with the command gives the first option. If the user supplies a name, the second option (.DAT -14) will be taken.

The GET command determines the size of the image by examining the copy of the register in the .SCOM register in the image. If the image size is larger than either the size of the ↑Q area or the current system memory size, the following question is asked:

IMAGE TOO LARGE - CONTINUE (Y/N)?

If the reply is "N", control is returned to the Nonresident Monitor. If the reply is "Y", the minimum of the ↑Q area size and the current

system memory size is copied into the ↑Q area and into memory. If the image size is smaller or equal to both the ↑Q area size and the current memory size, the image will be copied into the ↑Q area and into memory.

Execution of the restored core image is resumed by one of the Program Start/Restart Commands described in Paragraph 8.11 (CTRL P, CTRL T, or CTRL S). These commands can be issued either manually from the keyboard or automatically by argument in the GET command string.

> When a restored core image is to be restarted manually (i.e., by typing CTRL P, CTRL T, or CTRL S), the user should wait at least 8 seconds after issuing the GET to ensure the complete transfer of the core image before typing the command.

When restoring a core image file from .DAT -14, the Monitor also places the core image in the QAREA of the system device unit Ø. This permits the user to conveniently use the DUMP program to obtain listings of core image files.

Form: GET [{P / T / S}] [␣filnam␣ext]↵

where:

P,T,S = Perform automatic program start

P = CTRL P start address
T = CTRL T start address
S = CTRL S start address

Note that there must be no space between GET and P, T, or S.

If not specified, the appropriate CTRL character must be issued from the keyboard.

filnam ext = Name and extension of core image to be retrieved from .DAT -14. If not specified the contents of the QAREA are restored.

NOTE

All system conditions and parameters extant when core was dumped are restored. This includes the DATE and TIME .SCOM registers.

Examples:

1. Restore the QAREA:

   type: GET↵
   or G↵

2. Restore a core image file called DMPFIL ØØ1 and automatically start at the CTRL P address:

   type: GETP␣DMPFIL␣ØØ1↵

3. Restore a core image file called DMPFIL ØØ2 to the QAREA:

   type: GET␣DMPFIL␣ØØ2↵

   Notice that a manual start must be performed, since P, T or S was not specified with the GET.

8.8 VT15 DISPLAY COMMANDS

The commands described in this section provide users who have configurations which include a VT15 Display Processor and a VTØ4 Display Console with the ability to display any text normally directed to the teleprinter on the screen of the VTØ4 Display Console. The control commands are issued from the teleprinter keyboard and permit rapid switching between hard and soft copy output when operating with either the Monitor, system programs, or user programs. Up to fifty-six 72-character lines can be displayed. Keyboard input is echoed both on the display (when ON) and the teleprinter. When operated in this manner, the VT15/VTØ4 Display System functions as an extension of the teleprinter and communicates directly with its device handler.[1]

8.8.1 Operating Features

8.8.1.1 Display Modes - Two modes of display are provided which are controlled by the two rightmost pushbuttons (#5 and #6) on the VTØ4 console.

[1] To operate the display as a separate I/O device, the user must use the software package described in VT15 XVM Graphics Software Manual.

a. Continuous (Scroll) Mode - In this mode of operation, pushbutton #5 must be in the OFF position (i.e., not illuminated). Each text line is displayed on the screen starting at the top and progressing to the bottom. When 56 lines have been displayed, or the display buffer is full (as with HALF ON), each additional line causes all displayed lines to move up one line position and the top line to be deleted.

b. Paging Mode - This mode of operation causes the display to stop after 56 lines have been output, or the display buffer is full. The display will then wait for the user's signal. Paging Mode is entered by setting pushbutton #5 to the ON position (i.e., button #5 is illuminated). The next display page is obtained by depressing pushbutton #6 once.

8.8.1.2 Clearing the Display Screen - The display screen can be erased at any time by depressing pushbutton #6 once and typing a Carriage RETURN.

8.8.1.3 Editing - Both single characters and entire lines can be deleted during input from the teleprinter keyboard using the standard keyboard editing commands RUBOUT and CTRL U. The only difference on the VT is that, when using RUBOUT, no backslash (\ ) is echoed on the display; the unwanted character is simply deleted.

8.8.2 Display Command Descriptions

The following paragraphs describe the three keyboard commands required for operating the display.

8.8.2.1 VT ON/OFF - The ON argument of this command instructs the Monitor to load the routines which interface the VT15 to the teleprinter's device handler and set up the display buffer to the size specified by the HALF ON/OFF command (or its default setting). After this command has been typed, the user can switch at will between teleprinter and VT15 output using the CTRL X command described below. The routines and buffer space for display operation occupy approximately $13\emptyset\emptyset_{1\emptyset}$ locations (when HALF is on) or $2\emptyset\emptyset\emptyset_{1\emptyset}$ locations (when HALF is off). The OFF argument of this command erases the display screen and releases the core area occupied by the display routines and display buffer.

Form: VT␣{ON / OFF}↵

8.8.2.2 HALF ON/OFF - This command is used to set the size of the display buffer. This, in turn, limits the maximum number of characters which can be displayed on the VTØ4 screen. The QFF argument permits a full size buffer to be created. The full size buffer allows $4Ø32_{1Ø}$ characters to be displayed (i.e., fifty-six 72-character lines).

The ON argument allows only a half size display buffer to be loaded. A half size buffer allows 2016 characters to be displayed (e.g., 28 72-character lines). Since most lines are not 72 characters long, more than 28 lines can usually be displayed with HALF ON. This feature is particularly useful during assembly or compiling operations when additional symbol table space is required.

Form: HALF␣{ON / OFF}↵

8.8.2.3 CTRL X - This command, formed by typing CTRL and X simultaneously, alternately switches text output either to the teleprinter or to the VTØ4 screen. Once VT ON has been issued, CTRL X can be typed at any time (i.e., with the Monitor, a system program, or a user program) to change output control. An up-arrow (↑) is echoed on the device to which control is transferred.

8.8.2.4 Command Default Settings - The commands VT ON/OFF and HALF ON/OFF can be initially set during system generation to meet user requirements. The default settings for the XVM/DOS system as initially supplied are: VT OFF and HALF OFF.

8.9 MISCELLANEOUS COMMANDS

8.9.1 API ON/OFF

This command controls the status of the Automatic Priority Interrupt System for machine configurations having this option. The ON argument enables the API and the OFF argument disables the API.

Form: API␣{ON / OFF}↵

If the API hardware is not available, the following message is output:

API NOT AVAILABLE

and the system parameter is set to reflect its absence. Whenever the API ON/OFF command is given, a mode specification message is output as described in Section 8.3.4.

8.9.2 TAB ON/OFF and FILL ON/OFF

The TAB command specifies whether or not the system should simulate the tab character. Specifying "TAB ON", conditions the XVM/DOS Monitor to simulate tabs with the appropriate number of spaces.

Form: TAB␣{ON / OFF}↵

The FILL command enables (ON) and disables (OFF) fill character insertion. Fill characters are necessary to allow completion of carriage returns on LA3Øs running at 3ØØ band.

Form: FILL␣{ON / OFF}↵

The TAB and FILL specification settings are independent of each other.

The following chart summarizes the TAB and FILL settings necessary for several devices:

TAB \ FILL	ON	OFF
ON	LA30 (300 baud)	LA30 (110 baud) LA36 KSR33
OFF	-	KSR35

8.9.3 CHANNEL 7/9

This command sets a constant in the Monitor which is used by the Magtape device handlers as the default operation parameter for 7- or 9-channel operation. Refer to the description of the Magtape handlers in Chapter 9 for further information. 9-25

Form: C[HANNEL] ␣ {7 / 9} ↵

8.9.4 LP ON/OFF

This command permits the text output resulting from the System Information Commands REQUEST, REQUEST USER, and SCOM to be output to a line printer, if one is available. The ON argument directs output to the line printer and the OFF argument restores output to the teleprinter.

NOTE

For INSTRUCT and INSTRUCT ERROR: Assign LP to .DAT -12 to output to the line printer

Form: LP ␣ {ON / OFF} ↵

8.9.5 BANK ON/OFF - PAGE ON/OFF

These commands permit the user to select either Bank or Page Mode operation. In Page Mode, relocatable user programs and device handlers are loaded within 4K memory pages, and Index Register usage is permitted. Library routines are loaded from the library residing in the PAG UFD. In Bank Mode, Index Register usage is not permitted, and user programs (including device handlers) are loaded within 8K memory banks, and system library routines are loaded from the library contained in the BNK UFD. Either BANK OFF or PAGE ON sets the system to operate in Page Mode. Conversely, BANK ON or PAGE OFF sets the system to operate in Bank Mode.

Form: BANK ␣ {ON / OFF} ↵ or PAGE ␣ {ON / OFF} ↵

Whenever the BANK ON/OFF or PAGE ON/OFF commands are given, a mode specification message is output as described in Section 8.3.4.

8.9.6 POLLER ON/OFF[1]

This command controls the UNIBUS device error 'poller' initiated by the XVM on a UNICHANNEL-15 system. The ON argument enables the operation of the poller while the OFF argument disables the operation of the poller. This command is provided to temporarily permit usage as a single integrated system or as two distinct and separate systems.

Form: POLLER { ON / OFF } ↵

The poller checks both the "UC15" and POLLER switches to determine if it should run. The "POLLER ON/OFF" command has no effect with "UC15 OFF", however, if the "UC15 ON" command is subsequently given, the setting of the "POLLER" switch determines whether or not the poller should run.

The poller periodically queries the UC15 PIREX monitor for the presence of error conditions. UC15 option system users wanting to either place the UNICHANNEL system off line for maintenance or to use the UC15 processor to run other PDP-11 monitors (such as RSX-11M or RT11) will want to turn the POLLER OFF. Whenever the POLLER ON/OFF command is given, a mode specification message is output as described in Section 8.3.4.

8.9.7 DATE

This command is used either to enter a calendar date or to examine the calendar date currently stored in the Monitor. This information is used by the system to indicate the date of creation of mass storage files. If no date has been entered into the system, each time control is returned to the Monitor it outputs a message requesting that a date be entered. When a date is to be entered, the entire date (month, day and the last two digits of the year) must be typed. The DATE command with no argument causes the date currently stored in the Monitor to be typed out. When 24ØØ hours have passed, the system automatically requests a fresh date to be entered. The clock is required with the UC15 system.

Form: D[ATE]␣[mm[/]dd[/]yy]↵

[1]For UC15 systems only.

where:

mm = Month (Ø1-12)
dd = Day of Month (Ø1-31)
yy = Year (7Ø-99)

If the slash delimiters (/) are used, leading zeroes can be omitted; otherwise all six digits must be typed.

8.9.8 TIME

This command is used either to enter or to examine the time of day currently stored in the Monitor. This information is updated by the Monitor every second. When time is to be entered, it must be typed as a 4-digit number (ØØØØ-2359) in 24-hour clock notation. The TIME command with no argument causes the current time of day (as kept in the Monitor) to be typed out.

Form: T[IME][␣hhmm]↵

where:

hh = Hours (ØØ-23)
mm = Minutes (ØØ-59)

8.9.9 TIMEST

This command instructs the Monitor to terminate the current operation and restart itself after a specified time interval has elapsed. The command is particularly applicable when operating in Command Batching Mode (discussed in 8.12), since it permits users to unconditionally terminate a program's operation if its execution time exceeds that value expected for normal operation. The time interval can be specified in minutes, seconds, or a combination of both. The maximum time interval (total minutes and seconds) can not exceed 131,071 seconds. Once executed, a TIMEST command can only be nullified by a subsequent LOGIN or LOGOUT command.

Form: TIMEST␣ { mm: / mm:ss / ss← }↵

where:

mm - Minutes
ss = Seconds

Example: TIMEST 5281 (5281 seconds)

or

TIMEST 88:1 (88 minutes, 1 second)

8.9.10 LOG

This command instructs the Monitor to ignore subsequent keyboard input and is used primarily for making operator comments. Typing ALT MODE restores the keyboard for normal command input.

Form: L[OG] ↵

comment
.
.
.
comment ALT MODE

8.9.11 HALT

This command will cause the Monitor to halt computer operation after terminal IOPS errors. Press CONTINUE, followed by CTRL P or CTRL C to continue system operation.

Form: H[ALT] ↵

8.9.12 CTRL D

This command, formed by simultaneously striking the CTRL and D keys on the teleprinter keyboard, is used to indicate an end-of-file condition when the keyboard is used as an ordinary input device (as opposed to its being used as a command input device). CTRL D signals the teleprinter's device handler, or other keyboard device handler, if available, to transmit a header word pair to the requesting program's I/O buffer in which bits 14-17 are set to the end-of-file code $Ø1Ø1_2$. Any information currently in the buffer is lost. In IOPS ASCII mode, therefore, a Carriage RETURN should always precede a CTRL D in order to assure output of the last line.

8.9.13 UC15 ON/OFF

This command is used to enable and disable the use of the UC15 peripheral processor.

Form: UC15␣ {ON / OFF} ↵

If the UC15 hardware is not present, it is not enabled and the following message is output:

UNICHANNEL NOT AVAILABLE

When the UC15 is disabled, the POLLER ON/OFF command has no effect (see paragraph 8.9.6).

Whenever the UC15 ON/OFF command is given, a mode specification message is output as described in Section 8.3.4.

8.10 SYSTEM PROGRAM LOADING COMMANDS

The commands described in Table 8-1, System Program Loading Commands, are used to instruct the system loader within the Monitor to load the various language and utility programs which are part of the XVM/DOS Software System. Each command must be typed as shown, terminated by either a Carriage RETURN or ALT MODE.

8.11 PROGRAM START/RESTART/CONTINUE COMMANDS

The commands described below are used to provide keyboard control during system and user program operation. The format of these commands has been previously described in paragraph 8.2.1.

8.11.1 CTRL C

This command returns control to the Nonresident Monitor. It can be typed at any time, signaling the Monitor that the user wishes to abort a program, or enter a keyboard command. In returning control to the Nonresident Monitor from a previously executing program, all device and UFD assignments are returned to their default settings unless the KEEP ON command is in effect, in which case, device and UFD assignments are not affected (see 8.5.2 and 8.5.3). CTRL C can also be used to terminate the Monitor's processing of all keyboard commands except for TIMEST (see 8.9.8).

8.11.2 CTRL P

This command restarts system programs, terminating current operation. Upon execution of this command, control is transferred to the start address specified in the last .INIT I/O Macro to the teleprinter. User programs may use this restart facility by issuing a .INIT macro

to .DAT -2 (which is normally permanently assigned to the teleprinter handler). See 6.7.6 and the description of the teleprinter handler in Chapter 9. CTRL P is ignored until the teleprinter handler receives the proper .INIT.

8.11.3 CTRL S

This command is used to start a program loaded by the Linking Loader via the LOAD command.

8.11.4 CTRL T

This command is used only with the DDT program and the Monitor's Command Batching Mode (see BATCH described below). When used with DDT, CTRL T terminates execution of the program being debugged and causes DDT to enter command mode. When used with the Monitor's Command Batching Mode, it causes the current job to be terminated and skips to the next job.

8.11.5 CTRL R

This command permits the user to continue program operation either after an IOPS4 (I/O device not ready) error occurs, or after the execution of a $PAUSE (see 8.12.3.4). Prior to typing this command, the user must first correct the condition which caused the error (e.g., DECtape unit incorrect, OFF LINE, etc.).

8.12 BATCHING KEYBOARD COMMANDS

Most of the Monitor's keyboard commands, as well as the keyboard commands for most system programs, can be issued from the card reader, a mass-storage file or the paper tape reader. This is possible when the system is in the Monitor's Command Batching Mode. Batching Mode allows many programs to be run in sequence with minimum operator intervention. A typical sequence of operations might include file editing, assembly and program execution. All commands input from the batch device are echoed on the console teleprinter.

8.12.1 Preparation

For Command Batching Mode, the programmer must prepare a file, a paper tape or a deck of punched cards which contains the keyboard

commands for the operations performed. These commands should be in the same order and form as they would normally be issued from the teleprinter keyboard. The only exception to this is that certain job control commands may be inserted in the command sequence. When preparing batch files, the user will find it convenient to use the system's Text Editor Program EDIT. When preparing commands for input from cards, the user can prepare his cards using a card punch which punches either 029 or 026 Hollerith codes (see Appendix F).

8.12.2 Operator Commands

The following commands are provided for operator control:

8.12.2.1 BATCH - This command is used to enter Command Batching Mode. Once this command is issued, the Monitor begins to read from the batch device specified (PR = paper tape reader, CD = CRØ3B or CR15 card reader).

Form: B[ATCH]␣dv{unit}␣{<uic>}␣{filnam}↵

where dv is any device name (see Section 8.5.2) uic, (see Section 8.5.2), and filnam is the name of the batch command file, with the fixed extension of 'BAT'.

8.12.2.2 CTRL T - This command causes the Monitor to skip to the next job (i.e., skip to the next $JOB job separator. See 8.12.3.1).

8.12.2.3 CTRL C - This command is used to terminate Command Batching Mode operation. It operates in all other respects as it does with normal keyboard operation (see 8.11.1).

8.12.2.4 CTRL R - This command is used to recover from either the execution of a $PAUSE (see 8.12.3.4) or an IOPS4 (see 8.11.5).

8.12.3 Job Control Commands

The following commands are inserted into the normal keyboard command sequences on the batch device medium to provide job control.

8.12.3.1 $JOB - This command separates one job from the next. The command operates, within the context of Command Batching Mode, in a manner similar to the CTRL C command since it causes the Batching Mode Nonresident Monitor to be reloaded, and all .DAT/.UFDT slot assignments to be reset to their default setting, if KEEP is OFF.

$JOB may occur as the first command on the batch medium and can be used thereafter each time the user wishes to exit from the current program and issue another command to the Monitor.

Form: $JOB[␣comment]↵

where: comment = User comments

8.12.3.2 $DATA - When the batch device is to be used for data input (as it might be in non-batch mode), this command marks the beginning of the data. Unlike BATCH commands, data is not printed on the teleprinter.

Form: $DATA[␣comment]↵

where: comment = User comments

8.12.3.3 $END - This command follows the $DATA command and data to signify the end of the data, and causes an end of file indication to be sent to the program.

Form: $END[␣comment]↵

where: comment = User comments

8.12.3.4 $PAUSE - This command terminates input from the batch device until the operator types a CTRL R. It has particular application when the user wishes to signal the operator to mount a DECtape, reload the batch device, or perform some other manual operation.

Form: $PAUSE[␣comment]↵

where: comment = User comments

8.12.3.5 $EXIT - This command signals the Monitor to leave Command Batching Mode and resume operation using commands from the console keyboard.

Form: $EXIT[␣comment]↵

where: comment = User comments

8.12.4 Restrictions

When operating in Command Batching Mode, the following restrictions apply:

a. The following commands are illegal: QDUMP, HALT, GET (all forms), PUT, BATCH, BOSS, LOAD, DDT, API ON/OFF.

b. Any ASSIGN command which references the pseudo-device name 'CM' automatically obtain the services of the current batch device handler.

8.13 KEYBOARD ERROR DETECTION AND HANDLING

The Monitor performs comprehensive error checking on all keyboard commands typed. Upon detection of an error, an appropriate message is output to the teleprinter indicating the nature of the error, and the remainder of the line from the error is ignored. The dollar sign ($) prompting symbol is then output to indicate the Monitor's readiness to accept another command. Keyboard errors which result during operation of system programs are explained in the appropriate reference manual for the particular program (see Preface). Error messages which are prefixed by "IOPS", "IOPSUC", ".SYSLD" or ".LOAD" are listed and explained in Appendices D, J and E, respectively.

CHAPTER 9
I/O DEVICE HANDLERS

9.1 INTRODUCTION

This chapter describes the I/O device handling routines which are supplied as a part of the XVM/DOS system software. Included in this chapter are their operating characteristics and their applicability for use with the various language and utility programs in the XVM/DOS system.

Each I/O device handler has a unique three-character name which is used when assigning it to a .DAT slot via the ASSIGN keyboard command (see 8.5.2). The first two characters of the name designate the device with which the handler operates. For example: DK = DECdisk; DP = Disk Pack; RK = Disk Cartridge; TT = Teleprinter, etc. The third character specifies a particular version of a handler, since some I/O devices have several handlers. Thus, DK<u>A</u> designates the "A" version of the DECdisk device handler. Similarly, DT<u>C</u> is the "C" version of the DECtape handler. Table 9-1 lists the standard XVM/DOS I/O device handlers.

The availability of several handler versions allows the user who is concerned with core utilization to select a particular version with the size and capabilities most nearly suited to his needs. Device handler versions differ from one another in the number of I/O functions (Macros) and Data Modes allowed, and in the number of files which can be accessed concurrently. The fewer capabilities allowed, the smaller the handler. "A" version handlers are the largest, but also provide the greatest capabilities. Other versions are more limited, and consequently are smaller.

In selecting a handler, the user most consider all I/O requirements for the program with which it is to run. What I/O Macros and Data Modes are used? Is output required? How many files may be concurrently open? To assist the user in selecting handlers, paragraph 9.2 lists all versions of the handlers which can be assigned to the various .DAT slots used by the various XVM/DOS System programs, and paragraph 9.3 describes the specific functional characteristics of the handlers.

Table 9-1

XVM/DOS I/O DEVICE HANDLERS[1]

Device Name	Version					
	A	B	C	D	E	F
DK (DECdisk)	✓	✓	✓			
DP (Disk Pack)	✓	✓	✓			
RK (Disk Cartridge)	✓	✓	✓			
DT (DECtape)	✓		✓	✓	✓	✓
TT (Teleprinter)	✓					
PR (Paper Tape Reader)	✓	✓				
PP (Paper Tape Punch)	✓	✓	✓			
LP (Line Printer)	✓					
CD (Card Reader)		✓				
VP (VP15A Display)	✓					
MT (Magtape)	✓		✓			✓
XY (XY11/XY311 Plotter)	✓					

9.2 DEVICE HANDLERS ACCEPTABLE TO SYSTEM PROGRAMS[2]

The following paragraphs provide listings of .DAT Slot assignments for the various system programs and the I/O device handlers which may be assigned to each.

[1] By convention, all system programs use .DAT -2 for command input and .DAT -3 for output. Both .DAT slots are permanently assigned to the console device or the batch device file, if in Batch mode. PIP uses these .DAT slots whenever teletype I/O is requested, thus freeing positive slots for other devices. Further all system programs use .DAT -11 and/or -14 for input, -13 and/or -15 for output, -12 for listings and -10 for secondary input.

[2] Users having a VT15 Graphic Display System should refer to the VT15 XVM GRAPHICS Software Manual for descriptions of the associated device handlers, VTA, LKA, and VWA.

NOTE

Only one version of an I/O handler for a particular device may be in core at the same time, since there is no communication between the interrupt handling routines.

9.2.1 FORTRAN IV (F4)

.DAT Slot	Use	Handler
-13	Output	PPA PPB PPC DKA, DPA, DTA, MTA DKB, DPB, DTD, DTE RKA DTF, MTF RKB
-12	Listing	TTA LPA VPA PPA DKA, DPA, DTA, MTA DKB, DPB, RKA DTD RKB DTE DTF, MTF
-11	Input	TTA PRA PRB DKA, DPA, DTA, MTA DKB, DPB, DKC, DPC, DTC, MTC RKA DTD, RKB DTE, RKC DTF, MTF

9.2.2 MACRO

.DAT Slot	Use	Handler
-14	Macro Definitions File	TTA PRA PRB CDB DKA, DPA, DTA, MTA RKA DKB, DPB, DTD RKB DKC, DPC, DTE RKC
-13	Output	PPA PPB DKA, DPA RKA DKB, DPB RKB
-12	Listing Output	*TTA LPA VPA PPA, DPA, DTA, MTA RKA DKA, DPB, DTD RKB DKB, DTE
-11	Input	TTA PRA PRB CDB DKA, DPA, DTA, MTA RKA DKB, DPB, DTD RKB DKC, DPC, DTE RKC
-10	Parameter File Input	TTA PRA PRB CDB DKA, DPA, DTA, MTA RKA DKB, DPB, DTD RKB DKC, DPC, DTE RKC
-1	System MACRO File Input	Same as -11 (input)

9.2.3 MAC11

.DAT Slot	Use	Handler
-12	Listing Output	TTA LPA VPA PPA, DPA, DTA, MTA RKA DKA, DPB, DTD RKB DKB, DTE
-11	Input	TTA PRA PRB CDB DKA, DPA, DTA, MTA RKA DKB, DPB, DTD RKB DKC, DPC, DTE RKC

9.2.4 FOCAL

.DAT Slot	Use	Handler
3	Library Input	TTA PRA PRB CDB DKA, DPA, DTA, MTA RKA DKB, DPB, DTC, MTC RKB DKC, DPC, DTD RKC DTE
5[1]	Library Output	TTA PPA DKA, DPA, DTA, MTA RKA DKB, DPB, DTD RKB LPA, DTE VPA
7	Data File Input	TTA PRA PRB CDB DKA, DPA, DTA, MTA RKA DKB, DPB, DTC, MTC RKB DKC, DPC, DTD RKC DKE, DTE
10	Data File Output	TTA PPA DKA, DPA, DTA, MTA RKA DKB, DPB, DTD RKB LPA DTE VPA

[1]Prior to loading FOCAL, this .DAT slot must be assigned to one of the devices listed, if library output is desired.

9.2.5 EDIT, EDITVP, and EDITVT

.DAT Slot	Use	Handler				
-15	Scratch/ Output	TTA VPA LPA PPA DKA, DKB,	 DPA, DPB,	 DTA, DTD DTE	 MTA	 RKA RKB
-14	Input	TTA PRA PRB CDB DKA, DKB, DKC,	 DPA, DPB, DPC,	 DTA, DTD DTE	 MTA	 RKA RKB RKC
-13	Secondary Output	Same as .DAT -15.				
-12	Listing Output	LPA	TTA	PPA PPB PPC	VPA	
-10	Secondary Input	TTA PRA PRB CDB DKA, DKB, DKC,	 DPA, DPB, DPC,	 DTA, DTD DTE	 MTA	 RKA RKB RKC
10	Display Output (EDITVP only)	VPA				

9.2.6 Linking Loader and DDT

.DAT Slot	Use	Handler			
-5	External User Library Input	NON (same as .DAT slot -4 when used for user library)			
-4	User Program Input	PRA DKA, DKB, DKC,	 DPA, DPB, DPC,	 DTA, DTC DTD DTE	 RKA RKB RKC
-1	System Library Input	PRA DKA, DKB, DKC,	 DPA, DPB, DPC,	 DTA DTC DTD DTE	 RKA RKB RKC

9.2.7 PIP (Peripheral Interchange Program)

PIP uses all positive .DAT slots and -2 and -3 for TTY I/O. Prior to use, any non-standard device assignments should be made via the ASSIGN command to the Monitor. If several functions are to be used with a variety of peripherals, assignment of these devices all at the same time avoids the necessity for returning to the Monitor to reassign devices and for repeatedly reloading PIP after each operation that requires a new device.

NOTE

The device handlers used with PIP should normally be those having the greatest capability (i.e., PRA, PPA, DTA, DKA, etc.). If both input and output are to occur on the same device (e.g., DECtape), separate .DAT Slots must be assigned. Both .DAT Slots must be assigned to the same handler.

Positive .DAT Slot assignments for the system, as initially supplied, are as follows:

.DAT Slot	Use	Handler
1	I/O	*DKA or *DPA or *RKA
2	I/O	*DKA or *DPA or *RKA
3	I/O	*DKA or *DPA or *RKA
4	I/O	*TTA
5	Input	*PRA
6	Output	*PPA
7	I/O	*DTA
1Ø	I/O	*DTA
11	I/O	*NON
12	I/O	*NON
13	I/O	*NON
14	I/O	*NON
15	I/O	*NON
16	I/O	*NON
17	I/O	*NON
2Ø	I/O	*NON

9.2.8 SGEN (System Generator)

.DAT Slot	Use	Handler
-14	Input/Output	DKA or DPA or RKA

9.2.9 PATCH

.DAT Slot	Use	Handler
-14	I/O	DKA, DPA, DTA, RKA DTD DTE
-1Ø	Secondary Input	TTA PRA DKA, DPA, DTA, RKA

9.2.10 UPDATE

.DAT Slot	Use	Handler
-15	Output	PPA PPB PPC DKA, DPA, DTA, MTA RKA DKB, DPB RKB
-14	Input	PRA DKA, DPA, DTA, MTA RKA DKB, DPB RKB DKC, DPC RKC
-12	Listing	LPA TTA VPA PPA DKA, DPA, DTA, MTA, RKA DKB, DPB RKB
-1Ø	Secondary Input	TTA PRA DKA, DPA, DTA, MTA, RKA DKB, DPB RKB DKC, DPC RKC

9.2.11 DUMP

.DAT Slot	Use	Handler
-14	Input	DKA, DPA, DTA, MTA, RKA DKB, DPB, DTD RKB DKC, DPC, DTE RKC
-12	Listing	TTA LPA VPA PPA DKA, DPA, DTA, MTA, RKA DKB, DPB, DTD RKB DTE

9.2.12 CHAIN

.DAT Slot	Use	Handler
-5	User Library	*NON (Same as .DAT slot -4 when used)
-4	Input	PRA DKA, DPA, DTA, RKA, MTA DKB, DPB, RKB DKC, DPC, DTC, RKC, MTC DTD DTE Use only if no other DT, DK, or MT is assigned.
-1	System Library	Same as for .DAT -4

NOTE

Use the smallest handlers possible since they are not recoverable as user handlers in the overlay system.

9.2.13 EXECUTE

.DAT Slot	Use	Handler
-4	Load CHAIN-Built Program	RKA, DKA, DPA, DTA, MTA DTD, DTE

9.2.14 SRCCOM

.DAT Slot	Use	Handler
-15	New File Input	TTA (if not assigned to -14) PRA (if not assigned to -14) CDB (if not assigned to -14) *DKA, *DPA, DTA, MTA, *RKA DKB, DPB, DTD, RKB DKC, DPC, DTE RKC
-14	Original File Input	TTA (if not assigned to -15) PRA (if not assigned to -15) CDB (if not assigned to -15) DKA, DPA, DTA, MTA, RKA DKB, DPB, DTD RKB DTE

SRCCOM (Cont.)

.DAT Slot	Use	Handler
-12	Listing	TTA PPA LPA VPA DKA, DPA, DTA, MTA, RKA DKB, DPB, DTD, RKB DTE

9.2.15 8TRAN

.DAT Slot	Use	Handler
-15	Input	PRA CDB TTA DKA, DPA, DTA, MTA RKA DKB, DPB, DTD RKB DKC, DPC, DTE RKC
-14	Output	PPA LPA TTA VPA DKA, DPA, DTA, MTA, RKA DKB, DPB, DTD RKB DTE

9.2.16 MTDUMP

.DAT Slot[1]	Use	Handler
1	Input	MTA MTF
3	Output	VPA, LPA, TTA

[1] Prior to loading this program, the .DAT slots must be reassigned to one of the handlers listed here.

9.2.17 SPOOL[1]

.DAT Slot	Use	Handler
-14	Load SPOOLER	RKA, DPA, DKA

9.2.18 SPLGEN[1]

.DAT Slot	Use	Handler
-14	Temporary File Storage	RKA, DPA, DKA

9.2.19 SPLOAD[1,2]

.DAT Slot	Use	Handler
-14	Install SPOL11 Image	RKA, DPA, DKA
none	SPOL11 Input	Paper tape (no handler)

9.2.20 MCLOAD[1,2]

.DAT Slot	Use	Handler
-14	Install MAC11 Image	RKA, DPA, DKA
none	MAC11 Input	Paper tape (no handler)

[1] UNICHANNEL systems only.

[2] Prior to loading this program, the paper tape output of MAC11 containing the spooler should be readied in the paper tape reader.

9.3 I/O HANDLER DESCRIPTIONS

The following paragraphs describe the operating features of the standard XVM/DOS I/O Device Handlers. The XVM/DOS System Manual describes functions which are internal to the handlers and provide instructions to assist users in creating their own special device handlers. Users having a VT15 Graphics Display System should refer to the VT15 XVM GRAPHICS Software Manual for descriptions of the associated device handlers VTA, LKA, and VWA.

9.3.1 Teleprinter Handler (TTA)

9.3.1.1 General Description - The teleprinter handler is embedded in the Resident Monitor and provides all functions necessary for teleprinter input/output. The handler performs I/O using either IOPS ASCII (Mode 2) or Image Alphanumeric (Mode 3) data. Table 9-2 lists the handler's responses to the various I/O Macros.

Table 9-2

TELEPRINTER I/O FUNCTIONS

Macro	Response
.INIT	Accept
.FSTAT	Ignore
.RENAM	Ignore
.DLETE	Ignore
.RAND	Illegal
.RTRAN	Illegal
.SEEK	Ignore
.ENTER	Ignore
.CLEAR	Ignore
.CLOSE	Accept
.MTAPE	Ignore
.READ	Accept
.WRITE	Accept
.WAIT	Accept
.WAITR	Accept
.TRAN	Illegal

Illegal = Illegal Function (IOPS6)

9.3.1.2 Device Dependent Characteristics - The following paragraphs describe the characteristics which are unique to the teleprinter handler in its response to certain I/O Macros and characters:

a. .INIT 1) Maximum I/O buffer size returned: $42_8(34_{10})$

2) Set up CTRL P restart address from address specified in .INIT argument "restrt". Refer to the XVM/DOS System Manual for setup of the restart address for CTRL C and CTRL T.

3) Output Carriage RETURN/LINE FEED

4) A .INIT issued to the teletype handler will not cancel a .READ/.WRITE which is in progress.

.SCOM+35 contains the instruction "DZM/TTIOSW". If a program desires to abort teletype I/O, it can execute the instruction "XCT* (.SCOM+35)", which will clear out the busy flag.

b. .CLOSE - Output Carriage RETURN/LINE FEED.

c. .WRITE - When in IOPS ASCII Mode a LINE FEED is normally output automatically before the line (logical record) is output unless an Overprint (2Ø) is the first character (see Table 9-3).

d. Non-Printing Function Characters - The non-printing function characters contained in Table 9-3 have special significance when input and output in IOPS ASCII mode.

Table 9-3

SPECIAL NON-PRINTING FUNCTION CHARACTERS FOR IOPS ASCII TELEPRINTER I/O

FUNCTION (ASCII in parentheses)	TRANSFER DIRECTION	ACTION
Carriage RETURN (Ø15)	Input	Insert all characters typed, including this Carriage RETURN, since the last Carriage RETURN or ALT MODE, into the requesting program's I/O Buffer. Echo a LINE FEED on the printer.
	Output	Terminate output of the contents of the requesting program's I/O buffer. Output Carriage RETURN.

Table 9-3 (Cont.)

SPECIAL NON-PRINTING FUNCTION CHARACTERS FOR IOPS ASCII TELEPRINTER I/O

FUNCTION (ASCII in parentheses)	TRANSFER DIRECTION	ACTION
ALT MODE (33, 175, 176)	Input	Terminate the current line and insert all characters typed since the last Carriage RETURN or ALT MODE into the requesting program's I/O buffer. Map into the I/O Buffer as 175.
	Output	Terminate output of the contents of the requesting program's I/O buffer.
LINE FEED (Ø12)	Input	Insert in requesting program's I/O buffer.
	Output	Ignore if this is the first character in the I/O buffer; otherwise, output.
VT (Vertical Tab) (Ø13) or FORM Feed (Ø14)	Input	Insert in requesting program's I/O buffer.
	Output	Model 35 teleprinters output FORM Feed. LA36, Model 33 and LA30 teleprinters ignore.
Horizontal TAB (Ø11)	Input	Insert in requesting program's I/O buffer.
	Output	If Monitor Command TAB OFF is in effect (no TAB simulation). If TAB ON is in effect, output sufficient number of SPACEs (Ø4Ø) to position printer at columns 9, 17, 25,... etc.
Skip One Line (Ø21)	Output Only	If this is the first character in the requesting program's I/O buffer, skip 1 line. Otherwise, ignore.
Overprint (Ø2Ø)	Output Only	If this is the first character in the requesting program's I/O buffer, suppress the LINE FEED normally output. Otherwise, ignore.
RUBOUT (177)	Input	Delete the last character typed previous to this and echo a backslash (\ ).
CTRL U (Ø25)	Input	Delete all characters typed since the last Carriage RETURN or ALT MODE, and echo an "at" sign (@).
	Output	If typed while output is under way, truncate the remainder of the logical record being output.

Table 9-3 (Cont.)

SPECIAL NON-PRINTING FUNCTION CHARACTERS FOR IOPS ASCII TELEPRINTER I/O

FUNCTION (ASCII in parentheses)	TRANSFER DIRECTION	ACTION
Null (ØØØ)	Input/ Output	Ignore
CTRL D (ØØ4)	Input	Transmit to the requesting program's I/O buffer a logical record which consists of a header word pair only, with the I/O Mode Bits (14-17) set to Ø1Ø1 to indicate end-of-file. (See paragraph 6.3.1.1.)

9.3.1.3 Program Control Characters - The teleprinter retains its role as control device during all I/O operations. The CTRL characters (CTRL C, CTRL P, CTRL S, and CTRL T) are recognized when typed, regardless of Data Mode or transfer direction. These characters perform specific system functions as described in 8.1 . Experienced users who wish to alter their meaning may do so using procedures described in the XVM/DOS System Manual.

9.3.2 Paper Tape Punch Handlers (PPA, PPB, and PPC)

9.3.2.1 General Description - There are three paper tape punch handlers: PPA (571_8 registers), PPB (416_8 registers), and PPC (322_8 registers). Sizes listed are approximate. All three handlers respond identically to the I/O Macros, but differ as to the various data modes which are acceptable to them. Table 9-4 lists the data modes which are acceptable and Table 9-5 shows the handlers' responses to the I/O Macros.

Table 9-4

PAPER TAPE PUNCH DATA MODES

Data Mode	Handler		
	PPA	PPB	PPC
IOPS ASCII	X	-	-
IOPS Binary	X	X	X
Image Alphanumeric	X	X	-
Image Binary	X	X	-
Dump	X	X	-

Table 9-5

PAPER TAPE PUNCH I/O FUNCTIONS

Macro	Response
.INIT	Accept
.FSTAT	Ignore
.RENAM	Ignore
.DLETE	Ignore
.RAND	Illegal
.RTRAN	Illegal
.SEEK	Illegal
.ENTER	Ignore
.CLEAR	Ignore
.CLOSE	Accept
.MTAPE	Ignore
.READ	Illegal
.WRITE	Accept
.WAIT	Accept
.WAITR	Accept
.TRAN	Illegal

Illegal = Illegal Function (IOPS6 Error)

9.3.2.2 Device Dependent Characteristics - The following paragraphs describe the characteristics which are unique to the Paper Tape Punch Handlers in their response to certain I/O Macros and characters.

a. .INIT - 1) Maximum I/O buffer size returned: 64_8 (52_{10}).

2) Punches two fanfolds of leader.

b. .CLOSE- 1) Output EOF (end-of-file) header word pair (see 6.3.1.1) as last record on the tape (for IOPS Binary mode Ø, only).

c. Special Characters - The characters listed in Table 9-6 have special significance to the handlers when output in IOPS ASCII Mode. These characters, except TAB, are ignored if they do not appear as the first character in a logical record (line). If a LINE FEED, VT (Vertical TAB), or FORM Feed does not appear as the first character in a logical record, a LINE FEED is supplied by the handlers.

Table 9-6

SPECIAL FUNCTION CHARACTERS FOR FIRST CHARACTER IN LINE

FUNCTION	ACTION[1]
LINE FEED (Ø12)	Output.
VT (Vertical TAB) (Ø13)	Output, followed by four RUBOUTs[1] (177).
FORM FEED (Ø14)	Output, followed by 40 Nulls[1] (ØØØ).
Horizontal TAB (Ø11)	Output, followed by one RUBOUT[1] (177).

9.3.3 Paper Tape Reader Handlers (PRA and PRB)

There are two paper tape handlers: PRA (673_8 registers) and PRB (446_8 registers). Sizes shown are approximate. Both handlers respond identically to the I/O Macros, but differ as to the data modes which are acceptable. Table 9-7 lists the data modes which are acceptable, and Table 9-8 shows the handlers' response to the I/O Macros.

NOTE

PRA and PRB ignore any nulls or rubouts encountered in IOPS ASCII mode. PRA ignores any nulls encountered in IOPS BINARY mode.

Table 9-7

PAPER TAPE READER DATA MODES

Data Mode	Handler	
	PRA	PRB
IOPS ASCII	X	X
IOPS Binary	X	-
Image Alphanumeric	X	-
Image Binary	X	-
Dump	X	-

[1]The RUBOUT and NULL functions which follow output of the desired characters are used for hardware timing purposes when the paper tape is to be transmitted to a printer in an off-line environment.

Table 9-8

PAPER TAPE READER I/O FUNCTION

Macro	Response
.INIT	Accept[1]
.FSTAT	Ignore
.RENAM	Ignore
.DLETE	Ignore
.RAND	Illegal
.RTRAN	Illegal
.SEEK	Ignore
.ENTER	Illegal
.CLEAR	Illegal
.CLOSE	Accept
.MTAPE	Ignore
.READ	Accept
.WRITE	Illegal
.WAIT	Accept
.WAITR	Accept
.TRAN	Illegal

Illegal = Illegal Function (IOPS6 Error)

9.3.4 DECtape Handlers (DTA, DTC, DTD, DTE, and DTF)

9.3.4.1 General Description - There are five DECtape handlers for TU55/TU56 DECtape operation. Sizes shown are approximate.

- DTA ($3ØØØ_8$ locations) is the most general DECtape handler provided. It has a simultaneous three-file capacity, either input or output. Input files can be referenced on the same or different DECtape units; however, not more than one output file can exist on the same unit. (I.e., file creation on the same unit must occur sequentially.)
- DTC (1264_8 locations) is the most limited and also the most conservative of core (for IOPS data mode operations) of the handlers. It is an input only handler with a one-file capacity.
- DTD (3071_8 locations) provides single file operation, either input or output.
- DTE (2674_8 locations) is similar to DTD, differing only in its I/O function capabilities as shown in Table 9-10.

[1]Maximum I/O buffer size returned: 64_8 (52_{10}).

- DTF (1151_8 locations) is a handler which simulates the non-directoried, sequential access file structure of Magtape. It accommodates (serially) up to eight DECtape units, both input and output. When the last block of a tape on a particular unit has been accessed, DTF causes the Monitor to output an IOPS4 message (Device Not Ready) to permit the operator to remove the current tape from the DECtape drive and mount another. The operator can then type a CTRL R to continue processing.

Table 9-9 illustrates the data modes which are acceptable to these handlers, while Table 9-10 shows the handlers' responses to the various I/O Macros.

Table 9-9

DECTAPE DATA MODES

Data Mode	Handler				
	DTA	DTC	DTD	DTE	DTF
IOPS ASCII	X	X	X	X	X
IOPS Binary	X	X	X	X	X
Image Alphanumeric	X	-	X	X	-
Image Binary	X	-	X	X	-
Dump	X	-	X	X	-

9.3.4.2 Device Dependent Characteristics - The following characteristics are unique to the DECtape Handlers in their responses to certain I/O Macros.

a. .INIT - Maximum I/O buffer size returned: 377_8 (255_{10}).

b. .MTAPE 1) DTD accepts REWIND and BACKSPACE RECORD subfunctions only.

2) DTF accepts REWIND, BACKSPACE RECORD and SKIP RECORD subfunctions only.

Table 9-10

DECTAPE I/O FUNCTIONS

Macro	Handler				
	DTA	DTC	DTD	DTE	DTF
.INIT	Accept	Accept	Accept	Accept	Accept
.FSTAT	Accept	Accept	Accept	Accept	Accept
.RENAM	Accept	Illegal	Accept	Accept	Ignore
.DLETE	Accept	Illegal	Accept	Accept	Ignore
.RAND	Illegal	Illegal	Illegal	Illegal	Illegal
.RTRAN	Illegal	Illegal	Illegal	Illegal	Illegal
.SEEK	Accept	Accept	Accept	Accept	Ignore
.ENTER	Accept	Illegal	Accept	Accept	Ignore
.CLEAR	Accept	Illegal	Accept	Accept	Illegal
.CLOSE	Accept	Accept	Accept	Accept	Accept
.MTAPE	Illegal	Illegal	Accept	Illegal	Accept
.READ	Accept	Accept	Accept	Accept	Accept
.WRITE	Accept	Illegal	Accept	Accept	Accept
.WAIT	Accept	Accept	Accept	Accept	Accept
.WAITR	Accept	Accept	Accept	Accept	Accept
.TRAN	Accept	Illegal	Accept	Accept	Illegal

9.3.5 DECdisk, Disk Cartridge and Disk Pack Handlers (DKA/RKA/DPA. DKB/RKB/DPB, and DKC/RKC/DPC)

9.3.5.1 General Description - Three handlers are provided for RF15 DECdisk, RKØ5 Disk Cartridge and RPØ2 Disk Pack operations. Version for version, these handlers are identical in their functions with these two exceptions:

a. All disks DECdisk, Disk Cartridge and Disk Pack are block addressable for direct access operations (.TRAN and .RTRAN Macros). Also, the DECdisk is word addressable.

b. The Disk Pack and Disk Cartridge have a unit structure while the DECdisk does not. This means that each DECdisk is treated as a single addressable unit regardless of the actual number of platters incorporated (up to 8).

All versions of these handlers support the disk file structures described in Chapter 3. There is no fixed limit to the number of input or output files which can be simultaneously accessed, except as determined by the amount of available buffers. To this end, the handlers perform dynamic buffer allocation from the Monitor's buffer pool, using the .GTBUF and .GVBUF Monitor Macros described in Chapter 5. At run time, the operator need only be concerned that the number of files concurrently accessed is not greater than the number of buffers allocated by the BUFFS Keyboard command (see 8.6.1).

The following commands obtain buffers from the pool, and return them immediately upon completion of the operation:

.DLETE
.RENAM
.CLEAR

The following commands obtain a buffer from the pool and do not return it until a subsequent .CLOSE, .INIT or Rewind (.MTAPE) is performed:

.FSTAT
.ENTER
.SEEK
.RAND

The following commands return a buffer to the pool, if any were taken:

.INIT
.CLOSE
.MTAPE (Rewind subfunction)

The handlers operate in all data modes (i.e., IOPS, Image, and Dump). Table 9-11 lists the I/O Macros which are acceptable to the various handler versions.

Table 9-11

DECDISK, DISK CARTRIDGE AND DISK PACK I/O FUNCTIONS

Macro	DKA $(4343)_8$ DPA $(4642)_8$ RKA $(4517)_8$	Handler[1] DKB $(3677)_8$ DPB $(3677)_8$ RKB $(3775)_8$	read only DKC $(2172)_8$ DPC $(2316)_8$ RKC $(2273)_8$
.INIT	Accept	Accept	Accept
.FSTAT		Accept	Accept
.DLETE			
.RENAM			
.RAND		Illegal	Illegal
.RTRAN			
.SEEK		Accept	Accept
.ENTER		Accept	Illegal
.CLEAR		Illegal	Illegal
.CLOSE		Accept	Accept
.MTAPE		Illegal	Illegal
.READ			Accept
.WRITE		Accept	Illegal
.WAIT			Accept
.WAITR			Accept
.TRAN		Illegal	Illegal

9.3.5.2 Device Dependent Characteristics - The following characteristics are unique to these handlers in responding to certain I/O Macros:

a. .INIT

1) Maximum I/O buffer size returned: 376_8 (254_{10}).

2) The disk handlers allow write verification on output files. If the file is defined as an output file, the user has the opportunity of guaranteeing the integrity of his data by using 11_8 as the "dd" argument to the .INIT macro. The handler will then check every block of data it writes out, to ensure that the transfer occurred without error. A second .INIT command must be executed to remove or add the write checking feature. (i.e., repeated 11_8 argument causes write check switch 1 to be turned ON/OFF/ON/OFF/ON..)

3) An output file already opened on a .DAT slot referenced by .INIT will be deleted. If a .INIT references a .DAT slot with an opened input file, the handlers will close it, and give back the buffer it was using.

4) Control is retained until all necessary I/O is complete.

5) The .INIT macro uses the relationship between the User File Directory Table and the Device Assignment Table to get the correct UIC from the User File Directory Table. If the user is changing UIC's under program control (via .USER macros), the operation must be accomplished before a .INIT in order to obtain the desired UFD.

[1]Sizes listed are approximate.

b. .DLETE

1) Control is not returned until all necessary Disk I/O is complete. If the UIC associated with this .DAT slot does not exist in the MFD, or if the named file can not be found, the disk handlers ignore this macro. If the name is found, the handlers return the first block number of the file in the AC. The buffer used by the handlers to delete the named file is given back to the buffer pool upon completion of the .DLETE.

2) The .DLETE macro follows the protection rules for directory modification. That is, .DLETE will not work on a protected directory, but returns an IOPS 63 error.

c. .RENAM

1) The first block number of the renamed file is returned in the AC after a successful operation. If the file or UFD does not exist, the handlers return to LOC+3 with Ø in the AC. .RENAM changes the renamed file's date to the current date (maintained by the DATE keyboard command described in Chapter 8).

2) At completion of the .RENAM function, the handlers return the buffer to the buffer pool.

3) The .RENAM macro follows the protection rules for directory modification. That is, .RENAM will not work on a protected directory (IOPS 63).

d. .ENTER

1) The handlers check for directory protection. If any of the following conditions is satisfied, the handlers will allow successful operation. If none is satisfied, the handlers will terminate with an IOPS 63.

Conditions for gaining UFD access:

Entry in .UFDT equals the logged-in UIC

Logged-in UIC equals MIC

Directory protection code equals Ø

2) Once the entry in the UFD has been made via the .ENTER, the file is defined as being opened and truncated. Upon a subsequent .CLOSE, the file will exist as a closed file, but not truncated.

3) When a .ENTER is done with a file name that already exists, the old file is deleted only after the new file (just .ENTERed) is .CLOSEd, if the old file is not truncated. If the old file is a truncated file, it is deleted immediately, before the new file is listed in the UFD. The process of deletion of identically named truncated files continues until a non-truncated file with the same name is found. At this point, the new directory entry is made. Truncated files which follow are not deleted. In all cases, UFD searches are sequential starting at the beginning.

e. .CLEAR

1) The disk handlers will not honor .CLEAR unless the user has logged in under the MIC. The .CLEAR function deletes all files and directories on the entire disk. All bit maps are closed and indicate only the space which they occupy. The MFD will have no UFD's, SYSBLK or BAT. An I/O buffer is obtained from the pool for this operation and is subsequently returned on its completion.

f. .CLOSE

1) On input, the handlers give the buffer back (if one was acquired) and make the .DAT slot available for subsequent .INITs. On output, the handlers write an end-of-file record (if the user did not already write one), and then proceed as on input.

g. .READ

1) All .READ commands executed after an end-of-file (EOF) header has been reached will return an EOF in header word Ø (ØØ1ØØ5).

h. .MTAPE

1) DKA, RKA and DPA accept the REWIND and BACKSPACE subfunctions during input only. REWIND is effectively a .CLOSE.

i. .TRAN

1) The .TRAN is not included as part of the disk file structure. That is, all blocks read or written are done so at the user's discretion. MFD's, UFD's, Bit Maps, and RIB's are not considered, and are not protected from the .TRAN macro. The .TRAN macro is allowed to any .DAT slot that has been .INITed, and not .CLOSEd or rewound (via a .MTAPE).

2) For the RF DECdisk, the user can reference a specific platter just by identifying the block number he wants. The block numbers and platter relationships are shown below:

Platter Number	Block Number
Ø	Ø-1777
1	2ØØØ-3777
2	4ØØØ-5777
3	6ØØØ-7777
4	1ØØØØ-11777
5	12ØØØ-13777
6	14ØØØ-15777
7	16ØØØ-17777

j. .FSTAT

1) .FSTAT functions normally, except that a subsequent .SEEK to a file found via .FSTAT will not require redundant disk access. That is, both .FSTAT and .SEEK ordinarily require a minimum of three disk accesses -- one to the MFD, one to the UFD, and one to the file. If the user does a .FSTAT to an existing file, and then a .SEEK, the disk handlers "remember" the successful .FSTAT, and do not do an extra disk access.

k. .RAND

1) .RAND commands to a nonexistent file cause an IOPS13. Those to a nonexistent UFD cause an IOPS51. Those to an empty UFD cause an IOPS71.

l. .RTRAN

1) The disk pack and disk cartridge handlers ignore the word number argument (assumed to be Ø) and return the whole block. If the word number plus the word count exceeds 254_{10}, the disk handlers will return IOPS67.

Output .RTRAN to the RP or RK disk requires 256_{10} -word buffers to allow the handlers to supply the correct links in the last two words. (Otherwise, random files would require two buffers from the pool.)

If the block number argument requested by the .RTRAN is less than one, or greater than the number of blocks in the file, an IOPS66 will result.

9.3.6 Magtape Handlers (MTA, MTC, and MTF)

9.3.6.1 General Description - Three handlers are provided for operation of Magtape drives TU10, TU20A, TU20B, TU30A, and TU30B. These handlers permit control of up to eight transports. The sizes shown are approximate.

- MTA (4703_8 locations) is the most general and permits DECtape file structuring using .SEEK and .ENTER Macros (refer to Chapter 4). Up to three files can be concurrently referenced, each on a different transport, either input or output.

- MTC (1253_8 locations) is a read-only handler designed for operation using DECtape file structuring only. It has a single file capacity; sequential file references are, of course, allowed.

- MTF (1312_8 locations) is designed for Magtape file structuring only. It accommodates up to eight concurrently referenced transports, both input and output.

The track count (either 7- or 9-channel) can be set at System Generation, or by using the CHANNEL Keyboard Command. In addition, it can also be set dynamically, along with parity and recording density parameters, using the .MTAPE I/O Macro (see paragraph 6.7.7) when using Magtape file structuring. When using DECtape file structuring, parity and density are fixed at odd parity and 800 BPI recording density.

Table 9-12 lists the Data Modes acceptable to the handlers and Table 9-13 lists the I/O Macros and their responses.

Table 9-12

MAGTAPE DATA MODES

Data Modes	Handler		
	MTA	MTC	MTF
IOPS ASCII	X	X	X
IOPS Binary	X	X	X
Image Alphanumeric	X	-	-
Image Binary	X	-	-
Dump	X	-	-

Table 9-13

MAGTAPE I/O FUNCTIONS

Macro	Handler		
	MTA	MTC	MTF
.INIT	Accept	Accept	Accept
.FSTAT			
.RENAM		Illegal	
.DLETE			
.RAND	Illegal		Illegal
.RTRAN	Illegal		
.SEEK		Accept	
.ENTER		Illegal	
.CLEAR		Illegal	
.CLOSE		Accept	Accept
.MTAPE	Accept	Illegal	
.READ		Accept	
.WRITE		Illegal	
.WAIT		Accept	
.WAITR		Accept	
.TRAN		Illegal	

Illegal = Illegal Function (IOPS6 Error)

9.3.6.2 Device Dependent Characteristics - The first .INIT to a Magtape unit causes the system default parameters for track count, recording density, and parity to be assigned as follows:

Parity odd - Density 800 BPI
Track Count - System Generated Default unless otherwise specified in the CHANNEL Keyboard Command. (See Paragraph 8.9.3.)

The following characteristic is unique to the operation of the A and C handlers in responding to the I/O Macros below:

a. .INIT - Maximum I/O buffer size returned: 376_8 (255_{10}).

The following describes characteristics which are unique to the operation of the MTF handler.

CHANGING THE BUFFER SIZE

The user may dynamically change the standard buffer size (377_8 returned on .INIT) with the following instructions:

```
.GLOBL MTBSIZ
  :
  :
LAC    BUFSIZ  /(Any desired buffer size)
DAC*   MTBSIZ
  :
  :
WRITE
  :
.READ
```

Thus, since FIOPS re.INITs a handler every time I/O transfer direction is changed, one can input from Magtape drive N with one record size and output to drive M with another record size, bearing in mind that MTBSIZ must be altered before transfer direction is changed. Record size must be less than or equal to 256_{10} words for FORTRAN .READs and .WRITEs. Larger record sizes are possible via MACRO routines.

.INIT

1. Return standard buffer size of 377_8 words. Buffer size can be changed by modifying the .GLOBL MTBSIZ through a call to a MACRO subroutine.

2. Call .SETUP - API channel register 45_8.
3. Set up transfer direction (input or output).
4. If first .INIT to this device, the assigned default values will be odd parity and 8ØØ BPI. Track count will be specified by bit 6 of the SCOM+4 register. (Ø=7 channel, 1+9 channel.)
5. Update the reference drive table to indicate this unit is open for I/O transfers.

.OPER - .FSTAT is allowed and implemented. AC=Ø on return.

.SEEK - Undefined. Error return IOPS6.

.ENTER - Undefined. Error return IOPS6.

.CLEAR - Undefined. Error return IOPS6.

.CLOSE

1. Checks transfer direction. If not output, an error return IOPS6 is issued.
2. Writes an end-of-file mark and returns to caller.

.MTAPE - Honors the subfunction specification as follows:

ØØ - Rewind: A rewind is issued to the specified drive and control is returned to the caller.

Ø1 - Undefined: Error return IOPS6.

Ø2 - Backspace: A backspace is issued to backspace one physical record.

Ø3 - Backspace file: Issue backspace until two end-of-file marks have been passed over, then space forward over the last end-of-file mark. If beginning of tape is sensed during the backspacing, the function is terminated and control is returned to the caller.

Ø4 - Write EOF: Issue a write end-of-file mark.

Ø5 - Forward space: A space forward is issued to forward space one physical record. If the end-of-tape is sensed, an error return IOPS65 is issued.

Ø6 - Forward space file: A space forward is issued to forward space until an end-of-file is sensed. If the end-of-tape is sensed, an IOPS65 is issued.

Ø7 - Space to logical EOT: A space forward is issued until two consecutive end-of-file marks are passed, then a backspace one physical record is issued. If the end-of-tape is sensed, an IOPS65 is issued.

10 17 - Describe the tape configuration. The reference drive table will be updated for the unit referenced. Subsequent I/O transfers will be performed in the density, parity, and channel count given below:

Subfunction	Channel Count	Parity	Density
1Ø	7	Even	2ØØ BPI
11	7	Even	556 BPI
12	7	Even	8ØØ BPI
13	9	Even	8ØØ BPI
14	7	Odd	2ØØ BPI
15	7	Odd	556 BPI
16	7	Odd	8ØØ BPI
17	9	Odd	8ØØ BPI

.READ

1. Check if the referenced unit is allowed to input. Error IOPS6 is issued if not.
2. Check if the data mode is legal: mode Ø or 2. Error return IOPS7 is issued if the data mode is illegal.
3. Initiate the data transfer.
4. Read errors:

   a. Parity, checksum, and record length incorrect. The appropriate header bits are set and returned to the data buffer.

   b. Bad tape or data late. These are considered unrecoverable tape errors, and an error return IOPS65 is issued.

   c. End-of-file. The appropriate header bits are set to a 5 and returned to the data buffer.

   d. End-of-tape. The appropriate header bits are set to a 6 and returned to the data buffer.

   e. Any Real Error (parity, data late, or bad tape) is given 25 rereads before error action is taken.

.WRITE

1. Check if referenced unit is allowed to output. Error return IOPS6 is issued if not.
2. Check if the data mode is legal: mode Ø or 2. Error return IOPS7 is issued if the mode is illegal.
3. Initiate the data transfer.
4. Write errors:

a. Any write error is given 25 rewrites. If unsuccessful, six inches of tape are skipped and the record is rewritten.

b. End-of-tape. The appropriate header bits are set to 6 and returned to the data buffer.

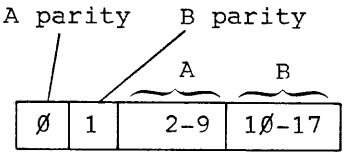
.WAIT, .WAITR - Check I/O underway.

1. Busy: Return to CAL or to address in CAL+2 (.WAITR).
2. Nonbusy: Return to CAL+2 or to CAL+3 (.WAITR).
3. After a .WAIT or a nonbusy .WAITR, the accumulator contains the contents of the TC59 magnetic tape status register as it appeared on completion of the previous I/O operation.

.TRAN

Allows standard core dump mode (both 7 and 9-track look like 7-track), and true 9-track mode. Bit 9 of LOC+Ø=1 yields true 9-track.

Nine-track data arrives in core with parity set in bits Ø and 1, and with two 8-bit bytes in the low-order 16 bits, as follows:



LEGAL DATA MODES

Output - IOPS BINARY (Mode Ø) and IOPS ASCII (Mode 2) - The checksum is completed and stored in the second word of the line in the IOPS buffer area. Bits 12-13 of the first header word are set to zero. The count of words to write is taken from the word pair count in the header and transfer from the IOPS area is initiated.

Input - IOPS BINARY (Mode Ø) and IOPS ASCII (Mode 2) - The count of words to transfer is taken from the CAL sequence and input is initiated from the next physical block on the tape directly into the IOPS buffer area. When the read is complete, the line validity bits are modified under the following conditions: bits 12-13 of header word Ø are set if buffer overflow occurred. A checksum is calculated and compared with the checksum read. If they differ, bits 12-13 are set to 1Ø. Finally, a check is made to assure that the line was transferred without hardware-detected error. If an error occurred, bits 12-13 are set to Ø1.

RECOVERABLE ERRORS

Transport not ready. IOPS4

1. Results from:
   a. Write request with write lock.
   b. Nine-channel I/O request to a 7-channel transport, and vice versa.
   c. Transport off line or otherwise not ready.

2. Remedy:
   a. Ready the device.
   b. Type control R on the teletype.

UNRECOVERABLE ERRORS

1. Illegal function. IOPS6.
2. Illegal data mode. IOPS7.
3. End-of-tape encountered on a file spacing command. IOPS44.
4. Unrecoverable Magtape error (data late or bad tape). IOPS65.

9.3.7 Line Printer Handler (LPA)

9.3.7.1 General Description - LPA (533_8) locations for the LP15 version and 657_8 locations for the LP11 version[1] are designed to operate the 80-column and 132-column LP15 and LP11 Line Printers respectively. The sizes shown are approximate. The handlers accept data in either IOPS ASCII or Image Alphanumeric data modes. Table 9-14 lists the various I/O Macros and the handlers' response to them.

Table 9-14

RESPONSES TO LINE PRINTER I/O FUNCTIONS

Accept	Illegal (IOPS 6)	Ignore
.INIT	.RAND	.FSTAT
.CLOSE	.RTRAN	.RENAM
.WRITE	.SEEK	.DLETE
.WAIT	.READ	.ENTER
.WAITR	.TRAN	.CLEAR
		.MTAPE

[1]The LP11 version also supports the LS11 Line Printer and the LV11 (as a printer and only when connected to the LP11 printer interrupt vector).

9.3.7.2 Device Dependent Characteristics - The following paragraphs describe characteristics which are unique to the Line Printer Handler in its response to certain I/O Macros and characters.

a. .INIT - 1) Maximum I/O buffer size returned: 70_8 (56_{10}) for 132 column printers; 44_8 (36_{10}) for 80 column printers.

2) Output FORM Feed.

3) Test Bit 6 of the .INIT CAL (see 6.7.6). If this bit is zero, the handler will perform automatic paging by outputting a form feed after every 57_{10} lines. (This bit is set by using a 5 rather than a 1 in the "dd" argument of the .INIT.)

b. .CLOSE - Output a FORM Feed (if not inhibited in the .INIT).

c. .WRITE - 1) Examine header word Ø in the user's I/O buffer as follows:

Bit	Meaning
Ø	Ø = Enter Single Line Mode 1 = Enter Multiple Line Mode
1-8	Contains Line Count for Multiple Line Mode
14-17	Data Mode 2 = IOPS ASCII 3 = Image Alphanumeric

2) Check the first character of the user's I/O buffer for the following vertical form control characters, all of which are output by the FORTRAN IV Object Time System:

Ø14 FORM Feed
Ø2Ø Overprint
Ø21 Print every second line
Ø12 Line Feed

To effect the Overprint function for FORTRAN users, it is necessary to simulate certain vertical form control characters. If the first character of a line is Ø12, Ø14, or Ø21, the handler automatically enters Multiple Line Mode (by setting bit Ø of the first word in the user's I/O buffer to 1) and prints two lines, the first line being the vertical control character, and the second line being the actual data. If the first character is Ø2Ø (overprint), it is replaced in the user's I/O buffer by Ø15 (Carriage Return) which does not affect the page position and both lines are printed. All other characters cause a Line Feed to be output from the handler's internal buffer followed by the line from the user's buffer. After output, any data in the user's I/O buffer which was changed (i.e., header word Ø or the first data word) is restored.

If the user intends to output to another device from the same I/O buffer (e.g., two sequential .WRITEs), a .WAIT should be used after the .WRITE referencing the Line Printer to permit the restoration of any data which may have been replaced in the user's I/O buffer by LPA.

3) Output in either Single Line Mode or Multiple Line Mode as applicable.

4) Restore modified portions of the user's I/O buffer (if changed).

d. Carriage Control Characters - The control characters in Table 9-15, except horizontal TAB, cause line termination, in both IOPS and Image Modes, except for special cases described under .WRITE above.

Table 9-15

LINE PRINTER CARRIAGE CONTROL CHARACTERS

Character	Action
Line Feed (Ø12)	Space one line
VT (Vertical Tab) (Ø13)	Space 20 lines
Form Feed (Ø14)	Move to top of form
Carriage Return (Ø15)	Reset column count to zero (no implicit LINE FEED function)
DLE (Ø2Ø)	Overpoint
DC1 (Ø21)	Space 2 lines
DC2 (Ø22)	Space 3 lines
DC3 (Ø23)	Space 1 line
DC4 (Ø24)	Space 10 lines
ALT MODE (175)	Reset column count to zero (no implicit LINE FEED function)
Horizontal Tab (Ø11)	Output sufficient number of spaces to position printer at column 9, 17, 25, ...,etc. This is not a line terminator and may occur anywhere in the line.

Refer to Appendix A for alternate designations

NOTE[1]

The user is provided with the facility of halting the output on the LP11/LS11 Line Printer (in a UNICHANNEL-15 system) at the end of the current file (CLOSE causes the end of file condition) in the spooled mode. This permits the user to perform operations like tearing off the output listing, changing paper, etc. This is enabled through console switch 1 on the PDP-11. The user requests the PIREX system to halt output on the line printer

[1]For UC15 or UC15 Option Systems only.

at the end of the current file by setting the PDP-11 switch in the 'UP' or '1' position. The system periodically monitors this switch. When PDP-11 switch 1 is in the 'DOWN' or '0' position, the system resumes output to the line printer shortly thereafter.

9.3.8 Card Reader Handler (CDB)

9.3.8.1 General Description - CDB is designed to operate the CR03B (770_8 locations), CR15 (1000_8 locations) and CR11 (613_8 locations) card readers respectively. Sizes shown are approximate. The handler transmits data in IOPS ASCII mode only. As initially supplied, it interprets Hollerith code as punched in DEC029 Card Code. CDB is also supplied in source form which can be assembled to produce a version of the handler which interprets Hollerith punched in DEC026 Card Code[1]. Appendix F contains a table of 029 and 026 Hollerith Codes and the corresponding IOPS ASCII codes. Table 9-16 lists the handler's response to the various I/O macros.

Table 9-16

CARD READER I/O FUNCTIONS

MAcro	Response
.INIT	Accept
.FSTAT	Ignore
.RENAM	Ignore
.DLETE	Ignore
.RAND	Illegal
.RTRAN	ILlegal
.SEEK	Ignore
.ENTER	Illegal
.CLEAR	Illegal
.CLOSE	Accept
.MTAPE	Ignore
.READ	Accept
.WRITE	Illegal
.WAIT	Accept
.WAITR	Accept
.TRAN	Illegal

Illegal = Illegal Function (IOPS6 Error)

[1]Refer to the XVM/DOS V1A System Installation Guide for procedures for assembling and installing CDB.

9.3.8.2 Device Dependent Characteristics - The following paragraphs describe the characteristics which are unique to the Card Reader Handler in its response to certain I/O Macros.

a. .INIT - Macimum I/O buffer size returned: 44_8 (36_{10})

b. .READ - Eighty card columns are read and interpreted as 029 or 026 Hollerith data, mapped into the corresponding 64-graphic subset of ASCII, and stored in the user's I/O buffer in 5/7 format (36_{10} locations are required to store an 80 column card). Compression of internal blanks to tabs and truncation of trailing blanks is not performed (all 80 characters appearing on the card are delivered to the user's buffer). In addition, a Carriage RETURN (Ø15) character is appended to the input line; thus, a total of 81 characters are returned to the user.

c. Illegal punch configurations - all illegal punch configurations (i.e., those not appearing in the 029 or 026 character set shown in Appendix F) are interpreted as validity errors and will cause an IOPS 4 error condition. The card containing the error must be repunched.

d. Special Codes - In addition to the Hollerith character set, the handler recognizes the ALT MODE terminator which is necessary for some system programs. ALT MODE, recognized as a 12-8-1 code (multiple-punched A8) is mapped into the standard ALT MODE character (175_8) in the user's buffer.

   Each file must be terminated with an EOF card punched in either of two ways: a) multiple punch the characters +-0123456789 which punches all positions in card column 1, or b) multiple punch characters AØ- which produces a 12-11-Ø-1 punch in card column 1.

9.3.9 VP15A Storage Tube Display (VPA)

VPA operates the VP15A Storage Tube Display. It accepts data in IOPS ASCII, Image Alphanumeric and Dump Modes. In IOPS ASCII and Image Alphanumeric Modes, up to 70_8 (56_{10}) 72-character lines can be displayed. In Dump Mode, the handler interprets each 18-bit word as a coordinate for point plotting operations. The binaries of VPA.S and VPA. are produced from the same source: VPA. xxx, where xxx is the current edit number. Two versions of VPA are provided. The version

called VPA. (1260_8 locations) automatically erases the screen when an attempt is made to display the 57th ASCII line and that line is placed at the top of the screen. The other version of VPA, which has a file name VPA.S, is slightly larger (1266_8 locations). Sizes shown are approximate. It permits the user to operate the display in a "paging" mode by setting console data switch Ø (when set to 1) to inhibit further output once the 56th ASCII line is displayed. Operation is resumed by pressing the ERASE pushbutton. Paging is stopped by resetting console data switch Ø to Ø[1].

Table 9-17 lists the handler's response to the various I/O Macros. The user should refer to the VP15A XVM Graphics Software Manual for a detailed description of the handler's capabilities and programming considerations.

Table 9-17

VP15A DISPLAY I/O FUNCTIONS

Macro	Response
.INIT	Accept
.FSTAT	Ignore
.RENAM	Ignore
.DLETE	Ignore
.RAND	Illegal
.RTRAN	Illegal
.SEEK	Illegal
.ENTER	Ignore
.CLEAR	Ignore
.CLOSE	Accept
.MTAPE	Ignore
.READ	Illegal
.WRITE	Accept
.WAIT	Accept
.WAITR	Accept
.TRAN	Illegal

Illegal = Illegal Function (IOPS6 Error)

[1]Refer to the XVM/DOS V1A System Installation Guide for procedures for installing the VPA.S or VPA.

9.3.10 UC15 XY11/XY311 PLOTTER (XYA)

XYA (approximately 1145_8 locations) drives a pen-type plotter interfaced onto the PDP-11 in UNICHANNEL-15 systems. The legal data modes are IOPS ASCII and IOPS BINARY.

IOPS ASCII is purely for character plotting. The characters have the size and orientation specified by the last "set character attribute" call (mode=10). The default setting calls for characters to be drawn each in 1/5" square box, in a horizontal direction. The box includes the intercharacter spacing, so that adjacent boxes touch. Horizontal is the length of the plotter paper roll, with the feed roll the direction of positive X. The ASCII characters of value 40-137 octal are plotted. Those below 40 (blank) are replaced by blank; those 140-177 are mapped back to 100-137. Thus, a lower case "a" is plotted as an upper case "A". Note that this means that carriage return, line feed, etc., have no meaning on the plotter. IOPS ASCII output can be obtained, for instance, from FORTRAN FORMAT statements.

IOPS BINARY writes are used to output data and control information to the plotter. The format of integer data in the buffer is a mode variable followed by several optional variables. The variables are integer. Co-ordinate addresses are in plotter steps (1/100").

MODE	FUNCTION	ADDITIONAL VARIABLES[1]
0	pick up the pen	none
1	put down the pen	none
2	pen up, move to absolute position	IX, IY [,IX,IY,...][2]
3	pen down, move to absolute position	IX, IY [,IX,IY,...][2]
4	pen up, move relative to present position	IX, IY [,IX,IY,...][2]
5	pen down, move relative to present position	IX, IY [,IX,IY,...][2]
6	draw characters from array	ICNT, ARRAY
7	present pen position defined as co-ordinate	IX, IY [,IX,IY,...][2]
8	move to absolute position (no pen change)	IX, IY [,IX,IY,...][2]
9	move relative to present position (no pen change)	IX, IY
10	set character	IX, IY, ISIN, ICOS
11	set XY311 pen (1, 2, or 3)	IPEN

[1] The mode and additional variables are simply the data referenced by a .WRITE MACRO, or a FORTRAN WRITE operation.

[2] Up to 126_{10} vector co-ordinate pairs may be specified on a single output request.

where IX is the horizontal size of the box surrounding each character, in plotter steps, and IY is the vertical size. ISIN and ICOS are integers to describe rotation of the character string counter-clockwise from the horizontal position. ISIN is the sin of the angle times 65536_{10}, and ICOS is the cosine times 65536_{10}.

Interface routines may easily be written in FORTRAN to emulate a given plotter package. A CALL with certain arguments places those arguments in a .WRITE to the plotter. Scaling of inches to plotter steps should also be done here, if necessary.

UC15 CALCOMP PLOTTER I/O FUNCTIONS

MACRO	RESPONSE
.INIT	Accept
.CLEAR	Ignore
.CLOSE	Accept
.DLETE	Ignore
.ENTER	Ignore
.FSTAT	Ignore
.MTAPE	Illegal
.RAND	Ignore
.READ	Accept
.RENAM	Ignore
.RTRAN	Ignore
.SEEK	Illegal
.TRAN	Illegal
.WAIT	Accept
.WAITR	Accept
.WRITE	Accept

Illegal results in IOPS 6 error message

A .INIT (issued by FORTRAN) lifts the pen, sets the default character attributes, and set present X, Y to '0,0'. .INIT returns buffer size of 400_8. The maximum character string accepted is 204_8 characters. Buffers larger than this are ignored with no error message.

A .READ returns in IOPS BINARY seven words (eight for XY311) of handler status. (Note, in the case of spooling, this status may not yet have been obtained by the plotter itself.)

1) present X co-ordinate in plotter steps
2) present Y " "
3) horizontal character size
4) vertical " "
5) ISIN for character rotation, format given above
6) ICOS for " "
7) 0 is pen up, 32768_{10} if down.
8) current pen selected (for XY311 only)

For instance, from FORTRAN:

READ (7) LASTX, LASTY, ISX, ISY, ISIN, ICOS, IPEN, ISEL

MACRO users note that the first returned word is an octal 400000 for FORTRAN.

.CLOSE forces the handler to send to the PDP-11 any writes held in the handler. Normally, 10 decimal writes are grouped together to make up one buffer which is sent to the PDP-11.

A FORTRAN STOP statement will force a .CLOSE automatically.

NOTES:

It is possible to direct plotter output to a file on the disk, for instance. This file can then later be PIP-ed in IOPS binary mode to the plotter. Plotter files to be PIP-ed must be in the IOPS binary mode only.

The offline switch for the plotter is switch #2 of the PDP-11 console switches, regardless of whether spooling is enabled or not. The plotter will stop when the switch is raised, and resume when lowered. An IOPSUC XYU 4 message (indicating offline) at the console is for information only, it is neither necessary nor possible to resume plotter operation by typing CTRL R.

The user is provided with the facility of halting the output on the XY11 plotter (in a UNICHANNEL-15 System) at the end of the current file in the spooled mode. This permits the user to perform operations like positioning plotter pen on paper, changing pen/paper, etc.

PDP-11 Console switch 3 on the PDP-11 is used for this purpose. The user requests the PIREX system to halt output on the plotter at the end of the current file by setting PDP-11 switch 3 to the 'UP' or '1' position. The system periodically monitors this switch. When PDP-11 switch 3 is in the 'DOWN' or '0' position, the system resumes output on the plotter shortly thereafter.

The XY11 plotter is 1ØØ steps per inch with an 11-inch width. The XY311 plotter is 5ØØ steps per inch with a 3Ø-inch width. The X axis is in the direction of the paper roll. The character modes handle their own pen up-down. To initialize the plotter (XY11);

```
C  PUSH PEN TO PAPER EDGE

     IM=4
     IX=Ø
     IY=-12ØØ
     WRITE(7) IM,IX,IY

C  ESTABLISH CO-ORDINATES
     IM=7
     IY=Ø
     WRITE(7) IM,IX,IY
```

The first write in MACRO would be;

```
     .WRITE 7,Ø,BUFF,3
     .WAIT 7

BUFF 3ØØØ
        Ø
        4
        Ø
   77552Ø
```

APPENDIX A
XVM IOPS ASCII STANDARD CHARACTER SET

The following table shows the 7-bit ASCII characters interpreted by the XVM/DOS Monitor and System Programs either as meaningful data and command input or as control characters. The 7-bit octal code and the corresponding graphic characters conform to the standard ASCII 1968 64-character graphic subset.

The control characters (codes 00-37, and 175-177) are used for system control purposes. The characters shown in brackets are not used by the system and are available for user applications. Characters in parentheses denote the 1963 character set.

XVM IOPS ASCII Standard Character Set

7-BIT CODE	ASCII CHAR.	7-BIT CODE	ASCII CHAR.	7-BIT CODE	ASCII CHAR.	7-BIT CODE	ASCII CHAR.
000	NUL	040	SP	100	@	140	Not
001	[SOH	041	!	101	A	↕	recognized
002	STX]	042	"	102	B	174	by XVM/DOS
003	ETX (CTRL C)	043	#	103	C	175}	ALT MODE
004	EOT (CTRL D)	044	$	104	D	176}	
005	[ENQ]	045	%	105	E	177	DEL(rubout)
006	[ACK]	046	&	106	F		
007	[BELL]	047	'	107	G		
010	[BS]	050	(	110	H		
011	HT	051	)	111	I		
012	LF	052	*	112	J		
013	VT	053	+	113	K		
014	FF	054	,	114	L		
015	CR	055	-	115	M		
016	[SO]	056	.	116	N		
017	[SI]	057	/	117	O		
020	DLE (CTRL P)	060	0	120	P		
021	DC1 (CTRL Q)	061	1	121	Q		
022	DC2 (CTRL R)	062	2	122	R		
023	DC3 (CTRL S)	063	3	123	S		
024	DC4 CTRL T)	064	4	124	T		
025	NAK (CTRL U)	065	5	125	U		
026	[SYN]	066	6	126	V		
027	[ETB]	067	7	127	W		
030	CAN (CTRL X)	070	8	130	X		
031	[EM]	071	9	131	Y		
032	[SUB]	072	:	132	Z		
033	ESC (ALT MODE)	073	;	133	[		
034	[FS]	074	<	134	\		
035	[GS]	075	=	135	]		
036	[RS]	076	>	136	^(↑)		
037	[US]	077	?	137	_(←)		

NOTES: 1. Codes 33, 175 and 176 are interpreted as ESC (ALT MODE) and are converted on input to 175 by IOPS.

2. The left bracket, backslash, and right bracket (i.e. [, \ ,and]) characters are formed by typing SHIFTs K, L, and M, respectively, on Teletypewriters.

APPENDIX B
SAMPLE IOPS ASCII PACKING AND UNPACKING ROUTINES

This appendix contains two subroutines for use in packing and unpacking IOPS (5/7) ASCII. These routines are self-contained and can be incorporated in user programs directly or contained in the XVM/DOS System Library (.LIBR BIN) or user-created library and referenced globally (.GLOBL pseudo-op). If they are to be incorporated directly into a program, the .GLOBL and .END pseudo-ops must be removed.

Unpacking Routine

This routine has two entry points, GT.FST and GT.CHR. The GT.FST entry is used to initialize the routine for each unpacking sequence. Upon entry via GT.FST, the AC must contain the address of the first location in the buffer. (The routine automatically skips over the Header Word Pair.) On return the AC is unchanged. After initialization, each entry via GT.CHR returns a 7-bit ASCII character right justified in the AC. The user program must detect the end of the logical record, normally by checking for a Carriage RETURN or ALT MODE terminator. Entry to the routine from the user program is made by either a JMS or a JMS* instruction as follows:

If this routine is directly incorporated in the user's program:

```
        LAC     ADDRESS     /BUFFER ADDRESS
        JMS     GT.FST      /ENTRY TO INITIALIZE UNPACKING
         .
         .
        JMS     GT.CHR      /ENTRY TO UNPACK A CHARACTER
```

If this routine resides in a library:

```
        .GLOBL GT.FST,GT.CHR/EXTERNAL GLOBL DECLARATION
          .
          .
        LAC     ADDRESS     /BUFFER ADDRESS
        JMS*    GT.FST      /ENTRY TO INITIALIZE UNPACKING
          .
          .
        JMS*    GT.CHR      /ENTRY TO UNPACK A CHARACTER
        DAC     SAVCHR      /SAVE CHARACTER RETURNED IN AC
```

Sample IOPS ASCII Packing and Unpacking Routines

Packing Routine

This routine also has two entry points, PK.FST and PK.CHR. The PK.FST entry is used to initialize the routine for each new packing sequence. Upon entry via PK.FST, the AC must contain the address of the first location in the buffer (the routine ignores the header word pair automatically); on return, the AC is unchanged. After initialization, each entry via PK.CHR with a character in the AC will pack that character in 5/7 format. The characters to be packed must be right justified in the AC. On return to the user, the AC will still contain that character, however, bits 0-10 of the AC will be set to zero. The user program must detect the end of the logical record, normally by testing for a terminator such as Carriage RETURN or ALT MODE. Further, the user must also set up the header word pair for the logical record. Entry to the packing routine is made from the user's program either by a JMS or JMS* instruction as follows:

If this routine is directly incorporated in the user's program:

```
        LAC     ADDRESS         /BUFFER ADDRESS
        JMS     PK.FST          /ENTRY TO INITIALIZE PACKING
          .
          .
          .
        LAC     CHAR            /CHARACTER TO BE PACKED
        JMS     PK.CHR          /ENTRY TO PACK A CHARACTER
```

If this routine resides in a library:

```
       .GLOBL   PK.FST, PK.CHR /EXTERNAL GLOBL DEFINITION
          .
          .
        LAC     ADDRESS         /BUFFER ADDRESS
        JMS*    PK.FST          /ENTRY TO INITIALIZE PACKING
          .
          .
          .
        LAC     CHAR            /CHARACTER TO BE PACKED
        JMS*    PK.CHR          /ENTRY TO PACK A CHARACTER
```

Sample IOPS ASCII Packing and Unpacking Routines

```
/IOPS ASCII UNPACKING SUBROUTINE
/
/GT.FST-INITIALIZE 5/7 .ASCII UNPACKING. ON ENTRY
/AC CONTAINS ADDRESS OF I/O BUFFER TO BE
/UNPACKED. ON RETURN AC IS RESTORED.
/GT.CHR-AFTER 5/7 .ASCII UNPACKING HAS BEEN
/INITIALIZED BY GT.FST, GT.CHR WILL RETURN
/SUBSEQUENT CHARACTERS IN AC.
/
        .GLOBL  GT.FST,GT.CHR
GT.FST  0               /INITIALIZE
        DAC     GT.TMP  /SAVE AC
        TAD     L2      /SET BUFFER POINTER
        DAC     GT.PTR  /TO SKIP OVER HEADER WORD PAIR
        LAW     -1      /CHARACTER COUNTER
        DAC     GT.5
        LAC     GT.TMP  /RESTORE AC
        JMP*    GT.FST
GT.CHR  0
        ISZ     GT.5
        JMP     GT.MO   /WORD PAIR STARTED
        LAC*    GT.PTR  /NEED NEXT PAIR
        ISZ     GT.PTR
        DAC     GT.WD1  /FIRST PAIR
        LAC*    GT.PTR
        ISZ     GT.PTR
        DAC     GT.WD2  /SECOND PART
        LAW     17773   /RESET CHARACTER COUNTER.
        DAC     GT.5
GT.MO   LAW     17770   /SHIFT LOOP TO 7 1/2 TIMES
        DAC     GT.WD3
GT.LUP  LAC     GT.WD2
        RAL
        ISZ     GT.WD3
        JMP     GT.MOR
        AND     L177    /GOT CHARACTER
        SAD     SPACE   /IF SPACE DONT UPDATE GT.LST
        JMP     GT.EXT
GT.EXT  JMP*    GT.CHR  /EXIT
GT.MOR  DAC     GT.WD2
        LAC     GT.WD1
        RAL
        DAC     GT.WD1
        JMP     GT.LUP  /BACK TO LOOP
/
GT.PTR  0
GT.5    0
GT.WD1  0
GT.WD2  0
GT.WD3  0
GT.TMP  0
L2      2
L177    177
SPACE   40
        .END
```

Sample IOPS ASCII Packing and Unpacking Routines

```
/IOPS (5/7) PACKING SUBROUTINE
/
/PK.FST - INITIALIZE 5/7 .ASCII UNPACKING. ON ENTRY
/AC CONTAINS ADDRESS OF I/O BUFFER TO CONTAIN PACKED ASCII,
/ON RETURN AC IS UNCHANGED.
/PK.CHR IS NORMAL ENTRY POINT AFTER INITIALIZATION (PK.FST).
/AC CONTAINS CHARACTER TO BE PACKED, ON RETURN, AC CONTAINS
/THE SAME CHARACTER BUT HIGH ORDER BITS (0-10) ARE ZEROED
/OUT.
/
        .GLOBL  PK.FST,PK.CHR
SHAL=660000
PK.FST  0               /INITIALIZE
        DAC     KLCHR2  /SAVE AC
        TAD     L2      /SET KLPUTP TO BUFFER ADDRESS+2
        DAC     KLPUTP  /TO SKIP OVER HEADER WORD PAIR
        DZM     KL57
        DZM     CHRCNT
        LAC     KLCHR2  /RESTORE AC
        JMP*    PK.FST  /EXIT
PK.CHR  0
        ISZ     CHRCNT
        AND     L177
        DAC     KLCHR2
        CLL
        LAC     KL57    /CHAR. POSITION.
        TAD     (JMP*   KLJ57)
        DAC     .+2
        LAC     KLCHR2
        XX              /MODIFIED JMP
KLJ57   KL571           /CHAR1
        KL572           /CHAR2
        KL573           /CHAR3
        KL574           /CHAR4
        KL575           /CHAR5
KL571   ALS!SHAL        13      /11 LEFT
KL571A  DZM*    KLPUTP  /CLEAR DATA WORD
        JMP     KLND57
KL572   ALS!SHAL        4       /4 LEFT
        JMP     KLND57
KL573   RTR                     /3 RIGHT-1ST HALF
        RAR
        AND     L17
        XOR*    KLPUTP
        DAC*    KLPUTP
        ISZ     KLPUTP  /LAST WORD OF PAIR.
        LAC     KLCHR2  /2ND HALF
        ALS!SHAL        17      /15 LEFT
        JMP     KL571A
KL574   ALS!SHAL        10      /8 LEFT
        JMP     KLND57
KL575   RCL                     /1 LEFT
        DZM     KL57            /RESET 5/7 COUNTER
        SKP
```

Sample IOPS ASCII Packing and Unpacking Routines

```
KLND57  ISZ     KL57
        XOR*    KLPUTP
        DAC*    KLPUTP
        LAC     KL57
        SNA
        ISZ     KLPUTP  /2ND WORD COMPLETE
        LAC     KLCHR2
        JMP*    PK.CHR  /EXIT.
KL57    0
KLPUTP  0
KLCHR2  0
CHRCNT  0
L2      2
L177    177
L17     17
/
        .END
```

APPENDIX C

INPUT/OUTPUT DATA MODE TERMINATORS

All handlers determine the data mode from the .READ or .WRITE macro as shown in the following table. Abbreviations and acronyms are defined at the end of the table.

SPECIFIC DEVICE HANDLERS

DATA MODE	HANDLER	INPUT	OUTPUT
IOPS ASCII	DT (all versions)	HWP, WC, EOM, EOF whichever is smaller	HWP, EOM
	DK, DP, RK (all versions)	HWP, WC, EOM, EOF, whichever is smaller	HWP, EOM
	CDB.	WC, EOM, EOF	Not applicable
	LPA.15 LPA.11	Not applicable	HWP, CR, AM, IOPS 37 on overflow VC
	MTA.	HWP, WC EOM, EOF whichever is smaller	HWP, EOM
	MTC.	HWP, WC EOM, EOF whichever is smaller	HWP, EOM
	MTF.	WC, EOM, EOF	HWP, EOM
	TTA.	CR, AM, CTRL D, WC (no CR appended)	HWP*, CR, AM, EOF
	PP (all versions)	Not applicable	HWP, CR, AM, EOM
	PR (all versions)	WC, CR, AM, EOM, EOF	Not applicable
	XYA.	NA	CR, EOM, AM

*HWP with word pair count less than two will cause carriage return/line feed only. If the word pair count is two or more, only a carriage return or an ALT MODE will terminate output.

SPECIFIC DEVICE HANDLERS (Cont)

DATA MODE	HANDLER	INPUT	OUTPUT
IMAGE ALPHA/ IMAGE BINARY	DT (all versions)	HWP, WC EOM, EOF whichever is smaller	HWP, EOM
	DK, DP, RK (all versions)	HWP, WC EOM, EOF whichever is smaller	HWP, EOM
	CDB.	Not applicable	Not applicable
	LPA.15 LPA.11	Not applicable	Not applicable for BIN HWP, CR, AM, VC for ALPHA IOPS 37 if line is exceeded
	MTA.	HWP, WC EOM, EOF whichever is smaller	HWP, EOM
	MTC.	HWP, WC EOM, EOF whichever is smaller	HWP, EOM
	MTF.	WC, EOM, EOF	HWP, EOM
	TTA.	Not applicable for BIN CTRL D, WC for ALPHA	Not applicable for BIN HWP
	PP (all versions)	Not applicable	HWP, EOM for BIN HWP, AM, CR, EOM for ALPHA
	PR (all versions)	WC, EOM, EOF	Not applicable

SPECIFIC DEVICE HANDLERS (Cont)

DATA MODE	HANDLER	INPUT	OUTPUT
IOPS Binary	DT (all versions)	HWP, WC whichever is smaller EOM, EOF	HWP, EOM
	DK, DP, RK (all versions)	HWP, WC whichever is smaller EOM, EOF	HWP, EOM
	CDB.	Not applicable	Not applicable
	LPA.15, LPA.11	Not applicable	Not applicable
	MTA.	HWP, WC whichever is smaller EOM, EOF	HWP, EOM
	MTC.	HWP, WC whichever is smaller EOM, EOF	HWP, EOM
	MTF.	WC, EOM, EOF	HWP, EOM
	TTA.	Not applicable	Not applicable
	PP (all versions)	Not applicable	HWP, EOM
	PR (all versions)	WC, EOM, EOF	Not applicable
	XYA.	NA	DD

SPECIFIC DEVICE HANDLERS (Cont)

DATA MODE	HANDLER	INPUT	OUTPUT
DUMP MODE	DT (all versions)	WC, EOM, EOF	WC, EOM
	DK, DP, RK (all versions)	WC, EOM, EOF	WC, EOM
	CDB.	Not applicable	Not applicable
	LPA.15, LPA.11	Not applicable	Not applicable
	MTA.	WC, EOM, EOF	WC, EOM
	MTC.	Not applicable	Not applicable
	MTF.	Not applicable	Not applicable
	TTA.	Not applicable	Not applicable
	PP	Not applicable	WC, EOM
	PR	WC, EOM	Not applicable

List of Abbreviations and Acronyms:

Handlers:

DT	DECtape
DK, DP, RK	Disk (DECdisk or Disk Pack or Disk cartridge)
CDB.	Card Reader Handler
LPA.15	Line Printer Handler for XVM users
LPA.11	Line Printer Handler for UC15 users
MTA.	Magtape "A" Handler
MTC.	Magtape "C" Handler
MTF.	Magtape "F" Handler
TTA	Teleprinter Handler
PP	Paper Tape Punch Handler
PR	Paper Tape Reader Handler

Abbreviations:

AM	ALT MODE key
CR	Carriage RETURN (RETURN) key
EOF	End-of-File
EOM	End-of-Medium
HWP	Header Word Pair's word pair count
VC	Vertical Control Character
WC	Word Count in an I/O Macro
DD	Data dependent

APPENDIX D

IOPS ERROR CODES

The following table lists the Meaning and Data Output for those IOPS error codes that are unrelated to UNICHANNEL operation. The UNICHANNEL IOPS error codes (IOPSUC) are described in Appendix J.

MEANING FOR IOPS ERROR CODES

Error Code	Meaning	Error Data Output[1]
0	Illegal CAL Function Code - The function code immediately following the offending CAL instruction is not legal	CAL address
1	CAL* Illegal - The Monitor does not permit execution of CAL* (indirect) instructions	CAL address
2	.DAT Slot Error - a. The .DAT Slot number (bits 9-17 of the CAL) is either Ø or outside the range of legal numbers established when the system was created (at System Generation). b. No .IODEV has been issued for this .DAT Slot.	CAL address
3	Illegal Interrupt -An interrupt originated from a device when either its handler was not core resident or its handler was not previously initialized (via .INIT).	Contents of the IORS word at the time of the interrupt.†
4	Device Not Ready - a. Device "OFF LINE", "WRITE PROTECTED" or Unit Number not selected. b. Line Printer or Paper Tape Punch out of paper. c. Line Printer Alarm Status d. Card Reader stacker full, mis-punched card, card jam, hopper empty, EOF card missing. e. 9-channel I/O request to 7-channel Magtape transport (or vice versa). Remedy error condition and type CTRL R to continue interrupted operation.	Disk - dv & unit,CAL fcn,UIC Card Reader - dv, message Line Printer - dv Teleprinter - dv Magtape - dv, & unit,CAL fcn DECtape - dv & unit Other - CAL adr
5	Illegal Setup CAL - A CAL to set up API/PI linkage was issued by a handler (as a result of a .INIT) when no skip IOT existed in the Monitor's Skip Chain for that device (IOTs are placed in the Skip Chain during System Generation.).	CAL address

†Refer to Chapter 3 of the PDP-15 User's Handbook, Vol. I Processor

MEANING FOR IOPS ERROR CODES (Cont)

Error Code	Meaning	Error Data Output[1]
6	Illegal Handler Function - A CAL has been issued to a handler which is incapable of performing that function (e.g., .READ to the paper tape reader, .TRAN to the disk in the reverse direction, etc.)	Disk - CAL adr,dv & unit,CAL fcn,UIC Card Reader - dv,message Line Printer - dv Teleprinter - dv Magtape - CAL adr,dv & unit, CAL fcn Other devices - CAL address
7	Illegal Data Mode - a. A .READ or .WRITE was issued using a Data Mode unacceptable to the handler. b. An attempt was made to change transfer direction prior to issuing a new .INIT via that .DAT Slot. c. For DL11 - attempt to change from non-file oriented to file oriented I/O.	Disk - CAL adr,dv & unit,CAL Fcn, UIC Card Reader -dv,message Line Printer - dv Teleprinter - dv Magtape - CAL adr,dv & unit, CAL fcn Other devices - CAL address DL11 - CAL adr, dv
10	File Still Active - Failure to close (.CLOSE) a file before another .SEEK, .ENTER, .RAND, .RENAM, .FSTAT, .DLETE or .CLEAR is issued via the same .DAT Slot.	Disk - CAL adr,dv & unit,CAL fcn,UIC, filnam Magtape - CAL adr,dv & unit, CAL fcn DECtape - CAL adr
11	.SEEK/.ENTER/.RAND Not Executed - A .READ, .WRITE, or .RTRAN was issued to a directoried device with no prior .SEEK/.ENTER/.RAND.	Disk - CAL adr,dv & unit,CAL fcn,UIC Magtape - CAL adr,dv & unit,CAL fcn DECtape - CAL adr
12	Terminal Device Error - a. DECtape mark track error (tape must be reformatted) b. Magtape EOT encountered on space forward	DECtape - The contents of device status register "B" (bits 0-11) and unit no. (bits 15-17) Magtape-CAL adr, dev & unit, CAL fcn
13	File Not Found - The file name specified in the CAL argument (CAL+2) is not in the file directory of the device associated with the specified .DAT Slot (.SEEK, .ENTER, .RAND).	Disk - CAL adr,dv & unit,CAL fcn, UIC, filnam Magtape - CAL adr DECtape - CAL adr
14	Directory Full - In response to a .ENTER the DECtape or MAGtape handler has determined that there is no space for another file name.	Magtape - CAL adr DECtape - CAL adr

MEANING FOR IOPS ERROR CODES (Cont)

Error Code	Meaning	Error Data Output[1]
15	Device Full - No space available on the device medium for data storage.	Disk - CAL adr, dv & unit, CAL fcn, UIC, Magtape - CAL adr,dv, & unit, CAL fcn DECtape - CAL adr
16	Output Buffer Overflow - The word pair count on the current .WRITE is greater than 177_8. (This error is obsolete and has been replaced by IOPS 23.)	CAL adr
17	Too many Files for Handler - Too many files are currently open on the handler to be referenced by this CAL. (See handler descriptions in Chapter 9 for limitations.	Disk - CAL adr,dv & unit, CAL fcn, UIC Magtape - CAL adr,dv & unit,CAL fcn DECtape - CAL adr
20	Disk Hardware Failure -	Block no., dv & unit, CAL fcn, UIC
21	Illegal Disk Address - An attempt was made to reference a block number which was either 0 or greater than the maximum number of blocks available on the disk.	Block no., dv & unit, CAL fcn, UIC
22	Two Output Files on One Unit - An attempt was made to reference more than one output file concurrently on the same DECtape or Magtape unit.	Magtape - CAL adr,dv & unit, CAL fcn DECtape - CAL address
23	Illegal Word Pair Count - The word pair count in header word Ø of the logical record currently being transferred is either Ø or greater than 177_8. For DL11, this error is given when an attempt is made to transmit more than 73_8 characters.	Disk - CAL adr, dv & unit, CAL fcn, UIC filnam Magtape - CAL adr,dv & unit,CAL fcn Other devices - CAL adr. DL11 - CAL ab, dv.
24	Illegal unit number.	Card Reader - CAL adr DL11 - CAL adr, dv
25	Negative or 0 character count (IOPS ASCII write) X or Y increment too large (>2**14) (binary write)	XY Plotter - CAL address
27	Illegal write type.	XY Plotter - CAL address
30	API Software Level Error - An API break occurred to a software API level which did not have the appropriate transfer vector(s) setup in .SCOM+12 through .SCOM+15.	Contents of the API Status Register
31	Nonexistent Memory Reference - A nonexistent memory reference occurred with memory protect mode ON without a user-defined violation routine.	Program Counter

MEANING FOR IOPS ERROR CODES (Cont)

Error Code	Meaning	Error Data Output[1]
32	Memory Protect Violation - A reference was made to a location in memory below the memory protect boundary without a user-defined violation routine.	Program Counter
33	Memory Parity Error - A memory parity error occurred without a user-defined error routine.	Program Counter
34	Power Fail Skip Not Setup - The power failure interrupt detected a power low condition with no user-defined service routine to save appropriate registers.	Program Counter
35	Control character routine returned to IOPS error processor.	Control Character routine address
37	Print Line Overflow - The 81st or 133rd character (depending on the line printer type) of the line currently being output is not a legal terminator. (Carriage RETURN, ALT MODE, FORM FEED, LINE FEED, Vertical TAB, etc.). The rest of the line is lost.	CAL adr,dv
40	Header Label Error - During processing of a .SEEK to a Magtape file, the handler calculated file name disagrees with name present in file header label.	CAL adr
41	Directory Format Error - Illegal or meaningless data was found in the Magtape file directory.	CAL adr
42	Accessibility Map Overflow - During the processing of an .ENTER to the Magtape unit, the accessibility map is found to be full. (Too many files.) Use MTDUMP to delete unwanted files to obtain space.	CAL adr
43	Directory Recording Error - The file directory of the referenced Magtape has been contaminated. Use MTDUMP to reformat the directory.	CAL adr

MEANING FOR IOPS ERROR CODES (Cont)

Error Code	Meaning	Error Data Output[1]
44	Logical EOT Detected - The Magtape handler detected a logical End-of-Tape during the processing of a .SEEK or .ENTER.	CAL adr,dv & unit,CAL fcn
45	Long Input Record - The record being input from Magtape is too long for the handler's internal buffer (255_{10} words maximum).	CAL adr,dv & unit,CAL fcn
46	Attempt to Delete A System File - An attempt has been made via a .DLETE to delete a file having a "SYS" extension (applies to Advanced Monitor System DECtapes only).	CAL address
47	Illegal Horizontal Tab - An attempt has been made to issue a Horizontal TAB operation on the Line Printer which caused the column count to exceed the device's capacity for line length.	CAL adr,dv
51	Illegal User File Directory - When performing Disk I/O: a. A .USER was issued using -1, ???, or @@@ as a UIC. b. A .SEEK was attempted to a nonexistent UFD.	CAL adr,dv & unit,CAL fcn, UIC
55	No Buffers Available - A .GTBUF Macro was issued from either a handler or a user program with an insufficient number of buffers allocated (see BUFFS for DL11, no TCB area available.	CAL adr,dv & unit,CAL fcn,UIC
56	Mass Storage Busy Table Full - An attempt to open a file on a handler which uses the MSBT when there was not an empty entry in the MSBT. This can occur only when the user is making dynamic .DAT assignments.	CAL adr,dv & unit,CAL fcn
61	Parity Error in Directory or File Bit Map - Defective data, device medium, or hardware (see Recovery procedure for DECtape in note 2 below).	Disk - CAL adr,dv & unit,CAL fcn,UIC DECtape - CAL address
63	Protected User File Directory - Attempt to create (.ENTER) or delete (.DLETE) a file in a protected directory (see 9.3.5).	CAL adr,dv & unit,CAL fcn, UIC

MEANING FOR IOPS ERROR CODES (Cont)

Error Code	Meaning	Error Data Output[1]
64	Protected File - Attempt to access a file via .RAND with protection codes 2 or 3, or to .SEEK a disk file with protection code of 3.	CAL adr,dv & unit,CAL fcn, UIC
65	Unrecoverable Magtape Error - For DL11 - Unrecoverable Telecommunications Error	Magtape status word, dv,CAL,fcn DL11 - status word,dv
66	Relative Block Not Within File - Attempt to access (via .RTRAN) a block not within the limits of the current file [i.e., block Ø or n+1]).	CAL adr,dv & unit,CAL fcn,UIC,filnam
67	Illegal DECdisk Word Transfer Starting Address or Count - When issuing an .RTRAN: a. The argument which specifies the first word in the DECdisk block to be transferred is either Ø or greater than 376_8. b. The argument which specifies the number of words to be transferred exceeds the physical block size (i.e., Ø< no. words< 253-word starting address). I/O buffer above 32K.	CAL adr,dv & unit,CAL fcn,UIC,filnam DL11 - CAL adr,dv
70	Buffer Size Too Small - The size of the buffer allocated by .GTBUF and .GVBUF Macros (established during System Generation) is not large enough for the handler attempting to utilize them.	CAL adr,dv & unit, CAL fcn,UIC
71	Empty UFD - A .SEEK or .RAND was attempted to a UFD which did not contain any files.	CAL adr,dv & unit,CAL fcn,UIC,filnam
72	Input Parity or Write Check Error - Hardware error detected; type CTRL R to continue.	block no.,dv & unit,CAL fcn,UIC
73	Null File Name - A .SEEK, .ENTER, .DLETE, .FSTAT or .RAND was issued with a null filename argument.	CAL adr,dv & unit,CAL fcn,UIC

MEANING FOR IOPS ERROR CODES (Cont)

Error Code	Meaning	Error Data Output[1]
74	Disk System File Structure Degradation[3] - a. Attempt to turn off a bit in a submap that is already off. b. Attempt to write block Ø in a sequential file. c. Attempt to use block Ø as a UFD block. d. Nonexistent submap.	CAL adr,dv & unit,CAL fcn, UIC,filnam
75	Disk System File Structure Degradation - Word 1 of submap is greater than word Ø or is Ø or negative.	CAL adr,dv & unit,CAL fcn,UIC,filnam
76	Disk System File Structure Degradation - Word 376_8 of the first UFD or MFD block is not -1. Type CTRL R to continue.	CAL adr,dv & unit,CAL fcn,UIC,filnam
77	Undersized or Nonexistent CTRL Q AREA - The system attempted to utilize a CTRL Q area (via CTRL Q, QDUMP, GET, PUT, GETP, GETT, GETS keyboard commands) which was nonexistent or of insufficient size for the amount of core available. (The CTRL Q area is created during System Generation.)	The address (15-bits) to which control would have been passed if the requested operation had been successful.

NOTES

[1] Abbreviations: addr = address dv = device fcn = function filnam = file name

[2] Recovery procedures for IOPS 61:

1) Repeat operation which causes error.
2) If error persists, remount DECtape on another drive and repeat step 1.
3) If error still persists and you are very familiar with DECtape file structure and have a reasonably current directory listing, proceed as follows:
   a. Using DUMP, obtain a listing of each file in the directory listing. (The directory listing provides the starting block number for each file. The last (link) word in each block points to the next block. Negative block numbers indicate reverse recording. Last block has a link of 777777.)
   b. Use PIP to block copy each file onto a good tape.
   c. Use PATCH to construct a directory on the new tape. DO NOT WRITE ON THIS TAPE - IT HAS NO BIT MAPS.
   d. Use PIP to transfer each reconstructed file to still another tape (this reconstructs the master and file bit maps).

[3] These errors usually result from hardware failure or inadvertent manipulation of disk structure data areas.

APPENDIX E
LINKING LOADER AND SYSTEM LOADER ERRORS

The following error codes are output by both the Linking Loader and the System Loader. When output by the Linking Loader, the errors are identified as shown below. When output by the System Loader, the errors are identified as ".SYSLD n" instead of ".LOAD n".

Error	
.LOAD 1	Memory overflow - the Loader's symbol table and the user's program have overlapped. At this point the Loader memory map will show the addresses of all programs loaded successfully before the overflow. Increased use of COMMON storage may allow the program to be loaded, as COMMON can overlay the Loader and its symbol table, since it is not loaded into until run time.
.LOAD 2	Input data error - parity error, checksum error, illegal data code, or buffer overflow (input line bigger than Loader's buffer).
.LOAD 3	Unresolved Globals - any programs or subroutines required but not found, whether called explicitly or implicitly, are indicated in the memory map with an address of 000000. If any of the entries in the memory map has a 000000 address, loading was not successful; the cause of trouble should be remedied and the procedure repeated.
.LOAD 4	Illegal .DAT slot request - the .DAT slot requested was: a. Out of range of legal .DAT slot numbers, b. Zero, c. Unassigned; that is, was not set up at System Generation Time or was not set up by an ASSIGN command.
.LOAD 5	Program segment greater than 4K - the program segment being loaded in Page Mode exceeds a Page Bound (i.e., program is greater than 4K).

COMMON block length error

.LOAD 6

a. Common Block Too Large - An attempt was made to allocate a common block of zero length or larger than 32K-1 words. A loader map line of the form:

C blknam length

will be output on the console printer immediately prior to the line containing the error code. The program which made the offending common block reference will be the most recent program name typed out via the "P" (program load address) loader map option.

b. COMMON block requested length is greater than its length when first loaded. This error may be the result of loading the modules in the wrong order. It is remedied by defining all instances of the same COMMON block to be of identical length.

.LOAD 7

Value of Relocated Global Symbol Too Large - An attempt was made to define a global symbol with a relocated value greater than 32K-1. A loader map line of the form:

G symnam value

where:

symnam = global symbol name
value = attempted value

will be output on the console printer immediately prior to the line containing the error code.

.LOAD 8

Data initialization attempt outside of range of program or COMMON block. This can be the result of an input data error but is usually caused by an illegal data initialization attempt via the MACRO .CBC pseudo-op.

.LOAD 9
device

Duplicate handler request - More than one handler for the same device has been requested. The first two letters of the duplicate file names are output along with the error message. Loading continues despite this message but when the loaded programs are executed, interrupts initiated by I/O requests may not properly distinguish among the multiple handlers causing an error.

APPENDIX F
XVM ASCII/HOLLERITH CORRESPONDENCE

The following table shows the correspondence between the XVM 64 character graphic subset of ASCII and the DEC 029/026 Hollerith codes. Both 029 and 026 codes are identical for numeric and alphabetic characters but vary for symbol representation. The 029 code, except as indicated by brackets , is a subset of the standard Hollerith punched card code specified in ANSI standard X3.26-1970. Characters in parentheses denote the 1963 character set.

XVM ASCII/Hollerith Correspondence

ASCII		HOLLERITH		ASCII	HOLLERITH		
CHAR.	7-BIT CODE	DEC 029 CODE	DEC 026 CODE	CHAR.	7-BIT CODE	DEC 029 CODE	DEC 026 CODE
Space	40			@	100	8-4	8-4
!	41	[11-8-2]	12-8-7	A	101	12-1	12-1
"	42	8-7	0-8-5	B	102	12-2	12-2
#	43	8-3	0-8-6	C	103	12-3	12-3
$	44	11-8-3	11-8-3	D	104	12-4	12-4
%	45	0-8-4	0-8-7	E	105	12-5	12-5
&	46	12	11-8-7	F	106	12-6	12-6
'	47	8-5	8-6	G	107	12-7	12-7
(	50	12-8-5	0-8-4	H	110	12-8	12-8
)	51	11-8-5	12-8-4	I	111	12-9	12-9
*	52	11-8-4	11-8-4	J	112	11-1	11-1
+	53	12-8-6	12	K	113	11-2	11-2
'	54	0-8-3	0-8-3	L	114	11-3	11-3
-	55	11	11	M	115	11-4	11-4
.	56	12-8-3	12-8-3	N	116	11-5	11-5
/	57	0-1	0-1	O	117	11-6	11-6
0	60	0	0	P	120	11-7	11-7
1	61	1	1	Q	121	11-8	11-8
2	62	2	2	R	122	11-9	11-9
3	63	3	3	S	123	0-2	0-2
4	64	4	4	T	124	0-3	0-3
5	65	5	5	U	125	0-4	0-4
6	66	6	6	V	126	0-5	0-5
7	67	7	7	W	127	0-6	0-6
8	70	8	8	X	130	0-7	0-7
9	71	9	9	Y	131	0-8	0-8
:	72	8-2	11-8-2	Z	132	0-9	0-9
;	73	11-8-6	0-8-2	[	133	12-8-2	11-8-5
<	74	12-8-4	12-8-6	\	134	11-8-7	8-7
=	75	8-6	8-3	]	135	0-8-2	12-8-5
>	76	0-8-6	11-8-6	^ (↑)	136	12-8-7	8-5
?	77	0-8-7	12-8-2	_ (←)	137	0-8-5	8-2

NOTES:
1. ASCII codes 00-37 and 140-177 have no corresponding codes in the DEC 026 & 029 Hollerith sets and therefore, are not presented here.
2. ALT MODE is simulated by a 12-8-1 punch (multiple punch A8).
3. End-of-file corresponds to a 12-11-Ø-1 punch (multiple punch AØ-).
4. The card reader hardware supplies the binary equivalent of Hollerith code which in turn is mapped into 7-bit ASCII by the Card Reader Handler.

APPENDIX G

SPECIAL DEBUGGING TOOLS
(QDMP XVM and UDMP11)

Two special dump tapes are being issued with the XVM/DOS system. These tapes, which use hardware read in mode (HRM), are very special purpose dumps, and are provided as a convenience to those customers with the exact configurations needed for their use. These are unsupported items. No commitment to improve these products or to make them more applicable to general usage is implied.

QDMP XVM

This paper tape performs a ↑Q dump of XVM memory to DECtape. It overlays most of the XVM/DOS bootstrap. The standard XVM/DOS DUMP program can be used to print the data after the system has been recovered.

Operating Instructions

1. Load the QDMP XVM paper tape into the reader.
2. Load 77650_8 (32K) into the ADDRESS Switches.
   57650_8 (24K)
3. Set the console data switches to zero to signal QDMP XVM to dump core up to and including itself. If the data switches are set non-zero, QDMP XVM will use that as the upper limit of memory (up to 17 bits of address may be specified).
4. Put an empty directoried DECtape on DECtape unit #7 WRITE enabled. The tape need not be initialized with space reserved for a QDMP. If such is not done, however, subsequent writing of user files to the tape may destroy the dumped data.
5. Depress STOP and RESET simultaneously.
6. Depress READIN.
7. The Dump should now occur. If it does not occur, a problem such as DECtape offline, etc., has been detected. Rectify the problem and press CONTINUE.
8. Reload XVM/DOS.
9. Use the ALL command (to the DUMP program) to list the dump from DECtape #7. The amount of core listed by DUMP is a function of the amount of core in use when DUMP is loaded.

UDMP11

This program performs a dump of the PDP-11 local memory to the PDP-11 listing device (LP11/LS11/LV11). It loads into common memory. All registers except the PC, PS, and device registers are listed.

Operation:

1. Load the UDMP11 paper tape into the Reader.
2. Halt the PDP-11.
3. Load the XVM address switches with $177\emptyset\emptyset_8$.
4. Depress STOP and RESET on the XVM console.
5. Depress READIN on the XVM console.
6. The tape should now read in. If it fails to read in, repeat steps 3, 4 and 5.
7. Load 100000 for 8K or 120000 for 12K, the starting address, into the PDP-11 switch register.
8. Raise the PDP-11 HALT switch.
9. Insure that the listing device is ON-LINE.
10. Depress the PDP-11 START switch. The Dump should now proceed.

APPENDIX H

XVM/DOS TERMS AND ACRONYMS

Terms unique to the XVM/DOS Software System are listed and described in the following table. The acronyms for each term are also given.

TERM	ACRONYM	DEFINITION
Bad Allocation Table	BAT	A device (disk) table which indicates, in storage blocks, any faulty disk areas in which data cannot be stored.
Master File Directory	MFD	A master device (disk) file directory which contains pointers to all user directories (UFD's) within a disk device.
Monitor Identification Code	MIC	The master system password which permits full access to all files within the system. This code identifies the system manager and should be used only by him.
Storage Allocation Table	SAT	The device (disk) table which stores busy, not-busy indicators for the disk storage area.
System Block	SYSBLK	The system table which contains the names, locations, and loading and starting parameters for all system programs within the operating system.
User File Directory	UFD	File directories for each user who established disk file storage areas within the system.
User File Directory Table	UFDT	The system directory table which maintains the relationship between the system's .DAT slots and each unique user identification code (UIC).
User Identification Code	UIC	A password entered by a user to uniquely define himself and any files which he may enter. If necessary, a user may enter more than one UIC to establish several unique sets of files. Since only one user may employ the system at any one time, the current UIC is the last logged-in UIC.

APPENDIX I

UC15 SPOOLER ERROR MESSAGES

The following is a list of the error messages printed by the 'SPOOL' program and the conditions that cause these:

MESSAGE	CONDITIONS
("$" BAD HERE)	An altmode is not allowed as the answer.
.DAT -14 SYSTEM FAILURE - ABORT	The system disk (RKØ or DPØ or DKØ) may be failing.
(# > 6 DIGITS)	Too many digits in answer.
DISK BUFFER - NOT IN COMMON MEMORY - ABORT	The disk buffer used to copy (in 256 word blocks) the spooler into memory from the system disk (RKØ, DPØ or DKØ) does not reside in common memory. Thus, PIREX cannot move the spooler into local PDP-11 memory.
FATAL ERROR - DOS LACKS TCB AREA FOR SPOL15 USE - ABORT	The XVM/DOS monitor must have a TCB area that can be used by "SPOOL". This TCB area is the one pointed to by the second entry in the TCB table pointed to by .SCOM+100.
FATAL RK DISK ERROR - ABORT	An error was detected during I/O on the chosen RK unit. Ten retries were attempted by PIREX and the error was not corrected. This is either a bad disk cartridge or a hardware problem.
FATAL TCB ERROR ON SOFTWARE DIRECTIVE - ABORT	A TCB sent to PIREX was rejected.
("n" HAS NO SPOOLER AREA)	The chosen RK unit has no (Ø size) spooler area defined on it.
INSUFFICIENT FREE LOCAL-11 MEMORY TO LOAD SPOOLER - ABORT	The spooler (SPOL11) will not fit into the remaining LOCAL-11 memory.
("n" IS A WRITE PROTECTED UNIT)	A chosen RK unit is write locked.
("a" ISN'T OCTAL #)	An octal number was expected in the answer.
("x" ISN'T SYMBOL)	A symbol (non-numeric answer) was expected.

UC15 Spooler Error Messages

MESSAGE	CONDITIONS
("n" NON-EXISTENT RK UNIT #)	The chosen RK unit does not exist; e.g., choosing RK3 on a system with only RKØ and RK1.
(NON-PRINTING CHAR)	A non-printing character was entered in an answer.
SPOOLER TCB ERROR - ABORT	The spooler TCB sent to PIREX was not accepted.
SPOOL NOT ON .DAT -14 DEVICE - ABORT	The SPOOL program, SPOL15, which has printed this error message, is not installed on the system device under the name SPOOL. The spooler SPOL11, cannot be loaded into the PDP-11 unless both it and SPOL15 have been installed in SPOOL on the system disk.
SPOL11 VERSION NOT AT LEAST DOS V3BØØØ - ABORT	The version of SPOL11 being loaded does not seem to be at least V3BØØØ. This is determined by the presence of the SPOL11 size in word Ø of the spooler (SPOL11) and the entry point offset in word 1. If word Ø is equal to Ø, this error is reported.

APPENDIX J
UC15 SYSTEM ERROR MESSAGES

The Error Messages from tasks running under PIREX have the following format.

IOPSUC YYY XXXX

Where YYY denotes on the following:

EST	Stop all I/O	task
ESD	Software Driver	"
DTU	DECtape	"
LPU	Line Printer	"
CDU	Card Reader	"
GRU	Plotter	"
ESP	Spooler	"
EMA	MAC11	"
PRX[1]	PIREX	Monitor

XXXX denotes one of the following:

3 - Illegal interrupt to driver

4 - Device not ready

12 - Device failure

15 - Spooler full warning message

20 - Spooler disk error-spooling terminated

25 - IOPSUC XYU 25 means that illegal data is being passed to the handler. This error is given under the following conditions.

1. When the mode=6 and the character count in the buffer is zero or negative, and

2. When the mode=2/3/4/5/$10/11_8$ and ΔX or ΔY is greater than 2^{14} (ΔX^8 and ΔY represent the change or difference between the current X, Y and the previous X, Y).

27 - IOPSUC XYU 27 means an illegal mode has been specified. This message will be given whenever the mode is any value other than $0-12_8$ (both inclusive).

[1] PRX is shown only in the following message: IOPSUC PRX 4 which means that the PIREX executive is not running or that it is excessively slow in responding.

45 - Greater than 80 column card
55 - No spooler buffers available
72 - Illegal punch combination
74 - Timing error--card column lost--retry card
75 - Hardware busy--driver not
76 - Hardware error between cards
77 - Unrecognized task request--device not present
400 - Spooler empty--XVM input request pending

Additional IOPS error messages:

Error Code	Meaning
200	Non-existent task referenced.
300	Illegal API level given (illegal values are changed to level 3 and processed).
400	Illegal directive code given.
500	No free core in the PDP-11 local memory.
600	ALT node for this TCN missing.
777	Request node was not available from the POOL; i.e., the POOL was empty and the referenced task was currently busy or the task did not have an ATL node in the Active Task List; i.e., specific device task was not installed in either PIREX or the spooler.
26	SPOOLER DISK RUN DOWN WHILE SPOOLER RUNNING

INDEX

READER'S COMMENTS

NOTE: This form is for document comments only. Problems with software should be reported on a Software Problem Report (SPR) form.

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

_____

_____

_____

_____

_____

_____

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

_____

_____

_____

_____

_____

_____

Is there sufficient documentation on associated system programs required for use of the software described in this manual? If not, what material is missing and where should it be placed?

_____

_____

_____

_____

_____

_____

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

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Non-programmer interested in computer concepts and capabilities

Name _____ Date _____

Organization _____

Street _____

City _____ State _____ Zip Code or Country _____

If you require a written reply, please check here. ☐

Please cut along this line.

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

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

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

digital

Software Communications
P. O. Box F
Maynard, Massachusetts 01754