CHAPTER 6
PRIVILEGED OPERATIONS

6.1 INTRODUCTION

The system manager is responsible for installation of the system and its daily operation. The following chapter contains all commands, programs and documentation concerning privileged operations which are performed on a regular basis.

Any job can be made privileged if bit Ø of the protection code of the current account is set. This bit may be set by the system manager when he creates the account. In addition to LOGging IN to a privileged account, you must execute the PRIV command with an argument of 4XXX to become a privileged user (see 6.2.1). A privileged user can write on SYS and can also execute some special commands in SCALE. Privilege is a potentially dangerous feature of ETOS operations. The system manager should limit the accounts that can be privileged to those that definitely require it. Account [ØØ,Ø3] is privilegable when the system is delivered.

6.2 PRIVILEGED SCALE COMMANDS

The concept of SCALE and its operation is explained in the User's Guide, Chapter 3. This chapter should be read before you attempt to use SCALE commands. The SCALE commands presented in this section may only be executed by privileged users. Since these commands could be used to affect other users or corrupt the ETOS software, they are not listed in the System User's Guide. It is the system manager's responsibility to acquaint privileged users with these commands and their inherent dangers.

6.2.1 The PRIVilege Command

PR[IV] [n]

where "n" can be set in accordance with Table 6-1.

Table 6-1
Privilege Word Format

Bits	Meaning
Ø	1=privileged Ø=non-privileged
1-5	Unused
6-8	Run queue number (Ø has higher priority than 7)
9-11	Index into LSLICE table

The PRIV command is utilized for three distinct functions.

The most common use of the command is that of privileging the user. If you want to perform any privileged operation described in this chapter, you must be logged into a privileged account and set bit Ø of the PRIV word. The PRIV command for privileging a user is of the following form.

!PRIV 40XY

X and Y are used for the other two functions of the PRIV command.

The second use of the PRIV command is controlled by the third octal digit of the PRIV word. ETOS has an internal priority structure which you cannot change. Terminal input/output has the highest priority and card reader input has the lowest. ETOS automatically adjusts these values, based on the job's current status. ETOS also has a priority queue which you can change. This queue has eight positions. Ø is the highest priority and 7 is the lowest priority. When you LOGIN, you are automatically assigned priority level 4. A job which has a lower priority

number than another job will get scheduled more often. Therefore, you can place your most critical jobs at priority level 0 and your jobs which are not time-critical at level 7.

The fourth octal digit of the PRIV word controls quantum length. The LSLICE table (see 3.13) is a table which contains quantum lengths. A quantum is the period of time which you run before another user is scheduled. If you have a large quantum, you will run a given job faster than a user with a smaller quantum. The reason for this decreased time is that you receive a greater percentage of the central processing unit resources. The LSLICE table has eight settings. A value of 0 for the last digit at the PRIV word corresponds to the first entry. A value of 7 corresponds to the last entry. When you LOGIN, this digit is set to 0. If you have not changed the LSLICE table from its distribution form, your quantum is 30 (base 8) or 24 (base 10) tenths of a second. You can increase your quantum if your job is time-critical.

Each time you LOGIN, your PRIV word is set to 0040. This value means that you are non-privileged, you are at priority level 4, and your quantum is equal to the first entry of the LSLICE table. If you have not changed the LSLICE table, the following command causes your non-privileged job to execute at the fastest possible rate.

```
!PRIV 0007
```

The following command causes your non-privileged job to execute at the slowest possible rate.

```
!PRIV 0070
```

If you execute the PRIV command with no argument, your current PRIV word is displayed. A sample of the PRIV command is

contained in Figure 6-1.

Figure 6-1
Sample Use of the PRIV Command

```
!PRIV 0023
!PRIV
0023
!
```

6.2.2 PEEK and POKE Commands

```
PE[EK] loc [n]
PO[KE] loc v1[v2[...]]
```

These commands are analogous to the EXAMINE and DEPOSIT commands (see 3.8, System User's Guide). However, EXAMINE and DEPOSIT deal with virtual memory, whereas PEEK and POKE deal with physical memory.

To examine physical memory, enter PEEK followed by the location whose contents you wish to examine (1-5 digits). If the location is five digits, the first digit is the instruction field. To examine a sequence of location, enter the starting address and the octal number of locations to be examined. When printing the contents, ETOS performs a carriage return, line feed after each location with a last digit of 7 (XXXX7). This assists the user in determining the contents of a variety of locations. The number of locations displayed must be less than 100 (base 8). A sample execution is shown in Figure 6-2.

Figure 6-2
Sample Use of the PEEK Command

```
!PEEK 10200
 7777
!PEEK 10200 15
 7777  7602  0354  1264  7664  3452  4017  2000
 0000  0000  1027  7700  3315
```

To deposit in physical memory, enter POKE, followed by the address at which to start depositing (1-5 digits), and the string of octal values which will go into sequential locations. If the start address is five digits, the first digit is the instruction field. If the start address is less than five digits, field Ø is modified. Figure 6-3 contains a sample use of the POKE Command.

Figure 6-3
Sample Use of the POKE Command

```
!PEEK Ø2Ø4
 7777
!POKE Ø2Ø4 2ØØ7 52Ø4 3216 7215 73ØØ 74Ø2
!PEEK Ø2Ø4 6
 2ØØ7 52Ø4 3216 7215
 73ØØ 74Ø2
!
```

If you POKE data into a field which does not exist in the physical machine, the data is lost. If you PEEK from a field which does not exist, the data returned is always "ØØØØ".

The POKE command should be used with extreme caution. When you POKE into fields Ø and 1, you are changing data in the ETOS resident monitor. This could cause the machine to crash or data to become corrupted. If you POKE into fields 2-7, you are changing data in user programs. This is not usually desirable. As a general rule, you should not use the POKE command unless the arguments were supplied by QUODATA or you have the source to the ETOS monitor.

6.2.3 BROADCAST Command

BRO[ADCAST] message

This command sends a message to all initialized terminals, simultaneously. A terminal is initialized when any key is struck on it while ETOS is running. The message can be any text up to

8Ø characters. The terminal does not have to be LOGged IN. After you execute the BROADCAST command, the following message is displayed on all terminals.

```
** KB n ** message
```

"n" is the console number which corresponds to the entry of the physical terminal interface in the IOT TABle (see 3.9).

The BROADCAST command should be utilized before you SHUTUP the system (see 6.2.7). This warns users that ETOS processing will be interrupted. A sample execution is shown in Figure 6-4.

Figure 6-4
Sample Use of the BROADCAST Command

```
!BROADCAST ETOS WILL BE SHUT DOWN IN 2 MINUTES
** KBØØ * ETOS WILL BE SHUT DOWN IN 2 MINUTES
!
```

6.2.4 FORCE and ECROF Commands

```
FO[RCE] n; command
EC[ROF] n
```

The FORCE command allows you to force a command on the user with the specified console number, "n". The command is entered at that user's terminal, exactly as if he had typed it himself. The console number corresponds to the entry of the physical terminal interface in the IOT TABle (see 3.9). The following rules define the use of FORCE.

1. A semicolon separates the console number from the text.
2. If the text is terminated by a line feed, the character before the line feed is the last character forced.
3. If a carriage RETURN terminates the text, a carriage RETURN will be forced.

4. The uparrow "^" character indicates that the next character is to be forced as a control character. For example, to force a CTRL/V enter uparrow V (^V), not CTRL/V.

5. The underscore "_" (or backarrow) indicates that the next character is to be taken literally. In order to force an uparrow, the uparrow must be preceded by an underscore (backarrow). To force an underscore, precede the underscore by an underscore, (_).

The FORCE command may be used to force a user to DEASSIGN a device which you wish to utilize, LOGOUT a user who is abusing his privileges, or to control the processing of a DETACHED terminal without ATTACHing to it. To FORCE a user on console 1 to LOGOUT from the system, enter the commands of Figure 6-5.

Figure 6-5
Sample Use of FORCE Command

```
!FORCE 1;^VS                    (^ = uparrow, not control key)
!FORCE 1;LOGOUT
!
```

ECROF (FORCE spelled backwards) can be used only on detached consoles, i.e., those console numbers not associated with physical terminals (see 3.9, System User's Guide). Its function is to empty the detached console n's output buffer into the user's output buffer. The ATTACH command yields the same result, except that after the ATTACH, the physical terminal is then associated with job n. If you attempt to use ECROF on a console which is not DETACHed, the message "?CONSOLE IN USE" is printed. To obtain the output of a DETACHed BASIC computational job associated with console 1Ø, enter the command of Figure 6-6.

Figure 6-6
Sample Use of the ECROF Command

```
!ECROF 10
TOTAL = 8.36
READY
```

6.2.5 Write Lock and Write Enable Commands

WE n
WL n

Initially, at LOGIN, channel 0 (OS/8 SYS:) is write locked and channels 1 and 2 are write enabled. If the user enters COS via the COSBO program, channel 0 (COS DK0) is write enabled. In order to change the status of the channels the above commands may be issued. The WE command write enables the specified channel, "n" and the WL command write locks a specified channel. These commands affect only logical write status, not the physical characteristics of the drive.

For more documentation on writing on OS/8 SYS:, see 5.13.7.

6.2.6 MOUNT and DISMOUNT commands

MO[UNT] DKn:PASSWORD,offset
DI[SMOUNT] DKn:

The two commands documented in this section are not actually PRIVileged commands. They are discussed in the Manager's Guide because they are system manager functions.

If you want to use packs other than the system disk for ETOS account storage, you must create a public pack. See 5.11 for details on this operation. Before you can utilize a public pack, you must logically mount it. The MOUNT command performs this function.

For RKØ5s, "n" in "DKn:" refers to the drive number (1-3). For System Industries disks, if n = 1, you're accessing the removable pack in port Ø; if n = 2, you're accessing the fixed pack in port 1; if n = 3, you're accessing the removable pack in port 1. "password" refers to the password of account [Ø,1] on that pack. If you rename the password from its current value of PASSWORD, use the new value in the MOUNT command. The offset for system packs or copies of system packs is 12ØØ. The offset for public packs is 1.

A sample execution of the MOUNT command is shown in Figure 6-7.

Figure 6-7
Sample Use of the MOUNT Command

```
!MOUNT DK1:PASSWORD,1          (public pack)
!
```

A disk does not need to be physically write enabled to be MOUNTed. You cannot mount a disk which is ASSIGNed (see 3.1Ø, System User's Guide).

There is a SYSCAL function which allows you to MOUNT a disk via an assembler program. This command corresponds exactly to its corresponding SYSCAL in argument structure (see 6.3.5). Rather than specifying the arguments within your program, the SCALE command accepts arguments in the command line.

MOUNT error messages are returned in the form "n? FUNCTION FAILURE" where "n" corresponds to the error returned in the AC by the equivalent SYSCAL. A list of errors are given in Table 6-2.

Table 6-2
MOUNT Error Messages

Error	Meaning
1	Nonexistent device
2	Device already MOUNTed or ASSIGNed
3	Wrong password

The DISMOUNT command is used to release a disk drive on which an ETOS file-structured device is MOUNTed. After dismounting a pack logically and removing the pack, you can insert another pack and ASSIGN it or MOUNT it. A pack may not be DISMOUNTed if any files are open on the device by any user. This is the most common error and it can be frustrating to track down and close all channels with files open on a particular pack.

This command corresponds exactly to its corresponding SYSCAL in argument structure (see 6.3.7). Rather than specifying the arguments within your program, the SCALE command accepts arguments in the command line. Error messages listed in Table 6-3 are returned in the form "n? FUNCTION FAILURE" where "n" corresponds to the error code returned in the AC by the equivalent SYSCAL.

Table 6-3
DISMOUNT Error Messages

Error	Meaning
1	Nonexistent device
2	Device not MOUNTed
3	File open on device

A sample execution of the DISMOUNT command is shown in Figure 6-8.

Figure 6-8
Sample Use of the DISMOUNT Command

```
!DISMOUNT DK1:
!
```

6.2.7 SHUTUP Command

SH[UTUP]

This command is used to shut down time sharing mode and return to stand-alone OS/8. After this command is executed, the only active terminal is the console terminal. See 4.7 for more information about shutting down the system.

6.3 PRIVILEGED SYSCAL FUNCTIONS

SYSCAL (6200) has five functions which can be used only in the privileged mode. See 6.7, System User's Guide for a full description of SYSCAL's other functions.

6.3.1 LOCK (Function 4)

This function LOCKs the user's virtual instruction and data fields into physical memory. These fields will be memory resident until an UNLOCK (see 6.3.2) is performed. The AC may be any value on entry, and the AC is cleared upon return. A sample calling sequence is shown in Figure 6-9.

Figure 6-9
Sample Use of the LOCK Function

```
SYSCAL=6200
LOCK=4                          /LOCK IS FUNCTION 4
LOKFLD=X                        /DATA FIELD TO LOCK IN
FLDCHG=LOKFLD^10                /ARGUMENT FOR CDF
TAD        (LOCK                /MQ = FUNCTION
MQL                             /AC IS IGNORED
CDF FLDCHG                      /ADDITIONAL FIELD TO LOCK IN CORE
SYSCAL
/ AC = 0 UPON RETURN
```

You must insure that a minimum of 16K words is left unLOCKed. 8K words are used for the ETOS resident monitor and 8K words are used for other jobs.

6.3.2 UNLOCK (Function 5)

UNLOCK applies to all fields assigned to the calling user, which are currently locked in core. Figure 6-10 contains a simple calling sequence.

Figure 6-10
Sample Use of the UNLOCK Function

```
SYSCAL=6200
UNLOCK=5                        /UNLOCK IS FUNCTION 5
TAD       (UNLOCK
MQL
SYSCAL
/ AC = 0 UPON RETURN
```

6.3.3 HOOK (Function 6)

This command performs a LOCK and HOOKs the user's instruction field into the resident monitor's interrupt skip chain. Only one job may be hooked into the resident system at any given time. An attempt to hook more than one job will result in the error message "ILLEGAL IOT" being displayed. If an interrupt occurs while a real-time task is in core, the monitor preserves the

registers (AC and FLAGS) and JMS's to location Ø in the calling field.

If the user's interrupt service will not use the clock, a -1 must be loaded into the user's AC before the HOOK SYSCAL is executed.

For a more lengthy description of using the HOOK function see Chapter 7. A sample calling sequence is shown in Figure 6-11.

Figure 6-11
Sample Use of the HOOK Function

```
SYSCAL=6200
HOOK=6                      /HOOK IS FUNCTION 6
CLA CLL
TAD       (HOOK
MQL
CLA CMA                     /USER NOT USING CLOCK
SYSCAL
```

6.3.4 UNHOOK (Function 7)

This command UNHOOKs the user's instruction field from the resident monitor's interrupt skip chain and performs the UNLOCK function. Figure 6-12 contains a sample calling sequence.

Figure 6-12
Sample Use of the UNHOOK Function

```
SYSCAL=6200
UNHOOK=7                    /UNHOOK IS FUNCTION 7
CLA CLL
TAD      (UNHOOK
MQL
SYSCAL
```

6.3.5 MOUNT (Function 17)

Mount an ETOS file-structured public pack. The pack becomes available to all users. This function is equivalent to SCALE's MOUNT command (see 6.2.6). The MQ contains the function number. The AC contains a pointer to a ten-word argument block in the user's data field. AC = (block address) minus one. The block is of the form

Word 1-2 = Name of the device the public pack will be mounted on. The name is in packed 6-bit ASCII.

Word 3 = Master file directory account on the public pack. This is always equal to 0001.

Word 4 = Pack identification account on the public pack. This is always equal to 0001.

Word 5-8 = Pack identification. This is normally "PASSWORD" unless changed by the system manager.

Word 9-10 = Offset for MOUNT. These two words make up a double precision offset, which is 00000001 for public packs and 00001200 for system packs. Word 9 is the high order portion of the offset.

An error code is returned in the AC in accordance with Table 6-4.

Table 6-4
Error Codes for the MOUNT Function

AC	Meaning
0	MOUNT succeeded
1	Nonexistent device
2	Device already mounted or owned by other user
3	Wrong I.D.

Figure 6-13 contains a sample calling sequence.

Figure 6-13
Sample Use of the MOUNT Function

```
SYSCAL=6200
MOUNT=17                        /MOUNT IS FUNCTION 17
ARGFLD=Y                        /FIELD CONTAINING ARGUMENT BLOCK
FLDCHG=ARGFLD^10                /ARGUMENT FOR CDF
.
.
.
TAD             (MOUNT          /MQ = FUNCTION
MQL
TAD             (ARGBLK-1       /AC = (ADDRESS OF ARG BLOCK) - 1
CDF FLDCHG
SYSCAL                          /MOUNT A PUBLIC PACK
SZA                             /CHECK ERROR IN AC
JMP             ERROR           /ERROR ROUTINE
.
.
.
FIELD ARGFLD
ARGBLK,         DEVICE DEVI     /DEVICE NAME
                1               /MFD ACCOUNT
                1               /PACK ID ACCOUNT
                TEXT "PASSWORD" /PACK ID
                *.-1
                0000            /HIGH ORDER OFFSET
                0001            /LOW ORDER OFFSET
```

6.3.6 XTRAP (Function 21)

In the ETOS V5A monitor, this function put the user into DEBUG mode. Since this mode is not supported under V5B, this function is now an ILLEGAL IOT.

6.3.7 DISMNT (Function 23)

Dismount a public pack, making the drive available for other use. This function is equivalent to SCALE's DISMOUNT command (see 6.2.6). The AC points to a 2-word argument block in the user's data field that contains the device name to dismount. AC = (address of block)-1.

Table 6-5 contains the errors which are returned in the AC.

Table 6-5
Error Codes for the DISMOUNT Function

AC	Meaning
Ø	DISMNT succeeded
1	Nonexistent device
2	Device not MOUNTed
3	File open on device

A sample calling sequence is given in Figure 6-14.

Figure 6-14
Sample Use of the DISMNT Function

```
SYSCAL=6200
DISMNT=23                       /DISMNT IS FUNCTION 23
ARGFLD=Y                        /FIELD CONTAINING ARGUMENT BLOCK
FLDCHG=ARGFLD^10                /ARGUMENT FOR CDG
.
.
.
TAD          (DISMNT            /MQ = FUNCTION
MQL
TAD          (ARGBLK-1          /AC = POINTER TO ARGUMENT BLOCK
CDF FLDCHG
SYSCAL                          /DISMOUNT PUBLIC PACK
SZA                             /CHECK ERROR CODE IN AC
JMP          ERROR              /ERROR ROUTINE
.
.
.
FIELD ARGFLD
ARGBLK, DEVICE DEVI             /DEVICE NAME TO DISMOUNT
```

6.4 PRIVILEGED OS/8 FUNCTIONS

Certain functions which are performed under OS/8 require you to be privileged, i.e., PRIV 40XX. Any access of the TD8E DECtape, TCØ8 DECtape, RXØ1 floppy disk or PT8E high speed reader/punch is privileged. Changing terminal or line printer attributes is privileged. Execution of system ETOS cusps (e.g., line printer spooler) is privileged. Since OS/8 SYS is write-protected, any process which writes on SYS must use the "WE" SCALE command and must therefore be privileged. The following sections list all standard OS/8 privileged operations.

6.4.1 Changing Device Handlers in OS/8

When ETOS is distributed, certain device handlers are activated under stand-alone OS/8 and virtual OS/8. You may wish to activate other handlers for your installation. A list of the distributed handlers is provided in section 1.3.

To change device handlers under stand-alone OS/8 use the following procedure.

1) Boot OS/8 on the disk, using the procedure outlined in 2.2 (RKØ5) or 2.3 (S.I.).

2) Use the dialogue of Figure 6-15 to print current device handlers.

Figure 6-15
Printing the Current Device Handlers

```
.RUN SYS:BUILD
$PRINT
```

(On an RKØ5 system, the following is printed)

```
ETOS:*SYS
RKØ5:RKAØ *RKBØ *RKA1 *RKB1 RKA2 RKB2 RKA3 RKB3
RXØ1:*RXAØ *RXA1
TD8A:*DTAØ DTA1
TC:DTAØ DTA1 DTA2 DTA3 DTA4 DTA5 DTA6 DTA7
RF: RF NULL
BAT:BAT
CR8E:*CDR
PT8E:*HSP *HSR
KL8E:*TTY
KS33:*PTP *PTR
DUMP:DUMP
VT5Ø:LST
LQP:LPT
TM8E:MTAØ MTA1 MTA2 MTA3 MTA4 MTA5 MTA6 MTA7

TA8A:CSAØ CSA1
LPSV:*LPT
DSK=ETOS:SYS
$
```

Figure 6-15 (continued)

(On a System Industries disk, the following is printed)

```
ETOS:*SYS DSKØ *DSKl *DSK2 *DSK3
RKØ5:*RKAØ *RKBØ RKAl RKBl RKA2 RKB2 RKA3 RKB3
RXØ1:*RXAØ *RXAl
TD8A:*DTAØ DTAl
TC:DTAØ DTAl DTA2 DTA3 DTA4 DTA5 DTA6 DTA7
RK:RF NULL
BAT:BAT
CR8E:*CDR
PT8E:HSP HSR
KL8E:*TTY
KS33:PTP PTR
DUMP:DUMP
VT5Ø:LST
LQP:LPT
TM8B:MTAØ MTAl MTA2 MTA3 MTA4 MTA5 MTA6 MTA7
TA8A:CSAØ CSAl
LPSV:*LPT
SI4T:SYS DSKØ DSKl DSK2 DSK3
SINS:LPT *DSK4 DSK5 DSK6 DSK7

DKS=ETOS:SYS
$
```

All device handlers loaded into BUILD are printed. All active device handlers are indicated with an asterisk (*) preceding the handler name. A list of all device handlers loaded into BUILD for RKØ5 and System Industries disks is contained in Table 6-6.

Table 6-6
Stand-alone OS/8 Device Handlers

Group Name	Handlers in Group	Description
BAT	BAT	1-page handler, which can be called from a BATCH job to read from the batch stream.
CR8E	CDR	2-page card reader handler.
DUMP	DUMP	1-page handler which allows you to dump the contents of an OS/8 mass storage device.
ETOS	SYS DSKØ DSK1 DKS2 DSK3	1-page RKØ5 system handler modified to be device type 4Ø. System Industries handler modified to be device type 4Ø. DSK1-3 access stand-alone portions of the system disk. Since the ETOS disk has only one 12ØØ (base 8) portion, these handlers are used only for the BACKUP program.
KL8E	TTY	2-page terminal input/output handler.
KS33	PTP PTR	2-page low speed paper tape punch and reader handlers.
LPSV	LPT	1-page line printer handler.
LQP	LPT	2-page letter quality printer handler.
PT8E	HSP HSR	2-page high speed paper tape punch and reader handlers.
RF	RF NULL	RFØ8 fixed head disk and NULL device handlers.
RKØ5	RKAØ RKBØ RKA1 RKB1 RKA2 RKB2 RKA3 RKB3	2-page non-system RKØ5 handlers for drives Ø-3.

Figure 6-6 (continued)

Group Name	Handlers in Group	Description
RX01	RXA0 RXA1	2-page handler for two RX01 floppy drives.
SINS	LPT DSK4 DSK5 DSK6 DSK7	System Industries non-system handlers. If ETOS SYS is enabled, those handlers access OS/8 portions of removable disk in port 0. If SI4T SYS is enabled, these handlers access OS/8 portions of fixed disk in port 0.
SI4T	SYS DSK0 DSK1 DSK2 DSK3	Handlers which allow you to run OS/8 off the removable System Industries disk in port 0.
TA8A	CSA0 CSA1	2-page cassette handlers.
TC	DTA0 DTA1 DTA2 DTA3 DTA4 DTA5 DTA6 DTA7	2-page TC08 dectape handlers.
TD8A	DTA0 DTA1	2-page TD8E dectape handlers.
TM8E	MTA0 MTA1 MTA2 MTA3	2-page industry standard magnetic tape handlers.
VT50	LST	2-page handler which allows VT50 display processing.

In addition to the device handlers listed in Table 6-6, DSK: is active. This default device is used whenever DSK: is specified or when the device name is omitted. It is set up as SYS on the distribution pack but any device may be ASSIGNed to DSK.

OS/8 allows only fifteen device handlers to be active. Therefore, if you wish to activate a handler, you must first deactivate an existing handler. For example, the dialogue of Figure 6-16 can be used to activate the TC08 DECtape handler.

Figure 6-16
Activation of Resident Device Handler

```
$DEL DTA0          (deactivates TD8E handler)
$IN TC DTA0        (activates TC08 handler)
$
```

If you wish to activate a handler which is not displayed with the PRINT command, you must first LOAD the handler. BUILD has a limited space for device handlers. Therefore, you must unload an existing handler. For example, the dialogue of Figure 6-17 can be used to activate a DF32 disk handler.

Figure 6-17
Activation of Non-resident Device Handler

```
$UNLOAD TA8A              (unload cassette handler)
$LOAD dev:DF32            (handler on device called "dev")
$IN DF DF
$
```

3) After you have made all desired changes to device handlers, enter the commands of Figure 6-18.

Figure 6-18
BOOTing a Revised OS/8 System

```
$BOOT                     (boots OS/8 with revised system)
SYS BUILT
.SAVE SYS:BUILD           (saves changes in BUILD. If you do not
                          have 33 free blocks on SYS, you must
                          SAVE BUILD on another device and copy it
                          back to the system device).
.
```

4) The file ETOSOS.SY contains a copy of the distribution OS/8 head. If you wish to change this file to reflect changes made in device handler selection, use the dialogue of Figure 6-19.

Figure 6-19
Updating the Stand-alone OS/8 System File

```
.R PIP
*SYS:ETOSOS.SV<SYS:/Y/D
*^C
.
```

To change device handlers under virtual OS/8, use the following procedure.

1) Initiate ETOS via the procedures of 4.5.

2) LOGIN to a privileged account. The operator's account is one privileged account.

3) Enter the commands of Figure 6-20.

Figure 6-20
Printing the Current Virtual Device Handlers

```
.^VS
!PRIV 4040
!WE 0
!CONT
RUN SYS:BUILD
$PRINT
```

The following output is printed on the terminal.

```
ETOS:*SYS CHN0 CHN1 *CHN2 *CHN3 *CHN4 *CHN5 CHN6 CHN7
ETRX:*RXA0 RXA1
ETTD:*DTA0 DTA1
ETLP:*LPT
ETKS:*PTP *PTR
ETPT:HSP HSR
TC01:DTA0 DTA1
TC23:DTA2 DTA3
TC45:DTA4 DTA5
TC67:DTA6 DTA7
ETCR:*CDR
SPLP:*QLP
ETKL:*TTY
BAT:BAT
DUMP:DUMP
DSK=ETOS:CHN2
$
```

All device handlers loaded into BUILD are printed. All active device handlers are indicated with an asterisk (*) preceding the handler name. A list of all device handlers loaded into BUILD are listed in Table 6-7.

Table 6-7
Virtual OS/8 Device Handlers

Group Name	Handlers in Group	Description
BAT	BAT	1-page handler, which can be called from a BATCH job to read from the batch stream.
DUMP	DUMP	1-page handler which allows you to dump the contents of an OS/8 mass storage device.
ETCR	CDR	ETOS version of the OS/8 card reader handler.
ETKL	TTY	ETOS version of the OS/8 terminal handler. All SET TTY functions may be used with this handler.
ETLP	LPT	ETOS version of the line printer handler.
ETLS	PTP, PTR	ETOS version of the line printer handler.
ETLT	LT	Handler which simulates line printer characteristics on a terminal.
ETOS	SYS CHNØ CHN1 CHN2 CHN3 CHN4 CHN5 CHN6 CHN7	ETOS system device handler (SYS) and channel handlers (CHNX:).
ETPT	HSP HSR	ETOS versions of the high speed paper tape punch and reader handlers.
ETRX	RXAØ RXA1	ETOS versions of the RXØ1 floppy disk handlers.
ETTD	DTAØ DTA1	ETOS versions of the TD8E DECtape handlers.
ETTY	TTY	ETOS terminal handler, which does not allowo SET TTY functions but does allow different margins on each terminal.
SPLP	QLP	2-page spooling device handler.

Table 6-7 (continued)

Group Name	Handlers in Group	Description
TCØ1	DTAØ DTA1	ETOS versions of handlers for TCØ8 DECtape drives Ø and 1.
TC23	DTA2 DTA3	ETOS versions of handlers for TCØ8 DECtape drives 2 and 3.
TC45	DTA4 DTA5	ETOS versions of handlers for TCØ8, DECtape drives 4 and 5.
TC67	DTA6 DTA7	ETOS versions of handlers for TCØ8 DECtape drives 6 and 7.

In addition to the device handlers listed above, DSK: is also active. This default device is used whenever DSK: is specified or when the device name is omitted. It is set up as CHN2 on the distribution pack, but any device may be ASSIGNed to DSK.

OS/8 allows only fifteen device handlers to be active. Therefore, if you wish to activate a handler, you must first deactivate an existing handler. For example, the dialogue of Figure 6-21 could be used to activate the TCØ8 DECtape handler.

Figure 6-21
Activation of Resident Virtual Handler

```
$DEL DTAØ          (deactivates TD8E handler)
$IN TCØ1 DTAØ      (activates TCØ8 handler)
$
```

If you wish to activate a handler which is not displayed with the PRINT command, you must first LOAD the handler. BUILD has a limited space for device handlers. Therefore, you must unload an existing handler. For example, the dialogue of Figure 6-22 could be used to activate an ETOS DF32 handler (if one existed).

Figure 6-22
Activation of Non-resident Virtual Handler

```
$UNLOAD TCØ1            (unload TCØ8 handler)
$LOAD dev:ETDF          (handler ETDF on device called "dev:")
$IN ETDF DF
$
```

4) After you have made all desired changes to device handlers, enter the commands of Figure 6-23.

Figure 6-23
BOOTing a Revised Virtual OS/8 System

```
$BOOT                   (boots virtual OS/8 with revised system)
SYS BUILT
.SAVE SYS:BUILD         (saves changes in BUILD.  If you do not
                         have 33 free blocks on SYS, you must save
                         BUILD on another device and COPY it back
                         to the system device).
.
```

5) The file ETOSET.SY, which resides on the stand-alone OS/8 portion of the ETOS pack, contains a copy of the distribution virtual OS/8. If you wish to change this file to reflect changes made in device handler selection, use the dialogue of Figure 6-24.

Figure 6-24
Updating the Virtual OS/8 System File

```
.^VS
!CLOSE 3
!CLOSE 4
!LOOKUP 3=[Ø,2]OS8.OS8
!LOOKUP 4=[Ø,2]OS8.RTS
!WE 3
!CONT
R PIP
*CHN3:ETOSET.SY<CHN4:/Y/D
*^C
.
```

In the previous dialogue, the system device ([Ø,2]OS8.RTS) was looked up on channel 4. The reason that you can't copy directly from SYS is that ETOS maps portions of OS/8 SYS to the scratch blocks. Looking OS8.RTS on channel 4 circumvents this problem. It is important to update the file ETOSET.SY if you wish to make

device handler changes permanent. This file is transferred to OS8.RTS if the "H" option is used in start-up (see 4.3).

6.4.2 TD8E DECtape

The TD8E DECtape is not a data break device. It cannot be used simultaneously with timesharing. However, it is often desireable to store or retrieve files on the DECtape without shutting down ETOS. QUODATA supports this device via two OS/8 device handlers (DTAØ and DTA1) in the following manner. Each time that a DECtape record is read or written, timesharing is stopped for an average of 27 milliseconds. This period is so short that it is not normally noticed. However, this device should be used with extreme caution. It can cause subtle problems, depending upon what other users are doing. There are certain definite restrictions in the use of DECtape: the job must be privileged; only one job at a time can access the TD8E; the DECtape should not be used at the same time as the card reader or the card reader will lose characters. Each time the DECtape is used, the ETOS time of day loses a fraction of a second. Users performing terminal input may lose characters, which will not be echoed. A job running the DECtape should never be stopped with double ^C or ^VS. This causes the DECtape to spin off onto one reel. If this occurs, each attempt to mount a DECtape will result in the tape spinning off the reel. The best cure for this situation is to run TDSTOP.

The program TDSTOP.SV is a utility program which operates under ETOS. An entry of ^VS;BOOT;R TDSTOP<RET> halts TD8E drives Ø and 1 almost instantly. Like the TD8E handler itself, TDSTOP is a privileged program.

In the example of Figure 6-25, a file called FILE1.EX, is written to the TD8E DECtape.

Figure 6-25
Sample Access of the TD8E DECtape

```
.^VPRIV 4040
COPY DTA0:<FILE1.EX
FILES COPIED:
FILE1.EX
^VPRIV 40
```

6.4.3 TC08 DECtape

The TC08 DECtape is a data break device. However, unlike the card reader, disk and line printer, this device is not supported by the ETOS resident monitor. The TC08 is supported by eight OS/8 device handlers (DTA0-7) which stop timesharing for short periods of time and execute the DECtape IOTs directly. These periods are so short that they are not normally noticed. However, this device should be used with caution. Its use can cause subtle problems, depending upon what other users are doing. There are certain restrictions in the use of the TC08 the job accessing the TC08 must be privileged; only one job at a time can access the TC08; it must access the DECtape via the OS/8 device handler, not an internal handler. The DECtape should not be used at the same time as the card reader or the card reader will lose characters. Each time the DECtape is used, the ETOS time of day loses a fraction of a second. In the example of Figure 6-26, a file called FILE1.EX is written to the TC08 DECtape.

Figure 6-26
Sample Acess of the TC08 DECtape

```
.^VPRIV 4040
COPY DTA0:<FILE1.EX
FILES COPIED:
FILE1.EX
.^VPRIV 0040
```

When the system is distributed, the TC08 DECtape is not enabled under stand-alone or virtual OS/8. To disable TD8E DECtape and

enable TC08 DECtape, follow the procedures of 6.4.1.

6.4.4 RX01 Floppy Disk

The RX01 floppy disk is not a data break device. Therefore, it cannot be operated in a true timesharing mode. The RX01 single density floppy disk is supported by two OS/8 device handlers (RXA0 and RXA1) which stop timesharing for short periods of time and executes the RX01 IOTs directly. These periods are so short that they are not normally noticed. However, this device should be used with caution. Its use may cause subtle problems, depending on what other users are doing. There are certain restrictions in the use of the RX01, the job accessing the RX01 must be privileged; only one job at a time can access the RX01; it must access the floppy disk via the OS/8 device handler, not an internal handler. The RX01 should not be used at the same time as the card reader or the card reader will lose characters. Each time the RX01 is used, the ETOS time of day loses a fraction of a second. In the example of Figure 6-27, a file called FILE1.EX is written to the single density floppy disk drive 0.

Figure 6-27
Sample Access of the RX01 Floppy Disk

```
.^VPRIV 4040
COPY RXA0:<FILE1.EX
FILES COPIED:
FILE1.EX
.^VPRIV 0040
```

6.4.5 PT8E High Speed Reader/Punch

The PT8E high speed reader/punch is not supported by the ETOS resident monitor. The PT8E is supported by two OS/8 device handlers (HSP and HSR) which stop time sharing for short period of time and execute the reader/punch IOTs directly. These periods are so short that they are not normally noticed.

However, this device should be used with caution. Its use may cause subtle problems, depending on what other users are doing. There are certain restrictions in the use of the PT8E: the job accessing the PT8E must be privileged; only one job at a time can access the PT8E; it must access the reader/punch via the OS/8 device handler, not an internal handler. The PT8E should not be used at the same time as the card reader or the card reader will lose characters. Each time the PT8E is used, the ETOS time of day loses a fraction of a second. In the example of Figure 6-28, a file called FILE1.EX is read from the high speed reader and a file called FILE2.EX is output to the high speed punch.

Figure 6-28
Sample Access of High Speed Reader/Punch

```
.^VPRIV 4040
R PIP
*FILE1.EX<HSR:
*HSP:<FILE2.EX
*^C
.^VPRIV 0040
```

6.4.6 User-Written Device Handlers

There are two basic methods of implementing user device handlers into the ETOS system. One method is discussed in Chapter 7. In this procedure, the device handler acts as a job which may hook itself into the ETOS interrupt system to handle interrupt requests. That method of treating a device handler as a job could also be used for non-interrupt peripherals.

The second method for implementing user device handlers is to build in the handler as an OS/8 handler (e.g., LPT, HSP, etc.). Since the virtual OS/8 system currently contains the maximum number of handlers allowable, one existing handler needs to be deleted for every user implemented device added. The IOTs for the user's peripheral are not defined in the resident ETOS

monitor. Therefore, the device handler will have to stop ETOS for very short periods of time and use the machine as if it were running stand-alone.

Because the handler is not hooked into the ETOS interrupt system, the device to be operated must not generate interrupts at any time. If the device has an interrupt enable/disable facility, be sure that it is set to the interrupt disable state. To operate the device, the job running the program must be privileged. This allows the handler to clear the user mode flip/-flop via the CUF IOT.

It is very important that the you fully understand the characteristics of running with the user mode disabled.

First, all IOTs, CDFs, HLTs, OSRs and RDFs will be executed as they are on a stand-alone machine.

Second, the user should be careful in changing data or instruction fields with the user flag cleared because the CDFs and CIFs will go to real space instead of virtual space. The real fields may be relocated due to swapping at any time so don't rely on a RDF or RIF to tell you where you are.

Third, to re-enter user mode, use the SUF IOT but note that the user flag is not set until the next JMP or JMS following the SUF IOT.

Fourth, the scheduler may set the data field equal to the last value it had in user mode, even if a swap doesn't occur. Therefore, if you intend to use the CUF so that a program may examine or modify the resident monitor, briefly turn the interrupt system off while changing the data field to that of the resident monitor. This is illustrated in Figure 6-29.

Figure 6-29
Sample Use of CUF IOT

```
/ROUTINE TO LOOK AT ETOS MONITOR LOCATION.
/ENTRY ASSUMES POINTER CONTAINS THE ETOS LOCATION
/TO PEEK AT.  AC SHOULD BE ZERO, AND TEMP AND POINTER
/SHOULD BE ON THE CURRENT PAGE.  ROUTINE EXISTS WITH
/VIRTUAL DATA FIELD=VIRTUAL INSTRUCTION FIELD.  AC
/CONTAINS THE CONTENTS OF THE ETOS LOCATION.

CDF N              /N = VIRTUAL INSTRUCTION FIELD
CUF                /CLEAR USER FLAG
IOF                /DISABLE INTERRUPT, SO NO SWAPPING
CDF X              /X = REAL FIELD OF INTEREST
TAD I    POINTER   /FETCH DATA WORD FROM MONITOR
DCA      TEMP      /SAVE THE DATA
RIF                /FIND OUT WHERE WE ARE
TAD      (CDF      /CONSTRUCT A PROPER CDF
DCA      .+1       /PREPARE TO EXECUTE CDF
NOP                /SET DF TO IF
ION                /REENABLE INTERRUPT SYSTEM
SUF                /SET USER FLAG ON NEXT JMP/JMS
JMP      .+1       /FORCE USER FLAG ON
TAD      TEMP      /DATA WORD
```

A set of subroutines contained in Figure 6-30, which control the XY8-E incremental plotter, is presented as an example of the preceding explanation. These subroutines do not have to turn the interrupt off, since the hardware is capable of disabling its own interrupt facility, via a special software IOT.

Figure 6-30
Sample User-written Subroutines

```
SETUP,  0               /SET UP PLOTTER
        CUF             /CLEAR USER FLAG
        PLCE            /CLEAR PLOTTER FLAG
        SUF             /SET USER FLAG ON NEXT JMP/JMS
        JMP I   SETUP   /RETURN WITH USER FLAG ON
/
XPLSF,  0               /SKIP ON PRINTER FLAG
        CUF             /CLEAR USER FLAG
        PLSF            /PLOTTER FLAG UP?
        SKP
        ISZ     XPLSF   /YES, BUMP RETURN
        SUF             /SET USER FLAG ON NEXT JMP/JMS
        JMP I   XPLSF
/
/
XPLCF,  0               /CLEAR PLOTTER FLAG
        CUF             /CLEAR USER FLAG
        PLCF            /CLEAR PLOTTER FLAG
        SUF             /SET USER FLAG ON NEXT JMP/JMS
        JMP I   XPLCF
/
/
XPLPU,  0               /PUT THE PEN UP
        CUF             /CLEAR USER FLAG
        PLPU            /PEN UP
        SUF             /SET USER FLAG ON NEXT JMP/JMS
        JMP I   XPLPU
/
/
XPLLR,  0               /MOVE PEN RIGHT OR LEFT
        CUF             /CLEAR USER FLAG
        PLLR            /LOAD DIRECTION REGISTER
        SUF             /SET USER FLAG ON NEXT JMP/JMS
        JMP I   XPLLR
/
/
```

The device handler must be in the standard format for an OS/8 device handler (see OS/8 Software Support Manual). An example of a handler which has been implemented in the preceding manner is the high speed paper tape reader/punch. A listing of this handler is given in Figure 6-31.

Figure 6-31
Sample User-written Device Handler

```
                /HI SPEED READER/PUNCH HANDLER FOR ETOS
                /ETPT V2.00  12:38 AM 3/22/79

      0000              *0
00000 7776              -2
00001 0524      DEVICE ETPT;DEVICE PTP;1020;PIP&177;ZBLOCK 2
00002 2024
00003 2024
00004 2000
00005 1020
00006 0000
00007 0000
00011 0524      DEVICE ETPT;DEVICE PTR;2010;PTR&177;ZBLOCK 2
00012 2024
00013 2024
00014 2200
00015 2010
00016 0106
00017 0000

      6264              CUF=6264        /CLEAR USER FLAG
      6274              SUF=6274        /SET USER FLAG
      0001      PTVERSION="A&77

      0200              *200
                        /THIS IS THE REAL HIGH SPEED HANDLER.
                        /HIGH SPEED PAPER TAPE HANDLER FOR BLEEP
                        /MONITOR
                        /PACKS 3 CHARACTERS IN 2 WORDS ON INPUT,
                        /UNPACKS ON OUTPUT
                        /PAGE RELOCATABLE
00200 0001      PIP,    PTVERSION
00201 7320              CLA CLL CML     /SET LINK ON TO INDICATE
                                        /PUNCH
00202 4223              JMS PSETUP      /DO COMMON STUFF
00203 1711      PTPLP,  TAD I PTPCA
00204 4300              JMS PTPPCH      /FIRST CHAR IN LOW ORDER 8
                                        /BITS OF WORD 1
00205 3306              DCA PTR
00206 2371              ISZ PIPCA
00207 7700      PT7700, 7700
00210 1771              TAD I PTPCA
00211 4300              JMS PTPPCH      /SECOND CHAR IN LOW ORDER 8
                                        /BITS OF WORD 2
00212 7112              CLL RTR
00213 7012              RTR
00214 1306              TAD PTR
00215 7012              RTR
```

Figure 6-31 (continued)

```
00216  7012            RTR             /THIRD CHARACTER NOW IN AC
00217  4300            JMS PTPPCH
00220  2371   PTPEND,  ISZ PTPCA
00221  0070   PT70,    70              /JUST IN CASE WERE PUNCHING PG 7600
                                       /KEEP THIS LITERAL HERE
00222  5663            JMP PTPISZ

00223  0000   PSETUP,  0
00224  6214            RDF             /GET FIELD OF CALLING PROGRAM
00225  1367            TAD PTPCIF
00226  3273            DCA PTPXIT          /SET UP RETURN SEQUENCE
00227  1600            TAD I PIP
00230  0221            AND PT70
00231  1366            TAD PCDF
00232  3247            DCA PTPCDF
00233  7010            RAR                 /GET LINK(1=PTP,0=PTR)
00234  1600            TAD I PIP           /GET FUNCTION WORD
00235  2200            ISZ PTP
00236  7510            SPA                 /CHECK CORRECT MODE
00237  5364            JMP PTPERR          /SIGNAL "UNRECOVERABLE
                                           /DEVICE ERROR"
00240  0207            AND PT7700
00241  7040            CMA                 /SET UP -(WORD COUNT)/2-1
00242  3370            DCA PTPWC
00243  1600            TAD I PTP           /SET UP STARTING ADDRESS
00244  2200            ISZ PTP
00245  3371            DCA PTPCA
00246  1600            TAD I PTP
00247  0000   PTPCDF,  0                   /SET DATA FIELD TO ACCESS
                                           /BUFFER
       0247            PTPEOF=PTPCDF
00250  7640            SZA CLA             /BLOCK 0?
00251  5262            JMP INICUF          /NO: SKIP BLOCK 0 INITIAL-
                                           /IZATION
00252  7630            SZL CLA             /READ OR WRITE?
00253  5275            JMP INIPTP          /WRITE
00254  1354            TAD PTP336          /READ: PRINT " ";ON CONSOLE
00255  6046            TLS
00256  6036            KRB                 /READ AND THROW OUT A CHAR
00257  6264            CUF                 /CLEAR USER FLAG
               /******NOW IN EXECUTIVE MODE******
00260  6014            RFC                 /FETCH NEXT CHAR FROM READER
00261  4340            JMS PTRGCH          /INITIALIZE THE ONE CHAR
                                           /BUFFER JUST BARELY OK TO
                                           /LEAVE JUNK IN AC
00262  6264   INICUF,  CUF                 /FOR BLOCKS OTHER THAN 0
               /******NOW IN EXECUTIVE MODE******
00263  2370   PTPISZ,  ISZ PTPWC
00264  5623            JMP I PSETUP        /LOOP FOR BUFFER SIZE (128
                                           /WORDS)
00265  6274            SUF                 /SET USER FLAG
00266  5267            JMP .+1             /SUF WAITS UNTIL NEXT JUMP
```

Figure 6-31 (continued)

```
                /******NOW BACK IN USER MODE******
00267   1247            TAD PTPEOF
00270   7640            SZA CLA             /DID WE RUN OUT OF TAPE?
00271   2200    FATAL,  ISZ PTP
00272   2200            ISZ PTP
00273   7402    PTPXIT, HLT             /EXIT CDF GOES HERE
00274   5600            JMP I PTP

00275   6264    INIPTP, CUF                 /CLEAR USER FLAG
                /******NOW IN EXECUTIVE MODE******
00276   6026            PLS                 /INITIALIZE PUNCH
00277   5263            JMP PTPISZ

00300   0000    PTPPCH, 0
00301   6021            PSF                 /ISN'T THIS A LOVELY COM-
                                            /PUTE BOUND LOOP TO BE
00302   5301            JMP .-1             /DOING UNDER TIMESHARING?
00303   6026            PLS
00304   0335            AND PT7400
00305   5700            JMP I PTPPCH

00306   0001    PTR,    PTVERSION           /***CORRECT ENTRY IN MAIN
                                            /ASSEMBLY IF THIS IS MOVED!

                                            /WHAT DOES THE ABOVE COMMENT
                                            /MEAN?
                                            /PTR ENTRY POINT WAS 112.
00307   7300            CLA CLL
00310   1306            TAD PTR
00311   3200            DCA PTP
00312   4223            JMS PSETUP          /SET UP ADDRESS, COUNT, FIELDS
00313   4340    PTRLP,  JMS PTRGCH          /READ FIRST CHARACTER OF 3
00314   3771            DCA I PTPCA
00315   4340            JMS PTRGCH
00316   3300            DCA PTPPCH
00317   4340            JMS PTRGCH
00320   7006            RTL
00321   7006            RTL
00322   3306            DCA PTR
00323   1306            TAD PTR
00324   0335            AND PT7400
00325   1771            TAD I PTPCA
00326   3771            DCA I PTPCA         /HIGH ORDER 4 BITS INTO WORD 1
00327   1306            TAD PTR             /GET THE CHAR FROM THE PTR
                                            /BUFFER
00330   7006            RTL
00331   7006            RTL
00332   0335            AND PT7400
00333   1300            TAD PTPPCH
00334   2371            ISZ PTPCA
00335   7400    PT7400, 7400
00336   3771            DCA I PTPCA         /LOW ORDER 4 BITS INTO WORD 2
```

Figure 6-31 (continued)

```
00337  5220           JMP PTPEND

00340  0000   PTRGCH, 0
00341  1247           TAD PTPEOF
00342  7650           SNA CLA
00343  5740           JMP I PTRGCH
00344  3247           DCA PTPEOF
00345  2247   PPTIME, ISZ PTPEOF          /THIS LOOP OVERFLOWS IN APPROX.
                                          /100 MS ON A PDP-8,
00346  5351           JMP I PTRGCH

00351  1355   PGCHLP, TAD PTPTMP
00352  3306           DCA PTR
00353  0232   PTP232, 232
00354  0336   PTP336, 336                 /FALL THROUGH CONSTANTS TO STALL
                                          /FOR TIME
00355  0000   PTPTMP, 0                   /ALWAYS LESS THAN 400
00356  6011           RSF
00357  5345           JMP PTTIME          /READER NOT READY - CHECK TIMING
00360  6016           RRB RFC             /READER READY - READ CHAR
00361  3355   PTRXX,  DCA PTPTMP          /BUFFER READER BY ONE CHARACTER
                                          /TO ELIMINATE
00362  1306           TAD PTR             /GARBAGE CHARACTER AT END OF TAPE
00363  5740           JMP I PTRGCH        /AND RETURN

               /COULD SAVE A COUPLE OF WORDS HERE
00364  7330   PTPERR, CLA CLL CML RAR     /SIGNAL A "PERMANENNT I/O
                                          /ERROR" ON THE DEVICE
00365  5271           JMP FATAL

00366  6201   PCDF,   CDF 0
00367  6203   PTPCIF, CIF CDF 0
00370  0000   PTPWC,  0
00371  0000   PTPCA,  0
               $$$$$$$$
```

To operate some devices it may be necessary to turn off the interrupt system. The user must be careful not to leave interrupts off for long periods of time. Otherwise, use of the handler may disrupt other jobs. It may also be desirable to LOCK the handler in core so that it will never have to wait to be swapped in (see 6.3.1). This can be dangerous, particularly on a 16K system, because it may block SCALE or DMON from running. The handler should minimize the amount of time it is LOCKed in core, and it should avoid calling SCALE or DMON during this period. The handler must always UNLOCK its memory when it has completed

its operations.

Once the handler has been written, it should be assembled into binary format using PAL8. It should then be built into the virtual OS/8 system, using BUILD (see 6.4.1).

6.4.7 Reading Binary Paper Tapes

When reading binary paper tapes, 6 bits of each frame are used for a location setting or data deposited in a location. ETOS does not know that these characters are not user input characters. It therefore traps on the special control characters, if enabled. Reading of an ASCII 203 causes a "^C" to be executed. Before reading a binary tape from the low speed reader, the user must disable all ETOS handling of control characters. The non-privileged user may disable all the control character handling except "^V". It is for this reason that a user must be privileged to read binary paper tapes. The ETOS low speed paper tape reader handler performs this function automatically for the user and re-enables control character handling upon exit. To read a binary paper tape, the user enters the dialogue of Figure 6-32.

Figure 6-32
Sample Read of a Binary Paper Tape

```
.^VPRIV 4040
R PIP
*FILE.EX<PTR:/B
*^C
.^VPRIV 0040
```

6.4.8 Enabling CCL

CCL (Concise Command Language) provides the OS/8 user with an extended set of keyboard monitor commands. The user may also write his own CCL commands and add them to CCL. See page 1-52,

OS/8 Handbook for more information on using CCL.

After a new version of OS/8 has been transferred to OS8.RTS via the "H" start-up option (see 4.3) or BUILD (see 6.4.1), it may be necessary to re-enable CCL. You will know that CCL needs to be enabled if you attempt to use a CCL command (e.g., DIRECT, COPY) and the system does not recognize the command or executes the wrong program. For example, if you entered

```
.DIR
```

and the system printed

```
DIR?
```

indicating that it didn't understand the command or

```
#
```

indicating that it was running SET instead of DIRECT, you would know that CCL should be re-enabled. To enable CCL, enter the dialogue of Figure 6-33 after LOGging IN to a privileged account.

Figure 6-33
Enabling CCL Under Virtual OS/8

```
.^VPRIV 4040
^VWE 0
R CCL
.^VWL 0
^VPRIV 40
```

6.4.9 Setting Terminal Specifications

ETOS has a system-wide OS/8 terminal handler called "TTY:". This handler is used to perform output from some OS/8 programs (e.g., DIRECT) and input to some OS/8 programs (e.g., PIP). Many

programs in the system use their own internal terminal handler (e.g., BASIC, OBASIC, FORT, FIV). These terminal handlers are tailored to that program's use. (See Appendix J, OS/8 Handbook Update.) The SET program can be used to change operating characteristics of the TTY handler. These changes will not affect the programs which utilize an internal terminal handler. Before you execute the SET command, enter the commands of Figure 6-34 after logging into a privileged account.

Figure 6-34
Commands Before Execution of TTY SET Options

```
.^VPRIV 4040
^VWE 0
```

After the above commands are entered, you can enter your SET commands. After your SET commands are specified, enter the commands of Figure 6-35.

Figure 6-35
Commands Terminating TTY SET Options

```
.^VWL 0
^VPRIV 0040
```

A list of all valid SET TTY options follows.

SET TTY [NO] ARROW

If the arrow feature is enabled, all control characters output by the TTY handler are printed in the form:

^X

where "^" indicates that a control character is being typed and "X" specifies which control character is being typed (100 + code for control character). If a CTRL/E character is output, "^E" is

printed on the terminal.

Using this attribute with the NO modifier causes each control character typed by the TTY handler to print with no modification. If you send a CTRL/E to the terminal, the actual code for a CTRL/E (205) is sent. As a result, no readable character is output.

SET TTY COL[UMN]

This attribute changes the default number of columns used to print the directory (using the DIR command) to the number you specify as "N". The initial default number of columns is equal to one. The number you specify must be in the range 1 to 7.

SET TTY [NO] ECHO

This attribute causes all characters entered to the TTY handler to be printed as they are entered. Using this attribute with the NO modifier causes character echoing to be disabled.

SET TTY [NO] ESC

This attribute causes the escape or alt mode character (ASCII code 233) to print as a control character. If you enable this option, you should specify NO ARROW. This allows you to send escape sequences to CRT terminal. If you enable the escape feature and do not SET NO ARROW, escape will print as "^[". Using this attribute with the NO modifier causes escape to print as dollar sign ($).

SET TTY [NO] FILL

This attribute causes two null fill characters to be printed following every tab (ASCII code 211) which is output. Enabling

this feature has no effect unless you also enable the TAB feature. Using this attribute with the NO modifier causes no fill characters to be printed.

SET TTY [NO] FLAG

This attribute causes all lower case characters to be output as upper case characters preceded by a quote ("). For example, "a" would be output as "A". This feature is enabled if you wish to determine which characters are lower case when you're using an upper case only terminal. Using this attribute with the NO modifier causes lower case characters to be output in lower case.

SET TTY HEIGHT n

This attribute sets the number of lines which are printed on the terminal between pauses. The default value is twenty-four lines. This attribute has no effect unless the PAUSE feature is enabled.

SET TTY [NO] LC

This attribute allows you to input lower case characters. These characters are stored in lower case. Using this attribute with the NO modifier causes input lower case characters to be converted to upper case.

SET TTY PAGE

This attribute allows you to interrupt terminal output by entering CTRL/S. Terminal output ceases until you enter a CTRL/Q. Both the CTRL/S and the CTRL/Q are ignored. Using this attribute with the NO modifier disables the interruption of terminal output.

SET TTY [NO] PAUSE [n]

This attribute causes terminal output to pause for approximately "n" seconds each time the number of lines specified in the HEIGHT command is output. Using the NO modifier disables terminal pauses entirely. SET TTY PAUSE Ø is equivalent to SET TTY NO PAUSE.

SET TTY [NO] SCOPE

This attribute causes the rubout or delete key to physically remove characters it deletes. This removal is performed by printing a backspace, space, backspace on the terminal. This attribute only is valid if your terminals are CRT terminals. Using the NO modifier causes the rubout or delete key to print the character deleted, enclosed by slashes.

SET TTY [NO] TAB

This attribute causes a tab (ASCII code 211) to be output as a control character. This setting will enable tabs to function properly only on terminals with a hardware tab. Terminals without the hardware tab will ignore this character. Using the NO modifier causes tabs to be simulated with spaces. This simulated tab is the normal setting for most terminals.

SET TTY WIDTH n

This attribute sets the width of terminal output to "n" decimal positions. "n" must be a multiple of eight in the range 1 to 255. However "n" must not be 128. Any lines which are longer than the specified width are output, "n" characters at a time. A carriage return line feed separates each "n" characters. The terminal handler keeps track of the number of characters by counting each character that is printed. If the TAB feature is enabled, this method does not function properly. A tab is output as one character, but a terminal which has the hardware tab

feature converts the tab to spaces. This discrepancy makes the character count meaningless. If you have NO TAB specified, tabs are output as spaces. The character count is incremented properly and the WIDTH feature functions as specified.

6.4.10 Setting Line Printer Characteristics

ETOS has a system-wide OS/8 line printer called "LPT:". This handler is used to perform non-spooled line printer output from most OS/8 programs (e.g., DIRECT, BASIC). Some programs in the system use their own internal line printer handler (e.g., FORT, F4, BATCH). These handlers are tailored to that program's use. The SET program (see Appendix J, OS/8 Handbook Update) can be used to change operating characteristics of the LPT handler. Many of the standard SET commands are no longer valid, because the basic line printer operating charactertistics are specified via CONFIG (see 3.4). These changes will not affect the programs which utilize an internal line printer handler. They also will not affect programs utilizing the spooling handler QLP. The SPOOLR program is used to set QLP handler characteristics (see 6.4.13). Before you execute the SET command, enter the commands of Figure 6-36 after logging into a privileged account.

Figure 6-36
Commands Before Execution of LPT SET Options

```
.^VPRIV 4040
^VWE 0
```

After the above commands are entered, you can enter your SET commands. After your SET commands are specified, enter the dialogue of Figure 6-37.

Figure 6-37
Commands Terminating LPT SET Options

```
.^VWL 0
^VPRIV 0040
```

A list of all valid LPT SET options follows.

SET LPT [NO] LC

This attribute allows lower case characters to be printed as lower case characters. This option should only be enabled if your line printer has the capability of printing both upper and lower case. Using this attribute with the NO modifier causes all lower case characters to be converted to upper case before they are printed.

SET LPT WIDTH n

This attribute sets the width of line printer output to "n" decimal positions. "n" must be in the range 1 to 255. Any lines which are longer than the specified width are output, "n" characters at a time. A carriage return, line feed separates each "n" characters. The line printer handler keeps track of the number of characters by counting each character that is printed.

6.4.11 Setting General Device Characteristics

ETOS has system-wide OS/8 handlers for every device which ETOS supports (e.g., CDR, DTAØ, TTY). Most OS/8 programs utilize these external device handlers. Some programs in the system use their own internal handlers (e.g., FORT, F4, EPIC). These handlers are tailored to that program's use. The SET program can be used to change operating characteristics of the external handlers. These changes will not affect programs which utilize internal handlers. Before you execute the SET command, enter the commands of Figure 6-38 after logging into a privileged account.

Figure 6-38
Commands Before Execution of Device SET Options

```
.^VPRIV 4040
^VWE 0
```

After the above commands are entered, you can enter your SET commands. After your SET commands are specified, enter the dialogue of Figure 6-39.

Figure 6-39
Commands Terminating Device SET Options

```
.^VWL 0
^VPRIV 0040
```

A list of valid SET commands for all devices follows.

SET BLOCK b, LOC n [=m]

This attribute is used to change the contents of a handler located in block "b" on the virtual OS/8 system device. "n" specifies the location which is modified. This location must be in the range 0 to 177 for one-page handlers and 0 to 377 for two-page handlers. "m" may be specified as the new contents of location "n". If "m" is not entered, the system prints the current contents of the specified location, followed by a slash (/). You can enter a new value in that location by typing that value followed by a carriage return. If you want to leave the contents unchanged, press carriage return only.

SET dev DVC n

This attribute sets the IOT device code used by the handler to the decimal number you specify as "n". "n" must be in the range 30 to 77. If the device is handled by the ETOS monitor (e.g., RK05, System Industries disk, card reader, line printer,

terminal), this command has no effect. You must change the IOT code in the ETOS monitor to affect these devices. However, if the device is implemented as a privileged device handler (e.g., TD8E DECtape, TCØ8 DECtape, RXØ1 floppy disk, PT8E high speed paper tape reader/punch), this command can be used to implement non-standard device codes. For example, if your floppy disk interface had a device code of 64 instead of 75, you could utilize this option to set the OS/8 handler's device code to 64.

SET dev [NO] FILES

This attribute allows the specified device to act as a file-structured device handler. Some programs check to see if a device is file-structured before output is sent to it. For example, FOTP disables input from non-file structured devices. This attribute allows you to circumvent these tests by treating a device like a paper tape reader/punch as a file-structured device. Using the NO modifier with this attribute causes the system to treat the device as a non-file structured device. Any changes performed by this option are removed the next time the virtual machine is booted.

SET dev LOC n [=M]

This attribute is used to change the contents of the device handler specified by "dev:". "n" specifies the location which is modified. This location must be in the range Ø to 177 for one-page handlers and Ø to 377 for two-page handlers. "m" may be specified as the new contents of location "n". If "m" is not entered, the system prints the current contents of the specified location, followed by a slash (/). You can enter a new value in that location by typing that value followed by a carriage return. If you want to leave the contents unchanged, press carriage return.

The use of this option is identical to the BLOCK option. The difference between the two options is that the BLOCK option specified a handler by its block location on the system device, whereas the LOC option specifies a handler by its device name.

SET dev [NO] READO

This attribute causes the device specified by "dev" to become a read-only device. Any output sent to this device causes an error message, informing you that the device is read-only. Using this attribute with the NO modifier restores the handler to its original state. Any changes performed by this option are removed the next time that the virtual machine is booted.

6.4.12 Setting Automatic Login Batches

It is desirable to automate as much of the system as possible. For novice users, the automatic processing simplifies their use. For experienced users, the automatic processing relieves some of the repetition and increases their efficiency. The following command allows you to automate your system.

SET SYS [NO] INIT [command]

If you enable this option, a command is executed each time that OS/8 is booted. Under stand-alone OS/8, this process is initiated each time that you boot the physical machine (see 2.2 or 2.3). Under virtual OS/8 this process is initiated each time you LOGIN to an account or reBOOT the virtual machine. If you specify a command after the keyword INIT, this command is executed. Therefore, the command SET SYS INIT HELP causes a help message to be printed each time a user LOGged IN or BOOTed OS/8 on his virtual machine. Under ETOS, this form of the command is restrictive in that all users automatically execute the same command. The command is also limited to five characters.

If you do not specify a command after the keyword INIT, the command is taken from the file DSK:INIT.CM. Under ETOS, this file is taken from the user's OS8DISK area. If a user LOGs IN to an account without an OS8DISK area, the message "BAD DIRECTORY" is printed when the system attempts to locate the INIT.CM file. This message can be ignored. If a user LOGs IN to an account which has an OS8DISK area which does not contain a file called INIT.CM, the error message "FILE NOT FOUND" is printed. This message can be ignored.

The file INIT.CM is an ASCII file which contains one CCL command (see page 72 Appendix J, OS/8 Handbook Update). This CCL command is typically a SUBMIT command which initiates a batch stream (see page 1-72, OS/8 Handbook). Since BATCH under ETOS allows all SCALE commands to be processed in addition to OS/8 commands, you can automatically ASSIGN devices, LOOKUP channels or perform any other ETOS functions. You can also include FORCE commands in the batch, which will control other terminals.

The options discussed in this section allow full automation of the system. If you have a hardware bootstrap, you may set the system up so that pressing the boot key starts ETOS, asks you to enter the time and date and brings all terminals up and running in specified accounts. If you don't have a hardware bootstrap, executing the boot program from another OS/8 peripheral or booting the disk from the switch register starts this process. To automate your system, use the following steps.

1) Under stand-alone OS/8, execute the command line, SET SYS INIT. This causes the system to execute the command contained in the file INIT.CM each time stand-alone OS/8 is booted.

2) Create INIT.CM under stand-alone OS/8 with the commands of Figure 6-40.

Figure 6-40
Creating INIT.CM under Stand-alone OS/8

```
.CREATE INIT.CM
#A
LOAD SYS:ETOS.SV=14000/I/G
<CTRL/L>
#E
.
```

When this INIT.CM file is executed, it causes ETOS to be initiated. The LOAD command is used instead of an R Command because your command must use CCL.

3) Enter the commands of Figure 6-41 to enable automatic LOGIN of the console terminal.

Figure 6-41
Enabling Automatic LOGIN of Console TTY

```
.R CONFIG
'CONFIG' V3.010
*AUTO
*NOOPTN
*EXIT
.
```

The above dialogue assumes that you wish to automatically LOGIN to account [0,3] and that you have not changed the password to the operator's account. If either of those assumptions is not true, use the INITLN option (see 3.17) to implement your changes.

At this point, the system is set up to automatically initiate ETOS and LOG the console terminal INto account [0,3].

4) To allow the LOGIN of account [0,3] to automatically execute a batch, enter the commands of Figure 6-42.

Figure 6-42
Enabling Automatic ETOS Batch Execution

```
.R ETOS
ETOS V5B

H;0,3 OPERATOR                (automatically executed)
JOB 03 LOGGED IN ON CONSOLE 00
.^VPRIV 4040
^VWE 0
SET SYS INIT
.^VWL 0
^VPRIV 0040
```

At this point, each time a user LOGs INto any account or boots virtual OS/8, the system executes the command contained in the file INIT.CM.

5) Enter the dialogue of Figure 6-43 to create an INIT.CM which allows you to execute multiple commands. This write-up assumes that you have an OS8DISK area for account [0,3].

Figure 6-43
Creating INIT.CM Under Virtual OS/8

```
.CREATE INIT.CM
#A
SUBMIT BAT03.BI/Q
<CTRL/L>
#E
.
```

This write-up assumes that you have OS/8 BATCH. If you do not have BATCH, you cannot automate your system any further. You can create an INIT.CM file for each account which will execute one command each time the user LOGs IN. You cannot automatically LOG other users IN from [0,3].

6) You may now enter the batch stream which will LOG others INto the system. A sample creation of a batch stream which brings a three terminal system up and running is listed in Figure 6-44.

Figure 6-44
Sample Creation of Automatic Batch Stream

```
.CREATE BATØ3.BI
#A
$JOB
.R INIT
!FORCE 1;LOGIN
!FORCE 1;
!FORCE 1;1,4
!FORCE 1;
!FORCE 1;COSACCT
!FORCE 2;LOGIN
!FORCE 2;
!FORCE 2;Ø,11
!FORCE 2;TEST
!FORCE 2;
.R PATTERN
$END
<CTRL/L>
#E
.
```

This batch initiates the time and date, LOGs console 1 INto account 1,4, LOGs console 2 INto account Ø,11 and initiates a pattern generation program on the console terminal. Notice that in this batch there were some lines which FORCEd no commands to a console. These were included to wait for the system to complete one command before starting another command. In some circumstances, you might have to include many null FORCE commands to avoid timing problems.

7) The final step in automating the system is to LOGIN to accounts [1,4] and [Ø,11] and create INIT.CM files and batch streams which will execute each time you LOGIN to these accounts. This write-up assumes that accounts [1,4] and [Ø,11] have an OS8 disk area already created. A sample creation of commands is shown in Figure 6-45.

Figure 6-45
Sample Creation of INIT.CM Files

```
.LOGOUT
JOB Ø3 LOGGED OUT ON CONSOLE ØØ
Ø1:3:Ø5 P.M. ON TUE Ø6-APR-79
ELAPSED TIME:
1Ø MINUTE(S) 27.6 SECONDS
GOOD AFTERNOON
!LOGIN
ETOS V5B AT Ø1:13:Ø5 P.M. ON TUE Ø6-APR-79
ACCOUNT? 1,4
PASSWORD? COSACCT                (this does not echo)
JOB Ø3 LOGGED IN ON CONSOLE ØØ
FILE NOT FOUND                  (INIT.CM does not exist yet)
.CREATE INIT.CM
#A
SUBMIT BAT14.BI/Q
<CTRL/L>
#E
.CREATE BAT14.BI
#A
$JOB
!ASSIGN DK1:
!LOOKUP 3=DK1:
!ASSIGN DK2:
!LOOKUP 4=DK2:
!LOOKUP 5=DKØ:[Ø,77]DATAFI.LES
!ASSIGN LPT
.R COSBO
$END
<CTRL/L>
#E
.LOGOUT
JOB Ø3 LOGGED OUT ON CONSOLE ØØ
Ø1:36:27 P.M. ON TUE Ø6-APR-79
ELAPSED TIME:
Ø2 MINUTES 21.5 SECONDS
GOOD AFTERNOON
!LOGIN
ETOS V5B AT Ø1:36:32 P.M. ON TUE Ø6-APR-79
ACCOUNT? Ø,11
PASSWORD? TEST           (this does not echo)
JOB Ø3 LOGGED IN ON CONSOLE ØØ
FILE NOT FOUND                  (INIT.CM does not yet exist)
.CREATE INIT.CM
#A
SUBMIT BAT11.BT/Q
<CTRL/L>
```

Figure 6-45 (continued)

```
#E
.CREATE BAT11.BI
#A
$JOB
.R BCOMP
*PROG.BA/G
$END
<CTRL/L>
#E
.LOGOUT
JOB 03 LOGGED OUT ON CONSOLE 00
01:40:52 P.M. ON TUE 06-APR-79
ELAPSED TIME:
03 MINUTES 45.9 SECONDS
GOOD AFTERNOON
!
```

The system is now fully automated. When the batch stream SUBMITted from account [0,3] LOGs console 01 INto account [1,4] and console 02 into account [0,11], the INIT.CM files in those accounts execute their batch processes. If you boot stand-alone OS/8 at this time, ETOS is initiated, the time and date is initialized, console 01 is running COS in account [1,4], console 02 is running an OS/8 basic program called PROG.BA and console 00 is running an assembler program called PATTERN.SV.

To disable automatic execution of the INIT.CM file, LOGIN to a privileged account and enter the commands of Figure 6-46.

Figure 6-46
Disabling Automatic ETOS Batch Execution

```
.^VPRIV 4040
^VWE 0
SET SYS NO INIT
.^VWL 0
^VPRIV 0040
```

After you have automated your system, there is one problem. When you execute the SHUTUP command (see 6.2.7) or reboot the disk to return to stand-alone OS/8, the INIT.CM is executed which initiates ETOS. Therefore, it is not possible to return to

stand-alone OS/8. If you wish to run RKCOPY or one of the other stand-alone OS/8 programs, you must circumvent this problem. To circumvent the problem, you must RENAME the INIT.CM file which resides on the stand-alone OS/8 portion of the pack, while you are running under ETOS. When you execute the SHUTUP command, the system attempts to execute the command contained in the file INIT.CM. Since the file does not exist under that name, the system prints "FILE NOT FOUND" and returns to OS/8. To RENAME the INIT.CM file, LOGIN to a privilged account under ETOS and enter the commands of Figure 6-47.

Figure 6-47
Disabling Automatic OS/8 Batch Execution

```
.^VS
!PRIV 4040
!LOOKUP 3=[0,2]OS8.OS8
!WE 3
!CONTINUE
RENAME CHN3:INIT.BK<CHN3:INIT.CM
FILES RENAMED:
INIT.CM
.^VS
!SHUTUP

FILE NOT FOUND
.                          (stand-alone OS/8 is now running)
```

After you have run the stand-alone OS/8 programs, enter the commands of Figure 6-48 under stand-alone to re-enable automatic ETOS initiation.

Figure 6-48
Reenabling Automatic OS/8 Batch Execution

```
.RENAME SYS:INIT.CM<SYS:INIT.BK
FILES RENAMED:
INIT.BK
.
```

6.4.13 Line Printer and Terminal Spooling

Section 4.9 in the User's Manual explains the concept of spooling. To enable spooling, the system manager must initiate the spooling controller program each time that the system is brought up. Permanent files are placed in an output device queue for eventual printing. A unique I.D. number is generated for each permanent file. These permanent files are usually created by the QLP handler. They are considered permanent only when a CTRL/Z is sent to the QLP handler, indicating that the file is complete.

Initiation of the SPOOLR is performed by entering the commands of Figure 6-49.

Figure 6-49
SPOOLR Initiation Commands

```
.^VPRIV 4040
R SPOOLR
```

The system responds with "PHYSICAL DEVICE?". If you are to spool to a line printer, the device name is "LP@@". If you are spooling to a terminal, the device name is "TTnn", where "nn" is the console number. This console number corresponds to the entry number of the terminal's interface in the IOT TABle (see 3.9). If you are spooling to a terminal, the terminal must be initialized. To initialize a terminal, type a character on the terminal's keyboard or FORCE a character into the terminal's input buffer. If the terminal is not initialized, the message of Figure 6-50 will appear.

Figure 6-50
Error Message in SPOOLR for Non-initialized Terminal

```
?%VM HALT
!PC 4507 AC 0 FL ...
```

The system responds with "LOGICAL DEVICE?". Enter the specification used in the QUEUE command. The QLP: handler assumes that the line printer name is "LPØØ". If you change the logical name, you must change the QLP handler. If you have two spoolers running simultaneously, you must BUILD in two copies of the QLP handler with different logical names.

The system responds with "OUTPUT DEVICE WIDTH?". Enter the decimal width of the output device. All lines which are printed which are longer than the specified length are printed on multiple lines, with the continuation lines indented eight spaces.

The system responds with "HARDWARE FORMFEED?". The spooler will simulate form feeds on any spooler output device which does not handle form feeds in the hardware. Most terminals do not have hardware form feeds. Enter Y or N according to the characteristics of your output device.

The system responds with "LENGTH?". Enter number of vertical lines per page. The number is utilized to simulate hardware form feed and to count the number of pages output. The standard response for 11-inch high paper is 66. Even if the device has hardware form feed, this question is asked. It is used to count pages for the QS command.

After all questions have been answered, a "$" is printed, indicating that spooling is initiated. If the system had been brought down without stopping the spooler, the spooler continues printing the file it was printing before shutdown. After spooling has been initialized, you can enter one of the commands contained in Table 6-8 to examine the queue, print summary statistics, etc. If you are finished with your command specification, you probably want to DETACH the spooler so that it does not occupy a physical terminal.

Table 6-8 contains the four commands that can be specified to the spooler program.

Table 6-8
SPOOLR Commands

Command	Meaning
QP	list all permanent files
KP n	kill permanent file with ID equal to "n"
E	shut down the spooler
QS	print a lists of spooler usage for the current permanent file and for all programs listed by this spooler

Killing a permanent file will remove the item from the queue and the output will never be printed. If the permanent file is currently listing on the output device, output will cease at the end of the current buffer.

If the spooler is shut down by the E command or by the system crashing, the queue remains intact. When the spooler is initiated, it begins outputting the file it was listing at the time of shut down. All permanent files in the queue will be printed.

Initialization of the spooler must be performed under the operator's account [0,3].

A sample run of the SPOOLR Program is contained in Figure 6-51.

Figure 6-51
Sample Run of SPOOLR

```
!LOGIN;0,3 OPERATOR

JOB 3 LOGGED IN ON CONSOLE 0

.^VPRIV 4040
R SPOOLR

PHYSICAL DEVICE? LP@@          (system line printer)
LOGICAL DEVICE? LP00           (for use with QLP)
OUTPUT DEVICE WIDTH? 80

HARDWARE FORM FEED? Y
LENGTH? 66                     (required for page count)

$QP                            (check on files being printed)

ID OWNER FILE  OFFSET
0021 [01,10] DK0@:[01,10]TEMP00@.LPT      0000000
0022 [00,05] DK0@:[00,05]TEMP00@.LPT      0000000
ID 0021 OWNER [00,03] NOW LISTING
$QS

                PAGES            CHARS
LISTING         000000000001     0000000000122
TOTALS          000000000342     000001152771
$KP 21
$QP
ID    OWNER      FILE                              OFFSET
0022  [00,05]    DK0@:[00,05]TEMP00@.LPT           00000000
$^VDET

DETACHING FROM JOB 3
```

When the spooler is initiated, it creates a file queue (FQ.DAT) called [0,3]XXXXSPL.QUE, where XXXX is the logical device name of the spooler. This file is used to store the list of files waiting to print. I.D. numbers start with 0001 and is incremented by 1. Therefore, the first file listed after the spooler is initiated has I.D. 0001. This I.D. number is used to identify files in the KP command.

The QLP device handler creates ETOS files in the form TEMPXX.LPT on the system disk in the account of the user requesting

spooling. "XX" starts at "ØØ" and is incremented by 1. The first file spooled by a user in an account is of the form TEMPØØ.LPT. The second file is of the form TEMPØ1.LPT. Once these files are listed, these temporary files are deleted and the names can be used again.

The OFFSET column in the listing of permanent files represents the OFFSET specified in the QUEUE command (see 3.11, System User's Guide). If you use the QLP handler to print your files, this offset is always "ØØØØØØØØ".

The QS command prints the number of pages (in octal) and the number of characters (in octal) which the spooler has output for the current permanent file. It also lists a summary of page count and character count for all files listed by the spooler.

The KP command allows you to delete a permanent file which is printing or is waiting to be printed. If you delete a file which is printing, the system will stop printing the file and begin printing the next file.

If you want to shut the SPOOLR down, you must ATTACH back to the job which is running SPOOLR. If you do not remember which job the SPOOLR is associated with, run SYSTAT to print the status of all jobs (see 4.5, System User's Guide). A sample execution of shutting down the SPOOLR is given in Figure 6-52. This example assumes that you are LOGged OUT before you start this dialogue.

Figure 6-52
Sample Shutdown of SPOOLR

```
!ATTACH
JOB NUMBER? 3
PASSWORD? OPERATOR              (this does not echo)

$E
%END SPOOLING
.LOGOUT

JOB Ø3 LOGGED OUT ON CONSOLE ØØ
Ø3:15:17 P.M. ON TUE Ø6-APR-79
ELAPSED TIME:
2 HOUR(S) 1Ø MINUTE(S) 39.4 SECONDS
GOOD AFTERNOON
!
```

If you have initiated the SPOOLR properly and files are not being spooled, you know that there is a problem. The most common reason for the SPOOLR not running properly is a lack of available disk space. To determine what the problem is, first ATTACH to the SPOOLR job. Any error messages which have been executed will print at this point. Figure 6-53 contains sample execution.

Figure 6-53
Sample SPOOLER Error

```
!ATTACH
JOB NUMBER? 3
PASSWORD? OPERATOR              (this does not echo)

?ERRFNA?
$
```

A list of all messages printed by the spooler is presented below.

DEVICE ALREADY OWNED

Another job has the line printer ASSIGNed. To determine who owns the printer, enter ^C to exit SPOOLR and run SYSTAT (see 4.5, System User's Guide) to print busy devices. You can use TALK (see 3.9, System User's Guide) to request that the user DEASSIGN

the printer and then you can run SPOOLR.

%END SPOOLING

When a spooler is shut down by the E command, this message is printed. A spooler should never be shut down via any other method.

?ERRFNA?

There was an error when attempting to LOOKUP the permanent file which is going to be listed. To recover from this error, delete the permanent file in error with the KP command, end spooling with the E command and reinitialize by running SPOOLR.

ERROR ENTERING FQ.DAT

There was an error when the system attempted to create the queue file for this spooler. The queue file name is of the form [Ø,3]XXXXSPL.QUE, where XXXX is the logical spooler device. The most common cause of this error is a lack of space in the ETOS directory. Run the ACCNT program (see 5.4) to list the ETOS directory and the number of free blocks. To recover from this error, DELETE or REDUCE an ETOS file or files and run SPOOLR again.

?ERRRLF?

There has been an error in READing from the permanent file currently being listed. To recover from this error, delete the permanent file in error with the KP command, end spooling with the E command and reinitialize by running SPOOLR.

?HUH?

A command was entered to the spooler which was not one of the four valid options listed in Table 6-8.

ID XXXX OWNER [YY,ZZ] NOW LISTING

If a permanent file is being listed on the output device, this message is printed at the end of the QP or QS command.

ID NOT FOUND

No permanent file is found with the ID specified in the KP command.

INVALID DEVICE

The physical device name is not of the form "LP@@" or "TTXX", where XX is the console number.

INVALID NUMBER

An output device length was entered with non-numeric characters.

INVALID RESPONSE

An answer to "HARDWARE FORM FEED?" was entered which is not Y or N.

NOT LISTING

If no files are currently being listed on the output device, this message is printed at the end of the QS command.

% QUEUE IS EMPTY

If no permanent files are in the queue file for listing, this

message is printed by the QP command.

SPOOLER ALREADY EXISTS

SPOOLR has been run previously with the same physical device name as the spooler you are attempting to initialize.

TOO MANY SPOOLERS

You cannot initialize a spooler, because seven spoolers are active on the system.

As explained previously, the temporary files created by the QLP handler of the form TEMPXX.LPT are cleared in the user's account on the system disk. To change the QLP handler so that it creates the temporary files in the user's account on a public pack, LOGIN to a privileged account and enter the commands of Figure 6-54.

Figure 6-54
Sample Creation of SPOOLR's Temporary Files on a Public Pack

```
.^VPRIV 4040
^VWE 0
RUN SYS:BUILD
$ALTER SPLP,262=XX00          (6-bit ASCII drive designation)
$BOOT
SYS BUILT
.SAVE SYS:BUILD
.VWL 0
^VPRIV 0040
```

This write-up assumes that you have 33 contiguous free blocks on the system device for SAVing BUILD. If you do not have this space, SAVE BUILD on an area which contains 33 contiguous free blocks and then copy BUILD to the system area. In the ALTER command, the 6-bit ASCII code for the drive designation is contained in the high order six bits. Therefore, setting location 262 equal to 6000 causes temporary files to be created

in the user's account on DKØ. Setting it equal to 6100 causes temporary files to be created on DK1. Setting it equal to 6200 causes temporary files to be created on DK2. Setting it equal to 6300 causes temporary files to be created on DK3. The DK designations refer to public packs mounted in a disk drive, for RKØ5 systems, DKX is the designation for drive X. For System Industries systems, DK1 represents the removable pack in port Ø; DK2 represents the fixed pack in port 1; DK3 represents the removable pack in port 1.

If you desire to run two spoolers, you must LOAD another copy of the QLP handler into the system. The assembler source to the QLP handler is stored on the distribution pack in account [Ø,6] in the file QLP.PA. Before you compile this file, change the group name which is currently SPLP, change the device name which is currently QLP and change the logical name which is currently LPØØ. After the file is compiled, LOAD it into the system, using the procedures of 6.4.1. To initiate the two spoolers, run SPOOLR with one set of specifications, DETACH the job, run SPOOLR with the second set of specifications and DETACH this job also.

6.4.14 Printing Cumulative Job Statistics

SYSTAT is used to display the usage of the machine at any one point in time. You also can display cumulative job statistics by account number with the USAGE program. To run USAGE, LOGIN to a privileged account and enter R USAGE<RET>.

The system responds with an asterisk (*). At this point, enter =account number, where "account number" is the account whose job statistics you wish to display. If the account number is "Ø" or the "=" option is omitted entirely, the current user's account is assumed. The account number should be entered without commas. To exit the program, end your last command with an escape or alt mode. A sample run of USAGE is given in Figure 6-55.

Figure 6-55
Sample Run of USAGE

```
.R USAGE
'USAGE.SV'  V4.001  10:21:27  A.M. ON SAT 07-APR-79
*=3$                          ($ - escape or alt mode)

THIS MONTH'S STATISTICS
RUN-TIME = 0000:00:00:03.6
CONNECT-TIME = 000:00:05:41.4
TIME OF LAST LOGIN 10:18:01 A.M. ON SAT 07-APR-79
TIME OF LAST LOGOUT 10:18:38 A.M. ON SAT 07-APR-79
.
```

This program may be run under OS/8 BATCH. Therefore, you can set up a batch stream which will display cumulative job statistics for all accounts on the system. In addition to the "=" option, there are "/" options which are discussed below.

The USAGE program prints out statistics from files in account [00,03] which are called ACCTIMX.YYY. X is the first digit of the account number and YYY represents the last three digits of the account number. Therefore, the statistics file for account [17,20] is called ACCTIM1.720. These statistics files are created by LOGging OUT of the system. Due to the fact that usage statistics are not updated until the user LOGs OUT, you should not SHUTUP the system without insuring that all users are LOGged OUT. If you do not wish to maintain these statistics files, see 8.13.1. Before you utilize USAGE for an account, you must initialize the statistics file for that account. This intialization may only be performed on the console terminal. For each account you wish to initialize, enter =account number/N/Q. For example, to initialize the statistics file for account [70,21], enter the command contained in Figure 6-56.

Figure 6-56
Initialization of USAGE Statistics File

```
*=7021/N/Q
*
```

If you delete an account, you may want to delete the statistics file for the account. This deletion will restore one additional free block to the ETOS directory. To delete an account statistics file, enter =account number/K/Q. This deletion can only be performed on the console terminal. For example, to delete the statistics file for account [01,04], enter the command contained in Figure 6-57.

Figure 6-57
Sample Deletion of Usage Statistics File

```
*=104/K/Q
*
```

In addition to the monthly statistics which are updated by LOGIN, there are four other statistics which may be kept. A list of these statistics follows.

CURRENT STATISTICS - These statistics represent a running total of all run-time and connect-time used by this account since the statistics file was initialized via the "/N" option. These statistics are updated by the LOGIN program which performs the LOGOUT. The current statistics are not kept in the ACCTIMX.YYY file, but are computed by adding the month's statistics to the total statistics.

TOTAL STATISTICS - These statistics may be updated at any time by the user via the "/U" option. The current month's statistics are added to the previous totals and the results are stored as the new totals. The previous total statistics are saved in the old statistics area and the current month's statistics is zeroed.

This process may only be initiated on the console terminal. When this process is complete, the total statistics are equivalent to the current statistics. However, during the month, the current statistics reflect usage at the end of the previous month.

SAVED STATISTICS - These statistics may be updated at any time by the user via the "/S" option. Copies of the month's statistics are moved into the saved statistics. The month's statistics are not affected. This option may only be executed on the console terminal. These saved statistics may be used to reflect usage on a different cycle than the monthly statistics. For example, you could save the current month's statistics every two weeks. These saved statistics would reflect the statistics of the last full two-week period.

OLD STATISTICS - Before the total statistics are updated via the "/U" option, the existing totals are saved in the old statistics area. If you update your totals monthly, the old statistics area reflects last month's statistics and can be used for a comparison. Also "/M" will place a copy of the month's statistics in the old statistics.

When you enter the account number to USAGE, the month's statistics for that account is printed. In addition to the account number, you may specify options to USAGE which are each preceded by a slash (/). Some of these options have been discussed previously. A list of all options and their meanings are given in Table 6-9.

Table 6-9
USAGE Options

Option	Meaning
/A	Print all of the statistics (month's, current total, saved, old).
/C	Print current statistics in addition to the current month's statistics.
/D	Disable printing of last LOGIN and last LOGOUT. These times are not printed again until the E option is used.
/E	Enable printing of the last LOGIN and last LOGOUT.
/G	Get and print the saved statistics in addition to the month's statistics.
/H	Print the USAGE header line in addition to the current statistics.
/K	Kill the statistics file. This should only be done if the account no longer exists. This option can only be specified on the console terminal.
/M	Move the month's statistics into the saved statistics. This option can only be specified on the console terminal. The month's statistics are not effected.
/N	Set up a new statistics file for the account. This process must be performed only once, after you create an account.
/O	Print the old statistics in addition to the current statistics.
/Q	Quell output. Don't print any statistics. This option is usually used in conjunction with one of the file modification switches.
/S	Save the month's statistics in the saved statistics area. This option can only be specified on the console terminal. The month's statistics are not affected.

Table 6-9 (continued)

/T	Print the total statistics in addition to the month's statistics.
/U	Update the total statistics. Set them equal to the previous total plus the month's statistics. The month's statistics are zeroed. This option may only be specified on the console terminal.
/Z	Zero the month's statistics used to start a timing period without disrupting other statistics.

A sample run illustrating the use of some of the options is shown in Figure 6-58.

Figure 6-58
Sample Run of USAGE With Options

```
.R USAGE

'USAGE.SV' V4.001  10:29:17 A.M. ON SAT 07-APR-79
*/A

THIS MONTH'S STATISTICS
RUN-TIME = 0000:00:00:03.6
CONNECT-TIME = 000:00:05:41.4
TIME OF LAST LOGIN 10:18:01 A.M. ON SAT 07-APR-79
TIME OF LAST LOGOUT 10:18:38 A.M. ON SAT 07-APR-69

CURRENT STATISTICS
CONNECT-TIME = 000:01:27:26.2
RUN-TIME = 0000:00:00:27.8

TOTAL STATISTICS
RUN-TIME = 0000:01:21.35.2
CONNECT-TIME = 000:00:00:24.1

SAVED STATISTICS
RUN-TIME = 0000:00:00:00.0
CONNECT-TIME = 000:00:00:00.0

OLD STATISTICS
RUN-TIME = 0000:00:00:00.0
CONNECT-TIME = 000:00:00:00.0
*/S/Q
*/A
```

Figure 6-58 (continued)

```
THIS MONTH'S STATISTICS
RUN-TIME = 0000:00:00:03.6
CONNECT-TIME = 000:00:05:41.4
TIME OF LAST LOGIN 10:18:01 A.M. ON SAT 07-APR-79
TIME OF LAST LOGOUT 10:18:38 A.M. ON SAT 07-APR-79

CURRENT STATISTICS
CONNECT-TIME = 000:01:27:26.2
RUN-TIME = 0000:00:00:27.8

TOTAL STATISTICS
RUN-TIME = 0000:01:21.35.2
CONNECT-TIME = 000:00:00:24.1

SAVED STATISTICS
RUN-TIME = 0000:00:00:03.6
CONNECT-TIME = 000:00:05:41.4

OLD STATISTICS
*/V/Q
*/A$                              ($ - escape or alt mode)

THIS MONTH'S STATISTICS
RUN-TIME = 0000:00:00:00.0
CONNECT-TIME = 000:00:05:00.0
TIME OF LAST LOGIN 10:18:01 A.M. ON SAT 07-APR-79
TIME OF LAST LOGOUT 10:18:38 A.M. ON SAT 07-APR-79

CURRENT STATISTICS
CONNECT-TIME = 000:01:27:26.2
RUN-TIME = 0000:00:00:27.8

TOTAL STATISTICS
RUN-TIME = 0000:01:27.26.2
CONNECT-TIME = 000:00:00:27.1

SAVED STATISTICS
RUN-TIME = 0000:00:00:03.6
CONNECT-TIME = 000:00:05:41.4

OLD STATISTICS
.
```

USAGE sets your PRIV word to 4040, closes channel 7 and LOOKS UP a specified account statistics file on channel 7. When you exit the program via an escape or alt mode, channel 7 is closed and your PRIV word is set to 0040.

A list of all possible messages printed by USAGE follows.

?NO STATISTICS AVAILABLE

There is no statistics file for the specified account. You must create an account statistics file via the "/N" option in USAGE.

?PROTECTION VIOLATION

The USAGE program was run from a non-privileged account. You must LOGOUT of the current account, LOGIN to a privileged account and run USAGE.

?X READ ERROR

A READ error occurred while READing the specified account statistics file. "X" corresponds to the errors in the READW CHANIO instruction (see 7.4.1, System User's Guide). If this error occurs, the account statistics file should be initialized with the "/N" option.

?X WRITE ERROR

A WRITE error occurred while WRITing the specified account statistics file. "X" corresponds to the errors in the WRITEW CHANIO instruction (see 7.4.3, System User's Guide). If this error occurs, the account statistics file should be initialized with the "/N" option.

A layout of the account statistics files is contained in Table 6-10.

Table 6-10
Layout of the Account Statistics File

Location	Meaning
0-3	Current month's connect-time. These four words make up a forty-eight bit counter which contains the total LOGIN time for this account in system ticks. (1 second = 10 system ticks.) Location 0 is the low order word and location 3 is the high order word.
4-6	Current month's run-time. These three words make up a thirty-six bit counter which contains the month's total CPU time for this account in system ticks (1 second = 10 system ticks). Location 4 is the low order word and location 6 is the high order word.
7-12	Time of last LOGIN. The format of this time is identical to the current time of day (see 6.7.2, System User's Guide).
13-16	Time of last LOGOUT. The format of this time is identical to the time of last LOGIN.
17-22	Current statistics area's connect-time. The format of these words is identical to the current months' connect-time.
23-25	Current statistics area's run-time. The format of these words is identical to the current month's run-time.
26-31	Total statistics area's connect-time. The format of these words is identical to the current month's connect-time.
32-34	Total statistics area's run-time. The format of these words is identical to the current month's run-time.
35-40	Saved statistics area's connect-time. The format of these words is identical to the current month's connect-time.
41-43	Saved statistics area's run-time. The format of these words is identical to the current month's run-time.

Table 6-10 (continued)

44-47	Old statistics area's connect-time. The format of these words is identical to the current month's connect-time.
50-52	Old statistics area's run-time. The format of these words is identical to the current month's run-time.
53-377	Unused at this time, and may be used for user's purposes.

6.5 PRIVILEGED COS FUNCTIONS

Certain functions which are performed under COS require you to be privileged, i.e., PRIV 40XX. Any access of the RX01 floppy disk is privileged. Changing scope characteristics is privileged. Since writing into another user's COS account may require that you Write Enable the account, transferring files to other accounts may be privileged. The following sections list all standard COS privileged operations.

6.5.1 RX01 Floppy Disk

The RX01 floppy disk is not a data break device. Therefore, it cannot be operated in true timesharing mode. The RX01 floppy disk is supported by an internal COS handler which stops time sharing for short periods of time and executes the RX01 IOTs directly. These periods are so short that they are not normally noticed. However, this device should be used with caution. Its use may cause subtle problems, depending on what other users are doing. There are certain restrictions in the use of the RX01: the job accessing the RX01 must be privileged; only one job at a time can access the RX01; the RX01 should not be used at the same time as the card reader or the card reader will lose characters. Each time the RX01 is used, the ETOS time of day loses a fraction of a second. You can access only the first set of RX01 drives under ETOS. The ETOS names for these drives are DT0 and DT1, not RX0 and RX1. The example of Figure 6-59 copies a source file

from floppy disk drive Ø to floppy disk drive 1.

Figure 6-59
Sample Access of the RXØ1 Under COS

```
.^VPRIV 4040
FETCH FILE1,DTØ
.WRITE FILE1,DT1
.^VPRIV ØØ4Ø
```

6.5.2 VTØ5 and VT52 Scopes

When ETOS COS is BOOTstrapped, it attempts to determine what type of terminal it is running on. VT5Ø and VT52 terminals normally have an answer-back feature. If you send the terminal an ESC Z, the VT5X terminal should respond with ESC/Y, where Y determines the exact type of VT5X. The problem with this structure is that not all VT5X terminals have the answer back feature enabled. Therefore when COS is BOOTed, ETOS assumes that the terminal is a VTØ5 and modifies the DISPLAY code accordingly.

To solve the problem, you can enable the answer-back feature on the terminal or you can modify COS to force the terminal to be treated as a VTØ5 or VT5X. Since each user has his own copy of COS, you must modify COS for each account which will use the terminal. This modification removes some of the flexibility of COS in that it restricts certain accounts to a specified type of terminal. In the distributed system, a COS account may be utilized properly by a VTØ5 or VT5X.

To modify a COS account to disable the answer-back feature, LOGIN to a privileged account and enter the dialogue of Figure 6-6Ø.

Figure 6-60
Disabling the Answer Back Feature Under COS

```
.^VS
!PRIV 4040
!CLOSE 3
!LOOKUP 3=[X,Y]COSDISK.RTS          ([X,Y] is the COS account
!WE 3                                you are modifying)
!PRIV 0040
!CONTINUE
R FUTIL

SET DEVICE CHN3

0.20/  1177  5027<RET>
```

To force the COS account to output VT05 cursor controls, enter the following line

```
70/  7305  5110<RET>
```

To force the COS account to output VT5X cursor controls, enter the following line

```
70/  7305  5114<RET>
```

No matter which type of scope you enabled, enter the following commands to complete the modification.

```
WRITE
EXIT
.^VS
!CLOSE 3
!CONTINUE
```

If you have additional accounts to modify, go back to the first step. In the LOOKUP command, enter the new account number.

6.6 ON-LINE BACKUP

The programs RKCOPY (see 2.5.1) and SICOPY (see 2.5.2) are used to copy one disk to another disk for back-up. Since these programs run under single-user OS/8, you must shut down timesharing to run them. An ETOS program called DKCOPY.SAV may be used to copy one disk to another disk while ETOS is running.

An advantage of this program is that you do not have to interface with time sharing users in order to perform back-up operations. A disadvantage of this program is that since the program runs for a minimum of five minutes, you might copy a directory to the back-up disk which has been updated while the copy is taking place. The directory might not reflect the actual placement of the programs. This mismatched directory situation is not likely to happen often and can be eliminated by BROADCASTing a message requesting users not to enter commands while DKCOPY is running. This situation is preferable to shutting ETOS down, because users can continue processing where they left off, after the copy is completed.

DKCOPY must run as an ETOS file. The OS/8 file SYS:DKCOPY.SV is transferred to an ETOS file DKCOPY.SAV in account [0,2] using the OSETOS program (see 5.12.2).

This program must be run from a privileged account on console 00. There are two methods which can be used in running the program. You can assume that no critical directory entries will take place while the program is running. In this situation, you want DKCOPY to disturb other users as little as possible. Therefore, you set your PRIV word to 4070 by entering ^VPRIV 4070<RET> before you run the copy program. The other method of using DKCOPY is to BROADCAST a message to all users, informing them that they are not to enter commands until another message is BROADCAST. In this situation, you want DKCOPY to run as quickly as possible. Therefore, you set your PRIV word to 4007 by entering ^VPRIV 4007<RET> before you run the copy program.

After you enter one of the two PRIV commands, DKCOPY may be run by following the dialogue contained in Figure 6-61.

Figure 6-61
Sample Initiation of DKCOPY

```
.^VS
!R [0,2]DKCOPY.SAV
'DKCOPY' V1.014

FROM DK #?
```

FIRST USE OSETOS.SAV (see 5.12.2) TO LOAD DKCOPY.

Enter the DK number (0-3) which contains the pack which you wish to copy. On an RK05 system the DK number corresponds to the physical drive number. On a System Industries system, DK1 refers to the removable pack in port 0; DK2 refers to the fixed pack in port 1; DK3 refers to the removable pack in port 2. On both types of systems, DK0 is the system disk. The ETOS system pack must be on DK0 when you are copying. Therefore, you may only copy DK0 if you have a two drive RK05 system or a one drive System Industries system. After the input disk is entered, the system responds with

```
TO DK?:
```

Enter the DK number (1-3) which contains the pack which you wish to copy onto. You cannot copy onto DK0. The pack you are copying onto must be a formatted pack contained in a drive which is write enabled and up to proper speed. The pack must also not be ASSIGNed to MOUNTed by any user.

After you have entered the output DK number, DKCOPY makes a copy of the specified disk. If no errors occur, OS/8 is recalled and BOOTed, causing a "." to appear on the console terminal. At this point, you may wish to BROADCAST a message indicating that the copy is complete. You also should return your PRIV word to its original value of 0040. A sample run of DKCOPY which copies the sytem disk to DK1 is contained in Figure 6-62.

Figure 6-62
Sample Run of DKCOPY

```
.^VPRIV 4007
.^VBROADCAST DO NOT ENTER COMMANDS UNTIL NOTIFIED
.^VS
!R [0,2]DKCOPY.SAV
'DKCOPY' V1.014

FROM DK#? 0          (system disk is input)
TO DK #? 1           (DK1 is output)

.^VBROADCAST RESUME PROCESSING
^VPRIV 0040
```

When DKCOPY is run, it CLOSes channel 0 and LOOKs UP [0,2] DKCOPY.SAV on channel 0. It ASSIGNs output disk and LOOKS UP channel 7 on the output disk. If you are copying from a non-system disk, DKCOPY assigns the input disk and LOOKS UP channel 7 on the input disk. If you are copying from DK0, DKCOPY looks up channel 7 as the file [0,2]IMAGE.RK5. This dummy IMAGE file allows you to access the entire system disk via CHANIO commands. It does not take up disk blocks because it simply contains a map to the entire system disk.

Copying is done in groups of 14 (decimal) blocks. DKCOPY reads 14 blocks from channel 6 into virtual field 0. It then writes the 14 blocks from field 0 to the output disk. For verification, the same 14 blocks are read from the output disk into virtual field 1 and compared word for word with the contents in field 0. Together with the cyclic redundancy checksum (CCC) tests performed routinely by the hardware, the verification procedure provides a strong gaurantee that the output is a true image of the input.

When DKCOPY is finished, channel 6 and 7 are automatically closed. The output disk is DEASSIGNed and the input disk is DEASSIGNed if it is not DK0. Channel 0 is closed on [0,2]OS8.RTS is LOOKed UP on channel 0. OS/8 is BOOTed by jumping to location

00000.

Most errors occurring during a run of DKCOPY are printed on the console terminal. However, if you attempt to run DKCOPY without being privileged or from a terminal other than console 00, the program LOgs you out of the account. All errors which generate error message are listed below.

BAD COPY, BLOCK # BBBB.LLLL

An error occurred while attempting to verify that data written to the output disk is the same as the data read from the input disk. "BBBB" represents tthe starting block number of the 14 block segment. LLLL represents the address in memory which does not verify properly. To compute the physical disk block number, divide LLLL by 400 (base 8). Add the result to BBBB to obtain the block number. This error causes the program to terminate and OS/8 is reBOOTed.

LOOKUP:X = ERROR AT YYYY

An error occurred while attempting to perform a LOOKUP operation. X corresponds to the errors produced by the CHANIO LOOKUP function (see 7.4.6, System User's Guide). YYYY represents the address in field 0 where the error occurred. This address explains what type of LOOKUP was being attempted. Table 6-11 contains a list of LOOKUP error addresses and their meanings.

Table 6-11
DKCOPY LOOKUP Error Messages

Address	Meaning
407	An error occurred while attempting to LOOKUP channel 6 as the input disk if not a system disk or the IMAGE.RKJ file on a system disk. The program is terminated oand OS/8 is reBOOTed.
422	An error occurred while attempting to LOOKUP channel 7 as the output disk. The program is terminated and OS/8 is reBOOTed.
622	An error occurred while attempting to LOOKUP channel Ø as [Ø,2]OS8.RTS upon program termination. This error LOGs you out of the system.

READ: X = ERROR AT YYYY

An error occurred while attempting to perform a READ operation. X corresponds to the errors produced by the CHANIO READW function (see 7.4.1, System User's Guide). YYYY represents the address in field Ø where the error occurred. This address explains what type of READ was being attempted. Table 6-12 contains a list of READ error addresses and their meanings.

Table 6-12
DKCOPY READ Error Addresses

Address	Meaning
27	An error occurred while attempting to read in the rest of the DKCOPY program. Block Ø is automatically read in by the R command.
41	This error LOGs you out of the system.
416	An attempt was made to copy onto DKØ. This error LOGs you out of the system.
44Ø	An error occurred while attempting to READ a 14 block segment from the input disk. The program is terminated and OS/8 is reBOOTed.
456	An error occurred while attempting to READ a 14 block segment from the output disk for verification purposes. The program is terminated and OS/8 is reBOOTed.
622	An error occurred while attempting to READ block Ø of [Ø,2]OS8.RTS into virtual memory. This error LOGs you out of the system.

WRITE: X = ERROR AT YYYY

An error occurred while attempting to perform a WRITE operation. X corresponds to the errors produced by the CHANIO WRITEW function (see 7.4.3), System User's Guide). YYYY represents the address in field Ø where the error occurred. This address explains what type of WRITE was being attempted. Table 6-13 contains a list of WRITE error addresses and their meanings.

Table 6-13
DKCOPY WRITE Error Addresses

Address	Meaning
446	An error occurred while attempting to write a 14-block segment onto the output disk.

6.7 TEMPORARY PRIVILEGE

It is sometimes desirable for a non-privileged user to be able to execute a privileged program. SYSTAT is an example of this type of program. To accomplish this, you must transfer the program to an ETOS file with bit 0 of the protection word set. This protection word is explained in 5.2.1. When you run an ETOS file with bit 0 set in a non-privileged account, this file can execute privileged functions. This privilege is temporary and is removed when the program exists to SCALE, when a programmed XSCALE is executed (see 6.7.11, System User's Guide) or when the program is interrupted by ^V or ^C. This method insures that a user cannot become privileged permanently by running a privileged program. The OSETOS program may be used to create the ETOS file version of an OS/8 core image file. The dialogue of Figure 6-63 illustrates this procedure.

Figure 6-63
Sample Setting of Temporary Privilege

```
.^VS
!PRIV 4040
!CLOSE 3
!ENTER 3=[0,2]TEST.SAV<4050>        (bit 0 set)
!WE 3
!CONT
R OSETOS
*TEST.SV                            (OS/8 assembler program)
TO CHANNEL? 3
.^VS
!R TEST.SAV                         (run resultant ETOS file)
```

If you want a user to be able to execute this ETOS file from OS/8 or COS, make the program XSCALE the R command. SYSTAT performs this XSCALE function.

CHAPTER 7
ETOS REAL TIME PROGRAMMING

7.1 INTRODUCTION

ETOS has been designed to permit simultaneous timesharing and real time programming. The privileged system functions "HOOK" and "UNHOOK" enable and disable an extension to the ETOS interrupt service. Only one extension will be permitted at a given time, but this extension may run a number of real time tasks. The real time extension has priority over all interrupt requests except the CDF, CIF, CIF/JMP, and CIF/JMS requests by user mode programs. Under normal conditions, entry into the real time interrupt extension occurs within 100 microseconds of an interrupt request. Conditions that may delay this time are a system crash or improper coding of a privileged user job running in executive mode which disables the interrupt system for too long a period.

The real time extension is allowed to request the responsibility of handling the system clock. If a real time job uses the clock, it must signal the ETOS system at 100 millisecond time intervals. (the method of signalling is explained later) This time interval is called a system tick.

The system HOOK function immediately locks the user program instruction and data fields in memory, with the user's instruction field assumed to be the field in which the user's real time interrupt service exists. The user must be aware of the differences between referencing virtual fields and real fields. If any field change instructions must be issued in the interrupt service, the real field of any virtual field reference must be determined. This can be easily done by following the sample program's technique.

Upon entry to the real time interrupt service, the FLAGS (GTF) status and the AC have been preserved by the ETOS interrupt service. If the MQ register (or the EAE) is to be used, its status must be preserved and restored by the realtime interrupt entry and exit routines. The background task of the job containing the real time interrupt extension must be very careful not to disturb the extensions's interrupt code, which begins at location X0000. If it does, ETOS will crash. Therefore, that user must be very careful if he runs OS/8, since it uses virtual fields 0 and 1. Remember to disable the ^C^C and ^P intercepts if necessary. Do not type ^VS;BOOT without unhooking the interrupt extension.

If the system clock will not be used, the AC should equal -1 when the HOOK function is issued. Any other value will indicate that the user's real time service is handling the clock. If the clock is being handled, the user must take a skip return from his interrupt service (i.e., ISZ 0000 before exit), every 100 milliseconds. The real time interrupt service must exit via a CDF 10; CIF 0; JMP I 0000 instruction sequence. The AC must be 0 when exiting.

Warning: Do not issue the HLT instructions in executive mode. These will stop the system. Also be aware that you cannot put ODT breakpoints in any EXEC mode routine. RDF, CIF, etc. operate on real fields, not virtual fields.

7.2 SAMPLE PROGRAM

In the Real Time example of Figure 7-1 assume that a user has written an OS/8 program that will run a foreground/background task with interrupt service in virtual field 2 and additional data field in virtual field 3 for a real time buffer. Note that this program locks out two of the available fields for timesharing. This particular example should be run only on systems with 24K+ memory. The example is very inefficient in core usage and is used only for illustrative purposes.

The real time task gathers data at each clock tick and stores it in virtual field 3. The background user program doubles buffers the output to a file on DSK:. The sample rate for the real time task is 100 HZ. Every 10 seconds, the background task tests for buffer full condition (10 seconds is roughly half the time to fill one buffer).

Figure 7-1
Sample Real-Time Program

```
SYSCAL=6200     /SYSTEM CALL FUNCTION IOT
W100MS=2        /WAIT 100 MILLISECONDS FUNCTION
HOOK=6          /HOOK INTO INTERRUPT SERVICE
UNHOOK=7        /UNHOOK...
CUF=6264        /CLEAR USER FLAG (IF PRIVILEGED)
SUF=6274        /SET USER FLAG
SETSTAT=6047    /SET STATUS
      FIELD 0
      *200
START,CLA CLL
      TAD       (STTARG
      SETSTAT             /RESET ^C^C RESTART
      CIF 20
      JMS       SETUP     /SETUP INTERRUPT SERVICE (FIELD 2)
LOOP, JMS       SLEEP     /SLEEP FOR 10 SECONDS
      TAD I     (PTR
      SMA CLA             /ARE WE FILLING THE SECOND BUFFER YET?
      JMP       LOOP      /NOT YET
      JMS       WRITE
LOOP2,JMS       SLEEP
      TAD I     (PTR
      SPA CLA             /STARTED TO FILL FIRST BUFFER YET?
```

Figure 7-1 (continued)

```
        JMP     LOOP 2
        JMS     WRITE
        ISZ     BUFCNT      /DOUBLE BUFFER COUNT
        JMP     LOOP        /CONTINUE TAKING SAMPLES

DONE,   TAD     (UNHOOK     /UNHOOK INTERRUPT SERVICE
        MQL
        SYSCAL
        JMP I   (7600       GO BACK TO OS/8

BUFCNT,X                    /- # OF 4K BUFFERS TO SAMPLE
STTARG,4375     /DUPL,ENCTLC,ENCTLO,ENCTLS,ENCTLU,ENCTLV,ENRUBO
        607     /BREAK MASK
        DONE    /!C RESTART ADDRESS
        0       /NO !P RESTART ADDRESS

SLEEP,0000
        TAD     (W100MS
        MQL
DECIMAL
        TAD     (-100
OCTAL
        SYSCAL              /SLEEP FOR 10 SECONDS
        JMP I   SLEEP

WRITE,0000
        CDF 0
        JMS I   OUTDEV      /SET UP SOME TIME EARLIER
        6030                /WRITE 20 PAGES FROM FIELD 3
BUFADD,0                    /BUFFER ADDRESS
BLKADD,.-.                  /BLOCK ADDRESS (SETUP EARLIER)
        JMP     ERROR
        CDF 20              /SET FOR FIELD 2 TEST ON PTR
        TAD     BUFADD
        RAL
        CML RAR             /SWITCH BUFFERS
        DCA     BUFADD
        TAD     BULKADD
        TAD     (10         /10 BLOCKS PER BUFFER
        DCA     BLKADD
        JMP I   WRITE
ERROR,.....
        JMP     DONE        /DON'T FORGET TO UNHOOK INTERRUPT SERVICE
/***********************************
FIELD 2                     /PUT INTERRUPT SERVICE IN VIRTUAL FIELD 2

*0
INTRUP,0000
        JMP I   [INTSRV

*10
PTR,7777                    /USE AUTOINDEX
```

Figure 7-1 (continued)

```
                        /NOTE THAT IF THE SLEEP DOES NOT PRECEDE
                        /THE TEST ON THE BUFFER IN THE FIELD 0
                        /BACKGROUND TASK, THERE COULD BE A POTEN-
                        /TIAL PROBLEM UNTIL THE FIRST SAMPLE IS
                        /TAKEN.

*200
SETUP,0000
      CDF 30            /LOCK IN VIRTUAL FIELD 3
      TAD       (HOOK
      MQL               /NOTE AC NOT -1, USING CLOCK
      SYSCAL            /HOOK IN THE SKIP CHAIN
      CLA               /CLA MUST FOLLOW HOOK
      CUF               /CLEAR THE USER FLAG
      RDF               /THIS WILL READ THE REAL DATA FIELD
      TAD       (CDF
      DCA       BUFCDF  /SET IN CDF TO BUFFER FIELD AND ENABLE
                        /REAL TIME SERVICE
      SUF               /SET USER FLAG BACK ON (ON NEXT JMP)
      JMP       .+1     /FORCE USER FLAG ON
      CIF               /RETURN TO VIRTUAL FIELD 0
      JMP I     SETUP   /BACKGROUND TASK

INTSRV,CLSK             /IS INTERRUPT THE CLOCK?
      JMP       XIT     /NO
      ISZ       SYSTIC  /KEEP TRACK OF SYSTEM TICKS
      JMP       BUFCDF  /NOT A SYSTEM TICK
      ISZ       0000    /TAKE SKIP RETURN ON SYSTEM TICK
      TAD       (-12    /10 TICKS PER SYSTEM TICK
      DCA       SYSTIC
BUFCDF,JMP      XIT     /NOTE, WHEN THE REAL FIELD FOR THE
                        /VIRTUAL FIELD 3 IS DETERMINED THIS
                        /INSTRUCTION BECOMES A CDF TO THAT
                        /FIELD, AND SAMPLING BEGINS
      (SAMPLE IOT)
      DCA I     PTR     /STORE ONE DATUM IN BUFFER
XIT,  CDF 10            /DATA FIELD MUST BE 10 FOR EXIT
      CIF 0             /INSTRUCTION FIELD 0
      JMP I     0000    /AND RETURN TO RESIDENT MONITOR

SYSTIC,-12
$
```

7.3 SYSTEM STRUCTURE

ETOS is a collection of routines that share the time of a single PDP-8 processor. The functions that ETOS performs include: Device and extended memory reference IOT emulation, file mapping, task scheduling, and data buffering. A simple discussion of the fundamental aspects of ETOS follows. A thorough understanding of this material is necessary for efficient use of the real time linkage option. It is also necessary to understand this material to be able to write a system disk handler. For more information and specific implementations of some of the ideas presented, consult the file PRVFNC.PA in account [Ø,1Ø] on the distribution pack.

7.4 SCHEDULING

The ETOS monitor uses a software simulation of variable level priority. In this case, "priority" does not refer to an interrupt priority. For this reason, the word "level" will be used instead.

7.4.1 Levels

ETOS operates on four levels. This structure allows scheduling to be simplified and codified; it strictly defines program execution flow and eliminates many race conditions. Details of the funcational characteristics of the four levels follow. It is recommended that this description be read several times.

Level Ø

This is the condition of the processor immediately following an interrupt. The user flag is cleared and the interrupt is off. The data field is Ø and the instruction field is Ø. The following conditions are checked as possibly processed: user

executing a virtual CDF (executing a CDF IOT on level 3); user executing a virtual IOT, HLT, or OSR; user executing a virtual CIF; and user executing a JMP/JMS while a virtual CIF is pending.

Level 1

Conditions are the same as level Ø except that the data field is 1. While on level 1, ETOS executes code to resolve the interrupt.

Level 2

On level 2, the data field is 1, instruction field is Ø, the user flag is cleared, and the interrupt is on. This level is the one on which the bulk of computation of the resident monitor is performed. Because the interrupt is on, level Ø and level 1 routines may be activated.

Level 3

This is the level designated for user tasks. The interrupt is on and the user flag is set. As a result of the foregoing, all extended memory references are mapped; the monitor may suspend the user's task; the user's task may be swapped to the disk; and all IOTs and HLTs are emulated.

7.4.2 Scheduling Algorithms

While on level 2 or 3, a machine interrupt causes a hardware JMS Ø to field Ø. The machine is then, by definition, running on level Ø. If the level Ø interrupt processor cannot resolve the interrupt, level 1 is entered. If there is a real time task present, the level 1 processor immediately activates it. If the real time task cannot resolve the interrupt, the level 1 processor polls the active devices attached to the machine.

This polling process may take a long time (typically 500 microseconds maximum). For this reason, a high priority interrupt routine is called within 100 microsecond intervals. This routine checks the disk units and, if present, the real time task. This is the routine called upon entry to level 1. After the interrupting device has been identified, it is serviced. If the processing required is minimal, then all of the computation for the servicing occurs at this point. The interrupt is then dismissed and the level 2 or 3 task is resumed. If there is a large amount of computation the level 1 routine enters level 2. This is peformed by calling an internal monitor routine.

If the system was running on level 2 at the time of the interrupt, the new level 2 routine execution is deferred by queuing it into a FIFO buffer called the level 2 Queue. If the system was executing a level 3 task, the machine state is preserved so that upon completion of the last level 2 function, the level 3 task may be resumed.

If it is necessary for the monitor to execute the level 2 service as soon as possible, three words are available to store the address of the desired routine. These words are checked before the queue. Two of these words are dedicated to the disk unit, while one of them is for the real time task.

This structure allows ETOS to run with interrupts on most of the time. It also prevents many race conditions from occuring.

Scheduling of level 3 tasks is performed independently of the above process. The routine to select the next level 3 task is referred to as the scheduler. The scheduler may be activated for many reasons. If the current level 3 task is compute bound the scheduler will run whenever the task has used up its time quantum. In this case, the clock interrupt will be the event causing execution. The sequence is

1. Clock interrupts
2. Level 1 detects the clock interrupt
3. If ths is a system tick (these occur every 1/10 second) the level 2 clock routine is entered. If this is not a system tick a counter is incremented and the interrupt is dismissed.
4. The level 2 clock processor determines if the current level 3 task has run its maximum amount of time. If so, the level 2 clock processor queues the scheduler for eventual execution.
5. The scheduler eventually runs and selects the next job.

The scheduler may be run for other reasons. For example, if the user executes an extended memory reference instruction and it is necessary to swap in a user's field, the level 3 task is suspended and the scheduler is called to select a resident runnable job, while the original level 3 task is swapping.

7.5 MONITOR AND MEMORY ARCHITECTURE

ETOS uses 24K, but not all of ETOS needs to be resident. ETOS is divided into 3 logical sections of 8K. These sections are RMON (resident monitor), KMON or SCALE (keyboard monitor or system command and login logout executive) and DMON (disk monitor). A description of each of these components follows.

RMON Resident Monitor

The resident monitor is the nucleus of the system. It's responsibilities include buffering I/O, emulating IOT's and scheduling.

KMON Keyboard Monitor

KMON is the user interface to the system. KMON interprets and acts upon commands made to it. KMON allows users to access and set the registers in their virtual machine, access and set real and virtual core, manipulate their I/O channels, and attach to other jobs.

DMON Disk Monitor

DMON is responsible for processing window turns and non-I/O disk requests. A window turn is the rearranging of the monitor data base to allow a file request to be mapped. A non-I/O disk request is a CHANIO function such as LOOKUP, ENTER, EXTEND, REDUCE, etc., or SYSCAL function MOUNT or DISMNT.

RMON executes on levels Ø, 1, and 2. It is permanently resident in fields Ø and 1. KMON and DMON run on levels 3 and 2.

When they execute on level 3, they run just like any other user job. This allows the system to perform needed level 2 processing. This also allows them to access their own internal multi-field data bases. Because they are privileged they may request the resident monitor to run them as a level 2 task instead of a level 3 job. In this case the level 2 task will not be running in physical field Ø. Since level 2 runs in executive mode, the entire architecture of the operating environment changes. The field references are no longer virtual and mapped but are real and unmapped. IOT's are directly executed. Since the task is running on level 2 the scheduler's execution if requested will be deferred. This, coupled with the fact that all level 3 job swaps originate from the scheduler means that the program will never swap.

To restore level 3 status (which is necessary to avoid a level 2 queue overflow), a monitor routine is called.

7.6 REAL TIME TASKS

ETOS provides the capability to execute one user written real time task. Any privileged user may do this by issuing the 'HOOK' SYSCAL. This makes a portion of the user's program an extension of the resident monitor during interrupt processing.

While running as the real time task, the user's program executes on level 1. Note that the user's level 3 task remains unchanged (i.e., it is still active). The real time task is activated when ETOS enters level 1. Since the level 3 task and the level 1 task (the real time task) occupy the same virtual machine, the level 3 task must be very careful not to destroy the level 1 program area. For example, a SETSTAT should be issued before the HOOK so that a ^C^C doesn't bootstrap OS/8 over the real time task area. If this capability is wanted [boot-strapping OS/8] the real time task should be located in a virtual field greater than 1. This location will somewhat protect the level 1 task from the level 2 task. Note that the OS/8 bootstrap destroys location 3646 in the highest virtual field.

The amount of time ETOS remains on level 1 should be minimized. If too much time is spent on level 1, interrupt servicing for devices will be degraded. To give the real time task maximum flexibility, a monitor call exists to put the real time task on level 2. At this point the interrupt will be on so that devices will operate at maximum throughput.

There is a special high-priority word in the level 2 queue reserved for the real-time task. When the real time task requests to enter level 2, the sequence is as follows.

1. Real-time task issues a level 2 entry request.

2. If level 2 was not running at the time of the real time interrupt, the real time task enters level 2 immediately.

3. If level 2 was running, execution of the real time tasks's level 2 service must be deferred. The monitor linkage routine checks to see if the high priority word for the real time task in the level 2 queue is available. If it is, it will be used, and the real-time task will receive level 2 service next (before all other items in the queue, but after high-priority disk service).

4. If the high priority word is in use, the request enters the level 2 queue just like any other resident monitor level 2 entry. The real time task receives level 2 service after all other items in the queue.

When the level 2 processing is complete, the level 2 routine must release control to ETOS. This is done by calling another monitor routine. This call signifies the end of processing by the real time task. Before the real time task enters level 2, the MQ is not available unless it is preserved and restored upon return to the monitor.

7.7 PRIVILEGED LEVEL 3 PROGRAMMING

Privileged level 3 jobs are allowed to alter the environment in which their virtual machine is running. One of the ways this can be accomplished is by using the real time task. Two other methods are available.

The simplest of these is the use of the CUF (Clear User Flag) instruction. This instruction causes the user's virtual machine to run in executive mode. Therefore, IOTs are no longer trapped and emulated but are directly executed. Any change field instructions go to real fields instead of virtual fields. A TLS will directly affect the console with the Ø4 device code.

Since the interrupt is on, the clock generates interrupts. At

system tick time the scheduler runs and the user's field may be swapped. When it is swapped back in there is no restriction on where it is placed; therefore, the real field may change. For this reason, if the field number is critical, an IOF should be executed wherever a swap is undesirable. To restore normal level 3 status, the SUF IOT should be used. The user flag is not set until the next JMP or JMS. Interrupts are inhibited between a CIF and the JMP/JMS that effects the CIF.

For a full discussion of user mode/executive mode see page 5-18 in the Small Computer Handbook from Digital Equipment Corporation.

The second method of altering the environment is to request the monitor to execute its job on level 2. This action is accomplished by issuing the 6007 IOT. In executive mode, this IOT is called CAF (Clear All Flags). In the ETOS level 3 environment, this IOT is called RMON. Control returns to the location following the RMON instruction. ETOS is then operating on level 2. The data field is initially set to real field 1 but this may be changed to any field. Since the machine is running on level 2 the program will not be swapped. The interrupt is on so devices will be serviced. The user must insure that his program does not remain on level 2 too long. If this happens, the level 2 queue will overflow, causing monitor failure. Reinitialization of ETOS is necessary.

To return to level 3, a monitor routine is called.

7.8 REAL-TIME DEMONSTRATION

As an illustration of the real-time facilities of ETOS, the program RTDEMO.PA is included on the distribution pack in account [0,10]. This program allows you to play a blockade game on a VT52 terminal. The game starts up with a twenty-four by eighty

position grid. There are two players on the VT52. The left-hand player starts in the left-hand side of the screen in the twelfth row.

The right-hand player starts at the extreme right-hand side of the screen in the twelfth row. Every second, two asterisks (*) are placed on the screen. One asterisk is placed next to the last asterisk displayed for each player. In the beginning of the game, the left-hand player's asterisks are placed to the direct right of the last asterisk. The right-hand player's asterisks are placed to the direct left of the last asterisk. If any player attempts to place an asterisk in a position which already has an asterisk or in a position off the screen, the player loses the point and the game starts over again. Each player can change the direction of the placing of the asterisks by hitting a particular key on the VT52. The keys and their directions are listed in Table 7-1.

Table 7-1
RTDEMO's Keys and Directions

Left Player Key	Right Player Key	Direction
W	8	UP
E	9	UP-RIGHT
D	6	RIGHT
C	3	DOWN-RIGHT
X	2	DOWN
Z	1	DOWN-LEFT
A	4	LEFT
Q	7	UP-LEFT

To stop the game, press the carriage return key on the VT52. To list this demonstration program on a terminal, LOG INto account [Ø,1Ø] on a copy of the distribution pack and enter TYPE RTDEMO.PA<RET>.

To assemble the program and save it in your account, copy the

source program to your account storage area and enter the commands of Figure 7-2.

Figure 7-2
Compiling and Saving RTDEMO

```
.COMPILE RTDEMO

LINKS GENERATED:Ø
ERRORS DETECTED:Ø
.LOAD RTDEMO
.SAVE DSK:RTDEMO
.
```