ETOS

Version 5B

SYSTEM USER'S GUIDE

PREFACE

The ETOS User's Manual contains the basic information necessary for running under ETOS. For easy reference the appendices include a summary of SCALE commands, error messages, a demonstration run, and a glossary.

Various references were helpful in the preparation of the glossary. These includes OS/8 Handbook, DEC Press, 1974; Small Computer Handbook, DEC Press, 1973; Computer Dictionary, by Donald D. Spencer, Abacus Computer Corporation, 1973; Computer Dictionary and Handbook, by Charles J. Sippl and Charles P. Sippl, Howard W. Sams & Co., Inc., 1972; and COS 300 System Reference Manual, DEC Press, 1972.

TABLE OF CONTENTS

SYSTEM OVERVIEW	1- 1
1.1 INTRODUCTION	1- 1
1.2 SYSTEM ARCHITECTURE	1- 2
1.3 CONVENTIONS USED IN THIS MANUAL	1- 5
USING ETOS	2- 1
2.1 INTRODUCTION	2- 1
2.2 GETTING ON-LINE	2- 1
2.2.1 Account Numbers	2- 1
2.2.2 Logging into ETOS	2- 2
2.2.3 Messages	2- 6
2.3 CALLING PROGRAMS	2- 8
2.3.1 Calling OS/8 Programs	2- 8
2.3.2 Calling COS Programs	2- 9
2.4 MODES OF OPERATION	2-11
2.5 EXITING THE SYSTEM	2-12
THE MONITOR AND SCALE COMMANDS	3- 1
3.1 INTRODUCTION	3- 1
3.2 HELLO, LOGIN AND LOGOUT COMMANDS	3- 2
3.3 STOP AND CONTINUE COMMANDS	3- 3
3.4 CORE COMMAND	3- 6
3.5 BOOTSTRAP COMMAND	3- 7
3.5.1 OS/8 Bootstrap	3- 7
3.5.2 COS Bootstrap	3- 8
3.6 R COMMAND	3- 8
3.7 VIRTUAL REGISTER COMMANDS	3- 9
3.7.1 AC, PC and MQ Commands	3-10
3.7.2 FLAGS Command	3-10
3.7.3 SC Command	3-11
3.7.4 SR Command	3-11
3.7.5 BREAK Command	3-12

3.7.6 KSTAT Command 3-13
3.7.7 WHERE Command 3-15
3.8 EXAMINE AND DEPOSIT COMMANDS 3-15
3.9 TALK COMMAND 3-17
3.10 ASSIGN AND DEASSIGN COMMANDS 3-18
3.11 QUEUE COMMAND 3-20
3.12 VERSION COMMAND 3-21
3.13 DETACH AND ATTACH COMMANDS 3-21
3.14 SYSTAT COMMAND 3-23
3.15 ^C AND ^P COMMAND 3-23
3.16 RESTRICTIONS ON THE USE OF THE COMMANDS 3-24
3.17 SCALE ERROR MESSAGES 3-25
3.18 EFFECTS OF NO FREE BUFFERS ON SCALE 3-28
3.19 TTY READER CONTROL 3-28

DIFFERENCES IN OS/8 4- 1
4.1 INTRODUCTION 4- 1
4.2 OS/8 MONITOR CHANGES 4- 2
4.3 CHANGES TO OS/8 SYSTEM PROGRAMS 4- 5
4.4 CONVERTING STAND-ALONE OS/8 PROGRAMS TO ETOS 4-14
4.5 SYSTAT PROGRAM 4-17
4.6 TIME PROGRAM 4-20
4.7 WHO PROGRAM 4-22
4.8 INQUIR PROGRAM 4-23
4.9 LINE PRINTER AND TERMINAL SPOOLING 4-23

DIFFERENCES IN COS 5- 1
5.1 INTRODUCTION 5- 1
5.2 COS MONITOR CHANGES 5- 2
5.3 CHANGES TO COS SYSTEM PROGRAMS 5- 5
5.4 CONVERTING STAND-ALONE COS PROGRAMS TO ETOS 5- 6
5.5 SYSGET PROGRAM 5- 8
5.5.1 Operating Instructions 5- 8
5.5.2 Input Sysgen Table 5- 8
5.5.3 Space Allocation Algorithms 5-11

5.5.4 Error Messages 5-13
5.6 BREAK AND UNBRK PROGRAMS 5-17

ASSEMBLER PROGRAMMING UNDER ETOS 6- 1
6.1 INTRODUCTION 6- 1
6.2 TERMINAL IOT 6- 2
6.2.1 KCF 6- 2
6.2.2 KSF 6- 2
6.2.3 KCC 6- 2
6.2.4 KCL 6- 2
6.2.5 KRS 6- 2
6.2.6 KRB 6- 3
6.2.7 GETSTAT 6- 3
6.2.8 TSF 6- 3
6.2.9 TCF 6- 4
6.2.10 TCL 6- 4
6.2.11 TPC 6- 4
6.2.12 TSK 6- 4
6.2.13 TLS 6- 4
6.2.14 SETSTAT 6- 4
6.3 EXTENDED MEMORY IOTs 6- 5
6.3.1 CDF 6- 5
6.3.2 CIF 6- 6
6.3.3 CIF CDF 6- 6
6.3.4 RDF 6- 6
6.3.5 RIF 6- 6
6.4 TIME SHARING IOTs 6- 6
6.5 CARD READER IOTs 6- 7
6.5.1 RCRA 6- 7
6.5.2 RCRP 6- 7
6.5.3 RCRB 6- 8
6.5.4 RCRC 6- 8
6.5.5 RCCR 6- 8
6.5.6 Illegal Card Reader IOTs 6- 9
6.6 LINE PRINTER IOTs 6-10

6.6.1 LSF 6-10
6.6.2 LLS 6-10
6.6.3 Miscellaneous Line Printer IOTs 6-10
6.7 SYSTEM FUNCTIONS 6-11
6.7.1 SSR 6-12
6.7.2 TOD 6-12
6.7.3 W100MS 6-13
6.7.4 W1M1N 6-14
6.7.5 WAIT 6-14
6.7.6 DATAPK 6-15
6.7.7 VERSION 6-19
6.7.8 TXTSET 6-20
6.7.9 LOGOUT 6-20
6.7.10 JOBNUM 6-21
6.7.11 XSCALE 6-21
6.7.12 ENDRUN 6-22
6.7.13 CORE 6-22
6.8 MISCELLANEOUS IOTS 6-25
6.8.1 SGT 6-25
6.8.2 HLT 6-25
6.8.3 OSR 6-25
6.9 UNIMPLEMENTED IOTS 6-25

ETOS CHANNELS AND ASSEMBLER PROGRAMMING OF CHANNELS
7.1 INTRODUCTION 7- 1
7.2 CHANNEL INPUT/OUTPUT COMMANDS 7- 3
7.2.1 READ Command 7- 4
7.2.2 WRITE Command 7- 5
7.2.3 ENTER Command 7- 5
7.2.4 LOOKUP Command 7- 8
7.2.5 EXTEND Command 7- 9
7.2.6 REDUCE Command 7-11
7.2.7 DELETE Command 7-12
7.2.8 RENAME Command 7-13
7.2.9 CLOSE Command 7-13

7.2.10 INQUIRE Command 7-14
7.3 RESTRICTIONS ON THE USE OF CHANNEL COMMANDS 7-15
7.4 ASSEMBLER PROGRAMMING OF CHANNEL INPUT/OUTPUT IOTS 7-16
7.4.1 READW 7-16
7.4.2 READC 7-18
7.4.3 WRITEW 7-18
7.4.4 WRITEC 7-19
7.4.5 ENTER 7-19
7.4.6 LOOKUP 7-22
7.4.7 EXTEND 7-24
7.4.8 REDUCE 7-26
7.4.9 DELETE 7-28
7.4.10 RENAME 7-28
7.4.11 WAITW 7-30
7.4.12 WAITC 7-31
7.4.13 CLOSE 7-32
7.4.14 INQUIRE 7-32

APPENDICES

Appendix A -- SCALE Commands A-1
Appendix B -- SCALE Error Messages B-1
Appendix C -- OS/8 Error Messages C-1
Appendix D -- COS Error Messages D-1
Appendix E -- OS/8 Character Codes E-1
Appendix F -- COS Character Codes F-1
Appendix G -- Demonstration Run G-1
Appendix H -- Glossary H-1

INDEX

READER'S COMMENT FORM

List of Figures

Figure 2-1	Sample LOGIN	2- 4
Figure 2-2	Alternate LOGIN Sample	2- 4
Figure 2-3	Sample Call of OS/8 Programs	2- 9
Figure 2-4	Sample Call of COS Programs	2-10
Figure 2-5	Exiting the System	2-12
Figure 2-6	Exiting the System From OS/8	2-12
Figure 3-1	SCALE Semicolon Construction	3- 1
Figure 3-2	Sample LOGOUT	3- 3
Figure 3-3	Sample LOGOUT From OS/8	3- 3
Figure 3-4	Sample CORE Command	3- 6
Figure 3-5	Sample BOOT Command	3- 7
Figure 3-6	Sample Virtual Register Command	3- 9
Figure 3-7	Sample WHERE Command	3-15
Figure 3-8	Sample EXAMINE Command	3-16
Figure 3-9	Sample DEPOSIT Command	3-16
Figure 3-10	Sample ASSIGN and DEASSIGN	3-19
Figure 3-11	Sample QUEUE Command	3-20
Figure 3-12	Sample VERSION Command	3-21
Figure 3-13	Sample DETACH Command	3-21
Figure 3-14	Sample ATTACH Command	3-22
Figure 4-1	Sample Use of Type-Ahead	4- 3
Figure 4-2	Sample OS/8 Batch	4- 7
Figure 4-3	EPIC Example	4- 9
Figure 4-4	Sample Use of FUTIL "A" Variable	4-11
Figure 4-5	Sample Use of OS/8 ASSIGN	4-15
Figure 4-6	Sample Run of SYSTAT	4-17
Figure 4-7	Three Alternate Systat Uses	4-20
Figure 4-8	Sample Run of Time	4-21
Figure 4-9	Output of Time to a File	4-22
Figure 4-10	Sample Run of Who	4-22
Figure 4-11	Sample Run of INQUIR	4-23
Figure 4-12	Sample Use of Spooling	4-24
Figure 4-13	Sample Spool from SCALE	4-25
Figure 5-1	Sample Use of Type-Ahead	5- 3
Figure 5-2	Sample Run of SYSGET	5- 8
Figure 5-3	Sample SYSGET Table	5-10
Figure 5-4	SYSGET Error Format	5-14
Figure 5-5	BREAK and UNBRK Example	5-18
Figure 6-1	Sample Use of GETSTAT	6- 3
Figure 6-2	Sample Use of SETSTAT	6- 5
Figure 6-3	Sample Use of SKETOS	6- 7
Figure 6-4	Sample Use of ETOS Card Reader IOTs	6- 9
Figure 6-5	Sample Use of SSR	6-12
Figure 6-6	Sample Use of TOD	6-13
Figure 6-7	Sample Use of W100MS	6-13
Figure 6-8	Sample Use of WAIT	6-14
Figure 6-9	Sample Use of DATAPK	6-18
Figure 6-10	Sample Use of VERSION	6-19
Figure 6-11	Sample Use of TXTSET	6-20
Figure 6-12	Sample Use of LOGOUT	6-20

Figure 6-13 Sample Use of JOBNUM 6-21
Figure 6-14 Sample Use of XSCALE 6-21
Figure 6-15 Sample Use of ENDRUN 6-22
Figure 6-16 Sample Use of CORE 6-22
Figure 6-17 OS/8 CORE Usage 6-23
Figure 6-18 Batch CORE Usage 6-24
Figure 7-1 Sample SCALE READ 7- 4
Figure 7-2 Sample SCALE WRITE 7- 5
Figure 7-3 Sample SCALE Variable Lenth Enter 7- 7
Figure 7-4 Sample SCALE Fixed Length Enter 7- 7
Figure 7-5 Sample SCALE File Structured LOOKUP 7- 8
Figure 7-6 Sample SCALE Non-File Structured LOOKUP 7- 9
Figure 7-7 Sample SCALE EXTEND 7-11
Figure 7-8 Sample SCALE REDUCE 7-12
Figure 7-9 Sample SCALE DELETE 7-12
Figure 7-10 Sample SCALE RENAME 7-13
Figure 7-11 Sample SCALE CLOSE 7-14
Figure 7-12 Sample SCALE INQUIRE 7-14
Figure 7-13 Sample CHANIO READW 7-17
Figure 7-14 Sample CHANIO WRITEW 7-18
Figure 7-15 Sample CHANIO ENTER 7-21
Figure 7-16 Sample CHANIO File Structured Lookup 7-23
Figure 7-17 Sample CHANIO Non-File Structured Lookup 7-24
Figure 7-18 Sample CHANIO EXTEND 7-26
Figure 7-19 Sample CHANIO REDUCE 7-27
Figure 7-20 Sample CHANIO DELETE 7-28
Figure 7-21 Sample CHANIO RENAME 7-30
Figure 7-22 Sample CHANIO WAITW 7-31
Figure 7-23 Sample CHANIO WAITC 7-31
Figure 7-24 Sample CHANIO CLOSE 7-32
Figure 7-25 Sample CHANIO INQUIRE 7-34
Figure G-1 Demonstration Run #1 G-2
Figure G-2 Demonstration Run #2 G-3
Figure G-3 Demonstration Run #3 G-3

List of Tables

Table 3-1	Flags Word Format	3-10
Table 3-2	BREAK Mask Bit Assignment	3-13
Table 3-3	KSTAT Register Bit Assignement	3-14
Table 4-1	OS/8 Programs Not Supported Under ETOS	4- 5
Table 4-2	DIRECT Options	4- 9
Table 4-3	SYSTAT States	4-19
Table 4-4	SYSTAT Options	4-19
Table 4-5	TIME Options	4-21
Table 5-1	COS Programs Not Supported Under ETOS	5- 5
Table 6-1	Illegal Card Reader IOTs	6- 9
Table 6-2	System Functions	6-11
Table 6-3	Wait Bits	6-14
Table 6-4	Privilege Word Format	6-15
Table 6-5	EAE Register Format	6-16
Table 6-6	Teleprinter Status Format	6-17
Table 7-1	Channel I/O Defaults	7- 3
Table 7-2	SCALE READ Errors	7- 4
Table 7-3	SCALE WRITE Errors	7- 5
Table 7-4	Protection Code Format	7- 6
Table 7-5	Scale ENTER Errors	7- 7
Table 7-6	Scale File Structured Lookup Errors	7- 8
Table 7-7	Scale Non-File Structured Lookup Errors	7- 9
Table 7-8	Scale Extend Errors	7-10
Table 7-9	Scale Reduce Errors	7-11
Table 7-10	Scale Delete Errors	7-12
Table 7-11	Scale Rename Errors	7-13
Table 7-12	CHANIO Functions	7-16
Table 7-13	CHANIO READW Errors	7-17
Table 7-14	CHANIO WRITEW Errors	7-18
Table 7-15	Protection Code	7-19
Table 7-16	Date Word Format	7-20
Table 7-17	CHANIO Enter Errors	7-20
Table 7-18	CHANIO File Structured Lookup Errors	7-22
Table 7-19	CHANIO Non-File Structured Lookup Errors	7-24
Table 7-20	CHANIO Extend Errors	7-25
Table 7-21	CHANIO Reduce Errors	7-27
Table 7-22	CHANIO Delete Errors	7-28
Table 7-23	Protection Code Format	7-29
Table 7-24	Date Word Format	7-29
Table 7-25	CHANIO Rename Errors	7-29
Table 7-26	CHANIO WAITW Errors	7-30
Table 7-27	CHANIO WAITC Errors	7-31
Table 7-28	Protection Code Format	7-33
Table 7-29	CHANIO Inquire Errors	7-34
Table A-1	Non-Privileged Scale Commands	A-1
Table B-1	Scale Error Messages	B-1
Table C-1	OS/8 Error Messages	C-1
Table D-1	COS Error Messages	D-1
Table E-1	OS/8 Character Codes	E-1
Table F-1	COS Character Codes	F-1

CHAPTER 1
SYSTEM OVERVIEW

1.1 INTRODUCTION

The Extended Timeshared Operating System (ETOS) is a powerful, multi-programming system for the Digital Equipment Corporation PDP-8/E, /F, /M and /A computers. The system provides multiple users with shared access to the central processor (computer) and its peripherals (mass storage devices, line printer, and so on). Each user terminal is provided with a complete "virtual" PDP-8/E; that is, the operating system makes it appear to each user that an entire computer is at their disposal. The system executes programs originally written for a stand-alone omnibus PDP-8 with up to 32K words of internal memory, even if the physical machine in use has less memory. A wide range of peripherals is supported by ETOS and is present on each virtual machine, and therefore is available to each user. The system manager can restrict the use of these peripherals.

The functions normally performed by the front panel controls of a stand-alone machine are available at each terminal through the System Command and Login/Logout Executive (SCALE), a system job which simulates these functions. Information about SCALE commands is presented in Chapter 3 (The Monitor and Scale Commands).

ETOS derives its power from OS/8 and COS, Digital Equipment Corporation's single-user operating systems for the PDP-8 family of computers. OS/8 is available to each ETOS user on a virtual machine (VM), giving a wide range of system programs, a Keyboard Monitor, command Decoder, User Service Routine, and device handlers.

The system manager can also provide the user with accessibility

to COS, which provides the capability to run DIBOL. Unlike OS/8, COS is not automatically available to all users. In addition to the facilities provided by OS/8 or COS, each user has accessibility to the special monitor facilities provided by SCALE.

It is assumed in what follows that the reader has some familiarity with PDP-8 programming in general and with the OS/8 operating system in particular (see Introduction to Programming, DEC Press, 1972, OS/8 Handbook, DEC Press, 1974, and Small Computer Handbook, DEC Press, 1973). The COS user should have some familiarity with the COS System Reference Manual, DEC Press, 1972 and will require only enough knowledge of OS/8 to run a program.

1.2 SYSTEM ARCHITECTURE

This section explains the architecture of ETOS. This information is useful but it is not necessary to comprehend all of the ideas which are presented.

ETOS is composed of a number of related programs collectively called the monitor. The monitor manages input/output, scheduling, and memory utilization to effect time sharing. Each active task runs for a short period of time (a "quantum") repetitively; to users, it appears that all tasks are running at the same time. Most of these managerial functions are performed by that part of the monitor which is memory-resident in physical fields Ø and 1. However, SCALE and some of the functions of the virtual disk handlers are not resident. These run as special system jobs. They are ordinary jobs except that (1) they are invoked by the resident monitor to perform specific functions, (2) they can run in executive mode, as does the resident monitor.

User programs normally run in "User Mode", a special hardware

mode that disables input/output instructions (IOTs), change field instructions (CIFs), halt (HLT), and switch register instructions (OSR) from having their usual effect. Instead, these instructions cause interrupts, or traps, to the resident monitor, which then processes them in its own way. For example, a "halt" instruction does not stop the machine (and thus stop all virtual machines) when executed by a user program. Rather, it stops only the virtual machine that executed it. SCALE is called to print "?%VM HLT" (virtual machine halt) on the terminal, the terminal is put into SCALE mode waiting to accept SCALE commands, and the program stopped. This user mode hardware enables the monitor to step between the virtual and physical machines to some extent, and thus to isolate separate users from each other.

The monitor sees a job as being in one of three major states: runnable, halted, or waiting for I/O. From the monitor's standpoint, a "job" is a virtual machine which is allowed to make requests to the system for computing, input/output, and memory.

When a job is runnable, it is compute-bound and receives repeated quanta of runtime. When a job is halted, after being stopped by a programmed HLT or a ^VS (a SCALE command) from the terminal, it can make no further requests upon the system for any services until requests are initiated at the terminal through SCALE. Hence, it disappears from the system's run queues, and is no longer considered as one of the jobs competing for system services.

Similarly, when a job is waiting for I/O, it can make no further demands on system resources. Consequently, when a job requests an input/output transaction that cannot be executed immediately, it disappears from the system run queues and receives no more service until the transaction completes. After completion, the job re-enters the run queues and it will again begin receiving service until it requests I/O or halts.

Not all input/output requests cause the job to be stopped; only those requests for I/O that cannot be performed immediately do so. For instance, when the program outputs a character to the teleprinter, it is not stopped unless the monitor buffer for that teleprinter is already full. The size of the teleprinter buffer is initially set at 250 (base 10) characters, but it can be changed by the system manager with the CONFIG option BUFMAX. If the buffer is full when the program attempts to output a character, the job would be put to sleep until some minimum number of characters (defined by the system) is left in the buffer. The number of characters left in the buffer which causes the job to be restarted is initially set to 75 (base 10) characters, but it also can be changed by the system manager with the CONFIG option BUFMIN.

On the other hand, if the job requested input and the user typed ahead, the job would receive the input immediately and proceed in the normal fashion. The size of the keyboard buffer for characters typed ahead is initially set at 250 (base 10) characters, but can be changed by the CONFIG option BUFMAX. When you type ahead too many characters, ETOS sends an XOFF to the terminal. No further input from the terminal will be accepted until characters are emptied from the input buffer. XOFF is sent when the input buffer is within eight characters of being filled. The number of characters which triggers the XOFF can be changed by the system manager with the CONFIG option, XOFFCT.

If a user job must wait for input, the monitor "puts the job to sleep" (stops running the job) until input is completed. The job then "wakes up" (is rescheduled) and runs. The crucial point is that the job has no way of knowing if, or how long, it has been "sleeping". All input appears to be instantaneous.

1.3 CONVENTIONS USED IN THIS MANUAL

Certain documentation conventions are used throughout this manual to clarify examples of ETOS syntax. These conventions are

a. Capital letters and special characters such as commas and colons appear verbatim in the user's command. Keywords in capital letters may sometimes be abbreviated. For example

    ASSIGN dev: may be abbreviated as AS dev:

b. Square brackets indicate that the bracketed item is optional. For example

    AC [n]

    Square brackets are also used to enclose ETOS account numbers, and in this context they do not indicate that the item is optional. For example

    [proj,prog]
    or
    [account]
    or
    [56,77]

    However, an entire account specification may be enclosed in brackets to indicate that it <u>is</u> optional. For example

    [[proj,prog]]

c. Ellipses (successive periods) ... are used in conjunction with square brackets to indicate that the previous parameter may be repeated indefinitely, as when inputting a list of numbers. For example

    DEPOSIT n,m1[,m2[,...]]

d. Items in lower case type (channel, device, length, etc.) are supplied by the user according to rules explained in the text. For example

    DELETE channel

e. Words that the user enters are underlined. Words that the computer prints are not underlined. For example

    OPTION? <u>CREATE</u>

f. Special input characters are indicated by a code enclosed in angle brackets, as listed below

Code	Character	ASCII Code
<RET>	Carriage Return	215
<ALT>	Altmode	233, 375, or 376
<RO>	Rubout (Delete)	377
<LF>	Line Feed	212

g. Control characters are indicated by an uparrow followed by an alphabetic character. For example

Code	Character
^C	Control-C
^V	Control-V

h. User input is terminated by a carriage return unless otherwise indicated.

i. If a reference is made to a section in another manual, the manual name is included. If a reference is made to a section in this manual, the manual name is omitted.

CHAPTER 2
USING ETOS

2.1 INTRODUCTION

There are two general modes of operation in ETOS: SCALE mode and normal mode. Before the virtual computer is first "bootstrapped", it is in the ETOS monitor (SCALE). In SCALE mode the user may execute instructions analogous to the hardware operations that are performed on a single user PDP-8 computer (e.g., press ADDR LOAD, EXAMine, etc.). If OS/8 is bootstrapped, a complete OS/8 system is available to the user. If COS is bootstrapped, a complete COS system is available to the user.

After OS/8 or COS is bootstrapped, you may utilize SCALE commands by stopping the virtual machine, entering your commands and continuing standard processing. The program that you were running is continued at exactly the point at which it was stopped. See Chapter 3 for documentation on SCALE commands.

2.2 GETTING ON-LINE

Before the user attempts to use ETOS, he should be familiar with the operation of the terminal and the ETOS LOGIN procedure. Each user must have a limited knowledge of OS/8 in order to run ETOS system programs (INIT, TIME, etc.). The COS user must have a familiarity with standard COS.

2.2.1 Account Numbers

An account number is assigned to each user by the system manager. The account number consists of two portions: the project number (p) and the programmer number (pn). The account number is referred to symbolically as [p,pn]. The project number is a general identification. All personnel in one department, or all

students in one course, could have the same project number. the programmer number is the individual identification number. Only one person in the group would have this number. Both parts of the account are octal numbers between Ø and 77.

A typical account number might be

```
[12,77]
```

In this example, 12 is the project number and 77 is the programmer number. Associated with the account number is a password consisting of one to eight alphanumeric characters. This password does not echo during LOGIN, so that a degree of security may be maintained.

2.2.2 Logging into ETOS

Before using a terminal the user should determine whether or not it has been logged out by holding down the CTRL key and pressing the letter V (called a Control V and subsequently referred to as ^V). After releasing the CTRL key, the letter S is pressed and then RETURN (hereafter referred to as <RET>) is pressed. This sequence of ^VS<RET> should always cause a terminal to produce one of the two following messages

```
     ?LOGIN PLEASE                [Case A]
     !
or
     !                            [Case B]
```

In either case, an exclamation point is printed indicating that the virtual machine associated with this terminal has halted. It may be that the machine was already stopped, in which case, ^VS<RET> is harmlessly redundant. In case A above, the terminal is not yet logged into the system. In case B, the terminal is logged into an account and the previous user simply failed to log

out (sign off properly). Before proceeding in case B, enter LOGOUT<RET> which produces a LOGOUT message and ultimately a !. You are now at the same point as in the first case.

It is now time to log into the system. Once the terminal is on-line, enter

LOGIN

This command instructs ETOS that a user is ready to utilize the system. The command HELLO is synonymous with the LOGIN command. ETOS responds with the current time and date and requests input as follows.

```
ETOS V5B AT Ø9:55:ØØ A.M. ON TUE Ø3-APR-79
ACCOUNT?
```

Input your project number and programmer number, separated by a comma. ETOS responds with

```
PASSWORD?
```

Input your alphanumeric password. Remember that these characters are not printed on the terminal. If the account number or password is invalid, the system displays "ACCOUNT NOT FOUND". The user must then follow the entire LOGIN procedure again. If the input is valid, the user is logged into the system. ETOS displays the job number (a number assigned by the computer for this LOGIN) and console number of the current user. A message specified by the system manager is also displayed. This message provides current information about the system and may change periodically.

A sample LOGIN is contained in Figure 2-1. In all examples in the manual, each line is terminated by a carriage return unless

otherwise specified. Characters entered by the user are underlined to distinguish them from characters displayed by ETOS.

Figure 2-1
Sample LOGIN

```
^VS
?LOGIN PLEASE
!LOGIN
ETOS V5B AT 10:23:35 A.M. ON THU 15-MAR-79
ACCOUNT? 0,5
PASSWORD? TRS                          (this does not echo)
JOB 4 LOGGED IN ON CONSOLE 1

WELCOME TO QUODATA'S EXTENDED TIMESHARED OPERATING SYSTEM
    (this message may be changed)
```

There are several alternate ways of LOGging INto the system. The user may enter the account number on the same line as the LOGIN command, separated by a semicolon. If the user logs in this way, the header line will not be typed on the console. He may also enter the password on the same line, separated by a space. However, if he uses this construction, the password is displayed on the terminal. This method should not be used if security is important.

An example of an alternate LOGIN method is contained in Figure 2-2.

Figure 2-2
Alternate LOGIN Sample

```
^VS
?LOGIN PLEASE
!LOGIN;10,10 ACCT
JOB 5 LOGGED IN ON CONSOLE 0

WELCOME TO QUODATA'S EXTENDED TIME-SHARED OPERATING SYSTEM
   (this message may be changed)
```

Note that the heading information line is not printed when the account number is placed on the same line as LOGIN. After the user is logged into the system, ETOS automatically assigns the job either 8K, 12K, 16K or 20K words of memory and BOOTs up OS/8. The system manager determines how much memory is automatically assigned when he creates the account. If you need more memory, you can obtain it at any time with the CORE command.

If a user desires to have a special OS/8 system that is different than [0,2]OS8.RTS, the manager creates a special file "OS.SAV" for the user's account that contains the specialized OS/8 head for that user. OS.SAV can reside on any disk as long as it is under the user's account number. ETOS will look for this file to boot all of the mounted disks. If it is not found on any of the disks then [0,2]OS8.RTS will be booted up. If OS.SAV is found, the OS/8 scratch blocks and the OS/8 disk area are not LOOKed UP.

The user's private OS/8 disk "OS8DISK.DSK" can reside on any of the mounted public disks. LOGIN will look for it by checking drive 0, then drive 1, and so on. If OS8DISK.DSK is not found on any of the disks, the message "?CANNOT FIND OS8DISK.DSK" will be printed, and OS/8 will be BOOTed without a private OS/8 disk area. This message is only a warning and is proper for COS users.

If you require more core or wish to use the virtual front panel, enter

```
.^VS
```

This places your terminal in SCALE mode. If you need more memory, enter CORE X, where X is the amount of virtual memory you require. After you complete your SCALE commands, you may return to OS/8 by entering

```
!BOOT
```

You may now proceed to use OS/8. There are three devices immediately available to you: SYS, DSK, and TTY. SYS is a read-only storage device having the OS/8 system library. All public files and programs are contained here. DSK is the user's account storage device which has a size allocated by the system manager. TTY is the device name of the terminal itself.

All system maintenance functions are performed under OS/8 (initializing time and date, creating accounts, etc.). If you are LOGged IN under an account which can use COS, you can invoke COS by entering R COSBO. The user's virtual COS disk "COSDISK.RTS" can reside on any of the mounted public disks. COSBO will look for it by checking drive Ø, then drive 1, and so on. If COSDISK.RTS is not found on any of the disks, the message "CANNOT FIND COSDISK.RTS" will be printed, and OS/8 will be reBOOTed. This message indicates that you cannot run COS from the current account. If you require more core or wish to use the virtual front panel after invoking COS, enter

```
.^VS
```

This places you in SCALE mode. After you complete your SCALE commands, you may return to COS by entering

```
!BOOT
```

You may now proceed to use COS. Your COS account and all programs stored therein are immediately accessible.

2.2.3 Messages

The following is a list of all possible messages which may be printed during LOGIN.

```
?ALREADY LOGGED IN
```

This error indicates that you are attempting to LOGIN to an account, while you are LOGged IN to another account. You must LOGOUT (see 2.5) before you can switch accounts.

```
?CANNOT FIND OS8DISK.DSK
```

This is a warning message, which indicates that there is no OS/8 account storage area for this account. COS users should ignore this message.

```
?CANNOT FIND OS8.RTS
?%VM HLT
PC 1021 AC 0 FL 1100 MQ 500 SC 1 SR 0
!
```

If this error occurs, you must manually initiate operation of an operating system. OS/8 is not available to you.

```
?CANNOT FIND SCRATCH BLOCKS
?%VM HALT
PC 1021 AC 0 FL 1100 MQ 500 SC 1 SR 0
!
```

This is a fatal error which indicates that the file [0,3]JOBXX.SBK is not found (XX is the current job number). You must LOGOUT and wait until the system manager creates this file before you may LOGIN.

```
?ERROR BOOTING OS/8
%VM HALT
PC 1021 AC 0 FL 1100 MQ 500 SC 1 SR 0
!
```

If this error occurs, the system manager must shut ETOS down and transfer a fresh copy of the OS/8 system head to the file [0,2]OS8.RTS.

?JOBMAX EXCEEDED. TRY LATER

This message indicates that all sixteen job slots are being used. You must wait until another user LOGs OUT before you can LOGIN.

?NO SWAP TRACKS AVAILABLE

This message indicates that the disk file, which stores the user jobs when they are not running, is full. You must wait until swap tracks are made available by another user LOGging OUT or diminishing his virtual memory with the CORE command. If this error is printed frequently, request that the system manager increase the size of the swap tracks files.

n OTHER JOB(S) USING SAME ACCOUNT

This is a warning message, which indicates that other users are currently LOGged IN to the current account. If you wish to create files, you should LOGOUT and LOGIN to a different account.

2.3 CALLING PROGRAMS

After invoking OS/8 or COS, you will eventually want to execute a program. This section contains instructions for calling OS/8 and COS programs.

2.3.1 Calling OS/8 Programs

Most programs used under OS/8 can be called from SYS: by use of the R command (e.g., .R PIP, .R EDIT, .R FORT). Core image programs which are created on or copied to the user's account must be called with the RUN command. A sample of what may happen when you wish to run a core image program called OS8SAV on a disk area which has not been initialized is contained in Figure 2-3.

Figure 2-3
Sample Call of OS/8 Programs

```
1)  .RUN DSK OS8SAV
2)  MONITOR ERROR 2 AT 00524 (DIRECTORY I/O ERROR)
3)  .DIRECT
4)  BAD INPUT DIRECTORY
5)  .ZERO DSK:
6)  .RUN DSK OS8SAV
7)  OS8SAV.SV NOT FOUND
8)  .COPY DSK:<SYS:OS8SAV.SV

    FILES COPIED:
    OS8SAV.SV

9)  .DIRECT

    15-APR-79

    SYS [0003] OS8DISK.DSK
    <0050>     [434 BLOCKS]

10) OS8SAV.SV 73 01-APR-79
11) 354 FREE BLOCKS

12) .RUN DSK OS8SAV
```

You wish to run OS8SAV (1) but OS/8 returns an error message (2) telling you that the directory of DSK is bad. You confirm the bad directory (3 and 4) and initialize the directory (5). An attempt to RUN OS8SAV (6) fails again (7) because it cannot be found on the just-zeroed DSK. OS8SAV is copied (8) from the public SYS to the private DSK. It now appears in the directory of DSK (9) as using 73 blocks of 427 available (10 and 11). OS8SAV is now available for use (12).

2.3.2 Calling COS Programs

All programs executed under COS must be initiated with the RUN or R commands (e.g., .RUN PIP, .R COMP). You can only run programs which are in the COS account you are currently using. If you wish to run a program in another account or on another disk, you must copy the program to the current COS account or BOOT the COS

account containing the program. A sample of what may happen when you wish to compile a source program called COSSRC from another COS user's disk area is contained in Figure 2-4.

Figure 2-4
Sample Call of COS Programs

```
1)  FETCH COSSRC
2)  FILE NOT FOUND
3)  .^VS
4)  !CLOSE 3;LOOKUP 3=[10,25]COSDISK.RTS
5)  !!CONTINUE
6)  FETCH COSSRC,DK3
7)  .R COMP/N
8)  .
```

You wish to fetch COSSRC (1) but COS returns an error message (2) telling you that the source file is not found. You return to SCALE (3) and associate another COS disk area with ETOS channel 3 (4). You return to COS (5) and fetch the source file from DK3, the COS designation for channel 3 (6). You compile the file which is now in the source buffer (7) and the file compiles successfully, returning you to the COS monitor (8).

2.4 MODES OF OPERATION

ETOS requires more control over the virtual machine than OS/8 or COS allows. The equivalent of a front panel is necessary for a number of operations (e.g., starting OS/8, examining memory).

Through SCALE, each terminal can perform all the functions that a real front panel does, as well as additional operations. Two modes of operation have been defined. The first is normal mode, in which the terminal communicates with the program as it would on a real machine. The second is SCALE mode, in which keyboard input is routed to SCALE for interpretation as system commands. The convention is that the terminal is always in normal mode, communicating with whichever program happens to be running, except when (1) the virtual machine is halted, or (2) the user has typed ^V at the beginning of the line.

When a terminal is running in normal mode, typing ^V (CTRL/V) clears the user's keyboard buffer and the input line following is interpreted by SCALE as a command. The terminal returns to normal mode as soon as SCALE has executed the command, unless the command that follows ^V is an S (STOP), which stops the virtual machine and places the terminal in SCALE mode.

If the terminal is in SCALE mode, the command !R [0,2]OS8.RTS brings in a fresh copy of the OS/8 monitor; !R COSDISK.RTS will bring in a new COS monitor; !CONTINUE restarts the virtual machine where it left off or at a location specified by the user, and ^C restarts the machine at the current restart address (07600 for OS/8 and COS keyboard monitors). ^C may or may not return to the virtual machine depending on the program that it is running. See Chapter 3 (The Monitor and Scale Commands) for further explanation of these SCALE commands.

2.5 EXITING THE SYSTEM

After you have completed your work, you must leave the system such that the terminal is ready for another user. To exit the system, enter the dialogue contained in Figure 2-5.

Figure 2-5
Exiting The System

```
.^VS          (not required in OS/8)
!LOGOUT
JOB n LOGGED OUT ON CONSOLE m
12:19:28 P.M. ON WED Ø4-APR-79

ELAPSED TIME:
Ø2 MINUTE(S) Ø1.2 SECONDS
GOOD AFTERNOON

!
```

To eliminate the LOGOUT message, type LOGOUT;F.

If you are at the OS/8 keyboard monitor, enter the dialogue contained in Figure 2-6.

Figure 2-6
Exiting The System From OS/8

```
.LOGOUT
JOB n LOGGED OUT ON CONSOLE m
12:33:21 P.M. ON WED Ø4-APR-79

ELAPSED TIME:
Ø2.9 SECONDS
GOOD AFTERNOON
```

If you LOGOUT from OS/8, you may not eliminate the LOGOUT message.

The LOGOUT procedure closes all of your ETOS channels and DEASSIGNs all devices ASSIGNed to your job. It does not DISMOUNT any disks MOUNTed by your job.

CHAPTER 3
THE MONITOR AND SCALE COMMANDS

3.1 INTRODUCTION

SCALE is the user's method of communicating with the ETOS monitor. An example of the general form of each SCALE command is given immediately after the appropriate section heading. Rules for interpreting the general forms are listed in 1.3. See Appendix A for a reference list of all SCALE commands. See Appendix B for a reference list of all SCALE error messages.

SCALE commands may be entered with lower case alphabetic characters. This should simplify the use of terminals with no CAPS LOCK key.

SCALE commands may optionally be terminated by a semicolon. When this is done, SCALE will not read past the semicolon unless the VM is halted, thus leaving the remaining characters in the input buffer. This makes it possible to enter several SCALE commands followed by one user program command in one input line. The general form of this construction is

```
commanda[;commandb[;...]]
```

Examples of applications of the semicolon construction are contained in Figure 3-1.

Figure 3-1
SCALE Semicolon Construction

```
(1) .^VS;LOOKUP 3=DK1:OS8DISK.DSK;CONT
    !!
(2) COS MONITOR 3.Ø7G
    .^VS;R [Ø,2]OS8.RTS;DIRECT/Q
    !
    .
     Ø1-APR-79
     77 FREE BLOCKS
    .
```

3.2 HELLO, LOGIN, AND LOGOUT COMMANDS

H[ELLO]
LOGI[N]
LOGO[UT]

These commands are used at the start and at the end of a session at a terminal. HELLO and LOGIN are identical and are used to gain entrance into the system. See 2.2 for more information. HELLO and LOGIN initiate in a virtual machine (JOB number) being assigned to the terminal. Either 8K, 12K, 16K or 2ØK words of memory are allocated to the virtual machine and a BOOT command is executed so that the virtual machine is now running OS/8.

Note: The system manager determines how much memory is automatically assigned upon LOGIN when he creates the account. You can change this memory allocation at any time via the CORE command.

After you have LOGged IN, enter the following to initiate COS.

```
.R COSBO
```

When you have finished a session at a terminal, you must LOGOUT. See 2.5 for more information. A sample LOGOUT is contained in Figure 3-2.

Figure 3-2
Sample LOGOUT

```
.^VS                          (not needed in OS/8)
!LOGOUT
JOB n LOGGED OUT ON CONSOLE m
10:21:02 A.M. ON FRI 16-MAR-79

ELASPED TIME:
01 HOUR(S), 03 MINUTE(S) 22.1 SECONDS
GOOD MORNING
!
```

To eliminate the LOGOUT message when in SCALE, type LOGOUT;F. If you are at the OS/8 keyboard monitor, the sequence contained in Figure 3-3 can be used.

Figure 3-3
Sample LOGOUT From OS/8

```
.LOGOUT
JOB n LOGGED OUT ON CONSOLE m
10:21:02 A.M. ON FRI 16-MAR-79

ELAPSED TIME:
01 HOUR(S), 03 MINUTE(S) 22.1 SECONDS
GOOD MORNING
!
```

If you LOGOUT from OS/8, you may not eliminate the LOGOUT message by typing LOGOUT;F.

The LOGOUT procedure closes all ETOS channels and DEASSIGNs all devices ASSIGNed to your job. It does not DISMOUNT any disks MOUNTed by your job.

3.3 STOP AND CONTINUE COMMANDS

```
S[TOP]
CON[TINUE] [[if[df]]loc]
```

Entering ^VS<RET> produces the same result as a HLT instruction

in a program - the terminal goes into SCALE mode. The S command is used to exit infinite loops, to stop OS/8 or COS when a program is not listening to ^C, etc.; i.e., whenever the HALT switch or HLT key on the front panel would be used.

CONTINUE is similar to the CONT switch on the PDP8/E, /F, or /M or the RUN key on the PDP8/A. Entering CONT with no argument picks up execution where the program halted (this location is specified by your PC in the field specified by your flag word). The CONTINUE command takes an optional 1 to 5 digit octal argument. Entering CONT and 1 to 4 digits resumes execution at that location in field Ø. For example,

```
!CONT 2ØØ
```

causes execution to begin at location ØØ2ØØ. Entering CONT and a five digit argument recommences execution in the field specified by the first digit at the location specified by the remaining four. For example,

```
!CONT 21754
```

restarts execution at location 1754 in field 2. The data field retains its original value. Entering CONT and a six digit argument allows you to continue execution with split fields. The first digit is the instruction field, the second the data field, and the remaining four the address. For example,

```
!CONT 2Ø7756
```

starts the RIM loader in field 2 (assuming it was there, i.e., that you had deposited it there), with the Data Field=Ø, so that it would load into field Ø.

All of the above can, of course, be accomplished somewhat less

conveniently using PC, FL, then CONT with no argument. The command

```
!CONT 207756
```

is equivalent (assuming the LINK and GT flags can be cleared) to

```
!FLAGS 20;PC 7756;CONT
```

3.4 CORE COMMAND

CORE[E] [n]

The command CORE n is used to request that "n" K words of virtual memory be allocated to the user's virtual machine. "n" is Ø, 4, 8, 12, 16, 2Ø, 24, 28 or 32. When CORE n is executed, the job's current core is saved. Therefore, under COS it is unnecessary to reBOOT the VM in order to return to the program executing prior to the CORE command. It is necessary to reBOOT the VM if you wish to run OS/8, because the software core size is only set by a BOOT. If you continue the program without reBOOTing, some OS/8 software will not recognize the additional memory assigned (see OS/8 Software Support Manual, page 1-6). If you wish to run OS/8 or COS, you must assign a minimum of 8K words. After the CORE n command is used, SCALE returns the message FIELDS ASSIGNED Ø,1, etc. depending on n. If the CORE command is used without an argument, the current assignment is displayed. A sample execution is contained in Figure 3-4.

Figure 3-4
Sample Core Command

```
.^VS
!CORE
FIELDS ASSIGNED Ø,1
!CORE16
FIELDS ASSIGNED Ø,1,2,3
!BOOT          (or CONT if COS)

.MEMORY
16K MEMORY
```

3.5 BOOTSTRAP COMMAND

BO[OT] [chan]

The command BOOT performs a bootstrap of a virtual system device similar to the bootstrapping of a single user OS/8 or COS system. It may be necessary if a user program destroys the resident OS/8 or COS monitor in the virtual machine. A sample execution is contained in Figure 3-5.

Figure 3-5
Sample Boot Command

```
^VS
!BOOT
.       (OS/8 or COS is now bootstrapped depending on which
         system was previously running).
```

The BOOT command operates by reading block 0 of channel "chan" into virtual core at location 00000 and then jumping to location 00000. If "chan" is not specified, channel 0 is assumed. If a "%VM HLT" at PC=0 occurs, there was an error reading the bootstrap block. AC contains the CHANIO error (see 7.4.1).

3.5.1 OS/8 Bootstrap

The OS/8 bootstrap sets the software core size (location 07777) to the maximum field in the virtual machine, sets the text area to KBDMON.OS, jumps to location 07605 (which sets the OS/8 date words (locations 07777 and 17666) to the ETOS date, and enters the OS/8 monitor. If channel 1 is closed, the scratch blocks are looked up and write enabled. This can only be done if the user is privileged. If a BOOT is initiated (or OS8BO is executed) when channel 1 is closed and you are non-privileged, the BOOT process fails, displaying the message

```
?%VM HLT
```

```
 PC 3541 AC 1 FL 5100 MQ 501 SC 1 SR 0
!
```

When this %VM HLT message is printed, you should LOGOUT and LOGIN again to restore OS/8.

If channel 0 is closed when the BOOT is executed, the error message "SYSTEM ERROR" is printed. If this occurs, type the following to restore OS/8.

```
.^VS;R [0,2]OS8.RTS
```

3.5.2 COS Bootstrap

The COS bootstrap sets the COS date word (loc 07777) to the ETOS date, sets the text area to COS, and dynamically alters itself for a VT05 or VT52. The user can use the BOOT command to return to the COS-300 monitor if he has corrupted his virtual core. If a BOOT is executed when channel 0 is closed, the error message "?ILLEGAL IOT" is printed. If this occurs, type the following to restore COS.

```
.^VS;R COSDISK.RTS
```

3.6 R COMMAND

```
R [[p,pn]]filenam.ext
```

SCALE's R command is used to run a program contained in an ETOS file. It operates by looking up the ETOS file on ETOS channel 0, and then simulating the BOOT command (see 3.5). If [p,pn] is omitted, the default is the user's account. It is the responsibility of block 0 of the ETOS file to read in the rest of the program, which is assumed to be LOOKed UP on channel 0.

Under current conventions, any ETOS file with an extension of "RTS" or "SAV" is an ETOS program which can be run with SCALE's R command. Therefore, both the virtual OS/8 and COS operating systems can be regarded as ETOS programs. For example, the following command reads into the user's virtual memory a fresh copy of OS/8 from the disk and initiates the bootstrap procedure.

```
!R [0,2]OS8.RTS
.                          (now in OS/8 monitor)
```

The following command bootstraps the COS system device into the user's account.

```
!R COSDISK.RTS
COS MONITOR 3.07G
.                          (now in COS monitor)
```

3.7 VIRTUAL REGISTER COMMANDS

The ETOS front panel (SCALE) has virtual registers which correspond to their physical counterparts. All SCALE commands which set a register in the VM have the same form unless noted otherwise. If just the command keyword is entered, the current value of the register is displayed. If the keyword is followed by an octal number of one to four digits, the register is set to that new value. A sample virtual register command is contained in Figure 3-6.

Figure 3-6
Sample Virtual Register Command

```
!AC 200                    (sets the value of the AC to 200)

!AC                        (displays the current value of the AC)
200
```

3.7.1 AC, PC, and MQ Commands

AC [n]
PC [n]
MQ [n]

The AC command prints or sets the current value of the accumulator for the program.

The PC command prints or sets the value of the program counter, the address from which the next instruction to be executed will be fetched.

The MQ command prints or sets the value of the multiplier quotient.

3.7.2 FLAGS Command

F[LAGS] [n]

The FLAGS command prints or sets the value of the FLAGS word. This word is in the same format as the STATUS word on the front panel. Its format is contained in Table 3-1.

TABLE 3-1
FLAGS WORD FORMAT

Bit	Meaning
Ø	Link
1	GT flag
2-4	Unused
5	User mode
6-8	Virtual instruction field
9-11	Virtual data field

The link is a one-bit overflow of the accumulator, which functions identically like it does on a stand-alone machine.

The GT flag is an EAE bit, and is meaningful only on systems with the extended arithmetic element (EAE).

User Mode (UM) is internal to the system. This bit keeps your program in user mode when it is running. SCALE will not let you clear this bit; it performs a logical OR on any value you attempt to put in the FLAGS word with 100 (base 8).

The instruction and data fields are used identically as they are on a stand-alone machine; they are the fields that your program is currently using. The field numbers are virtual, and have no meaningful correlation with the physical fields that your program is currently using.

3.7.3 SC Command

SC [n]

The SC command prints or sets the value of your step counter if the extended arithmetic element (EAE) option is installed. If your computer has no EAE, this command serves no function. This word also indicates the current EAE mode. When bit 11 is on, your EAE is in mode B; when it is off, the EAE is in mode A. These modes are discussed in Chapter 5 of the Small Computer Handbook.

3.7.4 SR Command

SR [n]

The SR command prints or sets the current value of your switch register. Each VM has its own virtual switch register; no one has access to the physical front panel switch register. The virtual switch register may also be set via an assembler program instruction (see 6.7.1).

3.7.5 BREAK Command

B[REAK] [n]

In stand-alone OS/8 programs, each character typed at the terminal must be read in, echoed to the user, and interpreted immediately. Under ETOS, programs which execute in this fashion may cause a swap to another job each time that a key is struck. This results in a great deal of system overhead and slower response times. Most programs do not need to check each character immediately. They only need to take special action on a few special characters (RET, LF, RUBOUT, etc.). If the user sets the break mask to break only on specified characters, ETOS reads all other input into an internal buffer. ETOS also echoes these characters and handles some control characters, depending on the value of KSTAT (see 3.7.6). When a break character is entered, the program wakes up and reads the characters from this buffer, exactly as if it was reading directly from the terminal. Since ETOS is always resident, this method is very efficient.

BREAK prints or sets the current value of your Break Mask as indicated in Table 3-2. If a bit is set, the corresponding characters act as BREAK characters. The OS/8 and COS monitors set this word to 0607.

Table 3-2
Break Mask Bit Assignment

Bit	BREAK Characters
Ø	Any character
1	Letters
2	Numerals
3	Horizontal tab
4	LF, VT, FF, RET
5	! " # $ % & ' ( ) * + - / ; ,
6	SPACE
7	< = > ? @
8	[ ] ^ _ (On some terminals the underscore appears as a back arrow.)
9	RUBOUT
1Ø	ALTMODE (376,377)
11	2ØØ-237 (all other control characters including ESCAPE (233))

3.7.6 KSTAT Command

K[STAT] [n]

In stand-alone OS/8 programs, there exist routines for interpreting special characters (CTRL/C, CTRL/U, etc.). There are also routines for echoing input characters. ETOS will handle some characters and echo input depending on the value of a special register.

KSTAT prints or sets the current value of this special 12-bit register associated with your terminal as indicated in Table 3-3. The OS/8 and COS keyboard monitors set this register to 4374. The KSTAT word is organized so that each bit corresponds to an ETOS feature. Setting the bit enables the feature. Clearing the bit disables the feature.

With ETOS handling the defined control characters, user programs run more efficiently. Also the user may remove routines which currently handle these characters, thus freeing program space. Note that all of the control characters mentioned in Table 3-3,

except RUBOUT and CTRL/V, are left in the input buffer even if ETOS is enabled to handle them.

Table 3-3
KSTAT Register Bit Assignment

Bit	Meaning
Ø	Terminal input echoed by ETOS. User routines which echo characters should be removed or double echoing will occur.
1	XOFF/XON for terminal input. If the user's input buffer is nearly full, ETOS sends an XOFF (^S) to the user's terminal to disable further input. When the input buffer empties, ETOS sends an XON (^Q) to reenable keyboard input. This bit is set if the system has sent an XOFF (i.e., input is disabled), indicating that ETOS must transmit an XON when the input buffer becomes sufficiently empty. It is the responsibility of whatever is sending data to ETOS to respond properly to ETOS's XOFF/XON Control signals.
2	Job in SCALE mode. This bit is always set when the user executes the KSTAT command. Setting the bit or clearing the bit has no effect.
3	Ignore all terminal input. This bit is set only by the spooler when spooling to a terminal. Neither the KSTAT command nor the SETSTAT IOT will set this bit.
4	CTRL/C Trap. Double CTRL/C (^C^C) entered on the user's console traps to the CTRL/C RESTART address in field Ø.
5	CTRL/O. Entering CTRL/O (^O) on the terminal suppresses output. The computer continues to output, but it won't display on the terminal. Execution of a KRS (see 6.2.5) is necessary to turn output on again. This is standard OS/8 convention (see the OS/8 Handbook).
6	XON/XOFF control of user's keyboard. This bit enables the feature discussed in conjunction with bit 1.
7	XON/XOFF for terminal output. When the terminal is printing, XOFF (^S) entered through the keyboard suspends output. XON (^Q) reenables terminal output. Note that some terminals do not have this feature labeled as XOFF/XON. XOFF/XON differs from CTRL/O in that no output is lost. It main application is in scope output or asynchronous interprocessor communication.

8 CTRL/U. CTRL/U (^U) deletes the current input line, or as much of it that has not yet been read by the program. The ^U is also placed in the user's input buffer.

9 CTRL/V. Entering CTRL/V (^V) interrupts the current job and puts the user in SCALE mode for that line. This feature is automatically enabled by ETOS. Only a privileged user may disable this feature.

10 CTRL/P trap. Entering CTRL/P (^P) on the terminal traps to the CTRL/P RESTART address in field 0.

11 RUBOUT. Pressing RUBOUT (or DEL) on the terminal causes the previous character in the input buffer to be ignored. RUBOUT deletes CARRIAGE RETURNs and LINE FEEDs, allowing editing of previous input lines which are still in the input buffer. If there are no characters in the input buffer, ETOS deposits a RUBOUT in the buffer.

3.7.7 WHERE Command

W[HERE]

The WHERE command prints the values of the AC, PC, FLAGS, MQ, SC and SR. This command takes no arguments. A sample execution of this command is contained in Figure 3-7.

Figure 3-7
Sample WHERE Command

```
!WHERE
 PC 757 AC 0 FL 4101 MQ 77 SC 2000 SR 0
!
```

3.8 EXAMINE AND DEPOSIT COMMANDS

E[XAMINE] loc[n]
DEP[OSIT] loc v1[v2[...]]

To examine virtual memory, enter EX followed by the location whose contents you want to examine (1-5 digits). If the location is five digits, the first digit is the virtual instruction field.

To examine more than one location, also enter 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. A sample execution is contained in Figure 3-8.

Figure 3-8
Sample Examine Command

```
!EX 7756
 6032
!EX 7751 5
 6032 6031 5357 6036 7106
!
```

To deposit in virtual memory, enter DEP, the address at which to start depositing (1-5 digits), and then a string of octal values which will go into sequential locations. If the start location is five digits, the first digit is the virtual instruction field. A sample execution is contained in Figure 3-9.

Figure 3-9
Sample Deposit Command

```
!EX 4
 0000
!DEP 4 1005 3410 5004 5404 11 2010
!EX 4 6                          (these instructions zero
 1005 3410 5004 5404              memory)
 0011 2010
!
```

3.9 TALK COMMAND

T[ALK] console message

Issue this command when you wish to send a message to another terminal. The text message is sent to the terminal number specified in "console". The console number corresponds to the entry in the IOT table.

After the TALK command is executed, the receiving terminal displays

```
** KB m ** message
```

where "m" is the number of the sending console. This message does not affect the processing of the console, you're sending to, in any way.

For example, if you were LOGged IN on console 5 and you wanted to request that the user at the console terminal (console Ø) mount a DECtape, enter the following command.

```
!TALK Ø PLEASE MOUNT DECTAPE NUMBER 175
!
```

The following message appears at console Ø.

```
**KB5 ** PLEASE MOUNT DECTAPE NUMBER 175
```

The system manager may make the TALK command privileged. If this is done, the following error message "?NOT PRIVILEGED" is printed when you attempt to utilize TALK command.

3.10 ASSIGN AND DEASSIGN COMMANDS

AS[SIGN] dev[:]
DEA[SSIGN] dev[:]

The ASSIGN command allows the user to request the use of a private disk, a line printer, or a card reader if available on the system, by assigning it to his job as follows

!ASSIGN DKn[:] or L[PT:] or C[DR:]

In the above example, "n" ranges from 1 to 3, depending on the disk drive number. A disk needs to be ASSIGNed only if it is a non ETOS file-structured device. Standard OS/8 and COS disks are in this category and should be ASSIGNed and then accessed via the non file-structured LOOKUP command (see 7.2.4). A disk cannot be MOUNTed as a public pack if it is ASSIGNed. DK1 through DK3 may be assigned by any privileged job, even if the disk has previously been ASSIGNed by another user. DK1 through DK3 are automatically accessible by any job with the same project number as the first job to ASSIGN the disk. It is not recommended that two users access a COS or OS/8 format disk, unless only one user is creating programs or files.

For an RK05 installation, non-system packs are mounted in physical drives DK1, DK2 and DK3. For a System Industries installation, the removable pack in port 0 is DK1. The fixed disk in port 1 is DK2. The removable disk in port 1 is DK3.

A disk cannot be DEASSIGNed if any channels are open which access the disk. If this is attempted, the following error "?CHANNEL OPEN ON DEVICE" is printed.

Once a device is ASSIGNed, no one else on the system may use this device until you issue the command DEASSIGN dev: or log off. If

someone else has assigned the device, the message "DEVICE OWNED BY JOB n CONSOLE m" is printed. If this message is printed, you can use the TALK command (see 3.9) to request the owner of the device to DEASSIGN it. Once this DEASSIGN is complete, you may then ASSIGN the device to your job. All devices are automatically DEASSIGNED when you LOGOUT.

If the user attempts to utilize the line printer or card reader without ASSIGNing them, an "ILLEGAL IOT" message appears. One exception is the use of the line printer in the spooling mode (see 3.11 and 4.9). If you receive the "ILLEGAL IOT" message, you can ASSIGN the device, set the PC back one location and continue. If the PC printed out by the error IOT message is 3216, the following command would continue a line printer listing.

```
!ASSIGN LPT;PC 3215;CONT
```

If you attempt to ASSIGN a device other than the ones listed above, the message "?UNKNOWN DEVICE" is printed. If you attempt to DEASSIGN a device which is not ASSIGNed to your job, the message "?WASN'T ASSIGNED" is printed.

A sample ASSIGN and DEASSIGN is contained in Figure 3-10.

Figure 3-10
Sample ASSIGN and DEASSIGN

```
!ASSIGN CDR:
!DEASSIGN LPT:
!
```

3.11 QUEUE COMMAND

Q[UEUE] device:=filespc[,offset] or
Q[UEUE] device:<filespec[,offset]

The QUEUE command invokes the line printer spooler to perform output.

QUEUE prints the ETOS file on the specified device, starting with the block specified by "offset" and continuing until a CTRL/Z is found or an end-of-file occurs. Note: remember that this file must be contained in the ETOS directory, not an OS/8 or COS directory.

The output device is a logical device name which is assigned by the system manager when he initiates the spooler. This name is normally LP00:.

The characters in the file are assumed to be packed, three 8-bit characters per two 12-bit words. The format is the same as OS/8's ASCII files.

Each QUEUE command immediately queues the ETOS file for printing. Normally, the QUEUE command is initiated by the QLP handler. See 4.9 on how to use the line printer spooler under OS/8. The spooler cannot be directly accessed under COS.

If you attempt to QUEUE to a device which does not have a spooler initialized, the message "?BAD QUEUE DEVICE" is printed.

A sample QUEUE command is contained in Figure 3-11.

Figure 3-11
Sample QUEUE Command

```
!QUEUE LP00:<TEMP01.LPT,22
!
```

3.12 VERSION COMMAND

V[ERSION]

The VERSION command displays the current ETOS version number. It may be issued at any time before or after LOGIN. This version number changes each time that a software update is performed on the ETOS monitor. A sample execution is contained in Figure 3-12.

Figure 3-12
Sample Version Command

```
!VERSION
ETOS V5.15
!
```

3.13 DETACH AND ATTACH COMMANDS

DET[ACH]
AT[TACH]

The DETACH command disassociates the terminal from the current job, leaving the terminal logged out. It displays a confirmation message to remind the user which job he is leaving. DETACH does not modify virtual memory, so it is possible to DETACH a job while it is running. Terminal output is placed in a buffer. When the buffer is full, the job goes to sleep. A sample DETACH is contained in Figure 3-13.

Figure 3-13
Sample DETACH Command

```
^VDETACH
DETACHING FROM JOB  n
```

The ATTACH command is used to associate the console with a

specified job. If the console is already logged in, a DETACH is automatically done first. After the ATTACH is specified, supply the job number you wish to attach to, and that job's account password. If the password is incorrect, the message "ACCOUNT NOT FOUND" is printed and the terminal is left logged out. The memory of the job being ATTACHed to is not modified, so it is possible to DETACH and ATTACH back to a running job without disturbing it. If the DETACHed job has output any characters to the terminal, they are displayed within a few seconds of ATTACHing to the job. If you have forgotten the job number of the job you wish to ATTACH to, you can utilize the SYSTAT program to print all DETACHed jobs (see 4.5). A sample ATTACH is contained in Figure 3-14.

Figure 3-14
Sample ATTACH Command

```
^VATTACH
DETACHING FROM JOB    n     (only if already logged in)
JOB? m
PASSWORD? password          (does not echo, for security)
```

Note: No prompt occurs after you type the password. A <RET> will usually tell you the state.

The DETACH command assigns your job to a dummy console. If there are no dummy consoles available when you attempt to DETACH your job, the message "?CANNOT DETACH" is printed. You must wait until a dummy console becomes available.

3.14 SYSTAT COMMAND

SY[STAT]

The SYSTAT program provides a list of current system usage. Information may be obtained in a variety of areas, such as the number of users currently logged into the system, what these users are doing, and how long they have been using ETOS. For information on the details of this output, refer to 4.5.

To run SYSTAT, enter SYSTAT<RET> in response to SCALE's exclamation point.

3.15 ^C AND ^P COMMAND

^C
^P

If you enter a ^C while in SCALE mode, you begin execution of the contents of virtual memory at the ^C restart address (see 6.2.14). If you were operating under OS/8 before you entered SCALE mode, ^C will normally return you to the OS/8 monitor. If you were operating under COS before you entered SCALE mode, ^C will normally return you to the COS monitor. If bit 4 of your KSTAT word is set to Ø (see 3.7.6), the ^C command has no effect.

If you enter a ^P while in SCALE mode, you begin execution of the contents of virtual memory at the ^P restart address (see 6.2.14). Since each program sets a different ^P restart address, the results of entering ^P vary greatly according to which program was just executed. If bit 1Ø of your KSTAT word is set to Ø (see 3.7.6), the ^P command has no effect.

3.16 RESTRICTIONS ON THE USE OF THE COMMANDS

Certain SCALE commands are illegal unless the VM (virtual machine) is halted. They are dangerous to the user, to the system, or to both if they are executed while the VM is running. These are BOOT, CONTINUE, CORE, HELLO, LOGIN LOGOUT, R, and SYS. Attempting to execute these commands while the VM is running will be unsuccesful, and the message

?PLEASE TYPE <^VS><CR> FIRST

will be displayed on the terminal. The commands that are listed above are by no means the only ones which might be dangerous to a user program. For example, entering ^VPC n while the VM is running is very likely to produce unwanted results.

Certain commands may be executed when the terminal is not logged in. They are ATTACH, HELLO, LOGIN, SYS, and VERSION. Attempting to execute other commands when the terminal is not logged in will be unsuccessful, and the message

?LOGIN PLEASE

will be displayed on the terminal. A summary of commands and when they may be executed is provided in Appendix A.

In addition to the SCALE commands presented in Chapter 3, there are an additional group of SCALE commands called channel input/output commands. Since these commands are complex and unnecessary for normal operation, they are not discussed until Chapter 7.

3.17 SCALE ERROR MESSAGES

A complete list of SCALE error messages is contained in Appendix B. The section on each SCALE command contains the errors pertaining to that command. In addition to these specified error messages, the following is a list of general error messages which may occur after a number of SCALE commands.

?BAD NUMBER

Too many digits were entered, no number was entered where one was required, or non-octal digits were entered in an argument which must be octal.

?FILE SPECIFICATION ERROR

An illegal syntax for an ETOS file was specified. An invalid ETOS device, account number, file name, file extension or protection code causes this error.

?UNKNOWN COMMAND

A command is entered which is not one of the valid SCALE commands listed in Appendix A.

?WHAT? WHERE?

A command is utilized in a manner which is invalid.

In addition to the error messages printed in response to SCALE commands, there are several error messages which may be printed by SCALE while a user program is running. These messages stop the virtual machine and return it to SCALE.

?ILLEGAL IOT

An input/output instruction, which ETOS does not support, was executed by the user program. This message will be printed if a non-privileged user attempts to run a privileged program or if you attempt to access a device which was not ASSIGNed.

The PC contains the (address +1) of the instruction which generated the error. The FLAGs word indicates the field of the address in bits 6-8.

?JMP/JMS TO NON EXISTENT FIELD

The user program executed a CIF to a non-existent field and attempted to execute a JMP or JMS. The PC contains the (address +1) of the JMP/JMS and bits 6-8 of the FLAG word contains the Instruction Field which the program tried to jump to. There is no indication which field the JMP/JMS came from. If you wish to locate the JMP/JMS error, all existing virtual fields must be examined at the location contained in the PC for a JMP or JMS.

?% VM HLT

The user program executed a HLT (7402) instruction. The PC contains the (address +1) of the instruction which contained the HLT. The FLAGs word indicates the field of the address in bits 6-8.

There are several errors which indicate a system malfunction. This malfunction may be due to hardware or software. If these errors occur frequently, there is a problem with the hardware, an ETOS system disk pack or the file structure. Diagnostics can be run to determine if there is a hardware problem. A copy of the system disk may be used to determine if the problem is caused by a disk pack. The file structure may be recreated by the system

manager. A list of serious errors follows.

?READ ERROR

A READ error occurred while attempting to read information from an ETOS disk.

?SYSTEM SWAP ERROR

A READ or WRITE error occurred while swapping a user into or out of memory.

?SYSTEM R/W ERROR

A READ or WRITE error occurred while attempting to read or write information from the ETOS system pack.

?WAIT FOR I/O

Disk input/output has been initiated but not completed.

3.18 EFFECTS OF NO FREE BUFFERS ON SCALE

Under extreme circumstances, the monitor's character buffering space may be entirely used. In this case, attempts at output by SCALE will not succeed or only partially succeed. For instance, if the user typed ^VW, the entire output from the W command (see section 3.7.7) might not be printed if at the moment there was not enough space for it in the output buffers. The remainder is discarded. This action can be annoying in the case of a program which displays a textual message and then halts. If it just fills the output buffers, the message "?%VM HLT" is not printed, although the HLT will, of course, have its usual effect. In this case, the terminal might appear to be hung. Pressing <RET> determines the state of the terminal.

3.19 TTY READER CONTROL

If the XON/XOFF feature on the terminal is not enabled, reading paper tapes when the system is heavily loaded may result in the reader getting so far ahead of the program executing the read, that the input buffer overflows. It may be necessary to stop the reader about every 150 characters for a moment so that no characters are lost. This should occur only under severe loading.

If the user overflows the input buffer and his program does not read the buffer, no further characters may be entered. In this case, the only character that may be entered is ^V. Entering ^V completely clears the input bufffer, and puts the TTY in SCALE mode for the rest of the current line (if the VM is running) or leaves its mode unaffected if it was already in SCALE mode. To escape a full input buffer, enter ^V<RET>.

CHAPTER 4
DIFFERENCES IN OS/8

4.1 INTRODUCTION

The differences between stand-alone OS/8 and ETOS OS/8 may be separated into three areas. There are changes made to the operating system itself (OS/8 monitor and command decoder), changes made to the standard OS/8 system programs (e.g., DIRECT, PIP) and there are additional programs under ETOS OS/8 which were written expressly for ETOS (e.g., TIME, SYSTAT).

During LOGIN, ETOS allocates between 8K and 20K words of virtual memory to you and starts (bootstraps) OS/8. The date word is set by the bootstrap to the ETOS date. After the bootstrap is completed, you have automatic access to an OS/8 system area called SYS. This area contains programs which must be run by all users (e.g., PIP, DIRECT). You cannot write or delete files in this area. In addition to SYS, you may have access to a private storage area which is separate from all other users. This area is accessed as DSK or CHN2. If no device is specified, this area is assumed. In addition to SYS and DSK, you may have access to non-disk peripherals (i.e., TTY, CDR, HSP, HSR, LT, LPT, QLP, RXA0, RXA1, DTA0, DTA1). You do not have automatic access to any other device on the system. If you wish to access other accounts or non-system disks, you must use the data channels explained in Chapter 7.

If you destroy the copy of OS/8 which resides in virtual memory, you can bring in a new copy from the disk by entering .^VS;R[0,2]OS8.RTS.

4.2 OS/8 MONITOR CHANGES

ETOS OS/8 appears to the user almost the same as stand-alone OS/8. However, there are some differences: a CTRL/U entered immediately after a RUBOUT will not enclose the deleted characters in slashes. Entering ^VS stops any program from running and returns the user to SCALE mode. If an OS/8 program is not in input wait, a double ^C is needed to return control to OS/8. This allows ^C to be used in type-ahead commands. ^V^C may be needed to return to OS/8.

When an OS/8 program is running, the user may type ahead commands which are to be executed when the current operation is finished. These commands will echo, providing ETOS is enabled to echo characters. This allows you to set up a batch stream from the terminal. If you type ahead too many characters, no more input is accepted or echoed until some characters are read from the input buffer. Commands which are typed ahead are lost if the input buffer is cleared. Some OS/8 programs clear the input buffer in case of an error. Entering ^V also causes the input buffer to be cleared. A sample use of type-ahead is contained in Figure 4-1.

Figure 4-1
Sample Use of Type Ahead

```
.^VASSIGN LPT:
R BITMAP
*LPT:<BINFIL.BN
```

While the BITMAP is being produced on the line printer, the following commands are entered on the terminal.

```
^C                  exit BITMAP
LOAD FILE1          load the resulting binary
SAVE DSK FILE1      save core image
DIR/Q               OS/8 directory of DSK
```

When the BITMAP is finished, the following is displayed on the terminal.

```
.                   (CTRL/C)


.

.                   (LOAD)
.                   (SAVE)

1-APR-1979          (DIRECT)

BINFIL.BN 10  1-APR 79
FILE1 .PA 52 15-MAR-79
FILE1 .BN  7 15-MAR-79
FILE1 .SV  7 15-MAR-79

   04 FILES LISTED, USING 76 BLOCKS
  672 FREE BLOCKS

.
```

The keyboard monitor and command decoder enable ETOS to perform character echoing. The keyboard monitor sets the BREAK Mask to 207 and the KSTAT to 4374. The command decoder sets the BREAK Mask to 207 and the KSTAT to 4374 upon entry, but restores the original status values when it is exited.

All device handlers have been modified to operate under ETOS. TTY is accessible to all users. CDR and LPT are accessible unless ASSIGNed by another user. RXA0 and RXA1, DTA0 and DTA1, HSP and HSR are accessible if you are privileged. In addition to the standard OS/8 devices, you may have access to an LT: handler and a QLP: handler. The LT: handler simulates a line printer on a terminal by using separated line feeds to handle pagenation. The QLP: handler is used to access the line printer when the system manager has initiated spooling. ETOS also has a modified version of the terminal handler (TTY) which allows you to specify the margin following an equal (=) sign. All devices which are not supported under ETOS do not have active handlers under virtual OS/8. In addition, the RK05 or System Industries system and non-system handlers are not active under ETOS. All disk access is controlled via data channels (see Chapter 7).

4.3 CHANGES TO OS/8 SYSTEM PROGRAMS

In addition to the changes in the OS/8 monitor, there are also changes in many of the OS/8 system programs.

Some of the changes were required to allow the programs to run under ETOS. Some of the changes were made to make programs more efficient. The general method for converting an assembler program to run under ETOS is presented in section 4.4.

Some of the standard OS/8 programs are not appropriate for use under ETOS. These programs access hardware in a mode not supported under ETOS. The OS/8 programs which are not supported under ETOS are listed in Table 4-1.

Table 4-1
OS/8 Programs Not Supported Under ETOS

Program Name	Stand-Alone Function
BOOT	Bootstraps OS/8 monitor on any supported system device.
CAMP	Transfers programs to and from an industry standard magnetic tape.
DTCOPY	Performs image copies of TCØ8 DECtapes.
DTFRMT	Formats TCØ8 DECtapes.
MCPIP	Transferrs programs to and from cassestes.
PIPlØ	Transfers programs between the PDP8 and PDPlØ DECtapes.
RK8FMT	Formats RKØ1 disk packs.
RKLFMT	Formats RKØ5 disk packs.
RXCOPY	Performs image copies of RXØ1 floppy diskettes.
TDCOPY	Performs image copies of TD8E DECtapes.
TDFRMT	Formats TD8E DECtapes.

A list of the OS/8 system programs, which are supported under ETOS, and a list of changes follows. Since these programs have been modified to run under ETOS, the stand-alone versions of the programs should never replace these modified programs on the ETOS disk.

ABSLDR

ETOS is enabled to perform CTRL/C checking.

BASIC

EDUCOMP BASIC (EB BASIC) creates a seventeen block ETOS file called "BASIC.TMP" in the user's account, when it is executed. If BASIC.TMP already exists, it is deleted and recreated. This file is used for temporary storage and is deleted when BASIC is exited normally. EB BASIC can be restarted with the START command if it has been exited from the BASIC compiler or the run-time system, but the only function you can perform is to save the text. ETOS is enabled to perform echoing for the editor and run-time input statement.

OBASIC

OS/8 BASIC is called OBASIC to distinguish it from EB BASIC (BASIC). The temporary file created by OS/8 BASIC is called OBASIC.TM and is created on DSK instead of SYS. The real-time functions of OS/8 BASIC are not currently supported. ETOS is enabled to perform echoing for the editor and run-time INPUT statement.

CBASIC

is called CBASIC to distinguish it from EB BASIC (BASIC). The temporary file created by Commercial BASIC is called CBASIC.TM and is created on DSK instead of SYS. ETOS is enabled to perform echoing for the editor and run-time INPUT statement.

BATCH

The /C switch and the /P switch in BATCH are inoperable. Stand-alone OS/8 BATCH attempts to output its log of operations performed to the line printer. If the printer does not respond, the log prints on TTY. ETOS BATCH makes the opposite assumption. The log prints on TTY unless you send it to LPT by using the /L option. The job must have the line printer assigned to function properly. The BATCH monitor is output to the scratch blocks 23-30 ([0,3]JOBXX.SBK), instead of the file SYS:BATCH.SV. ETOS BATCH allows SCALE commands to be specified, prefixed by a "!". SCALE commands, which cannot be executed while the virtual machine is running, must be preceded by "!S;" and terminated by ";CONT". A sample batch is contained in Figure 4-2.

Figure 4-2
Sample OS/8 Batch

```
$JOB ILLUSTRATING SCALE COMMANDS
!ASSIGN LPT:
!S;LOOKUP 3=[10,25]OS8DISK.DSK;CONT
.DIRECT CHN3:-L
!
!
!DEASSIGN LPT:
$END
```

Notice that some batches require filler lines. In the case of the above batch, you must wait until the DIRECT command is finished, before the DEASSIGN command is activated.

BITMAP

ETOS is enabled to perform CTRL/C checking.

BUILD

This program performs its own echoing, so that BUILD sets the BREAK mask to 4000 and the KSTAT to 1004. The BUILD command has been disabled. Device handlers modifed for ETOS have been LOADed into ETOS BUILD. The version of BUILD on the stand-alone OS/8

portion of the ETOS pack has a system device handler of device type 40. The type of the device handler has been changed so that PIP will set the size of the directory to 1200 (base 8) or 640 (base 10) blocks (option /S or /Z).

CCL

The following commands are not supported: BACKSPACE, BOOT, EOF, REWIND, SKIP and UNLOAD. CCL and SET can only be run by the system manager, since channel 0 must be write enabled. QBOL is used to run the QBOL program. TIME runs a program called TIME.SV (see 4.6). SYSTAT runs the SYSTAT program to print a summary of the system's status (see 4.5).

COBOL

The compiler and fairly short user programs will run in 8K. Longer user programs will need ^VS;CORE 12;BOOT before they are loaded or the system manager must allocate a minimum of 12K to the account. The device library (COBOL9.RL) has been modifed, so that stand-alone instructions which clear flags are disabled.

CREF

ETOS is enabled to perform CTRL/C checking.

DIRECT

ETOS is enabled to perform CTRL/C checking. The ETOS version of DIRECT displays the ETOS filename and characteristics for the channel which is being accessed. In addition, this version of DIRECT has options which are not contained in the standard OS/8 version of DIRECT. A list of all options is contained in Table 4-2.

Table 4-2
DIRECT Options

Switch	Meaning
/A	Alphabetize directory before processing.
/B	Print starting block numbers (octal).
/C	List only files with current date
/D	List files by date, the oldest files first
/E	Include empties in the listing
/F	Fast mode (no lengths or dates)
/H	List contents of the parameter block (header)
/I	Print additional information words (octal)
/M	List only the empties
/N	Print dates in numeric form (MM/DD/YY)
/O	List only files with other than today's date.
/P	Paginate -- add form feeds to output
/Q	Quell ETOS file information. Do not print ETOS header.
/R	List remainder of files after the first one
/T	List the files backwards by date.
/U	Treat each input specification separately
/V	List files not of the form specified
/W	Print the version number (disables other options)
/X	Alphabetize by extension, then by name.
=XYZ	Use Z columns for output. XYØ is passed to output device. DIRECT also prints the total number of files listed and the total number of blocks used by these files.

EDIT

ETOS is enabled to echo user input.

EPIC

Files compared with EPIC do not have to reside on the system disk. They must reside on SYS: or on a device co-resident with SYS. Under ETOS, SYS, DSK, and all channels (CHNX:) are legal devices. The example in Figure 4-3 compares two files under ETOS.

Figure 4-3
EPIC Example

```
.R EPIC
*DSK:OLDFIL.EX<CHN3:NEWFIL.EX/2$  ($=escape or alt mode)
```

The high speed reader/punch options are not supported.

FOCAL

ETOS is enabled to perform echoing of character input and CTRL/C checking.

FORTRAN II

Teletype input routines for FORTRAN II support special control characters (CTRL/U, RUBOUT, etc.). Temporary files created in compiling a FORTRAN program are created on DSK instead of SYS. The internal device handlers for high speed reader/punch and card reader are not supported. Device independent I/O must be used to access these devices.

FORTRAN IV

Temporary files created in compiling a FORTRAN IV program are created on DSK instead of SYS. Real-time functions and floating point processor functions are not currently supported. The internal device handlers for high speed reader/punch and card reader are not supported. Device independent I/O must be used to access these devices.

FOTP

ETOS is enabled to perform CTRL/C checking.

FUTIL

The program performs its own echoing, so that FUTIL sets the BREAK mask to 4000 and the KSTAT to 1004. Stand-alone FUTIL has no convenient facilities for accessing OS/8 files as pure binary data relative to the beginning of the file. In particular, it is almost impossible to do this in an OS/8 BATCH stream, because it is necessary to enter the starting block number of the file. The ETOS version of FUTIL has a variable "A" which is set to the starting block number of the file selected by the file statement. A sample use of the "A" variable is contained in Figure 4-4.

Figure 4-4
Sample Use of FUTIL "A" Variable

```
.R FUTIL

FILE ETOS.SV
ETOS.SV 0600-0762   0163 (0115)  1.247
EVAL A
= 000000600 (00000384)
```

Note that the value of "A" is equal to the first number in the data line output by the FILE command. This feature allows accessing of data relative to the beginning of the file, even in an OS/8 BATCH stream. Continuing the above example, block 1, location 3 of ETOS.SV can be accessed as follows.

```
(A + 1).3/ nnnn
```

Specifying a nonexistant file in the FILE command does not affect the "A" variable.

The change to add the "A" variable has been made to the ETOS version of FUTIL under virtual OS/8 and the stand-alone version of FUTIL under stand-alone OS/8.

HELP

The HELP command for DIRECT has been modified to display the additional switches added under ETOS.

KSORT

ETOS is enabled to perform echoing.

PAL8

PAL8 creates the temporary file CREFLS.TM on DSK instead of SYS. ETOS is enabled to perform CTRL/C checking.

PIP

PIP performs its own echoing for the answer to the question "ARE YOU SURE?", so that it sets the BREAK mask to 3777 and the KSTAT to 1364. On initialization, PIP performs an INQUIRE on all ETOS channels. The squish option and zero option therefore sets the size of the ETOS OS/8 directories to the proper length. Since PIP performs the INQUIRE only on initialization, you should not enter SCALE in the middle of PIP, LOOKUP a channel, and continue PIP without restarting. PIP assumes that the length of non file-structured devices is 6260 (base 8) or 3248 (base 10). The version of PIP on the stand-alone OS/8 portion of the ETOS pack set the size of device number 40 to 1200 (base 8) blocks. The system device handler in BUILD has been modified so that its device number is 40. The squish option and zero option therefore set the size of the stand-alone OS/8 directory to the proper length.

QBOL

QBOL uses the SKETOS (6107) IOT to determine if it's running under stand-alone OS/8 or ETOS. It modifies itself so that the same version of QBOL runs under single-user or time sharing. Under ETOS, QBOL performs its own echoing for the ACCEPT statement. Therefore it sets the BREAK mask to 4000 and the KSTAT to 4374.

RESORC

ETOS is enabled to perform CTRL/C checking. RESORC reconstructs device names from the OS/8 system head. It reconstructs some device names improperly. The version of RESORC under virtual OS/8 is modified to print ETOS device handler names properly. The version of RESORC under stand-alone OS/8 is modified to print OS/8 device handler names properly.

SET

SET performs its own echoing so that it sets the BREAK mask to

4000 and the KSTAT to 1004. The INIT.CM command file is taken from the OS/8 disk area (channel 2) if you set SYS INIT. All SET commands are privileged.

SQUASH

SQUASH is a program which compresses OS/8 directories to eliminate empties. It is preferable to the Squish option in PIP since it writes the directory after each file move. If the SQUASH is interrupted, the directory is valid. To execute this program, type R SQUASH DEV, where DEV is the device that you wish to compress. SQUASH does not rewrite the directory length. If you have EXTENDed or REDUCEd an OS/8 account area, you must use PIP, option S to rewrite the OS/8 directory length.

SRCCOM

ETOS is enabled to perform CTRL/C checking.

TECO

ETOS is enabled to echo character input.

3040

The System Industries 30/40 controller diagnostic under stand-alone OS/8 is modified to accept input from the console terminal instead of the switch register. Users with limited function panels are therefore able to utilize this diagnostic effectively.

6SORT

6SORT uses the SKETOS IOT (6107) to determine if it's running under stand-alone OS/8 or ETOS. It modifies itself so that the same version of 6SORT runs under single-user or time sharing. Under ETOS, 6SORT performs it's own echoing for any questions that it asks. Therefore, it sets the BREAK Mask to 4000 and the KSTAT to 4374.

4.4 CONVERTING STAND-ALONE OS/8 PROGRAMS TO ETOS

Source programs written in higher level languages (e.g., BASIC, FORTRAN) do not have to be modified to run under ETOS. However, some compiled files have to be recompiled from the source. OS/8 BASIC and Commercial BASIC compiled programs and core image programs (created with /K) have to be recompiled if the program accepts input from the terminal. If the programs are not recompiled, each input line is echoed twice. All FORTRAN II compiled programs and core image programs (created with SAVE command) have to be recompiled. The stand-alone version of FORTRAN inserts instructions into the compiled file which clear flags on devices which ETOS does not support. If you attempt to run a stand-alone FORTRAN II compiled program under ETOS, "ILLEGAL IOT" error messages are generated.

Even though no source programs have to be modified, there are some changes which might be desirable. Device names for disk drives are different under ETOS. Under stand-alone mode, each RKØ5 drive is accessed as RKAX and RKBX, where "X" represents the drive number. Each System Industries disk drive is referred to as DSKØ through DSK7, where DSKØ through DSK3 represents the fixed disk and DSK4 through DSK7 represents the removable disk. Under ETOS, all disk access is controlled with the channel handlers CHN2 through CHN7 (see Chapter 7). To run a program which uses a stand-alone device name, you can either modify the device names for ETOS or you can use the OS/8 ASSIGN command to equate stand-alone names to ETOS names. For example, assume that you have a stand-alone program which accesses SYS as RKAØ and a private area as RKBØ. Further assume that under ETOS, all accesses to RKAØ can be mapped to SYS and all accesses to RKBØ can be mapped to your private storage area CHN2 or DSK. You could edit the program and replace RKAØ with SYS and RKBØ with DSK. The alternative method is to use ASSIGNs to equate the devices. A sample run of this method is contained in Figure 4-5.

Figure 4-5
Sample Use of OS/8 ASSIGN

```
.ASSIGN SYS RKAØ
.ASSIGN DSK RKBØ
.RUN DSK PROG
```

If you have a program which accesses SYS, you must insure that no writing to SYS is performed. Under ETOS, SYS is an area accessible to all users and is therefore write-protected. Programs which write to SYS must be modified to write to another device such as DSK.

Unlike programs written in a higher level language, assembler programs should be modified for ETOS.

Assembler programs written for stand-alone OS/8 run more efficiently if they are modified to run under ETOS. There are several changes that should be made to the terminal service.

1. ETOS should be enabled to handle CTRL/C, CTRL/O, CTRL/P, CTRL/U and RUBOUT (see 6.2.14).

2. ETOS should be enabled to echo characters (see 6.2.14).

3. The program should allow ETOS to read in the characters and should break only on special characters (see 6.2.14).

4. The ^C^C restart address should be set so that ETOS can handle ^C checking (see 6.2.14).

Programs which have unmodified terminal service can be run without modification. ETOS can be enabled so that the program breaks on every character, handles its own control characters, and echoes all input characters. For example, if PROG is an unmodified stand-alone program, it can be properly run using the following procedure. After calling PROG, type ^VS to enter SCALE mode. To disable special ETOS character handling, issue the commands

```
!KSTAT 0304;BR 4000;CONT
```

and proceed in the usual manner.

In addition to the terminal service, there are several other changes which may be necessary to convert a stand-alone program to ETOS. Any temporary files which are written on SYS have to be written to DSK, since SYS is write-protected. Code which attempts to access a device not supported under ETOS, must be disabled or "ILLEGAL IOT" errors occur. Interrupt driven code must be disabled, since ETOS controls all interrupt processing. Any program which needs to know the length of an ETOS disk must be modified to INQUIRE (see 7.4.14) as to the length of a virtual device.

When you modify programs, you can modify them so that one version of the program runs under single-user OS/8 and ETOS. The IOT SKETOS skips the next instruction if you are running under ETOS. This IOT acts as a NOP if you are running under single-user OS/8. Therefore a program can determine if it is running under single-user or time sharing mode and modify itself accordingly.

4.5 SYSTAT PROGRAM

The SYSTAT program provides a list of current system usage. Information may be obtained in a variety of areas, such as the number of users currently logged into the system, what these users are doing, and how long they have been using ETOS.

To run SYSTAT, enter

```
.^VS
!SYSTAT
```

A sample output of SYSTAT is shown in Figure 4-6.

Figure 4-6
Sample Run of SYSTAT

```
SYSTAT VERSION 2.Ø24

ETOS V5B FOR QUODATA CORPORATION ON APRIL Ø1, 1979 AT 11:47:39 A.M.
UPTIME : Ø2:57:18

JOB  ACCT      WHERE  PROGRAM     SIZE  STATE     RUN-TIME      CONNECT-TIME
Ø1   [**,**]          SCALE       Ø8K   KEY NR    ØØ:ØØ:Ø3.7    Ø2:57:18
Ø2   [**,**]          DMON        Ø8K   KEY NR    ØØ:ØØ:Ø4.5    Ø2:57:18
Ø3   [ØØ,Ø3]  ØØ*     DIRECT.SV   Ø8K   LPT       ØØ:ØØ:Ø2.1    ØØ:Ø8:11
Ø4   [ØØ,Ø5]  Ø7*     KBDMON.OS   Ø8K   KEY NR    ØØ:ØØ:ØØ.2    ØØ:Ø1:Ø3
Ø5   [ØØ,Ø6]  Ø6      TECO.SV     16K   FIO       ØØ:1Ø:39.9    Ø2:5Ø:29
Ø6   [1Ø,1Ø]  Ø2      ODT.        12K   KEY NR    ØØ:Ø5:52.9    Ø1:15:Ø6
Ø7   [ØØ,Ø5]  Ø1*     DIRECT.SV   32K   TTY NR    ØØ:ØØ:1Ø.6    ØØ:15:16
1Ø   [ØØ,Ø4]  Ø5      SYSTAT.SV   Ø8K   RUN       ØØ:ØØ:Ø1.Ø    ØØ:ØØ:57

DEVICE  JOB
DSKØ    Ø1
LPT     Ø3
```

The header record consists of the following items.

1. The version number of ETOS. This version number is obtained from the monitor and changes if ETOS changes.

2. The installation name. The installation name is obtained from the ETOS file [Ø,2]INSTALL.TXT. If the user wishes to change the name, he may create a new installation file with EDIT or TECO and move the file to [Ø,2] with the OSETOS program. If no file [Ø,2]INSTALL.TXT exists, the installation name is taken as "QUODATA CORPORATION".

3. The current date and current time.

4. The uptime. The uptime is the time that ETOS has been running since system installation.

After the header, a list of system usage by job number is printd. The first two jobs are the system tasks, SCALE and DMON. The list consists of:

1. JOB number. The job number is an identification number assigned sequentially, when the user logs into the system.

2. ACCounT number. The account number consists of a project number and a programmer number, separated by a comma.

3. WHERE. This console number is associated with the respective terminal on which the user logged in. This number is followed by an asterisk (*) if the job is detached.

4. PROGRAM. This is the program that the user is currently running. If the user is at the OS/8 keyboard monitor (.), "KBDMON.OS" is displayed. If the user is running COS, "COS" is printed. If the user is running ODT, "ODT." is displayed. If the user is currently in SCALE, the last program executed is displayed.

5. SIZE. The size represents the amount of virtual core currently allocated to the user.

6. STATE. The state of a job corresponds to its current activity on the system. The possible states are contained in Table 4-3.

Table 4-3
SYSTAT States

STATE	Meaning
CDR	Card reader I/O wait
DMN	DMON wait
FIO	File I/O wait
KEY	Keyboard wait
LPT	Line printer I/O wait
NR	Job is currently non-resident
SCA	Scale mode
SMI	Sleeping in increments of minutes
STE	Sleeping in increments of tenths of a second
TTY	Teletype output wait

7. RUN-TIME. This is the actual amount of computing time that the user has received since LOGIN.

8. CONNECT-TIME. This is the amount of time that the user has been logged into ETOS.

The "SYSTAT" command can be executed while the user is logged out. Under OS/8, there are two alternate ways of calling SYSTAT. If your enter R SYSTAT, the program responds with an asterisk. Pressing carriage return starts execution of the program. A system status can also be initiated via the CCL command "SYSTAT". Regardless of how SYSTAT is started, the available options are contained in Table 4-4.

Table 4-4
SYSTAT Options

Option	Meaning
B	Display busy devices only
D	Disply detached jobs only
J	Do not display busy devices
L	Display system status to line printer
N	Do not display the header portion of SYSTAT
S	Display system jobs only (SCALE and DMON)
U	Display user jobs only. Skip system jobs
=n	Display job "n" only

Under OS/8, the options are entered, separated by slashes. Under SCALE, the options are entered after a semicolon. Therefore, the three alternate ways of obtaining a system status of job Ø5 to the line printer, with no header are contained in Figure 4-7.

Figure 4-7
Three Alternate SYSTAT Uses

```
.R SYSTAT                    (run SYSTAT from OS/8)
*/L/N = 5                    (specify options)

.SYSTAT/L/N = 5              (run under OS/8 via CCL)

!SYSTAT;5LN                  (run from SCALE)
```

The options can be specified in any order. The "L" option is ignored if you are logged out. You cannot specify more than one job number for one run of the program.

4.6 TIME PROGRAM

The current time of day may be obtained by entering

```
.R TIME
*<RET>
```

The time is in the form: hour:minute:second A.M. or P.M. If the user invokes the TIME program via the CCL command TIME, there are several options available in addition to the current time. These options may be entered in any order. A list of these options and their meanings are contained in Table 4-5.

Table 4-5
TIME Options

Option	Information Printed
/C	Connect time for the current user
/R	Run-time for the current user
/U	Amount of time ETOS has been running
/V	Version number of the TIME program
=n	Job number for connect and run times

The connect-time and run-time are normally displayed for the current user. The job number for the user is also displayed, enclosed in parentheses. The user may want to see the run-time and/or connect-time for another job. To accomplish this, enter "=n" in addition to the options. "n" represents the job number that he wants information about. Two sample executions are contained in Figure 4-8.

Figure 4-8
Sample Run Of Time

```
.R TIME
*/C/R/U/V
'TIME' V2.Ø13
Ø2:12:3Ø P.M.
UP-TIME    - Ø6:2Ø:1Ø
CONNECT-TIME (Ø4) - Ø1:Ø2:3Ø
RUN-TIME     (Ø4) - ØØ:Ø1:37.2

.TIME/C/R/=5
Ø2:12:5Ø P.M.
CONNECT-TIME (Ø5) - Ø6:Ø1:5Ø
RUN-TIME     (Ø5) - ØØ:1Ø:27.9
```

The TIME program can also be used to output the time of day or system times to an OS/8 file. To output to a file, enter an OS/8 filename in response to the "*" or on the same line as the TIME CCL command. For example, to output to the file TIMSET.TX, enter one of the sequences contained in Figure 4-9.

Figure 4-9
Output Of Time To A File

```
.R TIME
*TIMSET.TX/C/R
.
```

or

```
.TIME TIMSET.TX/R=5
```

The file created by the TIME program may be printed on the terminal via the TYPE command.

4.7 WHO PROGRAM

Current job data may be obtained by entering

```
.R WHO
```

The current job number, account number and console number are printed. WHO is provided as a quick means of determining the ownership of unattended terminals, keeping track of jobs when ATTACHing and DETACHing, and so forth. This program is distributed under OS/8, but may also be run under COS. A sample execution is contained in Figure 4-10.

Figure 4-10
Sample Run Of WHO

```
.R WHO
'WHO' V1.00Ø2
JOB #Ø3, ACCOUNT #[Ø0,03],CONSOLE #Ø1
.
```

4.8 INQUIR PROGRAM

The status of all ETOS channels (see Chapter 7) may be obtained by entering

```
.R INQUIR
```

For each open channel, the device, account number, ETOS file name, protection code and length are printed. For non file-structured devices, only the the device name is displayed. The offset is unavailable. This program is distributed under OS/8 but may also be run under COS. A sample execution is contained in Figure 4-11.

Figure 4-11
Sample Run Of INQUIR

```
.R INQUIR
'INQUIR' V1.004

CHN # DEV ACCOUNT FILE NAME PROTECTION LENGTH
CHN 0 SYS:[00,02] OS8     RTS    <1152>   00003300
CHN 1 SYS:[00,03] JOB03   SBK    <0025>   00000030
CHN 2 SYS:[00,03] OS8DISKDSK     <0050>   00001630
CHN 3 DK1:
CHN 5 DK2:[10,10] COSDISKRTS     <0050>   00000512
.
```

4.9 LINE PRINTER AND TERMINAL SPOOLING

Under ordinary usage, the line printer is assigned to a single user for his exclusive use for as long as needed. If other users want to use it, they must wait until the owner releases it. Also, the job using the line printer must keep swapping in and out of core while it waits for the comparatively slow device. Under spooling, the line printer is not assigned directly to the jobs that use it. Requests for output are stored temporarily on disk by the special OS/8 device handler QLP:. In this way, several users can appear to be using the line printer

simultaneously. Each program also runs faster, since the disk is faster than the line printer. Programs now spend less time in core, reducing the swapping overhead of the system. When the entire listing has been put on disk, the spooler puts the temporary file containing the output in a queue for actual printing. This queueing function, and the actual listing of the intermediate ETOS file, is performed by a special system job known as the spooler. In this way, the line printer is utilized as efficiently as possible.

Under OS/8, the QLP: device handler produces spooled line printer output if spooling is in effect. For example, the command contained in Figure 4-12 could be used to spool a CREF listing file.

Figure 4-12
Sample Use Of Spooling

```
.CREF PROG,QLP:<PROG
```

The system manager has to initiate a spooling controller each time that the system is started. QLP: directs the output to logical spooler device LPØØ:. If no spooler job has been initiated for this logical device and device QLP: is used, the message "?BAD QUEUE DEVICE" is displayed.

In this case, SCALE's QUEUE command (see 3.11) must be used to direct the temp file created by QLP: to the correct logical spooling device. For example, suppose the spooling device LPØØ has not been initialized. Figure 4-13 contains a sample of using SCALE to spool the temporary file.

Figure 4-13
Sample Spool From SCALE

```
.CREF PROG,QLP:<PROG
?BAD QUEUE DEVICE
```

At this point, assume that the System Manager initiates the spooler for LPØØ on another terminal.

```
.^VS
!QUEUE LPØØ:=TEMPØØ.LPT
!CONT
```

Note: If TEMPØØ.LPT already exists, QLP creates TEMPØ1.LPT, then TEMPØ2.LPT, etc.

When a process involving the QLP device handler does not terminate normally (as with the "?BAD QUEUE DEVICE" message), the TEMP??.LPT file it was creating should be QUEUEd (as above) or DELETEd. Otherwise the temp file will never be deleted.

If no spooler jobs are running, the user must ASSIGN LPT to his job and use the LPT: device handler to do the output.

The system manager can set up the system so that LPT: is the queueing device handler. If this is done, the non-spooling line printer handler must be renamed. If the manager renamed the spooling device to LPT: and the non-spooling device to LPN:, you would type CREF PROG,LPT:<PROG<RET> to spool a CREF listing or ^VASSIGN LPT<RET>, CREF PROG,LPN:<PROG<RET> to directly print a CREF listing.

The line printer spooler is not directly accessible under COS. COS must print to a file and then spool the file under OS/8 after running CONVEX to convert it to proper format.

CHAPTER 5
DIFFERENCES IN COS

5.1 INTRODUCTION

The differences between stand-alone COS and ETOS COS can be separated into three areas. These are changes made to the operating system itself (COS monitor), changes made to the standard COS system programs (e.g., DIRECT, PIP) and there are additional programs under ETOS COS which were written expressly for ETOS (e.g., BREAK, SYSGET).

During LOGIN, ETOS allocates between 8K and 20K words of virtual memory to you and starts (bootstraps) OS/8. After the bootstrap is completed, you have full access to OS/8. To enter ETOS COS, enter R COSBO<RET> in response to the OS/8 monitor dot (.). The system manager must have set up a COS area in your account, or the error message "CANNOT FIND COSDISK.RTS" is printed and OS/8 is reBOOTed. After the COS bootstrap is completed by COSBO, you have automatic access to a private COS storage area called DK0. This storage area is separated from all other users. In addition to DK0, you also have access to non-disk peripherals (i.e., KBD, TTY, CDR, LPT, DT0 and DT1). You do not have automatic access to any other device on the system. If you wish to access other accounts or non-system disks, you must use the data channels defined in Chapter 7.

If you destroy the copy of COS which resides in virtual memory, bring in a new copy from the disk by entering .^VS;R COSDISK.RTS.

To return to OS/8 while you are in COS, enter R OS8BO in response to the COS monitor dot (.). If the program OS8BO does not reside in your COS area, enter ^VS;R [0,2]OS8.RTS.

5.2 COS MONITOR CHANGES

ETOS COS appears to the user almost the same as stand-alone COS. However, there are some differences: a RUBOUT entered on a scope terminal does not physically delete the character from the screen. The deleted character is re-echoed. Entering ^VS stops any program from running and returns the user to SCALE mode. ^V therefore cannot be used to suppress repetition of the message "COS MONITOR 3.Ø7GB". If a COS program is not in input wait, a double ^C is needed to return control to OS/8. This allows ^C to be used in type-ahead commands. ^V^C is sometimes required to return to the COS monitor.

When a COS program is running, the user may type ahead commands which are to be executed when the current operation is finished. These commands will echo during their entry. This allows you to set up a batch stream from the terminal. If you type ahead too many characters, no more input is accepted or echoed until some characters are read from the input buffer. Commands which are typed ahead are lost if the input buffer is cleared. Entering ^V causes the input buffer to be cleared.

A sample use of type-ahead is contained in Figure 5-1.

Figure 5-1
Sample Use of Type-ahead

```
.VASSIGN LPT:
R DFDIR/L
```

While the data file directory is being produced on the line printer, the following commands are entered on the terminal.

```
FETCH PROG          (fetch a program into source buffer)
R COMP/N            (compile the program)
SAVE PROG           (save the compiled output)
DI/T                (COS directory of DKØ)
```

When the data file directory is finished, the following is displayed on the terminal.

```
COS MONITOR 3.Ø7GB          (end of DFDIR)
..                          (FETCH)

COS MONITOR 3.Ø7GB          (compilation)
.

COS MONITOR 3.Ø7GB          (SAVE)
.

DIRECTORY      Ø3-JAN-79

NAME     TYPE      LN      DATE

COMP     V         14      25-JUL-73
PROG     S         Ø1      Ø1-JAN-79
PROG     B         Ø1      Ø3-JAN-79
<ØØ29 FREE BLOCKS>

COS MONITOR 3.Ø7GB
.                           (end of Directory)
```

The COS monitor enables ETOS to perform character echoing. The COS monitor sets the BREAK mask to 6Ø7 and the KSTAT word to 4375. All keyboard input via the ACCEPT statement must be terminated by a control character such as carriage return. If you must terminate input with a non-control character, see 5.6.

When a carriage return is entered in response to an ACCEPT statement, a line feed is automatically generated by ETOS. Usually, this causes no problems. However, if the carriage return is entered on the last line of a screen, the top line of the screen is scrolled up and lost.

The DISPLAY statement may be used to output codes other than the standard scope codes (e.g., clear screen, position cursor) with the literal or field features (see COS300/310 System Reference Manual, Section 1.4.5). However, as the manual states, equivalent results are not guaranteed on different or future versions of COS. For example, the DISPLAY of a decimal code 95 under COS310 version 6.05G outputs an underline (_). Under ETOS COS300, the same code outputs an "A".

All device handlers have been modified to operate under ETOS. KBD and TTY are accessible to all users. CDR and LPT are accessible unless ASSIGNed by another user. If you are privileged, the RX01 floppy disk may be accessed as DT0 and DT1. The RK05 or System Industries handlers (DK0-DK3) are not active under ETOS. DK0-DK7 represent ETOS channels (see chapter 7). All disk access is controlled with these data channels.

Versions of COS after and including Version 8 utilize the date format dd-mmm-yy (e.g., 05-JAN-79). The date format for ETOS COS is mm/dd/yy (e.g., 01/05/79). This format is the same as versions of COS prior to Version 8. The base year for single-user COS is 1972. Since the date offset is stored in three bits, the largest year that can be represented is 1979. Under ETOS V5B, the base year for COS has been moved up to 1979. The largest year that can be represented is 1986. Dates on newly saved programs and files are correct. Dates on previously saved programs and files are represented as seven years later than their actual creation date. To correct these dates, re-write the files.

5.3 CHANGES TO COS SYSTEM PROGRAMS

In addition to the changes in the COS monitor, there are also changes in some of the COS system programs. Since all of the programs use COS monitor device handlers, changes do not need to be made in each program's terminal services as they are required in OS/8.

Some of the standard COS programs are not appropriate for use under ETOS. These programs access hardware in a mode not supported under ETOS. The COS programs which are not supported under ETOS are listed in Table 5-1.

Table 5-1
COS Programs Not Supported Under ETOS

Program Name	Stand-Alone Function
BOOT	Boots COS monitor on any supported system device.
DTMARK	Formats TCØ8 DECtapes.
FILEX	Converts floppy diskettes to various formats.
RKEMRK	Formats RKØ5 disk packs.
RK8MRK	Formats RKØ1 disk packs.
TDMARK	Formats TD8E DECtapes.

A list of the modified COS system programs, which are supported under ETOS, and a list of changes follows. Since these programs have been modified to run under ETOS, the stand-alone versions of the programs should never replace these modified programs on the ETOS disk.

PIP

Stand-alone COS3ØØ PIP allows device designations DKØ through DK3, which refer to physical disk drive. The ETOS version of PIP has been modified to accept DKØ through DK7. These DK designations refer to ETOS channels (see Chapter 7).

SYSGEN

SYSGEN is not used under ETOS. The reason SYSGEN cannot be used is that it always assumes that a disk area is 406 segments. Under ETOS, the size of a disk area varies from user to user. The functions of SYSGEN are handled by COSBLD and SYSGET. COSBLD (discussed in the manager's guide) is used to build a COS system head in a disk area and initialize a COS system device directory. SYSGET (see 5.5) is used to specify logical units and change the COS directory length after a COS storage area has been EXTENDed or REDUCEd.

5.4 CONVERTING STAND-ALONE COS PROGRAMS TO ETOS

Programs written in DIBOL do not have to be modified to run under ETOS. However, under some version of COS, compiled programs must be recompiled from the source. To determine if you must recompile your programs, copy a compiled program onto the ETOS disk and see if it runs properly.

Batch streams must be modified to run under ETOS. There are two changes which must be made. The program SYSGET (see 5.5), which replaces SYSGEN, can accept logical unit assignments only from the source buffer. Therefore, the batch stream line

```
100 RUN SYSGEN, AR310A
```

must be modified to

```
100 FETCH AR310A
101 RUN SYSGET
```

The other type of change which is required is due to different device names under ETOS. Under stand-alone mode, two dual RX01 floppy disk drives may be access as RX0 and RX1, and RX2 and RX3. Under ETOS, only one dual RX01 floppy disk drive may be accessed.

The ETOS device names are DTØ and DT1. Disk names are also different under ETOS. Under stand-alone mode, DKØ through DK3 refer to physical drive numbers. Under ETOS, DKØ through DK7 refer to data channels (see Chapter 7) which handle all disk communication. Instead of a program disk and several file disks, programs and files are maintained in separate areas (accounts) on the ETOS public disks.

In addition to the required changes, you may want to make some modifications due to the fact that multiple systems are on-line at the same time. To acquire enough space to run multiple systems, you may have to reduce some of the file sizes. If you are converting from a floppy disk system, you may be able to extend file sizes with the additional space. Mounting instructions for floppy disk batch streams should also be changed to avoid operator confusion.

If you wish to develop an assembler program which runs under COS (e.g., OS8BO, SYSGET), you must develop the program under OS/8 and copy it to a COS system area to run it. Since the program is developed under OS/8, it must conform to the rules specified in section 4.4, with the exception of terminal service. All COS programs utilize the COS monitor handler for the terminal, which has been modified to run under ETOS.

5.5 SYSGET PROGRAM

SYSGET is an 8K COS-300 CUSP that runs under ETOS. It must be used instead of SYSGEN whenever a sysgen is performed under ETOS. (The name "SYSGET" is intended to suggest "SYSGEN/T"). SYSGET sets up the sysgen table and adjusts the length of file storage area of the COS system device (DK0). Under ETOS, SYSGEN does not assign the proper lengths to ETOS files or private packs. SYSGET circumvents this problem and also offers many other useful features not available in SYSGEN.

5.5.1 Operating Instructions

SYSGET has no options. It reads the input sysgen table from the editor text buffer and performs the sysgen. A typical call example is contained in Figure 5-2.

Figure 5-2
Sample Run of SYSGET

```
.FETCH SYSTAB          (get sysgen table into text buffer
.R SYSGET              (perform sysgen)
'SYSGET' V1.026        (header & version number)
DONE                   (indicates successful completion)
COS MONITOR 3.07GB     (back in COS-300 monitor)
.
```

5.5.2 Input SYSGEN Table

The COS editor's text buffer is assumed to contain SYSGET's input file (which contains the sysgen's parameters). The syntax of this file is a superset of SYSGEN's input files.

As in COS SYSGEN, the first line of the input file defines logical unit #1; the second line defines logical unit #2, etc. Any text past the fifteenth line is ignored. If there are fewer than 15 input lines, the remaining logical units are undefined.

The general form of each input line is

line dvn[,[length][,[start][,filespec]]][;comment]

"line" is the COS-300 text editor line number.

"dvn" is a recognized COS-300 device, composed of two alphabetic characters followed by an octal digit. Legal devices are DT0, DT1, and DK0 through DK7.

"length" is the decimal length of the logical unit in segments (1 segment = 16 blocks). If "length" is not specified, then all available space is allocated to the logical unit.

Note: The preceding three values constitute the entire syntax of the SYSGEN program. If only these specifications are used, then the command .RUN SYSGET will operate analagously within the ETOS environment to the command .RUN SYSGEN on a standalone COS system. All remaining specifications are features that only SYSGET provides.

"start" is the decimal starting segment number of the logical unit. If "start" is not specified, then the logical unit will start immediately after all other logical units on the device. If this feature is utilized, the next logical unit is assumed to start immediately after the last unit assigned without a "start".

"filespec" is the ETOS file specification that corresponds to the COS-300 device. "dev" must be of the form "DKn", but it must not be "DK0", since this would reassign the COS-300 system device. DK1 and DK2 should not be used unless you need more than five channels. channels 1 and 2 may be required if you return to OS/8 in this account. "filespec" is of the form

(1) [dev:][[proj,prog]]filename[.ext]
(2) [dev:][[account]]filename[.ext]
(3) dev:[,offset]

Forms 1 and 2 are basically the same, and are only used for ETOS file structured devices. Form 3 is for non-ETOS file structured private disk packs. Non-ETOS file structured disk packs are ASSIGNed by SYSGET.

"filespec" must not be specified more than once for the same COS device, and it must appear on the first line in which

the COS device is mentioned. If all these conditions are met, ETOS channel "n" (COS300 device DKn) has the ETOS file or device "filespec" LOOKed UP on it. Any ETOS file previously open on channel "n" is closed.

"dev:" is a recognized ETOS random-access (Channel I/O) device name. If not specified, "SYS:" is assumed.

"proj" is the project number of the ETOS file.

"prog" is the programmer number of the ETOS file.

"account" is the account number of the ETOS file. "account" is equivalent to "proj,prog"; if neither are specified, the user's account is assumed.

"filename" and ".ext" together specify the ETOS file. If ".ext" is omitted, then a null extension is assumed. ".ext" should not be specified without also including "filename". The filename and extension specification is normally DATAF1.LES. This is only due to the fact that COS data file areas are normally created with this name by the system manager.

"offset" is a 23-bit octal block offset for non-ETOS file structured devices.

"comment" is any text that the user cares to include. It is ignored by SYSGET. Note that comments are preceded by semicolons, just as in DIBOL source programs.

Note: Any spaces or tabs in the input file are completely ignored. They are also omitted from error message display of erroneous input lines.

A sample SYSGET table is contained in Figure 5-3.

Figure 5-3
Sample SYSGET Table

```
100 DK3,16,DK1:[20,12]DATAF1.LES;CUSMAS/1
110 DK3,3;CUSIDX/2
120 DK4,3,9,DK1:[70,1]DATAF1.LES;WORK1/3
130 DK4,3,12;WORK2/4
140 DK4,3,15;WORK3/5
150 DK5,7,DK0:[20,6]DATAF1.LES;SALES/6
160 DT0,20;MONSLS/7
170 DT0,20;MONSCH/8
180 DK6,7,23,DK2:;INVMAS/9
```

This SYSGET table allocates nineteen segments of data files to relative segments 0-18 in the ETOS file DK1:[20,12]DATAF1.LES. Nine segments of data files are allocated to relative segments 9-17 in the ETOS file DK1:[70,1]DATAF1.LES. Seven segments are allocated to relative segments 0-6 in the ETOS file DK0:[20,6]DATAF1.LES. Forty segments are allocated to relative segments 0-39 on the floppy disk drive 0. Seven segments are allocated to relative segments 23-29 on a standard COS data pack in drive 2.

5.5.3 Space Allocation Algorithms

SYSGET has two major functions: (1) set up the sysgen table, and (2) adjust the length of the file storage area on DK0 (COS system device). It uses the same algorithms for these functions as the SYSGEN program, as long as the starting segment number is not specified. SYSGEN does not have this parameter.

If no starting segment number is specified, logical units begin immediately after all other logical units on the device. If no length is specified, the logical unit is expanded to fill all available space. It is not assumed that a logical unit will fit in place where it is allocated; complete error checking is performed to ensure that logical units do not overlap or extend past the end of the device. SYSGET must know the length of a device to allocate logical units on it. The CHANIO INQUIRE function (see 7.4.14) is used to determine the length of virtual disks DK0-7. Other devices such as DECtape and floppy disk have fixed lengths.

If logical units are being allocated on the COS system device (DK0), one must consider the system directory. The space taken up by the COS system head and existing files is fixed; but the free space (empty entry) at the end is variable, and SYSGET adjusts this every time it runs. The goal is to provide as much

space as possible for logical units (LU) but also make the empty entry as large as possible. Logical units have priority, with those of unspecified length being allocated every available segment. If only fixed length LUs are allocated on DKØ, the empty entry gets all remaining space. If no LUs are allocated, the empty entry gets all space. If the starting segment is specified for a logical unit on DKØ, the segment number is relative to the beginning of logical unit storage. The total length of the directory storage area cannot exceed 4Ø95 blocks (just under 16 segments). The total length of the directory, plus all logical units cannot exceed 4Ø95 segments. It is doubtful this limit will ever be reached since there are few devices for the PDP-8 that are this large. Due to the complexity of determining the location of files allocated on DKØ, it is recommended that files be stored in different ETOS areas than programs.

It is possible to map logical units used by different applications into the same area. For example, to distribute accounts receivable data to the general ledger, you may want to map an accounts receivable logical unit into the general ledger file, which is also mapped by the general ledger system. ETOS COS does not have a record lockout facility, which prevents two users from updating the same record at the same time. It is possible that if two users updated the same record at the same time, one of the updates would be lost. There are two ways to circumvent this problem. One method is to implement a record lockout facility in the applications programs. A list of active records could be maintained in the header record of each file. This header record could be checked by each program updating the file. The other method is an operational solution. You can organize your processing so that you are never running two programs which update the same file. This organization could be accomplished by scheduling certain updates for specific periods of time.

5.5.4 Error Messages

Whenever SYSGET is run, it displays an identification header line. This line contains the program name and version number. The current header line is

```
'SYSGET' V1.Ø26
```

After displaying the header line, SYSGET processes the input. If there are no errors

```
DONE
```

is displayed and then returns to the monitor, which is indicated with

```
COS MONITOR 3.Ø7GB
.
```

If an error occurs during SYSGET, a message is printed in the format contained in Figure 5-4.

Figure 5-4
SYSGET Error Format

```
? error message[IN LINE line, L.U. n: rest of input line]
ABORTING SYSGET
?%VM HLT
PC 14 AC 1 FL 5100 MQ 1500 SC 1 SR 0
!
```

The bracketed data is included in all error messages unless otherwise noted.

"error message" is the particular error.

"line" is the decimal line number that contained the error.

"n" is the decimal current logical unit number.

"rest of input line" is the unprocessed remainder of the input file line, including (in most cases) the character that caused the error. Any spaces or tabs in the input file are omitted from the display.

As the line "ABORTING SYSGET" implies, all errors are fatal and cause immediate exit from COS to SCALE. The reason that COS is exited is that you may be running under BATCH. It is important not to continue a batch stream after a SYSGET error. At this point, SYSGET has changed nothing. The core, resident sysgen table, the disk resident sysgen table, and the system directory are unmodified. To return to COS from SCALE, enter BOOT<RET>. If the COS resident monitor displays "I/O ERROR ON DK; RETRY?", it is possible that the disk data is bad. The recommended response is Y<RET>. If the disk data is bad, the COS system device may have to be rebuilt.

Following is a list of all SYSGET errors.

?DIRECTORY OVERFLOW

There is not enough room in the system device directory for the

final empty entry. Consolidate the directory with PIP's option "E" and try again. Note: the bracketed data in the general error format is not included in this error message.

?DKn NOT OPEN

The user did not specify an ETOS file, and there was no file already open on ETOS channel "n". In this case, there is no way for SYSGET to assign logical units on DKn because it cannot tell how long it is.

?ETOS FILESPEC NOT ALLOWED HERE

(1) The user has associated an ETOS file with a COS device other than DK1 through DK7; or

(2) He has associated an ETOS file with a particular COS device more than once in the same input file; or

(3) He has associated an ETOS file with a COS device after he has defined a logical unit on the device.

?ETOS LOOKUP FUNCTION FAILURE n

The system cannot access the ETOS file that was specified. For the meaning of "n", inspect the error codes specified in section 7.2.4.

?LOGICAL UNIT n OVERLAPS

The user has specified a starting segment for the current logical unit that causes it to overlap the previously defined logical unit #n.

?NO ROOM FOR LOGICAL UNIT

(1) There are no free segments on the device; or

(2) The user has specified a length for the logical unit that is longer than the free space on the device.

?NO ROOM ON SYS:

SYSGET has found that the COS system directory maps a file past the end of the ETOS file that corresponds to the COS system device. In other words, actual system file storage extends past the physical end of the COS system device. The file(s) at the end of the directory are at least partially lost. Note: the bracketed data in the general error format is not included in this error message.

?NUMBER TOO BIG

A number in the input line was too big.

?OFFSET TOO BIG

The offset on a private pack was larger than the full size of the entire pack.

?SYNTAX ERROR

A syntax error or error of form was found in the input file.

?UNKNOWN COS DEVICE "dv"

The user's specification for "dvn" (see 5.5.2) was an unknown device to SYSGET.

5.6 BREAK AND UNBRK PROGRAMS

These two programs manipulate the keyboard status of the COS job. They are included to give the user the option of having the ACCEPT statement under ETOS COS function exactly as stand-alone COS.

Under COS-300, the ETOS monitor normally buffers input characters until a special "break" character is typed. Normally, this break character must be a control carriage (e.g., carriage return, escape, line feed). This is done for efficiency so that the system does not have to run the program (possibly causing a swap) each time an input character is typed. To further reduce overhead, the ETOS monitor echoes input characters and handles the special characters ^C,^O,^S,^U, and RUBOUT.

This scheme works well for most of COS. Unfortunately, it is incompatible with certain details of DIBOL's ACCEPT statement. The ACCEPT statement reads a fixed number of characters, and does not require a delimiting carriage return. However, ETOS will not pass the input characters to the ACCEPT statement until an ETOS break character was entered. This is particularly disruptive for one and two character fields, and can result in the user destroying the screen format by entering past the end of the input field.

The BREAK program provides some relief from these problems. It causes ETOS to break on every character, and also disables handling of the special characters ^U and RUBOUT. The ETOS monitor still echoes characters and handles ^C,^O, and ^S. It is less efficient to break on every character, but it normalizes the operation of the ACCEPT statement. The UNBRK program restores the ETOS keyboard handling parameters to their normal values for COS. BREAK changes these parameters until the UNBRK program is run or COS is re-booted. Because the BREAK program results in

lower system efficiency, we recommend that it only be used when needed, as in the example contained in Figure 5-5.

Figure 5-5
BREAK and UNBRK Example

```
.RUN BREAK      (set keyboard status to break on every character)
.RUN PROG       (DIBOL program that uses ACCEPT statement)
.RUN UNBRK      (keyboard status back to normal)
.
```

BREAK and UNBRK have no options. Both may be used under COS BATCH. The BREAK programs sets the ETOS break mask to 4000. The UNBRK program sets the ETOS break mask to 0607. For information about the break mask, see Section 3.7.5.