

The word "digital" is written in a lowercase, sans-serif font, with each letter contained within its own white rectangular box. The boxes are arranged in a single horizontal row.

digital

The title is centered within a white rectangular box. The text is in a bold, sans-serif font.

**VAX-11 EDT Editor
Reference Manual**

The order number is centered within the same white rectangular box as the title. It is in a standard sans-serif font.

Order No. AA-H944A-TE

The logo consists of the letters "VAX11" in a large, bold, sans-serif font. The letters are white and set against a blue background. The "1" is slightly smaller than the other characters.

VAX11

April 1980

This manual describes how to use the EDT interactive text editor. The manual is intended for all users.

**VAX-11 EDT Editor
Reference Manual**

Order No. AA-H944A-TE

OPERATING SYSTEM AND VERSION: VAX/VMS V2.0
SOFTWARE VERSION: EDT V2.0

To order additional copies of this document, contact the Software Distribution Center, Digital Equipment Corporation, Maynard, Massachusetts 01754

digital equipment corporation · maynard, massachusetts

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license, and may only be used or copied in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by DIGITAL or its affiliated companies.

Copyright © 1980 Digital Equipment Corporation

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

The following are trademarks of Digital Equipment Corporation:

DEC	FOCAL
DECnet	IAS
DECsystem-10	PDP
DECtape	RSX
DECUS	UNIBUS
DIBOL	VAX
DIGITAL	VMS

Contents

	Page
Preface	<i>ix</i>
Chapter 1 Introduction	
1.1 EDT Editor Summary	1-1
1.2 EDT Commands	1-2
1.2.1 EDT Control Commands	1-2
1.2.2 Line Editing Commands	1-2
1.2.3 Character Editing	1-3
1.2.4 Nokeypad Character Editing	1-3
1.2.5 Keypad Character Editing	1-3
1.3 Text Buffers	1-3
1.4 Line Numbers.	1-4
1.5 Startup Command Files	1-4
1.6 Journal File.	1-4
1.7 Help Facility	1-4
Chapter 2 EDT Tasks	
2.1 General.	2-1
2.1.1 Operating System	2-1
2.1.2 System Command Level Prompt	2-1
2.1.3 EDT Command Level	2-1
2.1.4 Your Terminal Position Marker	2-2
2.2 Starting EDT	2-2
2.2.1 The Login Procedure	2-2
2.2.2 The Command Line	2-3
2.3 Line Editing Commands: Creating New Files	2-4
2.4 EDT's Protection of Your Input	2-5
2.5 More Line Editing: New Files	2-6
2.5.1 More Line Editing: Revising Files.	2-9
2.5.2 And More Line Editing: Multiple Files	2-11
2.6 Using Character Editing: Nokeypad Editing	2-14
2.7 Using Character Editing: Keypad Editing.	2-19
2.7.1 How to Start Keypad Editing.	2-19
2.7.2 How to Get Help.	2-20
2.7.3 Keypad Functions	2-20
2.7.4 Inserting Text	2-20
2.7.5 Moving the Cursor	2-21
2.7.6 Marking Text with Select Ranges	2-24
2.7.7 Deleting Words and Characters	2-26
2.7.8 Deleting Lines	2-27
2.7.9 How to Change Case	2-31
2.7.10 Exiting Keypad Character Editing	2-31

Chapter 3 Protection Provided by EDT

3.1	The Journal File	3-1
3.2	Consistency Check.	3-3
3.3	Error Messages	3-3

Chapter 4 Line Numbers and Range Specifications

4.1	Line Numbers.	4-1
4.2	Range Specifications.	4-2
4.2.1	Single Line Ranges.	4-2
4.2.2	Contiguous Line Ranges	4-4
4.2.3	Noncontiguous Range Specifiers.	4-4
4.2.4	Text Buffer Range Specifications	4-5

Chapter 5 Editor Buffers

5.1	Text Buffers	5-1
5.1.1	Main Buffer	5-1
5.1.2	Paste Buffer	5-1
5.1.3	Additional Text Buffers.	5-1
5.2	Other Buffers	5-2
5.2.1	Line Buffer	5-2
5.2.2	Word Buffer	5-3
5.2.3	Character Buffer	5-3
5.2.4	String Search Buffer	5-3
5.2.5	Substitute Buffer.	5-3

Chapter 6 The Command Line and Command Files

6.1	EDT Command Line	6-1
6.2	Command Qualifiers.	6-1
6.3	Command Files	6-3

Chapter 7 Line Editing Commands

7.1	COPY Command	7-1
7.2	DEFINE MACRO Command	7-2
7.3	DELETE Command.	7-4
7.4	FIND Command	7-5
7.5	INCLUDE Command	7-6
7.6	INSERT Command	7-7
7.7	MOVE Command	7-7
7.8	Null Command	7-8
7.9	PRINT Command	7-9
7.10	REPLACE Command	7-9
7.11	RESEQUENCE Command	7-10
7.12	SUBSTITUTE Command	7-11
7.13	SUBSTITUTE NEXT Command	7-13
7.14	TYPE Command	7-14
7.15	WRITE Command	7-16

Chapter 8 Editor Control Commands

8.1	CHANGE Command	8-1
8.2	EXIT Command	8-2
8.3	HELP Command	8-3
8.4	QUIT Command	8-4
8.5	SET Command	8-4
8.5.1	SET CASE	8-5
8.5.2	SET CURSOR Top:bottom	8-5
8.5.3	SET ENTITY	8-6
8.5.4	SET KEYPAD	8-7
8.5.5	SET LINES Number	8-7
8.5.6	SET MODE	8-7
8.5.6.1	SET MODE CHANGE	8-7
8.5.6.2	SET MODE LINE	8-8
8.5.7	SET [NO]NUMBERS	8-8
8.5.8	SET [NO]QUIET	8-8
8.5.9	SET SCREEN Width	8-8
8.5.10	SET SEARCH	8-9
8.5.10.1	SET SEARCH EXACT	8-9
8.5.10.2	SET SEARCH BOUNDED	8-9
8.5.10.3	SET SEARCH END	8-9
8.5.11	SET TAB N	8-9
8.5.12	SET TERMINAL	8-12
8.5.13	SET [NO]TRUNCATE	8-12
8.5.14	SET [NO]VERIFY	8-14
8.5.15	SET [NO]WRAP N	8-14
8.6	SHOW COMMAND	8-14
8.6.1	SHOW BUFFER	8-15
8.6.2	SHOW CASE	8-15
8.6.3	SHOW CURSOR	8-16
8.6.4	SHOW ENTITY	8-16
8.6.5	SHOW KEY	8-16
8.6.6	SHOW SCREEN	8-16
8.6.7	SHOW SEARCH	8-16
8.6.8	SHOW TERMINAL	8-16
8.6.9	SHOW VERSION	8-16

Chapter 9 Nokeypad Character Editing

9.1	Entities of Text	9-1
9.2	Nokeypad Command Structure	9-3
9.3	Format 1 Commands	9-4
9.3.1	ADV (Advance)	9-4
9.3.2	BACK (Backup)	9-4
9.3.3	EX (Exit)	9-5
9.3.4	EXT (Extended)	9-5
9.3.5	I (Insert)	9-6
9.3.6	QUIT	9-6
9.3.7	REF (Refresh)	9-7
9.3.8	SEL (Select)	9-7

9.3.9	TAB.	9-7
9.3.10	TC (Tab Compute).	9-9
9.3.11	TOP.	9-9
9.4	Format 2 Commands (Without Direction).	9-9
9.4.1	ASC (ASCII).	9-9
9.4.2	SHL (Shift Left)	9-10
9.4.3	SHR (Shift Right)	9-10
9.4.4	TD (Tab Decrement)	9-11
9.4.5	TI (Tab Increment)	9-11
9.4.6	UNDC (Undelete Character)	9-11
9.4.7	UNDW (Undelete Word)	9-11
9.4.8	UNDL (Undelete Line)	9-11
9.4.9	^ (Circumflex)	9-11
9.5	Format 2 Commands (With Direction)	9-11
9.5.1	S/s1/s2/ (Substitute)	9-12
9.5.2	SN (Substitute Next)	9-12
9.6	Format 3 Commands	9-13
9.6.1	APPEND	9-13
9.6.2	CUT	9-13
9.6.3	D (Delete)	9-14
9.6.4	FILL	9-15
9.6.5	[Null] (Move Cursor)	9-16
9.6.6	PASTE	9-17
9.6.7	R (Replace)	9-17
9.6.8	TADJ (Tab Adjust)	9-18

Chapter 10 Keypad Functions In Change Mode

10.1	The Keypad and its Functions	10-2
10.2	Starting and Ending an Editing Session	10-3
10.2.1	Beginning the Editing Session	10-3
10.2.2	What it Means to Save or Delete Edits	10-3
10.2.3	Saving Your Edits	10-6
10.2.4	Deleting Your Edits	10-6
10.3	Essential Functions	10-6
	GOLD	10-6
	HELP	10-7
	RESET	10-8
10.4	Inserting Text and Lines.	10-8
10.4.1	Entering Text	10-8
10.4.2	Using OPEN LINE.	10-9
10.5	Moving the Cursor	10-10
10.5.1	The Arrow Keys	10-10
	UP	10-10
	DOWN	10-11
	LEFT	10-12
	RIGHT	10-13
10.5.2	Setting the Direction of Cursor Movement.	10-14
	ADVANCE	10-14

	BACKUP	10-14
10.5.3	Movement by Entity	10-14
	CHAR (Character)	10-14
	WORD	10-14
	EOL (End of Line)	10-15
	LINE	10-15
	BACK SPACE	10-15
10.5.4	Movement throughout the Buffer	10-16
	PAGE	10-16
	SECTION	10-16
	TOP	10-16
	BOTTOM	10-16
10.6	Finding Text	10-16
	FIND	10-16
	FNDNXT (Find Next)	10-17
10.7	Deleting and Reinserting Text	10-19
10.7.1	Deleting and Reinserting by Character	10-19
	DELETE	10-19
	DEL C (Delete Character)	10-19
	UND C (Undelete Character)	10-19
10.7.2	Deleting and Reinserting by Word.	10-20
	LINE FEED	10-20
	DEL W (Delete Word)	10-20
	UND W (Undelete Word).	10-21
10.7.3	Deleting and Reinserting by Line	10-21
	CTRL/U	10-21
	DEL EOL (Delete to End of Line)	10-22
	DEL L (Delete through End of Line)	10-22
	UND L (Undelete Line)	10-22
10.8	Selecting and Moving Text	10-23
	SELECT	10-23
	CUT	10-24
	PASTE	10-26
	APPEND	10-27
10.9	Replacing and Substituting Text	10-30
	REPLACE	10-30
	SUBS	10-32
10.10	Entering Commands	10-33
	ENTER	10-33
	COMMAND	10-34
10.11	Special Characters, Changing Case, and Filling Lines	10-34
	SPECINS	10-34
	CHNGCASE	10-35
	FILL	10-35
10.12	Control Character Functions	10-36
	CTRL/C	10-36
	CTRL/U	10-36

CTRL/W	10-36
CTRL/Z	10-36
CTRL/A	10-37
CTRL/E	10-37
CTRL/D	10-37
CTRL/T	10-37
CTRL/K	10-37
10.13 DEFINE KEY Command	10-37

Appendix A Differences In Terminal Responses

A.1 Hardcopy Terminals.	A-1
A.2 Nokeypad Display Terminals.	A-2
A.3 Keypad Display Terminals.	A-3

Appendix B Error Messages

Appendix C ASCII Decimal Equivalents

Glossary

Figures

10-1 The VT52 Keypad and Its Functions	10-4
10-2 The VT100 Keypad and Its Functions	10-5
10-3 Keypad Numbers	10-39

Tables

6-1 EDT Command Qualifiers	6-2
9-1 Entities.	9-2

Commercial Engineering Publications typeset this manual using DIGITAL's TMS-11 Text Management System.

Preface

Document Objectives

The *VAX-11 EDT Editor Reference Manual* describes the EDT editor, EDT commands, the command qualifiers, and their usage. The manual provides the user with all the information necessary to use EDT.

Intended Audience

This manual introduces all users to the interactive text editor EDT and provides reference material on EDT. EDT is useful for creating and editing programs and text.

Document Structure

This manual is divided into 10 chapters, 3 appendixes, a glossary, and an index. Users new to text editors should read the complete manual, concentrating on Chapters 2, 4, 6, 8, and either 7, 9 or 10. Users with experience in using text editors should read Chapters 4, 6, 8, and either 7, 9 or 10 before using EDT. Users whose interest is primarily line editing should read Chapter 7; users whose interest is primarily nokeypad character editing should read Chapter 9; readers whose interest is primarily keypad editing should read Chapter 10. The examples use the VT100 terminal; the response differences for other terminals is in Appendix A.

Chapter 1 is an introduction with a general description of EDT's features.

Chapter 2 presents a series of sample tasks in order of increasing complexity. The first tasks are presented with the most detail. The tasks are presented as tutorial lessons that, when followed step by step, show EDT's responses to certain inputs.

Chapter 3 describes some of the protection provided by EDT. This protection gives the user visible proof of an EDT operation.

Chapter 4 describes the line numbering system and the range specifications used in EDT.

Chapter 5 describes the buffers used in EDT.

Chapter 6 describes the EDT command line and the command files.

Chapter 7 describes EDT line editing commands with examples of their use.

Chapter 8 describes editor control commands with examples of their use.

Chapter 9 describes EDT character editing commands with examples of their use.

Chapter 10 describes EDT keypad character editing commands with examples of their use.

Appendix A describes the differences in terminal displays in response to EDT.

Appendix B describes the error messages contained in EDT.

Appendix C lists the ASCII character codes in decimal.






The glossary describes the unfamiliar terms and abbreviations used in this manual.

Associated Documents

The user should refer to the associated manual on the operating system for information on file handling external to EDT. Users unfamiliar with VAX/VMS should read the *VAX-VMS Primer* (order no. AA-D030A-TE).

Conventions

The symbols used in this manual have the following meaning:

Symbol	Meaning
CTRL/x or 	The phrase CTRL/x indicates that you must press the key labeled CTRL while you simultaneously press another key; for example,  or  .
	The DELETE key symbol signifies the deletion of the character to the left of the cursor.
	The form feed symbol signifies the end of a page.

GOLD / or **GOLD/x** The phrase GOLD/x indicates that you must press the GOLD keypad key and then another key; for example, **GOLD/T** or **GOLD/A**.

RET The RETURN key symbol signifies a carriage return.

[] Brackets enclose optional parameters within command statements. Parameters describe the object of the command (that on which the command performs, for example, a file, a line, a word).

{ } Braces enclose a list of alternatives within a command, one of which you must specify.

| The OR symbol separates qualifiers for the same operation and the alternatives contained within braces.

\$ The DIGITAL command language (DCL) prompt at the left margin indicates the operating system is at DCL level.

* The EDT command level prompt at the left margin indicates that EDT is at command level.

Abbreviations The minimum abbreviation for a command is given in the command description. The minimum abbreviation is shown in bold type; the remainder of the word is shown in standard type. EDT accepts any input from the minimum abbreviation up through the complete command. The complete command word is used in this manual.

Color Red type indicates user input.

... Ellipses show that not all of the statements in an example or figure are shown. Ellipses may be horizontal or vertical. Also used for indefinite repetition in command formats.

Chapter 1

Introduction

This chapter contains an overall description of the EDT editor.

1.1 EDT Editor Summary

The EDT editor is an interactive, general purpose text editor. You can use EDT for generating new files and modifying old files. The files can contain programs or text.

Features include:

- Decimal line numbers
- English language based commands
- Original (fixed) line numbers
- Hard copy and screen display
- On-line error diagnosis
- Help facility
- File and buffer input/output handling
- Keypad editing
- Line editing
- Macro capability
- On-line help facility
- Journaling (a record of edit operations)

1.2 EDT Commands

EDT provides commands for all usual editing operations. EDT commands are divided into four groups:

EDT control commands

Line editing commands

Nokeypad character editing commands

Keypad character editing commands

There are two modes of operation: command mode and change mode. You enter EDT control commands and line editing commands from command mode. You enter character editing commands from change mode.

1.2.1 EDT Control Commands

EDT control commands control the operation of EDT. The control commands are:

CHANGE	Places EDT in either keypad or nokeypad character editing. See Chapters 9 and 10.
EXIT	Leaves EDT, saving the edited input as a file.
HELP	Displays help information.
QUIT	Leaves EDT without saving the input as a file.
SET	Sets editor operating parameters.
SHOW	Shows status of selected parameters.

1.2.2 Line Editing Commands

The line editing commands are those commands (other than the control commands and the HELP command) which you enter from EDT command mode, that is, after the asterisk (*) prompt.

The line editing commands are:

COPY	Copies a selected number of lines from one location in an EDT buffer to another.
DEFINE MACRO	Names a sequence of commands which are then performed when the macro name is entered as a command.
DELETE	Deletes selected line(s).
FIND	Finds a selected line.
INCLUDE	Copies external files to a specified text buffer.

INSERT	Opens a text buffer for the insertion of text.
MOVE	Moves a selected number of lines from one location in an EDT buffer to another.
Null	Signifies an implied TYPE command.
PRINT	Generates a printable file from a range of lines.
REPLACE	Deletes and inserts selected lines in a text buffer.
RESEQUENCE	Renumbers lines in a text buffer.
SUBSTITUTE	Replaces one string of characters with another string of characters.
SUBSTITUTE NEXT	Replaces the next occurrence of one string of characters with another string of characters.
TYPE	Displays selected lines.
WRITE	Copies selected text in an EDT buffer to a file.

You can modify the action of some of these commands with command qualifiers and range specifications.

1.2.3 Character Editing

To use either the keypad or the nokeypad character editing commands, enter the CHANGE command. For VT52 and VT100 terminals, EDT defaults to the keypad character editing commands. For other terminals, EDT defaults to the nokeypad character editing commands. To use the nokeypad commands on the VT52 or VT100, use the SET NOKEYPAD command.

1.2.4 Nokeypad Character Editing

The nokeypad character editing commands are entered through the keyboard; you also use the keyboard to enter text. You end the command entries with a **RET**, which tells EDT to execute the commands.

1.2.5 Keypad Character Editing

The keypad character editing commands are entered through the keypad keys; you use the keyboard to insert text. You can change the command assignment for the keypad keys with the DEFINE KEY command.

1.3 Text Buffers

You store and change text in areas called text buffers. Text buffers are organized into lines containing 0 to 255 characters. There is no explicit limit on the number of text buffers that you can use during an editing session.

1.4 Line Numbers

EDT assigns each line in a text buffer a unique line number when that line is input to the buffer. When you add a line that is not at the end of the buffer, EDT assigns the line a number between the number of the line preceding and the line following. When there are no unassigned whole numbers between the preceding and succeeding line, the current line is given a decimal number (for example, a line inserted between lines 3 and 4 is given the number 3.1).

Fixed line numbers are a fixed part of the file containing them. When you copy a file which has fixed line numbers, EDT maintains a record of them in a second field. The numbers in this field, which is not displayed, can be used as range specifications. You can also use EDT to write fixed line numbers to a file (see EXIT and WRITE commands).

1.5 Startup Command Files

A startup command file is a file containing EDT commands that are executed when you start EDT. You can define macros and keys and set editor operating parameters with startup command files. You can do this using the EDTINI.EDT file or the command file qualifier. If you do not use the command file qualifier, EDT searches your directory for the file EDTINI.EDT. EDT executes the commands in the command file before prompting you for input. When there is neither a command file nor the EDTINI.EDT file, EDT begins an editing session with the default values.

1.6 Journal File

EDT maintains a journal file containing all of your inputs to EDT for each editing session. EDT saves this file if there is a system failure. Therefore, the journal file provides protection against system failures during the edit session. You can direct EDT to execute the contents of the journal file at the beginning of another editing session. EDT deletes the journal file when you make a normal exit from the editing session unless you specify that it be saved. You can also edit the journal file.

1.7 HELP Facility

EDT has a help facility. Use the HELP command to display information about EDT commands on your terminal. A HELP key is available in nokeypad character editing. The help facility provides on-line documentation of EDT.

Chapter 2

EDT Tasks

This chapter contains examples of sample tasks using EDT commands. The examples are in order of increasing complexity. The introduction to each example lists the EDT commands used in the task. See Chapters 7, 8, 9, and 10 for descriptions and examples of the individual commands.

The illustrations for the examples show the interaction between you and the operating system. Red shows your input.

2.1 General

EDT is an easy-to-use text editor. The more you use EDT, the more confidence you will have in it and your ability to use it. Remember that EDT protects your work.

2.1.1 Operating System

EDT is a program included with your operating system. The operating system is the software package that controls the operation of the computer. This includes enabling you to communicate with the computer through your terminal. The operating system, including EDT, has a defined set of symbols to describe its status. Learn these symbols and the proper responses to them. The examples in this chapter use the VAX/VMS operating system symbols.

2.1.2 System Command Level Prompt

The command level prompt for the VAX/VMS operating system is a dollar sign (\$). This is present when you have access to the DIGITAL command language (DCL). You access EDT from the DCL level with the command line (see Chapter 6).

2.1.3 EDT Command Level

The EDT command level prompt is an asterisk (*). This is present when you can enter EDT command level commands. The * is not present when you are in insert or change mode.

2.1.4 Your Terminal Position Marker

Your terminal has a position marker or cursor.

▒ VT100

— VT52

(On VT100s, you can set the cursor to be either an underline or a blinking rectangle.) The cursor indicates your position for the entry of text and commands from the terminal. The cursor is present when your display terminal is working. The cursor presentation for other terminals is given in Appendix A.

2.2 Starting EDT

To use EDT, you must first have access to system command level. You have this access when you log in to your operating system.

2.2.1 The Login Procedure

To log in, you must have a working terminal and a user name and password that the operating system recognizes. This is your key to the operating system.

Example:

```
(RET)
Username: GUNDERS (RET)
Password: (RET)
Welcome to VAX/VMS Version n
A message of the day follows the welcome.
$ ▒
```

When you see the cursor on the display, enter a (RET). If the terminal is not logged in, the response to your (RET) input is 'Username:'. Enter your account name and a (RET). Respond to the 'Password:' prompt with your account password and a (RET). Your password is not displayed. When your entries are recognized, the operating system responds with a \$ prompt. The \$ is the DCL command prompt, after which you can enter any system command.

```
(RET)
Username: GUNDDRS (RET)
Password: (RET)
▒ Login - user validation error
```

If you make a mistake in your login sequence, you receive an error message.

The message indicates that the system does not recognize the username/password combination you used to log in.

2.2.2 The Command Line

When the \$ is present, you can enter any of the DCL commands. The DCL command for EDT is EDIT/EDT (see Chapter 6 for details on the command line). Determine the form of the command line you want to use. If you do not enter the input file specification, the operating system prompts you for a file name.

Examples:

```
$ EDIT/EDT(RET)
$_File: TRIAL(RET)
Input file does not exist
[EOB]
* █
```

In this example, you enter the command line without defining an input file. The operating system prompts you for an input file name. You can name a new file or enter the name of an existing file. After the * prompt, you can enter any EDT command level command.

```
$ EDIT/EDT NEXT(RET)
Input file does not exist
[EOB]
* █
```

In this example you enter the input file in the command line. In the first two examples, an input file with the name you specify does not exist. EDT displays an information message ('Input file not found'), an end of buffer symbol ([EOB]), and an EDT command prompt. (If you think your input file exists, check your file name against the directory.) The [EOB] indicates that EDT is at the end of the text in the current text buffer. The EDT command prompt indicates that you are at EDT command level; you can enter EDT commands.

```
$ EDIT/EDT TYPEA /OUTPUT=TYPEB(RET)
1 This contains text.
* █
```

In this example, you define an existing input file and a name for the output resulting from this editing session. EDT displays the first line of the input file and the EDT command prompt.

Remember that to gain access to EDT, you must have access to the operating system. This requires that you:

1. Log in to the operating system
2. Enter the DCL command for EDT

2.3 Line Editing Commands: Creating New Files

Once you enter EDT command level, you can start entering text into a text buffer; EDT always opens the text buffer MAIN. Assume, unless stated otherwise, that all references to a text buffer refer to the text buffer MAIN. When you exit an editing session, EDT moves the contents of the text buffer MAIN to a file. You control the name of the file through the command line you used to begin the editing session.

The simplest form of file generation is the insertion of new text. When you are at EDT command level, enter the INSERT command; this opens the text buffer for the insertion of text. Insert the desired text, ending each line with `(RET)`. When you wish to exit insert mode, enter `(CTRL/Z)`. End the editing session with an EXIT.

The following example is the simplest form of file creation using EDT. The EDT commands used are:

- INSERT
- EXIT

Example:

```
$ EDIT/EDT TEST (RET)
Input file does not exist
[EOB]
*INSERT (RET)
                                The text buffer is open for inserting text. (RET)
                                This is line two. (RET)
                                This is line three. (RET)
                                (CTRL/Z)
*EXIT (RET)
DB1:[GUNDERS]TEST. ;1 3 lines
$
```

In this example, your command line defines the input file TEST, which does not exist. Enter INSERT after the EDT command prompt. EDT opens the text buffer and enters insert mode. Enter the text on a line by line basis. The

text you enter is automatically indented because the first 16 spaces of each line are reserved for line numbers. The line numbers appear when EDT displays the line in response to other line editing commands.

As you enter a line, you can delete characters by pressing `DEL`. You end each line with `RET`. However, when you end a line, you can no longer delete characters from that line in insert mode. You can delete a portion of a line, from the cursor to the left margin, by entering `CTRL/U`. To exit insert mode, enter `CTRL/Z`; this returns you to EDT command level.

To end the editing session, enter the `EXIT` command. EDT then moves the contents of the text buffer to a file and names the file `TEST`. When the transfer is complete, EDT displays a message stating the location and size of the file. This is followed by the `DCL` prompt.

If you now enter `TEST` as the input file on your command line, EDT transfers a copy of the file `TEST` into the text buffer.

2.4 EDT's Protection of Your Input

It is important that you feel confident that EDT protects your work. One feature of EDT is the journal file, which is a record of your input to the terminal (text and commands). EDT adds to the journal file incrementally rather than continuously. When you end an edit session with other than the `EXIT` or `QUIT` command (without the `SAVE` qualifier) EDT automatically saves the journal file. You can restore your files to the condition that existed when the edit session ended by using the `RECOVER` qualifier in the command line.

Example:

```
$ EDIT/EDT RET
$_File: TESTER RET
Input file does not exist
[EOB]
*INSERT RET
          This is line one. RET
          This is line two. RET
          This is line three. RET
          This is line four. RET
          CTRL/Y
$ DIRECTORY RET
TESTER.JOU;1

Total of 1 file.
$
```

You enter the `INSERT` command and EDT opens the text buffer for the insertion of data. You end each line of data with `RET`. Assume that you make a mistake and enter `CTRL/Y` instead of `CTRL/Z`. EDT responds to `CTRL/Y` in much the same way it does to a system failure; the editing session is ended and the

journal file is saved. You can reenter EDT at the point where you entered **CTRL/Y** by entering CONTINUE (DCL command). Any DCL command that requires the execution of an image breaks that link to EDT. Assume you react to the \$ prompt with a DIRECTORY command before you think through the options; DCL responds by displaying the files in your directory (the TESTER journal file). Your link to EDT is now broken and you cannot use CONTINUE.

```
$ EDIT/EDT(RET)
$_File: TESTER/RECOVER(RET)
Input file does not exist
INSERT
This is line one.
This is line two.
This is line three.
[EOB]
*
```

To use the journal file, enter the same command line as before but add the RECOVER qualifier. EDT reads the journal file as if it were input to the terminal and displays the results. Note that the fourth line of your input is not recovered. EDT transfers your input to the terminal to the journal file at set intervals and the portion of your input between transfers can be lost.

You can continue your edit session. The journal file retains the input from the previous editing session with the input from this session added at the end. In this way you can have successive recoveries with one journal file until you exit with the EXIT or QUIT command.

2.5 More Line Editing: New Files

Editing sessions are usually more complex than the previous ones. The next example shows how to use more of the line editing commands. The commands used are:

- Null
- REPLACE
- TYPE
- DELETE
- EXIT

Example:

```
$ EDIT/EDT Chilly.Dat
Input file does not exist
[EOB]
*INSERT
    The town of Chilly, Me. is located 40(RET)
    miles north of Bangor.(RET)
    (RET)
    Chilly's main industry is tourism and(RET)
    it now contains no heavy or light(RET)
    industries. There is a sufficient(RET)
    labor base for a small light industry,(RET)
    but much of the labor force is unskilled.(RET)
    (RET)
    The average yearly temperature is 68(RET)
    degrees F; mean annual snowfall is(RET)
    172 inches.(RET)
    (CTRL/Z)
```

In this example, you enter text into the text buffer with the INSERT command. When the initial entry is complete, you enter (CTRL/Z) to exit insert mode. In reviewing the material, you decide to add a statement after the second line.

```
*TYPE 2(RET)
    2          miles north of Bangor.
*(RET)
    3
*REPLACE 3(RET)
1 line deleted
    Major access is by route 70/81,(RET)
    which is frequently snowbound.(RET)
    (RET)
    (CTRL/Z)
```

There are several ways to find a location (line) in the text buffer. One method is to make a best guess on the line number and then search from that point. To locate the position for adding the line, enter TYPE 2 and (RET); EDT displays line 2 and an EDT command prompt. Enter (RET); EDT automatically displays the next line in the text buffer, the blank line 3. This is the implied TYPE command; in the absence of any command preceding the (RET), EDT displays the next line in the text buffer.

You can insert the statement in place of line 3 by using the REPLACE command. Enter REPLACE 3; EDT deletes line 3, displays a message telling how many lines were deleted, and enters insert mode. After you enter the

statement and a blank line to replace the deleted line, you exit insert mode with **CTRL/Z**.

In a second review, you decide to add a recommendation to the report.

```
.  
.   
.   
*INSERT END  
  
      (RET)  
      Because of its inaccessibility and its (RET)  
      lack of a skilled labor force, we do not (RET)  
      recommend Chilly, Me. as a plant site. (RET)  
      (CTRL/Z)
```

To add the recommendation to the end of the buffer, enter the **INSERT END** command; **EDT** opens the text buffer for the insertion of text at the end of the current text. Enter **(RET)** to make the first line of this insert blank. Enter the statement, **(RET)**, and **CTRL/Z**.

```
*TYPE WHOLE (RET)  
  1      The town of Chilly, Me. is located 40  
  2      miles north of Bangor.  
  2.1    Major access is by route 70/81,  
  2.2    which is frequently snowbound.  
  2.3  
  4      Chilly's main industry is tourism and  
  5      it now contains no heavy or light  
  6      industries. There is a sufficient  
  7      labor base for a small light industry,  
  8      but much of the labor force is unskilled.  
  9  
  10     The average yearly temperature is 68  
  11     degrees F; mean annual snowfall is  
  12     172 inches.  
  13  
  14     Because of its inaccessibility and its  
  15     lack of a skilled labor base, we do not  
  16     recommend Chilly, Me. as a plant site.  
[EOB]  
*EXIT (RET)  
DBB1:[GUNDERS]CHILLY.DAT;1 18 lines  
$
```

To review the contents of the text buffer, enter the **TYPE WHOLE** command. **EDT** displays each line of the text buffer, in sequence, beginning with the first line. You can interrupt the scrolling action of the **TYPE WHOLE** command

with **CTRL/S**. To restart the display of the text buffer, enter a **CTRL/Q**. Notice that the lines you inserted between lines 2 and 3 are numbered with decimal numbers.

When you enter the **EXIT** command, EDT moves the contents of the text buffer to a file. When the move is complete, EDT displays a message with the complete name of the file and the number of lines in the file.

2.5.1 More Line Editing: Revising Files

You deal with existing files in much the same way you deal with new files. When the input file in your command line is an existing file, EDT loads the input file into the text buffer before prompting you for input. You can modify and add to the file.

For the next example, assume that you want to add to the **CHILLY.DAT** file. The EDT commands used include:

- **INSERT**
- **DELETE**
- **TYPE**
- **REPLACE**
- **SUBSTITUTE**
- **RESEQUENCE**

Example:

```
$ EDIT/EDT CHILLY.DAT RET
  1      The town of Chilly, Me. is located 40
*TYPE 4 RET
  4      which is frequently snowbound.
*INSERT 5 RET
          The nearest airport is in Bangor, RET
          although a seaplane lands once a week RET
          on nearby Lake Okichibwa. RET
          CTRL/Z
* RET
```

When EDT completes copying the file **CHILLY.DAT** into the text buffer, it displays the first line of the file and an EDT command prompt. In the first sequence, you use the **TYPE** command to locate where you are going to enter new text. The **INSERT** command opens the text buffer just before line 5 for the insertion of text. Enter **CTRL/Z** to exit insert mode.

```

*TYPE 2 (RET)
  2          miles north of Bangor.
*(RET)
  3          Major access is by route 70/81,
*SUBSTITUTE !70/81!193!(RET)
  3          Major access is by route 193,
1 Substitution
*(RET)
  4          which is frequently snowbound.
*TYPE 12 (RET)
  12         The average yearly temperature is 68
*DELETE 12 (RET)
1 line deleted
  13         degrees F; mean annual snowfall is
*DELETE (RET)
1 line deleted
  14         172 inches.
*DELETE (RET)
1 line deleted
  15
*INSERT END (RET)
          (RET)
          Factors considered include access, (RET)
          present industry, and labor force. (RET)
          (CTRL/Z)

```

Use the SUBSTITUTE command to replace the highway number. The SUBSTITUTE command uses a search string and a substitute string. The substitute string replaces the search string. The strings are separated by and enclosed by delimiters. A delimiter is a character that limits a string and therefore cannot be contained in the string.

To use the SUBSTITUTE command, you must place EDT at the line containing the string you want to substitute. To do this, move to the line prior to entering the SUBSTITUTE command or enter the line in the range specification. (Range specifications define line numbers for use with EDT commands.) The default range specification is the current line. Because the string you want to replace contains a slash, you cannot use that character as a delimiter. You can use any other special character. When EDT completes the substitution, it displays a message on the number of substitutions made and displays the corrected line. Use the implied TYPE command to see the next line, line 4.

You then decide to delete the lines about temperature and snowfall. You use the TYPE command to display the lines, and the DELETE command to delete the lines. EDT deletes the line, tells you how many lines were deleted, and displays the next line.

You decide to insert a final statement. Type INSERT END and insert the general statement, ending the insert with (CTRL/Z).

```

*RESEQUENCE (RET)
21 lines resequenced
*TYPE WHOLE (RET)
  1           The town of Chilly, Me. is located 40
  .
  .
  .
  21          present industry, and labor force.
[EOB]
*EXIT (RET)
DBB1:[GUNDERS]CHILLY.DAT;2 21 lines

```

To change the decimal line numbers, enter the RESEQUENCE command. EDT rennumbers the lines from 1 through 21. Enter a TYPE WHOLE command to review the contents of the text buffer and then use the EXIT command to end this editing session.

When EDT has moved the contents of the text buffer to a file, it displays a message with the complete name of the file and the number of lines in the file. Because this is the second time you have worked with this file, its version number is now 2.

To practice creating and revising files, make two new files similar in contents to CHILLY.DAT. You will use these files in the next example. Name the files TEPID.DAT and WARM.DAT.

2.5.2 And More Line Editing: Multiple Files

You can use several files as inputs to one editing session and generate several files during one editing session. The next example uses the following commands:

- INCLUDE
- PRINT
- WRITE
- SHOW
- MOVE
- COPY
- RESEQUENCE
- DELETE
- INSERT

Example:

```
$ EDIT/EDT SITES.DAT
Input file does not exist
[EOB]
*INSERT (RET)

                STATISTICS ON POTENTIAL PLANT SITES (RET)
                (RET)
                (RET)
                This report is a summary of the information (RET)
                about the 3 towns proposed as plant sites: (RET)
                (RET)
                Chilly, Me. (RET)
                (RET)
                Warm, Ariz. (RET)
                (RET)
                Tepid, Mo. (RET)
                (RET)
                (CTRL)Z

* █
```

In this example you generate a report. The report consists of new text that ties together information contained in other files. The material you inserted above introduces the text that exists in other files. Use the INCLUDE command to bring a copy of those files into this editing session.

```
*INCLUDE CHILLY.DAT =CHILLY (RET)
*INCLUDE WARM.DAT =WARM (RET)
*INCLUDE TEPID.DAT =TEPID (RET)
*SHOW BUFFER (RET)
MAIN      12      lines
PASTE     0       lines
CHILLY    21      lines
=TEPID    18      lines
WARM      26      lines
* █
```

Here you use the INCLUDE command to copy the external files into separate text buffers. The SHOW BUFFER command displays the names of the text buffers defined for this editing session. The = indicates the current text buffer.

```

*=CHILLY(RET)
  1           The town of Chilly, Me. is located 40
  .
  .
  .
  21          Present industry, and labor force.
*COPY 20, 21 to =MAIN END(RET)
2 lines copied
*

```

You can use a form of the TYPE command to move EDT from the text buffer TEPID to the text buffer CHILLY; enter =CHILLY. EDT displays the contents of the text buffer CHILLY with line numbers. The cursor is left at line 1. To copy the general statement at the end of the text buffer CHILLY to the end of the text buffer MAIN, use the COPY command. The COPY command inserts a copy of lines 20 and 21 at the end of the buffer MAIN. EDT positions the cursor just after the moved text.

```

*MOVE =CHILLY TO END(RET)
21 lines moved
*MOVE =WARM TO END(RET)
26 lines moved
*MOVE =TEPID TO END(RET)
18 lines moved
*RESEQUENCE(RET)
65 lines resequenced
*PRINT SITES.SUM(RET)
*EXIT(RET)
DBB1:[GUNDERS] SITES.DAT;1 77 lines
$

```

Use the MOVE command to move the contents of the text buffers to the end of the text buffer MAIN. MAIN now includes the title of the report, the introductory material of the report, the text from CHILLY.DAT, WARM.DAT, and TEPID.DAT, and the concluding lines copied from line 20 and 21 in CHILLY.DAT. The RESEQUENCE command renumbers the lines in MAIN from 1 to n. The PRINT command copies the contents of MAIN to the file space. The file is in a printable format and is assigned the name SITES.SUM. The EXIT command copies the contents of the text buffer MAIN into the file; this file is named SITES.DAT.

2.6 Using Character Editing: Nokeypad Editing

In character editing, EDT lets you modify text at the character level. To enter nokeypad character editing, you use the SET NOKEYPAD command for VT52 and VT100 terminals and then enter the CHANGE command. Nokeypad is the default mode for all other terminals; you enter the CHANGE command after the * prompt without using the SET command. The nokeypad commands are described in detail in Chapter 9.

In nokeypad character editing, the screen displays up to 22 lines of text; line numbers are not shown. EDT displays nokeypad character editing commands at the bottom of the screen. When you enter a command, the cursor moves to the command entry position. After EDT executes the command, it returns the cursor to the displayed text.

You can move the cursor to any position in the display with cursor movement commands. The format for a cursor movement command is a count of entities ending with `(RET)` (an entity is a quantity of characters), such as 3L for “3 lines.” You can modify the contents of the text buffer with additional nokeypad character editing commands. Your display represents the current contents of the displayed range within the buffer.

The example shows how to:

- Enter nokeypad change mode
- Move the cursor
- Insert text
- Delete and insert text
- Substitute text
- Cut and paste text
- Exit nokeypad change mode

Example:

```
$ EDIT/EDT CHILLY.DAT(RET)
1 The town of Chilly, Me. is located 40
*SET NOKEYPAD(RET)
*CHANGE(RET)

The town of Chilly, Me. is located 40
miles north of Bangor.
Major access is by route 193,
which is frequently snowbound.
.
.
.

[EOB]
```

You use the SET NOKEYPAD command to set EDT to nokeypad character editing when the CHANGE command is entered. When you enter the CHANGE command, the screen clears and then displays the contents of the text buffer. The cursor is at the beginning of the text buffer.

```
The town of Chilly, Me. is located 40
miles north of Bangor.
Major access is by route 193,
which is frequently snowbound.
.
.
.
[EOB]
3L
```

```
The town of Chilly, Me. is located 40
miles north of Bangor.
Major access is by route 193,
which is frequently snowbound.
.
.
.
[EOB]
3L(RET)
```

You decide to move the cursor down 3 lines and enter the 3L command. The cursor moves to the command entry position. When you press (RET), it moves to the first character in the fourth line.

You then decide to shorten the file in order to work with it. Move the cursor to the beginning of the eighth line. You can either use the command 4L or press the down arrow function key four times. Delete all the following lines with the command D14L. The cursor is at the end of the buffer.

```
.
.
.
on nearby Lake Okichibwa.
[EOB]
4L(RET)
D14L(RET)
```


“Lake Okichibwa” should be spelled “Okechibwa.” Use the command sequence:

```
-1L (RET)
17C (RET)
```

This moves the cursor back (up) one line and to the right 17 characters. The minus sign is necessary in the first command to set the direction for the cursor movement. The cursor is now on the letter “i”:

```
on nearby Lake Okichibwa.
[EOB]
```

To delete the incorrect letter and insert the letter “e,” use the command sequence:

```
D1C (RET)
Ie (CTRL/Z) (RET)
```

EDT inserts the letter “e” in the correct place:

```
on nearby Lake Okechibwa.
```

The name of the lake is actually “Okechibwa Lake.” Move the cursor to the beginning of the word “Lake” with the command -8C. (Remember that the minus sign indicates movement to the left.)

```
on nearby Lake Okechibwa.
```

Delete the whole word with the command D1W. The word “Lake” disappears from the screen and is stored in a buffer. Move the cursor forward (right) 9 spaces with the command 9C:

```

      .
      .
      .
on nearby Okechibwa.
[EOB]
9C (RET)
```

Now you can reinsert the word "Lake" with the undelete word command:

```
      .  
      .  
      .  
on nearby OkechibwaLake .  
[EOB]  
  
UNDW(RET)
```

To enter a space between the two words, enter the command:

```
I (CTRL/Z) (RET)
```

Move the cursor to the space before the period and enter the delete character command. The line now stands corrected:

```
      .  
      .  
      .  
on nearby Okechibwa Lake .  
[EOB]  
  
D1C(RET)
```

If you decide to change the word "frequently" to "often" in the fourth line, you can use the SUBSTITUTE command:

```
s/frequently/often/(RET)
```

However, when you try this, you get an error message:

```
The town of Chilly, Me. is located 40  
miles north of Bangor.  
Major access is by route 193,  
which is frequently snowbound.  
The nearest airport is in Bangor,  
although a seaplane lands once a week  
on nearby Okechibwa Lake .  
[EOB]  
  
String was not found
```

To understand why this happened, look at the cursor position in the text area. The SUBSTITUTE command began a forward search when the cursor was near the end of the text and immediately reached the end of the buffer. Use the BACK command to reverse the search direction. The cursor is now at the beginning of the search string:

```
which is frequently snowbound.
```

Try the SUBSTITUTE command again.

This results in:

```
which is often snowbound.
```

EDT will continue reverse searching unless you negate the BACK command with ADV (ADVANCE). All searches will now be forward.

You now decide to perform one last edit on this text. You want to move the second sentence to the end of the buffer. Move the cursor to the beginning of the second sentence:

```
major access is by route 193,  
which is often snowbound.
```

Use the CUT command (CUT 2L) to cut the sentence and move it to an auxiliary buffer called the paste buffer:

```
CUT2L (RET)
```

Your text now reads:

```
The town of Chilly, Me. is located 40  
miles north of Bangor.  
The nearest airport is in Bangor,  
although a seaplane lands once a week  
on nearby Okechibwa Lake.
```

To reinsert the cut sentence at the end of the text, move the cursor to the end of the buffer and use the PASTE command:

```
The town of Chilly, Me. is located 40
miles north of Bangor.
The nearest airport is in Bangor,
although a seaPlane lands once a week
on nearby Okechibwa Lake.
Major access is by route 193,
which is often snowbound.
[EOB]
```

```
PASTE[RET]
```

To go back to the EDT command level, enter EX [RET].

These examples show a small portion of the nokeypad character editing commands. It is recommended that you experiment with nokeypad character editing, using the examples in Chapter 9.

2.7 Using Character Editing: Keypad Editing

The keypad is the set of keys at the right of the keyboard. You can edit with the keypad if you have a VT52 or VT100 terminal. There are diagrams of the VT52 and VT100 keypads in Chapter 10.

The function keys used in the examples are enclosed by broken lines; for example:

```
GOLD
```

2.7.1 How to Start Keypad Editing

Start EDT and create the text file COLD.DAT:

```
$EDIT/EDT COLD.DAT[RET]
Input file does not exist
[EOB]
*
```

Type the CHANGE command after the line editing prompt to start a keypad editing session:

```
*CHANGE[RET]
```

The screen clears and the cursor appears at the upper left of the screen. On the line below the cursor is the [EOB] symbol. Whatever you type at this point appears on the screen before the cursor. The [EOB] symbol always appears after the last character in the text buffer.

2.7.2 How to Get Help

You can get two kinds of help during keypad editing. First, you can use the HELP function to get a diagram of the default keypad functions. Press the HELP key for this diagram. (Requesting help this way does not affect your editing session.)

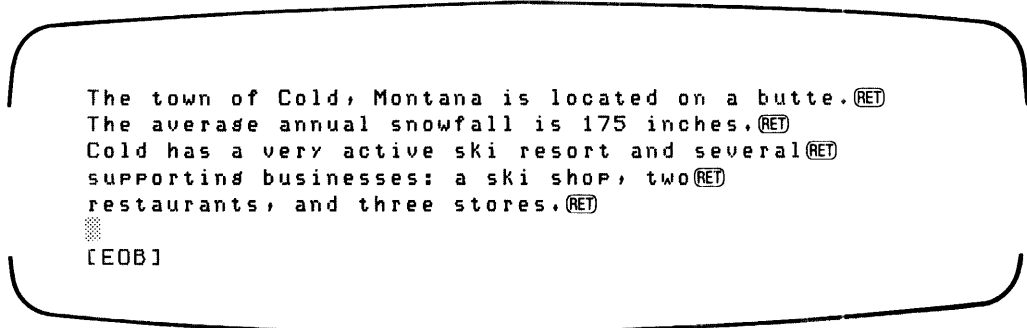
Second, once you have the diagram, you can press any keypad key to see a description of what the key does. If you press the key numbered 5 on the VT100 keypad, for example, a description of the BACKUP and TOP functions appears on the screen. (BACKUP and TOP are the functions associated with key 5.) At the bottom of each description are directions for seeing the keypad diagram again, reading about other functions, or getting out of help and returning to your editing session.

2.7.3 Keypad Functions

In keypad character editing the keypad keys have assigned functions which are performed when you press the key. The function is one or more of the nokeypad character editing commands. Most of the keypad keys have a standard and an alternate function. You access the standard function by pressing the key. The GOLD key causes EDT to perform the alternate function on any keypad key; you press and release the GOLD key and then press the desired keypad key. In text, a statement such as “Use the TOP function key” assumes that you use the GOLD key and then the TOP function key. The functions for the keypad keys are shown in Figures 10-1 and 10-2.

2.7.4 Inserting Text

You can insert text as soon as you start keypad editing. Whatever you type on the keyboard is inserted directly into the buffer. The following paragraph appears on your screen as you type it.



```
The town of Cold, Montana is located on a butte.(RET)
The average annual snowfall is 175 inches.(RET)
Cold has a very active ski resort and several(RET)
supporting businesses: a ski shop, two(RET)
restaurants, and three stores.(RET)
[EOB]
```

The cursor appears after the last character you enter.

2.7.5 Moving the Cursor

You can use the TOP function to move the cursor to the top of the text:

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.
[EOB]

GOLD + BACKUP
TOP

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.
[EOB]

You can also move the cursor to the bottom of the buffer with the BOTTOM function:

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, and three stores.
[EOB]

GOLD + ADVANCE
BOTTOM

The town of Cold, Montana is located on a butte. The average annual snowfall is 175 inches. Cold has a very active ski resort and several supporting businesses: a ski shop, two restaurants, and three stores.


[EOB]

The arrow keys on the keyboard let you move the cursor in each of four directions:

UP 


Supporting businesses: a ski shop, two restaurants, and three stores.

[EOB]

 (3 times)

Supporting businesses: a ski shop, two restaurants, and three stores.

[EOB]

DOWN 

Supporting businesses: a ski shop, two restaurants, and three stores.

[EOB]



SUPPORTING businesses: a ski shop, two
restaurants, and three stores.
[EOB]

RIGHT



SUPPORTING businesses: a ski shop, two
restaurants, and three stores.
[EOB]



(4 times)

SUPPORTING businesses: a ski shop, two
restaurants, and three stores.
[EOB]

LEFT



SUPPORTING businesses: a ski shop, two
restaurants, and three stores.
[EOB]



(2 times)

```
sUPPORTING businesses: a ski shop, two  
restaurants, and three stores.  
[EOB]
```

Some keys let you move text by character, word, and line entities. For example, the WORD function key moves the cursor to the next word:

```
sUPPORTING businesses: a ski shop, two  
restaurants, and three stores.  
[EOB]
```

```
WORD  
CHNGCASE
```

```
sUPPORTING businesses: a ski shop, two  
restaurants, and three stores.  
[EOB]
```

2.7.6 Marking Text with Select Ranges

This section shows how to use the SELECT, CUT, and PASTE functions. When you use the SELECT function, you mark some text for one of several keypad operations. The diagrams in this section show VT100 select ranges, since the reverse video feature makes select ranges obvious. The VT52 also has the select range function, but it does not have reverse video.

The following series of frames shows what happens when you mark a select range and delete the selected text with the CUT function. The text is then reinserted with the PASTE function.

1. Move the cursor to the start of the text that you want to cut. Press the SELECT function key, and then press the WORD function key three times:

```
SUPPORTING businesses: a ski shop, two
restaurants, and three stores.
[EOB]
```

SELECT
RESET + **WORD**
CHNGCASE (3 times)

```
SUPPORTING businesses: a ski shop, two
restaurants, and three stores.
[EOB]
```

If you make a mistake with the SELECT function key, press GOLD and the RESET function key, and then try again.

2. Press the CUT function key to delete the text:

```
SUPPORTING businesses: a ski shop, two
restaurants, and three stores.
[EOB]
```

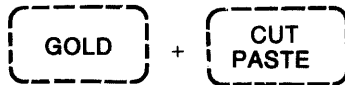
CUT
PASTE

```
SUPPORTING businesses: a ski shop, two
stores.
[EOB]
```

The text that you delete with the CUT function is stored in the paste buffer until the next CUT operation.

3. You can reinsert the deleted text with the PASTE function one or more times. To reinsert the text at the original location press the PASTE function key. Move the cursor to a second location and press the PASTE function key. The contents of the paste buffer is inserted at both locations.

```
SUPPORTING businesses: a ski shop, two
stores.
[EOB]
```



```
SUPPORTING businesses: a ski shop, two
restaurants, and three stores.
.
.
two restaurants, and three stores.
[EOB]
```

2.7.7 Deleting Words and Characters

You can delete words and characters several ways. For this example, press LEFT arrow function key six times to move the cursor. Then use the DELETE function to delete characters preceding the cursor:



To delete whole words that follow the cursor, use the DEL W function:



You can reinsert the last word you deleted with UND W:



2.7.8 Deleting Lines

There are three keypad functions (DEL L, DEL EOL, and UND L) that let you delete and undelete lines:

1. The DEL L function lets you delete a line, up to the end of the line plus the line terminator. (In this example, move the cursor to the beginning of the fourth line.) Press the DEL L function key. The text following the deleted line moves next to the cursor.

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, three stores.

[EOB]

DEL L
UND L

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
restaurants, three stores.

[EOB]

Undelete the line with UND L:

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
restaurants, three stores.

[EOB]

GOLD

+

DEL L
UND L

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, three
restaurants, two stores.

[EOB]

2. Delete all or part of a line with DEL EOL (DELe to End Of Line). This deletes all of the text from the cursor to the following RETURN. The text following the deletion remains in its former location.

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
supporting businesses: a ski shop, two
restaurants, three stores.

[EOB]

GOLD

+

EOL
DEL EOL

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
restaurants, three stores.

[EOB]

You can undelete the line with UND L:

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
■ restaurants, three stores.

[EOB]

GOLD

+

DEL L
UND L

The town of Cold, Montana is located on a butte.
The average annual snowfall is 175 inches.
Cold has a very active ski resort and several
■ supporting businesses: a ski shop, two
restaurants, three stores.

[EOB]


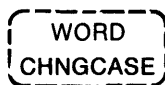
2.7.9 How to Change Case

You can change the case of text by individual characters or by whole strings. Move the cursor to the second letter of the word "Cold," and then change the case of one character at a time:

Cold  COld  COLd  COLD

If you mark a select range, everything within the range changes case. Move the cursor to the word "town." Press SELECT, then press the WORD function key four times; this gives you your select range. When you press the GOLD and CHNGCASE function keys, the characters you select change case.

The town of COLD, Montana is located on a butte.



 + 

The TOWN OF cold, mONTANA is located on a butte.

2.7.10 Exiting Keypad Character Editing

There are two ways to exit keypad character editing:

1. Enter **CTRL/Z**, which returns you to line mode editing. You can still type CHANGE after the asterisk prompt, which returns you to KEYPAD editing. The cursor will not have moved.
2. Use the COMMAND function and type EXIT (to save your edits in a file) or QUIT (to exit without saving your edits), followed by ENTER:

 + 

Command: EXIT 

Either of these commands lets you exit EDT.

Chapter 10 gives you more information about editing with keypad functions. It describes not only the basics presented here, but also many different ways of using each function.

Chapter 3

Protection Provided by EDT

This chapter describes the user-visible protection provided by the editor: the journal file, the consistency check, and error messages.

3.1 The Journal File

EDT records your input to the terminal in a journal file and saves the journal file under any of the following conditions:

- A system failure
- A user abort
- An EXIT command with the SAVE qualifier
- A QUIT command with the SAVE qualifier

EDT saves the journal file as filename.JOU, where filename is the output file name. When you do not specify the output file name, EDT uses the input file name by default.

You can use the journal file to recover edit operations (your inputs to the terminal). You repeat the command line, adding the RECOVER qualifier. The input file is loaded into the text buffer MAIN, and the journal file is executed by EDT as if it were input from the terminal.

You can edit the journal file, with changes or additions, by entering it as the input file in the command line. If you exit normally when you are through editing the journal file, the file is given a version number two above the input file version number. The intervening number is assigned to the journal file created during the editing of the input journal file. (See Chapter 6 for a description of the command line.)

In the following example, you are instructed to abort the editor during a session. This is not a normal practice and is used here only to demonstrate the operation of EDT when it is ended abnormally.

Examples:

```
$ EDIT/EDT(RET)
$_File: ANGLICIZE(RET)
Input file does not exist
* (Enter text using EDT and other EDT commands.)
.
.
.
(CTRL/Y)
$DIRECTORY(RET)
Directory _DB1:[YOUR NAME]
ANGLICIZE.JOU;1 n lines
.
.
.
.
Total of n files.
$
```

When you do not specify an input file in the command line, the operating system prompts for a file. You must enter a file name. When no file with that name exists, EDT responds with the 'input file not found' message. EDT is then ready for you to insert text.

Exit the edit session with a (CTRL/Y) to simulate a system failure. You then return to system command level; you can reenter EDT by entering a CONTINUE command. If you enter a system command that requires system execution, such as DIRECTORY, your link to EDT is ended. EDT saves the journal file, and your directory has the entry ANGLICIZE.JOU;1.

```
$ EDIT/EDT ANGLICIZE/RECOVER(RET)
*
```

Assume there is an existing file named CHAPTONE.RNO and you have performed a series of edit operations on the file before one of the four conditions for saving the journal file occurred. You now have two files:

1. CHAPTONE.RNO — unchanged by the edit session
2. CHAPTONE.JOU — which is the text you entered at the terminal from the time of the command line entry to the departure from the edit session

When you enter the command line, EDT loads the file CHAPTONE.RNO into the main text buffer and executes the file CHAPTONE.JOU as a command file. The result is displayed as it occurs. The last few entries during the preceding editing session may not be in the journal file.

```
$EDIT/EDT CHAPTONE.JOU(RET)
*
```

EDT loads the journal file into the main text buffer. You then edit the journal file. You can change the journal file to make corrections to the input from a previous editing session or to create a new output file. EDT starts a journal file on this edit session; the name for the journal file is CHAPTONE. The version number is one above the version number of the input file. When you exit this editing session, the output file is CHAPTONE and its version number is one above the highest existing version number.

3.2 Consistency Check

EDT also maintains a count of all lines and characters entered into the text buffers during an edit session.

When you exit the edit session, EDT makes a count of the contents of the text buffers and checks it for consistency with the input. When there is a discrepancy in the consistency check, EDT displays a warning message on exit from the session:

```
' Consistency check failed, please check your files
```

3.3 Error Messages

EDT contains a set of error messages to warn you when something that it cannot accept has occurred. Error messages can be a direct result of your input or a result of operations performed during the edit session. The error messages point to corrective action. Appendix B contains a list of the error messages.

Chapter 4

Line Numbers and Range Specifications

This chapter describes line numbering and the range specifications used in EDT.

4.1 Line Numbers

Line numbers are an editing tool that aid you in the manipulation of text. EDT assigns a number to each line in a text buffer. The numbers range from 0.00001 to 42949.67295. Some of the range specifications in the commands address these line numbers.

When you specify an input file, EDT assigns numbers to the lines which result from loading the text buffer with that file. The line numbers are from 1 to n in increments of 1. When n (the number of lines in the file) exceeds 42949, EDT renumbers the contents of the text buffer using decimal numbers.

EDT also has the capability to handle fixed line numbers. Fixed line numbers are numbers associated with records in a file; the line numbers are a part of the file. Fixed line numbers are associated with files which have variable length and an additional fixed-length control area. When the input file has fixed line numbers, EDT saves those numbers; this record of line numbers is not visible. EDT assigns the line number 0, in this record, to lines added to a text buffer with fixed line numbers. You can assign fixed numbers to a file through the `WRITE` command or the `EXIT` command by using the `SEQUENCE` qualifier.

You can renumber the lines in a text buffer with the `RESEQUENCE` command. If that text buffer had fixed line numbers, you can still address lines by their original line numbers (see “Single Line Ranges”).

4.2 Range Specifications

Range specifications define the lines upon which EDT commands operate. They specify the lines on which EDT performs the command. Ranges can be single lines or multiple lines. In addition, the multiple line ranges can be contiguous or noncontiguous. Ranges can be specified for text buffers other than the current text buffer.

4.2.1 Single Line Ranges

A single line range defines a single line in a text buffer for the command to operate on. The following ranges specify a single line in a text buffer:

<code>.</code> (period)	The current line. The line the cursor is on.
<code>number[.decimal]</code>	The line specified by the number.
<code>'string' ! "string"</code>	The next line containing the string you specify. If you do not define the search string (that is, you enter an empty quoted string), EDT uses the last search string.
<code>-'string' ! -"string"</code>	The most recent preceding line containing the string you specify. If you do not define the search string (that is, you enter an empty quoted string), EDT uses the last search string.
<code>[range] + [number]</code>	The line which is the specified number of lines after the specified range (where range is a single line range and number is an integer). The default range is the current line and the default number is one.
<code>[range] - [number]</code>	The line which is the specified number of lines before the specified range (where range is a single line range and number is an integer). The default range is the current line and the default number is one.
<code>BEGIN</code>	The first line in the text buffer.
<code>END</code>	An empty line following the last line of text in the text buffer.
<code>LAST</code>	The line position in the most recent text buffer before entering the current text buffer.
<code>ORIGINAL number</code>	The line numbers assigned to the text in text buffer MAIN from the primary input file, when the primary input file has fixed line numbers. You can locate the text by the original line number even after it has been assigned a new line number.

Examples: In the following examples, typical commands are used to show samples of range specifications.

```
*DELETE .(RET)
1 line deleted
```

EDT deletes the current line, the line where the cursor is located, and displays the message "1 line deleted."

```
*TYPE 12.11(RET)
12.11 This is a decimally numbered line.
*␣
```

EDT types the contents of the line.

```
*TYPE - 'PAYROLL' (RET)
129 The payroll records are file 72.a.
*'' (RET)
247 payrolls are made up weekly.
*␣
```

EDT displays the first line preceding the current line that contains the string 'payroll'. The next entry is an implied TYPE (empty string) command. EDT searches forward and displays the next occurrence of the string 'payroll'.

```
*INSERT .+12(RET)
```

EDT opens the text buffer 12 lines past the cursor location for the insertion of text.

```
*'ABC' + 3(RET)
27 This line is three lines later.
*␣
```

EDT searches forward for the first occurrence of the string 'ABC', counts forward three lines, and displays the line at that position.

```
*FIND LAST(RET)
*␣
```

EDT moves the cursor to the line last addressed in the text buffer that was used before the current text buffer. The line is not displayed.

```
*TYPE ORIGINAL 27(RET)
247 This is the line of interest
*␣
```

The input file contained fixed line numbers. During the edit session you have added text and resequenced the line numbers and you now want to change a line of text. You know the original line number but not its present line number. You enter the TYPE command with the original range specification and EDT displays the line of interest with its current line number.

4.2.2 Contiguous Line Ranges

Contiguous line ranges define sets of consecutive lines within the text buffer. Range specifiers for contiguous sets of lines are as follows:

[range-1] : [range-2] !	The set of lines from range-1 through range-2.
[range-1] THRU [range-2]	The default for either range entry is the current line. Range-1 and range-2 are any single line range specification.
[range] # number !	The specified number of lines beginning with the line specified by range (where range is any single line range specification). The default range is the current line.
[range] FOR number	
BEFORE	All lines preceding the current line in the current buffer.
REST	All lines after and including the current line.
WHOLE	The current text buffer.

Examples: In the following examples, typical commands are used to show samples of range specifications.

```
*DELETE 10 THRU 22(RET)
13 lines deleted
```

EDT deletes lines 10 through 22 from the current text buffer and displays a message indicating how many lines were deleted. The number of lines from 10 through 22 is determined by the number of lines you have inserted and deleted since the line numbers were assigned.

```
*DELETE BEFORE(RET)
12 lines deleted
```

EDT deletes all of the lines before the cursor position in the current text buffer and displays a message about the number of lines deleted.

4.2.3 Noncontiguous Range Specifiers

Noncontiguous range specifiers define multiple lines in a text buffer that are not necessarily adjacent to one another. Range specifiers for noncontiguous ranges are as follows:

[range, range, ...] !	All lines specified by each range; each range must be a single line.
[range AND range AND ...]	
[range] All 'string'	All lines in the range containing the specified string. If range is not used, the default is the entire text buffer.

Examples: In the following examples, typical commands are used to show samples of range specifications.

```
* MOVE 3,7,10,22,26 TO 60(RET)
*
```

EDT moves the specified lines to the location just before line 60 and renumbers them for that location.

```
* TYPE ALL 'occurrence'(RET)
```

EDT displays all lines in the the current text buffer that contain the string 'occurrence'.

4.2.4 Text Buffer Range Specifications

You can specify ranges in a buffer other than the current text buffer. Type the buffer name before the range specification.

[= buffer] [range] ; The default is the current text buffer. When you
[BUFFER buffer] [range] use a buffer without a range specification (null
 range), the default is the entire text buffer, and
 the cursor is placed at the first line in the text
 buffer.

Examples: In the following examples, typical commands are used to show samples of range specifications.

```
* TYPE =MAIN .(RET)
```

The period in the command is a current line range. EDT enters the text buffer MAIN at the cursor position (current line) when the text buffer was exited and displays the contents of the line at that position.

```
* TYPE BUFFER PASTE ALL 'EDITOR'(RET)
```

EDT displays all lines in the text buffer PASTE that contain the string 'editor'.

Chapter 5

Editor Buffers

This chapter describes the two types of buffers contained in EDT: text and other.

5.1 Text Buffers

EDT works on text stored in locations called buffers. When you enter EDT, it creates two text buffers: MAIN and PASTE.

EDT creates additional text buffers as you call for them.

5.1.1 Main Buffer

The name of the main text buffer is MAIN. EDT puts the contents of the input file into the text buffer MAIN. When you exit the editing session, EDT transfers the contents of the main text buffer to the output file.

5.1.2 Paste Buffer

The paste text buffer is named PASTE. The paste text buffer is empty at the beginning of each edit session and then stores text from cut and append operations. The contents of the paste text buffer are transferred to the current text buffer for paste operations. You can also use the paste text buffer for other edit operations (as a text buffer).

5.1.3 Additional Text Buffers

You can name and use additional text buffers as needed (for retaining parts of other buffers, for including other files). EDT creates the text buffer when you name it. You can give these text buffers any name other than MAIN or PASTE. The name consists of 1 to 30 alphanumeric characters with the first character alphabetic. To call or to name a text buffer, you enter =name or BUFFER name as a part of the range specification (see Chapter 4).

Examples:

```
*COPY 10 THRU 100 TO =TEMP(RET)
```

A copy of lines 10 through 100 of the current text buffer is written to a text buffer named TEMP. If TEMP does not exist, EDT creates it.

```
*INCLUDE NEWFILE.TYP BUFFER PASTE(RET)
```

A copy of the file NEWFILE.TYP is written to the text buffer PASTE.

```
*MOVE =HOLD 1 THRU 32 TO 88(RET)
```

EDT moves lines 1 through 32 of the text buffer HOLD to the location just ahead of line 88 in the current text buffer.

There is no restriction on the size of any single text buffer except the restriction placed on the set of all text buffers. The maximum size for the set of all text buffers in EDT is approximately 250,000 lines of 80 characters each. The maximum size can be limited by the space restrictions on the system device containing EDT's work space.

The buffer status is displayed through the SHOW BUFFER command (see Chapter 8). The SHOW BUFFER command displays the name of all the text buffers in use during the edit session, the number of lines they contain, and an equal sign (=) preceding the name of the current text buffer.

5.2 Other Buffers

EDT also contains buffers (storage locations) for use by editor commands. The contents of these buffers result from the command operation. These buffers include the following:

- Line
- Word
- Character
- String search
- Substitute

5.2.1 Line Buffer

The line buffer is a 256-character buffer which is empty at the beginning of each edit session. EDT loads the line buffer with text from character editing commands DL, DEL, DNL, and DBL (see Chapter 9). Loading the buffer erases the previous contents of the buffer. You copy the contents of the line buffer with an undelete line command. When you copy the line buffer, EDT inserts the contents of the line buffer just in front of the cursor position. Copying does not erase the contents of the line buffer.

5.2.2 Word Buffer

The word buffer is an 80-character buffer which is empty at the beginning of each edit session. EDT loads the word buffer with the text from character editing commands DW, DEW, DNW, and DBW (see Chapter 9). Loading the buffer erases the previous contents of the buffer. You copy the contents of the word buffer with each undelete word command. When you copy the word buffer, EDT inserts the contents of the word buffer just in front of the cursor position. Copying does not erase the contents of the word buffer.

5.2.3 Character Buffer

The character buffer is a one-character buffer which is empty at the beginning of each edit session. EDT loads the character buffer from each character editing command that deletes a character. Loading the buffer erases the previous contents of the buffer. You copy the contents of the character buffer with each undelete character command. When you copy the character buffer, EDT inserts the contents of the character buffer at the cursor position. Copying does not erase the contents of the character buffer.

5.2.4 String Search Buffer

The string search buffer is a 64-character buffer which is used for storing the object of a substitute or a string search operation. The search string buffer is empty at the beginning of each edit session. Loading the buffer erases the previous contents.

5.2.5 Substitute Buffer

The substitute buffer is a 64-character buffer which is used for storing the replace string in a substitute command. The substitute buffer is empty at the beginning of each edit session. Loading the buffer erases the previous contents.

Chapter 6

The Command Line and Command Files

This chapter describes the command line used for starting EDT and the command files for presetting EDT operating parameters.

6.1 EDT Command Line

You start EDT through the command line. If you type `Ⓡ` after EDIT/EDT without a file specification, you receive the File: prompt.

The format for the EDT command line is:

```
EDIT/EDT [/qualifiers] file-spec [/qualifiers]
```

where:

`EDIT/EDT` specifies the text editor as EDT on VAX/VMS.

`file-spec` specifies the file to be created or edited. The file is created if it does not already exist. EDT does not provide default file types. If you do not include a file type, it is null. The file must be a disk file on a files-11 formatted volume. Wild card characters are not allowed in the file specification.

`/qualifiers` specify command qualifiers. Qualifiers may be placed either before or after the file specification.

See the *VAX/VMS Command Language User's Guide* for details on file specification format.

6.2 Command Qualifiers

The EDT command qualifiers provide the system with additional information on how to handle EDT text during an editing session. Table 6-1 summarizes the qualifiers, which are then described in detail.

Table 6-1: EDT Command Qualifiers

Qualifier	Default
/[NO]OUTPUT[=file-spec]	/OUTPUT=file-spec
/[NO]COMMAND[=file-spec]	/COMMAND=EDTINI.EDT
/[NO]JOURNAL[=file-spec]	/JOURNAL
/[NO]RECOVER	/NORECOVER
/[NO]READ__ONLY	/NOREAD__ONLY

/OUTPUT=file-spec
/NOOUTPUT

The /OUTPUT qualifier defines the file specification of the output file created during the edit session. If you do not specify /OUTPUT, the output file has the same name and file type as the input file and a version number one higher than the highest existing version of the file.

/COMMAND[=file-spec]
/NOCOMMAND

The COMMAND qualifier controls whether or not EDT searches for and executes the named file before prompting you for input. If you do not specify /COMMAND, EDT searches your directory for and executes the contents of file EDTINI.EDT before prompting you for input (see Section 6.3, “Command Files”). If there is no file EDTINI.EDT or you specify /NOCOMMAND, EDT prompts you for input without modifying the default values for the operating parameters.

/JOURNAL[file-spec]
/NOJOURNAL

The JOURNAL qualifier controls whether or not the journal file is assigned a name other than the input file’s name with extension .JOU. If you specify /NOJOURNAL, EDT does not maintain a journal file for that edit session. (See Chapter 3.)

/RECOVER
/NORECOVER

The RECOVER qualifier controls whether the editor will execute the journal file created in the previous editing session. The rest of the command line must be the same as it was for the command line that started the editing session when the journal file was created. (See Chapter 3.)

```
/READ_ONLY
/NOREAD_ONLY
```

The `READ_ONLY` qualifier sets the `NOOUTPUT` and `NOJOURNAL` qualifiers. With `READ_ONLY`, if you attempt to exit the editing session without specifying an output file, an error occurs. Use the `READ_ONLY` qualifier when you do not have write access and want to examine the file without editing it.

Examples:

```
$ EDIT/EDT TEXT.DAT
```

EDT copies the contents of the input file `TEXT.DAT` into the main buffer and opens a journal file named `TEXT.JOU`. EDT assigns the name `TEXT.DAT` to the output file.

```
$ EDIT/EDT PAYROLL.FRM/OUTPUT=PAYROLL.JAN/JOURNAL=REC.ONE
```

EDT copies the contents of the file `PAYROLL.FRM` into the main buffer and opens a journal file named `REC.ONE`. When you end the edit session with the `EXIT` command, EDT copies the contents of the main text buffer to the file named `PAYROLL.JAN`. If the editing session ends in other than a normal exit (such as a system failure), EDT retains the journal file `REC.ONE`.

```
$ EDIT/EDT PAYROLL.FRM/OUTPUT=PAYROLL.JAN/RECOVER/JOURNAL=REC.ONE
```

EDT copies the contents of the file `PAYROLL.FRM` into the main buffer and then reads the contents of the journal file `REC.ONE` (the result of a previous editing session) as a command file. At the end of reading the journal file, EDT is in the same state as when the previous editing session ended. The text buffers contain the input files and your input to the terminal. The journal file contains all of your inputs to the terminal. The journal file continues to add to its contents during the recovery edit session. The editor assigns the name `PAYROLL.JAN` to the output file.

```
$ EDIT/EDT TEXT.DAT/OUTPUT=TEXTA.DAT/COMMAND=FORMC.DAT
```

EDT copies the contents of the file `TEXT.DAT` into the main buffer and then reads the contents of the command file. After the contents of the command file is executed, EDT prompts you for input. EDT assigns the name `TEXTA.DAT` to the output file.

6.3 Command Files

A command file contains instructions that can be read and executed by EDT. The name for the command file is `EDTINI` with a default filename string of `.EDT`. The file used may be changed by defining the logical name `EDTINI` to be the file specification of the desired file. EDT loads the input file into the

main buffer and performs the instructions in the command file before prompting you for input. When you do not specify a command file in your command line, the editor searches for the EDTINI.EDT file. If there is no EDTINI.EDT file, EDT is entered directly and you are prompted for input.

Examples:

```
TABLEA.COM  
  
    SET ENTITY PAGE '.PAGE'  
    SET WRAP 65  
    INCLUDE HEAD1.COM =HEAD1  
    INCLUDE HEAD5.COM =HEAD5  
    INCLUDE DATA.FIL =MAIN
```

When you specify the TABLEA.COM file as the command file in the command line, EDT executes the commands in the file and prompts you for input. With this command file, you have set the user-definable entity page to .PAGE and you have set the right margin for a word wrap of 65 characters. EDT loads the files HEAD1.COM and HEAD2.COM into text buffers named HEAD1 and HEAD2 respectively. EDT loads the file DATA.FIL into the main buffer just ahead of the file loaded into main buffer by the command line.

```
EDTINI.EDT  
    SET WRAP 65  
    SET MODE CHANGE
```

The EDTINI.EDT file is the default command file, so you do not have to name it in the command line. You have set the right margin for word wrap of 65 characters. EDT places you in the keypad change mode before prompting you for input.

Chapter 7

Line Editing Commands

The line editing commands operate on a range of lines defined through the range specification (see Chapter 4). You enter line editing commands from the EDT command level, indicated by the * prompt. In character editing, you enter line editing commands through the COMMAND command. The commands in this chapter are listed in alphabetical order.

The minimum abbreviation for these commands is shown in boldface type (in the line showing the form of the command). EDT accepts any input from the minimum abbreviation to the complete command word.

7.1 COPY Command

The COPY command copies text within editor text buffers. You copy text from one text buffer to another or from one location to another within a text buffer. The text is not deleted from the original location. (To copy text from an external file, use the INCLUDE command.)

The COPY command has the form:

```
COPY range-1 TO [range-2] [/QUERY] [/DUPLICATE:n]
```

EDT copies the set of lines you specify by range-1 to the location in front of the line specified by range-2. Range-2 is a single line range; the default for range-2 is the current line. Range-2 can be contained in range-1.

If the destination (range-2) is not in the current text buffer, you specify the name of the receiving text buffer (=buffer) immediately after TO. You must give the full name of the text buffer. When your source (range-1) is not the current text buffer, you name the source text buffer (=buffer) immediately before range-1.

When you use the QUERY qualifier, EDT prompts you for verification of each

line of the range in the COPY command. The prompt is a question mark (?). You have four valid responses to the QUERY prompt:

- Y (yes) Copy this line to range-2.
- N (no) Do not copy this line to range-2.
- A (all) Copy all remaining lines in range-1 to range-2.
- Q (quit) Quit the copy operation.

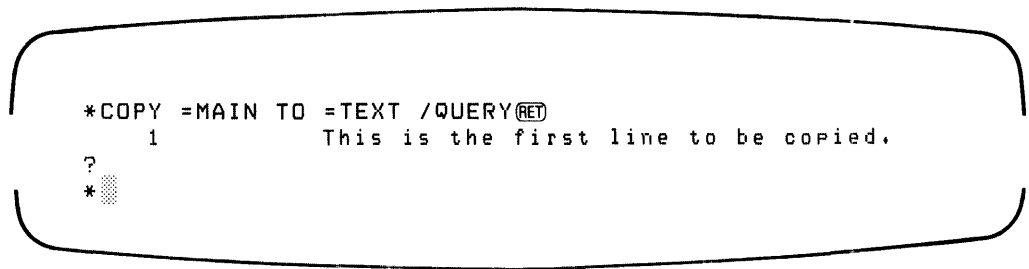
Use the DUPLICATE:n qualifier to make n copies of the source text (range-1).

The first line of range-2 becomes the first line after completion of the COPY command.

Examples:

```
*COPY 15 THRU 60 TO 95(RET)
```

EDT copies lines 15 through 60 of the current text buffer to the position just before line 95 of the current text buffer.



```
*COPY =MAIN TO =TEXT /QUERY(RET)
1 This is the first line to be copied.
?
*
```

EDT copies the contents of the text buffer MAIN to the buffer TEXT one line at a time. EDT shows each line to be copied followed by the ? prompt. You must respond with one of the prompt answers before EDT continues.

```
*COPY BEGIN TO 65 /DUPLICATE:9(RET)
```

EDT copies the first line of the current text buffer and places it just before line 65 of the current text buffer. This operation is repeated 9 times.

7.2 DEFINE MACRO Command

The DEFINE MACRO command assigns a name to a sequence of editor commands.

The DEFINE command has the form:

```
DEFINE MACRO macro-name
```

The DEFINE MACRO command assigns a name to a text buffer. You can enter the text buffer and insert any number of EDT commands. You end your

entries with a `CTRL/Z`. The macro-name is a line editing command and becomes a part of the EDT command list for the duration of this editing session. You can define the macro-name to be the same as an existing line editing command, but this results in that command no longer being available. For example, assume that you want to insert a block of text in a number of places in the text buffer. You create the macro containing the text and name the macro INSERT. Now when you enter INSERT as a command, EDT will execute the contents of the macro INSERT.

You enter the macro-name in response to the editor command level prompt. EDT makes a search of the EDT command list for every command you enter before it processes the command. When a macro and a command have the same name, EDT redefines the command as the macro.

You can have macros address other macros to a depth limited only by the available memory.

Examples:

```
10          This is a line in the text buffer.
*DEFINE MACRO FORM(RET)
*FIND=FORM(RET)
*INSERT(RET)
          SET TAB WRITE(RET)
          SET ENTITY SENTENCE ';' (RET)
          SET NOWRAP(RET)
          SET NOTRUNCATE(RET)
          FIND LAST+1(RET)
          INSERT; THIS IS A SAMPLE OF THE NEW PROGRAM(RET)
          CTRL/Z
*FIND=MAIN.(RET)
*(RET)
10          This is a line in the text buffer.
*
```

At line 10 of the current text buffer you determine a need for a macro. The DEFINE MACRO command defines FORM to be a valid EDT command for the duration of this edit session. A text buffer is assigned the name FORM. You transfer EDT to the text buffer FORM with the FIND command and you enter the command string into the text buffer FORM. At the completion of the command entries, you exit the insert mode with a `CTRL/Z`. The FIND=MAIN. command returns you to your position in the text buffer prior to the DEFINE MACRO command. Enter a `RET` and EDT displays the current line, line 10.

When you enter FORM in response to the command level prompt, the commands in the text buffer FORM are executed:

The first tab indent is set to 4.

The delimiter for the sentence entity is set to ;.

The NOWRAP parameter is set.

The NOTRUNCATE parameter is set.

EDT locates the last line previous to entering the FORM command and enters the message

```
'THIS IS A SAMPLE OF THE NEW PROGRAM'
```

on the next line.

7.3 DELETE Command

The DELETE command deletes lines of text specified by the range.

The DELETE command has the form:

```
DELETE [range] [/QUERY]
```

When you do not specify the range, EDT defaults to the current line. When you delete a range of lines, EDT deletes the lines and displays a message stating the number of lines deleted. The message is followed by a display of the next line in the text buffer.

When you use the QUERY qualifier, EDT prompts you for verification of each line of the range in the DELETE command. The prompt is a question mark (?). You have four valid responses to the QUERY prompt:

Y (yes) Delete this line.

N (no) Do not delete this line.

A (all) Delete all remaining lines in the specified range.

Q (quit) Quit the delete operation.

Examples:

```
*DELETE(RET)
1 line deleted
  21          The line preceding this line was deleted.
*  
*
```

EDT deletes the line at the cursor position and displays the contents of the next line, line 21.

```
*DELETE 25(RET)
1 line deleted
 26          There are no lines between line 25 and 26.
*█
```

EDT deletes line 25 of the current buffer and displays the contents of the next line, line 26.

```
*DELETE BEFORE(RET)
10 lines deleted
 11          There are no decimal numbers in the range deleted.
*█
```

EDT deletes all lines preceding the current line and displays the contents of the current line, line 11.

```
*DELETE 12 THRU 42 /QUERY(RET)
 12          This is the first line of the range.
? N(RET)
 13          This is the second line of the range.
? Y(RET)
1 lines deleted
 14          This is the third line of the range.
? A(RET)
29 lines deleted
 43          This line is not in the range.
*█
```

EDT displays each line of the range and the ? prompt.

You must enter one of the prompt QUERY responses before EDT continues to the next line or exits the delete operation.

7.4 FIND Command

The FIND command locates the line specified by range. EDT does not display the line located.

The FIND command has the form:

```
FIND range
```


Range is a single line range. Use the FIND command to move between text buffers. EDT locates the selected range and positions the cursor at the start of that line. EDT then displays the command prompt.

Examples:

```
*FIND 45 (RET)
*
```

EDT finds line 45 of the current text buffer. EDT displays the * prompt when the line is found.

```
*FIND =STORE 45 (RET)
*
```

EDT finds line 45 of the text buffer STORE. EDT displays the * prompt when the line is found.

7.5 INCLUDE Command

The INCLUDE command copies files into text buffers. Lines added through the INCLUDE command are numbered by EDT to ensure that their position immediately precedes the specified range.

The INCLUDE command has the form:

```
INCLUDE file name [range]
```

The file name is the name of the file you want to copy.

EDT copies the specified file to the current text buffer immediately in front of the first line of [range]. The transfer does not modify the source file. When the INCLUDE operation is complete, EDT displays the command prompt.

Examples:

```
*INCLUDE TEXT.DAT (RET)
*
```

EDT copies the contents of the file named TEXT.DAT into the current text buffer just in front of the present line.

```
*INCLUDE TEXT.FIL =TEXT (RET)
*
```

EDT copies the contents of the file named TEXT.FIL into the buffer you have named TEXT. EDT copies the text in front of any text in the text buffer.

```
*INCLUDE FILE.TEN =ELEVEN 60 (RET)
*
```

EDT copies the contents of the file named FILE.TEN into the text buffer you have named ELEVEN. EDT loads the text into the text buffer ELEVEN just in front of line 60.

7.6 INSERT Command

The INSERT command opens a text buffer for the insertion of lines of text. The lines added through the INSERT command are numbered by EDT to ensure that their position immediately precedes the specified range.

The INSERT command has the form:

```
INSERT [range]
```

To exit the INSERT command, enter `CTRL/Z`.

When you use the range qualifier, EDT inserts the text you enter before the first line of the specified range. Range is a single line range. When you do not use the range qualifier, EDT inserts the text you enter just before the current line.

When you end the INSERT command with a semicolon, you can use the rest of the command line (until you press `RET`) to insert text. EDT inserts the text at the specified range and exits the insert operation. When you do not use the semicolon, the insert operation continues until you enter a `CTRL/Z`.

Examples:

```
*INSERT 12 RET
*
```

EDT opens the current text buffer, just before line 12, for the insertion of text. Enter as many lines of text as you wish. EDT assigns decimal line numbers to all lines entered. EDT remains in the insert mode until you enter a `CTRL/Z`.

```
*INSERT =TEXT 30;ENTER NEW EMPLOYEE NUMBERS RET
*
```

EDT opens your buffer TEXT, inserts "ENTER NEW EMPLOYEE NUMBERS" just before line 30, and then exits the insert mode of operation.

7.7 MOVE Command

The MOVE command moves the lines defined by one range to the location preceding the line specified by another range. The copied lines are numbered by EDT to ensure that their position immediately precedes the specified range.

When you use the MOVE command, EDT deletes the original copy of the text. Use the COPY command to copy without deleting the original text.

The MOVE command has the form:

```
MOVE [range-1] TO [range-2] /QUERY
```

Range-1 is the text you wish to move. Range-2 is the destination for the text of range-1 and becomes the current line on completion of the move. If you omit either range-1 or range-2, the default for the omitted range is the current line. When the move is complete, EDT displays a command line prompt.

When you use the QUERY qualifier, EDT prompts you for verification of each line of range-1 in the MOVE command. The prompt is a question mark (?). You have four valid responses to the QUERY prompt:

- Y (yes) Move this line to range-2.
- N (no) Do not move this line to range-2.
- A (all) Move all of the remaining lines in range-1 to range-2.
- Q (quit) Quit the operation.

Examples:

```
*MOVE TO 65(RET)
*  
.
```

EDT copies the current line to line 65 of the current buffer and deletes the current line.

```
*MOVE =TEXT WHOLE TO 47(RET)
*  
.
```

EDT copies the contents of your text buffer TEXT to the position just in front of line 47 of the current text buffer and then deletes the contents of text buffer TEXT. EDT assigns new line numbers to the moved lines.

```
*MOVE 24 THRU 88 TO =MAIN 60(RET)
*  
.
```

EDT copies lines 24 through 88 of the current text buffer to the point just in front of line 60 in the text buffer MAIN and then deletes those lines in the current buffer. EDT assigns new line numbers to the moved lines.

7.8 Null Command

The null command is the implied TYPE command. The null command includes a range specification followed by a (RET). The range specifications REST, WHOLE, BEGIN, END, LAST, and ALL are not allowed in this statement unless they are preceded by a % (see Chapter 4).

Examples:

```
*-(RET)
```

The minus sign (-) reverses the direction so that EDT displays the line preceding the current line.

```
*15 THRU 60(RET)
```

EDT displays each line in the range in sequence.

*%REST **(RET)**

EDT displays the rest of the text in the current text buffer.

7.9 PRINT Command

Use the PRINT command to copy text from a text buffer into a file. EDT puts the transferred text into a page printable format.

The PRINT command has the form:

PRINT file name [range]

Use the range to select a portion of the text buffer to be copied to the file in page printable format. When you do not make a range entry, the default is the current text buffer.

When the transfer is complete, EDT returns you to your last position in the current text buffer and displays the command level prompt.

Examples:

```
*PRINT YOUR.FIL (RET)
*
```

EDT copies the contents of the current text buffer to a file named YOUR.FIL. The file is in a page printable format.

```
*PRINT THIS.FIL=TEXT 1 THRU 86 (RET)
*
```

EDT copies lines 1 through 86 of your text buffer named TEXT to a file named THIS.FIL. The file is in a page printable format.

7.10 REPLACE Command

Use the REPLACE command to delete the lines specified by range and to add new text. When the lines are deleted, EDT displays a message stating the number of lines deleted and then enters insert mode. The text buffer is opened at the first line in the range specification for the insertion of new text.

The REPLACE command has the form:

REPLACE [range]

To exit the REPLACE command, enter **(CTRL/Z)**.

When you do not specify a range, EDT deletes the current line and opens the text buffer for entering the new text in that location. EDT renumbers the inserted text, using the numbers of the deleted lines. When the text you insert exceeds the line count of the deleted text, EDT assigns decimal line numbers to ensure that the new line numbers do not conflict with line numbers outside the specified range. When there are more deleted lines than inserted lines, the excess line numbers are not used.

To replace the specified range a single line of text, you end the REPLACE command with a semicolon and enter the new text on the same line. When you press **(RET)** to end the line, EDT displays the command line prompt.

To return to EDT command level when you do not use the semicolon, you enter a **(CTRL/Z)**. The editor places you at the line following the last line deleted.

Examples:

```
*REPLACE 3 THRU 12(RET)
10 lines deleted
█
```

EDT deletes lines 3 through 12 of the current text buffer and opens the text buffer at line 3 for the insertion of text. When you are through inserting text, enter a **(CTRL/Z)**.

```
*REPLACE 9; Delete and replace this line.(RET)
10          This is the line following the replaced line.
*█
```

EDT deletes line 9 and inserts the text between the semicolon and the **(RET)** as the new line 9. EDT displays the contents of the next line, your present position in the current text buffer.

7.11 RESEQUENCE Command

The RESEQUENCE command assigns new line numbers to the contents of a buffer.

The RESEQUENCE command has the form:

```
RESEQUENCE [range] [/SEQUENCE [:initial:increment] ]
```

The lines you specify for resequencing by the range specification must be contiguous. When you do not specify a range, EDT resequences all lines in the current text buffer. When you use the sequence qualifier, EDT sets the first line to the initial value and increments the succeeding line numbers with the value of increment. When the lines are resequenced, EDT displays the command line prompt.

If you do not use the SEQUENCE qualifier, EDT sets the increment to 1, and the initial number is the first line of the specified range.

EDT prevents the assignment of nonsequential or duplicate line numbers.

Examples:

```
*RESEQUENCE 22 THRU END␣  
*
```

EDT resequences the line numbers of the current buffer starting with line 22. EDT increments the line numbers by a count of 1.

```
*RESEQUENCE =TEXT /SEQUENCE:10:10␣  
*
```

EDT resequences the line numbers of the contents of your buffer named TEXT. The first line number is 10, and EDT increments the line numbers by 10.

7.12 SUBSTITUTE Command

The SUBSTITUTE command replaces occurrences of one specified string with another string. A string consists of from 0 to 64 characters.

The SUBSTITUTE command has the form:

```
SUBSTITUTE /string-1/string-2 [/range] [/BRIEF[:n]][/QUERY] [/NOTYPE]
```

Any nonalphanumeric character, such as slash (/), can be used as a string delimiter, provided the same delimiter is used in all places. However, the delimiter character cannot be contained in string-1 or string-2.

When you do not specify a range, EDT replaces the first occurrence in the current line of string-1 in the current line with string-2. When you specify a range, EDT replaces all the occurrences of string-1 within the range with string-2. EDT displays the line after the substitution is made. EDT returns you to the first line in the specified range at the end of the substitution.

When you use the BRIEF:n qualifier, EDT displays the first n characters of the lines containing string-1.

When you use the QUERY qualifier, EDT precedes each substitution with a question mark (?) prompt. The response qualifiers are:

- Y (yes) Substitute this string.
- N (no) Do not substitute this string.
- A (all) Make all of the remaining substitutions without further queries.
- Q (quit) Quit the substitution operation.

When you use the NOTYPE qualifier, the lines where EDT makes the substitutions are not displayed.

Examples:

```
*SUBSTITUTE /USED/PREOWNED/WHOLE/QUERY(RET)
```

EDT searches the current buffer for the word "USED". For each occurrence, EDT prints the line containing "USED", followed by a question mark (?). You read the line to determine the validity of making the substitution. The string delimiter used in this example is the same as the qualifier delimiter.

```
22          This is a command used for substitutions.
? N (RET)
46          It can be used for a range of lines.
? N (RET)
No substitutions
*(RET)
1          This is the current line.
*
```

After you respond with one of the four answers to the ? (in this case no), EDT continues searching the text buffer for occurrences. If no substitutions are made, EDT places you at the second line of the text buffer.

When you respond Y, EDT replaces "USED" with "PREOWNED". If there is more than one occurrence of "USED" in the line, the process is repeated. On each occurrence of "USED" in the line, EDT displays the corrected line and, after the last substitution, the number of substitutions made. After the last substitution is made EDT places you at the second line in the text buffer.

```
20          The word list appears once in this sentence.
*SUBSTITUTE #LIST#TABLE#/BRIEF:20(RET)
20          The word table appea
1 substitution made
21          This is the next line in the buffer.
*
```

EDT searches the current line for the word "LIST", replaces it with "TABLE" and displays the first 20 characters of the corrected line and the number of substitutions made. EDT then displays the next line in the current buffer and the command line prompt. The string delimiter used differs from the qualifier delimiter.

```
*SUBSTITUTE @SECTION@CHAPTER@WHOLE/QUERY/NOTYPE(RET)
*
```

EDT searches the current buffer for the word "SECTION" and then displays in sequence each line containing the word "SECTION", followed by a question mark (?). When you respond with one of the four QUERY qualifiers, EDT continues. When your response is a Y, the editor replaces "SECTION" with "CHAPTER" and searches for the next occurrence of "SECTION". The string delimiter used differs from the qualifier delimiter.

7.13 SUBSTITUTE NEXT Command

The SUBSTITUTE NEXT command replaces the next occurrence of a specified string with another string. The cursor remains at the line where the substitution is made.

The SUBSTITUTE NEXT command has the form:

```
[SUBSTITUTE] NEXT [/string-1/string-2/]
```

EDT searches for the next occurrence of string-1 from the current location forward. When EDT makes a substitution, the line in which the substitution is made becomes the current line.

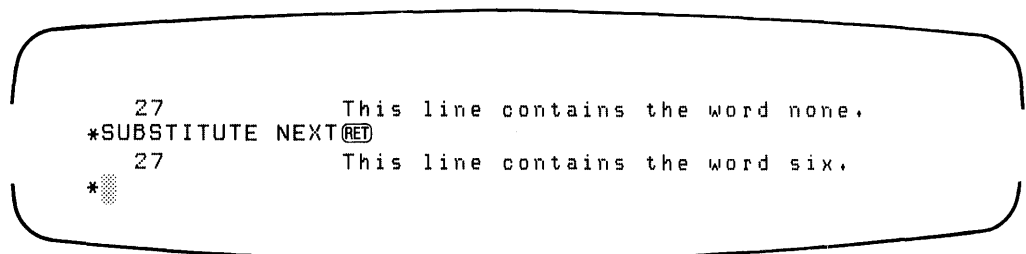
If neither string-1 nor string-2 is specified, EDT uses the strings specified by your last SUBSTITUTE command.

Examples:

Assume that your last substitute command was

```
SUBSTITUTE/NONE/SIX/
```

You wish to find the next occurrence of the word "NONE" to replace it with "SIX".



```
27          This line contains the word none.  
*SUBSTITUTE NEXT(RET)  
27          This line contains the word six.  
*
```

EDT searches for the next occurrence of your last SUBSTITUTE command string-1 and replaces it with string-2. EDT replaces the word "NONE" with "SIX" and displays the corrected line.


```

*NEXT/REASON/EMOTION/(RET)
  43          This line now contains the word emotion.
*

```

EDT searches for the next occurrence of "REASON" and replaces it with "EMOTION". EDT displays the corrected line.

7.14 TYPE Command

The TYPE command displays a specified range of lines.

The TYPE command has the form:

```
TYPE [range] [/BRIEF[:n]] [/STAY]
```

The range you select can be a single line or multiple lines. The multiple lines do not have to be contiguous. The first line you specified in the range becomes the current line.

When you select the BRIEF qualifier, EDT displays the first n characters of the selected lines. The default for n is 10 characters.

When you select the STAY qualifier, EDT does not change the cursor position.

Examples:

```

  30          There are two lines left in the current buffer.
*TYPE REST(RET)
  30          There are two lines left in the current buffer.
  31          This is the last line.
[EOB]
*

```

EDT displays the remainder of the current text buffer from the current line. The first line displayed is the current line.

```

*TYPE 27(RET)
  27          Line 27 is for display.
*

```

EDT displays line 27, the current line.

```

*TYPE 3,5,6,9,24(RET)
  3      This is line 3.
  5      This is line 5.
  6      This is line 6.
  9      This is line 9.
 24     This is line 24.
*

```

EDT displays lines 3, 5, 6, 9, and 24. Line 3 is the current line.

```

*TYPE 26#3(RET)
 26     This is line 26.
 27     This is line 27.
 28     This is line 28.
 29     This is line 29.
*

```

EDT displays lines 26 and the following three lines.

```

*TYPE 40:END ALL 'NEWS'(RET)
 41     This is the first line after 40 containing news.
 44     This is the last line after 40 containing news.
*

```

EDT displays all lines, beginning with line 40, that contain the string 'NEWS'.

```

*TYPE =TEXT 'JOURNAL'(RET)
 33     Line 33 contains the word Journal.
*

```

EDT searches your buffer TEXT for the word "JOURNAL" and then displays the first line containing "JOURNAL".

```

    32          This is the current line.
*TYPE 1 THRU 15 /STAY(RET)
    1          This is line one.
    .
    .
    .
    15          This is line fifteen.
*

```

EDT displays lines 1 through 15 of the current text buffer. The cursor remains at line 32.

7.15 WRITE Command

The WRITE command copies the defined range of text from a text buffer to the specified file. The WRITE command does not change the contents of the text buffer.

The WRITE command has the form:

```
WRITE file name [range] [/SEQUENCE[:initial:increment]]
```

When you specify a range, EDT copies that text to the file. When you do not specify a range, EDT copies the contents of the current text buffer to the file.

When you use the SEQUENCE qualifier, EDT writes the line numbers, as defined by initial and increment, as a fixed field in the file. These fixed line numbers are part of the file. The default for line numbers are the fixed line numbers from the input file.

Examples:

```

*WRITE PART.FIL 10 THRU 90(RET)
DB1:[USERNAME]PART.FIL;1 81 LINES
*

```

EDT transfers a copy of the contents of lines 10 through 90 to the file PART.FIL. When the transfer is complete, EDT displays the complete name of the new file, the number of lines written to the file, and the command prompt.

```
*WRITE NEW.DAT =TEXT1/SEQUENCE:10:10␣  
DB1:[USERNAME]NEW.DAT;1 145 LINES  
*  
█
```

EDT transfers a copy of the text buffer TEXT1 to your file NEW.DAT. EDT rennumbers the lines (beginning with line 10 and incremented by 10) and places them in a fixed field in the file. When the transfer is complete, EDT displays the complete name of the new file, the number of lines written to the file, and the command prompt.

Chapter 8

Editor Control Commands

Editor control commands are commands that control EDT rather than operate on text buffers. The commands in this chapter are listed in alphabetical order. The minimum abbreviation for these commands is shown in boldface type (in the line showing the form of the command). EDT accepts any input from the minimum abbreviation to the complete command word.

8.1 CHANGE Command

The **CHANGE** command places the editor in character editing mode. You select either keypad or nokeypad character editing through the **SET** command (one of them is always set). EDT defaults to keypad character editing for VT52 and VT100 terminals and to nokeypad character editing for all other terminals.

The **CHANGE** command has the form:

```
CHANGE [range]
```

The range specification in the command line is a single line range that defines where the buffer is opened (the line at which the cursor is positioned) upon entry into character editing. To exit character editing mode, you enter **CTRL/Z** in keypad and **EX** in nokeypad editing.

EDT accepts a set of commands for character editing. The full set of character editing commands is available in nokeypad and a portion of the character editing commands are available under keypad. In keypad character editing, you enter the commands through the keypad. In nokeypad character editing, you enter the commands through the keyboard. See Chapter 9 for details on nokeypad character editing and Chapter 10 for details on keypad character editing.

Your terminal type controls the visual response you receive during character editing.

On hardcopy (HCPY) terminals or terminals other than the VT52 or VT100, EDT prints the edited line after each `(RET)`.

On video terminals operating in nokeypad character editing, the subcommands are accepted at the last line on the screen. EDT updates the screen after each `(RET)`.

For video terminals using keypad character editing, the subcommand operations are immediately visible to the user; the portion of the display modified by the subcommand is updated.

For additional detail on terminal responses, see Appendix A.

Keypad character editing places a substantially larger demand on system resources than line editing does.

For those applications using slow speed input/output devices (300 baud or less), you should use a smaller display window. Use `SET LINES` to decrease the number of lines in the window, `SET CURSOR` to limit the cursor movement over a smaller range of lines, and `SET SCREEN` to limit the length of the lines displayed.

Examples:

```
*SET NOKEYPAD (RET)
*CHANGE (RET)
```

EDT is placed in nokeypad character editing mode. For VT52 and VT100 terminals, the setting of the other `SET` command parameters controls the screen display presentation. For all other terminals, the display is the line where the cursor is located followed by the character editing command prompt.

```
*CHANGE 43 (RET)
```

EDT enters the default character editing mode (keypad for VT52 and VT100 terminals and nokeypad for other terminals) at line 43 of the current text buffer.

8.2 EXIT Command

The `EXIT` command ends an editing session. EDT transfers the contents of text buffer `MAIN` to the file you specify.

The `EXIT` command has the form:

```
EXIT [file name] [/SEQUENCE[:initial:increment]] [/SAVE]
```

You define the file name for the contents of the buffer `MAIN` in either the command line or the `EXIT` command. When you define the file name in the `EXIT` command, that name takes precedence.

When you use the `SEQUENCE` qualifier, EDT assigns integer line numbers before the text transfer and places them in a fixed field in the file. These are

fixed line numbers and are a part of the file. You define the starting line number through the initial qualifier and the increment between line numbers through the increment qualifier.

The **SAVE** qualifier saves the journal file. The journal file is file name.JOU, where the file name is specified in the command line on the output file name. For information on the journal file, see Chapter 3.

Examples:

```
*EXIT (RET)
```

EDT transfers the contents of the text buffer **MAIN** to the file you defined in your command line. EDT returns you to system command level.

```
*EXIT NAME.NEW/SEQUENCE:5:5/SAVE (RET)
```

EDT transfers the contents of the text buffer **MAIN**, with line numbers, to your file **NAME.NEW**. The number of the first line in the file is 5 and each of the following lines is incremented by 5. The editor saves the journal file.

8.3 HELP Command

The **HELP** command displays information on requested topics.

The **HELP** command has the following form:

```
HELP topic subtopic
```

A topic or subtopic can have the following format:

1. An alphanumeric string (for example, a command name or a qualifier)
2. The wild card or match-all symbol (*)

To obtain a list of the valid topics, enter **HELP**.

The subtopics available on a topic are listed at the end of each topic.

Examples:

```
*HELP SUBSTITUTE NEXT (RET)
```

EDT displays the **HELP** text for **SUBSTITUTE NEXT** command.

```
*HELP CHANGE SUBCOMMAND (RET)
```

EDT displays the **HELP** text for **CHANGE** subcommands.

```
*HELP * (RET)
```

EDT scrolls through all help text alphabetically. If you enter **(CTRL/S)**, EDT stops scrolling. Enter **(CTRL/Q)** to restart scrolling.

When you want help on the keypad commands in keypad character editing, press the HELP function key. To exit the keypad help files, press the space bar or any of the keyboard keys. See Figures 10-1 and 10-2 for keypad layouts.

8.4 QUIT Command

The QUIT command ends the current editing session without saving the contents of text buffer MAIN.

The QUIT command has the following form:

```
QUIT [/SAVE]
```

You use the SAVE qualifier to save the journal file. For more information about the journal file, see Chapter 3.

Examples:

```
*QUIT (RET)
```

EDT returns you to system command level and saves nothing from the current session.

```
*QUIT/SAVE (RET)
```

EDT saves the contents of the journal file under the name you defined in the command line. EDT returns you to system command level.

8.5 SET Command

The SET command sets editor operating conditions. Once set, these conditions remain in effect until you exit the editor or until you change them through another SET command.

The SET command has the following form:

```
SET  { [ CASE { UPPER | LOWER | NONE } ] ;  
      [ CURSOR top:bottom ] ;  
      [ ENTITY { WORD | SENTENCE | PARAGRAPH | PAGE } 'string' ] ;  
      [ KEYPAD | NOKEYPAD ] ;  
      [ LINES number ] ;  
      [ MODE { LINE | CHANGE } ] ;  
      [ NUMBERS | NONUMBERS ] ;  
      [ QUIET | NOQUIET ] ;  
      [ SCREEN width ] ;
```

(Continued on next page)

```

[ SEARCH { BEGIN | END } ] ;
[ SEARCH { BOUNDED | UNBOUNDED } ] ;
[ SEARCH { GENERAL | EXACT } ] ;
[ TAB n | NOTAB ] ;
[ TERMINAL { HCPY | VT52 | VT100 } ] ;
[ TRUNCATE | NOTRUNCATE ] ;
[ VERIFY | NOVERIFY ] ;
[ WRAP n | NOWRAP ] ;

```

8.5.1 SET CASE

When you SET CASE (UPPER or LOWER), EDT flags the selected case characters with a preceding apostrophe. The flag is displayed on the terminal, but it is not transferred to the file when you exit EDT. The EDT default for CASE is NONE, which does not flag any characters. The primary use for this qualifier is on terminals with single case.

Example:

```
* SET CASE UPPER(RET)
```

With this command EDT marks all uppercase characters with a preceding apostrophe. The display on a terminal with upper and lower case for the first part of the preceding paragraph is as follows:

```
When you 'S'E'T 'C'A'S'E ('U'P'P'E'R or 'L'O'W'E'R) ...
```

For terminals with single case, the same copy reads as follows:

```
WHEN YOU 'S'E'T 'C'A'S'E ('U'P'P'E'R OR 'L'O'W'E'R) ...
```

8.5.2 SET CURSOR Top:bottom

SET CURSOR is applicable to screen terminals and character editing only. The CURSOR top:bottom parameter sets the number of lines over which the cursor moves on the display, where top is the number of lines for the upper limit and bottom is the number of lines for the lower limit. The top and bottom counts start from the top line on the screen, which is number 0.

When you reach either limit, EDT scrolls the text so you can access the desired line(s). The allowable limits for top and bottom are 0 through 21. Do not set top greater than bottom. The EDT default for this parameter is top=7 and bottom=14. On initial text entry into a text buffer, you use the lines up to and including the value set for top.

Example:

```

.
.
.
*SET CURSOR 10:12(RET)
*CHANGE(RET)

```

```

Line 0
Line 1
Line 2
Line 3
Line 4
Line 5
Line 6
Line 7
Line 8
Line 9
Line 10 This is the top line of the cursor range on the screen.
Line 11 This line is in the cursor range.
Line 12 This is the bottom line of the cursor range.
.
.
.

```

The 11th line from the top is set as the first line area and the 13th line is the last of the edit area. When you try to move the cursor to a position outside of these lines, EDT scrolls the display up or down until the line is within the cursor range.

8.5.3 SET ENTITY

The SET ENTITY command sets the user definable entities for character editing (WORD, SENTENCE, PARAGRAPH, and PAGE). Each of the entities is defined by a string of characters. For the WORD and SENTENCE entities, each character in the defining string can be a delimiter. For the PAGE and PARAGRAPH entities, the entire defining string is the delimiter.

The default values for the entity delimiters are:

```

WORD          (LF) or (TAB) or (FF) or (RET)
SENTENCE      . or ? or !
PARAGRAPH     (RET) (RET)
PAGE          (FF)

```

Example:

```
*SET ENTITY WORD ' ',';'(RET)
*SET ENTITY SENTENCE ';' (RET)
*CHANGE(RET)
```

The delimiters for **WORD** are space or comma. All the commands using word as an entity work up to the delimiter. When the delimiter, space, is on the right, the delimiter is included in the operation. When the delimiter, space, is on the left, the delimiter is excluded from the operation. All other word delimiters are treated as words. When a delete word command is positioned such that the first character to be deleted is a comma, EDT will delete the comma.

The delimiter for **SENTENCE** is semicolon. All commands using a sentence as an entity work up to the delimiter. The semicolon delimiter is useful in some programming languages.

8.5.4 SET KEYPAD

When the **KEYPAD** parameter is set, the keypad controls the character editing operations. When you set the **NOKEYPAD** parameter, the keyboard controls the character editing operations. The default for this parameter, for VT52 and VT100 terminals, is **KEYPAD** (see Chapter 10).

8.5.5 SET LINES Number

The **SET LINES** number command sets the number of lines that EDT displays on your terminal for character editing. The number of lines can be set from 1 to 22. When you are editing at slow baud rates (terminal speeds), you can use this parameter to reduce the number of lines. The default for this parameter is 22.

Example:

```
*SET LINES 8(RET)
*CHANGE(RET)
```

EDT displays 8 lines at the top of the screen.

8.5.6 SET MODE

The **SET MODE** parameter is used in the startup command file to control the editing mode which is entered at the end of the initialization process.

8.5.6.1 SET MODE CHANGE — The **SET MODE CHANGE** command causes EDT to begin a session in change mode; you do not have to enter the **CHANGE** command.

8.5.6.2 SET MODE LINE — The SET MODE LINE command causes EDT to begin an editing session in line mode; this is the default mode.

8.5.7 SET [NO]NUMBERS

The SET NONUMBERS command causes EDT not to display the line numbers. The EDT default for this parameter is NUMBERS, which causes the line numbers to be displayed.

8.5.8 SET [NO]QUIET

The SET QUIET command suppresses the ringing of the bell when an error occurs in change mode. The default for this parameter is NOQUIET.

8.5.9 SET SCREEN Width

The SET SCREEN width command controls the maximum length of a line EDT displays. In change mode, when you insert more characters than the length set for character editing, EDT displays all overflow characters as one solid rectangle on the VT52 and a diamond on the VT100. In line mode EDT may display the overflow characters on succeeding lines. This depends on the terminal type. The default for this parameter is 80 characters.

Example:

```

.
.
.
The limit for a line is 40 characters.
When you exceed that limit the overflow is indicated by a ♦
CTRL/Z
*SET SCREEN 40(RET)
*CHANGE(RET)
```

```

.
.
.
The limit for a line is 40 characters.
When you exceed that limit the overflow ♦
```

In the first part of the example, SET SCREEN is set to the default of 80 characters and there is no overflow. You then set the maximum number of characters EDT displays on a line to 40. The first line shown in frame 2 does not exceed 40 characters. The diamond indicates that the second line exceeds 40 characters.

8.5.10 SET SEARCH

The SET SEARCH command sets parameters for string searching in character editing.

8.5.10.1 SET SEARCH EXACT — The SET SEARCH EXACT command causes EDT to search for exact comparisons of case in the specified string(s). The default for this parameter is GENERAL, which disregards case.

8.5.10.2 SET SEARCH BOUNDED — The SET SEARCH BOUNDED command causes EDT to stop the search for the specified string at the next page entity marker. The default for this parameter is UNBOUNDED, which searches the entire buffer.

8.5.10.3 SET SEARCH END — The SET SEARCH END command causes EDT to leave the cursor at the end of the search string when it is found. The default for this parameter is BEGIN, which leaves the cursor at the beginning of the string. If the string is not found, the cursor position is unchanged.

```
.  
. .  
. .  
*SET SEARCH EXACT (RET)  
*SET SEARCH BOUNDED (RET)  
*SET SEARCH END (RET)  
*TYPE 'SUBStitute' .
```

EDT searches the current text buffer, from the current cursor position to the next page marker, for the first occurrence of the string SUBStitute. The first three characters in the string must be uppercase, and the remaining characters must be lowercase. When a string is found, EDT puts the cursor at the character position following the string.

8.5.11 SET TAB N

The SET TAB n command sets the number of spaces for the first tab stop in keypad character editing. The remaining tab stops for the terminal are unchanged (multiples of 8 spaces). After you have set the first tab stop with SET TAB, you can increase the first tab stop position by the value of SET TAB with (GOLD/E). You can decrement the first tab stop by the value of SET TAB with (GOLD/D).

EDT maintains an indentation level count. The count is set to one when you enter the SET TAB command. The indentation level count is incremented by one when you enter (GOLD/E) and decremented by one when you enter (GOLD/D). The first (TAB) entry for a line is the product of the value of SET TAB and the indentation level count. When the cursor is at position n times the initial SET TAB, you can set the first tab set to that position with (GOLD/A).

You can adjust the tab setting over a range of lines so that all lines in the selected range are indented. You enter the character editing SELECT function to define the start of the range. You then move the cursor to the end of the range of lines and enter `(GOLD/T)` preceded by a GOLD function and a + or - repeat count. The lines between the SELECT function entry and the `(GOLD/T)` function entry are indented to the value of SET TAB times the + or - repeat count.

Examples:

The first example shows the effect of the SET TAB command on the tab settings.

```
This is the first line in the text buffer.  
This is the next line in the text buffer.  
This is the last line.
```

```
(CTRL/Z)  
*SET TAB 5(RET)  
*CHANGE(RET)
```

```
This is the first line in the text buffer.  
(TAB) This is the next line in the text buffer.  
(TAB)(TAB)(TAB) This is the last line.
```

Entering a `(CTRL/Z)` places EDT in line editing mode. The SET TAB command sets the first tab indent to 5 spaces. The CHANGE command returns EDT to keypad character editing. Move the cursor to the start of the second line, and press `(TAB)`. EDT indents the line 5 spaces (the SET TAB value). Move the cursor to the start of the last line, and press `(TAB)` three times. EDT indents the line 16 spaces. The cursor is indented 5 spaces for the first tab, 3 spaces for the second tab, and 8 spaces for the third tab.

The next example shows the result of incrementing and decrementing the first tab indent by using `(GOLD/E)` and `(GOLD/D)`.

```
.  
. .  
(CTRL/Z)  
*SET TAB 12(RET)  
*CHANGE(RET)
```

```

For this example there are four lines.
This is the second line.
This is the third line.
This is the fourth line.
(TAB) The indent from the left margin is 12 spaces.(RET)
(GOLD/E) (TAB) The indent is now 24 spaces.(RET)
(GOLD/D) (TAB) The indent is back to 12 spaces.(RET)

```

With the SET TAB command, set the first tab for keypad character editing to 12 spaces, this sets the indentation level count to 1. Remember the remaining tab settings for a line are the default values, multiples of eight spaces from the left hand margin. Move the cursor to the end of the current text buffer (the empty line preceding [EOB]). Indent the cursor one tab space before entering text. After adding a line to the buffer, press (GOLD/E) and (TAB). This increments the indentation level count to 2; the first tab is set to 24 (the SET TAB value times the indentation level count). After adding a line indented 24 spaces, press (GOLD/D) and (TAB). This decrements the indentation level count by one, setting the indentation for the start of the next line to 12 spaces.

The next example shows how to indent a selected number of lines by using the SELECT and (GOLD/T) function keys. (See Chapter 10.)

```

(CTRL/Z)
*SET TAB 4(RET)
*CHANGE(RET)
<SELECT>
For this example there are four lines.
This is the second line.
This is the third line.
This is the fourth line.
(GOLD/T)

```

```

For this example there are four lines.
This is the second line.
This is the third line.
This is the fourth line.

```


Using the SET TAB command, set the first tab to 4 spaces and enter the CHANGE command. (The text is from the example above.) EDT displays the contents of the text buffer. Move the cursor to one end of the range of lines you wish to indent, and press the SELECT function key. Move the cursor to the other end of the range, and press (GOLD/T). EDT indents the selected range of lines four spaces; the remainder of the text buffer is unchanged. A (GOLD/3) followed by (GOLD/T) would shift the selected lines 12 spaces (3 times the SET TAB of 4).

The next example shows the use of the (GOLD/A) function key to set the indentation level count and subsequent first tab indents.

```

.
.
.
(CTRL/Z)
*SET TAB 5 (RET)

*CHANGE (RET)
For this example there are four lines.
This is the second line.
This is the third line.
This is the fourth line.
(Move cursor to 25 spaces) (GOLD/A)
(TAB) The first tab stop is now 25 spaces. (RET)
(GOLD/D) (TAB) The first tab stop is now 20 spaces.
[EOB]

```

Assume that the text buffer contains the four lines of text. The (CTRL/Z) places EDT in line editing mode. Using the SET TAB command, set the first tab to 5 spaces (the indentation level count is set to 1), and enter the CHANGE command. Move the cursor to the line following the text, and indent the cursor 25 spaces (a multiple of the SET TAB value). Enter a (GOLD/A); this sets the first tab to 25 spaces and sets the indentation level count to 5 (25/5=5). You can decrement or increment the tab level counter by pressing the (GOLD/D) and (GOLD/E) function keys respectively (each increment is 5 spaces).

8.5.12 SET TERMINAL

The SET TERMINAL command is used to identify the type of terminal in use. EDT gets the terminal type from the operating system. You can override the terminal type with the SET TERMINAL command. You can set the terminal to VT100, VT52, or HCPY.

8.5.13 SET [NO]TRUNCATE

The SET TRUNCATE command ends the display of a line at the value of SET SCREEN. The default is SET TRUNCATE. When you use

SET NOTRUNCATE, the input text which exceeds the SET SCREEN value is displayed on succeeding lines. You use SET NOTRUNCATE when you want to display lines that exceed the screen width.

Example:

```
.  
. .  
*SET SCREEN 40 (RET)  
*CHANGE (RET)
```

```
The line of text is limited to 40 chara ◆ (RET)  
. .  
CTRL/Z  
*SET NOTRUNCATE (RET)  
*CHANGE (RET)
```

```
The line of text is not limited to 40 c  
◆ haracters.  
. .
```

In the first part of the example, the screen display limit is set to 40 characters. The CHANGE command places EDT into character editing. The input line exceeds 40 characters; therefore, EDT truncates the line to the limit set by SET SCREEN. The solid diamond indicates that there are additional characters in the line that EDT does not display.

The CTRL/Z places EDT in command level. Enter SET NOTRUNCATE and the CHANGE command. The input line again exceeds the 40 characters of SET SCREEN, but this time the excess is displayed on the next line. The solid diamond indicates that this line is an overflow from the preceding line.

8.5.14 SET [NO]VERIFY

When you use the SET VERIFY command, the commands from command files and macro commands are printed when they are executed. The default for this parameter is NOVERIFY — the command files and the macros are not printed when executed.

8.5.15 SET [NO]WRAP N

The SET WRAP n command sets a line length limit of n character positions. This command is used in two ways:

1. When you are inserting text in character editing and the cursor position exceeds the value of n, EDT tries to wrap a full word to the next line. The word that exceeds the limit of n characters is moved to the following line. If there are no spaces in the character string, the line can be any length up to 255 characters. When you insert text in the middle of a line, the line can extend beyond the right hand margin without wrapping.
2. When you use the FILL command, SET WRAP sets the right hand margin for the fill. EDT fills each line in the range to the word delimiter nearest the limit n.

The default for this parameter is NOWRAP.

Example:

```
*SET WRAP 40(RET)
*CHANGE(RET)
This line contains 40 characters of text
This line contains more than 40
characters of text.█
```

EDT sets the right margin to 40 characters. The first line of input contains the limit of 40 characters. If you placed a period at the end of the first sentence (after “text”), EDT would write the word “text” and the period on the next line. The next line exceeds the limit of 40 characters within the word “characters,” so EDT moves “characters” to start a new line.

8.6 SHOW Command

The SHOW command displays selected information on the current state of EDT.

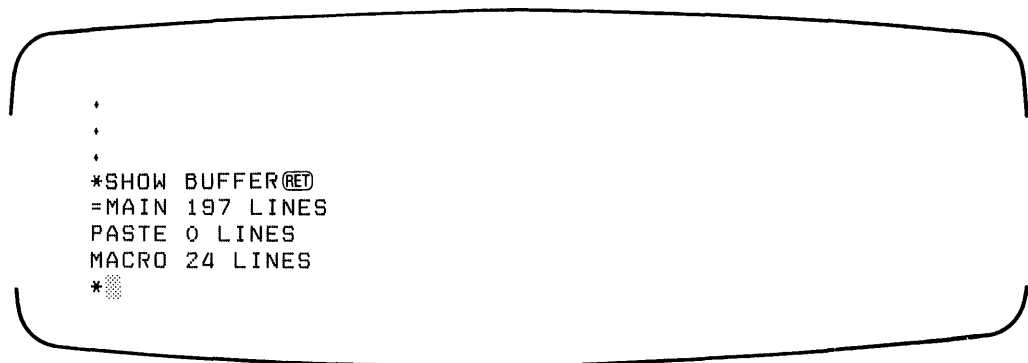
The SHOW command has the following form:

```
SHOW { BUFFER ;
      CASE ;
      CURSOR ;
      ENTITY { WORD ; SENTENCE ; PAGE ; PARAGRAPH } ;
      KEY [ { CONTROL letter ; [GOLD] n } ] ;
      SCREEN ;
      SEARCH ;
      TERMINAL ;
      VERSION }
```

8.6.1 SHOW BUFFER

The SHOW BUFFER command lists the buffers in use during the current edit session and the number of lines of text in each buffer. The main and paste buffers are always displayed under the SHOW BUFFER command even when they are empty. EDT marks the name of the current text buffer with an equal sign (=). If the line count is followed by an asterisk, there are more lines in the input file which have not been read by EDT; thus, the number of lines listed is not always exact.

Examples:



```
.
.
.
*SHOW BUFFER(RET)
=MAIN 197 LINES
PASTE 0 LINES
MACRO 24 LINES
***
```

EDT displays the names of all text buffers used in this editing session. MAIN is the current text buffer.

8.6.2 SHOW CASE

The SHOW CASE command shows the current case setting (UPPER, LOWER, or NONE). CASE is defined in the SET CASE command.

8.6.3 SHOW CURSOR

The SHOW CURSOR command shows the current cursor range. CURSOR is defined in the SET CURSOR command.

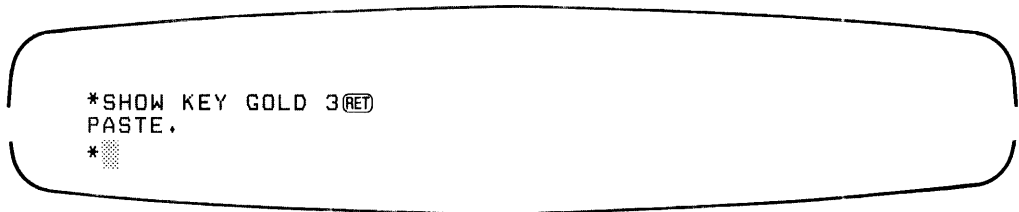
8.6.4 SHOW ENTITY

The SHOW ENTITY command shows the current setting for the specified entity (WORD, SENTENCE, PARAGRAPH, and PAGE). The entities are defined in the SET ENTITY command.

8.6.5 SHOW KEY

The SHOW KEY command (CONTROL letter ; [GOLD]n) shows the definition of the specified key in change mode. You can change the definition of the key with the DEFINE KEY command.

Example:



```
*SHOW KEY GOLD 3(RET)
PASTE.
*
```

The example assumes that EDT is at command level. Enter the SHOW KEY command and the key (the numbers for the keypad keys are shown in Figure 10-3). EDT responds by displaying the current command value for that key.

8.6.6 SHOW SCREEN

The SHOW SCREEN command shows the current setting for the maximum length of a line EDT displays. SCREEN is defined in the SET SCREEN command.

8.6.7 SHOW SEARCH

The SHOW SEARCH command shows the current search parameters. SEARCH is defined in the SET SEARCH command.

8.6.8 SHOW TERMINAL

The SHOW TERMINAL command shows your terminal type (HCPY, VT52, or VT100).

8.6.9 SHOW VERSION

The SHOW VERSION command shows the version number of EDT.

Chapter 9

Nokeypad Character Editing

The character editing commands operate on a range of characters defined within each command entry, either specifically or by default. You enter the commands through the keyboard.

You enter nokeypad character editing through the CHANGE command and if you have a VT52 or VT100 terminal, you must also have used the SET NOKEYPAD command.

You enter nokeypad character editing commands after the command prompt. For VT52 and VT100 terminals, the command prompt is the cursor. The cursor defines your position in the text buffer (the reference point for command entries). When you enter character editing commands, EDT displays your keyboard entries at the bottom of the screen. Press **RET** to execute the command. When the command sequence is complete, the cursor returns to your position in the text buffer. For a description of the prompt on other terminals, see Appendix A.

9.1 Entities of Text

An entity is a definable quantity of text. The character editing commands work on multiples of these entities. Table 9-1 describes the defined entities for the change mode commands.

Table 9-1: Entities

Entity	Description
C (Character)	A C is a single character. The characters include all of the alphabetic, numeric, and special characters on the keyboard.
W (Word)	A W is a string of characters delimited by one of a set of delimiter characters, including spaces. The default word terminators are spaces, <code>␣</code> , tabs, and <code>␣</code> (form feed). You can define the word delimiters with the SET ENTITY command. Each delimiter other than space is a word.
BW (Beginning Of Word)	A BW is the string of characters from the cursor position to the beginning of the word. When the cursor is at the beginning of a word, BW is the previous word.
EW (End Of Word)	An EW is the string of characters from the cursor position to the end of the word. This includes the character at the cursor position.
L (Line)	An L is a single line of text, where line is a string of characters between line terminators.
BL (Beginning Of Line)	A BL is the string of characters from the cursor position to the beginning of the line. When the cursor is at the beginning of a line, BL is the previous line.
EL (End Of Line)	An EL is the string of characters from the cursor position to the end of the line. This includes the character at the cursor position.
NL (Next Line)	An NL is the string of characters from the cursor position to the beginning of the next line.
PAR (Paragraph)	A PAR is a user-defined entity. EDT defaults to two successive RETURNs for a paragraph delimiter. You define the paragraph delimiter with the SET ENTITY command.
BPAR (Beginning Of Paragraph)	A BPAR is the string of characters from the cursor position to the beginning of the paragraph.
EPAR (End of Paragraph)	An EPAR is the string of characters from the cursor position to the end of the paragraph (not including the paragraph delimiter). This includes the character at the cursor.
SEN (Sentence)	A SEN is a user-defined entity. EDT defaults to a period (.), a question mark (?), or an exclamation mark (!) for sentence delimiters. You define the sentence delimiters through the SET ENTITY command.
BSEN (Beginning Of Sentence)	A BSEN is the string of characters from the cursor position to the beginning of the sentence.
ESEN (End Of Sentence)	An ESEN is the string of characters from the cursor position to the end of the sentence (not including the sentence delimiter).

(continued on next page)

Table 9-1: Entities (Cont.)

Entity	Description
PAGE	A PAGE is a user-defined entity. The default for a page is the text between two form feed characters (including the second form feed). You define the page delimiters with the SET command.
BPAGE (Beginning Of Page)	A BPAGE is the text from the cursor position to the beginning of the page.
EPAGE (End Of Page)	An EPAGE is the text from the cursor position to the end of the page (not including the page delimiters).
BR (Beginning Of Range)	A BR is the string of characters to the left of the cursor in the current text buffer. This is equal to a BL command plus all preceding lines.
ER (End Of Range)	An ER is the string of characters to the right of the cursor in the current text buffer. This is equal to an EL command plus all remaining lines.
V (Vertical)	A V is equivalent to L except that the cursor stays in the same column.
'string'	A 'string' is a group of characters within quotation marks. It is used in search operations. When you specify the string entity with the D (delete) or CUT commands, EDT includes all characters from the cursor to the string.
SR (Select Range)	An SR is the text between the last SEL (select) command and the cursor position. When select range is not active (no select command entered) and the cursor is at the present search string, SR selects the search string. When neither of the above conditions exists, SR applies to the CHGC (changecase) command. For the CHGC command, SR selects a single character in the current direction.

9.2 No keypad Command Structure

The command keywords have one and only one form (no abbreviations are allowed). You can put no keypad character editing commands together in a series; no delimiter is required between the commands but one or more spaces are allowable.

The no keypad character editing commands consist of one or more of the following elements in a specified sequence:

Command The command specifies EDT's action.

Count The count specifies the number of entities the command acts upon. The count must be from 1 to 32767 decimal.

Direction	The minus sign (-) indicates that the action performed by the command is to the left or up from the cursor. The plus sign (+) indicates the action performed by the command is to the right or down from the cursor.
Entity	The entity defines the basic element of text (for example, character, word, or paragraph).
Buffer	The buffer defines a storage location and is used in the CUT, PASTE, and APPEND operations.

The sequence in which the nokeypad commands use these elements takes one of the following formats:

1. command
2. [+ | -] [count] command
3. [+ | -] [repeat-count] command [+ | -] [entity-count] [+ | -] entity [=buffer]

The first are the fixed format commands, which specify EDT's action. You cannot modify the action of these commands.

The second are those commands with variable count and direction parameters (or fields). With all but two of the format 2 commands, the direction has no meaning and is ignored by EDT. EDT defaults to a count of 1 in the current direction.

The third are those commands which specify an action on an entity (character, word, line, or sentence). These commands have variable count and direction fields. Some commands can operate in a buffer other than the current buffer. The direction sign can be in any of the positions shown. Two counts are allowed; they are multiplied together.

You can repeat any string of change mode commands by enclosing the string with parentheses and preceding the parentheses with a count. When a command in the string fails, EDT exits the repeated string.

9.3 Format 1 Commands

The format 1 commands follow.

9.3.1 ADV (Advance)

The ADV command sets all commands forward (the defined operation is to the right and down from the cursor position) unless you override ADV with a minus sign (-).

9.3.2 BACK (Backup)

The BACK command sets all commands backward (the defined operation is to the left or up from the cursor position) unless you override BACK with a plus sign (+).

9.3.3 EX (Exit)

The EX command exits EDT from the nokeypad character editor change mode back to EDT command level.

9.3.4 EXT (Extended)

The EXT command enters line mode commands while EDT is in change mode. EDT accepts the rest of the line (after EXT) as an editor command level command. When EDT completes the specified command, it returns you to change mode.

Example:

```
Line 1
Line 2
Line 3
Line 4
Line 5
Line 6
Line 7
Line 8
Line 9 is the first line we want to retain.
Line 10 is also to be retained.
.
.
.
EXT DEL 1 THRU 8(RET)
```

```
Line 9 is the first line we want to retain.
Line 10 is also to be retained.
.
.
.
EXT COPY=HEADER TO 9(RET)
```

```
This is from the text buffer =HEADER.
Line 9 is the first line we want to retain.
Line 10 is also to be retained.
.
.
.
```

EDT deletes all lines in the range from 1 through 8 from the current text buffer. For illustrative purposes it is assumed that the text buffer line numbers match numbers shown in text. EDT then refreshes the screen with the line succeeding the last line deleted at the top of the display. With the second EXT command, EDT copies the contents of text buffer HEADER preceding line 9 and refreshes the screen to display line 9 and one or more of the lines of the inserted text.

9.3.5 I (Insert)

The I command opens the current text buffer for the insertion of text. The text is inserted in front of the cursor position. EDT displays the text, at its proper location in the text buffer, as you insert it. To exit from the insert operation, enter a `CTRL/Z`.

Example:

```
This is line one.  
This is line two.  
.  
.  
.  
I (RET)
```

```
This is line one.  
This is line two.  
This line is inserted after line two. CTRL/Z  
.  
.  
.
```

In this example the first two lines of the display are shown; the cursor is at the beginning of line 3. To insert text, enter I followed by a `RET`. The I appears at the bottom of the display until you exit the INSERT command. When you wish to exit the insert command, enter `CTRL/Z`.

9.3.6 QUIT

The QUIT command exits from EDT and returns you to the system command level prompt. EDT does not save any files.

9.3.7 REF (Refresh)

The REF command forces EDT to refresh the entire screen. EDT normally refreshes only those portions of the screen changed by the command operations. This command is useful when the screen contains extraneous text, such as when someone sends a message to your terminal while you are using EDT.

9.3.8 SEL (Select)

The SEL command is used to select a range of text. SEL marks the present cursor position. To select a range, you enter SEL at one end of the range and then move the cursor to the other end. The select range is the text between the cursor and the position marked by SEL. You can use the SEL range as an entity.

Examples:

```
  This is line one.  
  This is line two.  
  This is inserted after line two.  
  This is line three.  
  .  
  .  
  .  
  SEL(RET)
```

```
  This is line one.  
  This is line two.  
  This is inserted after line two.  
  This is line three.  
  .  
  .  
  .
```

The cursor location is the beginning of line one. By entering the SEL command, you set the beginning of line one as the start of your SEL range. The cursor marks the other end of the range; moving the cursor moves the other end of the range. With the cursor between “inserted” and “after”, the range is from the start of line one to the beginning of “after”.

9.3.9 TAB

The TAB command sets the tab position in one of two ways:

1. When you have not set a tab size with SET TAB (see Chapter 8) or when

the cursor is not positioned at the start of a line, the TAB command inserts a tab character at the cursor position.

2. When you have set a tab size with SET TAB and the cursor is positioned at the beginning of a line, the TAB command inserts a number of tab characters and spaces to move the cursor to the column position which is equal to the SET TAB value times the indentation level count.

The indentation level count defines the current level of indentation in terms of the tab size. The number of columns to indent is equal to the indentation level count multiplied by the tab size. The counter is set to 1 when you enter a SET TAB value. The counter is incremented by 1 with the TI command and decremented by 1 with the TD command. The counter is set by the TC command also.

Example:

```
SET TAB 4(RET)
SET NOKEYPAD(RET)
CHANGE(RET)
This is line one.
This is line two.
This is line three.
.
.
.
TAB(RET)
```

```
    This is line one.
This is line two.
This is line three.
.
.
.
TC(RET)
TI(RET)
```

You set the value for the first tab setting to 4 spaces, select no keypad character editing, and enter the CHANGE command. Entering the TAB command moves the cursor four spaces, to character space 5. If you then move the cursor to a multiple of the SET TAB value plus one character and enter TC, the indentation level count is set to the result of dividing the value for the present cursor position by the SET TAB value (the left margin is position 0). In this example the indentation level count is set to 3. The TI command increments the indentation level count by one to the value 4.

9.3.10 TC (Tab Compute)

The TC command sets the indentation level count to the value obtained from dividing the current cursor column position by the SET TAB number. An error message occurs if the cursor is not at a multiple of the SET TAB number. EDT uses the product of indentation level count and tab size to set the first tab position for each line. Subsequent tab positions on the same line are a multiple of the terminal's tab setting (8 spaces). See the example in TAB above.

9.3.11 TOP

The TOP command places the current line at the top of the screen.

9.4 Format 2 Commands (Without Direction)

The format 2 commands that don't accept a direction follow.

9.4.1 ASC (ASCII)

With the ASC command you enter the decimal number representation (the value 0 through 255 for count) of the desired ASCII character, and EDT displays the character. EDT ignores a minus sign (-) preceding the count. See Appendix C for the ASCII code decimal equivalents.

Example:

```
This is line one.  
This is line two.  
.  
.  
.  
10ASC(RET)
```

```
This is line one.  
This is line two.  
<LF>  
.  
.  
.
```

The cursor location is just after line two. When you enter 10ASC and (RET), EDT enters <LF> at the cursor position.

9.4.2 SHL (Shift Left)

The SHL command shifts the screen image to the left. You control the amount of shift through your count entry, where shift equals count times 8 (8 equals one tab stop). EDT does not display the left portion (count times 8) of the text. You use the SHL command to counter the SHR command. See the example in SHR below.

9.4.3 SHR (Shift Right)

The SHR command shifts the screen image to the right. You control the amount of shift through your count entry, where shift equals count times 8 (8 equals one tab stop). You use the SHR command to counter the SHL command.

Example:

```
*SET SCREEN 40 (RET)
*CHANGE (RET)
The line of text is limited to 40 charact
To be able to read that which is beyond t
This is line three.
.
.
.
4SHL (RET)
```

```
characters.
beyond the limit use the SHL command.
.
.
.
4SHR (RET)
```

```
The line of text is limited to 40 charact
To be able to read that which is beyond t
This is line three.
.
.
.
```

In the first part of the example, the lines are set to truncate at 40 characters. The first two lines extend beyond the 40 character limit and are truncated. The 4SHL command moves the left margin 32 characters to the right. The remainder of the first two lines can now be read. The third line is less than 32 characters long; none of the third line is visible. The SHR command restores the text to its original position.

9.4.4 TD (Tab Decrement)

The TD command decreases the indentation level count. When the indentation level count is zero, the count is not changed. See the example under TAB in the format 1 commands description.

9.4.5 TI (Tab Increment)

The TI command increases the indentation level count. See the example under TAB in the format 1 commands description.

9.4.6 UNDC (Undelete Character)

The UNDC command inserts the contents of the character buffer into the current text buffer (just in front of the cursor). The character buffer contains the last character deleted by one of the delete character commands. You can use count to repeat the UNDC operation; EDT inserts the contents of the character buffer into the text buffer count times.

9.4.7 UNDW (Undelete Word)

The UNDW command inserts the contents of the word buffer into the current text buffer (just in front of the cursor). The word buffer contains the last word deleted by one of the delete word commands. You can use count to repeat the UNDW operation; EDT inserts the contents of the word buffer into the text buffer count times.

9.4.8 UNDL (Undelete Line)

The UNDL command inserts the contents of the line buffer into the current text buffer (just in front of the cursor). The line buffer contains the last line deleted by one of the delete line commands. You can use count to repeat the UNDL operation; EDT inserts the contents of the line buffer into the text buffer count times.

9.4.9 ^ (Circumflex)

The ^ enters a control character in your text. You enter the desired control character (A through Z) after the circumflex.

9.5 Format 2 Commands (With Direction)

The two format 2 commands that accept a direction with count follow.

9.5.1 S/s1/s2/ (Substitute)

The S/s1/s2/ command replaces string s1 with string s2. Use count to define the number of substitutions and a minus sign (-) when you want a backward search. Any nonalphanumeric character can be used as a delimiter. The search string (string-1) is stored in the string search buffer; the substitute string is stored in the substitute buffer.

9.5.2 SN (Substitute Next)

The SN command uses the s1 and s2 defined in the last substitute command (stored in the search string and substitute buffers) to replace the next occurrence of s1 with s2. Use count to define the number of substitutions and a minus sign (-) when you want to search backward in the buffer.

Example:

```
This is line one.  
This is line two.  
This is line three.  
. .  
S/line/sentence/RET
```

```
This is sentence one.  
This is line two.  
This is line three.  
. .  
2SNRET
```

```
This is sentence one.  
This is sentence two.  
This is sentence three.  
. .
```

With the cursor at the start of line one, enter the S/line/sentence/ command and a `(RET)`. EDT replaces the string “line” in the first line with the string “sentence”. If you then enter the command 2SN, EDT substitutes the next two occurrences of the string “line” with the string “sentence”. You can achieve the same result by preceding the S command with a count of 3.

9.6 Format 3 Commands

The format 3 commands follow:

9.6.1 APPEND

The APPEND command moves the specified entities to another text buffer. EDT deletes the moved text from the current text buffer. Use the =buffer range specification to name the receiving text buffer; EDT defaults to the paste text buffer. EDT appends the specified entities to the end of the contents of the receiving text buffer. See the example in the CUT command description.

9.6.2 CUT

The CUT command moves the specified entities to another text buffer. EDT deletes the moved text from the current text buffer. Use the =buffer range specification to name the receiving text buffer; EDT defaults to the paste text buffer. EDT deletes the previous contents of the receiving text buffer.

Example:

```
This is line one.  
This is line two.  
This is line three.  
This is line four.  
This is line five.  
. . .  
CUTZL=SHOW(RET)
```

```
This is line three.  
This is line four.  
This is line five.  
. . .  
ZAPPENDL=SHOW(RET)
```

```
This is line five.
```

```
█  
.  
.  
.
```

```
PASTE=SHOW(RET)
```

```
This is line five.  
This is line one.  
This is line two.  
This is line three.  
This is line four.
```

```
█  
.  
.  
.
```

The CUT command moves the two lines following the cursor to the text buffer you have named SHOW. EDT deletes these two lines from the current text buffer. The 2APPEND command moves the two lines following the cursor position to the text buffer SHOW. These two lines, which are deleted from the current text buffer, are inserted at the end of the present contents of text buffer SHOW. After you move the cursor to the left margin following line five, enter the PASTE = SHOW command. EDT copies the contents of the text buffer SHOW to the present cursor location.

9.6.3 D (Delete)

The D command deletes a specified number of entities.

Example:

```
This is line one.  
This is line two.  
This is line three.
```

```
█  
.  
.  
.
```

```
D2W(RET)
```

```
line one.  
This is line two.  
This is line three.  
.  
.  
.  
LDL(RET)
```

```
line one.  
This is line three.  
.  
.  
.
```

Assume the cursor is located at the beginning of line one. To delete the first two words, you enter D2W and (RET). EDT deletes the first two words and leaves the cursor at the beginning of the third word.

To delete line two, you enter LDL and (RET). EDT moves down one line in response to the first L and deletes the line in response to the DL command. The cursor is at the beginning of line three.

9.6.4 FILL

The FILL command places the maximum amount of text on each line within the bound set by the SET WRAP command. The text from the succeeding lines are moved up to fill out the line. Text is broken between words; the line is filled to the word delimiter that occurs before the limit of SET WRAP is exceeded. FILL does not fill across blank lines; the lines between blank lines are filled. The default value for the fill line is 80 characters.

Example:

```
Here is a text buffer that you have modified  
through a series of  
edits such that the right  
margin is ragged over the first four lines.  
.  
.  
.  
FILL4L(RET)
```

```
There is a text buffer that you have modified through a
series of edits such that the right margin is ragged
over the first four lines.
```

```
█
.
.
.
```

The first four lines represent the text buffer after a series of edits. The SET WRAP is set at 64. When you enter FILL4L, EDT fills lines, with up to 64 characters in each line, until the end of the fourth line is reached.

9.6.5 [Null] (Move Cursor)

The null command moves the cursor the specified number of entities. You enter the entity, the quantity, the direction, and then press (RET).

Example:

```
█ This is line one.
This is line two.
This is line thr
.
.
.
2L (RET)
```

```
This is line one.
This is line two.
█ This is line thr
.
.
.
3W3C (RET)
```

Assume the cursor is located at the first character of line one and that you want to move to the end of the third line. Enter 2L and (RET); EDT moves the cursor to the start of the third line. Enter 3W3C and (RET) and EDT moves the cursor three words and three characters. The cursor is at the end of line three. You can perform the same sequence with one command string (2L3W3C) or with three command strings (2L, 3W, 3C). EDT executes the commands in the string in the order you entered them.

9.6.6 PASTE

The PASTE command copies the contents of the specified text buffer in front of the cursor location. The contents of the specified buffer is unchanged by the PASTE operation. When you do not specify a text buffer, EDT defaults to the paste text buffer. See the example in the CUT command description.

9.6.7 R (Replace)

The R command replaces deleted text with inserted text. EDT deletes the selected entities and then enters the insert mode. To exit the insert mode, you enter a `CTRL/Z`.

Example:

```
This is line one.  
This is line two.  
This is an improper entry.  
This is also an improper entry.  
This is line six.  
. .  
R2L RET
```

```
This is line one.  
This is line two.  
This is line three. RET  
This is line four. RET  
This is line five. CTRL/Z  
This is line six.  
. .
```

Use the R command to replace the third and fourth lines, which are in error, with new text. Move the cursor to the first character position of line three and enter the R2L command. EDT deletes lines three and four and opens the text buffer for the insertion of text. You exit the insert mode with `CTRL/Z`.

You can use the replace command on other entities. EDT deletes the selected entities and opens the text buffer for the insertion of text.

9.6.8 TADJ (Tab Adjust)

The TADJ command adjusts the tab level for the selected range of lines. You set the number of lines through the range specification. The tab size and repeat-count set the tab level (see TC). The tab level is adjusted by the value of repeat-count; it is incremented for a plus repeat-count and decremented for a negative repeat-count. The TADJ tab setting is the product of the SET TAB value and the indent level.

Example:

```
This is the first line of text with a list following:
Item a
Item b
Item c
Item d
The list of items is complete.
.
.
.
TADJ4L (RET)
```

```
This is the first line of text with a list following
Item a
Item b
Item c
Item d
The list of items is complete.
.
.
.
```

With the cursor at the beginning of second line, you enter TADJ4L. EDT indents the next four lines to the value of SET TAB. To increase the indent, you use the repeat-count. EDT sets the indent level to the product of repeat-count and the value of SET TAB.

Chapter 10

Keypad Functions in Change Mode

This chapter describes keypad editing, which is available when the editor is in change mode. Because a key is not labeled with what it does, this chapter discusses functions rather than keys.

A key is found on the keyboard or keypad and is labeled with characters such as “,” or “ENTER”. By contrast, a function is the operation performed when you press that key. For example, the SUBSTITUTE and ENTER functions are associated with the ENTER function key.

The functions in this chapter are grouped by what they do:

- Essential functions
- Entering commands and functions
- Moving the cursor
- Changing cursor movement
- Deleting and undeleting text
- Replacing text
- Inserting lines and moving sections
- Special characters, changing case, and redefining keys
- Keyboard commands
- Control character functions

10.1 The Keypad and its Functions

In change mode you can select from the set of either keypad functions or nokeypad commands. Keypad functions let you perform much the same operations as nokeypad commands, which are described in the preceding chapter. However, there are differences between the two ways of editing:

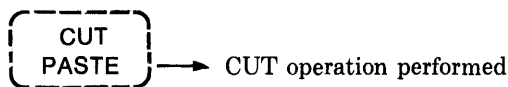
- You can enter nokeypad commands from any terminal. Keypad functions work only on VT52 and VT100 terminals.
- You type most nokeypad commands at the keyboard, whereas you simply press keys for keypad functions.

You can start keypad editing in change mode with the CHANGE command. At this level EDT can perform keypad functions or nokeypad commands. VT52 and VT100 terminals default to keypad editing; other terminals default to nokeypad editing. (You can change this default by using the SET command.)

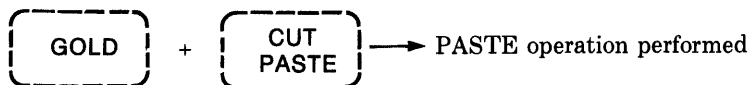
Figure 10-1 shows the VT52 keypad and its corresponding functions; Figure 10-2 shows the keypad and functions of the VT100.

Notice that most of the keys in Figures 10-1 and 10-2 have two functions. The upper of the two is the “standard” function, and the lower one is the “alternate” function. You can use either of the two functions, depending on whether or not you press the GOLD key first. Press the key itself to use the standard function of that key. To use the alternate function, press the GOLD key and then that function key.

For example, CUT is the standard function of the VT100 keypad key labeled “6”. PASTE is the alternate function of this key. To use the CUT function, you press the CUT function key:



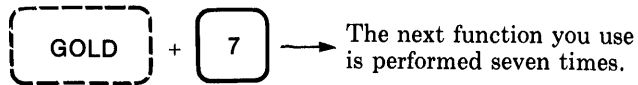
To use the PASTE function, press the GOLD key and then the PASTE function key:



Throughout this chapter, a statement such as “Use the PASTE function key” assumes that you use the GOLD key and then the PASTE function key.

Figure 10-2 shows that there are four arrow keys at the top right of the VT100 keyboard. These arrow keys let you move the cursor about in the text. In addition, the LINE FEED and BACK SPACE keys have special functions in keypad mode. You can also use several CONTROL functions by pressing the CTRL key (CTRL) and one of several alphabetic character keys.

The keys in the diagrams are enclosed by solid lines. Functions are shown with broken lines. The result of a keypad operation is shown with an arrow. For example, if you are told to “press the GOLD function key on the keypad and then the 7 key on the keyboard,” the diagram looks like this:



There are 36 KEYPAD functions that you can perform with the VT52, 38 with the VT100.

10.2 Starting and Ending an Editing Session

You can use keypad editing only if you have a VT52 or VT100 terminal. When you enter change mode with these terminals, the default is keypad editing.

10.2.1 Beginning the Editing Session

When you type the CHANGE command, you can begin editing with the keypad immediately. If you are editing an existing file, a copy of the file’s contents appears on the screen. If you are creating a new file, the screen clears for your input. All entries that you make on the keyboard are entered as text before the cursor.

The following describes how to start change mode and enter text. The asterisk (*) shows that you are already using EDT. Type CHANGE after the asterisk prompt:

```
* CHANGE (RET)
```

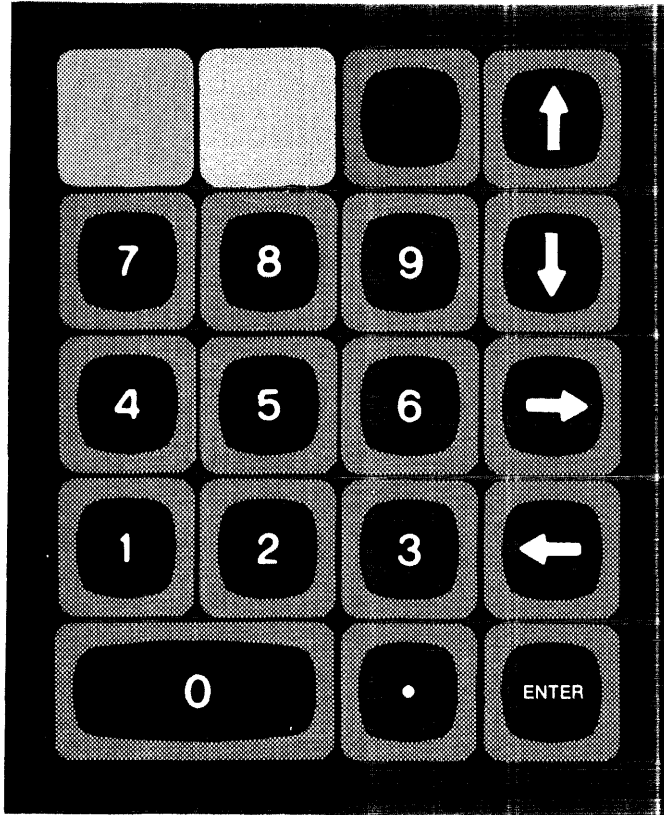
The screen becomes blank, since there is no text in the new file. The cursor appears in the upper left of the screen. The message [EOB] shows that you are at the end of the buffer in the new file. What you type appears before the cursor. For example, enter the following text:

```
Here is an example (RET)
of how to enter (RET)
text with the keypad. (RET)
[EOB]
```

10.2.2 What it Means to Save or Delete Edits

The edits that you make during an EDT editing session are stored in a text buffer. When you “save” or “delete” edits, you are really saving or deleting the contents of the text buffer. Therefore, when you create a new file, you

Figure 10-1: The VT52 Keypad and Its Functions

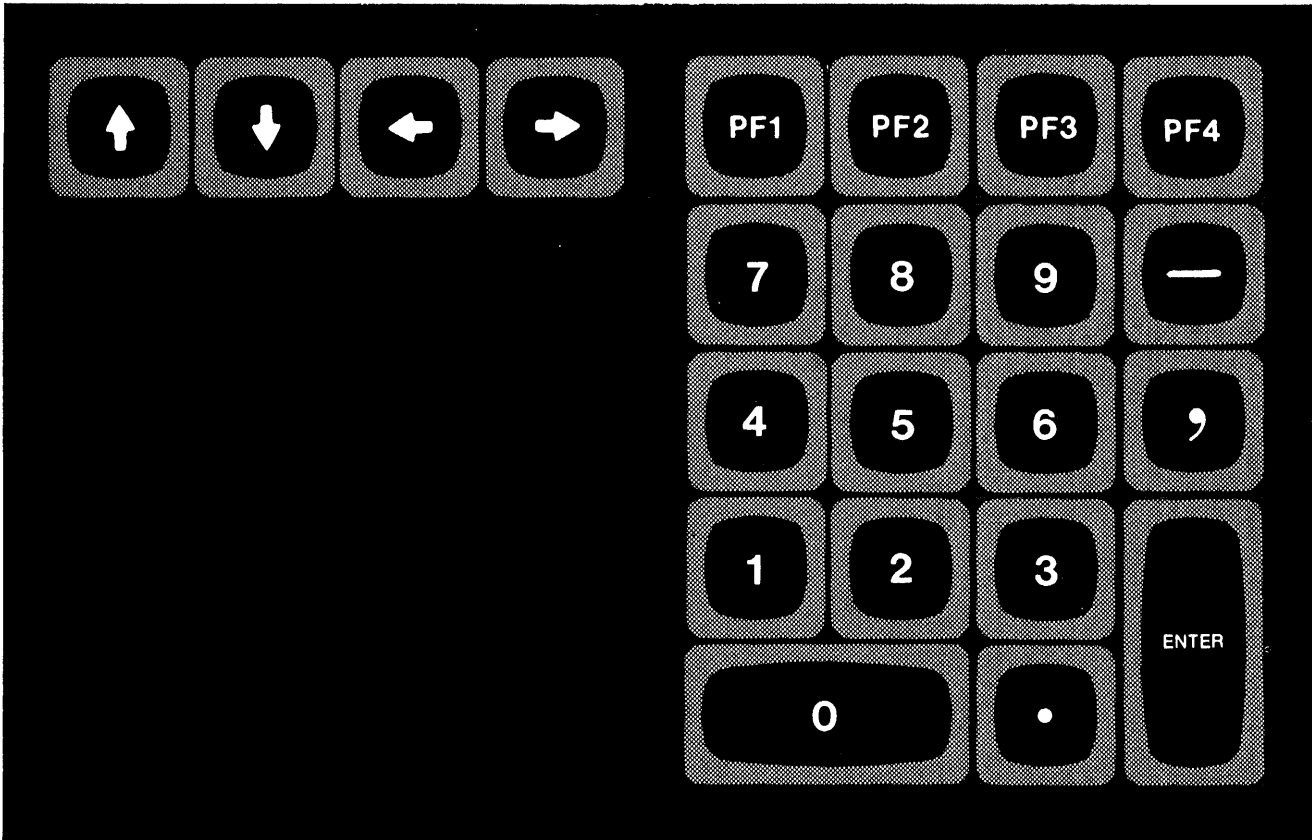


A. The Keypad

GOLD	HELP	DEL L UND L	UP REPLACE
PAGE COMMAND	FNDNXT FIND	DEL W UND W	DOWN SECT
ADVANCE BOTTOM	BACKUP TOP	DEL C UND C	RIGHT SPECINS
WORD CHNGCASE	EOL DEL EOL	CUT PASTE	LEFT APPEND
LINE OPEN LINE		SELECT RESET	ENTER SUBS

B. Keypad Functions

Figure 10-2: The VT100 Keypad and Its Functions



A. The Keypad and the Arrow Keys

UP	DOWN	LEFT	RIGHT
----	------	------	-------

GOLD	HELP	FNDNXT FIND	DEL L UND L
PAGE COMMAND	SECT FILL	APPEND REPLACE	DEL W UND W
ADVANCE BOTTOM	BACKUP TOP	CUT PASTE	DEL C UND C
WORD CHNGCASE	EOL DEL EOL	CHAR SPECINS	ENTER SUBS
LINE OPEN LINE		SELECT RESET	

B. Keypad Functions

decide whether or not to store the text buffer copy as the new file. When you work on a copy of an existing file, you decide whether to incorporate your edits into the file or to delete your work from the editing session.

10.2.3 Saving Your Edits

You can save the edits you make during a keypad editing session several ways. One way is to press **CTRL/Z** to return to line mode. At this point you type EXIT after the asterisk and press the ENTER function key on the keypad.

Another way to save your edits is to press GOLD and then the COMMAND function key. Type EXIT after the prompt and press the ENTER function key:

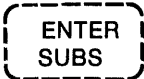
```
COMMAND:  EXIT 
```

Whether you type EXIT in line mode or after the COMMAND prompt, you save the file with your edits and exit EDT.

10.2.4 Deleting Your Edits

You can delete all the edits you make during a keypad editing session two ways. First, you can return to line mode by pressing **CTRL/Z** and type QUIT after the asterisk.

Second, you can press GOLD and then the COMMAND function key. Type QUIT after the prompt and press the ENTER function key:

```
COMMAND:  QUIT 
```

Both ways delete all your edits and end your session with EDT.

10.3 Essential Functions

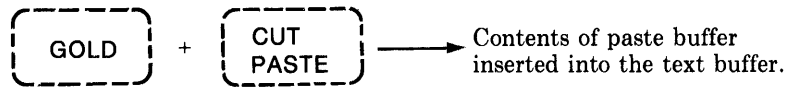
These functions are either required for or basic to keypad operations.

GOLD

The GOLD function causes EDT to perform the alternate (lower) function on any of the keypad keys. Use GOLD by pressing the GOLD function key and then a keypad key.

The following example shows how the GOLD function is used in a PASTE

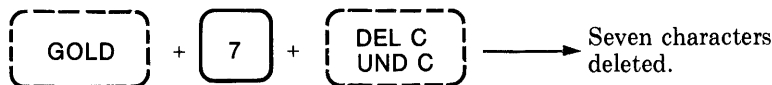
operation (to actually perform this operation, you need information described later in this chapter):



As shown above, when you press the GOLD key and then the key for the alternate function, the function is performed.

GOLD remains in effect until you press one of the other keypad keys. To cancel the effect of GOLD, press the RESET function key.

GOLD also lets you execute a function any given number of times. First, press GOLD and enter a number from the keyboard. This number appears at the bottom left of the screen and remains until you complete the operation. Press a keypad key for the function you want performed:



If you make a mistake in the middle of this operation, you can edit what you have typed two ways:

- To delete the last character typed, press **DEL**.
- To delete the entire operation, press **CTRL/U**.

Note, however, that the preceding operation is executed when you press the last function key in the operation (DEL C in this example). Neither **DEL** nor **CTRL/U** will undo an operation once it is executed.

HELP

Pressing the HELP function key causes a picture of keypad functions and CONTROL key descriptions to appear on your screen. When you then press one of the keypad keys, EDT displays a description of the standard and alternate functions of that key.

At this point you can:

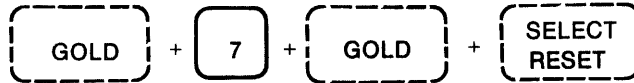
- Return to the keypad diagram by pressing the RETURN key
- Get help on another keypad key by pressing the key
- Exit from HELP by typing a space

When you exit from HELP, you return to keypad editing. The cursor stays in the same location while you use HELP.

RESET

You can use **RESET** to cancel the effects of the **GOLD** and **SELECT** functions or any key sequence that is partially entered. To use **RESET**, press **GOLD** and then the **RESET** function key. **RESET** restores the keypad to the way it was before your last command.

For example, suppose you enter a sequence of functions and then decide against using it. The following example shows how to cancel the sequence:



The repeat operation is cancelled by the **GOLD** and **RESET** key sequence. Thus, you can use **RESET** to cancel any operation that has not yet been executed. **RESET** will not undo an operation once it is performed.

10.4 Inserting Text and Lines

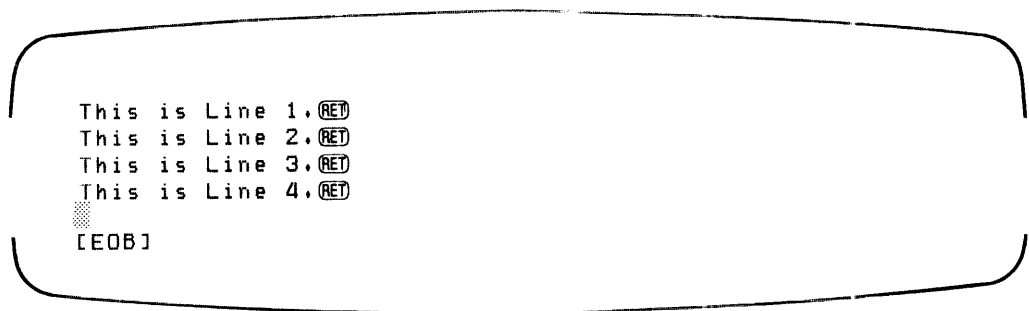
You can enter text into the text buffer as soon as you are in the keypad editing part of change mode.

10.4.1 Entering Text

Whatever you type on the keyboard when doing keypad editing is inserted directly before the cursor as text. Enter change mode by typing **CHANGE** after the line mode asterisk prompt and pressing the **RETURN** key:

```
* CHANGE (RET)
```

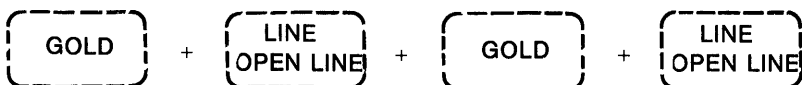
Then type in text at the keyboard:



10.4.2 Using OPEN LINE

You can use OPEN LINE to insert a RETURN after the cursor. The cursor position does not change. If the cursor is at the beginning or end of a line, the effect of this function is to insert a blank line:

```
This is Line 1.  
This is Line 2.  
This is Line 3.  
This is Line 4.  
█  
[EOB]
```

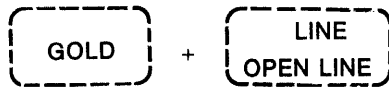


```
This is Line 1.  
This is Line 2.  
This is Line 3.  
This is Line 4.  
█  
[EOB]
```

Notice that EDT inserts two blank lines after the cursor, one for each OPEN LINE operation.

If the cursor is in the middle of a line, EDT moves any text at the right of the cursor down to the beginning of the next line:

```
This is Line 1.  
This is █ine 2.  
This is Line 3.  
This is Line 4.  
[EOB]
```

```

This is Line 1.
This is █
Line 2.
This is Line 3.
This is Line 4.

[EOB]

```

10.5 Moving the Cursor

The functions in this section let you move the cursor and set the direction of cursor movement during your editing session.

10.5.1 The Arrow Keys

You can move the cursor up, down, to the left, and to the right with the four arrow keys. These keys are found on the VT52 keypad and on the VT100 keyboard.



The UP (up arrow) function key moves the cursor to the character in the line above. If the line above does not extend to the present cursor position, EDT places the cursor at the end of the line. The cursor moves to the original character space when successive lines extend that far. For example:

```

This is the longest of the three lines.
This is a line.
This is a slightly longer line.█

```



```

This is the longest of the three lines.
This is a line.█
This is a slightly longer line.

```



```
This is the longest of the three lines.  
This is a line.  
This is a slightly longer line.
```

The following diagrams illustrate cursor movement with the UP function key. The first frame shows the original cursor position, and the second frame shows what happens when you press the UP function key twice:

```
This is an example of  
how the arrow keys  
let you move the cursor.
```



(2 times)

```
This is an example of  
how the arrow keys  
let you move the cursor.
```

DOWN



The DOWN (down arrow) function key moves the cursor to the character below. If the line does not extend to the present cursor position, EDT places the cursor at the end of the line. The cursor moves to the original character space when successive lines extend that far.

The following diagrams illustrate cursor movement with the DOWN function key. The first frame shows the original cursor position, and the second frame shows what happens when you press the DOWN function key once.

This is an example of
how the arrow keys
let you move the cursor.



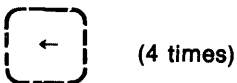
This is an example of
how the arrow keys
let you move the cursor.



The LEFT (left arrow) function key moves the cursor to the preceding character. When the cursor is on the left margin, EDT moves the cursor after the rightmost character on the line above.

The following diagrams illustrate cursor movement with the LEFT function key. The first frame shows the original cursor position, and the second frame shows what happens when you press the LEFT function key four times:

This is an example of
how the arrow keys
let you move the cursor.



(4 times)


```
This is an example of
how the arrow keys
let you move the cursor.
```

RIGHT 

The RIGHT (right arrow) function key moves the cursor to the next character. When the cursor is after the rightmost character of the line, EDT moves the cursor to the first character on the next line.

The following diagrams illustrate cursor movement with the RIGHT key. The first frame shows the original cursor position, and the second frame shows what happens when you press the RIGHT key three times:

```
This is an example of
how the arrow keys
let you move the cursor.
```

 (3 times)

```
This is an example of
how the arrow keys
let you move the cursor.
```

10.5.2 Setting the Direction of Cursor Movement

ADVANCE and BACKUP change the direction in which the cursor moves. The following functions are affected by ADVANCE and BACKUP:

CHAR	WORD	FIND	FNDNXT	CHNGCASE
LINE	EOL	PAGE	SECTION	SUBSTITUTE
				TAB ADJUST

ADVANCE

ADVANCE sets the cursor direction forward. The cursor direction is toward the end of the text buffer, that is, to the right and down.

ADVANCE is the default direction of cursor movement. This default remains in effect until you press BACKUP.

BACKUP


BACKUP sets the cursor direction backward. The direction of cursor movement is toward the start of the text buffer, that is, to the left and up. When you select BACKUP, it remains in effect until you press the ADVANCE function key.

10.5.3 Movement by Entity

You can move the cursor by entity; the direction depends on your advance/backup mode status.

CHAR (Character)

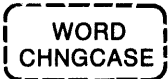
CHAR moves the cursor one character to the right or left. When necessary, the cursor moves up or down a line to move to the next character. For example:

Move the cursor.  Move the cursor.

The CHAR function is available with the VT100 only.

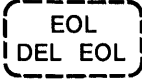
WORD

WORD moves the cursor one word forward or one word back. When the cursor is at the end of a line and the direction for the move carries off the end of the line, the cursor moves to the next line. For example:

Move the cursor.  Move the cursor.

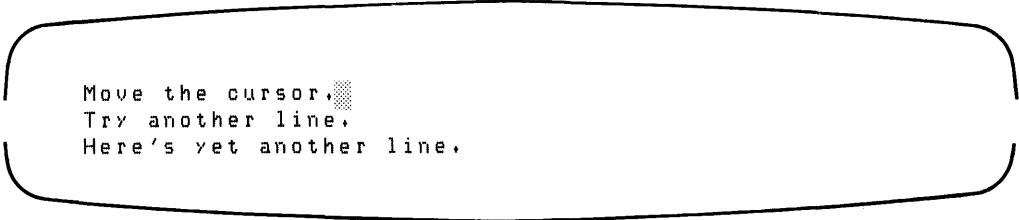
EOL (End Of Line)

EOL moves the cursor forward to the end of the current line, or backward to the end of the previous line. The cursor movement includes any blank spaces before the last line terminator. For example:

Move the cursor,  Move the cursor,

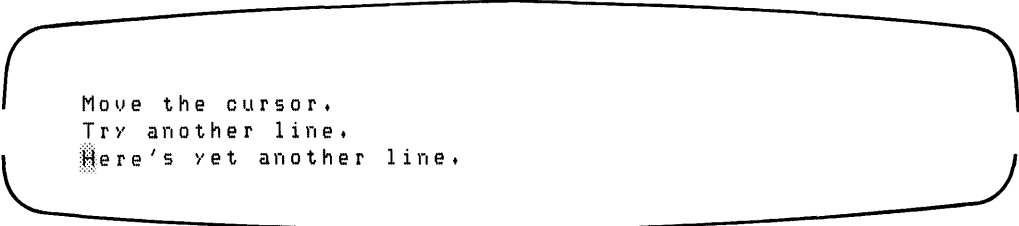
LINE

LINE moves the cursor down to the start of the next line or up to the start of the previous line (in other words, to the left hand margin). When the editor is in backup mode, the cursor moves to the beginning of the line the cursor is on. When you press the key again, the cursor moves to the beginning of the preceding line. For example:



Move the cursor,
Try another line,
Here's yet another line.

 (2 times)



Move the cursor,
Try another line,
Here's yet another line.

BACK SPACE

BACK SPACE places the cursor at the beginning of the line, that is, at the leftmost character position. For example:

Move the cursor,  Move the cursor,

10.5.4 Movement Throughout the Buffer

You can move the cursor to the top or bottom of the text buffer, or by sections and pages.

PAGE

PAGE moves the cursor to the top of the next page. You can define the page delimiter with the SET command. The default delimiter for a page is form feed, an ASCII character that determines the start of each line printer page. (Refer to Appendix C for a list of ASCII character codes.) The direction of cursor movement depends on your advance/backup mode status.

SECTION

SECTION moves the cursor one 16-line section. The direction depends on your advance/backup mode status.

TOP

TOP positions the cursor at the top of the text buffer. EDT displays text for as many lines as you set with the SET SCREEN command. (See Chapter 8 for a description of the SET SCREEN command.) If you do not set the number of lines to be displayed on the screen, the default of 22 lines appears.

BOTTOM

BOTTOM positions the cursor at the bottom of the text buffer. EDT displays text for as many lines as you set with the SET SCREEN command. (See Chapter 8 for a description of the SET SCREEN command.) If you do not set the number of lines to be displayed on the screen, the default of 22 lines appears.

10.6 Finding Text


The functions in this section let you locate strings of characters.

FIND

FIND prompts with "Search for:" on the lower left of your screen. Press GOLD and then the FIND function key to use this function. You enter the desired string through the keyboard and end the string either with an ADVANCE or BACKUP request to set the direction of search, or with ENTER to use the direction currently specified. For example:

A diagram illustrating the key sequence for the FIND function. It shows two dashed rectangular boxes. The first box contains the word "GOLD". To its right is a plus sign "+". The second box contains the words "FNDNXT" and "FIND" stacked vertically. To the right of the second box is the text "Search for:".

After the prompt, enter the text you want to find and set the direction of the search with ADVANCE or BACKUP:

Search for: word 

The string you enter in response to the FIND prompt is stored in the search string buffer. This buffer is also used for the FNDNXT, SUBSTITUTE, CUT, CHNGCASE, and REPLACE functions.

FNDNXT (Find Next)

FNDNXT searches for the string that is stored in the search buffer. For example, if the last search string you used in a FIND operation was “word”, pressing FNDNXT would cause a search for the next occurrence of “word”.

You can search forward or backward; the direction depends on your advance/backup mode status. EDT stores the search string in a buffer until you enter a new search string or exit EDT.

Examples:


You can use FIND and FNDNXT to search for occurrences of the word “the” in the following text:

Use the FIND function to locate the first occurrence of the search string, and use FNDNXT thereafter.

When you press GOLD and the FIND function key, you are prompted for a search string:

 +  Search for:

Type “the” and press ADVANCE:

Search for: the 

The cursor moves forward to the first occurrence of the search string:

Use the FIND function to locate the first occurrence of the search string, and use FNDNXT thereafter.

Pressing the FNDNXT function key moves the cursor to the next occurrence:

FNDNXT
FIND

Use the FIND function to locate the first occurrence of the search string, and use FNDNXT thereafter.

You can repeat the FNDNXT function:

FNDNXT
FIND

Use the FIND function to locate the first occurrence of the search string, and use FNDNXT thereafter.

If you want to search backward, press the BACKUP function key and then press FNDNXT:

BACKUP
TOP

+

FNDNXT
FIND

Use the FIND function to locate the first occurrence of the search string, and use FNDNXT thereafter.

10.7 Deleting and Reinserting Text

These functions let you delete text from a buffer and reinsert deleted text. Each of the functions that delete characters, words, or lines stores these entities in buffers for possible reinsertion. The functions are also useful for moving these entities of text about in the buffer.

10.7.1 Deleting and Reinserting by Character

DELETE

The DELETE key, which is located on the keyboard, lets you delete the character to the immediate left of the cursor. When the cursor is at the leftmost character position on a line, EDT deletes the line terminator to the left and moves the text on the line to the right of the text in the line above.

The following is an example of how to delete the characters preceding the cursor:



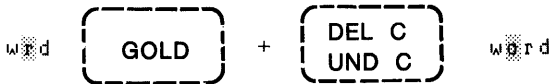
DEL C (Delete Character)

DEL C lets you delete the character that the cursor is on and stores it in the character buffer. For example:



UND C (Undelete Character)

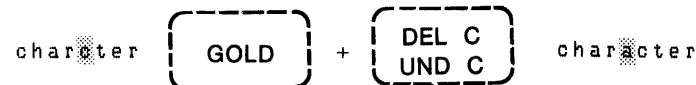
UND C reinserts the character that was deleted by the last DEL C or DELETE command. You are therefore able to replace a character that you accidentally deleted. For example:



DEL C and UND C are useful for correcting typographical errors. For example, you can use DEL C to delete a misplaced character:



Move the cursor to the character's proper location and then press GOLD and UND C:



10.7.2 Deleting and Reinserting by Word

LINE FEED

LINE FEED deletes the characters from the cursor back to the beginning of the word. EDT does not delete the character that the cursor is on. When the cursor is at the first letter in a word, EDT deletes the preceding word and the trailing spaces.

The LINE FEED function works as follows:

Here is a LINE FEED

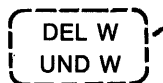


Here is a LINE

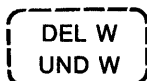
DEL W (Delete Word)

DEL W deletes the text up to the first character of the next word. Each DEL W operation clears the contents of the word buffer and inserts a new entry. For example:

Delete a very large word.



Delete a large word.



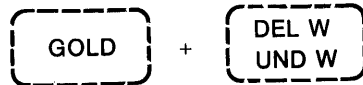
Delete a word.

The word buffer now contains the word “large”, which replaces the word “very” from the previous DEL W operation.

UND W (Undelete Word)

UND W inserts the contents of the word buffer in front of the cursor in the current buffer. The word buffer contains the text deleted by the last DEL W (keypad function) or LINE FEED (keyboard command). You are therefore able to replace a word that you accidentally deleted. For example:

Delete a word.

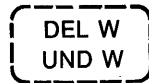


Delete a large word.

DEL W and UND W are especially useful when words are misplaced. You can delete a word, move it to the proper location, and undelete it as follows:

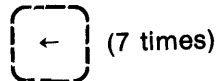
1. Delete the word with DEL W:

The first three applicants are Baker, Anderson, and Carter.



The first three applicants are Baker, and Carter.

2. Move the cursor where you want to reinsert the deleted word.



The first three applicants are Baker, and Carter.

3. Press GOLD and then the UND W function key.



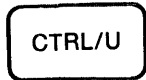
The first three applicants are Anderson, Baker, and Carter.

10.7.3 Deleting and Reinserting by Line

CTRL/U

CTRL/U deletes the text from the cursor position to the beginning of the line. When the cursor is at the leftmost character position on a line, EDT deletes the line above. EDT loads the deleted text into the line buffer.

This sentence is useless. This sentence is better.

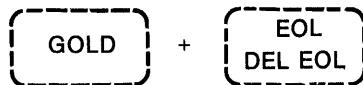


This sentence is better.

DEL EOL (Delete to End Of Line)

DEL EOL deletes all the characters to the right of the cursor, up to the end of the line, including the character on which the cursor is positioned. If you are at the end of a line, DEL EOL deletes the next line. The DEL EOL function works as follows:

Here is one line of
text for the example.

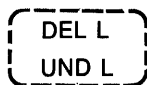


Here is
text for the example.

DEL L (Delete through end of Line)

DEL L deletes the text from the cursor position through the next line terminator. The first character of the following line moves up to the cursor.

Here is one line of
text for the example.



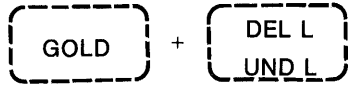
Here is text for the example.

UND L (Undelete Line)

UND L reinserts the text deleted by the last CTRL/U, DEL EOL, DEL L, or UND L command. You can therefore replace a line that you accidentally deleted.

The following example shows how UNDL restores the contents of the buffer after the DEL L function was performed:

Here is `text for the example.`



Here is `one line of text for the example.`

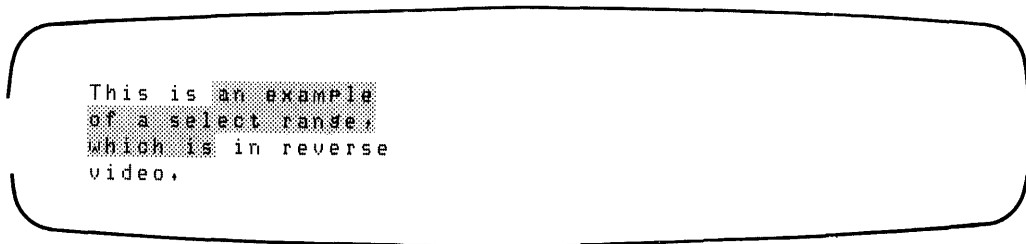
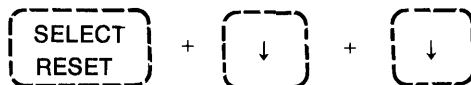
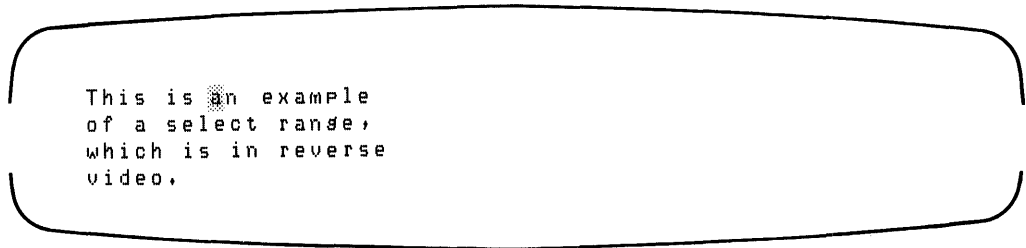
10.8 Selecting and Moving Text

This section describes how to use a select range for moving text about in a text buffer.

SELECT

SELECT lets you mark one end of a string for cutting, appending, or replacing. You mark one end of the string by pressing the SELECT function key. Then you move the cursor to the other end of the string and press the CUT, APPEND, or REPLACE function key. (The search string directly precedes, but does not include, the cursor position.)

On a VT100 you can see what characters are in the range you select because a VT100 select range appears in reverse video. The following occurs when you press the SELECT function key and move the cursor down two lines:



CUT

CUT deletes the selected string from the file and stores it in the paste buffer. This function is especially convenient when you want to delete or move large sections of text. The selected string is all the text between the cursor location when SELECT was pressed and the position the cursor was moved to.

The following example shows one of the uses of CUT:

1. Mark the first character to be CUT by pressing the SELECT function key.

SELECT
RESET

```
Here is some text.  
Here is some text to CUT and PASTE.  
This line stays here.  
This is the last line so far.
```

Then move the cursor with the WORD function key to the end of the string of characters:

WORD
CHNGCASE (3 times)

```
Here is some text.  
Here is some text to CUT and PASTE.  
This line stays here.  
This is the last line so far.
```

2. Press the CUT function key. The text disappears from the screen:

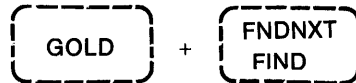
CUT
PASTE

```
Here is some text.  
Here is CUT and PASTE.  
This line stays here.  
This is the last line so far.
```

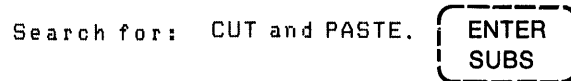
The text you cut stays in the paste buffer until you do another CUT operation or until you end the editing session. In the previous example, the words “some text to” are stored in the paste buffer.

If you do a search for some text and then press CUT, the search string is moved into the paste buffer.

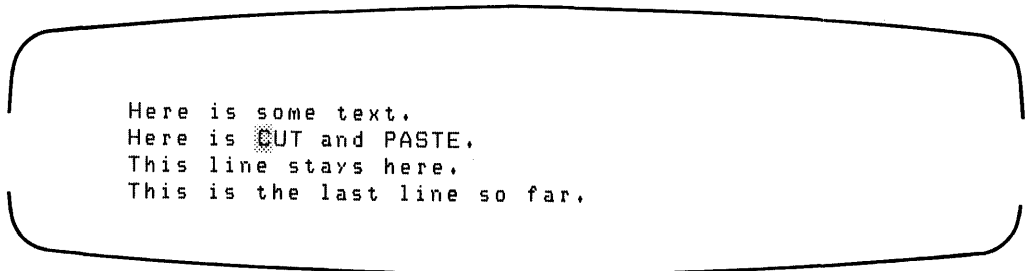
1. Press the GOLD key and the FIND function key.



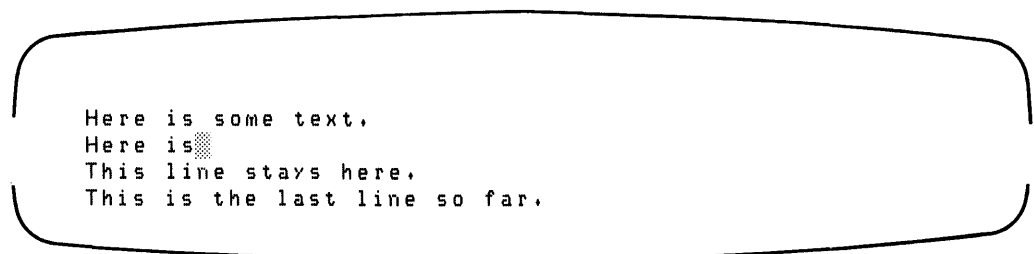
2. Type the search string after the prompt and press ENTER.



The cursor moves to the first character in the search string:



3. When you press CUT, the search string disappears from the screen.



The new search string (CUT and PASTE) replaces whatever was in the paste buffer.

PASTE

PASTE inserts the contents of the paste buffer in front of the cursor position.

If you wanted to cut the first line in the following example and move it down two lines, you would put the first line in a select range with the SELECT and LINE functions. You then press CUT:

```
Here is a line to CUT and PASTE.  
Here is some text.  
This line stays here.  
This is the last line so far.
```

SELECT + LINE
RESET OPEN LINE

```
Here is a line to CUT and PASTE.  
Here is some text.  
This line stays here.  
This is the last line so far.
```

CUT
PASTE

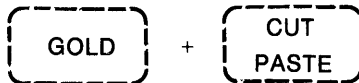
```
Here is some text.  
This line stays here.  
This is the last line so far.
```

Then you press the down arrow key twice, and press GOLD and the PASTE function key:



```
Here is some text.  
This line stays here.  
Here is a line to CUT and PASTE.  
█ This is the last line so far.
```

If you press GOLD and PASTE again, you duplicate the line:



```
Here is some text.  
This line stays here.  
Here is a line to CUT and PASTE.  
Here is a line to CUT and PASTE.  
█ This is the last line so far.
```

APPEND

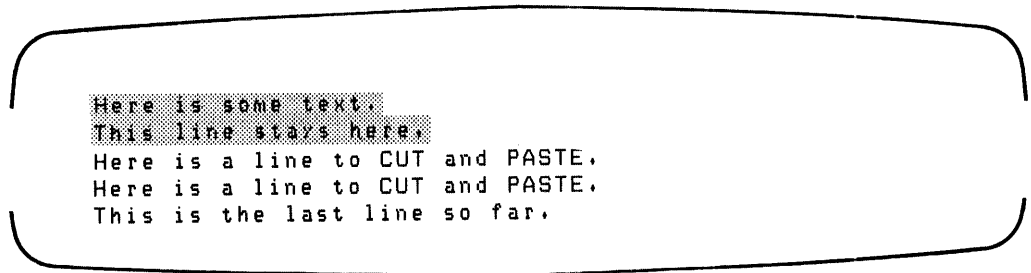
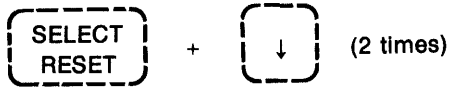
APPEND deletes the selected string from the current buffer and stores it at the end of the paste buffer. The selected string is the text between the SELECT entry and the cursor position at the time when you press APPEND.

For example, assume that you want to append the fourth line to the first two lines in the following example and move all three lines to the end of the text buffer:

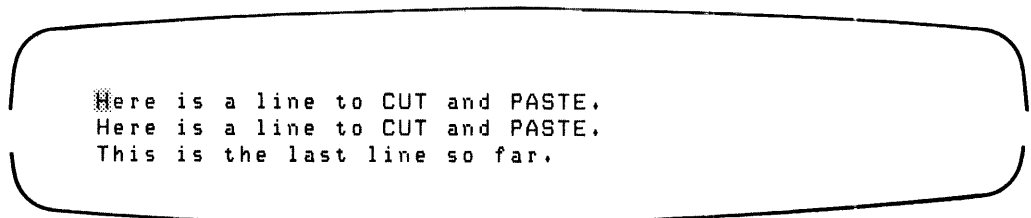
```
█ Here is some text.  
This line stays here.  
Here is a line to CUT and PASTE.  
Here is a line to CUT and PASTE.  
This is the last line so far.
```

The procedure to append is as follows:

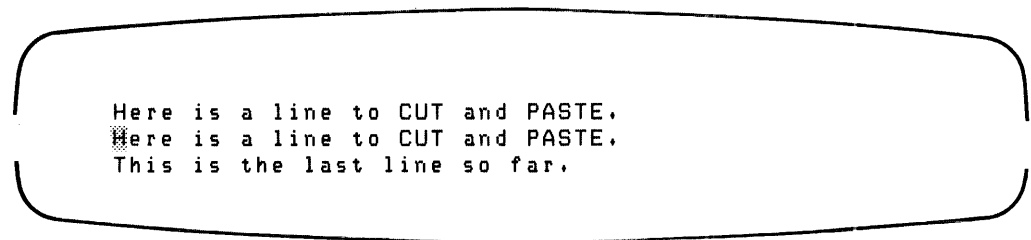
1. Mark the first two lines with a select range:



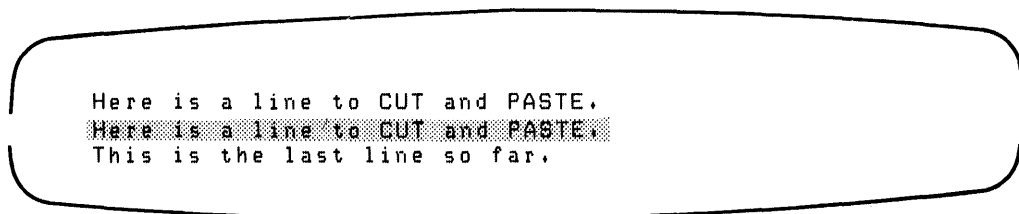
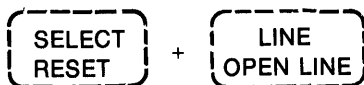
2. Put the contents of the select range into a paste buffer by pressing the CUT function key:



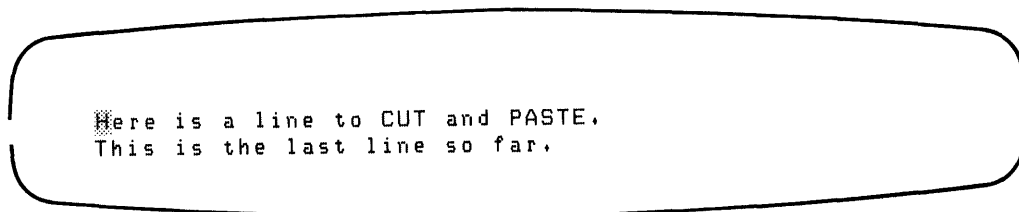
3. Move the cursor to the line that you want to append:



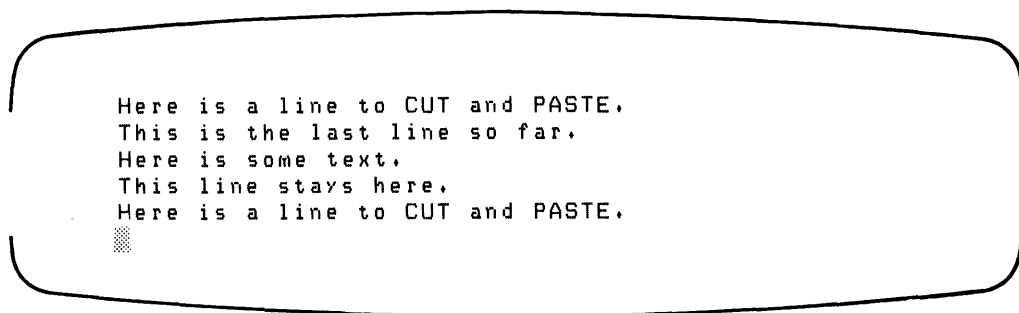
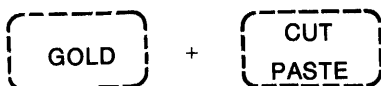
4. Put the line to be appended in a select range:



Now press the APPEND function key. The line disappears from the screen and is stored in a buffer. The text now reads:



5. Move the cursor to the end of the file and press the GOLD and PASTE function keys:



10.9 Replacing and Substituting Text

The following two functions, REPLACE and SUBS, are similar.

REPLACE

REPLACE deletes the text that you put in a select range and replaces it with the contents of the paste buffer. The following example shows how to replace the word "brown" with the words "red and white" in the following sentence:

The quick brown fox jumped over the lazy dog.

The steps are as follows:

1. Press the SELECT function key and type the words "red and white":

SELECT
RESET

red and white The quick brown fox jumped over the lazy dog.

2. Press CUT to store the phrase in a buffer:

CUT
PASTE

The quick brown fox jumped over the lazy dog.

3. Move the cursor to the word you want to replace and put this word in a select range:

WORD
CHNGCASE (2 times) + SELECT
RESET + WORD
CHNGCASE

The quick brown fox jumped over the lazy dog.

4. Press GOLD and the REPLACE function key. The sentence shows the change you made:

GOLD + APPEND
REPLACE

The quick red and white fox jumped over the lazy dog.

You can also use the FIND function in REPLACE operations. In the following example, the word "silver" replaces the strings "red and white" and "lazy".

1. Press the SELECT key and enter some text:



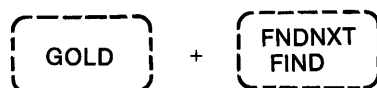
silverThe quick red and white fox jumped over the lazy dog.

2. Press CUT to place the text in a buffer:




The quick red and white fox jumped over the lazy dog.

3. Use FIND to locate the text you want to replace:



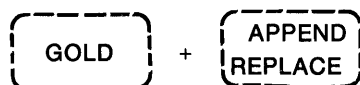
The FIND function causes a prompt to appear on your screen:

Search for: red and white 

The sentence reads:

The quick ed and white fox jumped over the lazy dog.

4. Press the GOLD key and the REPLACE function key to replace the phrase "red and white" with the contents of the PASTE buffer:



The quick silver fox jumped over the lazy dog.

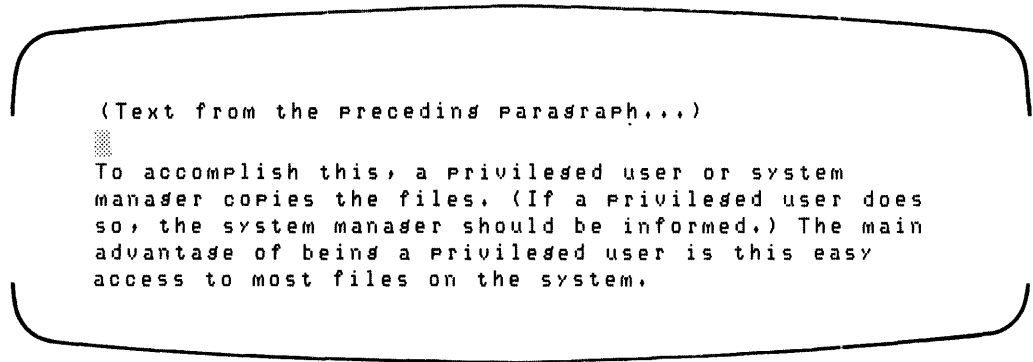
You can repeat the last two steps if you want to replace another word with the contents of the PASTE buffer. Use FIND to locate the word "lazy" and press GOLD and REPLACE again. The result is as follows:

The quick silver fox jumped over the silver dog.

SUBS

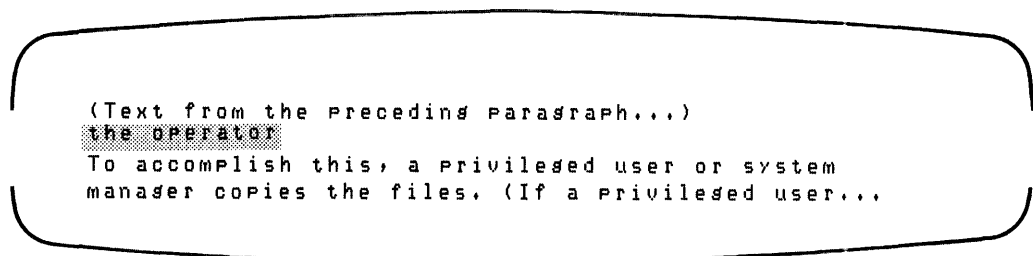
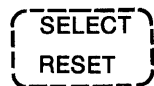
SUBS lets you find a string of characters and replace it with the contents of the paste buffer. SUBS works like the REPLACE function, except that the SUBS function automatically includes a “find next” operation.

The next example shows how to change several times the phrase “a privileged user” to “the operator” in the following frame:



To make this change, place the new phrase in a paste buffer, find the existing phrase, and substitute it throughout the paragraph. The steps are as follows:

1. Press the SELECT function key and type the words “the operator”. This puts the words you type in a SELECT range.



2. Press the CUT function key to store the words in a PASTE buffer. The words disappear from the screen. (The words are removed from their location in the file to the PASTE buffer.)



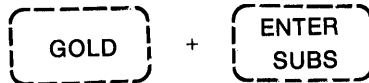
(Text from the preceding paragraph...)

To accomplish this, a privileged user or system manager copies the files. (If a privileged user...

3. Move the cursor to the first occurrence of the phrase "a privileged user" with the FIND function. You must use FIND (which places the phrase in a search buffer) in order for subsequent SUBS operations to work.

To accomplish this, a privileged user or system manager copies the files. (If a privileged user...

When you press GOLD and SUBS, the change is made and the cursor moves to the next occurrence of the phrase:



To accomplish this, the operator or system manager copies the file. (If a privileged user does so, the system manager should be informed.)

⋮

4. Continue to substitute the phrase throughout the paragraph by pressing SUBS. If you want to skip an occurrence of the phrase, press FNDNXT instead of SUBS.

10.10 Entering Commands

ENTER

ENTER lets you enter commands and do searches when you use the FIND and COMMAND functions.

For example, you press GOLD and the FIND function key to prompt for a search string:



When you type a string of characters, pressing ENTER saves the string and causes a search for it:



When you press the COMMAND function key and type in a command, pressing ENTER executes the command.

COMMAND

COMMAND lets you enter EDT command level and line editing commands. Press the COMMAND function key and type in these commands at the keyboard. To execute the commands, press the ENTER function key. For example:

1. Press the GOLD key and then the COMMAND function key. The COMMAND prompt appears at the bottom of the screen.



2. Type a command level command, which appears after the prompt:

```
COMMAND: SET NOTRUNCATE
```

3. Press the ENTER function key to execute the command.

Upon completion of a line editing command, the screen is updated as required and you return to keypad editing. If you enter a command such as EXIT or QUIT and press the ENTER function key, you end the EDT editing session.

10.11 Special Characters, Changing Case, and Filling Lines

These functions let you insert special characters, change characters to uppercase or lowercase, and perform a FILL operation on a section of text.

SPECINS

SPECINS lets you insert non-printing ASCII characters into your text. (See Appendix C for a list of ASCII character code decimal equivalents.) Press the GOLD key, enter the decimal representation of the ASCII character at the keyboard, and then press the SPECINS key.

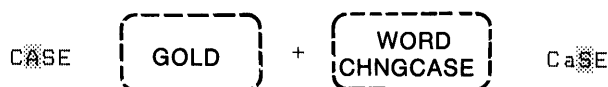
For example, suppose you wanted to insert a line feed character into the file. You would use the following procedure:

1. Press the GOLD key and type the ASCII numeric equivalent of the character. Line feed is ASCII 10.
2. Press the GOLD key and then the SPECINS function key.

SPECINS is the only way to insert a carriage return character without having it interpreted as a line terminator.

CHNGCASE

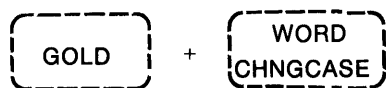
CHNGCASE lets you change uppercase alphabetic characters to lowercase, and the reverse. Press the GOLD key and the CHNGCASE function key to change the case of the character the cursor is on:



Note that the cursor moves one character to the right when you perform a CHNGCASE operation. This happens when the editing status is ADVANCE, which is the default. If EDT is in backup mode, the cursor moves one character to the left.

You can also change the case of text in a select range or a search string. If you press CHNGCASE when the cursor is on a search string or select range, the case of all these characters is changed:

Here is `an example of changing case.`



Here is `AN EXAMPLE OF changing case.`

FILL

FILL fills the selected range of lines to the limit of the line width. This function exists on the VT100 as a keypad key and as CTRL/F on the VT52.

For example, you may want to limit the width of a section of text to 20 characters. The FILL function allows the maximum number of characters on a line without breaking words:

1. Press the GOLD key and the COMMAND function key.
2. Type "SET WRAP 20" after the command prompt to limit the width of the text in the select range. Then press the ENTER function key.

3. **SELECT** the range of lines that should not exceed 20 characters in length. Use the **SELECT** function key and move the cursor to the end of the range of lines:

```
Here is some text across the page.  
This text will be no more than twenty characters  
wide. See how the FILL function works?  
Here, again, is the regular page width.
```

4. Now press the **GOLD** key and the **FILL** function key:

```
Here is some text across the page.  
This text will be no  
more than twenty  
characters wide.  
See how the FILL  
function works?  
Here, again, is the regular page width.
```

← 20 characters →

10.12 Control Character Functions

To use the control character functions, hold down the **CTRL** key while you press the alphabetic character key.

CTRL/C

CTRL/C ends the current operation and returns you to the keypad command level.

CTRL/U

CTRL/U deletes the text from the cursor position to the beginning of the line. When the cursor is at the leftmost character position on a line, **EDT** deletes the line above. **EDT** loads the deleted text into the line buffer.

CTRL/W

CTRL/W refreshes the screen display. The screen becomes blank and then the characters in the buffer reappear. Refreshing the screen deletes extraneous characters, such as from system messages or electronic mail, that might appear on the screen during your editing session.

CTRL/Z

CTRL/Z returns you to the line mode prompt (*).

CTRL/A

CTRL/A sets the tab position to the present cursor position. If the present cursor position is not a multiple of the SET TAB number, an error message occurs.

CTRL/E

CTRL/E increases the TAB level one count, where the count is set by the SET TAB command.

CTRL/D

CTRL/D decreases the TAB level one count, where the count is set by the SET TAB command.

CTRL/T

CTRL/T performs a TAB ADJUST operation on a select range. The characters in the select range move one tab stop to the right, unless you use a minus (-) character to move the select range one tab stop to the left. You can also use a repeat count for this function by pressing GOLD and a number from the keyboard before pressing the CTRL/T.

NOTE

You can also use the GOLD key on the keypad, instead of the CONTROL key on the keyboard, for the following functions:

CTRL/A	CTRL/E
CTRL/D	CTRL/U
CTRL/Z	CTRL/W
CTRL/T	

For more information on these keys, see the description of the SET TAB command in Chapter 8.

CTRL/K

CTRL/K lets you use the DEFINE KEY function. You can assign new functions to any of the keypad keys (except GOLD) and to all of the CONTROL keys, except CTRL/C, CTRL/Y, CTRL/Z, and CTRL/K itself. See the following description of DEFINE KEY for more information.

10.13 DEFINE KEY Command

The DEFINE KEY command assigns functions to the keypad keys and the control keys. You can redefine the value for all keypad keys (except GOLD) and most control keys, and you can define new control keys. During keypad character editing, use the CTRL/K function to define keys.

The DEFINE KEY command for the command level has the form:

DEFINE KEY {CTRL/x ; [GOLD]n} AS 'string'

where:

CTRL/x refers to any CONTROL key combination, such as **CTRL/S** or **CTRL/U** .

GOLD refers to the GOLD keypad key, which lets you access the alternate functions of the keypad keys.

n refers to the number of the selected keypad key. See Figure 10-3.

To define the standard (non-GOLD) function for a keypad key, enter the key number. To define the alternate function for a keypad key, press GOLD and the key number.

Each of the keypad keys and CTRL keys has a string of characters associated with it which define the function of the key. The string consists of nokeypad character editing commands. When you define keys with the DEFINE KEY command, the same rules apply as for nokeypad command strings:

1. A string can contain multiple commands.
2. A string ending in a period is executed immediately when you press the key associated with that string.
3. If a string does not end in a period, EDT buffers it (holds it for execution) until you enter a command that does end in a period.
4. You can precede a string with a question mark; the ? prompts the user for input.

Examples:

```
*DEFINE KEY 7 AS +D(RET)
*CHANGE(RET)
```

This example uses the DEFINE KEY command to define a string without an ending period. You define the standard function for keypad key 7 as delete (keypad key 7=<D>); you provide the entity to be deleted through additional key entries. EDT buffers the D function until you press a key with a definition ending in a period, such as a function key.

Figure 10-3: Keypad Numbers

GOLD	10	11	12
7	8	9	13
4	5	6	14
1	2	3	15
0	16	21	

A. VT52 Keypad

12	13	15	14
-----------	-----------	-----------	-----------

GOLD	10	11	17
7	8	9	18
4	5	6	19
1	2	3	21
0	16		

B. VT100 Keypad

```

This is line one.
This is line two.
This is line three.
.
.
.

```

```

D      WORD
COMMAND + CHNGCASE

```

```

This is one.
This is line two.
This is line three.
.
.
.

```

The cursor is positioned at the "l" in line one. You press function keys <D> and <WORD>. (Nokeypad character editing command W. is the definition of function key <WORD>.) EDT buffers +D until you press <WORD> and then executes +D W. as a single command.

```

*DEFINE KEY GOLD 1 AS ?'This is a special cmd:'D+3W L TAB(RET)
*CHANGE(RET)

```

The alternate function for keypad key 1 is redefined with the ? character followed by a string enclosed in single quote marks. When you use the redefined key, the quoted string is displayed at the bottom of the screen as a prompt. Your message can be a reminder of the special function you have assigned to the key. The command string does not end in a period and therefore the command string is buffered until you press a key whose function ends with a period. The command string deletes 3 words, moves down one line, and sets the tab for that line.

This prompting feature is used in a slightly different form in the FIND function, which is defined as:

```

"? 'Search for:'. '

```

Here, the quoted string prompts you for an input which can be up to 64 characters long. You end your search string input by pressing any keypad key.

The double quote marks are the nokeypad character editing form of the string search command.

The DEFINE KEY command for the keypad editing level has the form:

CTRL/K

EDT responds with:

Press the key you wish to define

You select the key by pressing it.

EDT responds with:

Now enter the definition terminated by ENTER

You enter the function definition using the rules for string entries. Enter the definition, either using the nokeypad character editing commands or pressing the keypad function keys. Your entries for the function appear at the bottom of the screen. Press the ENTER function key when your command string is complete. Use the SHOW KEY command to display the definition of the key.

Use DEL to edit errors in the commands you type. You can use CTRL/U to cancel the DEFINE KEY sequence.

You may use control characters in your key definition. When you press a control key, the circumflex and control characters are inserted into the command string (CTRL/Z is entered as ^Z). EDT inserts the circumflex character combination for keys that generate control characters, such as RET and LF, CTRL/M and CTRL/J respectively).

Examples:

Assume you are not a very good typist and frequently transpose characters. To fix these errors manually, you delete the first character, move forward one character, and retype the deleted character. After making another typing error you decide to use the DEFINE KEY function to make the correction operation automatic.

This sentence has a trasnposed letter.

CTRL/K

Press the keypad key to be defined:

GOLD

5

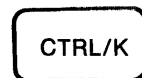
Then enter your definition, ending it with ENTER:



You enter CTRL/K and EDT prompts you with “Press the key you wish to define”. You press the keypad key GOLD/5, and EDT prompts you with “Now enter the definition and end with ENTER”. When you press keys to define GOLD/1, the nokeypad editing values for those keys appears at the bottom of the screen. The finished string is D+C+CUNDC., which disappears from the screen when you press ENTER.

Move the cursor to the “s” in “trasnposed” and press GOLD/1. The “s” and the “n” are transposed.

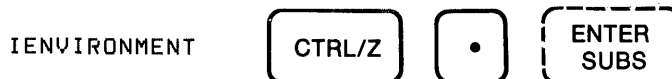
The next example shows how you use a defined key to insert a frequently used word or phrase.



Press the key you wish to define



Now enter the definition terminated with ENTER:



The finished string for your entry is IENVIRONMENT[^]Z., where the CTRL/Z is displayed as [^]Z. The CTRL/Z, which ends the insert operation, could be entered by typing [^]Z. Enter [^]Z when using the command level DEFINE KEY command.

When you want to insert “ENVIRONMENT”, press GOLD/1

The next example shows how to define two keys to do multiple string substitutions in keypad character editing. This definition uses the ? to prompt you for both the search string and the substitute string. You define one key to accept the search and substitute strings and do the first substitution and another key to substitute for the next occurrence of the search string.

CTRL/K

Press the key you wish to define

CTRL/F

Now enter the definition terminated by ENTER

S/?'SUBS: '/?' WITH: '/"'

ENTER
SUBS

CTRL/K

Press the key you wish to define

CTRL/R

Now enter the definition terminated by ENTER

SN" ",

ENTER
SUBS

The first definition is a substitute command, and EDT prompts you for the two strings when you press **CTRL/F**. The pair of double quotes at the end of the command string causes EDT to search for the next occurrence of the string. If you enter a slash in either string, the command is invalid; the string is delimited by the slash. To avoid this, replace the slashes in the definition with a character you are not likely to use in the strings, such as a control character.

The second definition is a substitute next command; it replaces the next occurrence and searches for another one.

If you define a key as a series of commands and wish to use a repeat count with the key, enclose the series of commands in parentheses. EDT builds a command from your input and interprets your input when you press the key.

+L I*^Z.

The function defined for this key is 'move to the beginning of the next line and insert an asterisk.' Now suppose you used a repeat count of 10 to insert asterisks at the beginning of the next 10 lines. The command EDT sees after you press the keys is:

10+L I*^Z.

The repeat count of 10 applies only to the +L command, so EDT moves forward 10 lines and then inserts the asterisk. This is not what you had in mind. Now change the definition to:

```
10(+L I*^Z).
```

The repeat count now appears in front of the left parentheses and the entire string inside the parentheses is executed ten times. This has the desired effect of putting an asterisk at the front of each of the next ten lines.

Appendix A

Differences in Terminal Responses

EDT is an interactive editor; it responds to your commands as you enter them. EDT can be used with three types of interactive terminals:

Hardcopy terminals

Display terminals without a keypad

Display terminals with a keypad

A general description of the display responses to the editor commands on the three terminal types follows.

A.1 Hardcopy Terminals

You can designate any interactive terminal as a hardcopy terminal through the SET TERMINAL HCPY command. The hardcopy terminals (such as the LA36) respond to the command level commands and to the nokeypad commands. In line mode, the hardcopy terminals respond in a manner similar to display terminals (except for deleting characters). When you press `DEL`, the terminal responds with a backslash and repeats the last character entered, the character being deleted. When you make additional deletions before new entries, the characters are repeated without the backslash. When you make a new entry, the editor responds with another backslash and the new data. For example:

```
errir\ri\or
```

All line mode commands are entered at the command level after the * prompt. When you enter line editing commands, they are displayed immediately after the asterisk and executed when you end the command line with a `RET`. When the command is complete, EDT responds with another asterisk; the editor is ready to execute another command. Each time you enter a `RET`, EDT returns to command level.

When you enter change mode on an LA36, the cursor is a pair of brackets. The cursor encloses a character to mark your position in the text buffer. When the first entry is a tab character (eight characters spaces), the brackets enclose seven blanks ([]). When the first one to seven characters are empty, the brackets enclose one blank ([]). When the position is the end of a line, the brackets enclose the carriage return symbol [<CR>].

To exit the change mode, enter EX.

Example:

```
*12(RET)
  12          Commands follow:
*14(RET)
  14          Hardcopy Terminals
*CJAMGE\EGMAJ\HANGE(RET)
[H]ardcopy Terminals
C*L(RET)
[<CR>]
C*L(RET)
[T]he hardcopy terminals can respond to all of the command level
C*4W(RET)
The hardcopy terminals can [r]espond to all of the command level
```

In the first line of the example, EDT is at command level; the asterisk (*) prompt is present. When you enter a line number, EDT responds by displaying the lines, lines 12 and 14.

You try to enter change mode but misspell the command. To correct the spelling error, you press DEL once for each letter to be deleted and then enter the correct characters. EDT responds by displaying the deleted letters between backslashes.

When you enter the CHANGE command, EDT displays the current line and the position of the cursor on that line, the first character. The change mode prompt is C*.

When you enter the L command, EDT displays the next line. The [<CR>] indicates that the line is empty. The next line contains text; EDT positions the cursor at the first character in the line. In response to your 4W command, EDT displays the current line with the cursor at the beginning of the fourth word following the current word.

A.2 Nokeypad Display Terminals

Both the VT52 and the VT100 terminals can be used in the nokeypad mode. In nokeypad mode the commands are entered through the keyboard.

The VT52 cursor is a blinking underline (—) and the VT100 cursor is either a blinking underline or a rectangle (■) with the character in reverse video. The

cursor marks your position in the text buffer for entering change mode commands. To insert data, you enter the I command. EDT inserts the data just before the cursor. With each character entry, the cursor moves to the right. At the end of the line, it moves to the start of the next line. To exit the insert operation, enter a `CTRL/Z`.

All line mode commands are entered at the command level prompt (*). When you enter line editing commands, EDT displays them immediately after the asterisk and executes them when you end the command line with a `RET`. When the command is complete, EDT responds with another *; EDT is ready to execute another command. For those commands which require additional input, the INSERT and the REPLACE commands, you enter a `CTRL/Z` to return to the command level.

When you enter a command, the cursor moves to the bottom of the screen and EDT displays the command you enter. When you end the command line with `RET`, EDT performs the command and returns the cursor to the correct position in the text.

A.3 Keypad Display Terminals

Both the VT52 and the VT100 terminals can be used in the keypad mode. The examples in the manual are based on a VT100 terminal unless otherwise specified.

Appendix B

Error Messages

EDT contains a set of error messages that identify problems and assist you in the completion of the present editing operation.

Most error messages consist of a pointer line, which is a circumflex character (^), and the error message. The pointer indicates the position of the error in the command.

The error messages, arranged alphabetically, and their explanations follow.

`'.' required`

The command requires a period (.) or other special character shown enclosed in single quote marks.

`Aborted by CTRL/C`

This message occurs when you are in keypad change mode and enter a `CTRL/C` after entering a command.

`Advance past bottom of buffer`

The command indicates that you attempted to move the cursor past the [EOB].

`Attempt to CUT or APPEND to current buffer`

This message occurs when you have tried to perform a CUT or APPEND and have named the current text buffer as the destination text buffer. The default destination is the paste text buffer.

`Attempt to PASTE the current buffer`

The PASTE command attempted to paste text into the same text buffer containing the text to be pasted.

`Backup Past top of buffer`

The command entered would move the cursor to a position preceding the first line (top) of the buffer.

`Change mode may be entered only from the terminal`

You cannot enter change mode when you run EDT from a batch command file or from your startup command file. There is no terminal associated with this job, so you cannot enter change mode.

`Command buffer exhausted`

The string of change mode commands exceeds 255 characters.

`Command file could not be opened`

The command file given in the command line cannot be opened. EDT will also display an associated RMS message for the error.

`Command file does not exist`

The command-file parameter in the command line does not exist in the specified directory.

`Consistency check failed, please check your file`

There is a discrepancy between the number of lines and characters entered during the editing session and the number of lines and characters present when the editing session ends. You should check for possible errors in your output file. This message indicates that there is a problem in EDT.

`Could not align tabs with cursor`

The cursor is at a position where it is not evenly divisible by the tab size when the tab compute function was used.

`Destination for MOVE or COPY not found`

The range specification in the command does not exist.

`Entity must be WORD, SENTENCE, PAGE, or PARAGRAPH`

The SET ENTITY command must include one of the four entity options listed.

`Error in command`

The command entered is invalid.

`Error in command option`

The command includes a /name where name is not a valid option.

`Error in range specification`

The command requires a range specification which must be complete.

File specification required

A file specification is required as a part of the command (WRITE, PRINT, or INCLUDE).

Help file could not be opened

The requested help file cannot be accessed. This indicates an error in system storage (for example, improper installation of the operating system).

Input file could not be opened

The command contains faulty syntax, or you specified a nonexistent directory.

Input file does not exist

The input file is not contained in the specified directory.

Input record too large, truncated to 255 characters

A record in the input file exceeds 255 characters.

Insufficient memory

There is insufficient memory to complete the last command. This message can occur when you define a new text buffer or use the DEFINE KEY command.

Invalid buffer name

You have used improper syntax for the buffer name in the command.

Invalid entity

The entity portion of the change mode command is not recognized.

Invalid option for that command

You have used an /OPTION where it is not allowed for that command.

Invalid parameter for SET or SHOW

The SET or SHOW command does not use one of the listed parameters (see Chapter 8).

Invalid subcommand

You have used an improper name for the change mode command.

Invalid value in SET command

The command has an invalid keyword.

I/O error on work file

EDT is unable to access its text storage area for the file. An additional message will explain the error further, such as "Device full" or "Device write locked".

Journal file could not be opened

The journal file is not within your defined privilege.

Line exceeded 255 characters, truncated

The input for the line exceeds 255 characters; the excess is deleted.

MACRO or KEY required

The DEFINE command is incomplete. You must include either MACRO or KEY in the command.

No definition

You have requested a SHOW KEY definition for an undefined key.

No output file name

You have used the EXIT command without having specified a file name, either in the EXIT command or in the command line.

No select range active

You did not create a select range prior to entering the APPEND or CUT command.

No such line

There are no original line numbers for the specified range.

Numeric value required

The command must have a numeric value at the point of the ^ in the command.

Output file could not be opened

EDT will display another message describing the error.

Parenthesis mismatch

The number of right hand parentheses () do not match the number of left hand parentheses [(]. This error occurs when you are entering a change mode command string.

Parsing stack overflow

The command has caused the memory space for the parse data to be filled before the command could be validated. Check your command string. If it is valid, reenter the command in segments.

Please answer Y(es), N(o), Q(uit), or A(11)

This is the prompt that occurs when you have selected the query qualifier and have failed to answer with one of the above in response to the ? prompt.

`Quoted string required`

The command requires a quoted string. The ^ indicates the position of the required quoted string.

`Range must be contiguous`

The range specification for the RESEQUENCE command must be contiguous lines.

`Range specified by /SEQUENCE would cause duplicate or non-sequential Numbers`

You have range specifications in the RESEQUENCE command that would cause duplicate or non-sequential numbers.

`String was not found`

The string defined in the range specification cannot be found.

`That key is not definable`

The key selected for the DEFINE KEY command is not available for definition.

`Unexpected characters after end of command`

The command contains a string of one or more characters at the end of the command which are not part of the command. The rest of the command is valid.

`Unrecognized command`

EDT does not recognize or support the command entered. Most likely you have incorrectly specified the command.

`Unrecognized command option`

The command includes an invalid option/qualifier.

`Work file overflow`

You have exceeded 65536 blocks of text in this editing session.

`Working`

This message is displayed when the command operation requires more than a minimum amount of time to complete. It tells you that EDT is responding to your command.

Appendix C

ASCII Decimal Equivalents

Decimal Value	ASCII Character	Decimal Value	ASCII Character	Decimal Value	ASCII Character
0	NUL	43	+	86	V
1	SOH	44	,	87	W
2	STX	45	-	88	X
3	ETX	46	.	89	Y
4	EOT	47	/	90	Z
5	ENQ	48	0	91	[
6	ACK	49	1	92	\
7	BEL	50	2	93]
8	BS	51	3	94	^ or †
9	HT	52	4	95	— or ←
10	LF	53	5	96	` Grave accent
11	VT	54	6	97	a
12	FF	55	7	98	b
13	CR	56	8	99	c
14	SO	57	9	100	d
15	SI	58	:	101	e
16	DLE	59	;	102	f
17	DC1	60	<	103	g
18	DC2	61	=	104	h
19	DC3	62	>	105	i
20	DC4	63	?	106	j
21	NAK	64	@	107	k
22	SYN	65	A	108	l
23	ETB	66	B	109	m
24	CAN	67	C	110	n
25	EM	68	D	111	o
26	SUB	69	F	112	p

(Continued on next page)

Decimal Value	ASCII Character	Decimal Value	ASCII Character	Decimal Value	ASCII Character
27	ESC	70	G	113	q
28	FS	71	H	114	r
29	GS	72	I	115	s
30	RS	73	J	116	t
31	US	74	K	117	u
32	SP	75	L	118	v
33	!	76	M	119	w
34	"	77	N	120	x
35	#	78	O	121	y
36	\$	79	P	122	z
37	%	80	Q	123	{
38	&	81	R	124	Vertical Line
39	"	82	S	125	}
40	(83	T	126	~Tilde
41)	84	U	127	DEL RUBOUT
42	*	85			

Glossary

alphanumeric

A contraction of alphabetic-numeric; the set of characters that compose text; characters include letters, numerals, and exclude special characters.

ASCII

The acronym for American Standard Code for Information Interchange; a standardized code representing the 128 characters in which text is recorded. The decimal values for the ASCII characters are contained in Appendix C.

authorization file

See user authorization file.

buffer

A temporary storage area used to contain text.

case

The state of the alphabetic characters. Capital letters are upper case.

character

A symbol representing an ASCII code. See also alphanumeric character.

character buffer

A temporary storage area used to store the last character deleted by an EDT delete character operation.

character editing

Editing text at the character level; an operational mode of EDT, entered through the CHANGE command.

character string

A sequence or group of connected characters.

command

An instruction typed by the user at a terminal or included in a command file which requests the operating system to perform some well-defined procedure.

command file

A file containing command strings.

command line

The command for entering EDT.

command string

A line (or set of continued lines), normally terminated by typing a **RET**, containing a command and (optionally) information modifying the command. The fullest form of the command string contains a command, its qualifiers, and its parameters (file specifications, for example) and their qualifiers.

contiguous

Contiguous lines or characters are those which are adjacent to one another.

control key

The keyboard character that causes a control action; a control key is usually the combination of the CTRL key and an alphabetic key.

CTRL/A or GOLD/A

Sets the tab position to the present cursor position.

CTRL/C

End the current keypad operation and returns you to the keypad command level.

CTRL/D or GOLD/D

In keypad character editing, decreases the tab indentation level count by one.

CTRL/E or GOLD/E

In keypad character editing, increases the tab indentation level count by one.

CTRL/I

Duplicates the function of the TAB key.

CTRL/K

In keypad character editing, lets you use the DEFINE KEY function.

CTRL/Q

Restarts terminal output that was suspended by **CTRL/S**.

CTRL/S

Suspends the terminal output until you press **CTRL/Q**.

CTRL/T or GOLD/T

In keypad character editing, sets the tab indent over a range of lines.

CTRL/U or GOLD/U

In keypad character editing, deletes the text from the cursor position to the left margin.

CTRL/W

In keypad character editing, refreshes the screen display.

CTRL/Z

Exits you from insert mode or keypad character editing and returns you to EDT command level.

decimal number

A number in the numbering system with a base of 10; a numbering system composed of 10 possible symbols (0 through 9) with each number position representing that symbol times some power of 10.

default

The act of omitting information in a computer operation so that the computer makes a predefined assumption about the operation; an assumption made by the operating system when no specific value is provided by the user.

dellimiter

A character that limits a string of characters and therefore cannot be a member of the string.

directory

A file used to locate files on a mass storage device; a list of file names (including type and version number) and their unique internal identifications.

directory name

The field in a file specification that identifies the directory in which a file is listed. The directory name begins with a left bracket and ends with a right bracket.

direction

Direction can be forward or backward; the symbol for forward direction is the plus sign (+) and the symbol for backward direction is the minus sign (-).

editing session

The interval between entering and exiting EDT.

entity

Units of text from 1 to n characters on which EDT commands operate; for example, character, word, line.

error message

A message displayed by EDT indicating that EDT cannot perform the operation you requested.

exit

The means of halting a computer cycle of operation (EDT); the ending of an editing session. When capitalized, it is a command.

field

A group of characters which can be treated as a single unit of information, for example, line numbers.

file

A set of data stored on a mass storage device.

file name

The field preceding a file type in a file specification. This field contains a 1- to 9-character name for a file.

file specification

A unique name for a file on a mass storage device. It identifies the node, the device, the directory name, the file name, the file type, and the version number under which a file is stored.

fixed line numbers

Line numbers fixed to lines of text in a file. EDT maintains a record of these line numbers during the editing sessions in which you use those files. The line numbers are copied to the file when you end the editing session.

form feed (FF)

Moves the cursor position to the start of a new page. One of the default word delimiters; the default page delimiter.

GOLD key

A key on the keypad. When you press GOLD, the alternate function for the next keypad key you press is enabled.

help facility

A set of on-line messages describing the operation of EDT.

increment

To add a quantity to another quantity; the quantity added.

integer

A whole number containing no fractional or decimal part.

journal file

A file containing the data input to the terminal for one editing session.

journaling

The recording of your input during an editing session.

language

A systematic, unambiguous means of communication with a computer; a set of representations, conventions, and associated rules used to convey information.

line buffer

A storage area used to store the last line deleted by an EDT delete line operation.

line feed (LF)

Moves the cursor position down one line. One of the default word delimiters. In keypad character editing, deletes the characters from the cursor position to the left word delimiter.

line number

A number used to identify a line of text in a file or in an EDT text buffer.

log in

The process of accessing a computer system; an identification procedure.

macro

A statement that requests EDT to perform a predefined set of instructions.

main text buffer

The default text buffer for keyboard input and for input files, and the source for output files.

memory

A device on which data can be stored and from which it can be retrieved; internal computer storage.

nested parentheses

A parenthetical operation embedded within another parenthetical operation in an expression.

noncontiguous lines

Lines which are not adjacent to one another.

nonprinting character

A character in the computer code set for which there is no corresponding graphic symbol.

null

The character with the ASCII code 000; an absence of information.

null string

A string without content; an empty string represented by adjacent quotation marks.

operating system

Software that controls the execution of computer programs and performs system functions; an integrated collection of programs that supervise computer operation.

parameter

The object of a command. A parameter can be a file specification or a keyword option.

password

The character string assigned to a user to verify the user's access privileges; a code for assuring file confidentiality on time-sharing systems.

paste buffer

The default text buffer for EDT cut and paste operations.

program

The complete sequence of instructions, text, and routines necessary for the computer to perform a desired operation.

programming

The process of planning, writing, testing, and correcting the steps required for a computer to solve a problem or perform a desired operation.

prompt

A symbol indicating that the user must provide input.

punctuation

Special language characters such as commas, separators, etc.

qualifier

A keyword that modifies the operation of a command. Qualifiers are always preceded by slash (/) characters.

range of lines

The number of lines in a range specification; the range can define single or multiple lines and contiguous or noncontiguous lines.

range specification

A means for defining a string of text.

refresh

To update the display with the most current text for the range displayed.

rubout

Synonymous with delete.

screen

The display surface on a display terminal.

screen width

The number of character positions that can be displayed on a line.

search string

A group of characters you define in a command; the object of a search operation.

single line ranges

Ranges which address one line of a text buffer.

string

A set of contiguous items of a similar type; a connected sequence of characters.

string search buffer

A buffer used to store the string being searched for.

substitute buffer

A buffer used to store the string to be substituted for the search string.

symbol

An identifier used to represent a value; a character or group of characters of a language that are used to represent another entity.

syntax

The rules governing command structure in a computer language; the structure of the language.

syntax error

A mistake in your use of the governing command structure.

TAB

Moves the cursor a number of character positions. The default number of positions is eight. One of the default word delimiters.

system

A combination of hardware and software that performs specific processing operations; a collection of components that forms a functional unit.

terminal

The general name for those peripheral devices that have keyboards and video screens or printers. Under program control, a terminal enables you to type commands and text on the keyboard and receive messages on the video screen or printer.

text buffer

An EDT storage area for text (either terminal input or file input).

truncate

To set a limit on the number of characters in a line; characters entered after the limit is reached are used to start a new line.

user authorization file

A file containing an entry for every user that the system manager authorizes to gain

access to the system. Each entry identifies the user name, password, default account, User Identification Code, quotas, limits, and privileges assigned to individuals who use the system.

user name

The name that a person types on a terminal to log in to the system.

value

A quantity; the information represented by a data item.

variable

An entity that can assume any of a given set of values; a symbol whose value can change during a program.

version number

The field following the file type in a file specification. It is separated from file type by a period (.) or semicolon (;) and consists of a number that generally identifies it as one version among all files having the identical file specification except for version number.

Index

- ^ command
 - description of, 9-11
 - purpose of, 9-11
- A**
- ADV command
 - purpose of, 9-4
- ADVANCE function
 - description of, 10-16
 - purpose of, 10-16
- APPEND command
 - description of, 9-13
 - purpose of, 9-13
- APPEND function
 - description of, 10-27
 - example, 10-28
 - purpose of, 10-27
- ASC command
 - description of, 9-9
 - example, 9-9
 - purpose of, 9-9
- Authorized user file, 2-2
- B**
- BACK command
 - description of, 9-4
 - purpose of, 9-4
- BACKSPACE function
 - description of, 10-15
 - purpose of, 10-15
- BL entity, description of, 9-2
- BOTTOM function
 - description of, 2-21, 10-16
 - example, 2-22
 - purpose of, 10-16
- BPAGE entity, description of, 9-3
- BPAR entity, description of, 9-2
- BR entity, description of, 9-3
- BSEN entity, description of, 9-2
- Buffer line ranges, example, 4-5
- BW entity, description of, 9-2
- C**
- C entity, description of, 9-2
- CHANGE command
 - description of, 1-2, 8-1
 - example, 2-14, 2-15, 2-19, 8-2, 10-3
 - purpose of, 8-1
- Changing case, how to, 2-31. *See also*
CHNGCASE function
- CHAR function
 - description of, 10-14
 - purpose of, 10-14
- Character buffer
 - purpose of, 5-3
 - using, 5-3
- Character editing
 - deleting characters, 2-16
 - deleting lines, 2-15
 - inserting text, 2-16
 - keypad, 1-3, 10-1
 - moving text, 2-17
 - moving the cursor, 2-14
 - nokeypad, 1-3, 2-14, 9-1
 - substituting text, 2-17
- CHNGCASE function
 - description of, 10-35
 - example, 2-31, 10-35
 - purpose of, 10-35
- Command
 - ^, 9-11
 - ADV, 9-4
 - APPEND, 9-13
 - ASC, 9-9
 - BACK, 9-5
 - CHANGE, 8-1
 - COPY, 1-2, 7-1
 - CUT, 9-13
 - D, 9-14
 - DEFINE KEY, 10-38
 - DEFINE MACRO, 1-2, 7-2
 - DELETE, 1-2, 2-10, 7-4
 - EX, 9-5
 - EXIT, 8-2
 - EXT, 9-5
 - FILL, 9-15
 - FIND, 1-2, 7-5
 - HELP, 8-3
 - I, 9-6
 - INCLUDE, 1-2, 7-6
 - INSERT, 1-3, 2-4, 7-7
 - INSERT END, 2-8
 - MOVE, 1-3, 7-7
 - Null, 1-3, 7-7, 9-16
 - PASTE, 9-17
 - PRINT, 1-3, 7-8
 - QUIT, 8-4, 9-6
 - R, 9-17

REF, 9-7
 REPLACE, 1-3, 2-7, 7-8
 RESEQUENCE, 1-3, 2-11, 7-9
 S/s1/s2/, 9-12
 SEL, 9-7
 SET, 8-4
 SET CASE, 8-5
 SET CURSOR, 8-5
 SET ENTITY, 8-6
 SET KEYPAD, 8-7
 SET LINES, 8-7
 SET MODE, 8-7
 SET NOKEYPAD, 8-2
 SET NUMBERS, 8-8
 SET QUIET, 8-8
 SET SCREEN, 8-8
 SET SEARCH, 8-9
 SET TAB, 8-9
 SET TERMINAL, 8-11
 SET TRUNCATE, 8-11
 SET VERIFY, 8-13
 SET WRAP, 8-13
 SHL, 9-10
 SHOW, 8-14
 SHOW BUFFER, 8-14
 SHOW CASE, 8-15
 SHOW CURSOR, 8-16
 SHOW ENTITY, 8-16
 SHOW KEY, 8-16
 SHOW SCREEN, 8-16
 SHOW SEARCH, 8-16
 SHOW TERMINAL, 8-16
 SHOW VERSION, 8-16
 SHR, 9-10
 SN, 9-12
 SUBSTITUTE, 1-3, 7-11
 SUBSTITUTE NEXT, 1-3, 7-13
 TAB, 9-7
 TADJ, 9-18
 TC, 9-9
 TD, 9-11
 TOP, 9-9
 TYPE, 1-3, 2-7, 7-14
 TYPE WHOLE, 2-8
 UNDC, 9-11
 UNDL, 9-11
 UNDW, 9-11
 WRITE, 1-3, 7-16
 Command format
 action on an entity, 9-4
 fixed, 9-4
 variable count and direction, 9-4
 COMMAND function
 description of, 10-34
 example, 2-31, 10-34
 purpose of, 10-34
 Command level prompt, EDT, 2-1
 Command line, EDT, 2-3, 6-1
 Commands
 character editing, 9-1, 10-1
 control command, 1-2, 8-1
 EDT, 1-2
 line editing, 1-2, 7-1
 nokeypad, 9-3
 Consistency check
 description of, 3-3
 purpose of, 3-1, 3-3
 Contiguous line ranges, example, 4-4
 COPY command
 description of, 1-2, 7-1
 DUPLICATE, 7-2. *See also qualifier*
 example, 7-2
 examples of, 2-13
 purpose of, 7-1
 QUERY, 7-1. *See also qualifier*
 CTRL/A function, description of, 10-37
 CTRL/C function, description of, 10-36
 CTRL/D function, description of, 10-37
 CTRL/E function, description of, 10-37
 CTRL/K function
 description of, 10-37
 example, 10-42
 CTRL/U function
 description of, 10-21
 example, 10-22
 purpose of, 10-21
 CTRL/U function, description of, 10-36
 CTRL/W function, description of, 10-36
 CTRL/Y, entering of, 2-5
 CTRL/Z function, description of, 10-36
 Cursor
 moving, 2-21
 VT100, 2-2
 VT52, 2-2
 Cursor movement, 10-10
 arrow keys, 2-22
 BOTTOM function, 2-21, 10-16
 DOWN arrow key, 2-22
 DOWN function, 10-11
 LEFT arrow key, 2-23
 LEFT function, 10-12
 RIGHT arrow key, 2-23
 RIGHT function, 10-13
 TOP function, 2-21, 10-16
 UP arrow key, 2-22
 UP function, 10-10
 CUT command
 description of, 9-13
 example, 2-18, 9-13
 purpose of, 9-13

CUT function
description of, 10-24
example, 2-25, 10-7, 10-25, 10-28,
10-31, 10-33
purpose of, 10-24

D

D command
description of, 9-14
example, 9-14
purpose of, 9-14

DCL
command prompt, 2-2
CONTINUE command, 2-6, 3-2
directory command, 2-6

DEFINE KEY command
description of, 10-38
example, 10-40
purpose of, 10-37

Define key function, 10-42. *See also*
CTRL/K

DEFINE MACRO command, 1-2
description of, 7-2
example, 7-3
purpose of, 7-2

DEL C function
description of, 10-19
purpose of, 10-19

DEL EL function
description of, 10-22
purpose of, 10-22

DEL EOL function
description of, 10-22
example, 2-29, 10-22
purpose of, 10-22

DEL L function, example, 2-27

DEL W function
description of, 10-20
example, 2-26, 10-20
purpose of, 10-20

DELETE command
character editing, 2-15
description of, 1-2, 7-4
example, 2-10, 7-4, 7-5
purpose of, 7-4
QUERY, 7-4. *See also* *qualifier*

DELETE function
description of, 10-19
example of, 10-19
purpose of, 10-19

Deleting characters, 2-5
example of, 2-17

Deleting characters and lines, 2-5

Deleting lines, 2-5

Deleting words, example, 2-17

Deletions
of characters with keypad, 10-19
of lines with keypad, 2-27, 10-22
of words with keypad, 10-20

DOWN function
description of, 2-22, 10-11
example, 2-23, 10-12
purpose of, 10-11

E

Editing, character, 1-3, 9-1, 10-1

Editing, line, 1-2, 7-1

EDT
features of, 1-1
purpose of, 1-1

EDT command files
description of, 6-4
example, 6-5
purpose of, 6-4

EDT command level prompt, 2-1

EDT command line, 2-3
example, 6-4
examples, 2-3
format of, 6-1
purpose of, 6-1
qualifiers, 6-1

EDT control commands, 1-2
example, 8-1

EL entity, description of, 9-2

Ending an editing session, 2-5

ENTER function
description of, 10-33
purpose of, 10-33

Entity
BL, 9-2
BPAGE, 9-3
BPAR, 9-2
BR, 9-3
BSEN, 9-2
BW, 9-2
C, 9-2
EL, 9-2
EPAGE, 9-3
EPAR, 9-2
ER, 9-3
ESEN, 9-2
EW, 9-2
L, 9-2
NL, 9-2
PAGE, 9-3
PAR, 9-2
SEN, 9-2
SR, 9-3
"string", 9-3
V, 9-3

- W, 9-2
- [EOB]
 - description of, 2-21
 - End of buffer symbol, 2-3
- EOL function
 - description of, 10-15
 - purpose of, 10-15
- EPAGE entity, description of, 9-3
- EPAR entity, description of, 9-2
- ER entity, description of, 9-3
- Error messages
 - description of, 3-3
 - list of, B-1
 - purpose of, 3-1, 3-3
- ESEN entity, description of, 9-2
- EW entity, description of, 9-2
- EX command
 - description of, 9-5
 - purpose of, 9-5
- EXIT command, 2-4
 - description of, 1-2, 8-2
 - example, 2-5, 2-9, 2-13, 8-3, 10-6
 - fixed line numbers, 8-3
 - purpose of, 8-2
 - SAVE, 8-3. *See also qualifier*
 - SEQUENCE, 8-3. *See also qualifier*
- EXIT function, 10-6
- EXT command
 - description of, 9-5
 - example, 9-5
 - purpose of, 9-5

F

- File
 - examples of revising, 2-9
 - message describing, 2-5
 - specification version number, 3-1
- File generation
 - EXIT command, 2-4
 - INSERT command, 2-4
- Files, multiple, 2-11
- FILL command
 - description of, 9-15
 - example, 9-15
 - purpose of, 9-15
- FILL function
 - description of, 10-35
 - example, 10-36
 - purpose of, 10-35
- FIND command
 - description of, 1-2, 7-6
 - example, 7-6
 - purpose of, 7-5
- FIND function
 - description of, 10-16

- example of, 10-18
- purpose of, 10-16
- Fixed line numbers, 7-17
 - assigning, 4-1
 - definition of, 1-4, 4-1
 - saving of, 4-1
 - source of, 4-1
- FNDNXT function
 - description of, 10-17
 - purpose of, 10-17
- Function
 - ADVANCE, 10-16
 - APPEND, 10-27
 - BACKSPACE, 10-15
 - BOTTOM, 2-21, 10-16
 - CHAR, 10-14
 - CHNGCASE, 2-31, 10-35
 - COMMAND, 2-31, 10-34
 - CTRL/A, 10-37
 - CTRL/C, 10-36
 - CTRL/D, 10-37
 - CTRL/E, 10-37
 - CTRL/K, 10-37, 10-42
 - CTRL/U, 10-21, 10-36
 - CTRL/W, 10-36
 - CTRL/Z, 10-36
 - CUT, 10-7, 10-24
 - CUT, 2-25
 - DEL C, 10-19, 10-20
 - DEL EL, 10-22
 - DEL EOL, 2-29, 10-22
 - DEL L, 2-27
 - DEL W, 2-26
 - DELETE, 10-19
 - DOWN, 2-22, 10-11
 - ENTER, 10-33
 - EOL, 10-15
 - EXIT, 10-6
 - FILL, 10-35
 - FIND, 10-16
 - FINDNXT, 10-17
 - GOLD, 2-20, 10-6
 - GOLD/A, 8-11
 - GOLD/D, 8-10
 - GOLD/E, 8-10
 - GOLD/T, 8-11
 - HELP, 10-7
 - LEFT, 2-23, 10-12
 - LINE, 10-15
 - LINE FEED, 10-20
 - OPEN LINE, 10-9
 - PAGE, 10-16
 - PASTE, 2-26, 10-7, 10-26
 - QUIT, 10-6
 - REPLACE, 10-30

RESET, 10-8
RIGHT, 2-23, 10-13
SECTION, 10-16
SELECT, 2-24, 10-23
SELECT ranges, 2-26
SPECINS, 10-34
SUBS, 10-32
TOP, 2-21, 10-16
UND C, 10-19
UND L, 2-28, 2-30, 10-22
UND W, 2-26, 10-21
UP, 2-22, 10-10
WORD, 10-15

G

GOLD function
description of, 10-6
example, 10-7
GOLD key, description, 2-20
GOLD/A function
description of, 8-11
example, 8-11
purpose of, 8-11
GOLD/D function
description of, 8-10
example, 8-10
purpose of, 8-10
GOLD/E function
description of, 8-10
example, 8-10
purpose of, 8-10
GOLD/T function
description of, 8-11
example, 8-11
purpose of, 8-11

H

Help, to obtain, 1-4, 2-20, 8-3, 10-7
HELP command, 1-4
description of, 1-2, 8-3
example, 8-3
purpose of, 8-3
Help facility, definition of, 1-4
HELP function
description of, 2-20, 10-7
purpose, 10-7
purpose of, 2-20

I

I command
description of, 9-6
example, 9-6
purpose of, 9-6

Implied TYPE command, 2-7. *See also null command*

INCLUDE command
description of, 1-2, 7-6
example, 2-11, 2-12, 7-6
purpose of, 7-6

Indentation level counter
decrementing, 8-11, 9-9
definition of, 8-11
incrementing, 8-11, 9-9
using, 8-9

Input file, prompt for, 3-2

INSERT command, 2-4
description of, 1-3, 7-7
example, 2-5, 2-7, 2-8, 2-9, 2-10, 7-7
exiting, 2-5, 2-7
purpose of, 7-7

INSERT END command, 2-8

INSERT mode
EDT entry into, 2-4
exiting, 2-5, 2-7
Inserting text, 10-8

J

Journal file
definition of, 1-4
description of, 2-5
example, 3-1
executing contents of, 3-2
purpose of, 3-1
saving, 3-1

K

Key, definition of, 10-3
Keypad and nokeypad, comparison of, 10-2
Keypad editing
cursor movement, 2-21
definition of, 1-3
deleting lines, 10-22
deleting results of, 10-6
deleting text, 2-26, 10-19
deleting words, 10-20
description of, 10-2
entering, 2-19, 10-3
essential functions, 10-6
exiting, 2-31, 10-6
HELP, 2-20
inserting text, 2-20, 10-8
moving text, 2-24
quitting, 10-6
saving results of, 10-6
to start, 10-2, 10-3
undeleting text, 2-26

Keypad functions
 VT100, 10-5f
 VT52, 10-4f
Keypad functions, how to use, 10-3
Keypad keys
 VT100, 10-5f
 VT52, 10-4f
Keypad layout, description of, 10-3

L

L entity, description of, 9-2
LEFT function
 description of, 2-23, 10-12
 example, 2-24, 10-12
 purpose of, 10-12
Level prompt, EDT command, 2-1
Line, EDT command, 2-3
Line buffer
 purpose of, 5-2
 using, 5-2
Line editing commands, 2-4
 abbreviations for, 7-1
 example, 7-1
LINE FEED function
 description of, 10-20
 purpose of, 10-20
LINE function
 description of, 10-15
 purpose of, 10-15
Line numbers
 assigning, 4-1
 changing, 2-11
 decimal numbers, 4-1
 definition of, 1-4
 display of, 2-5
 fixed, 4-1
 purpose of, 4-1
 range of, 4-1
 renumbering, 4-1
Lines, deleting, 2-10
Login command file, 2-2
Login procedure
 example, 2-2
 password, 2-2
 user name, 2-2

M

MAIN, text buffer, 5-1
MOVE command
 description of, 1-3, 7-7
 example, 2-13, 7-7
 purpose of, 7-7
 QUERY, 7-7. *See also qualifier*

N

New files
 example of, 2-6
 generation of, 2-4
NL entity, description of, 9-2
Nokeypad character editing
 definition of, 1-3
 description of, 2-14, 9-1
 entering, 9-1
 entities, 9-1
Nokeypad commands
 buffer definition, 9-4
 command definition, 9-3
 command formats, 9-4
 count definition, 9-3
 description of, 9-3
 direction definition, 9-4
 element sequence, 9-4
 elements of, 9-3
 entity definition, 9-4
 format of, 9-3
Noncontiguous line ranges, example, 4-4
Null command
 description of, 1-3, 7-7, 9-16
 example, 7-7
 purpose of, 7-7, 9-16

O

OPEN LINE function
 description of, 10-9
 example, 10-9

P

PAGE entity, description of, 9-3
PAGE function
 description of, 10-16
 purpose of, 10-16
PAR entity, description of, 9-2
PASTE, text buffer, 5-1
PASTE command
 description of, 9-17
 example, 2-19
 purpose of, 9-17
PASTE function
 description of, 10-26
 example, 2-26, 10-7, 10-26, 10-29
 purpose of, 10-26
 Position marker, 2-2. *See also cursor*
PRINT command
 description of, 1-3, 7-8
 example, 2-13, 7-8
 purpose of, 7-8
Prompt, EDT command level, 2-1

Prompts, DCL command, 2-2

Q

Qualifier

BRIEF, 7-11, 7-14
of COPY command, 7-1, 7-2
of DELETE command, 7-4
DUPLICATE, 7-2
EDT command line, 6-2, 6-3t
EDT command line, description of, 6-3
of EXIT command, 8-2
of MOVE command, 7-7
NOTYPE, 7-11
QUERY, 7-1, 7-4, 7-7, 7-11
of QUIT command, 8-4
RECOVER, 2-6
RECOVER, example, 2-5, 2-6, 3-1
of RESEQUENCE command, 7-9
SAVE, 8-3, 8-4
SEQUENCE, 7-9, 7-16, 8-2
STAY, 7-14
of SUBSTITUTE command, 7-11
of TYPE command, 7-14
of WRITE command, 7-16

QUIT command

description of, 1-2, 8-4, 9-6
example, 8-4
purpose of, 8-4, 9-6
SAVE, 8-4. *See also qualifier*

QUIT function, 10-6

R

R command

description of, 9-17
example, 9-17
purpose of, 9-17

Range specification

ALL "string", 4-4
AND, 4-4
BEFORE, 4-4
BEGIN, 4-2
BUFFER, 4-5
contiguous line, 4-4
default, 2-10
definition of, 4-2
description of, 4-1
END, 4-2
example, 4-2
FOR or, 4-4
LAST, 4-2
minus(-), 4-2
noncontiguous line, 4-4
number, 4-2

ORIGINAL number, 4-2

period (.), 4-2
plus (+), 4-2
purpose of, 4-2
REST, 4-4
single line, 4-2
"string" 4-2
text buffer, 4-4
THRU or :, 4-4
WHOLE, 4-4

Ranges, CUT function and SELECT, 2-26

RECOVER, 2-6, 2-6. *See also qualifier, example*

RECOVER qualifier, example, 2-5, 3-1

REF command

description of, 9-7
purpose of, 9-7

REPLACE command

description of, 1-3, 7-8
example, 7-9
purpose of, 7-8

REPLACE command, example, 2-7

REPLACE function

description of, 10-30
example, 10-31
purpose of, 10-30

RESEQUENCE command

description of, 1-3, 7-9
example, 2-11, 2-13, 7-10
purpose of, 7-9
SEQUENCE qualifier, 7-9

RESET function

example, 10-8
purpose of, 10-8

Revising files, examples of, 2-9

RIGHT function

description of, 2-23, 10-13
example, 2-23, 10-13
purpose of, 10-13

S

S/s1/s2/ command

description of, 9-12
purpose of, 9-12

Screen display, nokeypad, 2-14

SECTION function

description of, 10-16
purpose of, 10-16

SEL command

description of, 9-7
example, 9-7

SELECT command, example, 8-10

SELECT function

- description of, 10-23
- example, 2-25, 10-23
- example of, 10-31
- purpose of, 10-23
- SELECT range
 - and CUT function, 2-25
 - marking text with, 2-24
- SEN entity, description of, 9-2
- SET CASE command
 - description of, 8-5
 - example, 8-5
 - purpose of, 8-5
- SET command
 - description of, 1-2, 8-4
 - example, 8-4
 - purpose of, 8-4
- SET CURSOR command
 - description of, 8-5
 - example, 8-6
 - purpose of, 8-5
- SET ENTITY command
 - description of, 8-6
 - example, 8-7
 - purpose of, 8-6
- SET KEYPAD command
 - description of, 8-7
 - purpose of, 8-7
- SET LINES command
 - description of, 8-7
 - example, 8-7
 - purpose of, 8-7
- SET MODE command
 - description of, 8-7
 - purpose of, 8-7
- SET NOKEYPAD command, example, 2-14, 2-15, 8-2
- SET NUMBERS command
 - description of, 8-8
 - purpose of, 8-8
- SET QUIET command
 - description of, 8-8
 - purpose of, 8-8
- SET SCREEN command
 - description of, 8-8
 - example, 8-8
 - purpose of, 8-8
- SET SEARCH command
 - BOUNDED parameter, 8-9
 - description of, 8-9
 - END parameter, 8-9
 - EXACT parameter, 8-9
 - example, 8-9
 - purpose of, 8-9
- SET TAB command
 - description of, 8-9
 - example, 8-10, 9-8
 - purpose of, 8-10
- SET TERMINAL command
 - description of, 8-11
 - purpose of, 8-11
- SET TRUNCATE command
 - description of, 8-11
 - example, 8-12
 - purpose of, 8-11
- SET VERIFY command
 - description of, 8-13
 - purpose of, 8-13
- SET WRAP command
 - description of, 8-13
 - example, 8-13
 - purpose of, 8-13
- SHL command
 - description of, 9-10
 - example, 9-10
 - purpose of, 9-10
- SHOW BUFFER command
 - description of, 8-14
 - example, 2-12, 5-2, 8-14
 - purpose of, 8-14
- SHOW CASE command
 - description of, 8-15
 - purpose of, 8-15
- SHOW command
 - description of, 1-2, 8-14
 - purpose of, 8-13
- SHOW CURSOR command
 - description of, 8-16
 - purpose of, 8-16
- SHOW ENTITY command
 - description of, 8-16
 - purpose of, 8-16
- SHOW KEY command
 - description of, 8-16
 - example, 8-16
 - purpose of, 8-16
- SHOW SCREEN command
 - description of, 8-16
 - purpose of, 8-16
- SHOW SEARCH command
 - description of, 8-16
 - purpose of, 8-16
- SHOW TERMINAL command, purpose of, 8-16
- SHOW VERSION command, purpose of, 8-16
- SHR command
 - description of, 9-10
 - example, 9-10
 - purpose of, 9-10

Single line ranges, example, 4-3

SN command
 description of, 9-12
 example of, 9-12
 purpose of, 9-12

SPECINS function
 description of, 10-34
 example, 10-34
 purpose of, 10-34

SR entity, description of, 9-3

Startup command files, definition of, 1-4

String search buffer
 purpose of, 5-3
 using, 5-3

“string” entity, description of, 9-3

SUBS function
 description of, 10-32
 example, 10-32
 purpose of, 10-32

Substitute buffer
 purpose of, 5-3
 using, 5-3

SUBSTITUTE command
 BRIEF, 7-12. *See also qualifier*
 description of, 1-3, 7-11
 example, 2-10, 7-12
 NOTYPE, 7-12. *See also qualifier*
 purpose of, 7-11
 QUERY, 7-12. *See also qualifier*

SUBSTITUTE NEXT command
 description of, 1-3, 7-13
 example, 7-13, 7-14
 purpose of, 7-13

T

TAB command
 description of, 9-7
 example, 9-8
 purpose of, 9-7

TADJ command
 description of, 9-18
 example, 9-18
 purpose of, 9-18

TC command
 description of, 9-9
 example, 9-9
 purpose of, 9-9

TD command
 description of, 9-11
 example, 9-9
 purpose of, 9-11

Text, to insert, 2-20

Text buffer
 definition of, 5-1

displaying contents of, 2-8
 inserting text, 2-8
 MAIN, 2-4
 MAIN, example, 3-1, 5-1
 others, example, 5-1
 PASTE, example, 5-1
 transferring between, 2-13

Text buffers
 definition of, 1-3
 examples of using, 5-2
 limits on, 5-2

TI command
 description of, 9-11
 example, 9-9
 purpose of, 9-11

TOP command
 description of, 9-9
 purpose of, 9-9

TOP function
 description of, 2-21, 10-16
 example of, 2-21
 purpose of, 10-16

TYPE command
 BRIEF, 7-14. *See also qualifier*
 description of, 1-3, 7-14
 example, 2-7, 2-8, 2-10, 7-14, 7-15
 implied, 2-7
 purpose of, 7-14
 STAY, 7-14. *See also qualifier*

TYPE WHOLE command, 2-8

U

UND C function
 description of, 10-19
 purpose of, 10-19

UND L function
 description of, 10-22
 example, 2-28, 2-30, 10-23
 purpose of, 10-22

UND W function
 description of, 10-21
 example, 2-26, 10-21
 purpose of, 10-21

UNDC command
 description of, 9-11
 purpose of, 9-11

UNDL command
 description of, 9-11
 purpose of, 9-11

UNDW command
 description of, 9-11
 purpose of, 9-11

UP function
 description of, 2-22, 10-10

example, 2-22, 10-10, 10-11
purpose of, 10-10
User buffers, definition of, 5-2

V

V entity, description of, 9-3
VT52 and VT100 Keypad numbers, 10-39f

W

W entity, description of, 9-2
Word buffer
 purpose of, 5-3
 using, 5-3
WORD function
 description of, 2-24, 10-15
 example, 2-24
 purpose of, 10-15
WRITE command
 description of, 1-3, 7-16
 example, 7-16, 7-17
 purpose of, 7-16
 SEQUENCE, 7-16. *See also qualifier*

Reader's Comments

Note: This form is for document comments only. Digital will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement. _____

Did you find errors in this manual? If so, specify the error and the page number. _____

Please indicate the type of user/reader that you most nearly represent.

- Assembly language programmer
- Higher-level language programmer
- Occasional programmer (experienced)
- User with little programming experience
- Student programmer
- Other (please specify) _____

Name _____ Date _____

Organization _____

Street _____

City _____ State _____ Zip Code
or
Country _____

---Do Not Tear - Fold Here and Tape---

digital

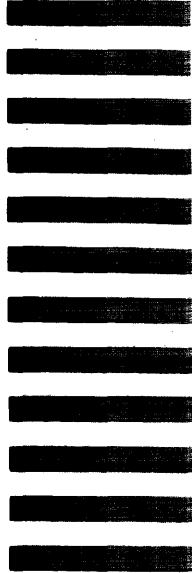


No Postage
Necessary
if Mailed in the
United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

ATTN: Commercial Engineering Publications MK1-2/ H3
DIGITAL EQUIPMENT CORPORATION
CONTINENTAL BOULEVARD
MERRIMACK N.H. 03054



--- Do Not Tear - Fold Here and Tape ---