digital

TECO
Text Editor and Corrector Program

PDP-10

TECO
Text Editor and Corrector Program

Programmer's Reference Manual

Order No. DEC-10-ETEA-D Price $0.70
Direct comments concerning this manual to Software Quality Control, Maynard, Mass.

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

CONTENTS

CONTENTS (cont)

CHAPTER 1
TECO

TECO (Text Editor and Corrector) is a sophisticated text editor for use on all five PDP-10 systems; 10/10, 10/20, 10/30, 10/40, 10/50. TECO allows many simple editing requests, macro definitions, and conditional statements. The macro language permits highly sophisticated search, match and substitution operations and provides simple typographical corrections to text. TECO editing is performed on files recorded in ASCII characters. The minimum core requirement is 4K. All standard I/O devices are acceptable to TECO but the most commonly used are DECtape, paper tape reader, paper tape punch, disk, magnetic tape, Teletype and line printer.

1.1 GENERAL DESCRIPTION

TECO indicates its readiness to accept commands by typing an asterisk (*). At this stage the user types in commands to TECO to describe the location of the material to be edited (the "input"). By further commands the user, (1) reads the first "page" of the input into the buffer; (2) inspects any lines in which he is interested, by having them typed on his Teletype; (3) makes changes by typing in the necessary commands; (4) outputs this first page to his declared output device, reads another page and similarly proceeds to the end of his file; (5) after the last page of the file is processed, the user closes the output file and thereby releases the output device. This is by no means the only way in which the user may proceed but it is the most common.

Each command to TECO consists of one or more characters. Many commands may be sent to TECO as a character string. TECO will sequentially execute these commands when two consecutive ALTMODES are received. Some commands use ALTMODE as a terminating character. In these cases the terminal ALTMODE may be used as the first of two ALTMODES needed for TECO to execute the command string.

TECO was developed at Massachusetts Institute of Technology on the PDP-1 and by Project MAC on the PDP-6. The work of the following people is acknowledged: Daniel L. Murphy, Stewart Nelson, Jack Holloway, and Richard Greenblatt.

1.2 TECO ENVIRONMENT

Monitor	All
Minimum Core	4K
Additional Core	Takes advantage of any additional core available. Each 1K additional core augments the basic 6500 character buffer by 5K additional characters.
Equipment Required	One input device and one output device.

1.3 TECO INITIALIZATION

.R TECO↵	Loads the Text Editor and Corrector program.
*	TECO is ready to accept a command.

NOTES: When typing command strings to TECO, the following points should be noted.

$\boxed{\text{ALTMODE}}$ - One ALTMODE is used to terminate the text within a command string, where applicable; two successive ALTMODE's terminate the entire command string sequence and generate a RETURN, LINE-FEED. ALTMODEs type back as $'s.

$\boxed{\text{RUBOUT}}$ - The RUBOUT key can be used to erase the preceding typed-in character(s) of a command string. Each character erased is echoed back on the Teletype (e.g., ABD RUBOUT DC...). Successive RUBOUT's can be used to erase more than one character.

N.B. To erase a carriage return (which generates RETURN, LINE-FEED), two RUBOUT's are required, one RUBOUT to erase the LINE-FEED and one to erase the RETURN.

Two successive ↑G's can be used to wipe out the entire command string currently being typed.

TECO commands in the form↑x (where "x" is any character) can be entered by either holding down the CTRL key while striking the "x" key or typing up-arrow (shift N) followed by the "x" character. These alternatives are not true where ↑x is a character within a command string (such as in a Search argument); in this case, the CTRL key must be used.

1.4 TECO TERMINATION COMMANDS

↑C	Returns control to the Monitor without waiting for any I/O operations to finish.
↑G (BELL)	Returns control to the Monitor after completing all current output requests.

1.5 TECO DEFINITIONS

A. The storage area in TECO is used for three types of storage:

(1) The Buffer is the area where text to be edited is examined and modified. At all times it contains a (possibly null) character string.

The Pointer is associated with the buffer. It is a movable flag which always sits between two adjacent characters in the buffer (or at either end of the buffer). The pointer has a numeric value equal to the number of characters to its left in the buffer. If at the moment there are n characters in the buffer, the value of the pointer may range from 0 to n.

(2) The Command String Area

Commands to TECO are written as a character string. TECO interprets the characters in the command string area as a series of commands.

(3) The Q-Registers are locations for storing quantities, or strings of text for later use. There are 36 Q-Registers which are designated by the symbols 0,...9,A, B...Z. Each Q-Register may then (a) be undefined, (b) contain a positive or negative integer, (c) contain a character string. The Q-Registers are typically used in the more sophisticated requests to TECO.

B. A Page

A page is the amount of data transferred by a Y-request or a P-request. The buffer always contains a (possibly null) page of information.

1.6 LEARNING TECO

This manual contains two logical sections. In the first section, emphasis is on simple text editing methods for,(a) declaring input and output devices;(b) reading "pages" of the input file; (c) examining lines in each page;(d) making deletions and insertions;(e) writing edited information to the output device, and (f) closing the output file. Many examples are given throughout the section.

In the second section more sophisticated TECO commands are described, for example, searching for text within a file; inserting character strings into the text by the use of Q-registers; and using the macro language to develop iterative requests. Easy methods for using TECO effectively are shown. A debugging aid is also described.

CHAPTER 2
BASIC TECO COMMANDS

Editing with TECO is achieved by creating a new destination file from the source file (or files). Editing is not achieved by alteration of the source file(s). One source file can be defined at any time. If a new source file is defined, the previous file is automatically superseded. Similarly, if a new destination file is defined, the previous filė is closed and superseded. This chapter describes the TECO commands required for simple text editing and provides many examples.

2.1 SELECT THE INPUT DEVICE

*ERdev:filename.ext [proj,prog] ALTMODE

Selects the input device and file (if specified).

dev:	DTAn:	(DECtape)
	PTR:	(paper tape reader)
	DSK:	(disk)
	MTAn:	(magnetic tape)
	CDR:	(card reader)
	TTYn:	(Teletype)
filename.ext		(DSK: or DTAn: only)

[proj,prog] (DSK: only)

Specified only if file is located in other than user's area.

2.2 SELECT THE OUTPUT DEVICE

*EWdev:filename.ext [proj,prog] ALTMODE

Selects the output device and file (if specified).

*EZdev:filename.ext [proj,prog] ALTMODE

Selects the output device and file (if specified), and rewinds the tape (if magnetic tape) or zeros the directory (if DECtape).

dev:	DTAn:	(DECtape)
	DSK:	(disk)
	MTAn:	(magnetic tape)
	PTP:	(paper tape punch)
	LPT:	(line printer)
	TTYn:	(Teletype)
filename.ext		(DSK: or DTAn: only)

[proj,prog] (DSK: only)

Specified only if file is located in other than user's area.

EF

Terminate output on the current output file and close the file without selecting a new output file.

2.3 MAGNETIC TAPE POSITIONING

EM	Rewind the currently selected input magnetic tape.
nEM	Depending upon the value of n, perform one of the following operations on the currently selected input magnetic tape.

n[1]	Operation
1	Rewind tape to load point.
3	Write end of file.
6	Skip one record.
7	Backspace one record.
8	Skip to logical end of tape.
9	Rewind and unload tape.
11	Erase 3 in. of tape.
14	Advance tape one file.
15	Backspace tape one file.

NOTE 1: Throughout TECO, all numbers in command strings are interpreted as decimal.

2.4 INPUT COMMANDS

Y Read from current input device into buffer until

a. A FORM character is read (i.e., a "page" has been input) or

b. The buffer is at least 2/3 full and a LINE-FEED character is read or

c. The end of file occurs or

d. The buffer is completely full.

NOTES:

1. The FORM character, if read, does not enter the buffer.
2. Any data previously residing in the buffer is destroyed.
3. The pointer is positioned immediately before the first character in the buffer.

A Read from the current input device and append the incoming data to information already residing in the buffer. Terminate reading on the same conditions as in Y.

NOTES:

1. No previous data is destroyed.
2. The pointer is not moved.

2.5 OUTPUT COMMANDS

PW Output the entire buffer to the selected output device, with a FORM character appended as the last character. Do not alter the contents of the buffer or move the pointer.

nP	Equivalent to a PW command followed by a Y command (i.e., output the current contents of the buffer followed by a FORM character, and then read in more data from the input device). If n is specified, repeat this operation n times. If n is omitted, it is assumed to be equal to "1."
m,nP	Output the m+1 through the nth character from the buffer to the current output file. Do not append a FORM character at the end. Do not alter the contents of the buffer or move the pointer.

2.6 EDITING COMMANDS

2.6.1 Move the Pointer

nJ	Move pointer to right of the nth buffer character and give the pointer symbol (.) the value of n. If n is omitted, set pointer to before the first buffer character (same as 0J).
nC	Set the pointer to the right of the nth character beyond the pointer's present position (equal to .+nJ).
nR	Set the pointer to the left of the nth character prior to the pointer's present position (equal to .-nJ).
nL	+n - Move the pointer to the right, stopping after it has passed over n LINE-FEED characters. -n - Move the pointer to the left, stopping after it has passed over n+1 LINE-FEED characters, then move to the right of the last LINE-FEED character passed over. If n is omitted, assume 1L.

2.6.2 Delete Text

nD	Delete n characters. +n - Delete them just to the right of the pointer. -n - Delete them just to the left of the pointer.
nK	+n - Move the pointer to the right, stopping after it has passed over n LINE-FEED characters. Delete all characters the pointer passes over. -n - Move the pointer to the left, stopping after it has passed over n+1 LINE-FEED characters, then move it to the right of the last LINE-FEED character passed over. Delete all characters between this point and the pointer's previous position.
m,nK	Delete the m+1 through the nth characters of the buffer. Set the pointer where the deletion occurred.

2.6.3 Insert Text

Iaaaa....aaaa ALTMODE

Insert the text following the "I" up to, but not including, the ALTMODE character, beginning at the current pointer position. Move the pointer to the right of the inserted material.

nI	Insert at the pointer location a character whose ASCII code is n (n must be a decimal value). Move the pointer to the right of the inserted character.
nn...nn\	Insert at the current pointer location the ASCII numbers equal to nn...nn. Move the pointer to the right of the inserted text.
⇥ text ALTMODE	Insert at the current pointer location a TAB (⇥) character and the following text up to but not including the ALTMODE character. Move the pointer to the right of the inserted text.
@ I/text/	Insert at the current pointer location the text which follows. The text is delimited by a character (/) which can be any character not appearing in the text.

2.6.4 Type Text

NOTE: T commands do not move the pointer.

nT	Type out the string of characters beginning at the current pointer position and terminating after the nth LINE-FEED character is encountered.
	+n - n lines to the right of the current pointer position.
	-n - n lines to the left of the current pointer position.
	If n is omitted, the value is assumed to be "1."
m,nT	Type out the m+1 through the nth characters of the buffer.

2.7 STAND-ALONE EXAMPLES (BASIC)

2.7.1 Open an Input File

a) ERDTA5:SOURCE.MAC ALTMODE	Open the input file called SOURCE.MAC located on DTA5.
b) ERDSK:SRCE.MAC [12,24] ALTMODE	Open the input file called SRCE.MAC located in area 12,24 on the disk.
c) ERPTR: ALTMODE	Open an input file on the paper tape reader.

2.7.2 Open an Output File

a) EWDTA3:EDITED.MAC ALTMODE	Open an output file on DTA3 and call it EDITED.MAC.
b) EZDTA1:DEBUG.MAC ALTMODE	Zero the directory on DTA1, open an output file on it, and call the file DEBUG.MAC.

2.7.3 Read a Page

a) Y	Read a page into the buffer from the current input file, destroying the previous contents of the buffer.
b) A	Read a page into the buffer, appending the data to the end of the information currently in the buffer.

2.7.4 Output Data

a) PW	Output the entire buffer, followed by a FORM character.
b) 6P	Execute the write and read cycle six times.
c) 12,50P	Write out the 13th through the 50th characters of the buffer.

2.7.5 Pointer Positioning

a) Y18J	a) Read in a page of information and position the pointer after the 18th character of the buffer; b) Then move the pointer left to between characters 13 and 14.

2.7.6 Delete Text

a) J19C3D	Move the pointer to the right of the 19th character in the buffer and then delete the next three characters to the right (characters 20, 21, and 22).
or	
b) 19,22K	Delete the 19+1 (20th) through the 22nd characters of the buffer.
c) HK	Delete all the characters in the buffer (for a description of numeric values and arguments see Paragraph 3.4).

2.7.7 Insert Text

a) J2LITAG: MOVE 1,AMT↵ ALTMODE	Move the pointer to a position following the second line of the buffer; insert the text "TAG: MOVE1,AMT" between the second and third lines of the buffer.

b) 69\	Insert the digits "69" in ASCII at the current pointer position (same as I69 or 54I57I). NOTE: \ is typed with a SHIFT $\boxed{\text{FORM}}$.
c) $\boxed{\text{TAB}}$ ERROR IN JOB $\boxed{\text{ALTMODE}}$	Insert a tab followed by the text "ERROR IN JOB" at the current pointer position.
d) @I#ERDSK:PROG $\boxed{\text{ALTMODE}}$ #	Insert the text "ERDSK:PROG $\boxed{\text{ALTMODE}}$" at the current pointer position. NOTE: The use of delimiters is the only method for inserting an ALTMODE in the text.

2.7.8 Typing Text

a) 3T	Assuming that the pointer is at the beginning of the buffer, type out the first three lines of the buffer.
b) 25,100T	Type out the 25+1 (26th) through the 100th character of the buffer.

2.8 EXAMPLES OF EXECUTION OF BASIC TECO COMMANDS

.R TECO↵	
*ERDTA1:SCFILE.MAC $\boxed{\text{ALTMODE}}$ $\boxed{\text{ALTMODE}}$ ↵	Open the file called SCFILE.MAC on DTA1 for input.
*EWDTA2:EDFILE.MAC $\boxed{\text{ALTMODE}}$ $\boxed{\text{ALTMODE}}$↵	Open an output file on DTA2 and call it EDFILE.MAC.
*Y0,20T $\boxed{\text{ALTMODE}}$ $\boxed{\text{ALTMODE}}$↵ aaaaa.......aaaaa↵	Read a buffer of information from the input file and type the first 20 characters of the buffer.
*3LT $\boxed{\text{ALTMODE}}$ $\boxed{\text{ALTMODE}}$ ↵ bbbbb.......bbbbb↵	Move the pointer to the right, stopping when three LINE-FEED characters have been encountered; type the text of the fourth line in the buffer.
*ITHIS IS A SAMPLE INSERT↵ $\boxed{\text{ALTMODE}}$ $\boxed{\text{ALTMODE}}$↵	Insert the text "THIS IS A SAMPLE INSERT↵ " between the third and fourth lines of the buffer and position the pointer after the inserted material.
*10PT $\boxed{\text{ALTMODE}}$ $\boxed{\text{ALTMODE}}$↵ ccccc......ccccc↵	Write out the current buffer to the output device; read in and write out the next nine "pages" of data; read in the 11th page of data and position the pointer at the beginning of the buffer; type out the first line of the buffer.

*K $\boxed{\text{ALTMODE}}$ $\boxed{\text{ALTMODE}}$	Delete this line from the file; position the pointer at the beginning of the (now) first line in the buffer.
*200PPWEF $\boxed{\text{ALTMODE}}$ $\boxed{\text{ALTMODE}}$	Repeats the write and read cycle 200 times and writes out the last page before terminating the output file.
*↑G $\boxed{\text{ALTMODE}}$ $\boxed{\text{ALTMODE}}$ EXIT↵ ↑C ↵	Return control to the Monitor after all output requests have been completed.
.KJOB↵ .	Kill the job, deassign all devices, release core.

CHAPTER 3
ADVANCED TECO COMMANDS

This chapter describes the more sophisticated TECO functions, the use of Q registers and the functions available through the use of macros, iterations, and conditionals. The use of these functions is decided by the user's imagination but several practical examples are given. Characters which may appear in command strings to develop numeric values are described in Paragraph 3.4. These characters are frequently used and are presented in this chapter in tabular format.

3.1 SEARCH COMMANDS

3.1.1 Summary

a. All searches begin at the current location of the pointer.

b. Modifiers:

Each search command can be preceded by a modifier character, : or @ .

: causes the search command to have a numeric value at completion;
0 if the search has failed (the requested text was not found) or
-1 if the search was successful (the requested text was found).
@ indicates that the text to be matched is delimited by some character (same as in the @I command).

c. Numeric Arguments:

A numeric argument can appear following the modifier (if any) but preceding the command. If the numeric argument is n, TECO searches for the nth occurrence of the text. If n is not used, the value of n is assumed to be "1."

d. Pointer Positioning:

If search is successful, the pointer is positioned to the right of the matched text.

If the search fails, the pointer is positioned at the beginning of the buffer.

e. Use of Special Characters Within Text:

↑S - Match any separator character (any character not a letter, number, period, dollar sign, or percent symbol).

↑X - Match any (arbitrary) character. Used when the contents of some position within the text is unimportant.

↑N x - Match any character except x.

↑Q - Takes the next character literally, even if it is one of these four special characters.

S↑Q↑X $\boxed{\text{ALTMODE}}$ - Find the character ↑X.

NOTE: See note on page TECO-2.

Table TECO-1
Search Commands Summary

Command	Action at End of Buffer	Action at End of File	Values		Typeout ? if Failure
			Success	Fail	
S	Failure	N/A	N/A	N/A	Yes
:S	Failure	N/A	-1	0	No
N	Performs a P command and resumes search.	Failure	N/A	N/A	Yes
:N	Performs a P command and resumes search.	Failure	-1	0	No
←	Performs a Y command (read only) and resumes search.	Failure	N/A	N/A	Yes
:←	Performs a Y command (read only) and resumes search.	Failure	-1	0	No

3.2 Q REGISTER COMMANDS

Q registers are provided for storing quantities, command strings, or buffer contents for later use. Thirty-six Q registers, labeled 0 through 9 and A through Z, are available.

nUi	Places the numeric value n in Q-register i.
Qi	Represents the current value in Q-register i.
%i	Adds 1 to the value in Q-register i and represents the new value.
m,nXi	Places characters m+1 through the nth character of the buffer into Q-register i.
nXi	Places the buffer characters between the current pointer position and the nth LINE-FEED character in Q-register i.
Gi	Place the text contained in Q-register i into the buffer beginning at the current pointer location. Set the pointer to the right of the insertion.
[i	Push the contents of Q-register i onto the Q-register pushdown list.
]i	Pops the top entry of the Q-register pushdown list into Q-register i. The Q-register pushdown list is cleared each time two successive ALTMODE's are typed.

3.3 MACRO, ITERATION, AND CONDITIONAL COMMANDS

Mi	Perform the text in Q-register i as a series of commands.
<>	Iteration brackets. When > is encountered, command interpretation is sent back to <.

;	If not in an iteration, an error results. If most recent search failed, send command interpretation to just beyond the matching > on the right; otherwise, no effect.
n;	If not in an iteration, an error results. If the value of n is 0 or greater, send command interpretation just past the matching > to the right; otherwise, no effect.
!tag!	Tag definition. Tag is the name of the location in which it appears in a command string.
Otag $\boxed{\text{ALTMODE}}$	Go to the named tag, which must appear in the current macro or command string.
n"G	If n ≥ 0, send command interpretation to the next matching '; if n >0, no effect.
n"L	If n ≥0, send command interpretation to the next matching '; if n <0, no effect.
n"N	If n = 0, send command interpretation to the next matching '; if n ≠ 0, no effect.
n"E	If n ≠ 0, send command interpretation to the next matching '; if n = 0, no effect.
n"C	If n is <u>not</u> one of the ASCII characters (a letter, number, period, dollar sign, or percent symbol), send command interpretation to the next matching '; otherwise, no effect.

NOTE: The " and ' symbols are matched in the same way as the (and) symbols.

3.4 NUMERIC VALUES AND ARGUMENTS IN COMMAND STRINGS

Many command string formats permit arguments with numeric values. The following characters may appear in a command string to develop these values in any instance where a numeric value is permissible.

0 through 9	Represent their corresponding numeric values.
B	Equivalent to 0.
Z	Equivalent to the number of characters in the buffer.
.	Equivalent to the number of characters to the left of the current pointer position (or in other words, equal to the current pointer position).
Qi	Equivalent to most recent numeric value placed in Q-register i.
nA	Equivalent to ASCII value of character to right of pointer; n is used to differentiate this argument from an Append command (A) and has no other significance.
↑H	Equivalent to value of elapsed time in 60th of a second since midnight.
↑F	Equivalent to the value of the console data switches.
↑↑	(Hold down both the CTRL and SHIFT keys and type "N".) Equivalent to the value of the next character in the command string; this character is not interpreted as a command.
↑T	Stops command execution until user types a character on the Teletype; ↑T then becomes equivalent to the ASCII value of the character typed.
\	Equivalent to the value represented by the digits (or minus sign) immediately following the current pointer position. The value is terminated by the first nonnumeric character encountered. The pointer is positioned immediately following the last numeric character.

m+n	Add	Take one or two arguments. A space is equal to +.
m-n	Subtract	
m*n	Multiply	Take one or two arguments.
m/n	Divide (truncates)	
m&n	Logical AND; bitwise and of binary representations m and n.	
m#n	Logical IOR; bitwise inclusive or of binary representations m and n.	
( )	Operators +, -, *, /, #, and & are normally performed left to right. This sequence can be overruled by use of parentheses.	
n=	Causes the value of n to be typed out.	
H	Abbreviation for B, Z. (0 through the last location of the buffer; in other words, the whole buffer.)	

NOTES: If a command takes two numeric arguments, a comma is used to separate them.

3.5 STAND-ALONE EXAMPLES (ADVANCED)

3.5.1 Search

J3SMOVE ALTMODE IM ALTMODE	Within the current buffer, search for the third occurrence (3S) of the text "MOVE", position the pointer immediately after it, and insert an "M" at that point.

3.5.2 Search for a Special Character

a) S↑NA ALTMODE	Search for any character except A within the current buffer.
b) S↑S ALTMODE	Search for any separator character within the current buffer.

3.5.3 Q-Registers, Macros, Iterations, and Conditionals

a) JOUN<S LINE-FEED ALTMODE ;
%N>QN=

Count the number of LINE-FEED characters in the buffer as follows:

a. Position the pointer at the beginning of the buffer (J),

b. Place 0 in Q-register N (0UN),

c. Perform a search for a LINE-FEED character (S LINE-FEED ALTMODE); if one is found, add 1 to Q-register N ('%N). Go back (<>) and repeat this cycle until the end of the buffer is reached and the test fails (;); at this point type out the contents of Q-register N (QN=).

b) J<SJUMPA $\boxed{\text{ALTMODE}}$; ↵
-4DIRST $\boxed{\text{ALTMODE}}$>

Replace all instances of the text "JUMPA" with "JRST" in the current buffer.

a. Position the pointer at the beginning of the buffer (J).

b. Search for JUMPA; when found, backspace the pointer four positions and delete the four characters passed over (;-4D).

c. Replace these four characters with the characters "RST" (IRST).

d. Repeat this routine (<>) until the test fails (end of the buffer has been reached) and exit (;) to >.

3.5.4 To Place a Command in a Q-Register for Later Execution

@I#JOUN<S $\boxed{\text{LINE-FEED}}$ $\boxed{\text{ALTMODE}}$ ↵
%N>QN=#HXP

a. Insert the text "JOUN<S $\boxed{\text{LINE-FEED}}$ $\boxed{\text{ALTMODE}}$;%N>QN=" into the buffer (@I#.........#).

b. Copy the contents of the buffer into Q-register P (HXP).

To Execute the Command:

ERDTA3:FN.EX $\boxed{\text{ALTMODE}}$ YMP

a. Read in a page of a file to search. (ERDTA3:FN.EX $\boxed{\text{ALTMODE}}$ Y).

b. Execute the command stored in Q-register P (MP).

3.5.5 To Read in Text to be Inserted in Several Places in a File and to Store it in a Q-Register

ERPTR: $\boxed{\text{ALTMODE}}$ YHXP

a. Assume that the text to be inserted is on paper tape. Open an input file on the paper tape reader (ERPTR:); read the text into the buffer (Y); copy the contents of the buffer into Q-register P (HXP).

ERDTA4:TXTEDT (ALTMODE)↵ EWDSK:TXTEDU (ALTMODE)	b. Open the input file to be edited and the output file to contain the edited version.
YNCALC: (ALTMODE) GP	c. Read a page from the input file and initiate a search for the text "CALC:". When found, insert the text stored in Q-register P at that point (GP).
NTOT: (ALTMODE) GP	d. Search for the text "TOT:" and, when found, insert the text stored in Q-register P after it.

3.6 EXAMPLES OF EXECUTION OF ADVANCED TECO COMMANDS

.R TECO↵	
*ERMTA1: (ALTMODE) EM14EM (ALTMODE)(ALTMODE)↵	Select MTA1 for input; rewind the tape (EM) and advance the tape one file (14EM).
*EZDTA1:REVFIL (ALTMODE)(ALTMODE)↵	Select DTA1 for output; zero the directory; open a file and call it REVFIL.
*YNTAXRT (ALTMODE) 0LT (ALTMODE)↵ 1X1 (ALTMODE) (ALTMODE)↵ aaaa...TAXRT aaaa......aaaaa↵	Read in the first page from the input file; search for the text "TAXRT"; if not found, write the buffer out, read in the next page, search again, etc.; continue this cycle until either TAXRT is found or end of file is reached. If TAXRT is found, position the pointer at the beginning of the line containing it, type the line, and place the line in Q-register 1.
*JNTXRTE (ALTMODE) 0LT (ALTMODE)↵ G1 (ALTMODE)(ALTMODE)↵ bbb...TXRTE bbb......bbbbb↵	Search the buffer for the text "TXRTE"; if not found, write out the buffer, read the next page, search again; continue this cycle until either TXRTE is found or end of file is reached. If TXRTE is found, position the pointer at the beginning of the line containing it, type the line, and insert the contents of Q-register 1 immediately before that line.
*NTXTEND: (ALTMODE)↵ J<SA (ALTMODE) ;1A-47"G1A-58"L-DIB (ALTMODE)">↵ PWEF (ALTMODE)(ALTMODE)↵	Read pages from the input file and write them on the output file until end of file (marked by the text "TXTEND:") is found. At that point, move the pointer to the beginning of the buffer (J), and search for all A's in the buffer (SA); if the character following the "A" is a digit, 0 through 9 (ASCII codes 48_{10} through 57_{10}), change the "A" to a "B"

	(IB); continue searching and modifying until end of buffer is reached; write out last page and write end of file on output device.
*↑G (ALTMODE)(ALTMODE)	Return control to the Monitor after all output requests have been completed.
EXIT ↵	
↑C ↵	
.KJOB ↵	Kill the job, deassign all devices, release core.
.	

CHAPTER 4
DIAGNOSTIC MESSAGES

TECO Diagnostic Messages are as listed in Table TECO-2.

Table TECO-2
TECO Diagnostic Messages

Message	Meaning
?	An illegal or otherwise meaningless command has been entered. TECO has ignored the remainder of the command string and has returned to the idle state. At this point, the user can type ? back in, causing TECO to type out the command string terminated by the bad command.

NOTE: Search commands are considered illegal if they fail and the : modifier was not used; in this case, the message typed is "? SEARCH? ."

4.1 DEBUGGING AIDS

As an aid in debugging macros and iterations, TECO can be set in the trace mode by typing ? as any character other than the first in a command string. When in trace mode, TECO types out each command as it is interpreted, interspersed with requested output. Typing a second ? in the same manner takes TECO out of trace mode; the ? can be typed each time it is desired to change the current mode.

The user can also type comments on this Teletype sheet as he executes TECO by typing:

↑Atext ↑A

This causes all text entered to be printed on the Teletype (with the exception of terminating↑A character).

NOTE: Since the terminator ↑A is not a command, it must be typed by holding the CTRL key down while typing "A"; it cannot be entered as "up arrow, A."

If DDT (Dynamic Debugging Technique program) has been loaded along with TECO by Linking Loader, control can be transferred to DDT by using the command

↑D

READER'S COMMENTS

Digital Equipment Corporation maintains a continuous effort to improve the quality and usefulness of its publications. To do this effectively, we need user feedback: your critical evaluation of this manual and the DEC products described.

Please comment on this publication. For example, in your judgment, is it complete, accurate, well-organized, well-written, usable, etc? __________

Did you find this manual easy to use? __________

What is the most serious fault in this manual? __________

What single feature did you like best in this manual? __________

Did you find errors in this manual? Please describe. __________

Please describe your position. __________

Name __________ Organization __________

Street __________ State __________ Zip __________

.... Fold Here ....

.... Do Not Tear - Fold Here and Staple ....

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

digital

Digital Equipment Corporation
Software Quality Control
Building 12
146 Main Street
Maynard, Mass. 01754

digital

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

Printed in U.S.A.