T.S. HOLDEN

T E D

D.C.R. TEXT EDITING PROGRAM

JAN. 1974

TABLE OF CONTENTS

3. ASCII DOCUMENTATION

4. EXAMPLES

SUMMARY OF TED COMMANDS

COMMAND	MEANING	SECTION
General		
; or / or !	(a) Text delimiters	1.2
	(b) Exit from relaxed mode	1.7.2.2
↑; etc.	Set relaxed mode	1.7.2.1
Q	Quit	2.6.1.1
@	Perform next command 10000 times	2.6.1.2
(LF)	Line-feed = display pointer position	2.6.1.4
n"	Set tab at column n	2.6.1.3
" (in text)	Jump to next tab position	2.6.1.3
0" or "	Clear all tabs	2.6.1.3
nP	Print n lines	1.5.1
?/text/	Print the text string	2.9.2.1
?'state'	Print current state, e.g. DATE,SIZE,etc.	2.9.2.4
x,	Restrict output lines to x characters	1.5.2
U;c.c.;	Use charge-code, c.c.	2.1.10
U;;	Revert to users charge-code	2.1.11
n=	Skip to nth comma in command string	2.6.1.5
,	Used by n= command - else ignored	2.6.1.6
Document Commands		
nC;name;	Copy BCD/BIN doc. (ed. n) to workspace	2.1.1
nM;name;	Make workspace into document (ed. n)	2.1.2
nV;name;	Insert the doc.(ed. n) into workspace	2.1.3
n&;name;	Add named doc. to doc. formed by #	2.1.4
n#;name;	Create new doc. using &, ? and P commands	2.1.5
#//	Finish any preceding # command	2.1.6
'nC;name;	Copy ASCII doc. (ed. n) to workspace	2.1.7
n];name;	Save boxes in the named doc.	2.1.8
n[;name	Restore boxes from named doc.	2.1.9
Writing on Workspace		
C;?;	Type a BCD doc. direcly onto workspace	2.2.1
'C;?;	Type an ASCII doc. onto workspace	2.2.2
V;?;	Insert typed records into the workspace.	2.2.3
Pointer Commands		
0	Rewind pointer to origin	2.3.1.1
J	Left justify pointer to start of record	2.3.1.2
↑J	Right justify pointer to end of record	2.3.1.3
L	Move pointer one column left	2.3.1.4
R	Move pointer one column right	2.3.1.5
B	Move pointer one record back	2.3.1.6
F	Move pointer one record forward	2.3.1.7
S;text;	Search forward for the text string	2.3.2.1
↑S;text;	Search records in reverse order	2.3.2.2

Deletion Commands

-	Delete current character	2.4.1.1
E	Erase current line	2.4.1.2
↑E	Erase the previous line	2.4.1.3
D;text	Delete next occurrence of the text	2.4.1.4
nD;text;	Delete nth occurrence of text	2.4.1.4
↑D;text;	Search records upwards and delete the text	2.4.1.5

Insertion Commands

+	Insert a blank	2.4.2.1
I;text;	Insert the text string	2.4.2.2
Z;text;	Insert a new record	2.4.2.3

Exchange Commands

%	Swap character with next	2.4.3.1
↑%	Swap character with previous	2.4.3.2
X	Exchange current record with the next	2.4.3.3
↑X	Exchange current record with prior record	2.4.3.4

Replacement Commands

Y	Replace character with a space	2.4.4.1
A;text;	Replace text with blanks	2.4.4.2.
↑A;text;	Search upwards and replace text with blanks	2.4.4.3
G	Split line into two at pointer position	2.4.4.4

Execution Requests

*/MABEL/	Execute the MABEL short request	2.5.1
K;name;	Execute HA doc. on CYBER and copy the o/p	2.5.2.1
K;;	Again request output from previous K command	2.5.2.2

Search Modifying Commands

nW	Restrict search to first n columnsof records	2.6.2.1
nN	Search from column n of each record	2.6.2.2
@N	Restrict the next search to current line	2.6.2.3
\S;text;	Ignore blanks in search	2.6.2.4
S;\;	Search for first non-blank character	2.6.2.4
\S;\;	Search for \	2.6.2.4
S;...\..;	\ in string will match any character	2.6.2.4

Box Commands

nH;string;	Store character string in box n	2.7.1.1
nH;;	Clear box n	2.7.3.1
[//	Clear all boxes	2.7.3.2
n<	Store current line in box n	2.7.1.2
n>	Replace current line by box n	2.7.2.2
n$	Execute contents of box n	2.7.2.1
x'n'	Use box n as text for the chosen command, x	2.7.2.3
x'state'	Use state n as text for the chosen command	2.7.2.4
x'0'	Type text for command x on TTY	2.7.2.3
?//	Display contents of all boxes	2.9.2.3
?'n'	Display contents of box n	2.9.2.2

Reserved words - BCD

*EOS	CYBER end-of-section record	2.8.1.1
*EOP	CYBER end-of-partition record	2.8.1.2
*FFF	CYBER free-form-flag delimiter	2.8.1.3
*EOF	3600 end-of-file record	2.8.1.4
	- ASCII	
*BOP	Beginning of paragraph mode	3.2.1
*END	End of paragraph mode	3.2.6
*STP	Stop printing until user types a line	3.2.2
*EOF	Page eject	3.2.3
*HD	Lines down to *END form a page header	3.2.4
*FT	Lines down to *END form a page footing	3.2.5

Reserved words - Status Determination

DATE	Determine the date in the form dd/mm/yy	2.8.2.1
TIME	Determine the time in the form hhmmss	2.8.2.2
SIZE	Determine no. of records in the workspace	2.8.2.3
LINE	Determine the record no. of current record	2.8.2.4
COLM	Determine column no. of pointer position	2.8.2.5
CODE	Determine current charge-code in use	2.8.2.6
EDIT	Determine edition no. of last doc. copied	2.8.2.7
STOP	Determine value of x as set by x, command	2.8.2.8

ASCII Printing

'nP	Print n lines of formatted text	3.4.2
↑'nP	As above with character justification	3.4.3
↑m'nP	As above with continuous pagination	3.4.4

1. OVERVIEW

1.1 Introduction

TED is a console program (DAD Manual Ch.23-25) used for program development or for general text editing of documents stored on the document region (DR) of the CSIRO 3600 computer. The documents should consist of one or more records where each record contains up to 160 characters coded in BCD or ASCII code. TED is commonly the main link between the interactive system and the user, and can be accessed from any teletype-like device attached to the CSIRONET system (e.g. ASR and KSR teletypes, the CDC 713, Tektronix storage oscilloscopes, etc.). However users of DD210's should use its predecessor, FRED.

TED is logged into from a console by typing

*LG,charge-code,TED (LF)

directly after the colon. (LF) refers to line-feed. TED replies with the time and date, the line TED x.y = where x.y is the edition number of TED, then a colon on the next line. The printing of the colon tells the user that TED is waiting for a command. If he types in a command or command string followed by (LF) TED will execute the commands, type out any necessary or required lines, then finish with a colon to show that it is ready for a new command.

The user does not directly edit a document stored on DR. Instead he must request that the document be copied from the document region (DR) of the 3600 disc to a random access (RA) area on the drum - referred to as the users workspace. It is this copy which is edited - the original is unaffected. Thus the request C/TEST/ would copy a users document called TEST to his workspace area; he may then alter this copy and finally make the resultant document into a new 3600 document - e.g. the command M/TEST2/ would copy the contents of the workspace area back onto the DR of the disc with a name TEST2. This document could later be accessed in a similar way.

Both 3600 and CYBER 76 jobs can be prepared or edited, the job executed and the output examined without logging out of TED. When the user does log out of TED - via the Q command - or when he copies a document to the workspace then all the previous contents of his workspace are lost. The workspace contents can be irreparably damaged or lost by a careless command or an unexpected 3600 shutdown. Hence it is recommended that the user guard against serious losses by making the workspace into a 3600 document once every 20-30 minutes whilst logged into TED.

For a more detailed explanation of general concepts such as documents, disc, RA areas, etc. other CSIRO publications should be consulted, i.e. "Introduction to the DAD system", "Consoles and the DAD system", and the "DAD Manual".

1.2 Command Structure

Document editing is performed using commands rather than directly typing corrections into a document. Commands are all of the form,

mnc or mnc;text;

where

n = an optional integer (whose meaning varies with the command).

m = an optional modifier (whose meaning varies with the command) - modifiers are ^(up arrow), '(quote) or \(backslash).

c = a single character representing a required command.

text = any string of characters e.g. a document name, a charge code, a string of text, a number etc. The text string is mandatory for some commands and meaningless for all other commands.

; = a text delimiter. Delimiters can either be ! or ; or / . The three are equivalent; a choice is given to permit delimiters to be included in the text string. A fourth delimiter ' is used in special cases. Strings must be started and ended by the same delimiter.

e.g. a permissable command is

2D;*EQUIP;

which says delete the second occurrence of the word *equip - no modifier was used. The positions of modifiers and the integer are generally interchangeable. However more commonly the optional integer and modifier are not used.

On terminals which contain both upper and lower case letters any command or text string can be typed in using either lower or upper case letters provided the terminal is in the normal mode, i.e. BCD mode - the default option (section 1.7.1). However in special cases (e.g. documentation) ASCII mode (section 1.7.1) must be used and

then lower case text characters are not mapped onto upper case.

1.3 Command Strings

A command string consists of several commands all typed on the one line. The string is terminated by a line-feed (LF). The command string is then sent to the 3600 where it is interpreted (by TED) and executed, one command at a time, in order from left to right.

If an error is detected in the command string then ↑(up arrow) is typed beneath the command in error and a diagnostic message is given. All commands to the left of the error would have been executed, and all to the right - plus the detected error - ignored.

Commands may be nested in parentheses - any missing right parenthesis is assumed to be present at the end of the input line. An unmatched right parenthesis, or a blank (other than in a text string) is interpreted as the end of the command string. Any characters to the right are ignored. An integer immediately preceding the left parenthesis is always a repetition factor for the enclosed command string.
e.g.

(a) 2D/END/ delete the second occurrence of END
(b) 2(D/END/) delete the next two occurences of END
(c) D/READ/I/WRITE/D;60;I;61;

In (a) the number 2 is a modifier which has a special significance to the D command.
In (b) 2 is merely a general repetition count which causes the contents of the parentheses to be executed twice.
In (c) the command string requests that the next occurrence of READ in the workspace be deleted and (if successful) the word WRITE be inserted in its place, then the next occurrence of the number 60 be deleted and the number 61 be inserted in its place.

The situation of an unsuccessful search is discussed later (section 2.3.2)

1.4 The Pointer

All text editing in TED is performed relative to an internal (invisible) pointer which can point to any column of any record in the workspace. The pointer is initially at column 1 of record 1 of the document. However it is moved by any command which searches for a text string, or by a number of general pointer commands of the form nnc. Thus J moves the pointer to the start of the record, 3F moves it three records forward, B moves it one record back, etc. (refer section 2.3.1). The pointer position can be determined by various status-determining commands descibed in sections 2.8.2 and 2.9.2.4.

Alternatively the user may get the contents of the current record printed out as a final reply to the command string; this is often done as a default or can be done by the P command at the end of the command string. Then typing only a line-feed after the colon results in an up-arrow being printed. This points to the column at which the pointer resides. Successive line-feeds repeat the printing of the pointer position whereas typing blank then line-feed causes the current record to be output.
e.g.

```
:S/IT/P    search for next occurrence of IT and print
      WRITE (61,100) A
:(LF)
```

In this example the P command (to cause one line of output) is unnecessary as printing would occur by default. Note that TED is very literal-minded - it searched for the string IT, not the word IT! If the user then requested that the next occurrence of ,100 be deleted (via D/,100/) then the pointer would, after the deletion, move to the right parenthesis. If an insertion was then done (e.g.I/,200/) then ,200 would be placed where the ,100 was. The pointer would still point to the right parenthesis.

1.5 Display of Workspace

1.5.1 nP PRINT

A command string which operates on the users workspace usually results in the display of at least part of one record of the document. If nP is absent from the command string then the record printed is taken from the record at which the pointer ends. The command nP at any stage in the

command string asks that n records of the document be printed - starting from the current record. The pointer position is unaffected by this command. Several nP commands may exist in the one command string.

Most terminals can only print about 80 characters per line although up to 160 characters may be supplied to the console by TED. To allow for this two features have been incorporated - truncation and windowing. The first is a user option, the second is automatic and mandatory.

(a)Truncation

1.5.2 x. where x is a number from 1-160 causes any reply from TED to the user to be restricted to (x-1) characters per line. Any excess characters are not printed. At log-in time x=0; this removes any restrictions. Once the value of x is set it remains set until another x. command is used or until the user logs out of TED.

(b) Windowing

If, when a print command is requested, the pointer is in column 1-20 of the current record then the output lines are printed normally - no windowing occurs.

If the pointer is in column 21-160 then the output lines are printed such that, if the pointer points to column c of a record, then the output line is shifted c-20 positions left. The line is output with the character that was in column c-20 printed in column 1 on the output device; the pointer is therefore at the character printed in the 20th column.
e.g.

```
:JP    move pointer to column 1 and print
THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG.
:30.   restrict output to 29 characters
THE QUICK BROWN FOX JUMPED OV
:35R   move pointer 35 positions to right
FOX JUMPED OVER THE LAZY DOG.
:(LF)  display the pointer position
       ↑
```

Note that although the pointer was moved 35 positions to the right (thus to column 36) the output line was only shifted 16 places to the left (i.e. 36-20 positions). Had 5 lines been printed, then all five would have lost the first 15 characters of the record. Note also that the command 30. had no print command in it but still caused one

line of output to be printed (see section 1.5.1).

1.6 Boxes

Instruction strings or text strings which may be used frequently can be stored in reserved regions called boxes.

Each box is numbered and the contents of the box can be accessed at any time by using the box number. Thus the instruction I'2' would insert the contents of box 2 in the current line whereas I/2/ would merely insert the number 2. The instruction 1$ requests that the contents of box 1 be used as a command string and executed. These and other features (discussed in section 2.7) make boxes an extremely powerful facility.

1.7 Modes

1.7.1 Text modes

Characters are usually handled on the 3600 by giving each character an octal code and then packing these octal numbers together into 48-bit words. Two main codes are used.

(a) BCD

This is the standard code that the 3600 uses for character handling. Each character is coded as a 6-bit number and packed 8 characters per computer word. The 64 possible numbers in a 6-bit byte each represent one character in the CSIRO 64-character set. This code is the standard default used for storing documents arising from Hollerith card input, paper-tape input, console programs, or running jobs.

TED assumes all documents are in BCD mode unless the user directly specifies that he wishes to change modes (i.e. via the command 'C;name;)

(b) ASCII

In this mode characters are stored as 12-bit numbers - 4 characters to a 3600 word. This allows many extra characters to be coded, including lower case letters. However all non-printing characters (except backspace and horizontal tab) are deleted from any command string sent to TED and from any document copied by TED. ASCII documents can be input from binary cards, from paper-tape or from a

teletype; however most teletypes do not have a full ASCII set of graphic characters. In all cases the user must take special precautions to ensure that the ASCII document is not treated as BCD, or converted to BCD (refer DAD manual).

ASCII documents cannot be run as jobs on the 3600 or CYBER 76 computers, although they can be used as data for jobs which are set up to handle ASCII characters. More commonly, they are used by TED for preparing documentation, letters, reports, etc. These documents are usually directly output to an ASCII printing device, such as an ASCII line-printer or the IBM 2741 typewriter in Canberra.

Many special features have been incorporated in TED in order to allow flexible formatting of documents. These features are discussed in Chapter 3.

1.7.2 Command modes

(a) Normal

Many of the TED commands require text strings as arguments. In the normal mode these strings are delimited by either / or ; or ↓. The special delimiter ' may also be used when accessing boxes (or special reserved words). However if the user wishes to use some single characters as text, then he may switch from normal mode to relaxed mode.

(b) Relaxed

1.7.2.1 ↑; RELAXED MODE ON

This is entered via any of the three commands ↑/, ↑; or ↑↓. In this mode delimited text strings may have a different format. They can be replaced by just a single character. Thus if TED expects a text delimiter and does not find one it uses the next character after the command as the text string. Once entered relaxed mode remains switched on until a command is given to turn it off.

1.7.2.2 / or ; or ↓ RELAXED MODE OFF

Relaxed mode is turned off when a / or ; or ↓ is encountered as a command. Encountering these as delimiters does not turn the mode back to normal.
e.g., the command string

↑/CAPSIZE↓

would turn on relaxed mode (↑/), copy a document called A (CA), print the first line of the document (P), search for the first occurrence of the letter I (SI), insert a line containing only the letter E (ZE), and then switch the user

back to normal mode (↓).

1.8 Teletype Commands

1.8.1 Local Editing

The user of a console teletype has several commands that he can use to modify input and output strings. These commands can be used on any teletype-like device (except the IBM 2741) and for any console program or any communication with the 3600 computer. They are commands which are obeyed by the PDP-11 computer to which the teletype is attached. The 3600 is completely unaware of this editing. Fuller details are given in Chapter 22.3.2 of the DAD manual.

(a) Input

Any input line is set up and stored, character by character, in a PDP-11 buffer area until the user types line-feed. This action sends the full record to the 3600 computer. The user may alter the line as it is being set up by the following commands. If in ASCII mode then each usage of these commands must be preceded by a DLE or CTRL/P character. The notation CTRL/P means that on some teletypes the DLE character is typed by typing P whilst the CTRL button is depressed.

1.8.1.1 BELL (CTRL/G)

Removes the last character stored in the PDP-11 buffer

1.8.1.2 CR (carriage return)

Unless this is directly followed by LF it erases the current line from the buffer. Hence if a user decides he doesn't want the input line he can erase it and recommence a new line.

1.8.1.3 VT (CTRL/K)

Displays the current contents of the line stored in the PDP-11 input buffer.

(b) Output

1.8.1.4 CR

Immediately discard the remainder of the current record being output.

1.8.2 Interrupt

Two commands may be used to interrupt the 3600 computer at any time.

1.8.2.1 DLE A (CTRL/P A)

Interrupt the 3600 and stop it processing the present command. This can be used to interrupt a long loop e.g. @(S/A/jp)

1.8.2.2 DLE T (CTRL/P T)

Interrupt the execution of the present command and log out of the program. This command can be used instead of Q, or when the program is stuck in a loop. If done whilst performing a MABEL short request (section 2.5.1) the user is logged out of the MABEL region and returned to TED. Otherwise he is logged out of TED and returned to MABEL !

1.9 Summary of Basic TED Commands

TED contains several dozen different commands which allow the user great power in manipulating documents. However the beginner can manage quite well with only a small sub-set of these instructions. A summary of the most common ones are given below, along with the section in which they are fully explained. Modifiers are not included as for most cases they are unnecessary.

COMMAND	MEANING	SECTION
C/doc/	Copy the named 3600 document to the workspace	2.1.1
M/doc/	Make the TED workspace into a new 3600 document	2.1.2
K/doc/	Make the workspace into a 3600 document, run it as a CYBER job, copy result back onto workspace.	2.5.2.1
K//	Extend wait time for another minute	2.5.2.1
D/text/	Delete the next occurrence of the text string	2.4.1.4
I/text/	Insert the text at current pointer position	2.4.2.2
S/text/	Search for the next occurrence of the text string	2.3.2.1
Z/text/	Insert a new line after current line.	2.4.2.3
+	Insert a blank character at pointer position	2.4.2.1
-	Delete the next character	2.4.1.1
0	Move pointer to start of document	2.3.1.1
J	Move pointer to start of current record	2.3.1.2
L	Move pointer one position left.	2.3.1.4
R	Move pointer one position right	2.3.1.5
B	Move pointer one record back	2.3.1.6
F	Move pointer one record forward	2.3.1.7
E	Erase the current line	2.4.1.2
Q	Log out of TED	2.6.1.1

In the above commands doc is any 3600 document name and text is any string of text. The modifer, n, where n is an integer can be used before any command. For all except the first four commands it is merely a repeat factor.

2. COMMANDS

2.1 Document Commands

The prime function of TED is to prepare new documents - generally by modifying copies of existing documents. This means several document manipulation commands are required. These are all of the form

mnc;name;

where name is a 1-8 character name of a 3600 document, m is a modifier (generally absent), c is the command and n is a number (generally absent) which specifies the edition number of the document in question. If n is absent then any command which references a document will reference the highest edition number, and any command which makes a new document will give it an edition number one higher than the previous highest.

2.1.1 nC;name; COPY

Edition n of the named document is copied from the 3600 document region to the workspace. If n is absent the highest edition is copied. The copy can then be edited without affecting the original document. The document should

(a) be in BCD/binary mode
(b) contain no more than 160 characters per record
(c) be less than 100 sectors long (1 sector =256 48-bit words i.e. 2048 BCD characters or 1024 ASCII characters)

The C command, if successfully obeyed, clears the previous contents of the workspace. If this was an ASCII document (i.e. the terminal was in ASCII mode) then the terminal would be switched to BCD mode and any further commands in the string ignored. No other command causes this switch.

Thus 1C;JOBA; will copy edition 1 of a document called JOBA onto the workspace - JOBA having previously been stored on the document region (DR) under the users charge-code.

Relocatable binary documents may be copied using this command, although any printed lines will be meaningless. Absolute binary documents should not however as they have a record length greater than 20 computer words and truncation of records will occur.

2,1,2 nM;name; MAKE

This command copies the contents of the users workspace back to the 3600 disc (DR region) and labels it as edition n of the document with the chosen name. If n is absent the edition number is one higher than the previous highest edition with the same name and charge-code. The mode of the document - BCD, binary, or ASCII - is preserved.

The MAKE command in no way affects the workspace document or the pointer position; further editing of the workspace may continue. The contents of the users workspace are irretrievably lost whenever one copies another document with the C command or whenever one logs out of TED.
e.g.
If the highest edition of a document called JOBA is edition 2 then the instruction

C/JOBA/M/JOBA/

would copy edition 2 onto the workspace and then copy it back to disc as a new document - edition 3 of JOBA. The workspace copy and edition 2 would both still exist.

2,1,3 nV;name; INSERT

The named 3600 document is copied to the workspace where it is inserted in the current RA document, record by record, immediately after the current record. At the end of the operation the pointer is in column 1 of the last record to be inserted. The command is inefficient especially if one wishes to insert large documents as each record must be inserted one at a time and TED requires about twice the workspace of the document being inserted. Where possible it is better to concatenate documents using the # and & commands.

2,1,4 n&;name; ADD

This command must be preceded by a # command in the same command string. It causes the named document (edition n) to be added to the new document being created as a result of the # command. The & command allows many small documents to be concatenated into one large document (999 sectors maximum). Record lengths are no longer restricted to 160 characters; hence absolute binary documents can be copied using this command.

2,1,5 n#;name; CREATE

This creates a new 3600 document (with the name

specified) out of previous 3600 documents, parts of the workspace area, or both, depending on the contents of the rest of the command string.

(a) The remaining instruction string may request that the pointer be moved throughout the workspace and various lines be printed (using nP or ? commands). Any such lines will, instead of being printed, be added onto the end of the document being created. The # command thus allows the user to select any parts of his workspace document and string them together to create a new document. Note that this new document will be precisely what would have been printed - including any effects due to truncation and windowing. Hence it is recommended that users place a J command before each P command

(b) The remaining instruction string may have one or more &;name; requests in it. This will cause the document so named to be added to the end of the document being created.

e.g.

C/EQUIP/#/JOBA/J6P&/FORTRAN/3&/DATA/

This string copies a job called EQUIP onto the workspace, and creates, on DR, a new document, JOBA, consisting of the first six records from EQUIP, then the DR document FORTRAN, then the DR document DATA (edition 3). The workspace copy of EQUIP remains unaffected with the pointer in column 1 of record 1.

The instruction string

#/DOC1/&/DOC1/&/DOC2/

would concatenate the two highest editions of DOC1 and DOC2 and re-form the result as a higher edition of DOC1. The workspace area is unaffected and completion of the instruction string is indicated by a colon.

The # command is not completed until the entire text string has been processed or the command #// is encountered. Thus one should not have two #/name/ commands in the one string, nor refer to the document being created until a later string unless #// is used.

2.1.6 #//

This causes a preceding #/name/ command in the same command string to be completed. Normally it would not be

completed until the end of the command string.Thus the string

#/DOC1/012P#/DOC2/12F12P

although acceptable to TED would not create two separate documents - the command #/DOC2/ would be ignored. However the command string

#/DOC1/012P#//#/DOC2/12F12P

is quite acceptable and will create two new documents.

2.1.7 'nC;name; ASCII COPY

This command, if successful copies an ASCII document onto the workspace area. If the teletype was in BCD mode then the mode is changed, any further commands in the string are ignored and TED replies with an up-arrow, then DOCUMENT COPIED, followed by line-feed; if the mode was already ASCII then the first line of the document is printed and any further commands in the string are obeyed. A similar behaviour occurs when switching back to BCD mode via the C;name; command -i.e., further commands in the string are ignored.

2.1.8 n];name; SAVE BOXES

The present contents of the users boxes (section 2.7) are saved as a 3600 document (edition n). The contents of the boxes are otherwise unaffected. The document formed contains two records per box, an identification record and a text record. Since the identification record contains the box length in characters, the text records should not be modified. Pairs of records may be deleted from the document.

If the number n is absent from the command the edition number of the new document is one higher than the previous maximum.

On some teletypes] is typed via SHIFT/M. This can be remembered by the mnemonic M for MAKE since a 3600 document is made.

2.1.9 n[;name; REPLACE BOXES

The contents of boxes which have been saved via the] command can be replaced by the [command. If the box numbers which were saved are different from those currently in use then additional boxes will be created. If they are the same as existing boxes then the contents of the boxes

will be replaced and the old contents lost.

On some teletypes [is typed via SHIFT/K. This can be remembered by the mnemonic K for copy.

2.1.10 U:c.c.; SET CHARGE-CODE

This command allows documents of other users to be accessed. If any commands following this but in the same command string refer to a 3600 document then TED assumes the document is under the charge-code specified in the U command rather than the charge-code used to log into TED.
e.g.

U/CBC*****/C;RIOTGEN;

would copy the document RIOTGEN from under the CBC***** charge-code

The 3600 DAD system does not allow a user to create a document under another persons charge-code. Hence the commands M, #, and] cannot be used unless the document is made under the charge-code of the user. Thus if the user logs into TED under the charge-code CBCCS*XA then the command

U/CBC*****/C;TOTEGEN;M;TOTEGEN;

is illegal. However the following is legal

U/CBC*****/C/TOTEGEN/U/CBCCS*XA/M/NEWDOC/

This last instuction copies the document TOTEGEN from under the CBC***** charge-code and makes it into a new document NEWDOC under his own charge-code.
This could also be achieved in the following two ways

(a) :U/CBC*****/C;TOTEGEN; first copy document
:M/NEWDOC/ now make it

(b) :#/NEWDOC/U/CBC*****/&/TOTEGEN/

All three cases copy the CBC***** document TOTEGEN onto the 3600 document region and under the users charge-code. In the first two cases the users workspace is overwritten by TOTEGEN. In the third case (i.e.(b)) the users workspace and the pointer position are unaffected.

2.1.11 U;; RESET CHARGE-CODE

This causes the charge-code to be reset to that of the user. Thus if the command U:c.c.; occurs on a line the user can revert to his own charge-code via the command U;;. This

option is often useful when using TED boxes to simulate a console program.

2.2 Writing on the workspace

Instead of copying a document to the workspace and altering this with commands the user may directly type in records onto the workspace by using the reserved document name '?' in conjunction with the C or V commands

2.2.1 C;?;

Upon finding this command in a command string TED replies with

TYPE IN A DOCUMENT

then a colon. The user can then directly type in his document line by line. Each line is ended by typing (LF) or (CR-LF) then waiting for a colon to indicate that TED is ready for the next line.

In this mode of operation the following apply.

(a) The document is ended (i.e. the C;?; command completed) when the user replies to a colon with only (LF) or (CR-LF); the pointer is then positioned at column 1 of record 1 of the new document. Any further commands in the command string are then executed.

(b) The document cannot be ended by a line-feed until at least one record has been entered in it.

(c) To type in a blank line at least one blank character must be typed before typing line-feed.

(d) Reserved character strings,e.g. *EOS, *EOP, *EOF, *FFF may be used (section 2.8.1).

(e) Tabs may be used (see section 2.6.1)

(f) The previous contents of the users workspace is lost

If the copy command changes the mode from ASCII to BCD then at the end of the document TED will reply with

'DOCUMENT COPIED'

If this occurs the copy command will be the last command executed in the command string. An example of the copy command is given in chapter 4.

2.2.2 'C;?;

This command allows an ASCII document to be typed in directly onto the workspace. The above conditions apply except

(1) TED replies with

ENTER ASCII DOCUMENT

(2) Reserved words, which may be used, are *BOP, *END, *STP, *HD, *FT, *EOF (section 3.2).

2.2.3 V;?;

The command is the same as the C/?/ command except that it does not lose the current contents of the workspace. Instead the lines typed are inserted as records from the current pointer position. The mode of the document is maintained however. Thus ASCII records are inserted in ASCII documents and BCD records in BCD documents. It is not possible to mix modes. Upon exiting from the command the pointer is positioned at column 1 of the last record inserted.

2.3 Pointer Manipulation Commands

These commands allow the pointer to be moved throughout the workspace without altering the actual contents of the document. In each case the number n before the command is merely a repetition factor, e.g., 5F=3F2F=FFFFF

Pointer commands are as follows

2.3.1 General

2.3.1.1 0 ORIGIN

Return the pointer to column one in record one of the workspace document.

2.3.1.2 J JUSTIFY

Move the pointer to column 1 of the current record.

2.3.1.3 ^J

Move the pointer to the end of the record. The records are stored as 3600 word images with the last word of each record being blank filled if necessary. Hence BCD records are stored as multiples of 8 characters and ASCII lines as multiples of 4. The ^J command moves the pointer to the last 3600 word boundary. This is not necessarily the last non-blank character in the line.

2.3.1.4 L LEFT

Move the pointer one column left in the current record. If the pointer is in column 1 the command is ignored.

2.3.1.5 R RIGHT

Move the pointer one column right in the current line. If the pointer is in column 160 the command is ignored. The pointer may be moved beyond the end of the current record however.

2.3.1.6 B BACK

Move the pointer back one record. The current column position is unaltered. If the pointer is in record one the command is ignored.

2.3.1.7 F FORWARD

Move the pointer forward one record. The current column position remains unaltered. If the pointer is at the last record the command is ignored.

2.3.2 Specific - the search command

2.3.2.1 S;text; SEARCH

Search forward for the next occurrence of the text string. If this string is found, the pointer will be positioned immediately after the last character in the string. Note

(1) Searching starts from the current pointer position to the end of the current record and then continues on from column 1 of the next, and so forth, until the string is found.

(2) The object text string (i.e. the one searched for) must be contained within one record.

(3) If the search fails then the pointer will be positioned at column one of the last record in the document and all subsequent commands ignored unless the search command is preceded by the modifier @N (section 2.6.2), or unless it is within parentheses.

In this last case all subsequent commands up to the

matching right parenthesis are ignored and any following commands will be executed. This is the only conditional branching facility in TED - it applies to all other commands (↑S, D, A) which search for text strings.
e.g.
If the text string QWERTY does not exist in a users workspace then the command string

```
O(S/QWERTY/JP(FV/?/)S/100)/JP)3P
```

would have the same overall effect as 0@F3P. Hence all commands up to the matching right parenthesis (not necessarily the next right parenthesis) are ignored. Any parentheses within text strings are ignored in this matching process.

nS;text; = n(S;text;) i.e. this searches for the nth occurrence of the text string.

2.3.2.2 ↑S;text; SEARCH BACK

Each record (including the current one) is searched from left to right exactly as in the case of the S command. However if the search fails on any one record then it recommences from column 1 of the previous record, etc. If the search fails the pointer finishes at column 1 of record 1; again all further commands up to the matching right parenthesis or end of string are then ignored.

This ↑ modifier modifies the D and A commands in a similar way.

2.4 Text Editing Commands

2.4.1 Deletion commands

2.4.1.1 - REMOVE

Delete the character in the current column. The pointer is not moved but in effect all characters to the right of the pointer are moved one space left.
e.g.

```
:JP  justify and print
      PRINT,100,ALPHA,BETA,CX,D
:D/LPHA/D/ETA   pointer ends on the comma after the B
      PRINT,100,A,B,CX,D
:2R-JP  move pointer 2 right and remove the X
      PRINT,100,A,B,C,D
```

2.4.1.2 E ERASE

Erase the current record. The pointer is moved to the corresponding column of the next record. @E erases all following records including the current record provided it is not the first record. It is impossible to remove all records in the workspace.

2.4.1.3 ↑E ERASE BACKWARDS

Erase the previous record in the document. The pointer is unaltered. ↑@E or @↑E will erase all records prior to the current one.

2.4.1.4 D;text; DELETE

Search for the object character string and delete it. This command behaves exactly as the search command does except that it deletes the string. Thus the pointer is positioned immediately after the matched character string. nD;text; deletes the nth occurrence only of the text string

2.4.1.5 ↑D;text;

As for ↑S;text; except the text when found is deleted.

2.4.2 Insertion commands

With these commands the ↑ modifier has no significance except when it is used in the form ↑/, etc., to set relaxed mode. The number n immediately before an insertion command is always a repetition factor, and causes the instruction to be repeated n times.

2.4.2.1 + INSERT BLANK

A blank is inserted in the line just before the current pointer position. All characters to the right are moved one position right. The pointer also moves one position right. This has the same effect as the command I/ /.

2.4.2.2 I;text; INSERT

Insert the object text string in the current record just before the current pointer position. The current character, the pointer and all characters in the record

which are right of the pointer are all shifted sufficient columns right to allow the object text to fit in the line. Thus the pointer is positioned to the right of the last character inserted. Characters shifted past column 160 are lost.
e.g.

```
:P  print current line
FORMAT(8H RESULT:,I4)
:(LF)  display current pointer position
                  ↑
:I/3H A=,/   Insert 3H A
T(8H RESULT:,3H A=,I4)
:(LF)  (note windowing effect on output)
                  ↑
:JP  print the full line - no windowing.
1000  FORMAT(8H RESULT:,3H A=,I4)
```

2.4.2.3 Z;text; INSERT RECORD

The text string is inserted as a new record. The pointer will be moved to column one just prior to the insert and will end up positioned just after the last character of the inserted string when the insertion is complete.

```
e.g.
 :J2P  print next two lines
 1     READ (60,100) A,B,C
       WRITE (61,200) A,B,C,D,E,F
 :Z/      READ (60,110) D,E,F/BJ3P
 1     READ (60,100) A,B,C
       READ (60,110) D,E,F
       WRITE (61,200) A,B,C,D,E,F
```

2.4.3 Exchange commands

2.4.3.1 % SWAP CHARACTERS

The current character is interchanged with the character on its immediate right. The pointer is moved one character to the right, i.e., it follows the character.
e.g.
Suppose the command S/B/ produced

```
ABCDE
```

then the command 2% would produce the result

```
ABDEC
```

with the pointer still at C. Note that the S/B/ command resulted in the pointer ending at the letter C.

2.4.3.2 ↑% SWAP LEFT

Interchanges the current character with the one on the left. The pointer is moved one column to the left (i.e. it follows the character).
e.g.

```
:JP
ABCDEF
:3%
BCDAEF
:(LF)  display pointer position
  ↑
:↑2%
BACDEF
:(LF)
 ↑
```

2.4.3.3 X EXCHANGE

Exchange the current record with the one following it. The pointer is moved forward (i.e. follows the record). The column positon of the pointer is not changed. If the pointer is at the last record of the document, the command is ignored.
e.g.

```
:J2P  print the next two lines
THIS IS INITIALLY THE FIRST LINE
AND THIS IS THE SECOND.
:XB2P
AND THIS IS THE SECOND.
THIS IS INITIALLY THE FIRST LINE
```

The command nX moves a record down n lines or to the end of the document, whichever comes first. If one of the records swapped is a binary record then the X command can cause a change of mode to BCD. Hence binary end-of-section records should not be exchanged.

2.4.3.4 ↑X EXCHANGE UP

Exchange the current record with the one preceding it. The pointer moves with the record. Hence the command 10X↑10X would have no nett effect, unless there are less than ten lines to the end of the document.

2.4.4 Replacement commands

2.4.4.1 Y WIPE

Replace the current character with a space. The pointer is moved one column to the right. Hence 6Y would relace the next six characters with a space
e.g.

```
:S/1,00 /
1,00 1,01 2,00      3,00
:4Y
1,00      2,00      3,00
:(LF)
          ↑
```

2.4.4.2 A;text; ANNIHILATE

Search for the object character string and replace each character with a space. The pointer will be immediately to the right of the last character blanked out.

nA/text/ searches for and blanks out the nth occurrence of the text whereas n(A/text/) searches for and blanks out the next n occurrences of the text.

2.4.4.3 ↑A;text;

Search upwards (as in the ↑S;text; command - section 2.3.2.2) and replace the text with spaces.

2.4.4.4 G GROW

The current record is split into two at the pointer position. All characters to the left of the pointer remain in the first record, and the others are moved into a second record. The pointer finishes positioned at column 1 of the second record, i.e., at the same character as before the G command. e.g.

```
:JP  print current line
1000  FORMAT (12H SIZE OF A =,I4, 7H METRE,)
:S/=,/G5+I;*;B2P   Split the current line into two
1000  FORMAT (12H SIZE OF A =,
     *I4, 7H METRE,)
```

2.5 Execution Requests

2.5.1 MABEL short requests

MABEL short requests (DAD Manual Ch.23.2) are simple requests which may be made from a console without logging into a console program. They are handled by a special console program MABEL which normally handles requests made when users aren't logged into a console program. TED allows these requests to be made without having to log out. The requests are all of the form

*/MABEL request/

and may appear in commands strings like any other TED command. Permissible MABEL requests are listed below. In each case the charge-code field may be empty; then the charge-code used is that of the user. In the first four requests the charge-code, if present, must be the same as that used to log into TED.

*/BK,c,c,,id/ Enter a request for the named 3600 document to be broken in and run on the 3600.

*/CY,c,c,,id/ Enter a request for the named 3600 document to be run on the CYBER 76.

*/EX,c,c,,id,t,q/ Enter a request for the named 3600 document to be run on the 3600 as a t minute job with priority q.

*/DL,c,c,,id,ed/ Delete edition, ed, of the named 3600 document from the document region (DR).

*/LC,c,c,,id,ed,SVnn/ Locate edition, ed, of the named 3600 document and save it for nn days.

*/LP,c,c,,id,ed,dest,SVnn,nC/ Request that the document be line printed.

*/PL,c,c,,id,ed,dest,SVnn,nC/ Request that the document be plotted on the small plotter.

*/PB,c,c,,id,ed,dest,SVnn,nC/ Request that the document be plotted on the large plotter.

*/CP,c,c,,id,ed,dest,SVnn,nC/ Request that the document be card punched.

*/TP,c,c,,id,ed,dest,SVnn,nC/ Request that the document be paper-tape punched.

Several of the above fields may be missing, ignored, or in different order (Refer to the DAD manual).

e.g.

*/LP,,A40,MEB,SV3/

requests that the latest edition of the users document, A40, be output on the Melbourne line-printer and also saved as a 3600 document for the next three days.

*/EX,,JOBA,3,G1/
requests that the users document, JOBA, be placed on the execution list, EL001, although it is to be a three minute job.

Note that LOGIN (LG) is not a valid MABEL request from within TED.

2.5.2 Breakin request

The *CY MABEL short request has been combined with C and M commands in order to allow the user a simpler way of running jobs.

2.5.2.1 K/name/ KYBER

The users workspace is made into a 3600 document and a CYBER job execution request is made on this document. The terminal then remains dormant until a suitable output document is returned from the CYBER or until one minute is up. A suitable output document is one which has the users charge-code and terminal number. The first suitable document to return is copied to the users workspace region and the user is informed of the document name. If no such document returns after one minute the terminal is reawakened and control returned to the user with the diagnostic,

1 MIN, NO CYBER RESPONSE

The users workspace then remains unaffected. Any document that is returned from the CYBER to the 3600 as a result of the K/name/ command will be stored on the 3600 DR region irrespective of its hardware disposition type unless it was specifically disposed to some station other than the users terminal. Such documents are in all other respects (e.g. name, translations, etc.) treated as though they were to be disposed to the hardware device specified. Documents which are disposed to the DR region of the 3600 are given priority in returning from the CYBER. They should not be over 100 sectors long however or else they will be truncated.

Note that each time the K request is made the workspace is made into a 3600 document even if the rest of the command is not completely executed. The exception to this is the command K//.

2.5.2.2 K//

If after a K;name; command one minute elapses before a

CYBER document is returned to TED the user may extend the wait time by a further minute via the K// command. If a document returns from the CYBER with the correct charge-code and terminal number before the user types in the additional K// command then as soon as the command is typed, this document will be copied to the users workspace. The K// command can be repeated as often as necessary. If it fails to attach a document then any other commands following in the same command string are ignored.

2.6 Miscellaneous

2.6.1 General

2.6.1.1 Q QUIT

Log out of TED. This command terminates a session in TED - the users workspace is immediately lost. The user is then returned to the MABEL area.

2.6.1.2 @

Shorthand for 10,000 when used as a repetition count for commands. Some caution should be used with this command since it is possible to put TED into a long loop. If it is suspected that the command string is looping the interrupt command DLE A (or CTRL/P A) will terminate execution of the command string.

Thus @E will erase all following lines in a document merely by altering the value of an internal pointer. However the command @(J+F) will insert a blank space in every line of the document but upon reaching the end of the document the F command will be ignored whereas the rest of the command, J+, will continue to be executed on the last line until the 10,000 requests are made.

2.6.1.3 n" SET TABS

This command is used to set tabs in an internal tab register. If n is absent or zero the tab register is cleared. Otherwise each n" command sets a tab at column n. At log-in time the tab register contains one tab in column 7.

Tabs are activated by the occurrence of an " in either of two places (1) delimited text strings, (2) when typing text directly from a console (i.e. using the C/?/ or V/?/

commands). In these cases " is used as a reserved character in the string to indicate a skip to the next column at which a tab is set. Note that the tabs are set relative to the start of the delimited text string, and not relative to column 1 of the record. Hence if a tab is set at column 7 then the command S/"READ/ is identical to S/ READ/ and would search for any occurrence of six blanks followed by 'read' -not just 'read' starting in column 7.

To alter the tabs from the default of 7 for FORTRAN decks to tabs at columns 10, 20, and 40 for COMPASS decks the user would type

"10"20"40"

If the user wishes to insert an " in the actual text then he should either clear the tab register, or else insert " as a single character in the delimited text string.

2.6.1.4 (LF) LINEFEED

When line-feed or carriage-return line-feed (CR-LF) is typed immediately after a colon, the reply is a line containing only an up-arrow at the current pointer position. The command assumes that the last command given previously was a P command. Hence the up-arrow points to the current character assuming the current record had been printed. If the windowing effect occurred, then the arrow is automatically at column 20 on the terminal.

If (LF) is preceded by any other command then it merely notifies the PDP-11 that the input string is complete and can be sent to the 3600.

2.6.1.5 n= SKIP

Skips to the n-th comma in the command string and resumes execution of the commands from there. This command is most useful when following an S, D or A command as then it effectively becomes a conditional skip - conditional on whether or not the search is successful. Commas in text strings or another box are ignored in the count.

2.6.1.6 , COMMA

The comma in a command string is ignored by TED except during execution of the n= command.

2.6.2 Search modifying commands

The following commands set restrictions on search commands (i.e. D, A, and S commands). The restriction occurs only on the next search command which is executed.

2.6.2.1 nW RESTRICT WIDTH

The search is only performed over columns 1 to n (inclusive) of each record. Thus

```
0@(1WS/*/JPF)
```

says, starting from the origin of the document print all records which contain an * in column 1.

```
7W2(S/1000/P)
```

says search for the next occurrence of the number 1000 in columns 1-7 of a record and print the record; then search for the following occurrence of 1000 anywhere in the rest of the document and print that record.

2.6.2.2 nN COLUMN START

The next search command searches each record from column n onwards. Columns 1 to (n-1) are ignored in the search. Thus

```
7W7NS/P/JP
```

would search for the next occurrence of the letter P in column 7 of a record.
The command string

```
07WD;1000;I;2000;07ND!2000!I!1000!
```

would delete the first occurrence of 1000 which occurs in columns 1 - 7 of a record and replace it by 2000; it would then delete the first occurrence of 2000 which appears in columns 7 - 160 of a record and replace it by 1000.

2.6.2.3 @N LINE SEARCH

Restrict the next search command to the current record,Thus

```
S/WRITE/@ND/A,/
```

would search for the next occurrence of the text WRITE and then delete the first occurrence of the text A, that appears after this but in the same record.

The commands nN and @N, although quite different in meaning, are mutually exclusive and if both are used to modify the same search command, then only the last will be used.

2.6.2.4 \ BACKSLASH

This modifier makes the search commands (D, A and S) more general rather than restricting them. It has four different meanings depending on how it is used.

(1) Preceding a search command - when doing the search and comparision all blanks are ignored.
e.g.
\D/WRITE (61,2000)A/
would look for and delete the next occurrence of the sequence WRITE(61,2000)A , ignoring any blanks whilst searching. When a suitable text string is found, it is deleted. Any imbedded blanks in the string would also be deleted.

(2) As a one character text string of a search command - this causes a search to be made for the first non-blank character.
e.g.
06N6Ws/\/P
would search for the first non-blank character in column 6 of a record; thus it might be used to search for the next continuation card in a FORTRAN deck.

(3) As part of a search text string - When the search is made the \ is able to match any character including blanks. More than one backslash can be used in the delimited text string.
e.g.
@(7WS/10\\/JPF)
would go through a document, e.g. a Fortran program, and print out all records which contain within the first 7 columns the number 10 followed by any two characters - including blanks.

(4) If the user wishes to actually search for or delete the character backslash then he needs to combine the uses of \ described in (1) and (2) above. Thus the command
\D/\/
would search for and delete the next occurrence of a \.

On some teletypes \ is typed via SHIFT/L.

2.7 Storage Using Boxes

Each user of TED has a special storage area of 180 48-bit words in which he may store frequently accessed command strings or text strings. Each such string is given a number from 1 - 32767 by the user and is stored in part of

the special area called a box. The string can be accessed by various commands which refer to the box number. The user may have up to 40 boxes provided the total number of characters stored in the boxes is no greater than 1440 BCD characters or 720 ASCII characters. The contents of boxes may be stored on the 3600 DR region for later access. This feature makes boxes very useful for their application as small console programs.

2.7.1 Storage

The following commands store information in boxes. Any previous contents of the box are lost. The number n before a command is the box number and should be in the range 1 - 32767. If it is zero or absent it is taken to be 1. Some box numbers above 1000 have special meanings for ASCII mode documents.

2.7.1.1 nH;string; HOLD

This command stores the string in box number n. The string can be a text or command string. The contents of a box can be cleared by entering a null string (i.e. nH//); this reduces the number of boxes in use. The workspace is unaffected by this command.

2.7.1.2 n< STORE

Store a copy of the current record in box n. The current record and pointer are unaffected

2.7.2 Usage

Box contents can be used in the following ways

2.7.2.1 n$ EXECUTE

Execute the contents of box n as a command string. The string is executed in exactly the same way as any other command string. Thus the $ command and other box commands can be included in the string.

2.7.2.2 n> REPLACE

Replace the current record with the contents of box n. The contents of box n are otherwise unaffected.

2.7.2.3 x'n' USE BOX n

(a) n not 0

If x is any command which operates on a text string (i.e. Z, I, S, A, D, C, V, M, K, H, ?, #, &, [,] or U) then a number n delimited by ' tells TED to take the text string out of box n and operate on that.
e.g.
3H;ID=CBCCS*XA;S'3'
would have the same effect on a document as
S/ID=CBCCS*XA/
However in the first case the text string remains held in box 3 and can be used in other instructions.

(b) n=0

The command is exactly as for case (a) except that box zero cannot be saved. Instead whenever a call to box zero is made TED prints a colon on the teletype and waits for the user to type a line. This line is then used as the text string to the instruction. The instruction is executed and the rest of the string processed.
e.g.
The command
U'0'C/JOB/
would type a colon then wait for the user to type in a charge-code. It would then look for and copy the document JOB under the requested charge-code. This type of instruction is most powerful when stored in a box. Using the $ command a command string can be executed in which a question can be typed on the console, the user types in a reply which can then be executed as part of a text string, i.e., a search made or deletion performed, etc., then perhaps a reply made. In this way the boxes seem to communicate with the user as in a console program.

Special care must be taken, however, since box zero operates in units of 3600 words. Thus for, say, an insertion in BCD mode inserting the letter A would insert the letter A followed by seven blanks.

2.7.2.4 x'state'

If x is any command which operates on a text string and state is one of status-determining reserved words described in section 2.8.2 . i.e. DATE , TIME , SIZE , LINE , COLM , CODE , EDIT , STOP then the value of the particular state is determined and this value is used as a normal text string for the command.
e.g. Z'DATE' would insert in the users document a new line containing the current days date.

2.7.3 Clearing Boxes

2.7.3.1 nH;; CLEAR BOX n

TED can, at most, handle 40 boxes at any one time. This command clears box n and hence decreases the number of boxes used by one.

2.7.3.2 [// CLEAR ALL BOXES

All of the users boxes are cleared and the box count is decreased to zero.

2.8 Reserved Words

2.8.1 Binary records

These words allow special binary records to be put into documents via TED. The conversion to a binary record is made when the workspace is made into a 3600 document and the reverse conversion when and if this document is re-copied back to workspace.

2.8.1.1 *EOS END-OF-SECTION

*EOS in columns 1-4 of a record is treated as a CYBER end-of-section record and is converted to a binary record when the workspace is made into a 3600 document. This record is converted back to a *EOS when the DR document is copied to the workspace

2.8.1.2 *EOP END-OF-PARTITION

*EOP in columns 1-4 of a record causes the record to be treated as a CYBER end-of-partition record. As above this is converted to a binary record when on the DR region.

2.8.1.3 *FFF FREE FORM FLAG

*FFF in columns 1-4 of a record causes the record to be treated as a CYBER free-form-flag to delimit free-form

binary records. The flag is treated the same as a card with all of columns 1 and 2 punched.

2.8.1.4 *EOF END-OF-FILE

A *EOF in columns 1-4 of a record causes the record to be treated as a 3600 end-of-file.

2.8.2 Status determining

These words when delimited by the special delimiter ' can be used in place of a text string for any command which requires one. When TED encounters one of these reserved words it determines the value of a specific parameter and then uses this value as the text string for the command. Thus

Z'DATE'

would insert into the workspace a line containing the day's date. The command used may be A, C, D, H, I, K, M, S, U, V, Z, #, &, [,], or ?. It is most frequently used, however, with the ? command (section 2.9).

Possible reserved words are listed below -

2.8.2.1 DATE

This requests the current day's date in the form ddmmyy, where dd = day mm = month and yy = year.

2.8.2.2 TIME

This requests the time of day in the form hhmmss, where hh is the no. of hours, nn is the no. of minutes and ss is the number of seconds.

2.8.2.3 SIZE

This requests the number of records curently stored in the users workspace.

2.8.2.4 LINE

This requests the current record number at which the pointer resides.

2.8.2.5 COLM

This requests the current column number at which the pointer resides.

2.8.2.6 CODE

This requests the current charge-code

2.8.2.7 EDIT

This requests the edition number of the last document copied into the users workspace from DR.

2.8.2.8 STOP

This requests the value of x as set by the last x, command (section 1.5.2).

2.9 Print Commands

2.9.1 General

The print command, nP, as discussed in section 1.5.1 is the general command for printing n lines of output from the workspace region.

2.9.2 Specific

The ? command has ben modified from earlier editions of TED so that it now requires a text string. It requests that the text string be output to the users terminal without reference to the workspace and without altering the pointer position. Unlike other commands it is rarely used in its most simple form; hence some of its special uses are given below.

2.9.2.1 ?/text/

This command is only useful when stored in a box as part of console program. As such it can be used to give instructions to a user.
e.g. assume box 1 contains
```
?/TYPE IN A CHARGE-CODE/U'0'2$
```
then upon typing $ to start the console program , TED would type the reply
```
TYPE IN A CHARGE-CODE
```

then CR-LF and colon. The user would then type a charge-code and this would become operative for the rest of the string encountered in box 2.

2.9.2.2 ?'n' DISPLAY BOX n

If n is a number then TED uses this as a box number and types out the contents of that box. The workspace is unaffected. Thus the command ?'2' would cause the contents of box 2 to be displayed.

2.9.2.3 ?// DISPLAY ALL BOXES

The contents of all the users boxes are displayed on the terminal.

2.9.2.4 ?'state'

If state is one of the words DATE, TIME, SIZE, COLM, CODE, EDIT, or STOP (refer section 2.6.2) then the parameter requested is determined and its value printed.

3. ASCII DOCUMENTATION

3.1 The Basis

Documents which are prepared in ASCII mode may be edited or altered using the same commands as used for editing BCD mode documents. The main difference is that in ASCII mode upper-case letters in text strings are no longer mapped onto lower-case letters. Hence s/THE/ searches for a different text string to s/the/. Pointer movement, searching, etc. otherwise occurs as for BCD mode. The x. command (section 1.5.2) no longer causes record truncation however - it is used for another purpose.

ASCII mode documents are mainly used for preparing reports, letters or documentation (such as this TED manual). In order to aid such documentation various reserved words or symbols may be included in the original document. If this document is printed out via a normal P command then these words and symbols are printed as normal text. However the ' modifier to a P command causes them to trigger various actions which modify the printed output. These actions may split the output into pages (i.e. cause pagination), alter the width of output lines and hence the total number of output lines in the document, automatically include specified page headings and footings, arrange for all output lines to end at the same column position, automatically format paragraphs, and so forth. The above actions are only performed on the printed lines - the users RA document remains unaffected except for possible pointer movement. When the special symbols and control words are activated to cause formatting they are not themselves printed on the output document (except in one special case). Preceding the print request by a # command (section 2.1.5) causes the output to be stored as a document rather than being printed.

3.2 Control Words

Various control words may be added to the text of a source document in order to control the way output lines are formatted. These words do not have special significance unless they appear left-adjusted in columns 1 - 4 of a source record. Any other text in columns 4 - 160 of these records may be ignored on output. In each of these special records column 1 contains an asterisk and the remaining 2 - 3 characters are upper-case letters. The control words, when active, are not printed in the output text. They are activated by the ' modifier to the P command (section 3.4).

3.2.1 *BOP BEGIN PARAGRAPH JUSTIFICATION

Any text in the source document which lies between a *BOP record and a matching *END record is called formatted text. When this formatted text is output via the P command with a ' modifier the output is modified according to the following criteria.

(a) The pointer is immediately moved to column 1 of the current record. Hence no windowing (section 1.5) of lines occurs.

(b) The *BOP and *END records are not printed.

(c) The column position of the first non-blank character in the source record following *BOP defines the indentation of the first line of each new paragraph in the text, i.e. if it appears in column 15 then the start of each new paragraph of the formatted text is in column 15 of the output line. If the first record is blank column 1 is taken as the default.

(d) The column position of the first non-blank character on the second source record following the *BOP defines the left-hand margin of the output formatted text.

(e) Paragraphs in formatted text are separated by a blank record in the source text

(f) The output formatted text will not be allowed to extend past the print position x (as set by a prior x. command). If x is not set, is set less than 30, or is set greater than 160, then line width is set to a default of 30 columns (including any margin). Any words which one might expect to extend onto or over column x would be re-formatted on the next line. Thus record boundaries in the source text are ignored when the records are output and there need be no correspondence between the number of lines output and the number of records used to create this output. The output lines are word justified - words will not be split into two parts unless they are more than 15 characters long.

(g) Source words are separated by one or more blanks. When the formatted text is printed the blanks are squeezed out so that only one blank appears between successive words on a line. Since source record boundaries are ignored and blanks are squeezed it is common for words on separate source records to be output on the same output line. The user must take special precautions if he wishes to force a new line in the output.
Three exceptions to the blank squeezing are
(1) A blank source record is output without change.
(2) If two or more blanks occur after a full-stop then blanks are squeezed so that two blanks are left between the

full-stop and the next word.

(3) If the ↑ modifier is used as well as the ' modifier to the print command, P, then lines are character justified, i.e. after the output line is blank squeezed extra blanks are inserted between words in order to ensure that that the last column of each output line appears in the print position x defined by the x, command.

When setting up a region of formatted text the user may wish to alter the indentation and margin of areas of text within a formatted text section. This may be done by enclosing the inner area by another set of *BOP and *END records, i.e. *BOP - *END areas can be nested. Hence on encountering the first *BOP ensuing text is printed in one format; on encountering the second *BOP ensuing text is printed in the second defined format until another *BOP or *END is met; a *END record will result in ensuing text then being printed in the first format. A following *END will cause any text up to the next *BOP to be unformatted. Unformatted text is unaffected by the x, or 'P commands - output records are the same as source records.

3.2.2 *STP STOP

Stop printing the output until a (LF) is received. The user may then type in his own line in a form letter or change sheets of paper in the output device if single sheets are used.

When the *STP is encountered, CR-LF then colon are typed to inform the user. The colon may be eliminated by a prior DLE : (or CTRL/P :) sequence - refer DAD manual Chapter 22.3.4 . The pause is counted as one line of output for the purposes of line counts and pagination.

3.2.3 *EOF END-OF-FORM

The *EOF when encountered (in ASCII mode) causes a page throw, i.e. sufficient line-feeds are typed to fill the remainder of the page with blank lines. Section 3.4 discusses how to specify the number of lines per page.

3.2.4 *HD HEADER

Any source records between a *HD record and the next *END record are are not printed if encountered whilst outputting part of a document. Instead the records are stored in consecutive TED boxes, starting from box 1001. If paginating then each time a new page is started the contents of boxes 1001 up to the next empty box are printed at the head of the page. Any occurrence of the character ` within a header is printed as a page number (section 3.3). The *HD

to *END source records may appear anywhere in the source document except in areas of formatted text. If a second header is encountered in a document it replaces the first but does not reset the page count.

At the start of each P command with a ' modifier, boxes 1001 to 1010 are emptied. This ensures that no headers are printed until a new header is encountered.

3.2.5 *FT FOOTER

Text between a *FT record and the next *END record is not printed but is stored in boxes starting from box 2001. This text is printed at the bottom of each page. It is otherwise handled in a similar way to headers.

3.2.5 *END END

This record is used to denote the end of the current header, footer or *BOP field. A *END card is needed for each of these.

3.3 Special Control Symbols

3.3.1 ~ TILDE

Tilde when encountered in formatted text forces a new line to be printed. When formatted text is output using the P command with ' modifier, source record boundaries are usually ignored. However the user can force a new line with a tilde in the source document; if the tilde appears in column 1 of the source record then a new output line will be started from column 1 - if the tilde appears elsewhere in the source record then the new line will start at the left margin of the output text.

3.3.2 ` GRAVE

The grave accent is used in three separate ways depending in which region of source text it appears.

(a) In unformatted text - it is printed as a grave accent with no other effects.

(b) Within a header or footer - it acts as a page count when the print command causes pagination (section 3.4). i.e. it is printed as a number which is incremented every time a new page is started. It may occur more than once in headers or footers but always has the same numeric value on any one page. Redefining the header or footer does not reset the counter - it is reset only by the command causing pagination.

(c) Within formatted text - the grave symbol is treated like a normal non-blank character but is printed as a blank. This allows the user to insert spaces in the output lines without worrying about blank squeezing. It is also useful in place of symbols which will later be inserted by hand, such as Greek letters. Some care is needed as long words may be generated if one uses the grave instead of a space.

3.4 Print Commands

The print command, along with its various modifiers (', ↑ and x.), has special significance for ASCII mode documents. It allows the user great flexibility in outputting documents. In the following commands m, n, and x are integers.

3.4.1 nP NORMAL PRINT

This command, without modifiers is the normal print command. It causes n records of the source document to be output. The output lines are the same as the source records except that windowing may occur in the normal way. The x. command is ignored, however, and no truncation of lines occurs. Control words and special symbols are printed as stored without causing any special actions.

3.4.2 'nP FORMATTED PRINT

This command causes n lines of output to be printed. Reserved control words and special symbols in the source records are activated, and any formatted text is output as described in section 3.2. Note that n is the number of output lines - the number of source records needed to produce this may be greater than, less than, or equal to n. Unformatted areas of source records (excluding special control word records) are printed exactly as they occur - no windowing or truncation of lines occurs.

The command is useful for outputting a document onto separate sheets of paper. After each n lines are printed a colon is typed and output halts. A new sheet can then be inserted in the output device and a further n lines output by typing line-feed. This procedure may be continued until the entire document is output. Anything else typed after a colon will be used as a new instruction to TED.

If an *EOF source record is encountered whilst any page is being printed then the remainder of the page is filled with blank lines. Hence *EOF may be used a page throw. Headers, if present, are included in the count of n

lines and are printed on each page. If the number of lines of header exceeds n then the entire document is output without pagination. If footers are included then each page size is increased by the number of records in the footer.

3.4.3 ↑'nP

This command causes n lines of output to be printed with control words and special symbols activated as in the 'nP command. Any lines of formatted text are character justified as well as word justified, i.e. extra blanks are inserted between some output words in order to make each output line end at column x, where x is set by the x. command. The default and minimum value of x is 30.

3.4.4 ↑m'nP

This command is the most common method of outputting formatted ASCII documents on a teletype. The entire source document from the pointer on is output with control words and special symbols activated (as described in section 3.3), word and character jusification (as above), and pagination.

Page images consist of m+n lines of print. The first n lines of each page contain the header then lines of the document; the following m lines consist of the footer and blank lines. The command is useful for output onto fanfold paper as no user intervention is needed during output. *EOF again causes a page throw.

If m is set to one (or if the no. of lines of footer plus 2 is greater than m) the command behaves as ↑'xP (where x=m+n) with pauses at the end of each page of output.

3.5 Document Formation

3.5.1 #/name/.....P

The cross-hatch command can be used with any of the above print commands in order to save the formatted or unformatted output as a 3600 document. The output lines are not printed when the print command is preceded by the # command. The new document is exactly the same as one would expect from the output due to the print command with the following exceptions when the ↑'nP (or ↑1'nP) commands are used.

(a) Reserved words and special symbols are not omitted from the document. Hence, in this case, the newly formatted

document can later be reformatted. It can also be used as a reference to show where each *BOP, etc. occurs.

(b) The entire source document, from the pointer position on is used to create the new document - there is no requirement for the user to type (LF) after each page image is formed.