CP/M™

TEX

TEXT FORMATTER

USER'S GUIDE

DIGITAL RESEARCH

```
                        ;TEX PATCH #1: 27 JUN 78
                        ;THIS PATCH CORRECTS AN ERROR WHICH OCCURS
                        ;DURING ;".NA" MODE WHERE A WORD IS
                        ;DUPLICATED WHEN THE ;OUTPUT LINE IS EQUAL
                        ;TO THE LINE LENGTH.
                        ;PATCH FOLLOWS:

1D43 =              PATCH      EQU   1D43H
0DAB =              L606       EQU   0DABH
1DE5 =              TLINE      EQU   1DE5H
1DE6 =              WLINE      EQU   1DE6H
1DE8 =              ELINE      EQU   1DE8H
1DEA =              NLINE      EQU   1DEAH

0DAB                           ORG   L606

0DAB C3431D                    JMP   PATCH

                    RETPAT:

1D43                           ORG   PATCH
1D43 32E51D                    STA   TLINE
1D46 3AE61D                    LDA   WLINE
1D49 21E81D                    LXI   H,ELINE
1D4C BE                        CMP   M
1D4D F2AE0D                    JP    RETPAT
1D50 3AEA1D                    LDA   NLINE
1D53 32E61D                    STA   WLINE
1D56 C3AE0D                    JMP   RETPAT
1D59                           END

                        ;TEX PATCH #2, 28 JUN 78
                        ;THIS PATCH CORRECTS AN ERROR WHICH OCCURS
                        ;WHEN THE $F FORM FEED OPTION IS USED.  IN
                        ;PARTICULAR, THIS PATCH CAUSES A FORM FEED
                        ;ON THE FIRST PAGE, WHICH WAS NOT
                        ;PREVIOUSLY DONE.

1D59 =              PATCH           EQU   1D59H
0980 =              L445            EQU   0980H
1F11 =              SAV$LINELENGTH  EQU   1F11H
1F0C =              FORM$FEED        EQU   1F0CH
0672 =              PUTDEST          EQU   0672H
000C =              FF               EQU   0CH
0980                               ORG   L445
0980 C3591D                        JMP   PATCH

                    RETPAT:

1D59                               ORG   PATCH
1D59 32111F                        STA   SAV$LINELENGTH
```

```
1D5C 3A0C1F                     LDA   FORM$FEED
1D5F 17                         RAL
1D60 0E0C                       MVI   C,FF
1D62 DC7206                     CC    PUTDEST
1D65 C38309                     JMP   RETPAT
1D68                            END


A>DDT TEX.COM
DDT VERS 1.4
NEXT PC
1D80 0100

-L0DAB
     0DAB STA  1DE5
     0DAE LDA  1DE8

-L0980
     0980 STA  1F11

-ITEXPAT1.HEX
-R
NEXT PC
1D80 0100
-ITEXPAT2.HEX
-R

NEXT PC
1D80 0100

-L0DAB
     0DAB JMP  1D43
     0DAE LDA  1DE8

-L0980
     0980 JMP  1D59
     0983 LXI  H,1F10

-L1D43,1D68
     1D43 STA  1DE5
     1D46 LDA  1DE6
     1D49 LXI  H,1DE8
     1D4C CMP  M
     1D4D JP   0DAE
     1D50 LDA  1DEA
     1D53 STA  1DE6
     1D56 JMP  0DAE
     1D59 STA  1F11
     1D5C LDA  1F0C
     1D5F RAL
     1D60 MVI  C,0C
     1D62 CC   0672
```

```
      1D65 JMP   0983
      1D68 LDAX D
      1D69 LDAX D

-G0

A>SAVE 29 TEXP.COM
A>
```

COPYRIGHT

TRADEMARK

The names CP/M, SID, MAC, TEX, and Digital Research are trademarks of Digital Research.

DISCLAIMER

The "TEX User's Guide" was prepared using the TEX Text Formatter.

FORWARD

This manual is intended as a guide for users of a microcomputer system who wish to use the TEX Text Formatter program running under the CP/M Operating System. Because all users do not need to familiarize themselves with the entire contents of the Digital Research documentation for CP/M in order to use the TEX program, this manual should enable the casual user to operate the system with a minimum amount of study and training.

Most of the information in this manual is contained in more detail in the manuals, "CP/M Features and Facilities" and "ED: a Context Editor for the CP/M Disk System" which are available from Digital Research.

Although this manual does not assume a prior knowledge of CP/M on the part of the user, it does assume a familiarity with the user's particular hardware system.

The "TEX User's Guide" is divided into sections in the following manner. Section 1 gives an overview of the CP/M programming system, emphasizing those portions which are needed in the operation of TEX. The explanation of ED, the Context Editor for CP/M, is of particular importance because the TEX input source files are created with ED.

Sections 2, 3, and 4 describe the TEX program in detail. Section 2 introduces TEX to the user; this section presents various features of the program and explains how to use it. Section 3 describes each of the TEX commands in detail, includes examples, and highlights functions which warrant special attention. Section 4 describes the formatted output options which TEX allows the user to choose at run time.

The Appendices include (A) a quick reference table of TEX commands, (B) a summary of the TEX run-time parameters, (C) examples of the paging commands, and (D) a description of possible error messages.

TABLE OF CONTENTS

1 An INTRODUCTION TO CP/M

This section is written primarily for the TEX user who is not familiar with the CP/M Operating System. The user who already has a working knowledge of CP/M may wish to skip Section 1 and proceed to Section 2.

The intent of Section 1 is to describe those portions of the CP/M Operating System which are necessary for the operation of the TEX Text Formatter. The information contained in the following pages may be found in greater detail in the specific CP/M user's manuals which are referenced throughout the section.

1.1 A Typical Microcomputer System

The heart of an information processing system is the Central Processing Unit (CPU). In microcomputer systems, the CPU is often a single integrated circuit such as an Intel 8080 or a Zilog Z80.

In order to process information with a computer's input, output, and storage devices are also necessary. In the typical system, the input device is a keyboard and output devices include a CRT (cathode ray tube) and a printer. Other I/O devices might include a cassette tape recorder, paper tape reader and punch, and other special purpose devices. Storage is of two types: volatile (the semiconductor memory connected to the CPU within the computer) and non-volatile (disks, magnetic or paper tape, and other storage media).

Figure 1 illustrates the devices which comprise what will be referred to, in this manual, as the typical system. Usually, the CPU and the semiconductor memory (also called the "memory buffer," the "buffer," or simply, "memory") are internal to the computer unit. The disk drives, the keyboard, the CRT display unit, and the printer are external devices.

In our typical system, we will assume that the CPU and memory are enclosed within the main computer unit, although hardware configurations do vary widely. The disks, CRT, keyboard, and printer are external devices, connected to the computer's "main box" by cables.

In order to use the computer as a general-purpose computer, a program called an "operating system" is needed. This is a program which provides the necessary interactions among the various system components. The operating system

must accept commands from the keyboard and interpret them.
It must read from and write to the disks, as well as keep
track of where information is located on the disks.   The
operating system sends characters to the CRT for display or
to the printer for "hard-copy" output.

Until   the   operating   system   is   loaded   into   the
computer's memory and the CPU is set to access the program,
the computer cannot function.   Section 1 will explain how
to use the CP/M Operating System in order to provide the
user with a workable computer system for running the TEX
program and other programs.

Figure 1

2
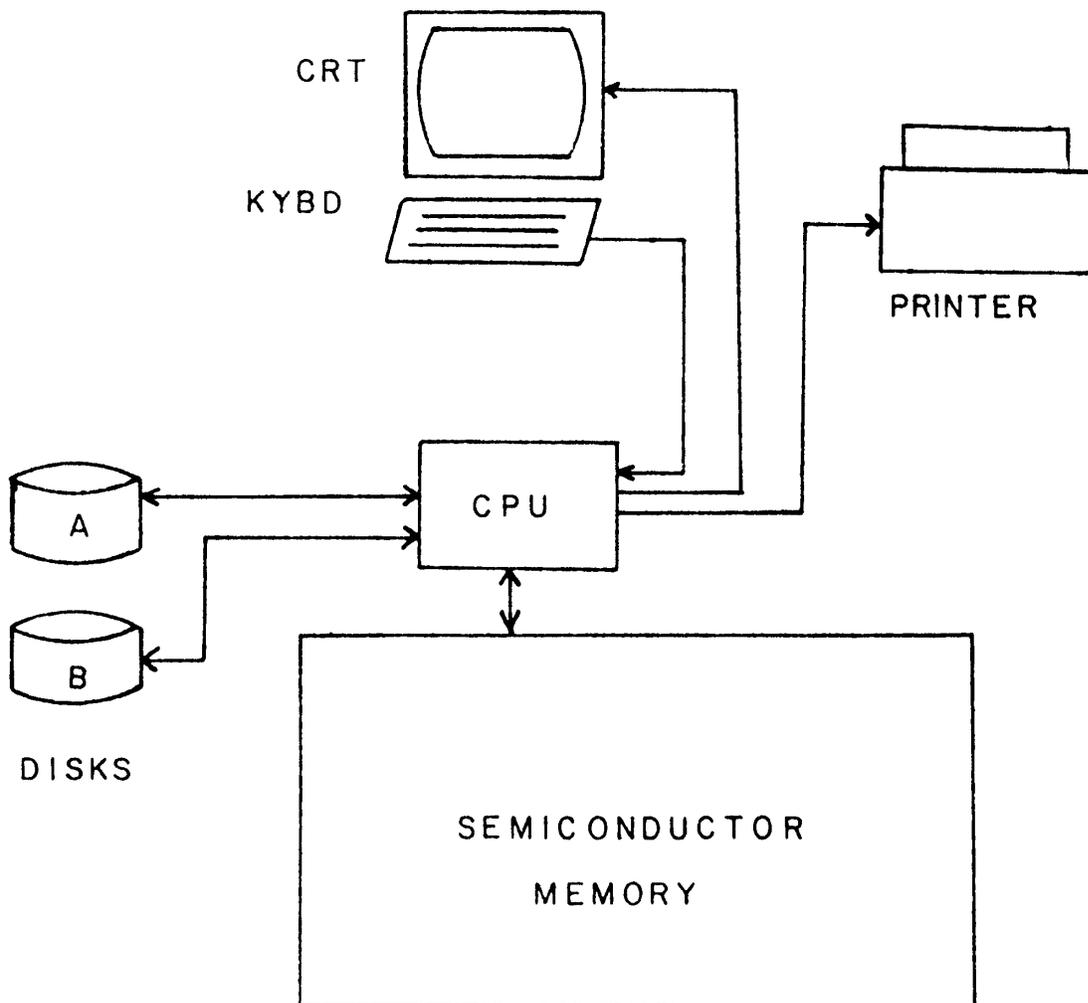
1.2 Overview of the CP/M Operating System

CP/M, a "Control Program for Microcomputers," is an operating system consisting of several parts. These parts include the Console Command Processor (CCP), the Basic Disk Operating System (BDOS), and the Basic Input/Output System (BIOS). The CCP interprets console commands and accesses one of the online disk drives: A, B, C, or D. The DBOS and BIOS control the interaction of the computer with the disk drive units and with the other I/O devices, respectively.

<div align="center">FILES</div>

A file is an area of a disk, which is reserved for data and given a name. This data is a stream of characters, which varies depending on the nature of the file. A file may contain words and sentences, as in a TEX file or PRN file; or it may contain a machine language program, a COM file for instance, which can be read only by the computer. If the user types the DIR built-in command, CP/M will display the names of all the files residing on the disk in the logged-in disk drive. A file may be as large as 240k bytes (use STAT to determine file size), or it may be empty. Generally, for the TEX user, a file will contain either the text source statements (a TEX file for example), or formatted output statements (usually a PRN file). A file is designated by a filename of up to eight characters and a filetype of not more than three characters. The filename and filetype are separated by a period. The format of a filename is as follows:

<div align="center">xxxxxxxx.yyy</div>

would designate a file of filename xxxxxxxx and filetype yyy. A file with no specified filetype is equivalent to one with a filetype of three blanks. Some examples of valid file designations are:

<div align="center">
REPORT.ASM
TRUNK.COM
FLIGHT.PRN
FILE
</div>

The user should note the following conventions in naming files:

(a) Lower-case in filenames.

(b) The following special characters may not be used within a file designation:

<div align="center">. [ ] ; : = ?</div>

In searching for files, the symbol ? means "any character is a match." For example, if the following two files exist on the disk:

REPORT1.BAS
REPORT2.BAS

a search for REPORT?.BAS in the disk directory (refer to the DIR command in Section 1.3.) would find and list both files.

The use of the * symbol indicates "all." In the example above, a search for *.BAS would retrieve both the REPORT1.BAS and REPORT2.BAS files, as well as all other files with a filetype of BAS. A search for *.* would retrieve all files on the disk.

DISK ACCESS

In order to indicate which disk is currently accessed ("logged in"), CP/M prompts the user with an A>, B>, C>, or D> to indicate the current disk drive. If the A disk is logged in and the user wishes to access the B disk, the following command logs out the A disk and logs in the B disk:

B:<cr>

(Note that the symbol <cr> indicates a carriage return.) To return to the A drive, the following command would be typed:

A:<cr>

It is important to note that the user need not be logged in to a particular disk drive in order to access or refer to a file on that drive. To access a file not on the logged-in disk, simply preceed the filename with the appropriate disk drive name (A, B, C, or D), followed by a colon W.

For example, if the A disk is logged in, but the user wishes to view the contents of a TEX source file on disk D, then the following command is given:

A>TYPE D:FILEX.TEX

where A> is the prompt message indicating the logged-in disk, TYPE is the CP/M built-in command (see Section 1.3.) which types the contents of a file onto the console device, and D: is the disk designation of the TEX source file, FILEX.TEX.

CONTROL CHARACTERS

    The following keyboard characters have special functions under the operation of CP/M:

DEL(delete) or Deletes, echoes last character.
RUB(rubout)

Any key but ^S   Aborts the typing of a file onto the console or printer.

^C   Reloads the CP/M operating System.

^I   Indicates a tab position. If a file is created using the ^I character (or TAB key) for spacing, each ^I character can be expanded into n spaces by using the PIP command with a Tn parameter (see Section 1.5.).

^L   Signifies the character pair <cr> <lf> (carriage return, line feed) in strings of characters in the edit environment. (See the ED User's Guide.)

^P   Copies all subsequent console output to the printer, as well as to the console. Typing ^P turns this feature on; typing ^P again, turns it off.

^R   Retypes the current line as it appears after deletions and corrections.

^S   Temporarily stops typing output onto the console or printer. Depressing any key (including ^S) restarts typing.

^U   Deletes an entire line.

^X   (Same as ^U.)

^Z   Ends an input string within the PIP and ED programs.

Table 1-1 Control Characters

NOTE: In the above list, the symbol ^ signifies a control character. The symbol ^U, pronounced "control u," is typed by holding the "ctrl" key down while typing a "u."

1.3 CP/M Commands

There are two general types of commands which can be
executed under CP/M. The two types are referred to as
"built-in commands" and "transient commands." (See Figure
2.)

The built-in commands are part of the CCP. These
functions are available whenever the CP/M system is
operating. The only software needed to use the built-in
commands, therefore, is a disk containing the CP/M
Operating System configured for the user's particular
microcomputer hardware.

The transient commands are individual programs which
reside on disk and are loaded from disk into memory when
the command is typed at the console. Under the CP/M
operating System, a large part of memory is left open for
transient programs. Several transient programs are
included as part of the CP/M system. These are listed in
the section on standard transient commands. Other
transient programs are those which are supplied by Digital
Research or independent software suppliers. These programs
may include sorting commands, text processing functions,
and an infinite variety of other possibilities. Transient
programs must have a filetype of COM (command) in order to
execute under CP/M.

Figure 2

FIGURE 2.

The diagram shows a hierarchical tree:

- CP/M → CCP
  - BUILT-IN COMMANDS
    - ERA (erase)
    - DIR (directory)
    - REN (rename)
    - SAVE
    - TYPE
  - TRANSIENT COMMANDS
    - STANDARD TRANSIENT
      - STAT (status)
      - PIP
      - EDIT
      - ASM
      - SYSGEN
      - LOAD
      - DUMP
      - DDT
    - USER'S TRANSIENTS
      - FORMAT
      - SORT
      - TEX
      - OTHERS
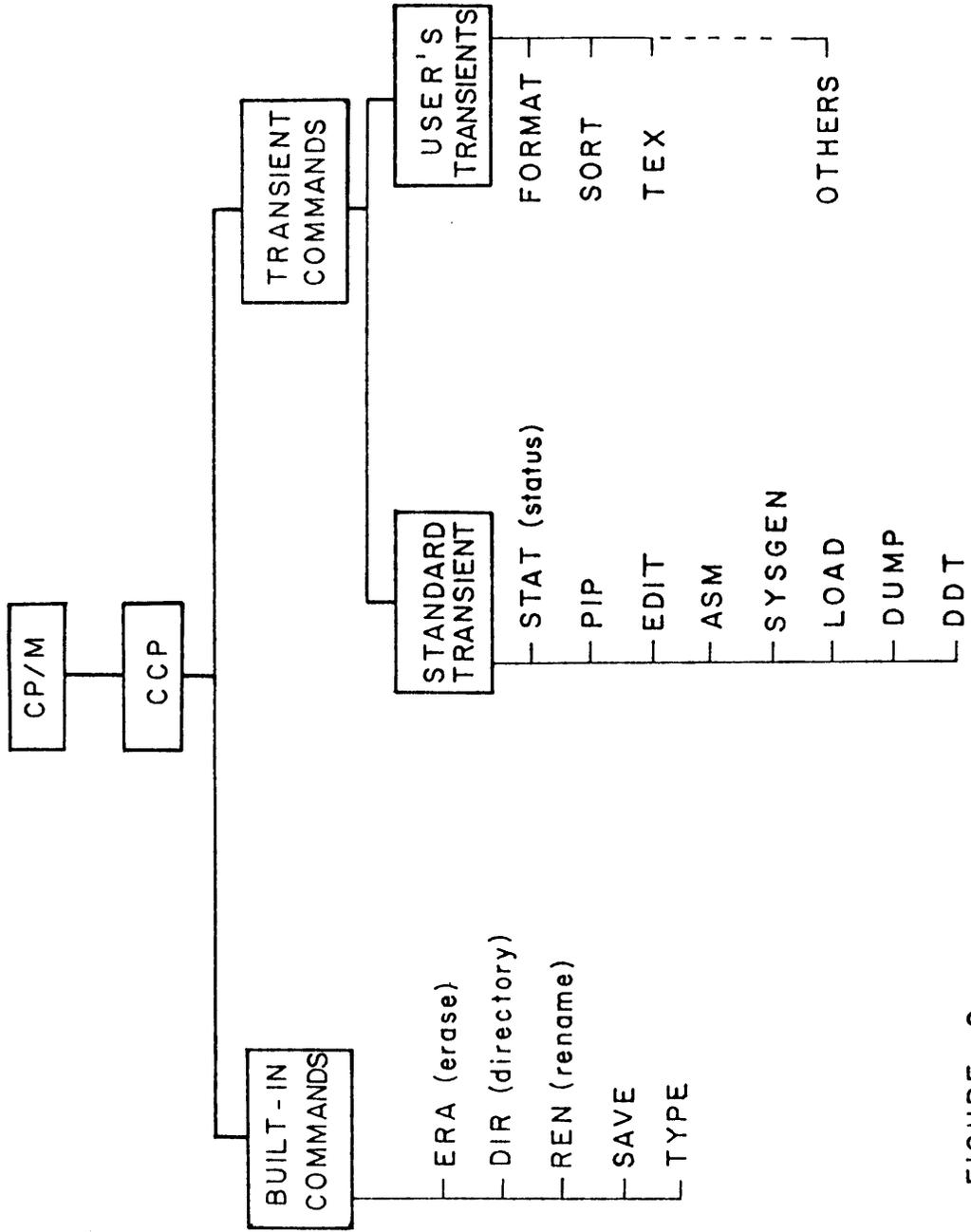
**BUILT-IN COMMANDS**

The following built-in commands are part of the CCP:

                        DIR (*directory*)
                        REN (*rename*)
                        SAVE
                        TYPE

```
ERA
```

ERASE A FILE

The ERA command erases files from the logged-in disk.
Examples are as follows (assume that the A disk is logged
in):

ERA RPT.BAS<cr>   Erases the file with filename RPT
                         and filetype BAS from the A disk.

ERA *.BAS<cr>   Erases   all   files   with   the
                         filetype BAS from the A disk.

ERA *.*<cr>   Erases all files from the A disk.

CAUTION: When using one of the above formats to execute
          the ERA command, the user should make certain
          that the file to be erased is on the logged-in
          disk; otherwise a file of the same name, residing
          on   the   currently   logged-in   disk,   may
          inadvertently be erased.   A file which is not on
          the logged-in disk may be erased by specifying
          the disk designation where the file resides.   For
          example:

ERA D:*.PRN<cr>   Erases all PRN files from disk D.

```
DIR
```

DIRECTORY LISTING

The DIR command lists the requested files on the
console.   Some examples (again assume that the A disk is
logged in):

DIR <cr>   Lists all files on the A disk.

DIR *.COM<cr>   Lists all files of filetype COM.

DIR X.Y<cr>   Lists   only   the   file   with   the
                         filename X and filetype Y, if
                         such a file exists on disk A.

        DIR C:<cr>   Lists all files on the C disk.

┌─────────┐
│ REN   │
└─────────┘

RENAME A FILE

    The REN command changes the filename and/or filetype
of a file. Examples:

    REN PROG1.OUT=PROG1.PRN<cr>   File PROG1.PRN is renamed
                                  as file PROG1.OUT.

    REN XFILE.IN=YFILE.TEX<cr>   File YFILE.TEX is changed
                                  to file XFILE.IN.

    REN C:X.Y=C:U.V<cr>   File U.V, on drive C, is
                                  renamed   as   file   X.Y.
                                  This is an example of
                                  renaming a file not on
                                  the logged-in disk.

    The REN command does not affect the contents of the
file in any way. The file is not moved or copied; only the
filename and/or the filetype is changed.

┌─────────┐
│ SAVE   │
└─────────┘

SAVE A FILE IN MEMORY

    The SAVE command is used to copy the contents of
memory to a disk file.   This occurs under certain
circumstances, such as in making alterations to the CP/M
system.   The   SAVE   command   is   included   here   for
completeness, but will not be used here.   See the "CP/M
Features and Facilities" manual for more information on
this command.

┌─────────┐
│ TYPE   │
└─────────┘

TYPE A FILE

    The TYPE command displays the contents of an ASCII
source file on the console. Examples:

                TYPE SAMPLE.ASM<cr>
                TYPE THESIS.TXT<cr>
                TYPE B:THESIS.TEX<cr>

    The TYPE command provides an easy method for viewing
the contents of a TEX or PRN file on disk.   Once the
display of the file has begun, the console output can be
stopped temporarily by pressing ctrl-S (see Section 1.2

CONTROL CHARACTERS); it can be restarted by pressing any key on the keyboard, such as another ctrl-S.

STANDARD TRANSIENT COMMANDS

Transient commands are programs which can be loaded by the user into the Transient Program Area (TPA) of CP/M. These commands are files which reside on the disk with a filetype of COM; they are loaded into memory from the disk each time they are executed. In order to use the transient commands, either the disk which is logged in must contain a copy of the transient command program, or the user must preceed the command with the prefix of the drive (A:, B:, C:, or D:) which contains the program. Standard transient commands included with the CP/M Operating System are:

STAT (Status)	Lists the amount of storage available on the logged-in disk (see Section 1.4.).
ASM (Assemble)	Loads and executes the 8080 assembler.
LOAD	Reads a hexadecimal file and produces a binary (object) file
PIP (Peripheral Interchange Program)	Copies or concatenates files (see Section 1.5.).
ED (Edit)	Used to create and edit TEX source files (Section 1.6.).
SYSGEN (System Generation)	Generates a new CP/M system disk.
SUBMIT	Executes a series of CP/M commands.
DUMP	Writes the contents of the specified file on the console in hexadecimal form. (Not used in this manual.)

1.4 The STAT Command

STAT is one of the standard transient programs supplied with the CP/M Operating System. Since STAT is transient, a copy of the file STAT.COM must be either on the logged-in disk, or on an accessible disk, in order to use the program.

The primary function of STAT is to provide the user with the status of all disks currently in use. STAT indicates whether the disks are in a read-write or read-only status, and gives the amount of usable space remaining on the disks. STAT can also be used to display the amount of disk space used by particular files.

Secondary functions of the STAT program are (a) to define disks as read-only and (b) to change the logical device assignments of the physical input/output devices. These secondary functions, however, are beyond the scope of this manual, and will not be discussed here. Refer to the "CP/M Features and Facilities" manual, for more information on the uses of STAT.

Possible forms of the STAT command are as follows:

> (a)   STAT<cr>
>
> (b)   STAT xxxxxxxx.yyy<cr>
>
> (c)   STAT *.*<cr>

The system will respond to form (a) with the read/write status of each disk and the number of kilobytes of space remaining on each disk.

The response to form (b) includes, for the file xxxxxxxx.yyy, the number of 128-byte records, the number of kilobytes, and the number of 128-byte record extensions used by the file.

The response to form (c) includes the same information as the second form, but for all files on the logged-in disk.

As with the built-in commands, any valid disk designation can be specified for a file accessed by the STAT program. For example:

> STAT B:*.*<cr>
> STAT C:*.PRN<cr>

1.5 The PIP Command

PIP, the Peripheral Interchange Program, is one of the transient programs which is a subsystem of CP/M. This program concatenates (merges several files), abstracts, or copies files on a single disk or from one disk to another. The PIP program also copies files onto the printer or the console from disk.

To enter the PIP program, simply type "PIP" following the usual CP/M "A>" prompt. While in the PIP environment,

the system prompts the user with a "" symbol. To exit from the PIP environment and return to CP/M, type a carriage return.

COPYING FILES

To make copies of a file on a single disk, statements of the form

PIP XX1.ASM=XX.ASM <cr>

and

PIP REPORT.TEX=SENDFILE.TEX,OLDFILE.TEX <cr>

can be used. The first example creates a file named XXl.ASM and copies the existing file XX.ASM into the new one. The second example creates a file named REPORT.TEX and concatenates and copies into the new file both of the existing files, SENDFILE.TEX and OLDFILE.TEX.

CAUTION: If a file already exists with the name of the new file to be created under PIP, the old file will be erased by the PIP operation.

To copy files from one disk to another, the same general form is used, except that the filenames are prefixed with the disk designator A:, B:, C:, or D:. For example,

PIP B:THESIS.TXT=A:THESIS.TXT <cr>

copies the file named THESIS.TXT from the A disk onto the B disk. The statement

PIP A:XXX.YYY=B:OLD.ASM <cr>

creates a file named XXX.YYY on the A disk, and copies into this newly created file the B-disk file OLD.ASM. In certain instances shortened forms of these commands may be used. The user may consult the "CP/M Features and Facilities" manual to learn the short forms. It is, however, recommended that the forms given here be used until the user becomes thoroughly familiar with the CP/M system.

NOTE: The copying operation under PIP may be aborted by depressing any key on the keyboard.

PRINTING FILES

To print a "hard copy" (copy on paper) of a file, type the command LST:= followed by the filename. Examples:

                        PIP LST:= XXX.TEX <cr>
                        PIP LST:= REPORT.PRN <cr>

NOTE: Only files containing printable characters (for
            instance, filetypes TEX or PRN) can be freely
            listed on the printer.

     If there are tab functions within the file, placing
the notation [T81 in a PIP statement prints the file with
tab settings every 8 spaces. For example:

                    PIP LST:= REPORT.PRN[T81 <cr>

would list the REPORT.PRN file on the printer with tabs
every eight spaces.

     There is a shorter form of the PIP command which
eliminates the necessity of continually reentering the PIP
environment when several PIP commands are to be executed.
Refer to the "CP/M Features and Facilities" manual for this
format.

     Other options available with PIP, using parameters,
include the following: the ability to specify a variable
tab setting; the ability to delete all characters past a
certain column; a capability for echoing the transferred
data to the console; capabilities for deleting forms feeds
and inserting line numbers; translation of alphabetic
characters totally to upper-case or to lower-case. When
the user becomes sufficiently familiar with the CP/M
system, he may wish to consult the PIP section of the "CP/M
Features and Facilities" manual for instruction on the use
of these more sophisticated PIP options.

     1.6   The ED Command

     ED is another subsystem of the CP/M Operating System.
It performs file creation and editing functions. To enter
the edit environment, type ED and the filename and filetype
of the file to be created. For example,

                    ED INSTRUC.TEX <cr>

opens the file INSTRUC.TEX for editing, or, if no such file
exists, creates a new one by that name. After the ED
command is given, the CP/M system loads the edit program
from disk into memory and passes control to the program.
The system is now in the "edit mode" or "edit environment."

LOADING THE FILE FOR EDITING

Before starting an edit session, the user should type the STAT command (see Section 1.4.) to ensure that there is enough space on the work disk to hold the edited file. For example, if the file THESIS.TEX is to be edited, the user can determine the size of the existing file by typing

STAT THESIS.TEX <cr>

If a subsequent STAT command does not indicate at least as much space remaining on the disk as the existing file occupies, then the user should take some action to acquire more space before editing. For instance, the user may erase unnecessary files (BAK files, PRN files) from the current disk, or he may use PIP to move the THESIS.TEX file to another disk which has more free space.

If no previous version of the file exists on the currently logged-in disk, the file is considered to be a new one and is automatically created. The "insert mode" must be entered, using the I command, in order to put information into this new file. Typing ^Z causes the editor to leave the insert mode and return to the edit mode. This input information is stored in memory, and the altered version is copied back onto disk for retention at the end of the edit session.

Figure 3 illustrates the steps which generally occur during an edit session.
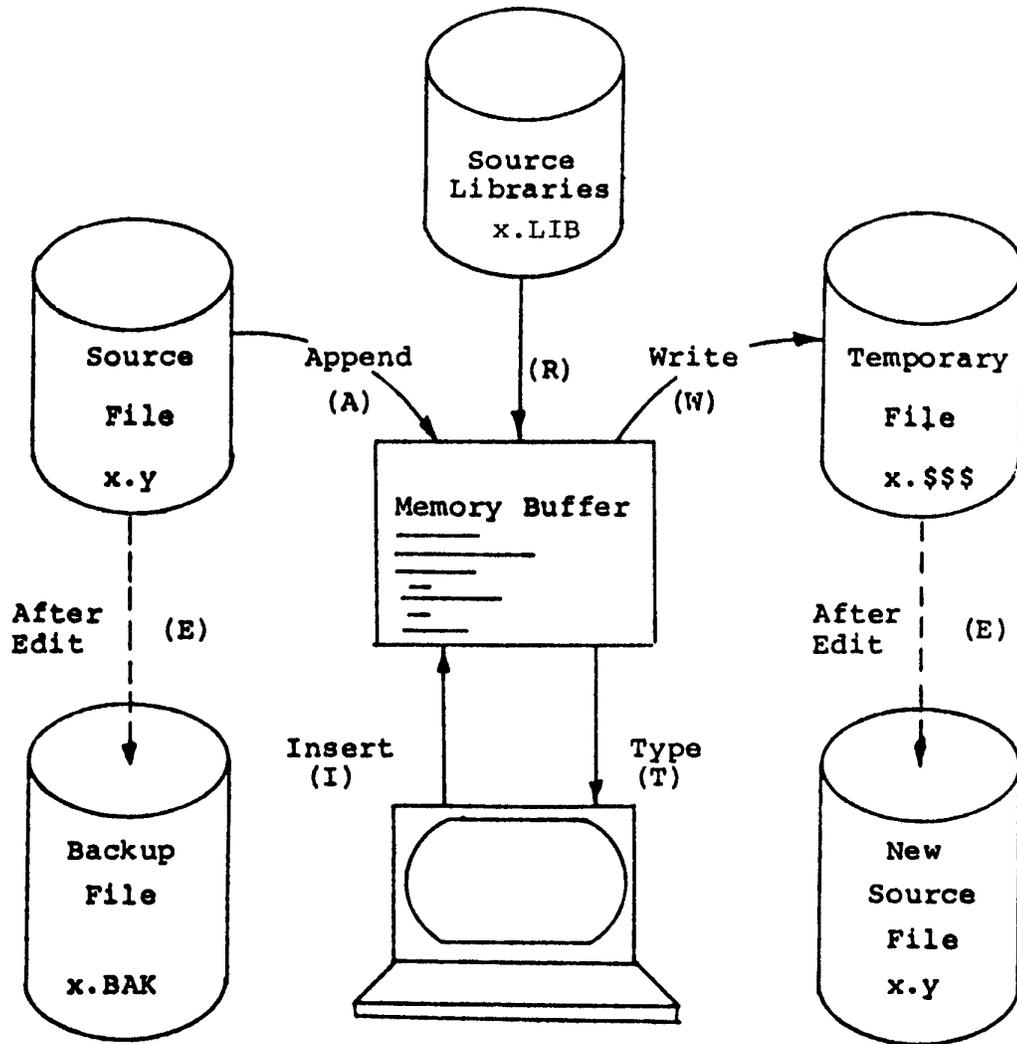
**Figure 3 Illustration showing overall
        operation of the ED Context Editor.**

    The A (append) command is used to copy an existing
file from disk to memory. Typing nA (where n is a positive
integer less than 65535) brings the first n lines of the
file from disk into memory. Typing the symbol "#" instead
of a number brings the whole file into memory, if it will
fit. If there is not enough memory space available for the
entire file, #A brings in as many lines as will fit into
memory.

    If all of the file does not fit into memory, the first
part can be edited; then n lines of the file can be written
back to disk to free more space in memory for the next part
of the file. This is done by typing nW, which moves the

first n lines from memory to disk; or type #W to move all
the lines in memory to disk, emptying the memory. Another
append command can then be used to bring the next part of
the file into memory. Files of virtually any length can be
"paged" through in this fashion.

     For convenience, the commands OA and OW can be used.
The OA command will fill half of the available memory of
the computer. The OW command will half empty the memory.
Further, the N command, described below, can be used to
automatically page through a file until a particular word
or word-sequence is encountered.

EDITOR CHARACTER POINTER

     In editing, the user must keep track of an imaginary
character pointer. This will take some practice, but an
understanding of the function of the character pointer is
absolutely essential in learning to use the ed it program.
(In this manual, we will designate the character pointer
with the symbol <cp>.)

     The character pointer is a device by which the editing
program keeps track of the specific place at which some
editing operation is to be performed in a file. For
example, if a certain number of characters are to be
deleted from a file, the D (delete) command is used. This
command specifies the number of characters to be deleted,
but not the location from which they are to be deleted. To
delete specific characters, the user must move the
character pointer to the desired location, and then give
the delete command. This applies to almost all the edit
commands.

     At the start of an edit session, the <cp> is always
positioned at the top of the file, before the first
character. It can be moved ahead or back a certain number
of character -positions using the C (character) command.
(Note that a character position may contain a character or
a blank.) It can be moved up or down a certain number of
lines by using the L (line) command or by issuing a "line
number:" when in the V (verify line numbers) mode. Refer
to Figure 4 for an illustration of character pointer <cp>
movement.

     To illustrate the use of the <cp>, consider the
following situation:

     We have just entered the edit mode and copied the
block of text shown below into memory. Note that the

invisible carriage return and line feed characters are at the end of each line and are included in the character count.

Initially, the <cp> is positioned at the top of the file, just before the capital I (location 1). The command 8C will move the <cp> along the line to the point between the "w" and the "e" of the word "twelve" (location 2). The next command, -2C, will move the <cp> back, between the space and the "t" of the word "twelve" (location 3). The command 2L moves the <cp> to the beginning of the line, two lines down, just before the "a" of "aid" (location 4). In this way, the <cp> can be moved up or down, ahead or back, to the appropriate location for the editing function required.

```
It is twelve o'clock.  Now is the time
  /\         /\ /\
 / 1\       /3  2\

for all good men to come to the


aid of their country.  The quick red
  /\
 / 4\

fox jumped over the lazy brown dog.
```
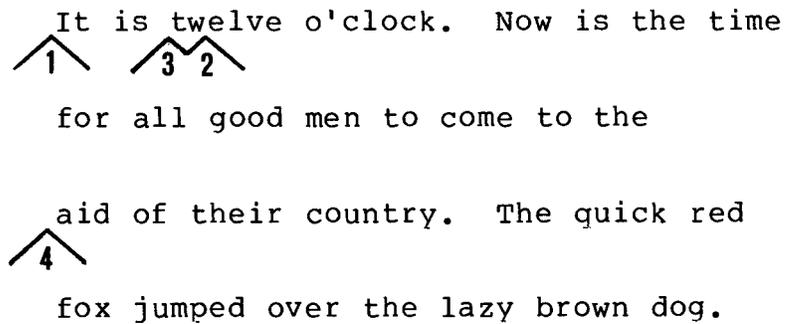
Figure 4   Illustration of text within a file, and the movement of the character pointer within the file for editing.

SUMMARY OF EDIT COMMANDS

The following is a summary of the more important commands used in the edit environment:

```
 _____
|           |
|    A      |
|_____|
```

APPEND

#A   Appends whole file into the memory buffer, if it will fit in the available memory space. If it will not, #A appends as much of the file into memory as will fit. After editing the first part of the file, the W command can be used to write part of the file from memory to disk, making more space in memory for the next part of the file.

OA   Appends lines of the file until the buffer is half full or until all of the file is in memory. This is a special form of the A command which assures that enough memory space will be available to handle editing commands. In other words, if memory is completely filled with the file data, there may not be enough work space in which to satisfactorily edit the file.

nA   Appends n lines of the file into the memory buffer.

```
 _____
|           |
|    B      |
|_____|
```

BEGINNING/BOTTOM

B   Moves the <cp> to the beginning of the memory buffer.

-B   Moves the <cD> to the bottom of the text in the memory buffer.

```
 _____
|           |
|    C      |
|_____|
```

CHARACTER

nC   Moves <cp> n characters to the right.

-nC   Moves <cp> n characters to the left.

```
┌──────────┐
│   D      │
└──────────┘
```

DELETE CHARACTER

nD   Delete the n characters after the <cp>.

-nD   Delete the n characters before the <c-o>.

```
┌──────────┐
│   E      │
└──────────┘
```

END EDIT

E   Ends the edit session, copies the edited
            version of the file from memory onto disk,
            and returns control to the CP/M operating
            System.

```
┌──────────┐
│   F      │
└──────────┘
```

FIND

nFxyz^Z   Searches that portion of the buffer after
            <cl)>   for   the   next   occurrence   of   the
            character string "xyz".   If the string is
            found n times, the <cp> is moved to a point
            immediately after the n-th occurrence of the
            string.   If the string is not found n times,
            <cp> is moved to the position immediately
            after the last occurrence of the string.
            (Note the use of ^Z to denote the end of the
            character string.)

```
┌──────────┐
│   I      │
└──────────┘
```

INSERT

I   Inserts characters after the <cp> in the
            file. The command I, on a line by itself or
            as the last item in a command line, places
            the system in the insert mode, which is
            terminated only by typing a ^Z.   While in
            the insert mode, as many lines of input as
            desired can be entered (up to the capacity
            of memory) at the point at which the
            character pointer was positioned before
            entering the insert mode.

Iabc^Z   Inserts the character string "abc" into the
            file immediately preceeding the <cp>.

```
┌─────────┐
│    K    │
└─────────┘
```

KILL LINE

nK   Kills (deletes) the next n lines, including
            the characters in the current line after the
            <cp>.

-nK   Kills (deletes) the last n lines, including
            the characters in the current line before
            the <cp>.

```
┌─────────┐
│    N    │
└─────────┘
```

NEXT

nNabc^Z   Searches the buffer after <cp> for the n-th
            occurrence of the "abc" character string.
            If the memory buffer is empty, or if the
            string is not in the buffer, text will be
            continuously appended and written in half-
            buffer increments (see the OA and OW
            commands) until either the n-th occurrence
            of the string is found or the end of the
            text file is reached. The repositioning of
            the <cp> is the same as for the F command.
            This command is a combination of the A
            (append), W (write), and F (find) commands.

```
┌─────────┐
│    L    │
└─────────┘
```

LINE

OL   Moves <cp> to beginning of current line.

nL   Moves <cp> to beginning of line, n lines
            down.

-nL   Moves <cp> to beginning of line, n lines up.

```
┌─────────┐
│    R    │
└─────────┘
```

READ

R   Reads the contents of the temporary library
            file, which was created by a previous X
            command, into the memory buffer. The
            characters are placed into the edited source
            file at the position immediately preceeding
            the character pointer <cp>.

Rf   Reads the contents of some file having a
            filename denoted by f and filetype LIB, into
            the memory buffer.

```
Q
```

QUIT

Q   Ends the edit session with no change to
            original version.   All editing changes made
            during the session are lost.   When this
            command is typed, ED responds with the
            message

                        Q-(Y/N)?

    meaning "End this edit: Yes or No?".   The user must
respond with "Y" or "N" to indicate the desired action.

```
S
```

SUBSTITUTE

nSxyz^Zabc^Z   Locates the next n occurrences of the
            string   "xyz",   and   substitutes   for   each
            occurrence of that string, the string "abc".
            (Note the use of the ^Z function to show the
            ends of the strings.)

    This command is actually a combination of the F
(find), D (delete), and I (insert) commands.

```
T
```

TYPE

OT   Types the contents of the current line up to
            <cp>.

T   Types contents of the current line, from
            <cp> to the end of the line.

nT   Types the current line, from the <cp> on,
            and the n-l lines which follow.

-nT   Types the previous n lines, and the current
            line, up to the <cp>.

```
┌─────────┐
│    V    │
└─────────┘
```

VERIFY LINE NUMBER

V   Displays relative line numbers of format 41nnnnn:" before each line of text in file being edited.

-V   Disables the line number display.

OV   Prints the statistics of the memory buffer in the following format:

<div align="center">free / total</div>

where "free" is the number of free bytes in the buffer and "total" is the size of the memory buffer.

```
┌─────────┐
│    W    │
└─────────┘
```

Write

#W   Writes all of the lines in the memory buffer into a temporary file (which will become the new edited file if the E command is given before ending the edit session).

OW   Writes the first half of the buffer to the temporary file. The lines of text remaining in the buffer are shifted to the beginning of the buffer. This is a special form of the W command often used in conjunction with the OA command.

nW   Writes the first n lines of the memory buffer to the temporary file. The remaining lines of text are shifted to the beginning of the buffer.

```
┌─────────┐
│    X    │
└─────────┘
```

XFER (transfer)

nX   Transfers the next n lines of text into a temporary library file. The X command does not erase the lines transferred; an nK command given after the nX command will accomplish this process, if desired.

OX   Clears the temporary library file.

```
┌─────────┐
│   n     │
└─────────┘
```

MOVE <CP> n LINES

n   Moves <cp> to the beginning of the line n
           lines down, and types that line.

-n   Moves <cp> to the beginning of the line n
           lines up, and types that line.

           This command form is an abbreviated
           combination of the L (line) command and the
           T (type) command.

EDITING: AN EXAMPLE

For an example of how to delete and insert characters
in a line, suppose the <cp> is positioned at the beginning
of a line which reads: "Now is the time for all good men."
We wish to change this to read: "Now is the place for all
good men." The statement sequence which does this is:

Ftime<cr>
-4D<cr>
Iplace<cr>

A breakdown of this edit sequence is as follows:

Ftime<cr> Finds the character string "time" and
           positions the <cp> after the letter "e".

-4D<cr>   Deletes the four characters (t, i, m, e)
           immediately preceeding the <cp>.

Iplace   Inserts the string "place" at that point.

The above sequence could also have been typed as one
continuous string of commands:

Ftime^Z-4DIplace^Z<cr>

Note the use of -Z to delimit the strings within the
command sequence.

STORING THE EDITED FILE ON DISK

If an edit session is producing undesirable results,
the user may wish to leave the edit mode and return to
CP/M. The Q (quit) command is used to discard the edited
version of the file, keeping, instead, the original,
unedited version. All files are retained as they were
before the edit session was begun.

Upon completion of a successful edit session, use the E (end) command to leave the edit environment and store the edited file on the disk. This stores the edited file under the original filename and filetype, but keeps the old file, with the original filename and the filetype, BAK (backup). If a BAK file for the file being edited already exists on the disk (from a previous edit), it will be automatically erased at this point by the ED program.

For instance, after editing the file REPORT.TEX, the following two files will reside on the disk containing the original file: REPORT.TEX, the new edited version; and REPORT.BAK, the original version, which is now the backup file. The BAK file can be erased as soon as the new file is checked for accuracy. Do not edit a BAK file as the backup version will be lost and no new BAK file will be created. If the BAK file must be edited, first rename it to TEX or some other filetype.

2   AN INTRODUCTION TO TEX

TEX is a text formatter program designed to run under the CP/M Operating System. To use the program, two-letter commands are interspersed with lines of text in a source file. The user creates a source file using ED, the CP/M Context Editor. The Text Formatter program is then loaded from disk to process the text lines in the source file according to the embedded TEX commands. The formatted output file is created by default on disk, or at run time may be redirected to the console or line printer (see Section 4.). The combination of TEX and ED, running under the CP/M Operating System, provides the user with powerful text formatting capabilities.

With TEX, the user has complete control over horizontal and vertical spacing, and pagination. For instance, TEX commands are available to accomplish the following functions: set the number of lines per page (PL); set the line spacing to single space (SS), double space (DS), or multiple line spacing (LS); justify the right margin (AD); set the left (PO) and right (LL) margins, as well as the top (MT) and bottom (MB) margins; define and place a page heading (HE) in the top margin and a page number (PN) in the bottom margin; indent by block (IN) or by line (TI); and paragraph (PP), center lines (CE), embed comments within the text (IG), and copy text in a literal mode, exactly as typed (LI).

Refer to Figure 5 for an illustration of the initial values of various TEX commands. This figure depicts a page setting as it appears at the invocation of the TEX program.

```
       .PO8          .LL70
```
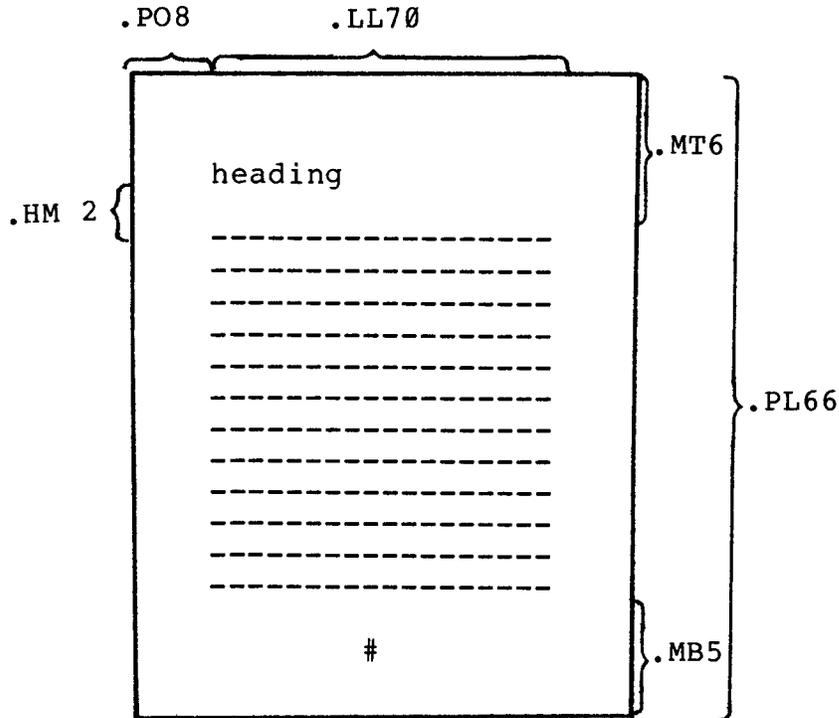

**Figure 5   Illustration showing initial integer values of
the spacing commands.**

Vertical	Horizontal
.PL 66	.PO 8
.MT 6	.LL 70
.HM 2	.IN 0
.MB 5	

2.1   Executing the TEX Program

Unless   otherwise   specified,   source   files   to   be
processed by the TEX program are assumed to have a filetype
of TEX.   The formatted output file created by the TEX
program will default to a filetype of PRN if a filetype is
not specified.

Once a source file is ready for processing, the call
to   execute   the   TEX   program   takes   one   of   the   following
forms:

> (a)   TEX x
>
> (b)   TEX x u
>
> (c)   TEX x.y
>
> (d)   TEX x.y u.v

If form (a) is used, the source file is x.TEX and the output file is x.PRN.

If the form (b) is used, the source file is x.TEX and the output file is u.PRN.

Using form (c), the source file is x.y and the output file is x.PRN.

With form (d), the source file is x.y and the output file is u.v. Note that x and u may be the same in this case.

If the user wants the input file and the output file to be stored on different disks, the input or output filename can be prefixed with the appropriate disk designation. For example, if disk A is logged in, and the user wants the output file to appear on disk B, the correct command is in one of the following formats:

```
TEX x B:
TEX x B:u
TEX x.y B:u.v
```

2.2   TEX Command Format

The commands used by the TEX program are embedded within the source file. Each command occupies a line by itself; it begins at the left margin and is preceeded by a period. The commands control such features as margin size, line length, justification of the right-hand margin, line spacing, centering, and pagination with page numbers and headings. TEX commands take one of the following forms:

(a)   ab

(b)   ab n

(c)   ab +n

(d)   ab -n

where ab is a two-character keyword constituting the actual command and n is an integer value which may or may not be required, depending on the particular command. The integer will be either signed or unsigned. The keyword may be typed in upper-case or in lower-case. A space may separate the keyword and the integer value, but it is not required.

Form (a) represents a command which cannot take an integer value. Examples of this form are AD, BR, and DS.

Form (b) represents a command which requires an absolute, unsigned integer. Generally, a command of this

form has a default value associated with it; therefore, if the integer value is omitted from a command, the default value is assumed.   Commands of this format include CE n, .LS n, PA n, and PP n.

Commands of the forms (c) and (d) require either an absolute integer, n, or a relative integer value, +n or -n. The default value applies as in form (b).   If a signed integer value is given, it specifies an incremental change to the previous value for this particular command function. Commands of form (c) and (d) include BP n, IN +n, LL -n, .MB+n, and MT n.

For example, LL is the command used to set the line length of the text in the formatted output file.   Three possible forms of the LL command are:

                    .LL 72
                    .LL +3
                    .LL -12

The first form sets the length of all lines following this command to 72 characters.   The second form increases the length of subsequent lines to three characters more (+3) than the length of preceeding lines.   The third form decreases the current line length by twelve (-12) characters.   By using this command, in one of its three forms, the line length can be changed at any point within a file.

DEFAULT VALUES

If an integer value is required by a particular command but the user omits the value, the TEX program generally assumes a default value for the command.   Before omitting a required integer, however, the user should refer to Appendix A to determine that the default value exists and is satisfactory.   If no default value is listed, an integer value should be specified in the command. otherwise, unpredictable results may occur.

The commands for the TEX program and their default values are given in the following section.   Appendix A contains a brief reference table.

INITIAL VALUES

At the beginning of the TEX program, initial integer values are assigned to many of the commands.   Therefore, if a required signed integer (see form (c) above) is used, but

no previous absolute value has been specified, the command
will increment or decrement the initial integer by +n or -n
to produce the new integer value.

For instance, if the first occurrence in the text of
the MB (bottom margin) command is MB+2, the bottom margin
value will be set to 7 lines. That is, the initial value
of 5, incremented by 2, yields 7.

If the initial value of a command parameter is
satisfactory for the user's ouptut, then that command need
not be specified. For example PL66, MT6, and PO 8.

BREAK

An important concept in TEX is the line-fill "break"
which takes place when various commands are encountered.
TEX automatically builds (or fills) output lines with input
text to comDlete a line of the specified length (LL
command). If a break occurs in a command, then the normal
line fill ceases at precisely the location of the command
in the text file. The remainder of the line is left blank.
Some commands cause a break to occur; others do not.

2.3   Run-Time Parameters

The formatted output from a TEX run is directed to a
disk file, usually a PRN file. However, the user can make
certain decisions concerning the destination of the output
at "run time," i. e., at the time of invoking the TEX
program. This is accomplished by adding a "run-time
parameter" in the initial startup of TEX. For example,

              TEX FILE $L

redirects the output to the list device.

The user can choose to redirect the output to the
console ($C), to the printing device ($L), or to suppress
($S) the output completely. Error messages can be directed
to the list device instead of the console ($E). If
continuous-form paper is not being used for printed output,
the user can specify that TEX print in page mode ($P) to
allow the user time to insert paper for each separate page
of output. A forms feed ($F) option is also available.
The user is referred to Section 4, which describes the run-
time parameters in greater detail.

3 TEX COMMANDS

The following pages contain a brief description of
each of the TEX commands.   Each command is given in its
various forms, with examples provided for clarification.
Also included in these pages are the initial values and
default conditions for each command function.

Note that there are often two or more commands which
provide identical or similar formatting effects.

Following is a list of the TEX commands in the order
of their appearance in this section.

```
AD   adjust margins
BP   begin page
BR   break
CE   center lines
CP   conditional page
DS   double space
HE   heading
HM   heading margin
IG   ignore
IN   indent
LI   literal
LL   line length
LS   line spacing
MB   bottom margin
MT   top margin
NA   no margin adjustment
OP   omit page numbers
PA   page advance
PL   page length
PN   number pages
PO   page offset
PP   paragraph
QI   quit indenting
SP   space lines
SS   single space
TI   temporary indent
```

3.1 .AD   ADJUST MARGINS

The AD command adjusts the text by moving words up to
fill lines or down to shorten lines, and by padding lines
with spaces to align both the left and right margins. The
adjust mode is the initial environment of the text
formatter program.

Two other environments which affect the line fill of
source text can be specified.   If no line filling is
desired, the literal mode can be specified, using the LI
command.   If line filling is desired, but justification of
the right-hand margin is not desired, margin adjustment can
be turned off with the NA command.   Compare the following
input and formatted output for these commands:

    INPUT

    .LL 40
    .SP
    .NA
    The CP/M Operating System can be
    easily altered to execute with almost any
    microcomputer configuration.
    .SP
    .LI
    The CP/M Operating System can be
    easily altered to execute with almost any
    microcomputer configuration.
    .SP
    .AD
    The CP/M Operating System can be
    easily altered to execute with almost any
    microcomputer configuration.

    OUTPUT

The CP/M Operating System can be easily
altered to execute with almost any
microcomputer configuration.

    The CP/M Operating System can be
easily altered to execute with almost
any microcomputer configuration.

The CP/M Operating System can be easily
altered to execute with almost any
microcomputer configuration.

3.2 .BP +-n   BEGIN PAGE

The BP command causes an eject to a new page. This command can take any one of the following forms:

(a) BP

(b) BP n

(c) BP +n

(d) BP -n

Form (a) simply begins a new page. Unless the OP option (omit page numbers) is in effect, the new page will be numbered with the next sequential integer. If the OP option is set, the new page will not be numbered.

Form (b) begins a new page and assigns the page number n to the new page. For example, if the command BP 16 is given, the next page will be page 16, regardless of the number of the previous page. Subsequent pages will be numbered sequentially from 16, unless a new n is later specified.

Forms (c) and (d) begin a new page with a page number equal to the current page number, plus or minus n. For example, if the current page number is 12 and a BP +4 command is given, a new page is begun with the page number 16.

The initial value, as well as the default value, for page numbering is +1. See Appendix C for an example of the BP command.

NOTE: If the page numbering option has been disabled with the OP command, form (a) above will not change the status of this option. However, forms (b), (c), and (d) will cause page numbering to resume according to the integer value specified.

3.3 .BR   BREAK

     The BR command causes TEX to stop filling the current output line with words from the next line. That is, the text following this command begins on a new line; it is not moved up to fill the previous line. As an example of the use of this command, consider the two input file lists below. The results are shown in the corresponding output file lists.

    INPUT

    LIST I: LIST II:
    ALPHA   br
    BRAVO   ALPHA
    CHARLIE   br
    DELTA   BRAVO
    ECHO   br
                                     CHARLIE
                                     .br
                                     DELTA
                                     .br
                                     ECHO
                                     .br

    OUTPUT

LIST I: ALFA BRAVO CHARLIE DELTA ECHO

LIST II:
ALFA
BRAVO
CHARLIE
DELTA
ECHO

3.4 .CE n   CENTER LINES

The CE command centers (horizontally) the next n lines of text. Forms for the CE command are as follows:

(a) CE

(b) CE n

The default value for n is 1; therefore, if form (a) is used, TEX will center the single line of text following the command.

If form (b) is used, the n lines of text following CE will be centered.

This function is very useful for centering letterheads and setting titles. As an example, the following block of input text will result in the output below:

INPUT

```
.in +8
.ce 5
TECHNICAL REPORT ON SITE VISIT
XYZ ANTENNA ARRAY SITE
GENERAL NOISE CORPORATION
NOWHERE, MONTANA
January 3, 1976
.pp
This report will provide the technical details of
the site visit by three engineers
from our research institute.
```

OUTPUT

<div align="center">

TECHNICAL REPORT ON SITE VISIT

XYZ ANTENNA ARRAY SITE

GENERAL NOISE CORPORATION

NOWHERE, MONTANA

January 3, 1976

</div>

This report will provide the technical details of the site visit by three engineers from our research institute.

3.5 .CP n   CONDITIONAL PAGE

The CP command is a "conditional begin-page" function. This command is valuable when the user desires to begin a new page if, and only if, the next n lines of text will not fit on the current page.

For instance, at the beginning of a new section of a report which contains a title, it would be undesirable for the title to appear at the bottom of a page with no text immediately following it. To avoid this situation, the user can precede the title and text with a CP n command, where n specifies the minimum number of lines the user wants printed on the same page. Following is an example of this situation:

INPUT

.cp 10
.ce
PERIOD OF REPORT
.pp
The period of this report is from January 12, 1977, to October 30, 1977. The report covers all the activities of this office during this period.
The activities of this office will be enumerated in some detail, but will be discussed only in light of their impact on the net worth of the corporation as a whole. Activities of an internal nature will be discussed in another report.

NOTE: The above entry in an input file causes the text formatter to count the number of lines which can fit on the current page. If 10 lines (beginning with the title line "PERIOD OF REPORT") cannot fit on the current page, then TEX begins a new page, starting with the title line.

3.6 .DS   DOUBLE SPACE

The DS command causes the output text to be vertically double-spaced.   This   is   equivalent   to   using   the   line spacing command LS 2.

INPUT

```
.ll 40
.ds
To produce a double-spaced report,
just code the keyword DS
or, optionally, code LS 2.
```

OUTPUT

To produce a double-spaced report, just

code the keyword DS or, optionally,

code LS 2.

3.7   .HE string   HEADING

The HE command allows the user to define a heading to
be printed at the top of subsequent pages. The exact
location of the heading within the top margin is determined
by the HM command.   Once a heading is defined, it will
print on every page following the command, unless the
heading is changed or eliminated.   A heading can be changed
by simply entering another HE command to redefine the
heading.   A heading is eliminated, or nullified, by typing
an empty HE command into the source file (see form (b)
below).

Valid formats for the HE command are as follows:

                    (a)   HE string

                    (b)   HE

where "string" is any character string, not to exceed the
line length set by the LL command.   One space must separate
".HE" and the heading string.   All character positions
(characters, blanks, etc.) following this blank, and
preceding the carriage return, are considered to be the
contents of the heading.   If the heading exceeds the
current line length, it will be truncated (from the right)
to the current line length.   Form (b) sets the heading to
blank.

NOTE: The left margin set by a PO command remains in
           effect when printing the heading; however, other
           indenting values are ignored.

Following are some examples showing the heading
results of various HE input strings.   Assume a page offset
of 18 (.PO 18) and a line length of 45 (.LL45).   (Note the
truncation in the third example.)

     INPUT

  1->   .he THE CREATIVE PROGRAMMER   Chapter 1
  2->   .HE CENTERED TITLE
  3->   .he   Section Twenty

     OUTPUT

  1->   THE CREATIVE PROGRAMMER   Chapter 1
  2->   CENTERED TITLE
  3->   Section Twen

3.8 .HM +-n   HEADING MARGIN

The .HM command determines the location of the heading (see .HE command) within the top margin. It defines the number of lines to be left blank between the heading and the body of the text. The following command formats are valid:

<div align="center">

(a)   .HM n

(b)   .HM +n

(c)   .HM -n

</div>

Form (a) defines the absolute number of lines in the heading margin.

Form (b) defines an increase of n lines more than the initial, or previously defined, heading margin. The initial value of the heading margin is 2 lines.

Form (c) decreases the current heading margin by n lines.

Refer to the example of the following page for an illustration of the HM command.

NOTE: The size of the top margin should always be considered whenever the HM command is used. If the top margin does not contain at least one line more than the heading margin, TEX will automatically reduce the size of the heading margin in an attempt to fit the heading into the margin space available. A non-fatal message is printed to indicate the margin reset. If the user wishes to change the top margin and the heading margin simultaneously, the MT command should always be given first to avoid an automatic margin reset of this sort.

The following example illustrates the interaction of the HM (heading margin) command and the MT (top margin) command with the HE (heading) command. In the first example, the initial values for the top margin (.MT 6) and the heading margin (.HM 2) are in effect. Example 2 illustrates a simultaneous change to both margins, thus relocating the heading.

EXAMPLE 1:

INPUT

.he The CP/M Console Companion

OUTPUT

```
1:
2:
3:
4:The CP/M Console Companion
5:
6:
          (body of text)
```

EXAMPLE 2:

INPUT

```
.MT 5
.HM 3
.HE Chapter Two: DYNAMIC DEBUGGING TOOL
```

OUTPUT

```
1:
2:Chapter Two: DYNAMIC DEBUGGING TOOL
3:
4:
5:
            (body of text)
```

3.9 .IG   IGNORE

    The IG command causes the text formatter program to
ignore completely all lines in the input file between the
.IG command and the next command encountered.   This allows
the user to insert comments into the input file which will
not be reproduced in the output file.   For example:

INPUT

```
.LL 50
The parasitic oscillations were eliminated by
correcting the power supply filtration
inadequacies.
.ig
Note for technicians: The power supply problem was
caused by a leaky capacitor, C7.
.Ss
Elimination of the parasitics reduced the system
noise by about 87 decibels.
```

OUTPUT

    The parasitic oscillations were eliminated
by   correcting   the   power   supply   filtration
inadequacies.   Elimination of the parasitics
reduced the system noise by about 87 decibels.

3.10 .IN +-n   INDENT

The IN command indents text on subsequent lines until either a QI (quit indenting) command or another IN command is given. The IN command can take one of the following forms:

                    (a)   IN n

                    (b)   IN +n

                    (c)   IN -n

If form (a) is given, the subsequent text is indented spaces from the left-hand side of the page, or from the left margin, if a page offset value is in effect (see note below). Following the execution of this command, the current indent value is n.

Form (b) will increase the current indent position by n spaces. Note that the initial indent value is 0.

Form (c) will decrease the current indent position by n spaces, to a minimum value of 0. For instance, if the current indent value is 5 (.in 5) and a command is given to move the indent position to the left 10 spaces (.in -10), the resulting indent value will be 0, or the equivalent of an IN -5 command. TEX will not allow an indent value less than 0, even if a previous PO command has been given to provide a constant left margin.

NOTE: It is important to distinguish between the PO (page offset) command and the IN (indent) command. Both commands can change the left margin. Generally the PO command is specified once at the beginning of a report to provide a constant left margin setting (the page offset defaults to 8 if not specified). If an indent value is set by the IN command, it will be added to the page offset value to indicate the total number of spaces from the left side of the page to skip. Under no circumstances will printing occur within the left margin, i.e., to the left of the page offset value.

Consider this example of the IN command:

INPUT

.in 8
.ll 45
System Startup of a Hypothetical Machine.

```
        .pp
        System startup under CP/M is simple
        and straightforward, and
        consists of the following steps.  The steps
        must be performed in sequence.
        .in +5
        .sp
        1.  Turn on system power (CPU, disk drives,
        console).
        2.  Press the RESET switch.
        3.  Check to see that the CP/M sign-on message
        appears on the console.
        4.  If message does not appear, retry steps l.-3.
        .in -5
        If any difficulties are encountered, call a
        technician.
```

OUTPUT

System Startup of a Hypothetical Machine.

    System startup under CP/M is simple and
straightforward, and consists of the following
steps. The steps must be performed in sequence.

        1.   Turn on system power (CPU, disk drives,
        console).
        2.   Press the RESET switch.
        3.   Check to see that the CP/M sign-on
        message appears on the console.

    If any difficulties are encountered, call a
    technician.

    Note that, in the input file above, the command in 5
could have been used instead of in +5, and that .qi could
have been given in place of in -5.

3.11 .LI   LITERAL

The LI command is used to discontinue the filling and justification functions of the text formatter. This is usually done to enable the insertion of tables or columnized information. For example, if the user wishes to print the list shown below without having to insert BR (break) commands after each line, the LI command could be used. This command is terminated by any other command, generally one which will have no effect, such as .SS or .AD.

    INPUT

    .li

        ALFA   ONE   Al
        BRAVO   TWO   B2
        CHARLIE   THREE   C3
        DELTA   FOUR   D4
        ECHO   FIVE   E5
        FOX   six   F6

    OUTPUT

        ALFA   ONE   Al
        BRAVO   TWO   B2
        CHARLIE   THREE   C3
        DELTA   FOUR   D4
        ECHO   FIVE   E5
        FOX   six   F6

NOTE: The following commands, which set the physical limits of a page and define the margin contents, remain in effect during execution of the LI command:

          PL   page length
          PO   page offset
          MT   top margin
          HM   heading margin
          HE   heading
          MB   bottom margin
          OP   omit page numbers
          PN   number pages

Upon termination of the literal mode, these commands will again have control. All other commands are ignored during the literal copy.

3.12 .LL +-n   LINE LENGTH

The LL command sets the line length (in character
positions) of the formatted output text.   The initial value
is 70 characters per line.   Following are the valid formats
for this command:

    (a)   LL n

    (b)   LL +n

    (c)   LL -n

Form (a) sets the length of subsequent lines to n
character positions.

Forms (b) and (c) increase and decrease, respectively,
the current line length by n.

This command can be used at any point in a file to
change the line length temporarily, as well as at the
beginning of a file, to set the permanent line length for
the width of the paper being used.

See the following page for an example of changing the
line length in a file.

The line length of a file might be changed temporarily
in order to leave space for the insertion of a picture or
drawing on the page.   Consider this example:

INPUT

The use of a telephone coupler enables a terminal
to be connected to an on-line computer or remote
terminal by utilizing the telephone networks.
.LL −30
The Model 899, outlined herein and pictured at the
right, operates specifically with the type A terminal.
The compact Model 899 coupler converts the terminal's
output data into selective tones, which are then
transmitted over standard voice-grade telephone lines.
Conversely, this Model 899 telephone coupler decodes
selective audio tones into digital signals for
interface
to the terminal.
.LL +30
Full duplex and half duplex operation is selected at
the
terminal. A data set or an automatic answering
coupler
may be used to interface signals between the

        customer's
        telephone line and the computer.

        OUTPUT

The use of a telephone coupler enables a terminal to be
connected to an on-line computer or remote terminal, by
utilizing the telephone networks.   The model 899, outlined
herein and pictured at the right, operates specifically with
the   type   A   terminal.   The
compact   Model   899   coupler
converts   the   terminal   output
data   into   selective   tones,
which   are   then   transmitted
over   standard   voice-grade
telephone lines.   Conversely,
this Model 899 coupler decodes
selective   audio   tones   into
digital signals for interface
to the terminal.   Full duplex and half duplex operation is
selected at the terminal.   A data set or an automatic
answering coupler may be used to interface signals between the
customer's telephone line and the computer.

3.13 .LS n   LINE SPACING

The LS command sets the vertical spacing between lines to n spaces. For single spacing the SS command may be used and is equivalent to LS 1; for double spacing the DS command may be used and is equivalent to LS 2.

The initial value of n is 1: (single spacing).

INPUT

```
.in8
.ls 1
One of the CP/M built-in commands is DIR (directory).
Typing DIR at the console lists the names of all
files which satisfy a given ambiguous filename.
Examples of valid commands are:
.br
.ls3
DIR x.y
.br
DIR x?z.c??
.   br
DIR *.asm
```

OUTPUT

One of the CP/M built-in commands is DIR (directory). Typing DIR at the console lists the names of all files which satisfy a given ambiguous filename. Examples of valid commands are:

DIR x.y

DIR x?z.c?m

DIR *.asm

3.14 .MB +-n   MARGIN, BOTTOM

The MB command sets the number of lines to be left blank at the bottom of each page. The initial value is 5 lines. These blank lines are the bottom page margin. The user may eliminate the bottom margin by coding MB 0. Forms for the MB command are as follows:

(a) MB n

(b) MB +n

(c) MB -n

Form (a) leaves a bottom margin of n lines.

Form (b) will increment the current bottom margin value by n. If no bottom margin has been given explicitly, the initial value of 5 is used as the current value.

Form (c) decreases the last bottom-margin specified by n lines.

NOTE: If the automatic page numbering feature is used, the page number will be centered vertically and horizontally within the bottom margin.

3.15 .MT +-n   MARGIN, TOP

The MT command controls the size of the top margin. The number of lines to be left blank at the top of each page is specified by n.   The initial value for n is 6 lines.

The top margin may be blank or may contain one heading line to be printed on each page (see the HM command). Formats for the MT command include

> (a)   MT n
>
> (b)   MT +n
>
> (c)   MT -n

Form (a) defines a top margin of n lines.

Form (b) defines a top margin containing n lines more than the top margin last specified.   If no margin has been previously specified, the initial value of 6 is assumed.

Form (c) is identical to form (b) except that the current top margin value is decreased by n lines.

NOTE: If the user wishes to include a header title, the top margin must contain at least two lines.   If the heading option is in effect, a change to the top margin value may affect the placement of the heading.

3.16 .NA   NO ADJUST

The   NA   command   suppresses   the   normal   right-margin
alignment function of the AD command.   The filling of short
lines with words from subsequent lines remains in effect.
Following is an example of text processed with no margin
adjustment.

INPUT

```
.ll 40
.na
The ED Context Editor contains a number
of commands which enhance its usefulness
in text editing.  The improvements
are found in the addition of line numbers,
free space interrogation,
and improved error reporting.
```

OUTPUT

```
The Ed Context Editor contains a
number of commands which enhance
its usefulness in text editing.
The improvements are found in the
addition of line numbers, free
space interrogation, and improved
error reporting.
```

3.17 .OP   OMIT PAGE NUMBERS

The OP command suppresses the printing of page numbers on each page. Page numbering will resume upon encountering the next BP command with n specified or the next PN command.

NOTE: This command does not cause a break, nor does it cause paging. It takes effect at the beginning of the page immediately following the OP command.

3.18 .PA n   PAGE ADVANCE

The PA command causes n blank pages to eject. If the page numbering option is in effect, the pages will be numbered sequentially. Following are the possible formats of the PA command:

(a) PA

(b) PA n

If form (a) is used, the default value of 1 is assumed for n. Thus only the current page will eject. Note that this form is equivalent to the command BP (with no n value specified).

Using form (b) will cause n page ejects to occur. These pages will be numbered sequentially, provided the page numbering option has not been turned off with the OP command.

NOTE: The commands PA n and BP n are similar and should not be confused with one another.

- Both commands cause an immediate page eject. BP n will cause only one eject; PA n will cause n page ejects.

- The page numbering option is not affected by a PA n. Ejected pages are numbered consecutively, or not at all if the page numbering feature is turned off. The page numbering feature can be turned on, and the current page number reset, by a BP n.

  Please refer to the paging example in Appendix C.

3.19 .PL +-n   PAGE LENGTH

The PL command defines the length, in lines, of the paper being used for printing the formatted output file. This command is used in conjunction with the top and bottom margin commands to set the proper text length on each page. The initial value is 66 (the number of lines on a standard, 11-inch sheet of paper). Following are the forms for the page length command:

> (a) PL n
>
> (b) PL +n
>
> (c) PL -n

Form (a) resets the logical page length to n lines.

Form (b) increases the current page length by n lines. If no PL command has been previously specified, the initial value of 66 is assumed.

Form (c) is the same as form (b) except that the current page length is decreased by n lines.

NOTE:
Compare **Logical** **Page** **Length** **to** **Physical** **Page** **Length**

Although the page length value is generally set to reflect the physical page size, it is possible to set the page length to any value; n determines the logical, not the physical, size of the page.

For a page length value which is less than the physical page size, consider using the forms-feed parameter at run time (see Section 4.) to automatically eject the pages. Of course, this option applies only to printing devices with a forms-feed capability.

When changing the page length with a PL command, it may be necessary to adjust the top and bottom margins accordingly. If margins exceed a new page length, TEX will automatically reduce them, giving a non-fatal error message on the error output device.

Please refer to the paging example in Appendix C.

3.20 .PN +-n   PAGE NUMBER

The PN command turns on the page numbering option and sets the current page number according to the value of n. Subsequent pages are numbered consecutively.   The page numbering option is in effect initially; it can be turned off by the OP command.   The default page count value is 1 if n is omitted.   Page numbering command formats are as follows:

<div align="center">

(a)   PN

(b)   PN n

(c)   PN +n

(d)   PN -n

</div>

If form (a) is used, page numbering will resume with the current page number set to 1.

Form (b) sets the number of the current page to n.

Forms (c) and (d) will increment and decrement, respectively, the page number to print on the current page. For example, if the last page number is 3 and a PN+5 command is given, the new page number will be 9.   That is, the next number to print is 4; add 5 to get a new page number of 9.

The PN command may be used to (1) turn on the page numbering option following an OP command, (2) begin the page numbering sequence at a number other than 1, or (3) change the sequence of page numbers.

EXAMPLE 1:

If the first page is a title page and the user desires to start numbering pages on page 2, the usual TEX commands inserted at the beginning of the source file should include OP, and PN 2 should be embedded within the text between the first and last lines of the title page.

EXAMPLE 2:

The file being processed is chapter 3 of a book; chapter 2, when processed, ends on page 41. If the command PN 42 is inserted at the top of the input file for chapter 3, then the first page of the chapter 3 output file will be numbered page 42.

3.21 .PO +-n   PAGE OFFSET

The PO command sets the left page margin and provides a left-margin capability for those printers which do not have a mechanical margin setting. Although generally given at the beginning of a report, the page offset can be reset whenever necessary. The initial value for n is 8. Thus if no PO command is given, an automatic left margin of 8 spaces will be set. If the user does not want a left margin, a PO 0 should be specified at the beginning of the source file. Following are the forms for the PO command:

                    (a) (a) PO n

                    (b) (b) PO +n

                    (c) (c) PO -n

Form (a) offsets the formatted text n spaces to the right from the left-most printer position.

If form (b) is used, the text is offset n spaces more than the current page offset value. If no PO command has been previously issued, the text will be offset from the initial position of 8 spaces, or 8+n spaces.

Form (c) is similar to form (b), except that the current page offset value is decreased by n. If n is greater than the current value, the offset will be 0; printing will resume from the left-most printer position.

See the example on the following page.

INPUT

ED is the Context Editor for CP/M. With ED, the user can create and alter CP/M source files.
.sp
.PO 0
In general, ED reads segments of the source file into central memory, where the file is manipulated by the user and subsequently written back to disk after alterations.
.sp
.po 5
If the source file does not exist before editing, it is created by ED and initialized to empty.

OUTPUT

ED is the Context Editor for CP/M. With ED, the user can create and alter CP/M source files.

In general, ED reads segments of the source file
into central memory, where the file is
manipulated by the user and subsequently written
back to disk after alterations.

    If the source file does not exist before editing,
    it is created by ED and initialized to empty.

NOTE: In the above example, remember that the initial
           page offset value is 8 spaces.

    Note that the output example is simulated to
demonstrate the relative differences between the various PO
commands. An actual PO 0 would cause the text to print
against the left-hand edge of the page and be lost in the
binding.

    Note also that the line length is not changed (this
manual is printed with a line length of 60); therefore, an
increase in the page offset value causes the entire print
line to shift to the right. The IN command adjusts the
line length to keep an even right margin; the PO command
does not.

3.22 .PP n   PARAGRAPH

The PP command inserts one blank line space and indents the next line of text n positions. This command is a combination of the SP (space) command and the TI (temporary indent) command.   The initial value of n is 6 spaces. Forms for the paragraph command are

            (a)   PP

            (b)   PP n

Generally form (a) will be used for paragraph control; the last specified value of n is assumed.   If no previous .PP n command has been given, the initial value of 6 spaces is assumed.

Form (b) indicates paragraphing with an indent value of n spaces.   After issuing this command form, subsequent PP commands will also indent n spaces.

An example of the paragraph command is shown below.

INPUT

```
.ll 35
.pp
Good documentation is vital to
every EDP operation.
.pp 3
Documentation standards should be well-defined.
.pp
Active participation and support of EDP managers
in the process of standardization
is essential.
```

OUTPUT

```
    Good documentation is vital
to every EDP operation.

   Documentation standards should
be well-defined.

   Active participation and support
of EDP managers in the process of
standardization is essential.
```

3.23 .QI   QUIT INDENT

The QI command cancels a previously set IN (indent)
command.   The indent value is reset to 0.   This command has
no effect on the left margin value set by the PO (page
offset) command.

Consider the following example of the QI command:

INPUT

```
.ll 45
TEX sets up two margins on the left side of the page.
.sp
.in 8
The left margin, or page offset, is controlled by the
PO
command.  The IN command controls the indent margin.
.sp
.qi
A QI command cancels only the indent margin.
The left margin is altered only by another PO command.
```

OUTPUT

TEX sets up two margins on the left side of
the page.

The left margin, or page offset, is
controlled by the PO command.   The IN
command controls the indent margin.

A QI command cancels only the indent margin.
The left margin is altered only by another
.PO command.

3.24 .SP n   SPACE LINES

The SP command inserts n blank "line spaces" into the formatted output (see the note below). The initial value of n, as well as the default value, is 1. Valid forms for the SP command are as follows:

(a)   SP

(b)   SP n

Form (a) is the default form; it is identical to SP1. Entering this command in text will generate one blank line space (i.e., one blank line if single spacing, two blank lines if double spacing, and n blank lines if an LS n command is in effect) .

Form (b) will generate n blank line spaces. It is important to note that the SP n command will not allow spacing past the end of a page.

For example, if an SP 10 is encountered five lines from the bottom of a page, TEX will generate only enough blank line spaces to reach the bottom margin. The text following the SP10 command will print at the top of the next page, immediately after the top margin.

In some cases, it may be desirable to leave blank line spaces at the top ot a page. This is accomplished by preceding the SP command with a BP or PA command.

NOTE: One "line space" is the number of lines specified in a previous line spacing command:

.SS specifies 1 line per line space (default spacing).

.DS specifies 2 lines per line space.

.LS n specifies n lines per line space.

3.25 .SS   SINGLE SPACE

The SS command sets the vertical line spacing to
single space; this cancels a previously set LS command.
The single space condition is the initial line spacing
condition of the TEX program.

Note that SS is equivalent to the LS 1 command.

3.26 .TI +-n   TEMPORARY INDENT

The TI command causes the next line (one line only) to indent from the left margin of the page. The temporary indent, like the indent function, will indent to the right starting at any page offset value in effect. The TI command will indent to the left past the indent margin (set by the IN command) currently in effect; however, TEX will not allow indentation to the left past the left margin boundary (set by the PO command).

The following forms are valid for the temporary indent command:

(a) TI n

(b) TI +n

(c) TI -n

Form (a) will indent the next line n print positions from the left margin (see the PO command). If an indent margin has previously been set with the IN command, this indent value will remain in effect for subsequent lines.

Form (b) will indent the next line n positions more than the current indent value.

Form (c) will indent the next line n positions less than the indent value currently in effect. For this form, n cannot have a value greater than the last specified indent value.

See the following page for an example illustrating the flexibility of the TI command.

INPUT

BDOS Error Messages
.sp
.ti +3
There are three error conditions which the CP/M BDOS
intercepts during file processing.
.sp
.in 12
.ti -12
BAD SECTOR   Disk controller electronics have detected
an error condition in reading or writing the diskette.
.ti -12
SELECT   An attempt has been made to address a drive
other than drive A, B, C, or D.
.ti -12

READ ONLY User has attempted to write to a diskette
which has been designated as read-only.

        OUTPUT

BDOS Error Messages

   There are three error conditions which the CP/M BDOS
intercepts during file processing.

BAD SECTOR   Disk controller electronics have detected an
                error   condition   in   reading   or   writing   the
                diskette.
SELECT   An attempt has been made to address a drive
                other than drive A, B, C, or D.
READ ONLY   User has attempted to write to a diskette
                which has been designated as read-only.

4   TEX RUN-TIME PARAMETERS

Normally, the formatted output from a TEX program execution is placed in a file on disk. However, if the user wishes to direct the output to the console device or to the list device, he may do so at the time of executing the TEX program. other decisions may be made at run time which concern the actual print form and hardware capabilities.

The user may specify one or more run-time parameters when calling the TEX program. The format of the TEX program invocation, including parameters, is as follows:

TEX file $p

where "file" is one of the filename forms described in Section 2.1., and "p" is a string of one or more run-time parameters as described below. The first parameter must be preceded by a "~" symbol. If additional parameters are specified, each one may be optionally preceded by a $ or blanks. Consider the following possible statements for calling TEX with multiple parameters specified at run time:

TEX ABC $L F

tex filex $pe

TEX XYZ $S $E

NOTE: If more than one parameter is listed in the TEX program call, the attribute of the parameter specified last is in effect if conflicting attributes exist. For instance, the parameter string $PC will list the output on the console device instead of the list device, stopping after each page until the user presses the carriage return <cr> key of the console device.

4.1   Parameters Redirecting Output

The following parameters redirect the output from the disk file (either the default PRN file or one specified by the user); the last parameter redirects the error messages only.

The first three parameters (L, C, and S) describe alternate destination options for the formatted output from a TEX run. The redirected output will be the only formatted version to result from the TEX run. A PRN file will NOT be placed on disk. only one of these parameters should be specified at a time. If more than one is

inadvertently given at run time, the last parameter
specified will be the one to affect the output destination.
An exceptionally long listing which is directed to the
console or printer will be aborted if the user presses the
carriage return <cr> key.

The last parameter (E) in Section 4.1 specifies an
alternate output for the messages which the TEX program
displays when an error is detected. The formatted output
will be placed on disk as usual, in a PRN or user-specified
file, unless one of the output parameters is specified
simultaneously.

$L   OUTPUT TO LIST DEVICE

The L parameter causes the formatted output to print
on whatever device the user has assigned as the logical
LST: device while under control of the CP/M system. This
is generally, but not necessarily, a printer device.
Pressing the carriage return <cr> key will cause the
listing to abort.

$C   OUTPUT TO CONSOLE DEVICE

The C parameter causes the formatted output to print
on the device which has been assigned to the CON: logical
device name. This is usually a type of CRT console device.
A listing display to the console device may be aborted by
pressing the carriage return <cr> key.

$S   SUPPRESS OUTPUT

The S parameter suppresses the formatted output
completely. No output file is listed or preserved on file.
Error messages, however, are still printed on the console
device or on the list device (see the E parameter).

This parameter is particularly useful for a large text
file which the user wishes to check for TEX errors before
saving the formatted output.

$E   ERRORS TO LIST DEVICE

The E parameter redirects the error messages to the
list device. This parameter may be used in conjunction
with any of the above parameters (L, C, or S), as it
controls ONLY the error message output.

If a text input file has several errors, it is often
difficult to read the error messages on a console. This

parameter enables the user to obtain a hard copy of the
error messages for closer analysis.

4.2 Printing Aids

The following run-time parameters are somewhat
dependent upon the physical printing capabilities of the
user's hardware. They may be used in conjunction with
parameters from Section 4.1.

$P   PAGE MODE

The P parameter specifies that printing is to occur in
"page mode." This means that TEX will stop printing after
each logical page is complete; printing will continue ONLY
after the user presses the carriage return <cr> key on the
console keyboard. If output is not directed to the console
device, the TEX program will print a message on the console
instructing the user to insert a new page and press the
carriage return <cr> key to continue printing. If output
is directed to the console device ($C), the instructions to
the user will not be printed, but the same action is
required: when the printing stops, the user should change
paper (if applicable) and press the carriage return <cr>
key.

This parameter is necessary if the user wishes to
obtain any hard copy which is not on continuous form.

$F   FORMS FEED

The F parameter utilizes the forms-feed capability
which exists on many printers. This feature causes the
printer to skip to the physical end of a page in one
motion, instead of spacing line by line. TEX can only take
advantage of a printer's forms-feed ability in the
following situation:

If the logical page size differs from the physical
page size of the continuous form being used, the F
Parameter will cause one logical page to print on each
physical sheet of paper. For example if the user is
printing pages with page length 50 (.PL 50), but the
continuous form used has 66 lines per sheet, the F
parameter will cause a page eject, or a forms feed, to
occur after printing each page.

APPENDICES

Appendix A: SUMMARY OF TEX COMMANDS

Function	Break?	Command Format	Initial Value	Default Value
Adjust Margins	yes	AD	on	
Begin Page	yes	BP +-n		+1 *
Break	yes	BR		
Center	yes	CE n		1
Conditional Page	yes **	CP n		
Double Space	yes	DS	off	
Heading	no	HE s	off	blank
Heading Margin	no	HM +-n	2	
Ignore	no	IG		
Indent	yes	IN +-n	0	0
Literal	yes	LI		
Line Length	no	LL +-n	70	
Line Spacing	yes	LS n	1	
Margin, Bottom	no	MB +-n	5	
Margin, Top	no	MT +-n	6	
No Adjustment	yes	NA	off	
Omit Page Numbers	no	OP	off	
Page Advance	yes	PA n		1
Page Length	no	PL +-n	66	
Page Number	no	PN +-n	1	1
Page Offset	yes	PO +-n	8	0
Paragraph	yes	PP n	6	last n
Quit Indentation	yes	QI		
Space n Lines	yes	SP n	1	1
Single Space	yes	SS	on	
Temporary Indent	yes	TI +-n	0	

Table A–1

*   only if page numbering is in effect
**   only if page eject occurs

Appendix B: SUMMARY OF TEX RUN-TIME PARAMETERS

| PARAMETER	FORMAT	ACTION
Console Listing	$C	Redirects output to print on console device.
Listing of Errors	$E	Redirects error messages to print on list device.
Forms-Feed	$F	Enables user to forms-feed on printer, if capability exists.
Printer Listing	$L	Redirects output to print on list device.
Page Mode	$P	Stops printing at end of page, for paper change; resumes printing with <cr>
Output Suppression	$S	Produces no output file; prints error messages as usual.

Table B-1

Appendix C: PAGING EXAMPLE

It is difficult to present examples of the paging commands in text, without losing track of the overall effect of the commands. Therefore, consider these simulated diagrams of text which demonstrate the use of the following commands:

```
.BP +-n    Begin new page; number according to n.
.0p        omit page numbering.
.PA n      Advance n pages.
.PN +-n    Number pages according to n.
```
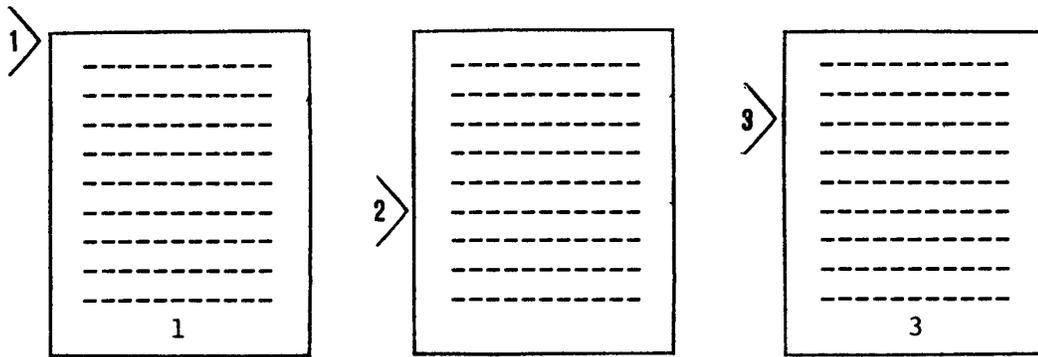

1) Normal default page numbering.
2) OP command issued at this point suppresses page numbering. Current PN value is 2.
3) PN+1 instructs TEX to begin numbering pages, startinq at the current page number plus one.
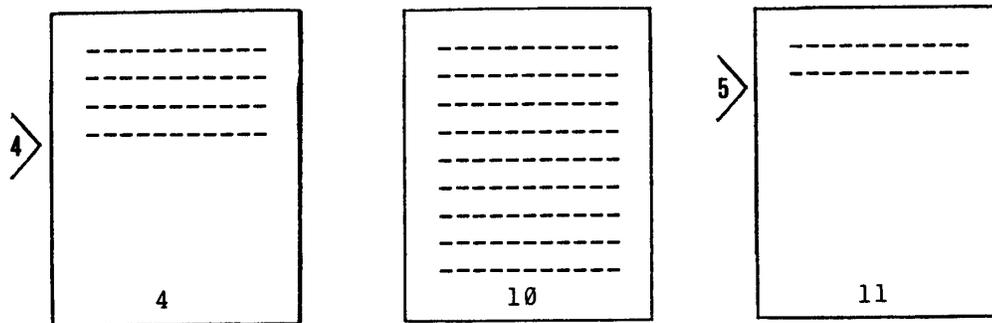
4) BP10 causes a page eject; page number is reset to 10.
5) PA 3 produces the eject of three pages, each of which is numbered consecutively.
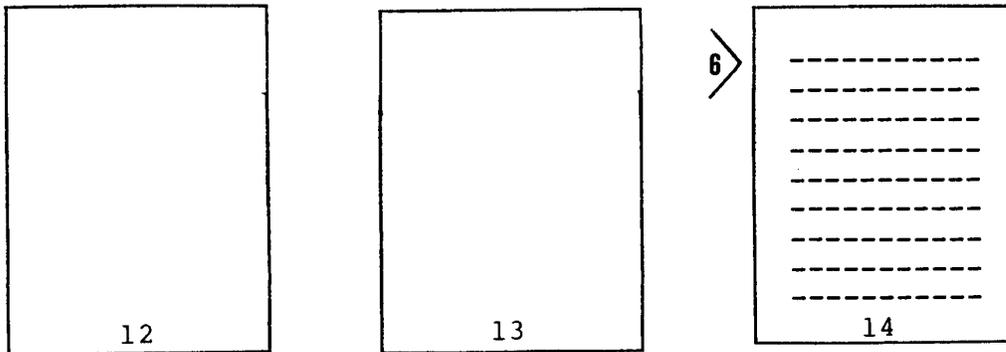
6) Formatted text continues printing at the top of the third
page.

Appendix D: SUMMARY OF TEX ERROR MESSAGES

TEX generates error messages to indicate various problems encountered in the input text file. Some of these messages are fatal, i.e., they cause the TEX program to terminate without further processing of the source file. Other messages merely indicate the occurrence of an error or of a problem situation. In such a non-fatal situation, TEX will continue to format the output as usual, following the printing of the error condition.

The format of all error messages is as follows:

ERROR DURING
error-condition statement

INPUT CONTEXT:
text containing error condition

If the error is a fatal error, an additional statement prints on the error device:

THIS ERROR TERMINATES TEX.

The error-condition statement will vary according to the error. See the following page for a summary of these conditions.

The text following the INPUT CONTEXT: designation is printed to give the user as much information as possible for locating the problem area. The error generally as found at the end of this segment of text.

ERROR-CONDITION STATEMENTS:	EXPLANATION CAUSE:	ACTION TO CORRECT:
OPENING SOURCE *	Invalid or missing source file name.	Correct source file name.
PARAMETER SCAN *	Invalid character in parameter string.	Correct parameter (see Section 4).
FILE READ	Invalid file format.	Input file cannot be random mode.
DISK WRITE	Cannot write output file from memory to disk.	Disk or disk director full; condense disk or transfer files.

ERROR-CONDITION STATEMENTS:	EXPLANATION CAUSE:	ACTION TO CORRECT:
READING EOF	End-of-file mark detected where not expected.	Check for missing command or <cr>.
FITTING A WORD	Usually, a long word cannot be formatted successfully into a short line length.	Reword text or increase line length.
COMMAND VERIFY	Invalid command after a "." at beginning of input line.	Correct command; check for accidental "." at beginning of line.
HM COMMAND	Heading margin value greater than or equal to top margin.	Correct HM or MT value if changing both, reset MT value first. (TEX resets HM=MT-1)
MB COMMAND	Bottom margin greater than or equal to PL value - MT value.	Correct MB or PL value if changing both, reset PL value first.
MT COMMAND	Top margin greater than or equal to PL value - MB value.	Correct MT or PL value if changing both, reset PL value first.
PP COMMAND	Paragraph indent value greater than line length.	Correct PP or LL value

Table D-1