SOFTWARE ENGINEERING SERVICES

TXTCODE 2.0

Reference Manual

60460280 01

REVISION RECORD

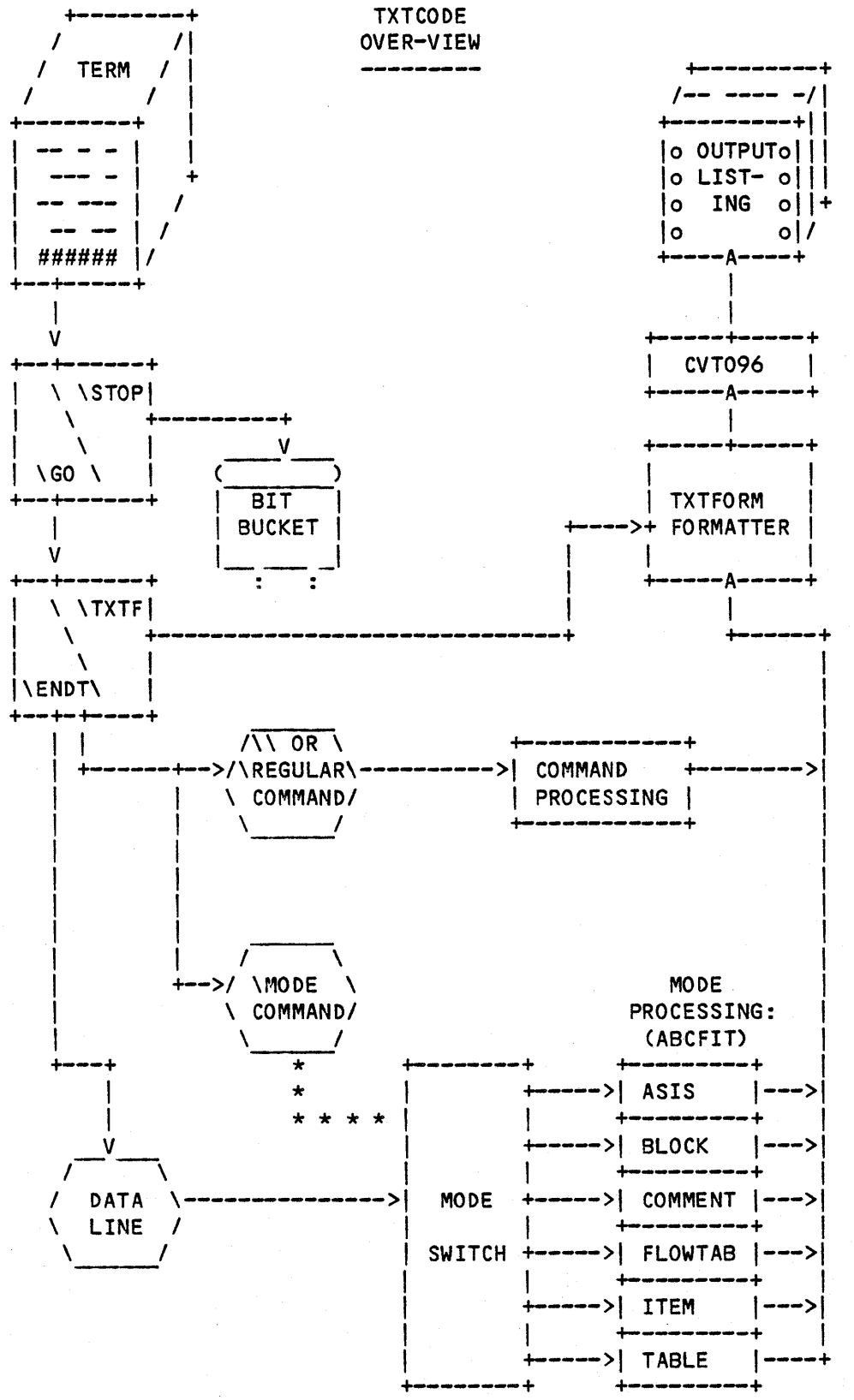
REVISION		DESCRIPTION
01	(02-17-84)	Preliminary manual released.

Address comments concerning this manual to:

Control Data Corporation
Software Engineering Services
4201 North Lexington Avenue
St. Paul, Minnesota 55112

60460280 01

Table of Contents


TXTCODE
OVER-VIEW
TERM
OUTPUT
LIST-
ING
CVT096
\STOP
\GO
BIT
BUCKET
TXTFORM
FORMATTER
\TXTF
\END
OR
REGULAR
COMMAND
COMMAND
PROCESSING
MODE
COMMAND
MODE
PROCESSING:
(ABCFIT)
DATA
LINE
MODE
SWITCH
ASIS
BLOCK
COMMENT
FLOWTAB
ITEM
TABLE

1.0 INTRODUCTION

Thus far, automated document generation here at CDD has been accomplished using low level commands which, for the most part, only work on immediate and limited portions of text. Few commands exercise significant control over large portions of text. As a result, document preparation is hampered by the relatively large amount of work needed to prepare a text file for submission to a text formatter such as TEXTJAB or TXTFORM. In TXTFORM, lines containing commands often outnumber lines of text. Thus, the input file to the formatter is hard to read and work on.

TXTCODE is designed to allow high level formatting instructions, which correspond directly to common text structures, to be used to direct document generation. In this way, commands of great power and range are able to format a whole "text structure" using, in some cases, only an escape code and a word.

TXTCODE is designed so that the text being input by the user will help to trigger in his mind the appropriate TXTCODE command, or format. The entered text also suggests to the user the format of the original. Since relatively few codes are present the input file is normally very readable. Thus on-line editing is simpler too - both visually and from an EDITing standpoint. TXTCODE's input format, which is basically free-form, allows lines and text strings to be freely replaced, added and deleted.

2.0 MODE OF OPERATION

2.0 MODE OF OPERATION

Philosophy

The basic idea of TXTCODE is to have a command which corresponds to the text structure which is being input, so that the formatter "knows" what it's trying to construct. Thus it will interpret the input in the context of what it is expecting and fit the input into the expected pattern. To keep things synchronized, each input "mode" (context) has some simple conventions which trigger formatting actions at the correct time. These conventions consist of:

- blanks at the start of some lines
- trigger characters embedded in the text for some modes

Definitions

BLANK LINE: a line which has no text data on it on input will normally generate one blank line at that point in the text on output.

COMMAND LINE: lines which start in position one with the master escape character (back slant \ by default, located to the right of the blue RETURN key on 713 terminals), are assumed to be TXTCODE command lines. Such lines are used to give various kinds of instructions to the formatter. In some cases, text is expected on the same line with the command.

DATA LINE: any line which is not a BLANK LINE or a COMMAND LINE is interpreted as a DATA LINE. This type of line is processed by the currently active mode (ASIS, BLOCK, COMMENT, FLOWTAB, ITEM, or TABLE) and is formatted into the corresponding text structure.

NOTE: Under certain circumstances, any or all of the above types of lines may be ignored by the formatter. This happens in two cases: 1) a \STOP command has been issued and a \GO has not been encountered, 2) a \TXTF command was given and a \ENDT has not yet been found. The STOP/GO takes precedence over the TXTF/ENDT command.

2.0 MODE OF OPERATION

Processing

The 6 mode commands and the section number commands are the heart of TXTCODE and with the occasional use of some of the other commands most text forms should be easily handled.

Before describing the commands and modes, an overview of TXTCODE's approach will be given.

At the highest level, \STOP and \GO have absolute control over the input file. All input text following a \STOP, until a \GO occurs will be utterly ignored. Thus sections of the input file may be "turned-off".

The next level down is TXTF/ENDT. Once \TXTF has been called, all input to TXTCODE (which reaches this level) is assumed to be for TXTFORM processing. Hence, the information is simply passed along to TXTFORM until a \ENDT is encountered which returns control to the current mode. TXTFORM would only be called though, if TXTCODE couldn't format a certain section of input text. With the addition of direct TXTFORM commands, there should be very few cases where \TXTF and \ENDT are actually needed. It should be used mainly to encapsulate existing TXTFORM input text.

Going down another level brings the input to TXTCODE proper. The input line is first checked to see if it starts with the escape code (usually \) and, if so, the mode is altered or some feature is set on/off etc. If the line does not start with the escape code, it is processed by the mode which is currently operational (i.e. interpreted according to the present context) with consideration given to any "feature" which is active at that time. (e.g. JUSTification ON in TABLE mode has no effect).

Normally a blank input line is used to generate a blank output line.

Input line length is 80 characters of information.

3.0 SUMMARY OF COMMANDS

3.0 SUMMARY OF COMMANDS

All commands may be entered in either lowercase or uppercase.

Input File Control Commands

```
\STOP
\GO
\TXTF
\ENDT
```

Mode Commands

```
\ASIS
\BLOCK,Jnn,Inn,Hnn                eg. \BLOCK,J1
\COMMENT
\FLOWTABxtt,tt,...tt,ff               \FLOWTAB;1,20
\ITEM,Jnn,Inn,Snn,Gnn,Hnn,Zxx,C=e,0   \ITEM,G2
\TABLExtt,tt,...tt                    \TABLE;10,20,30...
```

Mode Nesting Commands

```
\+[...+]mode command    \+[...+]
\-[...-]mode command    \-[...-]
\*mode command
\NESTIND                \NONESTIND
\EXITNEST,C
\CLEARNEST
```

Designation Text Control

```
\TITLEn=xxx...x          \TITLE=xxxx...x
\DATE=xxx...x
\FOLIO=xxx...x
\TABLCON
\AUTOSEC                 \NOAUTOSEC
```

Text Body Commands

. Section numbers

```
\ n.n....n xxxxx...x   or  \ n xxx...x   or  \ xxx...x
\SETSECnn
```

. Feature Selection De-selection

```
\BOLD    \NOBOLD
```

3.0 SUMMARY OF COMMANDS

```
\CODES
\HEADA   \HEADB    \HEADC
\JUST    \NOJUST
\KEEPn   \NOKEEP
\MARK
\PARA    \PARAnn   \NOPARA
\SEQ     \NOSEQ
\SINGLE  \DOUBLE
\UNDER   \NOUNDER
```

- Text Format Control

```
\MARGINnn          \MARGIN      \MARGIN+nn      \MARGIN-nn
\LENGTHnn          \LENGTH      \LENGTH+nn      \LENGTH-nn
\SKIPnn
\PAGE              \PAGEnnn
\CENTER=xxx...x    \CENTER,B,U=xxx...x
\BLANKnn
```

- Appendices

```
\SETAPP=a-xxx...x
```

- Box Generation Commands

```
\BOX      \BOX,[-]nn,A,nn,B,[-]nn,R,[-]nn
\MODBOX,[-]nn,A,nn,B,[-]nn,R,[-]nn
\BOXLINE
\NOBOX
```

- Escape Character Definition

```
\SETEx
\SETUx
\SETBx
\SETHx
```

Direct TXTFORM Commands

```
\\command
```

4.0 DESCRIPTION OF MODES

One mode of the 6 (ASIS, BLOCK, COMMENT, FLOWTAB, ITEM, or TABLE) is always active even if \STOP or \TXTF has been called. When the \GO or \ENDT is given, the mode resumes processing as if it had not been interrupted.

The initial mode is BLOCK. Transition to another mode is accomplished by simply calling the desired mode with or without any mode parameters. Initially the left margin is set at 10, and the right margin to 62 positions to the right of the left margin (i.e. LENGTH62).

Relevant features such as PARA or JUST are honored if they are applicable to the active mode.

All of these modes process commands issued within the mode and, unless specified otherwise, each mode processes any active trigger characters for underlining, boldfacing, and hard blanking.

4.1 ASIS MODE

\ASIS

Each data line encountered by the ASIS mode is copied as-is to output with no adjustment made to the line structure. The lines start at the margin and will be truncated if longer than 62 characters (default length). See the LENGTH command. A blank line produces a blank line on output.

PARA and JUST have no effect on as-is text. The underlining, boldfacing or hard-blank trigger characters are not recognized as trigger characters in ASIS mode. However the \BOLD command may be used to boldface one or more whole lines of as-is text.

Example:

```
SOME                +-------+
ASIS                | ASIS |
TEXT                +-------+
```

4.0 DESCRIPTION OF MODES
4.1 ASIS MODE

Input:

```
\asis
SOME                +-------+
ASIS                | ASIS  |
TEXT                +-------+
\block
```

4.2 BLOCK MODE

\BLOCK,Jnn,Inn,Hnn

Block mode forms paragraphs from "text BLOCKs" passed to it. Note that a text block of one character is permissible.

A "text block" is a set of one or more lines of data which is started by a line of text with a blank in column one, and ended by either the next line with a blank in column one or by a command which places text in the output at that point.

Each text block is started on a new line and is normally right justified between the left and right margins. A blank line produces a blank line. A series of lines which start with a blank in column one would appear as a list at the left margin.

If the data for a text block starts in column two, the generated paragraph will begin at the left margin. But if x additional blanks are left after the text block start blank, the whole text block will be indented by x positions from the left margin.

If the PARAgraph feature is on, the first line in a text block will be indented by three spaces from the margin. See the PARA command.

J specifies the number of lines (nn) to JUMP at the beginning of each text block. This is in addition to any lines skipped by explicit TXTCODE commands. The default is J0.

I INDENTs the first line of the text block nn positions from the margin. The default is I0.

H HANGs (indents) the second and succeeding lines of the block text nn positions from the margin. The default is HO.

Example:

This format, which is a BLOCK mode paragraph, is generated by the input shown below under Input.

Indentation like this is accomplished with additional blanks on the first line of the text block. The PARAgraph feature is on throughout.

Input:

```
 This format, which is a BLOCK mode paragraph, is
generated by the input shown below under Input.

    Indentation like this is accomplished with
additional blanks on the first line of the text
block.  The PARAgraph feature is on throughout.
```

4.3 COMMENT MODE

\COMMENT

Comment mode ignores all non-command lines. Note that commands are still processed, including text on the command line. This is in contrast to \STOP which ignores everything until a subsequent \GO is received. Comment is not intended to be used to "turn-off" sections of the input file - use STOP/GO.

4.4 FLOWTAB MODE

\FLOWTABxtt,tt,...tt,ff eg. \FLOWTAB;1,20

This mode allows text which is to "flow" between a left start position and the right margin to start on the same line

as text which has been tabbed into position.

The first character x after the command, if non-blank, resets the tab character to the given character. The tab positions follow. (Defaults are ;1,20.) The last position given specifies the start position for the flowing text. Note that text may be tabbed left and right of the text which is to flow between the given start position and the right margin. When tabbing to the right, it is usually necessary to first adjust the right margin using the \LENGTH-nn command. The original right margin may be restored by using the \LENGTH command.

Data is entered in the order of the tabbing sequence used in the command. Tabbing is relative to the left margin, so that a tab set to 1 will cause the associated text to start immediately to the right of the left margin.

Each line starting with a blank forces a new line. An all blank line leaves, as usual, a blank line in the output.

A text block which starts with a blank but which does not have the tab character specified next, is assumed to be flowing text and is placed at the last position specified in the command. Any x additional blanks after the first one, cause the text block as a whole to be indented x positions from the beginning of the flowing text.

NOTE: Once the flowing text is reached, the tab character may be used freely in the flowing text.

See the SETHx command if a "hard blank" (ie. non-compressible) character would be useful in positioning the tabbed information.

The following shows how to set up some FLOWTAB type text:

Example:

4.0 DESCRIPTION OF MODES
4.4 FLOWTAB MODE

COMMAND	PARAMETERS	ACTION
ADD	YES	Add the given files to the tape library and put the comment information on the last file given.
DAYFILE	NO	Leave dayfile for batch job on a permanent file when done. This starts at the flowing text position; the tab character may be used normally here. This is indented by four positions throughout the paragraph.

Input: (\SETU specified at start of document)

```
\keep17
\FLOWTAB;1,15,30
 ;"COMMAND" ;"PARAMETERS" ;"ACTION"

 ;ADD ;YES ;Add the given files to the tape library
and put the comment information on the last file given.

 ;DAYFILE;NO ;Leave dayfile for batch job on a
permanent file when done.

 This starts at the flowing text position; the tab character
may be used normally here.

     This is indented by four positions throughout
the paragraph.
\block
```

4.5 ITEM MODE

\ITEM,Jnn,Inn,Snn,Gnn,Hnn,Zxx,C=e,0 eg. \ITEM

Item mode allows the user to generate a properly indented and aligned list of items where each item is normally preceded by some character string such as: -, *, ., 1., (a), A., or 1)

etc. The above examples, plus any prefix character string, are handled automatically unless ITEM mode is called with its own settings. ITEM mode will set up default values for all parameters that are not supplied on the command.

Notice that the default value of one parameter can be automatically altered by specifying another parameter. E.g. giving G4 alters the default value of H.

(Preceeding the input data for the following six paragraphs with the command "\ITEM" would format that text as it appears below).

J specifies the number of lines (nn) to JUMP at the beginning of each new item. This is in addition to any lines skipped by explicit TXTCODE or TXTFORM commands. The default is J0.

I indicates the prefix characters are to be INDENTed from the margin by an amount nn. The default value for I depends upon the resulting value for S; see the table below.

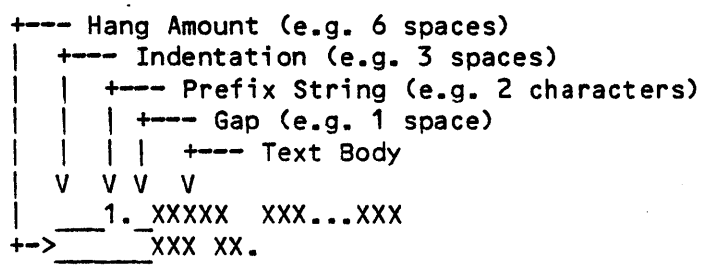
S specifies the SIZE of the prefix string as nn. (nn may be greater than 3). The first actual item encountered will set the SIZE for the prefix string area, unless the S parameter is given on the ITEM command. Subsequent prefix strings will left adjust in the prefix string field if shorter than the first prefix string, while longer prefix strings will be right adjusted in the prefix string field. By specifically giving a large or small S value, the prefix string information can be forced to be left or right justified within its field.

G gives the GAP after the prefix string; nn columns. The default value for G is determined from the table below using the resulting value for S.

H HANGS the text which overflows line one under the first non-blank character following the prefix string or as requested. The default value for H is equal to I+S+G.

Z,C,O These are the parameters controlling automatic item label generation. A full description of these parameters is given in the section on Automatic Label Generation of Itemized Lists which follows shortly.

The following diagram shows the meanings of the various terms used:



Blank lines produce blank lines on output. Each line which starts with a blank forces a new line to be started.

The prefix string ends at the second blank character in the line. Hence, if positions one and two are blank a null prefix string is assumed and text starting in column three would be aligned with the text body of the previous item.

Any x additional blanks in columns 3,4,... cause the following text including overflow portions to be indented from the text body of the previous item by x positions throughout.

Finally, text which starts in position one of the line and which follows a blank line, will be treated as a non-item and aligned at the left margin.

The following table shows how ITEM will set up its I, S, G and H values for various sizes of prefix strings when the first item is encountered:

SIZE	Inn	Snn	Gnn	Hnn (I+S+G)
1	3	1	2	6
2	3	2	1	6
3	2	3	1	6
4	1	4	1	6
5+	0	5+	2	7+

Item Parameter Defaults Table

Example:

The input shown below would produce the format shown above, where the J, I, S, G, H, Z, C, and O parameters are explained:

Input:

\item
J specifies the number of lines (nn) to JUMP at the beginning of each new item. This is in addition to any lines skipped by explicit TXTCODE or TXTFORM commands. The default is J0.

I indicates the prefix characters are to be INDENTed from the margin by an amount nn. The default value for I depends upon the resulting value for S; see the table below.

S specifies the SIZE of the prefix string as nn. (nn may be greater than 3). The first actual item encountered will set the SIZE for the prefix string area, unless the S parameter is given on the ITEM command. Subsequent prefix strings will left adjust in the prefix string field if shorter than the first prefix string, while longer prefix strings will be right adjusted in the prefix string field. By specifically giving a large or small S value, the prefix string information can be forced to be left or right justified within its field.

G gives the GAP after the prefix string; nn columns. The default value for G is determined from the table below using the resulting value for S.

H HANGS the text which overflows line one under the first non-blank character following the prefix string or as requested. The default value for H is equal to I+S+G.

Z,C,0 These are the parameters controlling automatic item label generation. A full description of these parameters is given in the section on Automatic Label Generation of Itemized Lists which follows shortly.
\block

Automatic Label Generation on Itemized Lists

The following parameters exist on the item command to implement the automatic labelling of items:

,Z[x..x][a..a][=[b..b][c][d..d]] ,C=e ,0

Each of the parameters are optional and have the following meaning:

Z = L or l - auto label with letters: A, B, C,.. or a, b, c,..

N or n - auto label with numbers: 1, 2, 3,..
R or r - auto label with roman: I, II, III, IV,.. or
i, ii, iii, iv,..

In the absence of the x..x portion of the command, the case of the "Z" determines the case of the generated labels. Otherwise, x..x is used to specify both a non-default starting value and the case.

x..x a starting value for labelling, if it is present and is compatible with "Z"'s type. Otherwise it is treated as a..a text.

a..a suffix text to follow each generated label when no "=" option is used in the parameter.

"=" used for more complex types of auto generated label text where b..b text is to be specified.

b..b text which is to precede the auto generated string. It might be a "(" for an (a), (b) type sequence.

c a point of substitution single character for the auto generated sequences, which is "." by default. It may be redefined using the ",C=e" parameter, where "C" stands for "character" and "e" is the new editing substitution character to assume both in the command and in the following text data prefix strings.

d..d suffix text to follow each auto generated string. It might be a ")." such as for an (A)., (B)., type sequence.

C=e e is a single character which is to define the point of substitution position for the auto generated strings. This is for both the command itself and the text data which follows the command.

O Override parameter. When this parameter is specified on the command, the point of substitution character is not sought for in the prefix string region of the text data. Any non-null character string for a prefix region is completely removed and replaced with the auto generated sequence. It may or may not contain the actual substitution character.

Functioning

The basic process that occurs is as follows:

The label type, starting value, and case is determined from the "Z" and any x..x. This material is the auto generated string and will be incremented for each substitution that occurs. If there is any a..a material, it is used to set up the area that supplies the suffix for each auto generated string. If there is an "=" in the command parameter, everything following the "=" (until the end of the parameter - "," or " " reached) until the point of substitution is found, becomes a prefix to the auto generated string.

When the substitution character is found, any text following this character until the end of the parameter is used to set up the suffix area material for the auto generated string.

At this point the command has been processed and the prefix, auto generated start string, and the suffix have been set up. These three regions as a whole, packed together, become the "item substitution text". This text, of course, changes each time it is used since it is being incremented for each use.

Text Data Substitution

Now, each time a non-null prefix string is encountered in the text data, the prefix string is scanned for the point of substitution character (usually "."), and the item substitution text is used to replace the point of substitution character. At this point, the text data prefix string has been constructed, and it is now treated according to the settings of the parameters I, S, G and H, whether set specifically or by the item command.

So, it is possible to specify standard preceding and following material for the auto generated string, such as parentheses, but still specify special text for a particular item that is unique to that item - an "*" for example, for a footnote. I.e.

(a). xxxx
*(b). yyy
(c). zzzz

```
TO DO:    1.      A.      (a)       i.       A.
          2.      B.      (b)       ii.      *B.
          3.      C.      (c)       iii.      C.
ISSUE:
        item,n.   item,L.  item,l=(.)  item,r.   item,L.
        ^.^----   ^.^----  ^.^------   ^.^-----  ^.^-----
        -------   -------  --------   ------    ------
        ^.^----   ^.^----  ^.^------   ^.^-----  ^*.^----
        -----     -----    ------     ------    -------
```

The "^" is used to represent a blank character.

4.6 TABLE MODE

\TABLExtt,tt,...tt \TABLE;10,20,30...

As the name implies, TABLE mode is for use in constructing tables or columns of tabbed information.

Unless redefined, the tab character x is ";" and the tabs are set to 10, 20, 30, 40, etc. If the tab character is not to be redefined, leave the position immediately after the mode name blank. The tabs are relative to the margin so that a tab set to 1 places text in the position right after the left margin.

Whenever a data line begins with a blank character, a new line is forced and the tabbing pointer is reset to tab number 1. (Data for a single line in a table may be input on several lines.) There are a maximum of 15 tabbing positions available.

An all blank line produces a blank line on output.

PARA and JUST are not applicable to this mode.

See the SETHx command if a "hard blank" (i.e. non-compressible) blank character would help in formatting the table.

Example: (\SETU specified at start of document.)

4.0 DESCRIPTION OF MODES
4.6 TABLE MODE

SIZE	Inn	Snn	Gnn	Hnn (I+S+G)
1	3	1	2	6
2	3	2	1	6
3	2	3	1	6
4	1	4	1	6
5+	0	5+	2	7+

Input:

```
\keep7
\table;7,17,26,35,44
 ;~SIZE~  ;~Inn~  ;~Snn~  ;~Gnn~  ;~Hnn~ (I+S+G)

\table;8,18,27,36,45
 ;1 ;3 ;1 ;2 ;6
 ;2 ;3 ;2 ;1 ;6
 ;3 ;2 ;3 ;1 ;6
 ;4 ;1 ;4 ;1 ;6
 ;5+;0 ;5+;2 ;7+
\block
```

5.0 MODE NESTING COMMANDS

This section describes the mode nesting facility of TXTCODE.

For example:

```
\+ITEM
 1. Level one in the nest.

\+ITEM,IO
 (a) Level two in the nest.

\-
 2. Level one in the nest again.
\-
```

The above example looks like this when formatted:

1. Level one in the nest.

   (a) Level two in the nest.

2. Level one in the nest again.

Mode nesting is a facility which allows an active mode to be temporarily interrupted so that another style of text may be processed, before returning to the interrupted mode and carrying on as though it had not been disturbed. The second mode can also be interrupted itself and so on. In addition, there is normally an automatic margin shift which causes the interrupting mode to be indented under the interrupted mode so that it aligns with the visual margin of the interrupted mode.

This provides a simple and powerful way of formatting complex text structures.

\+mode *mode \-mode

Nesting is entered by the first occurrence of "\+mode" and, until the next "\mode" command, which signals the exit, mode commands of the form "\+...", "*..." and "\-..." control movement within the nest. Exit is also accomplished when a

5.0 MODE NESTING COMMANDS

\EXITNEST command is given or when a "\-mode" command causes the nest to decrease to the zero level. ("\-" may also do this.)

The nest has a maximum number of active 'parentheses' of 10. Nesting beyond this maximum level is processed using the ASIS mode. Dropping back below the upper limit causes the specified modes at each level to resume.

The mode to use for processing data at each level in the nest is set by the mode given on the mode command that most recently set a mode for that level. Each of "\+mode", "*mode" and "\-mode" set the mode at the level that they are causing a shift to in the nest. It is possible to move to a new level in the nest without specifying a mode, by giving "\+", "*" or "\-" with no mode. In this case, the mode most recently given for that level is used. Any level which never had an initial mode specified, uses the ASIS mode as a default.

Multiple levels may be shifted, if desired, by giving "\++..." or "\--..." with or without a mode name.

Moving to a lower level in the nest does not wipe out the modes specified for higher nest levels. Even exiting from the nest as a whole will not cause the nest mode data to be destroyed. It will remain until it is overwritten by a new mode at the appropriate level or until a "\CLEARNEST" command is given.

This allows the user to take advantage of repetitive data structures to simplify the mode calling process and also help ensure consistency of formats. Note that for ITEM mode especially, even the last used automatically generated prefix string is remembered and resumed when the correct level is again reached.

\NESTIND

Another facility is provided when automatic indentation is on during nesting. This facility is on by default and is controlled using \NESTIND (nest indent) or \NONESTIND at any location in the document. With nest indent ON, whenever a "\+..." command is found, TXTCODE will cause the new mode to be right shifted to the "visual" margin of the previous mode. Actually, only the FLOWTAB and ITEM modes have visual margins which are usually shifted from the relative margin position. (BLOCK mode may also have a shift if the Hnn parameter is

used.) It is the relative margin that is being moved to the position of the "visual" margin of the previous mode.

When coming back out of the nest, the nest indent indicator is ignored. Shifting back is done if shifting forward was done at this level.

Level Shifting

The following is an example of some nested modes with an explanation of what is happening:

```
\block
\+item
\+FLOWTAB
\-
\+
\-
\+
\*asis
\-
\-
```

Initially, BLOCK mode is active. Then a nest is entered with the ITEM mode request. Then some FLOWTAB material is given under several particular items - all with the same FLOWTAB formats. After the last FLOWTAB section is some ASIS text which should have the same margin as the FLOWTAB mode had when it was called. Then ITEM mode is picked up and finished and finally the original mode (block) is resumed, which was active before the nest was entered.

"*mode" replaces the current mode at a particular level with a new mode. The new mode is not indented because it is at the same level as the one already there.

Restored Modes

It should be noted that when a mode is restored, it is the mode that is restored and not all the associated conditions that may have been active during that mode.

All such features remain under the users direct control alone. For example, if a trigger character for underlining had been set but was later cleared, reentering the mode during nesting would not reestablish the underlining trigger character.

5.0 MODE NESTING COMMANDS

In addition, margin control is also still under the user's direct control. The changes in the relative margin are made during the shifting process in going from one mode to another in the nest (when nest indent is active) in order to fashion the indentation structure, but the user may give extra margin adjustments of his own. He should be careful to move the margin back when the structure is completed, and before he changes levels preferably. Otherwise, arbitrary margin changes will fracture the nest structure as a whole at that point.

Visual Margin

The visual margin is defined as the relative margin for ASIS, COMMENT and TABLE modes. For FLOWTAB mode, the visual margin is where the flowing text starts. For ITEM or BLOCK mode, the visual margin is where the body of the text falls when line overflow occurs. (Note that the default visual margin for BLOCK mode is the normal margin.)

Mode Switching

To go into the nest another level give:

```
\+mode
```

To change modes at the same level give:

```
\*mode
```

To drop out of the nest one level give:

```
\-mode
```

These three commands, containing the "+", "*" or "-" in front of the mode, control the movement within the nest, entrance into a nest (first "\+..."), and exit from the nest (a "\-" at level one). Any mode command without the plus, asterisk or dash is a regular non-nest mode command and will terminate a nest if one is active.

Explanation

Each time a deeper mode is entered, the visual margin value is saved for the previous mode - that is, the amount to shift the relative margin by is saved, if nest indentation is active. Next the relative margin value is adjusted if necessary. Then the new mode is set up and the actual mode data is processed.

5.0 MODE NESTING COMMANDS

Nest Clearing

The entire mode nest may be cleared by using the \CLEARNEST (when outside a nest) or \EXITNEST,C commands. This destroys all memory of modes which have been defined in the nest structure.

\EXITNEST,C

This command causes an immediate exit from any level of a nest. The most recent \mode command governs the active mode on exit. If the "C" parameter is specified, the whole nest is completely cleared of mode information.

\CLEARNEST

The CLEARNEST command clears all nest levels beyond the current level. If nesting is inactive, it clears the whole nest - levels 1 to 10; if nesting is active, it clears levels n+1 through level 10. The current and lower levels are not affected and nesting is not disturbed.

6.0 DESCRIPTION OF COMMANDS

6.1 DESIGNATION TEXT COMMANDS

These commands apply to the title block (top of page title information), folio line (page footing information), and any supplemental lists generated such as a table of contents. (See \TABLCON command).

\TITLE

\TITLEn=xxxx...x or \TITLE=xxxx...x

The xxxx...x is the information which is to appear in the title position n (n=1,2...6) of subsequent pages. Up to six title lines may be in effect. Unused title lines will be squeezed out of the title block. The default title line is n=1, which appears between the page number and date lines. Other title lines follow the date title line.

\DATE

\DATE=xxxx...x

The current date (in YY/MM/DD format) will be replaced by xxxx...x at the upper right hand position of subsequent pages. It will appear on the current page too, if no text other than page header text has been given yet. E.g. \DATE=13 JUL 79

\FOLIO

\FOLIO=xxxx...x

At the bottom of the page on the folio line, xxxx...x will appear right justified on the current and following pages.

\TABLCON

If a table of contents is desired, this command should be issued once at the start of the input file. It should be given before any section numbers (\ n.n...n xxxxx...x) are given.

\AUTOSEC and \NOAUTOSEC

AUTOSEC turns on automatic section numbering. The level number is given by n on a section number command line. Any

n.n....n forms have their level calculated from the n.n...n. That is, n.0 is level 1; n.n is level 2; n.n.n is level 3; etc. The first level 1 becomes 1.0; the second level 1 becomes 2.0 etc., while the first level 2 becomes 1.1 and so on.

Automatic section causes page numbers to change to the n-n format; it also causes a section header box to appear at the top of each page. See Section Numbers for additional detail.

6.2 BODY TEXT COMMANDS

These commands control the text which is in the middle of the page - between the title block information and the folio line.

6.2.1 SECTION NUMBERS

\ n.n...n xxxxx...x or \ n xxx...x or \ xxx...x

This command establishes a section number and title data in the text and in the table of contents. (See the \TABLCON command for generating a table of contents). Normally the first format is used with \NOAUTOSEC and the second form with \AUTOSEC. However, the first format can be used by \AUTOSEC and the second format will produce titles with no section numbers if used with \NOAUTOSEC.

Paging (n.0 levels), spacing, underlining, and capitalization are produced automatically and are governed by the level being processed. Standard TXTFORM format for section numbering is used. See the Heading Level Selection Table.

Note: the space after the back slant (\) and also after the section number. One space is sufficient.

In the second form, n is a level number from 1 to 7, or 11 to 17. (See \AUTOSEC)

6.0 DESCRIPTION OF COMMANDS
6.2.1 SECTION NUMBERS

In the third form, no section number appears and the text is underlined. It is equivalent to \ 14 xxx...x. (Note: the third format should not be used if the first word contains any periods "."; use format 2 with n=14. The same applies if the first part of the title text is numeric.)

NB: Individual section number lines which are too long may be omitted from a table of contents by TXTFORM.

HEADING LEVEL SELECTION TABLE

	NEW PAGE	ALL CAPS	LINES SPACED BEFORE	LINES SPACED AFTER	UNDER LINED
H1(H11) SECTION \ 1 xxx...x \ 11 xxx...x \ n.0 xxx...x LEVEL 1	X	X		5	X
H2(H12) SECTION \ 2 xxx...x \ 12 xxx...x \ n.n xxx...x LEVEL 2					
\HEADA		X	2	1	X
\HEADB		X	3	2	X
\HEADC		X	2	1	X
H3(H13) SECTION \ 3 xxx...x \ 13 xxx...x \ n.n.n xxx...x LEVEL 3					
\HEADA		X	2	1	
\HEADB		X	3	2	
\HEADC		X	2	1	
H4(H14) SECTION \ 4 xxx...x \ 14 xxx...x \ n.n.n.n xxx...x \ xxx...x LEVEL 4					
\HEADA			1	1	X
\HEADB			3	2	X
\HEADC			2	1	X

6.0 DESCRIPTION OF COMMANDS
6.2.1 SECTION NUMBERS

	NEW PAGE	ALL CAPS	LINES SPACED BEFORE	LINES SPACED AFTER	UNDER LINED
H5(H15) SECTION					
\ 5 xxx...x					
\ 15 xxx...x					
\ n.n.n.n.n xxx...x					
LEVEL 5					
\HEADA		X	1		
\HEADB		X	1		X
\HEADC		X	1		X
H6(H16) SECTION					
\ 6 xxx...x					
\ 16 xxx...x					
\ n.n.n.n.n.n xxx...x					
LEVEL 6					
\HEADA			1		X
\HEADB					X
\HEADC		X			
H7(H17) SECTION					
\ 7 xxx...x					
\ 17 xxx...x					
\ n.n.n.n.n.n.n xxx...x					
LEVEL 7\HEADC			1		X

If AUTOSEC is off (default), the first form shown (\ n xxx..) is converted to the second form shown (\ 1n xxx..) and treated accordingly.

If AUTOSEC is on, (\AUTOSEC issued), page numbering changes from the simple sequential format to the n-n format. Also, a 4 line header box will be produced on each page which will contain the currently active major section (n.0 level) and the currently active heading level being processed, of whatever level. AUTOSEC doesn't produce numbering for (\ 1n xxx..) type headers, but it does replace any n.n...n forms, which effectively allows renumbering of old section numbers.

NOTE: that AUTOSEC effects do not occur until the first header command line (\ n xxx...x) has been encountered.

Also note that the form (\ xxx...x) is treated as the following form in all cases: (\ 14 xxx...x).

6.2.2 FEATURE SELECTION/DE-SELECTION

\PARA

\PARA \PARAnn \NOPARA

With the PARAgraph feature selected, BLOCK mode paragraphs which begin at the left margin have their first line indented by the number of spaces specified for the \PARAnn command or by three spaces for the \PARA command.

\UNDER and \NOUNDER

Specifying UNDERlining will cause the following lines of text to be underlined until a NOUNDERline command is given or until the mode is changed or a section number is encountered. This command is oriented toward underlining for which it is not convenient to use a trigger character for control. (See the SETUx command).

\JUST and \NOJUST

Right JUSTification occurs as a default. Text which is being built up so that it overflows to the next line when the right margin is met is normally aligned with the right margin by adding extra spaces between words. This feature has no effect, even if selected, if it is not appropriate to the currently active mode. TABLE and ASIS mode ignore this feature. Note that JUSTification only allows backing up in the line for 30 characters. Therefore, if a blank, -, or / is not found, a 'LINE OVERFLOW' message will be issued since a place could not be found to break for JUSTification.

\BOLD and \NOBOLD

The BOLDface command instructs the formatter to print each character which follows in such a way that the characters appear darker and thicker. BOLDfacing stops when the NOBOLDfacing command is given or when the mode is changed or when a section number is encountered. The command is oriented towards BOLDfacing text for which a trigger character in the text itself is not appropriate. (See the SETBx command).

\SEQ and \NOSEQ

If it is desired to have each output line in the document followed by the number of the line, issuing the SEQ command will initiate this process. (Default is NOSEQuencing). SEQuencing numbers will appear right justified on the fifth

character position to the right of the right margin or the length as established by the TXTFORM "FORMAT" command (default is 75), whichever is greater - absolute column 80 by default. SEQuence numbers start at 1 on each new page and appear only beside text in the body of the document. The feature may be turned on or off at any point in the document.

NOTE: The TXTFORM FORMAT command is used to give the margin values for heading and folio lines and to specify the total page depth. It may only be given once and must be before any text is input. Using the direct TXTFORM command capability, it may be specified as follows:

\\FORMATaa,bb,nn

where aa is the left margin (default is 7), bb is the absolute line length (default is 75), and nn is the total page depth (default is 60). These margin values do not apply to the main body of the text.

\KEEPnn and \NOKEEP

Sometimes material on different lines must be kept on the same page to be readable. To make sure that the next nn lines of output appear on the same page specify KEEPnn. NOKEEP serves only to mark the end of the kept text and is not actually necessary.

\SINGLE and \DOUBLE

It is often desirable to allow different standard line spacing on documents. For example, double spacing is quite useful for draft documents to which many changes are usually made. SINGLE and DOUBLE select single and double spacing respectively for the formatted output text.

\HEADA, \HEADB, and \HEADC

These commands allow selection of the TXTFORM heading styles A, B, and C respectively. See the heading level selection table for a description of the effect of the different heading styles.

\CODES

This is used to produce a TXTFORM command statistics and error summary list at the end of the formatted document.

\MARK

The MARK command is used to set a margin note in the line being formatted. An asterisk (*) is inserted at the current position in the line and a pointer (<) is placed in the right margin.

6.2.3 TEXT FORMAT CONTROL COMMANDS

\MARGIN

\MARGINnn \MARGIN \MARGIN+nn \MARGIN-nn

This command determines where text should normally start at the left hand side of the page. MARGINnn redefines the BASE MARGIN (default is 10) to the new value nn. Normally adjustments are made to the margin by specifying the change relative to the old margin instead of specifying the absolute margin each time. The +nn and -nn form is used to accomplish this. Note that a \MARGIN+2 followed by \MARGIN+3 command is equivalent to a \MARGIN+5 command.

At any time, the BASE MARGIN, the one defined by the last MARGINnn command, or MARGIN10 by default, may be returned to by asking for MARGIN with no value. In this way the margin may be temporarily redefined and then returned to the old value without remembering that value.

The ABSOLUTE margin is the extreme lefthand side of the page; the BASE margin is at 10 unless redefined using \MARGINnn; and the RELATIVE margin is initially equal to the BASE margin. The RELATIVE margin is moved by \MARGIN+nn and \MARGIN-nn commands as well as by mode nesting commands and may be reset to the BASE margin by \MARGIN.

\LENGTH

\LENGTHnn \LENGTH \LENGTH+nn \LENGTH-nn

This command specifies the LENGTH of the text line as measured from the BASE MARGIN (see above). The default value is 62. Hence, an ASIS line of up to 62 characters can be placed on each output line using the default value. (\MARGIN-nn and \MARGIN+nn will lengthen or shorten the effective line length.) Adjustments may be made relative to the old length in a similar manner to the MARGIN command above.

\SKIP

\SKIPnn

To write nn blank lines to output at any time, use the SKIPnn command. The nn defaults to 1. Note that if paging occurs in the process, the command does not leave the rest of the blank lines at the top of the next page. If an actual amount of blank paper is desired, say 2 inches, issue a KEEP12 (2 inches times 6 lines per inch = 12 lines) then a SKIP12.

\PAGE

\PAGE or \PAGEnnn

At any time, a PAGE eject may be forced using this command. Two consecutive page ejects do not, however, leave a blank page. (Entering the command BLANK between the two PAGE ejects will). If it is desired to cause the next PAGE number to be reset to a value nnn, include the value immediately after the command.

\BLANK

\BLANK or \BLANKnn

If this command is given, nnn blanks (default is 1) will be inserted in the text stream at the current position. See the SETHx command for generating "hard blanks" using a specially designated character instead of the BLANK command.

\CENTER

\CENTER=xxxx...x or \CENTER,B,U=xxx...x

The xxxx...x text will be CENTERed on a new line between the left and right margins. Consecutive embedded blanks are compressed to one blank. B and U parameters are for boldfacing and underlining xxx...x. Special trigger characters in the xxxx...x text will be recognized and processed, if set, if the first form of the command is used. See the SETU, SETB and SETH commands.

6.2.4 APPENDICES

\SETAPP

\SETAPP=a-xxx...x

This command is used to define appendix text. The character value of "a" defines the appendix identifier and "xxx...x" is the appendix title. This information appears in the table of contents (see the TABLCON command) and the user should set up his own appendix title page as desired as well as the body of the appendix text.

The appendix identifier "a" precedes both the section number and the page number wherever they appear. One method of setting up an appendix follows:

```
\SETSEC1                    - resets section numbers to 1
\SETAPP=A-APPENDIX A        - sets appendix character and title
\PAGE1                      - sets up title page as page 1
\BLANK
\SKIP20
\CENTER,B=* Appendix A *

\CENTER=Special Conditions
\PAGE                       - starts body of appendix text
\ 1 Introduction
  .
  .
  .
```

6.2.5 BOX GENERATION COMMANDS

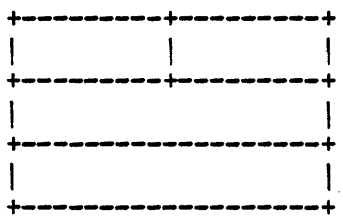
The following commands are used to draw boxes but the vertical spacing in boxes is controlled by commands or text given while a box is active. Usually TABLE or FLOWTAB mode is used to supply the text for a box.

\BOX

\BOX or \BOX,[-]nn,A,nn,B,[-]nn,R,[-]nn

The BOX command is used to initiate the drawing of vertical

lines which are all joined together by a horizontal line at the top. For example, the command \BOX,10,20,30 would be used to start the following box:



The two outside columns are at columns 10 and 30 with respect to the relative margin, while the center line (vertically) is at column 20. By default, the column positions are with respect to the relative margin. At any point in the command, the reference point may be changed to the BASE margin with the "B" parameter, to the ABSOLUTE margin using "A", or back to the RELATIVE margin with the "R" parameter. (See the MARGIN command for an explanation of margins.) The A, B, and R parameters may be used several times in the command if desired.

For the BASE and RELATIVE margins, it is possible to specify a column position to the left of the margin by preceding the column number with a minus sign "-". Therefore "-1" and "1" specify vertical lines which are adjacent to each other surrounding the margin ("0" is equivalent to "-1").

Up to 17 vertical lines may be in effect at once.

If \BOX is given with no parameters, the settings in effect for the previous box are used.

\MODBOX

\MODBOX,[-]nn,A,nn,B,[-]nn,R,[-]nn

This command is used to modify a box in progress. The parameters are the same as for the BOX command. For each column given, there are two possible actions. If that column already has a vertical line being drawn, the line is turned off. If it does not have a vertical line in effect, one is started. In addition, MODBOX joins the resulting outside verticals with a horizontal line. The above box had its center vertical terminated by the following command:

\MODBOX,20

\BOXLINE

This command draws a horizontal line between the inside and outside vertical lines of the box. The third horizontal in the box above was drawn this way.

\NOBOX

NOBOX is used to end a box currently in effect, drawing the final horizontal line terminating the box.

Example:

A	ABSOLUTE	This parameter causes the following values to refer to the ABSOLUTE margin.
B	BASE	This parameter causes the following values to refer to the BASE margin.
R	RELATIVE	Finally, this parameter causes the following values to refer to the RELATIVE margin.

Input:

```
\KEEP12
\BOX,-1,7,21,62
\FLOWTAB:3,10,24
\LENGTH-3
 ;A ;ABSOLUTE ;This parameter causes the following values to
refer to the ABSOLUTE margin.
\BOXLINE
 ;B ;BASE ;This parameter causes the following values to
refer to the BASE margin.
\BOXLINE
 ;R ;RELATIVE ;Finally, this parameter causes the following
values to refer to the RELATIVE margin.
\NOBOX
\LENGTH
```

6.2.6 ESCAPE CHARACTER DEFINITION

\SETEx

To reSET the master Escape character for commands to another character x issue this command. (Default is \)

\SETUx

To turn on a trigger character x to control the starting and stopping of Underlining give this command. The feature is turned off by giving a blank character for the x. (Default condition is no trigger character set). Also, note that the effect of a mode change or the occurrence of a section number command will turn off Underlining just as the second occurrence of the trigger character would. But the feature itself remains set. A suggested character to use for triggering underlining is _ or ~.

NOTE: If the underlining and boldfacing trigger characters are set to the same character at the same time, both functions will be controlled by the one trigger character.

\SETBx

This feature is exactly analogous to underlining (see above) except that it controls Boldfacing. A suggested character to use to trigger this feature on and off is @.

\SETHx

From time to time it is necessary to use special blanks in the text called "hard blanks", which will not be compacted into one (1) blank or expanded into more than one blank. The hard blank is also useful in forcing tabbed data into proper alignment if the left hand side is not straight. The BLANK command is not always convenient in such cases. This command, SETHx, allows the user to specify a character which is to be considered to be a hard blank until the feature is turned off. Each subsequent occurrence of the character forces a single blank to be shown in its place.

A suggested character to use is the circumflex (^) or the left apostrophe.

7.0 DIRECT TXTFORM COMMANDS

7.0 DIRECT TXTFORM COMMANDS

Any TXTFORM command may be given directly by prefixing that command by two master Escape characters rather than one. The preprocessor strips off the first master Escape character and passes the command directly to TXTFORM. No command validity checking is performed.

e.g. \\V10,20,30

* * *