

Norsk Data



PED User's Guide

ND-60.121.04



PED User's Guide

ND-60.121.04

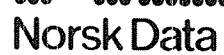
NOTICE

The information in this document is subject to change without notice. Norsk Data A.S. assumes no responsibility for any errors that may appear in this document. Norsk Data A.S. assumes no responsibility for the use or reliability of its software on equipment that is not furnished or supported by Norsk Data A.S.

The information described in this document is protected by copyright. It may not be photocopied, reproduced or translated without the prior consent of Norsk Data A.S.

Copyright © 1983 by Norsk Data A.S.

PED User's Guide Publ.No. ND-60.121.04 June 1983	
--	--



Norsk Data A.S
Graphic Center
P.O.Box 25, Bogerud
0621 Oslo 6, Norway

Manuals can be updated in two ways, new versions and revisions. New versions consist of a complete new manual which replaces the old manual. New versions incorporate all revisions since the previous version. Revisions consist of one or more single pages to be merged into the manual by the user, each revised page being listed on the new printing record sent out with the revision. The old printing record should be replaced by the new one.

New versions and revisions are announced in the ND Bulletin and can be ordered as described below.

The reader's comments form at the back of this manual can be used both to report errors in the manual and to give an evaluation of the manual. Both detailed and general comments are welcome.

These forms and comments should be sent to:

Documentation Department
Norsk Data A.S
P.O. Box 25, Bogerud
0621 Oslo 6, Norway

Requests for documentation should be sent to the local ND office or (in Norway) to:

Graphic Center
Norsk Data A.S
P.O. Box 25, Bogerud
0621 Oslo 6, Norway

Preface:

THE PRODUCT

This manual describes the second main version of the ND program editor

PED-ENG-J

which is both a fullscreen and a line based program editor incorporating extensive editing functions. Files are entered and edited from a video display terminal or a teletype terminal. On terminals with cursor addressing, the editor will normally be run in page mode. In line mode, PED functions like an extended version of QED. Files can be both program and data files. The Editor will not be able to read NOTIS-WP format (16 bits) files, and the reader is warned that none of the compilers of ND will accept 16 bit format files. Files generated by PED are accepted by all ND compilers.

THE READER

The information contained in this manual is primarily intended for users who require a detailed and formal explanation of the product, as well as an account of the features and facilities available to them. As the present product is quite different from previous versions, it will also be useful to those who have used the previous PED editor (E-version and earlier).

PREREQUISITE KNOWLEDGE

Although basic familiarity with the principles of electronic word processing, and in particular with the earlier versions of ND Program Editor (PED), is an advantage, the user needs no knowledge of computers or programming in order to use PED. It is also useful to be familiar with the ND operating system, SINTRAN III.

THE MANUAL

The present document describes the ND Program Editor PED, and has been written as a combined User Guide and Reference Manual. Appendices A, B, C and D are devoted to error messages, the editor menus, the help structure, and an ASCII conversion table.

RELATED MANUALS

SINTRAN III Utilities Manual ND-60.151 contains information about the Perform system.

SINTRAN III Introduction ND-60.125 and

SINTRAN III Timesharing/Batch Guide ND-60.132 provide useful information about the operating system.

NOTIS-TF Reference Manual - Text Formatter ND-63.007 covers the directives in the Text Formatter.

TOV 2200/9 User Guide (ND), part no. 398332, publication no. 5347, describes the NOTIS terminal.

FACIT 4420/ND User's Guide ND-12.024.01 describes the Facit NOTIS terminal.

TABLE OF CONTENTS

<u>Section</u>	<u>Page</u>
1 AN INTRODUCTION TO PED	1
1.1 Some comments on notation conventions	3
1.2 The Screen	4
1.2.1 The text area	4
1.2.2 The status line	6
1.2.3 The command line	6
1.2.4 The second command line	6
1.2.5 The tabulator line	7
1.2.6 The WAIT message	8
1.2.7 The LED display on the terminal	8
1.3 The command types	9
1.3.1 The HOME commands	9
1.3.2 The function key commands	9
1.3.3 Control key commands	10
1.4 How to start the program	10
1.5 To start a new program file	10
1.6 The PED work area	11
1.7 Area	11
1.7.1 Expressions	12
2 FILE HANDLING COMMANDS	13
2.1 Regions	15
2.2 Commands to PED in a file	16
2.2.1 Tabulator - TAB	16
2.2.2 Borders - {}	17
2.3 Commands to PED from the SINTRAN command line	18
2.4 Read file - R	19
2.5 Update file - U	23
2.6 Recover buffer - PED @	23
2.7 Read file to "number of pages" - R+	24
2.8 Write file - W	25
2.9 Append file - A	29
2.10 Exit from PED - E, EXIT, FUNC #	29
2.11 Activate PERFORM - FUNC \$, SHIFT + EXIT, C	29
2.12 Activate the Text Formatter NOTIS-TF - SHIFT + PRINT, FUNC " J	30
3 CHANGING PED PARAMETERS	31
3.1 Set new tabulator stops - T	33
3.2 Set/reset tabulator positions here - the TAB key, FUNC TAB, FUNC <T>, FUNC <I>	34
3.3 Set/reset secondary tabulator positions here - SHIFT + the TAB key, FUNC BTAB, FUNC <Y>	34

Section	Page
3.4 Set new editor borders - B	35
3.4.1 Set left editor border in this position - (..., FUNC (35
3.4.2 Set right editor border in this position - SHIFT + ...), FUNC)	36
3.5 Set the editor in line mode (QED compatible) - #	36
3.6 Change the terminal type to <type> - !	36
3.7 Set/reset expand mode - EXP, <E>	37
3.8 Set/reset append mode - INS, 	38
3.9 Set/reset alternative delete mode - FUNC Δ , FUNC DEL	38
3.10 Display menu no. 1 - 1	39
3.11 Display menu no. 2 - 2	39
 4 PED UTILITIES	 41
4.1 Evaluate a numeric expression - V	43
4.2 Verify current cursor position - <V>	47
4.3 Execute a SINTRAN command - @	49
4.4 Set/Reset LF/CRLF at end of line - FUNC LF	50
4.5 Display regions - X	50
4.6 Re-display the whole screen - space bar, FUNC @	51
4.7 The HELP function -- H, ?, HELP, FUNC ?	51
4.8 Exit from PED - E, EXIT, FUNC #	52
4.9 Activate PERFORM - FUNC \$, SHIFT + EXIT, C	53
4.10 Activate the Text Formatter NOTIS-TF - SHIFT + PRINT, FUNC ", J	54
 5 KEYS USED TO NAVIGATE	 57
5.1 Move the cursor one line up - \uparrow	59
5.2 Move the cursor one position to the left - \leftarrow	59
5.3 Move the cursor to and from HOME position - HOME key or slanted arrow	59
5.4 Move the cursor one position to the right - \rightarrow	60
5.5 Move the cursor one line down - \downarrow	60
5.6 Move the cursor down five lines - LF	60
5.7 Move the cursor to the left margin on the next line - \leftarrow , CR, RETURN	61
5.8 Move to the first character on the next line - FUNC \leftarrow , FUNC <M>	61
5.9 Move the cursor to the previous tabulator stop - \leftarrow , <U>, <Y>	62
5.10 Move the cursor to the next tabulator stop - \rightarrow , <T>, <I>	62
5.11 Move the cursor to the previous secondary tabulator stop - SHIFT + \leftarrow , FUNC V	63
5.12 Move the cursor to the next secondary tabulator stop - SHIFT + \rightarrow , FUNC K	63
5.13 Search for <string> - SHIFT + F7, FUNC <G>, G	63
5.14 Continue searching - F7, FUNC \, <G>	65
5.15 Scroll display window up - SCROLL-UP, FUNC \uparrow	65
5.16 Scroll display window right - SHIFT + SCROLL-DOWN, FUNC \rightarrow	66

Section	Page
5.17 Scroll display window left - SHIFT + SCROLL-UP, FUNC ← . . .	66
5.18 Scroll display window down - SCROLL-DOWN, FUNC ↓	66
5.19 Move to previous area, menu etc. - <==, FUNC B	67
5.20 Move to next area, menu, etc. - ==>, FUNC N	68
5.21 Move the cursor to the beginning of this line - SHIFT + <==, <R><R>	69
5.22 Move the cursor to the end of this line - SHIFT + ==>, <F><F>	69
5.23 Display the first text window - F	69
5.24 Display the last text window - L or \$	69
5.25 Display the next text window - N	70
5.26 Display the previous text window - P	70
5.27 Advance the displayed text by 5 lines - plus	70
5.28 Rewind the displayed text by 5 lines - minus	71
5.29 Move display to <position> - M	71
5.30 Enter the work area at <position> for editing - "	73
5.31 Move to another window - FUNC 1, FUNC 2, FUNC 3	73
 6 KEYS USED TO MARK TEXT AREAS	 75
6.1 Mark top left/bottom right corner of a box area - MARK, FUNC <V>	77
6.2 Mark this line - FIELD, FUNC F	78
 7 KEYS USED TO OPERATE ON MARKED TEXT AREAS	 81
7.1 Delete the marked area - DELETE, FUNC D, D	83
7.2 Replace the marked area with spaces - SHIFT + DELETE, FUNC <SPACE>	83
7.3 Copy the marked area, insert it at cursor position - COPY, FUNC C	83
7.4 Replace cursor position area with the marked area - SHIFT + COPY, FUNC I	84
7.5 Move the marked area, insert it at cursor position - MOVE, FUNC M	84
7.6 Replace cursor position area with marked area, spacefilling the marked area - SHIFT + MOVE, FUNC E	85
7.7 Insert area - I	85
7.8 Cancel the current area marking - CANCEL, FUNC X	86
7.9 Re-mark the last marked area - SHIFT + CANCEL, FUNC Q . . .	86
7.10 Convert the marked area to lower case characters - F6, FUNC L	86
7.11 Convert marked area to upper case characters - SHIFT + F6, FUNC U	87
7.12 Substitute <old string> with <new string> - S	87
 8 COMMANDS USED TO EDIT TEXT	 89
8.1 Function key commands	91
8.1.1 Delete this line - the F1 key, <D><D>, FUNC <D>	91

Section	Page
8.1.2	Insert a new line here - the F2 key, <L>, FUNC <L> . . . 92
8.1.3	Delete character - A , DEL, <A> . . . 92
8.1.3.1	Alternative delete mode . . . 93
8.1.4	Split the line at cursor position - F5, FUNC Y . . . 94
8.1.5	Link this line to the preceding line - SHIFT + F5, FUNC Z 94
8.1.6	Move the cursor to the beginning of this line - SHIFT + <===, FUNC <R>, <R><R> . . . 95
8.1.7	Move the cursor to the end of this line - SHIFT + ==>, FUNC <F>, <F><F> . . . 95
8.2	Control key commands . . . 95
8.2.1	Accept control character - <O> . . . 95
8.2.2	Undo editing on this line . . . 96
8.2.3	Copy one character from previous to current line - <C> . 96
8.2.4	Copy one character from next to current line - <N> . . . 96
8.2.5	Move cursor forwards to specified character on the line - <F>char . . . 96
8.2.6	Move cursor backwards to specified character on the line - <R>char . . . 97
8.2.7	Copy characters from previous line to current line - <P>char . . . 97
8.2.8	Copy characters from next line to current line - FUNC <P>char . . . 98
8.2.9	Delete characters up to and including char - <D>char . . 98
9	LINE MODE EDIT . . . 99
9.1	Shadow line . . . 101
9.2	The commands . . . 101

APPENDIX

A	PED AND SYSTEM ERROR MESSAGES . . . 107
B	THE MENUS . . . 113
C	HELP Structure diagram . . . 117
D	ASCII TABLE - NATIONAL CHARACTER SETS . . . 123
Index	127

C H A P T E R 1

AN INTRODUCTION TO PED

1 AN INTRODUCTION TO PED

1.1 Some comments on notation conventions

To make the manual readable some "shorthand" notations are being used. The main notations are:

<x> When occurring in the text it refers to a command performed by holding the CTRL key down while typing the key inside the angle brackets.

&x This is another way of writing a non-writable character. It is not much used in the manual, but is the way PED writes control characters. If PED cannot print a character, a & sign is put instead. Use the <V> command to get the value of character.

yB If y is a number, the B (or b) denotes that this is an octal number (ie., base 8). This is standard in all ND documentation.

FUNC On NOTIS terminals this is a function lead-in key placed on top of the numeric key pad. To perform this command on a non-NOTIS terminal, you have to input CTRL UNDERLINE (<_>). (on some terminals underline is a shift character and the command is then CTRL SHIFT UNDERLINE). The ascii value is 378.

↵ This sign is the marking of the carriage return key on a NOTIS terminal. (ascii 15B) On other terminals this may be marked as "RETURN" or "CR".

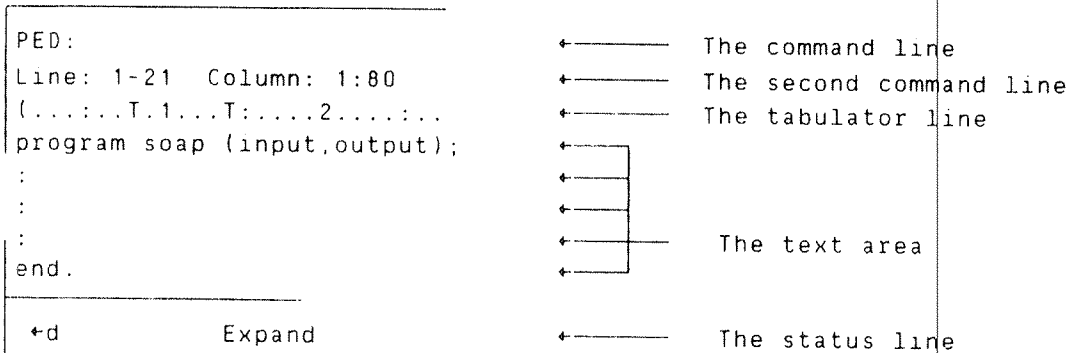
↖ This is the sign on the HOME key on the NOTIS terminal. This may be marked "HOME" on other terminals.

HOME position

Topmost left corner of the screen. It is marked with PED:.

1.2 The Screen

PED will use the screen for a number of different purposes. All of them have a fixed position on the screen, so that although much information is displayed, it should be easy to understand. On the NOTIS terminals, some lights (LEDs) are also used to make some frequently used messages more visible.



1.2.1 The text area

The text area can be divided into a maximum of 3 windows. Each window shows a part of a named region. See page 15 on what a region is. This feature with several windows is called a split screen. The screen may be divided either horizontally or vertically (not both at the same time). A split screen is defined in Menu no. 1 (the six fields at the bottom) in this way:

Vertical split (all windows occupy 21 lines, but different no. of columns).

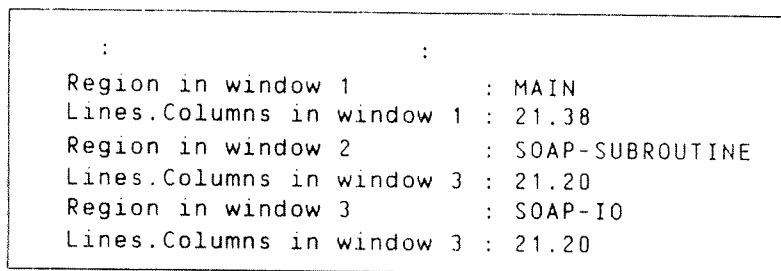


Diagram illustrating a windowed editor interface with a main window and three sub-windows.

Main Window Content:

- PED:
- Line: 39-59 Column: 1-80 Region: MAIN Position: _____
- (.....T.1...T:....2.....T.....) (.....T.1...T:....2 (.....T.1...
- Tabs etc. for window 1
- Text in region MAIN, window 1
- Status line

Sub-windows Content:

- Text in region SOAP-SUBROUTINE window 2
- Text in region SOAP-IO window 3

Diagram Labels:

- Active region
- Split lines

```

:
Region in window 1 : MAIN
Lines.Columns in window 1 : 11.80
Region in window 2 : SOAP-IO
Lines.Columns in window 3 : 9.80
Region in window 3 :
Lines.Columns in window 3 : 0

```

The diagram illustrates a window structure with several regions and tabs. At the top, a horizontal bar is labeled "Active region". Below this, the main content area is divided into two sections. The top section contains the text "PED:" followed by "Line:39-59 Coloumn: 1-80 Region: SOAP-IO" and "Position: _____". Below this text is a long line of characters: "(.....T.1...T:....2.....T.....T.....T.....T.....T...". Two vertical arrows point upwards from the text "Tabs for region SOAP-IO" to the first and second 'T' characters in this sequence. The bottom section of the main content area contains the text "The text of region SOAP-IO in window 2" and "Status line". A vertical arrow points upwards from the text "Split line" to the right edge of the window, indicating a split line.

1.2.2 The status line

The status line is the line in inverse video at the bottom of the NOTIS terminal screen. This line will at all times give the following information, which corresponds to values set in the menus or with the various commands and function keys.

- '+d' if alternative delete mode is set.
- 'Verify' if verify mode is active.
- 'Append' if append mode is active.
- 'Expand' if expand mode is active.

1.2.3 The command line

The command line is the first line of your terminal if you are in page-mode. The line is indicated by

PED:

This line is also used to display error messages and other messages from PED. Some of the commands use the command line to display parameter input. The right end of the line is used to indicate that file operation is in progress. The display has 10 segments, and indicates how much is done.

Example:

If you read a file, an arrow moves from left to right one step for each ten percent of the file. After 30% the display looks like this:

—————→—————

1.2.4 The second command line

The second command line is the second line displayed on the terminal. It can look like this (a little compressed):

Line: 261-281 Column: 1-80 Region: MAIN Position: ————+———

where:

- Line: The numbers displayed are the line numbers within the first and the last line on the screen window.
- Column: The numbers displayed are the column numbers of the leftmost and rightmost columns of your screen window. This is normally 80 character positions, and if the first number is different from 1, you are not looking at the leftmost part of your buffer.
- Region: The region is the name of your current region. Default is MAIN. The commands for marking areas, or working with marked areas take current region as default region.
- Position: Indicates where in the buffer you are, relative to the first and last line. The file is divided into 10 equally sized parts, and the cross indicates where you are now. Initially, when you start to write a new file, your position is "last" and the indicator is thus placed in the rightmost position.

1.2.5 The tabulator line

The tabulator line is the third line of your terminal display, and contains some tabulator and border information. When you first enter PED it will look like this for the first character positions:

(...:..T.1...T:....2.....T.....

- (The left parenthesis is placed in the position of the left border. A right parenthesis is displayed in the position of the right border. See also page 35 on Borders.
- . The dot appears in all positions where there is no other sign.
- : The colon appears every 10th character position, beginning at position 5. It will alternate with a number every 5 positions.
- number The numbers 1,2,3,4,5,6,7,8,9,0 are displayed in the positions 10, 20, 30,... Mathematically it may be said that the column position is displayed with only the 10's place unmasked.
- T A "T" denotes that a tabulator stop is placed in that position. To learn how to manipulate the tabulators, see page 33.

0 A "0" (not displayed as a standard) denotes a secondary tabulator stop.

Only one character at a time can be displayed in a given position. The priority is like this:

6	(and)
5	0
4	T
3	number
2	:
1	.

The highest number gives the highest priority.

1.2.6 The WAIT message

The wait message occurs in the upper right corner of the screen.

- WAIT -

It is used to indicate that the command you have given will take some time.

Example:

If you try to read or write a file, PED will ask the file system to open the file. At the same time the message - WAIT - will be displayed. The wait signal is switched off when PED starts the actual reading of the file.

1.2.7 The LED display on the terminal

Above the keys on the keyboard of the NOTIS terminals are some red LEDs. (light emitting diodes) Three of them are used by PED to indicate some status variables. Their meaning is:

EXP or EXPAND

The LED is on when expand mode is on.

INS or INSERT

The LED is on when insert mode is on.

BUSY

The LED is on when PED is doing things that may take some time. This can be when:

- Executing a READ/WRITE command
- Moving and deleting areas

It is used to notify the user that PED has accepted the command, although the screen is unaltered.

1.3 The command types

There are 3 types of commands in PED.

- HOME commands
- Function key commands
- Control key commands

1.3.1 The HOME commands

All HOME commands must be given in the screen HOME position, which is marked 'PED:'. You use the \uparrow key (HOME) to enter this position. To give the command you only type one character. All one character commands are HOME commands.



A HOME command may be cancelled by pressing the HOME key in reply to a prompt. The GET-STRING, SUSTITUTE, READ and WRITE commands can be interrupted by HOME.

Many of the HOME commands have one or several parameters. Only the mandatory ones are asked for if you type \downarrow after each parameter. If you want to give values to any other parameter, you have to use the cursor up (\uparrow) or left wide arrow (\leftarrow) to move to previous parameters, and cursor down (\downarrow) or right wide arrow (\rightarrow) to move to next parameter. You can edit the parameters you have input until you input \downarrow to the last mandatory parameter.

1.3.2 The function key commands

The NOTIS keyboards have a number of dedicated keys. Most of them have a function in PED. They can be activated both in the text windows and in the HOME position. On non-NOTIS terminals, the function commands are activated by a sequence of keys. Some of the function key commands have equivalent HOME commands.

1.3.3 Control key commands

Control key commands are used to edit text in the text windows and to edit parameters to HOME commands. Some of them have equivalent function key commands.

1.4 How to start the program

To enter PED, you give the command:




QPED

Note that this command may have to be given as QPED-ENG or QPED-NOR etc., depending on which language version you are using. The system responds with

PED:

in the HOME position and its identification on the command line of the screen.

1.5 To start a new program file

The cursor must first be placed on the first character position of line 1 with the  key (CR) or the  key (HOME). If you are accustomed to earlier versions of PED, you may be used to doing this with the cursor down arrow (). This is an option in PED versions later than E, and can be set by a flag in Menu no. 2. See also the chapter on changing PED parameters page 33.

When you have written a few lines, and created the first part of the text for your program, you will usually want to save your text. You will then have to create a file recognized by the file system.

To do this, you move the cursor into command position with the HOME key.



You then create your file, by giving it a name with the W command (WRITE FILE: <"file name">) explained under "Commands for file handling" page 25, and thereby store your file in the computer mass storage memory for further use. If you want to edit an existing file, you may read it into the work area by the R command (see page 19).

1.6 The PED work area

PED has a work area containing the text edited during the current run of the program. This area can be filled with text both from existing files and directly from the terminal. The current text is edited from the terminal by giving commands and data to PED. You can have lines up to 255 characters long, and PED will break the lines if they are longer.

NOTE:

Editing is only done in the work area and does not automatically store the text on a file. It is therefore necessary for you to write the file onto a disk file with the W command before you exit from PED. PED will ask you if you want to store the text if you forget.

1.7 Area

In PED you are sometimes asked to input an area. The concept is described here. Formally the definition is:

<region>.<first line>.<first column>:<second line>.<second column>

Most of these parameters can be omitted, and PED will use default values when necessary.

Region

Region is an alphanumeric name of a region. A hyphen is accepted inside a name. The region must be known to PED. If you want to create a new region, use double quotes around the name. The region MAIN will always exist. If region is omitted, current region is assumed. Region names can be abbreviated. Regions are described on page 15

First line

The first line to be affected by a command. This parameter can be given as an expression, as all the numeric parameters.

Second line

The last line to be affected by a command. If this one is omitted, and the first line is specified, the value given as first line will be filled in for the last line as well.

First column

The left margin of the text to be affected by a command. The default value of this parameter is column 1 for all commands.

Second column

The right margin of the text to be affected by a command. The default value for this parameter is the last possible column (column 255) for all commands.

All

All is a predefined area command meaning "the whole current region", and can, f.ex., be used to delete all the text in the current region, while keeping the region name. "ALL" cannot be abbreviated.

1.7.1 Expressions

All numeric parameters can be given as expressions. The expressions are evaluated as integer expressions. Therefore a dot is illegal as part of a number, (in fact a dot will be treated as delimiter in area input) and any decimal part due to division will be lost. The expressions are evaluated using 32 bit words. Legal expressions are described under the Value command (page 43) in the chapter on utilities.

C H A P T E R 2

FILE HANDLING COMMANDS

2 FILE HANDLING COMMANDS

This chapter contains the commands in connection with the file system. The commands are now safer than in previous versions. You are asked if you want your text buffer saved before any command is executed that might corrupt it, or use it for some other purpose. To unfamiliar users, this is preferable. Those with experience with earlier versions of PED, will hopefully learn to appreciate it too, as soon as they become accustomed to it.

PED has a work area containing the text edited during the current run of the program. PED uses file 100 as scratch file during editing, and the editor will write out an error message and stop if file 100 is not open. The easiest way to open the correct scratch file is to log out (QLOGOUT,]), and start anew. PED is able to handle about 40,000 lines, so if you have problems with size, the error is usually that user SCRATCH has no more free space. Contact your System Supervisor to help you. The largest size file 100 that PED can handle is about 6000 pages (à 2000 bytes). You can have lines up to 255 characters long, and PED will break the lines if they are longer. This will be marked by a "\$" in column 1 of the broken line. See the command SET/RESET LF/CRLF AT END OF LINE page 50.

2.1 Regions

The regions are a new feature in PED, and utilized, it will make life a lot easier for programmers using multiple sources, text and list files. A region is a named area in the editor's work area, and you can have a maximum of about 40 regions in PED at a time. The first time a name is referenced, it has to be created by placing double quotes around the name.

Example:

"SOAP"

The first character in a region name has to be an alphabetical character. In the read command { page 19 } you can specify region when asked for INSERT BEFORE:. If you want to read the file SOAP-MAIN into the region SOAP, it will look like this:

Example:

PED:Read file: <SOAP-MAIN>↓

Insert before: "SOAP"↓

Now the whole SOAP-MAIN is read into your region SOAP. If you do not specify any region, the current region is used. To move between regions you use the home command move (M), and specify region name when asked for "Move to position:". It is also possible to copy to another region, and in all the copy area/move area/delete area

commands, a region name is accepted as an area, or as part of an area specification.

Example:

```
Delete area:soap.1:200
```

will delete the 200 first lines of the region SOAP.

If you are using the MARK or FIELD keys to mark text, the region is implicit, so that you can mark an area in one region, move to another, and then copy or move the text you marked to that area, just as you would do inside a region.

2.2 Commands to PED in a file

On the first line in a file you can write commands to PED. These commands are read by the editor when you read a file into your input buffer. The commands are :

2.2.1 Tabulator - TAB

The tabulator command is equivalent to the home command T, and is described in the chapter on changing the PED parameters. { page 33 } The command is initiated by the letters TAB, and space is used as separator inside the command.

Example:

```
C TAB S3 5
```

The TAB command starts one or more spaces after the comment sign, and will order PED to set tabstops beginning in column 3 with a tab for each fifth column up to column 255. If you set a plus sign "+" or a space in the character position immediately behind the TAB, tabulator marks (ctrl I [ASCII 11B]) will be expanded on input. If you do not want to have your tabs expanded as in a print file with commands to a printer, you input a minus sign "-" after the TAB command. For full description of the tabulator command, see page 33.

2.2.2 Borders - ()

The border command is started by two parenthesis like this "()", with a normal border command following. For a full description of the border command, see page 35.

Example:

```
C ()5:80
```

This will set the left border to 5 and the right border to 80.

All non-blank characters from position 1 are reserved for the beginning of a comment. This will prevent the compilers from reading the PED commands. Comments can be :

```
C      Fortran
%      MAC, NPL and Planc
*      Cobol
(* and { Pascal
```

or whatever you like that your subsystem will accept as a comment.

The commands are separated by a semicolon ';', and can be combined in setting both borders and tabulators. The tabulator command has to be input first. If you want to write more on the same line you also have to terminate the last command to the editor with a ';'. Multiple spaces are accepted as one. The lines below will all give the same response from PED.

Example:

```
C TAB F;() 7:72
C      TAB F ; () 7:72
C      TAB F ; () 7:72
Ctext tab f; () 7:72; more text
```

This will set tabulators to standard FORTRAN, left border to 7 and right border to 72.

2.3 Commands to PED from the SINTRAN command line

Commands can be given to PED from the SINTRAN command line. They are given as normal commands, specified with parameters separated by space or comma. A semicolon terminates a command. If a command terminates due to an error, the rest of the commands will be skipped.

Example:

To start PED with SOAP-MAIN in region MAIN, SOAP-IO in region IO, and SOAP-SUBROUTINE in region SUB, this is how to write it:

@PED SOAP-MAIN;R SOAP-IO "IO";R SOAP-SUBROUTINE "SUBR";

PED will now:

- 1) read SOAP-MAIN into region MAIN (default region)
- 2) read SOAP-IO into region IO
- 3) read SOAP-SUBROUTINE into region SUBR

Your current region is now region SUBR.

Example:

If you want to have the PASCAL compiler as your exit program, and start editing in line 250 in the file SOAP-MAIN you can do it this way:

@PED SOAP-MAIN;1,,,PASCAL;" 250;

PED will now:

- 1) read SOAP-MAIN into region MAIN (default region)
- 2) go into Menu no. 1
- 3) write the text "PASCAL" in the third line (the Exit Program Name)
- 4) The semicolon returns to home position, ready to accept more commands.
- 5) The "enter position" command is executed.

PED will now display the lines 245-265 with the cursor in column 1 in line 250.

2.4 Read file - R

The R command will cause text to be fetched into the PED work area from a file which has already been edited and written in the computer's memory.

R

The basic format of this command is:

- Read file: <file name>↵

As you can see, <file name> is the only mandatory parameter in this command and if you give this parameter alone followed directly by ↵, your whole file from the first to the last line will be fetched into the work area. If no (DIRECTORY:USER) is specified in the file name, PED will assume that you want to read a file under your own user, and will put in empty parenthesis in front of the file. This will prevent the file system from looking for your file under user SYSTEM.

But you also have the possibility to fetch only a part of the file into the work area, by specifying the optional (from line),(to line) parameters.

The format of the command is then:

PED:Read file: <file name>↓

Insert before:↓ Area: <from line>:<to line>↵

Example:

PED:Read file: SOAP-MAIN↓

Insert before:↓ Area: 25:250↵

Only lines 25 through 250 of your file 'SOAP-MAIN' will then be fetched into the work area. This is useful if you want to use parts of a file only, since you can choose either to create a separate file with these few lines by giving it a new name, or to shorten your current file because you are no longer interested in the rest of it.

The abbreviation will be carried out if you write lines 25 through 250 of your file 'SOAP-MAIN' in the computer's memory under the same name 'SOAP-MAIN', because the lines before line 25 and after 250 will then be erased and only lines 25 - 250 of 'SOAP-MAIN' will remain. Although you have fewer lines in your file now, the size of the file in pages seen from the file system is not reduced. To reduce that size, you will have to delete the old file, and create it anew.

See the section devoted to the W command - Write current text to <file name> or to <"file name"> { page 25 }

The R command also provides the possibility to merge a new file (or parts of it) into the file you are currently editing, i.e., to fetch more text from another file while you are working on the current file.

You can also decide exactly where you want the additional text inserted.

You then give the R command a second time, in this manner:

```
PED:Read file: <second file name>↓
Insert before: <line number>↓ Area: <from line>:<to line>↓
```

Example:

```
PED:Read file: SOAP-SUBROUTINE↓
Insert before: 250↓ Area: 300:600↓
```

You have now borrowed 300 lines of text from your second file, 'SOAP-SUBROUTINE', and you have inserted these 300 lines before line 250 of your current file, 'SOAP-MAIN'.

You can also decide that you want the text inserted in another region than the one you are in.

You then give the R command a third time, in this manner:

```
PED:Read file: <second file name>↓
Insert before: <region>↓
```

Example:

```
PED:Read file: <SOAP-SUBROUTINE>↓
Insert before: "SUBR"↓
```

You have now placed the second file 'SOAP-SUBROUTINE' in the region SUBR. The region has to be surrounded by double quotes if it is not defined already. You will now have two files in PED: the SOAP-MAIN, and the SOAP-SUBROUTINE. The regions can be listed by the 'X' command. { see page 50 }

It is also possible to use the <V> command to give the line number as the parameter to the prompt Insert before:. If you have used the <V> in the text, you can write ↵ as response to the prompt, and the line number will be displayed with the cursor placed after the last number. You can then edit the number, or accept it by another ↵. The <V> command is described on page 47.

Note that when you fetch a second, complete file into the same region with this second R command, your command will result in one or more of the following prompts from PED:

Save edited text to <default file name> (Y/N)

This prompt occurs if you have modified something in your file before giving the second R command, and avoids your losing text that has not been written back to the computer's secondary memory yet.

The text is modified! Do you want to store it (Y/N)?

If there is text in the work area but no <default file name> the system will give this prompt and refuse to carry out the READ FILE command. The question has to be answered with Y or N before the READ command be executed. Type HOME to abort the command.

Do you want to clear current text before reading new file? (Y/N)

The prompt requests a confirmation that you really want <current file name> and <second file name> in the work area together.

If you reply YES your <current file name> will disappear from the work area. If you reply NO, it will remain in the work area when <second file name> is fetched.

Update the default file name to <second file name>?(Y/N)

The prompt requests confirmation of the default file name. The options are either to answer NO and thus maintain the <current file name>, or to reply YES and rename the merged files to the <second file name>.

In all these 'READ and MERGE' options the default values used by PED, whenever you simply type ↵ after <file name>, are:

- INSERT BEFORE: after the last line in the current region
- AREA: the whole file.

Some of the other subsystems on ND computers recognise special file types. PED will test for some of them on input if you do not specify something yourself. Listed are the file types in order of priority.

- 1) SYMB - Standard symbolic files.
- 2) PLNC - The ND Planc Compiler.
- 3) PAS - PASC is preferred by ND Pascal Compiler.
- 4) FTN - The FTN FORTRAN Compiler.
- 5) FOR - The ND-FORTRAN-100 Compiler.
- 6) COB - The ND COBOL Compiler.
- 7) NPL - The Nord PL Compiler.
- 8) MAC - The ND 10/100 Assembler.
- 9) ASM - The ND 500 Assembler.
- 10) BAS - The ND Basic Compiler.
- 11) MODE - For MODE files.
- 12) DATA - Only "symbolic" data files.
- 13) LIST - Preferred by many users for compiler listings.
- 14) TEXT - Text files in 7 bit format only.

It is possible to abort a READ command with \uparrow (home). PED will then ask:

Do you really want to abort the read operation (Y/N)?

If you answer N, the read operation will continue.

If you answer Y, PED will write out a message like this:

82 lines read (2048 bytes) (aborted)

or

Search aborted at line number -1

The last message is given if \uparrow is input while the WAIT message is displayed. \uparrow will only interrupt the READ if it was the first character input after the \downarrow terminating the READ command.

2.5 Update file - U

The Update command is a command to read a file into the work area. This command will prevent other users' access to the file while you are working on it. This is valuable if several programmers work on the same set of source files. It will also inhibit you from working on the same file from two terminals at the time. If you try to update several files, the last is the only one with that protection, and you should therefore not try to update several files at a time.



Pressing the character U will activate the update command, and PED will respond with

- Update file:

You then respond by

- Update file: < file name > ↵

There are no other parameters to this command.

2.6 Recover buffer - PED @

If you by accident are aborted from PED, or exit PED without saving what you intended to save, all is not lost. If you have not used file 100, you can write

@PED @

to SINTRAN. The "commercial-at" commands PED to not initiate the scratch file, but rather look for what it left in the scratch file when it was last terminated. If file 100 has been used for something else, the error message:

"MAIN" not found

will be written on the screen, and PED will terminate.

How often the scratch file is equal with your work area is determined by the parameter

Updates before save :

in Menu no. 2. If this parameter is ≤ 0 , the recover buffer command is useless, and PED will usually start with an empty buffer, or some rubbish. If the parameter is positive, a counter is incremented each time an internal function is called. This function is called each time

you move to a new line, at least twice if you edit a line, and at least three times if you insert a line. When the counter is equal to the value set, the buffer is flushed out onto the scratch file. The buffer is also flushed when :

- A READ command is performed.
- A move to another region is performed.
- An Exit command is performed.
- The buffer is expanded so that an index block has to be split.

The command (@PED @) will save all your regions if the machine stops, or if you by accident managed to start another program not using the scratch file.

You are warned that the editor will have quite a lot to do if you set the number to a small positive number. It will appear slower than if the number is negative, but this is a question of safety. A value of at least 100 is recommended, and if your system is stable and have many users, an even higher value should be chosen.

If you have not used any programs after you last used PED, it may be possible to restart PED by the SINTRAN command

@CONTINUE

This will only work if you have not been logged out, but if so, it is better than using the "@PED @" command.

2.7 Read file to "number of pages" - R+

If you have managed to copy a small file onto a big file by mistake, it is possible to save the last part of the file with the

Read: +<file name>

command. The + indicates to PED that it shall read beyond the max-byte-pointer, and read the number of pages as noted in the index table of the file. This will give you an average of 1k byte of rubbish at the end of the file, but this is easily deleted in PED. Remember to write the file so that the max-byte-pointer is set correctly. This command is most valuable if you have managed to copy (@COPY) the wrong way to a spooled device. You will usually not have read access to the file, so no data is lost. Only the max-byte-pointer is reset, and if you try the read +<file name>, you will get all your text back.

Example:

You have copied a line-printer file onto your file with the command

@COPY SOAP-MAIN:SYMB,LINE-PRINTER,↓

The command @FILE-STATISTICS gives the answer that the file SOAP-MAIN is empty:

```
@FILE-STATISTICS SOAP-MAIN:SYMB;]  
OUTPUT FILE:;]
```

```
FILE 117 : (PACK-ONE:SOAP-USER)SOAP-MAIN:SYMB;1  
           (INDEXED FILE)  
           PUBLIC ACCESS : READ  
           FRIEND ACCESS : READ, WRITE, APPEND  
           OWN ACCESS : READ, WRITE, APPEND, COMMON, DIRECTORY  
           OPENED 2 TIMES  
           CREATED 01.08.55  APRIL 7, 1983  
           OPENED FOR WRITE 01.09.18  APRIL 7, 1983  
           4 PAGES , 0 BYTES IN FILE
```

You then use PED like this:

```
@PED +SOAP-MAIN;]
```

and when the file is written back, the

```
@FILE-STATISTICS,SOAP-MAIN,,;]
```

command gives this answer:

```
FILE 117 : (PACK-ONE:SOAP-USER)SOAP-MAIN:SYMB;1  
           (INDEXED FILE)  
           PUBLIC ACCESS : READ  
           FRIEND ACCESS : READ, WRITE, APPEND  
           OWN ACCESS : READ, WRITE, APPEND, COMMON, DIRECTORY  
           OPENED 4 TIMES  
           CREATED 01.08.55  APRIL 7, 1983  
           OPENED FOR READ 01.13.21  APRIL 7, 1983  
           OPENED FOR WRITE 01.13.21  APRIL 7, 1983  
           4 PAGES , 7354 BYTES IN FILE
```

2.8 Write file - W

The Write command will cause text to be written from the PED work area into a file.



Pressing the character W will activate this command, and the following text will appear in the screen home position.

- PED:Write file:

The format of the answer is:

- Write file: <file name>↵

where:

<file name> is the name of the file to be written.

You then Write the whole file with name <file name> into the computer's disk storage.

Or

- Write file:<file name>↓

AREA:<area>↵

where

AREA:<area> is the portion of the text in the work area that you wish to write to the file in the computer disk storage with name <file name>.

The area format is: <region>.<from line>:<to line>

Defaults are:

region Your current region as indicated in line two on the screen.

from line 1

to line last line of the region specified.

It is not always necessary to indicate AREA; When you wish to write the whole file with name <file name> in the computer's memory you can leave this parameter out and simply give ↵ after <file name> instead of the ↓

If you get the message: <FILE NAME> NO SUCH FILE NAME from PED, just input <W>, and you will get the command back for editing.

PED only accepts ↵ as default file name. '*' will not be accepted, as it was in previous releases of PED.

A new file name must be given in double quotes when you want a file created for the first time with a Write file: <"file name">.

You can create it either with the first Write file command, in which case it will also be set in Menu no. 1 at the same time, or by setting it yourself: <"default file name"> in Menu no. 1 when you enter PED to start the editing of a new file.

Example:

Write file: <SOAP-MAIN>↵

results in the whole file named SOAP-MAIN being written, whereas

Write file: <SOAP-MAIN>↓

AREA:1:50↵

results in only lines 1 to 50 being written.

Example:

Write file: <"SOAP-FILE-IO">↓

AREA:20:150↵

In this example we have decided that we want to create a file called 'SOAP-FILE-IO' and write lines 20 - 150 of our current text in this file.

In the upper left corner of the screen the word Writing:<file name> will appear, and in the upper right corner a position arrow indicating how far the Write has progressed. Once the required number of lines have been written in the file, the reply 'n' LINES WRITTEN ('m' bytes) will appear in the upper left corner.

It is possible to interrupt the WRITE command with \. If you input \ while PED is writing a file, PED will stop and ask:

Do you really want to interrupt the write operation (Y/N)?

If you answer N, the write operation will continue. If you answer Y, the write operation is aborted, and PED will write:

xxx lines written (yyy bytes) (aborted) - OUTFILE IS INCOMPLETE!

If you manage to interrupt the write operation while the wait message is on, the message will be:

search aborted at line number -1

In this case no text has been written, and the text on the file you were writing to has not been changed. \ will only interrupt the WRITE command if it is the first key typed after the ↵ terminating the WRITE command.

When you Write text in the computer's memory under a name which is different from the <default file name> already set, PED will ask:

Update default file name to SOAP-FILE-IO:SYMB Y/N

It is asking you to confirm that you do not want this text to be called 'SOAP-MAIN', but that you want to name the new file 'SOAP-FILE-IO'.

If you answer Y, the new file named 'SOAP-FILE-IO:SYMB' is set to default file name in Menu no. 1. You will thus suddenly be working on the text of a file called 'SOAP-FILE-IO' and no longer on 'SOAP-MAIN'. On the other hand, your original file 'SOAP-MAIN' remains intact in the computer's memory.

If you answer N, the process is reversed: you will still be working on the text of your file 'SOAP-MAIN', but you will have created a new 'SOAP-FILE-IO' file with the same text. Hence, you never lose any text by writing text to a new file name.

In either case you find yourself with two files which can be fetched into the work area separately with the R command, each under its own name:

- a) your original 'SOAP-MAIN' file
- b) a new file called 'SOAP-FILE-IO', containing 130 lines of text borrowed from the 'SOAP-MAIN'.

Be aware that if you write only a part of your current file 'SOAP-MAIN' in the computer's memory again under that particular name, you will lose the part of the text that you do not write. This is therefore a way to shorten a file. This will not reduce the number of pages on disk that the file will occupy.

Some examples

- 1) Write file:<file name>↓

The file <file name> must already exist, and the full contents of the PED work area will be written to it.

- 2) Write file:<"file name">↓ AREA:10:140↓

A file with the name <file name:symb> is created, and lines 10 through 140 from the PED work area are written to this file.

3) Write file:<file name>↓ AREA:20↓

Only line 20 of <file name> is written.

4) Write file:<file name>↓ AREA:<region name>↓

The text in region <region name> is written to the file named <file name>.

5) Write file:<file name>↓ AREA:20:\$↓

All the lines from line 20 onwards from your current region are written into the file.

6) WRITE FILE:↓ and another ↓

Write all the lines from the PED work area under the <default file name> which has been set either with a previous READ FILE or WRITE FILE command, or in Menu no. 1.

Before any Write to the default file is carried out, the default file name is displayed, and must be confirmed by pressing ↓ once again.

2.9 Append file - A

A

The append command is a command to append the work area at the end of the file specified. If you use the down arrow (↓) after you specify the file name, you can specify area as in the Write command.

2.10 Exit from PED - E, EXIT, FUNC

When you have finished your work in PED, and stored your file text in the file named <file name> again, press:

EXIT

or FUNC

#

This will terminate PED, and return to SINTRAN. The command will ask you if you want to save what you have edited, if you have not already done so. For an extensive explanation see page 52.

2.11 Activate PERFORM - FUNC \$, SHIFT + EXIT, C

The subsystem PERFORM can be activated from PED by the following commands:

SHIFT + EXIT or FUNC \$

The command activates a program called PERFORM, which is a SINTRAN Utility Program. When used, PED will terminate, and will ask you to save all edited regions. For more information see page 53.

2.12 Activate the Text Formatter NOTIS-TF - SHIFT + PRINT, FUNC ", J

SHIFT + PRINT or FUNC "

The command activates NOTIS-TF with your current default file, unless the main file-name is specified in Menu no. 1. Then the main file is used. PED will terminate, and will ask you to save all edited regions. For more information see page 54.

CHAPTER 3

CHANGING PED PARAMETERS

3 CHANGING PED PARAMETERS

These commands are used to set a number of editor variables, such as input limits and tabulator marks. These include those marked on the status line, and those found in the menus.

3.1 Set new tabulator stops - T

The T command enables you to set tabulator positions at your own choice. See also the TAB function key in the next section.

The format of the command is:



- TABULATORS:<position-1>,<position-2>,...<position-'n'>,]

where the <position-number> is a number between 1 and 255, in increasing order.

The following may be specified as <position>:

- C for COBOL : Tabs = 8, 12, 16, etc.,..... max. 255
- F for FORTRAN : Tabs = 7, 11, 15, etc.,..... max. 255
- L : Tabs = 4, 7, 10, etc.,..... max. 255
- M for MAC : Tabs = 8, 14, 30, 40, etc.,... max. 250
- P for PLANC/PASCAL: Tabs = 5, 9, 13, etc.,..... max. 255
- T for Text : Tabs = 9, 18, 27, etc.,..... max. 250
- - for No tabulators ie., clears all tabstops.

The increment may be altered by specifying <increment>, ie., <position,increment>. Thus, <T,5> results in tabulator positions 9, 14, 19, 24, etc.

An eighth possibility is to specify:

- S for Special

S has two parameters:

the <first position> and

the <increment>.

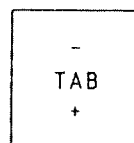
This may be added to a list of positions, eg., if You want tabulators in the positions: 5,7,13,20,30,40,50,60,70...., the quick way of writing this is to input the irregular tabs (5,7,13) succeeded by an S command: S,20,10 like this:

Tabulators: 5,7,13,S,20,10,]

They are at all times marked on the dotted line (third screen line) together with the () indicating the borders. Default tabulator setting is MAC setting (M).

3.2 Set/reset tabulator positions here - the TAB key, FUNC TAB, FUNC <T>, FUNC <I>

With the tabulator key:



or FUNC TAB or FUNC <T>
or FUNC <I>

you can set and reset tabulator positions for a program or a data file in the character positions of your choice.

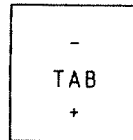
Pressing the tabulator key when the cursor is on the tabulator position you want will result in the position being marked with a T on the position line on the screen (the dotted line). Tabulator positions are deleted by pressing the key a second time.

The TAB key is used when the cursor is in the text area.

3.3 Set/reset secondary tabulator positions here - SHIFT + the TAB key, FUNC BTAB, FUNC <Y>

You also have a secondary tabulator set, marked by the letter "D" on the tabulator line. To set and reset the tab mark you do as you do with the standard tab set, using the commands

SHIFT +



or FUNC <Y>

There is no equivalent HOME command for the secondary tabulator set.

For commands to navigate between the tabstops, see the chapter on navigating commands.

3.4 Set new editor borders - B

The HOME command B enables you to set new borders for the text to be entered after the new borders have been set. See also the border functions in the next sections, and the default borders set in Menu no. 2.



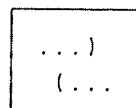
The format of the command is:

- Borders:<left column>.<right column>.]

Borders are used to detect the beginning and the end of a line, and to stop the user from unknowingly write program sentences and commands in illegal character positions. It is very useful for a FORTRAN programmer to be stopped at column 72, and some of the ND compilers have also some practical limits not demanded by the language, although they are usually so large that you will probably not know them. A more normal problem is the paper width of your printer. Usually this is either 132 or 80 characters, and this may be useful standard limits.

3.4.1 Set left editor border in this position - (.... FUNC (

Pressing the border key:



or FUNC (

sets the left editor border marked with (on the dotted line, in the character position under which the cursor is placed when the key is pressed. The border key updates the editor borders set in Menu no. 2. You thus have the possibility of modifying this border if you want a portion of your text to start (and end, too - see the next command) in another character position than the rest.

3.4.2 Set right editor border in this position - SHIFT + ...), FUNC)

If you press:

SHIFT +

...)
(...

 or FUNC)

while the cursor is in the character position where you want your right editor border to be, this position will be marked) on the dotted line.

3.5 Set the editor in line mode (QED compatible) - #

SHIFT +

#

The command sets PED in line mode (see chapter 9 LINE MODE EDIT), thus making it compatible with programs such as QED (see QED Users Manual ND-60.031). TELETYPE and DECWRITER terminals can run PED in line mode only.

This command is used to change the editor from running in page mode to run in line mode. The command has no parameters.

3.6 Change the terminal type to <type> - !

On TDV-2200/9-ND-NOTIS the terminal type is 53, and on FACIT-4420-ND-NOTIS it is 57. To change terminal type, you use the command

SHIFT +

!

The system's prompt and the parameter are:

PED:New terminal type: <terminal type number>

The present list of supported terminals is:

TELETYPE-ASR-33	(2)
TANDBERG TDV-2115	(3)
INFOTON 200-1	(4)
INFOTON 400	(5)
DEC-LA36 (DECWRITER-II)	(11)
TANDBERG TDV-2215-EXTENDED	(36)
TANDBERG TDV-2215-SDS-V2	(52)
TANDBERG TDV-2200/9-ND NOTIS	(53)
FACIT-4420-ND NOTIS	(57)

These are the terminal types the PED editor can presently be used on. If other terminals are used, contact should be taken with ND to find out if they can be implemented, and what it will cost you. It should not be expected that the user friendliness will be as good as on the NOTIS terminals.

The ! command is meaningless if one is using a terminal without an ability to run in different modes. It may be used if the initial terminal type was wrong, though. The command will cause the editor to reinitiate the screen, eg., some terminals will clear the screen, and reset terminal values to standard.

3.7 Set/reset expand mode - EXP, <E>

Pressing this key:

INS EXP

 or <E>

causes the word EXPAND to appear on the status line. The function enables you to insert new characters in the existing text, before the current cursor position. The expand mode is turned off by another typing of the expand key. The word EXPAND will then disappear from the status line.

Example:

You wanted to write 'existing', but you made a typing mistake and it came out 'exiting'. You now position the cursor on the 't', press the EXPAND key or <E> and type the missing 's'. The text will open up and make room for the 's'.

When expand mode is set, a LED marked EXP will switch on, and stay on as long as expand mode is on.

You can continue to type in new characters in this manner for as long as EXPAND mode is active, and it is thus possible to insert unlimited amounts of new text this way.

To get out of EXPAND mode you simply press the key a second time, and the word EXPAND disappears from the bottom line.

The EXPAND key or <E> respond both in HOME position and in the text.

3.8 Set/reset append mode - INS,

Pressing the same key in shift position or :

SHIFT +

INS
EXP

 or

causes the word APPEND to appear (or disappear) on the bottom line. The function enables you to insert new lines of text in the existing text, under the current line. The new lines are opened up each time you press \downarrow . The LED marked INS on the keyboard will be on as long as the append mode is set.

Example:

You have written a 10-page program and discover that you have forgotten a subroutine in the middle of it. There is no cause for despair, and you do not have to retype the whole second half of your document. You simply position the cursor on the line above the place in the text area where you want blank lines to be inserted to start editing the forgotten text, press SHIFT and the INS key, or , and give \downarrow . You will see the text open up and a blank line appear. It does not matter how long the text is that you had originally forgotten; the system will continue to give you blank lines in this manner until you press SHIFT + INS or again to get back into the usual edit mode.

The function responds both in HOME position and in the text.

3.9 Set/reset alternative delete mode - FUNC Δ , FUNC DEL

The

FUNC

Δ

 or FUNC DEL

command enables you to set/reset alternative delete mode. When alternative delete mode is ON, it leads to the 'd' (but not <A>) being interpreted in different ways according to prior input. The flag '+d' is placed on the bottom status line when alternative delete mode is on.

Example:

- if the cursor is positioned on a character, 'd' deletes the character and contracts the text, as usual.
- if EXPAND mode is OFF and you use 'd' while inputting text, the last character input is replaced by a space, ie., the text is not contracted.
- if EXPAND mode is ON, the last character input is deleted.

More information is given on page 83 under d.

3.10 Display menu no. 1 - 1

Pressing the digit 1 while the cursor is in the HOME position brings Menu no. 1: Editor Flags up on the screen (see Appendix B).

1

3.11 Display menu no. 2 - 2

Pressing the digit 2 while the cursor is in the HOME position brings Menu no. 2: Editor Defaults up on the screen.

2

CHAPTER 4

PED UTILITIES

4 PED UTILITIES

4.1 Evaluate a numeric expression - V

The home command

V

establishes the decimal, octal, hexadecimal and ASCII values of any 32 bits integer expression.

Example:

You want to establish the values for the figure 1234567.

You give the V command, and after the system prompt 'Value of: you input 1234567 as parameter and press ↵.

In the screen HOME position the operation looks like this:

```
PED:Value of: 1234567 ↵
Is: Decimal 1234567 Octal 4553207 Hex 12D687 Ascii
"&RV&G"
```

Generally speaking, all arithmetic expressions are 32 bit integer expressions (-2^{31} to $2^{31}-1$). There is no testing of overflow neither in the intermediary calculations nor in the results. A division by 0 is not detected.

For the benefit of those familiar with formal description of programming languages, a simplified BNF description is included. Terminating symbols are printed in lower case bold. (If you are not accustomed to this kind of notation, you should rather try the explanations after the BNF.)

Simplified BNF notation	
E	::= E xor E1 E or E1 E1
E1	::= E1 and E2 E2
E2	::= not E3 E3
E3	::= E3 + T E3 - T E3 shift T T
T	::= T * U T / U T (U U
U	::= + E3 - E3 F
F	::= abs P P ** F P mod F P
P	::= (E) ID
ID	::= \$ #c 'chrs' /string/ @ number

Explanation of the BNF:

The explanations refer to the terminating symbols in the BNF. That is: the symbols you are supposed to type.

\$ This corresponds to indicating the number of the last line in the current region. In the 'AREA' parameter of the R (retrieve document) command, \$ equals 65535.

Example:

If there are 56 lines, \$ will be understood as 56:

PED:Value of: \$,]

Is: Decimal 56 Octal 70 Hex 38 Ascii "8"

#c This corresponds to the internal representation of 'c':

Example:

The value of "A":

PED:Value of:#A_↵

Is: Decimal 65 Octal 101 Hex 41 Ascii "A"

The value of "<[>" (ctrl [):

PED:Value of:#<0><[>_↵

Is: Decimal 27 Octal 33 Hex 1B Ascii "&["

'chrs' This corresponds to 1 to 4 characters, right justified, in a 32 bit integer. If "" is to be included, it has to be written twice.

Example:

If you want to evaluate ab'c, you will have to type it like this: 'ab'c'

/string/ This implies searching for a string in the current region. The search starts at the current line of current region, and will only find the first occurrence of string. To get the next, you have to move to the line after the value returned from the VALUE command, and then start a new search. The line number is then displayed in decimal, octal, hex and ASCII. Instead of / one may use ! % & ? or |.

Example:

The input of /begin/ will give the result:

Is: Decimal 104 Octal 150 Hex 68 Ascii "h"

if the first begin in your program after your current line is line 104.

number This encompasses decimal numbers (0 to 9), octal numbers (0 to 7 terminated by B) and hexadecimal numbers (0 to 9 and A to F, the number is terminated by H and the first digit must be within the range 0 - 9).

Example:

```

- 99    = 99D    = 143B = 63H = "c"
- 10d   = 10D    = 12B  = 0AH = "&J"
- 17b   = 15D    = 17B  = 0FH = "&O"
- 0abh  = 171D   = 253B = 0ABH = "+"

```

Upper and lower case are equivalent.

-E3 +E3 Contrary to usual mathematics, these are not positive and negative numbers. Instead, the editor will add a number before the sign in order to make it an addition or a subtraction. -3 is therefore understood as x-3, where x depends on the context. The number -3 should be specified (0-3). x is equal to 0 in the value command, equal to the line the cursor goes down to when the HOME key is input in all AREA parameters, equal to the line number of the first line on the screen in the M (move to position) command and equal to the old contents of most of the parameters in the menu commands.

Example:

if line 30 to 50 are visible on the screen and you give the command M:<-5>, this is equal to giving the command M:<30-5> or M:<25>. In all these events lines 25 to 45 will be shown on the screen.

not Computes the bitwise not operation on the succeeding expression.

Example:

```

PED:Value of: not 100
Is:  Decimal -101   Octal 37777777633   Hex  FFFFFFF98
    Ascii "&-&-&-&["

```

xor	Computes the bitwise xor operation between the preceding and succeeding expression.
or	Computes the logical or between the expressions.
and	Computes the logical and between the expressions.
shift	Computes a bitwise shift operation on the preceding expression a number of times given by the succeeding expression. Positive number of times gives left shift, negative values give right shift.
abs	Computes the absolute value of the succeeding expression.
**	The first expression is raised to the power of the second expression.
mod	Computes the modulus of the two expressions.
@	Commercial 'at' is a variable containing the last computed result. It can be used in expressions, and is useful if your expression is too long, or too complicated to write on a line.

Priority between the operators is expressed in the BNF table. The last line has highest priority (computed first) and the first line is computed last. This means that the expression $3+4*5$ is calculated as $3+(4*5) = 23$. If one inputs a number just in front of or just after a parenthesis expression without an operator between, $*$ is presumed to be the operator. This means that $3(4+5) = 27$ and $(4+5)3 = (4+5)*3 = 27$. $2**3**2$ is calculated as $2**(3**2) = 2**9 = 512$, because the expression $F ::= P ** F$ results in the first link being raised to the expression that follows.

4.2 Verify current cursor position - <V>

The <V> command can be used for three purposes:

- It enables you to verify the current cursor position with respect to value, position and line length.

Example:

- You press <V> while the cursor is positioned on the character X. In the top right corner of the screen, the system now informs you of the line you are on, the column you are in, the total length of your line, the character your cursor is on, and the ASCII value of this character. The format is:

```

Line  Column  Length  Char  Ascii
4456      63      64      X    1308

```

- You will note that the word 'From:' precedes these indications. This is relevant when you use <V> to mark an area of text for use in the HOME commands Delete, Insert, Move and Substitute described in chapter 7.

To mark an area you proceed as follows:

- You position the cursor on the line and character position where you want your area to begin, and press <V>. The system displays:

```

From:  Line  Column  Length  Char  Ascii

```

- You now position the cursor on the line and character position where you want your area to end, and press <V> again. The system displays:

```

To:    Line  Column  Length  Char  Ascii

```

- The text area is now marked and will be 'remembered'. It may therefore be used in one of the HOME commands mentioned above.

To remember whole lines, the <V> must be input on the same character position of the first and the last line of the area you are marking.

If <V> has been used to mark a text area for one of the operations mentioned, no other <V> command nor operations causing renumbering of the lines in your document must be carried out before that particular area has been used for the purpose it was intended for. The line and column numbers 'remembered' would in such a case no longer be correct. If you move to another region, the <V> markings will not mark text in your previous region, but the equivalent area in your current region. We recommend the use of MARK if you want to copy between regions.

As HOME command, <V> sets/resets VERIFY mode. The word VERIFY appears on the status line while the mode is active. At the same time, the 'Line - Column - Length - Char - Ascii' indications explained above are displayed and the information given for each character every time the cursor is moved.

4.3 Execute a SINTRAN command - @

It is possible, while editing text in PED, to execute a SINTRAN command without having to exit from the editor. To do so, you press:

@

However, there is no check performed on the SINTRAN command given, and if this is not an actual command, such as LIST-FILES, but a valid :PROG file or reentrant system name, then this system will be run, and you will exit from PED and lose PED text that has not been stored with a W command.

Other strings may cause SINTRAN to stop PED. If this is the case, ie., if the @ suddenly appears, you may use the SINTRAN command:

@CONTINUE

to reenter PED without losing any of your current text.

Example:

You are editing text, and need to check some names of other files to be referred to.

Go into HOME position and press @.

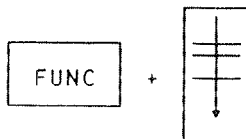
The prompt SINTRAN COMMAND: appears in the screen HOME position.

Give the SINTRAN command: LIST-FILES,]

The system responds by blanking the screen and with the acknowledgement: EXECUTING: LIST-FILES

Once you have found what you wanted, you press the space bar, and your screen of text is restored. It is thus unnecessary to exit from PED.

4.4 Set/Reset LF/CRLF at end of line - FUNC LF



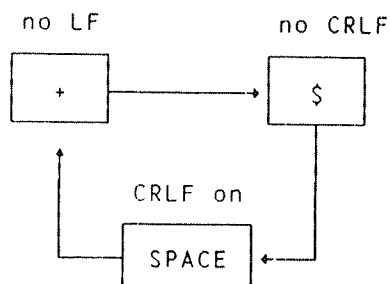
This command is used to set/reset LF/CRLF at the end of file. This can be useful when you want to alter a file that had for some reason lost CRLF/LF, or was transported from some system using another "end of line" marker than CRLF.

\$ A missing CRLF on input is marked by a '\$' in the first character position of the broken line. This will occur if the input line is longer than 255 characters.

+ A missing LF is marked with a '+' in the first character position. This means that on output to a printer, the next line will be written upon this line.

The character in the first character position will not be displayed, but is not lost, and can be examined by <V>.

The command sets the flag in circles when used repeatedly.



4.5 Display regions - X

The home command



will clear the screen, display all region names, the size of each region and the SINTRAN filename, and whether the region has been changed. An arrow (→) will point at the current region. The display will look like this:

<u>Region</u>	<u>Lines</u>	<u>Modified</u>	<u>Default file</u>
MAIN	1435	Y	SOAP-MAIN:SYMB
SUBROUTINE	548	N	SOAP-SUBROUTINE:SYMB
→FILE-IO	356	Y	SOAP-FILE-IO:SYMB

Here we have 3 regions, your current region is FILE-IO, and you have made some changes in that region. You have also made some changes in region MAIN. If you do not specify a region name in the READ-FILE command, your file is read into current region. Region MAIN will always exist, even if you do not use it, and it is the region you will see displayed when you start PED from SINTRAN.

If the list of regions is so long that it will not fit on the screen, PED will prompt:

PED:More (Y/N)?

The answer Y will make PED display another portion of the list until all regions are displayed.

The answer N will make PED return to HOME, ready to accept new commands.

4.6 Re-display the whole screen - space bar, FUNC @

Pressing the space bar on your keyboard from HOME position will redisplay the current window. If you are down in a text window, the command

FUNC

@

will have the same effect. The function is useful, for instance, if a broadcast is received on the screen during editing and 'scrambles' your text. This happens quite frequently when a computer is used by a large number of people for many different purposes.

4.7 The HELP function -- H, ?, HELP, FUNC ?

By pressing the HELP key on a NOTIS terminal, or by giving the HOME command H, (the equivalent command on non-NOTIS terminals is FUNC ?) you get access to the lists of available commands and functions in PED. These HELP lists will be different, depending on whether you are working from a NOTIS terminal, or not.

Home commands:

H

or

?

The function key commands:

HELP

or

FUNC

?

Navigate through the HELP lists with the cursor keys :

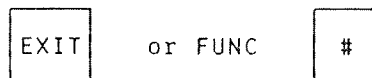


or by calling the lists up on the screen individually with the aid of the number keys, as explained in the online HELP information. An overview over the HELP structure is given in appendix D.

Due to the diversity of characters that exist on the keyboards in use in the various countries where PED is run, commands as given in this manual may in certain cases not conform with the characters on your particular keyboard. To find the correct key you may have to consult an ASCII table for your keyboard, and compare it to the ASCII table in appendix D.

4.8 Exit from PED - E, EXIT, FUNC

When you have finished your work in PED, and stored your text in the file called <file name> again, press:



If you are in HOME position you can also use the direct command



As in the READ FILE command, the system will prompt

Save edited text to <DEFAULT-FILE> (Y/N)?

if text has been added or changed after you last saved it with a WRITE FILE command.

If you have typed some text into the buffer without creating a file, PED will not accept the EXIT command and will give a warning and demand an answer:

The text is modified! Do you want to store it (Y/N)?

This question will have to be answered with Y or N! To return to COMMAND position you can input HOME:



For each region you have changed, PED will ask if you want to write it back to file.

The system will confirm the exit with an erase-page and the text:

PED:EXIT ND Program Editor
@

4.9 Activate PERFORM - FUNC \$, SHIFT + EXIT, C

The subsystem PERFORM can be activated from PED by the following commands:

SHIFT +

EXIT

or FUNC

\$

or the home command

C

The command activates a program called PERFORM, which is a SINTRAN Utility Program. When this command is given, the system will respond with:

Activate: PERFORM,,\ <current file-name>

The command activates PERFORM with your current default file, unless the main file-name is specified in Menu no. 1, in which case the main file is used instead.

The PERFORM starts when you confirm the response with ↵:

Activate: PERFORM,,\ <current file-name>↵

It is possible to edit the command string before starting the perform subsystem with the ↵. The relevant changes are:

- The first parameter after the perform call is default macro file PERFORM-LIBRARY:MCRO. If you want a macro on another macro file, the name of that file has to be input between the two comma signs.

- The \ will be interpreted by PERFORM so that it will ask for a macro name. You may therefore substitute it with a macro name. PERFORM will then not ask for a macro name, but use the one you input.
- If the file displayed is not the one you wanted to use in the PERFORM macro, it can be changed.

When you type `↵` the whole line is used as a command to the PERFORM system.

If you try to activate PERFORM without having stored all the edited text with a W command, the system will prompt

Save edited text to <FILE-NAME> (Y/N)?

The command transfers control from PED to PERFORM. PERFORM will ask you some questions, create a mode file and execute it. It is also possible to activate PERFORM directly from SINTRAN. For more information on the PERFORM subsystem, see the SINTRAN III Utilities Manual ND-60.151.

4.10 Activate the Text Formatter NOTIS-TF - SHIFT + PRINT, FUNC ", J

The Text Formatter NOTIS-TF is described in NOTIS-TF Reference Manual - Text Formatter ND-63.007. With the PRINT key in shift position, you activate the Text Formatter NOTIS-TF.

SHIFT +

PRINT

or FUNC

"

or the home command

J

When this command is given, the system will respond with:

ACTIVATE NOTIS-TF-xxx-J <current file-name>

'xxx' means the language version of the Text Formatter, and 'J' indicates the version number.

The J command activates NOTIS-TF with your current default file, unless the main file-name is specified in Menu no. 1, in which case the main file is used instead.

The formatting starts when you confirm the response with `↓`:

`ACTIVATE NOTIS-TF-xxx-J <file-name>↓`

If you try to activate the NOTIS-TF without having stored all the edited text with a W command, the system will prompt

`Save edited text to <file-name> (Y/N)?`

The command transfers control from PED to NOTIS-TF. Once the formatting is done, the system automatically enters INSPECT mode in NOTIS-WP where the formatted document can be scanned for accuracy. It is not possible to return directly to PED from NOTIS-TF. For information on the INSPECT mode, see NOTIS-WP REFERENCE MANUAL - EDITOR ND-63.002.

If you want to format a file with another name than main/default <file-name>, it is also possible to call NOTIS-TF up directly from SINTRAN. See the NOTIS-TF Reference Manual - Text Formatter ND-63.007.

CHAPTER 5

KEYS USED TO NAVIGATE

5 KEYS USED TO NAVIGATE

5.1 Move the cursor one line up - ↑

Pressing the up-arrow:



will move the cursor up one line, ie., to the current character position but on the line above. If the up-arrow is used in "HOME" position, the window is scrolled a number of lines according to the value of the parameter "vertical step in %" in Menu no. 1 and the number of lines in your current window. If the arrow is pressed repeatedly in a quick sequence, PED will compute how many lines you want to go, and only display the final window.

5.2 Move the cursor one position to the left - ←

Pressing the left arrow:



will move the cursor one character position towards the left.

In "HOME" position this command will move the window a number of columns to the left determined by the width of your window, and the "horizontal step in %" parameter in Menu no. 1.

5.3 Move the cursor to and from HOME position - HOME key or slanted arrow

Pressing the slanted arrow:



will move the cursor to the HOME position in the upper left corner of the screen.

If the cursor is already in HOME position when you press this key, your command will move the cursor back to the character position it was in when you last moved it HOME. If you have not used this region earlier, or used a first-page command, the cursor will be placed on the first line in the first column displayed.

5.4 Move the cursor one position to the right - →

Pressing the right arrow:



will move the cursor one character position towards the right.

If the command is used in "HOME" position, the window is scrolled "horizontal step in %" of the width of the window, to the right.

5.5 Move the cursor one line down - ↓

Pressing the down-arrow:

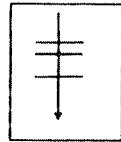


will move the cursor one line down, i.e., to the current character position but on the line below.

In "home" position the command will scroll the current window ("vertical step in %" number of lines in window) down. If repeated quickly, the number of entries are counted, and only the first, second and last window are displayed.

5.6 Move the cursor down five lines - LF

Pressing the arrow with the three bars:



or <J>

will cause the cursor to move five lines down. This is a quicker way of moving than pressing the ↓ five times.

The key can be used both in HOME position and in the text. On some terminals this key is marked with LF, LINE or LINE FEED.

5.7 Move the cursor to the left margin on the next line - ↵, CR, RETURN

Pressing the ↵ key



or



or



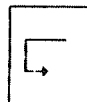
or <M>

will cause the cursor to move down to the first character position on the next line, or if left margin is set, to the left margin on the next line.

If append mode is active, the key will also cause an empty line to be inserted below the current line.

5.8 Move to the first character on the next line - FUNC ↵, FUNC <M>

Pressing the next line key:



or FUNC ↵ or FUNC <M>

will cause the editor to be placed on the first character position of the next non-empty line in the region. If you are at the end of current region, the next line command will stop at the line after the last line already used. The command is most useful if you are editing heavy indented programs.

Example:

If you have some program code with a lot of indentation, it may look like this

```

1  begin
2    a := 1;
3    while a > 0 do
4      begin      ↑
5        ↑  if (x + y) = 1 then
6          begin
7            if z * r then a := 0
8              write (output, 'big test');
9          end;
10     end;
11 end;
```

In this program the cursor is placed in PED in line 3, and then the



is activated. The new position of the cursor is determined by the first non-space character position in the next non-blank line. In this example this is the first character on the 4th line.

5.9 Move the cursor to the previous tabulator stop - ←, <U>, <Y>

Pressing the lower left arrow on the arrow pad:



or <Y> or <U>

will move the cursor one tabulator stop towards the left.

5.10 Move the cursor to the next tabulator stop - →, <T>, <I>

Pressing the lower right arrow on the arrow pad:




or <T> or <I>

will move the cursor one tabulator stop towards the right (see 4.6).

5.11 Move the cursor to the previous secondary tabulator stop - SHIFT + ←, FUNC V


Pressing the shift and the lower left arrow on the arrow pad:

SHIFT +  or FUNC V

will move the cursor one secondary tabulator stop towards the left.

5.12 Move the cursor to the next secondary tabulator stop - SHIFT + →, FUNC K


Pressing the shift and the lower right arrow on the arrow pad:

SHIFT +  or FUNC K

will move the cursor one secondary tabulator stop towards the right.

5.13 Search for <string> - SHIFT + F7, FUNC <G>, G

The command is used when you want to find a word or an expression (a string) used in your buffer, to check or modify it. The HOME command G is identical in function to SHIFT + F7.

SHIFT +  or FUNC <G>

When you press the SHIFT key and F7 you will get the following text in the screen HOME position:

- GET-STRING:

and your answer can be either:

1) GET-STRING:<string>.]

where <string> is the expression you want to find

or

2) GET-STRING:<string>↓

When you give ↓ in example 1, the system will be ready to search for <string> throughout the whole buffer.

When the ↓ is pressed after <string> in example 2, PED will ask for AREA: in the upper, right corner of the screen. This means that it is possible to tell PED to search for <string> only within a given part of your buffer. Now indicate where this area can be found by typing in the first line and the last line of the area, in the following manner:

- AREA: <from line>:<to line>
- or <from line.column>:to line.column>
- or <region.from line.column>:to line.column>

Example:

GET STRING: <ped>↓

AREA: 10:260.↓

or

GET STRING: <ped>↓

AREA: 10.5:260.20.↓

The search for <string> starts in the current screen window, and the system indicates

SEARCHING FOR "string"

in the HOME position. If you want to continue to look for further occurrences of the same string, you use the F7 key explained below. If the whole region is being searched, PED will carry out the search until it reaches the end of the buffer. It will then jump to the first line of the first screen window, and continue with further prompts from you until it reaches the point where the search was initiated.

Each time <string> is found, the system positions the cursor at the first character in <string> and waits for you to decide whether you will edit it or simply go on.

The search can be continued in this manner until PED replies

"string" NOT FOUND

which means that PED has searched through the search area, and that it can not find any "string" between the last occurrence and the position you initiated the search. If you want to restart the search, you may press F7 once more.

You may interrupt the search at any time by pressing the HOME key.

The SHIFT F7 (search for <string>) function will take into consideration the value of CASE SIGNIFICANCE in Menu no. 2. In other words, if the value of CASE SIGNIFICANCE is YES, SHIFT F7 will only find the exact replica of <string>.

Example:

GET-STRING: ped will not find 'PED'.

If CASE SIGNIFICANCE is NO, the system will find both upper-case and lower-case versions of your string and will therefore pick up ped, PED and Ped.

Note that SHIFT F7 will 'remember' the last string searched for, which enables you to restart the search for a string again much later. The absolute condition here is that you have not initiated a search for another string, or used the SUBSTITUTE command in the meantime.

5.14 Continue searching - F7, FUNC \, <G>

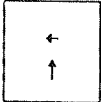
If you press the F7 key, the system will continue the search initiated with SHIFT F7 as explained above. F7 can be used either in HOME position or in the text.

 or FUNC \ or <G>

F7 'remembers' the last <string> searched for, and will resume the search for this last <string> requested if you press it again after the search has terminated or has been interrupted with \. If no search string is defined, the F7 key will act like the SHIFT + F7.

5.15 Scroll display window up - SCROLL-UP, FUNC ↑

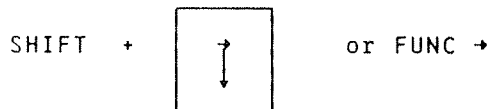
Pressing the topmost left double arrow:

 or FUNC ↑

will scroll the display window 'n' lines up, enabling you to re-examine 'n' lines of text already written, but no longer visible on your screen because it has scrolled out of the display. The value of 'n' will be computed as the percentage given in Menu no. 1 (vertical step), of the height of the window. This percentage is default 93. This means that if you have a window of 10 lines, the scroll up key will move you 9 lines up. If you only have one window on the NOTIS-2200/9 terminal, you have 21 lines. This will give a scroll of 19 lines.

5.16 Scroll display window right - SHIFT + SCROLL-DOWN, FUNC →

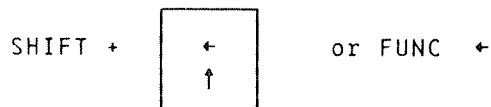
Pressing the topmost right double arrow in shift position:



will scroll the display window 'n' positions to the right, thus displaying text which lies beyond the right hand side of the window. The number n is computed from as a percentage of the window width. The percentage is given in horizontal step in Menu no. 1. Default value is 50, giving a horizontal scroll of half the window width.

5.17 Scroll display window left - SHIFT + SCROLL-UP, FUNC ←

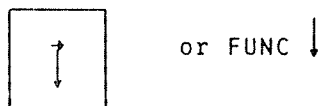
Pressing the topmost left double arrow in shift position:



will scroll the display window 'n' character positions towards the left again, thus resetting the display window in the usual editing position. The value of 'n' will be the percentage of the width of the window given in Menu no. 1 (horizontal step).

5.18 Scroll display window down - SCROLL-DOWN, FUNC ↓

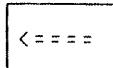
Pressing the topmost right double arrow:



will scroll the display window 'n' lines down. The value of 'n' will be computed as the percentage given in Menu no. 1 (vertical step), of the high of the window. This percentage is default 93. The command enables you to see existing text which is 'n' lines further on in your work area.

5.19 Move to previous area, menu etc. - <===, FUNC B

Pressing the wide left arrow:



or FUNC B

will enable you to :

- a) move to a previous parameter in a command with several parameter prompts.
- b) move to previous menu.
- c) move to previous HELP chapter when in HELP mode.
- d) move to the end of previous word if no text is marked, and you are in the text area.
- e) mark the previous line as FIELD if an area is marked.

Example:

READ FILE: <file name>↓

INSERT BEFORE:↓

AREA: (and you stop there!)

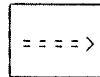
When you reach AREA: you discover that you have forgotten to write in the <line number> before which you want your text to be inserted. You then press

<====

The cursor jumps back to INSERT BEFORE: and you can correct your omission, press the cursor down arrow



or

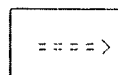


again to get the AREA: parameter once more, and now input <from line>:<to line> ↓ to finish the command.

You may also move back to a previous command parameter with the cursor up arrow

**5.20 Move to next area, menu, etc. - ===>, FUNC N**

Pressing the right wide arrow:



or FUNC N

will enable you to move to the next area or menu in exactly the same way as you move to the previous ones with the left wide arrow explained above.

5.21 Move the cursor to the beginning of this line - SHIFT + <===, <R><R>

Pressing the left wide arrow in shift position:

SHIFT + <=== or <R><R>

will move the cursor left, to the first character position on the current line. See also the <R> command.

5.22 Move the cursor to the end of this line - SHIFT + ===>, <F><F>

Pressing the right wide arrow in shift position:

SHIFT + ===> or <F><F>

will move the cursor to the character position after the last character you have input on the current line. See also the <F> command.

5.23 Display the first text window - F

Pressing the HOME command F will bring up on the screen the first screen window of edited text in the current region.

F

5.24 Display the last text window - L or \$

Pressing the L will bring up on the screen the last 10 lines of edited text in your buffer, and approximately 10 blank lines, depending on the terminal you are working on.

L or \$

When you now press the HOME key to move the cursor down into the text, the cursor will automatically be positioned one line below the last line of text.

5.25 Display the next text window - N

Pressing the N will bring the next screen window of edited text in the work area up on the screen.

N

The N command is different from the HOME command ↓, in that the N will scroll the full window height, and will not be affected by the "vertical step" parameter in Menu no. 1. If a rapid sequence of N's are input, PED will count them, and will only display the final window.

Example:

You are working in the screen window containing lines 1342 - 1362. You go into HOME position, press N, and thus bring up on the screen the window containing lines 1363 - 1383.

5.26 Display the previous text window - P

Pressing the P will bring the previous screen window of edited text in the work area up on the screen. The comments on the N command also affects this command.

P

Example:

You are working in the screen window containing lines 1342 - 1362. You go into HOME position, press P, and thus bring up on the screen the window containing lines 1321 - 1341.

5.27 Advance the displayed text by 5 lines - plus

The + command causes the window to be advanced by 5 lines of text. This is useful if you want to check a block of text, a part of which is found at the top of the next screen window.



5.28 Rewind the displayed text by 5 lines - minus

The - command causes the window to be rewound 5 lines. It thus has the opposite effect of the + command described above, and gives you the opportunity to review a few lines of text from the previous screen window, while keeping most of the text you are currently working on visible on the screen.



5.29 Move display to <position> - M

The M command will move the text displayed on the screen window so as to make the line number indicated in <position> appear as the first text line on the screen.



The command may be given relatively or absolutely. There are thus several ways in which the M command can be given:

- 1) MOVE TO POSITION:<line number>↵
where <line number> is the number of the line in the work area that you want to be displayed on the first line of the screen,

or
- 2) MOVE TO POSITION:<+> or <-> <number of lines>↵
where + or - <number of lines> will move the display the specified number of lines forwards or backwards in the work area.
- 3) MOVE TO POSITION:<line.column>↵
where <column> will be the character position displayed in the upper left corner of the screen window.
- 4) MOVE TO POSITION:<region>↵
where <region> will be a named area in the buffer. If the area does not exist, it has to be created by putting double quotes around the name. If position in the region is not specified, you will enter the region where you left it earlier, and with a new region you will be placed at the beginning. You may also

5) MOVE TO POSITION:<region.line.column>+]
where region is the region you move to and <line.column> will be the character position displayed in the upper left corner of the screen window.

Example 1:

If <line number> is 300, lines 300 through 320 will be displayed.

Example 2:

If you are working on a screen window displaying lines 700 to 720 and you give <line number> as +15, your command will move you 15 lines forwards in the work area relative to the current window position. You will thus display the window starting with line 715.

Example 3:

You want to move to column 15 in line no. 750. You input <750.15>. Line no. 750 will be displayed as the first line of text in the screen window, and the character which occupies column 15 in line 750 will be displayed in the first character position in that line. In other words, the text will be scrolled 14 character positions towards the left.

Example 4:

If region is FILE-I0, the current region will change to FILE-I0, and the window will display that text, and the new region name, and the new lines and columns are indicated in line 2 of the screen.

Example 5:

If you specify FILE-I0.100.10, your new region is FILE-I0, and the lines displayed starts with line 100, and the leftmost column displayed is column 10.

5.30 Enter the work area at <position> for editing - "

The " command enables you to move the cursor to the exact line and column in the text area where you want to edit text.



When you type " the system will prompt:

- ENTER POSITION:

The parameter is <line number> or <line number.column> or <region.line number.column>, which can be marked with a <V> (see 'Functions without special function keys') and 'remembered' for as long as it remains the last item to have been marked in this manner. The specified position will be placed in the upper left corner of your current window.

Example:

You have a column of figures starting on line 1500, column 50. You position the cursor on line 1500, column 50 and press <V>. You then move to another window to edit text, and when you want to return to your column of figures you simply press " in home position, and then \downarrow . The system will now list out the <line number.column> marked with <V>, and another \downarrow will place the cursor at that position.

- ENTER POSITION: \downarrow 1500.50 \downarrow

You may, of course, also input the <line number.column> parameter manually, if you can remember it:

- ENTER POSITION: 1500.50 \downarrow

5.31 Move to another window - FUNC 1, FUNC 2, FUNC 3

When using the split screen facility, the way to move between the three windows is to use FUNC and the number of the window you are moving to.

Example:

If you are in region MAIN in 1. window, and you want to edit something in region FILE-IO in 3. window, you input:

FUNC

 +

3

Now your cursor will move to window 3, and the region name in line two of the screen will change from MAIN to FILE-IO.

C H A P T E R 6

KEYS USED TO MARK TEXT AREAS

6 KEYS USED TO MARK TEXT AREAS

As you will see in the top row of function keys on your TDV-2200/9 keyboard there are two keys called MARK and FIELD. These keys are used to mark portions of text to be used for one of the purposes described in the chapter on 'Keys used to operate on marked text areas', ie., text to be moved, copied or deleted.

The wide arrows

and

or

FUNC B and FUNC N

can be used to mark the previous/next occurrence of the same type as that already marked, eg., if you mark a line, the left wide arrow will move you to the next line and mark that instead. The default value of area is field. When using the left wide arrow, the cursor position will be at the end of the selected area and when using the wide right arrow, the cursor position is at the beginning of the area.

If you regret a marking, press the CANCEL key (FUNC X) to cancel it. If you then regret the cancellation, you can make the marks reappear with SHIFT + CANCEL. (FUNC Q)

Marked text will appear in inverse video on your screen. Once text has been marked you have to carry out one of the operations mentioned above, before you can input more text in front of the marked area. If you try to enter more text before or in the marked area, before you have carried out an operation on the marked text you will receive a message in the top line of the screen:

PED:Press cancel key if you want to edit/input text

6.1 Mark top left/bottom right corner of a box area - MARK, FUNC <V>

When you wish to mark a box, eg., a table or form, you may do so with:

SHIFT +

or

or FUNC <V>
or FUNC <Z>

The box has to be rectangular or square. You press MARK the first time in the upper left corner of the box, and a second time in the lower right corner of the box. The marked box now comes out in inverse video and is ready for one of the operations mentioned earlier.

If the box you wish to mark for further use is alone in the middle of your text, as is the

MARK

in this section, you may well mark it by pressing the MARK keys in the first character position of the first and of the last line in the block. You will then see that not only the block itself, but the whole length of the marked lines from the first to the last character position will be marked in inverse video. The block is then considered as whole lines and will, if moved, be reinserted in exactly the same character positions as those in which it was located in the original position.

The text will always, if such an operation is carried out with full lines, be inserted above the line on which the cursor was placed when you pressed the COPY key, and existing text will open up to make room for it.

If you want to move the block into other character positions, it is only the upper left and lower right corners that should be marked before you place the cursor in the new character position where you want the block to start when inserted.

A marked block can of course also be used in other operations than the MOVE function.

6.2 Mark this line - FIELD, FUNC F

FIELD

or FUNC F

If you press this key while the cursor is somewhere on the screen line, in any character position, the whole line from character position 1 to character position 255 (maximum line length) will be marked in inverse video and can be used in one of the operations mentioned.

If you delete a line (a field) marked in this manner, the text will be contracted so that the line below is moved one line up.

C H A P T E R 7

KEYS USED TO OPERATE ON MARKED TEXT AREAS

7 KEYS USED TO OPERATE ON MARKED TEXT AREAS

When an area has been marked either with the MARK key or with the defined FIELD keys, you may use it in either one of the operations described in this chapter. Some function commands have equivalent HOME commands. The HOME commands are generally more flexible, and they can be simpler to use if you use non-NOTIS terminals. On NOTIS terminals, the function commands are faster to use.

7.1 Delete the marked area - DELETE, FUNC D, D

Once an area has been marked, you may delete it by pressing:

DELETE or FUNC D

The marked area is then deleted, and the text contracted to fill the space where the area previously was.

An area can also be deleted by the home command D. PED will answer with

PED:Delete area:

All legal area parameters are legal inputs. If an area is marked, and you type └─┘, PED will display the area, and wait for the user to type another └─┘, or edit the displayed values.

7.2 Replace the marked area with spaces - SHIFT + DELETE, FUNC <SPACE>

You may also choose not to have the text contract when an area has been deleted. In that case you press:

SHIFT + DELETE or FUNC <space>

The area is deleted as with the DELETE key, but the vacant space in the text area is filled with spaces and thus remains open.

7.3 Copy the marked area, insert it at cursor position - COPY, FUNC C

A marked area can also be copied into another part of your text document. This is done with:

COPY

 or FUNC C

You mark the area of your choice, position the cursor in the text area at the point where you want the text inserted, and press COPY.

The text will open up to make room for the insertion. If you are copying whole lines, the text will be inserted above the line where the cursor is positioned. If you are inserting an area which does not consist of whole lines, the insertion takes place before the character position the cursor is on.

The text area you have moved of course also remains intact in its original position.

7.4 Replace cursor position area with the marked area - SHIFT + COPY, FUNC I

This command will overwrite the text at the cursor position with the text taken from the marked area. This is done with:

SHIFT +

COPY

 or FUNC I

You mark the area of your choice, position the cursor in the text area at the point where you want the text replaced, and press SHIFT + COPY.

The text area you have moved also remains intact in its original position.

7.5 Move the marked area, insert it at cursor position - MOVE, FUNC M

Another possibility is to move a text area to a new location, at the same time deleting it where it was originally input. You then press:

MOVE

 or FUNC M

and otherwise you apply exactly the same rules as in the COPY function above.

7.6 Replace cursor position area with marked area, spacefilling the marked area - SHIFT + MOVE, FUNC E

Another possibility is to move a text area to a new location, at the same time spacefilling it where it was originally input. You then press:

SHIFT + MOVE or FUNC E

and otherwise you apply exactly the same rules as in the SHIFT + COPY function above.

7.7 Insert area - I

The insert area command I is a home command equivalent to the function commands COPY and MOVE. The parameters are:

- Insert area.
- Destination.
- Delete source.

If an area is marked, this is the default for the first parameter. If no area is marked by MARK or FIELD, the two last places <V> was used is the default parameter.

The destination parameter's default value is the current cursor position.

The default answer to "Delete source" is N.

Example:

PED:Insert area: 190.22:197.34

Destination: 188.39

Delete source?: N

Now you had already marked an area 190.22:197.34 with the MARK function. You moved the cursor to the position 188.39 where you wanted to copy the area. The first N as response to the parameters indicate that you want the default parameters displayed. You are satisfied with the defaults, and you want to copy rather than move the area. Legal inputs to the last question is Y, N, N, where N means N. While you are editing parameters, HOME will interrupt the command.

7.8 Cancel the current area marking - CANCEL, FUNC X

If you have marked an area and then decide not to do anything with it after all, you may cancel the marks by pressing:

CANCEL or FUNC X

The cancel command can also be used to recover the old version of a text region after a DELETE, COPY, or MOVE command on a marked area is executed. If there is no marked area when you press the CANCEL key, the command will undo the editing performed on the current line (whether this was replacing, adding or deleting text).

If there has been no editing on the current line, the command will cause the very last line deleted with the F1 key, or a marked area deleted with the DELETE key, to be restored and inserted above the current line.

If no lines have been deleted with F1, there is no marked area and no editing performed on the current line, the CANCEL key will cause a blank line to be inserted at cursor position. This is rare, however, since it is most probable that you have used the F1 key at some point in your text.

7.9 Re-mark the last marked area - SHIFT + CANCEL, FUNC Q

You may cancel a marked area, and then discover that you want to do something with it after all. You may then re-mark the same area, ie., the same line and column numbers and area type, without having to go looking for it. Press:

SHIFT + CANCEL or FUNC Q

and the area will be re-marked for a MOVE, COPY or DELETE operation. However, if you have for instance inserted a line before the marked area in the meantime, you will not get the same text remarked.

7.10 Convert the marked area to lower case characters - F6, FUNC L

If an area has been marked with one of the MARK functions described under 'Keys used to mark text areas', the marked area may be converted to lower case with

KEYS USED TO OPERATE ON MARKED TEXT AREAS

F6

or FUNC L

If there is no marked area, the function causes the remainder of the current line, ie., all text between cursor position and the end of the line, to be converted to lower case.

7.11 Convert marked area to upper case characters - SHIFT + F6, FUNC U

If an area has been marked with one of the MARK functions described under 'Keys used to mark text areas', the marked area may be converted to upper case with

SHIFT +

F6

or FUNC U

If there is no marked area, the function causes the remainder of the line, ie., all text between cursor position and the end of the line, to be converted to upper case.

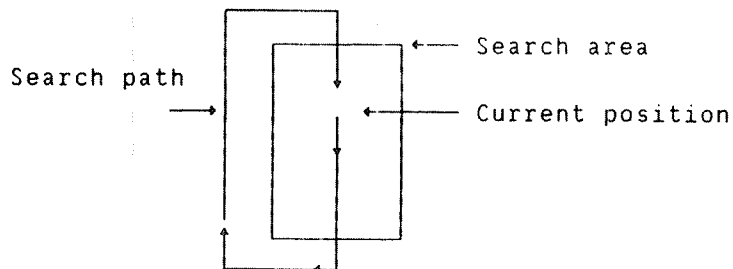
7.12 Substitute <old string> with <new string> - S

This command allows you to automatically replace one string within the text with another.

S

The substitutions may be carried out throughout your buffer, or only in a given AREA.

The search for the <string> to be substituted will be carried out starting at the current text window, and proceed through the chosen search area. If no area is specified, the whole region is searched. When the search has come to the end of the search area, PED will continue at the beginning if current position is not at the beginning.



Pressing the character S will activate the SUBSTITUTE command. The format of the command is:

```
SUBSTITUTE: <old string>↓      WITH: <new string>↓
            AREA:<from>:<to>    MANUAL CHECK: <yes or no>
```

where:

- <old string>: is the string to be replaced.
- <new string>: is the replacement string.
- AREA:<from>:<to> is the part of the text where you want the substitutions to be carried out.

If you have marked the AREA with <V>, you just input ↵.

- <yes> / <no>: is input Y or N.
↵ will be taken as Y.

The AREA: parameter is optional and if the <new string> is terminated with ↵, AREA will not be asked for.

If the answer to manual check is YES, and if <old string> is found, the cursor will be placed under the first character in the string, and PED will wait for a decision as to whether or not the <old string> should be replaced with the <new string>.

The following decisions are possible:

- S or Y: Substitute and search for next <old string>.
- \: Terminate substitute command.
The cursor is placed in the HOME position again, and the number of substitutions carried out is indicated: 'n' SUBSTITUTIONS.
- Anything else: Continue to next <old string> (ie., do not substitute).

If the answer to manual check is NO, PED will carry out all the substitutions automatically throughout the text and indicate: "'n' substitutions" in the HOME position at the end of the operation.

The new string will automatically be set as the string to be searched for, if a <G> or F7 command is given afterwards to initiate a 'continued search' operation.

C H A P T E R 8

COMMANDS USED TO EDIT TEXT

8 COMMANDS USED TO EDIT TEXT

This chapter is devoted to commands used in the edit buffer to edit text. Some commands are function key commands, while others are control key commands. When a command is written <X>, this means that you have to hold the ctrl key down when you push the character inside the <>. Upper and lower case is equal when used with the ctrl key. This means that <c> and <C> has the same meaning. Some places the command FUNC is found in the text. This is used for key input of the ASCII value 37B. On some terminals this is input as <UNDERLINE> or <SHIFT UNDERLINE>.

8.1 Function key commands

8.1.1 Delete this line - the F1 key, <D><D>, FUNC <D>

Pressing the F1 key:

or <D> <D> or FUNC <D>

will cause the line on which the cursor is positioned to be deleted.

If you find that you did in fact not want to delete the line, you may restore it with the CANCEL key.

or FUNC X

The last line or area deleted will be remembered and may be restored with the CANCEL key. You may therefore insert a line at one or several different locations in the text, by deleting it with the F1 key and restoring it with the CANCEL key.

You may similarly insert a block of whole lines at several different locations in the text by first marking the block of lines, deleting them with the DELETE key and then restoring them at the required locations with the CANCEL key.

When a line or block is deleted in this manner, the line below it is moved up and the gap in the text filled.

When the line or block is restored with the CANCEL key, the text is opened up again to give room for it above the line on which the cursor is positioned. If you only want to restore the last deleted line, the command <W> will insert it before the current cursor line. This command can be repeated several times, and works this way as a quick copy of one line at a time.

F1 only responds while the cursor is in the text area.

If you only want to delete part of your line, see the command <D>.

8.1.2 Insert a new line here - the F2 key, <L>, FUNC <L>

When you press the F2 key:

F2 or <L> or FUNC <L>

a blank line is inserted above the line on which the cursor is positioned. This enables you to insert one or several new program lines at a given point in your file. You may press this key as many times as you wish, thus opening up the number of lines you need.

F2 responds only when the cursor is in the text area.

8.1.3 Delete character - a, DEL, <A>

Pressing the a with the slash through it:

a or DEL or <A>

will cause the character on which the cursor is positioned to be deleted. The line will be contracted and the position where the character was deleted will be filled.

Example:

You write 'wrrite' instead of 'write'. You position the cursor on one of the r's, press the delete key, the extra 'r' is deleted and the word is contracted to 'write'.

If you press Δ while the cursor is positioned after the last character on the line, it is the character preceding the cursor position which will be deleted.

Example:

You have a line of text which ends with a full stop. The cursor is positioned immediately after the full stop. If you now press Δ , the full stop will be deleted.

If you use the Δ key while the cursor is placed on an empty line, or on the right hand side of the text, PED will then give a bell, and position the cursor immediately behind the last character on the line, or in position 1, if the line is empty.

8.1.3.1 Alternative delete mode

The function of the delete key is affected by a flag which is set by the command FUNC DEL (FUNC Δ). This is called alternative delete mode. If the indicator *d is set on the status line when this mode is on.

The change in function of Δ does not change the function of <A>.

The alternative delete mode is active when the last you did was typing text, and will then delete the last typed character, and move the cursor one step to the left. If you position the cursor with one of the arrows, or do anything but type text, the alternative delete mode will be passive until you type a character again.

- If expand mode is off, text to the right on the same line will not be moved.
- If expand mode is on, the text to the right is moved one character position to the left, following the cursor.

Example:

Alternative delete mode is on, and expand mode is on.

Before **␣** typed (last key typed is the e) the line looks like:

```
procedure readfrome_file (var inchar : char);
```

After the **␣** is typed the line is:

```
procedure readfrom_file (var inchar : char);
```

If the flag is not set, or if you input <A> instead, the f is deleted. This will also happen if you position the cursor prior to using the **␣** key. This is how it will look like:

```
procedure readfromeile (var inchar : char);
```

If expand mode is off, and you type the **␣** key, the line will look like:

```
procedure readfrom_file (var inchar : char);
```

As you see, the text to the right is not moved. This is particularly valuable for editing programs if you do not want to change your comments on the right side of the screen. This feature will also make editing of tables easier.

8.1.4 Split the line at cursor position - F5, FUNC Y

By pressing

F5

or FUNC Y

you cause the line on which you are working to be split at cursor position, and the part of the line which is behind cursor position to be moved down to the next line.

8.1.5 Link this line to the preceding line - SHIFT + F5, FUNC Z

By pressing the same key in shift position

SHIFT + F5 or FUNC Z

the line the cursor is on will be appended to the line above the cursor.

8.1.6 Move the cursor to the beginning of this line - SHIFT + <==, FUNC <R>, <R><R>

Pressing

SHIFT + <== or <R><R> or FUNC <R>

moves the cursor left, to the first character position on the current line.

8.1.7 Move the cursor to the end of this line - SHIFT + ==>, FUNC <F>, <F><F>

Pressing the right wide arrow in shift position or <F><F>:

SHIFT + ==> or <F><F> or FUNC <F>

moves the cursor to the character position after the last character on the current line.

8.2 Control key commands

8.2.1 Accept control character - <O>

The command allows any edit or control character to be accepted as a normal character.

Pressing <O> followed by a control character causes PED to insert the control character in the text at the current cursor position. Control characters will be written out on the screen as '&'. Their values can be verified with <V> while the cursor is positioned at the &, or with the HOME command V. The users are warned that PED will expand all <I> (ASCII 118) on input from file unless a TAB- command is placed on the first line of the file.

This function should not be used in text files, since the control keys may be taken as commands to either NOTIS-TF or a printer.

8.2.2 Undo editing on this line

The <W> command works like CANCEL, but limited to one line at a time. The functions are:

- Restore current line to original contents when the line is changed.
- REstore last line deleted by F1 or <D><D> when current line is not changed.

Note that <W> does not restore lines deleted as FIELDS.

<W> may also be used as a HOME command, to repeat the last command given in the HOME position. This function is attractive when you have typed the file name wrong in a read or write command. The <W> will recall the file name ready for you to correct it.

8.2.3 Copy one character from previous to current line - <C>

The <C> command is used to copy one character from the previous non-empty line into the current line. Position the cursor on the current line, directly under the character to be copied. Press <C>. The character will be copied into the current line at the cursor position, and the cursor moved one position to the right.

8.2.4 Copy one character from next to current line - <N>

The <N> command is used to copy one character from the next into the current line. The procedure is the same as for <C> above, except that the cursor must be positioned directly above the character to be copied.

8.2.5 Move cursor forwards to specified character on the line - <F>char

The <F>char command moves the cursor forwards to the first occurrence of 'char' on the current line.

Some special characters:

<F> The cursor is moved to the position after the last significant character on the current line.

↵ the cursor is moved forwards to the right margin.

When editing in the command line in HOME position, <F><F> moves the cursor to the end of the command parameter.

8.2.6 Move cursor backwards to specified character on the line - <R>char

The <R>char command moves the cursor backwards to the first occurrence of 'char' on the current line.

Some special characters:

<R> the cursor is moved to the first character position on the current line.

↵ the cursor is moved backwards to the left margin.

When editing in the command line in HOME position, <R><R> moves the cursor to the beginning of the command parameter.

8.2.7 Copy characters from previous line to current line - <P>char

The <P>char command enables you to copy into the current line all characters from the previous non-empty line that are to the right of cursor position, up to and including 'char'.

Some special characters:

<P> all the characters between cursor position and the end of the line are copied into the current line.

↵ all the characters between cursor position and the end of the line are copied into the current line, and the cursor moved to the beginning of the next line.

<I> or <T> all the characters up to the next tabulator position are copied into the current line.

→ or TAB all the characters up to the next tabulator position are copied into the current line.

SHIFT + (→) or FUNC K all the characters up to the next secondary tabulator position are copied into the current line.

The line may be restored by CANCEL or <W>.

8.2.8 Copy characters from next line to current line - FUNC <P>char

The FUNC <P>char command enables you to copy characters from the next line into the current line. It is otherwise carried out in exactly the same manner as the <P>char command above.

8.2.9 Delete characters up to and including char - <D>char

The <D>char command enables you to delete parts of a line of text, or the whole current line.

If char is:

<D> The whole line is deleted and the text below moved one line up to fill the gap.

↵ The line is deleted from cursor position to the end of the line.

<R> The line is deleted from the beginning of the line to but not including cursor position.

<I> or <T> or TAB or → |
The line is deleted from cursor position to the next tabulator position.

<U> or <Y> or BTAB or |←
The line is deleted from cursor position to the previous tabulator position.

FUNC K or SHIFT → |
delete characters from cursor position to next secondary tabulator stop.

FUNC V or SHIFT |←
Delete characters from the previous secondary tabulator stop to cursor position.

ordinary character

Delete up to and including first character equal to the character.

A line deleted with <D><D> can be restored with CANCEL or <W>. It is then restored on the line above cursor position.

Have you deleted inside a line, it can be restored if you have not moved out side the line.

C H A P T E R 9

LINE MODE EDIT

9 LINE MODE EDIT

Although this editor is best used on a terminal with cursor addressing capabilities, it can also be used on "teletype" style terminals, and on teletypes. The editor is also set in this mode if the VTM-table contain some serious error. VTM will continue to ask for a terminal type until the type specified is found on the file (SYSTEM)DDBTABLES-C-x:VTM. (The letter x may change.) The line mode may be used on all alphanumeric teletype-compatible terminals. To enter line mode from page mode, use the command #.

The editor is in many ways similar to the QED standard ND editor (see the QED Users Manual ND-60.031), but is more advanced than this. Most of the HOME commands from page mode (normal running mode) can be used. They are not immediate response commands however, but are of a "SINTRAN MATCH" type. Some commands are particular to this mode, and these are explained in more detail below.

9.1 Shadow line

The shadow line is used for supporting safe editing and fast input of similar lines. The content is different in the two modes:

Input mode: The previous line is copied into the shadow line.

Edit mode : The old content of the edited line is copied into the shadow line before edit is performed.

When the INSERT-LINE command is given, the initial value is the following line, not the previous. Some of the commands use the shadow line. If the terminal in use is a VDU, the shadow line will be displayed in edit mode.

9.2 The commands

The commands are listed with the QED equivalent inside { } parenthesis, if a QED command exists. The equivalent page mode command is displayed inside [] brackets. Parameters are inside <> brackets. If they are also inside () brackets, they are optional parameters.

PAGE-MODE:<terminal type>.

The command is used to return to page mode edit. If the terminal type specified does not permit page mode editing, the message:

Terminal is not defined with cursor control

will be given, and the editor will continue in line mode. If you only type page-mode,] the editor will test if your terminal type allows for cursor control, and if so, return you to page mode. If you are aborted, and use the @CONTINUE command, PED will restart in page mode if possible. You may then use the # command to enter line mode.

LIST-AREA <area>

{L} This command will display all lines as defined in the area parameter. If no area is specified, current line will be displayed.

NEXT-LINES <number of lines>

{N} Displays the next "number of lines". Current line is updated to the last displayed. Default number of lines is 1.

PREVIOUS-LINES <number of lines>

{P} Displays the previous "number of lines" in inverted order. Current line is the last displayed. If no parameter is given, previous line is displayed.

Example:

```

PED: list-area,
34>      end;
PED: previous-lines 5,
33>      end;
32>      count := count + 1;
31>      end;
30>      k := k + prim;
31>      flags [k] := false;
PED: list-area,
31>      flags [k] := false;

```

LINE-NUMBER-DISPLAY NO/LEFT/RIGHT

The value given determines the line numbering displayed on the terminal.

NO No line numbering is displayed.

LEFT Line numbering is displayed on the left side of the screen. The numbers are succeeded by a '>' and a space. The numbers are not a part of the text, and are only displayed as a help for the user. This numbering mode is default.

RIGHT Line numbering is displayed on the right side of the screen.

EDIT <line>

{E} is the command used to start editing a line. If <line> is omitted, the current line is edited. <L> at the beginning of the line will delete the line.

APPEND-NEW-LINES <behind line>

{A} PED is set in input mode, and all text input will be put after the specified line until a <L> is input. If the line parameter is omitted, current line is taken.

INSERT-NEW-LINES <before line>

{I} This works as APPEND-NEW-LINES, with the difference that the new lines will be put before the specified line.

DELETE <delete area>

{D} Default area is current line. See also the page command DELETE [D] on page 83.

INSERT-AREA <source area> <destination> (<delete source?>)

{I} See the page command INSERT [I] on page 85.

SUBSTITUTE: <string> with <new string> (<area>) (<manual check?>)

{SUBSTITUTE} See the page command SUBSTITUTE [S] on page 87.

CONVERT-LOWER-CASE <convert area>

See the equivalent page command F6 on page 86.

CONVERT-UPPER-CASE <convert area>

See the equivalent page command shift F6 on page 87.

READ-FILE <file> (<insert before>) (<area>)

{R} See the equivalent page command READ [R] on page 19.

INCLUDE-FILE <file> (<insert before>) (<area>)

{R} The include file will by default append the text of the specified file at the end of current region. See also the page command READ [R] on page 19.

UPDATE-FILE <file>

See the equivalent page command UPDATE FILE [U] on page 23.

WRITE-TO-FILE <file> (<area>)

{W} See the equivalent page command WRITE [W] on page 25.

WRITE-APPEND append to <file> (<area>)

{W with parameter A} See the equivalent page command APPEND [A] on page 29.

XFILE

{X} See the equivalent page command SHOW REGION [X] on page 50.

CHANGE-TERMINAL-TYPE <terminal type>

See the PAGE-MODE command, and the CHANGE-TERMINAL-TYPE command ['] on page 36.

@ <SINTRAN command>

Execute SINTRAN command. See page command [Q] on page 49.

EXIT {EX or F} See the equivalent page command EXIT [E] on page 52.

COMPILE See the equivalent page command COMPILE [C] on page 53.

TEXT-FORMATTER

See the equivalent page command ACTIVATE-NOTIS-TF [J] on page 54.

GET-STRING <string> (search area)

{L} See the equivalent page command GET-STRING [G] on page 63.

HELP Displays the line mode HELP.

SCALE {G} This command will write two full lines of text. The first will contain all tabulator positions, and the second will display the column numbering.

Example:

If you have default tabulators, it will look like this:

```
PED:scale,]
      T      T      T      T      T
123456789.123456789.123456789.123456789.123456789.12
PED: _
```

TABULATORS <tabulator positions>

{T} This command does exactly what the page mode TABULATOR [T] (page 33), and will also execute a SCALE command.

Example:

If you want to change to FORTRAN tabulators:

```
PED:tab f,]
      T      T      T      T      T      T      T      T      T      T
123456789.123456789.123456789.123456789.123456789.12
PED: _
```

VALUE <expression>

{V} See the equivalent page command EVALUATE [V] on page 43.

EDIT CHARACTERS (also in QED, but some differences should be noted)
PED will echo some of the control keys on hardcopy terminals (eg. terminal types 2 and 11).

<A>/DEL	Backspace one character (echo ^)
	Skip one character in old line (types %)
<C>	Copy one character from old line
<D>	Copy rest of old line and terminate edit
<E>	Insert string from terminal (echo <to start and> to finish)
<F>	Copy rest of old line (without typing) and terminate edit
<G>	As command: continue search (no function in edit mode)
<H>	Copy rest of old line without terminating edit
<I>	Insert spaces to next tabulation stop in new line
<J>	Terminate edit. If parameter input, ask for next (as , in page mode)
<K>	Restart the edit from the beginning (echo _)
<L>	Terminate edit and input mode in any event, except after <V>
<M>	Terminate edit (CR)(<u> </u>)
<N>	Copy next "word"
<O>x	Copy old line up to, but not including, character x
<P>x	Skip characters in old line up to, but not including, character x (% is echoed for each character skipped)
<Q>	Restart the edit from the beginning (types _)
<R>	Delete next word
<S>	Skip one character in old line (types %)
<T>	Type the rest of the old line and the new line aligned, then await more edit characters
<U>	Copy up to next tabulation stop
<V>x	Allows any edit character or control character to be accepted as normal characters (*)
<W>	Backspace one "word" (types backslash)
<X>x	Skip characters in old line up to and including the character x (% is typed for each character skipped)
<Y>	Append rest of old line to new line and edit the result
<Z>x	Copy old line up to and including character x
<J>	Abort input/command

(*) When listing lines on the terminal, QED will mark control characters in the line by inputting a & in front of each of them.

If line mode is used on a visual display terminal (a VDU), some visual screen editing is performed. Eg., when a character is erased with <A> or DEL, this is visible and no ^ is typed.

A P P E N D I X A

PED AND SYSTEM ERROR MESSAGES

ERROR TYPES

There are two different kinds of error messages, those from the file system and those from the editor (PED).

1) System messages

These error messages are described in the SINTRAN III Reference Manual ND-60.128.

2) Program Editor messages

These indications and/or error messages are given in the screen home position in connection with the commands you give.

We are listing the most usual messages, and would advise you to contact the System Supervisor if you are in doubt.

Unknown command

The message is given if an unrecognized HOME command is input.

Line indicated is greater than the last line number

The message is given if the first line number indicated in an AREA:<line nos> parameter is greater than the last line in the region.

Scratch file error - do you want to continue

The message is generally given because there is no more space under user SCRATCH.

This function cannot be used (now)

The message is given if you try to obtain a function which is authorized, but impossible; eg., giving a GET STRING command before fetching text into the work area.

"<string>" not found

The message is given both when a <string> given as parameter to a search initiated with the Get String command cannot be found in the file, and when the search has come to an end and all occurrences of <string> have been found in the search area.

The message also occurs when attempting to move (M) to a non-existing region.

Search aborted at line number "n"

The message occurs if a search initiated with SHIFT F7 or F7 is interrupted with the HOME key before the whole search area has been searched.

The text is modified! Do you want to store it (Y/N)

The message is given if you try to exit from PED or activate PERFORM or NOTIS-TF, without storing your text in a file-name. The question have to be answered before you can continue.

No area selected for this function

The message is given if you forget to define an area before you try to carry out one of the functions for which an area is needed.

Press cancel key if you want to edit/input text

The message is given if, after having marked an area for an operation, you try to enter more text or perform other editing in front of the marked area before the operation has been carried out.

Fatal editor error in routine "n", please report to ND

The message must immediately be reported to your System Supervisor, for further action. Please write down what happened before the error message was given and the name of the routine.

Library error number "n", please report to ND

The message must immediately be reported to your System Supervisor, for further action. Please write down the function which caused the error, the current run mode (page-mode, line-mode) and the current terminal type.

String is too long.

The message is given if the string length exceeds 255 characters (current maximum line length) due to input in line mode or to a substitution where the new string is longer than the old ones. The line input is lost in line mode, while in a substitution the replacements will be carried out up to the one that causes the error and leads to abortion of the operation.

Ambiguous command.

Given in line mode only, if an ambiguous command is input.

The function is not implemented in this version.

The message may be given if a function is omitted from a version of PED.

Serious error, back storage overflow.

The message will occur when the current version of PED is run, if the current use of the back storage (the scratch file) exceeds 6100 SINTRAN pages (approx. 40000 lines filled with 80 characters each).

Cannot open file

The message is likely to occur if another user has accessed the file while the current user was in SINTRAN, eg., outside the editor.

Terminal not defined with cursor control.

The message is given if the page mode command is input in line mode on a hard copy terminal or on a VDU defined without cursor positioning possibilities. Contact the System Supervisor, who will check the accuracy of the DDBxxx file installation.

Missing string/expression termination.

The message is given if one or more right parentheses are missing in a numeric expression, or if an end quote is missing in a string expression or when a search string is specified in line mode.

No help information available.

The file PED-ENG-x:HELP is missing or cannot be opened. Contact the System Supervisor, who will check if public access is set to R.

Invalid expression.

The message is given if there is an error in a numeric expression.

Attempt to create too many regions

PED can handle about 40 regions at a time. You have to delete a region in order to make room for the new one.

<region name> is already defined

You have already a region with name <region name>, so you have to find another name for your new region.

A P P E N D I X B

THE MENUS

Two menus can be called up on the screen, one by one, by pressing 1 or 2 on your keyboard while the cursor is in the HOME position. You may go back and forth between the two menus with the double arrows.

and

or

FUNC B and FUNC N

Move the cursor back into HOME position when you want to return to your text file.

The menus contain operating instructions to the program editor. You are free to modify these values to suit your particular purposes, and may do so from your terminal while you are logged in under your personal user name. Your modifications will not interfere with other users' menus. The options Y/N stand for YES or NO.

The two menus are listed below, with a short explanation.

Menu no. 1:
Editor flagsExplanations

Default file-name	The file to store (save) current text in when exiting
Main file-name	file to be sent to the formatter
Exit program name	Program to be started when exiting
Horizontal step in %	Width of horizontal scroll in percentage of window width Default 50
Vertical step in %	Depth of vertical scroll in percent of window height Default 93
Capital letters (Y/N)	Convert all letters to uppercase upon input (valid from A to Z only) Default N
Tab. spacefill (Y/N)	Fill in spaces when TAB function in use Default N
Region in window 1	Name of region in window 1 Default MAIN
Lines.Columns window 1	Window size of window 1 Default the whole text area
Region in window 2	Name of region in window 2
Lines.Columns window 2	Window size of window 2
Region in window 3	Name of region in window 3
Lines.Columns window 3	Window size of window 3

Menu no. 2:
Editor defaultsExplanations

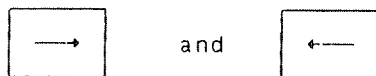
Case significance (Y/N)	YES if upper and lower case letters should be considered unequal in Get, Substitute and Sort functions Default N
Left border	Column where left border starts Default 1
Right border	Column where right border starts Default 255
Updates before save	If value is <0, the function is inactive. If value is >0, the buffer contents will be saved to the scratch file at regular intervals. Default 100
Scroll on down arrow	Start edit (NO)/Scroll display (YES) when using down arrow in HOME position Default Y
Write without trail. space	If YES, skip trailing spaces when writing Default Y

A P P E N D I X C

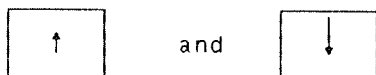
HELP Structure diagram

The HELP command initiates a tree structured help system. The help system is divided into two parts, one for NOTIS terminals and one for non-NOTIS terminals.

To navigate in the help structure you use the cursor arrows.

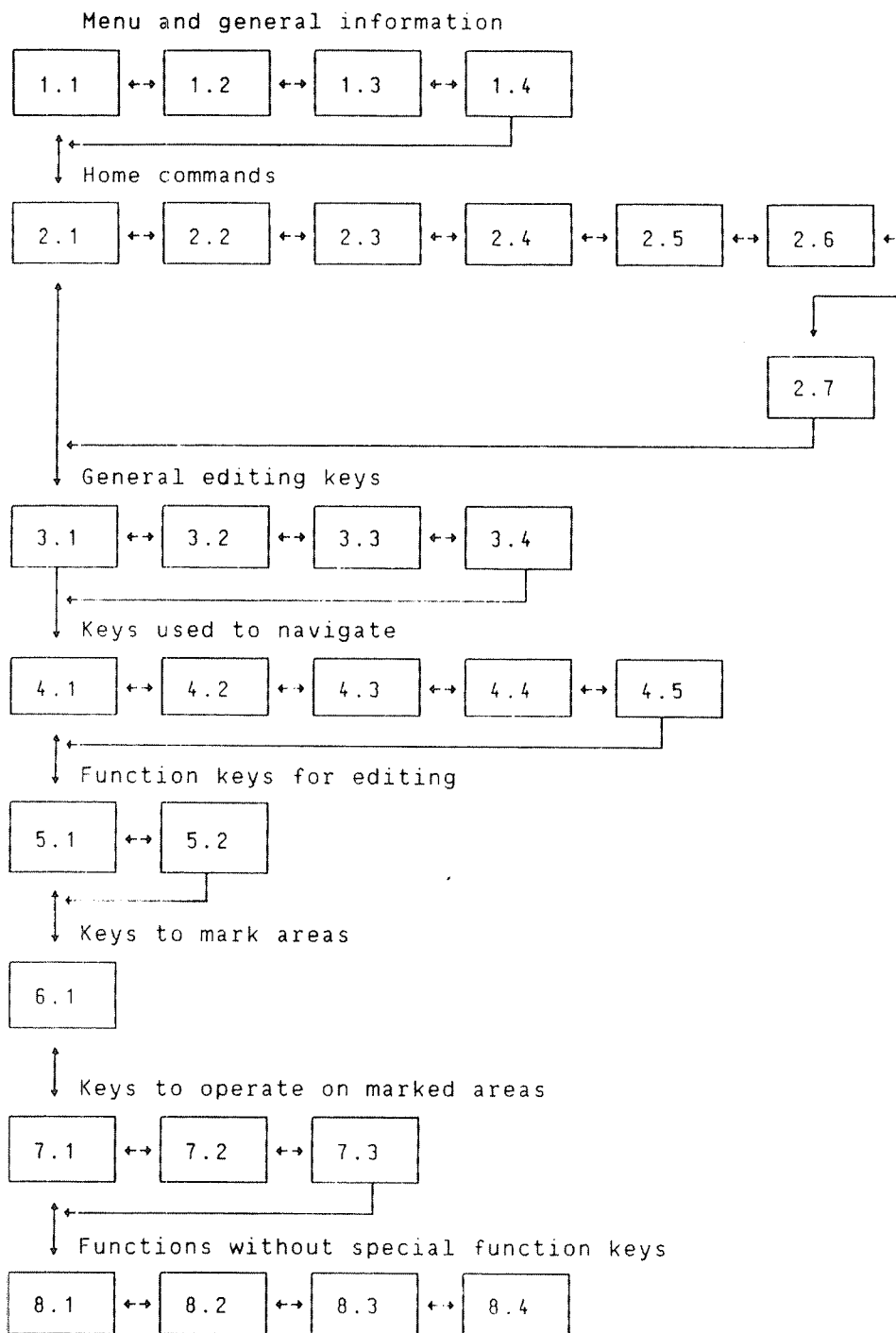


are used to navigate inside a group (horizontally), and the

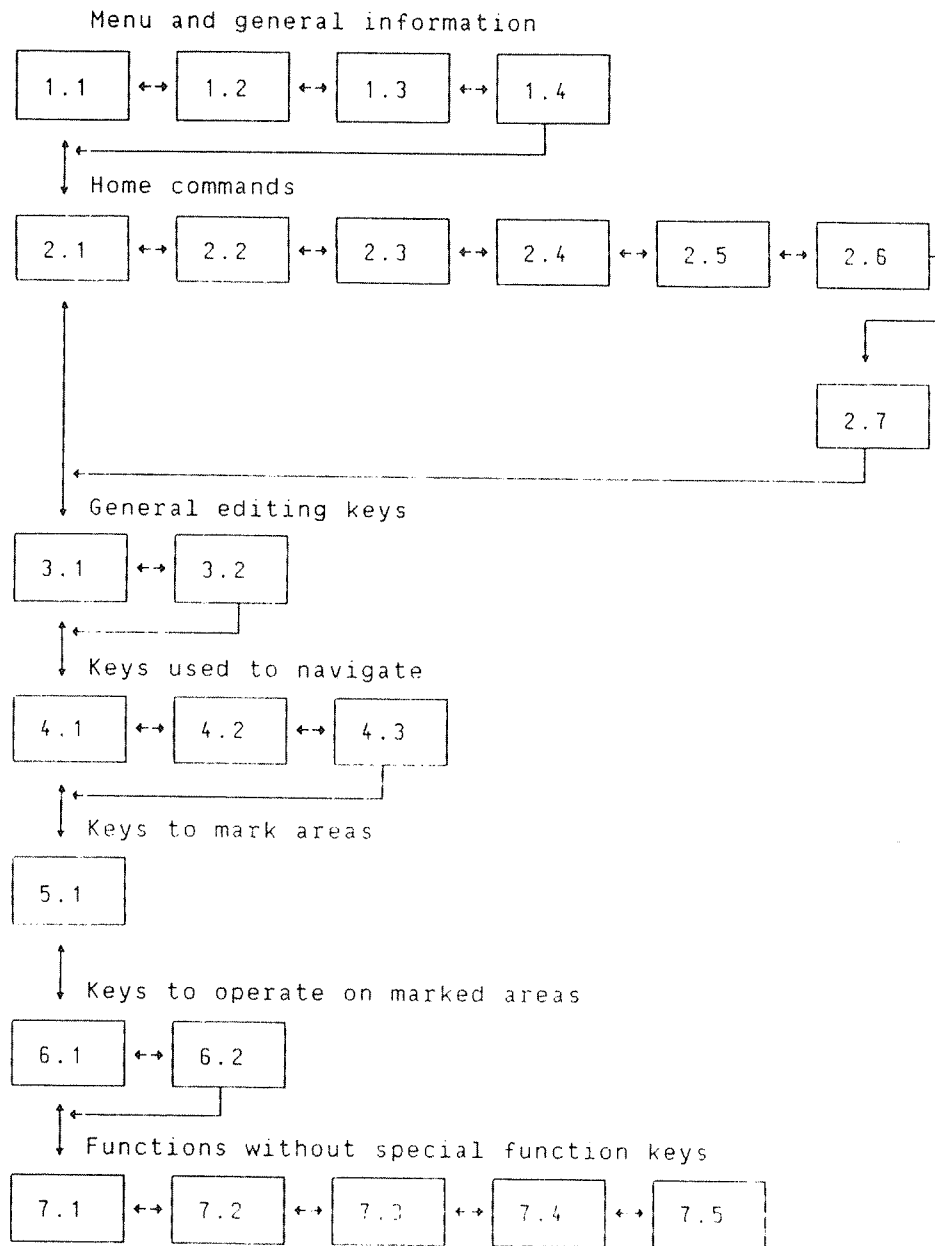


are used to navigate between the groups (vertically).

The structure of the help information for NOTIS terminals:



The structure of the help information for non-NOTIS terminals:



A P P E N D I X D

ASCII TABLE - NATIONAL CHARACTER SETS

ASCII TABLE - NATIONAL CHARACTER SETS

Some of the characters in the ANSI (ASCII) character set are defined as definable by national standardization committees. As this manual is country independent, we had to choose a character set for this manual. To help those with other character sets, we include a table for some languages:

Nationality	43B	100B	133B	134B	135B	173B	174B	175B	176B
International/US	#	@	[\]	{		}	~
French	£	à	•	ç	§	é	ù	è	…
German	#	§	Ä	Ö	Ü	ä	ö	ü	ß
British/Dutch	£	@	[\]	{		}	~
Norwegian/Danish	#	@	Æ	Ø	Å	æ	ø	å	~
Swedish/Finnish	#	@	Ä	Ö	Å	ä	ö	å	~
SDS Norwegian	§	@	Æ	Ø	Å	æ	ø	å	~

Index

all	12.
alternative deletemode	38.
append	38.
area	11.
convert	86, 87.
next	68.
previous	67.
remark	86.
ascii table	125.
borders	7, 17, 35.
left	35.
right	36.
cancel	86.
carriage return	61.
characters	
copy	96-98.
delete	98.
command	
line	6, 18.
repeat	96.
types	9.
commands	
function	9.
home	9.
homeposition	9.
concatenate	94.
continue	24.
control	
commands	10.
keys	10.
controlcharacter	95.
copy	83, 84.
characters	96-98.
country dependent	125.
cr	3, 61.
CTRL	3.
cursor	
backwards	97.
down	60.
forwards	96.
home	59.
left	59.
move	69, 95.
right	60.
up	59.
verify	47.
delete	38, 83, 91, 98.
character	92.
deletemode alternative	38.
display	
advance	70.
rewind	71.

editing	10.
undo	96.
editposition	73.
errors	
ped	109.
system	109.
evaluate	43.
exit	29, 52.
expand	37.
expressions	12.
file	
handling	15.
types	22.
find string	63.
formatter activate	30, 54.
func	3.
cr	61.
function commands	9.
functions	
link	94.
marking	77, 83.
secondary	34.
split	94.
tabulators	34.
get	
nextstring	65.
string	63.
help	
commands	51.
online	51.
structure	51.
home	3, 10.
commands	9.
insert	38, 92.
area	85.
LED display	8.
linemode	36.
lines	
delete	91.
insert	92.
restore	96.
link	94.
list regions	50.
lowercase	86.
mark	
again	86.
box	77.
cancellation	86.
convert	86, 87.
copy	83, 84.
delete	83.
line	78.

move	84, 85.
spacefill	83.
marked areas	83.
marking keys	77.
maxbyte-pointer	24.
menus	115.
next	68.
one	39.
previous	67.
two	39.
merge	20.
messages	109.
move	71, 73, 84, 85.
national characters	125.
navigation keys	59.
next line	61.
non-printable character	3.
notation	3.
notis-tf	30, 54.
manual	3.
octal	3.
ped editing	3.
perform	29, 53.
manual	3.
prerequisite knowledge	1, 2.
read	19.
plus	24.
reader	1, 2.
recover buffer	23.
redisplay screen	51.
region	20.
regions	15, 50.
related manuals	3.
reset	
CRLF	50.
LF	50.
restore	96.
return	61.
scratch file	15.
screen	4.
scroll	
down	66.
left	66.
right	66.
up	65.
search	63.
second command	6.
secondary	
tabulator	8.
tabulators	34.
sintran manual	3.
sintrancommand	49.

spacefill	83.
split	94.
screen	4.
startup	10.
statusline	6.
substitute	87.
tabulator	7, 16.
line	7.
tabulators	33, 34.
next	62, 63.
previous	62, 63.
secondary	63.
terminaltype	36.
text area	4.
textformatter	30, 54.
textwindow	
first	69.
last	69.
next	70.
previous	70.
Update	23.
updates before	23.
uppercase	87.
verify	47.
vertical split	4.
wait message	8.
windows	73.
current	51.
workarea	11.
write	25.
append	29.

SEND US YOUR COMMENTS!!!



Are you frustrated because of unclear information in this manual? Do you have trouble finding things? Why don't you join the Reader's Club and send us a note? You will receive a membership card — and an answer to your comments.

Please let us know if you

- * find errors
- * cannot understand information
- * cannot find information
- * find needless information

Do you think we could improve the manual by rearranging the contents? You could also tell us if you like the manual!



HELP YOURSELF BY HELPING US!!

Manual name: PED User's Guide

Manual number: 60.121.04

What problems do you have? (use extra pages if needed)

Do you have suggestions for improving this manual ?

Your name:

Date:

Company:

Position:

Address:

What are you using this manual for ?

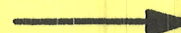
NOTE!

This form is primarily for documentation errors. Software and system errors should be reported on Customer System Reports.

Send to:

Norsk Data A.S
Documentation Department
P.O. Box 25, Bogerud
0621 Oslo 6, Norway

Norsk Data's answer will be found on reverse side



Answer from Norsk Data

Are you frustrated because of unclear information in the manual? Do you have trouble finding things? Why don't you join the Reader's Club and send us a note? You will receive a complimentary copy of the manual and answer to your comments.

Please let us know if you find errors, cannot understand information, or find needless information. Do you think we could improve the manual by reorganizing the contents? You could also tell us if you like the manual.

HELP YOURSELF BY HELPING US!!!

Manual name: The Reader's Guide
Manual number: 601.1.1.04

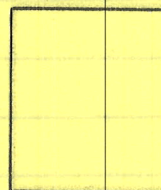
What problems do you have? (use extra pages if needed)

Answered by

Date

Norsk Data A.S

Documentation Department
P.O. Box 25, Bogerud
0621 Oslo6, Norway



Your name
Company
Address

What are you using this manual for?

NOTE!

This form is primarily for documentation errors. Software and system errors should be reported on Customer System Reports.

Send to:
Norsk Data A.S
Documentation Department
P.O. Box 25, Bogerud
0621 Oslo 6, Norway

Norsk Data's answer will be found on reverse side

Systems that put people first

NORSK DATA A.S OLAF HELSETS VEI 5 P.O. BOX 25 BOGERUD 0621 OSLO 6 NORWAY
TEL.: 02 - 29 54 00 - TELEX: 18284 NDN