



NEC

Reality v10.0

Terminal Definition Maintenance

NECSWS.COM

 **Orchestrating** a brighter world

Document control

Software Version	Document Status	Document Revision	Issue Date	Reason for Change
10.0	Published	0.1	June, 2003	Final draft

Prepared by

Name	Contact details
Pubali Pramanik	pubali.pramanik@necsws.com
Vijita Patel	vijita.patel@necsws.com

Table of Contents

Section 1: About this manual	4
1.1 Purpose of this manual	4
1.2 Conventions	4
Section 2: Introduction to Terminal Definition Maintenance	6
2.1 TDM defined	6
2.2 File structure for TDM	6
Section 3: Defining terminal types	7
3.1 Introduction	7
3.2 Invoking the TDM menu	7
3.3 Creating and modifying the TERMTYPE menu attributes	13
Section 4: Appendix A - Decimal, Hexadecimal and ASCII table	25
Section 5: Appendix B – Terminal definition examples	30
1.1 Example 1	30
1.2 Example 2	30
1.3 Example 3	31
Section 6: Appendix C – Column and row addresses	32
6.1 Column and row addresses for cursor addressing types A, L, T and H	32
Section 7: Appendix D – Creating additional definition fields for the TERMTYPE menus	35
7.1 The ELINE editor	35
7.2 Adding new definition fields	35
Section 8: Appendix E – Creating additional definition fields for the TERMTYPE menus	37
8.1 ELINE editor commands	37
8.2 Cutting and pasting with ELINE	38

Section 1: About this manual

This chapter describes the different sections of this manual and any conventions used.

1.1 Purpose of this manual


This manual tells you how to configure cursor addressing for foreign terminals to allow them to run under NEC's Reality operating system.

- **Chapter 1, About this Manual**, describes the different sections of the manual and any conventions used.
- **Chapter 2, Introduction to Terminal Definition Maintenance**, defines the Terminal Definition Maintenance (TDM) program and explains its supporting file structures.
- **Chapter 3, Defining Terminal Types**, tells you how to use TDM to define cursor addressing for your foreign terminal.
- **Appendix A is an ASCII chart.**
- **Appendix B** is a table of column and row addresses with their associated characters, for TDM-defined cursor addressing types A, H, L and T.
- **Appendix C** gives three sample terminal definitions: one for the Prism-IV and two others for foreign terminals.
- **Appendix D** tells you how to add new cursor addressing fields to the TERMTYPE menus.
- **Appendix E** is an explanation of the ELINE editor, which is used to create definition fields for the TERMTYPE menus.

1.2 Conventions

The following conventions are used in this documentation:

Conventions	Definition
Text	Bold text shown in this typeface is used to indicate input which must be typed at the terminal.
Text	Text shown in this typeface is used to show text that is output to the screen.
Bold text	Bold text in syntax descriptions represents characters typed exactly as shown. For example, WHO
Text	Characters or words in italics indicate parameters which must be supplied by the user. For example, in LIST <i>file-name</i> the parameter <i>file-name</i> is italicized to indicate that you must supply the name of the actual file defined on your system. Italic text is also used for titles of documents referred to by this document.
{ }	Braces enclose options and optional parameters. For example, in

Conventions	Definition
	<p>BLIST {DICT} <i>file-name item-id</i> {(<i>options</i>)}</p> <p>The word DICT can optionally be typed to specify the dictionary of the file.</p> <p><i>file-name</i> and <i>item-id</i> must be supplied.</p> <p>One or more single-letter options can be included, as defined for the command; these must be preceded by an open parenthesis, can be given in any order and are not separated by spaces. Any number of options can be used except where specified in text.</p>
...	In syntax descriptions, indicates that the parameters preceding can be repeated as many times as necessary.
SMALL CAPITALS	Small capitals are used for the names of keys such as RETURN.
CTRL+X	Two (or more) key names joined by a plus sign (+) indicate a combination of keys, where the first key(s) must be held down while the second (or last) is pressed. For example, CTRL+X indicates that the CTRL key must be held down while the X key is pressed.
Enter	<p>To enter means to type text then press RETURN. For instance, 'Enter the WHO command' means type WHO, then press RETURN.</p> <p>In general, the RETURN key (shown as ENTER or  on some keyboards) must be used to complete all terminal input unless otherwise specified.</p>
Press	Press single key or key combination, but do not press RETURN afterwards.
X'nn'	This denotes a hexadecimal value.
(Y/n)	This option means that you can select Y by pressing RETURN only.
(M/d)	This option means that you can select M by pressing RETURN only.
TERMTYPE	This word refers to the number entered in field 8 of the TERM or TERMINAL command.
terminal name	Refers to the item-id given to the TERMTYPE item for a particular terminal.

Section 2: Introduction to Terminal Definition Maintenance

This chapter defines the Terminal Definition Maintenance (TDM) program and explains its supporting file structures.

2.1 TDM defined

Because each terminal manufacturer has its own unique method for specifying cursor control strings and strings for performing special functions such as clearing the screen, moving the cursor back, and so on, it is difficult for an operating system manufacturer to provide tables for controlling all terminal types.

The program TDM (Terminal Definition Maintenance) allows end users to configure their own foreign terminals for Northgate's Reality operating system. Using TDM, you can define up to 43 different terminal types (TERMTYPEs 21 to 63, inclusive).

TERMTYPEs 0 to 20 are reserved by Northgate for defining native terminal types.

Once a terminal has been defined by TDM, you can make it available for use by your process by specifying its TERMTYPE number in field 8, TERMINAL TYPE, of the **TERM** or **TERMINAL** command.

2.2 File structure for TDM

Terminal definitions are stored in the CURSOR-DEPS file in the SYSPROG account.

The item-id for each item is the name of the terminal it defines, and the attributes of the item are the definition of the terminal type.

Attribute definitions are stored in the DICT CURSOR-DEFS file in the SYSPROG account. The item-id for each item is LINE*n*, where *n* is the number of the field in the TERMTYPE menu.

When you file a terminal definition by typing **FT** at one of the TERMTYPE menus, each item in CURSOR-DEFS is processed, and the resulting cursor addressing strings are loaded into Reality operating system tables.

Note

You cannot access the cursor positioning definition tables while a TERMTYPE definition is being loaded.

If you attempt to do so, Reality returns a CR/LF pair instead of the requested cursor control string.

The loading process is very brief, and the new table is available as soon as it has been loaded.

Section 3: Defining terminal types

This chapter tells you how to use TDM to define cursor addressing for your foreign terminal.

3.1 Introduction

This chapter tells you how to use the Terminal Definition Maintenance (TDN) program.

The control flow for this program is depicted in Figure 3-1.

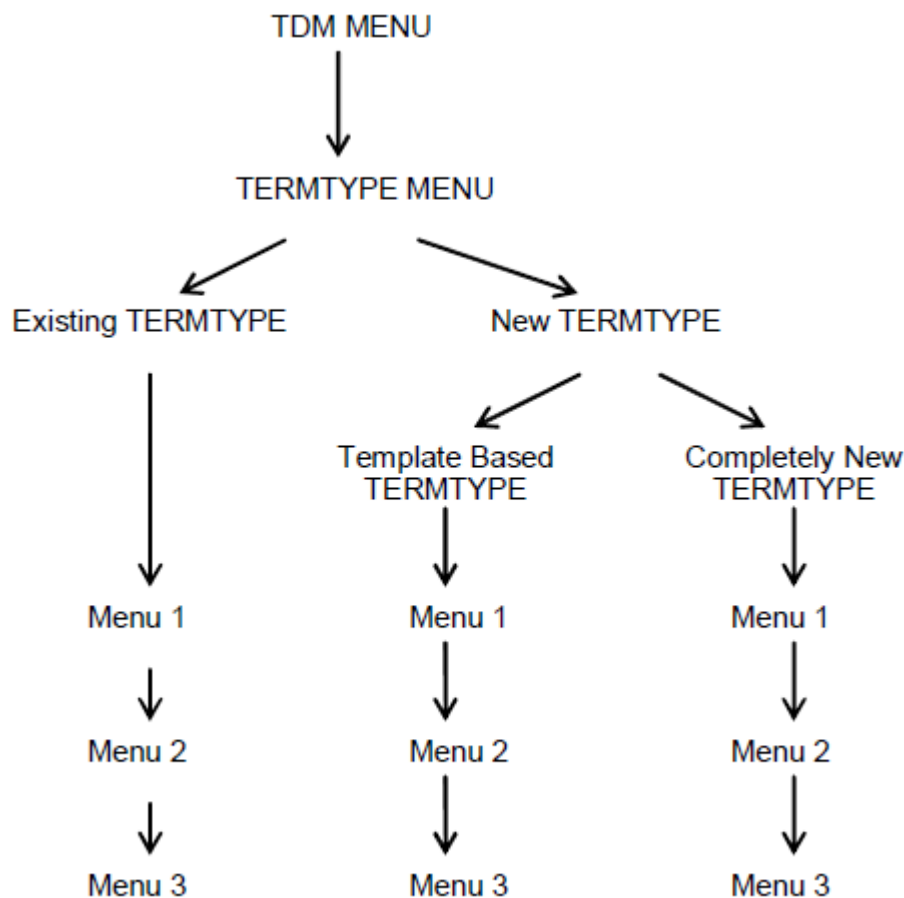


Figure 3-1: Menu control flow for TDM

3.2 Invoking the TDM menu

TDM is a cataloged DataBasic program on the SYSPROG account.

To invoke TDM, enter `TDM` at the TCL prompt (see Figure 3-2).

The program displays a list of existing terminal types (TERMTYPE) and asks you to select any TERMTYPE in the range of 21 to 63, inclusive.

```
:TDM
Terminal Definition Maintenance                10:49:49 23 JUL 1988

21 VIEWPOINT
22 HAZELTINE
23 QVT-102
24 QVT-511
25 ADM-12
26 ANPEX-210
27 5750/P99
28 PRISM-IV
29 PRISM-VIII

Input a TERMTYPE from 21 to 63, "EX" to exit, "F1 to build table or "?":29
```

Figure 3-2: Invoking the TDM menu

3.2.1 Exiting the TDM menu

To exit the TDM program and return to TCL, enter **EX** at the Input a TERMTYPE from 21 to 63, "EX" to exit, "F1" to build table or "?" prompt on the TDM menu.

3.2.2 Building a TERMTYPE table

To exit the TDM program and return to TCL, enter **F1** at the Input a TERMTYPE from 21 to 63, "EX" to exit, "F1" to build table or "?" prompt on the TDM menu.

3.2.3 Online Help messages

Online documentation is provided with the TDM software.

To access these online help screens, enter **?** at the:

Input a TERMTYPE from 21 to 63, "EX" to exit, F1 to build table or "?" prompt on the TDM menu,

Or at the

Field #,<cr> for next page, "EX" to exit, "F1" to file changes, or "?" prompt on the TERMTYPE menu.

3.2.4 Modifying an existing TERMTYPE

To modify an existing TERMTYPE, enter its number in response to the TDM menu. The TDM program asks you if you want to modify or delete the existing item.

Press RETURN or enter **M** and change the existing item.

Enter **D** to delete the item (see Figure 3-3).


```

Terminal Definition Maintenance                               10:49:49 23 JUL 1988

21 VIEWPOINT
22 HAZELTINE
23 QVT-102
24 QVT-511
25 ADM-12
26 ANPEX-210
27 5750/P99
28 PRISM-IV
29 PRISM-VIII

Modify or Delete terminal PRISM-VIII, TERMTYPE 29(M/d):M

```

Figure 3-3: Invoking the TERMTYPE menu

After you type **n** at the TDM menu, the program presents you with the TERMTYPE menu filled in for the TERMTYPE you want to modify (see Figure 3-4).

Make the necessary modifications and do the following:

- 1 Enter **F1** to file the changes, or
- 2 Enter **EX** to exit the menu at any time, or
- 3 Enter **?** to get online assistance.

```

Terminal Definition Maintenance                               10:49:49 23 JUL 1988
TERMTYPE: 31
Terminal: PRISM-VIII

1  TERMTYPE                               : 31
2  Max column, Max row                     : 131,23
3  Cursor Address type (A,L,T,H,D,X)      : A
4  @(C) cursor positioning                 : CHAR(16) C
5  @(C,R) cursor positioning              : CHAR(11) R CHAR(16) C
6  Visual attributes definition (D)        : D
7  "D" character for DIM attribute         : HEX(81)
8  "D" character for FLASHING attribute    : HEX(82)
9  "D" character for REVERSE attribute     : HEX(84)
10 "D" character for BLANK attribute       : HEX(88)
11 "D" character for UNDERLINE attribute  : HEX(90)
12 Visual attributes 'off' definition      : HEX(80)
13 Embedded visual attributes(Y/N)        : Y
14 Force vis. attr. to one column(Y/N)    :
15 Endow vis. attr. with effects(Y/N)     :

Field #,<cr> for next page,"EX" to exit,"FI" to file changes,or"?":?

```

Figure 3-4: TERMTYPE menu with characteristics for a prism 7 terminal filled in

3.2.5 Creating a new TERMTYPE

To create a new TERMTYPE, enter a number that isn't listed in the TDM menu display (see Figure 3-5).

```
Terminal Definition Maintenance                                10:49:49 23 JUL 1988

21 VIEWPOINT
22 HAZELTINE
23 QVT-102
24 QVT-511
25 ADM-12
26 ANPEX-210
27 5750/P99
28 PRISM-IV
29 PRISM-VIII

Input a TERMTYPE from 21 to 63, "EX" to exit, "FI to build table or "?" :30
```

Figure 3-5: Creating a new TERMTYPE

3.2.5.1 Verifying the new TERMTYPE number

The program asks you to verify that the number you have chosen is the one you want to use for your new TERMTYPE (see Figure 3-6).

```
Terminal Definition Maintenance                                10:49:49 23 JUL 1988

21 VIEWPOINT
22 HAZELTINE
23 QVT-102
24 QVT-511
25 ADM-12
26 ANPEX-210
27 5750/P99
28 PRISM-IV
29 PRISM-VIII

Create a new TERMTYPE 30?:Y
```

Figure 3-6: Verifying a new TERMTYPE number

3.2.5.2 Creating a new TERMTYPE name

After you verify the number of the new TERMTYPE, the program asks you for a name (see Figure 3-7).

```
Terminal Definition Maintenance                               10:49:49 23 JUL 1988

21 VIEWPOINT
22 HAZELTINE
23 QVT-102
24 QVT-511
25 ADM-12
26 ANPEX-210
27 5750/P99
28 PRISM-IV
29 PRISM-VIII

What is the terminal name for this new TERMTYPE?:PRISM-X
```

Figure 3-7: Creating a new TERMTYPE name

3.2.5.3 Selecting a TERMTYPE to use as a template

Once you specify a name, TDM prompts you to select an existing TERMTYPE to be used as a template for creating your new TERMTYPE (see Figure 3-8).

```
Terminal Definition Maintenance                               10:49:49 23 JUL 1988

21 VIEWPOINT
22 HAZELTINE
23 QVT-102
24 QVT-511
25 ADM-12
26 ANPEX-210
27 5750/P99
28 PRISM-IV
29 PRISM-VIII

Input a current terminal name or TERMTYPE to use as a template:PRISM-IV
```

Figure 3-8: Creating a new TERMTYPE from an existing TERMTYPE

The program then displays the template TERMTYPE (Figure 3-9) and waits for your input.

You can exit the menu at any time by entering **EX**.

```

Terminal Definition Maintenance                               10:49:49 23 JUL 1988
TERMTYPE: 30
Terminal: PRISM-X

1  TERMTYPE                                           : 31
2  Max column, Max row                               : 79,23
3  Cursor Address type (A,L,T,H,D,X)                 : A
4  @(C) cursor positioning                           : CHAR(16) C
5  @(C,R) cursor positioning                         : CHAR(11) R CHAR(16) C
6  Visual attributes definition (D)                   : D
7  "D" character for DIM attribute                    : HEX(81)
8  "D" character for FLASHING attribute                : HEX(82)
9  "D" character for REVERSE attribute                : HEX(84)
10 "D" character for BLANK attribute                  : HEX(88)
11 "D" character for UNDERLINE attribute              : HEX(90)
12 Visual attributes 'off' definition                 : HEX(80)
13 Embedded visual attributes(Y/N)                   : Y
14 Force vis. attr. to one column(Y/N)                :
15 Endow vis. attr. with effects(Y/N)                 :

Field #,<cr> for next page,"EX" to exit,"FI" to file changes,or"?:?

```

Figure 3-9: TERMTYPE menu with characteristics for a PRISM-IV terminal filled in

3.2.5.3 Creating a TERMTYPE without using a template

If you do not specify an existing TERMTYPE to use as a template, the program displays the first TERMTYPE menu with only the TERMTYPE information (field 1) filled in (Figure 3-10).

```

Terminal Definition Maintenance                               10:49:49 23 JUL 1988
TERMTYPE: 40
Terminal: NEWTERM

1  TERMTYPE                                           : 40
2  Max column, Max row                               :
3  Cursor Address type (A,L,T,H,D,X)                 :
4  @(C) cursor positioning                           :
5  @(C,R) cursor positioning                         :
6  Visual attributes definition (D)                   :
7  "D" character for DIM attribute                    :
8  "D" character for FLASHING attribute                :
9  "D" character for REVERSE attribute                :
10 "D" character for BLANK attribute                  :
11 "D" character for UNDERLINE attribute              :
12 Visual attributes 'off' definition                 :
13 Embedded visual attributes(Y/N)                   :
14 Force vis. attr. to one column(Y/N)                :
15 Endow vis. attr. with effects(Y/N)                 :

Field #,<cr> for next page,"EX" to exit,"FI" to file changes,or"?:?

```

Figure 3-10: TERMTYPE menu with no characteristics filled in

3.3 Creating and modifying the TERMTYPE menu attributes

This section explains how to use the three TERMTYPE menus to create or modify TERMTYPES for different vendors' terminals.

3.3.1 Menu 1

This menu defines the first 15 TERMTYPE attributes.

```

1  TERMTYPE                               :
2  Max column, Max row                    :
3  Cursor Address type (A,L,T,H,D,X)     :
4  @(C) cursor positioning                :
5  @(C,R) cursor positioning              :
6  Visual attributes definition (D)       :
7  "D" character for DIM attribute        :
8  "D" character for FLASHING attribute   :
9  "D" character for REVERSE attribute    :
10 "D" character for BLANK attribute      :
11 "D" character for UNDERLINE attribute :
12 Visual attributes 'off' definition     :
13 Embedded visual attributes(Y/N)       :
14 Force vis. attr. to one column(Y/N)   :
15 Endow vis. attr. with effects(Y/N)    :

Field #,<cr> for next page,"EX" to exit,"FI" to file changes,or"?:

```

3.3.1.1 TERMTYPE

Defines a decimal number between 21 and 63 (inclusive).

Any TERMTYPE name is valid as long as it:

- Is unique.
- Is not entirely numeric.
- Contains no blanks.
- Contains 50 or fewer characters.

The TERMTYPE is the number you type as parameter eight in the **TERM** and **TERMINAL** verbs. It is a pointer to the cursor tables used by Reality to perform input/output with the defined terminal.

3.3.1.2 Max column, max row

- Defines a decimal number pair, separated by a comma.
- The Max column is the highest numbered addressable column supported by the terminal.
- The Max row is the highest numbered addressable row supported by the terminal.
- If an application program tries to address an area greater than the maximum column or row, the program substitutes the maximum defined value instead.

For example, if a DataBasic program attempted to execute an @(85,10) statement to a terminal with a defined Max column, Max row of 79,23, TDM would substitute 79 for the 85 before generating a cursor address string.

3.3.1.3 Cursor address type (A, L, T, H, D, X)

Defines an ASCII character that determines a character generating scheme for cursor addressing.

Most cursor addressing schemes share a common element in that they emit the row and column as some form of number.

All address codes can be followed by a + to add one to both the row and column addresses before the row and column codes are calculated. This features are provided for terminals that define the upper left corner of the screen (home position) as physical location 1,1 rather than 0,0.

Use **Appendix C, Column and Row Addresses for TERMTYPES A, H, L, and T** for information on binary cursor addressing.

Each TDM cursor address type's method of emitting this number is discussed individually below.

Address type	Explanation
A	<p>ADDS. Formula is $COL=CHAR((INT(C/10)*6)+C)$ $ROW=CHAR(R+64)$ Column output is binary coded decimal (BCD).</p>
D	<p>Decimal Emits a one to three ASCII decimal character (bytes) string. May be followed by two one-digit padding specifications (for example, Dcr) to force the desired number of digits for column and row output. For example, D32 forces the output column string to be three digits and the output row string to be two digits. Output defaults to only those digits in the column or row. When padding is specified, leading zeros are added when necessary to reach the specified length.</p>
L	<p>Lear Siegler. Formula is $COL=CHAR(C+32)$ $ROW=CHAR(R+32)$</p>
H	<p>Hazeltine. Formula is $COL=CHAR(C)$ $ROW=CHAR(R)$</p>

Address type	Explanation
T	TEC. Formula is COL=CHAR(127-C) ROW=CHAR(127-4)
X	Hexadecimal. Emits a two-byte ASCII hexadecimal string.

3.3.1.4 @ (C) cursor positioning

Defines the formula used to position the cursor with a horizontal only cursor specification like DataBasic's @(I0) or PROC's T (10).

The formula must include the C specification.

F and L special leading characters are not valid for this definition.

If an L is specified preceding the @(c,r) cursor positioning definition, line feed delays also follow all column-only cursor positioning strings.

Most terminals do not support horizontal only cursor positioning. When setting up a TERMTYPE for terminals that use this type of cursor positioning, such as ADDS devices, use the normal control sequence, for example, CHAR(16) C.

When setting up a TERMTYPE for terminals that do not use horizontal only cursor positioning, you can simulate this function in one of two ways:

Character	Explanation
C	Specifies a column's place held in a cursor positioning string. The C character is replaced in the resulting string by the character(s) generated by processing the requested column position through the formula used to calculate the cursor address type (See Field 3 of Menu 1).
R	Specifies a row's place held in a cursor positioning string. The R character is replaced in the resulting string by the character(s) generated by processing the requested row position through the formula used to calculate the cursor address type (See Field 3 of Menu 1).
L	Specifies that the resulting string is to be appended with the number of ASCII null (NUL, X'00') characters specified in the TERM or TERMINAL parameter LF DELAY. This optional leading character is used when a terminal requires delay time after receiving a cursor control string before it can begin to receive more characters.
F	Specifies that the resulting string is to be appended with the number of ASCII null (NUL, X'00') characters specified in the TERM or TERMINAL parameter FE DELAY.

Character	Explanation
	This is an optional leading character.
D	<p>Specifies a video attributes definition's place held in a cursor positioning string.</p> <p>The D character is replaced in the resulting string by the hexadecimal character generated by logically ORing the values defined in Menu 1 Fields 7, 8, 9, 10, and 11.</p> <p>This character is valid only in Menu 1, Field 6, where it indicates the eventual location of the ORed video attributes value.</p>
'string' or "sting"	<p>Specifies a literal string.</p> <p>A literal string is any text delimited by single or double quotation marks.</p>
CHAR(<i>n</i>)	<p>Specifies a single character function with a binary value equal to the numeric <i>n</i>.</p> <p>For example, CHAR(32) is a blank and CHAR(65) is the capital A.</p>
CTRL(<i>char</i>)	<p>Specifies a single character function with a binary value equal to the value of holding down the CTRL and pressing the <i>char</i> key.</p> <p>Values for <i>char</i> are limited to the following:</p> <ul style="list-style-type: none"> • @ • Z • [• \ •] • ^ • - <p>This function is used, for example, when a terminal's functions are enabled or disabled by sending it CTRL+A or CTRL+U characters.</p>
HEX(<i>x</i>)	<p>Generates a string of the ASCII characters equivalent to the hexadecimal string defined by <i>x</i>.</p> <p>For example, HEX(1B20) generates an ESC character followed by a blank.</p> <p>HEX(30314142) generates the string 01AB.</p>
STR(<i>x,n</i>)	<p>Generates a repetition of the string <i>x</i> <i>n</i> times.</p> <p>The string <i>x</i> can be any combination of the following:</p> <ul style="list-style-type: none"> • A literal. • An ASCII control character name. • A CHAR(<i>n</i>) function. • A CTRL(<i>char</i>) function. • A HEX(<i>x</i>) function. <p>Blanks are not allowed within the function unless they are part of a literal string or as part of a multi-element string in which each element must be preceded by a blank.</p> <p>For example,</p> <ul style="list-style-type: none"> • "STR(ESC 'G' ,10)"

Character	Explanation
	<ul style="list-style-type: none"> • "STR(ESC CTRL(A), C)" <p>If <i>x</i> is only a single element, it does not require leading and trailing blanks.</p> <p>For example,</p> <ul style="list-style-type: none"> • "STR(CHAR(6) ,C)" <p>and</p> <ul style="list-style-type: none"> • "STR(CHAR:(6) ,C)" <p>are equivalent expressions.</p> <p><i>n</i> can either be a literal number or a special character C or R. When a C or R is specified, the value used for <i>n</i> is the input column or row value.</p> <p>The C or R is not preprocessed by the formulas defined in the cursor positioning types.</p> <p>For example, the string,</p> <ul style="list-style-type: none"> • "STR('a',5)" <p>generates the string aaaaa, and the string,</p> <ul style="list-style-type: none"> • "STR(CHAR(64),R)" <p>generates the string @@@@, and the string,</p> <ul style="list-style-type: none"> • "STR (CHAR (6), C)" • <p>generates a string of CHAR(6)s <i>C</i> characters long, where <i>C</i> is a column specification.</p>
ASCII name	<p>Specifies any ASCII control character (characters 0 - 31 and character 127) by its ASCII name.</p> <p>For example,</p> <ul style="list-style-type: none"> • ESC • NUL • ACK

3.3.1.5 @ (C) cursor positioning (continued)

Return the cursor to the leftmost column with a RETURN and then output a cursor forward command the number of times necessary to reach the specified column.

For example, with a terminal that uses CHAR(6) as a cursor forward, use STR(CRAR(6),C). The C in this statement is not processed by the cursor addressing formula. The actual requested column number is used in place of the C.

Note

This method does not work in all cases.

Use an internal table to store values of R.

If this field specifies the formula the same as the @(C,R) addressing, the value of R is looked up in the internal table.

This procedure works in the following example,

```
PRINT @(10,20) : 'A' : @(30) : 'B'
```

because the last specified row is 20 when the @(30) command is processed.

However, this procedure does not work in the following example,

```
PRINT @(-1) : 'A' : @(30) : 'B'
```

because the @(-1) command positions the cursor to row 9 but doesn't store the 0 in the internal table as the last used row.

Therefore when the @(30) is processed, the row it appears in is taken from the most recently specified command (i.e., @(c,r)) and the B doesn't necessarily appear on row 0.

If it works, this method executes more quickly than the first method discussed.

3.3.1.6 @ (C,R) cursor positioning

Defines the formula used to position the cursor to a specified column C and row R.

The formula must include both the C and the R specifications.

The L special leading character is not valid for this definition.

Examples:

- **ESC '=' R C**
- **CHAR(11) R CHAR(16) C**

where the R and C characters are replaced with the characters emitted by the cursor addressing formulas.

The L special leading character is used to specify that all cursor positioning strings are to be followed by line feed delays.

3.3.1.7 Visual attributes definition

Defines the character sequence used to enable terminal video attributes.

The actual D character used in the character sequence is computed by logically ORing the binary values of the characters defined for fields seven to eleven of this menu. The TDM program does this for you.

This field allows for any sequence of characters that includes the special character D.

When Reality processes this sequence, the D is replaced by the character produced by logically ORing the necessary elements of TDM definition fields seven to eleven to produce the desired video attribute bit pattern.

For example, an entry in field 6 of

```
SO ESC '0' D
```

when processed with a request for a dimmed underlined attribute, emits the characters SO, ESC, and 0 followed by the logically ORed characters from TDM definition fields seven and eleven, DIM and UNDERLINE.

The next example is based on the following: suppose characters 1, 2, 4, 8, and j enable the corresponding bit patterns X'31', X'32', X'34', X'38', and X'70'.

Now, if the calling program requested a video attribute list of dimmed, flashing, and reverse video, TDM would logically OR the characters 1, 2, and 4 to produce a character bit pattern of X'37'. The character X'37' would then be substituted for D in the definition statement for this field.

3.3.1.8 "D" character for DIM attribute

Defines the character to be used to define the DIM attribute when computing the D character.

3.3.1.9 "D" character for FLASHING attribute

Defines the character to be used to define the FLASHING attribute when computing the D character.

3.3.1.10 "D" character for REVERSE attribute

Defines the character to be used to define the REVERSE attribute when computing the D character.

3.3.1.11 "D" character for BLANK attribute

Defines the character to be used to define the BLANK attribute when computing the D character.

3.3.1.12 "D" character for UNDERLINE attribute

Defines the character to be used to define the UNDERLINE attribute when computing the D character.

3.3.1.13 Visual attributes 'off' definition

Defines the character sequence used to disable existing video attributes.

3.3.1.14 Embedded visual attributes (Y/N)

Defines whether a video attribute ON or OFF character requires a character position.

Enter **y** if a character position is required. If you answer **y** to this prompt, do not answer prompts 14 and 15 because they are irrelevant.

Enter **n** if a character position is not required. If you answer **n** to this prompt, answer prompts 14 and 15 as well.

3.3.1.15 Force vis. attr. to one column (Y/N)

Defines whether a video attribute character takes a character position (column) or not.

This feature is used to make code have the same video effects irrespective of the terminal it is run on. For example, if one terminal does not take a column to embed a video attribute and another does, the same code produces a different effect on the two terminals.

Enter **y** if you want to force a video attribute to take the place of a (blank) column when the defined terminal would not.

Enter **n** if you do not want to force video attributes to take up a column.

3.3.1.16 Endow vis. attr. with effects (Y/N)

Defines whether a non-displayed, forced column video attribute character takes the defined video attribute for itself or not.

This feature is used to specify, for example, whether an underlined word looks like this (video attribute endowed with effect) or if it looks like this (video attribute not endowed with effect).

Enter **y** if you want to endow video attributes with video effects.

Enter **n** if you don't want to endow video attributes with video effects.

Field #, <cr> for next page, "EX" to exit, "FI" to file changes, or "?":

To change a field, type its line number and press RETURN.

To move to Menu 2, press RETURN.

To exit Menu 1 and move back to the TDM menu, type **EX**.

To file your changes to the table being edited, type **FI**.

To get online help documentation, type **?** and press RETURN.

3.3.2 Menu 2

3.3.2.1 @(-1) Clear screen & home

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to indicate to clear the screen and position the cursor at the home position.

This information is used by the @(-1) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.2.2 @(-2) Cursor home

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to indicate to send the cursor to the home position.

This information is used by the @(-2) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.2.3 @(-3) Clear to end of screen

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to indicate to clear all lines up to the end of the screen.

This information is used by the @(-3) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.2.4 @(-4) Clear to end of line

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to indicate to clear to the end of the current line.

This information is used by the @(-4) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.2.5 @(-5) Start flashing

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to indicate to start flashing at the named position.

This information is used by the @(-5) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.2.6 @(-6) Stop flashing

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to indicate to stop flashing at the named position.

This information is used by the @(-6) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.2.7 @(-7) Start protect

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to indicate that the text string that follows, is protected from user input.

Fields 22 and 23 are used to delimit a protected text string.

This information is used by the @(-7) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.2.8 @(-8) Stop protect

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to terminate protection from user input.

Fields 22 and 23 are used to delimit a protected text string.

This information is used by the @(-8) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.2.9 @(-9) Cursor back

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to indicate to move the cursor back one space.

This information is used by the @(-9) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.2.10 @(-10) Cursor up

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to indicate to move the cursor up one line.

This information is used by the @(-10) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.2.11 @(-11) Cursor on

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to turn the cursor on.

This information is used by the @(-11) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.2.12 @(-12) Cursor off

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to turn the cursor off.

This information is used by the @(-12) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.2.13 @(-13) Begin status line

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to indicate the beginning of the status line.

This information is used by the @(-13) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.2.14 @(-14) End status line

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to indicate the end of the status line.

This information is used by the @(-14) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.2.15 @(-15) Cursor forward

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to indicate to move the cursor forward one space.

This information is used by the @(-15) function in DataBasic.

Examples for different terminals are provided in Appendix B.

Field #, <cr> for next page, "EX" to exit, "FI" to file changes, or "?":

To change a field, type its line number and press RETURN.

To return to Menu 1, type a line number from that menu and press RETURN.

To move to Menu 3, press RETURN.

To exit Menu 2 and move back to the TDM menu, type **EX**.

To file your changes to the table being edited, type **FI**.

To get online help documentation, type ? and press RETURN.

3.3.3 Menu 3

3.3.3.1 @(-16) Cursor down

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to indicate to move the cursor down one line.

This information is used by the @(-16) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.3.2 @(-17) Enable slave port

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to enable a slave port.

This information is used by the @(-17) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.3.3 @(-18) Disable slave port

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to disable a slave port.

This information is used by the @(-18) function in DataBasic.

Examples for different terminals are provided in Appendix B.

3.3.3.4 @(-19) Screen dump

Enter the hexadecimal character sequence, decimal character representation, or ASCII character representation used by this terminal to dump a screen.

This information is used by the @(-19) function in DataBasic.

Examples for different terminals are provided in Appendix B.

Field #, <cr> for next page, "EX" to exit, "FI" to file changes, or "?":

To change a field, type its line number and press RETURN.

To return to Menu 2, type a line number from that menu and press RETURN.

To return to Menu 1, press RETURN.

To exit Menu 3 and move back to the TDM menu, type **EX**.

To file your changes to the table being edited, type **FI**.

To get online help documentation, type ? and press RETURN.

Section 4: Appendix A - Decimal, Hexadecimal and ASCII table

This appendix lists the hexadecimal and ASCII values and keys or effects corresponding to decimal numbers 0 to 127 and 251 to 255.

Note

Key shown is typical but may vary with terminal type or configuration.

DEC	HEX	ASCII	KEY(S)	EFFECT
000	00	NUL	CTRL+@	
001	01	SOH	CTRL+A, HOME	
002	02	STX	CTRL+B	
003	03	ETX	CTRL+C	
004	04	EOT	CTRL+D	
005	05	ENQ	CTRL+E	
006	06	ACK	CTRL+F, →	
007	07	BEL	CTRL+G	
008	08	BS	CTRL+H, BACKSPACE	
009	09	HT	CTRL+I, TAB	
010	0A	LF	CTRL+J ↓	Line Feed
011	0B	VT	CTRL+K	
012	0C	FF	CTRL+L, CLEAR	
013	0D	CR	CTRL+M, RETURN	
014	0E	SO	CTRL+M, RETURN	
015	0F	SI	CTRL+O	
016	10	DLE	CTRL+P	
017	11	DC1	CTRL+Q	XON function
018	12	DC2	CTRL+R	Redisplay line
019	13	DC3	CTRL+S	XOFF function
020	14	DC4	CTRL+T	
021	15	NAK	CTRL+U ←	
022	16	SYN	CTRL+V	
023	17	ETB	CTRL+W	
024	18	CAN	CTRL+X	Cancel line
025	19	EM	CTRL+Y, SHIFT+TAB	Disconnect
026	1A	SUB	CTRL+Z ↑	
027	1B	ESC	ESC, CTRL+[

DEC	HEX	ASCII	KEY(S)	EFFECT
028	1C	FS		
029	1D	GS		
030	1E	RS		
031	1F	US		
032	20	SPACE	SPACEBAR	
033	21	!	!	
034	22	"	"	
035	23	#	#	
036	24	\$	\$	
037	25	%	%	
038	26	&	&	
039	27	`	`	
040	28	((
041	29))	
042	2A	*	*	
043	2B	+	+	
044	2C	,	,	
045	2D	-	-	
046	2E	.	.	
047	2F	/	/	
048	30	0	0	
049	31	1	1	
050	32	2	2	
051	33	3	3	
052	34	4	4	
053	35	5	5	
054	36	6	6	
055	37	7	7	
056	38	8	8	
057	39	9	9	
058	3A	:	:	
059	3B	;	;	
060	3C	<	<	
061	3D	=	=	
062	3E	>	>	

DEC	HEX	ASCII	KEY(S)	EFFECT
063	3F	?	?	
064	40	@	@	
065	41	A	A	
066	42	B	B	
067	43	C	C	
068	44	D	D	
069	45	E	E	
070	46	F	F	
071	47	G	G	
072	48	H	H	
073	49	I	I	
074	4A	J	J	
075	4B	K	K	
076	4C	L	L	
077	4D	M	M	
078	4E	N	N	
079	4F	O	O	
080	50	P	P	
081	51	Q	Q	
082	52	R	R	
083	53	S	S	
084	54	T	T	
085	55	U	U	
086	56	V	V	
087	57	W	W	
088	58	X	X	
089	59	Y	Y	
090	5A	Z	Z	
091	5B	[[
092	5C	\	\	
093	5D]]	
094	5E	^	^	
095	5F	_	_	
096	60	'	'	
097	61	a	a	

DEC	HEX	ASCII	KEY(S)	EFFECT
098	62	b	b	
099	63	c	c	
100	64	d	d	
101	65	e	e	
102	66	f	f	
103	67	g	g	
104	68	h	h	
105	69	i	i	
106	6A	j	j	
107	6B	k	k	
108	6C	l	l	
109	6D	m	m	
110	6E	n	n	
111	6F	o	o	
112	70	p	p	
113	71	q	q	
114	72	r	r	
115	73	s	s	
116	74	t	t	
117	75	u	u	
118	76	v	v	
119	77	w	w	
120	78	x	x	
121	79	y	y	
122	7A	z	z	
123	7B	{	{	
124	7C			
125	7D	}	}	
126	7E	~	~	
127	7F	DEL	DELETE	
.				
.				
.				
251	FB	[SB: start buffer
252	FC	\	CTRL+\	SVM: subvalue mark

DEC	HEX	ASCII	KEY(S)	EFFECT
253	FD]	CTRL+]	VM: value mark
254	FE	^	CTRL+^	AM: attribute mark
255	FF	_	CTRL+_	SM: segment mark

Section 5: Appendix B – Terminal definition examples

This appendix provides three terminal definition examples.

1.1 Example 1

```
1 TERMTYPE : 4
2 Max column, Max row : 79,23
3 Cursor Address type (A,L,T,H,D,X) : A
4 @(C) cursor positioning : CHAR(16) C
5 @(C,R) cursor positioning : CHAR(11) R CHAR(16) C
6 Visual attributes definition (D) : D
7 "D" character for DIM attribute : HEX(81)
8 "D" character for FLASHING attribute : HEX(82)
9 "D" character for REVERSE attribute : HEX(84)
10 "D" character for BLANK attribute : HEX(88)
11 "D" character for UNDERLINE attribute : HEX(90)
12 Visual attributes 'off' definition : HEX(80)
13 Embedded visual attributes(Y/N) : YES
14 @(-1) Clear screen & home : F HEX (0C)
15 @(-2) Cursor home : HEX(01)
16 @(-3) Clear to end of screen : ESC'J'
17 @(-4) Clear to end of line : ESC'K'
18 @(-5) Start flashing : HEX(82)
19 @(-6) Stop flashing : HEX(80)
20 @(-7) Start protect :
21 @(-8) Stop protect :
22 @(-9) Cursor back : CTRL(U)
23 @(-10) Cursor up : CTRL(Z)
24 @(-11) Cursor on : HEX(E2)
25 @(-12) Cursor off : HEX(E4)
26 @(-13) Begin status line : HEX(E3)
27 @(-14) End status line : HEX(E5)
28 @(-15) Cursor forward : CHAR(6)
29 @(-16) Cursor down : LF
30 @(-17) Enable slave port : CHAR(18)
@(-18) Disable slave port : CHAR(20)
32 @(-19) Screen dump : CTRL(S)
```

1.2 Example 2

```
1 TERMTYPE : 21
2 Max column, Max row : 80,24
3 Cursor Address type (A,L,T,H,D,X) : A
4 @(C) cursor positioning : CHAR(16) C
5 @(C,R) cursor positioning : CHAR(11) R CHAR(16) C
6 Visual attributes definition (D) : SO ESC '0' D
7 "D" character for DIM attribute : 'A'
8 "D" character for FLASHING attribute : 'B'
9 "D" character for REVERSE attribute : 'P'
10 "D" character for BLANK attribute : 'D'
11 "D" character for UNDERLINE attribute : HEX(60)
12 Visual attributes 'off' definition : SI
13 Embedded visual attributes(Y/N) : NO
14 @(-1) Clear screen & home : CHAR(12)
15 @(-2) Cursor home : SOH
16 @(-3) Clear to end of screen : ESC'k'
17 @(-4) Clear to end of line : ESC'K'
18 @(-5) Start flashing : SO ESC '0B'
19 @(-6) Stop flashing : SI
20 @(-7) Start protect :
21 @(-8) Stop protect :
22 @(-9) Cursor back : NAK
```

```
23 @(-10) Cursor up : SUB
24 @(-11) Cursor on :
25 @(-12) Cursor off :
26 @(-13) Begin status line :
27 @(-14) End status line :
28 @(-15) Cursor forward : ACK
29 @(-16) Cursor down : CHAR(10)
30 @(-17) Enable slave port : ESC '3'
31 @(-18) Disable slave port : ESC '4'
32 @(-19) Screen dump :
```

1.3 Example 3

```
1 TERMTYPE : 30
2 Max column, Max row : 80,24
3 Cursor Address type (A,L,T,H,D,X) : L
4 @(C) cursor positioning : CR STR(CTRL(L),C)
5 @(C,R) cursor positioning : ESC '=' R C
6 Visual attributes definition (D) : ESC 'G' D
7 "D" character for DIM attribute :
8 "D" character for FLASHING attribute : '2'
9 "D" character for REVERSE attribute : '4'
10 "D" character for BLANK attribute : '1'
11 "D" character for UNDERLINE attribute : '8'
12 Visual attributes 'off' definition : ESC 'GO'
13 Embedded visual attributes(Y/N) : YES
14 @(-1) Clear screen & home : CTRL(Z)
15 @(-2) Cursor home : CTRL(^)
16 @(-3) Clear to end of screen : ESC'Y'
17 @(-4) Clear to end of line : ESC'T'
18 @(-5) Start flashing : ESC'G2'
19 @(-6) Stop flashing : ESC'G0'
20 @(-7) Start protect :
21 @(-8) Stop protect :
22 @(-9) Cursor back : CTRL(H)
23 @(-10) Cursor up : CTRL(K)
24 @(-11) Cursor on : ESC'.'
25 @(-12) Cursor off : ESC'.'
26 @(-13) Begin status line :
27 @(-14) End status line :
28 @(-15) Cursor forward : CTRL(L)
29 @(-16) Cursor down : CTRL(J)
30 @(-17) Enable slave port : ESC'@'
31 @(-18) Disable slave port : ESC'A'
32 @(-19) Screen dump : ESC'P'
```

Section 6: Appendix C – Column and row addresses

This appendix contains column and row addresses and associated characters for cursor addressing types A, L, T and H.

6.1 Column and row addresses for cursor addressing types A, L, T and H

Column ("C")	Row ("R")	ADDS col	ADDS row	LSI	TEC	Hazeltime
0	0	NUL	@	Blank	X'7F'	NUL
1	1	SOH	A	!	~	SOH
2	2	STX	B	"	}	STX
3	3	ETX	C	#		ETX
4	4	EOT	D	\$	{	EOT
5	5	ENQ	E	%	Z	ENQ
6	6	ACK	F	&	Y	ACK
7	7	BEL	G	'	X	BEL
8	8	BS	H	(w	BS
9	9	HT	I)	V	HT
10	10	DLE	J	*	S	LF
11	11	DC1	K	+	T	VT
12	12	DC2	L	^	S	FF
13	13	DC3	M	-	R	CR
14	14	DC4	N	.	Q	SO
15	15	NAK	O	/	P	SI
16	16	SYN	P	0	O	DLE
17	17	ETB	Q	1	N	DC1
18	18	CAN	R	2	m	DC2
19	19	EM	S	3	L	DC3
20	20	blank	T	4	K	DC4
21	21	!	U	5	J	NAK
22	22	"	V	6	I	SYN
23	23	#	W	7	H	ETB
24		\$		8	G	CAN
25		%		9	F	EM
26		&		:	E	SUB
27		'		;	D	ESC
28		(<	C	FS

Column ("C")	Row ("R")	ADDS col	ADDS row	LSI	TEC	Hazetime
29)		=	B	GS
30		0		>	A	RS
31		1		?	`	US
32		2		@	_	blank
33		3		A	^	!
34		4		B]	"
35		5		C	\	#
36		6		D	[\$
37		7		E	Z	%
38		8		F	Y	&
39		9		G	X	`
40		@		H	W	(
41		A		I	V)
42		B		J	U	*
43		C		K	T	`+
44		D		L	S	`
45		E		M	R	-
46		F		N	Q	.
47		G		O	P	/
48		H		P	O	0
49		I		Q	N	1
50		P		R	M	2
51		Q		S	L	3
52		R		T	K	4
53		S		U	J	5
54		T		V	I	6
55		U		W	H	7
56		V		X	G	8
57		W		Y	F	9
58		X		Z	E	:
59		Y		[D	;
60		`		\	C	<
61		a]	B	=
62		b		^	A	>
63		c		_	@	?

Column ("C")	Row ("R")	ADDS col	ADDS row	LSI	TEC	Hazetime
64		d		`	?	@
65		e		a	>	A
66		f		b	;	B
67		g		c	<	C
68		h		d	;	D
69		i		e	:	E
70		p		f	9	F
71		q		g	8	G
72		r		h	7	H
73		s		i	6	J
74		t		j	5	K
75		u		k	4	I
76		v		l	3	J
77		w		m	2	K
78		x		n	1	L
79		y		o	0	M

Section 7: Appendix D – Creating additional definition fields for the TERMTYPE menus

This appendix tells you how to create new definition fields for the TERMTYPE menus. This is useful if you are attempting to define your own special functions to be used in the DataBasic @(negative-number) commands.

7.1 The ELINE editor

If you want to create your own definition fields, you should use the ELINE editor provided for this purpose, on the SYSPROG account.

The ELINE commands are described in Appendix E, Using the ELINE Editor to Create Additional Definition Fields for the TERMTYPE Menus.

7.2 Adding new definition fields

You can add extra definition fields to TERMTYPES if you want to use the DataBasic @(negative-number) commands (-20 to -127) as they are defined in the *DataBasic Programming Reference Manual*.

Never change the first 13 definition fields. TDM expects them to be in a particular format and if they are not, the results are unpredictable.

Definition Fields 14 to 32 should not be altered either. If you change any of these fields, they are no longer compatible with the standard definitions.

New definition fields are added as items in DICT CURSOR-DEFS and have the format described in the following table:

Attribute	Explanation
Item-id	Must be in the form LINE _n , where n is the next available sequential number for the TERMTYPE.
1	The prompting text that displays on the screen.
2	The help text that displays on the screen when you press the ? key while editing a field. You can use up to three values to contain this text.
3	Optional multivalued DataBasic format MATCH strings to be applied to input after Reality applies the conversion defined in attribute five of this item.
4	Optional special function definition characters: <ul style="list-style-type: none"> • L allows a leading L in the specification for line feed delays • F allows a leading F in the specification for form feed delays

Attribute	Explanation
	<ul style="list-style-type: none">• Q indicates a required input field
5	Defines an optional conversion code to be performed on any input before other checks are made.

Section 8: Appendix E – Creating additional definition fields for the TERMTYPE menus

ELINE is a line editor used to create additional definition fields for TERMTYPE menus.

Appendix D, Creating Additional Definition Fields for the TERMTYPE Menus, describes the restrictions of creating new definition fields. This appendix describes the editor used to create those new fields.

8.1 ELINE editor commands

ELINE editor commands and their explanation is as follows:

Attribute	Explanation
TAB CTRL+I	Word forward
<u>BACK TAB</u> CTRL+Y SHIFT+TAB	Word back
CTRL+E	Move cursor to end of field.
CTRL+B	Move cursor to beginning of field.
<u>SHIFT+4</u> CTRL+U LEFT ARROW	Cursor left
<u>SHIFT+6</u> CTRL+F RIGHT ARROW	Cursor right
CTRL+?	Insert characters.
<u>ESC</u> ESCAPE	Exit insert mode.
CTRL+D	Delete character
CTRL+W	Delete word
CTRL+X	Clear line from the cursor to the end of the line
CTRL+C	Cut. (See "Cutting and Pasting with ELINE")
CTRL+P	Paste (See "Cutting and Pasting with ELINE")
<u>DEL</u> DELETE+HOME	Cancel last change.

Command keystrokes that are underlined are only valid on Prism IV terminal keyboards.

Command keystrokes that are bold are only valid on Prism 7/8/9 terminal keyboards.

All other keystrokes are valid for both terminal types.

8.2 Cutting and pasting with ELINE

To cut a section of text, move the cursor to the beginning of the section to be cut. Press CTRL+C. Move the cursor to the end of the section to be cut. Press CTRL+C again.

To paste the section of text just cut, move the cursor to the place you want to insert the text, then press CTRL+P.

About NEC

Our customers change lives, so we create software and services that get them better outcomes. By innovating when it matters most, we help to keep people safer, healthier and better connected worldwide.

NEC

NEC.com

1st Floor, Bizspace, iMex Centre,
575-599 Maxted Rd,
Hemel Hempstead HP2 7DX
+44 (0)1442 768445