

# **Picture Editor Reference Manual**

**SW09-650**



**Implementation  
Engineering Operations - 2**

***Picture Editor  
Reference Manual***

**SW09-650  
Release 620  
12/00**



# Notices and Trademarks

© Copyright 2000 by Honeywell Inc.

Revision 03 – December 1, 2000

While this information is presented in good faith and believed to be accurate, Honeywell disclaims the implied warranties of merchantability and fitness for a particular purpose and makes no express warranties except as may be stated in its written agreement with and for its customer.

In no event is Honeywell liable to anyone for any indirect, special or consequential damages. The information and specifications in this document are subject to change without notice.

Fieldbus Foundation and FOUNDATION are trademarks of the Fieldbus Foundation.

**TotalPlant** and TPS are U.S. registered trademarks of Honeywell Inc.

Other brand or product names are trademarks of their respective owners.

Honeywell  
Industrial Automation and Control  
16404 N. Black Canyon Highway  
Phoenix, AZ 85053  
**1-800-343-0228**

---



# About This Publication

This publication provides reference information that helps you understand what can be done using the Universal Station's Picture Editor function, as well as the tools, language, and commands used in creating the pictures.

## Is this the right publication?

The Picture Editor Reference Manual will help you to understand the overall process of picture building and the tools used in the building displays and free format logs. It does not give you specific instructions on filling out the forms used to document custom displays or free format logs, nor does it give you specific instructions on calling up the Picture Editor function at the Universal Station and using the keyboard and touch targets to enter them. Those subjects are covered in other manuals.

## Associated Publications

Action sequences or Actors are used by some Picture Editor commands and are described in the *Actors Manual* (see References). *Picture Editor Forms*, *Picture Editor Form Instructions* and the *Picture Editor Data Entry* manual (see References) also contain related information.

## What should I be familiar with before using this publication?

The content of these publications:

*Implementation Overview* in the Implementation/Startup & Reconfiguration - 1 binder.

*Configuration Data Collection Guide* in the Implementation/Startup & Reconfiguration - 2 binder.

This publication supports **TotalPlant** Solution (TPS) System network Releases 500 - 620. TPS is the evolution of TDC 3000<sup>X</sup>.





# Table of Contents

<b>1 INTRODUCTION .....</b>	<b>1</b>
1.1 OVERVIEW .....	1
1.1.1 Custom Graphic Displays.....	1
1.2 REFERENCES.....	4
<b>2 TOOLS FOR PICTURE BUILDING.....</b>	<b>5</b>
2.1 FORMS AND FORM INSTRUCTIONS.....	5
2.2 DATA ENTRY INSTRUCTIONS .....	5
2.3 COMMANDS.....	6
2.4 FREE FORMAT LOGS .....	6
<b>3 PICTURE BUILDING .....</b>	<b>7</b>
3.1 SCREEN FORMAT FOR PICTURE BUILDING.....	7
3.1.1 Entering Commands .....	8
3.1.2 Command Abbreviations.....	9
3.1.3 Specifying Locations .....	9
3.1.4 Errors      10	
3.1.5 Help        10	
3.2 COMMANDS.....	11
3.3 COMMAND DESCRIPTIONS .....	12
3.3.1 General Purpose Commands.....	12
3.3.2 Fixed Behavior and Fixed Behavior Commands.....	74
3.3.3 Conditional Behavior .....	79
3.3.4 Graphic Commands .....	86
3.3.5 Text and Text Commands.....	99
3.3.6 Values     105	
3.3.7 Variants   111	
3.3.8 Bar Charts116	
3.3.9 Targets    122	
3.3.10 Subpictures .....	127
<b>APPENDIX A VALUE FORMATS.....</b>	<b>153</b>
A.1 INTEGER.....	153
A.2 REAL.....	154
A.3 BOOLEAN.....	154
A.4 STRING .....	155
A.5 ENUMERATION .....	156
A.6 SELF-DEFINING ENUMERATIONS .....	156
A.7 UNKNOWN.....	158

# Table of Contents

A.8 DATE TIME.....	159
<b>APPENDIX B NAME FORMS .....</b>	<b>163</b>
<b>APPENDIX C EXPRESSIONS.....</b>	<b>169</b>
C.1 EXPRESSION SYNTAX.....	169
C.2 INTRINSIC FUNCTIONS.....	171
C.2.1 Bit_Test	171
C.2.2 External	172
C.2.3 Internal	172
C.2.4 Status	172
C.3 ARRAYS AS PARAMETERS .....	173
C.4 SEMI-RESERVED NAMES .....	173
C.5 DATA TYPES IN EXPRESSIONS.....	174
C.5.1 Operations between Two Operands.....	174
C.5.2 Referencing AM/CL Custom Data Segment Boolean Parameters .....	175
<b>APPENDIX D CONDITIONAL BEHAVIOR SYNTAX .....</b>	<b>177</b>
<b>APPENDIX E VARIANT SYNTAX .....</b>	<b>181</b>
<b>APPENDIX F ENUMERATION CONSTANTS .....</b>	<b>183</b>
<b>APPENDIX G APPLICATION SUBPICTURES .....</b>	<b>185</b>
G.1 DESCRIPTION.....	185
G.1.1 Circle	186
G.1.2 Quarter	187
G.1.3 Three Dimensional Pie Charts .....	189
G.1.4 Trend Subpictures.....	191
G.1.5 Ring	192
G.1.6 Radar Chart (Fixed Axis).....	196
G.1.7 RADAR1 Chart (Variable Axis).....	199
G.1.8 LGRAPH	202
G.1.9 LGRAPHn.....	205
G.1.10 Custom Status Display Frame.....	208
G.1.11 Small Node Object Box .....	212
G.1.12 Large Node Object Box.....	213
<b>APPENDIX H COLLECTORS.....</b>	<b>215</b>
H.1 INTRODUCTION TO COLLECTORS .....	218
H.1.1 Collector Concepts .....	218

# Table of Contents

H.2 ACKNOWLEDGE STATE COLLECTOR.....	218
H.3 SYSTEM TIME/DATE COLLECTOR.....	219
H.4 HISTORY COLLECTORS .....	219
H.4.1 Historical Value Collectors.....	224
H.4.2 Historical Time Collectors.....	225
H.4.3 Historical Status Collectors.....	226
H.4.4 Using Indirect Collectors.....	227
H.5 ORIGINAL HISTORICAL COLLECTORS .....	228
H.5.1 Historical Value Collectors.....	228
H.5.2 Historical Time Collectors.....	230
H.6 TREND COLLECTORS.....	231
H.6.1 Trend Centerline Time.....	231
H.6.2 Trend Trace Color .....	232
H.6.3 Trend Trace Data Source.....	232
H.6.4 Trend Trace Engineering Units Descriptor (R530 and later systems) .....	232
H.6.5 Trend Trace Variable Name .....	232
H.6.6 Trend Trace Parameter (R530 and later systems).....	232
H.6.7 Trend Trace Point (R530 and later systems).....	233
H.6.8 Trend Trace Point Description (R530 and later systems).....	233
H.6.9 Trend Trace Range (High) .....	233
H.6.10 Trend Trace Range (Low) .....	233
H.6.11 Trend Trace Real Time Value (R530 and later systems) .....	233
H.6.12 Trend Scale Range (High).....	233
H.6.13 Trend Scale Range (Low).....	234
H.6.14 Trend Scroll Time Stamp.....	234
H.6.15 Trend Time Base.....	234
H.6.16 Hair Line Cursor Status (R510 and later systems) .....	235
H.6.17 Hair Line Cursor Trend Value (R510 and later systems).....	235
H.6.18 Trend Graph Cursor Position (R510 and later systems) .....	236
H.7 SCHEMATIC ALARM COLLECTORS.....	236
H.7.1 Schematic Alarm Status Collectors .....	236
H.7.2 Schematic Alarm Count Collectors.....	240
H.8 MISCELLANEOUS COLLECTORS.....	244
H.8.1 Universal Station Keylock Status.....	244
H.9 DOCUMENTATION TOOL QUERY COLLECTORS.....	245
H.9.1 QUERY DESCRIPTOR NAME.....	245
H.9.2 QUERY INVOCATION-DATE/TIME .....	245
H.9.3 QUERY STATUS.....	245
<b>APPENDIX I SCHEMATIC LOADING .....</b>	<b>247</b>
I.1 SCHEMATIC OPTIMIZATION .....	247
I.1.1 General Guidelines to Reduce Schematic Loading .....	247
I.2 REDUCING SCHEMATIC LOADING ON ALL NODES .....	249

# Table of Contents

- I.2.1 Using Longer Update Periods .....249
- I.2.2 Grouping by LCN Node Type and Unit .....249
- I.3 REDUCING SCHEMATIC LOADING ON UCN NODES.....249
  - I.3.1 Grouping of Data Access Requests by Processor Type .....250
  - I.3.2 Picture Editor Optimize Collection Groups (R530).....252
  - I.3.3 Use of Parameters at the Highest Data Owner Level .....254

# 1 INTRODUCTION

## 1.1 OVERVIEW

The Picture Editor is an engineering function of the Universal Personality (UP). With this running in a Universal Station, entry is made by selecting PICTURE EDITOR on the Engineering Menu. You can exit by executing an END command or by pressing the MENU key on the Engineering Keyboard (hold down the CTL key and press HELP). You can temporarily switch to the Command Processor (e.g., to use a File Utility function) by pressing the ESC key. Pressing MENU returns the Picture Editor.

With the Universal Personality running, press MENU to obtain the Engineering Menu, then select PICTURE EDITOR as before. To leave the Engineering functions, return to the Engineering Menu by again pressing the MENU key. If you then select some type of Operator display, Operator Personality functions become active.

The Picture Editor allows you to build and edit custom graphic displays without the need for extensive programming knowledge. The picture-building process consists mainly of drawing lines and typing data onto the Universal Station's screen.

### 1.1.1 Custom Graphic Displays

Custom graphic displays often represent a process in schematic form. Figure 1.1 is an example of such a schematic. The custom display can contain dynamic and interactive items. Dynamic items provide "live" information; that is, measured data that is periodically updated or shapes that change behavior to represent changing process conditions. For example, the measured pressure in pounds per square inch could be reported numerically below the outline of a pump. If the pump stops, the outline could turn red and blink.

Interactive items consist of visible or invisible target areas on the custom display. Activating a target by touch or by cursor can cause a number of user-defined actions, typically some type of process control or call up of another display.

#### 1.1.1.1 Building Custom Graphic Displays

Custom graphic displays are put together by using a series of commands to build the various shapes and features. The primary purpose of this manual is to describe those commands in detail.

As shown in 1.1.1.2, the Picture Editor commands are generally grouped according to the functions they support. There is one exception. Commands described in the General Purpose section may be needed while working with any of the other functions. The Picture Building section of this manual contains a quick index to locate specific command descriptions.

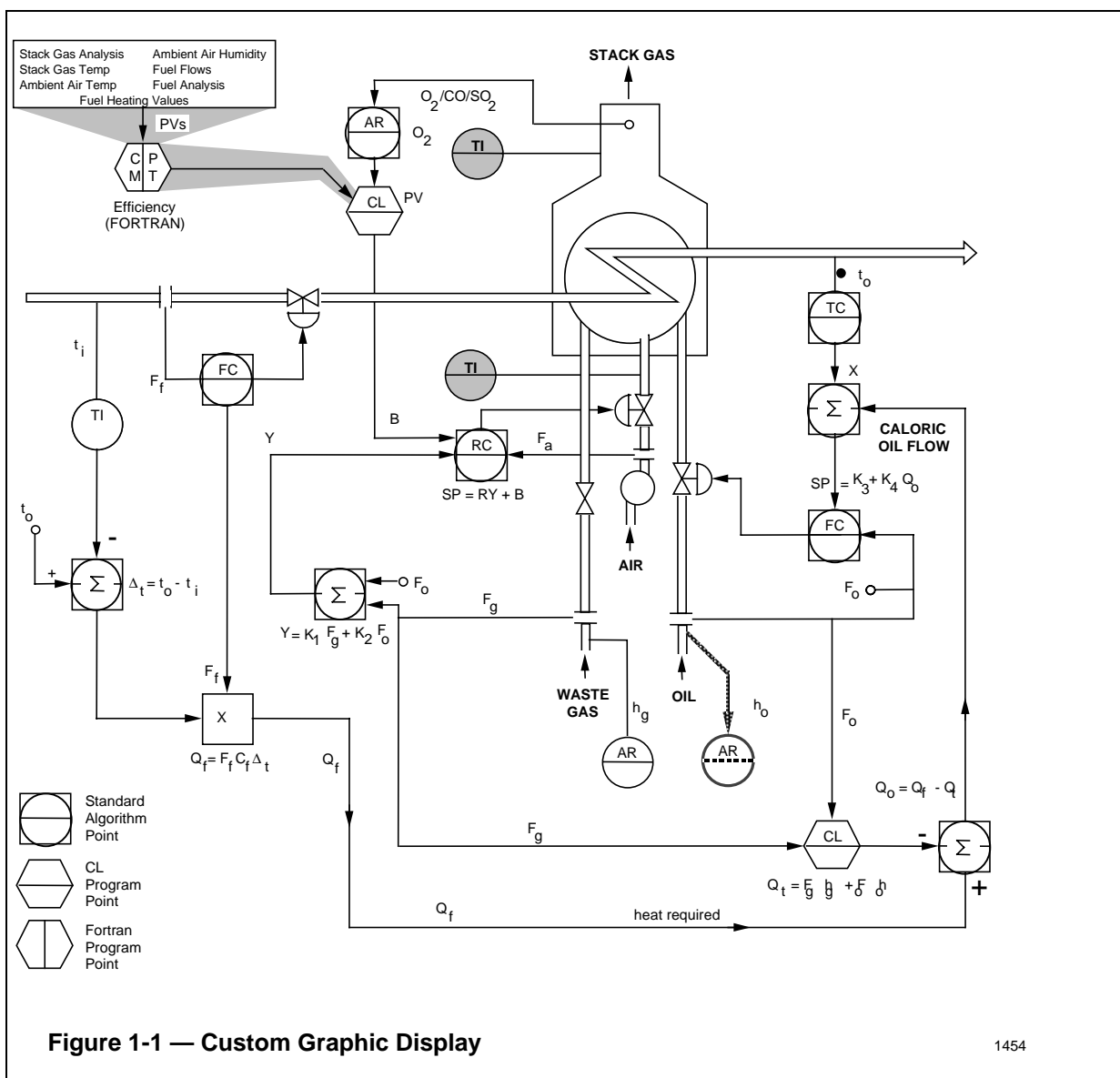


Figure 1-1 — Custom Graphic Display

1454

### 1.1.1.2 Command Overview

- General Purpose Commands

These are miscellaneous commands needed to clear the screen, to name/store/retrieve the display, and to move/copy/delete objects in the display.

- Drawing Lines and Shapes

These commands are used to draw and connect graphics objects, such as solid or hollow shapes, and connecting lines that represent pipes, etc. There are commands to enlarge/reduce or to reverse the direction of those objects.

- Behavior Commands

These commands are used to assign color, blink, intensity level, and reverse video. Conditional behavior commands allow objects to change behavior with changes in the process.

- Text

These commands add labels or other text to the picture.

- Values

These commands allow the picture to contain dynamic process measurements (i.e., "live" data such as pressure, temperature, flow, etc. appear on the screen).

- Variants

These commands make words or pictures change with process conditions.

- Bar Charts

These commands are used to build vertical or horizontal bars that change length according to process measurements.

- Targets

Target commands designate areas on the screen that can be activated by touch or by cursor. Activation causes user-specified action, such as calling up other displays, manipulation of the process, etc.

- Subpictures

Frequently needed pictures can be drawn once and used repeatedly as subpictures in the same or multiple displays. Subpictures can also be called into a display by a Variant.

### 1.1.1.3 Build Time and Run Time

In this publication, the term build time refers to the display-building phase, that is, when using the Picture Editor functions of the Universal Personality.

Run time means the operating functions are in use. Generally, it is only during run time that features built into the displays can be observed in their operating state.

If the Universal Personality is running, both the engineering functions (such as the Picture Editor) and the operating functions are available. Generally, the engineering functions are active when you enter a function through the Engineering Main Menu. If you exit the engineering functions and select an operating display, the operator functions are active. Only operator functions are available when the Operator Personality is running.

## 1.2 REFERENCES

Title	Binder
<i>Actors Manual</i>	Implementation/Engineering Operations – 2
<i>Picture Editor Form Instructions</i>	Implementation/Engineering Operations – 3
<i>Picture Editor Data Entry</i>	Implementation/Engineering Operations – 2
<i>Free Format Log Data Entry</i>	Implementation/ Engineering Operations – 1
<i>Button Configuration Form Instructions</i>	Implementation/ Engineering Operations – 1
<i>Button Configuration Data Entry</i>	Implementation/ Engineering Operations – 1
<i>System Startup Guide, Cartridge Drive</i>	Implementation/Startup & Reconfiguration – 1
<i>System Startup Guide, CD ROM</i>	Implementation/ Startup & Reconfiguration – 1
<i>System Startup Guide, Zip Drive</i>	Implementation/ Startup & Reconfiguration – 1
<i>Hiway Gateway Parameter Reference Dictionary</i>	Implementation/ Hiway Gateway – 1
<i>Application Module Parameter Reference Dictionary</i>	Implementation/AM – 2
<i>Computer Gateway Parameter Reference Dictionary</i>	Implementation/ Computer Gateway
<i>PM Parameter Reference Dictionary</i>	Implementation/PM – 2
<i>APM Parameter Reference Dictionary</i>	Implementation/APM – 2
<i>HPM Parameter Reference Dictionary</i>	Implementation/HPM - 2



## 2 TOOLS FOR PICTURE BUILDING

### 2.1 FORMS AND FORM INSTRUCTIONS

Honeywell provides a series of paper forms to collect the information necessary to create a custom display. The *SW88-651* display form is used to make a sketch of the intended display. Forms *SW88-652* through *SW88-659* are used to collect supporting information and to sketch subpictures. The *Picture Editor Form Instructions* manual (see References), describes the basic objects that can be built with the Picture Editor and explains how to use the forms. You should study that manual before attempting to design a custom graphic display.

### 2.2 DATA ENTRY INSTRUCTIONS

Until you become familiar with the display-building process, you should study Honeywell publication, *Picture Editor Data Entry*, which tells how to enter information collected on the paper forms into the Universal Station. The entry process consists of using the engineering keyboard to draw lines and to type data onto the screen. Entering some functions causes the Picture Editor to present a form on the screen and request, line-by-line, the information collected on the paper support forms.

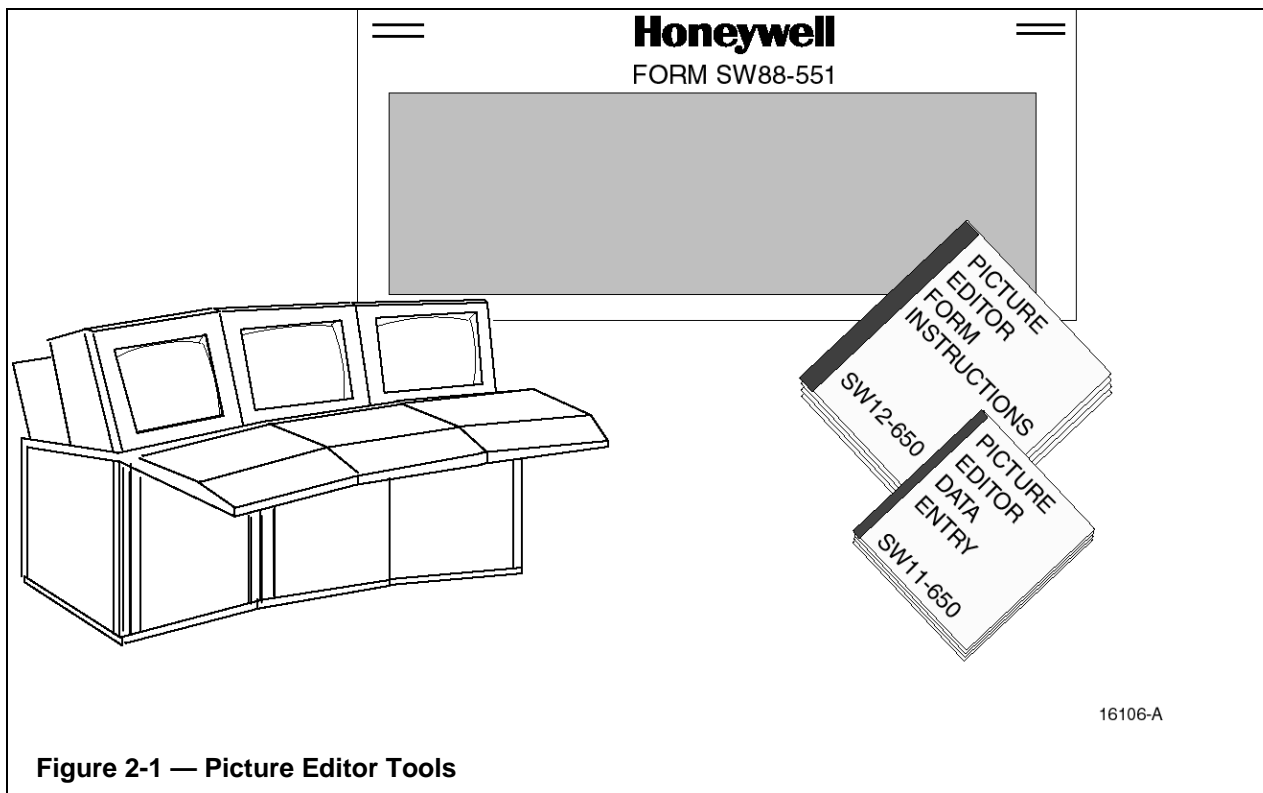


Figure 2-1 — Picture Editor Tools

## 2.3 COMMANDS

This manual provides detailed information about the Picture Editor commands, how they are used, and the exact nature of supporting information requested by the editor.

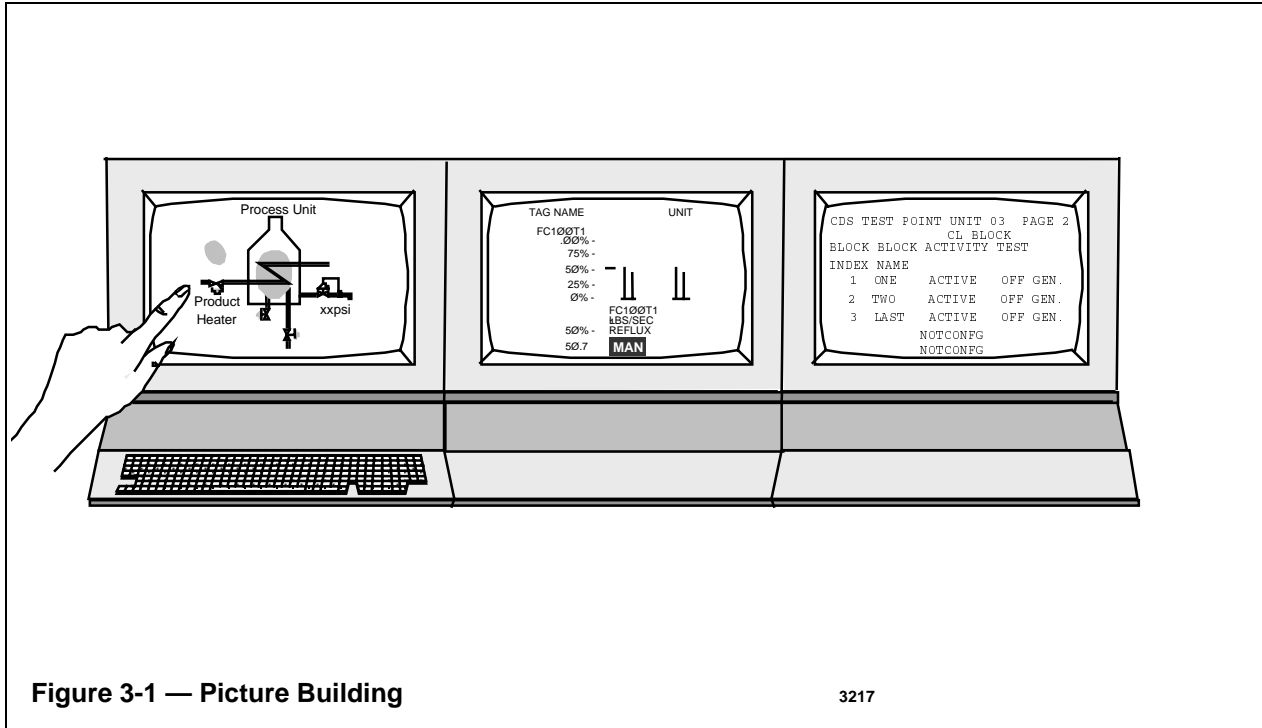
## 2.4 FREE FORMAT LOGS

While the FREE FORMAT LOG building function is called up by selecting a separate target on the Engineering Personality Main Menu, the Picture Editor is actually used to build Free Format Logs; however, there are some restrictions. In general, these are:

- Literal behavior is restricted to cyan, full intensity, no blink, no reverse video. Conditional behavior is not allowed.
- When entering the Free Format Log builder, the edit region is automatically rolled to the upper-left corner of the total drawing area.
- Text size is fixed in large mode.
- Only text, values, variants, and subpictures can be used in log displays. If variants or subpictures are used, they cannot contain any objects other than text, values, variants, or subpictures. Application subpictures are not allowed nor are graphics (lines, solids, bars etc.)
- The total drawing area is fixed at 132 columns wide and 66 lines deep (equivalent to one standard size printed sheet).
- When compiling, any object that is not valid in a Free Format Log is flagged with an error. The maximum area compiled is limited to 132 columns by 66 lines.
- The commands that can be used to build Free Format Logs are a limited set of the Picture Editor commands.

Refer to the *Free Format Log Data Entry* manual (see References), for more information and a list of the legal commands.

## 3 PICTURE BUILDING



### 3.1 SCREEN FORMAT FOR PICTURE BUILDING

Figure 3-2 shows the screen layout used by the Picture Editor. A careful examination of this format will help you to understand the command descriptions.

Top line—The current date, time, and station number appear on the right side.

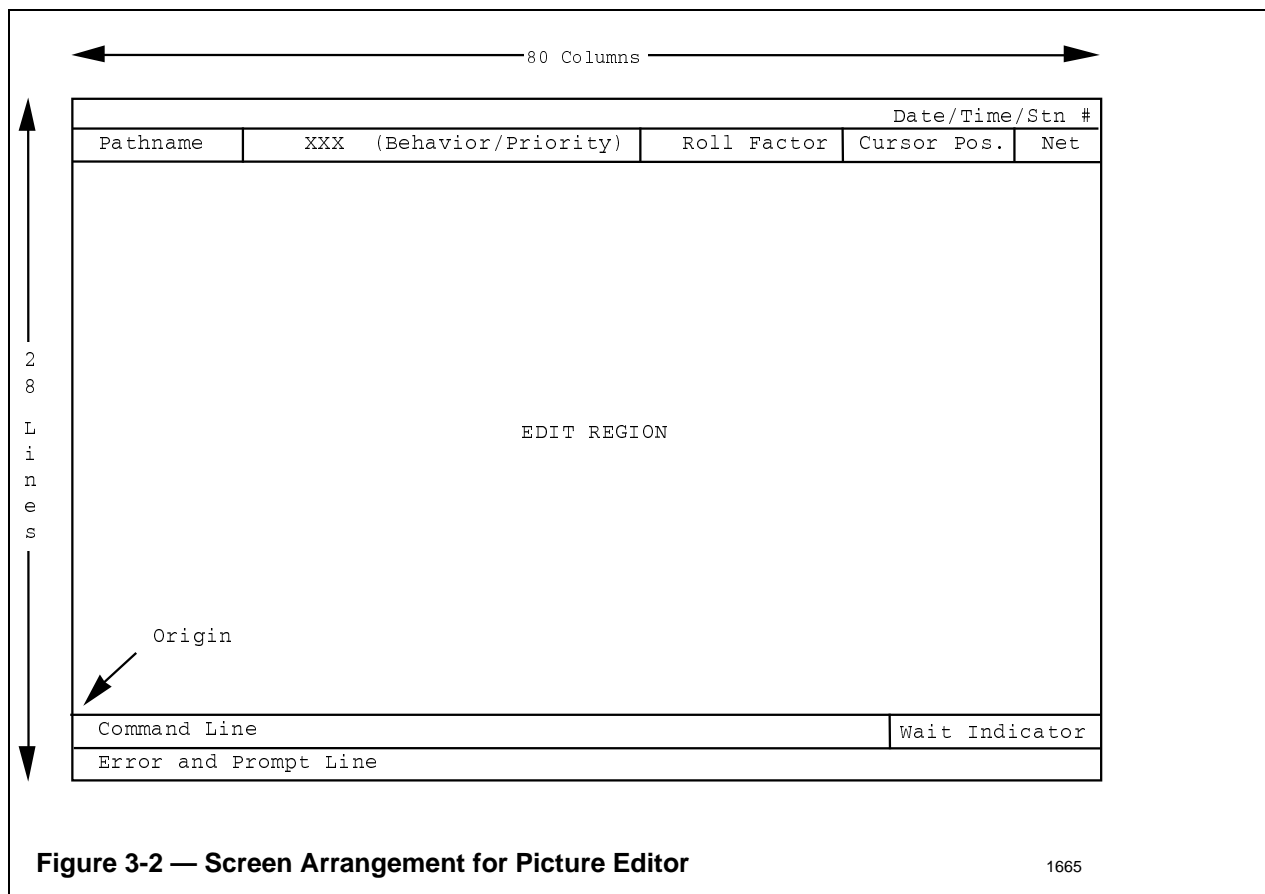
Second Line—Information on the second line is called the current settings. From left to right this is

- The pathname for the current source file—Pathnames are discussed in the General Purpose Commands section of this manual.
- A 3-letter priority code such as FGB, displayed in the current text size and behavior (color, intensity, etc.)—Text can be entered in large size (the default) or small size. Current behavior and priority settings deal with the next screen information entered. Behavior characteristics are color, intensity, steady state/blink, and reverse/normal video. Priority determines what is seen when parts of text or values overlap with graphic objects. Priority is discussed in the General Purpose Commands section of this manual.
- The roll factor—The viewable screen area is 80 columns wide by 28 lines long (see Figure 3-2) but, as explained later, the total available drawing area is much larger. The picture is built in the 80-column by 24-line edit region (assuming large size text). The edit region can be moved anywhere in the full drawing area by executing the Set Roll command. The roll factor is a number representing the current position (in character cells) for the origin of the edit region. A character cell is 8 picture elements (pixels) wide by 16 pixels high for large size text (unless otherwise stated, large size text is assumed in this manual when discussing lines or character cells). The Set Roll command is described in the General Purpose Commands section of this manual.
- The cursor position (during data entry), in pixel units relative to the bottom left-hand corner—The viewing area is 384 pixels from top to bottom and 640 pixels wide.
- Edit Region—the current work area (24 lines by 80 columns).
- Command Line—Picture Editor commands appear on the command line as they are typed at the Engineering Keyboard. At the far right, the word WAIT appears when the Picture Editor is busy.
- Prompt/Error Line—Many commands require additional input, so this line is used to request coordinate or other information when it is needed. Errors are reported on this line by the Picture Editor.
- Net—Indicates the current network access mode. N-ON means the Picture Editor has access to the network for type-checking of variables. N-OFF means network searching is disabled.

### 3.1.1 Entering Commands

Picture Editor commands are entered through the engineering keyboard by typing the name, or an abbreviated form of the name, on the command line (see Figure 3-2) and pressing the ENTER key.

Multiple commands can be entered by separating the command names with a semicolon, for example, SELECT; COPY; SELECT; SCALE. Each command is executed in order. The Picture Editor Data Entry manual contains more information and examples of command entry.



### 3.1.2 Command Abbreviations

Most command words have an abbreviated form of one to four letters that can be used instead of the full command. In the command descriptions that follow, the full command is followed by its abbreviated form(s) in parentheses. In some cases where confusion might result, the abbreviated form is shown separately. Either the full command or the abbreviated form can be used. For example A L is equivalent to ADD LINE when invoking the Add Line command.

### 3.1.3 Specifying Locations

Many commands require that one or more screen locations be specified, for example, the ends of a line. Locations can be entered in one of the following ways:

- Move the cursor to the desired screen location and press the SELECT key.
- If the touch-screen option is present, touch the screen at the desired location.

- Type the X-Y coordinates (in pixel units) on the command line, following the command. Example: ADD LINE 40 200 100 300 causes a line to be drawn between X-Y coordinates 40, 200 and 100, 300. Note that if an object is added (or moved) to a location that is out of the edit region, it cannot be seen in the picture at operating time.

### 3.1.4 Errors

If the Picture Editor detects illegal input, an error message is printed, illegal input is changed to red, and the command is stopped. To recover, type the correct information and press ENTER.

**Error symbols**—The following symbols are used throughout the Picture Editor-

Symbol	Error Type	Comments
?	Communication	May be communication across the LCN
*	Format Error	Often appears when screen space allotted is too small to present value returned
@	Configuration	Wrong ID, wrong parameter type, data unavailable, other.
-	Value Error	Bad value, limit exceeded, etc.
!	Unknown	Interpreter couldn't determine error type

If you get an Abstract Overflow message when writing or compiling a file, there are too many objects in the picture. You may want to split the information into two pictures. Release 400 and earlier users should note that the space available for compiled files has approximately doubled in the Release 500 Picture Editor.

If you get an out of memory message when writing or compiling a file, the specified volume has insufficient space remaining. Write or compile to another volume.

If you are trying to fit more schematics or larger schematics into resident Universal Station memory, you may be able to increase the memory space allocated for custom schematics and Free Format Logs. Refer to section 3.6, Table 3-5 of the *Network Configuration Forms Instructions* manual. To reconfigure an operating US node, also refer to table 7-42 (Modify Node) in the same manual.

### 3.1.5 Help

Whenever the Picture Editor is waiting for command input, several pages of on-screen help information are available. You can access them by either pressing the HELP key on the Engineering Keyboard or by entering HELP on the command line and then pressing the ENTER key. After entering the Help display, use the PAGE FWD or PAGE BACK keys to view the information.

To return to normal Picture Editor functions, press either the CANCEL or HELP key.

## 3.2 COMMANDS

The following picture editor commands are fully described in the **COMMAND DESCRIPTION** section (with suffix; i.e., Set Grid):

COMMANDS	
ADD	MOVE
COMPILE	NEW
COPY	MULTIPLE
COPYCLIP (R530)	PASTE (R530)
COLLECTINH (R600)	PRINT
CUT (R530)	READ
DEFINE	REPLACE
DELETE	SCALE
DESELECT	SELECT
END	SET
LISTEQ	SYSTEM ID
LOAD	UNLOADEQ
RECREATE (R620)	RECREATALL (R620)
MULTRECR (R620)	

## 3.3 COMMAND DESCRIPTIONS

### 3.3.1 General Purpose Commands

#### 3.3.1.1 Command: **NEW (N)**

**Use**—New clears the screen and resets all attributes to their default settings. This is the state when the Picture Editor is started. The behavior default values are

- Full Intensity
- No Blink
- No Reverse
- Cyan

The priority default is Foreground, Graphics, Background (FGB).

**Procedure**—Type NEW on the command line and press the Enter key.

**CANCEL**—If the CANCEL key is pressed at this point, the picture that was visible before the New command is redrawn.

#### 3.3.1.2 Command: **END (E)**

**Use**—This command is used to end the Picture Editor session and return to the main menu.

**Procedure**—Type END on the command line and press the Enter key.

If you have modified the Picture without executing a New, Write, or Compile command, the Picture Editor warns `Modified Picture Exists. End?` Press the CANCEL key to revoke the command, or press the ENTER key to exit from the Picture Editor.

#### 3.3.1.3 Command: **SET ROLL (S ROL) (S R)**

**Use**—The edit region can be pictured as a window that rolls over a larger drawing area. When the Picture Editor is started, the edit region is positioned in the bottom-left corner of the drawing area as shown by Figure 3-3 (for Free Format Logs, see Section 2.4). The Set Roll command moves the edit region around the drawing area to permit larger pictures to be built. The PAGE/DISP keys can also be used to roll the edit region. PAGE = vertical; DISP = horizontal. Each time one of these keys is pressed, the edit region is rolled by one-half the screen size, (unless limited by the edge of the drawing surface).

**Procedure 1**—Type SET ROLL on the command line, then press the ENTER key. The Picture Editor responds by requesting the roll coordinate (see Figure 3-4).



Move the cursor to the desired location and press the SELECT key. The location is echoed as a cross. If the touch-screen option is present, just touch the screen at the desired location.

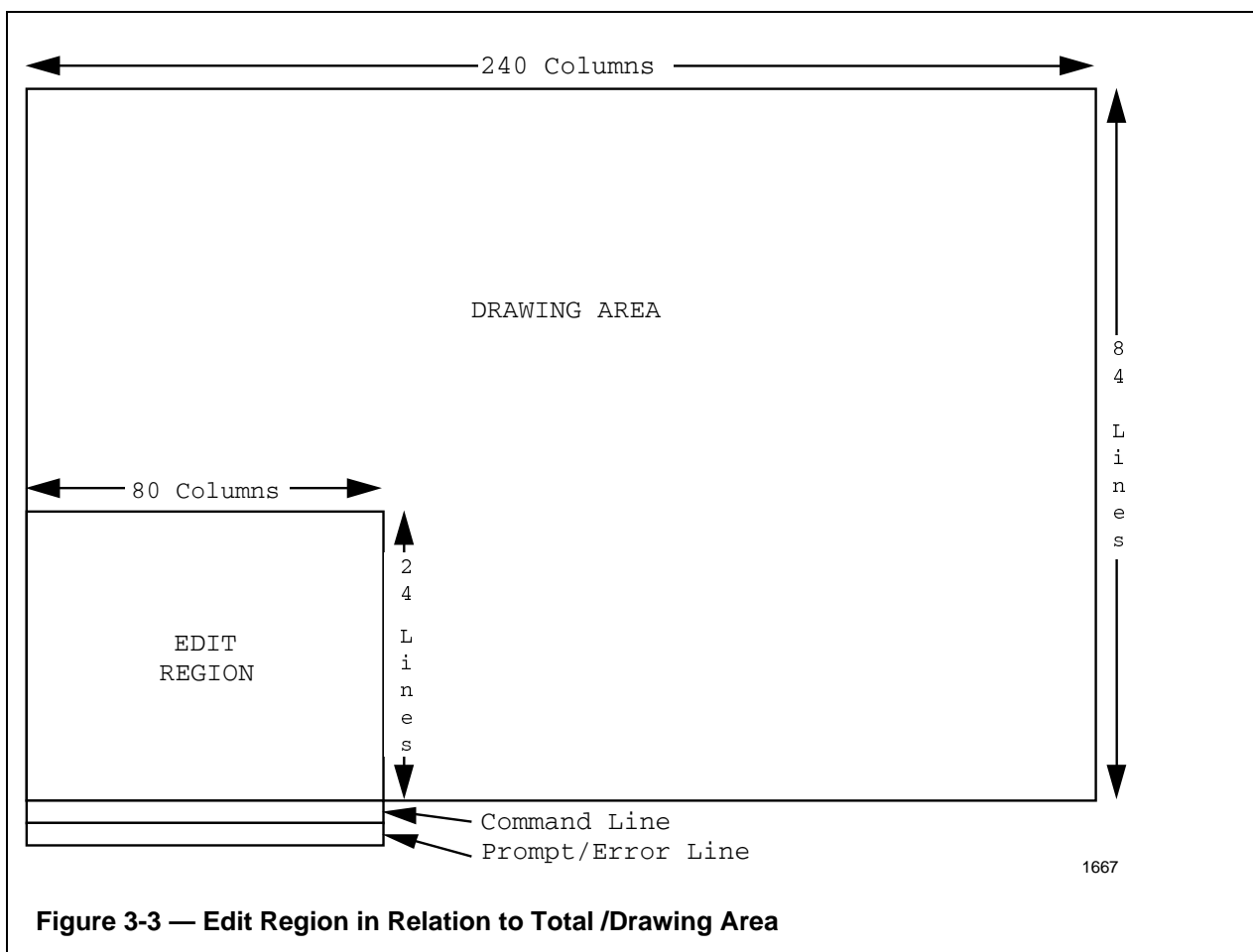
**DEL**—If the DEL key is pressed at this point, the cross is deleted and the location can be respecified.

**CANCEL**—If the CANCEL key is pressed at this point, the command is canceled.

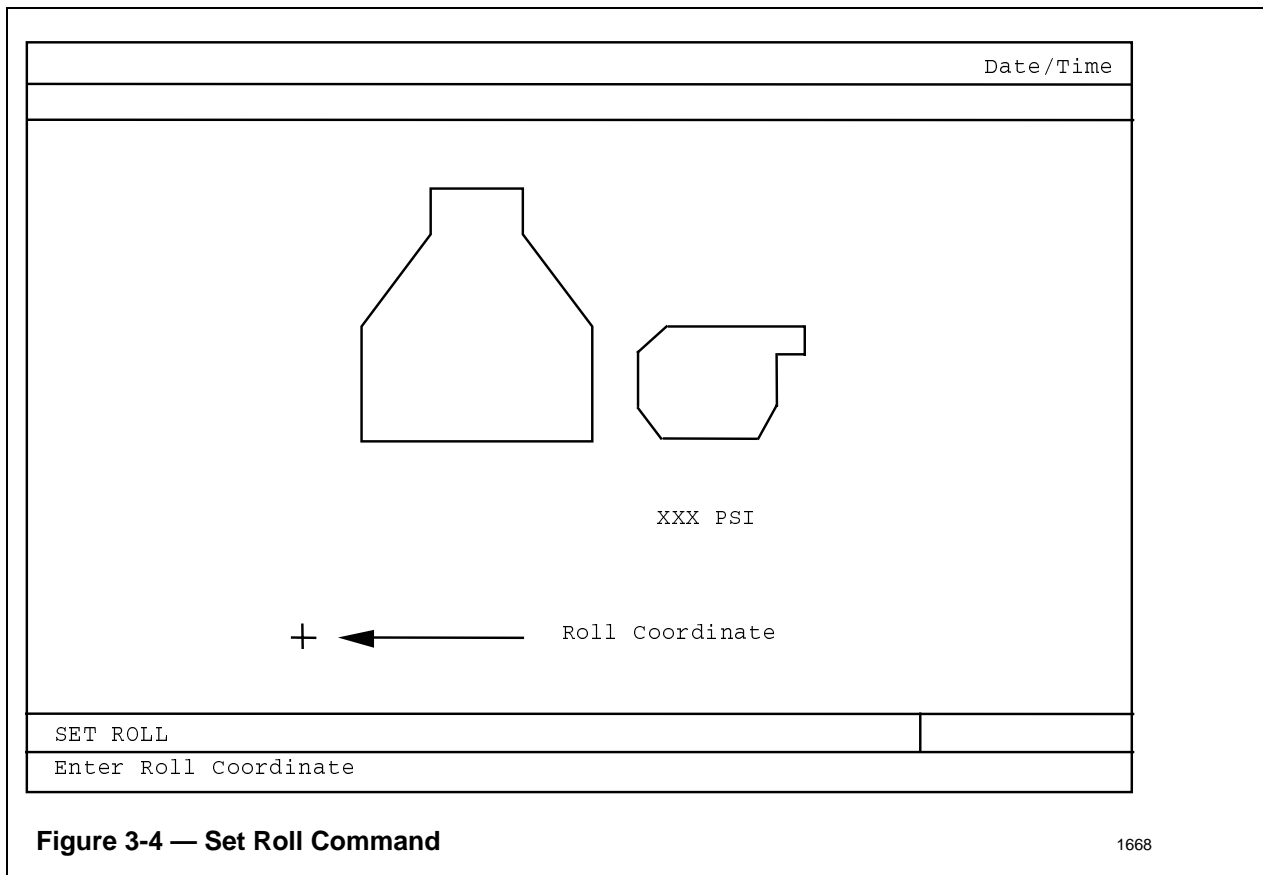
Press the ENTER key to complete the command. As shown in Figure 3-5, the bottom left-hand corner of the screen appears to move to the specified coordinate location. The Roll Factor shown on the top line of the screen changes to reflect the new position of the edit region.

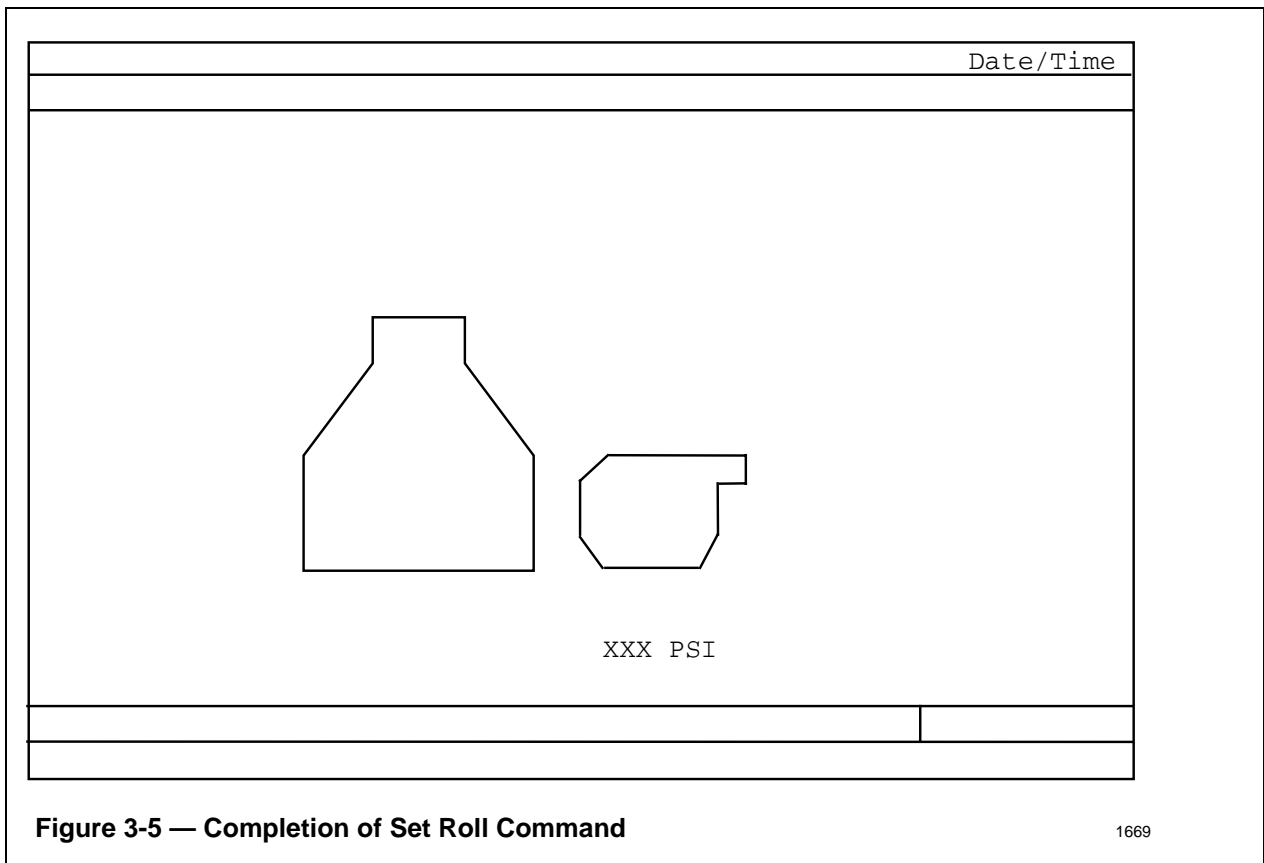
**Procedure 2**—Type SET ROLL XXXX YYYY on the command line, where XXXX YYYY are the roll coordinate in character units. Press the ENTER key to complete the command.

Examples: SET ROLL 2 3 causes the edit region to move right two columns and to move up three lines. SET ROLL 0 0 returns the edit region to its starting position.



**Figure 3-3 — Edit Region in Relation to Total /Drawing Area**





### 3.3.1.4 PRINT Commands

#### 3.3.1.4.1 Command: *PRINT \$Pn zzzz*

where *zzzz* is a print option

The options are

- None
- DEBUG
- COMMENT
- SELECTED (SELEC) (SEL)
- SYMBOLS (SYMBOL) (SYM) (SY)

**Use**—The Print command is used to print a description of the picture currently being displayed. You must designate the printer.

If the Set Pathname command was used to establish a default pathname, that pathname is printed on the listing.

**Procedure**—Type *PRINT \$Pn zzzz* on the command line (*n* is a valid printer identifier and *zzzz* is none or one of the valid print options described below). Press ENTER to complete the command.

If no option is specified, the printout contains a listing of all objects and text used in the picture, including their coordinates, colors, priorities, behaviors, and target actions.

**Examples**— *PRINT \$P1* a listing of all objects in the picture is printed  
*PRINT \$P1 SEL* a description of selected objects in the picture is printed.

The following table describes the other options—

Option	Output
DEBUG	Prints a complete description of objects in the picture.
SYMBOL	Data on variables in the picture
COMMENT	Comment data (i.e., data stored by a Define Comment command)
SELECTED	Information on objects that are selected (also, see text)

You can use either the **SYMBOL** or **SELECTED** option with the **DEBUG** option, e.g., *PRINT \$P1 DEBUG SEL* (some object must be selected in this case).

If the **SELECTED** option is used, the printout contains only those objects that were specified in the command (e.g., `PRINT $P1 SEL TEXT`) or objects selected in the picture before executing the Print command.

Objects that can be specified with the Print Select command are—

BAR	LINE	SUBPICTURE	VALUE
CONDITION	SOLID	TARGET	VARIANT
COMMENT	SPECIAL_TARGETS	TEXT	

Note that—

- If a subpicture is selected, all references to the subpicture are printed, plus one copy of the subpicture's objects including any defined comment pages.
- Special Targets refers to Initial/Final targets (see the Define command) and to defined comments (in the main picture only).

#### **3.3.1.4.2 Command: *PRINT* <pathname> zzzz**

where <pathname> is an optional pathname to a file.  
and zzzz is a print option

Pathname takes the form \$Fn>VOL>FILE to store on removable media, or NET>VOL>FILE to store on the History Module. If no pathname is specified, the default pathname is used. File is the name of a file where the data will be written.

The print options are—

- APPEND
- COMMENT
- DEBUG
- SELECTED (SELEC) (SEL)
- SYMBOLS (SYMBOL) (SYM) (SY)

**Use**—The Print File command is used to output a description of the picture currently being displayed. Except as described below, it works exactly like the Print \$Pn command but the output is to a file. The file can be edited with the Text Editor (it has a .DX extension).

**APPEND**—when output is sent to a file, the file is overwritten with new data unless the Append option is specified. If you specify the Append option, data is appended to a current file. Append is only valid if a pathname is specified, and the word Append directly follows the pathname; e.g., `PRINT NET>HMOV1>MYFILE APPEND SEL`.

**DEBUG**—if this option is specified, the printout contains a listing of all objects and text used in the picture, including coordinates, colors, priorities, behavior, and target actions.

**Example**—`PRINT NET>HMOV1>MYFILE SYM`

### 3.3.1.4.3 Multiple Print Command

**Command:** MPRINT <pathname>

Abbreviated forms: MPRN <pathname>  
MPR <pathname>  
MP <pathname>

where <pathname> is a pathname to a Schematic List file.

**Use**—The Multiple Print command accesses a file that contains a list of schematic source files. It then attempts to generate a schematic print file for each source file named on the list. There is no output to the printer. In addition, it generates a file that contains the result of each print request. The results file is created in the same directory as the schematic list file. The print file contains a list of objects and text in the picture, including their coordinates, colors, priorities, behaviors, and target actions.

**Schematic List File**—Before executing the MPRINT command you must build a Schematic List file with the Text Editor (refer to the *Text Editor Operation* manual). The file name can contain up to 8 characters and must end with a .EL suffix. EL stands for edited list. While using the Picture Editor, you can press the ESC key to temporarily use the Text Editor. Hold down CTL and press HELP when you are ready to return to the Picture Editor.

Note that when the MPRINT command is executed, the Default Pathname is initialized to that path specified for the Schematic List file. As explained below, this may be important if files specified on the list use a partial pathname.

Within the Schematic List file:

Any line beginning with { (a left curly brace) is treated as a comment line and is not executed. If a comment requires more than one line, each line must begin with { to be treated as a comment. A right curly brace (}) is optional at the end of a comment line.

Schematic source file names can be specified with a full or partial pathname. The .DS extension is optional. If a partial pathname is specified, it is used in conjunction with the Default Pathname. Specification of a full pathname reinitializes the Default Pathname.

An example of a Schematic List file follows:

```
$F2>WORK>SCHEM1
NET>HNV1>SCHEM2.DS
{The following file should execute as NET>HNV1>SCHEM3.DS}

{The next file should execute as NET>HNV1>SCHEM4.DS}
SCHEM4
{Use of the right curly brace on comments and addition {of
the .DS extension on source files is optional.
```

**Procedure**—On the command line, type MPRINT followed by a full or partial pathname for the Schematic List file, then press the ENTER key.

For example: MPRINT NET>HNV1>SLIST

The Default Pathname is modified to use the supplied name. Refer to the Set Pathname command for acceptable forms of the pathname. Press ENTER to continue or press CANCEL to cancel the command. After pressing ENTER, execution begins and continues until each file on the list is processed.

If you wish to halt before the operation is complete, hold down CONTROL and press BREAK. After the current file is processed, the operation is aborted.

**Options**—within the Schematic List file, several options can be added following the Schematic Source file names:

- APPEND will append the printed information to the existing print file for that source file.
- COMMENT will print only the comment sentences for that source file.
- DEBUG will include the debug sentences for that source file.
- SYMBOLS will print only the symbols sentences for that source file.

For example: NET>HNV1>SCHEM1 DEBUG

**Errors**—If an error occurs during execution, a print error is placed in the Results file and processing of the list file continues.

If the storage media becomes full during execution of the Multiple Print command, a print error is placed in the Results file for that schematic name and all remaining schematic names in the list file. If the Results file cannot be written to the storage media, then an error message appears on the screen.

Each Schematic Print file is generated in the same directory as its respective schematic source file. The Print file has the same name as the Schematic Source file, but has a .DX suffix.

Each Schematic Print file is ASCII formatted. Any Schematic Print file with the same name that existed before executing the Multiple Print command is overwritten (except see the Append option).

**Results File**—A Results file is produced by the Multiple Print command. It is generated in the same directory and with the same name as the Schematic List file except it has a suffix of .ER. Press the ESC key to temporarily use the Command Processor. Use the Print command to view the Results file. For example, PR Net>HNV1>SLIST.ER.

The Results file contains one line for each executable file name specified in the Schematic List file. Each line shows the full or partial file pathname and a print status. List file comments are not included. The .DS suffix does not appear in the Results file even if it was used in the Schematic List file.

If a previous Results file with the same file name exists, it is overwritten.

Only the first 30 characters of a defined pathname for each line in the list file appear in the Results file.

If the Multiple Print operation is halted by pressing the CONTROL-BREAK keys, the Results file contains the message: *Aborted by User Request*.

An example of a Results file is:

```
$F2>WORK>SCHEM1      Success!
NET>HVM1>SCHEM2      Print Operation Failed
SCHEM3                Read Operation Failed
SCHEM4                Success!
```

To view the individual schematic print files, escape from the Picture Editor, then use the Command Processor functions to view each file. For example:

PR NET>HVM1>SCHEM1.DX.

If you precede the print command with a Data Out command to the printer (DO \$Pn), you can obtain a paper copy.

### 3.3.1.5 Command: SET PATHNAME nnnn, (S P) nnnn

where the pathname is nnnn

**Use**—The Set Pathname command is used with the Write, Read, and Compile commands to specify a storage device volume and filename for the picture source and object files. The current pathname is used as the default pathname when entering the Picture Editor.

**Procedure**—Type SET PATHNAME nnnn on the command line, then press the ENTER key.

where the pathname nnnn consists of

- the word NET or a valid Logical Device Identifier (LDID)
- followed by a greater than symbol, >
- followed by a valid volume identifier
- followed by a greater than symbol, >
- followed by a valid filename, excluding the suffix.

Examples (using the abbreviated command form) are S P NET>HVM1>FRED  
S P \$F4>V123>FRED

After the full pathname is assigned, fields within the filename can be changed by entering only the changed part, for example, if the current pathname is NET>HVM1>FRED

- entering S P JOHN or S P >JOHN results in the pathname NET>HVM1>JOHN.
- Entering S P VOL2> or S P >VOL2> changes the volume identifier and results in the pathname NET>VOL2>FRED.

The word NET refers to the History Module.



A valid Logical Device identifier begins with \$ and contains a maximum of 10 characters. For example, \$F1, \$F12, etc. (\$F refers to a Floppy Disk Drive). The 10 characters can be alphabetic, numeric, underscore (\_), or dollar signs (\$).

A valid volume identifier contains up to four uppercase alpha or numeric characters. The following restrictions apply:

- It can contain underscore ( ) characters but not two consecutive underscore characters.
- It cannot begin or end with underscore characters.
- It cannot contain all numbers.
- It can contain exclamation points (!) and/or ampersands (&).

A valid filename can contain up to eight characters. Alphabetic, numeric, underscore ( ), and/or dollar sign (\$) characters can be used in any position.

The new pathname replaces the old pathname at the top of the screen. It is used when writing to or reading picture information from a source file, or when compiling.

#### **3.3.1.6 Command: SET NETWORK mmm,**

Abbreviated Forms: S NET mmm  
S N mmm  
where the mode is mmm

**Use**—The Set Network command is used to enable or disable variable type-checking through the data-access network. The mode (mmm) choices are ON or OFF. If ON is specified, network searching is enabled. If OFF is specified, network searching is disabled. Network access status is shown as N-ON (network on) or N-OFF (network off) on the second line from the top of the screen.

**Procedure**—Type SET NETWORK mmm on the command line, then press the ENTER key  
(mmm = ON or OFF).

Example: SET NETWORK OFF

**CANCEL**—If the cancel key is pressed after the command is completed, network searching reverts to the previous setting.

This feature is primarily used to disable type checking when the Universal Station is disconnected from the network. If network searching is enabled, variables not found in the local database and not previously defined will be presented to the data-access network for type checking. If the variable is not located, you are prompted for the variable type.

If network searching is disabled, variables not found in the local database and not previously defined will not be presented to the data-access network for type checking. Instead, you will be immediately asked for the type of the variable.

### 3.3.1.7 Command: WRITE (W)

**Use**—This command writes the current picture to a source file. The pathname used by the command is the current pathname shown at the top of the screen.

**Procedure**—Type WRITE on the command line, then press the ENTER key.

If the file exists, a warning that the source file will be overwritten is displayed. The source file is assigned a .DS suffix.

**ENTER**—Press the ENTER key again to allow the write. If the file does not exist, it is created and the write proceeds.

**CANCEL**—If the CANCEL key is pressed the write does not occur.

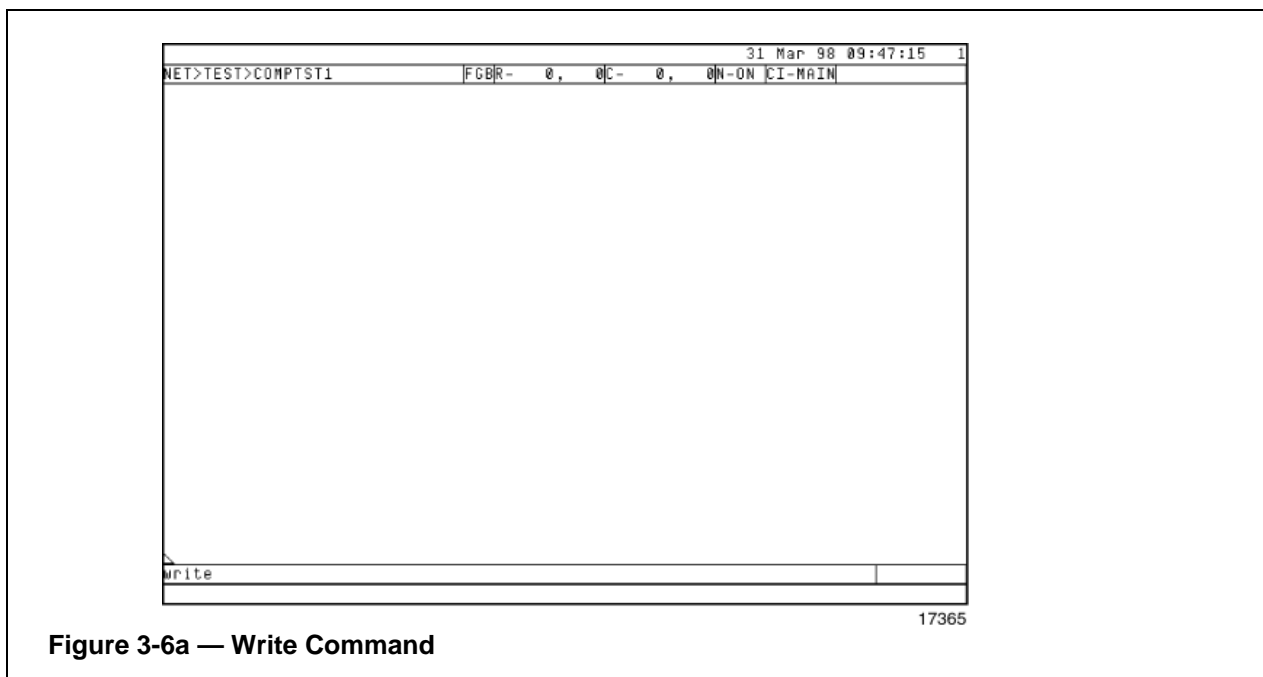


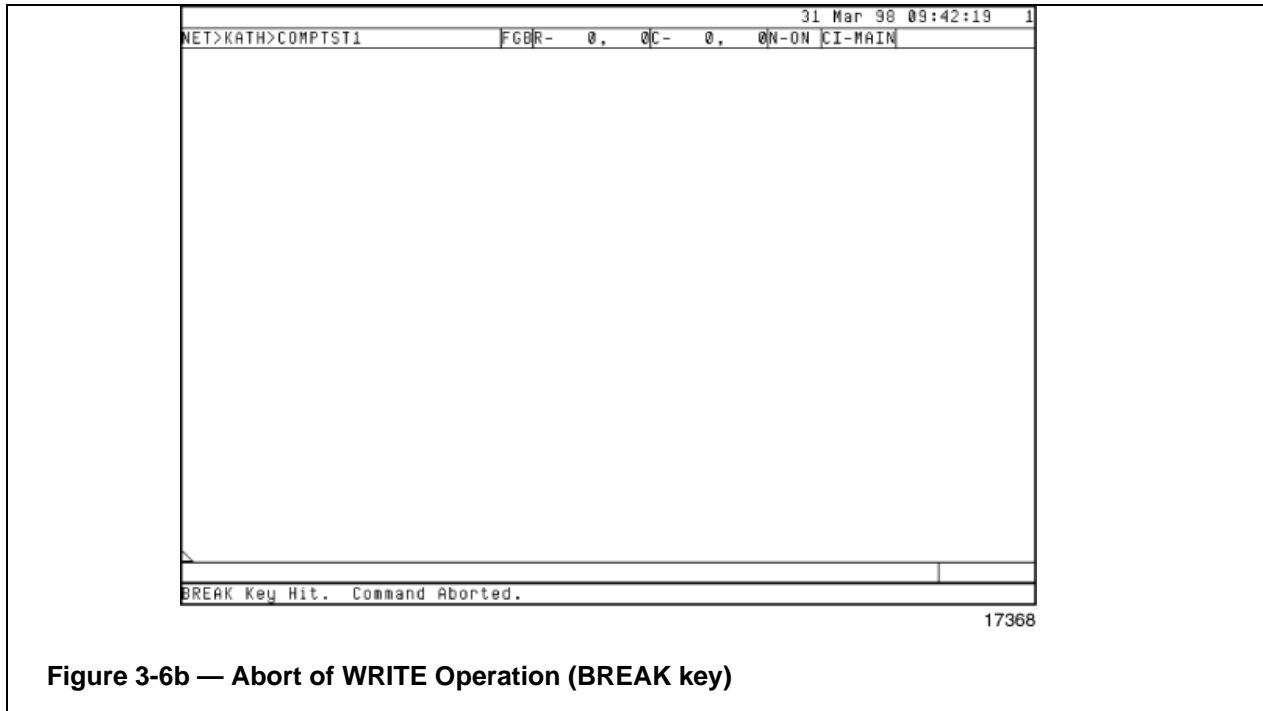
Figure 3-6a — Write Command

### (R600) Picture Editor Check for BREAK Key

The BREAK key is activated to allow the abort of the WRITE operation. If the BREAK key is pressed, the WRITE command terminates and the picture is left in the state prior to the execution of the command. A message BREAK key Hit. Command Aborted is output to inform that the command was ended by the BREAK key (see Figure 3-6b).

The BREAK key also is activated to allow the abort of the WRITE command when there is a prompt for input. During the overwrite prompt and the parameter form prompt, the CANCEL key is pressed to abort the operation. The BREAK key can also be used interchangeably with the CANCEL key in these steps.

During the WRITE operation, the word `WAIT` appears at the right-end of the command line. If you get an Abstract overflow message, refer to 3.1.4 Errors.



**Figure 3-6b — Abort of WRITE Operation (BREAK key)**

### 3.3.1.8 Command: **WRITE nnnn**

Abbreviated form: **W nnnn**

where nnnn is the full or partial pathname.

**Use**—This command writes the current picture to a source file. Refer to the Set Pathname command for acceptable forms of nnnn.

**Procedure**—Type **WRITE nnnn** on the command line and press the ENTER key (nnnn is the full or partial pathname).

Example: **W SAM** changes the file name of the current pathname to **SAM** during this Write command.

If the file exists, a warning that the source file will be overwritten is displayed. The source file is assigned a .DS suffix.

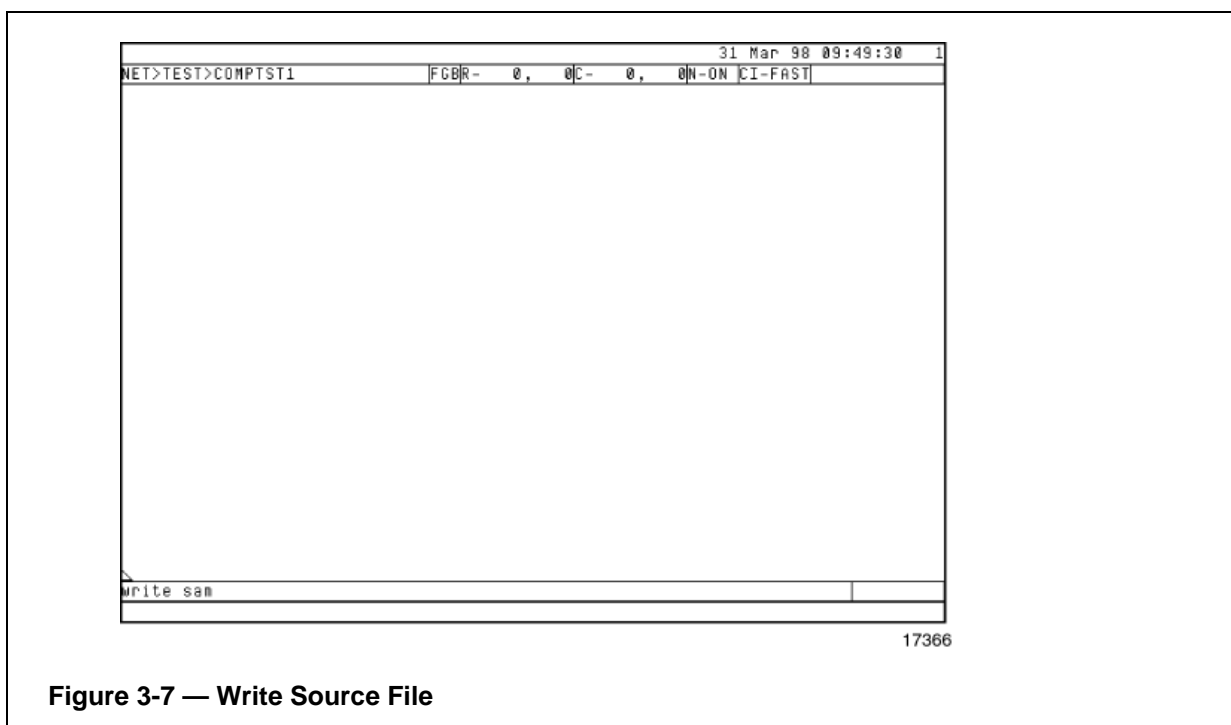
**ENTER**—Press the ENTER key again to write. If the file does not exist, it is created and the write proceeds.

**CANCEL**—If the CANCEL key is pressed, the write does not occur.

#### **(R600) Picture Editor Check for BREAK key**

See command: **WRITE (W)** in the Compile section (this manual) for R600 BREAK key change information.

During this command, the pathname used for the **WRITE** operation replaces the file name in the pathname on the second line from the top of the screen (see Figure 3-7); thereafter, the pathname returns to the previous pathname. During a **WRITE** operation, the word **WAIT** appears at the right-end of the command line.



**Figure 3-7 — Write Source File**

### 3.3.1.9 Command: READ (REA) (RD) (R)

**Use**—This command is used to read a previously constructed source file into the picture editor. The pathname used is that established by the Set Pathname command and is the pathname shown at the top of the screen.

**Procedure**—Type READ on the command line, then press the ENTER key.

During the read operation, the word WAIT appears at the right-end of the command line (see Figure 3-8).

**CANCEL**—If the CANCEL key is pressed at this point, the picture that was present before the Read Command is restored to the screen.

If a Read command is executed when the display currently on the screen has been changed, but not saved, you will be warned before the Read occurs.

		Date/Time
\$F1>VOL1>FRED		
PICTURE		
READ		

**Figure 3-9 — Read Command**

1672

### 3.3.1.10 Command: READ nnnn

Abbreviated forms: REA nnnn  
RD nnnn  
R nnnn

**Use**—This command is used to read a previously constructed source file into the picture editor. The pathname used is that specified in the command (nnnn), and can be a full or partial pathname as explained in the Set Pathname command discussion.

**Procedure**—Type READ nnnn on the command line, then press the ENTER key (nnnn represents the pathname).

Example: R SAM

The file name SAM replaces the current file name in the pathname at the top of the screen during this command.

During the read operation, the word WAIT appears at the right end of the command line (see Figure 3-9).

**CANCEL**—IF THE CANCEL key is pressed at this point, the picture that was displayed before the Read command reappears on the screen.

The previous pathname returns at the completion of this command.

If a Read command is executed when the display currently on the screen has been changed, but not saved, you will be warned before the Read occurs.

		Date/Time
\$F1>VOL1>FRED		
PICTURE		
READ SAM		

**Figure 3-9 — Read Source File**

1673



### 3.3.1.11 Compile Commands

#### 3.3.1.11.1 Command: **COMPILE (COMP) (COM)**

or

**Command: COMPILE nnnn**

Abbreviated forms: COMP nnnn  
COM nnnn

**Use**—These commands are used to generate an object file for the picture currently on the screen. An object file must be generated before the picture can be used during on-line operations. The system must be On-Net (i.e., connected to the network) to successfully compile.

This command can also be used to check the syntax of expressions in a subpicture. If objects in the subpicture have errors, these objects are automatically selected, which allows them to be easily modified. If the subpicture contains a parameter, but does not have any errors, the message Cannot Compile Subpicture appears.

**Procedure**—On the command line, type COMPILE, or COMPILE followed by a full or partial pathname, then press the ENTER key. If a pathname is not specified, the full default pathname is used. If a full or partial pathname is specified, the pathname is modified to use the supplied name. Refer to the Set Pathname command for acceptable forms of nnnn. If the specified file already exists, a warning is displayed, File Already Exists. Overwrite? Press ENTER to continue, or press CANCEL if you do not want to overwrite the file. Figure 3-11 illustrates the Compile command.

The picture is first checked for errors. If any errors are present, faulty objects in the picture are selected and the compile fails. If no errors are present, the source file is written (see the WRITE commands), then the object file is created. The object file has a .DO suffix.

#### **(R600) Picture Editor Check for BREAK key**

The BREAK key is activated to allow the abort of the COMPILE operation. If the BREAK key is pressed, the COMPILE command terminates and the picture is left in the state prior to the execution of the command. A message BREAK key Hit. Command Aborted is output to inform that the command was ended by the BREAK key. The BREAK key is activated to allow the abort of the compile through the end of the writing of the source file. To determine when the source file is finished writing, a message Compiling Object File is output informing that the object file is now being compiled. (See additional detail in the WRITE command section of this manual.)

If a full or partial pathname is specified, the pathname at the top of the screen is altered to show the pathname used during the command. Thereafter, the current pathname reverts to its previous form.

### CAUTION

The picture you want to compile must be on-screen when you execute the Compile command. The specified source file is overwritten with whatever is currently on the screen. When you change a picture that previously compiled OK, consider first compiling to a temporary file so that if the changed picture contains errors, the old source is preserved.

**Objects off the drawing surface**—If you get this error message when attempting to compile, some object(s) may have been accidentally moved or copied to a position off the allowable drawing area (see Figure 3-3). Because all such object(s) are **selected** by the compile attempt, you can immediately use a Delete command to erase the object(s) or you can try to bring the object(s) back onto the drawing area with a Move command. Note that an object that is off the current screen, but within the drawing area as shown in Figure 3-3, does not cause this error message.

**Cannot Compile**—If you get this message or some other error message when attempting to compile, execute a Verify command, then try to compile again. The Verify commands are described elsewhere in this publication.

If you get an `Abstract Overflow` message, refer to 3.1.4 Errors.

For Free Format Logs see subsection 2.4 in this manual.

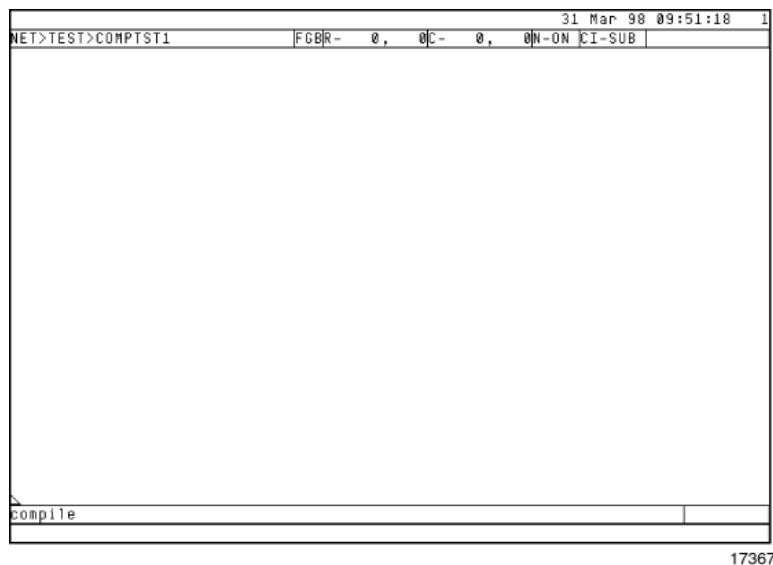


Figure 3-10 — Compile Command

### 3.3.1.11.2 Multiple Compile Command: **MCOMP** nnnn

Abbreviated forms:   MCOM nnnn  
                          MC nnnn

where nnnn is a full or partial pathname to a schematic List File

**Use**—The Multiple Compile command accesses a file that contains a list of schematic source files. It then attempts to generate an object file for each source file named on the list. In addition, it generates a file that contains the result of each compilation. The results file is created in the same directory as the schematic list file.

Each schematic source file on the list is overwritten as the multiple compilation begins. Like the single compile operation, object files are generated in the same directory as the source file and have a .DO suffix. Any of the listed schematic object files that exist before executing the Multiple Compile command are overwritten if the associated source file is successfully compiled. Unlike the regular Compile command, you do not need to read the schematic being compiled to the screen.

**Schematic List File**—Before executing the MCOMP command you must build a Schematic List file with the Text Editor (refer to the *Text Editor Operation* manual). The file name can contain up to 8 characters and must end with a .EL suffix. EL stands for edited list. While using the Picture Editor, you can press the ESC key to temporarily use the Text Editor. Hold down CTL and press HELP when you are ready to return to the Picture Editor.

Note that when the MCOMP command is executed, the Default Pathname is initialized to that path specified to the Schematic List file. As explained below, this may be important if files specified on the list use a partial pathname.

Within the source file:

Any line beginning with { (a left curly brace) is treated as a comment line and is not executed. If a comment requires more than one line, each line must begin with { to be treated as a comment. A right curly brace (}) is optional at the end of a comment line.

Schematic source file names can be specified with a full or partial pathname. The .DS extension is optional. If a partial pathname is specified, it is used in conjunction with the Default Pathname. Specification of a full pathname reinitializes the Default Pathname.

An example of a Schematic List file follows:

```
$F2>WORK>SCHEM1
NET>HNV1>SCHEM2.DS
{The following file should execute as NET>HNV1>SCHEM3.DS}
SCHEM3.DS
{The next file should execute as NET>HNV1>SCHEM4.DS}
SCHEM4
{Use of the right curly brace on comments and addition   {of
the .DS extension on source files is optional.
```

**Procedure**—On the command line, type MCOMP followed by a full or partial pathname for the Schematic List file, then press the ENTER key.

For example: MCOMP NET>HVM1>SLIST

The Default Pathname is modified to use the supplied name. Refer to the Set Pathname command for acceptable forms of the pathname. Press ENTER to continue, or press CANCEL to cancel the command. After pressing ENTER, compilation begins and continues until each file on the list is processed.

If you wish to halt any remaining compilation, hold down CONTROL and press BREAK. After the current file is processed, the operation is aborted.

**Errors**—If an error occurs during compilation, the error is placed in the results file and the list file is processed to completion.

If the storage media becomes full during execution of the Multiple Compile command, a compile error is placed in the results file for that schematic name and all remaining schematic names in the list file. If the Results file cannot be written to the storage media, then an error message appears on the screen.

**Results File**—A Results file is produced by the Multiple Compile command. It is generated in the same directory and with the same name as the Schematic List file except it has a suffix of .ER. Press the ESC key to use the Command Processor utilities. Then use the print command to view the Results file. For example:  
PR NET>HVM1>SLIST.ER.

The Results file contains one line for each executable file name specified in the Schematic List file. The .DS suffix does not appear in the Results file even if it was used in the Schematic List file. Each line shows the file pathname and a compilation status. List file comments are not included.

If a previous Results file with the same name existed, it is overwritten.

Only the first 30 characters of a defined pathname for each line in the list file appear in the Results file.

If the Multiple Compile operation is halted by pressing the CONTROL-BREAK keys, the Results file contains the message: *Aborted by User Request*. An example of a results file is:

\$F2>WORK>SCHEM1	Success!
NET>HVM1>SCHEM2	Compile Operation Failed
SCHEM3	Read Operation Failed
SCHEM4	Success!

**Equipment List/DDB Files**—Although not a part of the multiple compile command, you can as a matter of convenience place commands in the Schematic List file to load a user-defined DDB variable list or to load or unload an Equipment list.

- To load a user-defined DDB variable file, insert LOAD nnnn
- To load an Equipment List DDB variable file, insert LOADEQ nnnn
- To unload an Equipment List file, insert UNLOADEQ nnnn

Load commands and the names of schematic files that are to be compiled can be intermixed in the Schematic List file. If there is a schematic file named LOAD, LOADEQ, or UNLOADEQ, it can still be compiled if the suffix .DS is appended.

An example of a Schematic List file with the load commands follows:

```
$F2>WORK>SCHEM1
NET>HVM1>SCHEM2.DS
{The following file should execute as:
NET>HVM1>SCHEM3.DS
SCHEM3.DS
LOAD NET>HVM1>MYDDBS1
SCHEM4
UNLOADEQ REACTOR1
LOADEQ REACTOR2
```

An example of the Results file is:

\$F2>WORK>SCHEM1	Success!
NET>HVM1>SCHEM2	Compile Operation Failed
SCHEM3	Success!
LOAD NET>HVM1>MYDDBS1	Read Operation Failed
SCHEM4	Success!
UNLOADEQ REACTOR1	Success!
LOADEQ REACTOR2	Success!

### 3.3.1.11.3 Replace Subpicture in Bulk

#### Picture Editor MULTREP Command

The syntax for the MULTREP command will be:

**Command: MULTREP**

**MULTREP (MREP MR) <Filename>**

With R600, a new Picture Editor Multiple Replace command is provided called “MULTREP.” This command will be modeled after the Picture Editor “Multiple Compile” command in that a text file similar to the Multiple Compile text file will be read in allowing replacement of multiple subpictures in multiple schematic files.

An error file will be produced indicating errors if the number and types of the parameters of the subpictures don't match or if the subpicture or file names are invalid. The original source file will be replaced with the modified source file only if no errors occur on the Replace Subpicture commands. If no errors occur on the replace, allow compile (if indicated in the text file).

No parameter form will be provided for the user to modify in the automatic mode. A manual mode indicator, in the input text file, will bring up the parameter form for the subpicture and wait for the user to enter any changes to the subpicture parameters. The subpicture replace mode will remain in manual until the input file indicates to change the mode to automatic.

The new Picture Editor MULTREP command allows the user to replace multiple subpictures in multiple files without being present at the Universal Station. The filename must have an EL extension for edited list. The list file is a text file that contains a list of the commands to be executed.

**Multiple Replace Commands**—The following commands will be recognized by the Multiple Replace Command Interpreter:

Command	Action
REPL subpicA subpicB	<p>This command will replace either all occurrences of subpicA with subpicB (if no occurrences of subpicA are selected) or just selected occurrences of subpicA with subpicB. The format for the names of subpicA and subpicB are identical to the format required for the REPLACE SUBPICTURE command.</p> <p>Note that if the new subpicture (subpicB) has parameters, the REPLACE SUBPICTURE command will present a form containing the parameters of the new subpicture. This is where the following two commands come in. Auto Mode On will not present this parameter form.</p> <p>Auto Mode Off will present this parameter form. For more details, see the Auto Mode On and Auto Mode Off commands.</p>

Command	Action
AUTO MODE ON	<p>This command sets the Auto Mode for the multiple replace command to ON. When Auto Mode is On, the operator will NOT be prompted to enter new parameters for the replace subpicture action.</p> <p>If the parameters of the new subpicture do not match the parameters of the old subpicture, an error will be reported in the results file and the current replace command will be aborted. The default mode for the Multiple Replace command is Auto Mode On.</p>
AUTO MODE OFF	<p>This command sets the Auto Mode for the multiple replace command to OFF. When Auto Mode is Off, the operator will be prompted to enter any new parameters for the replace subpicture action.</p> <p>Execution of the list file will be suspended until the operator responds by entering the new subpicture parameters and completing the form. Once the form(s) has been successfully filled out, the next command in the list file will be executed.</p> <p>All normal engineer prompts will be made when the Auto Mode is set to off. This includes the overwrite prompt done on the Write and Compile commands.</p>
LOAD <filename>	<p>This command loads a user-defined DDB variables file. The syntax for the pathname rules is the same as for the schematic source files (with the exception of file extension DF).</p>
READ <filename>	<p>This command will read the specified file into the Picture Editor. The syntax for the pathname rules is the same as for the schematic source files.</p> <p>If a full pathname is specified, the default path will change to the specified pathname.</p>
COMPILE <filename>	<p>The Compile command compiles the currently loaded schematic file. As in the Compile command, the source and object files are overwritten. The syntax for the pathname rules is the same as for the schematic source files.</p>

Command	Action
WRITE <filename>	<p>The Write command signifies the end of the subpicture replace commands for the currently loaded schematic file and causes the schematic source file to be saved back to the current default or specified pathname.</p> <p>In order not to lose the results of the subpicture replace commands, this command or the Compile command should precede the reading in of the next schematic file. The syntax for the pathname rules are the same as for the schematic source files.</p>
LOADEQ <filename>	<p>This command loads an equipment list variables file. The syntax for the pathname rules are the same as for the schematic source files (with the exception of file extension EQ).</p>
UNLOADEQ <filename>	<p>This command unloads an equipment list variables file. The syntax for the pathname rules is the same as for the schematic source files (with the exception of file extension EQ).</p>
SET COLLI <options>	<p>This command sets the priority for which collection rate to use when adding a subpicture to a main picture and there are variable collisions between the subpicture and main picture.</p> <p>This is a new command being added to the Picture Editor in R600. The syntax is as follows:</p> <p>SET COLLI &lt; FAST / MAIN / SLOW / SUB &gt;</p>

**Replacing subpictures that have parameters**— When you are replacing a subpicture that does have parameters with one that does not have parameters, use the following work around:

- Turn Auto Mode Off,
- replace the subpicture,
- and then turn Auto Mode On again

Without these steps, the Replace Command will fail in Automatic Mode.



### Other Characteristics of the List File:

- If a full pathname is not specified for the list file, then the current path will be used. If a full pathname is specified for the list file, then the default path is changed to the pathname specified.

The default path is also changed for each successful READ of a source file that specifies a full pathname. Other list file commands can use either whole or partial pathnames, but the default path will not change as a result of running the command.

- Any specification of a full pathname will change the default path to the new specification. The default path is originally initialized to the pathname specified for the list file on the multiple replace command line.
- The execution of the command can be aborted by the “CONTROL-BREAK” key operation (i.e., the execution of the current command will complete, but remaining processing will be aborted).
- Note that all other Picture Editor operations are halted by hitting the “CANCEL” key. However, the Multiple Replace command is a batch operation, and batch operations are consistently halted by the “CONTROL-BREAK” key.
- When an error occurs during execution of a file, the error is placed in the results file. Any subsequent commands prior to the end of the list file; the next Read, Load, Loadq, Unloadq, or Auto Mode command, whichever comes first, will not be executed.
- If the storage media becomes full during execution of the Multiple Replace command, all schematics on the same media that encounter a media-full situation will show an error status in the results file.

If the storage media becomes full while attempting to update the results file, the results file will be closed, the remaining schematic names in the list file will be ignored, and an appropriate error message will appear on the Picture Editor screen.

- Lines beginning with a "{" will be treated as comment lines. These lines will not be executed. Note that it is not necessary to end the comment line with a "}".
- A schematic source file can be defined with a full pathname of up to 64 characters.
- A schematic source file can be defined with a partial pathname that will be used in conjunction with the Default Pathname.
- Specification of the schematic file extension "DS" is optional.

**Error Handling**—Each of the edited list file commands may cause an error action to occur. The table below lists the possible error actions for each of the corresponding list file commands:

Command	Action
REPL subpicA subpicB	<p>If the REPL command fails, a Replace Error will be reported in the error results file. This command may fail due to the file or subpicture not being found or due to a mismatch in subpicture parameters, if the Auto Mode is set to ON. If the replace error is due to subpicA not being found, subsequent commands will be executed.</p> <p>Otherwise, any subsequent commands prior to the end of the list file, the next Read, Load, Loadeq, Unloadeq, or Auto Mode command, whichever comes first, will not be executed.</p>
AUTO MODE ON	The Auto Mode On command will not fail. No specific error will be reported for this command.
AUTO MODE OFF	The Auto Mode Off command will not fail. No specific error will be reported for this command.
LOAD <filename>	This command loads a user-defined DDB variables file. If the filename cannot be found, a Load Error will be reported in the error file.
READ <filename>	If the Read command fails, a Read Filename error will be reported in the error results file. Any subsequent commands prior to the end of the list file, the next Read, Load, Loadeq, Unloadeq, or Auto Mode command, whichever comes first, will not be executed.
COMPILE <filename>	If the Compile command fails, a Compile Error will be reported in the error results file. If a previous READ, REPL, or LOAD command fails, then this command will not be executed and a Command Skipped indicator will be put in the error results file.
WRITE <filename>	If the Write command fails a Write Error will be reported in the error results file. If a previous READ, REPL, or LOAD command fails, then this command will not be executed and a Command Skipped indicator will be put in the error results file.
LOADEQ <filename>	If the LOADEQ command fails, a Loadeq Failed error will be reported in the error results file. Any subsequent commands prior to the end of the list file, the next Read, Load, Loadeq, Unloadeq, or Auto Mode command, whichever comes first, will not be executed.
UNLOADEQ <filename>	If the UNLOADEQ command fails, a Unloadeq Failed error will be reported in the error results file. Any subsequent commands prior to the end of the list file, the next Read, Load, Loadeq, Unloadeq, or Auto Mode command, whichever comes first, will not be executed.
Invalid Command (This can be a misspelled or any other unrecognized command.)	An Invalid Command error will be reported in the error results file. Any subsequent commands prior to the end of the list file, the next Read command, or the next Load command, Loadeq, Unloadeq, or Auto Mode command, whichever comes first, will not be executed.
SET COLLI <options>	Providing that the correct syntax is used to call this command, no error will be reported in the error results file. If the incorrect syntax is used to call this command, an "Invalid Command" error will be reported in the error results file.

### 3.3.1.12 Selecting Objects

Existing objects in the picture often must be selected for some action by a subsequent command, for example to move or copy them, or to change their size or behavior.

There are several ways to Select objects—

- with the Select command as described on the next page
- whenever the Picture Editor is waiting for command input, you can—

place the cursor at a point on the screen and then press the SELECT key. Repeat for a second point. The Picture Editor draws a box that includes the two points. Any objects enclosed or intersected by the box are Selected. Selected objects turn white and blink.

if the Universal Station has a touch screen, you can enter the two points by touching the desired screen locations.

The Select command allows you to select a specific type of object when spacing is close. The noncommand method of selecting objects is convenient and fast when object spacing allows.

#### **Command: SELECT (SEL)**

**Use**—The Select command is used to select existing objects in the picture for some special action by a subsequent command.

Objects are selected by using the cursor to define two points on the screen. The Picture Editor then draws a box that includes the two points. Objects within, or intersected by, the box are selected objects. Note that it is possible to specify a single location that intersects an object, and that location is considered to be a 1-pixel box. If the pixel intersects any part of an object, the object is selected.

**Procedure**—Type SELECT on the command line, then press the ENTER key. The Picture Editor prompts Enter Select Coordinates.

Place the cursor at the first position and press the SELECT key. If the touch-screen option is present, just touch the screen at the desired location. The selection is marked by a cross (see Figure 3-11).

Move the cursor to the second point and press the SELECT key again (or use the touch-screen). Selection of the second point causes the box to be drawn. Note that leaving the cursor in the same position for both selections creates the minimum-sized box (one dot).

**DEL**—If the DEL key is pressed at this point, the last location selected is deleted. If no further locations exist, the command is canceled.

Press the ENTER key.

Figure 3-12 shows how the resulting box intersects the pump, furnace, and value (XXX); therefore, they are selected but the text (PSI) is not selected.

Selected objects are displayed at full intensity, white, and blinking. Also, objects that have conditional behavior identical\* to the selected object are displayed as white and blinking at half intensity, however, they are not selected.

Selected objects remain selected until they are either deselected by the DESELECT command or acted on by any of the other commands that are used to manipulate selected objects (i.e., Move, Copy, Scale).

**Pixel Coordinates**—You can specify pixel locations for the top left and bottom right corners of the selection box as X-Y coordinates. For example:

**SEL 312 240 312 480**

specifies boundary coordinates: X-312, Y-240 and X-312, Y-480.

One way to determine the exact pixel coordinates is to invoke an Add Value command. As the cursor is moved, you can see the X-Y coordinates displayed at the top of the screen in the C- (cursor position) field. If you need to change the cursor to pixel resolution, hold down CTL and press 1. Note the coordinates for the desired selection box, then cancel the Add Value command. Enter the SEL command with those coordinates as explained above.

**Qualifiers**—a qualifier can be added to the Select command. In such cases, the command syntax is

**Command:   SELECT qqqq**

If the selection box intersects more than one object, a qualifier can be added to limit selection.

where the qualifier terms (qqqq) and their abbreviated forms are

BAR		SUBPICTURE	(SUBPIC)	(SUB)	(S)
LINE	(LIN)	(L)	TARGET	(TARG)	(TAR)
SOLID	(SOL)		VALUE	(VAL)	(V)
TEXT	(TEX)	(T)	VARIANT	(VAR)	

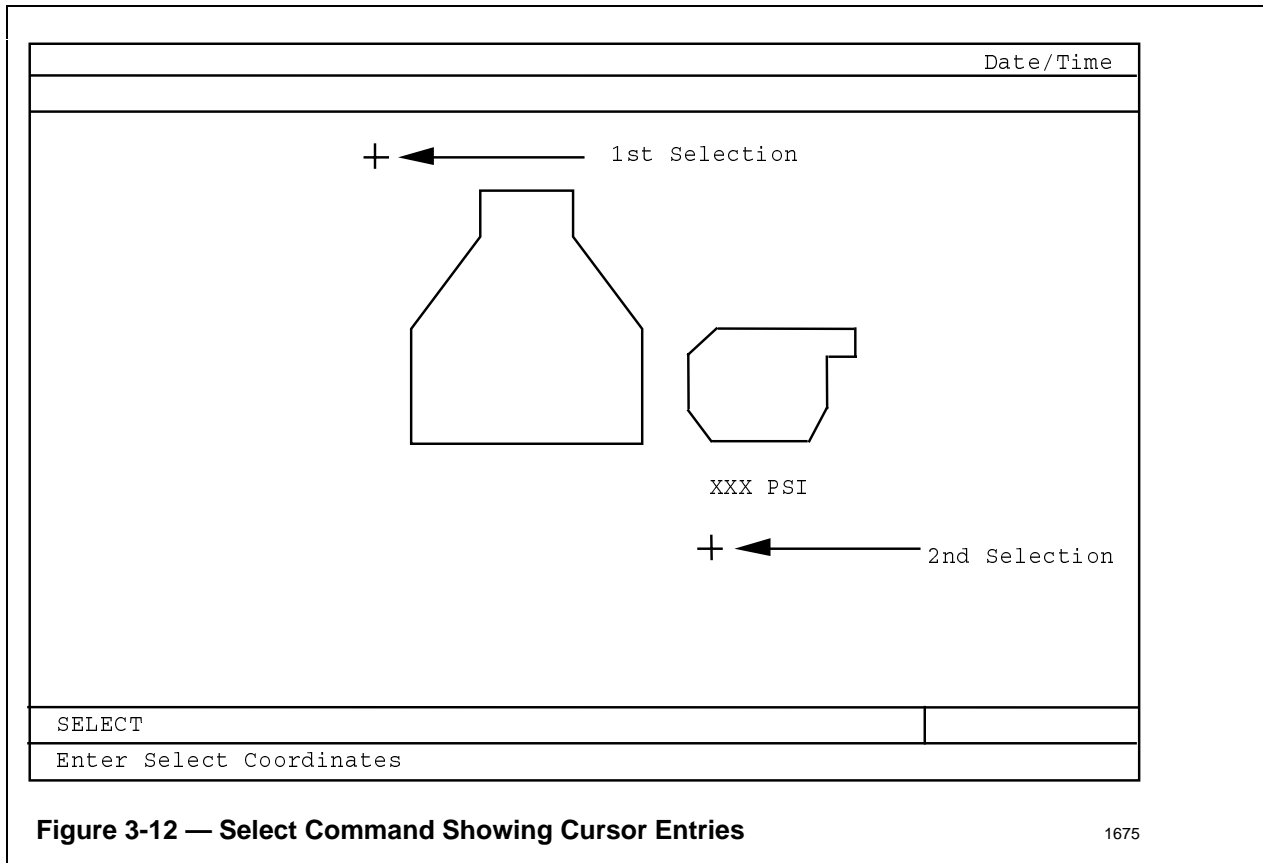
---

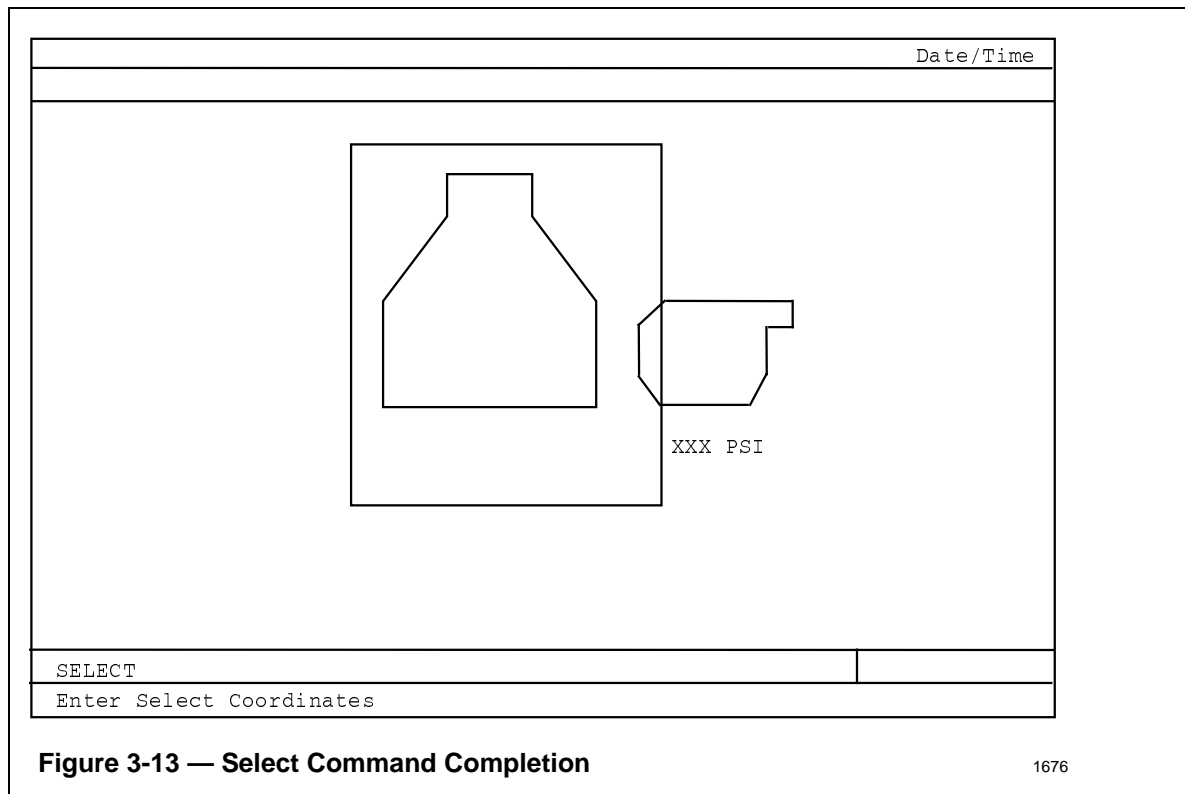
\* Refer to the Select Condition command for a definition of identical behavior.

In Figure 3-11, if the command Select Value had been used, only the value (XXX) would have been selected even though the box encloses and intersects other objects.

The Select Behavior, Select Condition, Select Inherit, and Select Priority commands are special forms of the Select command and are explained elsewhere in this manual.

The Select Subpicture command can be used to specify the name of a subpicture (e.g., SELECT SUB pump where pump is a subpicture name). In such cases, all occurrences of the specified subpicture are selected. Also, any variants that refer to the subpicture are selected.





### 3.3.1.13 Deselecting Objects

**Command:** DESELECT (DES)

**Use**—This command deselects selected objects in the picture.

**Procedure**—Type DESELECT on the command line, then press the ENTER key. The Picture Editor prompts Enter Select Coordinates. Objects to be deselected are chosen in the same way as for the Select Command.

**Qualifiers**—The DESELECT command can also be used with a qualifier (qqqq), in which case the command syntax is

**Command:** DESELECT qqqq

**Abbreviated form:** DES qqqq

where the qualifier terms (qqqq) and their abbreviated forms are

BAR		SUBPICTURE	(SUBPIC)	(SUB)	(S)
LINE	(LIN) (L)	TARGET	(TARG)	(TAR)	
SOLID	(SOL)	VALUE	(VAL)	(V)	
TEXT	(TEX) (T)	VARIANT	(VAR)		

Some special forms of the Deselect command are described elsewhere in this manual; they are the Deselect Priority, Deselect Condition, and Deselect Behavior.

#### 3.3.1.14 Command: **SELECT STRING "string" X Y X Y**

**Use**—The Select String command is used at build time to find a specified string of text. The optional X-Y coordinates specify a bounding box where the search is to take place.

**Procedure**—Type `SELECT "mnopq"` or `SELECT "mnopq" X Y X Y` on the command line (where `mnopq` is the string and `x,y` are coordinates that define a selection box), then press the ENTER key. If you did not specify the X-Y coordinates, the Picture Editor prompts you to select them now. The Picture Editor searches for the specified string and if the text string is found in an object intersecting or within the selection box, the object is selected (turns white and blinks).

**Examples**— `SELECT ".PV"`  
`SELECT "HG1001" 0 0 300 200`

**Undefined characters**—you can substitute symbols in a string to search for unspecified characters. If you precede the special character with `@`, the character is used literally. The following special characters can be used in a search—

Symbol	Search for
?	single Character
*	zero or more characters
[	begin set/range
-	range of characters within set
]	end set/range
~	reverse context of set/range
!	zero or more of the previous character
@	next character as a literal character

**Examples**—

`SELECT "A??"`. The Picture Editor will search for any three character string beginning with A.

`SELECT "abc*"`. The Picture Editor will search for any character string that begins with abc and has zero or more characters following.

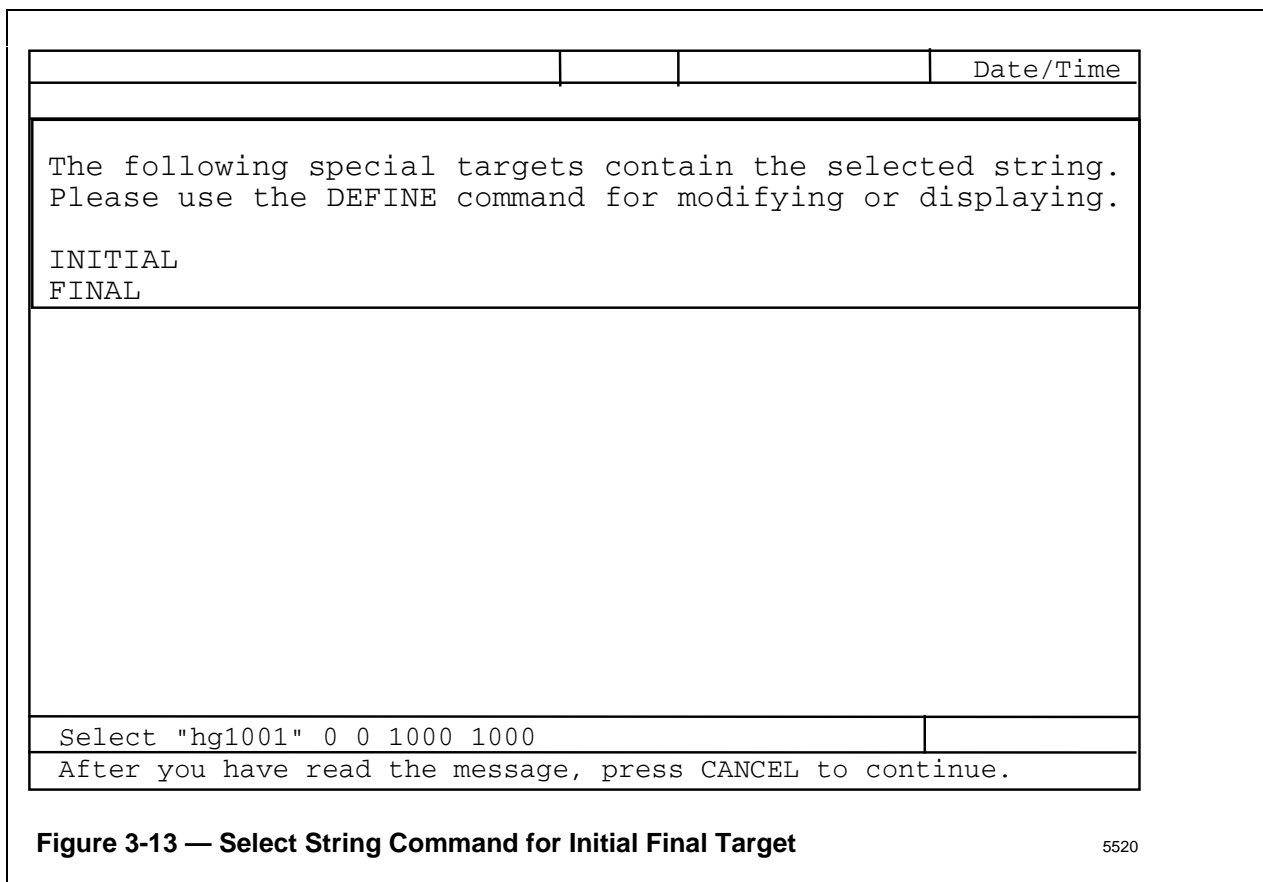
#### **Parameter Explanation**—

**String**—The text to be searched for is the string of characters between the quotation marks. The maximum size string length is 60 characters. The search is not case sensitive, therefore, if the specified string is "ABCD" the Picture Editor will search for "ABCD", "abcd", "AbCd" etc.

**Coordinates** —Coordinates are X-Y pairs that define the lower-left and upper-right corners of a selection box. If the coordinates are specified, the Picture Editor will search each object within or intersected by the selection box for the string pattern. If coordinates are not specified, the Picture Editor prompts the user to select coordinates on the picture.

Note that because comment ports, initial/final targets, help, and associated displays, are not displayable, they cannot be selected if the specified string is found in one of those items. Instead, if the string is found in one of these items, a message appears on screen to tell you in which objects the string was found. Press the CANCEL key to clear the message.

Figure 3-13 illustrates the message after searching for the string HG1001 and finding it in an Initial and Final target.





### 3.3.1.15 Command: DELETE (DEL) (D)

**Use**—This command deletes all selected objects in the picture (see the Select command for an explanation of selected objects).

**Procedure**—Type DELETE on the command line, then press the ENTER key.

**Qualifiers**—When used with a qualifier (qqqq) this command deletes all selected objects of the specified type. The object to be deleted must have been previously selected. In this case the command syntax is

**Command:** DELETE qqqq

Abbreviated forms: DEL qqqq  
D qqqq

The qualifiers and their abbreviated forms are \*

BAR		
INHERIT	(INH)	(IN) (I)
LINE	(LIN)	(L)
SOLID	(SOL)	
SUBPICTURE	(SUBPIC)	(SUB) (S)
TARGET	(TARG)	(TAR)
TEXT	(TEX)	(T)
VARIANT	(VAR)	
VALUE	(VAL)	(V)

Example: DELETE LINE, DELETE SOLID, etc.

---

\*Some of the Delete commands have special characteristics and are described elsewhere in this manual. They are Delete Behavior, Delete Condition, and Delete Priority.

### 3.3.1.16 Command: MOVE (MOV)

**Use**—The Move command is used to move objects within the picture. It only affects objects previously selected, and all selected objects are moved at the same time.

Two coordinates are entered to describe the distance and direction (i.e., a vector) of the move. The distance between cursor points represents the distance of the move. The direction of the move is represented by a vector whose tail is the first coordinate and whose head is the second coordinate (see Figure 3-14).

**Procedure**—Type MOVE on the command line, then press the ENTER key. The Picture Editor prompts ENTER MOVE COORDINATES.

Enter each of the coordinate points by positioning the cursor and pressing the SELECT key. If the Universal Station has a touchscreen, you can enter the two points by touching the desired screen locations. Crosses appear on the screen to mark coordinate points (see Figure 3-13).

**Pixel Coordinates**—You can specify pixel locations for the two locations as X-Y coordinates. For example:

**MOVE 312 240 312 480**

specifies vector coordinates: X-312, Y-240 and X-312, Y-480.

One way to determine the exact pixel coordinates is to invoke an Add Value command. As the cursor is moved, you can see the X-Y coordinates displayed at the top of the screen in the C- (cursor position) field. If you need to change the cursor to pixel resolution, hold down CTL and press 1. Note the coordinates for the desired locations and then cancel the Add Value command. Enter the MOVE command with those coordinates as explained above.

**DEL**—If it is necessary to redefine a coordinate, pressing the DEL key cancels the last entry. If no entry locations remain, the command is canceled.

When both coordinates have been specified, press the ENTER key. Selected objects move in the direction and distance specified (See Figure 3-15).

#### NOTE

Text, Values, Variants, and Subpictures have an attribute known as text size. When these objects are moved, the text size of the object determines the increment of movement. If the text size of the object is large, the object can be moved in 8 x 16 pixel increments (8 wide by 16 high). If the text size of the object is set to small, the object can be moved in 8 x 8 pixel increments. An attempt to move text objects to a position between incremental locations causes the object to be placed at the nearest incremental boundary to the desired location. Section 3.3.5 Text and Text Commands explains how text size is determined. Graphic objects (lines, solids, or bar graphs) can be moved anywhere within the edit region.

Qualifiers—The Move command can be used with a qualifier to move specified selected objects. Objects to be moved must have been previously selected. In this case, the command syntax is

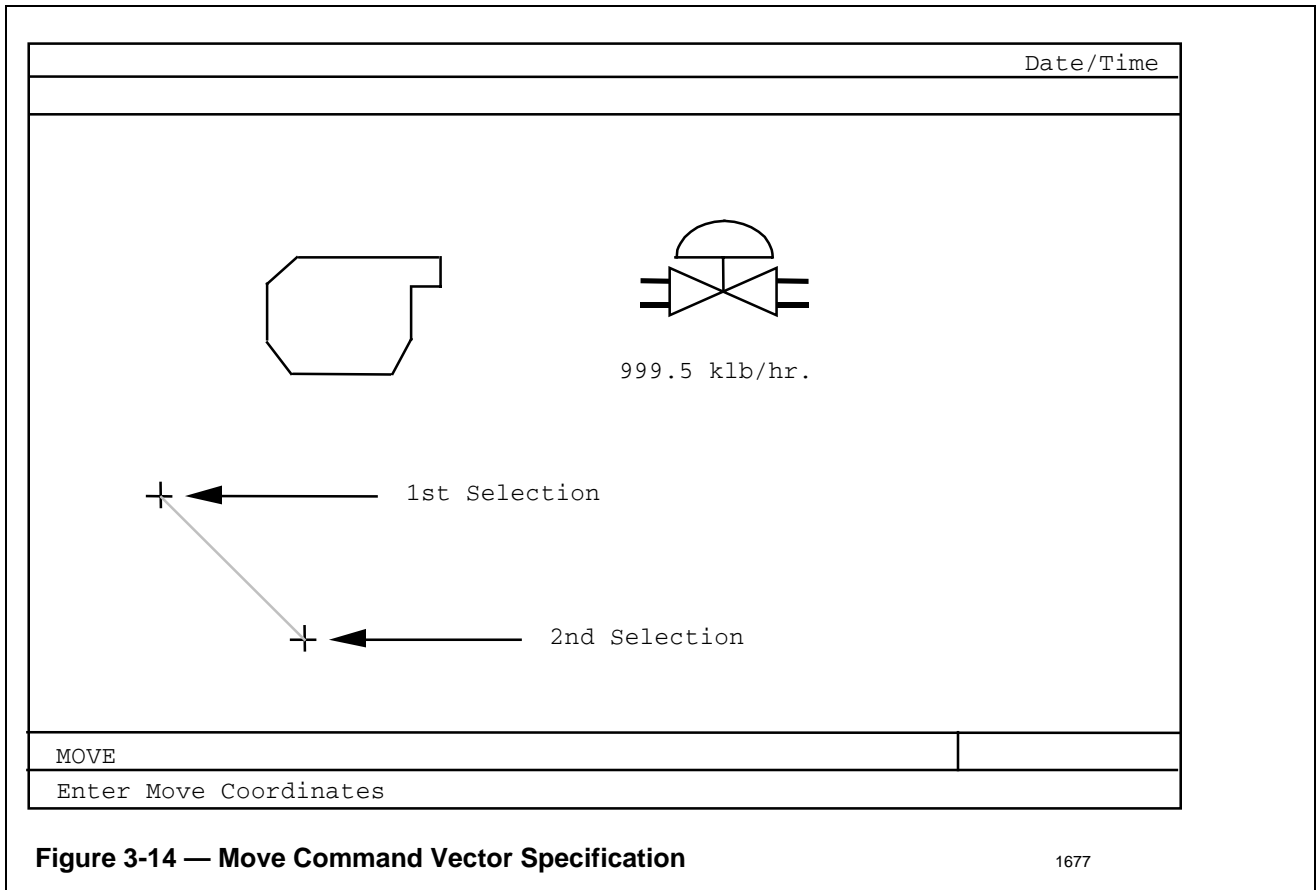
**Command:**    **MOVE**qqqq

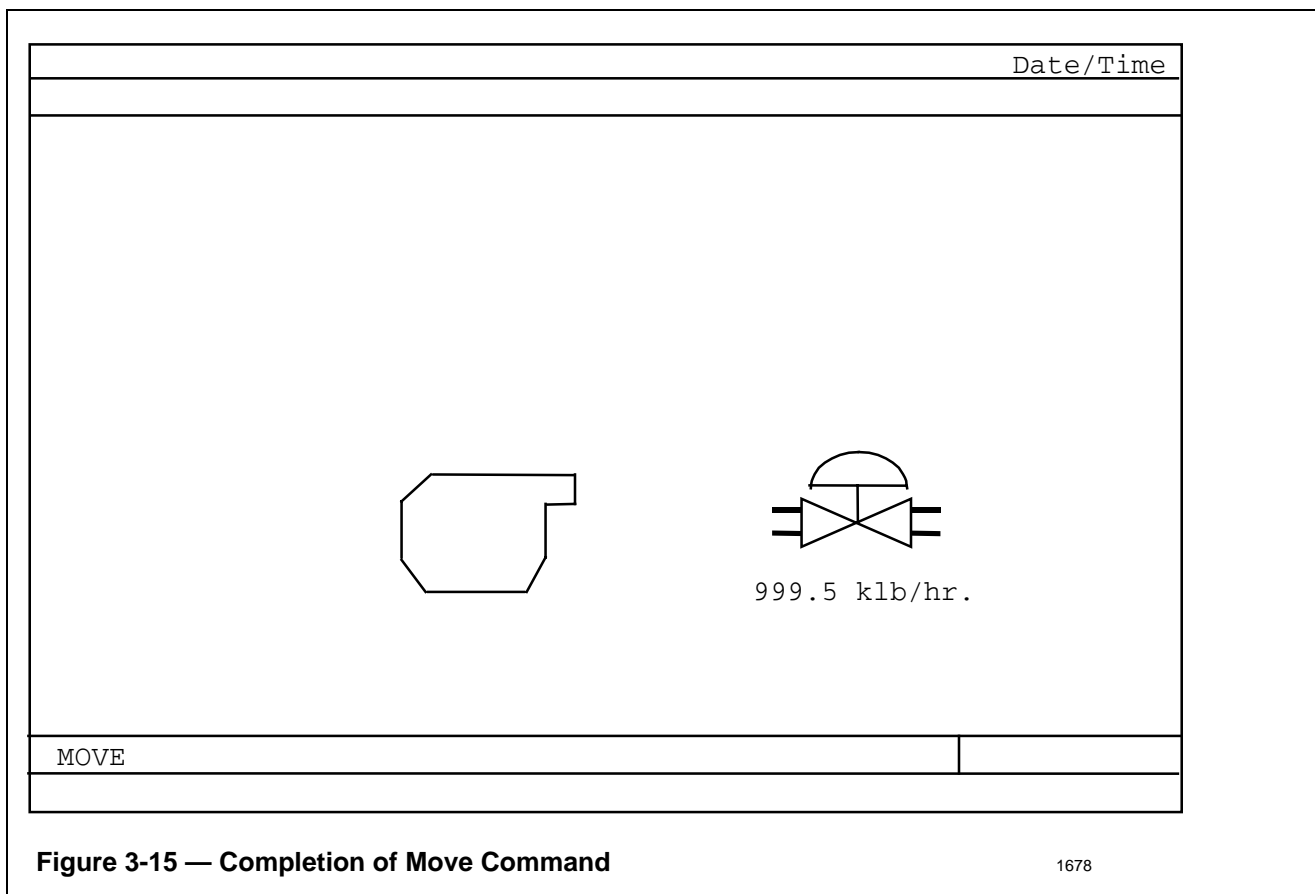
Abbreviated form:    **MOV** qqqq

where the qualifier terms qqqq and their abbreviated forms are

BAR		SUBPICTURE	(SUBPIC)	(SUB)	(S)
LINE	(LIN) (L)	TARGET	(TARG)	(TAR)	
SOLID	(SOL)	VALUE	(VAL)	(V)	
TEXT	(TEX) (T)	VARIANT	(VAR)		

Operation is the same as for the Move command.





### 3.3.1.17 Command: COPY (COP) (C)

**Use**—The Copy command is used to copy objects within the picture. It only affects objects previously selected, and all selected objects are copied at the same time (see new R530 copy command under COPYCLIP).

Two coordinates are entered to describe the distance and direction (i.e., a vector) where the second image is to appear. The distance between cursor points represents the distance between the first and second image. The direction between the first and second image is represented by a vector whose tail is the first coordinate and whose head is the second coordinate (see Figure 3-16).

**Procedure**—Type COPY on the command line, then press the ENTER key. The Picture Editor prompts ENTER COPY COORDINATES.

Enter each of the coordinate points by positioning the cursor and pressing the SELECT key. If the touch-screen option is present, just touch the screen at the desired location. Crosses appear on the screen to mark coordinate points (see Figure 3-16).

**Pixel Coordinates**—You can specify pixel locations for the two locations as X-Y coordinates. For example:

**COPY 312 240 312 480**

specifies vector coordinates: X-312, Y-240 and X-312, Y-480.

One way to determine the exact pixel coordinates is to invoke an Add Value command. As the cursor is moved, you can see the X-Y coordinates displayed at the top of the screen in the C- (cursor position) field. If you need to change the cursor to pixel resolution, hold down CTL and press 1. Note the coordinates for the desired locations and then cancel the Add Value command. Enter the COPY command with those coordinates as explained above.

**DEL**—If the DEL key is pressed while specifying locations, the last entry is deleted and the location can be re-entered. If no locations remain, the command is canceled.

When both coordinates have been specified, press the ENTER key. Selected objects are copied at a location in the direction and distance specified (see Figure 3-17).

**CANCEL**—If the CANCEL key is pressed at this point, the command is canceled.

#### NOTE

Graphic objects (lines, solids, or bar graphs) are pixel oriented and can be copied to any location within the edit region. Text, Values, Variants, and Subpictures have an attribute known as text size. Objects with an attribute of large text size can be copied in 8 x 16 pixel increments (i.e., to locations that are 8 pixels apart horizontally or 16 pixels apart vertically). Objects with an attribute of small text size can be copied in 8 x 8 pixel increments. An attempt to copy these objects to a position between incremental locations causes the object to be placed at the nearest incremental boundary to the desired location. Subsection 3.3.5 Text and Text Commands explains how text size is determined.

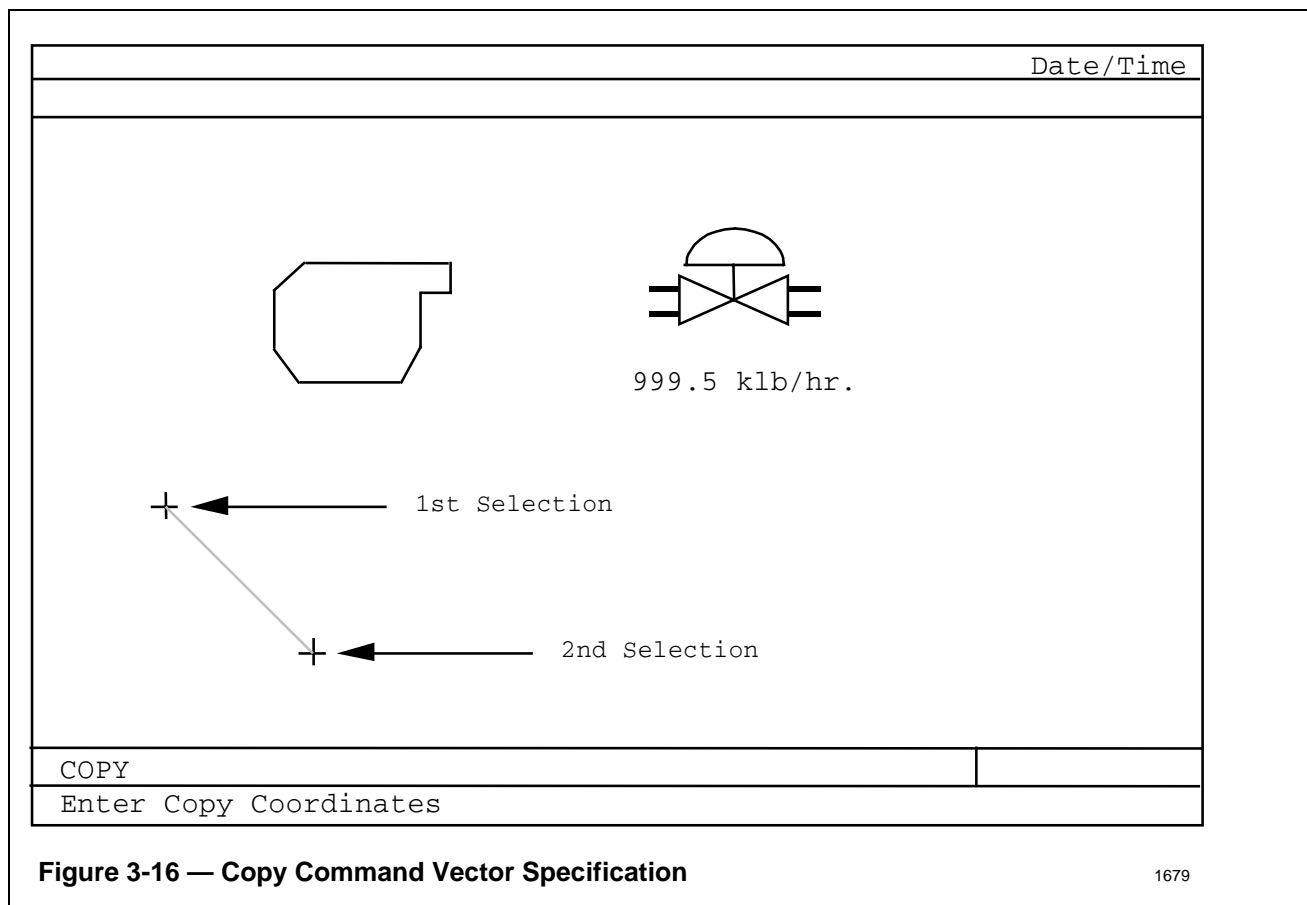
Qualifiers—The Copy command can be used with a qualifier to copy specified objects. The objects must have been previously selected. In this case, the command syntax is

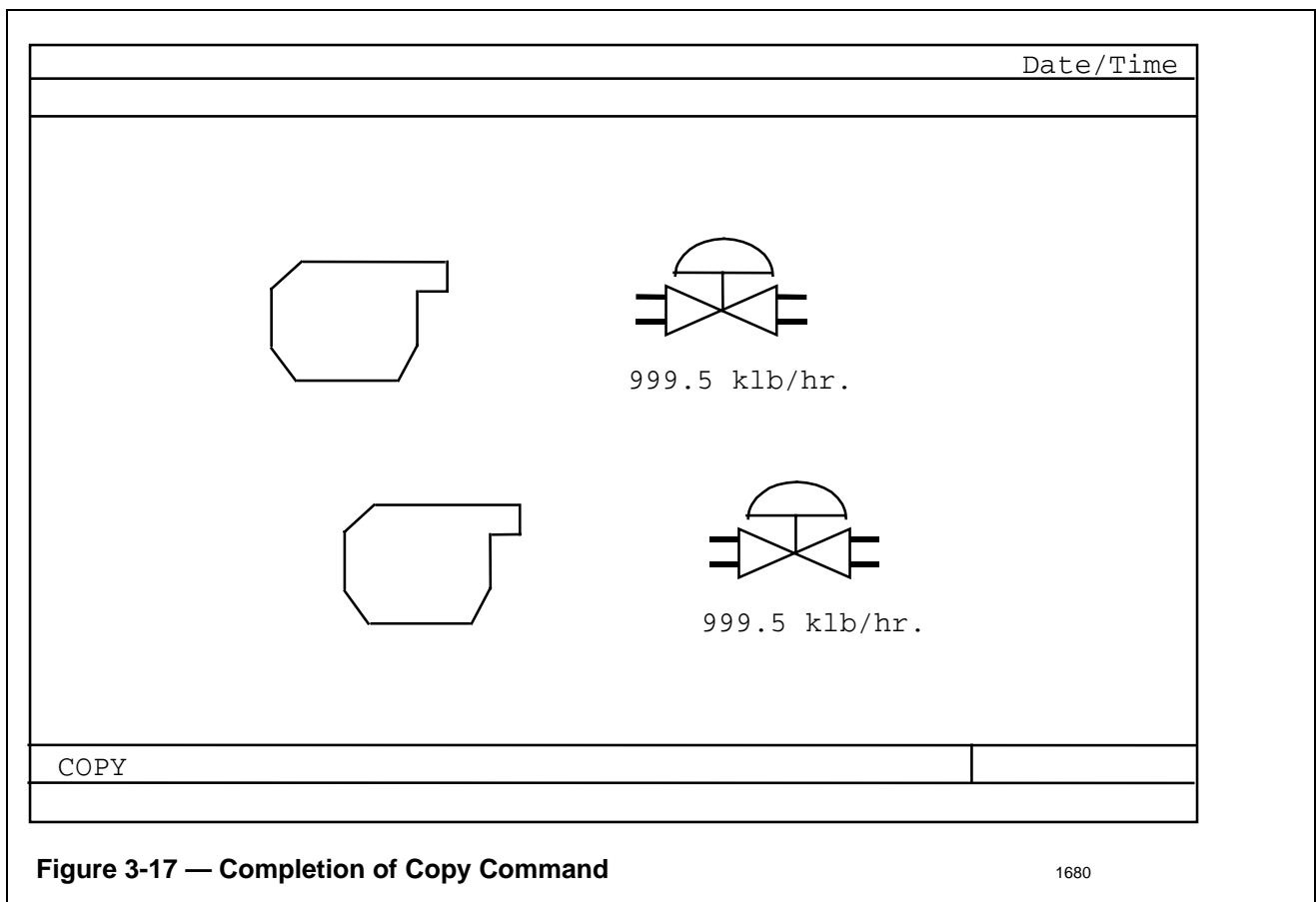
**Command:**   **COPY** qqqq

where the qualifier terms (qqqq) and their abbreviated forms are

BAR		SUBPICTURE	(SUBPIC)	(SUB)	(S)
LINE	(LIN) (L)	TARGET	(TARG)	(TAR)	
SOLID	(SOL)	VALUE	(VAL)	(V)	
TEXT	(TEX) (T)	VARIANT	(VAR)		

Operation is the same as for the Copy command.





## R530 functions - COPYCLIP, CUT, and PASTE

### 3.3.1.18 Command: COPYCLIP (CLIP CC)

**Use**—The new R530 COPYCLIP command facilitates copying selected objects from one picture or subpicture to another by copying the objects into an intermediate memory-resident file.

The command copies only objects previously selected and all selected objects are copied at the same time. No coordinates are necessary since the command copies only the objects to an intermediary memory-resident file and not immediately to another picture. The relative positions of the various selected objects are maintained. After the COPYCLIP command has successfully completed, the previously selected objects are no longer selected.

**Procedure**—Select objects to be copied, type COPYCLIP on the command line, then press the ENTER key.

The coordinates/boundaries of the selected objects can now be used to establish an origin for the copied objects for later use by the PASTE command (another new R530 function).

**Qualifiers**—The COPYCLIP command can be used with a qualifier to copy specified objects to the memory-resident file. The objects must have been previously selected. In this case the command syntax is:

**Command: COPYCLIP qqqq**

where the qualifier terms (qqqq) and their abbreviated forms are

BAR		SUBPICTURE	(SUBPIC)	(SUB)	(S)
LINE	(LIN)	(L)	TARGET	(TARG)	(TAR)
SOLID	(SOL)		VALUE	(VAL)	(V)
TEXT	(TEX)	(T)	VARIANT	(VAR)	

The above qualifiers are consistent with the current Picture Editor COPY command.

### 3.3.1.19 Command: CUT (CT)

**Use**—The new R530 CUT command facilitates the moving of objects from one picture or subpicture to another by deleting the objects out of the current picture and moving them into an intermediate memory-resident file. This command only affects objects previously selected and all selected objects are cut (deleted from the original schematic) at the same time. No coordinates are necessary since the CUT command only moves the objects to an intermediary memory-resident file and not immediately to another picture. The relative positions of the various selected objects are maintained. After the CUT command has been successfully completed, the previously selected objects are no longer selected.

**Procedure**—Select objects to be cut, type CUT on the command line, then press the ENTER key.

The coordinates/boundaries of the selected objects are used to establish an origin for the cut objects for later use by PASTE command



## R530 functions - COPYCLIP, CUT, and PASTE, continued

**Qualifiers**—The CUT command can be used with a qualifier to cut specified objects to the memory-resident file. The objects must have been previously selected. In this case, the command syntax is:

**Command:**    **CUT qqqq**

where the qualifier terms (qqqq) and their abbreviated forms are

BAR		SUBPICTURE	(SUBPIC)	(SUB)	(S)
LINE	(LIN) (L)	TARGET	(TARG)	(TAR)	
SOLID	(SOL)	VALUE	(VAL)	(V)	
TEXT	(TEX) (T)	VARIANT	(VAR)		

The above qualifiers are consistent with the current Picture Editor COPY command.

### ATTENTION

The memory-resident file will exist as long as the Picture Editor is active, including use of the ESCAPE feature. Also, exiting the Picture Editor will cause the memory-resident file to be deleted.

Additional copy or cut commands will cause previously saved data to be overwritten.

### 3.3.1.20 Command: PASTE (PA)

**Use**—The new R530 PASTE command is used to paste objects from an intermediate memory-resident file to the current picture. Execution of the PASTE command without specifying the coordinates will cause an “Enter Paste Coordinates” prompt to select the desired destination coordinates. The origin for the copied objects is used by the PASTE command to position the objects. The relative positions of the copied objects are maintained by the PASTE command. The objects will remain selected so that the objects can easily be moved to a new location in the picture without having to reselect the objects.

**Procedure**—Type PASTE on the command line, select one coordinate and then press ENTER.

Entering more than one set of coordinates for the PASTE command will result in an error. The PASTE command can be repeated with a new set of coordinates after the pasted objects are moved to their desired location and deselected.

## R530 functions - COPYCLIP, CUT, and PASTE, continued

The following error messages have been identified as new error messages that will be added to the Picture Editor for the COPYCLIP, CUT and PASTE R530 functions:

Error	This error will be reported...
Memory-Resident File Size Exceeded	by the new COPYCLIP or CUT commands if the size of the objects exceed the memory-resident file size.
Select Object Before Copying	if the COPYCLIP command is executed without first properly selecting objects to be copied.
Select Object Before Cutting	if the CUT command is executed without first properly selecting objects to be cut.
Deselect Objects Before Pasting	if the PASTE command is executed with objects selected.
Nothing to Paste	if the PASTE command is executed and the memory-resident file is empty.
Corrupted Clipboard File	if the PASTE command is executed because the format or contents of the clipboard file is incorrect.
No Mem/ Var Not Found - obj YYY: XXXXXXXX	if the COPYCLIP or CUT command is executed and the Picture Editor is not able to get enough memory for the variable list, or cannot correctly parse the variable to place it in the clipboard file symbol table.
Paste not allowed - Type mismatch: variable XXXXXXXX	if the PASTE command is executed and the variable type in the symbol table is different from the variable type in the clipboard file.
Paste not allowed - Text size mismatch in subpicture XXXXXXXX	if the PASTE command is executed and a large textsize subpicture is trying to be pasted into a small textsize schematic.
Paste not allowed - New subpicture does not match local subpicture XXXXXXXX	if the PASTE command is executed with a subpicture whose number of parameters or data types does not match the number of parameters or data types of a subpicture of the same name in the schematic where it is being pasted.
No More Coordinates Allowed	if more than one set of coordinates are specified for the PASTE command.
Paste not allowed - Undefined DDB variable: XXXXXXXX	if the PASTE command is executed with a variable (like a custom DDB) that is not defined in the target schematic.
Paste not allowed - Invalid subpicture variable in subpicture XXXXXXXX	if the PASTE command is executed and the clipboard file subpicture parameter will not parse correctly in the target schematic subpicture.
Paste not allowed - Variant bounds increased by local subpicture XXXXXXXX	if the PASTE command is executed and the target schematic subpicture causes the clipboard file variant bounding box to increase in size.

### 3.3.1.21 Command: **MODIFY (MOD) (M)**

**Use**—The modify command is used to change an existing object or, if applicable, to change entries on the screen form that was used when the object was built. The Modify command applies to bar charts, conditional behavior, lines, solids, subpictures, targets, text, values, and variants. If no qualifier is used, all selected objects are subject to modification. If only one object is selected, no qualifier is needed.

**Procedure**—Type MODIFY on the command line, then press the ENTER key.

The modification command as it applies to each type of object is discussed in the sections that follow under the heading: Modify [object].

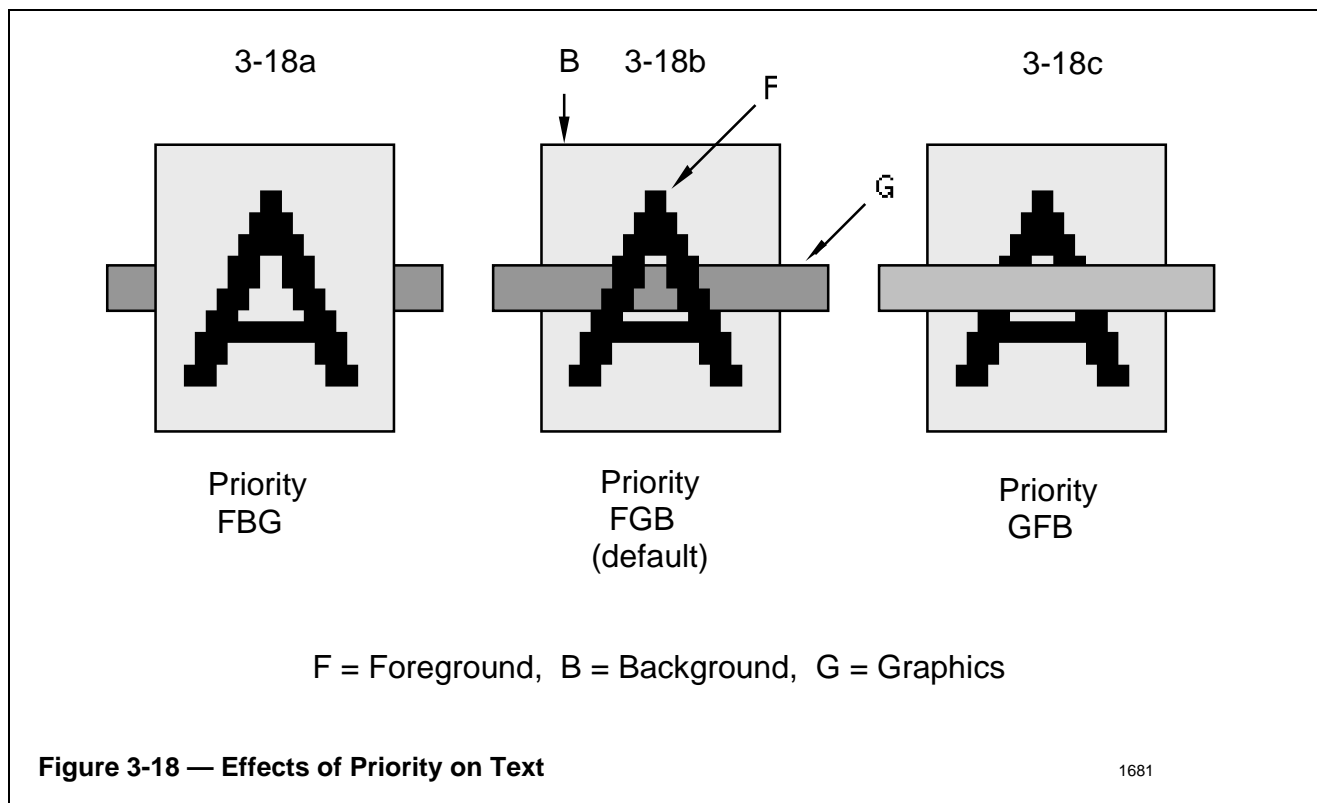
### 3.3.1.22 Priority Commands

Priority commands deal with text, values, or text within variants (i.e., nongraphic items). When there is an overlap between those items and graphic items, the priority order determines what is seen. Figure 3-18 illustrates the effect on text, the text background cell, and graphics for the three priority choices. Text or values are the foreground part (F), the cell around the text or value is the background (B), and drawn objects are the graphics part (G).

In Figure 3-18a, text has the highest priority followed by the text background cell, followed by a graphics object (a pipe in this example). You can compare the priority order to three layers of paint. The lowest priority item (the pipe) is the first layer. It is covered by the second layer of paint, which corresponds to the middle priority item (the cell). The pipe and cell are both covered wherever the highest priority item (text) is painted.

In Figure 3-18b, the pipe (graphics object) has higher priority than the background cell and now covers the cell. Text still has highest priority and paints over both the cell and graphics object.

In the figure below the graphics object has the highest priority and therefore covers both the text and cell.



Priority is affected by the following commands. When the picture editor is started, the default priority is FGB.

**Command:** SET PRIORITY qqq

Abbreviated forms: SET PRI qqq  
S PR qqq

where qqq is FGB, FBG, or GFB

**Use**—Set Priority is used to specify the current priority order. All objects subsequently added to the picture are added in the current priority.

**Procedure**—Type SET PRIORITY qqq on the command line, then press the ENTER key.

Executing this command, causes the new current priority setting to appear on the top line of the screen.

**CANCEL**—If the CANCEL key is pressed at this point, the current priority reverts back to the previous settings.

**Command:**    **ADD PRIORITY qqq**

Abbreviated Forms:    **ADD PRI qqq**  
                              **A PR qqq**

where the qualifiers qqq are FGB, FBG, OR GFB

**Use**—The Add Priority command is used to change the priority of objects within the picture. The objects must have been previously selected.

**Procedure**—Type ADD PRIORITY qqq on the command line, then press the ENTER key. All objects that are selected assume the specified priority (qqq).

**Command:**    **SELECT PRIORITY**

Abbreviated forms:    **SEL PRI**  
                              **SEL PR**

**Use**—The Select Priority command is used to select all objects with the same specified priority in the picture. This command is useful to identify those objects with common priorities.

**Procedure**—Type SELECT PRIORITY on the command line, then press the ENTER key. The Picture Editor prompts Enter Select Coordinates.

Move the cursor to the first position and press the SELECT key. If the touch-screen option is present, just touch the screen at the desired location. This selection is marked by a cross (see Figure 3-19). Move the cursor to the second point and press the SELECT key again (or use the touch-screen). Selection of the second point causes the box to be drawn.

**DEL**—If it is necessary to change a coordinate entry, press the DEL key to delete the last entry and then re-enter the coordinate.

As an example, refer to Figure 3-20 and assume that only the text Furnace F-101 and the text HPUMP have the priority FGB; all of the other objects have a different priority. The Select Priority command is invoked and the select box is drawn as shown in Figure 3-20. All objects in the picture with the same priority as the selected object are also automatically selected.

In this example, only the text Furnace F-101 was enclosed/intersected by the box; however, both this text and the text HPUMP are selected because they have the same priority.

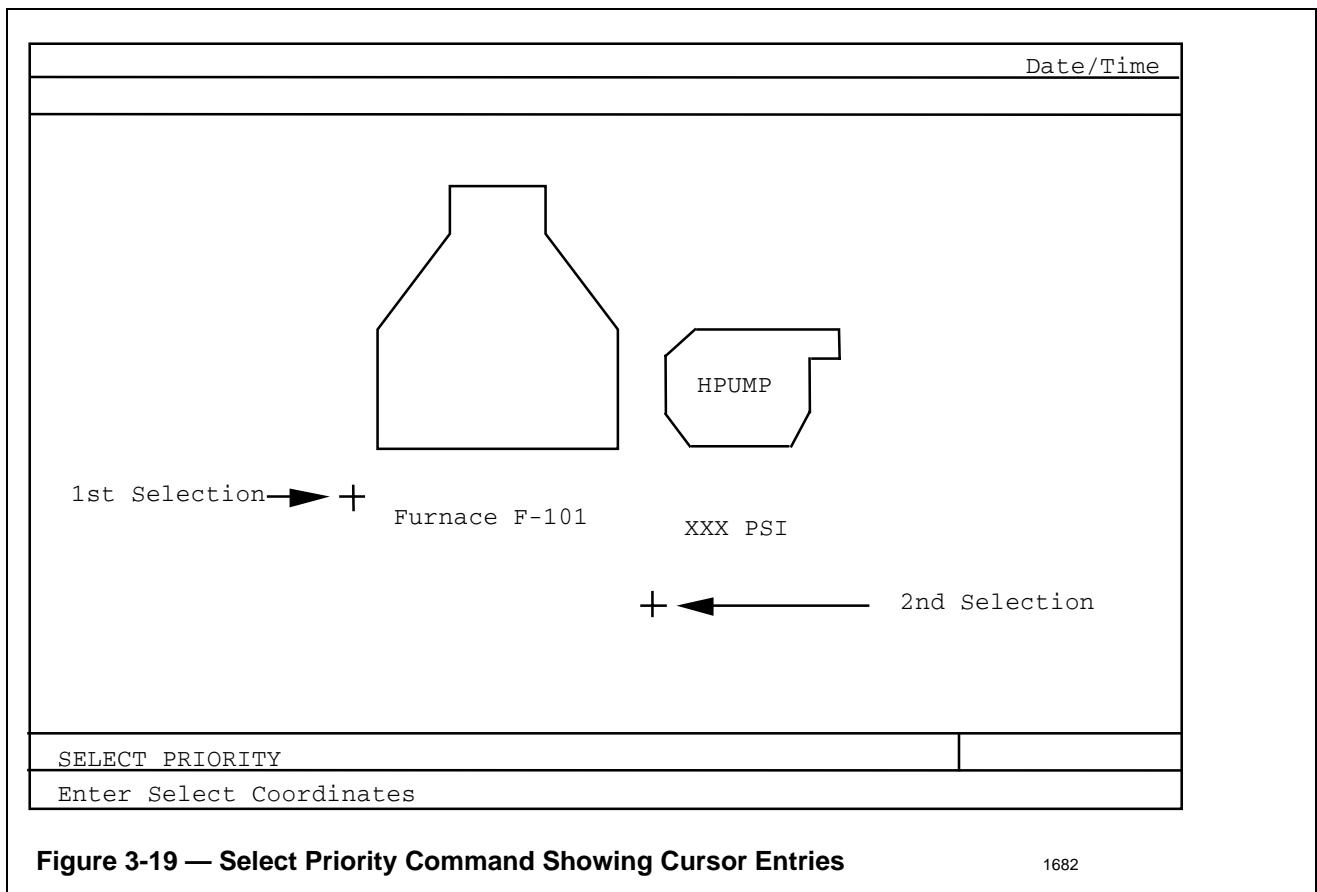
Selected objects are displayed as white and blinking, at full intensity. Selected objects remain selected until either deselected by a DESELECT command or until acted on by any of the commands that are used to manipulate selected objects (i.e., Move, Copy, Scale).

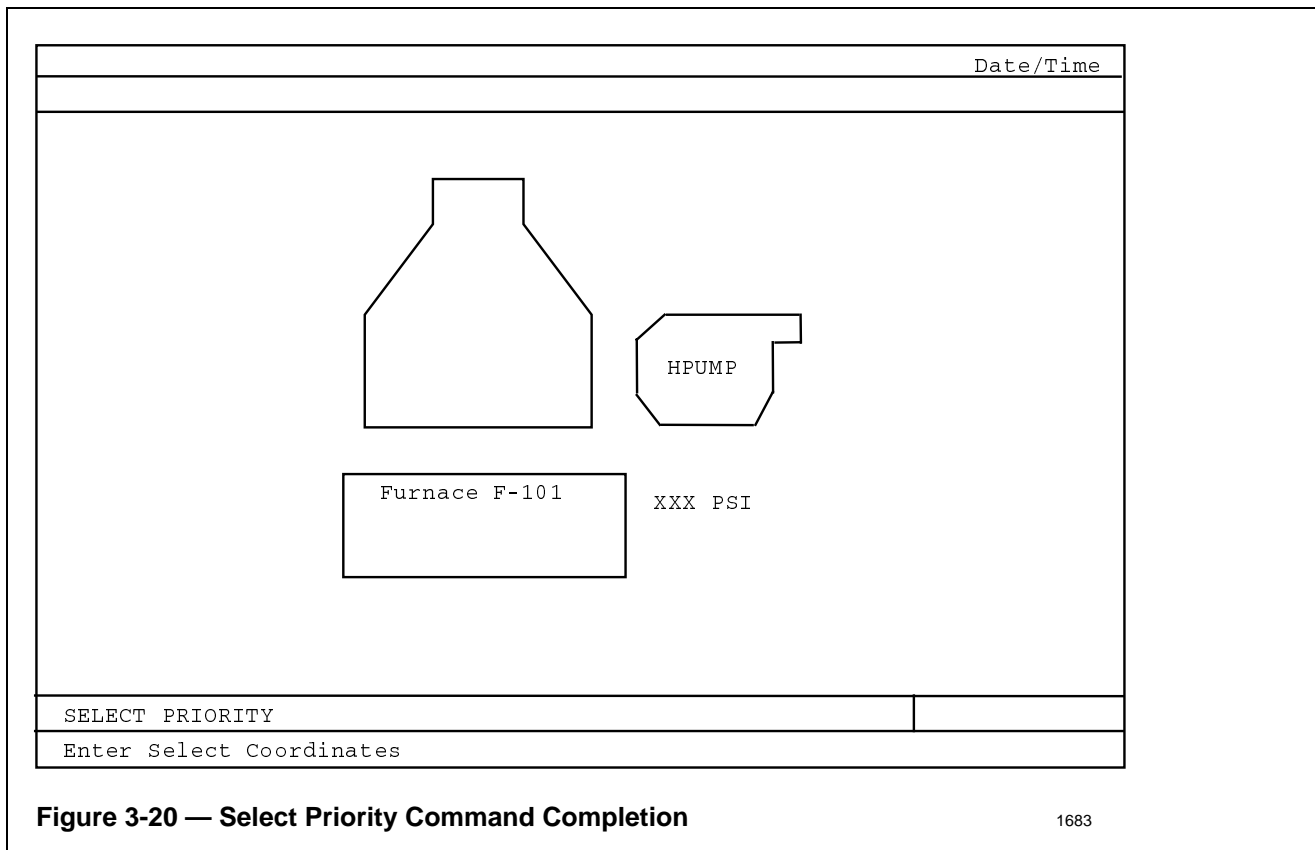
**Command:** DESELECT PRIORITY

Abbreviated forms: DES PRI  
DS PR

**Use**—This command is used to deselect all selected objects with the same priority.

**Procedure**—This command is the opposite of the Select Priority Command and the procedure is the same.





**Command:** DELETE PRIORITY

**Abbreviated forms:** DEL PRI  
D PR

**Use**—This command is used to delete the current priority setting for all selected objects and replace it with the new current priority setting. Objects must have been previously selected.

**Procedure**—Type DELETE PRIORITY on the command line, then press the ENTER key. All selected objects assume the current priority setting.

For example, assume that the value/text item XXX PSI shown in Figure 3-20 is selected, and has the priority GFB. If the current priority is FBG, executing the Delete Priority command causes this item to change to the current priority, FBG.

### 3.3.1.23 Command: SET COLLECTION, (S COL) (S C)

**Use**—The Set Collection command is used to change the data-collection properties for variables that have been used in the construction of a display. Data collection is the process of fetching current information for variables, collectors, and DDB variables.

Refer also to Appendix I which explains how collection groups are used to reduce system loading.

**Procedure**—Type SET COLLECTION on the command line; press the ENTER key.

The Picture Editor responds by presenting a screen form (see Figure 3-21) showing all of the currently defined variables. If the form continues beyond one page, use the Page Forward/Page Backward keys to view the entire form.

Date/Time																										
<table border="1"><thead><tr><th>SYMBOL</th><th>COLLECTION RATE</th><th>GROUP ID</th></tr></thead><tbody><tr><td>A100.NAME</td><td><input type="text" value="0"/></td><td><input type="text" value="0"/></td></tr><tr><td>A100.PV</td><td><input type="text" value="1"/></td><td><input type="text" value="FST"/></td></tr><tr><td>A100.SP</td><td><input type="text" value="1"/></td><td><input type="text" value="FST"/></td></tr><tr><td>\$CZ_VID1</td><td><input type="text" value="1"/></td><td><input type="text" value="CZ"/></td></tr><tr><td>TIC101.NAME</td><td><input type="text" value="0"/></td><td><input type="text" value="0"/></td></tr><tr><td>TC101.PV</td><td><input type="text" value="2"/></td><td><input type="text" value="1"/></td></tr><tr><td>TC101.SP</td><td><input type="text" value="2"/></td><td><input type="text" value="1"/></td></tr></tbody></table>			SYMBOL	COLLECTION RATE	GROUP ID	A100.NAME	<input type="text" value="0"/>	<input type="text" value="0"/>	A100.PV	<input type="text" value="1"/>	<input type="text" value="FST"/>	A100.SP	<input type="text" value="1"/>	<input type="text" value="FST"/>	\$CZ_VID1	<input type="text" value="1"/>	<input type="text" value="CZ"/>	TIC101.NAME	<input type="text" value="0"/>	<input type="text" value="0"/>	TC101.PV	<input type="text" value="2"/>	<input type="text" value="1"/>	TC101.SP	<input type="text" value="2"/>	<input type="text" value="1"/>
SYMBOL	COLLECTION RATE	GROUP ID																								
A100.NAME	<input type="text" value="0"/>	<input type="text" value="0"/>																								
A100.PV	<input type="text" value="1"/>	<input type="text" value="FST"/>																								
A100.SP	<input type="text" value="1"/>	<input type="text" value="FST"/>																								
\$CZ_VID1	<input type="text" value="1"/>	<input type="text" value="CZ"/>																								
TIC101.NAME	<input type="text" value="0"/>	<input type="text" value="0"/>																								
TC101.PV	<input type="text" value="2"/>	<input type="text" value="1"/>																								
TC101.SP	<input type="text" value="2"/>	<input type="text" value="1"/>																								
SET COLLECTION																										
Enter Collection Properties																										

11038

**Figure 3-21 — Screen Form for Set Collection Command**



Collection properties can be changed by typing new values into the associated entry ports. Collection rates represent 4-second multiples of the update rate. A collection rate of 0 means the variable will be collected only the first time the display is called up. A collection rate = 1 means update every 4 seconds, 2 = every 8 seconds, 3 = every 12 seconds, etc. (see the table below). The maximum collection **rate** number is 255.

If the Collection Rate entry is	Then the variable is updated
0	Only when first displayed
1	Every 4 seconds (default)
2	Every 8 seconds
3	Every 12 seconds
.	.
.	.
15	Once per minute
.	.
.	.
255	Every 17 minutes

Variables can be placed in different collection groups by group ID and those groups can be updated at different rates depending on the collection rate configuration. Collection **Group ID** numbers can range from 1-245, but the maximum number of Group IDs you can use per schematic is 127 (including FST, CZ, and 0). The maximum number of variables in a Collection Group is 512. The error message: COLLECTION SET OVERFLOW appears if there are more than 512 variables.

**FAST UPDATES**—Variables can be allowed to update at a 1/2 second rate by entering **FST** in the Group ID port. When the display is called up in the Operating Personality, fast updates (twice a second) are enabled by pressing the FAST button or executing the Que\_Key (Fast) target action. Pressing the FAST button again switches back to normal updates (4 seconds, unless otherwise specified in the Set Collection display). A light on the key indicates when FAST mode is enabled. CZ is automatically shown in the Group ID port for variables appearing in the standard Change Zones. To enable fast update of change zone variables, enter **FST** in the variable's Group ID port.

The following table summarizes how update considerations affect the collection rate.

<b>Update Requirements</b>	<b>For Use With ...</b>	<b>Collection Rate</b>
At Invocation only	Variables that do not change such as EUDESC, or those that change slowly or infrequently	0
Scheduled Update	Variables that require a regularly scheduled update according to how fast they change.	1 - 255 (4 seconds - 17 minutes)
Fast Update	Critical variables that at certain times require a faster update than the scheduled rate.  Fast update is activated using the FAST button; otherwise, the specified collection rate applies.	Twice a Second
On Demand	Variables that need to be updated under certain circumstances. This type of update operates independently of the configured Collection Rate.  These variables are updated by one of the following actors:  <b>UPDATE actor</b> —updates DDB variables, all variables, or variables assigned to a specific collection group in a specific screen region.  <b>DMD_UPD actor</b> —updates an entire region of the screen.	N/A

**CANCEL/DEL**—If either the cancel or DEL key is pressed at this point, the command is canceled and collection properties revert to their previous specifications.

Press ENTER to complete the command.

**CANCEL**—If the cancel key is pressed at this point, collection properties revert to their previous specifications.

#### **3.3.1.24 Command: SET GRID mmm**

Abbreviated Forms: S GRI mmm  
S GR mmm  
S G mmm

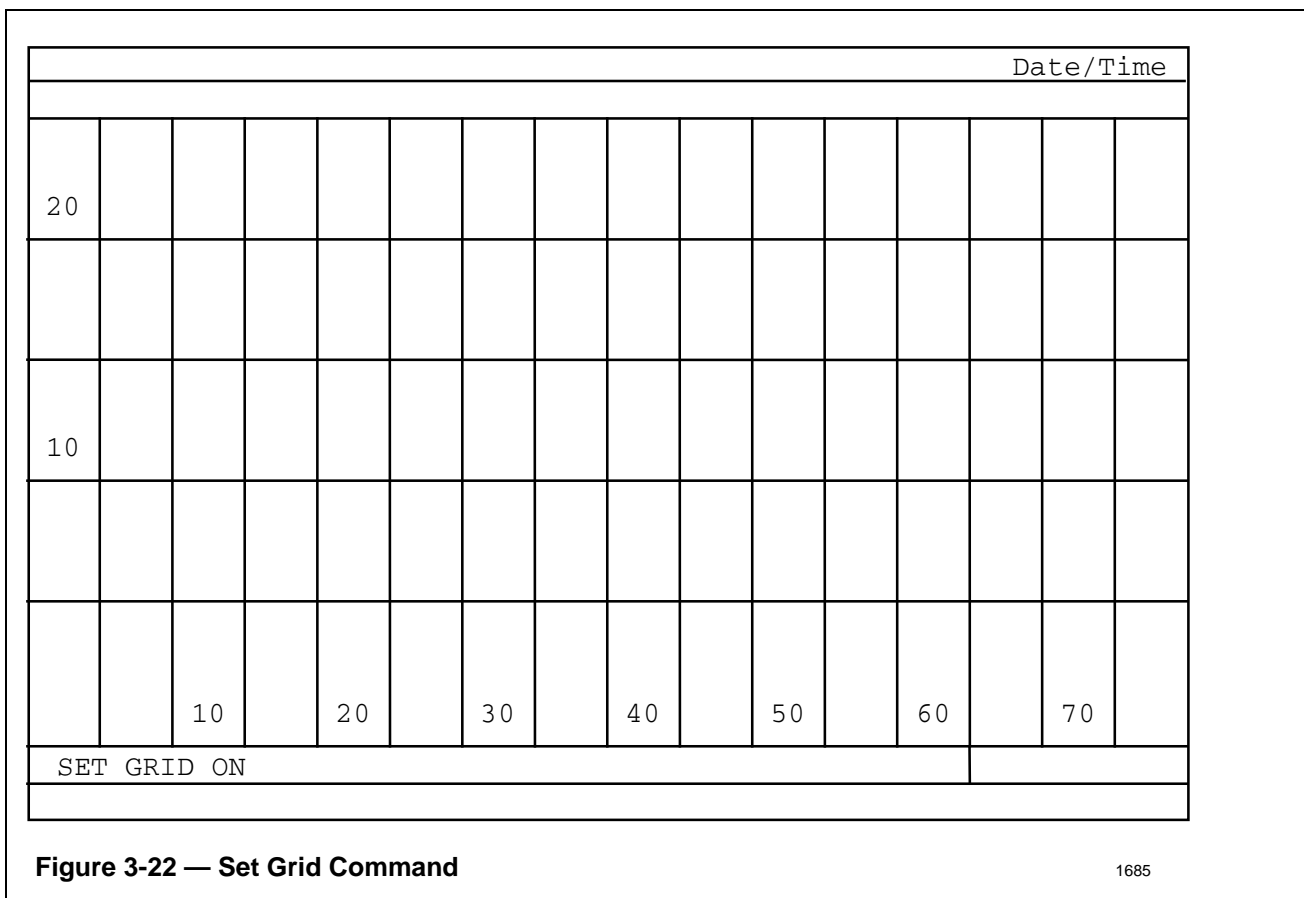
where the mode is mmm

**Use**—The Set Grid command is used to enable or disable a grid overlay of the drawing area. The mode (mmm) choices are ON or OFF. If ON is specified, the grid appears on the screen. If OFF is specified, the grid disappears. The picture is not otherwise affected.

**Procedure**—Type SET GRID mmm on the command line, then press the ENTER key (mmm = ON or OFF).

Example: SET GRID ON

This feature is primarily used to help align objects horizontally or vertically in the picture. The grid coordinates are numbered every 10 characters in both the horizontal and vertical directions. Figure 3-22 illustrates the grid.



### 3.3.1.25 Command: DEFINE

#### Calling Sequence— DEFINE nnnn

where **nnnn** specifies when the action is to be executed. The following table shows the choices.

nnnn	Synonyms	Action To Be Executed Upon
INITIAL	(INIT) (I)	Display invocation
FINAL	(FIN) (F)	Display cancellation
PAGE_FWD	(PAGE_F) (PF)	Pressing the PAGE FWD key
PAGE_BACK	(PAGE_B) (PB)	Pressing the PAGE BACK key
DISP_FWD	(DISP_F) (DF)	Pressing the DISP FWD key
DISP_BACK	(DISP_B) (DB)	Pressing the DISP BACK key
HELP	(H)	Pressing the HELP key
ASSOC	(A)	Pressing the ASSOC DISP key

**Use**—The Define command is used to designate an actor or string of actors that will execute when a specified action occurs. For example it can be used to—

- initialize (i.e., pre-load) DDB/system variables when calling up a display
- place points in a trend and start trending when the display is called up
- bring in an overlay with the display being called up
- configure specific keys (see table) for special use with certain displays

**Procedure**—Type **DEFINE nnnn** on the command line, then press the ENTER key. A screen form appears. The screen form is the same as used for configuring target or button actions. Enter the actors that are to execute when the nnnn action occurs. Actors are described in the *Actors Manual* (see References).

**Modification**—To change the command definition, re-issue the Define command.

- If you specify the same action (nnnn) as for a previously entered Define command, the existing actors for that command are displayed. You can then change the actors.
- The Define command is deleted by leaving the screen form blank.
- Specifying a different action (nnnn) than for a previously entered Define command is the same as invoking a new Define command.

Terminology—Throughout this description, an **initial action** (also called an initial target) means a display was configured with the Define Initial command. Its associated actors execute when the display is called up. A **final action** (also called a final target) means a display was configured with a Define Final command. Its associated actors execute when you leave that display.

### **WARNING**

Do not configure a display with an initial action that is configured to invoke itself. If so, when you call up the display, the Universal Station will lock up and must be reloaded.

Do not configure a display with a final action that is configured to invoke itself. If so, after you call up that display, you will not be able to call up any other display (i.e. the display will continually call itself when you try to exit). You can reload the Universal Station, or if the misconfigured display is network-resident, you can delete or rename it.

Do not use an initial action on an overlay (but you can use an initial action on a display; for example, to call in an overlay to that display).

#### **Example 1 Configuring an automatic overlay display**

When building a custom display, enter the DEFINE INITIAL command. When the screenform appears, enter the actor OVERLAY("DISP123"). Complete and compile the display. When this custom display is called up at run time (i.e., in the Operating Personality), the overlay named DISP123 is automatically brought in with it, thus eliminating the need for a separate target or button to call in the overlay.

#### **Example 2 Starting a trend display automatically**

After building a custom schematic that is to contain a trend display, the DEFINE INITIAL command is executed. Add Trace actors are entered into the screenform, for example—

```
TR_ADD(1,A100.PV,1); TR_ADD(1,A200.PV,1);, etc.
```

A TREND or TREND\_AX subpicture is added to the schematic, and the picture is compiled.

When this schematic is called up at run time, trending starts automatically. The need for a separate target that adds the traces and starts trending is eliminated.

#### **Example 3 Configuring a key to call up an associated display.**

A custom display is built and the command DEFINE ASSOC is executed. When the screenform appears, the actor GROUP(121,0) is entered and the display is then compiled.

At run time, when this display is in use, the operator can press the ASSOC DISP key to call up the Group display for Group 121.

Special help displays can be configured in a similar way, or the PAGE FWD/PAGE BACK keys could be configured to call up a series of displays.

## NOTE

### 1. Prompting for input within an Initial Action

When a display is configured with an initial action that requires input (for example, one of the Operator Input actors), and that display is called up (at run time), the screen is cleared except for the input prompt. After you type in the data, the called display appears. After two minutes, if no input is made, the input request times out and the requested display appears. If however, another display is called up, the originally requested display appears for an instant, after which the new display appears.

### 2. Calling another display from an Initial Action

If an initial action contains an actor that requests another display, the original display appears for an instant, then the new display is called in.

### 3. Some Actors should not be used for an Initial Action

Do not use the following actors in an initial action:

**UPDATE or DMD\_UPD  
COLLHIST**

An error message results if any of the above are used in an initial action and actor execution is terminated, but the display then appears.

### 4. Change Zone Actor

The Change Zone actor should not be used in an initial action.

### 5. Requesting input within a Final Action

If an actor within a Final Action requests input, the existing display is not cancelled until one of the following occurs—

- the entry is completed
- another display is called up
- the two minute time-out occurs

### 6. Calling another display from a Final Action

If an actor within a Final Action calls up another display, the display requested by the final action is always the one that is called. For example if display A has its final action configured to call display B, and if you try to leave display A by calling up display C, display B is called in instead.

### 7. Error Messages On Final Actions

If any operation performed by a Final Action causes an error, the error message is not displayed.

### 3.3.1.26 Command: **LOAD (LD) (L)**

**Use**—The Load command is used to

- define Display Database (DDB) variable names in a file
- declare DDB variable names to be part of the system Display Database.

**Procedure**—Type **LOAD nnnn** on the command line (where nnnn is the full or partial pathname of a previously created file that contains the defined DDB variables). Then, press the ENTER key.

You can just type LOAD if the pathname on the second line from the top of the screen is the desired pathname of the file nnnn. Refer also to the Multiple Compile command that shows how Load commands can be embedded in a Schematic List File.

#### **Example—LOAD \$F1>VOL1>MYDDBS**

Loads the file MYDDBS.DF from the floppy or cartridge named VOL1.

**Errors**—If the file containing the DDB variable declarations does not exist, the message **File Not Found** appears. If syntax errors exist in the file, the message **Syntax Error Detected** appears.

If the Load command executes with no errors, names declared in the file can then be referenced in the display. Note that if another DDB file, other than an Equipment List Object file (.QO file) is loaded while in the same editing session, the DDB variable names declared in the previous DDB file become unknown to the Picture Editor and an error results when you modify or compile the picture. If a .QO file was loaded before the Load command is executed, error checking is done for name or index conflict with the alias names in the .QO file. If errors are found, the same error display is presented as for the LOADEQ command.

#### **Creating the DDB File:**

The DDB file used with the Load command contains a declaration of the DDB variable names and must be created before executing the Load command. You can create the file with the Text File Editor. The procedure is described in *Actors Manual/Appendix A* (see User Defined DDB Files). A special Honeywell Developer's Display Database is described in the same Supplement.

Important points to note about user-defined DDB variables—

- User defined DDB variables are not initialized; therefore, reading from these variables before writing to them returns a "bad value" status.

For example—

```
Standard DDB: INT01      initial value = 0
User DDB:      MY_INT    initial value = @@@@ @@@@
```

- If user-defined DDB variables are used in a picture, the DDB file must be loaded when editing that picture.

### 3.3.1.27 Command: **LOADEQ nnnn**

Abbreviated form: **(LDQ ) nnnn**

nnnn is the pathname to an Equipment List Object File.

**Use**—The Load Equipment List command is used to load DDB locations (Alias Names) defined in an Equipment List Object File. The Alias Names are used the same way as if they were local Display Database (DDB) names in the schematic. While building a picture, up to eight different equipment list files can be loaded.

**Procedure**—Type **LOADEQ nnnn** on the command line (where nnnn is a pathname). Then, press the ENTER key.

**Examples**—**LOADEQ NET>ELB>REACTOR**  
**LDQ REACTOR**

Loads the alias name DDB definitions into the Picture Editor from the file: REACTOR.QO.

**Error Conditions**—When executing the LOADEQ command:

- If eight Equipment List Object Files (.QO files) are already loaded into the Picture Editor and you attempt to load an additional .QO file, an error message results and the file is not loaded.
- If you attempt to load a .QO file with the same name as one already loaded (even from a different pathname), an error message results and the file is not loaded.
- If the same name is used for an equipment list file and a user-defined DDB file, an error message results and the file is not loaded.
- If the same name is used as an alias name in an equipment list file and a system variable, the Picture Editor uses the standard variable name instead of the alias DDB variable name.
- If different Alias Names are assigned to the same location, an error message results and the file is not loaded.

Use the PAGE FWD/PAGE BKWD keys to move through the error displays if necessary. Press the CANCEL key to remove the error display.

If the Load Equipment List command executes with no errors, then you can use the alias name in the same manner as user-defined DDB names that were loaded with the Load command.

Refer also to the Multiple Compile command that shows how LOADEQ commands can be embedded in a Schematic List File.



### 3.3.1.28 Command: **UNLOADEQ zzzz** or **UNLOADEQ**

Abbreviated form: **(ULDQ ) zzzz**

where zzzz is an Equipment List Object File name (.QO file) that was previously loaded into the Picture Editor by the LOADEQ command.

**Use**—The Unload Equipment List command is used to delete the named Equipment List Object File (if it exists). If an Equipment List Object File name (zzzz) is not specified, all loaded .QO files are deleted from the Picture Editor.

**Procedure**—Type **UNLOADEQ zzzz** on the command line (where zzzz is a .QO filename). Then, press the ENTER key.

**Examples**—**UNLOADEQ REACTOR**  
**ULDQ REACTOR**

When a .QO file is unloaded, the Alias Names it contained cannot be used in the picture. If Alias Names were used in the picture before unloading the file, they must be removed or an error is generated at compile time.

**Cancel key**—if you press the Cancel key at this point, the command is aborted.

**Errors**—if the specified .QO file is not loaded into the Picture Editor, an error message results.

Refer also to the Multiple Compile command that shows how UNLOADEQ commands can be embedded in a Schematic List File.

### 3.3.1.29 Command: **LISTEQ (LSQ )**

**Use**—The List Equipment File names command is used to list all of the Equipment List Object Files (.QO file) that are currently loaded into the Picture Editor.

**Procedure**—Type **LISTEQ** on the command line. Then, press the ENTER key.

**Examples**—**LISTEQ**  
**LSQ**

**Cancel key**—if you press the Cancel key at this point the command is aborted.

### 3.3.1.30 Command: SET PALETTE nn

Abbreviated forms: SET PALETT nn  
S PALET nn  
S PAL nn

where nn is the palette number.

**Use**—The Set Palette command is used to activate one of 16 color palettes (if nn is 1–16) or the default palette (if nn = 0). A palette refers to a specific color combination of a background and the sixteen displayable colors available with that background. The standard default palette is the traditional palette consisting of a black background with eight colors in two intensities. The palette used at build time is the palette that appears at operate time.

**Subpictures**—Subpictures (even if built using another palette) appear in the main schematic's palette.

**Overlays**—If palette 0 (the default palette) is used to build an overlay, it doesn't affect the main schematic when it is called. If the overlay is built with any of the palettes numbered 1 through 16, when that overlay is called, it forces the main schematic to the same palette as specified in the overlay.

This command is effective with Release 320 and later software. Universal Stations must contain an EPDG electronic board with revision H or later firmware and an EPDG-P I/O electronic board to effectively use the Palette command. Note that there is also a Palette actor.

**Procedure**—Type SET PALETTE nn on the command line, then press the ENTER key (nn = 0–16).

Example: SET PALETTE 13

**CANCEL**—If the cancel key is pressed after the command is completed, the palette reverts to the previous setting.

At operating time, a schematic appears in the same palette colors specified at build time. Selecting a standard display, such as the Console Status display, causes the palette to revert to the default palette. The standard default palette consists of a black background with eight colors in two intensities.

## Color Keys

The engineering keyboard color keys select the following colors depending on the palette number selected—

---

### Full Intensity

	<b>- Key -</b>							
Palette	WHT	BLK	CYAN	BLUE	MAGN	RED	YEL	GRN
1	White	Black	Cyan	Blue	Magenta	Red	Yellow	Green
2	White	Lt Gry	Cyan	Blue	Magenta	Red	Yellow	Green
3	White	Med Gry	Cyan	Blue	Magenta	Red	Yellow	Green
4	White	Wrm Gry	Cyan	Blue	Magenta	Red	Yellow	Green
5	White	Black	Cyan	Blue	Magenta	Red	Yellow	Green
6	White	Lt Gry	Cyan	Blue	Magenta	Red	Yellow	Green
7	White	Med Gry	Cyan	Blue	Magenta	Red	Yellow	Green
8	White	Wrm Gry	Cyan	Blue	Magenta	Red	Yellow	Green
9	White	Lt Gry	Cyan	Blue	Magenta	Red	Yellow	Green
10	White	L/I Blue	Cyan	Blue	Magenta	Red	Yellow	Green
11	White	Med Gry	Cyan	Blue	Magenta	Red	Yellow	Green
12	White	Wrm Gry	Cyan	Blue	Magenta	Red	Yellow	Green
13	White	Lt Gry	Cyan	Blue	Magenta	Red	Yellow	Green
14	White	Lt Gry	Cyan	Blue	Magenta	Red	Yellow	Green
15	White	Med Gry	Cyan	Blue	Magenta	Red	Yellow	Green
16	White	Lt Gry	Cyan	Blue	Magenta	Red	Yellow	Green

---

### Half Intensity

	<b>- Key -</b>							
Palette	WHT	BLK	CYAN	BLUE	MAGN	RED	YEL	GRN
1	L/I White	Black	L/I Cyan	L/I Blue	L/I Magn	L/I Red	L/I Yel	L/I Green
2	L/I White	Lt Gry	L/I Cyan	L/I Blue	L/I Magn	L/I Red	L/I Yel	L/I Green
3	L/I White	Med Gry	L/I Cyan	L/I Blue	L/I Magn	L/I Red	L/I Yel	L/I Green
4	L/I White	Wrm Gry	L/I Cyan	L/I Blue	L/I Magn	L/I Red	L/I Yel	L/I Green
5	L/I White	Black	L/I Cyan	L/I Blue	L/I Magn	L/I Red	L/I Yel	L/I Green
6	L/I White	Lt Gry	L/I Cyan	L/I Blue	L/I Magn	L/I Red	L/I Yel	L/I Green
7	L/I White	Med Gry	L/I Cyan	L/I Blue	L/I Magn	L/I Red	L/I Yel	L/I Green
8	L/I White	Wrm Gry	L/I Cyan	L/I Blue	L/I Magn	L/I Red	L/I Yel	L/I Green
9	L/I White	Lt Gry	L/I Cyan	L/I Blue	L/I Magn	L/I Red	L/I Yel	L/I Green
10	L/I White	L/I Blue	L/I Cyan	L/I Blue	L/I Magn	L/I Red	L/I Yel	L/I Green
11	L/I White	Med Gry	L/I Cyan	L/I Blue	L/I Magn	L/I Red	L/I Yel	L/I Green
12	L/I White	Wrm Gry	L/I Cyan	L/I Blue	L/I Magn	L/I Red	L/I Yel	L/I Green
13	L/I White	Lt Gry	L/I Cyan	L/I Blue	L/I Magn	L/I Red	L/I Yel	L/I Green
14	Brown	Lt Gry	Blue-Gray	Sky Blue	Purple	Orange	Lt Yel	Mauve
15	Brown	Med Gry	Blue-Gray	Sky Blue	Purple	Orange	Lt Yel	Mauve
16	Brown	Lt Gry	Blue-Gray	Sky Blue	Purple	Orange	Lt Yel	Mauve

L/I = low intensity, Lt = light, Med = medium, Wrm = Warm

Note that the background color is the same as that specified for the Black key.

### VERIFY PROMPT (V P)

**Use**—The Verify command cleans up problems in a custom graphic display (picture) that may not allow the picture to compile. Typically, these problems arise when a picture is modified. The verify routine builds a new internal symbol table that contains only the proper variables and their data types for the picture currently on the screen. The Verify Prompt command allows you to change the type as each variable is checked.

**Procedure**—Read the picture onto the screen. Type **VERIFY** or (**VERIFY PROMPT**) on the command line, then press the ENTER key.

The Picture Editor responds: **Verification in Progress.**

The verify routine searches the system and Display Database trying to find the data type for every variable in the picture. If the command was:

- **VERIFY**, the verify routine accepts the old variable type (if found in the old symbol table). If all variable types are found, only a completion message is presented. If any variable type cannot be found, a screen form appears with the variable name and an input port. Enter the correct variable type and press ENTER.

A100.PV	
Variable Type	

- **VERIFY PROMPT**, the verify routine presents a screen form. If the variable type was found in the old symbol table, the variable type is displayed as the `Current Type`. Press **ENTER** to accept the old variable type or enter a new type in the `Variable Type` port (then press **ENTER**).

A100.PV	
Current Type	Real
Variable Type	

**Cancel**—If the Cancel key is pressed while the verify routine is waiting for input, the command is cancelled.

When all variables have been checked, a verification complete message is presented. The message tells if corrections were made. You must write or compile the picture to save any changes.

### 3.3.1.32 Command: **SYSTEM\_ID nn (SYS\_ID nn)**

where nn is a one or two character network identifier.

**Use**—The System ID command is used to prefix all system variables in a schematic with a network identifier corresponding to another network on which you want the points to reside.

**Procedure**—Read the picture onto the screen. Type **SYSTEM\_ID** and a system identifier on the command line; then press the ENTER key.

Each object in the picture is processed and the network identifier is assigned to all system entities.

When all variables have been changed, the Picture Editor responds:

System ID Command Complete.  
Please WRITE or COMPILE the abstract to save the changes.

#### **Notes**—

**System Identifier**—is the network identifier. It must follow the guidelines for network identifiers, i.e., up to two alphanumeric characters. The network delimiter \ is not specified on the command line, but it is added to all system entities.

**Where**—If a schematic exists on one network but the schematic is to be used on another network, run SYS\_ID on the network that owns the schematic.

If you copy a schematic to another network without running SYS\_ID on the network that owns the schematic, and then run SYS\_ID on the other network, the message: ERROR IN SYS ID COMMAND is displayed.

**& Entities**—If an entity part of the variable starts with &, no network identifier is assigned.

**DDB Entities**—If the entity part of the variable is found in any DDB file (standard, user defined, or equipment list), no network identifier is assigned.

**Existing ID**—If the entity already has a network identifier, no other network identifier is assigned.

If the entity does not begin with &, is not a DDB variable, and no network ID already exists, the \ delimiter and network identifier is added to the beginning of the entity.

**CAUTION**

Any user-defined DDB file or equipment list used to build the picture must be loaded before executing the System\_ID command; otherwise, the command will assign network identifiers to entity names when it shouldn't.

## 3.3.2 Fixed Behavior and Fixed Behavior Commands

### 3.3.2.1 Fixed Behavior

Literal behavior refers to the video characteristics color, blink, intensity, and reverse background. All new information added to the picture will assume the current literal behavior. The letters in the priority-code box on the second line from the top of the screen are displayed in the current literal behavior. The behavior choices and their abbreviated forms are

#### **Color** (in order of priority)

WHITE (WH) (W)  
CYAN (CY) (C)  
MAGENTA (PURPLE) (MAG) (M)  
BLUE (BLU)  
YELLOW (YELL) (YEL) (Y)  
GREEN (GR) (G)  
RED (R)  
BLACK (BLK) (BLA)

#### **Intensity**

FULL (FUL) (F)  
HALF (H)

#### **Background**\*

REVERSE (REV)  
NO REVERSE (NO REV) (N R) (NR)

#### **Blink**

BLINK (BLI)  
NO BLINK (NO BLI) (N B) (NB)

When graphic objects with different colors overlap, the color higher on the above list is predominant. For example, yellow covers red, cyan covers yellow, white covers all. When text and graphic objects overlap, refer to the Set Priority Command.

### **NOTE**

If you are using one of the optional color palettes, color priority is still determined by the color keys as listed above. For example, in palette 16, blue-gray (entered with the cyan key) has a higher priority than orange (entered with the red key).

Current literal behavior can be changed with either the Engineering Keyboard or by executing the Set Behavior Command.

Behavior of existing objects can be changed only with the Add Behavior and Delete Behavior commands.

---

\*Reverse background applies only to text and values.

### 3.3.2.2 Engineering Keyboard Behavior Keys

The Engineering keyboard's behavior keys are used to set the current literal behavior for objects that are thereafter added to the picture. The behavior keys are RED, YEL, GRN, INTEN, BKGND, etc., as listed above.

Note that the keys BLINK, INTEN (intensity), and BKGND (reverse/normal background), cause alternate actions. For example, after pressing INTEN once, objects that are subsequently added to the picture are displayed at half intensity. After pressing the key again, objects that are subsequently added will be displayed at full intensity.

### 3.3.2.3 Fixed Behavior Commands

**Command:** SET BEHAVIOR aaaa

Abbreviated forms: S BEH aaaa  
S B aaaa

where aaaa is a list of the behavior attributes color, intensity, background, or blink as previously listed; for example, S BEH RED HALF

**Use**—The Set Behavior command is used to set the literal behavior for all new objects added to the picture. Once set, this literal behavior applies to all objects that are added to the picture thereafter.

**Procedure**—On the command line, type SET BEHAVIOR followed by a list of one or more attributes, then press the ENTER key.

The attributes can be used in any order but must be separated by a space, for example,

S B HALF BLUE REV NB  
S B YEL BLINK FULL NR

Part of the current attributes can be changed without affecting the rest, for example, if the current behavior attributes are Green, Blink, Full, and Reverse, the command SET BEHAVIOR RED results in the new behavior of Red, Blink, Full, and Reverse.

**CANCEL**—If the CANCEL key is pressed at this point, the current literal behavior reverts back to the previous setting.

At the completion of this command the new current behavior is shown by the color, intensity, etc. of the 3-letter code on the top communication line.

**Command:**    **ADD BEHAVIOR aaaa**

Abbreviated form:    A BEH aaaa  
                          A B aaaa

where aaaa is a list that contains one or more of the behavior attributes listed under the Set Behavior command.

**Use**—This command is used to change the literal behavior for existing objects in the picture. The object or objects must have been previously selected.

**Procedure**—On the command line, type ADD BEHAVIOR followed by a list containing one or more of the behavior attributes.

Press the ENTER key. All selected objects assume the new behavior attributes specified in the command. Unspecified attributes are not changed.

Example: The command ADD BEHAVIOR BLINK REV causes selected objects to assume the behavior BLINK REVERSE and retain whatever color/intensity behavior they had before the command.

**Command:**    **DELETE BEHAVIOR**

Abbreviated form:    DEL BEH  
                          D B

**Use**—The Delete Behavior command removes the literal behavior of selected objects and replaces it with the current literal behavior. An object or objects in the picture must first be selected.

**Procedure**—Type DELETE BEHAVIOR on the command line, then press the ENTER key. All selected objects in the picture assume the current literal behavior.

For example, assume that current literal behavior is green, no blink, no reverse, and half intensity. An object is selected that has the behavior characteristics of red, blink, no reverse, and full intensity. After executing the Delete Behavior command, the object assumes the behavior of green, no blink, no reverse, and half intensity.



**Command:   SELECT BEHAVIOR**

**Use**—This command is used to select and identify all visible objects with the same literal behavior in the current display.

Abbreviated form:   SEL BEH  
                          SEL B

**Procedure**—Type SELECT BEHAVIOR on the command line, then press the ENTER key. The Picture Editor prompts Enter Select Coordinates.

Place the cursor at the first position and press the SELECT key. If the touch-screen option is present, just touch the screen at the desired location. This location is marked by a cross (see Figure 3-23). Move the cursor to the second point and press the SELECT key again (or use the touch screen). After selection of the second point, the Picture Editor draws a box that includes the two points (see Figure 3-24).

**DEL**—If it is necessary to change a coordinate entry, press the DEL key to delete the last entry.

Press the ENTER key to complete the command.

Selected objects are displayed as white and blinking at full intensity.

As an example, refer to Figure 3-23 and assume that only the furnace and pump have the behavior attributes blue, half, no reverse, and no blink. All other objects have a different behavior. The Select Behavior command is invoked and the select-location coordinates are entered as shown.

Only the Furnace was enclosed/intersected by the selection box; however, both it and the pump are selected because they have the same behavior. This command is useful to identify those objects with common behaviors.

Selected objects remain selected until deselected by the Deselect command or acted upon by one of the commands that are used to manipulate selected objects.

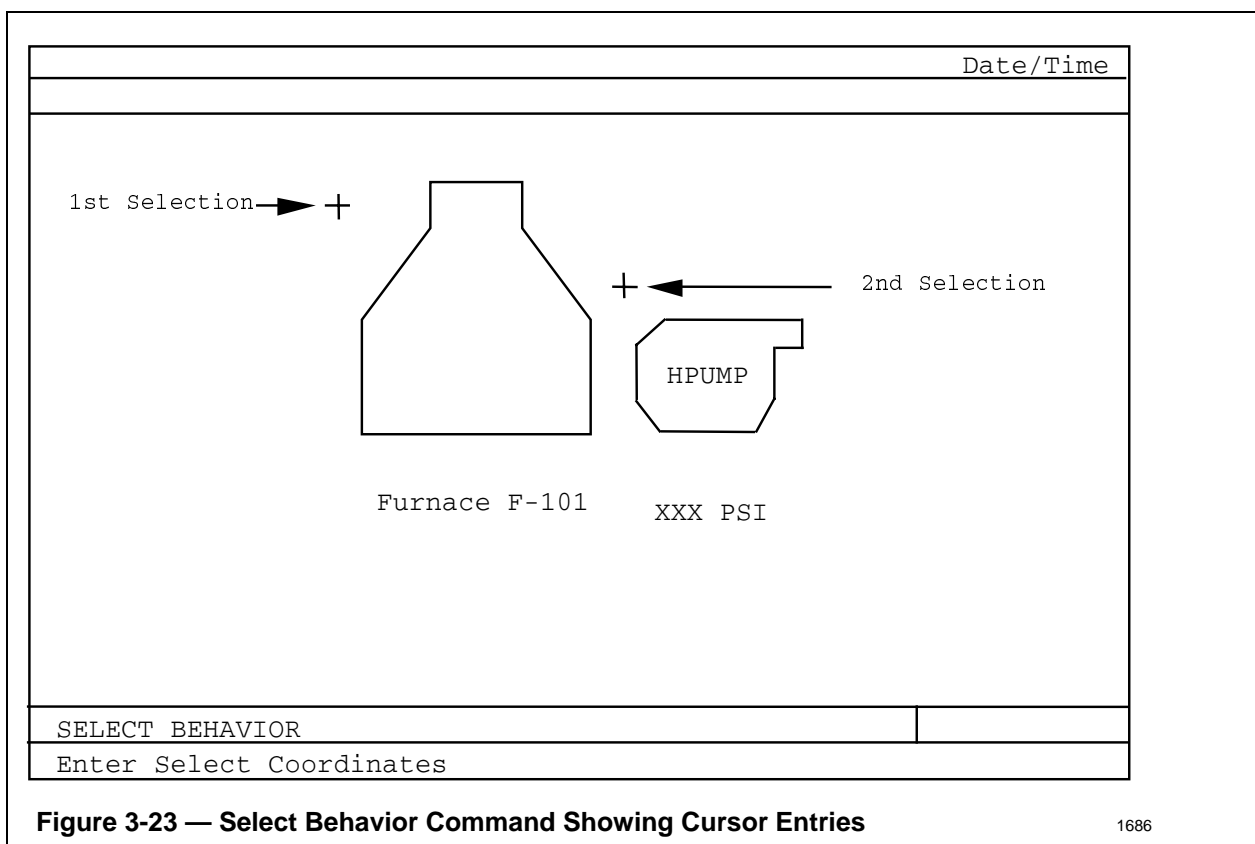
**Command:   DESELECT BEHAVIOR**

Abbreviated form:   DES BEH  
                          DS B

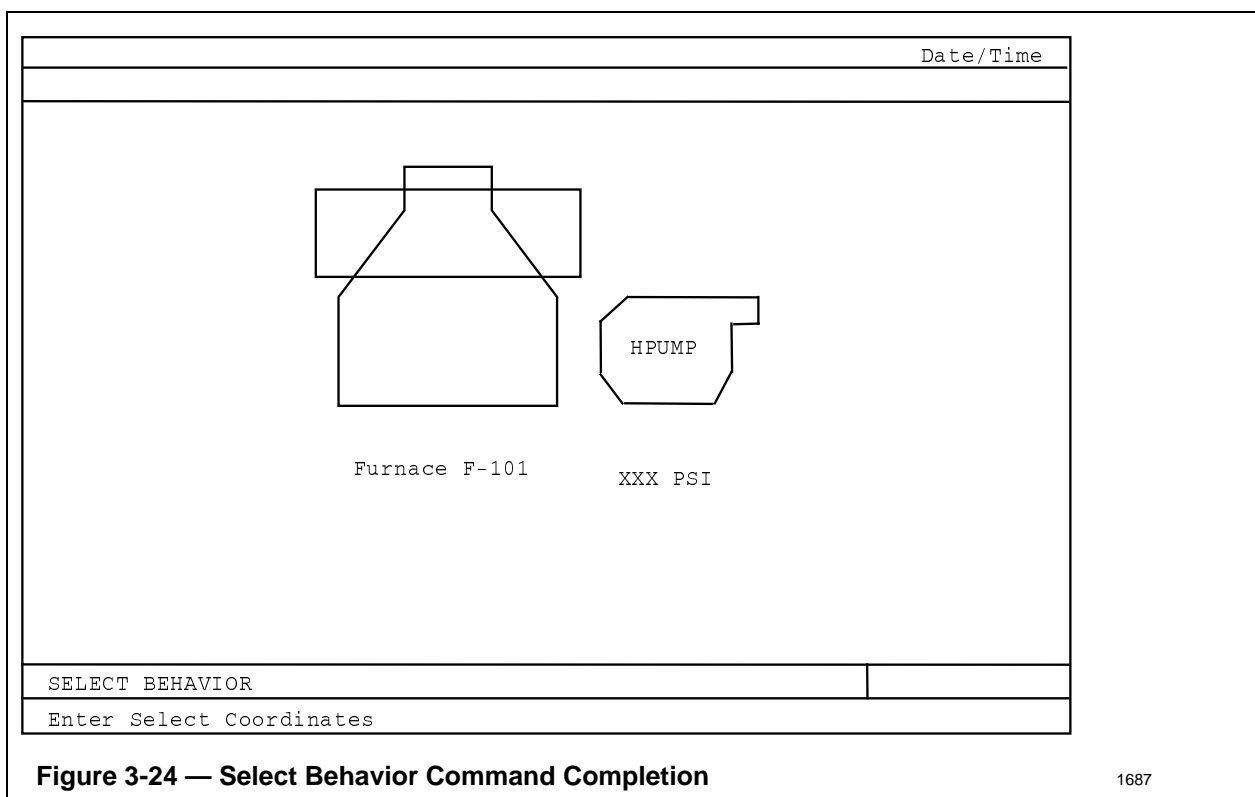
**Use**—This command is the opposite of the Select Behavior command. It is used when different behaviors or sets of behaviors exist for objects in the picture.

**Procedure**—Type DESELECT BEHAVIOR on the command line, then press the ENTER key. The Picture Editor prompts Enter Select Coordinates.

An object is selected as described for the Select Behavior command. The selected object and all objects with the same literal behavior as the selected object are deselected upon command completion.



1686



1687

### 3.3.3 Conditional Behavior

#### 3.3.3.1 Use of Conditional Behavior

Conditional behavior allows objects to

- change color (the choices are the same as for fixed behavior)
- change intensity (full or half)
- change between steady-state and blinking.

Text and values can also change between normal-video and reverse-video fields. As described in the following paragraphs, the process consists of selecting objects, invoking a behavior command, and filling in or modifying a form on the screen to describe the desired behavior. A conditional behavior statement must be written to describe the desired behavior. Conditional behavior syntax is explained in Appendix D of this manual.

#### 3.3.3.2 Conditional Behavior Commands

**Command:**    **ADD CONDITION**

Abbreviated form:    ADD COND  
                          A C

**Use**—This command is used to add conditional behavior to an object. The object or objects that are to have conditional behavior must already exist in the picture and must first be selected.

**Procedure**—Type ADD CONDITION on the command line, then press the ENTER key.

This causes the Picture Editor to present a form (see Figure 3-25). Only one form is presented each time the Add Condition command is used; therefore, if more than one object is selected, all of the selected objects will have the same behavior.

Refer to the following discussion and fill in the Conditional Behavior Form.

**CANCEL**—If the CANCEL key is pressed at this point, the form is erased and the Add Condition command is canceled.

Conditional Behavior Form:

Behavior For Bad Value	Specifies behavior for the object(s) if evaluation of the indicator gives an unreasonable value or the value cannot be obtained (for example, because of hardware failure).
Initial Behavior	The initial behavior specification is an implied ELSE statement for the conditional behavior statements. Also, all unspecified literal behavior attributes default to the initial behavior specification. The object(s) will assume initial behavior during build time.
Conditional Behavior	The conditional behavior statements tell how selected object(s) will appear for different states of an indicator, such as when a condition is true or false, when a certain level is exceeded, or when a point is in alarm. The following example illustrates the true/false status. Conditional behavior syntax is explained in Appendix D of this manual.

Press the ENTER key to complete the command.

**Unknown Variables**—If any of the variables referenced in the conditional behavior statements are unknown to the system or if there are parameters, the Picture Editor prompts for the type of the entity. Refer to Appendix B for a list of valid variable types.

Date/Time

Conditional Behavior

Behavior For Bad Value

Initial Behavior

Condition

<PAGE FWD> <PAGE BACK> to MOVE. <F2> for TFE. <F3>to JUMP <F4> to DELETE.

ADD CONDITION

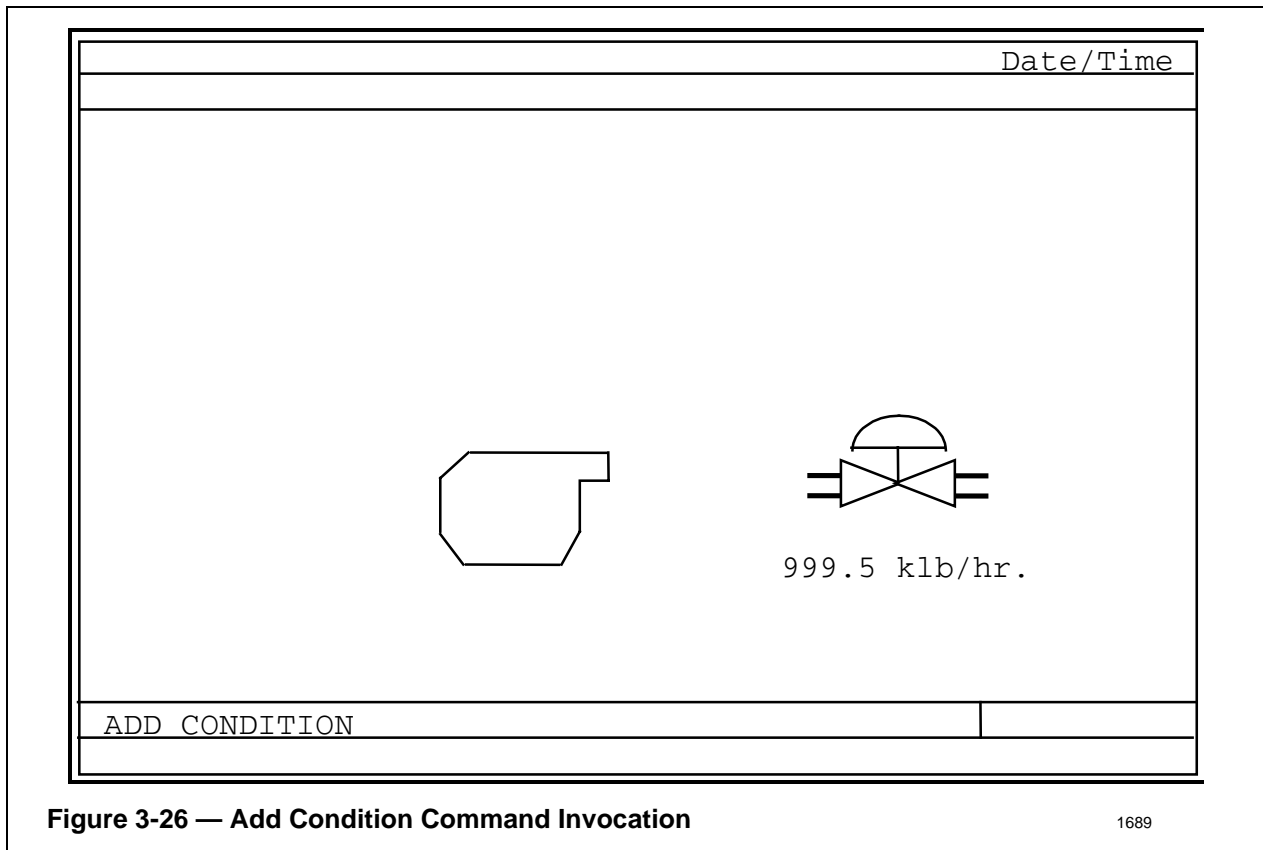
Figure 3-25 — Conditional Behavior Form

1688

Example—Assume that the pump shown in Figure 3-26 is selected and the Add Condition command has been invoked. Figure 3-27 illustrates the form that results and the information that is typed into the form.

In this example, at run time, (i.e., with the Operating Personality running) the pump will be red when the condition is true and blue when the condition is false.

See Appendix D for additional examples.



Date/Time	
<div style="border: 1px solid black; padding: 10px; margin: 10px auto; width: 80%;"> <p>Conditional Behavior</p> <p>Behavior For Bad Value <span style="border: 1px solid black; padding: 2px 10px;">BLACK NO BLINK NO REVERSE FULL</span></p> <p>Initial Behavior <span style="border: 1px solid black; padding: 2px 10px;">CYAN NO BLINK NO REVERSE FULL</span></p> <p>Condition</p> <div style="border: 1px solid black; padding: 10px; margin: 10px auto; width: 80%;"> <pre>IF A100.SP &gt; 100 THEN     SET RED ELSE     SET BLUE</pre> </div> <p style="font-size: small; text-align: center;">&lt;PAGE FWD&gt; &lt;PAGE BACK&gt; TO MOVE. &lt;F2&gt; for TFE. &lt;F3&gt; to JUMP. &lt;F4&gt; to DELETE</p> </div>	
<div style="border: 1px solid black; padding: 2px;"> <span style="float: left; padding-right: 10px;">ADD CONDITION</span> <span style="border: 1px solid black; width: 100%; height: 1.2em; display: inline-block;"></span> </div>	
<div style="border: 1px solid black; padding: 2px;"> <span style="float: left; padding-right: 10px;">Enter Condition Information</span> <span style="border: 1px solid black; width: 100%; height: 1.2em; display: inline-block;"></span> </div>	

**Figure 3-27 — Add Condition Example** 1690

**Screen Form Entry Aids**—The Picture Editor provides several screen form entry aids that can be used with this and several other commands. They are explained here for convenience and referenced by other commands where they apply.

**Comments**—Comments are useful to explain your sequences. Comments must be enclosed in curly braces { }.

Example—

```
IF A100.SP >100 {over temperature}
SET RED
ELSE{Normal operation}
SET BLUE
```

A multiple page port is allowed. If you need additional room to add condition statements or comments, press the PAGE FWD key and continue the entry. Do not allow identifiers or key words to be broken between two pages. For example, the variable ID A1000.PV\_ARR(25) cannot be broken. Do not break up an IF, THEN statement. Press PAGE BACK (or use the Jump function) to display previous pages. If you invoke the Print command, it will print out the multiple page port in a continuous stream.

**Jumping**—To jump to another page, hold down <CTL> and press <3>. Enter the page number that you want to jump to in the port that opens; then press <ENTER>.

**Delete**—To delete the on-screen page of a multiple page entry, hold down <CTL> and press <4>. Page one is not affected, but any other page is deleted and the entries concatenated.

**Text File Editor**—hold down <CTL> and press <2> to copy the entries into and invoke the text file editor (TFE). You can then edit multipage entries as one continuous stream. The text file pathname is determined by the entry in the Temporary File Directory of the Default Volume Pathnames display. Before invoking the TFD, you can escape from the Picture Editor and type SP to access/change the directory, or accept the default directory.

Note that the Picture Editor port is two characters shorter than the TFE line. If you use the last two characters on a line in the TFE, when you return to the picture editor, those characters are moved to the next line, possibly causing a break in identifiers or key words.

Do not delete or change the name of the text file being edited because the Picture Editor only remembers the TFE entry filename. If an upset occurs and the file was not stored, the Picture Editor will try to restore the data in its ports to the state when the TFE was invoked.

Use the Text File Editor commands and functions to edit the file. Get out of the TFE with the Exit function keys.

To return to the Picture Editor, hold down <CTL> and press <HELP>. The text file is copied into the Picture Editor port. Press <ENTER> to check syntax.

**Command:**    **MODIFY CONDITION**

Abbreviated form:    MOD COND  
                          M C

**Use**—This command allows you to change a conditional behavior screen form that was filled out during a previous Add Condition command.

The object or objects with conditional behavior must have been previously selected with the Select or Select Condition command.

**Procedure**—Type MODIFY CONDITION on the command line, then press the ENTER key. The conditional behavior form for the first selected object will appear on the screen. The form can be modified by typing over previous entries.

After modifying the form, press the ENTER key; to accept the form without change just press the ENTER key.

If more than one object with conditional behavior is selected, each form appears on the screen in turn and you can modify or accept it as described above. The command ends when all of the forms have been presented.

**DEL key**—If the DEL key is pressed while a form is present on the screen, the current form is erased and the form for the next object is presented. If no more forms remain to be presented, the command is canceled.

**Command:**     **DELETE CONDITION**

Abbreviated form:   DEL COND  
                          D C

**Use**—The Delete Condition command removes the conditional behavior from all selected objects that have it and replaces that behavior with the current literal behavior.

**Procedure**—An object or objects in the picture must be selected with the Select Condition command.

Type DELETE CONDITION on the command line and press the ENTER key. Selected objects assume the current literal behavior.

**Example**—Assume that current literal behavior is green, no blink, no reverse, and half intensity. A selected object has conditional behavior characteristics of red, blink, no reverse, and full intensity. After executing the Delete Condition command, the object assumes the behavior of green, no blink, no reverse, and half intensity.

**Command:**     **SELECT CONDITION**

Abbreviated form:   SEL COND  
                          SEL C

**Use**—This command is used to select visible objects with conditional behavior.

**Procedure**—Type SELECT CONDITION on the command line, then press the ENTER key. The Picture Editor prompts Enter Select Coordinates.

Move the cursor to the first position and press the SELECT key. If the touch-screen option is present, just touch the screen at the desired location. The selection is marked by a cross (see Figure 3-28). Move the cursor to the second point and press the SELECT key again (or use the touch screen). Selection of the second point causes the box to be drawn.

Press the ENTER key to complete the command.

**DEL**—If it is necessary to change a coordinate entry, press the DEL key to delete the last entry.

Selected objects are displayed as white, blinking, and with full intensity. If several objects have identical conditional behavior\*, selecting any of them, selects them all.

For an example of how this command is used, refer to Figure 3-28. Assume that the furnace and pump have identical conditional behavior attributes\*. The Select Condition command is invoked and the two coordinate points are entered. As shown in Figure 3-29, only the Furnace was enclosed/intersected by the box, however, both it and the pump are selected. A subsequent Modify Condition command could now be used to change conditional behavior attributes of selected objects.

---

\*Identical conditional behavior means that the behavior was added to all the objects with a single Add Condition command.



Selected objects remain selected until either deselected by the Deselect Condition command or until acted on by commands that are used to manipulate selected objects.

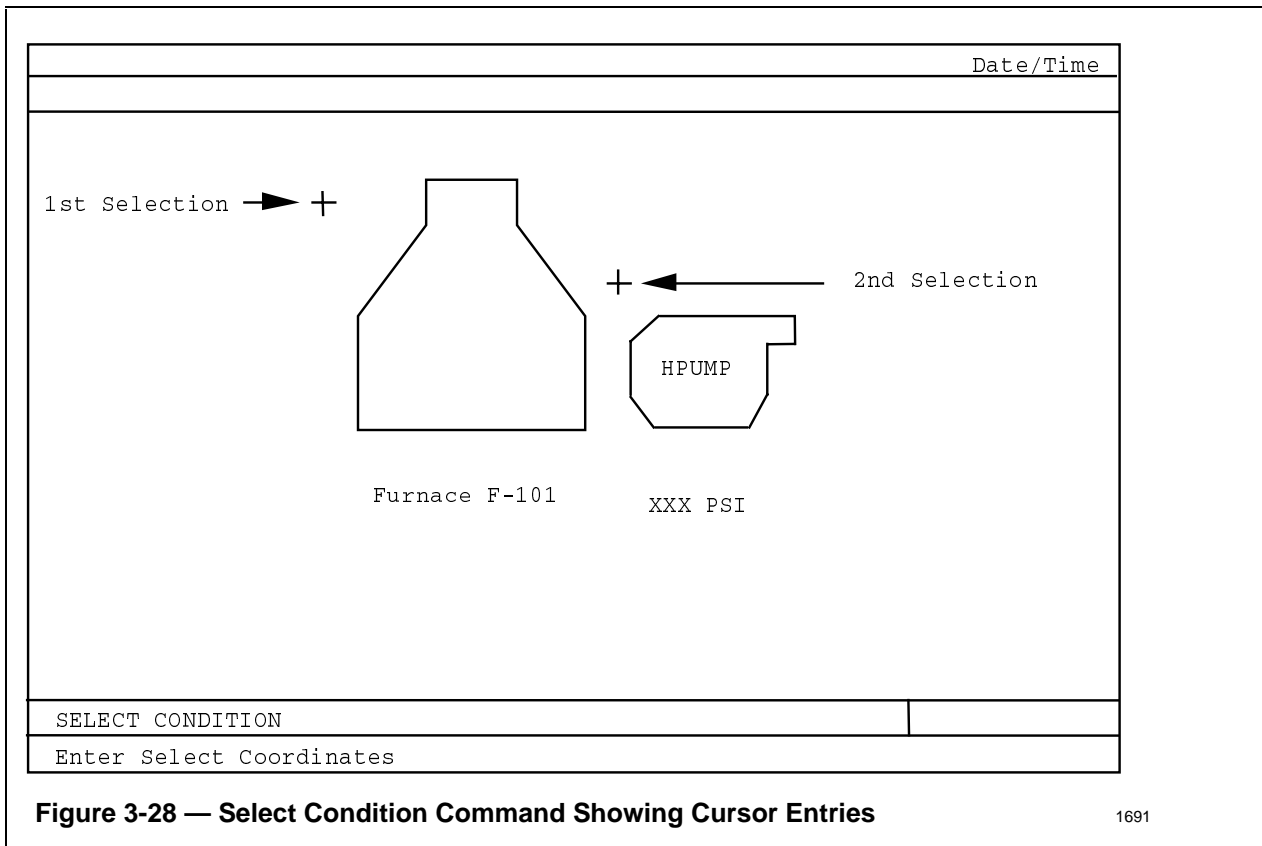
**Command:**     **DESELECT CONDITION**

Abbreviated form:   DES COND  
                          DS C

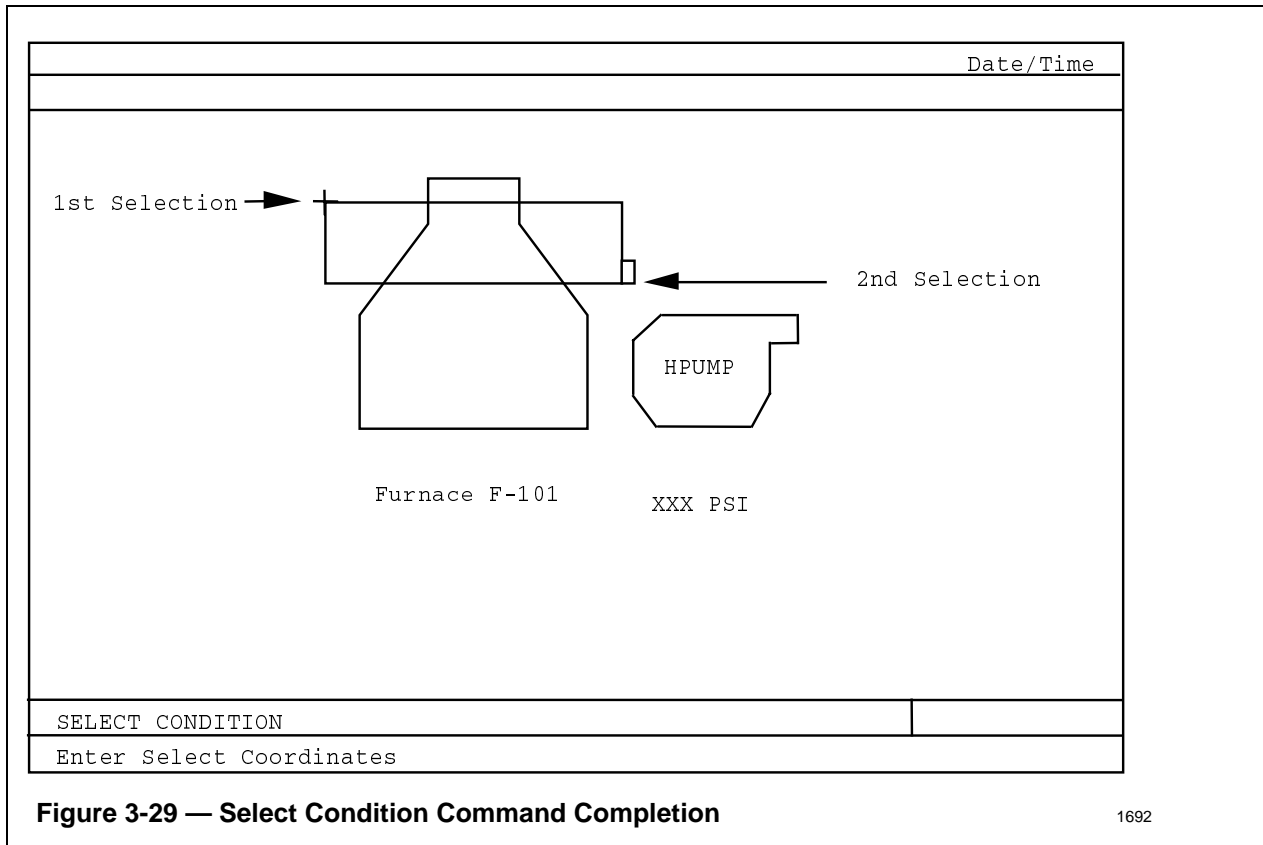
**Use**—This command is the opposite of the Select Condition command. It is used to deselect selected objects with conditional behavior (or sets of selected objects with identical conditional behavior<sup>\*</sup>).

**Procedure**—Type DESELECT CONDITION on the command line, then press the ENTER key. The Picture Editor prompts: Enter Select coordinates.

Coordinates are entered in the same way as described for the Select Condition command. Selected object(s) are deselected upon command completion. When several selected objects have identical conditional behavior, deselecting any one of them deselects them all.



<sup>\*</sup>Identical conditional behavior means that the behavior was added to all the objects with a single Add Condition command.



**Figure 3-29 — Select Condition Command Completion**

1692

### 3.3.4 Graphic Commands

The commands described in this section are used to draw or manipulate shapes and connecting lines that make up the picture. Hollow shapes (outlines) are drawn with the Add Line command. Solid shapes (filled with color) are drawn with the Add Solid command. Some related commands such as Move Line/Solid and Delete Line/Solid are described in other sections of this manual.

#### 3.3.4.1 Command: ADD LINE (ADD LIN) (A L)

**Use**—This command is used to add a line to the picture. Lines can be drawn in any direction and in multiple continuous segments, (for example, to form shapes).

**Procedure**—Executing the Add Line command causes the Picture Editor to prompt ENTER LINE COORDINATES.

Position the cursor at the first end-point and press the SELECT key. If the touch-screen option is present, just touch the screen at the desired location. The Picture Editor marks this end-point with a cross on the screen (see Figure 3-30).

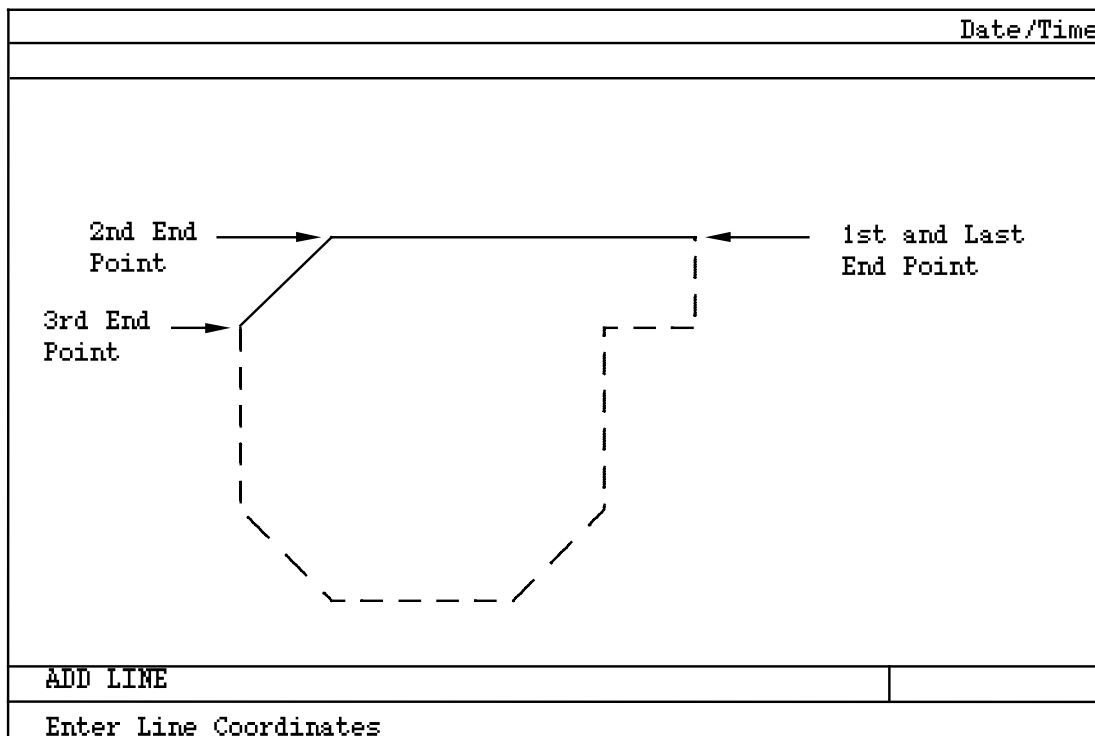
Move the cursor to each succeeding end-point and press the SELECT key (or use the touch-screen option). The end points are connected by straight-line segments.

Pressing the DEL key at this time deletes the previous end-point and allows re-entry if desired.

When all line segments have been specified, press the ENTER key. The line or line drawing is added to the picture in the current priority and with the current literal behavior (see Figure 3-31).

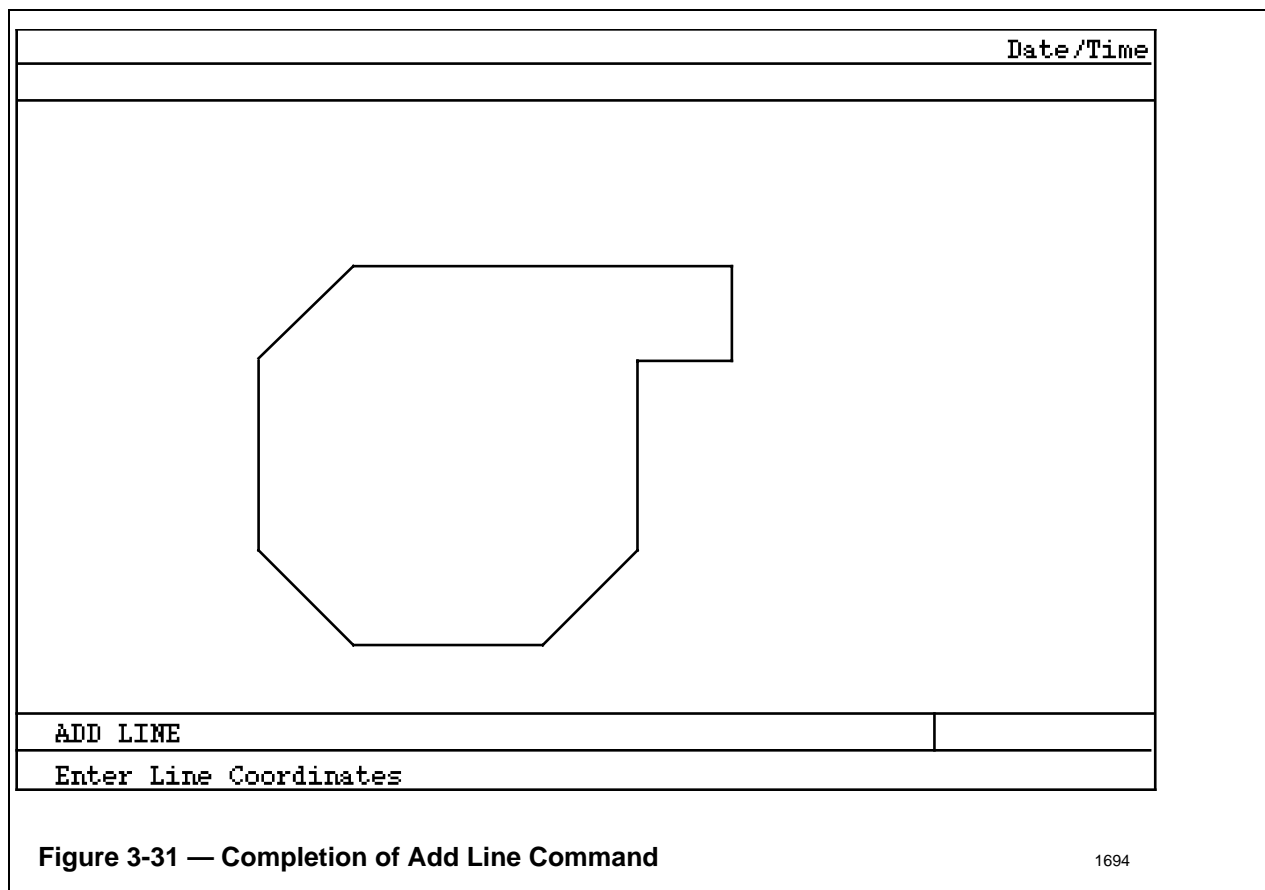
### NOTE

In the example, the pump was drawn oversized to clearly illustrate the procedure. If the object is drawn either too large or too small, it can be easily changed with the Scale command.



**Figure 3-30 — Drawing Shapes with the Add Line Command**

1693



**Figure 3-31 — Completion of Add Line Command**

1694

### 3.3.4.2 Command: **ADD SOLID (A SOL)**

**Use**—This command is used to add a polygon to the picture. The polygon is filled with the current literal color at the current intensity, etc., as called for by the current literal behavior attributes.

**Procedure**—Invoking the Add Solid command causes the Picture Editor to prompt: ENTER SOLID COORDINATES.

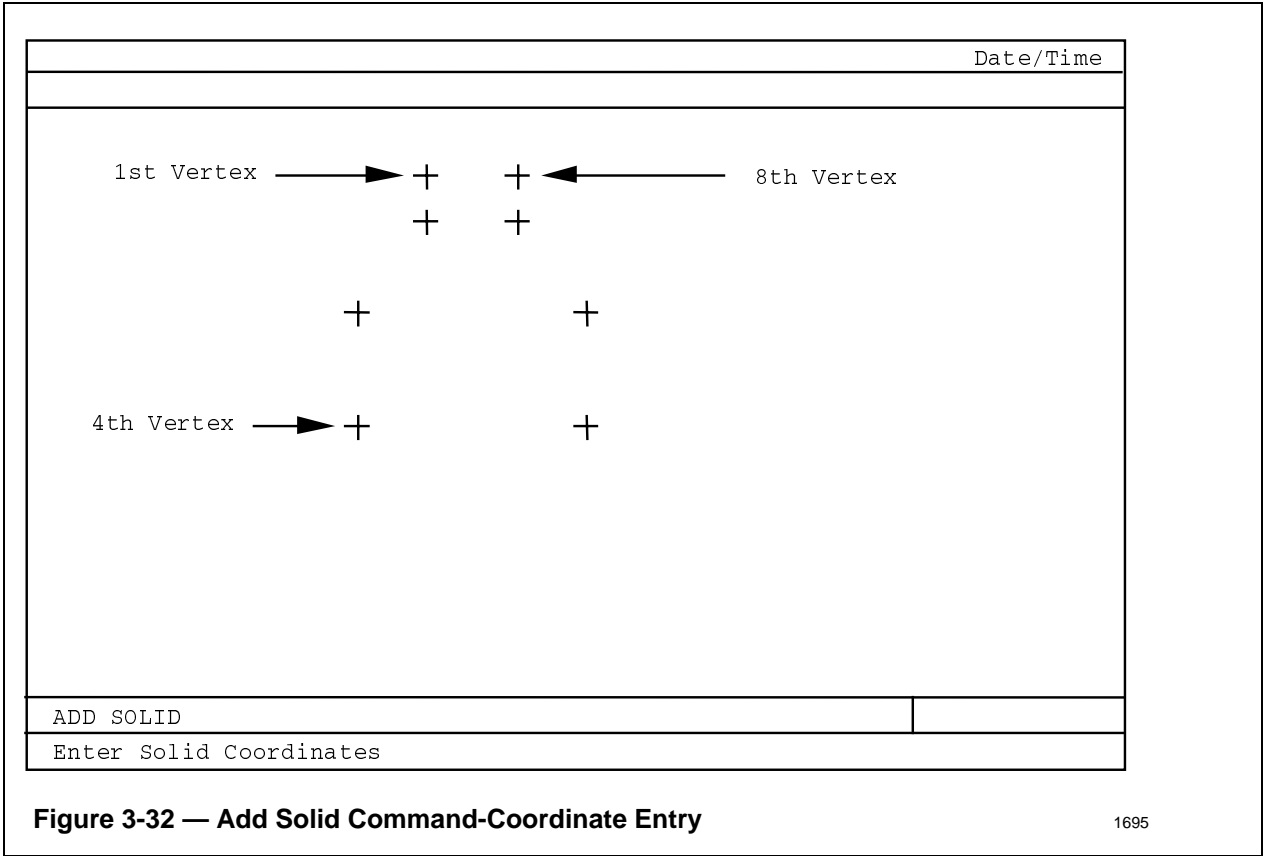
Move the cursor to the first vertex and press the SELECT key. If the touch-screen option is present, just touch the screen at the desired location. A cross appears to mark the first position (see Figure 3-32).

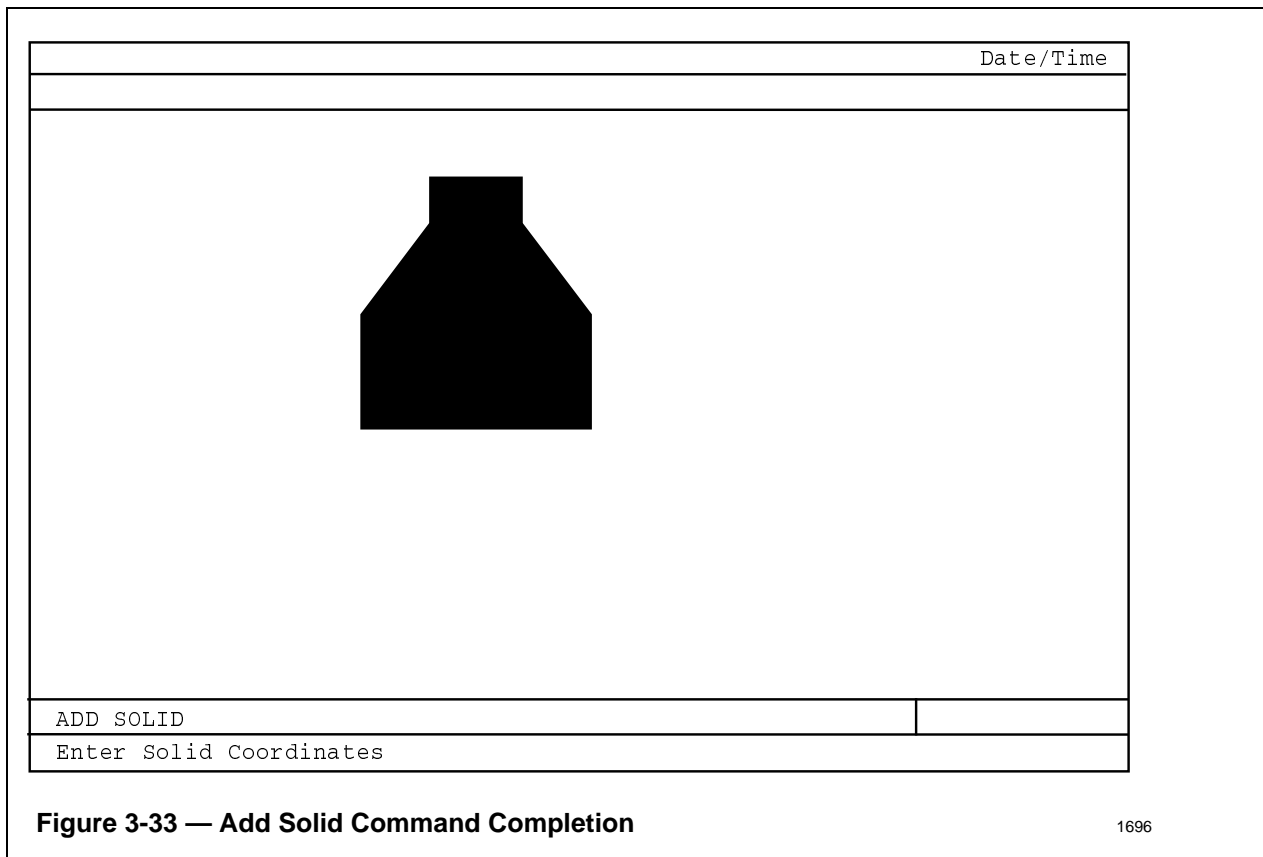
Continue in the same way until all vertices of the polygon are marked.

Press the ENTER key. The Picture Editor automatically connects the first and last vertex (see Figure 3-33).

If the DEL key is pressed at any time while specifying the vertices, the line between the last two locations is deleted and a new vertex can be entered. If the DEL key is pressed again, the previous line is erased.

The color-filled polygon is added to the picture in the current behavior and priority.





**Figure 3-33 — Add Solid Command Completion**

1696

### 3.3.4.3 Command: SCALE (SC)

**Use**—The scale command is used to change the size of selected objects or to rotate them about either their horizontal or vertical axis. Targets cannot be scaled, and objects that contain text or values cannot be scaled. Objects that you want to scale must first be selected. If only one object is selected, scaling occurs about the center of that object. If more than one object is selected, scaling occurs about the center of the group of objects.

**Procedure**—Type SCALE on the command line and press the ENTER key.

The Picture Editor responds with the prompt Enter Scale Bounding Box. There are two ways to proceed from this point

1. If you press the ENTER key, a screen form appears. Figure 3-34 illustrates the form with its default values. You can type over the default values to change the entries. Fill in the form as follows:

- Scale Factor

Enter the scale factors for X (horizontal) and Y (vertical) directions. Numbers smaller than one make the image smaller (e.g., 0.5 reduces the shape by half its current size in that

direction). Numbers greater than 1 make the image larger (e.g., 2 doubles the size in that direction).

Negative numbers and numbers greater than 100 are illegal.

The default values are 1.0 (no change).

- Right-Left Reflection and Top-Bottom Reflection

The Right-Left Reflection and Top-Bottom Reflection choices are answered yes or no. Yes to flip the image horizontally or vertically; No to remain as is.

The default values are No for both reflection choices.

- Rotation Angle

Solid or line objects can be rotated by entering the desired rotation amount in degrees (e.g., 30.7). Note that subpictures cannot be rotated (but lines and solids in pictures that you build for use as a subpicture (i.e., the source picture) can be rotated.

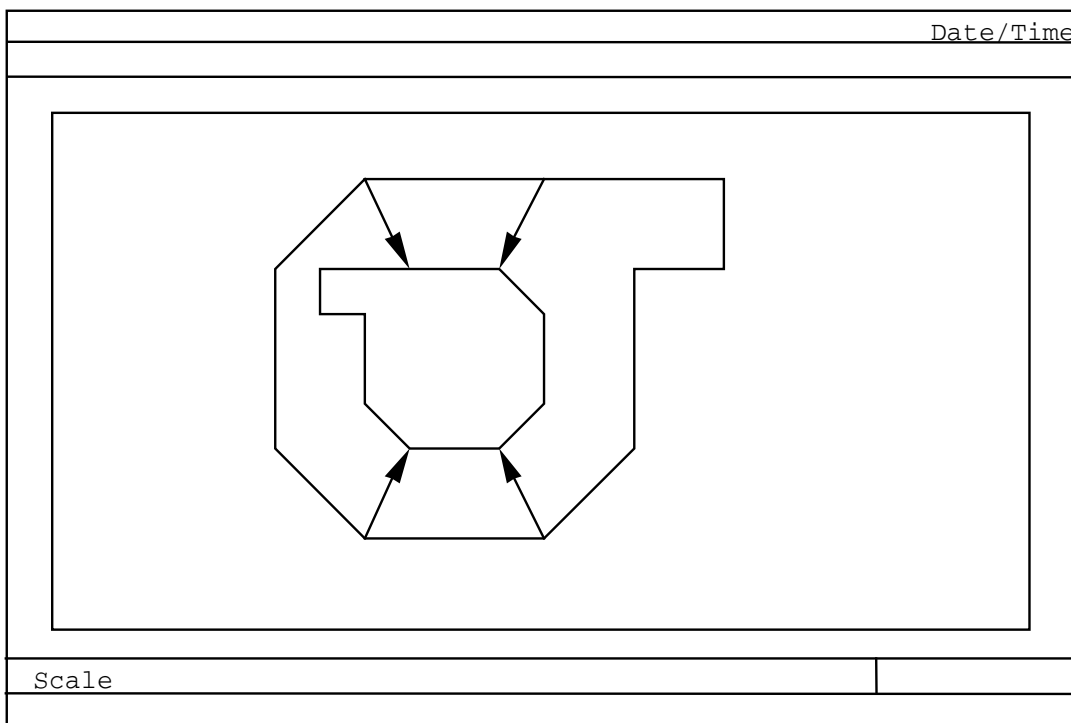
Figure 3-35 shows the effect of reducing the image with X/Y scale factors of 0.25 and answering the Right-Left Reflection with YES.

2. After invoking the Scale command, you can scale selected objects by specifying a bounding box. The bounding box is specified by entering two coordinate locations that become opposite corners of the box. When the ENTER key is pressed, all selected objects are scaled and moved to fit in the area described by the bounding box. The example illustrates this procedure.

Date/Time											
<div style="border: 1px solid black; padding: 10px; margin: 10px;"> <p>SCALING FACTORS</p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 40%;">X Scale Factor</td> <td style="border: 1px solid black; text-align: center;">1.0</td> </tr> <tr> <td>Y Scale Factor</td> <td style="border: 1px solid black; text-align: center;">1.0</td> </tr> <tr> <td>Right-Left Reflection</td> <td style="border: 1px solid black; text-align: center;">No</td> </tr> <tr> <td>Top-Bottom Reflection</td> <td style="border: 1px solid black; text-align: center;">No</td> </tr> <tr> <td>Rotation Angle (CW)</td> <td style="border: 1px solid black; text-align: center;">0</td> </tr> </table> </div>		X Scale Factor	1.0	Y Scale Factor	1.0	Right-Left Reflection	No	Top-Bottom Reflection	No	Rotation Angle (CW)	0
X Scale Factor	1.0										
Y Scale Factor	1.0										
Right-Left Reflection	No										
Top-Bottom Reflection	No										
Rotation Angle (CW)	0										
Scale											
Enter Scale Factor Information											

**Figure 3-34 — Scale Form with Default Values**

1697



**Figure 3-35 — Effects of Scale Command**

1698



## Bounding Box Example

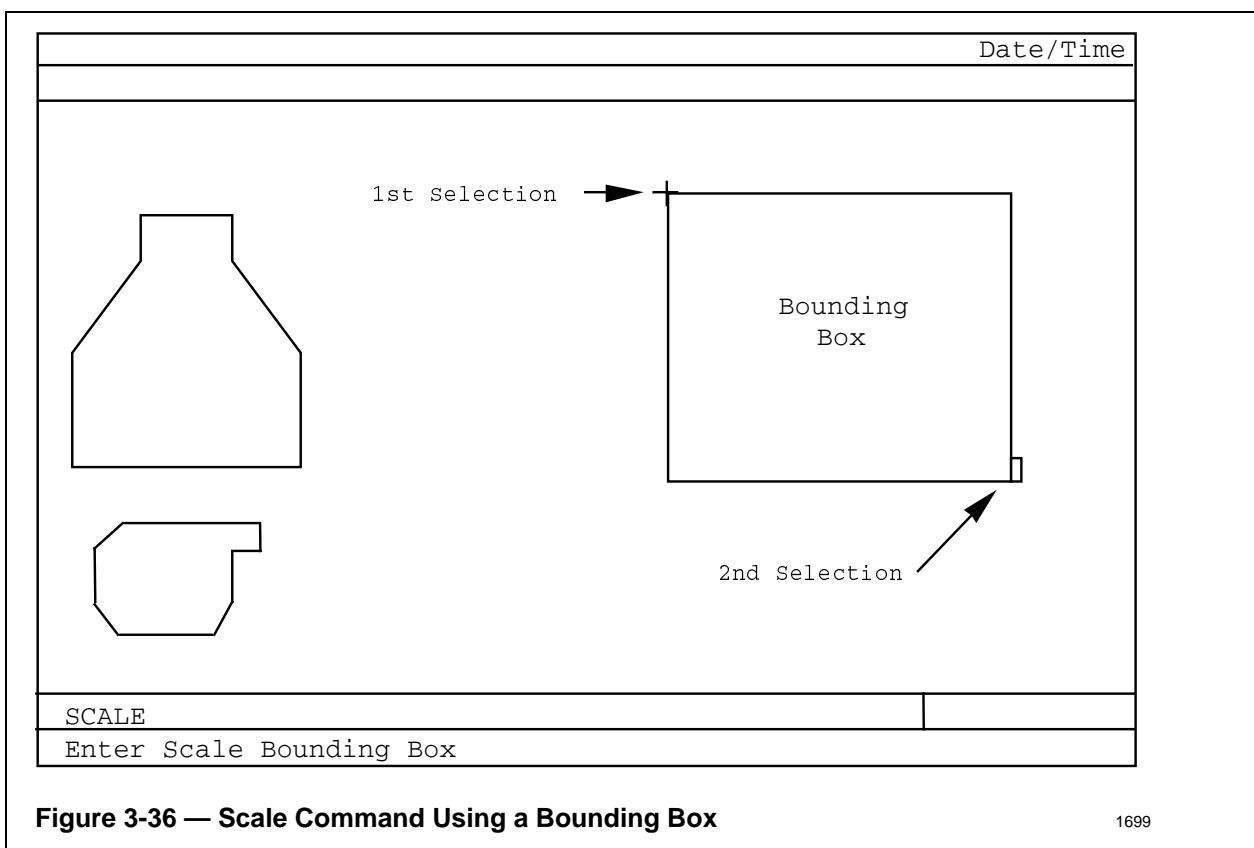
Figure 3-36 shows a furnace and pump that are to be scaled down and moved. Assume that the objects have been selected and that the Scale command was invoked. To create the bounding box, move the cursor to the first position and press the SELECT key. If the touch screen option is present, just touch the screen at the desired location. A cross appears on the screen to mark this location. Move the cursor to the next location and press SELECT (or use the touch screen). As the second location is entered, the Picture Editor draws a box that includes the two locations.

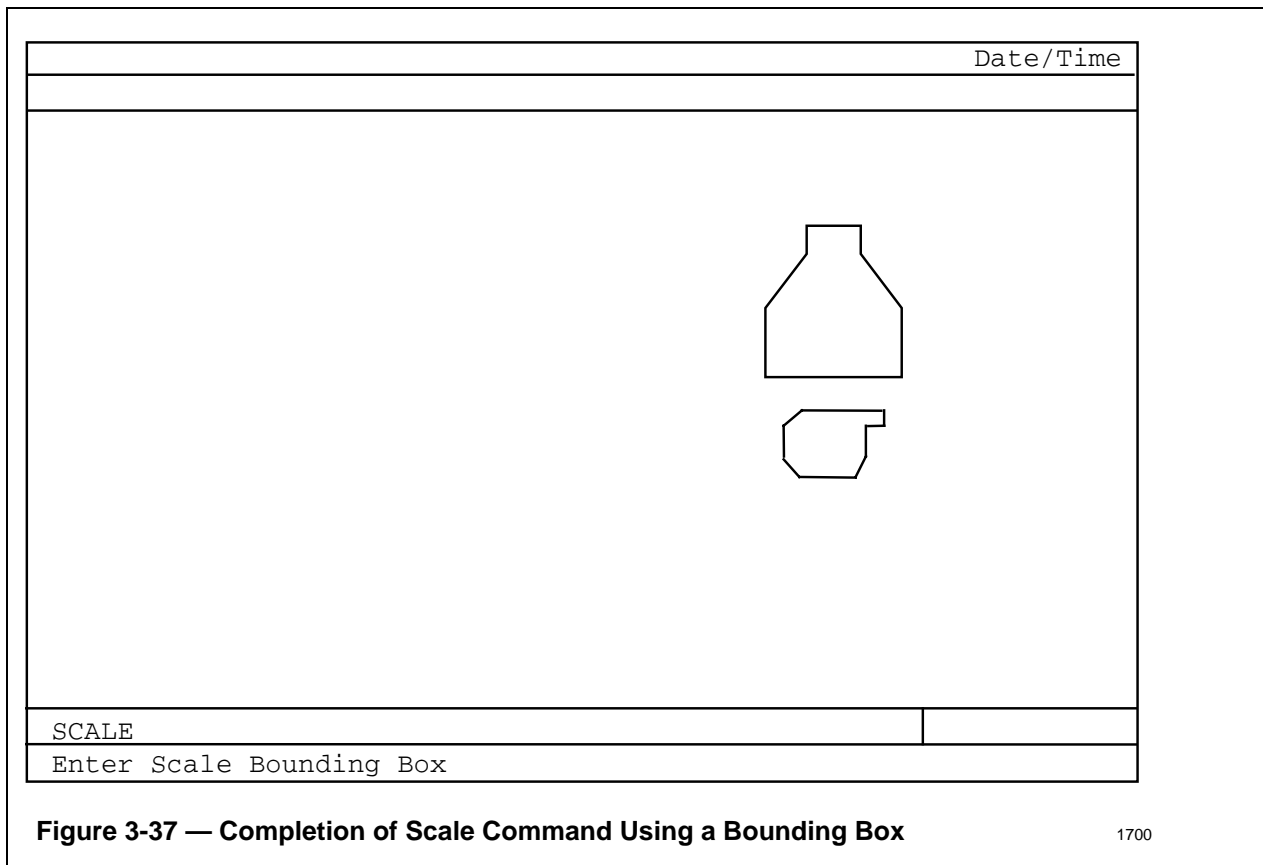
Here are three choices at this point:

**ENTER**—If the ENTER key is pressed, selected objects are scaled and moved to fit in the area described by the bounding box (see Figure 3-37).

**DEL**—If the DEL key is pressed, the last coordinate entered is deleted and you can re-enter that coordinate. If the DEL key is pressed when no coordinate locations remain, the command is canceled.

**CANCEL**—If the CANCEL key is pressed, the command is canceled.





**Figure 3-37 — Completion of Scale Command Using a Bounding Box**

1700

**Qualifiers**—The Scale command can also be used with a qualifier. In this case, the syntax is

**Command:**    **SCALE qqqq**

**Abbreviated form:**    **SC qqqq**

Where the qualifier qqqq can be any of the following:

Bar	Solid (SOL)	Variant* (VAR)
Line (L)	Subpicture (SUB) (S)	

If a qualifier is used, only objects of the type specified are scaled (and must be selected).

#### **3.3.4.4 Command: MODIFY LINE (MOD LIN) (M L)**

**Use**—The Modify Line command is used to change all or part of an existing line or multisegment-line drawing.

**Procedure**—The line must first be selected. If more than one line is selected, they are presented in order for modification.

Type MODIFY LINE on the command line and press the ENTER key.

\*Variants that display a text string for any event cannot be scaled.

The Picture Editor responds by positioning the cursor on the first end-point of the line.

If you want to change the end-point, move the cursor to a new position and press the SELECT key. If the touch-screen option is present, you can touch the screen at the desired location.

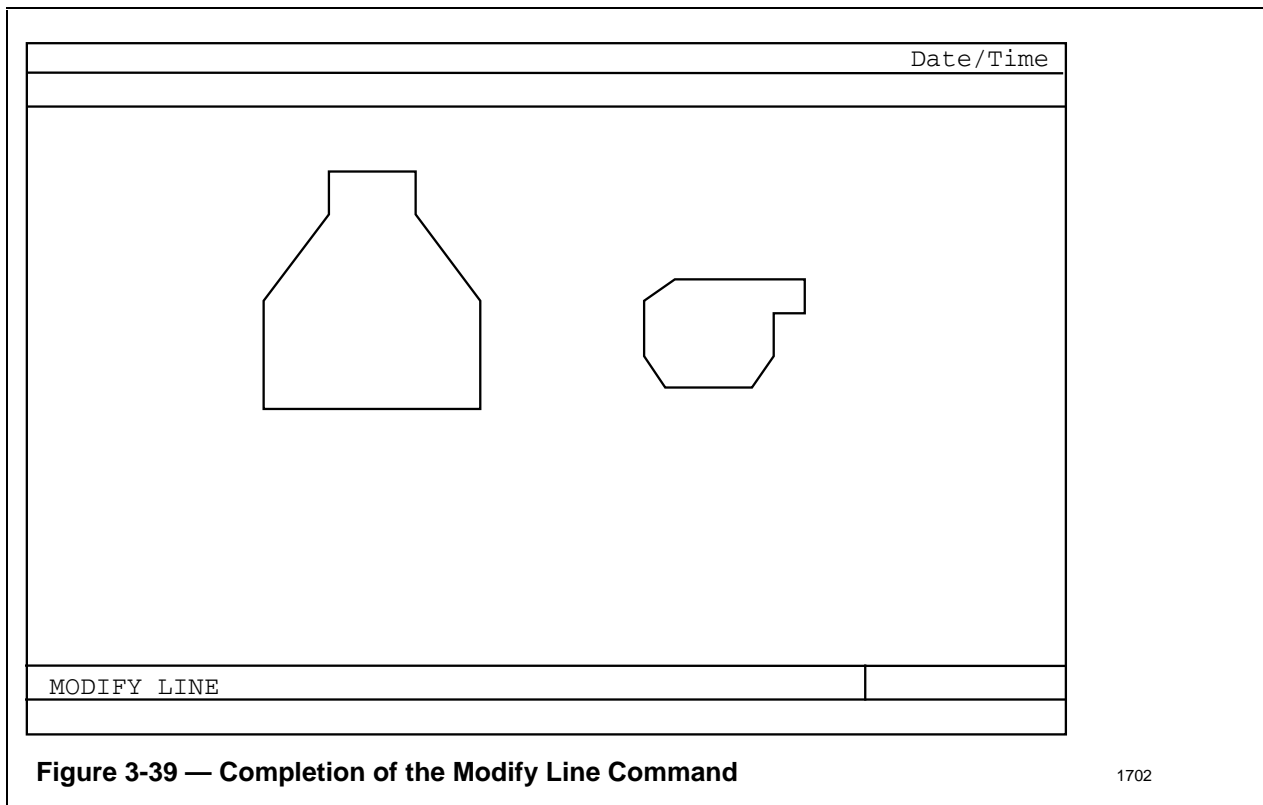
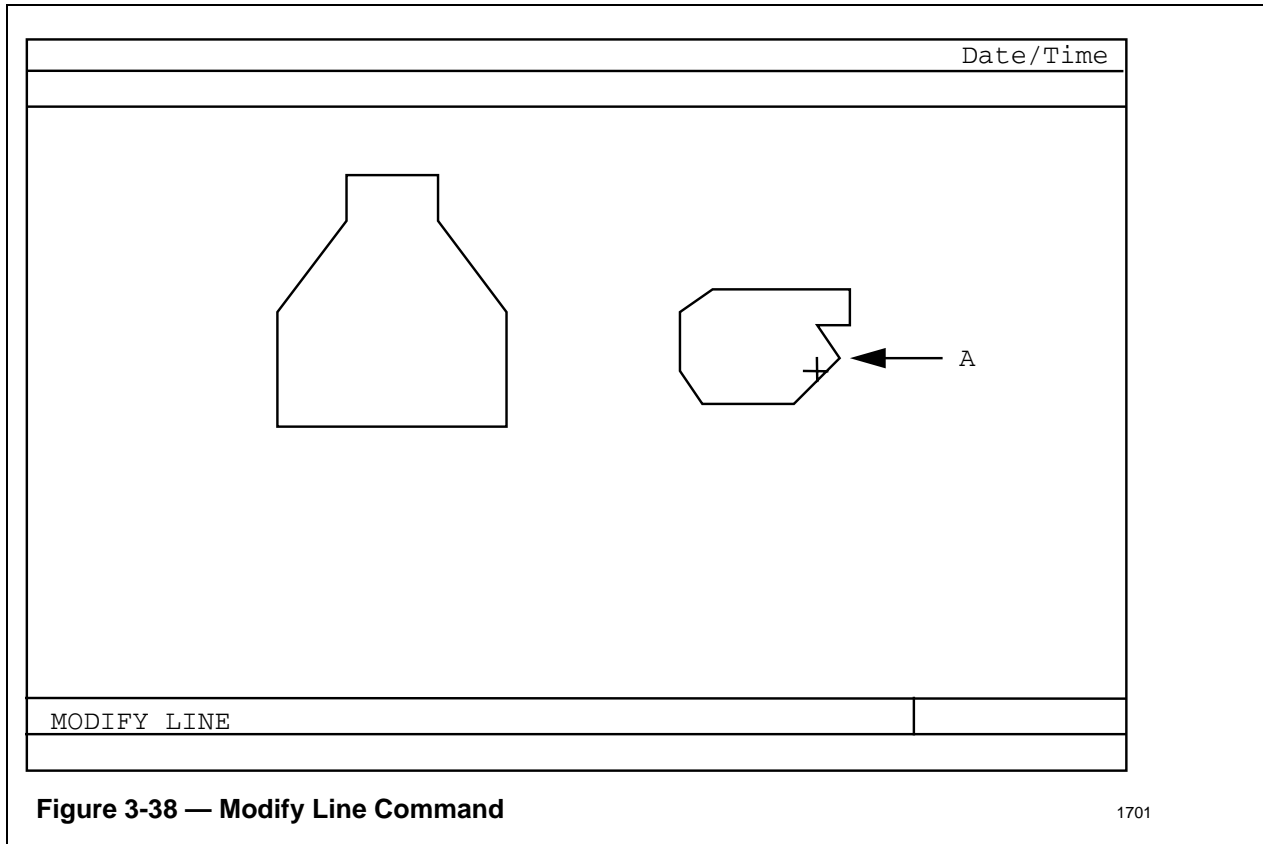
If you do not want to change the end-point, press the SELECT key without moving the cursor.

In either case, the picture editor moves the cursor to the next end-point. This point can be changed or accepted as described above.

End-points can be deleted by pressing the DEL key instead of the SELECT key.

The modification ends when all end-points have been presented or when the ENTER key is pressed. If any end-point of the line is off the drawing area, the line is not modified. If any end-point of the line is not currently visible, the edit region is rolled so that the end-point is visible.

Figure 3-38 illustrates a pump drawing that is to be modified. First the pump is selected, then the Modify Line command is invoked. The pump end-points are accepted by pressing the SELECT key until the cursor is positioned as shown by the arrow in Figure 3-38. The cursor is moved to a new end-point (marked by a cross in this illustration) where the Select key is pressed. The remaining end-points are accepted by pressing the ENTER key. The corrected drawing is shown in Figure 3-39.



#### 3.3.4.5 Command: **MODIFY SOLID (MOD SOL)**

**Use**—The Modify Solid command is used to redraw part of an existing color-filled polygon (called a solid or solid shape).

**Procedure**—The polygon must have been previously selected. If more than one polygon is selected, each is presented in order for modification.

Type **MODIFY SOLID** on the command polygon and press the **ENTER** key.

The Picture Editor responds by positioning the cursor on the first vertex of the polygon.

If you want to change the vertex, move the cursor to a new position and press the **SELECT** key. If the touch-screen option is present, you can touch the screen at the desired location.

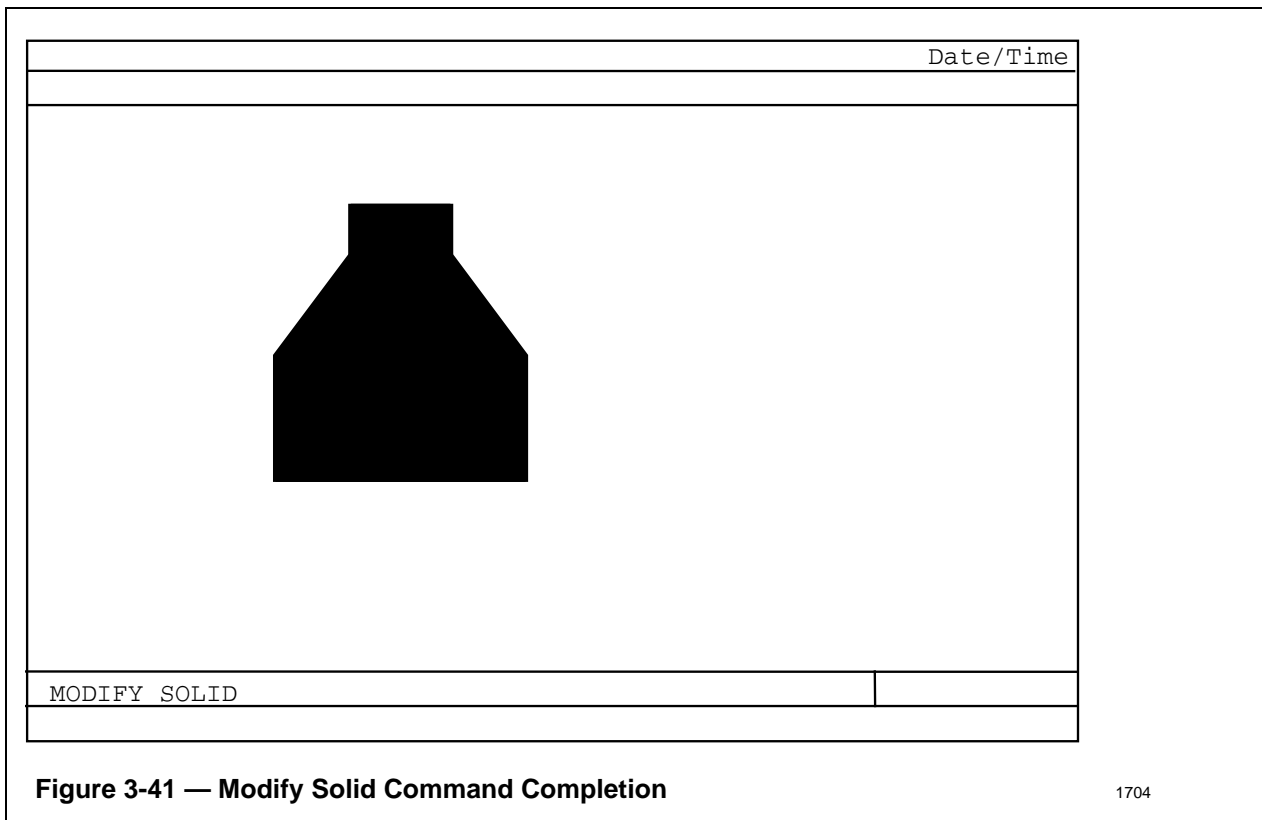
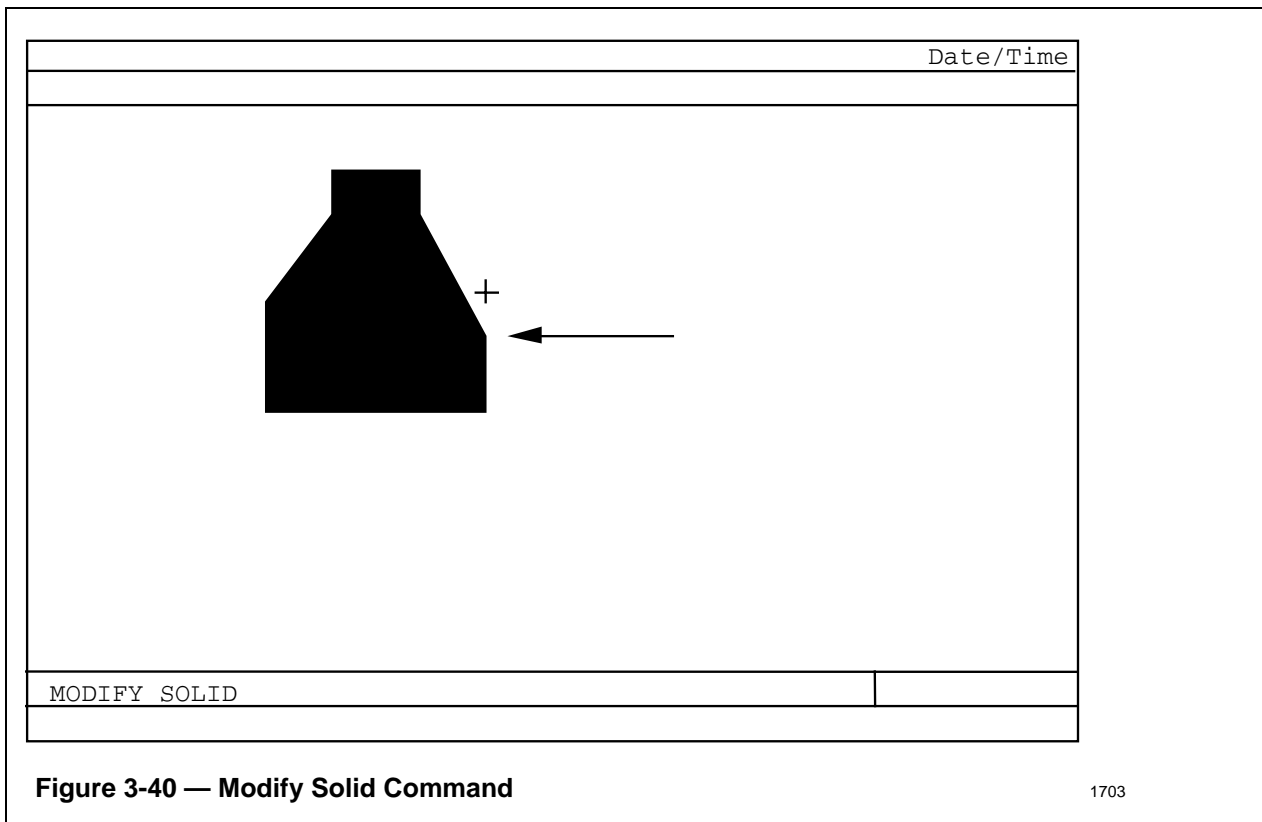
If you do not want to change the vertex, press the **SELECT** key without moving the cursor.

In either case, the picture editor moves the cursor to the next vertex. This point can be changed or accepted as described above.

Vertices can be deleted by pressing the **DEL** key instead of the **SELECT** key.

The modification ends when all vertices have been presented for change or when the **ENTER** key is pressed. If any vertex of the polygon is off the drawing area, the polygon is not modified. If any end-point of the polygon is not currently visible, the edit region is rolled so that the vertex is visible.

Figure 3-40 illustrates a furnace drawing that is to be modified. First, the furnace is selected, then the Modify Solid command is invoked. The furnace vertices are accepted, in turn, by pressing the **SELECT** key until the cursor is positioned as shown by the arrow on the vertex at the right side of the furnace. The cursor is moved to a new vertex (shown by a cross in this illustration) where the Select key is pressed. The result is shown in Figure 3-41.



### 3.3.5 Text and Text Commands

Text characters include all of the uppercase and lowercase alphabetical characters, the numbers 0-9, the standard punctuation characters, and symbols available on the Engineering Keyboard.

Text can be typed, copied, or moved into any of the character positions on the drawing area. Character positions are defined by the line/column format as described at the beginning of the Commands section in this manual (the lines and columns are not visible on the screen).

Two text sizes are available:

- Large size text is five pixels wide by nine pixels high on a cell that is eight pixels wide by 16 pixels high.
- Small size text is five pixels wide by seven pixels high on a cell that is eight pixels wide by eight pixels high.

Cursor movement is set to a granularity that is appropriate for use with the currently selected text size.

The default text size is large but can be changed with the Set Text size command. The size of existing text can be changed with the Add Textsize command. Both of these commands are described later in this section.

#### Special Considerations

When adding objects to a picture, the following text size rules apply:

**Subpictures**—A subpicture has a text size that is determined when it is saved. If a Subpicture contains any large text, or objects that are considered as large-text objects, the text size of the Subpicture is large. Otherwise, the text size of the Subpicture is Small.

**Variants**—The text size of a Variant is determined by the current text size (large or small) when the Variant is built. Furthermore, if the Variant has a text size of small and calls for a Subpicture that has a text size of large, the Add Variant command is aborted and an error results.

**Values**—The text size of a Value is determined by the current text size (large or small) when the Value is built.

**Targets**—Targets always align to large character cell boundaries (i.e., the text size attribute of a Target is always large).

**Lines, Solids, Bars**—These objects are pixel oriented and they have no text size.

### 3.3.5.1 Selected Text

You normally want to treat words that are logically grouped together as a single text-object so they can be selected as one unit. Text objects, like other objects, must be selected before certain commands can act on them (e.g., copy, move, etc.). The following precautions must be observed when entering text to avoid splitting the text object:

- Characters in a given string must be added in a single Add Text command
- The characters must have the same literal behavior
- Characters must be touching horizontally
- Blank characters typed-in by striking the space bar are considered part of the text-object. Separate text-objects are created when blank areas are created with the cursor-movement keys. Figure 3-42 illustrates four or five text-objects.

### 3.3.5.2 Text Commands

Some text-related commands are explained in the General Purpose Commands section of this manual. They are Move/Copy Text, Select/Deselect Text, and Delete Text.

**Command:**    **ADD TEXT (ADD TEX) (A T)**

**Use**—This command permits you to enter text into the picture.

**Procedure**—Type ADD TEXT on the command line, then press the ENTER key.

The Picture Editor prompts: Enter Text. Enter the text by typing on the Engineering Keyboard. Text size is determined by the current text size. Observe the following rules

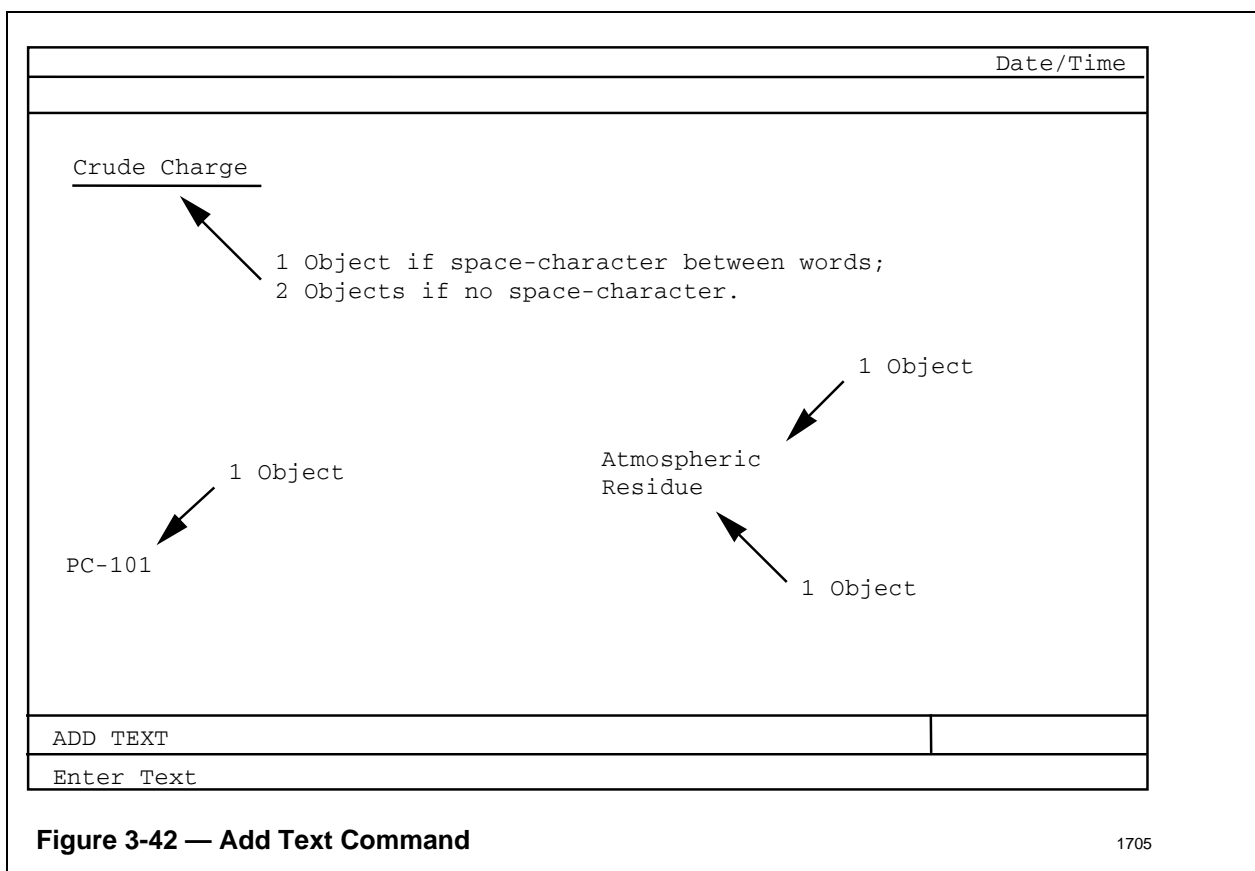
- Use the cursor controls to position the start of text.
- Use the Engineering-keyboard behavior keys to set literal behavior. You can change behavior as necessary, for example, different words are often entered in different colors.
- Characters must be adjacent and touching (typed spaces are considered blank characters) to be part of the same text-object.
- Multiple text strings can be entered with the same Add Text command; use the cursor-movement keys to separate text objects.

Do not use the insert character key (INS CHAR) with the Add Text command; refer to the Modify Text command if corrections are necessary.

When the desired text has been entered, press the ENTER key. Text is entered in the current priority (refer to the Set Priority command).

Figure 3-42 illustrates how multiple text objects were added with a single Add Text command.





**Command:** **ADD TEXTSIZE ssss (ADD SIZE) (A TS)**

where ssss is the choice of text size: large (**L**) or small (**S**).

**Use**—This command is used to change the size of existing textual objects. The objects must have been previously selected. When the Add Textsize command is executed, the size of all selected textual objects is changed to the specified text size.

**Procedure**—Type ADD TEXTSIZE ssss (or the equivalent synonyms) on the command line, then press the ENTER key.

Example: ADD TS L

When this command is executed, the size of all selected textual objects is changes to the specified text size. Also, refer to 3.3.5 Text and Text Commands, Special Considerations.

If the text size of an object is changed to large, the object is realigned to the nearest 8 x 16 pixel boundary.

Subpictures—if the object is a Subpicture and text size is changed to small, the change only occurs if the text size of the subpicture is small.

Variants—if the object is a Variant and the text size is changed to small, the change only occurs if the text size of all subpictures called by the variant is small.

An error results if none of the selected objects can legally be changed to the new text size. If the text size of only some of the selected objects can be legally changed, they are changed and the others are ignored.

**Command:**    **MODIFY TEXT (MOD TEX) (M T)**

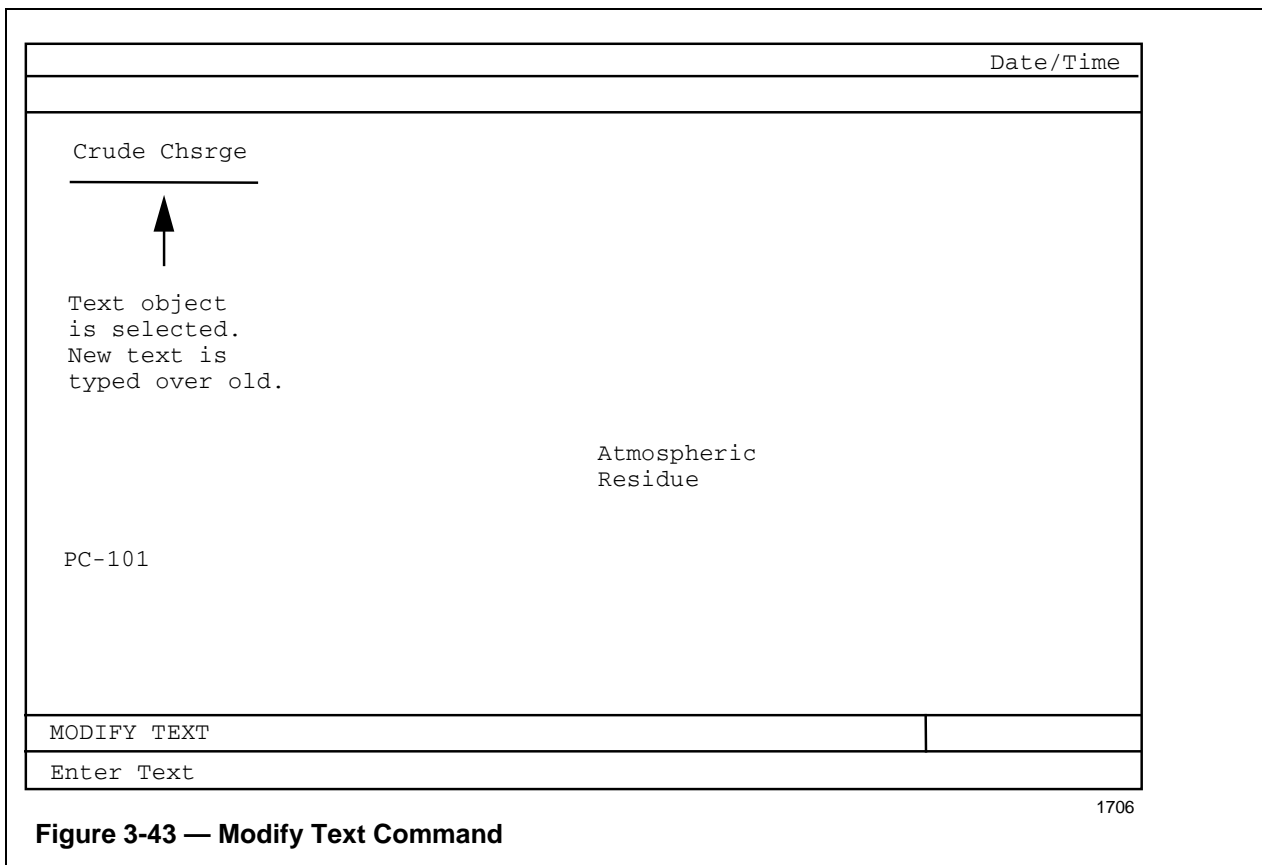
**Use**—This command is used to correct (or otherwise change) text in the picture. The text object must have been previously selected.

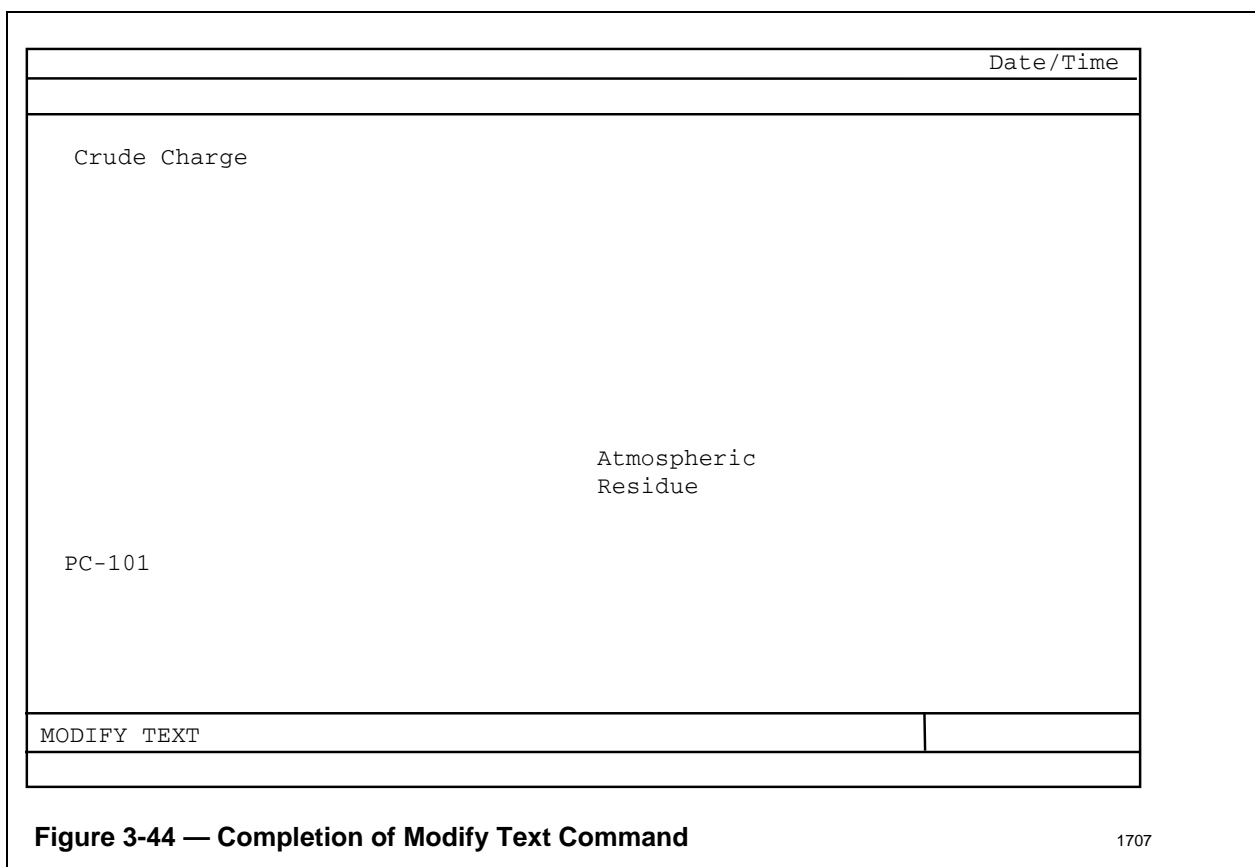
**Procedure**—Type MODIFY TEXT on the command line and press the ENTER key.

Correct any typing errors or change the text by typing over the selected text.  
Then press the ENTER key.

**CANCEL**—If the CANCEL key is pressed during or after text modification, the command is canceled and the text is not modified.

Figure 3-43 illustrates use of the Modify Text command. The text Crude Chsrge was selected. After invoking the Modify Text command, the new text Crude Charge was typed over the original text being replaced and the ENTER key was pressed. Figure 3-44 illustrates the result.





**Figure 3-44 — Completion of Modify Text Command**

1707

**Command:** SET TEXTSIZE sssss (SET SIZE sssss ) (S TS sssss )

where sssss is the choice of text size: **LARGE (L)** or **SMALL (S)**.

**Use**—This command is used to set the current text size (i.e., the next text that is entered). Large text cells are 8 pixels wide by 16 pixels high; small text cells are 8 pixels wide by 8 pixels high. The current text size is reflected on the second line from the top of the screen.

**Procedure**—Type SET TEXTSIZE sssss on the command line and press the ENTER key.

Example: SET TS SMALL

**CANCEL**—If the CANCEL key is pressed before pressing ENTER, the command is canceled and the current text size is not changed.

**Command:** DEFINE COMMENT

**Use**—This command allows you to document a picture or subpicture.

**Procedure**—Type DEFINE COMMENT on the command line and press the ENTER key.

A port opens up. Move the cursor into the port and type any comments. Multi page ports are allowed. If additional room is needed for comments, press the PAGE FWD key and continue entering text.

If the picture already contains a comment port, you can open it by invoking the Define Comment command. When the comment port is open, you can modify the contents. Figure 3-45 illustrates the Define Comment command.

Refer to the discussion of Screen Form Entry Aids following the Add Condition command in subsection 3.3.3 for an explanation of the function keys F2, F3, and F4.

**Cancel Key**—If you press the cancel key at this point, the command is cancelled.

To end the command, press the ENTER key.

**Subpictures**— when a subpicture that contains a comment port is added to a picture, the comment text is added to the main picture.

**Printing**—you can print comments with the print command.

The screenshot shows a window titled "Date/Time" at the top right. Inside the window, there is a large rectangular area containing the following text:

```
THIS SUBPICTURE DRAWS A PUMP AND HAS AN ORIGIN OF 0,0.  
  
THERE IS ONE PARAMETER (&P).  
  
IF THE PARAMETER VALUE EXCEEDS 100 THE PUMP TURNS RED AND  
BLINKS.
```

Below this text area, there is a line of text: "<PAGE FWD> <PAGE BACK> to MOVE. <F2> for TFE. <F3> to JUMP. <F4> to DELETE."

At the bottom of the window, there is a section with two fields:

DEFINE COMMENT	
Enter Text	

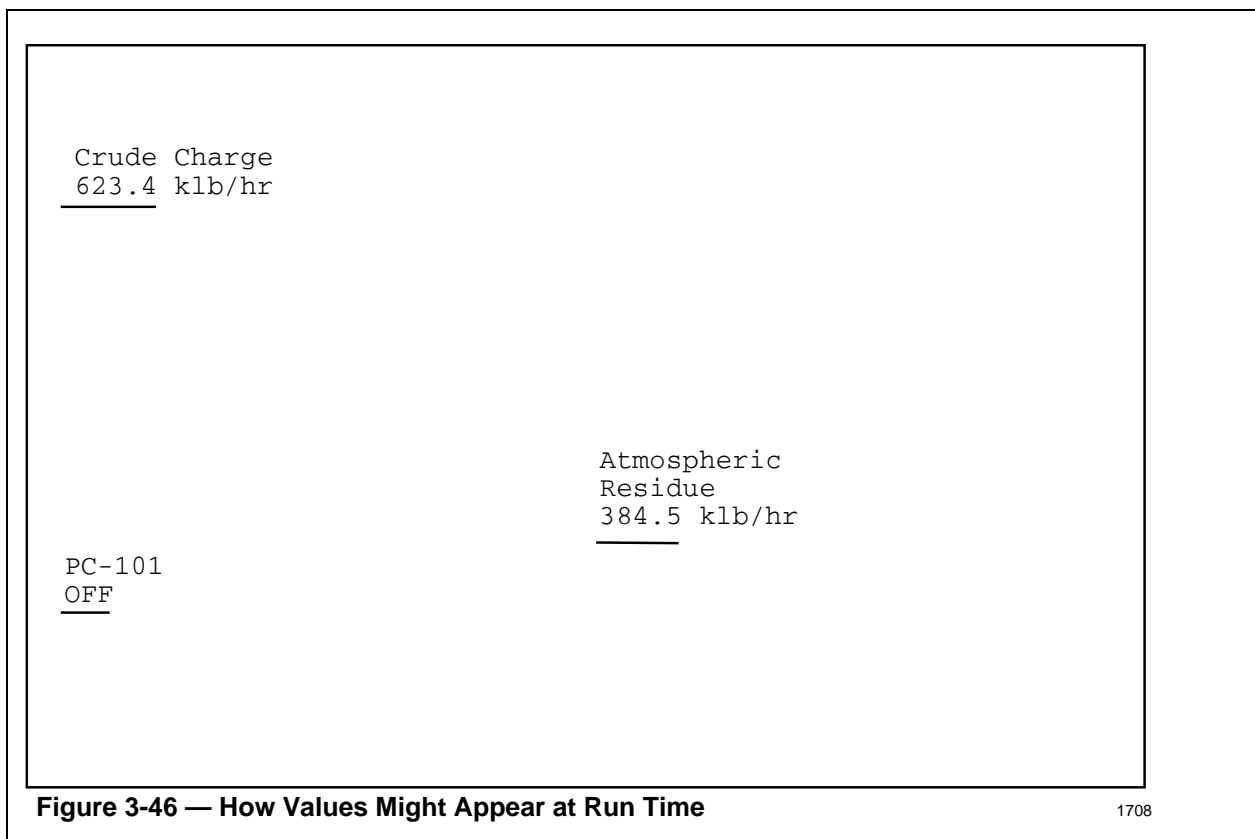
**Figure 3-45— Define Comment Command**

5521

### 3.3.6 Values

Values provide live numerical data (i.e., periodically updated values) or status conditions on the screen to represent measured quantities in the process. Text labels, such as pounds per hour, usually follow the value to provide its engineering units (e.g., 384.5 klb/hr where 384.5 is the current value). Figure 3-46 illustrates a picture with Values. The values have been underlined in this illustration.

Values have a text size that is determined by the current text size when the Value is built. Refer to subsection 3.3.5 for an explanation of text size.



### 3.3.6.1 Command: ADD VALUE (ADD VAL) (A V)

**Use**—The Add Value command is used to add value-objects (values) into the picture.

**Procedure**—Type ADD VALUE on the command line and press the ENTER key.

The Picture Editor prompts Enter Value Coordinates.

If the CANCEL or DEL key is pressed at this point, the command is canceled.

Move the cursor to each location where you want a value to appear and press the SELECT key. The Picture Editor marks each location with a cross (see Figure 3-47). Note the cursor position coordinates for each cross.

**DEL key**—If the DEL key is pressed while entering value locations, the last cross is deleted. If all locations have been deleted, pressing the DEL key again cancels the command.

When all value locations have been specified, press the ENTER key. The Picture Editor responds by presenting a form on the screen for the first Value location specified. Forms for each Value location are presented in turn. Fill in the forms as explained under Value Forms.

#### Value Forms

**Value Information Form**—Figure 3-48 illustrates the Value Information Form.

**Value at xxxx, yyyy**—The coordinates xxxx, yyyy identify the screen coordinates for the value.

**Expression**—The expression can be a simple entity, such as a point reference, or a combination of entities and literal values, for example, A100.PV + 2.0. The complete syntax for valid expressions is presented in Appendix C of this manual. It can also be a collector as described in Appendix H.

In Figure 3-48, the variable identifier A100.PV was typed into the form. At run time, the value displayed will be that for point A100.PV.

**Comments**—you can add comments to explain the expression. Comments must be enclosed in curly braces { }. Example: A100.PV + 2 {safety margin = 2}.

When the Value Information Form has been filled in, press the ENTER key.

**DEL key**—if the DEL key is pressed at this point, the value entry is deleted and the form for the next value is presented. If no other values need to be defined, the command is canceled.

Date/Time	
<div style="position: relative; width: 100%; height: 100%;"> <div style="position: absolute; top: 10%; left: 10%;">+</div> <div style="position: absolute; top: 30%; left: 40%;">+</div> <div style="position: absolute; top: 60%; left: 10%;">+</div> </div>	
ADD VALUE	
Enter Value Coordinates	

**Figure 3-47 — Entering Value Locations**

1709

Date/Time	
<div style="border: 1px solid black; padding: 10px; margin: 10px auto; width: 80%;"> <p>Value at xxxx, yyyy</p> <p>Expression <span style="border: 1px solid black; padding: 2px 20px;">A100.PV</span></p> </div>	
ADD VALUE	
Enter Value Information	

**Figure 3-48 — Entering the Expression**

1710

**Variable Type and Format Forms**—After the expression has been typed in and the ENTER key pressed, the Picture Editor checks for proper syntax of the expression and, if necessary, requests more information. There are several possible cases:

- For each entity name that is unknown to the system and for each expression that contains a parameter mark (&), the Picture Editor requests a definition of the variable type by presenting the form shown in Figure 3-49. The variable type for an entity is unknown to the system if the point has not yet been built or a parameter name appears in the expression. If this form is presented, type in the variable type and press the ENTER key (REAL was entered for this example). Refer to Appendix B for a list of name forms. If the expression contains a parameter mark (&), also refer to the section on Subpictures. After entry of the variable type, the Picture Editor prompts for the format as explained below.
- If the type of entity could be determined, or if you have entered the variable type as described above (and there are no syntax errors), the Picture Editor prompts ENTER THE FORMAT and presents the form shown in Figure 3-50. A new format can be typed over the default format on the screen, or the default format (shown as R-ZZZZ9.99 in the example)m, can be accepted. In either case, press the ENTER key to complete the format entry. Formats are described in Appendix A.

**DEL key**—If the DEL key is pressed before pressing the ENTER key, the form for this value is canceled and the form for the next value is presented. If all values have been defined, the Add Value command is canceled.

After entering all of the required information, the Picture Editor inserts the format into the Value location in the picture (see Figure 3-51). At run time, the expression or expressions are evaluated and the actual values appear.



Date/Time	
<div style="border: 1px solid black; padding: 10px; margin-bottom: 10px;"> <p>A100.PV</p> <p>Variable Type      <span style="border: 1px solid black; padding: 2px 20px;">REAL</span></p> </div>	
<div style="display: flex; justify-content: space-between;"> <span>ADD VALUE</span> <span></span> </div>	
Enter Value Type	

**Figure 3-49 — Entering Variable Type**

1711

Date/Time	
<div style="border: 1px solid black; padding: 10px; margin-bottom: 10px;"> <p>Value at xxxx, yyyy</p> <p>Format      <span style="border: 1px solid black; padding: 2px 20px;">R-ZZZZ9.99</span></p> </div>	
<div style="display: flex; justify-content: space-between;"> <span>ADD VALUE</span> <span></span> </div>	
Enter Value Format	

1712

**Figure 3-50 — Entering Format Information**

Date/Time	
RRRRRRRRR	
BBBBBB	
IIIIII	
ADD VALUE	

**Figure 3-51 — Completion of Add Value Command**

1713

### 3.3.6.2 Command: **MODIFY VALUE (MOD VAL) (M V)**

**Use**—This command allows you to change the Value Information Form that was filled out during a previous Add Value command.

The Value or Values to be modified must have been selected.

**Procedure**—Type **MODIFY Value** on the command line, then press the ENTER key. The information form for the first **selected** object appears on the screen. The form can be modified by typing over existing entries.

After modifying the form, press the ENTER key; to accept the form without change just press the ENTER key.

If more than one Value is **selected**, each form appears on the screen, in turn, and you can modify or accept it as described above. The command ends when all of the forms have been presented.

DEL key—If the DEL key is pressed while a form is present on the screen, the current form is deleted and the form for the next object is presented. If no more forms remain to be presented, the command is canceled.

### 3.3.6.3 Other Value Commands

Refer to the following commands in the General Purpose Commands section:

Copy	Move
Delete	Select
Deselect	

These are broad-based commands that affect Values and a number of other objects in a similar way.

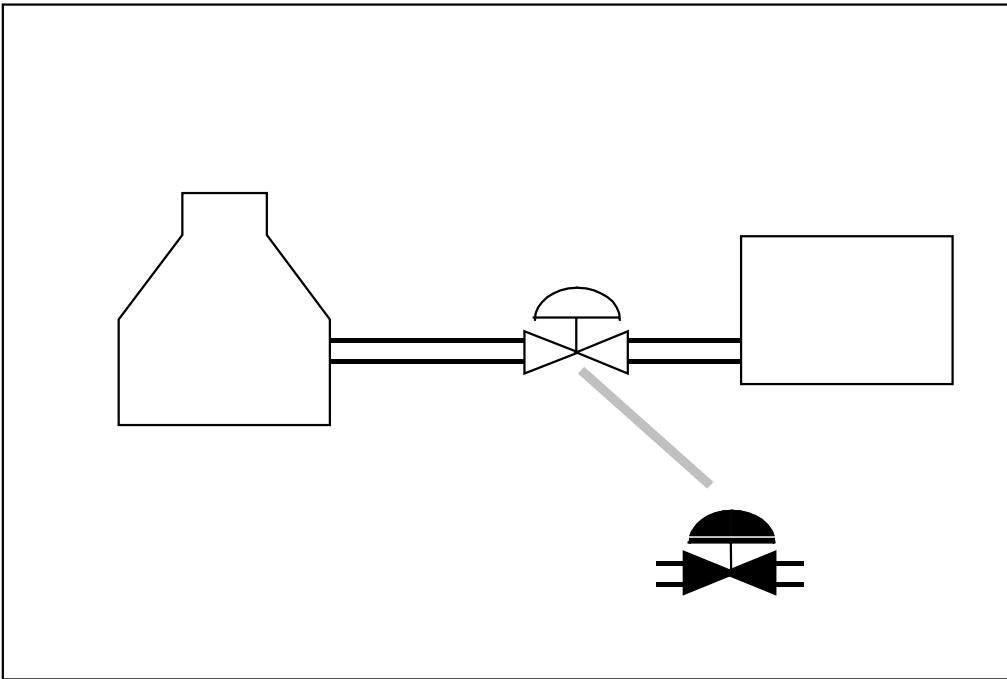
### 3.3.7 Variants

The valve in Figure 3-52 illustrates one way that a Variant can be used. The valve shape appears either solid or hollow to represent the valve as either open or closed. In this case, one subpicture or another is being switched into the picture, depending on evaluation of a Boolean expression. Subpictures are discussed in a separate section of this manual.

Variants are used to choose one of several objects for presentation in the picture. In R610 the Add Variant command is enhanced to accept a Value as one of its objects along with subpictures and text strings. The choices are between

- one of several subpictures
- one of several text strings
- one of several values (R610)
- a subpicture, text string or value (R610)

Variants have a text size that is determined by the current text size when the Variant is built. A Variant with its text size set to small cannot call a Subpicture that has its text size set to large. Refer to subsection 3.3.5 for a discussion of text size.



**Figure 3-52 — How a Variant Might Change at Run Time**

1714

Note that a subpicture called into the custom graphic by a Variant clears a rectangular area equal to or somewhat larger than the subpicture.

### 3.3.7.1 Command: **ADD VARIANT (ADD VAR)**

**Use**—The Add Variant command is used to add Variants into the picture. For Free Format Logs, see Section 2.4 in this manual.

**Procedure**—Type ADD VARIANT on the command line and press the ENTER key. The Picture Editor prompts:

Move the cursor to each location where you want a Variant to appear and press the SELECT key. If the touch-screen option is present, just touch the screen at the desired location. The Picture Editor marks each location with a cross (see Figure 3-53). Note the cursor-position coordinates for each cross.

**DEL KEY**—If the DEL key is pressed while entering value locations, the last cross is deleted. If all locations have been deleted, pressing the DEL key again cancels the command.

When all Variant locations have been specified, press the ENTER key. The Picture Editor responds by presenting a form on the screen for the first Variant location. Forms for each Variant location are presented in turn. Fill in the forms as explained under Variant Information Form.

**Variant Information Form**— Figure 3-54 — Entering Variant Information illustrates the Variant Information Form. Note that when a subpicture is called for, it must be preceded by the word SUBPICTURE (or SUB or S). If the subpicture file is not on the volume indicated by the default pathname, you can specify a full or partial pathname to the file. When a text string is called for, it must be enclosed in quote marks (example "BADVAL"). In R610 when a value is called for, it must be preceded by the word VALUE (or VAL or V).

**Variant at xxxx, yyyy**—The coordinates xxxx, yyyy identify the screen coordinates for the Variant.

**Subpicture, Text or Value (R610) for Bad Value**—Type in the subpicture name, text-string or value to be displayed in case evaluation of the expression gives an unreasonable value or if the value of any variable in the expression cannot be obtained. In Figure 3-54 the text string BADVAL was entered.

Note that the current text size determines the text size of the Variant and a Variant with its text size set to small cannot call in a Subpicture with a text size set to large. An attempt to do so causes an error and the Variant is not added into the picture.

**Variant Body**—Using Variant language, write a statement to describe the conditions that will cause each object to be displayed. Type the statement into the form and press the ENTER key. Variant language is described in Appendix E of this manual.

**DEL KEY**—If the DEL key is pressed instead of the ENTER key, the form is erased and the form for the next Variant is presented.

In this example, (at run time) if the magnitude of point A100.PV is equal to or greater than 50, the subpicture named CVALVE is displayed. If not, the subpicture named OVALVE is displayed. During edit time, the first condition is displayed. If a reasonable value cannot be obtained for A100.PV, the text string for Bad Value (BADVAL) is displayed. You can specify a pathname to call the subpicture from other media. For example, ELSE \$F1>LIB1>OVALVE might call the subpicture from your personal library of pre-built subpictures on a floppy disk. Once obtained, the subpicture is bound into the display.

**Variable Types and Formats**—If an entity name is unknown to the Picture Editor, it presents a form and prompts for the variable type.

If the Picture Editor encounters a subpicture that has a parameter, it presents a form and prompts for the variable type.

Date/Time	
<div style="position: relative; width: 100%; height: 100%;"> <div style="position: absolute; top: 10%; left: 10%;">+</div> <div style="position: absolute; top: 40%; left: 40%;">+</div> <div style="position: absolute; top: 70%; left: 10%;">+</div> </div>	
ADD VARIANT	
Enter Variant Information	

**Figure 3-53 — Entering Variant Locations**

1715

Date/Time	
<div style="border: 1px solid black; padding: 10px; margin: 10px;"> <p>Variant at xxxx, yyyy</p> <p>Subpic, Text, Or VAL for Bad Value <span style="border: 1px solid black; padding: 2px 20px;">"BADVAL"</span></p> <p>Variant Body</p> <div style="border: 1px solid black; padding: 10px; margin: 10px;"> <pre> IF A100.PV &gt; 50 THEN     "PUMP OFF"  ELSE IF A200.PV &gt; 50 THEN     SUB PRJR01  ELSE     VAL A200.PV </pre> </div> <p style="font-size: small; margin-top: 10px;">&lt;PAGE FWD&gt; &lt;PAGE BACK&gt; to MOVE. &lt;F2&gt; for TFE. &lt;F3&gt; to JUMP. &lt;F4&gt; to DELETE.</p> </div>	
ADD VARIANT	
Enter Variant Information	

1716

**Figure 3-54 — Entering Variant Information**

**Comments**—Comments are useful to explain your sequences. Comments must be enclosed in curly braces { }.

Example—

```
IF A100.PV > 25 {If flow is at high trip point}
THEN SUB HITRIP
ELSE IF A100.PV <10{if flow is at low trip point}
THEN {output a low flow message} "LOW FLOW"
```

Multiple page ports are allowed. If you need additional room to enter sequences or comments, press the PAGE FWD key and continue the entry. Do not allow identifiers or key words to be broken between two pages. For example the variable ID A1000.PV\_ARR(25) cannot be broken. Do not break up an IF, THEN statement. Press PAGE BACK or use the Jump function to display previous pages.

Refer to the discussion of Screen Form Entry Aids following the Add Condition command in subsection 3.3.3 for an explanation of the function keys F2, F3, and F4.

### 3.3.7.2 Command: MODIFY VARIANT (MOD VAR) (M VAR)

**Use**—This command allows you to change the Variant information form that was filled out during a previous Add Variant command.

The Variant or Variants to be modified must be selected.

**Procedure**—Type MODIFY VARIANT on the command line, then press the ENTER key. The information form for the first selected object appears on the screen. The form can be modified by typing over existing entries.

After modifying the form, press the ENTER key; to accept the form without change just press the ENTER key.

If more than one Variant is selected, each form appears on the screen, in turn, and you can modify or accept it as described above. The command ends when all of the forms have been presented.

**DEL key**—If the DEL key is pressed while a form is present on the screen, the current form is deleted and the form for the next object is presented. If no more forms remain to be presented, the command is canceled.

Note that a copy of each subpicture referenced by a Variant is bound into the picture at build time. If a referenced subpicture is later changed, every reference to that subpicture must be deleted from the picture in order to eliminate the copy stored with the picture. The subpicture reference(s) and images can then be added back.

### 3.3.7.3 Other Variant Commands

Refer to the following commands in the General Purpose Commands section:

Copy	Select	Move
Delete	Deselect	Scale (but not if a text string occurs for any Variant event)

These are broad-based commands that affect Variants and a number of other objects in a similar way.

### 3.3.8 Bar Charts

Bar charts are used to show relative magnitudes for various purposes, such as production reports, periodic use of materials, etc. A bar can also be used in process schematics to represent fluid level in a tank. Bars produced by the Picture Editor can be either horizontal or vertical, and either solid or hollow. At run time, the length of each bar is based on evaluation of an expression.

Honeywell manual *Picture Editor Form Instructions* (see References), provides additional information and illustrations of Bar Charts.

#### 3.3.8.1 Command: ADD BAR

**Use**—This command is used to add one or more bar charts to the picture.

**Procedure**—Type ADD BAR on the command line, then press the ENTER key.

After executing this command, the Picture Editor prompts Enter Bar Locations.

**DEL**—If the DEL or CANCEL key is pressed at this point, the command is canceled.

Bars are defined by using the cursor to specify points in opposite corners of a rectangle. The rectangle represents the width of the bar and the length of the bar when it has reached full scale. Move the cursor to the first position and press the SELECT key. If the touch screen option is present, just touch the screen at the desired location. A cross appears on the screen to mark this position. Move the cursor to the next position and press SELECT (or use the touch screen). As the second point is entered, the Picture Editor draws a bar that includes the two points (see Figures 3-55 and 3-56).



## NOTE

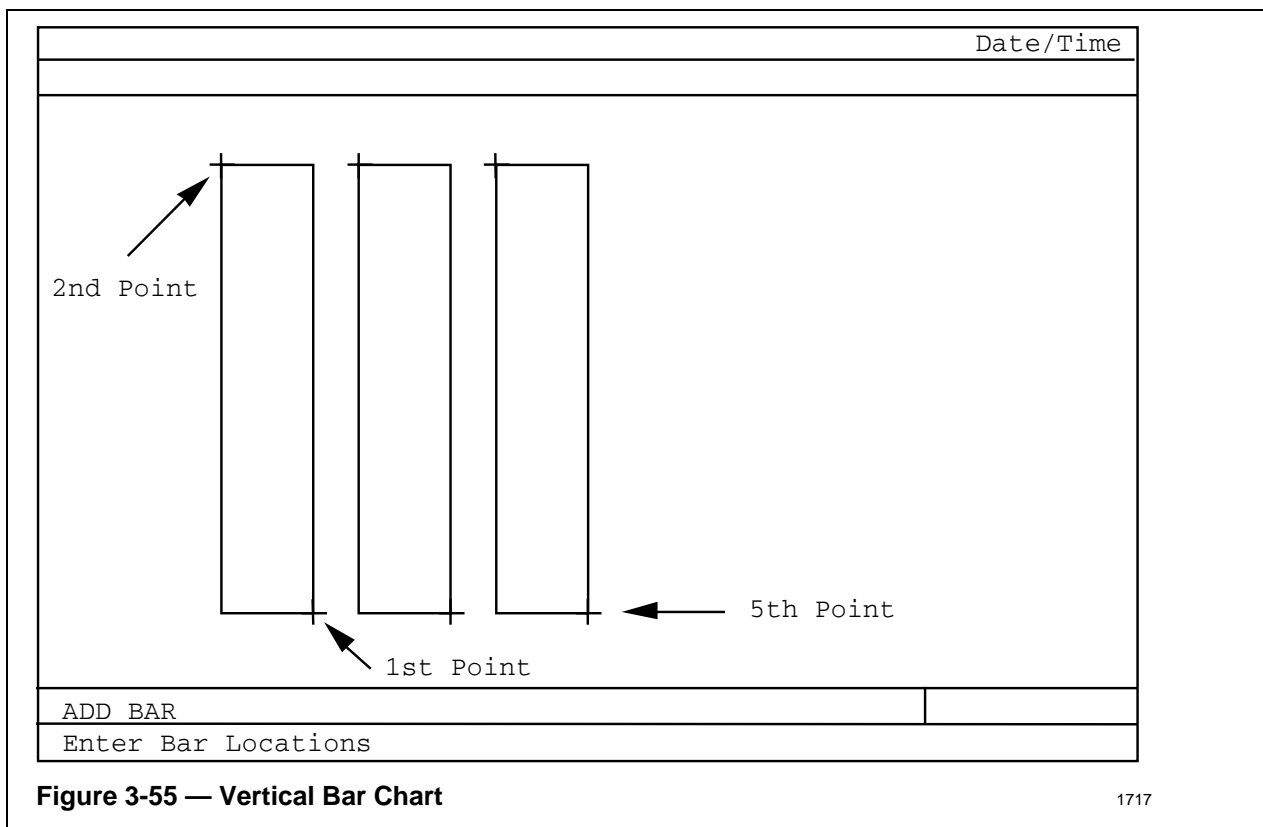
The bar chart that you define must be wider than 1 pixel to display properly at run time.

**DEL**—If the DEL key is pressed after specifying the first point of the pair, the cross is deleted and you can re-enter the first point. If the DEL key is pressed after specifying the second point of the pair, the box is erased and you can re-enter the second point. If no point locations remain when the DEL key is pressed, the command is canceled.

Additional bars can be drawn by continuing to specify point pairs.

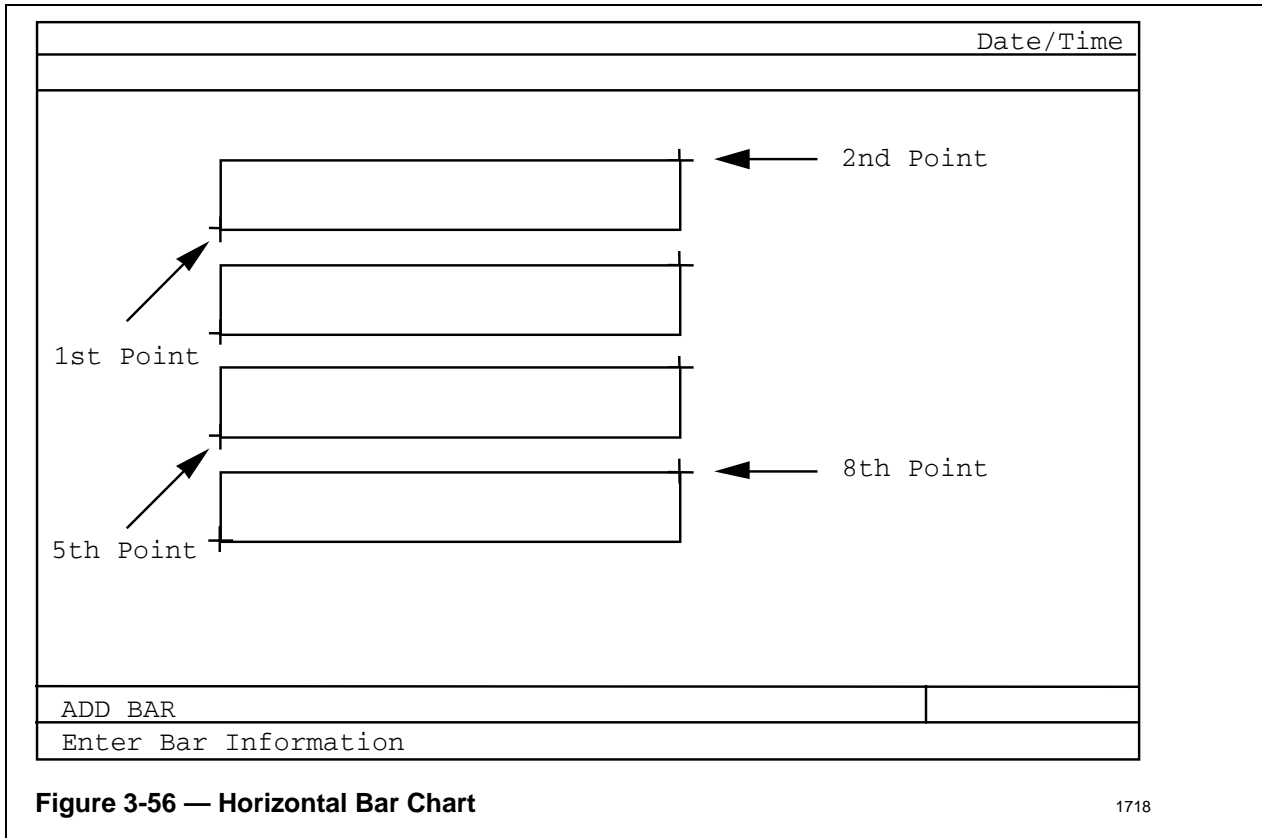
After all bar locations have been specified, press the ENTER key.

The Picture Editor presents a fill-in-the-blanks form on the screen and the editor prompts Enter Bar Information.



**Figure 3-55 — Vertical Bar Chart**

1717



**Bar Information Form**—Figure 3-57 illustrates the Bar Chart Information Form. Default values are shown in the information fields. You can change default values by typing over them or you can accept the default value in any field by leaving it unchanged. The form entries are explained below.

### **Bar at xxxx, yyyy**

The top line of the form gives coordinates for the bar being described.

<u>Field</u>	<u>Comments</u>
EXPRESSION	The expression (typically a point or parameter name) that is evaluated to determine the length or height of the bar. Legal expressions are defined in Appendix C.
SOLID/HOLLOW BAR	A solid bar is filled with color. A hollow bar is a rectangle with one end open.
VERTICAL/HORIZONTAL BAR	Direction in which the bars will be drawn.
LEFT/BOTTOM BOUND	Specifies the left boundary for horizontal bars or bottom boundary for vertical bars. The boundary is specified as an expression that yields a number used to scale the bar. Default value is 0.
RIGHT/TOP BOUND	Specifies the right boundary for horizontal bars or top boundary for vertical bars. The boundary is specified as an expression that yields a number used to scale the bar. The default value is 100.
ORIGIN	The origin value is used to specify the starting point for drawing the bar. The default value is 0.

When all of the Bar Information form has been completed, press the ENTER key. If additional bar locations were specified, a form for the next bar is presented.

**DEL key**—If the DEL key is pressed before the ENTER button, the form for the current bar is erased and the form for the next bar (if any) is presented. If no more bars remain to be defined, the command is canceled.

The Bar Chart is added to the picture in the current literal behavior and priority.

Date/Time	
Bar at xxxx, yyyy	
Expression	<input type="text"/>
Solid/Hollow Bar	<input type="text" value="Solid"/>
Vert/Horiz Bar	<input type="text" value="Vert"/>
Left/Bottom Bound	<input type="text" value="0"/>
Right/Top Bound	<input type="text" value="100"/>
Origin	<input type="text" value="0"/>
ADD BAR	
Enter Bar Information	

**Figure 3-57 — Bar Chart Information Form**

1719

### 3.3.8.2 Command: **MODIFY BAR (MOD BAR) (M BAR)**

**Use**—This command allows you to change the Bar coordinates and/or the Bar Chart information that was entered during a previous Add Bar command.

The Bar Chart or Charts to be modified must be selected.

**Procedure**—Type MODIFY BAR on the command line, then press the ENTER key. The Picture Editor responds by placing the cursor on the first coordinate of the first selected bar. You can move the cursor to a new coordinate and press the SELECT key, or accept the original coordinate by pressing the SELECT key.

The Picture Editor then moves the cursor to the next coordinate, which can be accepted or changed as before. When both coordinates have been entered, press the SELECT key.

Next, the Information Form for that bar appears on the screen. The form can be modified by typing over previous entries.

After modifying the form, press the ENTER key. To accept the form without change, just press the ENTER key.

If more than one Bar was selected, each coordinate pair and the related Information Form appear on the screen, in turn, and you can modify or accept them as described above. The command ends when all of the bars have been presented.

**DEL key**—If the DEL key is pressed while a form is present on the screen, the current form is erased and the form for the next object is presented. If no more forms remain to be presented, the command is canceled.

### 3.3.8.3 Other Bar Chart Commands

Refer to the following commands in the General Purpose Commands section:

Copy	Move	Select
Delete	Scale	Deselect

These are broad-based commands that affect Bar Charts and a number of other objects in a similar way.

### 3.3.9 Targets

Targets are rectangular areas in the picture that initiate some specified action when activated. Targets can be touch-activated on systems with touch screens or they can be activated on any system by placing the cursor in the Target rectangle and pressing the SELECT key. Typically, Targets are used to call up different displays and to communicate with the Display Database. The *Actors Manual* (see References) explains many of the ways that Targets can be used. Targets can be hollow (i.e., outlines) solid (i.e., color-filled), or invisible. An invisible Target can be superimposed over an object, such as a pump, that is controlled through the Target.

If two Targets overlap, only the last one created is executed when an overlapping Target area is selected. Overlapping Targets should be avoided.

Targets have a text size attribute that is set to large. If a Target is used in a Subpicture, the text size of the Subpicture is also set to large. Refer to section 3.3.5 for a discussion of text size.

#### 3.3.9.1 Command: ADD TARGET (ADD TARG) (A TAR)

**Use**—This command is used to add one or more Targets to the picture.

**Procedure**—Type ADD TARGET on the command line, then press the ENTER key.

After executing this command, the Picture Editor prompts: **Enter Target Coordinates.**

Targets are specified by using the cursor to identify points in opposite corners of a rectangle. Move the cursor to the first position and press the SELECT key. A cross appears on the screen to mark this position.

Move the cursor to the next position and press SELECT. As the second point is entered, the Picture Editor draws a rectangle that includes the two points (see Figure 3-58).

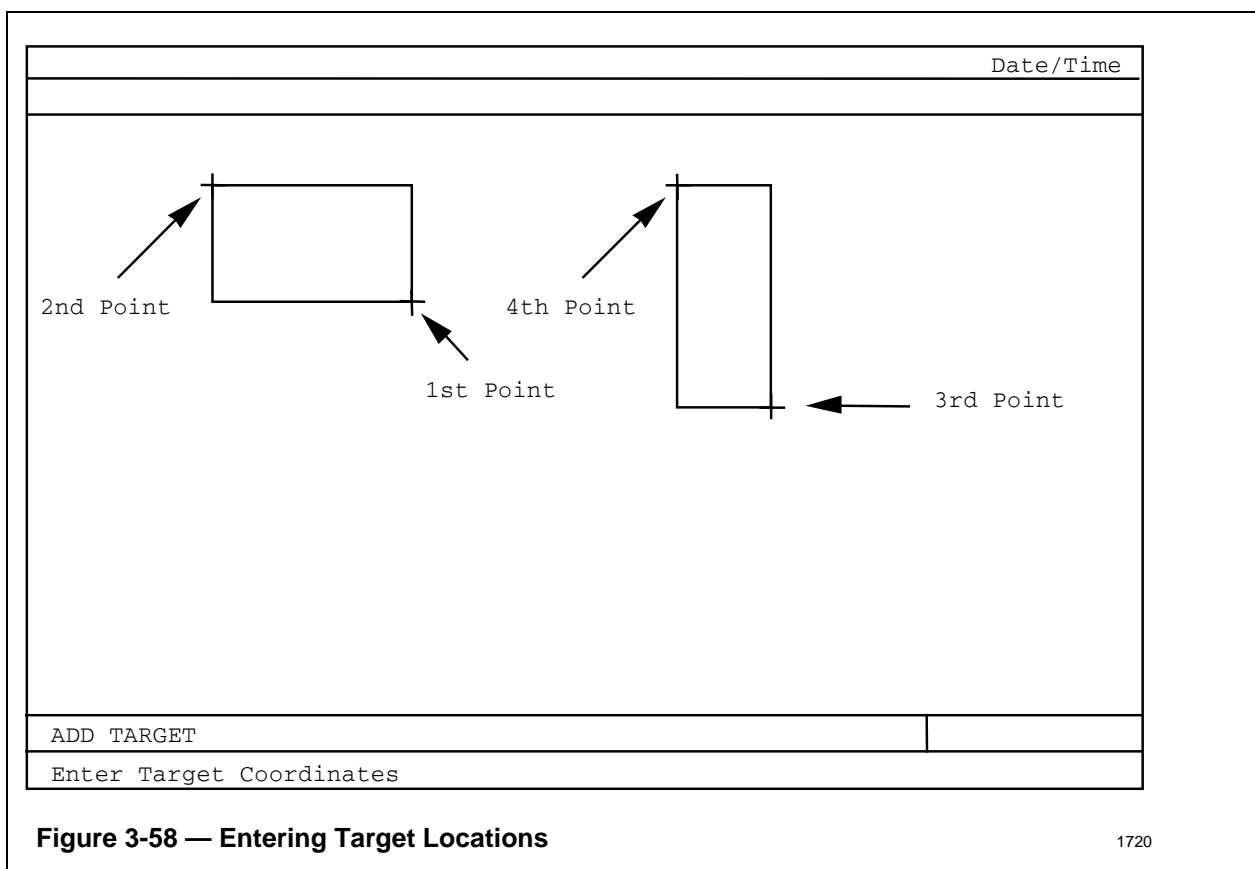
Target locations can be specified anywhere in the edit region, but the Picture Editor adjusts the edges of the rectangle to the nearest character-cell boundaries.

**DEL**—If the DEL key is pressed while specifying Target locations, the last location specified is deleted and you can re-enter that point. If no point locations remain when the DEL key is pressed, the command is canceled.

Additional Targets can be drawn by continuing to specify coordinate-pairs.

After all Target locations have been specified, press the ENTER key.

The Picture Editor presents a fill-in-the-blanks form on the screen and prompts **Enter Target Specifications.**



**Target Information Form**—Figure 3-59 illustrates the Target Information Form. Default values are shown in the information fields. You can change default values by typing over them or you can accept the default value in any field by leaving it unchanged. Abbreviated entry values are shown in parentheses. The form entries are explained below.

### Target at xxxx, yyyy

The top line of the form gives coordinates for the Target being described.

<u>Field</u>	<u>Comments</u>
SOLID/BOX/INVISIBLE	<p>Solid Targets appear as filled in rectangles at both edit and run time. Abbrev. forms: (SOL) (S)</p> <p>Box Targets appear as hollow rectangles at both edit and run time. Abbrev. form: (B)</p> <p>Invisible Targets appear as hollow rectangles at edit time, but are invisible at run time. This allows the Target to be placed over a picture object without obscuring the object. Abbrev. forms: (INVIS) (IN).</p>
ACTION	<p>Specifies what happens when the Target is activated. Refer to the discussion on action procedures in the <i>Actors Manual</i>.</p>

When the Target Information Form has been completed, press the ENTER key. If additional Target locations were specified, a form for the next Target is presented.

DEL key—If the DEL key is pressed before the ENTER button, the form for the current Target is erased and the form for the next Target (if any) is presented. If no more Targets remain to be defined, the command is canceled.

The Target is added to the picture in the current literal behavior and priority.

#### CAUTION

The total number of Targets that can be added to a display, including Targets in subpictures to that display (e.g., those called into the picture by Variants, Actors, etc.) is limited to 500. Exceeding this limit causes an error at compile time.



Date/Time	
<div style="border: 1px solid black; padding: 10px; margin: 10px;"> <p>Target at xxxx, yyyy</p> <p>Solid/Box/Invisible <span style="border: 1px solid black; padding: 2px 10px; margin-left: 20px;">Solid</span></p> <p>Action</p> <div style="border: 1px solid black; height: 100px; margin: 10px 0;"></div> <p style="font-size: small;">&lt;PAGE FWD&gt; &lt;PAGE BACK&gt; to MOVE. &lt;F2&gt; for TFE. &lt;F3&gt; to JUMP. &lt;F4&gt; to DELETE.</p> </div>	
<div style="border: 1px solid black; display: flex; justify-content: space-between; align-items: center;"> <span>ADD TARGET</span> <span></span> </div>	
<div style="border: 1px solid black; padding: 2px;">Enter Target Specifications</div>	

**Figure 3-59 — Target Information Form** 1721

**Comments**—Comments are useful to explain your sequences. You can enter comments before or after any action sequence (a semicolon is used to separate action sequences). Comments must be enclosed in curly braces { }.

Example—

```
IF(CMP_R(ENT01G.PV,GT,10.0)); {Set flag for variant action.}
CROSSCRN(01);MSGSUM;{ Write Message Summary on screen 1}
ELSE.....;
```

If you need additional room to enter comments or actors, press the PAGE FWD key and continue the entry. Do not allow an action sequence to be broken between two pages. For example the actor S\_ENT(ENT01G,R\_ENT(10,20,10,"ENTITY?",TRUE,2)); cannot be broken except immediately after the semicolon. Do not break up an IF, THEN statement. Press PAGE BACK to display previous pages.

Refer to the discussion of Screen Form Entry Aids following the Add Condition command in subsection 3.3.3 for an explanation of the function keys F2, F3, and F4.

### 3.3.9.2 Command: **MODIFY TARGET (MOD TARG) (M TAR)**

**Use**—This command allows you to change the Target Information Form that was filled out during a previous Add Target command.

The Target to be modified must have been selected.

**Procedure**—Type MODIFY TARGET on the command line, then press the ENTER key.

The Picture Editor places the cursor on the first coordinate of the selected Target. To accept the original coordinate press the SELECT key. To change the coordinate, move the cursor to the new location, then press the SELECT key.

Next, the Picture Editor moves the cursor to the second Target coordinate, which can be accepted or changed as before.

When both coordinates are specified, the information form for the associated Target will appear on the screen. The form can be modified by typing over previous entries.

After modifying the form, press the ENTER key. To accept the form without change just press the ENTER key.

If more than one Target was selected, each coordinate pair and its related form appear on the screen, in turn, and you can modify or accept them as described above. The command ends when all of the Targets have been presented.

**DEL key**—If the DEL key is pressed while a form is present on the screen, the current form is deleted and the form for the next object is presented. If no more forms remain to be presented, the command is canceled.

### 3.3.9.3 Other Target Commands

Refer to the following commands in the General Purpose Commands section:

Copy	Move
Delete	Select
Deselect	

These are broad-based commands that affect Targets and a number of other objects in a similar way.

When copying or moving a Target, take care that Actors associated with the Target are not moved to an illogical location. For example, if an Operator Input Actor creates a Text Input Port (TIP) at a location relative to the Target, moving the Target would also move the TIP (perhaps off the screen).

### 3.3.10 Subpictures

Subpictures are simply pictures created for convenient use when needed. Most of them are created with the Picture Editor as described in this section. They can be used repeatedly in the same, or other, drawings. Generally, it is useful to build a Subpicture library of shapes, such as pumps, valves, and similar objects because these objects are often needed in process schematics.

Subpictures can have different levels of complexity and versatility. The simplest of Subpictures contains no parameters and its behavior is not very flexible. In the discussions that follow, many of the Picture Editor commands are mentioned. You should refer to the complete description of that command if you are unfamiliar with what it does and how it is used.

Subpictures have an attribute called text size that is assigned when the Subpicture is created. If any object in the Subpicture has a text size of large, the Subpicture's text size is set to large; otherwise, it is set to small. For example, if a Subpicture contains a Target, the text size of the Subpicture is large. Values, Variants, and Text can have a text size of either large or small. Lines, solids, and Bar Charts do not have a text size. Note that a Variant with its text size set to small cannot call a Subpicture with its text size set to large. Text size also determines the exact point where a Subpicture can be added, moved, or copied in the picture (refer to the Move/Copy commands for more information and refer to section 3.3.5 for a discussion of text size).

#### 3.3.10.1 Creating The Subpicture

The picture is built by using the Picture Editor commands explained throughout this manual. Often, objects are drawn oversized and then scaled down to get well-defined shapes (refer to the Scale command). Rounded objects, for example, are drawn as a series of short straight lines and when reduced, the surface becomes a fairly smooth curve.

As an example of Subpicture building, a picture of a pump could be drawn by using the Add Line or Add Solid command and then reduced by using the Scale command (refer to the Graphic commands section of this manual).

Once the Subpicture is built to the correct specifications, an origin is added on or near the object (see the Set Origin command). The origin appears as a boxed cross on the nearest character boundary and is used as a reference point when adding the Subpicture to other displays. At this point, you should store the Subpicture (refer to the Write and Compile commands).

If several graphic displays are to be built and they contain similar objects, but orientation and size of the objects differ, another approach can save time later during the display-building sessions. The object is drawn and stored full size. It is then read back and modified for each different group of uses. For example, suppose that in order to construct a series of custom graphic displays, many Subpictures of pumps are needed but in four different sizes, and some of them face left while others face right. The main pump image is read back and scaled to the proper size or direction to meet one need. The origin is set and the altered picture is stored under another name. This procedure is repeated until Subpictures exist to fit all of the situations.

### 3.3.10.2 Using the Subpicture

As a custom graphic display is built, the Subpicture or Subpictures are placed into the picture by using the Add Subpicture command. This command allows you to specify where the Subpicture origin (and therefore the Subpicture) will appear. The Add Subpicture command is explained elsewhere in this section of the manual.

Subpictures can also be called into a display with Variants. Refer to the Add Variant command description in this manual.

Note that a Subpicture overlays the main schematic if it was added with the Add Subpicture command. A Subpicture called into a schematic by a Variant clears a rectangular section of the schematic equal to or somewhat larger than the Subpicture.

Also note that when a Subpicture is added into a picture, a copy is stored with that picture (actually in the picture's source file). If you decide to change the Subpicture, you must delete every use of it from that picture and from any others where you want the change to appear. Don't forget to delete references to that Subpicture from Variant statements too. Then call up the Subpicture file, make the necessary changes and add it back to the picture.

### 3.3.10.3 Subpictures with Parameters

More versatile Subpictures can be created by using user-visible parameters. User-visible parameters are displayed to the user when a subpicture is added to a picture and a prompt instructs the user to enter data so that actual values can be substituted for the & values.

Formal parameters are values, which after substitution has taken place are the actual values passed to the subpicture.

Prior to Release 510, the maximum number of parameters per Subpicture was 16 formal and 16 user-visible. For Release 510 and later systems, the maximum number of parameters per Subpicture is 64 formal and 64 user-visible, with 80 maximum of both types.

#### CAUTION

Subpictures with more than 16 parameters built in Release 510 can not be read by the Picture Editor in earlier software. The error message: `Illegal Record` results. The R510 object file may execute in previous releases but you can not modify the source file.

Before adding more than 16 parameters to an existing R500 Subpicture, you should save copies of the picture and Subpicture source/object (.DS/.DO) files in case it is necessary to go back to the earlier release.

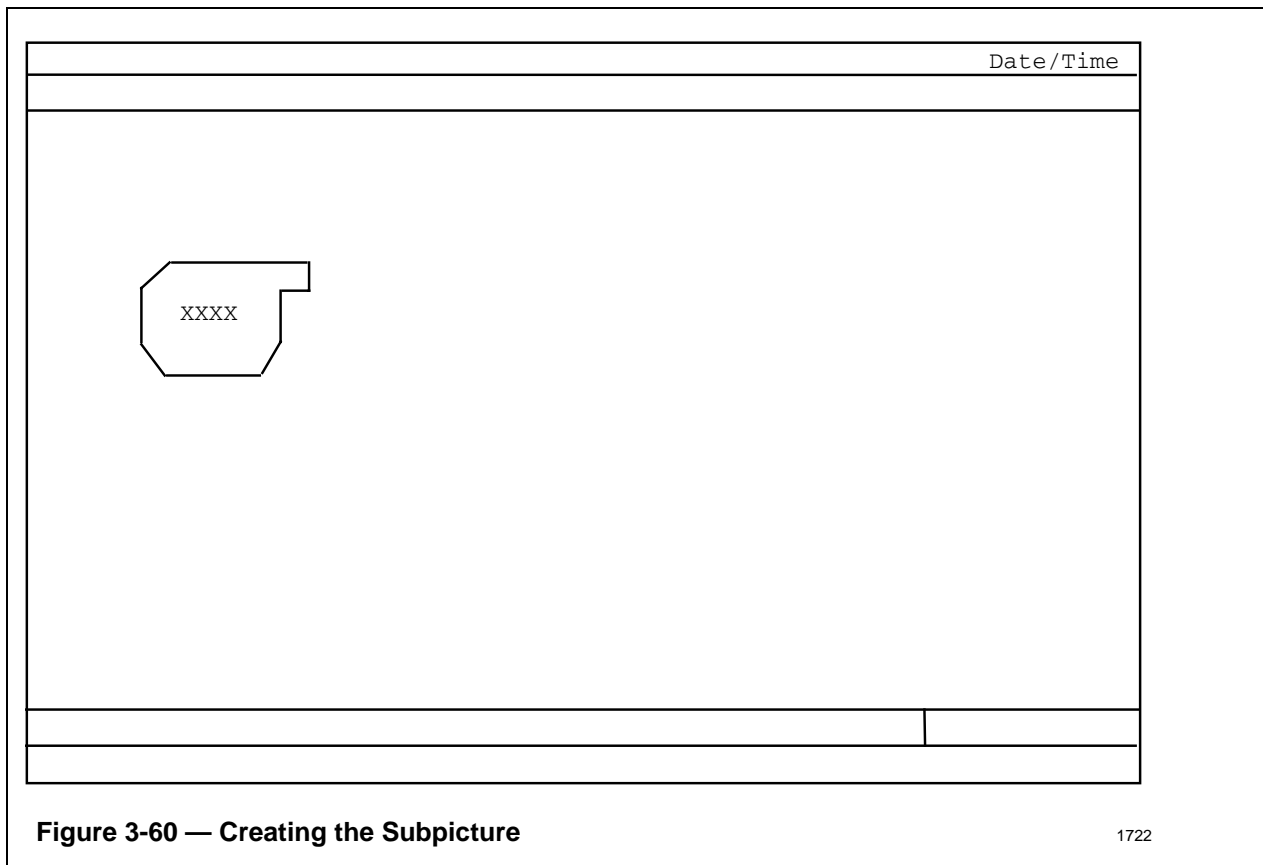
If you delete/modify more than one parameter in a subpicture, read the Caution statement in subsection 4.1.1.2 of the *Picture Editor Data Entry manual*.

User-visible parameters are denoted by preceding an alphanumeric string with an ampersand (&) character; for example, **&A**. When a Subpicture with a user-visible parameter is added to the display, specified data is substituted for the parameter(s).

The following examples illustrate how user-visible parameters are used with Values and Conditional behavior.

Suppose that several custom graphic displays are to be built and you want each to contain the image of a pump. We want the image of the pump to show the internal pressure of an actual pump. This can be done by using a Value. The Value is represented by XXXX in Figure 3-60. We also want to use a different expression to determine the Value in each case where the Subpicture is used.

The pump image is created by using the Add Line command.



The Add Value command is invoked and a Value location is specified on the pump. The Picture Editor then presents the Value Information Form. The expression is specified as a parameter by using the symbol & and an identifier (for example, in Figure 3-61 the parameter &A.PRESS was used). A parameter can be specified for any alphanumeric text string that is valid in an expression.

The figure shows a screenshot of the 'Value Information Form' within a software interface. At the top right, there is a 'Date/Time' field. The main area of the form contains a large rectangular box. Inside this box, the text 'Value at xxxx, yyyy' is displayed. Below this, the label 'Expression' is followed by a text input field containing the text '&A.PRESS'. At the bottom of the form, there are two buttons: 'ADD VALUE' on the left and 'Enter Value Information' on the right.

Because the Picture Editor cannot determine the variable type for this parameter, it presents the form shown in Figure 3-62 and prompts for the variable type. The type is specified by typing it into the form and pressing the ENTER key.

The Picture Editor then prompts for the format. When all specifications have been entered, the Value is added to the pump.

Date/Time	
<div><div>&amp;A.PRESS</div><div>Variable Type<div></div></div></div>	
ADD VALUE	
Enter Value Type	

**Figure 3-62 — Entering Variable Type**

1724

At this point the pump image can be stored. Invoking the Write command causes the Picture Editor to interrogate the picture and determine that it contains a parameter. The Picture Editor then presents a form that requests a string. The string, as typed in on the form, is presented as a prompter when the Subpicture is added to a graphic display. At that time, you can specify an exact expression that is to be evaluated for the Value. Each time the Subpicture is needed, a different expression can be typed in, for example, A100, B200, etc. Therefore, each Subpicture can have unique specifications for each occurrence (A100.PRESS, B200.PRESS, etc., for the example expressions). Note that a parameter can also be specified for another parameter.

Figure 3-63 illustrates the form that is presented during the Write command. In this example, the prompter string used is Point ID? and the file name chosen is HPUMP.

Date/Time

Subpicture HPUMP

Prompt for &A

Point ID?

WRITE PUMP

Enter Prompt Question

1725

**Figure 3-63 — Entering Prompter Question**



#### 3.3.10.4 Conditional Behavior

Unless otherwise designed, the Subpicture keeps the same literal behavior as its source and this cannot be changed when the Subpicture is incorporated into a display. One way to obtain different behaviors for different occurrences is to specify conditional behavior as a parameter. The procedure is explained below.

Suppose the pump picture has been drawn as shown in Figure 3-60 (with or without the Value). First, the pump is selected; then the Add Condition command is invoked. The Picture Editor presents a Conditional Behavior Form. The form is filled in to call for whatever behavior is desired; however, when writing the conditional behavior statements, a parameter is specified for the expression. Figure 3-64 illustrates a typical Conditional Behavior Form.

If the picture is stored at this point, the Picture Editor requests a prompt string, as described in the preceding discussion where parameters were used to specify the Value expression.

Date/Time	
Conditional Behavior	
Behavior For Bad Value	BLACK NO BLINK NO REVERSE FULL
Initial Behavior	CYAN NO BLINK NO REVERSE FULL
Condition	<pre>IF &amp;A.PV = 10 THEN SET     RED BLINK HALF ELSE SET     GREEN FULL N B N R</pre>
ADD CONDITION	
Enter Condition Information	

**Figure 3-64 — Conditional Behavior Using a Parameter** 1726

### 3.3.10.5 Inherited Behavior

Inherited Behavior is another way to give the Subpicture distinct behavior (other than with a parameter). When a Subpicture with Inherited Behavior is added to a display, it assumes the current literal behavior for that display. It can then be selected and given a distinct fixed or conditional behavior by using the Add Behavior or Add Condition commands, respectively. If a Subpicture with Inherited Behavior is used several times in the same custom graphic display, each use can have distinct fixed or conditional behavior.

The following example shows how Inherited Behavior is assigned.

Assume the Pump Subpicture is drawn (with or without the Value) as shown in Figure 3-60. The object that is to have Inherited Behavior (the pump in this case) must be selected.

Next, the Add Inherit command is executed. All selected objects in the display acquire the characteristic of Inherited Behavior. The Add Inherit command is described elsewhere in this section.

The picture is then stored for use as a Subpicture.

### 3.3.10.6 Subpicture Commands

**Command:**    **ADD SUBPICTURE nnnn**

Abbreviated forms:    **ADD SUBPIC nnnn**  
                              **ADD SUB nnnn**  
                              **A S nnnn**

Where nnnn is the name of the Subpicture.

**Use**—This command adds a previously created Subpicture to the current picture. If the Subpicture is not on the volume indicated by the default pathname you can specify a full or partial pathname (refer to the Set Pathname command in the General Purpose Commands section of this manual).

**Procedure**—Type ADD SUBPICTURE nnnn on the command line and press the ENTER key.

The Picture Editor responds:

Move the cursor to the first location where you want the Subpicture to appear and press the SELECT key. If the touch-screen option is present, just touch the screen at the desired location. The position is marked with a cross and corresponds to the origin position in the Subpicture.

You can continue to specify additional locations in the same way. When all locations have been entered, press the ENTER key (see Figure 3-65).

DEL—If the DEL key is pressed while specifying locations, the last coordinate selected is deleted. If no coordinates remain, the command is canceled.

If the Subpicture contains any parameters, the Picture Editor presents a Subpicture Information Form.

Subpicture Information Form—The first line on the form identifies the Subpicture and its coordinates.

The next line (or lines) prompts for parameters to the Subpicture. The word-string Point ID? in Figure 3-66 is an example of a prompter question that is stored with the Subpicture (refer to the discussion on using parameters in Subpictures).

Type the parameters into the form (A100 was used in this example, see Figure 3-66) and press the ENTER key.

DEL—If the DEL key is pressed at this point, the form is deleted and the form for the next Subpicture is presented. If no other Subpictures remain to be defined, the Picture Editor awaits the next command.

If any variables that are specified as parameters are unknown to the system, the Picture Editor prompts for the type of entity and presents a form where you must specify the variable type.

After all parameters are specified, the Picture Editor redraws the picture with the Subpicture image(s) visible. Subpictures are added so that the specified coordinate and the Subpicture origin coincide (see Figure 3-67).

Note that Subpicture information form entries can be changed later with the Modify Subpicture command.

Pathname	Date/Time
<div style="position: relative; width: 100%; height: 100%;"> <div style="position: absolute; top: 10%; left: 10%;">+</div> <div style="position: absolute; top: 30%; left: 30%;">+</div> </div>	
<div style="display: flex; justify-content: space-between;"> <span>ADD SUBPICTURE HPUMP</span> <input style="width: 20px;" type="text"/> </div>	
Enter Subpicture Coordinates	

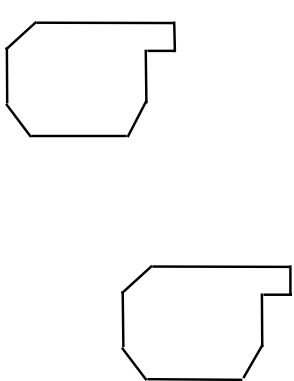
**Figure 3-65 — Specifying Subpicture Locations**

1727

Date/Time	
<div style="border: 1px solid black; padding: 10px; margin-bottom: 10px;"> <p>Subpicture HPUMP At xxxx,yyyy</p> <p>Point ID? <input style="width: 150px;" type="text" value="A100"/></p> </div>	
<div style="display: flex; justify-content: space-between;"> <span>ADD SUBPICTURE HPUMP</span> <input style="width: 20px;" type="text"/> </div>	
Enter Subpicture Information	

**Figure 3-66 — Subpicture Information Form**

1728

Pathname	Date/Time
	
<div style="display: flex; justify-content: space-between; align-items: center;"> <span>ADD SUBPICTURE HPUMP</span> <span></span> </div>	

**Figure 3-67 — Completion of Add Subpicture Command** 1729

**Command:**    **MODIFY SUBPICTURE (MOD SUBPIC) (M SUB) (M S)**

**Use**—This command allows you to change parameters on the Subpicture Information Form (if one exists) that was filled out when the Subpicture was added to the picture. Note that the contents of the Subpicture cannot be modified with this command.

The Subpicture images must first be selected.

**Procedure**—Type MODIFY SUBPICTURE on the command line, then press the ENTER key. The information form for the selected Subpicture image appears on the screen. The form can be modified by typing over previous entries.

After modifying the form, press the ENTER key. To accept the form without change just press the ENTER key. If more than one Subpicture image is selected, each form appears on the screen in turn and you can modify or accept it as described above. The command ends when all of the forms have been presented.

If the subpicture has no user visible parameters, the following message appears:

SUBPICTURE NNN AT X Y  
NO PARAMETERS TO THIS SUBPICTURE

NNN is the name of the subpicture and X Y are the coordinates of the subpicture's origin. Press ENTER to continue or CANCEL to abort the modify command.

**DEL key**—If the DEL key is pressed while a form is present on the screen, the current form is deleted and the form for the next subpicture is presented. If no more forms remain to be presented, the command is canceled.

**Command:**    **SET ORIGIN (SET ORIG) (S O)**

**Use**—The Set Origin command is used to specify an origin for a Subpicture. The Origin is used to position a Subpicture.

**Procedure**—Type SET ORIGIN on the command line and press the ENTER key.

The Picture Editor prompts Select New Origin.

**DEL key**—If the DEL key is pressed at this point, the Origin is not changed and the command is canceled.

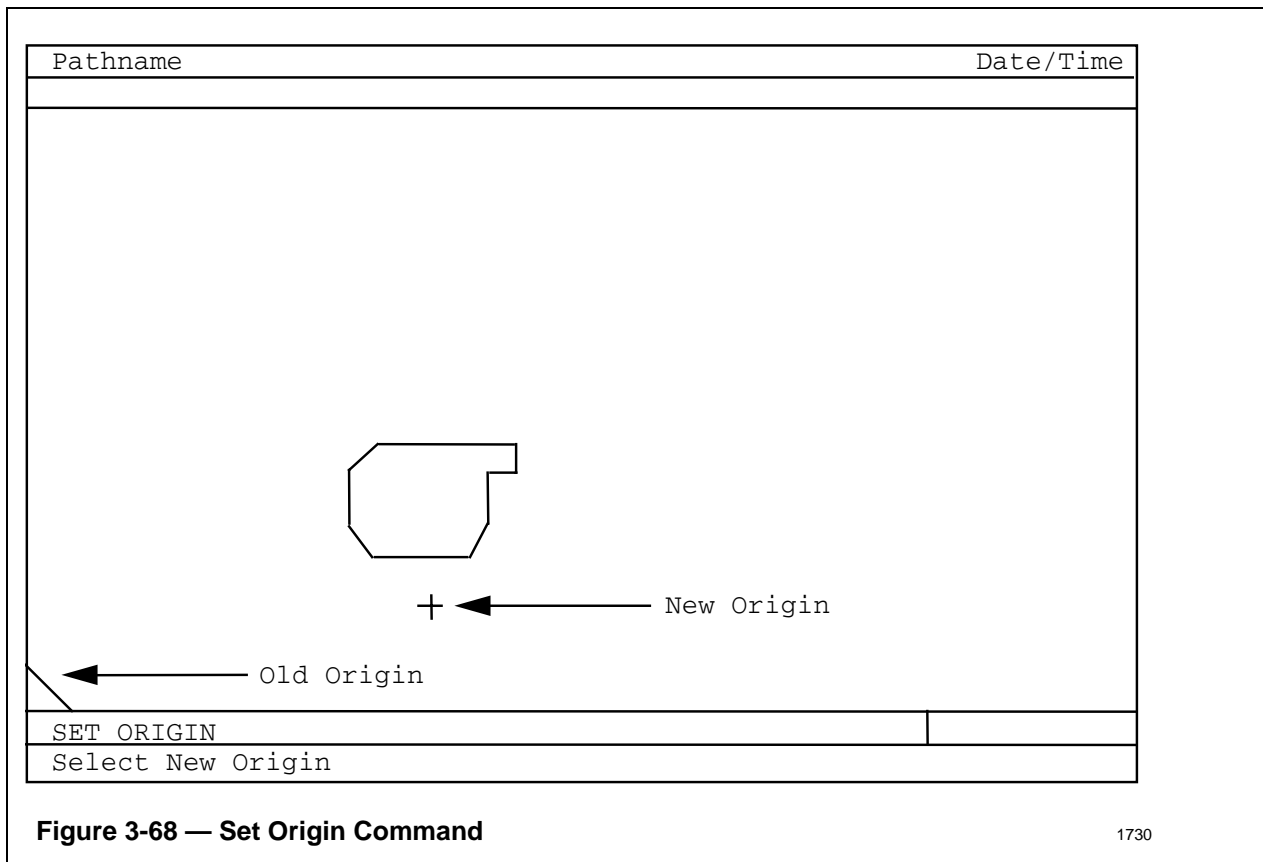
Move the cursor to the new location for the Origin and press the SELECT key. If the touch-screen option is present, just touch the screen at the desired location.

The new Origin is marked with a cross (see Figure 3-68). If the new Origin location is not on a character boundary, the Picture Editor adjusts it to the nearest character boundary (the line/column format is explained at the beginning of the Commands section of this manual).

**DEL key**—If the DEL key is pressed at this point, the cross that marks the new Origin is deleted and you can re-enter the new Origin location.

Press the ENTER key to accept the new Origin location.

**CANCEL**—If the CANCEL key is pressed at this point, the command is canceled and the old Origin is restored.



**Figure 3-68 — Set Origin Command**

1730

**Command:** **ADD INHERIT (ADD INH) (ADD IN) (A I)**

**Use**—This command is used to change the behavior of an object to Inherited Behavior. The objects must first be selected.

**Procedure**—Type ADD INHERIT on the command line and press the ENTER key.

All objects that are selected will assume the characteristic of Inherited Behavior.

Refer to the discussion on Inherited Behavior earlier in this section for more information.

**Command:** **SELECT INHERIT (SEL INH) (SEL IN) (SEL I)**

**Use**—This command is used to select and identify objects with Inherited Behavior.

**Procedure**—Type SELECT INHERIT on the command line and press the ENTER key.

The Picture Editor prompts for locations and these are entered as described for the Select command.

After entering the select locations, press the ENTER key.

All objects that are within or that intersect the select box and that have Inherited Behavior are selected.

Other Subpicture Commands—Refer to the following commands in the General Purpose Commands Section:

Copy	Move	Select
Delete	Scale	Deselect

These are broad-based commands that affect Subpictures and a number of other objects in a similar way.

**Command:** **REPLACE SUBPICTURE** (REPL SUBPIC) (REP SUB) (RE SUB) (RE S)  
(NAME PARTIAL\_PATHNAME [ORIGINAL (ORIG) (OR) (O)])

**Use**—This command allows you to replace the Subpictures in an existing custom graphic display (picture). Typically it is used to modify Subpictures that exist within a picture.

**R600 Enhancement**—With R600, the Picture Editor Replace Subpicture command has been improved. Prior to this function, all instances of a subpicture were replaced regardless of whether or not any or all of the subpictures were selected. With the enhancements made by this new function, the engineer can select a specific subset of subpictures and replace only the selected subpictures. If only a subset of subpicture A is selected, the subpicture to replace subpicture A must have a different name from subpicture A.

**Procedure**—Build a new Subpicture and save it with a Write command (the new Subpicture can be a change to the source file of the Subpicture you want to replace).

Read the desired custom graphic display onto the screen. On the command line, type REPLACE SUBPICTURE (or a synonym), the file name of the existing Subpicture, and the pathname to the new Subpicture. Then, press the ENTER key. The new Subpicture replaces every occurrence of the specified Subpicture in the custom graphic. Origins of the new Subpictures are aligned the same as those of the Subpictures that are replaced.

**Option**—If you want the new Subpicture scaled to the same proportions as the existing Subpicture(s), include the word ORIGINAL (or a synonym) at the end of the command.

NAME is the file name of the Subpicture being replaced.

PARTIAL\_PATHNAME is the pathname of the Subpicture that will replace the original Subpicture(s). It can take any of the following forms—

- filename
- volume>filename
- device>volume>filename

The new Subpicture's filename can be the same as the Subpicture it replaces, but it must be different from any other Subpicture in the custom graphic. Note that a longer new Subpicture filename could cause an overflow of statements in the Value/Variant's input port unless sufficient room exists to accommodate the extra characters. To avoid this possibility, keep the new Subpicture filename equal to or shorter than the original name.



**Examples**—A schematic is being edited. It contains the Subpictures PUMP1 and MYCIRCLE. The following commands are issued—

**REPL SUB PUMP1 PUMP2**

Every occurrence of the Subpicture PUMP1 is replaced by Subpicture PUMP2.

**RE SUBPIC MYCIRCLE CIRCLE OR**

Every occurrence of Subpicture MYCIRCLE is replaced by the Application Subpicture CIRCLE. CIRCLE is scaled to the same size as the Subpicture MYCIRCLE.

**Parameters**—If the new Subpicture contains parameters, a form is presented when the new Subpicture is read in. If identical parameters are used in both the original and the new Subpicture, the original parameters are presented as defaults and you can edit them or press ENTER to accept them. If the new Subpicture contains parameters that were not in the original Subpicture, a blank screen form is presented for your entries. A separate screen form is presented for each occurrence of the new Subpicture in the custom graphic display.

**CANCEL**—If the CANCEL key is pressed during execution or immediately after the REPLACE command is executed, (i.e., before executing any other command) the custom graphic display is restored to its original state.

**(R600)— Command: SET COLLECTINH <FAST/MAIN/SLOW/SUBPICTURE>**

**Use**—This R600 command allows the “set collection” properties (collection rate and group ID) of variables in objects or subpictures to be inherited by the destination picture when a Picture Editor command is executed (add/replace subpicture, paste, etc.).

**Procedure**— Allowable abbreviations for the command are:

<b>COLLECTINH:</b>	<b>COLLI, COLI, CI</b>
<b>FAST:</b>	<b>F</b>
<b>MAIN:</b>	<b>MN, M</b>
<b>SLOW:</b>	<b>SLO, SL</b>
<b>SUBPICTURE:</b>	<b>SUBPIC, SUB, S</b>

Where:

- SET COLLECTINH is the Picture Editor command.
- FAST indicates to use the faster of the collection rates as precedence.
- MAIN indicates to use the destination (main) picture’s data as precedence. This is the DEFAULT value and will give the same functionality that exists before R600.
- SLOW indicates to use the slower of the collection rates as precedence.
- SUBPICTURE indicates to use the subpicture’s data as precedence.

This “collection inheritance” priority is effective on a “per schematic” basis (until another “SET COLLECTINH” command is issued or until a new picture is read in). The “current” inherit priority is saved with the picture during the WRITE or COMPILE command. The READ command reads the priority and sets it as the “current” inherit priority. When the Picture Editor is first invoked, or when a picture that does not yet have a “current” inherit priority is read in, the inherit priority is defaulted to MAIN.

Both SET COLLECTION properties (collection rate and group ID) are set in the destination picture based on the above command.

The FST (fast) group ID (group #254) is not taken into account as a factor in determining the inheritance priority.

A collection rate of 0 indicates to only update a variable when the display is invoked. 0 is the slowest rate possible and will take precedence when the SLOW priority is in effect.

As for the FAST and SLOW priorities, when a “tie” occurs (both the subpicture and main picture have the same collection rate), the destination picture’s properties will be retained.

**Error Handling**—In R600 the error message, **Invalid Collection Priority Mode**, is added to the Picture Editor. This message is output if an invalid mode is entered or an incorrect mode abbreviation is used.

#### NOTE

If a subpicture is added to a main picture and that subpicture already exists (i.e., a new occurrence of an existing subpicture), or a subpicture is being modified, and that subpicture contains parameters, the “SET COLLECTION” properties for those parameters are unavailable. As a result, the “SET COLLECTION” properties for the final parameter values (specified by the user) cannot be transferred to the main picture. If a final parameter value does not exist in the main picture, its properties will be defaulted (rate = 1, group = 0). This happens after the following commands:

- ADD SUBPICTURE
- MODIFY SUBPICTURE
- ADD VARIANT
- MODIFY VARIANT
- PASTE

A new prompt message, `Check Collection Properties for Subpicture Parameters`, appears warning that the above has occurred if the collection inherit priority is set to something other than MAIN.

To modify the properties to their correct values when the above occurs, do one of the following:

1. Leave it as it is.
2. Use the SET COLLECTION command to modify the defaulted variables.
3. Use the REPLACE SUBPICTURE command (using the same name for old and new subs) to get the correct collection properties. Since the REPL SUB command reads the new subpicture from the specified device, the subpicture can be replaced by itself. The collection properties for the parameters are read in and transferred to all the “final” variables.

## **(R620)— Command: RECREATE {subpicture}**

**Use**—This command allows you to recreate a single subpicture source file automatically from the main schematic source file. The subpicture will be recreated from the main schematic, and saved as a new main source (.DS) file. The new source file for the subpicture can be edited for maintenance, and replaced into the original schematic.

After the command is executed, the original schematic remains on the Picture Editor screen. This will allow the user to execute more commands on the original schematic.

Abbreviated forms:

**RECREATE**  
**CREATE**  
**RECR**  
**CR**

### **Procedure—**

Type RECREATE {pathname >} subpicname on the command line and press the ENTER key.

Where:

- RECREATE is the Picture Editor command.
- pathname is the optional destination pathname for the new subpicture source file. Any parts of the pathname that are not specified will default to the current default Picture Editor pathname (displayed on the top line of the Picture Editor). If any part of the pathname is specified, the > character at the end delineates the pathname from the filename.
- subpicname is the name of the subpicture to be recreated. The Picture Editor will create the subpicture source file with the same name as the subpicture found in the source file.

This command will only recreate the top-level subpictures. Subpictures that are embedded inside subpictures can be recreated by first recreating the top-level subpicture, then recreating the embedded subpictures.

Example:

The user has a Picture Editor source file named MAIN whose location is NET>SCHM. MAIN contains subpicture SUBTOP which in turn contains a variant: IF BOOL01 THEN SUB LOW1 ELSE SUB LOW2. The RECREATE command will only recreate SUBTOP (the top-level subpicture). In order to recreate LOW1 and LOW2, first recreate SUBTOP. Then, read SUBTOP into the Picture Editor (SUBTOP is now the main picture), and finally execute the RECREATE command to recreate LOW1 and LOW2.

## NOTE

The command will first ensure that the subpicture is a Picture Editor built subpicture. If the subpicture is a phantom, the command will terminate and an error message is displayed. If the filename already exists, a warning message to overwrite the existing file will be displayed. The user can either press the ENTER key to overwrite the existing file, or CANCEL to terminate the command.

All the elements found in the subpicture will be transferred to the new source file, using default values found in the main picture (as needed).

Because the RECREATALL command utilizes the variable parsers, the variable information that was available when the variables were added needs to be available as well. For non-parameterized variables (e.g., A100.PV), the variable must exist on the system. If the variable is a user-defined DDB variable (defined in a .DF file loaded by the LOAD command), the .DF file must be loaded into the Picture Editor (by the LOAD command) prior to executing the command. If the variable is a user-define equipment-list variable, the equipment list must be loaded into the Picture Editor (by the LOADEQ command) prior to executing the command. If the information does not exist, the command will terminate, and an error message will be displayed. The only exception to this is if the variable is a subpicture parameter.

To guarantee parameter order, the Picture Editor first defaults the value types for the parameters. During the actual parsing phase of the command, the actual value types for the parameters are found. If the actual parameter is a user-define DDB variable, and the .DF file is NOT loaded, the default value type is used. The command will not terminate with an error message in this case.

The BREAK key will be enabled while the command is executing. Command execution time depends on the size of the subpicture being recreated. If the user presses the BREAK key at any time during the command, the command will terminate. A prompt message will be displayed indicating that the BREAK key was pressed. The command will clean up the aborted operation, and no new subpicture source file will be written.

The following prompt message is added to the Picture Editor for this command.

**Recreating Subpicture  
XXXXXXXXXX.**

This message will be displayed when the Picture Editor is recreating the subpicture source file. "XXXXXXXXXX" will be replaced with the subpicture name.

## **(R620)— Command: RECREATALL {pathname}**

**Use:** This command allows you to recreate all top level subpicture source files automatically from the main schematic source file. This command will execute in a manner that is similar to the RECREATE command.

### **Abbreviated forms**

RECREATALL  
CREATEALL  
RECRA  
CRA

### **Procedure—**

Type RECREATALL {pathname >} at the command line and press the ENTER key.

Where:

- RECREATALL is the Picture Editor command.
- pathname is the optional destination pathname for the new subpicture source files. Any parts of the pathname not specified will default to the current default Picture Editor pathname (displayed on the top line of the Picture Editor). If any part of the pathname is specified, the > character is required at the end. This indicates that the pathname is a directory name, not a filename.

This command will recreate all top-level subpictures found in the current schematic file in a similar manner as the recreate one subpicture command.

Example:

The user has a Picture Editor schematic source file named MAIN, located in NET>SCHM. MAIN contains subpictures EMBED and SUB1. Subpicture EMBED contains a value and a reference to subpicture SUBVAL. Subpicture SUBVAL contains a value and a reference to subpicture SUB2.

The RECREATALL command for the MAIN schematic source file will recreate only the subpictures EMBED and SUB1.

### **NOTE**

If a top-level subpicture is found to be a phantom, the command will ignore the subpicture and continue to the next subpicture. If no subpictures are found, or all subpictures are phantoms, a prompt message will be displayed indicating that no subpictures were found.

## NOTE

If all the top level subpictures are found to be phantoms, the command will ignore the subpicture and continue to the next subpicture. If the next subpicture is not found, a prompt message will be displayed indicating that no subpictures were found. The command will ignore any phantoms it finds. If all the top level subpictures are found to be phantoms, a prompt message will be displayed indicating that no subpictures were found.

## WARNING

If an error occurs during the recreation of a subpicture, the command will be terminated and an error message will be displayed. Any remaining subpictures found in the main picture will not be recreated.

The BREAK key will be enabled during the execution of the command. If the user presses the BREAK key at any time during the command, the command will terminate. A prompt message will be displayed indicating that the BREAK key was pressed. The command will clean up the aborted operation, and the current subpicture source file will not be written. Any remaining subpictures found in the main picture will not be recreated. If the BREAK key is pressed after the command has started writing subpictures, then it will execute to its completion.

**(R620)— Command: MULTRECR {filename}**

**Use:** This command allows you to recreate subpicture source files in batch mode. This will allow the user to create an edited list file of main schematic source files and the subpicture source files that are to be recreated. This list can then be executed as a batch file at some other time.

**Abbreviated Forms:**

**MULTRECR  
MRECR  
MCR**

**Procedure—**

Type MULTRECR {pathname >} filename at the command line and press the ENTER key.

Where:

- MULTRECR is the Picture Editor command.
- pathname is the optional source pathname for the edited list file (.EL extension). Any parts of the pathname not specified will default to the current default Picture Editor pathname (displayed on the top line of the Picture Editor). If any part of the pathname is specified, the > character at the end delineates the pathname from the filename.
- filename is the name of the edited list file (.EL extension).

This command operates in a similar manner to the existing multiple commands specifically, to the Multiple Replace (MULTREP) command. This command allows the user to recreate one or all top-level subpictures in multiple files without being present at the Universal Station. The filename must have an EL extension for edited list. The list file is a text file that contains a list of the commands to be executed.

The following table tells you the existing MULTREP commands that appear in the edited list file and recognized by the MULTRECR command.

Command	Action
AUTO MODE ON	This command sets the Auto mode for the multiple recreate command to ON. When the Auto mode is ON, the engineer will NOT be prompted to enter new subpicture values. Any existing subpicture source files will be automatically overwritten.
AUTO MODE OFF	This command sets the Auto mode for the multiple recreate command to OFF. When the Auto mode is OFF, the engineer will be prompted to overwrite an existing file. Pressing ENTER will overwrite the existing file, while CANCEL will abort the operation.

Command	Action
LOAD <filename>	This command loads a user defined DDB variables file. The syntax for the pathname rules is the same as for the schematic source files (with an exception of file extension DF).
READ <filename>	This command will read the specified schematic file into the Picture Editor. The syntax for the pathname rules is the same as for the schematic source files.  If a full pathname is specified, the default path will change to the specified pathname.
LOADEQ <filename>	This command loads the equipment list variables file. The syntax for the pathname rules is the same as for the schematic source files (with the exception of file extension EQ).
UNLOADEQ <filename>	This command unloads the equipment list variables file. The syntax for the pathname rules is the same as for the schematic source files (with the exception of file extension EQ).

The new commands for recreating subpictures that can appear in the edited list file and are recognized by the MULTRECR command are as follows:

RECREATE <subpicname>	recreate one subpicture
RECREATALL <pathname>	recreate all top-level subpictures

The results of executing the command will be placed into a results file (<filename>.ER), in a similar manner as the MULTREP command.

The BREAK key will operate in a similar manner as with other multiple (batch) commands. When the BREAK key is detected during the multiple command processing, the command currently executing runs to completion, and then the command file is aborted before the next command starts. In other words, the current RECREATE or RECREATALL command will execute to its completion.

## Scenario

This scenario is used when the engineer wants to recreate the subpictures in a schematic named MAIN, and also recreate all the top-level subpictures for all the schematics in the current directory.

For this example, the directory is on volume USR, and the files currently in NET>SCHM are as follows:

MAIN PUMP1      TANK25      OVERVIEW   BIGVALVE

The engineer also wants to guarantee that all the subpictures are recreated. That is, if subpicture EMBED also exists in OVERVIEW, the engineer wants to make sure that both EMBED subpictures are recreated (just in case they are not identical subpictures).



The engineer needs to first ensure that no pictures exist with (possible) duplicate names of the subpictures. The easiest way to do this is to create new directories to place the subpictures. However, the File Manager only allows four characters as a maximum size for directory names. Using the Command Processor, the engineer has to create new directories with names as close as possible to the original schematics as follows:

```
CD NET>USR MAIN
CD NET>USR PMP1
CD NET>USR TK25
CD NET>USR OVWV
CD NET>USR BVLV
```

Next, using the text file editor, the engineer creates the edited list file (named SCHEMS.EL) with the commands to recreate the subpictures. The basic file commands would be as follows.

```
AUTO MODE ON

READ MAIN
RECREATALL NET>MAIN>
READ PUMP1
RECREATALL NET>PMP1>
READ TANK25
RECREATALL NET>TK25>
READ OVERVIEW
RECREATALL NET>OVWV>
READ BIGVALVE
RECREATALL NET>BVLV>
```

Note that other commands, and comments can be added as needed (ex, load commands if user-defined display databases are needed).

Now, using the Picture Editor multiple recreate command, the engineer can execute the edited list file above.

The user enters the following command in the Picture Editor:

```
MULTRECR SCHEMS
```

Some time later, the multiple recreate command finishes execution. The engineer can print out the results file from the Command Processor (PRINT NET>SCHM>SCHEMS.ER).

The results file should look as follows if everything worked correctly:

AUTO MODE ON	Automatic On
READ MAIN	Success!
RECREATALL NET>MAIN>	Success!
READ PUMP1	Success!
RECREATALL NET>PMP1>	Success!
READ TANK25	Success!
RECREATALL NET>TK25>	Success!
READ OVERVIEW	Success!
RECREATALL NET>OVVW>	Success!
READ BIGVALVE	Success!
RECREATALL NET>BVLV	Success!

The engineer can also list the source files found in the directories to see all the subpictures that were recreated. If embedded subpictures inside the top-level subpictures are needed, the user can now repeat this scenario using the newly recreated subpictures.

#### **Error Handling During Subpicture Recreation**

The following error messages are added to the Picture Editor.

##### **Subpicture is a Phantom – Cannot Recreate.**

This message will be displayed while using the RECREATE command if the subpicture attempting to be recreated is a phantom.

##### **Error Recreating Subpicture XXXXXXXXXX.**

This message will be displayed while using the RECREATALL command if an error occurs while recreating a subpicture. “XXXXXXXXXX” will be replaced with the subpicture name.

The following error messages are added to the multiple recreate results:

##### **Recreate Operation Failed**

This message will be displayed if the RECREATE or RECREATALL command fails.

##### **No Subpictures Found**

This message will be displayed if the RECREATALL command either encounters no subpictures, or if all the subpictures encountered are phantoms. This is not an error – just an information message.

Since the main picture is not affected by the RECREATE or RECREATALL commands, if an error occurs in either command, the subsequent commands will still be processed. In contrast, multiple replace commands affect the main picture – errors result in subsequent commands being skipped until a new file is read (or new database definitions loaded).

### 3.3.10.7 PE Compile Error Involving Subpicture Parameters

A compile error can occur in a Picture Editor schematic containing subpictures with user-visible (parameterized) variables. The error manifests itself in the following situation:

A user-visible (parameterized) variable is used BOTH as a “stand-alone” variable AND as an index in another variable IN THE SAME OBJECT.

Some examples (user-visible variable in bold type):

(value object)	<b>&amp;A</b> + PT.PAR( <b>&amp;A</b> )
(behavior object)	IF INT01 = <b>&amp;B</b> THEN SET REV; IF PT.PAR( <b>&amp;B</b> ) = INT02 THEN SET BLINK;
(variant object)	IF INT01 = <b>&amp;C</b> THEN “INT01” ELSE IF PT.PAR( <b>&amp;C</b> ) = INT02 THEN “INT02”

A subpicture is written that contains the above object. If the subpicture is added to a main picture, and the user-visible variable is not used separately in the subpicture in another object (either as a “stand-alone” variable, or as an index in a variable), an error will occur when the main picture is compiled. The error message `Compile Error` will appear and no object will be blinking to signal it had an error.

The error occurs as a result of the parse of the object. The user-visible variable, in this situation, is not correctly flagged to be a formal parameter to the subpicture. There is no “final” (bound) variable returned to the subpicture for the user-visible variables at compile time.

#### **Solution #1: (EASIEST)**

Use two different user-visible variable names in the above situation. When the subpicture is added to the main picture, use the same “final” (bound) value for both subpicture parameters.

1. ADD the desired object to a picture. Use two different user-visible variable names.

Example – behavior object from above: (user-visible variable in bold type)

```
IF INT01 = &B1 THEN SET REV;  
IF PT.PAR(&B2) = INT02 THEN SET BLINK;
```

2. WRITE the subpicture (SUBPIC1). Supply appropriate prompts for the user-visible variables.

PE write screen example:

Prompt for &B1	<u>ENTER VALUE</u>
Prompt for &B2	<u>ENTER INDEX</u>

3. ADD the subpicture in step 2 (SUBPIC1) to the desired main picture. Supply the actual (final) variable desired for BOTH parameters when prompted.

PE add subpicture screen example:

ENTER VALUE	<u>A100.PV</u>
ENTER INDEX	<u>A100.PV</u>

### **Solution #2:**

This solution should only be used if one user-visible parameter is necessary for the “final” product (i.e., customer subpicture; max number of parameters would be exceeded, etc.). It involves “nesting” the original subpicture inside another subpicture to obtain the one user-visible parameter “feel”.

1. ADD the desired object to a new blank picture. Use two different user-visible variable names (in other words, do solution #1).

Example: (value object from above)

**&VALUE + PT.PAR(&INDEX)**

2. WRITE the subpicture to a file (SUBPIC1). Supply appropriate prompts for the user-visible variables.

PE write screen example:

Prompt for &VALUE      ENTER VALUE

Prompt for &INDEX      ENTER INDEX

3. ADD the subpicture in step 2 (SUBPIC1) to a new blank picture. Supply the actual (final) user-visible variable desired when prompted.

PE add subpicture screen example:

ENTER VALUE              &A

ENTER INDEX              &A

4. WRITE this subpicture to a new file (SUBPIC2). Supply the actual (final) prompt for the user-visible variable.

PE write screen example:

Prompt for &A              VAT001 INDICATOR

5. At this point the subpicture in step 4 can be added to the main picture. If the subpicture in step 4 is not the final subpicture desired, the subpicture can be added to another subpicture (as necessary) to obtain the final subpicture desired.

## APPENDIX A VALUE FORMATS

The ADD VALUE command allows the addition of values to a picture. Legal types for values are as follows: INTEGER, REAL, STRING, BOOLEAN, ENUMERATION, UNKNOWN, and DATE TIME.

In order to display these values at run time, a format specification must be present. This appendix describes the format specification used for each value type and defines the default format.

### A.1 INTEGER

Keyword                      Synonyms

INTEGER                    INT I

The following characters are legal in an integer format (symbols within single quotes are literals):

- I     =     Indicates an integer format.
- +     =     Causes either a '+' or a '-' to be printed. Each '+' after the first one represents a digit position that the sign floats over if the digit is zero.
- =     Causes a '-' to be printed if the number is negative. Each '-' after the first one represents a digit position that the sign floats over if the digit is zero.
- Z     =     Represents a digit position that is blanked if the digit is zero.
- L     =     Represents a digit position that is nulled by left-justification if the digit is zero.
- 9     =     Represents a digit position.

A legal integer format consists of an 'I', followed by zero or more '+'s or zero or more '-'s, followed by zero or more 'Z's or zero or more 'L's, followed by one or more '9's.

The default format for values of type INTEGER is

**I-ZZZZ9**

Examples:

Value	Format					
	<b>I99999</b>	<b>I+ZZZ99</b>	<b>I++++99</b>	<b>I-ZZ999</b>	<b>I---Z99</b>	<b>I-LLLL9</b>
123	00123	+ 123	+123	123	123	123
-12	00012	- 12	-12	- 012	-12	-12
2345678	#####	#####	#####	#####	#####	#####
0	00000	00	00	000	00	0

The limit for integers is -32767 to +32767. A result that exceeds the limit is presented as ----- (Not a Number).

## A.2 REAL

Keyword                      Synonyms

REAL                        NUMBER NUM R

The legal characters in a REAL format are 'R', '+', '-', 'L', 'Z', '9', and '.'. The characters have the same definitions as the characters in an Integer format except that the 'R' indicates a REAL format is to follow and the '.' indicates the position of the decimal point.

The 'R' indicates a REAL format with a decimal point that floats to the right if the magnitude of the number requires more digit positions than is specified in the format.

A legal REAL format consists of an 'R' followed by zero or more '+'s or zero or more '-'s, followed by one or more '9's, followed by a '.', followed by one or more '9's, or nothing.

The default format for values of type REAL is

**R-ZZZZ9.99**

The maximum number of significant digits is six. The Picture Editor allows you to choose more than six.

Examples:

Value	Format				
	<b>R++++Z9.9</b>	<b>R-ZZZ99.99</b>	<b>RLLLL99</b>	<b>R-ZZZ9.999</b>	<b>R999</b>
345.87	+345.9	345.87	346	345.870	346
-345678	-345678.	-345678.0	345678	-345678.0	***
0.12345	+0.1	00.12	00	0.123	000
-1.1E-56	-0.0	- 00.00	00	- 0.000	000
0	+0.0	00.00	00	0.000	000

## A.3 BOOLEAN

Keyword                      Synonyms

BOOLEAN                    LOGICAL L B

The format specification used for values of type BOOLEAN is the format specification defined elsewhere in this section for STRING.

The default format for values of type BOOLEAN is

**TEXTL1:5**

## A.4 STRING

Keyword                      Synonym

STRING                      S

The legal characters for a text format specification are as follows:

TEXT	=	Indicates a text format.
L	=	Signifies left-justify.
R	=	Signifies right-justify.
C	=	Signifies center-justify.
p	=	A digit from 1 to 9.
d	=	A digit from 0 to 9.
:	=	Separator between the field specifiers.

A legal text format consists of the word TEXT followed by L or R or C, followed by a number from 1 to 99, followed by a :, followed by a number from 1 to 99.

The first number specifies the starting character of the string and the second number specifies the output-field width.

Examples:

Value	Format		
	TEXTL1:5	TEXTR2:10	TEXTC3:15
Fred	Fred	red	ed
Lazy Fox	Lazy	azy Fox	zy Fox
Twenty-Two	Twent	wenty-Two	enty-Two

The default format for values of type STRING is

TEXTL1:10

## A.5 ENUMERATION

Keyword	Synonym
---------	---------

ENUM	E
------	---

Example entry— ENUM:MODE

The format specification used for values of type ENUMERATION is the format specification defined elsewhere in this section for STRING.

The textual representation of the enumerated value is written as text. The field width is defined in the format specification.

The default format for values of type ENUMERATION is

TEXTL1:10

## A.6 SELF-DEFINING ENUMERATIONS

Keyword	Synonym
---------	---------

SD_ENUM	SD_ENM
---------	--------

Self-defining enumerations can be added to a schematic as variables and in relational expressions. Self-defining enumerations consist of internal and external values, or states. The external value is a character string that you configure using the Data Entity Builder. For example, a point X.Y can be configured as a self-defining enumeration with two states. These states are displayed to the user as **UP** or **DOWN** (see Appendix C for intrinsic functions used to retrieve the internal and external forms of a self-defining enumeration).

The default format for self-defining enumerations is a 10-character text string, left-justified, beginning with character 1 (TEXTL1:10). Values are displayed during the picture-building process with the letter E representing each character of the string.

Self-defining enumerations can also be added to subpictures in the form of parameters (e.g., &P1). Parameters follow the same rules as values.

In relational and conditional expressions, self-defining enumerations can only be compared for equality or inequality (= or <>). The left side of the relational must be a variable or user-visible parameter. The right side can be a variable, user-visible parameter, or a constant value (e.g., AUTO, DOWN).



Type checking for self-defining enumerations is deferred until the display is called up. At that time, if a self-defining enumeration is being compared to a variable, parameter, or constant value, and the value string on the right side of the expression is not in the list of valid strings, the FALSE limb is always taken.

Example: X.Y is a self-defining enumeration with states UP and DOWN. The expression

```
IF X.Y = NO ACK THEN "NOT ACKNOWLEDGED" ELSE "ALARM
ACKNOWLEDGED"
```

will always appear on the screen as ALARM ACKNOWLEDGED.

When the display is called up, values and parameters that are self-defining enumerations are retrieved to determine the corresponding text string (defined at configuration time). Constants are "transformed" to corresponding text strings. Strings for relationals are compared, and based on the expression, the TRUE or FALSE limb of the relational is taken. For values, the string is displayed according to the format you specify.

#### Examples

<u>Value</u>	<u>Displayed As</u>
• X.Y: format TEXTL1:10 value "FRED"	FRED
• &P1: format TEXTL2:6 value "TENCHARS"	ENCHAR

<u>Relationals/Conditionals</u>	<u>Displayed As</u>
• IF X.Y = OLD THEN "OLD" ELSE "NEW" value of X.Y is not OLD	NEW
• IF X.Y = OLD THEN SET RED HALF ELSE SET GREEN FULL value of X.Y is not OLD	GREEN FULL
• IF A.B <> C.D THEN "NOT EQUAL" ELSE "EQUAL" value of A.B = value of C.D	EQUAL
• IF A.B <> C.D THEN "NOT EQUAL" ELSE "EQUAL" A.B and C.D are not of the same type (e.g., A.B = OLD or NEW, C.D = YES or NO)	NOT EQUAL

## A.7 UNKNOWN

Keyword            Synonym

UNKNOWN        U

A value with the type UNKNOWN must be resolved at compile time. When the type of the value becomes known, a format is assigned. This format corresponds to the type of the value and is the default-format specification for that type.

In order to know the field width required for the value, a dummy format is specified. The format used is the general format, specified without a sign.

The legal characters in the dummy format are as follows.

G    =    Indicates a dummy format is to follow.  
9    =    Represents a position of the format.

A legal dummy format consists of a 'G' followed by one or more '9's.

The number of 9's corresponds to the field width of the value.

The default format for values of type UNKNOWN is

G999999

Examples:

Value	Format		
345.87	+345.870	345.9	###
-0.34567	-0.34567	-0.346	0.3
12345	+12345.0	#####	###
1.1E-56	#####	#####	###
0	+0.00000	0.000	0.0

## A.8 DATE TIME

Keyword      Synonym

DATE TIME    DATE/TIME    TIME    DATE  
DURATION    DUR

### Date Formats

A legal date format consists of the following:

- the word DATE,
- followed by at least one of the date items listed below,
- followed by the word ENDDATE.

A format can contain a maximum of 10 date items and/or ASCII characters between the words DATE and ENDDATE. The width of the format is the width of each date item plus the number of ASCII characters. The date format allows all printable characters including blanks. If the date requires less space than the format, it is left justified.

The Date Items are—

DD	=	Indicates day of month (01-31).
DY	=	Indicates day of year (001-366).
DW	=	Indicates day of week (1-7).
AD:d	=	Indicates alpha day of week (Sunday-Saturday). The 'd' is a digit from 1 to 9 that indicates how many digits of the alpha name are to be displayed. If the name of the day has fewer characters than specified, the name is displayed without extra spaces. Using a name width of 9 produces the complete name of any day.
WY	=	Indicates week of year (01-53).
MM	=	Indicates month of year (01-12).
AM:d	=	Indicates month of the year as a name (January-December). The 'd' is a digit from 1 to 9 that indicates how many digits of the name are to be displayed. If the name of the month has fewer characters than specified, the name is displayed without extra spaces. Using, a name width of 9 produces the complete name of any month.
YY	=	Indicates 2-digit year (for example, 93).
YYYY	=	Indicates 4-digit year (for example, 1993).

Examples:

Value	Format	Result
9/27/93	DATEMM:DD-YYENDDATE	09:27-93
9/27/93	DATEAD:3: AM:9 DD, YYYYENDDATE	MON: SEPTEMBER 27, 1993
1/01/93	DATEMM:DD-YYENDDATE	01:01-93
1/01/93	DATEAD:3: AM:9 DD, YYYYENDDATE	FRI: JANUARY 01, 1993

## Time Format

A legal time format consists of the following:

- the word **TIME**,
- followed by at least one of the time items listed below,
- followed by the word **ENDTIME**.

A format can contain a maximum of 10 time items and/or ASCII characters between the words **TIME** and **ENDTIME**. The width of the format is the width of each time item plus the number of ASCII characters. The time format allows all printable characters including blanks. If the date requires less space than the format, it is left justified.

The Time Items are—

HS	=	Indicates hours (1-12).
HH	=	Indicates hours (00-23).
MM	=	Indicates minutes (00-59).
SS	=	Indicates seconds (00-59).
AM	=	Indicates AM or PM.
MS	=	Indicates milliseconds (000-999).

Examples:

Value	Format	Result
13:06:57	TIMEHH:MM:SS MSENDTIME	13:06:57 123
13:06:57	TIMEHS:MM AMENDTIME	1:06 PM
12 Midnight	TIMEHH:MM:SS MSENDTIME	00:00:00 000
12 Midnight	TIMEHS:MM AMENDTIME	12:00 AM

### CAUTION

When you enter a time format, the first character inside the format port must be blank. The cursor is automatically positioned for you.



## Duration Format

A legal duration format consists of the following:

- the word DUR,
- followed by at least one of the duration items listed below,
- followed by the word ENDDUR.

A format can contain a maximum of 10 duration items and/or ASCII characters between the words DUR and ENDDUR. The width of the format is the width of each duration item, plus the number of ASCII characters. The duration format allows all printable characters, including blanks. If the duration requires less space than the format, it is left justified.

The Duration Items are—

DD	=	Indicates number of days in the duration (up to five Ds). By default, this is the duration from Jan. 1, 1979.
HH	=	Indicates hours (00-23).
MM	=	Indicates minutes (00-59).
SS	=	Indicates seconds (00-59).

The default format for values of type DATE TIME is

DATEMM-DD-YYENDDATE

Like the time format, the first character inside the format port must be blank.



## APPENDIX B NAME FORMS

Name forms are used for specifying point IDs to the Picture Editor.

Examples of simple name forms are

<b>A100.PV</b>	Point.Parameter
ENT02	A DDB variable (Display Database Variables are described in Appendix A to the <i>Actors Manual</i> )
ENT02.PV	

### Indexed Expressions—

An index is

- a constant of type Integer, Real or Standard Enumeration
- any DDB variable whose final data type is Integer, Real or Standard Enumeration
- a system variable of the form: **System Entity.Parameter** where the data type is Integer, Real, or Standard Enumeration. Note that a system variable can only be indexed as a constant. Examples: A100.PV(B100.PV(5)), A100.PV(B100.PV(MODE:AUTO)).

The following forms of indexing can be used in Values, Variants, Conditional Behavior, and Bar Charts:

A100.PV(<index>) (a parameter can be indexed by a constant or variable)  
A100(<index>).PV (an entity can only be indexed by an integer constant)  
A100(<index>).PV(<index>)  
ENT01.PV(<index>)  
ENT01.P1(<index>).P2(<index>)

For example—

A100.P1(INT01)  
A100.P1(B100.P2)

Index expressions can be in the form  $A * X + C$

where—

A and C are optional Integer constants (positive or negative)

X is

- a point. parameter value
- a parameter substitution
- a DDB Variable whose final type is Integer or Real

### Examples

INT01 + 5	-2*ENT10.PV	&J + 3
&J - 3	3*A100.PV - 25	

**Entity/Parameter Substitution**—The symbol **&** is used when substituting for an entity, entity index, parameter, or parameter index. This allows objects to be built for more general use. For example the expression **&A.PV** can be used in a subpicture. Each time the general subpicture is added to a schematic, the Picture Editor prompts for a value to substitute for **&A** in that situation.

Parameter substitution can be used in Target actions, Values, Conditional Behavior and Variant statements.

#### Examples

<b>&amp;A</b>	<b>&amp;A = FC100.PV</b>
<b>&amp;B.PV</b>	<b>&amp;B = A100</b>
<b>A100.&amp;C</b>	<b>&amp;C = PV</b>
<b>A100.&amp;D</b>	<b>&amp;D = TANK(3).LVL</b> illustrates indirection (see below) and indexing
<b>A100.PV(&amp;i)</b>	<b>&amp;i = Integer constant or variable</b>
<b>B100(3).STPTIM(&amp;STPIN X)</b>	<b>(3)</b> illustrates indexing and <b>&amp;STPINX</b> illustrates substitution
<b>&amp;A</b>	<b>&amp;A = TNKARA2.TANK(I).LVL</b>
<b>&amp;A.PV.LVL(&amp;i)</b>	<b>&amp;A = TNKARAI</b> and <b>&amp;i = an Integer constant or variable</b>

The Combine Variable and Arithmetic actors can be used in Target action phrases to generate fully indexed expressions with at least two levels of indirection.

For an indexed parameter, (e.g., **A.PV(&p1)**) use **C\_VAR (A,0,.PV,&p1)**

For an indexed entity, (e.g., **A(&p1).PV**) use **C\_VAR (A,&p1,.PV,0)**

#### Example—

Assume **A100.P1(INT01+INT01G)** is of type integer.

To store the number 44 into **A100.P1(INT01+INT01G)**, use the following action phrase:

```
SS_INT ( C_VAR ( A100 , 0 , .P1 , ADD_I ( G_INT ( INT01 ) , G_INT ( INT01G ) ) ) , 44 )
```



**Indirection**—Expressions used in display objects and Target action phrases, can contain up-to-two levels of indirection. If an expression contains two levels of indirection, the first level must be a DDB Variable.

For example—

ENT01.E1.PV

where—

ENT01 is a DDB variable that contains an entity name

E1 is a parameter on that entity

PV is a parameter

Other examples—

- ENT01.P1(INT01G).P2(B100.P4)
- ENT01.P1(A100.P3).P2(INT01G)
- A100.ENT.PV (where A100.ENT is a whole system variable and, therefore, only one level of indirection is indicated).

Consider the following expression:

ENT03.TANK(i).VAL(j)

where—

ENT03 is a DDB variable that contains the entity TNKARA02

TANK(i) is an array of entities in TNKARA02

VAL(j) is an array of parameters

An example of this expression used in a Variant statement is

```
IF ENT03.TANK(&i).VAL(&j) > 5 THEN
SUBPICTURE LOALARM
ELSE
SUBPICTURE NORMAL
```

In the following target action phrase, target execution assigns a numeric value to  
ENT03.TANK(&i).VAL(&j)

```
S_VAR (VAR01,C_VAR(GS_ENT(C_VAR(G_ENT(ENT03),0,.TANK,&i)),0,.VAL,&j));
RS_SYS(VAR01.,35,1,10,"Enter Value",TRUE,1)
```

## Syntax—

Terms in the syntax diagrams are defined as follows:

DDB_NAME	A DDB variable of a type other than 'Entity_ID' or 'Variable_ID'.
DDB_VAR	A DDB variable of type 'Variable_ID'.
DDB_ENT	A DDB variable of type 'Entity_ID'.
ENTITY	A system entity name.
PAR	A system parameter name.
int	An integer constant.
set_name	An enumeration set name.
member_name	An enumeration member name.

Note that—

1. Embedded spaces are not allowed in these variables.
2. Items enclosed in single quotes are literals.
3. A symbol of the form '&name' can be substituted for any of the following:

VARIABLE	ENTITY
DDB_NAME	PAR
DDB_ENT	

Note that if an entity or parameter is of the form **&NAME**, an index cannot be specified after it. The index must be substituted along with the formal entity or parameter.

Examples—

for an entity

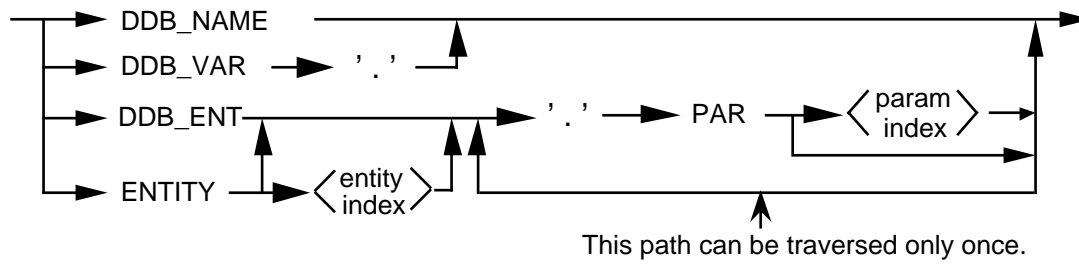
for a parameter

&A.PV where &A = X(5)

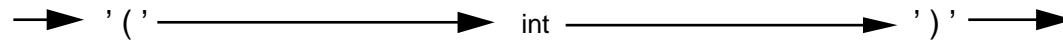
A100.&B (where &B = PV(INT01))

The syntax of name forms is—

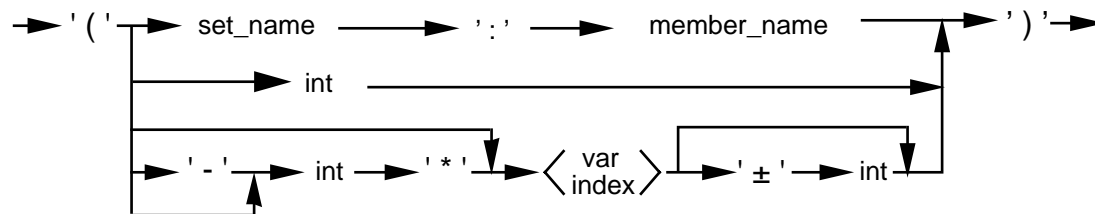
#### <variable>



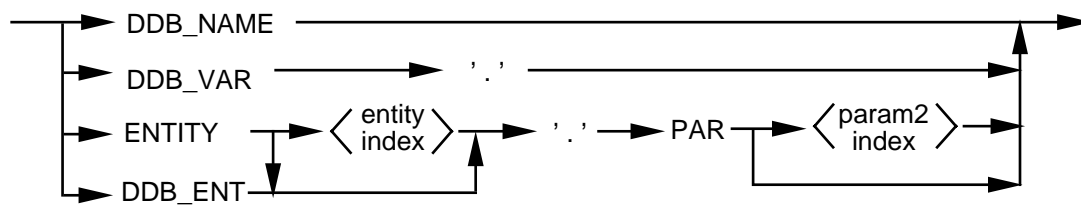
#### <entity index>



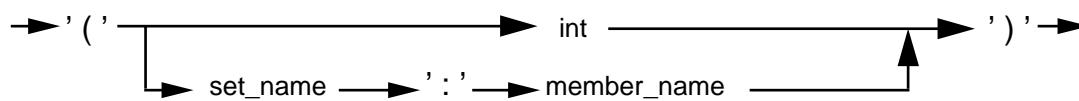
#### <param index>



#### <var index>



#### <param2 index>



Additional examples:

A100.PV INT01 ENT02.PV A100.PV\_ARR(INT01)

B100.PV\_ARR(A100.PV\_ARR(5)) ENT01.ENT.PV\_ARR(2\*A100.PV)

C100(5).ENT\_ARR(-5\*A100.PV\_ARR(1)).PV\_ARR(INT01-2)

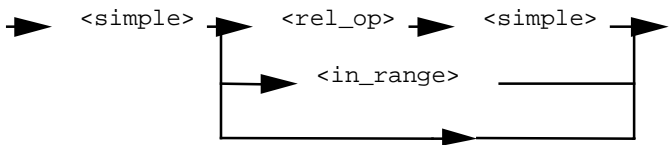
D100.PV(-5\*ENT02.PV\_ARR(1)+16)

# APPENDIX C EXPRESSIONS

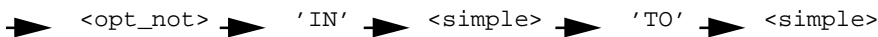
## C.1 EXPRESSION SYNTAX

The syntax of expressions is defined below. Note that an expression cannot contain more than six nested expressions which contain constant values. For example,  $(A+1) + (B+2) + (C+3) + (D+4) + (E+5) + (F+6) + 7$  is illegal.

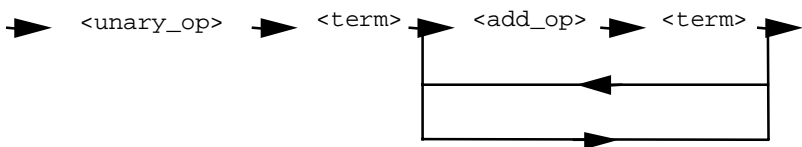
<expression>:



<in\_range>:

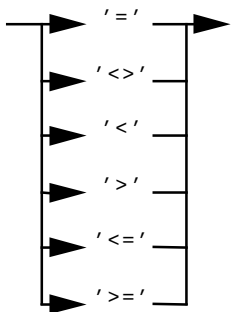


<simple>:



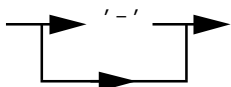
### Notes

<rel\_op>:

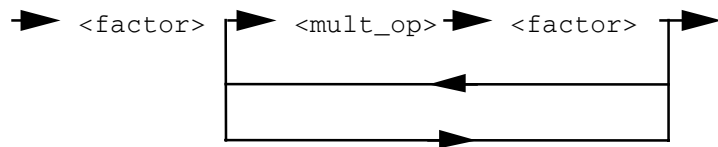


<u>Symbol</u>	<u>Meaning</u>
=	is equal to
<>	is not equal to
<	is less than
>	is greater than
<=	is equal to or less than
>=	is equal to or greater than

<unary\_op>



<term>:

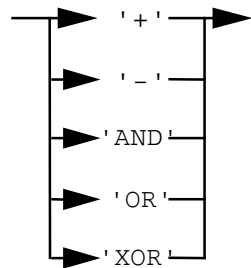


<add\_op>:

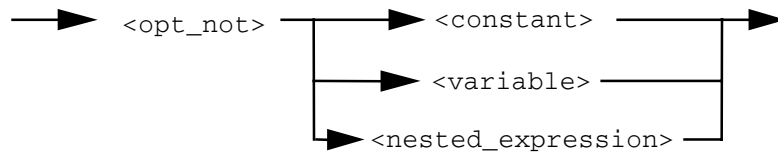
Examples:

A210+100

A210-100



<factor>:

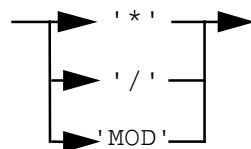


<mult\_op>:

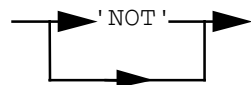
Examples:

A110\*2

A210.PV/10



<opt\_not>



<nested\_expression>

-> '(' --> <expression> --> ')' -->

## C.2 INTRINSIC FUNCTIONS

An intrinsic function is similar to a Pascal function in that it takes a parameter and returns a single result. These functions can be used in expressions wherever a variable of the expression's type is allowed.

The known intrinsic functions are—

### C.2.1 Bit\_Test

Format: **Bit\_Test** (Value, Bit\_Number)

Where:

**Value** is the real value to be checked. The contents of 'Value' must be between 0.0 and 65,535.0 inclusive. If the contents of 'Value' is within this range, the contents will be truncated to an integer. If the contents of 'Value' are outside the range, Bit\_Test will evaluate to a bad value.

**Bit\_Number** is the number of the bit to be tested. The contents of 'Bit\_Number' must be an integer between 1 and 16 inclusive, where a value of 1 represents the least significant bit and 16 represents the most significant bit. If the contents of 'Bit\_Number' is less than 1 or greater than 16, Bit\_Test will evaluate to a bad value.

A Boolean is returned.

Description: Tests a specified bit of the provided real number and returns ON or OFF if that bit number in the value is ON or OFF.

Example:           IF BIT\_TEST (A100.PV,3) THEN  
                      "The third bit is ON"  
                  ELSE  
                      "The third bit is OFF"

The third bit of A100.PV is tested and the appropriate limb of the Variant is executed.

## C.2.2 External

Format: **External** (enumeration state)

Description: Converts a value of type Self-Defining Enumeration to a value of type String.

Example: IF EXTERNAL(A100.PV) = A100.STATE1 THEN ...

A100.PV is of type SD\_ENUM  
A100.STATE1 is of type STRING

### CAUTION

Always use EXTERNAL to compare the state of a Digital Input, Digital Output, or Digital Composite point or Flag point to one of the STATE or PVSTATE parameters. If you compare to the actual state name, no type checking is done.

In the following example, if the actual state is CLOSED, but you misspell it as "CLOSE" then such a condition or variant will compile, but because the PV will never be equal to "CLOSE" the condition will never be true.

IF DCTAG.PV = CLOSE THEN SUB VALVE

## C.2.3 Internal

Format: **Internal** (enumeration state)

Description: Returns the internal form of the supplied Self-Defining Enumeration as an Integer.

Example: IF INTERNAL(A100.PV) = 3 THEN ...

A100.PV is of type SD\_ENUM

Where      Forward = 0  
             Stopped = 1  
             Reverse = 2

If A100.PV were currently stopped, the Internal function would return a 1.

## C.2.4 Status

Format: **Status** (value)

Description: evaluates the supplied expression and retrieves status (PTVALSTS Enm;) of the final value.



Status returns either: NO\_ERROR, VAL\_ERR, COMM\_ERR, or CONF\_ERR

Examples: IF STATUS(A100.PV) = COMM\_ERR THEN SET RED BLINK

IF STATUS(A200.OP) <> NO\_ERROR THEN SET RED

### C.3 ARRAYS AS PARAMETERS

Several array types are supported as parameters to Application Subpictures. These are array of integer, real, Boolean, data time, entity ID, parameter ID, enumerations, and self-defining enumerations.

When an Application Subpicture is added and a type must be specified, an array is identified by using the keyword ARRAY. For example, an integer array is identified by typing INTEGER ARRAY; likewise, an example of an enumeration array is identified by typing ENUM:ACCESTYP ARRAY.

### C.4 SEMI-RESERVED NAMES

The Picture Editor and Button Configurator have access to certain variables, in particular, those in the local and system Display Databases (refer to the *Actors Manual*, Appendix A).

If a system entity whose name matches one of these names is configured with the Data Entity Builder, the system entity will not be accessible from the Picture Editor or the Button Configurator. To avoid conflict, the following names should be reserved for use with the Picture Editor and Button Configurator.

ACKSTAT	STR01G - STR20G	TR_RNGLO
BOOL01 - BOOL20	STRING01 - STRING20	TR_SCRLL
BOOL01G - BOOL20G	SYS_TIME	TR_TIME
CZACTIVE	TMDAY	VAR01 - VAR20
DATIME1-DATIME4	TMHOUR	VAR01G - VAR20G
ENM01 - ENM10	TMMIN	\$ARAIID
ENM01G - ENM10G	TMMONTH	\$ARAIID01 - \$ARAIID10
ENT01 - ENT020	TMSHIFT	\$ARADSC
ENT01G - ENT20G	TMUSR_AV	\$ARADS01 - \$ARADS10
HMDAY	TREND01-TREND12	\$BUTTONS
HMHOUR	TR_COLOR	\$CONDSC
HMMIN	TR_DATA	\$CONDS01 - \$CONDS10
HMMONTH	TR_NAME	\$CONNUM
HMUSR_AV	TR_RNGHI	\$CZ_ENTY
INT01-INT20	\$STNNUM	\$GRPBASE
INT01G - INT20G	\$MY_AREA	\$KEYLEVL
REAL01 - REAL20	\$MY_PNA	\$REDYEL
REAL01G - REAL20G		

## C.5 DATA TYPES IN EXPRESSIONS

When the Picture Editor encounters a variable of an unknown type, a screen form is presented that requests the type. If the data type is known at that time, the type must be entered on the form as the Picture Editor uses the type to validate the expression.

The following types are legal in the Picture Editor.

<u>Keyword</u>	<u>Synonyms</u>	
BLIND REC	BLIND	BL (syntax is 'BLIND_REC:blind rec id' e.g., BLIND_REC:0)
BOOLEAN	B	LOGICAL L
DATE TIME	DATE/TIME	DATE TIME
ENTITY ID	ENTITY	ENT
ENUM	E	(syntax is 'ENUM: enum id' e.g., ENUM:MODE)
INTEGER	INT	I
PARAMETER	PARAM	P
REAL	NUMBER	NUM R
SD_ENUM	SD_ENM	
STRING	S	
UNKNOWN	U	

### C.5.1 Operations between Two Operands

The following table defines legal operations between two operands. Note that real/integer means either real or integer.

<u>Type of Op 1</u>	<u>Type of Op 2</u>	<u>Assumed Type</u>	<u>Legal Operators</u>	<u>Type of Result</u>
Boolean	Boolean		AND OR XOR NOT rel ops	Boolean
Boolean	Unknown	Boolean	AND OR XOR NOT rel ops	Boolean
	Unknown	Enum	= <>	Boolean
Enum	Enum		= <>	Boolean
Integer	Integer		+ - * / MOD rel ops	Integer/Boolean
Integer	Real		+ - * / MOD rel ops	Real/Boolean
Integer	Unknown	Integer	+ - * / MOD rel ops	Integer/Boolean
Real	Real		+ - * / MOD rel ops	Real/Boolean
Real	Unknown	Real	+ - * / MOD rel ops	Real/Boolean
Sd Enum	Sd Enum		= <>	Boolean
Sd Enum	Unknown	Sd Enum	= <>	Boolean
String	String		Rel ops	Boolean
String	Unknown	String	Rel ops	Boolean
Unknown	Unknown		All operators	Unknown/Boolean

### C.5.2 Referencing AM/CL Custom Data Segment Boolean Parameters

A Custom Data Segment (CDS) parameter that is defined as Logical in Application Module Control Language (AM/CL) is defined as Boolean in the Picture Editor.

A CDS parameter that is defined as Boolean in CL is defined as an Enumeration with a subtype of Boolean in the Picture Editor.

A problem arises when a CDS parameter is defined as Boolean in CL and is referenced in the Picture Editor. The picture compiles, but the condition may show as a bad value.

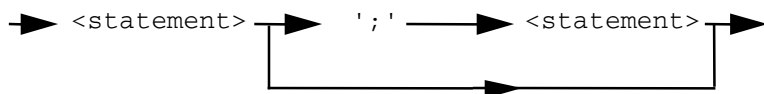
Solution — When using an indirect reference in a condition such as ENT01.PAR, where PAR is a CDS parameter that has been defined as Boolean, the variable type that must be entered is **ENUM:BOOLEAN**.



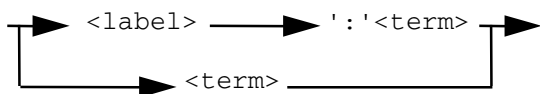
## APPENDIX D CONDITIONAL BEHAVIOR SYNTAX

The syntax for a conditional behavior is as follows:

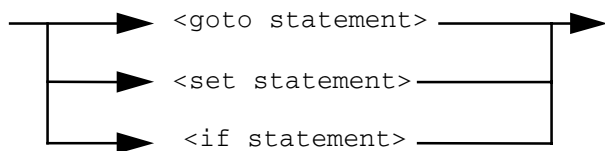
<conditional>:



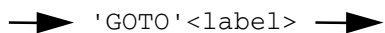
<statement>:



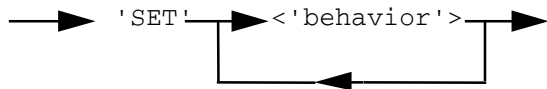
<term>:



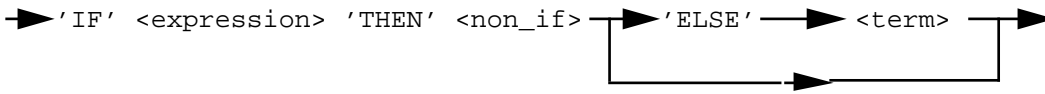
<goto statement>:



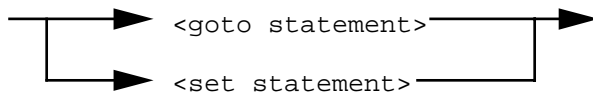
<set statement>:



<if statement>:



<non\_if>:



where:

<expression> is a valid expression as defined in Appendix C

<label> is an identifier that can contain

alphabetic character A to Z, a to z, and special foreign-language characters numeric characters 0 to 9

Restrictions:

uppercase and lowercase alphabets are considered identical.

must contain at least one non-numeric character

cannot begin or end with an underscore

cannot have two successive underscores

<behavior> is any valid behavior keyword as defined for the command SET BEHAVIOR

#### CAUTION

Check your statements carefully. An endless loop can cause station lockup at run time.

## Examples of Conditional Behavior Statements

Note that these examples are provided to show how conditional behavior is used and the statements have not been verified on an operating system.

### 1. Situation Description:

If the PV value from point A1 is greater than 100, you want an object in the picture to be red.

If point B1 is operating in Automatic mode, the object is to be red and to blink, otherwise, the object is to be cyan.

### Conditional Behavior statement:

**Initial behavior:** CYAN NO BLINK FULL NO REVERSE

Condition:

```
IF A1.PV > 100.0 THEN SET RED ELSE  
IF B1.MODE = AUTO THEN SET RED BLINK
```

### 2. Situation Description:

The value of A100.PV status can be enumerated as normal, uncertain, or bad. If the value of A100.PV status is bad and if the output from point B100 is greater than 100.2, you want an object in the picture to be red.

If A100.PV status is bad and if the output from point B100 is less than or equal to 100.2, the object is to be blue, otherwise, the object is to be cyan.

### Conditional Behavior statement:

**Initial behavior:** CYAN NO BLINK FULL NO REVERSE

Condition:

```
IF A100.PVSTS <> BAD THEN GOTO A  
IF B100.OP > 100.2 THEN  
SET RED  
ELSE  
SET BLUE  
A:
```

Note that A: is a label where additional conditions can be entered if necessary.

### 3. Situation Description:

AKSTAT is a collector that provides one of three acknowledge states for a point (see Appendix G to this publication).

If one or more alarms have been acknowledged on point A1 and if point B100 is operating in automatic mode, an object in the picture is to be yellow and blink.

If one or more alarms are unacknowledged on point A1 the object is to be red and blink.

If there are no alarms from point A1 and if point B100 is operating in manual mode, the object is to be green.

**Initial Condition statement:** MAGENTA HALF NO BLINK NO REVERSE

Conditional Behavior Statement:

```
IF ACKSTAT(A1) <> UNAKALRM THEN GOTO A
ELSE SET RED BLINK FULL;
GOTO B;
A: IF ACKSTAT(A1) = AKDALRM THEN GOTO C;
IF B100.MODE = MAN THEN SET CYAN FULL
ELSE IF B100.MODE = AUTO THEN SET GREEN FULL;
GOTO B;
C: IF B100.MODE = AUTO THEN SET YELLOW BLINK FULL;
B:
```

B: is a label where additional conditions can be specified.

Miscellaneous additional examples:

```
IF A210.HIGHAL <> NOALARM THEN SET RED BLINK

IF A230.PV - A303.SP > 20 THEN SET GREEN FULL

IF ACKSTAT(A210) <> NOALARM THEN SET RED

IF A100.PV = ACTIVE THEN SET REV

IF STRING01 = "MANUAL" THEN SET WHITE BLINK

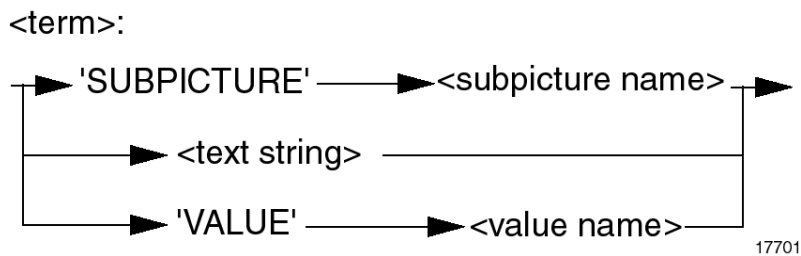
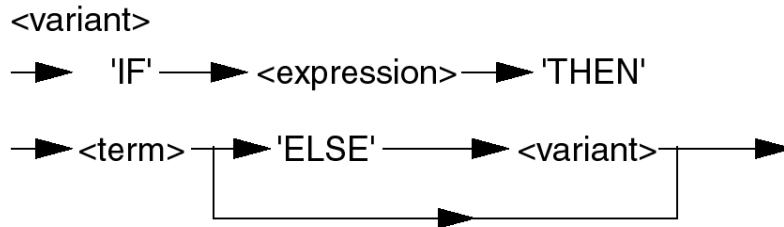
IF ENT01.OP = STOP THEN SET REVERSE

IF &A.MODE = AUTO THEN SET RED
```



## APPENDIX E VARIANT SYNTAX

The syntax for a variant is as follows:



where:

<expression> is a valid expression as defined in Appendix C

<subpicture name> is any valid subpicture name

<text string> is an ASCII text string enclosed in double quotes

<value name> (R610) is any valid picture editor value or expression as defined in Appendix B

Example:

```
IF BOOL01 THEN SUB PIX5 ELSE IF BOOL02 THEN "OVERLIMIT"
```

```
IF BOOL05 THEN SUB PUMP ELSE SUB PUMP2
```

IF NOT BOOL02 THEN "FAILED" ELSE IF BOOL07 THEN "RUNNING"

```
IF BOOL01 THEN VAL AP100.PV ELSE VAL INT01 + 5
```



## APPENDIX F ENUMERATION CONSTANTS

An enumeration is a data type consisting of two or more states. Enumeration constants can be specified in an expression at any point where a constant is legal (refer to Appendix C for the syntax of expressions). In many cases, a name cannot be identified as an enumeration constant from the context of the expression. In those cases, syntax must be used to positively identify the character string as an enumeration constant.

The cases in which the enumeration constant cannot be identified are:

- The constant appears as a literal in a value.
- The constant appears as a parameter to a subpicture.
- The constant appears on the left-hand side of an expression.
- The constant appears on the right-hand side of an expression where the left-hand side has been typed as unknown.
- The constant conflicts with a local name.

In these cases, a special syntax must be used to positively identify the character string as an enumeration constant.

### Standard Enumerations

Standard Enumerations are supplied by Honeywell. The syntax for standard enumerations is

**ENUMERATION\_SET:ENUMERATION\_MEMBER**

where: **ENUMERATION\_SET** is the set name of the enumeration constant

**:** is a literal colon

**ENUMERATION\_MEMBER** is the member name of the enumeration constant.

If a standard enumeration is used in a generic subpicture as &A.Mode, the type is E: MODE or ENUM: MODE.

**Examples:** MODE: AUTO PTEXECST: ACTIVE

## Self-Defining Enumerations

Self-Defining Enumerations are user-defined text strings that describe states of Digital points and Flag points. The syntax for self-defining enumerations is

**SD:** ENUMERATION\_STRING

where: **SD:** is the literal string SD:

**ENUMERATION\_STRING** is any text string. Generally, the text string will be a valid string for a particular point parameter typed as SD\_ENUM.

If a Self-Defining Enumeration is used in a generic subpicture as &A.STOP, the type is SD\_ENUM.

**Examples:** SD:ON SD:STATE1 SD:STOP

## Custom Enumerations

Custom Enumerations are created with the CL compiler and are used in custom data segments to create custom parameters.

The source file would contain these lines to create the custom parameter DAY–

```
ENUMERATION WEEKDAY = SUN/MON/TUES/WED/THURS/FRI/SAT
CUSTOM
PARAMETER DAY: WEEKDAY
END CUSTOM
```

If a custom enumeration is used in a generic subpicture as &A.DAY, the type is E:WEEKDAY.

Examples:

- When used in an ambiguous reference, you must use  
ENUMERATION TYPE:ENUMERATION MEMBER. For example–  
IF WEEKDAY: SUN = POINT.DAY THEN SET RED
- No enumeration type is needed if the reference is written this way–  
IF POINT.DAY = SUN THEN SET RED

## APPENDIX G APPLICATION SUBPICTURES

### G.1 DESCRIPTION

Application Subpictures are created by Honeywell and cannot be built or changed with the Picture Editor. They are added to displays with the Add Subpicture command and they can be used to build a display that is itself used as a subpicture. Application Subpictures cannot be used in Free Format Logs.

Application Subpicture names are reserved names. The following are the standard Application Subpictures. Each is described in detail in the following sections.

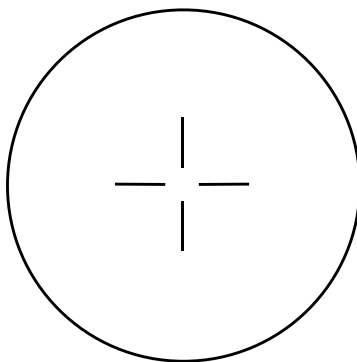
<u>Application Subpicture</u>	<u>Name</u>
G.1.1 Circle	CIRCLE
G.1.2 Quarter Circle	QUARTER
G.1.3 Pie Charts	PIEn (n=2-8)
G.1.4 Trend	TREND; TREND_AX
G.1.5 Ring	RING
G.1.6 Radar Chart, Fixed Axis	RADAR
G.1.7 Radar Chart, Variable Axis	RADAR1
G.1.8 Broken Line Graph	LGRAPH
G.1.9 Multi-Broken Line Graph	LGRAPHn (n=1-8)
G.1.10 Custom System Status Display Frame	SS_FRAME
G.1.11 Small Node Object Box	SS_PN_SM
G.1.12 Large Node Object Box	SS_PN_LG

### G.1.1 Circle

This Application Subpicture is a circle with the origin at its center (see Figure G-1).

When added to a display

- the size of the circle can be changed with the Scale Command.
- behavior of the circle can be changed with the Add Behavior or Add Condition commands.
- the circle assumes the current literal behavior of the display it is added to.



1757

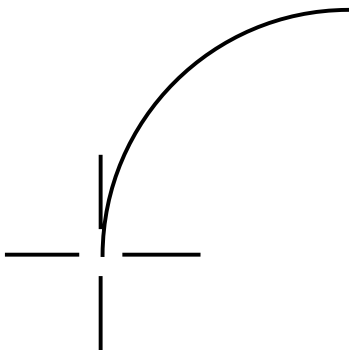
**Figure G-1 — Circle Application Subpicture**

### G.1.2 Quarter

This Application Subpicture is a quarter of a circle with the origin at its lower left end (see Figure G-2).

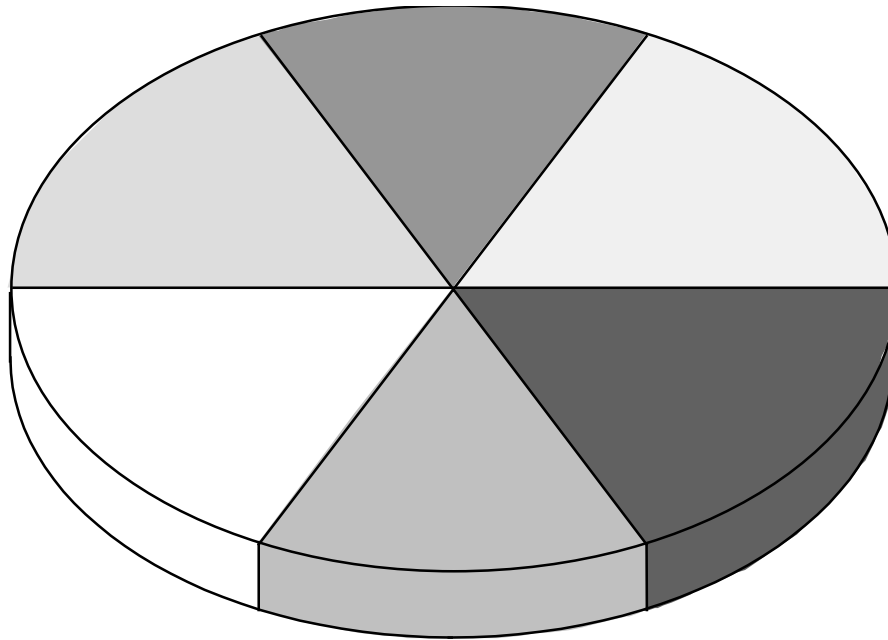
When added to a display

- the size of the subpicture can be changed, or it can be flipped horizontally or vertically with the Scale Command.
- two quarter circles can be aligned to form a half circle.
- behavior of the subpicture can be changed with the Add Behavior or Add Condition commands.
- the subpicture assumes the current literal behavior of the display it is added to.



**Figure G-2 — Quarter Application Subpicture**

1758



**Figure G-3 — Pie Chart Application with Six Slices (PIE6)**

2901



### G.1.3 Three Dimensional Pie Charts

Pie chart Application Subpictures are available with 2-through-8 slices (see Figure G-3). Specify the number of slices that you want by the suffix to the name PIE. PIE3, for example is displayed as a 3-slice pie chart, PIE8 is an 8-slice pie chart.

The size of each piece of the pie represents the ratio of the input parameter for that slice to the sum of all the input parameters for all slices.

#### Colors

Each slice of the pie has an associated color. The pie slices are colored as follows:

Slice 1 - Yellow	Slice 5 - Cyan
Slice 2 - Red	Slice 6 - White
Slice 3 - Green	Slice 7 - Blue
Slice 4 - Magenta	Slice 8 - Black (with white outline)

#### Specifications

When adding pie charts to a display, a screen form appropriate for the number of slices appears (see Figure G-4). The screen form requests parameters to describe the view of the pie and dimensions of each slice of the pie.

You must specify an expression or real number to define the angle of view (tilt angle) of the pie chart.

Thickness (edge height) of the pie is specified as a percentage of the size of the pie.

Enter values or expressions for each piece of the pie in the appropriate screen form boxes.

Date/Time	
<div style="border: 1px solid black; padding: 10px; margin: 10px auto; width: 80%;"> <p>Subpicture PIEn at xxxx, yyyy</p> <p>Enter Edge Height (real) <input style="width: 150px;" type="text"/></p> <p>Enter Tilt Angle (radians) <input style="width: 150px;" type="text"/></p> <p>Size of Piece 1 (real) <input style="width: 150px;" type="text"/></p> <p>Size of Piece 2 (real) <input style="width: 150px;" type="text"/></p> <p style="text-align: center;">⋮</p> <p>Size of Piece n (real) <input style="width: 150px;" type="text"/></p> </div>	
<div style="border: 1px solid black; padding: 5px;"> <div style="border-bottom: 1px solid black; padding-bottom: 5px;">ADD SUB PIEn</div> <div style="padding-top: 5px;">Enter Subpicture Information</div> </div>	

**Figure G-4 — Screen Form for Pie Chart of n Sections**

1759

## NOTE

At operating time, if the value of any slice is invalid or if the values for all slices are zero, a red X appears instead of the pie chart.

If the thickness value for the pie can increase at operating time, be sure to scale the pie chart such that the bounding area allows the size increase (see the Scale command elsewhere in this manual).

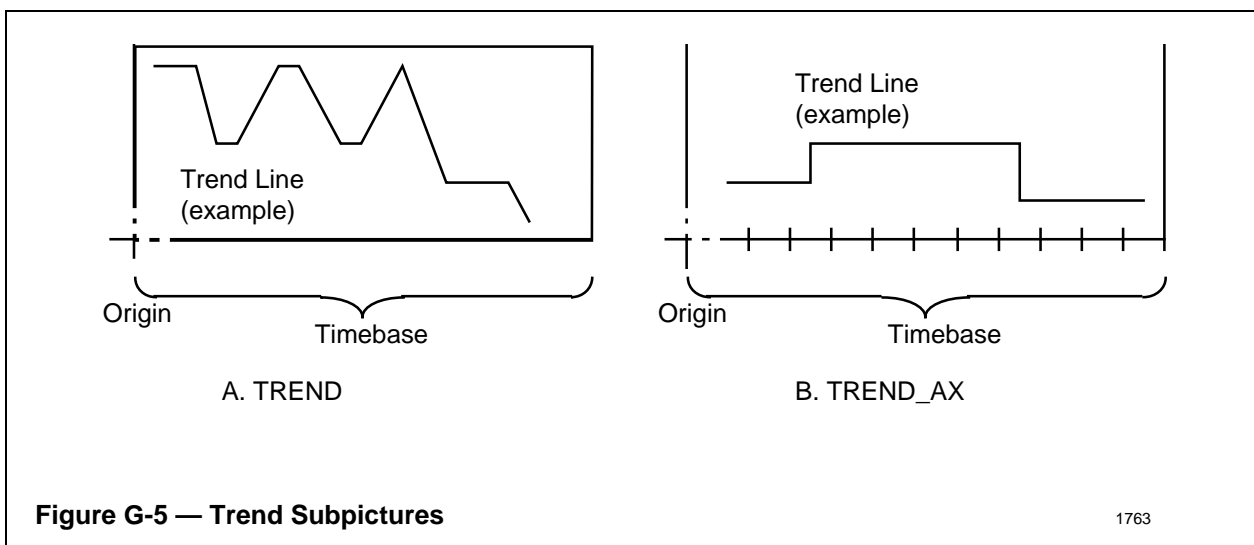
### G.1.4 Trend Subpictures

Trend subpictures provide a way to view trend lines in Custom Graphic displays. Each trend subpicture uses one Trend Record (see the *Actors Manual*, Section 13) and supports up to four trend lines (traces).

There are two Trend Subpictures: TREND\_AX and TREND. TREND\_AX presents a trend with axes; TREND presents a trend without axes, otherwise both subpictures function the same way (see Figure G-5).

Twelve Trend Records (named TREND01 through TREND12) are available, and each Trend Record should be used no more than once in any display. Therefore, up to twelve trends (each showing four trend lines) can be on the screen at the same time.

Trend Subpictures are used with the Trend Actors described in *Actors Manual*, and the Trend Collectors described in Appendix H. Trend Actors and Trend Collectors must specify the Trend Record ID number associated with the subpicture.



When a Trend subpicture is added to the display, the following screen form appears. TREND01 was entered as an example.

Subpicture TREND AX At XXX, YYY	
Trend Data Record	<input type="text" value="TREND01"/>

After the screen form information is entered, a dummy representation of the TREND or TREND\_AX subpicture appears in the display.

### G.1.5 Ring

This Application Subpicture is a variable radius circle with the origin at its center. The radius, color, and fill are specified by input parameters. The ring may contain a subpicture.

When you add the Ring Application Subpicture to a display at build time, a screen form appears. Figure G-6 illustrates the screen form with typical entries. The screen form requests parameters to describe the ring. Enter values or expressions in the appropriate screen form ports (see Specifications for details).

When the screen form parameters have been correctly typed in, press ENTER. The build time symbol for a ring appears at the selected coordinates. The same symbol is used for Rings with all options. See Figure G-7a. Note that:

- the size of the ring can be changed with the Scale Command.
- behavior of the ring cannot be changed with the Add Behavior or Add Condition commands.

Figures G7-b and G7-c illustrate the ring at operating time.

## Screen Form Parameter Specifications

**Color**—Table G-1 lists the choices for different colors and intensities of the subpicture. Entry is a Real number.

**Radius**—You must specify an expression or real number to define the radius of the Ring.

**Full**—The ring can appear as a circle, or as a circle filled with color. Any subpicture can be superimposed in the ring. Clearing options erase the previous ring drawing before updating; nonclearing options retain the previous image and draw the updated ring. Ring fill options are specified with an integer as follows—

Fill Integer	Fill Action
0	Ring with color fill; clearing
1	Ring only clearing
2	Ring with color fill; nonclearing
3	Ring only nonclearing

**Maximum Value**—Entry must be a Real number.

**Minimum Value**—Entry must be a Real number.

Date/Time

Subpicture RING at xxxx, yyyy

COLOR	(REAL)	7.0
RADIUS	(REAL)	A100.PV
FULL (0-FULL)	(INTEGER)	2
MAX VALUE	(REAL)	100.0
MIN VALUE	(REAL)	0.0

ADD SUB RING

Enter Subpicture Information

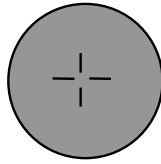
**Figure G-6 — Screen Form for Ring**

5522

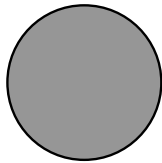
**Table G-1 — Color Specifications**

Color	Full Intensity	Half Intensity
Black	0.0	10.0
Cyan	1.0	11.0
Yellow	2.0	12.0
Red	3.0	13.0
Green	4.0	14.0
Blue	5.0	15.0
Magenta	6.0	16.0
White	7.0	17.0 (default)

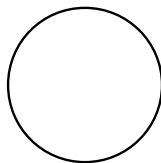
Note that the color parameter must be a real number.



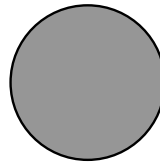
a. Ring symbol at build time;  
cross hair indicates origin



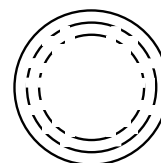
Fill = 0



Fill = 1

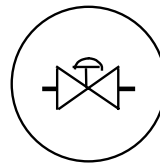
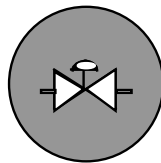


Fill = 2



Fill = 3  
(Changing  
Values)

b. Rings at operate time



c. Rings with superimposed subpicture

**Figure G-7 — Ring Application Subpicture**

5523

## NOTE

At operating time, if the value of any parameter is invalid (i.e., NaN), a red X appears instead of the ring.

### **G.1.6 Radar Chart (Fixed Axis)**

This Application Subpicture is a chart with multiple axes. The color of the graph, color of the axes, and number of axes are specified as parameters on the screen form. The angle between the axis is fixed and depends on the number of axes. The angle is  $120^\circ$  if three axes are specified,  $90^\circ$  if four axes are specified, etc. (see Figure G-9b).

When you add the Radar Application Subpicture to a display at build time, a screen form appears. Figure G-8 illustrates the screen form with typical entries. The screen form requests parameters to describe the chart. Enter values or expressions in the appropriate screen form ports (see Screen Form Specifications for details).

When the screen form parameters have been correctly entered, press the ENTER key. The build time symbol for a radar chart appears at the selected coordinates. The same symbol is used for all Radar Charts with fixed axis. The origin is at its center. See Figure G-9a. Note that:

- the size of the chart can be changed with the Scale Command.
- behavior of the chart cannot be changed with the Add Behavior or Add Condition commands.

#### **Screen Form Parameter Specifications**

Colors—Table G-1 (see Ring) lists the choices for different colors and intensities of the graph or axis. Entry is a Real number.

Max Value—Entry must be a Real number.

Min Value—Entry must be a Real number.

Number of Axis—You must specify a number 3 through 10.

Clear Flag—if this integer is 0, the previous trace on the graph is erased (at operating time) and the new trace is drawn. If this integer is 1, the graph is presented without axes, the new trace is drawn, and earlier traces are retained (see Figure G-9c).

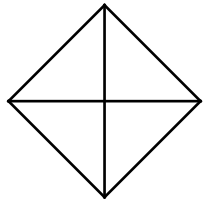
Parameters—Enter three to ten expressions. Enter 0.0 in all unused parameter ports.



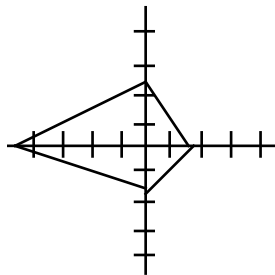
Date/Time																																								
<div style="border: 1px solid black; padding: 10px; margin: 10px;"> <p>Subpicture RADAR at xxxx, yyyy</p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 40%;">COLOR OF GRAPH</td> <td style="width: 10%;">(REAL)</td> <td style="width: 50%;"><input style="width: 90%;" type="text" value="2.0"/></td> </tr> <tr> <td>COLOR OF AXIS</td> <td>(REAL)</td> <td><input style="width: 90%;" type="text" value="7.0"/></td> </tr> <tr> <td>MAX VALUE</td> <td>(REAL)</td> <td><input style="width: 90%;" type="text" value="100.0"/></td> </tr> <tr> <td>MIN VALUE</td> <td>(REAL)</td> <td><input style="width: 90%;" type="text" value="0.0"/></td> </tr> <tr> <td>NUMBER OF AXIS (3 ~ 10)</td> <td></td> <td><input style="width: 90%;" type="text" value="3"/></td> </tr> <tr> <td>CLEAR FLAG (0 = CLEAR:INTEGER)</td> <td></td> <td><input style="width: 90%;" type="text" value="0"/></td> </tr> <tr> <td>PARAMETER 1</td> <td>(REAL)</td> <td><input style="width: 90%;" type="text" value="A100.PV"/></td> </tr> <tr> <td>PARAMETER 2</td> <td>(REAL)</td> <td><input style="width: 90%;" type="text" value="A101.PV"/></td> </tr> <tr> <td>PARAMETER 3</td> <td>(REAL)</td> <td><input style="width: 90%;" type="text" value="A102.PV"/></td> </tr> <tr> <td style="text-align: center;">⋮</td> <td></td> <td style="text-align: center;">⋮</td> </tr> <tr> <td>PARAMETER 9</td> <td>(REAL)</td> <td><input style="width: 90%;" type="text" value="0.0"/></td> </tr> <tr> <td>PARAMETER 10</td> <td>(REAL)</td> <td><input style="width: 90%;" type="text" value="0.0"/></td> </tr> <tr> <td colspan="3" style="height: 20px; border: 1px solid black; margin-top: 10px;"></td> </tr> </table> </div>		COLOR OF GRAPH	(REAL)	<input style="width: 90%;" type="text" value="2.0"/>	COLOR OF AXIS	(REAL)	<input style="width: 90%;" type="text" value="7.0"/>	MAX VALUE	(REAL)	<input style="width: 90%;" type="text" value="100.0"/>	MIN VALUE	(REAL)	<input style="width: 90%;" type="text" value="0.0"/>	NUMBER OF AXIS (3 ~ 10)		<input style="width: 90%;" type="text" value="3"/>	CLEAR FLAG (0 = CLEAR:INTEGER)		<input style="width: 90%;" type="text" value="0"/>	PARAMETER 1	(REAL)	<input style="width: 90%;" type="text" value="A100.PV"/>	PARAMETER 2	(REAL)	<input style="width: 90%;" type="text" value="A101.PV"/>	PARAMETER 3	(REAL)	<input style="width: 90%;" type="text" value="A102.PV"/>	⋮		⋮	PARAMETER 9	(REAL)	<input style="width: 90%;" type="text" value="0.0"/>	PARAMETER 10	(REAL)	<input style="width: 90%;" type="text" value="0.0"/>			
COLOR OF GRAPH	(REAL)	<input style="width: 90%;" type="text" value="2.0"/>																																						
COLOR OF AXIS	(REAL)	<input style="width: 90%;" type="text" value="7.0"/>																																						
MAX VALUE	(REAL)	<input style="width: 90%;" type="text" value="100.0"/>																																						
MIN VALUE	(REAL)	<input style="width: 90%;" type="text" value="0.0"/>																																						
NUMBER OF AXIS (3 ~ 10)		<input style="width: 90%;" type="text" value="3"/>																																						
CLEAR FLAG (0 = CLEAR:INTEGER)		<input style="width: 90%;" type="text" value="0"/>																																						
PARAMETER 1	(REAL)	<input style="width: 90%;" type="text" value="A100.PV"/>																																						
PARAMETER 2	(REAL)	<input style="width: 90%;" type="text" value="A101.PV"/>																																						
PARAMETER 3	(REAL)	<input style="width: 90%;" type="text" value="A102.PV"/>																																						
⋮		⋮																																						
PARAMETER 9	(REAL)	<input style="width: 90%;" type="text" value="0.0"/>																																						
PARAMETER 10	(REAL)	<input style="width: 90%;" type="text" value="0.0"/>																																						
<div style="border: 1px solid black; padding: 2px;"> <div style="display: flex; justify-content: space-between; padding: 2px;"> <span>ADD SUB RADAR</span> <span></span> </div> <div style="border: 1px solid black; padding: 2px; margin-top: 2px;"> Enter Subpicture Information </div> </div>																																								

**Figure G-8 — Radar Chart Screen Form**

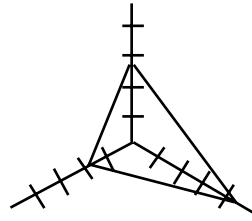
5524



a. Build time symbol

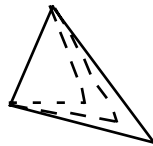


b. Four Axis



Three Axis

Operator Displays



c. Three Axis Radar Chart, Clear Flag = 1 (changing values)

**Figure G-9 — Radar Chart (Fixed Axis) Displays**

5525

## NOTE

At operating time, if the value of any parameter (except a graph data parameter) is invalid (i.e., NaN), a red X appears instead of the chart. If the value of any of the graph data parameters (Parameters 1- 10) is invalid, it is treated as if its value is 0.

### G.1.7 RADAR1 Chart (Variable Axis)

This Application Subpicture is a chart with multiple variable-angle axis. The color of the graph, number of axes and color of the axes are specified as parameters on the screen form. You can specify three to 45 axis. The angle between the axes depends on real data in a custom data segment array.

When you add the RADAR1 Application Subpicture to a display at build time, a screen form appears. Figure G-10 illustrates the RADAR1 screen form with typical entries. The screen form requests parameters to describe the chart. Enter values or expressions in the appropriate screen form ports (see Screen Form Parameter Specifications for details).

When the screen form parameters have been correctly typed in, press the ENTER key. The build time symbol for a radar chart appears at the selected coordinates. The same symbol is used for all Radar Charts with variable axis. The origin is at its center. See Figure G-11a. Note that:

- the size of the chart can be changed with the Scale Command.
- behavior of the chart cannot be changed with the Add Behavior or Add Condition commands.

#### Screen Form Parameter Specifications

Colors—Table G-1 lists the choices for different colors and intensities of the graph or axis. Entry is a Real number.

Max Value—Entry must be a Real number.

Min Value—Entry must be a Real number.

Axis Draw Flag—if this integer is 0, the immediate area needed to draw the graph is erased (at operating time) as each new trace is drawn (see Figure G-11b). If this integer is greater than 0, the axes are not presented and as the graph is updated, each new trace is drawn and earlier traces are retained (see Figure G-11c).

Axis Data—you must specify a custom data segment that contains an array of real axis data. These parameters determine the angle of the axis.

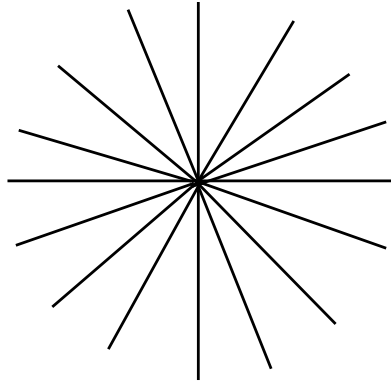
Number of Data—Enter an integer from 3 through 45 to specify the number of axis.

Parameters—you must specify a custom data segment that contains an array of real parameter data. These parameters provide the graph data.

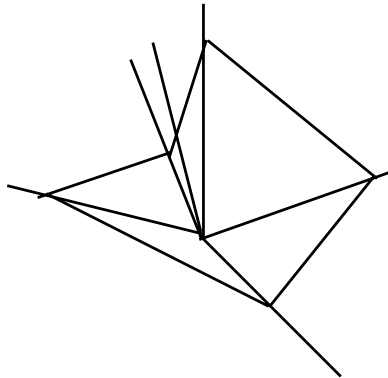
Date/Time	
Subpicture RADAR1 at xxxx, yyyy	
COLOR OF GRAPH (REAL)	<input type="text" value="7.0"/>
MAX VALUE (REAL)	<input type="text" value="100.0"/>
MIN VALUE (REAL)	<input type="text" value="0.0"/>
AXIS DRAW FLAG (0 = DRAW)	<input type="text" value="0"/>
COLOR OF AXIS (REAL)	<input type="text" value="17.0"/>
AXIS DATA (ARRAY OF REAL 0 - 360)	<input type="text" value="YY.D7"/>
NUMBER OF DATA (INTEGER)	<input type="text" value="6"/>
PARAMETER (ARRAY OF REAL)	<input type="text" value="YY.D8"/>
<input type="text"/>	
ADD SUB RADAR1	
Enter Subpicture Information	

Figure G-10 — RADAR1 Chart Screen Form

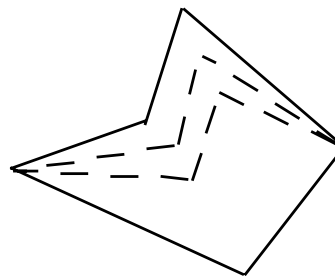
5526



a. Build time Symbol



b. With Axis (Draw Flag = 0)



c. Draw Flag = 1 (changing values)

Operating display

**Figure G-11 — RADAR1 Chart (Variable Axis) Displays**

5527

## NOTE

At operating time, if the value of any parameter (except for graph data), is invalid (i.e., NaN), a red X appears instead of the chart. Invalid graph data is treated as if its value is 0.

### G.1.8 LGRAPH

This Application Subpicture is a single broken line graph. Each line segment can be plotted by a different parameter.

When you add the LGRAPH Application Subpicture to a display at build time, a screen form appears where you can enter parameters to describe the graph (see Screen Form Parameter Specifications for details). Figure G-12 illustrates the LGRAPH screen form with typical entries.

When the screen form parameters have been correctly typed in, press the ENTER key. A build time symbol for the LGRAPH chart appears. The origin is at the left end of the line as shown in Figure G-13a.

Note that:

- the size of the chart can be changed with the Scale Command.
- behavior of the chart cannot be changed with the Add Behavior or Add Condition commands.

#### Screen Form Parameter Specifications

**Number of Plot**—maximum number of segments per line. Range is 2 to 10. Note that the horizontal space for each segment is determined by dividing the horizontal space available by the number of plots.

**Vertical Axis Max Value**—the maximum Y value that will be charted. Entry is a real number.

**Vertical Axis Min Value**—the minimum Y value that will be charted. Entry is a real number.

**Color Info**—This entry specifies the color choice and intensity for the graph (see Table G-1). You can enter a real number directly in the screen form, or specify a custom data segment (not an array) that contains the data.

**Plotmark Info**—Real data entries specify various small symbols that appear at points of inflection on the chart (see Table G-2). When 0.0 is specified, no plotmark symbol is displayed. Plotmark information can be specified directly in the screen form or in a custom data segment (not an array).

**Multiple Draw** —if this integer is 0, the immediate area required by the graph is erased (including the previous trace) as the graph is updated. If this integer is 1, the area is not erased as the graph is updated, and earlier traces or other information in the area can be retained.

**Parameter**—you can specify up to ten parameters. Each parameter specifies a point on the line. Entry is real data. Unused parameter entries must be set to 0.0.

Date/Time

Subpicture LGRAPH at xxxx, yyyy

Number of Plot (INTEGER)

Vertical Axis Max Value (REAL)

Vertical Axis Min Value (REAL)

Color Info. (REAL)

Plotmark Info. (REAL)

Multiple draw (1=YES :INTEGER)

Parameter NO.1 (REAL)

Parameter NO.2 (REAL)

⋮

Parameter NO.9 (REAL)

Parameter NO.10 (REAL)

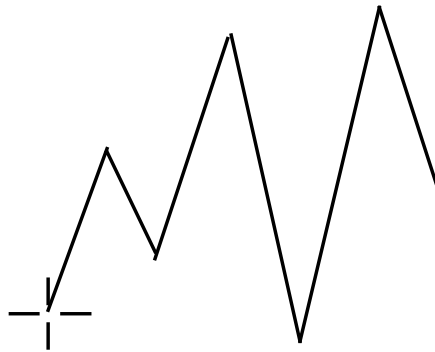
ADD SUB LGRAPH

Enter Subpicture Information

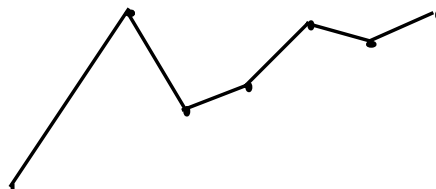
**Figure G-12 — LGRAPH Screen Form**
5530

Table G-2 Plotmarks

Entry (Real)	0.0	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0
Plotmark	None	□	△	◇	■	▲	◆	×	+



a. Build time symbol; origin at the cross hair.



b. Operating time LGRAPH display.

**Figure G-13 — LGRAPH Displays**

5531

## NOTE

At operating time, if a color value is invalid (i.e., NaN), the color of the graph changes to half intensity white. If axis data is invalid (i.e., NaN), neither the dot nor the lines that connect the dot appear.



### G.1.9 LGRAPHn

This Application Subpicture is a broken line graph with one to eight lines. The data and the color of the lines are specified in custom data segments.

When you add the LGRAPHn Application Subpicture to a display at build time, you must specify the number of lines that you want with the suffix "n", where n = 1 through 8. For example, use LGRAPH6 to build a six-line graph. A screen form appears where you can enter parameters to describe the graph (see Screen Form Parameter Specifications for details). Figure G-14 illustrates the LGRAPH screen form with typical entries.

When the screen form parameters have been correctly typed in, press the ENTER key. A build time symbol for the LGRAPHn chart appears. The number of lines in the build time symbol depends on "n". The origin is at the left end of the lowest line. See Figure G-15a. Note that:

- the size of the chart can be changed with the Scale Command.
- behavior of the chart cannot be changed with the Add Behavior or Add Condition commands.

#### Screen Form Parameter Specifications

Number of Plot—maximum number of segments per line. Range is 2 to 20.

Vertical Axis Max Value—the maximum Y value that will be charted. Entry is a real number.

Vertical Axis Min Value—the minimum Y value that will be charted. Entry is a real number.

Color Info—Entry specifies a custom data segment array that contains the color choice and intensity for the graph. Array data type is Real (see Table G-1).

Plotmark Info—Entry specifies a custom data segment array. Real data entries in the array specify various small symbols that appear at points of inflection on the chart (see Table G-2). When 0.0 is specified, no plotmark symbol is displayed.

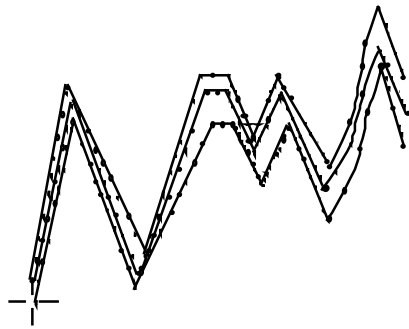
Multiple Draw—If this integer is 0, the immediate area required by the graph is erased (including the previous trace) as the graph is updated. If this integer is 1, the area is not erased as the graph is updated, and earlier traces or other graphics can be retained. Figure G-15c illustrates an XYLOT1 and LGRAPH2 overlay using the multiple draw option.

Data No. 1 - Data No. n—For each entry, you must specify a custom data segment that contains an array of real data. The maximum number of elements in each data array is 20.

Date/Time	
<div style="border: 1px solid black; padding: 10px; margin: 10px;"> <p>Subpicture LGRAPHn at xxxx, yyyy</p> <p>Number of Plot (INTEGER) <input style="width: 150px;" type="text" value="10"/></p> <p>Vertical axis max value (REAL) <input style="width: 150px;" type="text" value="100.0"/></p> <p>Vertical axis min value (REAL) <input style="width: 150px;" type="text" value="0.0"/></p> <p>Color Info. (ARRAY OF REAL) <input style="width: 150px;" type="text" value="YY.COLOR"/></p> <p>Plotmark Info. (ARRAY OF REAL) <input style="width: 150px;" type="text" value="YY.TMARK"/></p> <p>Multiple draw (1=YES :INTEGER) <input style="width: 150px;" type="text" value="0"/></p> <p>Data NO.1 (ARRAY OF REAL) <input style="width: 150px;" type="text" value="YY.D1"/></p> <p>Data NO.2 (ARRAY OF REAL) <input style="width: 150px;" type="text" value="YY.D2"/></p> <p style="text-align: center;">⋮</p> <p>Data NO.n (ARRAY OF REAL) <input style="width: 150px;" type="text"/></p> <div style="border: 1px solid black; height: 20px; width: 500px; margin-top: 10px;"></div> </div>	
<div style="border: 1px solid black; display: flex; justify-content: space-between; align-items: center;"> <span>ADD SUB LGRAPHn</span> <input style="width: 50px;" type="text"/> </div>	
<div style="border: 1px solid black; padding: 2px;">Enter Subpicture Information</div>	

**Figure G-14 — LGRAPHn Screen Form**

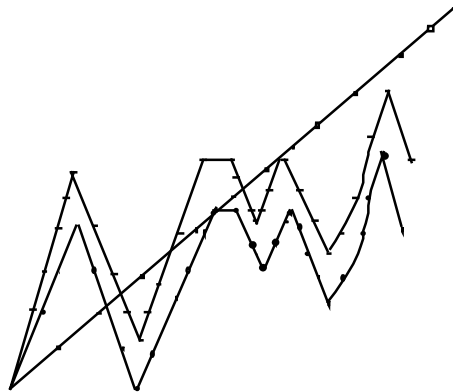
5532



a. Typical build time symbol with origin at the cross hair.



b. Operating display (LGRAPH2).



c. Overlay of XYPLOT1 and LGRAPH2 (Mult. draw = 1).

**Figure G-15 — LGRAPHn Chart Displays**

5533

## NOTE

At operating time, if a color value is invalid (i.e., NaN), the color of the graph changes to half intensity white. If axis data is invalid (i.e., NaN), neither the dot nor the lines that connect the dot appear.

### G.1.10 Custom Status Display Frame

Application subpicture **SS\_FRAME** is the template for a custom System Status display page (see Figure G-16). It must be added at the lower left hand corner (coordinates 0,0) as you build each System Status display page. You can then add small or large Node object boxes or combinations (see Figure G-18 and G-19) anywhere in the blank area. The top 4 lines and lower 5 lines of the display are reserved. You can build five custom System Status displays for each console number.

When adding subpicture **SS\_FRAME** to a display you are prompted to enter the page number of this custom status display page. The valid range is 1 - 5.

After a custom System Status display page is built, compile it to a user volume using the file name **Cnn\_SSp** where—

nn = the console number: 01 - 10

p = the page number of the custom status display (1 - 5).

Example: filename = C01\_SS1 for console 01, page 1.

You must copy the object files **Cnn\_SSp.DO** to volume &DSY (and &DS2 if used), otherwise the standard System Status display is invoked at operating time.

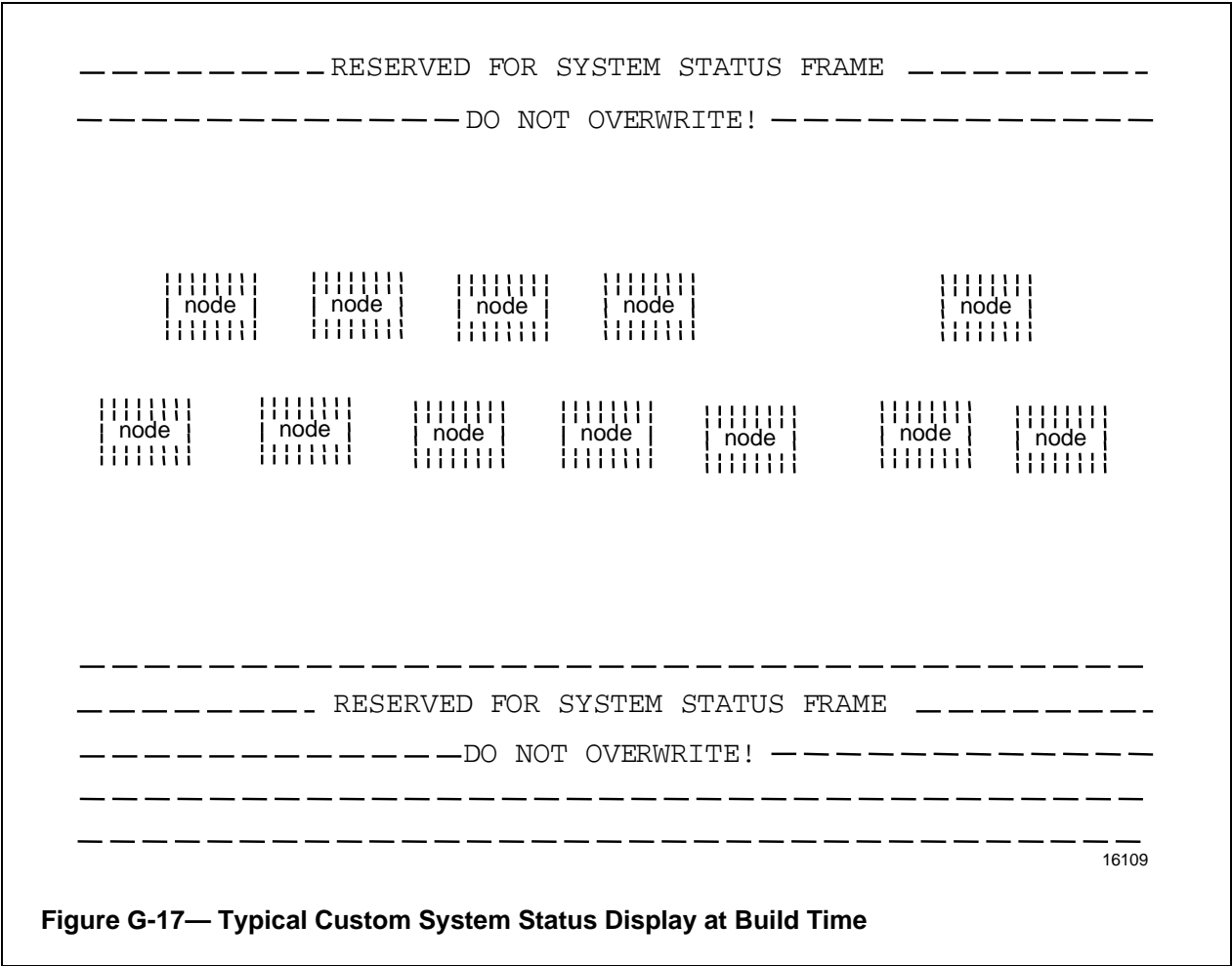
----- RESERVED FOR SYSTEM STATUS FRAME -----  
----- DO NOT OVERWRITE! -----

-----  
----- RESERVED FOR SYSTEM STATUS FRAME -----  
----- DO NOT OVERWRITE! -----  
-----  
-----

16107

**Figure G-16— SS\_Frame**

Figure G-17 illustrates how a simple custom System Status display appears at build time. Refer to the Easy Reload subsection of the Process Operations Manual for information on use of custom System Status displays.



**Figure G-17— Typical Custom System Status Display at Build Time**

### G.1.10.1 General Notes on Building Custom System Status Displays

SS\_Frame can be added to any schematic however it is not recommended for use in applications other than the System Status displays.

Node object boxes can be added anywhere on the SS\_Frame display, however we recommend that you only add them in the designated region.

Node object boxes can be overlapped, however it is not recommended.

Node object boxes can be built and placed on a character boundary or on a pixel boundary.

Other Picture Editor objects such as lines, text, and targets can be added in the blank area of SS\_Frame. The *Picture Editor Form Instructions* manual contains an example that shows how to build a simple custom System Status display.

Note that you must perform an Area Change (typically to/from the same Area) or reload the Universal Station to get a new custom System Status display into US memory.

#### NOTE

When building a custom System Status display, you may need to move or delete an Object that has been added to the SS\_Frame subpicture. Selecting an object also selects the frame. To deselect the frame part, just touch the frame area twice (or place the cursor on the frame and press the Select key twice to deselect the frame if the US doesn't have a touchscreen).

### G.1.11 Small Node Object Box

Application subpicture **SS\_PN\_SM** is the small node object box used in a custom System Status display. It is 6 characters wide and 3 characters high. At operating time, status words such as **WARNING** is inserted in the small box in an abbreviated format such as **WARN**. Up to 13 small boxes will fit across the display. Up to 64 node boxes can appear on a page. You do not have to add a box for every configured node.

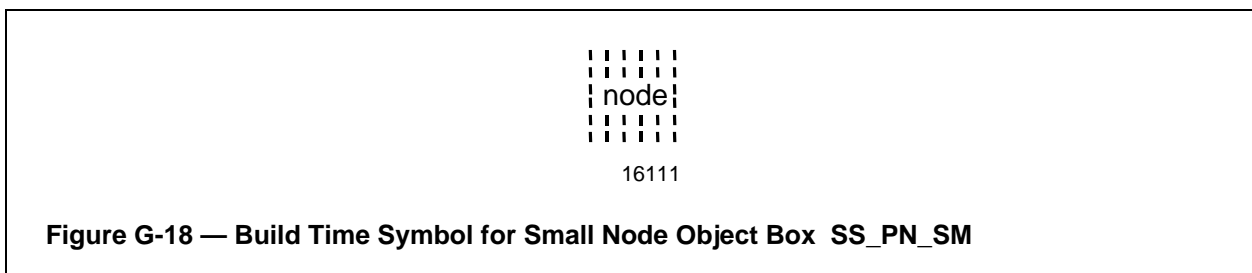
You can add small node object box subpictures anywhere within the blank area of the custom System Status display frame (see Figures G-16 and G-17).

After entering the add subpicture command and subpicture coordinates, the message: **Enter Node Number** appears.

Subpicture SS\_PN\_SM At XXX, YYY

Enter Node Number

Enter the desired one or two character LCN node number and press the Enter key.





### G.1.12 Large Node Object Box

Application subpicture **SS\_PN\_LG** is the large node object box used in a custom System Status display. It is 8 characters wide and 3 characters high. Up to 10 large boxes will fit across the display without overlap.

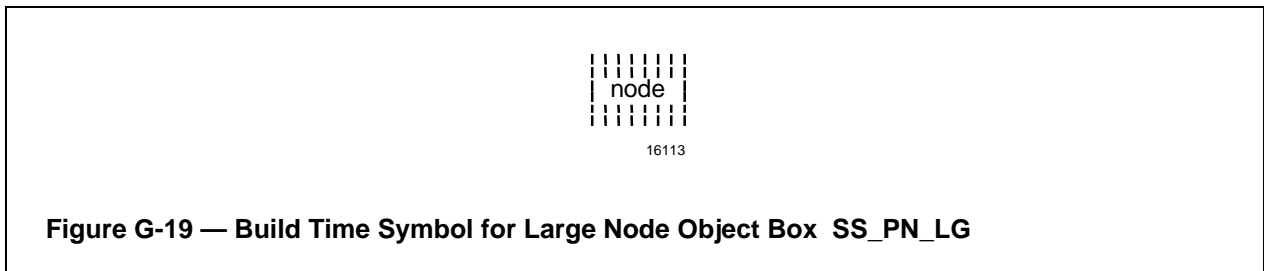
You can add large node object box subpictures anywhere within the blank area of the custom System Status display frame (see Figure G-16 and G-17). You do not have to add a box for every configured node.

After entering the add subpicture command and subpicture coordinates, the message: **Enter Node Number** appears.

Subpicture SS\_PN\_LG At XXX, YYY

Enter Node Number

Enter the desired one or two character LCN node number and press the Enter key.





## APPENDIX H COLLECTORS

### COLLECTORS, ALPHABETICALLY BY CALL NAME

Call Name	Description	Section
\$ADDBCNT	Returns the number of alarms for the assigned DDBs that contain the Annunciator Group Title and the alarm status category	H.7.2.1
\$ADDBSTS	Returns the composite alarm status for an assigned DDB that contains an annunciator group title	H.7.1.1
\$ANNCNT	Returns the number of alarms in the specified alarm status category for the specified Annunciator group title	H.7.2.2
\$ANNSTS	Returns the comp. alarm status of a specified annunciator group	H.7.1.2
\$AREACNT	Returns the number of alarms in a specified alarm status category for the local area	H.7.2.3
\$AREASTS	Returns the composite alarm status of the local area	H.7.1.3
\$KEYLEVL	Returns the status of the US Keylock Level	H.8.1
\$PDDBCNT	Returns the number of alarms for the assigned DDBs that contain the Primmod name and the alarm status category	H.7.2.4
\$PDDBSTS	Returns the composite alarm status for an assigned DDB that contains a Primmod name	H.7.1.4
\$PNTCNT	Returns the number of alarms in a specified alarm status category for the specified point	H.7.2.5
\$PNTSTS	Returns the composite alarm status of a specified point	H.7.1.5
\$PRIMCNT	Returns the number of alarms in a specified alarm status category for the specified Primmod name	H.7.2.6
\$PRIMSTS	Returns the composite alarm status of a specified Primmod name	H.7.1.6
\$QNAME	Returns the name of the last Doc. Tool query from this US	H.9.1.1
\$QTIME	Returns the date/time of the last Doc. Tool query from this US	H.9.1.2
\$QSTATUS	Returns the status of the last Doc. Tool query from this US	H.9.1.3
\$UNITCNT	Returns the number of alarms in a specified alarm status category for the specified unit	H.7.2.7
\$UNITSTS	Returns the composite alarm status of a specified unit	H.7.1.7
ACKSTAT	Returns acknowledge state for point alarm	H.2
DAY_S	Reports status of averaged-value, n days offset	H.4.3
DAY_T	Returns date/time stored with variable, n days offset	H.4.2
DAY_V	Returns Max/Min/Daily Avg/Sum value of variable, n days offset	H.4.1

<b>Call Name</b>	<b>Description</b>	<b>Section</b>
HMDAY	Returns daily average value for n days back	H.5.1
HMHOUR	Returns hourly average value for n hours back	H.5.1
HMMIN	Returns current minute (snapshot) value for n minutes back	H.5.1
HMMONTH	Returns monthly average value for n months back	H.5.1
HMSHIFT	Returns shift average for n shifts back	H.5.1
HMUSR_AV	Returns user average value for n periods back	H.5.1
HOUR_S	Reports status of averaged-value, n hours offset	H.4.3
HOUR_T	Returns date/time stored with variable, n hours offset	H.4.2
HOUR_V	Returns Max/Min/Average/Sum value, n hours offset	H.4.1
I_DAY_S	Reports status of averaged-value, n days offset	H.4.3
I_DAY_T	Returns date/time stored with variable, n days offset	H.4.2
I_DAY_V	Returns Max/Min/Daily Avg/Sum value of variable, n days offset	H.4.1
I_HMDAY	Returns daily average value for n days back	H.5.1
I_HMHOUR	Reports hourly average value, n hours offset	H.5.1
I_HMMIN	Returns current minute (snapshot) value for n minutes back	H.5.1
I_HMMON	Returns monthly average value for n months back	H.5.1
I_HMSHFT	Returns shift average for n shifts back	H.5.1
I_HMUSR	Returns user average value for n periods back	H.5.1
I_HOUR_S	Returns status of hourly averaged value, n hours offset	H.4.3
I_HOUR_T	Returns date/time stored with variable, n hours offset	H.4.2
I_HOUR_V	Returns Max/Min/Hourly Average/Sum value, n hours offset	H.4.1
I_MIN_T	Returns date/time stored with variable, n minutes offset	H.4.2
I_MIN_V	Returns snapshot of variable, n minutes offset	H.4.1
I_MON_S	Reports status of averaged-value, n months offset	H.4.3
I_MON_T	Returns date/time stored with variable, n months offset	H.4.2
I_MON_V	Returns Max/Min/Monthly Average/Sum value, n months offset	H.4.1
I_SHFT_S	Reports status of averaged-value, n shifts offset	H.4.3
I_SHFT_T	Returns date/time stored with variable, n shifts offset	H.4.2
I_SHFT_V	Returns Max/Min/Shift Average/Sum value, n shifts offset	H.4.1
I_USER_S	Reports status of averaged-value of variable	H.4.3
I_USER_T	Returns date/time stored with variable	H.4.2
I_USER_V	Returns Max/Min/User Average/Sum value of variable	H.4.1
MINUTE_T	Returns date/time stored with variable, n minutes offset	H.4.2
MINUTE_V	Returns snapshot value of variable, n minutes offset	H.4.1
MONTH_S	Reports status of averaged-value, n months offset	H.4.3
MONTH_T	Returns date/time stored with variable, n months offset	H.4.2
MONTH_V	Returns Max/Min/Monthly Average/Sum value, n months offset	H.4.1
SHIFT_S	Reports status of averaged-value, n shifts offset	H.4.3
SHIFT_T	Returns date/time stored with variable, n shifts offset	H.4.2
SHIFT_V	Returns Max/Min/Shift Average/Sum value, n shifts offset	H.4.1
SYS_TIME	Returns system time or date	H.3

<b>Call Name</b>	<b>Description</b>	<b>Section</b>
TMDAY	Returns day date/time as of n days back	H.5.2
TMHOUR	Returns hour date/time as of n hours back	H.5.2
TMMIN	Returns minute date/time as of n minutes back	H.5.2
TMMONTH	Returns month date/time as of n months back	H.5.2
TMSHIFT	Returns shift date/time as of n shifts back	H.5.2
TMUSR_AV	Returns user average date/time as of n periods back	H.5.2
TR_CLINE	Returns trend center line date and time	H.6.1
TR_COLOR	Returns trend trace color	H.6.2
TR_DATA	Returns data source for the trace being trended	H.6.3
TR_EUDSC	Returns the engineering units descriptor for the given trend id and trace number	H.6.4
TR_HLSTS	Returns status indicating if hair line-cursor is on a collected value	H.6.16
TR_HLVAL	Returns the nearest numeric value to the hair-line cursor position	H.6.17
TR_LTIME	Returns date/time for the cursor position	H.6.18
TR_NAME	Returns name of variable being trended	H.6.5
TR_PARAM	Returns the parameter name for the given trend id and trace number	H.6.6
TR_POINT	Returns the point name for the given trend id and trace number	H.6.7
TR_PTDESC	Returns the point description for the given trend id and trace number	H.6.8
TR_RNGHI	Returns high-range value for trace	H.6.9
TR_RNGLO	Returns low-range value for trace	H.6.10
TR_RTVAL	Returns the current (real time) value for the given trend id and trace number	H.6.11
TR_SCLHI	Returns high-scale value being used	H.6.12
TR_SCLLO	Returns low-scale value being used	H.6.13
TR_SCRLL	Returns scroll date and time	H.6.14
TR_TIME	Returns the time base used for the trend	H.6.15
USER_S	Reports status of averaged-value of variable	H.4.3
USER_T	Returns date/time stored with variable	H.4.2
USER_V	Returns Max/Min/User Average/Sum value of variable	H.4.1

## H.1 INTRODUCTION TO COLLECTORS

### H.1.1 Collector Concepts

Collectors provide a way to access data that would not otherwise be available for use in custom graphic displays and Free Format Logs. Generally, collectors can be referenced in any expression entered through the Picture Editor. All collectors are functions that return values. The expressions use the value returned by the collector.

**Collector Entry**—Enter the Collector statement in the “expression” field for an Add Value command. You can enter the entire collector statement and, in some cases as described later, you only need to type the collector name and press the Enter key. A screen form appears with ports for the parameters.

All of the collectors are created by Honeywell, and collector names are reserved names.

## H.2 ACKNOWLEDGE STATE COLLECTOR

This collector is used to return the Acknowledge State of a point.

**Calling Sequence**— ACKSTAT(Point ID)

It returns an enumeration set. The values of this set are

- NOALARM—No alarm currently on the point
- UNAKALRM—The point has one or more unacknowledged alarms.
- AKDALRM—The point has one or more acknowledged alarms.

The ACKSTAT collector can be used in the expression for a Value; for example

Expression    ACKSTAT(A100)

At operating time, one of the three values of the set appears on the screen.

This collector can also be used in Variant and Conditional Behavior statements; for example

```
IF ACKSTAT(A100)=UNAKALRM  
THEN SUBPICTURE REDPUMP
```

or

```
IF ACKSTAT(A100)=NOALARM  
THEN SET GREEN
```

ACKSTAT alarms are acknowledged by pressing the ACK key on the keyboard. Alarms are acknowledged at all stations in the console that are assigned to the same area as the station that acknowledged the alarm.

#### CAUTION

Ackstat can be used in a schematic or in an overlay to that schematic. There is a limit of 511 Ackstat collectors per schematic or overlay. Exceeding this limit causes an error at compile time.

References to the Ackstat collector in a picture no longer need to be contained in the same collection group or at the same collection rate, but the number of collection groups and rates should be kept at a minimum to conserve memory.

Do not use ACKSTAT through a Network Gateway.

### H.3 SYSTEM TIME/DATE COLLECTOR

This collector returns the current system time for use in a picture or a Free Format Log.

**Calling Sequence**— SYS\_TIME.

The returned value can be either a time or date depending on the format that was assigned when the collector was added (see Date/Time formats in Appendix A to this manual). The default format is TIMEHH:MM SENDTIME.

**Example**— Two values are added to a Free Format Log. SYS\_TIME is entered as the expression for both. When the system prompts for the format, Value 1 is given the format DATEMM-DD-YYENDDATE and the format TIMEHH:MM AMENDTIME is assigned to Value 2. At operating time the current date is displayed for Value 1 (e.g., 03-15-86 and the current time is displayed for Value 2 (e.g., 07:10 AM).

### H.4 HISTORY COLLECTORS

Two broad categories of collectors are described in this subsection—

In one case, the variable is specified directly in the collector statement. These are referred to as direct collectors. Direct collectors can be used in either Free Format Logs or pictures.

In the other case, the collectors allow indirect reference to points and parameters in local/global Display Database (DDB) variables. These are referred to as indirect collectors and can only be used in pictures. Indirect collectors provide a way to build general purpose pictures where the variable/entity can be specified at operating time. Indirect collectors are preceded by I\_. DDB variables are described in the *Actors Manual*, Appendix A.

Both collector types use the same parameters in the collector statement and return the same information. Starting with software release 500, all of the history collectors (including those in subsection H.5) can be used in pictures.

If you are building new pictures or Free Format Logs that require history collectors, use those described in this subsection. If you have existing Free Format Logs that contain the original HM/TM collectors described in subsection H.5, you are encouraged to convert to the collectors described in this subsection. The original collectors return only the PV, and the time offset is always relative to current time however release 500 also provides an indirect version of the original HM series collectors (for use in pictures).

### NOTES

Direct history collectors can be used in either Free Format Logs or pictures.  
Indirect history collectors can be used in pictures but not Free Format Logs.

When either type history collectors are used in a picture, you must include a Collect History Actor (COLLHIST) to specify the starting time and to initiate history collection. COLLHIST is described in section 2 of the Actors manual.

Each of the history collectors in this subsection has a series of parameters. The general form of the direct collector is—

**Collector Name (Variable ID, Start Time, Time Offset)**

The general form of the indirect collector is—

**I\_Collector Name (Variable/Entity DDB, Start Time, Time Offset)**

Parameter Explanation—

**Variable ID**—The Variable ID is a point,parameter for which history is being collected. Variable ID examples are:

HG1001.PV  
HG1001.SP  
HG1001.PVEUHI

**Variable/Entity DDB**—(for indirect collectors) specifies the DDB Variable that will contain the ID for which history is being collected. For example: VAR01.



**Start Time**—The start time parameter is relative to the reference time as explained in the following paragraphs. Start time is used by the Time Offset parameter and is represented by one of the following numbers:

**Table H-1 Start Time**

Number	Start Time
0	Current time
1	Beginning of the hour
2	Beginning of the shift
3	Beginning of the day

**Reference Time**—The reference time and date for History Collectors in a Free Format Log (FFL) is operator entered through the START TIME/START DATE targets on the Report/Log/Trend/Journal Menu when selecting a log. The collector's Start Time parameter is then relative to the operator entered reference time.

For History Collectors in a picture, you must use a COLLHIST Actor to establish the reference time. When the picture contains indirect collectors, you can build operator entry type targets to bring in the reference time and date. These are then stored in a local DDB variable used by the COLLHIST Actor. The collector Start Time argument is then relative to the operator entered reference time. Alternatively, you can use a target with the COLLHIST actor Current Time flag = True, in which case the collector's Start Time is relative to the current time. See also H.4.4.

**Time Offset**—The Time Offset parameter tells how many time periods and which way to move relative to Start Time:

If Start Time equals–	Time Offset tells –
1, 2, or 3	how many time periods to move <u>forward</u> from the Start Time.
0 (Current time)	how many time periods to move <u>backward</u> from the Start Time.

**Table H-2 Offset Periods**

The Offset for–	is (period)–
hourly collectors	hours
user collectors	the user average period
Minute collectors	Minutes
Shift collectors	Shifts
Daily collectors	Days
Monthly collectors	Months

### CAUTION

There is no validity check for offset values. If an offset value is out of range, it is ignored at operating time. There is no error message.

Note that you must not create collectors that attempt to collect data not yet recorded.

The HM can have a 2-3 minute delay in recording collector data, therefore a short offset from current time (for example, 1 or 2 minutes) may not return the expected data. Use a time collector and a value collector together to verify the time the value was stored.

Most of the Historical Value Collectors also have a fourth parameter for the Value Type (average, maximum, minimum, etc.) which is explained where it applies.

**Example:** The collector statement: HOUR\_V(A101.PV,3,2,1) asks for the second hourly average of the day for A101.PV. Explanation: Start time = 3 (beginning of the day), 2 = second period, 1 = get average value. Other examples are provided for each group.

When the Add Value command request the expression, if you enter only a History Collector name and press the Enter key, a screen form appears to prompt for the parameters. For example, if SHIFT\_V is entered as the expression, the following screen form appears:

```
Collector SHIFT_V:

ENTER VARIABLE ID:      
ENTER START TIME (0,1,2,3): 
ENTER TIME OFFSET:      
ENTER DATA VALUE (1,2,3,4) 
```

**Figure H-1 Typical Direct Collector Screen Form**

5535 Revised

If the indirect collector I\_SHIFT\_T is entered as the expression, the following screen form appears:

Collector SHIFT\_T:

ENTER VARIABLE DDB:

ENTER START TIME (0,1,2,3):

ENTER TIME OFFSET:

**Figure H-2 Typical Indirect Collector Screen Form**

New

Type in the parameters and press the ENTER key.  
If the entity is unknown to the system, the system also prompts for the format.

Other screen forms are similar.

### H.4.1 Historical Value Collectors

Direct Historical Value Collectors have the general form—

**Collector(Variable ID,Start Time,Time Offset,Value Type)**

Indirect Historical Value Collectors have the general form—

**Collector(Variable DDB,Start Time,Time Offset,Value Type)**

The Historical Value Collector names are—

Direct Version	Indirect Version
USER_V	I_USER_V
MONTH_V	I_MON_V
DAY_V	I_DAY_V
SHIFT_V	I_SHFT_V
HOUR_V	I_HOUR_V
MINUTE_V	I_MIN_V

Note that USER means user-defined units.

Minute collectors do not use the Value Type parameter and return a snapshot value.

#### Parameter Explanation—

**Start Time**—See Table H-1

**Time Offset**—See Table H-2

**Value Type**—The value type parameter is specified by one of the following numbers:

**Table H-3 Value Types**

Number	Value Type
1	Average
2	Maximum
3	Minimum
4	Summation (average times number of samples)

Note that for the USER\_V or I\_USER\_V collectors to work properly, the user-defined time interval must be less than the Start Time interval. For example, if start time is set to **beginning of the current day**, then the user-defined time must be less than one day.

Also note that a Minute\_V collector may return @@@@ (data unavailable) at operating time if configured with an offset of 1 or 2 minutes. This happens if the requested data has not yet been written to the HM and primarily depends on how busy the HM is.

#### H.4.1.1 Historical Value Collector Examples

Example 1— HOUR\_V(HG1001.PV,3,2,1)

Returns the second hourly average (PV) from the beginning of the day.

Indirect version: I\_HOUR\_V(VAR01,3,2,1), where VAR01 contains HG1001.PV.

Example 2— MINUTE\_V(HG1001.PV,3,2)

Returns the second PV snapshot value, since the beginning of the day (the first value is at the beginning of the day).

Indirect version: I\_MIN\_V(VAR01,3,2), where VAR01 contains HG1001.PV.

Example 3— USER\_V(HG1001.PV,3,2,1)

Returns PV Average Value, for the second user defined time unit from the beginning of the day.

Indirect version: I\_USER\_V(VAR01,3,2,1), where VAR01 contains HG1001.PV.

#### H.4.2 Historical Time Collectors

Historical Time Collectors return a date and time. Data returned is the actual date/time stored with the variable value for the specified time.

Direct Historical Time Collectors have the general form—

**Collector(Variable ID, Start Time, Time Offset)**

Indirect Historical Time collectors have the general form—

**Collector(Variable DDB, Start Time, Time Offset)**

The Historical Time Collectors names are—

Direct Version	Indirect Version
USER_T	I_USER_T
MONTH_T	I_MON_T
DAY_T	I_DAY_T
SHIFT_T	I_SHFT_T
HOURL_T	I_HOURL_T
MINUTE_T	I_MIN_T

Note that USER means user-defined units.

## Parameter Explanation—

**Start Time**—See Table H-1

**Time Offset**—See Table H-2

**Example** —`HOURL_T(HG1001.PV,3,1)`

Returns date/time stored with the PV Average Value, for the first hour of the day (contains the starting time of the day).

Indirect version: `I_HOURL_T(VAR01,3,1)`, where VAR01 contains HG1001.PV.

## H.4.3 Historical Status Collectors

Status Collectors indicate whether or not enough samples were collected to provide a valid average. When the minimum number of samples have not been used to compute the average, an asterisk \* is displayed or printed. When the average was computed using more than the minimum number of samples, a blank space is displayed or printed.

Direct Status collectors have the general form—

**Collector(Variable ID, Start Time, Time Offset)**

Indirect Status collectors have the general form—

**Collector(Variable DDB, Start Time, Time Offset)**

The Status Collectors names are—

Direct Version	Indirect Version
USER_S	I_USER_S
MONTH_S	I_MON_S
DAY_S	I_DAY_S
SHIFT_S	I_SHFT_S
HOURL_S	I_HOURL_S

Note that USER means user-defined units.

## Parameter Explanation—

**Start Time**—See Table H-1

**Time Offset**—See Table H-2

### Example —`HOUR_S(HG1001.PV,2,1)`

Returns an asterisk \* if the minimum number of samples were not used to compute the first PV average value of the shift.

Indirect version: `I_HOUR_S(VAR01,2,1)`, where VAR01 contains HG1001.PV.

Refer also to the Combination Example on the next page.

**Combination Example**—You want to find the average set point value for point HG1001, four hours from the start of the current shift (and to indicate if enough values were used to compute the average), the following three collectors could be added to a Free Format Log:

```
HOUR_T(HG1001.SP,2,4,1)
HOUR_V(HG1001.SP,2,4)
HOUR_S(HG1001.SP,2,4)
```

The three collectors above might return the following:

**11:00 378.6 \***  
(for a bad average)

**11:00 378.6**  
(for a good average)

Note that what is actually reported is the fourth value since the beginning of the current shift. Power failures or loss of data, or a time change could have affected what was stored in the history module. It is important to pair Value and Time Collectors for this and other reasons.

## H.4.4 Using Indirect Collectors

The indirect collectors allow an operator to enter variables or entities at operating time. For example, one or more indirect collectors can be built into the picture (or subpicture) that expect the variable in VAR01. You can use a target with the actor string:

```
S_VAR(VAR01,R_VAR(0,0,20"Enter Variable ID",TRUE,0))
```

to allow the operator to type in the variable.

---

\*Note that a Minute\_V collector may return @@@@ (data unavailable) at operating time if configured with an offset of 1 or 2 minutes back. This happens if the requested data has not yet been written to the HM and primarily depends on how busy the HM is.

If the point names appear on the screen, they can be overlaid with a target and the actor string S\_VAR such that the operator can store them into the DDB by touching the name.

A picture requires a Collect History actor to establish the reference time and initiate history collection. Unless the actor's Current Time Flag = True, you will need actors that allow the operator to enter the date and time and a C\_DATTIM actor to combine and store them into the local DDB specified by the COLLHIST actor's third parameter. Activating the Collect History actor then indicates that the operator has entered all parameters and is ready to retrieve history. Refer to section 2 of the Actors Manual for a description of the Collect History Actor.

## H.5 ORIGINAL HISTORICAL COLLECTORS

These were the only historical collectors available through Software Release 300. They will continue to work in release 400 and 500. However, you should prefer the Historical Collectors described in subsection H.4 when building new pictures or Free Format Logs. Do not mix these with the newer type collectors in a table of values versus time data. The direct version HM or TM collectors can be used in either Free Format Logs or pictures. Effective with release 500, an indirect version of the original HM type collectors is available for use in pictures.

Note that when either the direct or indirect version of these collectors is used in a picture, you must include a Collect History Actor to initiate history collection.

### H.5.1 Historical Value Collectors

The Historical value collectors and values returned are—

Calling Sequence	Value Returned
Direct Versions:	
HMMIN(Entity, No. of periods back)	Current Minute (snapshot)*
HMHOUR(Entity, No. of periods back)	Hourly Average
HMSHIFT(Entity, No. of periods back)	Shift Average
HMDAY(Entity, No. of periods back)	Daily Average
HMMONTH(Entity, No. of periods back)	Monthly Average
HMUSR_AV(Entity, No. of periods back)	User Average
Indirect Versions:	
I_HMMIN(Entity DDB, No. of periods back)	Current Minute (snapshot)
I_HMHOUR(Entity DDB, No. of periods back)	Hourly Average
I_HMSHFT(Entity DDB, No. of periods back)	Shift Average
I_HMDAY(Entity DDB, No. of periods back)	Daily Average
I_HMMON(Entity DDB, No. of periods back)	Monthly Average
I_HMUSR(Entity DDB, No. of periods back)	User Average



When the Add Value command requests the expression, if you enter only a direct History Collector name and press the Enter key, a screen form appears to prompt for the parameters. For example, if HMHOUR is entered as the expression, the following screen form appears:

```
Collector HMHOUR

ENTER ENTITY ID:      

ENTER TIME OFFSET    
```

**Figure H-3- HMHOUR Direct Collector Screen Form**

ENTITY ID is an entity for which history is being collected. If the entity is unknown to the system, a screen form is also presented for the format. Time Offset is a positive integer that specifies the number of periods to go back for the historical value.

If the indirect collector I\_HMHOUR is entered as the expression for a value, the following screen form appears

```
Collector I_HMHOUR

ENTER ENTITY DDB:    

ENTER TIME OFFSET:   
```

**Figure H-4- I\_HOUR Indirect Collector Screen Form**

## H.5.2 Historical Time Collectors

The Historical Time collectors are—

Calling Sequence	Period
Direct Version:	
TMMIN(No. of periods back)	Minute
TMHOUR(No. of periods back)	Hour
TMSHIFT(No. of periods back)	Shift
TMDAY(No. of periods back)	Day
TMMONTH(No. of periods back)	Month
TMUSR_AV(No. of periods back)	User Average Period

Indirect Version: none.

The value displayed, time or date, depends on the format assigned when the collector was added. Remember that this time is a calculated time and not the time stored with the data values.

Entering only a direct Historical Time collector name causes a screen form to prompt for the parameters. For example, if TMMONTH is entered as the expression for a value, the following screen form appears.

Collector TMMONTH

ENTER INTEGER VALUE

**Figure H-5- TMMONTH Direct Collector Screen Form**

If the entity is unknown to the system, a screen form is also presented for the format.

INTEGER VALUE is a positive integer. It specifies the number of periods to go back for the date or time (the offset).

## H.6 TREND COLLECTORS

The Trend Collectors are described in the following paragraphs.

### Parameter Explanation

All of the Trend Collectors use one or both of the following parameters:

#### TREND ID

Trend ID is an integer from 1 to 12 that designates one of the 12 Trend Records (TREND01-TREND12) associated with a TREND or TREND AX Application Subpicture.

TRACE NUMBER is an integer from 1 to 4 that specifies the trace.

Entering only the Trend Collector call-name causes a screen form to appear that prompts for the parameter(s). For example, if TR\_COLOR is entered as the expression for a value, the following screen form appears; 1, designating TREND01, and 2, specifying trace 2, were typed in as examples.

Collector TR_COLOR	
ENTER INTEGER VALUE	<input type="text" value="1"/>
ENTER INTEGER VALUE:	<input type="text" value="2"/>

### H 6.1 Trend Centerline Time

A14-character string is returned that indicates the date and time for the centerline of a trend, if centerline trending is on. The format of the 14-character string is MM/DD/YY HH:MM. A blank string indicates that centerline trending is turned off.

**Calling Sequence**— TR\_CLINE(TREND ID)

## H 6.2 Trend Trace Color

**Calling Sequence**— TR\_COLOR(TREND ID,TRACE NUMBER)

An integer is returned that indicates the color of the specified trace. This can be used in a conditional behavior statement to set the color of some value on the screen that provides a visual link to the trace. The integer values and their meanings are

- 0 Trace is inactive
- 1 Cyan
- 2 Yellow
- 3 Red
- 4 Green
- 5 Blue
- 6 Magenta
- 7 White

### H.6.3 Trend Trace Data Source

A 2-character string value is returned indicating the data source for the specified trace: HM (History Module), HG (Hiway Gateway), or RT (Real Time). Two blank characters are returned if the trace is inactive.

**Calling Sequence**— TR\_DATA (TREND ID,TRACE NUMBER)

### H.6.4 Trend Trace Engineering Units Descriptor (R530 and later systems)

Returns the engineering units descriptor for the given trend id and trace number. A blank value is returned if the trace is inactive.

**Calling Sequence**— TR\_EUDSC(TREND ID, TRACE NUMBER)

### H.6.5 Trend Trace Variable Name

A string value is returned that is the external name for the variable being trended by the specified trace. A blank value is returned if the trace is inactive.

**Calling Sequence**— TR\_NAME(TREND ID, TRACE NUMBER)

### H.6.6 Trend Trace Parameter (R530 and later systems)

Returns the parameter name for the given trend id and trace number. A blank value is returned if the trace is inactive.

**Calling Sequence**— TR\_PARAM(TREND ID, TRACE NUMBER)

### **H.6.7 Trend Trace Point (R530 and later systems)**

Returns the point name for the given trend id and trace number. A blank value is returned if the trace is inactive.

**Calling Sequence**— TR\_POINT(TREND ID, TRACE NUMBER)

### **H.6.8 Trend Trace Point Description (R530 and later systems)**

Returns the point description for the given trend id and trace number. A blank value is returned if the trace is inactive.

**Calling Sequence**— TR\_PTDESC(TREND ID, TRACE NUMBER)

### **H.6.9 Trend Trace Range (High)**

A real value is returned that indicates the high-range value currently being used by the specified trace. A blank value is returned if the trace is inactive.

**Calling Sequence**— TR\_RNGHI(TREND ID, TRACE NUMBER)

### **H.6.10 Trend Trace Range (Low)**

A real value is returned that indicates the low-range value currently being used by the specified trace. A blank value is returned if the trace is inactive.

**Calling Sequence**— TR\_RNGLO(TREND ID, TRACE NUMBER)

### **H.6.11 Trend Trace Real Time Value (R530 and later systems)**

Returns the current (real time) value for the given trend id and trace number. A blank value is returned if the trace is inactive.

**Calling Sequence**— TR\_RTVAL(TREND ID, TRACE NUMBER)

### **H.6.12 Trend Scale Range (High)**

A real value is returned that indicates the high-scale value currently being used.

**Calling Sequence**— TR\_SCLHI(TREND ID, TRACE NUMBER)

### H.6.13 Trend Scale Range (Low)

A real value is returned that indicates the low-scale value currently being used.

**Calling Sequence**— TR\_SCLLO(TREND ID, TRACE NUMBER)

### H.6.14 Trend Scroll Time Stamp

A 14-character string value is returned that is the Gregorian representation of the scroll date and time for the specified trend. All blank characters are returned if scrolling is inactive.

**Calling Sequence**— TR\_SCRLL(TREND RECORD ID)

### H.6.15 Trend Time Base

An 8-character string value is returned indicating the time base being used for the specified trend.

**Calling Sequence**— TR\_TIME(TREND RECORD ID)

Note that when testing the returned 8-character string, upper case characters and blank positions within the string being compared are important. Examples:

```
IF TR_TIME (1) = " 1  MIN " THEN....
IF TR_TIME (1) = " 2  MIN " THEN....
IF TR_TIME (1) = " 5  MIN " THEN....
IF TR_TIME (1) = " 10 MIN " THEN....
IF TR_TIME (1) = " 20 MIN " THEN....
IF TR_TIME (1) = " 1  HOUR " THEN....
IF TR_TIME (1) = " 2  HOURS" THEN....
IF TR_TIME (1) = " 8  HOURS" THEN....
IF TR_TIME (1) = "16  HOURS" THEN....
IF TR_TIME (1) = "24  HOURS" THEN....
IF TR_TIME (1) = "48  HOURS" THEN....
IF TR_TIME (1) = "96  HOURS" THEN....
IF TR_TIME (1) = "HOURLY  " THEN....
IF TR_TIME (1) = "SHIFT   " THEN....
IF TR_TIME (1) = "DAILY   " THEN....
IF TR_TIME (1) = "MONTHLY  " THEN....
IF TR_TIME (1) = "  USER   " THEN....
```

### H.6.16 Hair Line Cursor Status (R510 and later systems)

Because plotted data can be obtained at different collection rates, the hair-line cursor may not be positioned on a collected value. Collector TR\_HLSTS can be used to indicate if the collected value has actually been retrieved. TR\_HLSTS returns a Boolean; TRUE when the hair line cursor is positioned on a collected value or FALSE when the cursor is between collected values.

Example: Two traces are displayed on the same trend graph. The first trace has a collection rate of 20 seconds and the second trace has a collection rate of one minute. The hairline cursor moves at 20 second intervals due to the first trace. The second trace only has a value collected every minute. Collector TR\_HLSTS always returns TRUE for the first trace. For the second trace, the value returned by Collector TR\_HLSTS is TRUE at each minute interval and FALSE at the 20 second intervals between each minute interval.

**Calling Sequence**— TR\_HLSTS(TREND ID,TRACE NUMBER)

#### NOTE

A suggested technique for use with this collector, is to add a conditional behavior statement such that when the status returned by TR\_HLSTS = TRUE, the displayed value returned by TR\_HLVAL changes from half intensity to full intensity.

### H.6.17 Hair Line Cursor Trend Value (R510 and later systems)

This collector returns a real number value that depends on the hair-line cursor position. If the hair-line cursor is positioned on a collected value, the value returned is the collected value. Otherwise, the value returned is the nearest collected value to the hair-line cursor position. To determine where the hair-line cursor is positioned, use the TR\_HLSTS collector. When the hair-line cursor is positioned on a collected value, TR\_HLSTS returns TRUE; otherwise it returns FALSE.

Blanks are returned if the trend graph is not in hair-line mode.

**Calling Sequence**— TR\_HLVAL(TREND ID,TRACE NUMBER)

## H.6.18 Trend Graph Cursor Position (R510 and later systems)

This collector returns the date and time at the cursor position on a trend graph—

- If the trend graph is in hair-line mode, this collector returns a date/time value for the hair-line cursor position.
- If the trend graph is in center line mode, this collector returns a date/time value for the center line.
- If the trend graph is in right axis mode and scrolled, this collector returns a date/time value for the right axis.
- If the trend graph is in right axis mode and scrolling is inactive, this collector returns blanks for the date and time.

The date and time are returned as a 17 character string in Gregorian format where—

MM/DD/YY HH:MM:SS

specify the month, day, year, hour, minutes and seconds.

**Calling Sequence**— TR\_LTIME(TREND ID)

## H.7 SCHEMATIC ALARM COLLECTORS

There are two types of Schematic Alarm collectors—

- those that return the composite alarm status for an area, a point, or other categories.
- those that return the number of alarms (alarm count) for a unit, a point, or other categories.

### H.7.1 Schematic Alarm Status Collectors

The Schematic Alarm Status collectors are:

- \$ADDBSTS DDB Annunciator Group Alarm Status collector.
- \$ANNSTS Annunciator Group Alarm Status collector.
- \$AREASTS Area Alarm Status collector.
- \$PDDBSTS Primmod DDB Alarm Status collector.
- \$PNTSTS Point Alarm Status collector.
- \$PRIMSTS Primmod Alarm Status collector.
- \$UNITSTS Unit Alarm Status collector.



Each Schematic Alarm Status collector can return any of the following values:

Value	Definition
UNACKEM	Highest alarm condition is <b>unacknowledged emergency</b> alarm
UNACKHI	Highest alarm condition is <b>unacknowledged high</b> alarm
UNACKLO	Highest alarm condition is <b>unacknowledged low</b> alarm
ACKEM	Highest alarm condition is <b>acknowledged emergency</b> alarm
ACKHI	Highest alarm condition is <b>acknowledged high</b> alarm
ACKLO	Highest alarm condition is <b>acknowledged low</b> alarm
NOALARM	No alarms exist

**Usage as a Value**—These collectors can be used in the expression for a value. For example—  
\$AREASTS used as a value returns one of the eight values listed above.

**Usage With a Variant or Conditional Behavior** —These collectors can be used in Variant or Conditional Behavior statements. For example:

IF \$AREASTS=UNACKEM THEN

Each of the collectors is discussed in the following paragraphs.

Note that that entities (such as a unit, a point or a DDB) referenced by the collectors must exist at schematic build time.

## NOTE

If a Primmod or \$MPROD name is used in a collector:

For R500 - R510 systems the Primmod must exist as an entity in the system, otherwise the value returned at run time is NOALARM (NOALARM is also a valid status if no points are in alarm).

For R520 and later systems, if the AM Multiple Primmod Alarming option in the Network Configuration File is set to—

- Multiple Primmod Option Disabled, the Primmod must exist as an entity in the system.
- Multiple Primmod Option Enabled Exclusive or Inclusive, the Primmod Name does not have to exist as an entity in the system.

#### **H.7.1.1 DDB Annunciator Group Alarm Status Collector**

This collector returns the composite alarm status for an assigned DDB that contains an Annunciator Group title.

**Calling Sequence**— \$ADDBSTS(DDB String Name)

where DDB String Name is the assigned DDB that contain an Annunciator Group title.

The Annunciator Group title must be valid, otherwise the value returned for the collector at run time is @@@@ @@@@.

If this collector refers to an invalid Annunciator Group title when used in a Variant or Conditional behavior statement, it causes an error at run time. The error action is whatever is configured for the Bad Value response by the Add Variant or Add Condition command.

#### **H.7.1.2 Annunciator Group Alarm Status Collector**

This collector returns the composite alarm status of a specified Annunciator Group.

**Calling Sequence**— \$ANNSTS("Annunciator Group Title")

where Annunciator Group title is the 8-character Annunciator Group title as configured in the Area data base.

The Annunciator Group title must be valid, otherwise the value returned for the collector at run time is @@@@ @@@@.

If this collector refers to an invalid Annunciator Group title when used in a Variant or Conditional behavior statement, it causes an error at run time. The error action is whatever is configured for the Bad Value response by the Add Variant or Add Condition command.

#### **H.7.1.3 Area Alarm Status Collector**

This collector returns the composite alarm status of the local area.

**Calling Sequence**— \$AREASTS

#### H.7.1.4 DDB Primmod Alarm Status Collector

For an assigned DDB that contains a Primmod or \$MPROD name\*, this collector returns one of the following alarm enumeration values:

ACKEM	ACKHI	UNACKEM	UNACKHI
ACKLO	NOALARM	UNACKLO	

**Calling Sequence**— \$PDDBSTS(DDB Name)

where DDB Name is the assigned DDB that contain a Primmod or \$MPROD name.  
In Release 500 - 510, the DDB must be of type String (for example, STRING01).

In Release 520 and later systems, the DDB can be of type STRING or type VAR (for example, STRING01 or VAR01). If it is of type VAR, the VAR DDB must contain an Entity.Parameter; for example: AM100.\$MPROD1.

Refer to the Primmod note in subsection 7.1.

#### H.7.1.5 Point Alarm Status Collector

This collector returns the composite alarm status of a specified point.

**Calling Sequence**— \$PNTSTS(Entity ID)

where Entity ID is the desired entity name.

The Entity ID must be valid, otherwise the value returned at run time is NOALARM.

#### H.7.1.6 Primmod Alarm Status Collector

This collector returns the composite alarm status of a specified Primmod name.

**Calling Sequence**— \$PRIMSTS("Primmod Name")

where Primmod Name is a Primmod Name of 16 characters or less.

Refer to the Primmod note in subsection 7.1.

---

\*\$MPRODn (where n = 1 - 4) values are available in Release 520 and later systems.

### H.7.1.7 Unit Alarm Status Collector

This collector returns the composite alarm status of a specified Unit.

**Calling Sequence**— \$UNITSTS("Unit ID")

where Unit ID is a two character Unit identifier.

The Unit ID must refer to a configured unit, otherwise the value returned at run time is @@@@ @@@@.

If this collector refers to an invalid Unit ID when used in a Variant or Conditional behavior statement, it causes an error at run time. The error action is whatever is configured for the Bad Value response by the Add Variant or Add Condition command.

### H.7.2 Schematic Alarm Count Collectors

The Schematic Alarm Count collectors are:

- \$ADDBCNT DDB Annunciator Group alarm count collector.
- \$ANNCNT Annunciator Group alarm count collector.
- \$AREACNT Area alarm count collector.
- \$PDDBCNT DDB Primod alarm count collector.
- \$PNTCNT Point alarm count collector.
- \$PRIMCNT Primmod alarm count collector.
- \$UNITCNT Unit alarm count collector.

Each Schematic Alarm Count collector can return the number of alarms (from 0 - 600) for the following specified alarm categories:

Value	Definition
UNACKEM	Highest alarm condition is <b>unacknowledged emergency</b> alarm
UNACKHI	Highest alarm condition is <b>unacknowledged high</b> alarm
UNACKLO	Highest alarm condition is <b>unacknowledged low</b> alarm
ACKEM	Highest alarm condition is <b>acknowledged emergency</b> alarm
ACKHI	Highest alarm condition is <b>acknowledged high</b> alarm
ACKLO	Highest alarm condition is <b>acknowledged low</b> alarm

The specified alarm category must be expressed in the form SET:MEMBER. For example:

\$ALRMSTS:UNACKLO.

**Usage as a Value**—These collectors can be used in the expression for a value. For example:

\$AREACNT(\$ALRMSTS:UNACKEM) returns an integer between 0 and 600.

**Usage With a Variant or Conditional Behavior** —These collectors can be used in Variant or Conditional Behavior statements. For example:

```
IF $AREACNT ($ALRMSTS:UNACKEM) <>0 THEN
```

Note that that entities (such as a unit, a point, or a DDB) referenced by the collectors must exist at schematic build time.

Each of these collectors is discussed in the following paragraphs.

#### **H.7.2.1 DDB Annunciator Group Alarm Count Collector**

This collector returns the number of alarms for the assigned DDBs that contain the Annunciator Group title and the Alarm Status Category.

**Calling Sequence**— \$ADDBCNT("DDB String Name", DDB Enm Name)

where DDB String Name is the assigned DDB that contain the Annunciator Group title, and DDB Enm Name is the assigned DDB that contains an Alarm Status Category.

Both the Annunciator Group title and Alarm Status Category must be valid, otherwise the value returned for the collector at run time is @ @ @ @ @ @ @ @.

If this collector refers to an invalid Annunciator Group title or an invalid Alarm Status Category when used in a Variant or Conditional behavior statement, it causes an error at run time. The error action is whatever is configured for the Bad Value response by the Add Variant or Add Condition command.

#### **H.7.2.2 Annunciator Group Alarm Count Collector**

This collector returns the number of alarms in the specified alarm status category for the specified Annunciator Group title.

**Calling Sequence**— \$ANNCNT("Annunciator Group Title", Alarm Status Category)

where Annunciator Group title is the 8-character Annunciator Group title as configured in the Area data base, and Alarm Status Category is any of the following:

\$ALRMSTS:UNACKEM	\$ALRMSTS:ACKEM
\$ALRMSTS:ACKHI	\$ALRMSTS:UNACKHI
\$ALRMSTS:UNACKLO	\$ALRMSTS:ACKLO

Both the Annunciator Group title and Alarm Status Category must be valid, otherwise the value returned for the collector at run time is @ @ @ @ @ @ @ @.

If this collector refers to an invalid Annunciator Group title or an invalid Alarm Status Category when used in a Variant or Conditional behavior statement, it causes an error at run time. The error action is whatever is configured for the Bad Value response by the Add Variant or Add Condition command.

### H.7.2.3 Area Alarm Count Collector

This collector returns the number of alarms in the specified alarm status category for the local area.

**Calling Sequence**— \$AREACNT(Alarm Status Category)

where Alarm Status Category is any of the following:

\$ALRMSTS:UNACKEM	\$ALRMSTS:ACKEM
\$ALRMSTS:ACKHI	\$ALRMSTS:UNACKHI
\$ALRMSTS:UNACKLO	\$ALRMSTS:ACKLO

The Alarm Status Category must be valid, otherwise the value returned at run time is @ @ @ @ @ @ @ @.

If this collector refers to an invalid Alarm Status Category when used in a Variant or Conditional behavior statement, it causes an error at run time. The error action is whatever is configured for the Bad Value response by the Add Variant or Add Condition command.

### H.7.2.4 DDB Primmod Alarm Count Collector

This collector returns the number of alarms for the assigned DDBs that contain the Primmod or \$MPROD\* name and the Alarm Status Category.

**Calling Sequence**— \$PDDBCNT(DDB Name, DDB ENM Name)

DDB Name is the assigned DDB that contain a Primmod or \$MPROD name.

In Release 500 - 510, the DDB must be of type String (for example, STRING01).

In Release 520 and later systems, the DDB can be of type STRING or type VAR (for example, STRING01 or VAR01). If it is of type VAR, the VAR DDB must contain an Entity.Parameter; for example: AM100.\$MPROD1.

The Primmod/\$MPROD name must be valid, otherwise the value returned for the collector at run time is 0 (but note that 0 can also mean no alarms).

DDB Enm Name is the assigned DDB that contains one of the following Alarm Status categories:

ACKEM	ACKHI	UNACKHI
ACKLO	UNACKEM	UNACKLO

---

\*\$MPRODn values (where n = 1 - 4) are available in Release 520 and later systems.

The Alarm Status Category must be valid, otherwise the value returned at run time is @@@@ @@@@.

If this collector refers to an invalid Alarm Status Category when used in a Variant or Conditional behavior statement, it causes an error at run time. The error action is whatever is configured for the Bad Value response by the Add Variant or Add Condition command.

Refer to the Primmod note in subsection 7.1.

#### H.7.2.5 Point Alarm Count Collector

This collector returns the number of alarms in the specified Alarm Status Category for the specified point.

**Calling Sequence**— \$PNTCNT(Entity ID, Alarm Status Category )

where Entity ID is the desired entity name, and Alarm Status Category is any of the following:

\$ALRMSTS:UNACKEM	\$ALRMSTS:ACKEM
\$ALRMSTS:ACKHI	\$ALRMSTS:UNACKHI
\$ALRMSTS:UNACKLO	\$ALRMSTS:ACKLO

The Entity ID must be valid, otherwise the value returned at run time is 0.

The Alarm Status Category must be valid, otherwise the value returned for the collector at run time is @@@@ @@@@.

If this collector refers to an invalid Alarm Status Category when used in a Variant or Conditional behavior statement, it causes an error at run time. The error action is whatever is configured for the Bad Value response by the Add Variant or Add Condition command.

#### H.7.2.6 Primmod Alarm Count Collector

This collector returns the number of alarms in the specified alarm status category for the specified Primmod name.

**Calling Sequence**— \$PRIMCNT("Primmod Name", Alarm Status Category)

where Primmod Name is a Primmod Name of 16 characters or less, and Alarm Status Category is any of the following:

\$ALRMSTS:UNACKEM	\$ALRMSTS:ACKEM
\$ALRMSTS:ACKHI	\$ALRMSTS:UNACKHI
\$ALRMSTS:UNACKLO	\$ALRMSTS:ACKLO

The Primmod Name must be valid, otherwise the value returned at run time is 0.

The Alarm Status Category must be valid, otherwise the value returned at run time is @@@@ @@@@.

If this collector refers to an invalid Alarm Status Category when used in a Variant or Conditional behavior statement, it causes an error at run time. The error action is whatever is configured for the Bad Value response by the Add Variant or Add Condition command.

Refer to the Primmod note in subsection 7.1.

#### H.7.2.7 Unit Alarm Count Collector

This collector returns the number of alarms in the specified alarm status category for the specified unit.

**Calling Sequence**— \$UNITCNT("Unit ID", Alarm Status Category)

where Unit ID is a two-character process Unit ID and Alarm Status Category is any of the following:

\$ALRMSTS:UNACKEM	\$ALRMSTS:ACKEM
\$ALRMSTS:ACKHI	\$ALRMSTS:UNACKHI
\$ALRMSTS:UNACKLO	\$ALRMSTS:ACKLO

The Alarm Status Category must be valid, otherwise the value returned at run time is @@@@ @@@@.

If this collector refers to an invalid Alarm Status Category when used in a Variant or Conditional behavior statement, it causes an error at run time. The error action is whatever is configured for the Bad Value response by the Add Variant or Add Condition command.

## H.8 MISCELLANEOUS COLLECTORS

### H.8.1 Universal Station Keylock Status

This collector is used to return an enumeration of the state of the Universal Station's Keyswitch.

The values of this set are

VIEW	(View)
OPR	(Operator)
SUP	(Supervisor)
ENGR	(Engineer)



**Calling Sequence—** \$KEYLEVL

**CAUTION**

Storing a new value in the parameter \$KEYLEVL causes a disagreement between the actual access level and that shown on the Console Status display.

## **H.9 DOCUMENTATION TOOL QUERY COLLECTORS**

With Release 510 and later software, the Documentation Tool query actors described in subsection 19 of the *Actors Manual* can be invoked at operating time from a custom built schematic. The Query collectors below can then be used from the same Universal Station that invoked the query (typically from the same schematic) to provide status and other information. Query status is obtained through the PSDP parameter QSTS(n) (refer to the *Engineering Reference Manual*). The QCLEAR actor can be used to clear status information in the QSTS parameter and therefore in the Query Collectors.

### **H.9.1 QUERY DESCRIPTOR NAME**

This collector returns the descriptor name of the last Documentation Tool Query that was invoked from this Universal Station. The name is returned as a string.

**Calling Sequence—** \$QNAME

A blank string indicates that no query was requested since the node was loaded or the query status values were cleared.

### **H.9.2 QUERY INVOCATION-DATE/TIME**

This collector returns the date and time that the last Documentation Tool Query was invoked from this Universal Station. The date/time is returned as a string.

**Calling Sequence—** \$QTIME

A blank string indicates that no query was requested since the node was loaded or the query status values were cleared.

### **H.9.3 QUERY STATUS**

This collector returns the status of the last Documentation Tool Query that was invoked from this Universal Station. The status is returned as a string.

**Calling Sequence—** \$QSTATUS

The status string returned is one of the following—

- **in progress**
- **completed**
- **completed but not displayable**
- **Printer Failure - Printout Aborted**
- **canceled**
- **device unavailable**
- **Fatal Error Occurred**
- **No Match Found**
- **Printing Results**
- **Multiple Errors Occurred**
- **----- (blank)**

A blank string indicates that no query was requested since the node was loaded or the query status values were cleared.

## APPENDIX I SCHEMATIC LOADING

### I.1 SCHEMATIC OPTIMIZATION

It is good engineering practice to optimize schematics wherever possible to reduce the load on the **TotalPlant** Solution (TPS) System. This is particularly true when complex plant operation requires the use of large, sophisticated schematics. It applies equally well to users with less demanding display requirements.

#### I.1.1 General Guidelines to Reduce Schematic Loading

There are several ways to improve schematics, reduce memory requirements, improve performance, and reduce the load on LCN and UCN nodes.

- **Use as few parameters as necessary in any schematic**—This seems obvious, but there may be a considerable difference between the schematic that gets the job done and one that does the job efficiently with the least system impact. You should evaluate the design carefully.
- **Keep the picture as simple as possible**—This principle has human factors impact as well as system implications.
- **Static parameters**—set the update rate to zero for any static parameters (those that don't need updating). They are collected only at schematic invocation.
- **Lines**—If possible, connect multiple lines with the same characteristics (behavior, etc.). For example, it is better to draw a box using four connected line segments, rather than four separate lines positioned to look like a box. Connected lines use less memory and take less time to draw on the screen.
- **Text**—If there is text on the same line with the same behavior, it is better to use a text object containing multiple words and spaces than to create separate text objects. For example, it is better to make the text "Boiler Feed Valve" with spaces between words, rather than create separate texts objects "Boiler," "Feed", and "Valve", placed on the same line. It uses less memory and reduces drawing time.
- **Similar objects**—Minimize the number of copies of the same object (for example, a reactor outline) in a schematic. It is easy to copy the object and change minor characteristics, like color, blink, etc. A better approach is to minimize the number of times the same (or very similar) objects are drawn by using conditionals to change color or other characteristics.

- **Variants**—Use Overlays Instead of Variants for target-initiated Changes. Any parameter in a Variant (every limb) is collected continuously as determined by its collection group number and update rate regardless of whether the Variant or limb is used. This is not true for overlays.

Overlay parameters are only collected when the overlay is on the screen. Therefore, rather than use a Variant with multiple limbs for target-initiated changes, try to use multiple overlays that produce the same effect. Obviously, you should still use Variants when you want changes in conditions to cause behavior changes in the display.

- **Custom Change Zones**—Consider the following when building custom change zones:
  - a. A Change Zone is always more efficient if built and used as an overlay. Build the Change Zone as an overlay, then have the actor that invokes the Change Zone call the appropriate overlay.
  - b. For fast update parameters in a schematic, build a custom change zone in a small part of the schematic and put the fast update items in this change zone. Put the slowest possible update rate on all other parameters in the picture.
  - c. Avoid the USER\_CZ actor when it is not necessary because it takes display update time. If you do use USER\_CZ, remember that it is unnecessary to do an update. After invoking USER\_CZ, the function does its own display update.
- **Multiple use of the same schematic**—Don't build a huge schematic that shows the entire process and all data and then have this same schematic on multiple Operator Stations in the same console. Instead, create an overview schematic that shows the whole process with as few parameters as possible. Then build separate schematics that show smaller parts of the process, with as few duplicate parameters as possible.
- **Elegant displays**—Some graphic capabilities produce elegant displays, but impact display update time and processor usage. These include:
  - a. Circles: octagons are as pleasing aesthetically and more efficient.
  - b. Scaleable text: this isn't really text but a set of lines and solids that impact performance.
  - c. 3-D Graphics: they significantly slow a graphics update, particularly when used with a condition or in a subpicture.
- **Subpictures**—Avoid lines and solids without conditions in subpictures if possible. Subpictures (including text, lines, and solids) are redrawn every update even if there are no conditions, variants, or values. The effort required to include these objects in each picture pays off in update speed.
- **Reverse video**—Use reverse video text and even reverse video spaces instead of highlighting solids. Because the display of text is much faster, you can improve a graphic by replacing boxes with reverse video spaces.

## **I.2 REDUCING SCHEMATIC LOADING ON ALL NODES**

The concepts that follow rely heavily on the assignment of parameter requests to collection groups. If you are not familiar with the concept of a collection group, refer to the Set Collection command in subsection 3.3.1.

### **I.2.1 Using Longer Update Periods**

It is important to reduce the rate of data gathering. The standard update rate for custom schematics is once every 4 seconds, but it is not always necessary to collect data at that rate. Consider collecting data at longer periods, particularly for data that you do not expect to change rapidly. The level of a large tank, or the temperature of a relatively slow process may not change significantly for tens of seconds or even minutes, so there is no need to update values such as these every 4 seconds.

Also, any group of values can be updated on demand using the UPDATE actor, so slow values can be collected when the operator needs them, using appropriate targets. A large class of data (point names, descriptors, engineering units, etc.) do not need to be updated at all. These data can be collected once, at schematic invocation.

### **I.2.2 Grouping by LCN Node Type and Unit**

To further reduce loading, each collection group should contain parameters from a single LCN node. In the case of AM and CG nodes, use separate collection groups for parameters from each Unit accessed by those nodes.

The same principles are true for the NIM, but you need to consider several other types of partitioning. This is discussed in depth later.

## **I.3 REDUCING SCHEMATIC LOADING ON UCN NODES**

There are two concepts that particularly apply when optimizing the UCN data gathering efficiency of a custom schematic. One is grouping of the data access requests by processor type and the other is the use of parameters at the highest possible data owner level.

A new R530 Picture Editor optimization command automatically performs the grouping of parameters (see I.3.2).

### I.3.1 Grouping of Data Access Requests by Processor Type

Custom schematics request data from the network through LCN Data Access. The mechanism used to make the request is called an Intermediate Data Block (IDB), but in this document, it is called a Data Request message.

Each Data Request message consists of a single collection group. A completely homogeneous request is generated by a group containing parameters from

- a single UCN node, and
- a single processor type (Control or IOL) within that node

However, if this criterion is applied strictly to more than a few UCN nodes or in cases where there are multiple collection rates, it creates a large number of IDBs and places a heavy transaction load on the NIM. It also creates a bookkeeping problem for the developer.

#### **NOTE**

The NIM sorts by node so display builders only need to be concerned about grouping by processor type and collection rate.

The combination of user-grouping by processor type and NIM sorting by node creates a UCN request message that is resident-homogeneous (single node/single processor in the same node).

The need for homogeneous UCN Request Messages arises from the way in which parameter requests are processed in the PM, APM, or HPM. Control parameters are (in effect) obtained directly from memory with relatively small cost in processing time in the node's communications Processor. IOL parameters require that the Communications Processor create a second request to the IOL processor with an associated cost in processing time.

This overhead cost occurs each time a new IOL parameter is encountered in a message. In a message that consists of a mixture of control-resident and IOL-resident parameters, the Communications Processor uses a significant amount of time switching between the two types of parameters. If the message consists of IOL parameters only, the number of IOL processor requests is considerably reduced because the Communications Processor can include several parameters in a single IOL request.

You can determine the processor type from the point type. It isn't necessary to be aware of the actual residence of a given parameter. The relationship of point type to processor type is—

#### IOL Processor Point Types

- Digital Input
- Digital Output
- Analog Input
- Analog Output

#### Control Processor Point Types

- Regulatory Control
- Regulatory PV
- Digital Composite
- Logic
- Process Module

All point types have some parameters that are NIM-resident, but access to those parameters has no effect on UCN node communications loading.

#### I.3.1.1 The Basic Rules for Grouping

Using the update rate considerations previously discussed, set up the collection groups as follows:

- groups to be updated at target selection
- groups to be collected at multiples of the schematic base rate
- groups to be included in fast update

Split groups that will update into subgroups by processor type (control or IOL).

#### Example

You are creating a schematic that requests both Control and IOL data from multiple APMs. Some of the data is non-updating (descriptors, point names, etc.).

You determine that the best overall system loading is achieved using two update rates (for example, the standard 4-second update and 8-second update).

Because the numbering of groups is arbitrary within the 0-245 limit allowed by the Picture Editor, you establish the following groups:

<u>Group</u>	<u>Rate</u>	<u>Membership</u>
0	0	All non-updating parameters
10	1	All ACKSTAT collectors
101	1	Control parameters collected every 4 seconds
102	1	IOL parameters collected every 4 seconds
201	2	Control parameters collected every 8 seconds
202	2	IOL parameters collected every 8 seconds

When the NIM constructs UCN parameter requests, it sorts each group's parameters by UCN node and each APM receives a homogeneous request.

### I.3.2 Picture Editor Optimize Collection Groups (R530)

The new Picture Editor optimization command puts UCN parameters into collection groups which are optimized for performance. The command can be negated by pressing the CANCEL key immediately following the optimization request.

**I.3.2.1 Command:**     **OPTIMIZE (O) (OPT) <UCN> <Control Group> <IOP Group>**

Where:

UCN is the UCN number. Valid range is 1-20.

Control Group is the group number to contain all parameters in the control processor. Valid range is 0-245.

IOP Group is the group number to contain all parameters in the IOPs on the IOL.  
Valid range is 0-245.

(See Point Types breakdown in section I.3.1)

**Cancel Key**—The use of the CANCEL key immediately following a request of the OPTIMIZE function will cause the group ids to revert back to their original values.

#### **NOTE**

The OPTIMIZE command will only affect symbols of the type "Point.Parameter." The function will optimize whatever parameters it can even if some run time errors are detected. Errors can be reviewed, corrected, and the OPTIMIZE command executed again, if desired.

Command line errors will appear in the Picture Editor prompt window.



### I.3.2.2 Run Time Errors

Run time error occurs if any Point.Parameter cannot be queried for the point type that occurs when the Point does not exist or the Data Owner is not responding. These errors are contained in an error file that can be viewed by ESC (Escaping) out of the Picture Editor and using the command processor print function. The file name for the error file will be the same file name as the schematic, but will have a file extension of ER. The file directory for the error file will be the same as the file directory for the schematic. The format of the error file is a title line “Residency Of The Following Point.Parameters Could Not Be Determined,” followed by a list of each Point.Parameter that could not be accessed. Once an ER file is created by the OPTIMIZE command, it remains intact until another OPTIMIZE command is executed for the same schematic.

#### NOTE

The OPTIMIZE command requires communication with the data owner nodes on the LCN; therefore, this function cannot be executed OFFLINE.

### I.3.2.3 Handling Errors

- If the OPTIMIZE function finds no matching parameters for the specified UCN, the following message will be displayed:  
**No Parameters Found For Specified UCN - Press <ENTER> to Continue**
- If the OPTIMIZE function cannot determine residency of all Point.Parameters in the schematic, an error file will be generated and the following message will be displayed:  
**Partial Optimization Occurred - See File XXXXXXXX.ER - Press <ENTER> to Continue**

The error file will contain a title, **Residency Of The Following Point.Parameters Could Not Be Determined**, followed by the list of Point.Parameters.

- The file directory for the ER file will be the same as the directory for the schematic source file being optimized. If an error file cannot be created, the error message will be:  
**Partial Optimization Occurred - Cannot Create Error File - Press <ENTER> to Continue**

The pathname/filename for the schematic can now be defined and the OPTIMIZE command executed again to view the errors.

The following new error conditions will have visible feedback:

- Invalid UCN Number
- Invalid Control Group Number
- Invalid IOP Group Number
- Missing Arguments (less than three arguments in the OPTIMIZE command)
- Extra Characters On Line (more than three arguments in the OPTIMIZE command)

### I.3.3 Use of Parameters at the Highest Data Owner Level

After determining the basic parameter requirements of an object (Variant, Conditional Behavior, etc.) you should ask the following questions if the logic requires PM/APM parameters:

- Is there a NIM-resident parameter that will allow creation of an equivalent effect?
- If the answer is no and if the parameter is IOL resident, can you use a parameter that is resident in the Control Processor?

For example, you may need a Digital Input point's PV that is already in use as a parameter of a logic point or digital composite point. In that event, the schematic can request a control-resident parameter, with less effect on Communications CPU loading.

Consider the following schematic Variant:

```
IF (A100.PVHHFL) THEN SUBPICTURE HIHIALRM ELSE
IF (A100.PVHIFL) THEN SUBPICTURE HI ALARM ELSE
IF (A100.PVLOFL) THEN SUBPICTURE LO ALARM ELSE
SUBPICTURE OTHRALRM
```

Assuming that A100 is an IOL resident point and that the parameters are in the same collection group, three separate transactions are required:

- the data request message (IDB) from the US to the NIM
- a UCN parameter request from the NIM to the APM which owns A100
- an IOL request generated by the host APMs Communications Processor.

Now consider the following schematic Variant, with basically the same effect:

```
IF (A100.HIGHAL=PVHH) THEN SUBPICTURE HIHIALRM ELSE
IF (A100.HIGHAL=PVHI) THEN SUBPICTURE HI ALARM ELSE
IF (A100.HIGHAL=PVLO) THEN SUBPICTURE LO ALARM ELSE
SUBPICTURE OTHRALRM
```

The data collection is reduced to a single NIM-resident parameter, HIGHAL, which is collected by a single transaction (a data request to the NIM).

This is a rather simple illustration of using parameters at the highest data owner level possible.

# Index

Abbreviations	3.1.2
Acknowledge State Collector	H.2
ACKSTAT, Collector	H.2
Action, Target	3.3.9.1
Actors	3.3.9.1
Add Commands	
Add Behavior Command	3.3.2.3
Add Condition	3.3.3.2
Add Inherit	3.3.10.5
Add Line	3.3.4
Add Priority	3.3.1.22
Add Solid	3.3.4.2
Add Subpicture	3.3.10.6
Add Target	3.3.9.1
Add Text	3.3.5.2
Add Textsize	3.3.5.2
Add Value	3.3.6.1
Add Variant	3.3.7.1
Alarms, Acknowledged/Unacknowledged	H.2
Annunciator Group Alarm Status collector	H.7.1.2
Annunciator Group Alarm Count collector	H.7.2.2
Application Subpictures	G
Area Alarm Count collector	H.7.2.3
Area Alarm Status collector	H.7.1.3
Arrays as Parameters	C
ASSOC action in Define Command	3.3.1.25
Average (User)	H.4.1
Periods Back, (Collector)	H.4.2
Daily (Collector)	H.4.1
Hourly (Collector)	H.4.1
Monthly (Collector)	H.4.1
Shift (Collector)	H.4.1
Bad Value, in Conditional Behavior Screen Form	3.3.3.2
Bad Value, in Variant Screen Form	3.3.7.1
Bar Charts	3.3.8
Bar Charts, Drawing	3.3.8
Behavior Keys on Eng. Keyboard	3.3.2.2
Behavior	
Conditional	3.3.3.1
Fixed (Literal)	3.3.2.
Bit Test (Intrinsic Function)	C.2.1
Blind Record as Data Type	C
Blink	3.3.2.1
Boolean Format	A
Boolean as Data Type	C

The letters A – H indicate Appendices to this publication. All other references identify the numbered section where the item appears. When multiple references are shown, bold print is sometimes used to indicate the most often needed information.

---

# Index

---

Bounding Box	3.3.4.3
Box, Bounding	3.3.4.3
Broken Line Graph Application Subpicture	G.1.8
Build Time	1.1.1.3
Character Units	3.1
Circle, Application Subpicture	G.1.1
Collector Concepts	H.1.1
Collectors	H
Collection Properties	3.3.1.20
Color Priority	3.3.2.1
Command (see also Type of Command)	
Abbreviations	3.1.2
Descriptions	3.3
Entry	3.1.1
Index	3.2
Line	3.1
Command Processor	1.1
Commands	
General Purpose	3.3.1
Overview of	1.1.1.2
Picture Editor	1.1.1.2, 3.3
Quick Index	3.2
Compile Source File	3.3.1.11.1
Compile Commands	3.3.1.11
PE Check for BREAK key (R600)	3.3.1.11.1
Conditional Behavior	3.3.3
Syntax	D
In Subpictures	3.3.10.4
Coordinates (Locations), Entering	3.1.3
Copyclip (R530)	3.3.1.18
Copy Commands	
Bar	3.3.1.17
Line	3.3.1.17
Solid	3.3.1.17
Subpicture	3.3.1.17
Target	3.3.1.17
Text	3.3.1.17
Value	3.3.1.17
Variant	3.3.1.17
Current Minute (Collector)	H.4.1
Current Priority	3.3.1.22
Cursor Position	3.1
Custom Graphic Display, Use	1.1
Custom Status Display Frame (subpicture)	G1.10
Cut (R530)	3.3.1.19

# Index

Daily Average (Collector)	H.4.1
Data Base	
Display (Local)	C.4
Data Entry Forms	2.1
Data Entry	
Of Picture Editor Commands	1.1.1.1, 2.2
Data Types	A
Data Types in Expressions	C.5
Date/Time	
Formats	A
As Data Type	A
Collector	H.3
Days Back (Collector)	H.4.2
DDB Annunciator Group Alarm Status collector	H.7.1.1
DDB Annunciator Group Alarm Count collector	H.7.2.1
DDB Primmod Alarm Count collector	H.7.2.4
DDB Primmod Alarm Status collector	H.7.1.4
Define Command	3.3.1.25
Define Comment	3.3.5.2
Delete Commands	3.3.1.15
Bar	3.3.1.14
Behavior	3.3.2.3
Condition	3.3.3.2
Inherit	3.3.1.15
Line	3.3.1.15
Priority	3.3.1.22
Solid	3.3.1.15
Subpicture	3.3.1.15
Target	3.3.1.15
Text	3.3.1.15
Value	3.3.1.15
Variant	3.3.1.15
Deselect Commands	
Bar	3.3.1.13
Behavior	3.3.2.3
Condition	3.3.3.2
Line	3.3.1.13
Priority	3.3.1.22
Solid	3.3.1.13
Subpicture	3.3.1.13
Target	3.3.1.13
Text	3.3.1.13
Value	3.3.1.13
Variant	3.3.1.13
Disp Fwd/Disp Back action in Define Command	3.3.1.25
Duration Format	A
Documentation Tool Query Collectors	H.9
Drawing Area	3.3.1.3, 3.3.1.11

---

# Index

---

Edit Region	3.1, 3.3.1.3
End Command	3.3.1.2
Engineering Keyboard	3.3.2.2, 3.3.5.2
Engineering Personality	1.1
Entity	B
Entity ID Data Type	C
Entry, Commands	3.1.1
Enumeration Constants	F
Member	F
Enumeration Set	F
String	F
As Data Type	C
Format	A
Equipment List	3.3.1.27–29
Error/Prompt Line	3.1
Errors	3.1.4
Escape Function	1.1
Expression Syntax	C.1
Expression, Regular Defined	A
Expressions	C.1
External Conversion (Intrinsic Function)	C.2.2
Fast Schematics (see Set Collection Command)	3.3.1.23
FBG, FGB Priority Codes	3.3.1.22
File, Object	3.3.1.11
Filename	3.3.1.5
Final Target/Action (see Define Command)	3.3.1.25
Fixed Behavior	3.3.2
Formats	A
Formats, Screen	3.1
Forms	2.1
Forms, Data Entry	2.1
Free Format Logs	2.4
GFB Priority Code	3.3.1.22
General Purpose Commands	3.3.1
Graphic Objects	3.3.1.15, 3.3.4
Grid Overlay	3.3.1.24
Help	3.1.5
Help action in Define Command	3.3.1.25
Historical Time Collectors	H.4.2
Historical Value Collectors	H.4.1, H.5.1
HMtime Collectors (HMDAY, HMHOUR, etc.)	H.4.1
HOURS Collectors	H.4
Hourly Average (Collector)	H.4.1
Hours Offset (Collector)	H.4.2

# Index

Identifier, Volume	3.3.1.5
Inherited Behavior	3.3.10.5
Initial Behavior on Conditional Screen Form	3.3.3.2
Initial Target/Action (see Define Command)	3.3.1.25
Integer	B
Integer Format	A
Integer Data Type	C.5
Intensity	3.3.2.1
Internal (Intrinsic Function)	C.2.3
Intrinsic Functions	C.2
Keyboard Event Actors (Actor)	3.0
Keyboard, Engineering	3.3.2.2, 3.3.5.2
Keypress Status (\$KEYLEVL) Collector	H.8.1
Large Status Node Object Box (subpicture)	G1.12
Lines, Drawing	3.3.4
List Equipment File Names	3.3.1.29
Literal Behavior	3.3.2.1
Locations, Specifying	3.1.3
Load Command	3.3.1.26
Load Equipment List Object File	3.3.1.27
Loading Due to Schematics	I
Logical Device Identifier	3.3.1.5
Logs, Free Format	2.4
Minute, Current (Collector)	H.4.1
Minutes Back (Collector)	H.4.2
MINUTE Collectors	H.4.2
Modify Commands	3.3.1.21, 3.3.8.2
Modify Bar	3.3.8.2
Modify Condition	3.3.3.2
Modify Line	3.3.4.4
Modify Solid	3.3.4.5
Modify Subpicture	3.3.10.6
Modify Target	3.3.9.2
Modify Text	3.3.5.2
Modify Value	3.3.6.2
Modify Variant	3.3.7.2
MONTH Collectors	H.4
Monthly Average (Collector)	H.4.1
Months Back (Collector)	H.4.2
Move Commands	3.3.1.15
Move Bar	3.3.1.15
Move Line	3.3.1.15
Move Solid	3.3.1.15
Move Subpicture	3.3.1.15
Move Target	3.3.1.15
Move Text	3.3.1.15
Move Value	3.3.1.15
Move Variant	3.3.1.15
Multiple Compile Command	3.3.1.11.2
Multiple Print Command	3.3.1.4.3

---

# Index

---

Multi-Broken Line Graph Application Subpicture	G.1.9
Name Forms, Syntax	B
Names, Reserved	C.4
Network Access Status	3.3.1.6, 3.1
Network Searching	3.3.1.6
New Command	3.3.1.1
Node Object Boxes (subpicture)	G1.11, G1.12
Object File	3.3.1.11
Objects	
Graphics	3.3.1.12, 3.3.4
Text	3.3.5.1
Origin	
Of Bar Chart	3.3.8.1
Of Subpicture	3.3.10.1, 3.3.10.6
Of Edit Region	3.1, 3.3.1.3
Original Historical Collectors	H.5
Page/Disp Fwd/Bkwd keys	3.3.1.3
Palette Command	3.3.1.30
Parameters	3.3.6.1, 3.3.10.2
As Data Type	C.5
Paste (R530)	3.3.1.20
Pathname	3.3.1.5
Personality, Engineering	1.1
Phantoms (Application Subpictures)	G
Picture Building	1.1, 1.1.1.1, 2.2
Picture Editor	1.1
Commands	1.1.1.2, 3.3
Data Entry	2.2
Optimize Collection Groups (R530)	I.3.2
Copyclip, Cut, Paste Commands (R530)	3.3.1.18-20
Pie Chart Application Subpicture	G.1.3
Pixel Units	3.1
Point Alarm Count collector	H.7.2.5
Point Alarm Status collector	H.7.1.5
Primmod Alarm Count collector	H.7.2.6
Primmod Alarm Status collector	H.7.1.6
Print Commands	3.3.1.4
Print to File Command	3.3.1.4.2
Priority	
Codes	3.3.1.22
Color	3.3.2.1
Current	3.3.1.19
Text/Graphics	3.3.1.19
Prompt/Error Line	3.1
Quarter Circle, Application Subpicture	G.1.2
Query Collectors	H.9



# Index

Radar Application Subpicture	G.1.6, G.1.7
Read Source File Command	3.3.1.10
Read Command	3.3.1.9
Real Format	A
Real Data Type	C.5
References	1.2
Region	
Edit	3.1, 3.3.1.3
Screen Regions (See <i>Actors Manual</i> )	
Regular Expression, Defined	A
Replace Subpicture	3.3.10.6
Reserved Names	C.4
Reverse Video	3.3.2.1
Ring Application Subpicture	G.1.5
Roll	3.3.1.3
Roll Factor	3.1
Roll, of Edit Region	3.3.1.3
Run Time	1.1.1.3
Scale Factor	3.3.4.3
Scale commands	3.3.4.3
Bar	3.3.4.3
Line	3.3.4.3
Solid	3.3.4.3
Subpicture	3.3.4.3
Variant	3.3.4.3
Schematic Alarm Collectors	H.7
Alarm Status type	H.7.1
Alarm Count type	H.7.2
Schematic Optimization/Loading	I.1
Reducing Schematic Loading on UCN Nodes	I.3
Optimize Collection Groups (R530)	I.3.2
Screen Format	3.1
Screen Form Entry Aids	3.3.3
Select Commands	3.3.1.12
Select Bar	3.3.1.12
Select Behavior	3.3.1.12
Select Condition	3.3.3.2
Select Inherit	3.3.10.6
Select Line	3.3.1.12
Select Priority	3.3.1.22
Select Solid	3.3.1.12
Select String	3.3.1.14
Select Subpicture	3.3.1.12
Select Target	3.3.1.12
Select Text	3.3.1.12
Select Value	3.3.1.12
Select Variant	3.3.1.12

---

# Index

---

Selected Objects	3.3.1.12
Self Defining Enumeration	
Format	A
Data Type	C.5
Syntax for	F
Semi-Reserved Names	C.4
Set Commands	
Set Behavior	3.3.2.3
Set Collectinh	3.3.10
Set Collection	3.3.1.23
Set Grid On/Off	3.3.1.24
Set Network On/Off	3.3.1.6
Set Origin	3.3.10.6
Set Palette	3.3.1.30
Set Pathname	3.3.1.5
Set Priority	3.3.1.22
Set Roll	3.3.1.3
Set Textsize Command	3.3.5.2
Shift Collectors	H.4
Shift Average (Collector)	H.4.1, H.5.1
Shifts Back (Collector)	H.4.2, H.5.2
Small Status Node Object Box (subpicture)	G1.11
Solids, Drawing	3.3.4.2
Source File	3.3.1.8
Compiling	3.3.1.11
Specifying Locations	3.1.3
Standard Enumerations, Syntax for	F
Status Display Frame (subpicture)	G1.10
Status (Intrinsic Function)	C.2.4
String Data Type	C.5
Subpictures	3.3.10, 3.3.7.1
Application	G
Adding to Display	3.3.10.2
How Used	3.3.10.2
MULTRECR	3.3.10.6
RECREATE	3.3.10.6
RECREATALL	3.3.10.6
Recreate All Subpictures	3.3.10.6
Recreate One Subpicture	3.3.10.6
Recreate Subpictures in Batch Mode	3.3.10.6
Replacing a Subpicture	3.3.10.6
Syntax Check	3.3.1.11
Text size considerations	3.3.10
With Parameters	3.3.10.3
Syntax	
Check of Subpictures	3.3.1.11
For Conditional Behavior	D
Of Expressions	C.1
System Display Data Base	C.4
System Time/Date (SYS_TIME) Collector	H.3
System Time Formats	A.8
System ID Command	3.3.1.32

# Index

Target Actions	3.3.9.1
Targets	
Drawing	3.3.9.1
Limit in Display	3.3.9.1
Text	
Commands	3.3.5
Format	A
Object	3.3.5
Moving	3.3.1.16
Textsize	
Commands	3.3.5.2
Time, use in collector	H.3
Touch Targets	3.3.9
TMtime Collectors (TMDAY, HMMHOUR, etc.)	H.4.2
Trend Application Subpictures	G.1.4
Trend (TR) Collectors	H
Hair-Line Cursor Status	H.6.16
Hair-Line Cursor Trend Value	H.6.17
Trend Centerline Time	H.6.1
Trend Graph Cursor Position	H.6.18
Trend Scale Range	H.6.12, H.6.13
Trend Scroll Time Stamp	H.6.14
Trend Time Base	H.6.15
Trend Trace Color	H.6.2
Trend Trace Data Source	H.6.3
Trend Trace Engineering Units Descriptor (R530)	H.6.4
Trend Trace Parameter (R530)	H.6.6
Trend Trace Point (R530)	H.6.7
Trend Trace Point Description (R530)	H.6.8
Trend Trace Range	H.6.9, H.6.10
Trend Trace Real Time Value (R530)	H.6.11
Trend Trace Variable Name	H.6.5
Type Checking (Variable)	3.3.1.6
Type, Data-Type in Expression	C.5
Unit Alarm Count collector	H.7.2.7
Unit Alarm Status collector	H.7.1.7
Unknown Format	A
Unknown Data Type	C.5
Unload Equipment List Object File	3.3.1.28
User Collectors	H.4.1
User Average Periods Back, (Collector)	H.5.2
Value Formats	A
Values	3.3.6
Variable, Type Checking	3.3.1.6
Variants	3.3.7, E
Variant Syntax	E, 3.3.7
Verify, Verify Prompt	3.3.1.31
Video, Reverse	3.3.2.1
Volume Identifier	3.3.1.5
Write Source File	3.3.1.8
Write Command	3.3.1.7
PE Check for BREAK key (R600)	3.3.1.7



---

**FAX Transmittal****FAX No.: (602) 313-4212**

---

**TO:** Information Development**Total FAX pages:** \_\_\_\_\_  
(including this page)

### Reader Comments

Title of Document: **Picture Editor Reference Manual**Document Number: **SW09-650 R620**Issue Date: **12/00****Comments:** \_\_\_\_\_

---

---

---

---

---

**Recommendations:** \_\_\_\_\_

---

---

---

---

---

**FROM:****Name:** \_\_\_\_\_ **Date:** \_\_\_\_\_**Title:** \_\_\_\_\_**Company:** \_\_\_\_\_**Address:** \_\_\_\_\_**City:** \_\_\_\_\_ **State:** \_\_\_\_\_ **ZIP:** \_\_\_\_\_**Telephone:** \_\_\_\_\_ **FAX:** \_\_\_\_\_**FOR ADDITIONAL ASSISTANCE:**

Write	Call
Honeywell Industrial Automation and Control 16404 N. Black Canyon Highway Phoenix, AZ 85053	1-800-343-0228 ( in the 48 contiguous states)





**Honeywell**

---

**Industrial Automation and Control**  
Honeywell  
16404 North Black Canyon Highway  
Phoenix, Arizona 85053