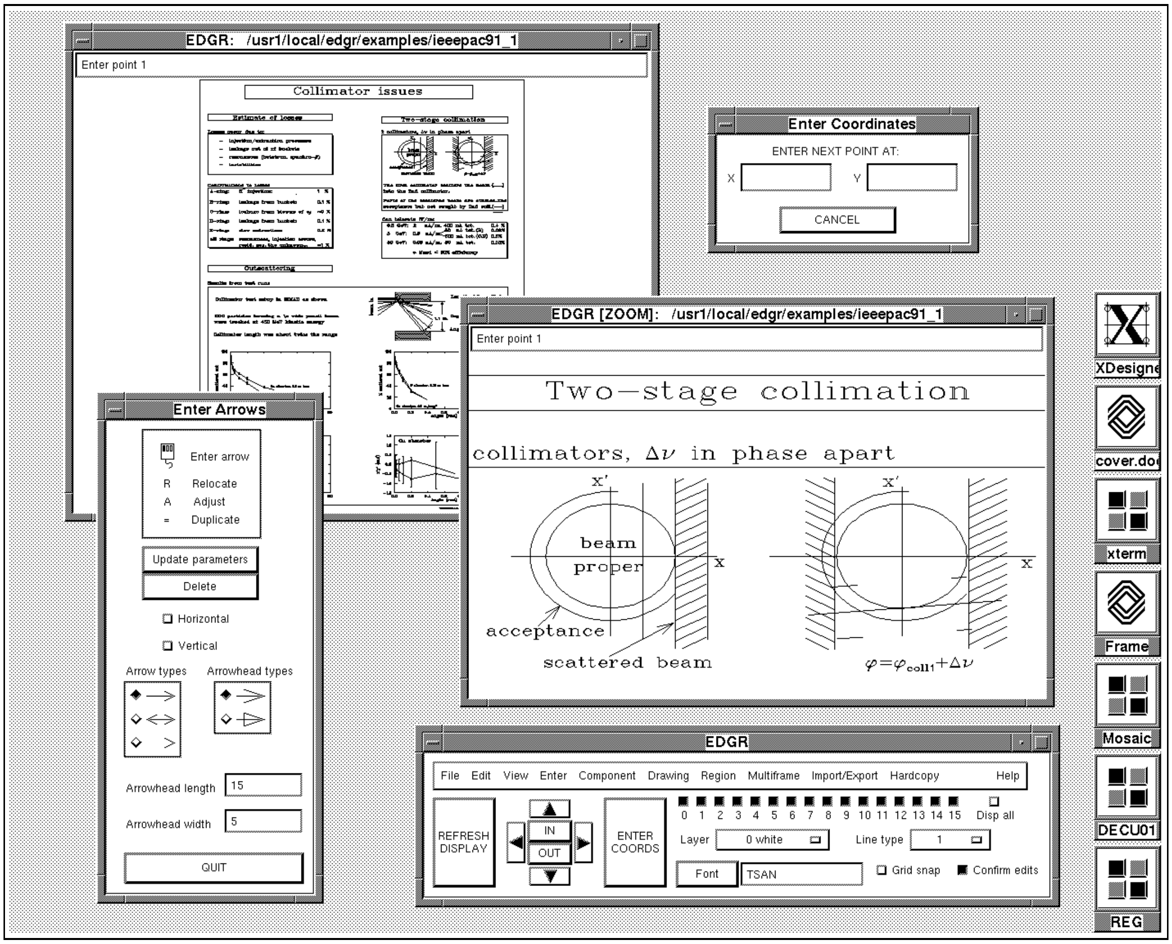
EDGR

Graphics Editor User’s Guide

F.W. JONES


EDGR: /usr1/local/edgr/examples/ieeepac91_1

Enter point 1

Enter Coordinates

ENTER NEXT POINT AT:

EDGR [ZOOM]: /usr1/local/edgr/examples/ieeepac91_1

Enter point 1

Enter Arrows

Arrowhead length 15

Arrowhead width 5

EDGR

Layer 0 white

Line type 1

Font TSAN

Version 6.7
April 1994
Computing Document TRI-CD-87-01

TRIUMF
4004 Wesbrook Mall
Vancouver, B.C.
Canada V6T 2A3

Contents

I OVERVIEW 1

1 Introduction 1

2 Applications of EDGR 2

3 Setting the Terminal Type 2

3.1 X Window displays 3

3.2 Generic Terminal Driver 3

4 EDGR's Two Interfaces 4

4.1 X/Motif interface 4

4.2 Command interface 5

5 Running EDGR 5

5.1 Getting started 6

6 Differences Between VMS and UNIX Versions of EDGR 7

7 Using the Mouse 8

7.1 Programmable mouse buttons 8

7.2 Additional mouse functions for X 8

8 Numerical Point Entry 9

9 Using Action Keys 9

9.1 X/Motif action keys 9

9.2 Command-mode action keys 10

9.3 Area Selection 12

10 Hardcopy Devices 13

11 Drawing and Editing Concepts 14

11.1 The Current Drawing 14

11.2 Drawing Coordinates 15

11.3 The Drawing Limits 15

11.4 Restrictions on Drawing Data 16

11.5 Drawing Format 16
11.6 The Hardcopy Image 17
11.7 Non-X Terminal displays 18
11.8 X Window displays 18
11.9 Selecting elements for editing 19
11.10 Verifying edits 19

12 Data Types **19**

13 Components **20**

14 Micro-editing of Text, Arcs and Components **20**

15 Attributes **21**
15.1 Fonts 21
15.2 Line types 22
15.3 Layers and colours 23
15.4 Layers and editing 24

16 Entering Text Strings **25**
16.1 Keyboard characters 25
16.2 Keywords for TRIUMF fonts 26
16.3 Keywords for all fonts 26
16.4 Hex codes 28
16.5 Text string recall 28

17 Grids **28**

18 The Status Line **29**

19 Script Files **29**

20 Generating Drawing Files with the DWG Package **30**
20.1 GPLOT Library Routines 31
20.2 DWG and DWG_CLOSE 31
20.3 DWG_OUTPUT 33
20.4 DWG_BATCH 33
20.5 Other DWG Routines 33

21 Running EDGR in a VMS Subprocess 33

22 Multiframe Drawing Files 34

23 Interface to Other Graphics Systems 34

II REFERENCE GUIDE 35

24 Introduction 35

24.1 Main Window 36
24.2 Command syntax 36
24.3 Command-line recall 37
24.4 Listing of drawing names 37
24.5 Starting EDGR with an immediate command 38

25 File 38

25.1 Edit new 38
25.2 Edit old 39
25.3 View 39
25.4 Close 39
25.5 Compress 40
25.6 Save as 40
25.7 List drawings [command mode only] 40
25.8 Execute script 41
25.9 Make survey file 41
25.10 Quit 42

26 Edit 43

26.1 Delete lines 43
26.2 Delete text 44
26.3 Delete components 44
26.4 Area delete 44
26.5 Area cut lines 44
26.6 Modify lines 45
26.7 Area modify lines 46
26.8 Modify text 46

26.9 Area modify text 47
26.10 Modify components 47
26.11 Smooth polyline 49
26.12 Convert to lines 49

27 View 50
27.1 Raise graphics 50
27.2 Clear graphics 50
27.3 Clear alpha [command mode only] 50
27.4 Refresh display 50
27.5 Zoom 50
27.6 End zoom 52
27.7 Pan 52
27.8 Layer control 53
27.9 Grid setup 54
27.10 Fast text on/off 55
27.11 Show border 55
27.12 Activate crosshairs [command mode only] 56

28 Enter 56
28.1 Enter lines 56
28.2 Enter arcs 57
28.3 Enter text 59
28.4 Enter components 61
28.5 Area fill 63
28.6 Enter Arrows 63
28.7 Enter Boxes 64
28.8 Enter ID stamp 65

29 Component 66
29.1 Component Make 66
29.2 Component Load 66
29.3 Component Maintenance 67

30 Drawing 67
30.1 Show status 67

30.2 Set edit tolerance 67
30.3 Change format 68
30.4 Move layers 68
30.5 Modify line types 68
30.6 Edit verification on/off 69

31 Region 69

31.1 Move 69
31.2 Transform 69
31.3 Duplicate 71
31.4 Extract 71
31.5 Insert 71
31.6 Insert tiles 72

32 Multiframe 73

32.1 Go to frame 73
32.2 Next frame 73
32.3 Previous frame 74
32.4 Multiframe Make 74
32.5 Multiframe Split 74

33 Import/Export 74

33.1 Convert plot file 74
  33.1.1 Transferring files to VMS 75
  33.1.2 Plotter files 75
  33.1.3 Tektronix files 75
  33.1.4 UGS plot files 76
33.2 Write net file 76
33.3 Read net file 76

34 Hardcopy 77

34.1 Hardcopy Entire drawing 77
  34.1.1 Colour assignments 79
  34.1.2 Pen plotters 79
  34.1.3 Line widths 79
  34.1.4 Postscript 79

34.1.5 Tektronix plot files 80
34.1.6 GKS metafiles 80
34.2 Hardcopy Current zoom 80

35 Help and Utility **80**
35.1 Info 80
35.2 News 81
35.3 General help 81
35.4 Command summary [command mode only] 82
35.5 Directory list [command mode only] 82
35.6 Text editor 82
35.7 DCL mode [VMS only] 83
35.8 Shell command [UNIX only] 83
35.9 Calculator [VMS only] 83

A Installing EDGR on VMS Systems **85**
A.1 Files for EDGR 85
A.2 Files for library support 85
A.3 GKS metafile support 86

B Installing EDGR on UNIX Systems **87**
B.1 List of required files 87
B.2 GKS metafile support 88

Command Index **89**

Part I

OVERVIEW

1 Introduction

EDGR (EDitor for GRaphics) is an interactive facility for creating, editing, printing, and storing graphical data. It has been developed at TRIUMF to allow users to create line drawings at their terminals and workstations, to store graphical output from other programs in a device-independent format, to edit and combine drawings, and to produce suitable hardcopy output for presentation, publication, or archiving.

EDGR runs on VMS and various UNIX platforms. It has a Motif-based graphical interface for X Window displays, and also provides a command-driven interface supporting a number of non-X graphics terminals. This allows a high degree of interoperability: it can be used effectively in environments ranging from a high-speed workstation to a PC-based terminal emulator over a modem line.

Functionally, EDGR is positioned somewhere between general-purpose drawing programs and CAD systems. It provides a basic set of drawing tools, where elements such as lines, arcs and text can be entered, moved and modified. Groups of elements or whole drawings can be cut, pasted, scaled, rotated, translated, and included in other drawings. Attributes such as colour, line type, or text font can be modified globally or on an element-by-element basis. Magnified views are available on both display and hardcopy. Other functions include area filling, text justification and alignment, and spline smoothing.

Some CAD-like features include grid overlays, independent graphics layers, full-precision coordinate storage, distance and angle measurement, and numerical coordinate input for any graphical operation. Often-used graphics objects can be defined as modular components that can be dynamically entered into any drawing. Sequences of editing functions that are used frequently can be automated by the use of script files. There are also some unique features for combining multiple drawings for presentation or archiving.

Hardcopy output is available for a number of different printers and plotters, with user-controlled colour assignments and line widths. Hardcopy files can be printed directly or saved for later printing or inclusion in TEX documents.

In addition to producing drawings from scratch, EDGR can edit the graphics produced by programs that use the TRIUMF GPLOT graphics library, notably the popular analysis/graphing program PLOTDATA and its successor PHYSICA. A subroutine package DWG, part of GPLOT, is provided to allow any user-written program to easily produce editable EDGR drawing files.

Graphics from other sources can also be imported: the HPGL, DM/PL, and Tektronix 4010/4014 formats, which are available from many graphics packages and programs, can be read by EDGR and converted into drawing files.

2 Applications of EDGR

Here are some typical ways that EDGR might be used:

1. To add some annotation to a graph produced by PHYSICA, before producing a hardcopy.
2. To store in a device-independent way the plots produced by a user-written data analysis program.
3. To produce a hardcopy using a format or device that is not supported by an application program.
4. To scale an image to the appropriate size and produce a plot file to be included as a figure in a $\TeX$ or $\LaTeX$ document.
5. To create a block diagram, schematic diagram, table or itemized list, and produce the output as a colour transparency.
6. To merge several plots or diagrams onto a single page, create captions, and produce a large-format plot for a poster session.
7. To capture, store, and produce hardcopies of the graphics output produced by a program that has no interface to the local graphics facilities.
8. To access a high-precision scale drawing and produce magnified views of specified areas.
9. To help design and lay out the graphics output for an application program that is under development.

3 Setting the Terminal Type

Before running EDGR you must specify the type of display or terminal you are using, by defining the logical name `TRIUMF_TERMINAL_TYPE` (on VMS systems) or the environment variable `TRIUMF_TERMINAL_TYPE` (on UNIX systems). The definitions for devices supported by EDGR are:

`X`	X Window workstation or terminal
`VT640`	Retro-graphics VT640 and emulators
`CIT467`	C. Itoh CIT-467
`PT100G`	Plessey PT-100G
`TK4107`	Tektronix 4100/4200 series and emulators
`GR1105`	Seiko GR-1105
`GENERIC`	User-programmed driver for other terminals

For example, if you usually use a workstation or an X Window terminal, you can put the following in your VMS `LOGIN.COM` file:

```
$ DEFINE TRIUMF_TERMINAL_TYPE X
```

or your UNIX `.login` file:

```
% setenv TRIUMF_TERMINAL_TYPE X
```

If `TRIUMF_TERMINAL_TYPE` is not defined or is set to some non-supported type, the default terminal type `VT640` is assumed by EDGR.

The `VT640` setting works not only with Retro-Graphics terminals but also with emulators such as Pericom terminals and the Atari ST running UNITERM or ST640.

The `TK4107` setting supports a family of Tektronix terminals including the 4105, 4107, 4205 and 4207. The EM4105 emulator for PC's also works well with this setting.

3.1 X Window displays

Specifying terminal type `X` indicates that you are using a workstation or terminal that supports the X Window System or a vendor's variant of it such as DECWindows. With this terminal type, EDGR can be run on your local workstation or on a remote computer. After logging in to a remote machine, and before running EDGR, the appropriate command must be issued to enable your local X Window display to be used.

If the remote system is a VMS system, the following command should be used:

```
$SET DISPLAY/CREATE/NODE=node_name/TRANSPORT=transport_name
```

The `node_name` parameter is the name of your local workstation or X terminal as it is known to the remote host. The `transport_name` parameter specifies the type of communications link to be used, typically `TCPIP` or `DECNET`.

If the remote system is a UNIX system, the following commands are used. For a TCP/IP connection,

```
% setenv DISPLAY node_name:0
```

and for a DECNET connection (if supported on the local and remote systems),

```
% setenv DISPLAY node_name::0
```

Once the X Window network connection is established, all graphics functions, including keyboard and mouse input, will be performed on the local workstation even though EDGR is running on the remote host.

3.2 Generic Terminal Driver

The `GENERIC` terminal type invokes a driver controlled by a data file that is created and configured by the user. The file contains lists of escape sequences for performing the various terminal functions. Creating such a file for a given terminal may allow it to be used with EDGR even though it is not explicitly supported.

The procedure for setting up a generic terminal is described in detail in the *Graph Plotting and Low Level Graphics* manual available from Computing Services, TRIUMF.

The minimum requirements for a generic terminal to be useable with EDGR are:

1. It uses a Tektronix or Regis vector graphics protocol.
2. It allows selective erase of vector graphics.
3. It has a VT100-compatible alphanumeric display plane
4. The alphanumeric plane can be erased and scrolled independently of the graphics plane.

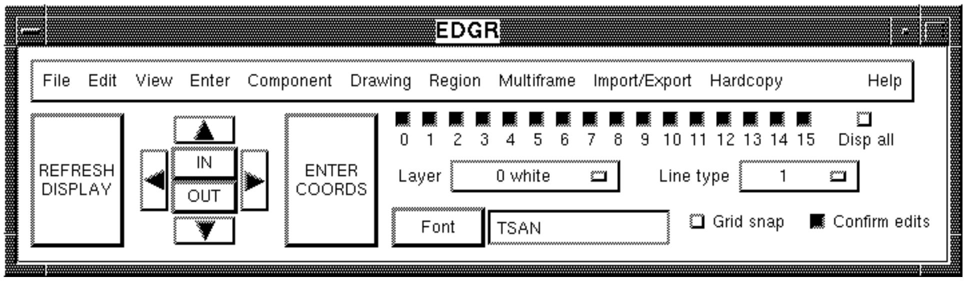
Although these might be considered minimum standards for any graphics terminal, a number of Regis terminals and some Tektronix emulators do not satisfy Requirement 4 and therefore are not supported by EDGR.

4 EDGR's Two Interfaces

4.1 X/Motif interface

For X Window display devices EDGR has a Motif-based point-and-click interface, rather than the command-based interface that is used with terminals. To use this interface you must define your terminal type to be X before running EDGR (see the previous section).

Along with the usual graphics window where the drawing is displayed, the main window for EDGR will appear on the screen. This window contains a menu bar with a series of drop-down menus that allow the user to invoke the various editing functions. For example, to enter lines and polylines, click on the Enter field of the menu bar and then click on Lines in the drop-down menu. This is equivalent to entering the `LINE` command when using a terminal. With a few exceptions, all of the terminal-mode commands have equivalent menu items.



■ 0 ■ 1 ■ 2 ■ 3 ■ 4 ■ 5 ■ 6 ■ 7 ■ 8 ■ 9 ■ 10 ■ 11 ■ 12 ■ 13 ■ 14 ■ 15 ☐ Disp all

Layer 0 white

Line type 1

Font TSAN

☐ Grid snap ■ Confirm edits

Main Window

When a menu item is selected, a window containing a dialog for the activity to be performed will pop up. For example, clicking on Edit old in the File menu will bring up a file selector dialog that allows you to select with the mouse an existing drawing file to be edited. Clicking on Text in the Enter menu brings up a dialog window that allows you to enter a text string and set the height and font of the text. Then, moving the mouse pointer to the graphics window and clicking at the desired location, the text is entered into the drawing. Additional function keys and buttons are provided for relocating the entered text, changing its parameters, setting the justification mode, etc.

Each dialog that involves mouse input into the graphics window will show a small mouse symbol. Next to this symbol will be shown the action that results from the mouse input. For the Enter Text command, for example, the mouse is used to enter the location of a new text string.

Several dialogs also use keyboard input into the graphics window to perform certain actions. This works in conjunction with the mouse pointer: the pointer is moved to the appropriate location in the graphics window and a keyboard key is pressed. For example, in the Enter Text dialog, the `R` key is used to relocate the current text to the current pointer position and the `Z` key is used to adjust the size of the current text so that it lines up with the current pointer location. See Section I for more information on keyboard input.

In addition to the menu bar, the main window contains numerous buttons and toggles for controlling the graphics display, zooming and panning, setting layers, line types, and the current font. These controls are available at all times, whereas only one menu bar function can be active at a time.

All user input is done through the various Motif-based dialog windows, the graphics window, and the zoom window if there is one. There is no input into the terminal window from which EDGR was run. Informational and diagnostic messages to the user are displayed in a one-line text region at the top of the graphics window. The zoom window has a similar region displaying the same messages. More lengthy output, and some error messages from the operating system or common library routines, may still be routed to the terminal window.

4.2 Command interface

This interface is primarily for use on non-X graphics terminals, although it can also be used on X displays if desired. Each editing function is invoked by entering a short command at the `EDGR>` prompt. Within an editing function, points are entered with the terminal's mouse or crosshairs, and additional actions can be performed using various keyboard keys called *action keys* (see Section I).

If you wish to use the command interface on an X Window display, it can be done by starting EDGR from a terminal window with the command:

```
edgr -c
```

A graphics window will be brought up, and EDGR commands can be typed into the terminal window as if you were using a terminal.

5 Running EDGR

The best way to avoid reading the rest of this document is just to run EDGR and try it out. Just as with a text editor, the best tutorial for many of the editing functions is to experiment with them.

If EDGR is properly installed on your system, you can run it by just typing `edgr` at your DCL prompt or shell prompt. *System Managers: see Appendices A and B for installation instructions.*

If EDGR isn't installed and you just want to try it out before doing a full installation, you can do a temporary "personal" set-up as follows:

VMS

```
$ DEFINE EDGR$DIR disk:[dir]
$ DEFINE TRIUMF$FONTS disk:[dir]
$ EDGR :== $EDGR$DIR:EDGR
$ COPY EDGR$DIR:EDGR.DAT SYS$LOGIN
```

In the above, replace `disk:[dir]` by the directory where the program `EDGR.EXE`, the font file `VAXFONT.DAT`, and the other files from the distribution are kept.

UNIX

```
% setenv EDGR_DIR directory
% setenv TRIUMF_FONTS directory
% alias edgr $EDGR_DIR/edgr
% cp $EDGR_DIR/Edgr $HOME
```

In the above, replace `directory` with the full path name of the directory where the program `edgr`, the font file `vaxfont.dat`, and the other files from the EDGR distribution are kept.

Note that the above set-ups put a copy of the EDGR application defaults file into your home directory. This is required for the X/Motif interface to work properly.

5.1 Getting started

If you have an X Window workstation or terminal, and you have properly defined the terminal type to be `X` as described in the previous section, then when you type `edgr` the Motif interface will start up, and you will see the main window and the graphics window appear on the screen.

On a non-X terminal, the command-mode interface will be started and you will see some introductory messages followed by the command prompt:

```
EDGR >
```

indicating that the editor is ready to accept your typed commands.

In the X/Motif interface, each of the main functions of the editor is invoked by selecting it from one of the dropdown menus in the main menu bar. In the command-mode interface each function is invoked by typing a short command.

After starting EDGR and ensuring that the terminal type setting is correct (read the previous section if it is not), you can try creating a drawing file, say `MYDRAWING`, by selecting

Edit new from the File menu and typing the name `MYDRAWING` into the file selection field. If you're in command mode, type the command `NEW MYDRAWING`. Then try out some of the data entry functions such as Enter Lines (`LINE` command), Enter Arcs (`ARC` command) and Enter Text (`TEXT` command).

In each case, you will be prompted to enter some points, either by a popup Motif dialog or by a prompt at the top of the terminal screen. Each point is entered by moving the mouse pointer (or terminal crosshairs if there is no mouse) to the desired location and pressing the `LEFT BUTTON` on the mouse, or the `SPACE` bar.

For many of the editing functions, there are additional actions that can be performed by pressing other keyboard keys. These are shown in the Motif dialog box, or in command mode in an action-key menu that can be displayed by pressing the `M` key on the keyboard.

Once you have entered some elements, they can be deleted or modified using the functions in the Edit menu, or in command mode by using commands like `DEL`, `DELT`, `MOD`, and `MODT`.

Other functions to try out are Region Transform (`TRAN`) and Region Duplicate (`DUP`), which transform and copy groups of elements.

Complete on-line documentation is available, through the Help menu or the `HELP` command.

6 Differences Between VMS and UNIX Versions of EDGR

The functioning of EDGR is almost identical on both VMS and UNIX systems. For those that are familiar only with the VMS version, the following points concerning the UNIX version should be noted:

1. Drawing files have the file types `.dwg` and `.dwt` in lower case. The drawing name itself can be upper, lower or mixed case.
2. When a new version of a drawing file is opened, the old version is renamed as *drawing_name*`.dwg~` and *drawing_name*`.dwt~`.
3. The `EDT` (text editor) command invokes the Emacs editor.
4. Of course, `DCL` mode does not exist. Any shell command can be executed from within EDGR by prefixing it with a "`%`" character.
5. The calculator `CALC` is not available. The command `%dc` will access the Unix `dc` calculator.

Drawings can be transported between VMS and UNIX systems using the following functions:

Write net file	(`NETW` command)
Read net file	(`NETR` command)

See Sections II and II for details.

7 Using the Mouse

Left button	`MB1`	Enter a point

MB1 is used to enter point locations in the graphics or zoom window. Its exact function will always be indicated in the current dialog window (in the X/Motif interface) or by a prompt (in the command interface). In the Enter Text function (`TEXT` command), for example, button 1 is used to position a new text element. For X displays, this function is indicated next to a small mouse symbol in the Enter Text dialog window. On terminals, it is indicated by a prompt at the top of the screen.

On terminals with no mouse, the graphics crosshairs are used instead, and the keyboard's space bar is the equivalent of button 1.

7.1 Programmable mouse buttons

When using terminal emulators such as the Atari/ST640 package, each mouse button can be programmed to send a keyboard character. Normally, the left mouse button would be assigned the space character so it can be pressed instead of the space bar. The right button, and the middle button if there is one, can be assigned to other frequently-used action keys, such as `R` (relocate) and `U` (update) which are used in a number of the editing commands.

7.2 Additional mouse functions for X

For X Window displays only, the following additional mouse button functions are available at all times:

Left button (drag)	`MB1`	Specify zoom rectangle
Middle button	`MB2`	Odometer on/off
Right button	`MB3`	Full-screen crosshairs on/off

MB1 is still clicked to enter points, but if it is pressed and held, and the pointer is moved, it will drag open a rectangle defining a zoom area. When the button is released, this area is displayed in a separate zoom window. If the zoom window does not yet exist, it will be created. This feature operates in addition to the usual `ZOOM` and `PAN` functions. For further information on the zoom window, see Section II).

MB2 opens a small window giving a continuous readout of the cursor coordinates. This window can be moved around on the screen as desired. The coordinates are given in drawing units for the current drawing. Pressing the middle button a second time stops the readout but does not close the window. The next time the middle button is pressed, the readout will resume and the window will be popped to the front if necessary.

`MB3` switches from the normal small crosshairs to ones that span the full active window. Pressing the right button again goes back to the small crosshairs.

8 Numerical Point Entry

In any EDGR function, whenever you are asked to enter a point with the mouse you also have the option of entering the numeric coordinates of the point. This allows greater precision in performing some graphics operations.

In the X/Motif interface, this is done by clicking on the ENTER COORDS button in the main window. A dialog box will pop up allowing you to type in the x and y coordinates of the point you want to enter. When you then click the mouse in the graphics or zoom windows, the actual mouse pointer location will be ignored and the point will be entered at the coordinates you have specified. This also applies if you enter the point with an action key rather than the mouse button.

In the command-mode interface, numerical coordinates can be entered using the K action key. See the following section for details.

9 Using Action Keys

In addition to the mouse (or terminal crosshair) input, EDGR uses keyboard keys to perform certain operations. Each EDGR function, such as entering lines, modifying text, etc., has its own set of such "action keys". As well, there are some general action keys that can be used in all functions.

9.1 X/Motif action keys

In the X/Motif interface most input is done using the mouse, so only a few action keys are required. Note that to use an action key, the graphics window (or the zoom window if there is one) must have the input focus! The window that currently has the input focus is usually highlighted by the window manager. To give a window the input focus, click on the title bar or frame of the window.

All the action keys relate to the pointer coordinates, so when the action key is pressed the mouse pointer must lie within the boundaries of the graphics window or the zoom window, whichever has the input focus.

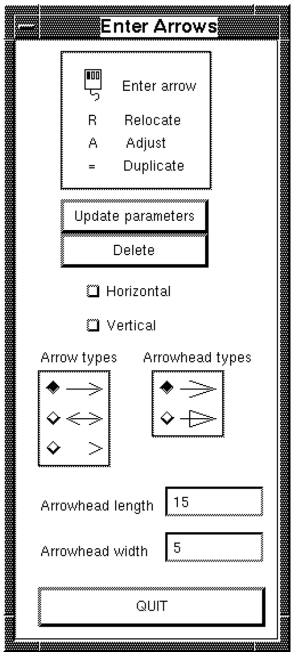
The following action keys are available at all times:

X	Display pointer coordinates
@	Set reference point
*	Place temporary marker

X displays the current location of the pointer in the drawing coordinate system. The display shows absolute coordinates, relative coordinates, distance and angle. The relative coordinates, distance and angle are all measured with respect to a reference point. The reference point is set using the $\boxed{@}$ key and has a default value of (0,0). These two keys can be used together to accurately align objects and perform distance and angle measurements.

$\boxed{*}$ places a temporary marker at the pointer location. This marker is not part of the drawing data and disappears the next time the screen is refreshed.

The various Enter and Modify functions (found in the Enter and Edit menus) define additional action keys, which are always shown in the dialog window that pops up when the function is selected from the menu.



Arrowhead length: 15

Arrowhead width: 5

Enter Arrows Dialog

For example, the Enter Arrows function uses the following keys:

$\boxed{R}$ relocates the current arrow, without changing its length and angle, so that it ends at the current pointer location.

$\boxed{A}$ adjusts the length and angle so that the arrow ends at the current pointer location.

$\boxed{=}$ makes a duplicate arrow that ends at the current pointer location.

These three keys are also used when entering lines, arcs, and boxes. The Enter Lines function has some more keys: $\boxed{S}$ to start a new polyline, $\boxed{H}$ to draw a horizontal segment, and $\boxed{V}$ to draw a vertical segment, as indicated in its dialog window.

9.2 Command-mode action keys

In the command-mode interface, action keys are used to perform operations similar to those of the mouse-activated push buttons and toggle buttons in the X/Motif interface. Hence there are many more key definitions to become familiar with, but the current list of valid keys can always be displayed for reference by pressing the `M` key (see below).

Note that the terminal must be in **graphics input mode** to use action keys. In this mode, the terminal's graphics crosshairs will be activated and EDGR will display a prompt indicating that graphics input is being solicited. For example, when you type the `TEXT` (Enter Text) command, the crosshairs appear and the following prompt is displayed at the top of the screen:

```
TEXT   Enter location of text                       M Menu
```

The prompt indicates what command is currently being invoked (`TEXT`), what happens when you press the mouse button or spacebar (new text is entered at the current crosshair location), and reminds you that pressing the `M` key will display a list of all the currently-valid action keys.

Some action keys are defined for all commands and are available at all times in graphics input mode:

`M`	Show menu
`X`	Display pointer coordinates
`@`	Set reference point
`*`	Place temporary marker
`K`	Keyboard input of coordinates
`G`	Grid snap on/off
`/`	Alpha clear
`\`	Refresh display
`#`	Calculator (VMS only)
`Q`	Quit (exit from current editing function)

`M` causes the menu of all currently-available actions to be displayed on the screen. This includes the standard menu as well as the particular menu for the editing function being used. The menu display is for reference only, and does not have to be on the screen in order for the menu keys to work. The menu display can be cleared by using the `/` key to erase the alpha screen.

`X` displays the current location of the pointer or crosshairs in the drawing coordinate system. The display shows absolute coordinates, relative coordinates, distance and angle. The relative coordinates, distance and angle are all measured with respect to a *reference point*. The reference point is set using the `@` key and has a default value of (0,0). These two keys can be used together to accurately align objects and perform distance and angle measurements.

`@` sets the reference point (for distance and angle measurements) to the current pointer location.

`*` places a temporary marker at the pointer location. This marker is not part of the drawing data and disappears the next time the screen is refreshed.

K exits from graphics input mode and allows one to specify the next point by entering in the coordinates numerically *at the keyboard* instead of by positioning the pointer or crosshairs. This allows points to be specified with a higher accuracy. In response to a prompt, the user types in the X and Y coordinates, optionally followed by an action key letter if a menu action is to be invoked. The effect is the same as entering a point at the given (X,Y) location. For example, to set a reference point at (100,80) one enters `100 80 @`. The entered coordinates can lie outside the current display limits, but if they do a warning message will be issued. Note: in the X/Motif interface, numerical coordinates can be entered by clicking on the "ENTER COORDS" button in the main window.

G enables and disables the snap grid, set up by the `GRID` command (see Section I). Pressing this key toggles the snap grid to the opposite of its previous state. Whenever the snap grid is on this is indicated in the status line at the top of the screen.

/ erases the alphanumeric screen or window, removing the menu if it is displayed. The menu can be brought back using the M key.

\ re-displays the current drawing, and is equivalent to issuing the `DISP` command.

# [VMS only] invokes a calculator facility which reads Fortran-like expressions and evaluates them numerically. Press RETURN to exit from the calculator and return to graphics input mode. The calculator can also be run from the command level with the `CALC` command (Section II).

Q exits from the current editing function and returns to EDGR command level or to the previous menu level if there is one. This key provides an ever-present "escape hatch". Pressing it one or more times will always get you out of any operation and back to the `EDGR >` prompt.

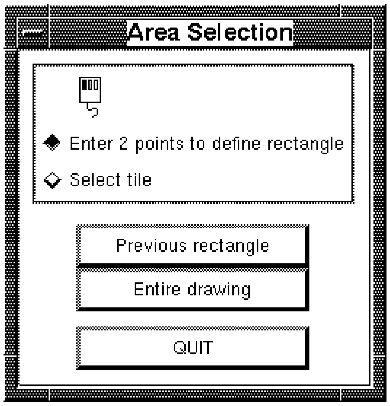
As well as the above universal keys, each editing function has its own set of keys. For example, the Enter Arrows function uses the R, A, and = keys (also used in the X/Motif interface) to relocate, adjust and duplicate the current arrow. Additional keys used only in command mode are:

Update parameters	U
Delete	D
Draw horizontal	H
Draw vertical	V
Set arrow parameters	P
Set layer	C

See Section II for information on how these keys work.

9.3 Area Selection

A number of the editing functions operate on rectangular areas of the drawing, and require the user to specify such areas. This is usually done by specifying two points to define



◆ Enter 2 points to define rectangle
◇ Select tile

Area Selection Dialog

diagonally opposite corners of the desired rectangle. Any two corners can be specified, in any order, provided they are diagonally opposite.

There are three other ways of specifying a rectangular area. In X/Motif an area selection dialog is popped up allowing the alternative methods to be used if desired. In command mode, equivalent action keys are provided.

Action	X/Motif	Command mode
Previous rectangle	[button]	$\boxed{\text{P}}$
Select tile	[toggle]	$\boxed{\text{T}}$
Entire drawing (or current window)	[button]	$\boxed{=}$

Previous rectangle: recalls the last rectangle that was entered. On terminals, this rectangle must lie within the current window.

Select tile: specifies a *tile* as the input rectangle. A tile is one of a series of equal rectangular areas into which the drawing is divided. The user simply moves the pointer over the relevant tile and presses the $\boxed{\text{T}}$ key to select it. This supposes that a *tiling scheme* has been defined by specifying the number or rows and columns in a regular grid that subdivides the drawing area. This can be done using the `TILE` command (see Section II). If no previous `TILE` command has been issued, the user will be asked to enter the number of rows and columns after selecting the tile.

Entire drawing (or current window): On X window displays this specifies the entire drawing. On non-X terminals it specifies a rectangle that exactly corresponds to the current display window, i.e. it can be used to select all items that lie within the display boundaries.

10 Hardcopy Devices

Hardcopy output is available from EDGR for the following devices:

- Printronix printer/plotter
- HP plotters (HP-GL)
- HP Laserjet printer
- HP Thinkjet printer
- DEC LA100 printer
- Houston Instruments plotters
- DEC LN03+ laser printer
- Imagen laser printer (imPRESS)
- HP Paintjet colour printer
- PostScript printers
- Roland GL plotters

On each device, hardcopies can be made in various sizes, resolutions, and orientations. In the case of pen plotters, the plotting speed can be varied, and the assignment of drawing colours to pen numbers can be changed. On bit-mapped hardcopy devices such as the HP Laserjet and HP Paintjet, the width of lines can be specified for the entire drawing or different line widths can be assigned for each colour layer in the drawing.

Hardcopy output is always through an intermediate plot file. Once the plot file is made, it can be directed to a device queue or saved for later printing.

Two types of generic plot file can also be produced:

- Tektronix 4010/4014 plot file
- GKS Metafile (where GKS is available)

These files may be printable or translatable to various devices not explicitly mentioned above.

11 Drawing and Editing Concepts

At this point we introduce a few concepts that are important in understanding the way EDGR works.

11.1 The Current Drawing

Drawings are called up using one of the following EDGR functions:

Menu: Item	Command	Function
File: Edit new	`NEW`	Open new drawing for editing
File: Edit old	`OLD`	Open existing drawing for editing
File: View	`VIEW`	Open existing drawing for viewing only

In each case, whatever drawing is called up becomes the *current drawing*.

For speed of processing, the contents of the current drawing are kept in memory at all times during the editing process. Note, however, that *all changes and additions to the drawing are performed simultaneously in memory and in the drawing file.* That is, the drawing file is always kept up to date and your work will not, in general, be lost if there is a hardware or software failure during editing. On the other hand, this means that changes made to a drawing cannot be "discarded" by quitting from the editing session. Therefore it is prudent to use the Save as function (`SAVE` command) to make a backup copy of your drawing before making extensive changes or additions.

For extremely large drawings there may be insufficient system resources available to load the entire drawing into memory. In this case an informational message is displayed indicating that only part of the drawing could be loaded. This does not restrict any editing operations, since all editing still takes place in the drawing file as well as in memory. This situation will, however, result in slower performance for searches and displays in the non-cached portion of the drawing.

11.2 Drawing Coordinates

The graphical elements of a drawing are defined in a terms of *drawing coordinates*. The drawing coordinate system can be any Cartesian coordinate system the user chooses, in any system of units. Some typical units are:

- Real-world distances (cm, m, *etc.*)
- Inches or centimeters on a hardcopy page
- Dots on a screen or hardcopy raster
- Normalized units (*e.g.* in the range 0 to 1)

What physical quantities the units ultimately represent is of no relevance to EDGR. One could, for example, have X units in milliseconds and Y units in centimeters. The only restriction is that the system be cartesian. Most applications use a *commensurate* system: one unit in X means the same thing as one unit in Y. In general, it is best to use commensurate units since this guarantees the correct appearance of text, arcs, *etc.*, on hardcopy output.

11.3 The Drawing Limits

The size and coordinate range of each drawing are defined by a rectangle in the drawing coordinate space. The coordinates at the borders of this rectangle are called the *drawing limits* and correspond to the edges of the drawing as it would appear on on a hardcopy.

If a drawing is being generated in an application program, the drawing limits are the same as those of the "world coordinate" limits that are in effect at the time the drawing is generated. These limits are usually set up by a call to HARDCOPY_RANGE:

```
CALL HARDCOPY_RANGE(XH1,XH2,YH1,YH2, XW1,XW2,YW1,YW2, IORIENT),
```

where `XW1`, `XW2`, `YW1`, `YW2` specify the world coordinate limits, and `XH1`, `XH2`, `YH1`, `YH2` specify a viewport into an internal bitmap. If a drawing file is opened subsequent to this call, the drawing limits will be `XW1`, `XW2`, `YW1`, `YW2`. If HARDCOPY_RANGE is not called, the world coordinate limits, and hence the drawing limits, are set by default:

```
XW1 = 0       XW2 = 639
YW1 = 0       YW2 = 479
```

(This system has a 3:4 aspect ratio and historically denotes pixel addresses on a VT640, the default graphics monitor)

If a new drawing is opened within EDGR, the drawing limits can be specified at the time the drawing file is created. Again, the default drawing limits are 0 to 639 in X and 0 to 479 in Y.

There are no restrictions on the drawing coordinate data: elements of the drawing can lie outside the drawing limits. Depending on how the drawing is displayed on the screen, such elements may not be visible or editable. Only the portion of the drawing within the drawing limits will appear on a hardcopy, except in the case of screen snapshots from a terminal (`SNAP` command).

Drawing limits can be expanded, contracted, or otherwise redefined using the Change format (`FORM`) function (see Sections I and II).

11.4 Restrictions on Drawing Data

It should be noted that EDGR is primarily a "page-oriented" system, where each drawing is assumed to correspond to a page of output on a hardcopy device, and the drawing coordinates have a linear and continuous correspondence with the hardcopy coordinates. Images involving multiple "world" coordinate systems, and multiple viewports in device coordinate space, are not in general supported. It is possible to generate such images using the GPLOT library routines, but it cannot be done in an integrated, device-independent fashion. It is assumed therefore, that each drawing is composed in a single world coordinate domain. This domain, bounded by the drawing limits, represents the full visible extent of the drawing as it will appear on on the output hardcopy page.

11.5 Drawing Format

Since the drawing's world coordinates can be in any units, and the X and Y units can be different, the drawing limits do not in general determine the intended size and shape of the drawing, as it will appear on the screen and on the hardcopy. One could specify for each drawing an intended "hardcopy size", but this may not always be determined in advance. The hardcopy size may also depend on the particular output device that is eventually used. To preserve as much device-independence as possible, we therefore divide drawings into three general classes, known as *formats*, according to their aspect ratio (ratio of height to width):

mode	**approximate height:width ratio**
landscape	3:4
portrait	4:3
square	1:1

The choice of 3:4 as the nominal ratio for a rectangular drawing is in keeping with the page-oriented graphics environment. Most hardcopy devices use page sizes that nicely accommodate it (e.g. 8.5"×11", on which one can make a 3:4 7.5"×10" plot with 0.5" margins).

The **square** format is appropriate for some applications where drawings are generated using commensurate units and equal limits in X and Y. The usual intention is that the drawing will occupy the largest square region that can be accommodated on the screen and hardcopy page.

Drawings for most applications conform reasonably well to these aspect ratio classes. For those that do not, the correct aspect ratio can be achieved by making an adjustment to the drawing limits. This is done using the Change format (FORM) function (see Sections I and II).

Some hardcopy devices are capable of producing multiple pages of fan-fold or roll paper output. If a drawing is intended to be plotted across several pages, a *page count* may be associated with the drawing format. By default, the page count is always 1. A page count of greater than 1 currently applies only to the HP Paintjet printer and is ignored for other devices. Multiple-page drawings may be in either landscape or portrait mode, where the mode refers to the orientation of the 8.5×11 fanfold pages when the plot is viewed.

When a new drawing is opened from an application program, the initial format is **landscape**. Once the drawing is opened, its format can be set as desired using the routine DWG_FORMAT:

```
CALL DWG_FORMAT(MODE,NPAGE)
```

where `MODE` is a character variable or literal, in upper or lower case. Valid modes are `'LANDSCAPE'`, `'PORTRAIT'`, or `'SQUARE'`. The parameter `NPAGE` (number of output pages) is normally set to 1, except for multiple-page HP Paintjet output, where it can be set to a maximum of 10.

The drawing format can be changed in EDGR using the Change format (FORM) function. Changing the format does not affect the drawing coordinate data or the drawing limits—it only affects the scaling and orientation of the drawing as it appears on the screen and hardcopy.

11.6 The Hardcopy Image

The size and orientation of the image of the drawing on the hardcopy page will depend on the drawing format: landscape, portrait, or square.

For landscape and portrait formats, the image will usually occupy most of the output page, with appropriate margins and orientation. For large-format 11×14 printers (Printronix and LA100) a smaller portion of the output page is used. For bit-mapped devices with variable resolution, such as the HP Laserjet and Thinkjet, a choice of hardcopy sizes and orientations may be offered, from which a selection is made prior to generating the hardcopy.

For square format, the image will be a square, usually centered on the output page and made as large as possible. For viewing, the output page will be in portrait orientation, except on the large-format printers where landscape orientation is used.

11.7 Non-X Terminal displays

When using EDGR with a terminal, the graphics display can either show the entire drawing or, through the `ZOOM` and `PAN` commands, a magnified portion of the drawing.

When a drawing file is first opened for editing by EDGR, the entire drawing is displayed so as to occupy as much of the screen as possible. For **landscape** format, the drawing and the screen have the same aspect ratio, so the drawing occupies the whole screen. For **portrait** and **square** formats, the drawing is centered on the screen and occupies the full height of the screen but only a portion of the width. In this case, the drawing limits are shown as dashed lines. For multi-page formats, the drawing occupies a centered screen area corresponding to the shape and size of the concatenated pages, and the fan-fold boundaries as well as the outer page boundaries are shown in dashed lines.

Once a drawing has initially been displayed in its entirety, the display limits can be redefined using the `ZOOM` and `PAN` commands. The `ZOOM` command allows the user to specify a rectangle whose edges correspond to the edges of the screen display. The contents of the rectangle will then be displayed so that they occupy the entire screen display area, thus producing a magnified (or even de-magnified) view of a portion of the drawing. The portion displayed can be changed using the `PAN` command, which has the effect of moving the "zoom rectangle" around in the drawing space.

If any of the drawing limits lie within the display area, they are shown as dashed lines. Some drawings may contain elements outside the drawing limits, and these may be visible on the screen display. However, they will not appear on a full-drawing hardcopy unless they are moved inside the drawing limits or the drawing limits are expanded to include them.

11.8 X Window displays

On X Window displays the main graphics window shows the full drawing and an optional zoom window shows a magnified view of any portion of the drawing. The name of the current drawing being viewed or edited is always shown in the title bars of the main window and zoom window.

Editing may be done in either graphics window. Both windows will be updated simultaneously when changes are made in the drawing. Each window can be moved and re-sized as required for editing convenience.

When the main window is re-sized, the full drawing display will be scaled to fit the new window size. When the zoom window is re-sized, the magnification remains the same and the portion of the drawing that is displayed will change depending on how the window is expanded or contracted.

Either the main window or the zoom window can be used for entering points. If you are entering points into a window using a keyboard key rather than a mouse button, you must ensure that the mouse pointer lies in the window and that the window has the keyboard input focus.

Selecting the `End zoom` menu item or issuing the `FULL` command will close the zoom window. A new one will be opened the next time a zoom is specified using the `Zoom` menu item or the `ZOOM` command.

There is also an "auto-zoom" feature: a zoom rectangle can be defined by moving the cursor into the main graphics window, pressing and holding mouse button 1, and dragging the cursor to define a rectangle. When the button is released, the zoom window will be redrawn to show the area within the rectangle.

11.9 Selecting elements for editing

When editing drawings, the mouse or crosshairs are used to select graphical elements that are candidates for editing changes. Generally the user enters a point on or near the element to be edited. For lines and arcs, all elements that are within a certain distance of the entered point will be selected by EDGR for editing. This distance is called the *edit tolerance*, and may be altered by the user. It is set by default to about 1/50 of the display width. To select a text element for editing, a point is entered anywhere within the bounds of the plotted text string. For a component, the point can lie anywhere within the bounding rectangle of the component.

For the "area" editing functions, multiple elements are selected by entering two points to define a rectangular area containing the elements. In this case a text element will be selected if its *base point*, which is at the lower left corner of the first letter of the text string, lies inside the rectangle. Similarly, components whose base (insertion) point lies within the rectangle will be selected.

11.10 Verifying edits

Once an element has been selected for an editing change, the action taken by EDGR depends on whether edit *verification* is enabled. With verification enabled, the user is asked if he wishes to perform the change on that element (e.g. delete it or change its colour), or to skip that element and go on searching for other elements that are candidates for editing. With verification disabled, all candidate elements are automatically edited *without confirmation* from the user. Verification is switched on and off using the main window's Confirm edits or the `VER` command.

Verification is useful not only as a safety feature but also as a way to edit certain elements that are within edit tolerance (or within a selection rectangle) but not others.

12 Data Types

The graphical elements that make up a drawing fall into four data types:

Lines This includes line segments and polylines (series of connected line segments). These elements are entered using the Enter Lines function or `LINE` command.

Arcs This includes circles, ellipses, and arcs thereof. These are entered using the Enter Arcs function or `ARC` command. For some of the editing commands, the designation "lines" refers to both line and arc data types.

Text Each text element is a character string associated with a base point. These elements are entered using the Enter Text function or TEXT command.

Components A component is a collection of graphics elements that have been defined as a unit, which can be placed at arbitrary locations in a drawing. Components are entered using the Component Load function (LOAD command) and the Enter Components function (PLACE command).

Some editing functions in EDGR restrict their operation to certain data types: for example, DEL deletes only lines and arcs while DELT is used to delete text and DELC is used to delete components. By contrast, the area delete function ADEL affects all data types.

13 Components

Some types of graphics objects such as symbols, logos, etc., may be frequently used in a given drawing or collection of drawings. Once such an object has been created, it is useful to be able to quickly and easily incorporate it into any drawing, in single or multiple copies, with arbitrary sizes and locations.

This can be accomplished using EDGR's **component** facility. Any graphics object which has been created as an EDGR drawing can be made into a component by creating a *component file* from the drawing. This file can then be loaded into any other drawing to define the component in that drawing.

Once a component definition is loaded into a drawing, the component can be used in the drawing by placing copies of it at the desired locations. Each copy of the component can be scaled to the required size and rotated by any angle.

Consult Part II of this guide for descriptions of the following functions for creating and using components:

Component Make	CMAKE	Section II
Component Load	LOAD	Section II
Enter Components	PLACE	Section II

14 Micro-editing of Text, Arcs and Components

In the EDGR system, a text element is normally represented as a character string associated with a base point. This enables text to be efficiently stored and easily edited. In some cases, however, the user may wish to make changes to the individual line segments that make up the plotted text characters. Similarly, arcs and components are stored in a form that does not allow access to these individual line segments.

The Convert to lines function and STROKE command are provided in EDGR to convert any text, arc, or component element into the **Lines** data type: that is, it becomes a series of editable line segments and loses its original identity.

This conversion can also be done when generating a drawing from an application program. For this purpose a routine DWG_STROKE has been provided:

`CALL DWG_STROKE('ON')`	to enable stroke conversion
`CALL DWG_STROKE('OFF')`	to disable stroke conversion

where keywords `'OFF'` and `'ON'` may be passed as character strings or literals, in upper or lower case. By default, stroke conversion is *off* when a drawing file is opened. DWG_STROKE should be called *subsequent* to DWG in order to enable stroke conversion.

15 Attributes

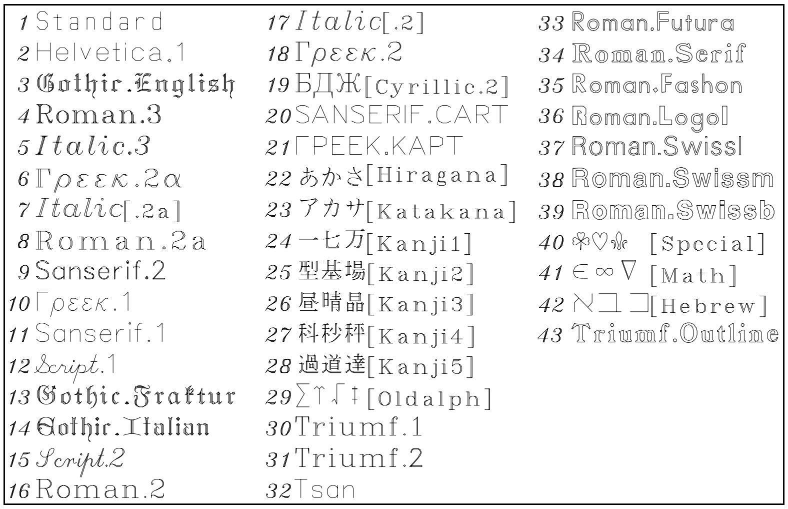
The appearance of each graphical element in a drawing is governed by a set of *attributes* that are stored in the drawing file. Each element carries its own set of attributes which can be modified by EDGR. Thus, the properties of an element or group of elements can be changed by using the appropriate editing operation. The following table describes the attributes carried by each data type.

Data type	*Attributes*
Lines	Layer number (colour) Line type
Arcs	Layer number (colour) Line type Vertex angle
Text	Layer number (colour) Height Rotation angle Font
Components	Layer number (colour) Component type Scale factors Rotation angle Colour rendition flag

Some attributes are discussed in the following sections. The others are described in the *Reference Guide* section of this document, under the relevant commands `LINE`, `ARC`, `TEXT` and `PLACE`.

15.1 Fonts

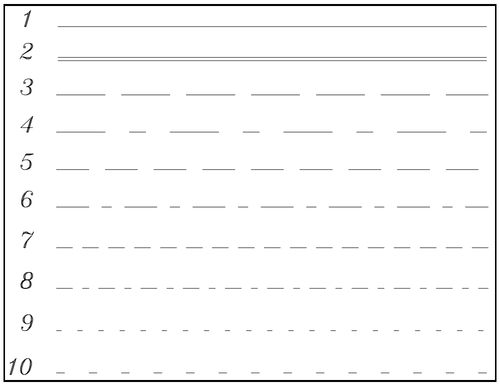
There are 43 fonts available for text, ranging from simple Sanserifs to elegant script and Gothic styles. Some of the fonts have special symbols that are not found on the terminal keyboard. In particular, the TRIUMF fonts (TRIUMF.1, TRIUMF.2 and TSAN) provide a comprehensive set of mathematics symbols and greek characters, in addition to the conventional keyboard characters. For these fonts, the symbols can be accessed using a system of keywords that conforms very closely to that used in TeX and LaTeX (see Section I for details). In the Motif interface, the current font can be set using a selection window opened by the Font button in the main window. In the command interface, fonts are specified by number. When a font number is requested, entering a `0` will produce a list of the available



Fonts

fonts by name and number. A set of tables for all the fonts is available separately from Computing Services, TRIUMF.

15.2 Line types



Line Types

For GPLOT-based application programs and within EDGR, there are ten line types available for drawing lines and curves. Type **1** is always an ordinary solid line, while types **2** through **10** are defined as double or dashed lines by a *line table*, in which each line type is defined by three floating-point parameters $P1$, $P2$ and $P3$, according to the following rules:

$P1$	$P2$	$P3$	*LINE STYLE*
0			*solid*
width	0	0	*double*
dash length	space length	0	*dashed*
dash length	space length	dash length	*dashed*

In an application program, the *default line table* is initially in effect, where line type **2** is a double line and types **3–10** are dashed lines with various dash and space lengths. This table can be altered as required by the application using the DLINESET routine. When a drawing is opened from the application program, the current line table is stored with the drawing. This line table is put into effect when the drawing is called up by EDGR.

When a new drawing is opened in EDGR, it is initially given the default line table. Alterations to the line table of any drawing can be made using the Modify line types function or `LMOD` command.

15.3 Layers and colours

Since EDGR supports both monochrome and colour devices, an effort has been made to allow maximum functionality and compatibility between colour and monochrome drawings and devices. To this end, a system of drawing *layers* is used to determine the colour of objects. Each object in the drawing belongs to one and only one layer, designated by a *layer number* that is carried with the object as one of its attributes.

Presently there are 16 layers, numbered **0** through **15**. Depending on what type of terminal or workstation is in use, each layer is assigned a particular colour. When a drawing is displayed on the screen, all objects in a given layer will be drawn in the designated colour for that layer.

For example, on a CIT-467 terminal, which has 7 display colours, objects in layer **0** will be drawn in **white** and objects in layer **1** will be drawn in **red**. The full list of layer colours is as follows:

0 white	**1** red	**2** blue	**3** violet
4 green	**5** yellow	**6** cyan	**7** white
8–15 white			

On Tektronix 4107/9 and 4207/9 terminals the following layer colours are used (assuming factory default colour palette):

0	white	**1**	white	**2**	red	**3**	green
4	blue	**5**	cyan	**6**	magenta	**7**	yellow
8	orange	**9**	green-yellow	**10**	green-cyan	**11**	blue-cyan
12	blue-magenta	**13**	red-magenta	**14**	dark grey	**15**	light grey

On workstations or X terminals with colour monitors, the following mapping is used:

0	white	**1**	red	**2**	blue	**3**	violet
4	green	**5**	yellow	**6**	cyan	**7**	white
8	orange	**9**	red-magenta	**10**	green-cyan	**11**	blue-cyan
12–15	white						

Note that the number of layers may exceed the number of available display colours, so that several layers must be drawn in the same colour. Of course, on a monochrome terminal all layers are drawn in the single available display colour.

The layer scheme also controls the colour of objects on hardcopies produced by printers and plotters. Each layer is drawn in a particular colour on the hardcopy. In this case the mapping of layers to colours is controlled by the user, who can select the colour for each layer at the time the hardcopy is made. On hardcopy devices that do not support colour, each layer can instead be drawn in a different line width. See Commands Reference section `HARD` for details.

When creating a drawing using EDGR, graphical elements can be entered into any layer. In the Motif interface, the current layer for data entry is set using the Layer option menu in the main window. In the command interface, the data entry functions `LINE`, `ARC`, `TEXT`, etc., allow one to set the layer number for subsequent entries, while the modification functions `MOD`, `MODT`, etc., allow the layer number of an element to be changed.

In an application program using the GPLOT graphics library, the routine PLOT_COLOR is used to set the drawing colour for subsequent graphics. The call sequence is:

```
CALL PLOT_COLOR(ICOL1,ICOL2)
```

where `ICOL1` and `ICOL2` are the integer colour indices for the first and second monitor devices. If a drawing file is generated from such a program, graphics are entered into the drawing layer number `ICOL1`.

Layer **0** is the default or "monochrome" layer. When a new drawing is created or an existing drawing is called up for editing, all new graphics elements will be entered into layer 0 unless otherwise specified. When a drawing is generated from an application program and PLOT_COLOR has never been called to select a drawing colour, then all graphics goes into layer 0 in the drawing file. Thus, a drawing with all elements in layer 0 could be considered as a "monochrome" drawing where no specific colours have been selected for the elements. A monochrome drawing can be made into a colour drawing by using EDGR's editing commands to move elements into different layers or adding new elements into various layers.

Although a colour display is preferable, a monochrome display does not preclude the editing of colour drawings. The layer number of an element can be determined using one of the Edit menu's Modify functions or the `MODx` commands, which display the layer number of any element that is selected for possible modification. The display can also be limited to selected layers of the drawing, as described in the next section.

15.4 Layers and editing

The layer system is used not only to assign colours or linewidths to objects but also to provide a logical separation between different groups of elements in the drawing. This provides more flexibility in editing, and is useful for making standard forms or templates, for making overlayed transparencies, and for protecting parts of a drawing from modification or deletion.

During an EDGR session, each layer can be turned on or off at will. Elements in layers that are *off* will not be affected by any editing operations. In the Motif interface, the layer states are controlled using the toggles, numbered 0 through 15, in the main window, and also through a popup dialog invoked by the Layer control menu item. In the command interface, the layer states are controlled using the `LAYER` command (see Section II) which brings up a menu where you can turn layers on or off by selecting them with the cursor keys.

The visibility of layers can also be controlled. By default, layers that are off do not appear on the screen or hardcopy, but if the DISPLAY ALL option is selected, all layers will be visible regardless of their state.

If the command interface is being used, whenever any layers are off (and therefore uneditable and possibly invisible) a *status line* (see Section I) will be displayed at the top of the screen, showing the state of each layer and the current layer for new elements being entered. When using layers it helps to have a colour terminal, but the selective display options allow monochrome terminals to be used successfully too.

16 Entering Text Strings

The functions Enter Text (TEXT command) and Modify text (MODT command) require text strings to be entered at the keyboard. Each text string specifies a series of characters to be plotted in the current font. Tables showing the character set for each font are available from Computing Services, TRIUMF.

The entered text string can contain *keyboard characters*, *keywords* and *hex codes* that specify the font characters used to compose a text element in the drawing.

16.1 Keyboard characters

These are the standard ascii characters on the terminal's keyboard:

```
abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ
1234567890-=`!@#$%^&*()_+~[]{};'\:"|,./?
```

These characters, together with a comprehensive set of math symbols and Greek letters, are provided in the three *TRIUMF fonts*: TRIUMF.1, TRIUMF.2 and TSAN. The character set for these fonts is shown in Table 1.

When using keyboard characters with other fonts, the font characters at the corresponding table locations will be plotted. *Examples:*

FONT:	TRIUMF.2
ENTERED STRING:	`abcklm`
PLOTTED TEXT:	abcklm
FONT:	SANSERIF.1
ENTERED STRING:	`abcklm`
PLOTTED TEXT:	abcklm
FONT:	GREEK.2A
ENTERED STRING:	`abcklm`
PLOTTED TEXT:	$\alpha\beta\gamma\kappa\lambda\mu$

Most of the Roman-style fonts have the same font table layout for keyboard characters, but some do not have the complete character set.

The keyboard characters "<" and ">" are reserved for special use as delimiters for keywords and hex codes. To generate these font characters the appropriate keyword (when using the TRIUMF fonts) or hex code (when using other fonts) must be entered:

Font character	Keyword	Hex code
$<$	`<lt>`	`<4C>`
$>$	`<gt>`	`<6E>`

16.2 Keywords for TRIUMF fonts

For font characters such as Greek letters and math symbols that are not found on the terminal keyboard, *keywords* can be used to generate the characters. These keywords are valid only when using the fonts TRIUMF.1, TRIUMF.2 and TSAN. For other fonts, hex codes must be used.

Each keyword must be surrounded by < and > characters, which act as delimiters. The keyword names used in EDGR adhere as closely as possible to those used in TeX and LaTeX for typesetting Greek letters and math symbols. Thus, except for minor differences, only one set of names need be learned. Keywords are always in *lowercase*, except for the first letter, which is sometimes in uppercase to indicate a variant form. Table 1 shows the valid keywords and, where applicable, their equivalent TeX symbols. For keywords that have no corresponding TeX symbol, the hex code for the font character is given. For ease of reading, the delimiters < and > have been omitted.

Examples of text entry using keywords:

FONT:	TSAN
ENTERED STRING:	`ABC<alpha><beta><gamma>123<infty>`
PLOTTED TEXT:	ABC$\alpha\beta\gamma$123∞
FONT:	TSAN
ENTERED STRING:	`ABC <lt> DEF`
PLOTTED TEXT:	ABC $<$ DEF

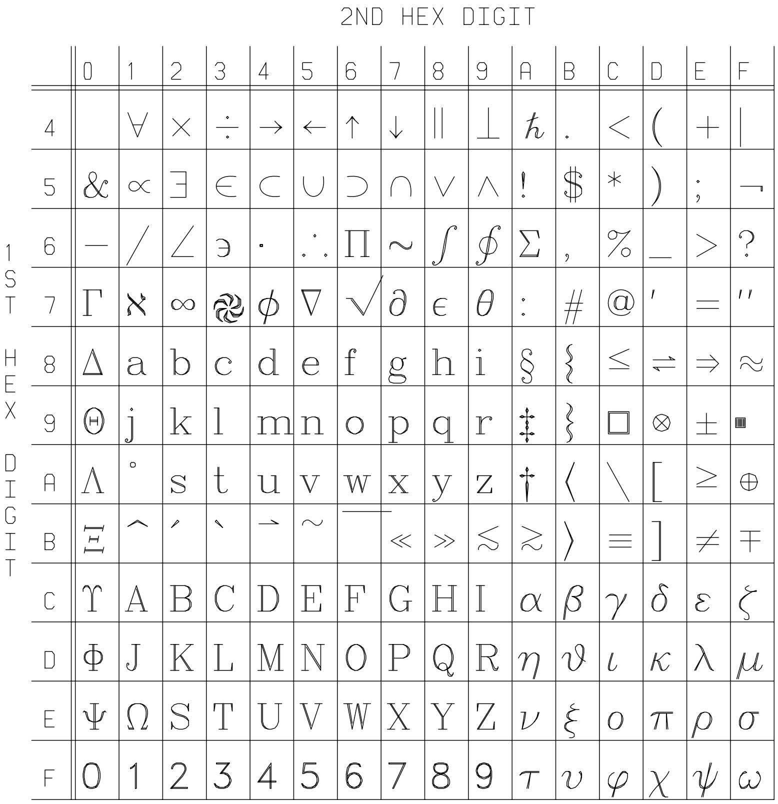
16.3 Keywords for all fonts

The following special "carriage control" keywords are valid for all fonts:

Keyword	**Hex code**	**Result**
`<^>`	09	Up (move up one super/subscript level)
`<_>`	38	Down (move down one super/subscript level)
`<bs>`	16	Backspace
`<cr>`	15	Carriage return

As with their TeX counterparts, the <^> and <_> keywords are used to generate superscripts and subscripts. *Example:*

FONT:	TSAN
ENTERED STRING:	`R<_>1<^> = X<^>2<_> + Y<^>2`
PLOTTED TEXT:	$R_1 = X^2 + Y^2$



2ND HEX DIGIT

1ST HEX DIGIT

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
4		∀	×	÷	→	←	↑	↓	‖	⊥	ℏ	.	<	(	+	|
5	&	∝	∃	∈	⊂	∪	⊃	∩	∨	∧	!	$	*	)	;	¬
6	—	/	∠	∋	·	∴	Π	∼	∫	∮	Σ	,	%	_	>	?
7	Γ	ℵ	∞		ϕ	∇	√	∂	ϵ	θ	:	#	@	′	=	″
8	Δ	a	b	c	d	e	f	g	h	i	§	{	≤	⇌	⇒	≈
9	Θ	j	k	l	m	n	o	p	q	r	‡	}	□	⊗	±	▪
A	Λ	°	s	t	u	v	w	x	y	z	†	⟨	\	[	≥	⊕
B	Ξ	^	´	`	⃗	~	‾	≪	≫	≲	≳	⟩	≡	]	≠	∓
C	ϒ	A	B	C	D	E	F	G	H	I	α	β	γ	δ	ε	ζ
D	Φ	J	K	L	M	N	O	P	Q	R	η	ϑ	ι	κ	λ	μ
E	Ψ	Ω	S	T	U	V	W	X	Y	Z	ν	ξ	o	π	ρ	σ
F	0	1	2	3	4	5	6	7	8	9	τ	υ	φ	χ	ψ	ω

∀	forall	×	times	÷	div	→	rightarrow	←	leftarrow
↑	uparrow	↓	downarrow	‖	parallel	⊥	perp	ℏ	hbar
<	lt	∝	propto	∃	exists	∈	in	⊂	subset
∪	cup	⊃	supset	∩	cap	∨	vee	∧	wedge
¬	neg	∠	angle	∋	ni	·	cdot	65	therefore
Π	Pi	∼	sim	∫	int	∮	oint	Σ	Sigma
>	gt	Γ	Gamma	ℵ	aleph	∞	infty	73	tlogo
ϕ	phi	∇	nabla	√	surd	∂	partial	ϵ	epsilon
θ	theta	Δ	Delta	§	S	≤	leq	⇌	rlharpoons
⇒	Rightarrow	≈	approx	Θ	Theta	‡	ddagger	□	box
⊗	otimes	±	pm	9F	squarebullet	Λ	Lambda	A1	degree
†	dagger	⟨	langle	≥	geq	⊕	oplus	Ξ	Xi
B4	vector	B6	overline	≪	ll	≫	gg	B9	lsimeq
BA	gsimeq	⟩	rangle	≡	equiv	≠	neq	∓	mp
ϒ	Upsilon	α	alpha	β	beta	γ	gamma	δ	delta
ε	varepsilon	ζ	zeta	Φ	Phi	η	eta	ϑ	vartheta
ι	iota	κ	kappa	λ	lambda	μ	mu	Ψ	Psi
Ω	Omega	ν	nu	ξ	xi	EC	omicron	π	pi
ρ	rho	σ	sigma	τ	tau	υ	upsilon	φ	varphi
χ	chi	ψ	psi	ω	omega				

Table 1: **Font table and keywords for TRIUMF fonts.**

The `<bs>` keyword results in a backspace to the previous character position, so that text can be overwritten. The `<cr>` keyword results in a carriage return to the starting point of the text string and a line feed to the next line, so that characters can be aligned vertically.

16.4 Hex codes

Font characters can also be specified using *hex codes*. Each character is selected by specifying a unique pair of hexadecimal digits, as shown in the font tables. As with keywords, a series of hex codes must be surrounded by the delimiters < and >. The hexadecimal digits `A` through `F` must be entered in *uppercase*. This allows hex codes to be easily distinguished from keywords. *Examples:*

FONT:	GREEK.2A
ENTERED STRING:	`<818283929394>`
PLOTTED TEXT:	$\alpha\beta\gamma\kappa\lambda\mu$
FONT:	ROMAN.2A
ENTERED STRING:	`A<4C>B`
PLOTTED TEXT:	A<B

16.5 Text string recall

When entering text using the `TEXT` command, whenever the prompt for a text string appears, the previously-entered strings can be recalled by pressing the up arrow key. Any recalled string can be edited and then entered by pressing the RETURN key. Up to 20 previous strings are stored. The same recall and edit facility can also be used when modifying text with the `MODT` command. The strings from both functions are stored in the same list.

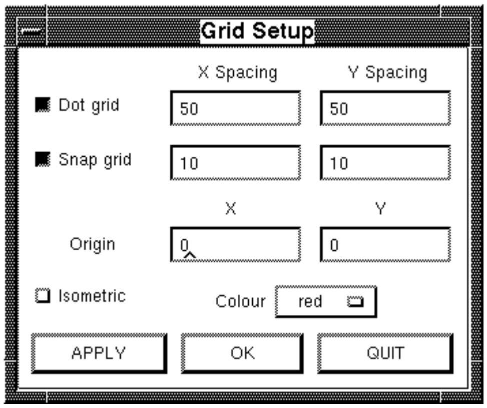
A similar recall/editing facility is available at the EDGR command level. See Section II for a description of the action of various keyboard keys when recalling and editing input.

17 Grids

There are two types of grids that can be used in EDGR. One is a reference grid of regularly-spaced dots that is displayed with the drawing but is not part of the drawing data. This grid allows one to more easily visualize the drawing coordinate system and see the coordinates of objects. The other grid is an invisible "snap" grid of regularly-spaced points in the drawing coordinate space. When the snap grid is enabled, points entered with the mouse or crosshairs are automatically aligned with this grid. In other words, any entered point snaps to the nearest grid point. This feature allows drawing elements to be entered at precise locations or intervals and is useful for aligning objects with each other.

To ease the creation of isometric drawings, an optional isometric grid is available.

The grid colour can be set to any desired display colour.



☒ Dot grid: X Spacing 50, Y Spacing 50

☒ Snap grid: X Spacing 10, Y Spacing 10

Origin: X 0, Y 0

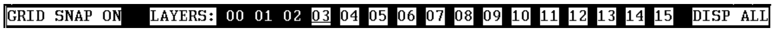
☐ Isometric

Colour: red

Grid Setup Dialog

The Grid setup menu item and GRID command, described in Section II are used to control the status and spacing of dot and snap grids. In addition, the Motif interface provides a toggle for turning the grid snap on and off, and in command mode the G action key performs this function.

18 The Status Line



```
GRID SNAP ON   LAYERS: 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15   DISP ALL
```

Status Line

When the command interface is used, EDGR provides a *status line* to indicate grid and layer conditions that affect the editing of graphical data. The status line is displayed at the top of the terminal screen if grid snap is enabled or if there are layers turned off. The layers that are on are highlighted and the layer corresponding to the current entry colour is underlined. If the DISPLAY ALL option is selected, this is also shown in the status line. The status line does not appear if grid snap is off and all layers are on.

In the Motif interface, the state of all the layers, and the DISPLAY ALL and GRID SNAP settings, are shown clearly by the toggle buttons on the main window.

19 Script Files

A simple facility is provided for execution of EDGR commands read from a file, known as a *script file*. This allows the automation of procedures that are frequently used or involve many predetermined coordinates. In the X/Motif interface, a script can be executed by selecting the Execute script menu item. In command mode, the command to invoke a script is

```
@filename
```

where the default file type is `.scr`. Once a script is invoked, all subsequent input is read from the script file until an end-of-file is reached, at which point control returns to the user terminal.

When executing a script, points that would normally be entered with the mouse or crosshairs are instead read from the script file in the following format (one record per point):

```
X      Y      [key]
```

where `X` and `Y` are the coordinates in drawing units and `[key]` indicates any action key for the current editing function. Omitting `[key]` is equivalent to entering the point with the left mouse button (or spacebar if there is no mouse). For example, the script

```
TEXT
300 200
Title
```

will place the text string "Title" at location (300,200). The sequence

```
0 0 H
10
320 240 R
0 0 Q
```

will change the height of the text string to 10, relocate it to location (320,240), and exit from the `TEXT` function.

Comments, indicated by a preceding "`!`", can be placed at the following locations in a script file: before an EDGR command, before a coordinate record, or appended to a coordinate record. Examples:

```
! Procedure to draw a title
TEXT
! Place the title
300 200      ! Coordinates of title
Title
```

Invalid EDGR commands or unreadable coordinate input will result in an exit from the script and return to terminal input. A diagnostic message will be be displayed together with the last record read from the script file.

20 Generating Drawing Files with the DWG Package

Note: Read this section if you are interested in generating EDGR drawings from your own graphics programs. Otherwise, it can be skipped.

EDGR provides a flexible environment for interactively creating drawings, but EDGR drawing files can also be generated by application programs that use the GPLOT graphics and graph-plotting library. This is done by calling a subroutine to enable the encoding of graphics data into a drawing file as the application is running. The resulting graphical elements and images can be edited and combined in the same way as their EDGR-generated counterparts. The way the application controls the generation of drawing files is described in the following sections.

20.1 GPLOT Library Routines

EDGR uses routines from TRIUMF's GPLOT library to support various graphics devices. These routines establish the transformations from the user's world coordinates to device coordinates, provide basic "move" and "draw" functions, do clipping against windows and viewports, and generate output for many different graphics devices. The primary low-level routine is PLOT_R, which generates a single vector on all active devices. At a higher level are routines which call PLOT_R, such as PSYM, for drawing text in various fonts; DLINE, for drawing dashed lines; and ARC, for drawing circular and elliptical arcs.

The GPLOT library also contains a comprehensive set of higher-level routines for drawing graphs, histograms, surface plots etc. The document *Low Level Graphics and Graph Plotting* provides a complete description of the library, and is available from Computing Services, TRIUMF.

In general, any program or package that uses the GPLOT graphics routines can generate EDGR drawing files. The drawing file is created and activated by a simple subroutine call. Then, as the program generates an image on the active graphics devices, the image is simultaneously encoded into the drawing file. When the image is complete, the drawing file can be closed and a new one opened to receive a subsequent image. Each completed drawing can later be retrieved and edited using EDGR.

We now describe the family of routines which control the generation of EDGR drawing files from application programs. The primary routine of this family is DWG, which interactively opens and activates a drawing file. In many cases this is the only routine needed in order to integrate a user's program with the EDGR system.

20.2 DWG and DWG_CLOSE

The subroutine DWG takes no parameters and is invoked simply by

```
      CALL DWG
```

When DWG is called, the user is prompted at the terminal to enter a name for the new drawing file to be generated:

```
Enter new drawing name or CTRL-Z to exit.  (Use * for directory list)
Drawing name >
```

On UNIX systems the prompt is slightly different:

```
Enter new drawing name or <RETURN> to exit.  (Use * for directory list)
Drawing name >
```

The drawing name may be from 1 to 39 alphanumeric characters, and may include embedded underscore (_) characters. A file type should not be included in the name. A device and directory specification can be prefixed to the drawing name, for a total of up to 80 characters. If these are omitted, the current default device and directory are used.

To help in choosing a drawing file name, a listing facility is included. This is invoked by entering a string containing one or more "*" wild card characters. All drawing names that match the specification will be listed and the drawing name prompt will be given again.

If you change your mind and do not want to open a drawing file, type CTRL-Z (on VMS) or RETURN (on UNIX) to exit from the prompt mode.

Upon receipt of a valid drawing name, DWG creates two files:

drawing_name`.dwg` and *drawing_name*`.dwt`

These files form an *interlocked pair* that is referred to loosely as a *drawing file* in this document. The use of a pair of files rather than a single file gives a substantial gain in both editing speed and convenience: the DWG file is used principally to store vectors while the DWT file contains mainly text strings. Both files must exist as a compatible pair in order for the drawing to be accessible and editable by EDGR.

As an example of how to use DWG, consider a program that uses GPLOT to draw some graphs:

```
REAL X(20),Y(20)       coordinates of plotted points
  ...
CALL SETNAM( ... )     Set up GPLOT parameters
  ...
CALL CLEAR_PLOT        Clear devices and plot files
CALL DWG               Open a drawing file
CALL GPLOT(X,Y,20,1)   Draw the graph
  ...
CALL CLEAR_PLOT        Clear in preparation for a new graph
CALL DWG               Open a new drawing file (old drawing file will be closed)
CALL GPLOT( ... )      Draw the next graph
CALL PSYM( ... )       Add some text to the graph
  ...
CALL DWG_CLOSE         Close the current drawing file
  ...
```

The routine DWG_CLOSE is used to close any drawing file that is currently open. It also prints a message on the standard output stream giving the name of the drawing what was closed. Whenever DWG is called, DWG_CLOSE is called automatically to ensure that there is only one open drawing file at any time. In the example, note that DWG was called again prior to doing the second graph. This is so the second graph will go into its own drawing file, rather than being overlayed on the first graph. Note that CLEAR_PLOT does not automatically open, close, or empty any drawing files—in fact it has no effect on drawing files at all. The drawing file status is completely controlled by the user program.

20.3 DWG_OUTPUT

In cases where it is desired to inhibit the flow of data to an open drawing file, without actually closing it, a routine DWG_OUTPUT is provided. Valid calls are:

`CALL DWG_OUTPUT('OFF')`	to disable output to drawing file
`CALL DWG_OUTPUT('ON')`	to enable output to drawing file

> **NOTE:** Keywords used in the DWG routines, such as `'OFF'` and `'ON'`, can be passed either as character variables or literals, and in upper or lower case.

20.4 DWG_BATCH

The routine DWG_BATCH is a non-interactive version of DWG, invoked as follows:

```
CALL DWG_BATCH(DNAME,IERR)
```

where `DNAME` is a character variable or literal containing a valid drawing name. If the specified drawing file already exists, a higher version is created. The integer variable IERR is returned as 0 if the drawing file was successfully opened, and returned as 1 if the drawing file could not be opened. In the latter case, a diagnostic message giving the reason for the failure is written on the standard output stream.

A companion routine DWG_BATCH_CLOSE exists to close any open drawing file, but without printing a message as DWG_CLOSE does.

20.5 Other DWG Routines

There are three other DWG routines for controlling drawing file generation. They are discussed later in this document:

DWG_NEXT	advance to next frame	(see Section I)
DWG_FORMAT	select format	(see Section I)
DWG_STROKE	enable or disable stroke conversion	(see Section I)

21 Running EDGR in a VMS Subprocess

In a program that generates drawing files, one may wish to use the graphics editor without stopping the program. On workstations or multi-session terminal connections, or on any UNIX system, this is a straightforward matter. In other cases, this can be achieved by having the program run EDGR in a subprocess, by adding the following call:

```
ISTAT = LIB$SPAWN('EDGR')
```

This spawns a subprocess, attaches the terminal to it, and runs EDGR. On exit from EDGR, the subprocess is deleted and control returns to the calling program. The above call assumes that the `EDGR` run command has been properly defined:

```
EDGR :== $EDGR$DIR:EDGR
```

22 Multiframe Drawing Files

Usually each drawing file contains one and only one image. For some applications, however, it is advantageous to store several images in a single file, particularly if the images are closely related to each other. Such a file is called a *multiframe drawing file*, and each image is called a *frame*. Once a drawing file has been opened in an application program, and a single image has been recorded in the usual way, additional frames can be appended using the DWG_NEXT (next frame) routine. For example, to make a 3-frame drawing file:

```
CALL DWG
...plot 1st frame...
CALL DWG_NEXT
...plot 2nd frame...
CALL DWG_NEXT
...plot 3rd frame...
CALL DWG_CLOSE
```

EDGR has commands for accessing any frame of a multiframe drawing for viewing, hardcopy, and data extraction. Facilities are also provided for building and decomposing multiframe drawings. The function Multiframe Make (command `FMAKE`) combines any set of drawings, single or multiframe, into a new multiframe drawing. The Multiframe Split function (command `FSPLIT`) generates a set of single frame drawings from a multiframe drawing.

It should be noted that multiframe drawing files can not be edited at the “element” level by EDGR. In order to do this, a multiframe drawing must first be decomposed using Multiframe Split into editable single frame drawings. If desired, the frames can later be recombined with Multiframe Make.

23 Interface to Other Graphics Systems

Programs or packages that do not use the GPLOT graphics library cannot use the integrated DWG routines to generate EDGR drawing files. However, EDGR provides a conversion facility that may allow drawing files to be generated indirectly, through the use of a plot file containing graphics output from the program or package. This plot file can be in one of the following formats: Hewlett-Packard HPGL, Houston Instruments DM/PL, or Tektronix 4010/4014. Most graphics systems support one or more of these formats. See Section II for information on how to convert these plot files to EDGR drawings.

Part II

REFERENCE GUIDE

24 Introduction

The remainder of this user's guide consists of documentation for the X/Motif interface main window, the command-mode interface, and the various EDGR functions. When using the X/Motif interface, it is easy to find out what functions are available by browsing through the various dropdown menus on the main window's menu bar. If you are using the command interface, it is useful to note the following commands which are helpful in getting started.

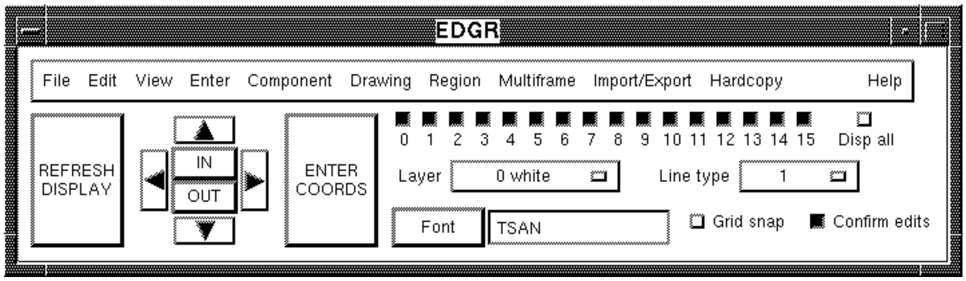
`COMM`	display command summary
`HELP`	enter help facility
`WHAT`	show drawing status
`NEWS`	show recent changes and new features

In the following sections, the functions are grouped in categories that roughly correspond to those of the dropdown menus on the main menu bar. The categories are:

- File
- Edit
- View
- Enter
- Component
- Drawing
- Region
- Multiframe
- Import/Export
- Hardcopy
- Help and utility

24.1 Main Window

The following table summarizes the various components of the main window and where to find information on them.



Component	Purpose	Reference Sections
Menu bar	Select an EDGR functions	II – II
REFRESH DISPLAY button	Re-draw current drawing	II
Pan and zoom buttons	Zoom window control	II and II
ENTER COORDS button	Numerical point entry	I
Layer toggles	Layer on/off	II
Disp all toggle	Display All mode on/off	II
Layer option menu	Select current layer	II
Line type option menu	Select current line type	I
Font button	Select current font	I
Grid snap toggle	Grid snap on/off	I, II
Confirm edits toggle	Verification on/off	II

24.2 Command syntax

In the command interface, a number of the commands accept or require additional parameters that are entered following the command name. For example, the command

```
OLD MYDRAWING
```

opens the existing drawing `MYDRAWING` for editing. The drawing name is a required parameter for the `OLD` command. In the following sections command names are shown in `UPPER CASE` and parameter names are shown in `lower case`. Optional parameters are enclosed in square brackets. For example, the syntax for the `HARD` (hardcopy) command is

```
HARD [drawing_name]
```

Typing the command `HARD MYDRAWING` will generate a hardcopy of `MYDRAWING` whereas typing `HARD` alone generates a hardcopy of the current drawing being viewed or edited.

If a *required* parameter is omitted from a command, EDGR displays a “template” showing the command syntax. Example:

```
EDGR > SAVE
Usage:     SAVE drawing_name       (use * for directory list)
EDGR >
```

The command line can then be recalled with the up arrow key (as in DCL or tcsh) and the required parameter can be added.

24.3 Command-line recall

Whenever EDGR prompts for a command, previous commands can be recalled by pressing the up arrow key. A recalled command can be edited, if desired, and entered by pressing the RETURN key. This recall/edit facility is similar to that of DCL in VMS and to that of the `tcsh` shell in UNIX. Up to 20 previous commands are stored.

The functions of the various keyboard keys in recalling and editing command lines are as follows:

Up arrow	scroll backwards through set of strings
Down arrow	scroll forwards through set of strings
Left arrow	move cursor left in current string
Right arrow	move cursor right in current string
Normal character keys	insert or overlay character at cursor location
BACKSPACE or CTRL-H	cursor to beginning of line
CTRL-E	cursor to end of line
CTRL-A	toggle insert/overstrike mode
CTRL-R	refresh current line
CTRL-U or CTRL-X	erase from cursor to beginning of line
DELETE	erase character to left of cursor
RETURN	enter the current line

On VMS, the insert/overstrike mode is initially set to be the same as that in the DCL environment. On UNIX it is initially set to insert.

24.4 Listing of drawing names

Commands which have `drawing_name` as a parameter have a built- in listing facility for existing drawing files. This is invoked by including an asterisk (*) in the entered name. For example, the command

```
OLD *
```

generates a list of all drawing files in the current directory. The command

```
OLD DSK1:[AAA.DRAW]DE*
```

will generate a list of all drawing file names beginning with `DE` in the directory `DSK1:[AAA.DRAW]`. Once the correct name is known, the command line can be recalled with the up arrow key and edited to complete the name specification.

24.5 Starting EDGR with an immediate command

The DCL or UNIX shell command line that runs EDGR can also contain any EDGR command, which will be executed immediately on start-up. Further commands are then typed in as usual. Examples of immediate commands are:

```
$ edgr old mydrawing
```

or

```
$ edgr hard mydrawing
```

This may be particularly useful when a VMS program runs EDGR in a subprocess (see Section I). For example, if a program produces a drawing file `MYDRAWING`, the calls

```
CALL DWG_CLOSE
ISTAT = LIB$SPAWN('EDGR HARD MYDRAWING')
```

will directly invoke EDGR's facilities to produce a hardcopy of the drawing, on any supported hardcopy device. Note: this example works only if the DCL symbol `EDGR` is properly defined as a *foreign command* that can take command-line arguments. See Appendix II for details.

> **NOTE:** On X Window displays, starting EDGR with a command on the command line will cause it to run in command mode, not with the X/Motif interface. This behaviour is provided for compatibility with some existing uses of EDGR in DCL and shell scripts. However, the behaviour may change in the future depending on user requirements.

25 File

The following functions deal with files, chiefly drawing files.

25.1 Edit new

Command: `NEW drawing_name`

This function creates and configures a new drawing file and makes it available for editing.

X/Motif: A file selector is brought up showing *existing drawings*. Type a new drawing name into the text field (the `.dwg` suffix can be omitted). Selecting an existing drawing name will begin a new version of the drawing under that name.

Command mode: The `drawing_name` parameter specifies the drawing to be created. As described in Section II, a list of existing drawings can be generated by including an asterisk (*) in the drawing name specification. The drawing name can then be edited by recalling the command line using the up arrow key. If a drawing with the specified name already exists, the user is prompted to confirm that a new version of the drawing should be created.

New versions of existing drawings: On VMS, files with the next higher version number will be created, rendering the old versions subject to deletion when the directory is purged. EDGR will only access the latest version of a drawing file. On UNIX systems, the old versions will be renamed as

`drawing_name.dwg~` and `drawing_name.dwt~`

removing any previous versions with these names.

When the new drawing file has been opened, the drawing format and drawing limits are entered. In X/Motif Mode, a Drawing Format dialog is brought up, whereas in Command Mode, the user is prompted to enter the format specification. The default format is landscape with drawing limits of 0–639 in X and 0–479 in Y. The default limits for portrait format are 0–479 in X and 0–639 in Y. For square format, the limits are 0–479 for both X and Y.

25.2 Edit old

Command: `OLD drawing_name`

This function opens an existing drawing file for editing.

X/Motif: A file selector is brought up showing existing drawings. If you manually type a name into the text field, the `.dwg` suffix can be omitted.

Command mode: The `drawing_name` parameter specifies the drawing to be opened. As described in Section II, a list of existing drawings can be generated by including an asterisk (`*`) in the drawing name specification. The drawing name can then be edited by recalling the command line using the up arrow key.

25.3 View

Command: `VIEW drawing_name`

This function opens an existing drawing file for *read only*. It is equivalent to the Edit old function except that all menu items or commands that may alter the drawing contents are disabled. Zoom, pan, hardcopy, and other read-only functions like Region Extract remain enabled, allowing the drawing information to be accessed while protecting it from inadvertent changes.

25.4 Close

Command: `CLOSE`

This function ends editing of the current drawing and closes the drawing file. It is necessary only when it is desired to end the editing of the current drawing without calling up a new drawing.

25.5 Compress

Command: `COMP`

During editing, the deletion of large numbers of drawing elements may result in large unused spaces in the drawing file. This function cleans up the drawing file, eliminating these spaces and minimizing the size of the drawing file.

On VMS, new versions of the files will be opened to receive the compressed data, and will become the current drawing after compression is complete. The old versions can be deleted or purged.

On UNIX, the old versions will be renamed as

`drawing_name.dwg~` and `drawing_name.dwt~`

removing any previous versions with these names.

During compression, a cleanup of component definitions will also take place. This removes all definitions of components that are not actually used in the drawing.

Drawings are also compressed when they are backed up with the Save as function (`SAVE` command).

25.6 Save as

Command: `SAVE drawing_name`

This function saves the contents of the current drawing into a new drawing file, providing a back-up copy of the current drawing. Since editing is done directly on the current drawing file (there is no intermediate work file or buffer), it is wise to use this function frequently when editing valuable drawings.

X/Motif: A file selector is brought up showing *existing drawings*. Type a new drawing file name into the text field (the `.dwg` suffix can be omitted). If you select an existing drawing file, a new version of it will be created, as described above for the Edit new function.

Command mode: The `drawing_name` parameter specifies the drawing file to be created. As described in Section II, a list of existing drawings can be generated by including an asterisk (`*`) in the drawing name specification. The drawing name can then be edited by recalling the command line using the up arrow key. If a drawing file with the specified name already exists, the user is prompted to confirm that a new version of the drawing file should be created.

During the save operation, the drawing is also compressed, as described in the previous section.

25.7 List drawings [command mode only]

Command: `LIST [drawing_spec] [list_file]`

This command displays a list of drawing files in a directory or directory tree, as specified by the `drawing_spec` parameter. This parameter can contain asterisk (`*`) wild cards. If the parameter is omitted, a list of all drawings in the current directory will be displayed. Optionally, the drawing list can also be written to the file specified by `list_file`. If the list file parameter is given, the drawing spec parameter must also be given. Example:

```
LIST * ALL.LIS
```

puts a list of all drawings in the current working directory into the file `ALL.LIS`.

A list file can be used as input to the `FMAKE` function which appends multiple drawings into a single drawing file (see Section II).

UNIX Note: the command `LIST layout` will look for a drawing called `layout` whereas `LIST layout/*` will list all drawings in the directory `layout`.

25.8 Execute script

Command: `@filename`

This function executes a specified script file. The default file type is `.scr`. For X/Motif, a file selector is brought up for choosing the script file.

The use of scripts is described in Section I.

25.9 Make survey file

Command: `SURVEY [log_file]`

This function allows selected drawing elements to be recorded in a text file. For each element, the file contains an identification tag and a list of coordinates for the element. This log file can then be used for reference or for reading by other programs.

X/Motif: a file selector is brought up for choosing the log file name, after which the Survey dialog will be brought up. Subsequently, logging will continue in the same log file unless the Start new log file button is clicked.

Command mode: the `log_file` parameter is required for the first `SURVEY` command and is optional thereafter. If the log file name is given, any current log file is closed and a new file is opened with the specified name. If the log file name is omitted, logging continues in the previous log file.

Once the log file is open for recording, the following actions are available:

Action	X/Motif	Command mode
Record element	`MB1`	—
Record line segment or arc	[toggle]	`MB1` or `SPACE`
Record polyline	[toggle]	`P`
Record single point	[toggle]	`S`
Write comment in log file	[button/textfield]	`C`
Edit log file	[button]	`E`

If verification is enabled, each selected element will be confirmed with the user prior to logging it in the file. The following describes the record format for each element type:

Single point	`S`	`x`	`y`		
Line segment	`L`	`x1`	`y1`		*start point*
		`x2`	`y2`		*end point*
Polyline	`P`	`x1`	`y1`		*start point*
		. . .			*intermediate points*
		`xn`	`yn`		*end point*
Arc	`A`	`xc`	`yc`	`ang`	*center point, vertex angle*
		`x1`	`y1`		*start point*
		`x2`	`y2`		*end point*
	`&`	`t1`	`t2`	`t3`	*X transformation*
		`t4`	`t5`	`t6`	*Y transformation*

Each record is echoed on the screen as it is written to the log file. The arc records marked by "&" are included only if the arc has a transformation associated with it. In this case the arc coordinates are transformed by:

$$X' = t_1 X + t_2 Y + t_3$$

$$Y' = t_4 X + t_5 Y + t_6$$

This mapping is used internally by EDGR to render ellipses and other non-circular arcs.

Two utility functions are provided: the Write comment in log file button (or `C` key in command mode) allows a comment to be typed in and recorded in the log file. The Edit log file button (or `E` key in command mode) invokes the EDT or Emacs editor on the log file, allowing corrections and annotations to be made during the recording process.

25.10 Quit

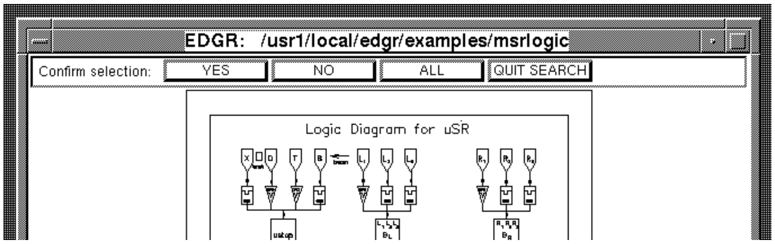
Command: `QUIT` or `STOP`

This function closes any open drawing file and exits from EDGR. Typing CTRL-Z on VMS or CTRL-D on UNIX has the same effect as the `STOP` command.

26 Edit

The following functions perform the deletion, modification, and movement of drawing elements.

In general, the user enters a point on or near the object (line segment, arc, or text) to be edited. For line and arc editing functions, the edit tolerance area that is being searched will be displayed as a rectangle around the specified point. If verification is enabled, elements that are selected will be marked with small rectangles as they are found. As each candidate element is found and marked, the user will be asked how to proceed. In the X/Motif interface, this is done by popping up a set of response buttons into the text area at the top of the graphics window:



Edit Confirmation

The same buttons will appear at the top of the zoom window if there is one.

In command mode, the user is prompted to enter a single-key response:

`y/N/q > Y`	O.K. to edit the element
`y/N/q > N` or RETURN	Do not edit the element
`y/N/q > Q`	Quit

For the *area* editing functions, where elements are selected from a rectangular area specified by the user, an additional editing option is available when verifying:

`y/N/q/all > A`	Edit the element and all subsequent elements in the area.

Selecting the `all` option causes the candidate element, and all subsequent elements found in the area, to be immediately edited without further confirmation. This does not affect the verification status for subsequent editing operations.

If verification is not enabled, all elements that are selected will be automatically edited *without confirmation*.

26.1 Delete lines

Command: `DEL`

This function deletes line segments, arcs, and polylines (contiguous sequences of connected line segments) from the drawing. The user enters a point on or near the segment, arc or polyline to be deleted. In X/Motif a toggle is provided for selecting polylines, and in command mode pressing the `P` key will select a polyline. If verification is enabled, candidate elements will be marked and the deletion will be confirmed with the user.

26.2 Delete text

Command: `DELT`

This function deletes text elements from the drawing. The user will be prompted to select elements for deletion. This can be done in two ways:

	X/Motif	Command mode
Enter a point on the text element	`MB1`	`MB1` or `SPACE`
Specify a search string	[button/textfield]	`S`

In the first case, the point must lie within the bounds of the plotted text string. In the second case, a character string can be entered and all text elements containing the string will be searched for. The search string can contain keywords or hex codes (see Section I for details).

26.3 Delete components

Command: `DELC`

This function deletes component placements from the drawing. It does not affect component definitions, which remain loaded even if all placements are removed. The user enters a point on each component to be deleted. This point must lie within the bounding rectangle of the component.

26.4 Area delete

Command: `ADEL`

This function deletes all drawing elements within a rectangular area. The user enters two points to define the rectangular area on the screen. All line segments and arcs whose two end points lie within the area will be deleted. All text elements and components whose base point lies within the area will be deleted. Elements in the area can be selectively deleted by ensuring that verification is enabled during the procedure.

26.5 Area cut lines

Command: `ACUT`

This function cuts a rectangular “hole” out of the drawing. The user specifies the rectangular area by entering two points. All line segments that lie within the rectangle will deleted. For line segments that intersect the edge of the rectangle, the portion that lies inside the rectangle is deleted, and the portion outside remains.

> **NOTE:** This procedure is performed *without* element-by-element verification.

26.6 Modify lines

Command: `MOD`

This function modifies the attributes and location of line segments, arcs and polylines. The location of single points (endpoints of line segments) can also be modified. The user first selects the item to be modified as follows:

Action	X/Motif	Command mode
Select element	`MB1`	—
Select line segment or arc	[toggle]	`MB1` or `SPACE`
Select polyline	[toggle]	`P`
Select single point	[toggle]	`S`

Once an item has been selected, it is modified using the following actions:

Action	X/Motif	Command mode
Relocate	`R`	`R`
Adjust arc radius	`A`	`A`
Modify layer	[option menu]	`C`
Modify line type	[option menu]	`L`
Vector move	[button]	`V`
New item	[button]	`E`

After each modification, the item is redrawn and additional modifications can be made.

Relocate: moves the item from its former location to the current location of the pointer.

Adjust arc radius: alters the radius of an arc, without changing its center location, so that the arc passes through the current pointer location.

Modify layer: Moves the element to a different layer. This action is not valid for single points.

Modify line type: changes the line type of the element. This action is not valid for single points.

Vector move: moves the item by amounts DX and DY entered by the user. When relocating line segments or single points, any connected line segments will be re-drawn to the new positions.

New item: ends the modifications of the current item and allows a new item to be selected.

26.7 Area modify lines

Command: `AMOD`

This function is used to change the attributes (layer number and line type) of a group of line segments. The user first enters a layer number and line type index to be applied to the elements. Then a rectangular area is specified around the elements to be modified. Only line segments and arcs whose end points lie within the area will be modified. As each element is modified it will be re-drawn with the new layer colour and line type.

26.8 Modify text

Command: `MODT`

This function is used to change the height, angle, font, colour, string contents, and location of text elements. First an element is selected, which can be done in two ways:

	X/Motif	Command mode
Enter a point on the text element	`MB1`	`MB1` or `SPACE`
Specify a search string	[button/textfield]	`S`

Then it can be modified as follows:

Action	X/Motif	Command mode
Relocate	`R`	`R`
Adjust height	`Z`	`Z`
Modify string	[text field]	`S`
Modify height	[text field]	`H`
Modify angle	[text field]	`A`
Modify font	[popup list]	`F`
Modify layer	[option menu]	`C`
Set justification	[toggle]	`J`
Freeze X or Y	[toggle]	`!`
Vector move	[button/textfield]	`V`
List font keywords	[button]	`?`
New item	[button]	`E`

Relocate: moves the text element to the current location specified by the pointer. The text will be positioned according to the current justification mode set in the dialog box or by the `J` key.

Adjust height: adjusts the height of the text element so that the text just fits in a rectangle defined by the element's justification point and the current pointer location.

Modify string: used to change the string contents of the text element. First, the current text string is displayed. If the text string contains any characters other than the standard keyboard characters, these will be shown as *keywords* and/or *hex codes*, depending on which font is being used. See Section I for information on the encoding of text strings using keywords and hex codes. **X/Motif:** the string is placed in a text field that can be edited at will. Click APPLY to enter the new string. **Command mode:** the string contents are put into an on-screen editing buffer. This buffer can be edited using the arrow keys in the same way that DCL command lines can be edited (see Section I for details). Once the string has been edited as desired, press `RETURN` to enter the new string.

Modify height: sets the height of the text element to the new value entered.

Modify angle: sets the angle of the text element to the new value entered.

Modify font: changes the font of the text element. **X/Motif:** a popup font list is displayed. Click on the new font. **Command mode:** enter the new font number.

Modify layer: changes the layer number of the text element.

Set justification: sets the justification mode for subsequent relocation of the text element. The default mode is **CC** (center-center).

Freeze X or Y: used to "freeze" the X or Y coordinate of the text element, so that the **Relocate** action will move the text only in a horizontal or vertical direction.

Vector move: used to move the text element by a precise amount $(\Delta x, \Delta y)$ where Δx and Δy are drawing coordinate values entered by the user.

List keywords: lists the valid keywords for the current font. These keywords can be used to enter special characters such as Greek letters and math symbols. See Section I for details.

New item: ends the sequence of modifications for the current element and allows a new element to be selected for modifications.

26.9 Area modify text

Command: `AMODT`

This function is used to modify the parameters of a collection of text elements. The parameters that can be modified are font, height, angle and layer number. When modifying font, height or angle, a justification code can also be specified. This allows the justification of text elements to be preserved in spite of changes to their size or orientation.

The user first chooses which parameters are to be modified and enters parameter values that will be assigned to the selected elements. Then a rectangle is entered around the text elements to be modified. Elements can be modified selectively if verification is enabled.

26.10 Modify components

Command: `MODC`

This function is used to modify the characteristics of any component. First, the component is selected by entering a point within its bounding rectangle. Then, the component can be modified as follows:

Action	X/Motif	Command mode
Relocate	R	R
Adjust size	A	A
Adjust angle	T	T
Change component type	[selection box]	N
Set scale factors	[text fields]	S
Commensurate on/off	[toggle]	=
Set angle	[text field]	L
Set layer	[option menu]	C
Set own colours flag	[toggle]	F
Base point on/off	[toggle]	B
New item	[button]	E

Relocate: moves the component so that its base point coincides with the current pointer location.

Adjust size: dynamically adjusts the scale factors of the component. The component is sized so that it just touches the horizontal and vertical boundaries defined by the current pointer location. The component's base point remains fixed. In each dimension, the sign of the scale factor will be determined by whether the pointer is on the positive or negative side of the base point. If commensurate scaling is in effect, the scale factors are forced to be equal in absolute value, but can differ in sign depending on the position of the pointer with respect to the base point.

Adjust angle: dynamically adjusts the rotation angle of the component. The angle is set equal to the angle of a line connecting the component's base point to the current pointer location. This angle is measured counterclockwise with respect to the positive X axis.

Change component type: the new type is selected from the list of loaded components.

Set scale factors: used to change the X and Y scale factors of the component. If commensurate scaling is enabled, only a single scale factor is set.

Commensurate on/off: enables and disables commensurate scaling. When commensurate scaling is in effect, a single scale factor is used for both X and Y scaling of the component.

Set angle: allows the rotation angle to be changed. This angle is measured in degrees counterclockwise with respect to the positive X axis. Rotation is always performed after scaling, and is done with respect to the component's base point.

Set layer: changes the layer number of the component. The component will be redrawn in the colour assigned to that layer, unless its "own colours" flag is set.

Set own colours flag: if this flag is set for the component, it will be drawn using its own colours rather than its layer colour. This allows the use of multi-coloured components. The component's own colours are those that were used in the original drawing that defined the component.

Base point on/off: enables and disables the marking of the base point for the component being modified. The base point marker is a visual aid and is not part of the drawing data.

New item: ends the modification of the current component and allows selection of a new component.

26.11 Smooth polyline

Command: `SMOOTH`

This function will render any polyline into a smooth curve. This is done by fitting a spline to the polyline vertices. After selecting a polyline with the pointer, the user will be asked to enter parameters for the spline:

Tension: controls the smoothness of the curve. A higher tension (e.g. 10) gives a smoother curve that lies farther from the vertex points; a lower tension (e.g. 0.1) increases the conformance of the curve to the polyline. Default = 1.

Endpoint weighting factor: controls the endpoints of the curve. The higher this value, the closer the endpoints lie to the endpoints of the polyline. Default = 10.

Number of output points: a larger value gives a smoother rendition of the spline curve, a lower value gives a choppier curve but economizes on the amount of drawing data generated. Default = 10 × number of input points. Note: a smoothed curve can actually be coarsened again by making the number of output points less than the number of input points.

After the parameters are entered the resulting curve is drawn. The user can can reject it and try again with different parameters, or accept it, causing it to replace the original polyline. The result is a new polyline that can again be modified using the Smooth polyline function, if desired.

26.12 Convert to lines

Command: `STROKE`

This function will convert text, arc or component elements to the **lines** data type. The individual strokes making up the element will be stored in the drawing as line segments and will be treated as such in all subsequent editing. It should be realized that once an element has been converted to line segments it loses its identity as a text, arc, or component element. This function is useful when scaling text: normal text can be scaled only in height, whereas text that has been converted to lines can be scaled in both height and width, e.g. with the Transform (`TRAN`) function.

When using this function, the user enters a point on the text string, arc or component to be converted. Alternatively, multiple elements can be selected by using an area selection. For each modified element, the element record will be deleted and the equivalent line segments will be added to the drawing file. As this is being done the element will be redrawn on the screen.

27 View

The following functions are used to control the graphics display.

27.1 Raise graphics

Command: *none*

Selecting this menu item causes the graphics window, and the zoom window if there is one, to be brought to the front of your display.

27.2 Clear graphics

Command: `CL`

This function clears the graphics screen or graphics windows. Note that on some terminals (e.g. VT640) this also causes the alphanumeric screen to be cleared. If a drawing is currently being edited, the drawing display can be restored after a clear using the REFRESH DISPLAY button or the `DISP` command.

27.3 Clear alpha [command mode only]

Command: `ACL`

This command clears the alphanumeric screen or window without affecting the graphics screen or window. In graphics input mode an alpha clear can be accomplished by pressing the `/` key.

27.4 Refresh display

Command: `DISP`

This function clears the screen or graphics window and re-displays the current drawing. It is useful for bringing the display up to date after extensive editing has taken place. If you are using the command interface and are in some editing function that uses graphics input mode, you can also refresh the display by pressing the `\` key.

27.5 Zoom

Command: `ZOOM [zoom_factor]`

This function specifies a rectangular portion of the drawing area to be displayed. On terminals the screen will be erased and the contents of the rectangular area will be displayed so as to occupy the entire screen, thus producing a magnified view. On workstations or X terminals the full drawing remains displayed in the main graphics window and a separate zoom window is opened to display the magnified zoom area. See Section I for details.

There are 5 ways to specify the zoom area:

Method	Command mode button/key
1 Enter corner points	`MB1` or `SPACE`
2 Enter center point and magnification	`C`
3 Enter center point and edge point	`E`
4 Enter tile address	`T`
5 Specify zoom factor	—

In method **1**, any two corner points may be entered in any order. The rectangle will be adjusted by expanding in the +X or +Y direction until it matches the screen aspect ratio. This is the default way to specify the rectangle.

In method **2**, a center point for the rectangle is entered and a numerical zoom magnification is specified. The magnification is defined with respect to the full drawing area, which has magnification 1. In command mode, the magnification of the previous zoom can be retained by pressing `RETURN` instead of entering a new value.

In method **3**, a center point and an edge point of the new rectangle are entered. This specifies a unique rectangle because of the screen aspect ratio requirement.

In method **4**, it is assumed that the drawing area is divided into rectangular *tiles* by a regular grid. On X window displays, the tile to be zoomed in on is determined by the pointer location. On non-X terminals, the tile row and column address must be entered. The new zoom rectangle will be centered around the tile. See Section II for information on how tiles are defined and inserted in a drawing. In command mode, if the tiling scheme has not been defined by a previous `TILE` command, the user is asked to enter the number of tile rows and columns before entering a tile address.

In method **5** a zoom factor is read from the command line or from the Zoom dialog, and the magnification of the current zoom is multiplied by this factor. The zoom rectangle is kept centered at the same location.

In command mode, after the new zoom rectangle has been specified it will be shown on the screen, if possible, so that the user can accept it or reject it and enter a different one. Once the rectangle has been accepted, the zoomed view will be displayed.

In command mode, an additional action key `P` will return the display to the previous zoom rectangle (the one that was in effect before the current one). This function can be used to "toggle" back and forth between two views.

The current magnification and current zoom coordinates can be determined at any time using the Show status menu item or `WHAT` command.

When using EDGR with an X Window display, the zoom window can also be opened or changed by pressing and holding the left mouse button and dragging open (in the main graphics window) a box to indicate the zoom area. If you are using the command interface, EDGR must be in graphics input mode for this to work.

27.6 End zoom

Command: `FULL`

On a terminal, this command is used to return from a zoomed or panned view of a drawing to a display of the entire drawing.

On workstations or X Window terminals, this function will close the zoom window and leave only the main graphics window which shows the entire drawing. A new zoom window will be opened the next time the Zoom function is used.

27.7 Pan

Command: `PAN`

This function is used to move the zoom rectangle around in the drawing area so that different portions of the drawing are displayed. A zoom in/out feature is also provided.

Action	X/Motif	Command mode
Pan to pointer	$\boxed{\texttt{MB1}}$	$\boxed{\texttt{MB1}}$ or $\boxed{\texttt{SPACE}}$
Pan left	[arrow button]	$\boxed{\texttt{L}}$ (or $\boxed{\leftarrow}$ on X displays)
Pan right	[arrow button]	$\boxed{\texttt{R}}$ (or $\boxed{\rightarrow}$ on X displays)
Pan up	[arrow button]	$\boxed{\texttt{U}}$ (or $\boxed{\uparrow}$ on X displays)
Pan down	[arrow button]	$\boxed{\texttt{D}}$ (or $\boxed{\downarrow}$ on X displays)
Vector pan	[button/textfields]	$\boxed{\texttt{V}}$
Zoom in	[button]	$\boxed{\texttt{I}}$
Zoom out	[button]	$\boxed{\texttt{O}}$
Set pan/zoom factors	[button/textfields]	$\boxed{\texttt{S}}$

Pan to pointer: moves the zoom rectangle so that it is centered at the current pointer location.

Pan left, right, up, down: moves the zoom rectangle left, right, up or down by a predetermined fraction DX or DY of its width or height. The default fraction is 0.4, which can be changed in the Pan dialog or using the $\boxed{\texttt{S}}$ key. **X/Motif note:** for ease of access the pan arrow buttons are located in the main window.

Vector pan: moves the rectangle in both X and Y by the predetermined fractions DX and DY of its width and height. These fractions have a default value of 0.4, which can be changed using the $\boxed{\texttt{S}}$ key.

Zoom in/out: zooms in or out by a predetermined zoom factor. The default zoom factor is 2, that is, the magnification is doubled when zooming in and halved when zooming out. This factor can be changed in the Pan dialog or by pressing the $\boxed{\texttt{S}}$ key. More comprehensive zooming facilities are offered by the Zoom menu item or the `ZOOM` command (Section II).

Set pan/zoom factors: sets the pan fractions DX and DY, and the zoom factor. DX and DY can lie between -1 and 1. A negative sign is ignored when using the X/Motif interface's arrow buttons or the $\boxed{\text{L}}$ $\boxed{\text{R}}$ $\boxed{\text{U}}$ $\boxed{\text{D}}$ keys, but is used for the vector pan. The default pan fractions are $DX = 0.4$, $DY = 0.4$.

27.8 Layer control

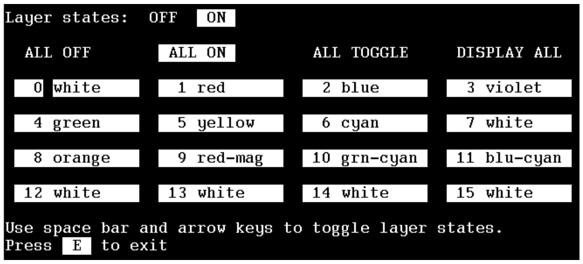
Command: `LAYER [list state]`

The system of drawing layers, described in Section II, allows graphical elements to be logically grouped together in independent sets. This allows selective control, by layer, over which drawing elements appear in the display and hardcopy and which drawing elements will be affected by the editing commands.

By default, all layers are **on**, that is, all elements are displayed and editable. If a layer is turned off, elements in that layer will not appear in the display or the hardcopy, unless the DISPLAY ALL option is selected. In this case, all elements are displayed regardless of which layers are on or off.

In X/Motif, the layer states are controlled by simple toggles in the main window, and a toggle for the DISPLAY ALL option. More efficient switching of multiple layers can be achieved via the Layer control menu item in the View menu. To show the effect of changes in the layer states, the REFRESH DISPLAY button must be clicked to re-draw the drawing.

For non-X terminals, layer control is provided through the LAYER command. If this command is given with no arguments, a layer menu is brought up. The layer menu shows the state of each layer, as well as its display colour, which depends on what type of terminal is being used. If a monochrome terminal is being used, the CIT467 colour assignments will be shown as a reference.



```
Layer states:  OFF  ON

  ALL OFF       ALL ON        ALL TOGGLE     DISPLAY ALL

  0 white       1 red         2 blue         3 violet

  4 green       5 yellow      6 cyan         7 white

  8 orange      9 red-mag     10 grn-cyan    11 blu-cyan

  12 white      13 white      14 white       15 white

Use space bar and arrow keys to toggle layer states.
Press  E  to exit
```

Layer Menu

The LAYER command also accepts command-line arguments. These can be used to change layer states from a script file or in other cases where it is not desirable to use the layer menu. The possible formats are:

```
LAYER list ON|OFF
LAYER ALL ON|TOGGLE
LAYER DISPLAYALL ON|OFF
```

where `list` consists of a series of layer numbers or layer ranges separated by blanks. Example:

```
LAYER 0 5 6-9 12 OFF
```

> **NOTE:** When the layer menu is used to change the state of some layers, the drawing display will be *automatically updated* to reflect the new layer states after exiting from the menu. However, when layer states are changed through the *command-line arguments*, the display will not be updated until an explicit `DISPLAY` command is given.

Elements in layers that are off will not be affected by any of the editing functions such as deletion, modification, transformation, etc. This applies even in DISPLAY ALL mode, so elements can be seen but can be protected from editing changes.

In command mode, if any layers are off, a *status line* will appear at the top of the screen and remain there until such time as all layers are turned on again. In the status line, the layer number for each layer that is on will be highlighted. The layer number for the current entry layer will also be underlined.

The current entry layer must always be a layer that is *on*. If this layer is subsequently turned off, the lowest-numbered layer that is on becomes the new entry layer. At least one layer must be on at all times. The status line also shows whether DISPLAY ALL is in effect.

When including or extracting other drawing files (Extract, Insert and Insert tiles functions) the effect of layering is basically "what you see is what you get." If DISPLAY ALL is in effect, all layers will be extracted or inserted. If DISPLAY ALL is not in effect, only visible layers (i.e. the layers that are on) will be extracted or inserted, and a warning message to that effect will be issued.

27.9 Grid setup

Command: `GRID`

This function controls the settings for a displayed dot grid and an invisible snap grid (see Section I). By default, both these grids are off. The grids share a common origin, but the spacing of each can be set independently. When the dot grid is on, it will be automatically displayed each time the current drawing is displayed. When the snap grid is on, points entered with the mouse or crosshairs will automatically be snapped to the nearest grid point. This feature allows drawing elements to be entered at precise locations or intervals and is useful for aligning objects with each other. For convenience during point entry, the grid snap can be turned on and off at any time, by clicking on the X/Motif Grid snap toggle or by using the $\boxed{\text{G}}$ action key available during any graphics input. Points that are entered by entering coordinates at the keyboard (ENTER COORDS dialog or $\boxed{\text{K}}$ action key) will not be affected by grid snap.

In X/Motif the grid settings are controlled by the Grid setup menu item, which brings up a dialog box. In command mode the GRID command displays a table showing the spacing for each grid, the status of each grid (on or off), and the grid origin. These parameters can be changed using three commands:

`D [x] [y]`	Sets the spacing and status of the dot grid
`S [x] [y]`	Sets the spacing and status of the snap grid
`I`	Sets the isometric grid option
`O x y`	Sets the grid origin
`C number`	Sets the grid colour

The `D` and `S` commands both work the same way. If `x` and `y` are given, the grid is activated with horizontal spacing `x` and vertical spacing `y`. If only `x` is given, both horizontal and vertical spacings will be set to `x`. If `x` and `y` are omitted, the grid is toggled on or off without changing the spacings.

The `I option` enables a grid suitable for isometric drawings and allows one to easily enter lines at multiples of 30 degrees from the horizontal. This grid consists of equilateral triangles. The X spacing, as entered with the `D` and `S` commands, determines the edge length of the triangles. The Y spacing is not used.

The `O` command sets the origin of both grids to `(x,y)`. Both `x` and `y` must be specified. By default, the origin is at `(0,0)`.

The `C` command sets the grid colour. The colour is set by specifying a layer number. The grid will be drawn in the display colour assigned to that layer. The default colour is that assigned to layer 0. The list of layer numbers and display colours will be shown if the `number` parameter is omitted.

After each command, the grid table will be updated to show the new parameters. Then another command can be entered or the `GRID` function can be exited by pressing `RETURN`.

27.10 Fast text on/off

Command: `FAST`

When fast text display is enabled, all text elements in the drawing will be displayed as rectangles rather than plotted characters. This speeds up the display considerably and is useful in cases where a drawing is being called up only for hardcopy or for minor editing where the contents of text elements need not be shown. The height, width, and angle of the rectangles correspond to those of the plotted text strings. The fast text mode is turned on and off using the Fast text toggle or the `FAST` command, which sets the mode to the opposite of its previous state. When fast text mode is in effect and a text element is selected for editing, the element is redrawn as characters before proceeding with verification or editing.

27.11 Show border

Command: `BORDER`

When the edges of the display area do not coincide with the drawing limits of the current drawing, by default a dashed border is displayed to show the drawing limits. The Border toggle or `BORDER` command can be used to disable this border display. This is useful when making direct screen hardcopies on a printer attached to the terminal. In this case, the border may not be wanted on the hardcopy. The border display can be re-enabled with the toggle or by typing the `BORDER` command a second time.

27.12 Activate crosshairs [command mode only]

Command: `CROSS`

This command enters graphics input mode but does not perform any editing functions on the drawing. It is provided in order to make available the coordinate display and measurement functions without the risk of making unwanted changes to the drawing.

See Section I for a description of the standard action keys available in graphics input mode.

28 Enter

The following functions are used to enter new elements into the current drawing.

28.1 Enter lines

Command: `LINE`

In this function, the user enters a series of points to define line segments which are entered into the drawing. Each connected series of segments is called a **polyline** and is defined by a start point and a series of end points.

Action	X/Motif	Command mode
Start/continue polyline	`MB1`	`MB1` or `SPACE`
Start new polyline	`S`	`S`
Continue horizontal	`H`	`H`
Continue vertical	`V`	`V`
Duplicate polyline	`=`	`=`
Relocate point	`A`	`A`
Relocate polyline	`R`	`R`
Update parameters	[button]	`U`
Delete	[button]	`D`
Close polygon	[button]	`P`
Fill polygon	[button]	`F`
Set layer	[option menu]	`C`
Set line type	[option menu]	`L`

Start/continue polyline: The first point entered specifies a start point, which will be connected by a line segment to the second point entered. Additional points can be entered to form a contiguous series of segments, called a *polyline*.

Start new polyline: begins a new polyline and establishes its start point at the current pointer location. A temporary marker will be placed at the start point and will be removed when the first line segment is drawn.

Continue horizontal: draws a horizontal line from the previous endpoint to the current X coordinate of the pointer.

Continue vertical: draws a vertical line from the previous endpoint to the current Y coordinate of the pointer.

Duplicate polyline makes a duplicate copy of the current polyline and adds it to the drawing, locating it so that it ends at the current pointer location.

Set layer: is used to set the layer number for lines subsequently entered.

Set linetype: is used to set the line type index for lines subsequently entered. **X/Motif:** the current line type is selected from an option menu in the main window.

Close polygon: closes the current polyline to form a polygon, by connecting the last end point to the start point. There must be at least 3 vertices to form a valid polygon. Subsequent points that are entered will continue the polyline from the closure point, and the closure operation can be repeated after more vertices are added.

Fill polygon: allows a polygon, as defined above, to be filled with crosshatch patterns. The polygon will be closed, if necessary, before filling. The crosshatch patterns are made up of hatch lines drawn at specified angles and spacings. For each set of hatch lines the user enters the angle (in degrees with respect to the positive X axis) and the spacing (in drawing units). This can be repeated to build up complex patterns. Each set of hatch lines is drawn and verified with the user before being added to the drawing data.

Relocate point: adjusts the last point entered to the present pointer location. The point and any associated segment will be re-drawn on the screen.

Relocate polyline: moves the current polyline in its entirety, so that it ends at the present pointer location.

Update: updates the parameters (layer number and line type) of the last segment entered to the current settings.

Delete: deletes the last point entered, together with any associated segment. This action can be repeated to delete any number of segments in the polyline, in reverse order.

28.2 Enter arcs

Command: `ARC`

Circles, ellipses, circular arcs, and regular polygons can be added to the drawing using this function. There are 10 arc entry modes, allowing arcs to be specified and entered in 10 different ways. The following table describes the entry modes. In each case, the arc is specified by entering one or more points. In some modes there is an additional radius or angle value that is used in generating the arcs.

Type	Mode	Entered points			Parameter
		p1	p2	p3	
CIRCLE	1	center	—	—	radius
CIRCLE	2	center	radius	—	—
CIRCLE	3	dia1	dia2	—	—
CIRCLE	4	circ1	circ2	circ3	—
ARC	5	start	end	—	radius
ARC	6	start	end	—	angle
ARC	7	center	start	—	angle
ARC	8	center	start	end	—
ARC	9	start	through	end	—
ELLIPSE	10	center	radius1	radius2	—

When arcs are drawn, they are approximated by line segments connecting a series of vertex points that lie at regular intervals on the arc. The angular increment between these points (measured with respect to the center of the arc) is called the *vertex angle*. This angle can be set for each arc by the user. The default value is 5 degrees which gives a reasonable approximation to a smooth curve in most cases. The smaller the angle is set, the better the curve looks and the longer it takes to display. N-sided regular polygons may be drawn by setting the vertex angle to $360/N$ degrees. For example, the vertex angle can be set to 60 degrees to generate hexagons.

Action	X/Motif	Command mode
Enter control point	`MB1`	`MB1` or `SPACE`
Relocate	`R`	`R`
Adjust	`A`	`A`
Duplicate	`=`	`=`
Fill	[button]	`F`
Update parameters	[button]	`U`
Delete	[button]	`D`
Set vertex angle	[text field]	`V`
Set layer	[option menu]	`C`
Set line type	[option menu]	`L`
Set entry mode	[toggles]	`E`

Enter control point: enters from one to three points which define the arc, according to the current entry mode.

Relocate: moves the current arc without changing its shape, so that the final defining point is at the pointer position.

Adjust: adjusts the current arc so that the final defining point is located at the present pointer position, while any other defining points remain at their previous locations. This can be used to change the size and curvature of the arc.

Duplicate: makes a duplicate copy of the last arc entered and adds it to the drawing at the current pointer location.

Fill: allows area fill, using crosshatch patterns, of the current arc. If the arc is not closed, the fill area can be defined by connecting the endpoints together (chord fill), or by connecting the endpoints to the arc center (sector fill). The crosshatch patterns are specified by a series of angle and spacing parameters for hatch lines. Each set of hatch lines is drawn and verified with the user before being added to the drawing data.

Update parameters: Updates the layer number, line type, and vertex angle of the current arc. If the entry mode uses a predetermined radius or angle, this is also updated. No change takes place in the defining points of the arc.

Delete: deletes the current arc.

Set vertex angle: sets the vertex angle for subsequent arcs. Generally speaking, the sign of the vertex angle determines the "sense" of the arc:

$vertex\ angle > 0$	arc will be drawn counterclockwise
$vertex\ angle < 0$	arc will be drawn clockwise

For entry modes 5, 6, 8 and 9, the arc is always drawn from the *start* point to the *end* point. Continuity with any connecting lines or arcs can be preserved by entering the points in the appropriate order (this may be important for pen plotters).

Set layer: sets the layer number for elements subsequently entered. **X/Motif:** the current layer is set in an option menu in the main window.

Set line type: sets the line type for elements subsequently entered. **X/Motif:** the current line type is set in an option menu in the main window.

Set entry mode: sets the arc entry mode and any associated parameters. For X/Motif, this is done using the toggle selections and text fields in the Arc dialog. In command mode, the user specifies the entry mode and then will be prompted for a radius or angle parameter if one is required. These parameters will apply to all arcs subsequently entered.

For editing purposes, arcs are grouped with lines as a single data type. The following editing functions apply to arcs as well as lines: Delete lines (`DEL`), Area delete (`ADEL`), Modify lines (`MOD`), and Area modify lines (`AMOD`).

If access to the individual chords that make up an arc is required, the arc can be converted to line segments using the Convert to lines function (`STROKE` command).

28.3 Enter text

Command: `TEXT`

This function enters new text elements into the drawing. In the X/Motif interface, the Text dialog pops up, the user enters a text string into the String field in the dialog, and then enters a point to position the text. In command mode, the user first enters the location point of the new text element, and then is prompted to enter the string.

In some cases, the text string must be specially encoded: for details on how to enter text strings see Section I. If the prompt for a text string is skipped by pressing `RETURN`, then no text element is entered and a return is made to graphics input mode.

Action	X/Motif	Command mode
Enter text	`MB1`	`MB1` or `SPACE`
Relocate	`R`	`R`
Adjust height	`Z`	`Z`
Update parameters	[button]	`U`
Delete	[button]	`D`
Set height	[text field]	`H`
Set angle	[text field]	`A`
Set alignment	[button]	`L`
Set justification	[toggle]	`J`
List font keywords	[button]	`?`
Set font	[popup list]	`F`
Set layer	[option menu]	`C`
Duplicate	—	`=`
Change string	—	`S`

Enter text: places a new text element at the current pointer location. The text will be justified according to the current justification mode.

Relocate: moves the current text element to the current pointer location. The current justification mode will determine the positioning of the text at its new location.

Adjust height: adjusts the height of the current text element so that the text just fits in a rectangle defined by the element's justification point and the current pointer location. The text will remain justified according to the current justification mode.

Update parameters: updates all the parameters of the current text element without changing its position.

Delete: deletes the current text element.

Set height: sets the height in drawing units for text subsequently entered.

Set angle: sets the angle in degrees of text subsequently entered.

Set alignment: allows the text angle to be defined by entering two points, rather than entering a numerical value for the angle. This is useful for aligning text with other objects.

Set justification: sets the text justification mode, which determines how the entered text is placed with respect to the entered point. A two-character code is entered to set the justification. Examples:

LL (*lower left*)	the lower left corner of the text will coincide with the entered point (*default*)
CC (*center center*)	text will be centered horizontally and vertically with respect to the entered point

There are 9 justification codes in all, shown schematically as follows:

UL	UC	UR
CL	CC	CR
LL	LC	LR

where the rectangle indicates the bounds of the plotted text string.

> **NOTE:** The justification mode does not affect the *base point* (origin) of a text element, which is always at the *lower left corner*. This base point is the location that is associated with the text element for all subsequent editing functions.

List font keywords: lists the valid keywords for the current font. These keywords can be used to enter special characters such as Greek letters and math symbols. See Section I for details.

Set font: sets the font for text subsequently entered. **X/Motif:** the current font is set using the Font button in the main window.

Set layer: sets the layer number for elements subsequently entered. **X/Motif:** the current layer is set using an option menu in the main window.

Duplicate: enters a new text element at the current pointer location, using the same the text string as the previous element. **X/Motif**: entering multiple points with `MB1` will make multiple copies of whatever text has been entered into the String field in the Enter Text dialog.

Change string: is used to change the last text string entered. The string is displayed in an on-screen edit buffer and is modified as desired, after which `RETURN` is pressed and the text element is updated and re-drawn. **X/Motif:** to change the text string, edit the String text field and click the Update button.

28.4 Enter components

Command: `PLACE`

A component is any collection of graphics objects that has been grouped as a single entity. This is done using the Component Make (`CMAKE`) function (Section II). Components to be placed must first have their definitions loaded into the current drawing. This is done with the Component Load `LOAD` function (Section II).

Initially, the user selects one of the loaded components to be entered. Then the following actions can be performed:

Action	X/Motif	Command mode
Enter component	`MB1`	`MB1` or `SPACE`
Relocate	`R`	`R`
Adjust size	`A`	`A`
Adjust angle	`T`	`T`
Update parameters	[button]	`U`
Delete	[button]	`D`
Commensurate on/off	[toggle]	`=`
Set scale factors	[text fields]	`S`
Set angle	[text field]	`L`
Set layer	[option menu]	`C`
Set own colours flag	[toggle]	`F`
Base point on/off	[toggle]	`B`
Set component type	[selection box]	`N`

Enter component: enters a new component with its base point at the pointer location. The component is entered into the current layer and is scaled by the current X and Y scale factors and then rotated by the current rotation angle.

Relocate: moves the current component so that its base point lies at the current pointer location.

Adjust size: dynamically adjusts the scale factors of the component. The component is sized so that it just touches the horizontal and vertical boundaries defined by the current pointer location. The component's base point remains fixed. In each dimension, the sign of the scale factor will be determined by whether the pointer is on the positive or negative side of the base point. If commensurate scaling is in effect, the scale factors are forced to be equal in absolute value, but can differ in sign depending on the position of the pointer with respect to the base point.

Adjust angle: dynamically adjusts the rotation angle of the component just entered. The angle is set equal to the angle of a line connecting the component's base point to the current pointer location. This angle is measured counterclockwise with respect to the positive X axis.

Update parameters: for the current component, updates the component type, scale factors, rotation, layer number, and colour flag to their current settings.

Delete: deletes the current component.

Commensurate on/off: enables and disables commensurate scaling. When commensurate scaling is in effect, a single scale factor is used for both X and Y scaling of the component.

Set scale factors: used to set the X and Y scale factors for subsequent entries. If commensurate scaling is enabled, only a single scale factor is set.

Set angle: allows the rotation angle to be set. This angle is measured in degrees counterclockwise with respect to the positive X axis. Rotation is always performed after scaling, and is done with respect to the component's base point.

Set layer: Sets the layer number for subsequent entries. **X/Motif:** the current layer is set using an option menu in the main window.

Set own colours flag: if this flag is set for a component, it will be drawn using its own colours rather than the layer colour. This allows the use of multi-coloured components. The component's own colours are those that were used in the original drawing that defined the component.

Base point on/off: enables and disables the marking of the base point for each component placed. The base point marker is not part of the drawing data but provides a visual aid to the placing of components.

Set component type: allows a new component type to be selected for subsequent entries. This is selected from a list of loaded components.

28.5 Area fill

Command: `FILL`

This function allows the filling of a polyline, arc, or polygonal area by a crosshatch pattern (sets of regularly-spaced parallel lines). The user has the option of selecting an existing arc or polyline to be filled, or of defining a polygonal area to be filled.

In the first case, the user enters a point on or near the arc or polyline to be filled. In the second case, the polygonal area to be filled is defined by entering the polygon vertex points in order.

Once the area to be filled has been defined, the user then enters an angle and spacing for a set of hatch lines. The angle is in degrees with respect to the positive X axis and the spacing is in drawing units. This process can be repeated to build up complex patterns. Each set of hatch lines will be confirmed with the user before it is added to the drawing file.

The hatch patterns are entered as a series of line segments in the drawing file, and can hence be edited later using such functions as Delete lines (`DEL`), Area delete (`ADEL`), Area cut lines (`ACUT`), Modify lines (`MOD`), and Area modify lines (`AMOD`).

28.6 Enter Arrows

Command: `ARROW`

This function generates the appropriate set of line segments to create arrows and arrowheads in the drawing.

To enter an arrow or arrowhead the user enters two points to define a directed line segment. There are three entry modes:

0	Draw the segment with an arrowhead at the end
1	Draw the segment with arrowheads at the beginning and end
2	Do not draw the segment but draw an arrowhead at the end

The default entry mode is mode **0**.

Action	X/Motif	Command mode
Enter arrow	`MB1`	`MB1` or `SPACE`
Relocate	`R`	`R`
Adjust	`A`	`A`
Duplicate	`=`	`=`
Update parameters	[button]	`U`
Delete	[button]	`D`
Draw horizontal	[toggle]	`H`
Draw vertical	[toggle]	`V`
Set layer	[option menu]	`C`
Set arrow parameters	[toggles & text fields]	`P`

Enter arrow: enters the two endpoints determining the arrow or arrowhead.

Relocate: moves the current arrow so that its endpoint is at the current pointer location, without changing its length or angle.

Adjust: adjusts the current arrow by moving its endpoint to the current pointer location while keeping the start point fixed, thus changing the length and angle of the arrow.

Duplicate: makes a duplicate copy of the current arrow and adds it to the drawing at the current pointer location.

Update parameters: for the current arrow, updates its arrow parameters and its layer number to the current settings.

Delete: deletes the current arrow.

Draw horizontal: (used when entering the second point) draws a horizontal arrow from the first point to the horizontal coordinate of the current pointer location.

Draw vertical (used when entering the second point) draws a vertical arrow from the first point to the vertical coordinate of the current pointer location.

Set layer: sets the layer number for subsequent entries. **X/Motif:** the current layer is set using an option menu in the main window.

Set arrow parameters: sets the following parameters for subsequent arrows:

1. Arrow type
4. Arrowhead type (open or closed)
2. Length of arrowhead
3. Width of arrowhead

28.7 Enter Boxes

Command: `BOX`

This function allows rectangular boxes to be entered into the drawing. For each box, two points are entered to specify two opposite corners of the box.

Action	X/Motif	Command mode
Enter box	`MB1`	`MB1` or `SPACE`
Relocate	`R`	`R`
Adjust	`A`	`A`
Duplicate	`=`	`=`
Fill	[button]	`F`
Update parameters	[button]	`U`
Delete	[button]	`D`
Set layer	[option menu]	`C`
Set line type	[option menu]	`L`

Enter box: used to place a new box by entering two opposite corners of the box.

Relocate: moves the current box so that its second corner point is at the current pointer location, without changing the size or shape of the box.

Adjust: adjusts the current box by moving the second corner point to the current pointer location while keeping the first corner point fixed, thus changing the size and shape of the box.

Duplicate: makes a duplicate copy of the current box and adds it to the drawing at the current pointer location.

Fill allows area fill, using crosshatch patterns, of the current box. The crosshatch patterns are specified by a series of angle and spacing parameters for hatch lines. Each set of hatch lines is drawn and verified with the user before being added to the drawing data.

Update parameters: updates the layer number and line type of the current box to their current settings.

Delete: deletes the current box.

Set layer: sets the layer number for subsequent entries. **X/Motif:** the current layer is set using an option menu in the main window.

Set line type: sets the line type for subsequent entries. **X/Motif:** the current line type is set using an option menu in the main window.

28.8 Enter ID stamp

Command: `STAMP`

This function adds an "identification stamp" to the current drawing. The stamp is a line of text containing the full drawing file specification (for the DWG file), and the date and time. The text will be drawn in the lower right-hand corner of the drawing. The stamp is a standard text element and can be moved, modified or deleted at will.

29 Component

The following functions are used to create component files and load components into a drawing. For a general description of components, see Section I.

29.1 Component Make

Command: `CMAKE [file_name]`

This function makes a component file from the contents of the current drawing. This allows the drawing to be used as a component in other drawings, where it can be manipulated as a single graphic entity. The default file type for component files is `.cmp`. If `file_name` is omitted, it defaults to the current drawing name. Before making the component file, a base point for the component must be entered. This can be done with the mouse, or you can specify exact numerical coordinates using the the ENTER COORDS facility (`K` key in command mode).

Before making the component file, if is sometimes useful to scale the component drawing. For example, scaling the drawing to fit a 1×1 unit square, with the lower left corner of the square as the base point, will allow the component to be easily used in any drawing, since the scale factors will simply specify the component size in drawing units. To make a component with a central base point, a 2×2 square should be used. Example drawing formats are:

Drawing format	*Drawing limits*	*Component base point*
SQUARE	(0,0) to (1,1)	(0,0)
SQUARE	(-1,1) to (1,1)	(0,0)

> **NOTE:** it is recommended to store component files and their original drawings in a designated directory. The original drawing file should always be retained in case a component design needs to be changed. In this case, the original drawing can be edited, a new component file made, and the component definition can be re-loaded into any drawing that uses it.

29.2 Component Load

Command: `LOAD [component_file_name]`

This function loads a component definition into the current drawing from the specified component file. See the preceding section for information on making component files. Each loaded component is identified by a unique name of up to 12 alphanumeric characters. This name will be the same as the component file name, minus the suffix and any directory specifications. The default suffix for component files is `.cmp`.

In X/Motif, a file selector is brought up to specify the component file to be loaded.

If the component is already loaded, the user will be informed of this and asked if a re-load should be done. If so, the existing component definition will be replaced by the one in the file and all the instances of the component will be redrawn according to the new definition.

29.3 Component Maintenance

Command: `CMAINT`

This function allows the maintenance of the loaded component definitions in the current drawing. The options are:

0	List loaded components [command mode only]
1	Rename a component
2	Delete a component and all its instances

Option **0** lists the names of all components loaded in the current drawing. Option **1** changes the name of a loaded component without changing its definition. The old name and new name will be prompted for. Option **2** deletes a loaded component definition and all instances of the component in the drawing – use with caution!

30 Drawing

The following functions involve the current drawing status, editing criteria, and global changes to the drawing.

30.1 Show status

Command: `WHAT`

This function displays various information pertaining to the current drawing and the options and attribute settings that are in effect:

- Name of current drawing
- Current frame number if multiframe drawing file
- Drawing limits
- Drawing format
- Magnification and limits of zoom window
- Edit tolerance and verification status
- Current layer number and line type

30.2 Set edit tolerance

Command: `TOL`

This function is used to change the edit tolerance, which defines the size of the region around an entered point which is searched for candidate elements for editing. The current tolerance value is displayed and a new tolerance can be set by entering two points. The default tolerance is $1/50$ of the width of the display.

30.3 Change format

Command: `FORM`

This function can be used to change the drawing format and drawing limits, which define how the drawing appears on the screen and hardcopy. These concepts are described in Section I.

In X/Motif, a dialog is brought up allowing editing of the relevant switches and parameters.

In command mode, first a new format (portrait, landscape, or square) is entered. The current format can be left unaltered by pressing `RETURN`. A page count for multiple-page drawings (to be printed on the HP Paintjet) can also be entered if desired.

Then, new drawing limits can be defined for the current drawing. Changing the drawing limits does not affect the coordinates of the drawing elements themselves, but the scaling of elements on the screen and hardcopy will be different. The effect of altering the drawing limits is similar to that of “framing” a picture in a camera viewfinder.

After the new format and limits have been entered, the drawing is re-displayed to reflect the changes.

30.4 Move layers

Command: `CMOD`

This function moves all elements in a given layer to a new layer, thus giving them a new display colour. The user is prompted for an old layer number and a new layer number. The drawing is re-displayed to show the display colour change.

> **NOTE:** This function is performed without element-by-element verification.

30.5 Modify line types

Command: `LMOD`

This function is used to alter the drawing's line table, which determines how dashed lines (line types 2 through 10) are plotted. See Section I for a description of the line type definitions. In X/Motif mode, a dialog is brought up allowing editing of the line table parameters. In command mode, the user first enters the number of the line type to be altered, and then enters from one to three new parameters to define dash and space lengths. Parameters can be left unaltered by pressing RETURN. The drawing must be re-displayed to reflect changes in the line table.

30.6 Edit verification on/off

Command: `VER [ON|OFF]`

X/Motif note: edit verification is controlled by the Confirm edits toggle in the main window.

This function controls the confirmation of editing operations to be performed on individual drawing elements. When verification is **on** and elements are selected for some editing operation, the operation can be confirmed or denied for each element. This applies to most element editing functions. When verification is **off**, editing of selected elements will take place immediately without further interaction with the user.

By default, verification is enabled whenever a drawing is called up for editing. To see the current verification state, issue the `VER` command without parameters.

> **NOTE:** Verification does not apply to the Transform (`TRAN`) and Area cut lines (`ACUT`) functions.

> **NOTE:** When verification is turned off using the `VER OFF` command, it will *remain disabled* until a `VER ON` command is issued or a new drawing is called up for editing.

31 Region

The following functions operate on rectangular regions of the drawing. See Section I for information on different ways to specify these regions.

31.1 Move

Command: `MOVE`

This function moves all the elements in the specified region by the same amount. The move is specified by entering a reference point for the region and then entering a destination point.

31.2 Transform

Command: `TRAN`

This function allows the transformation of objects in a rectangular region (the "source rectangle") by the following methods:

1 Scale, rotate and translate by specified amounts
2 Transform to fit a rectangle
3 Transform to fit a rectangle with commensurate scaling
4 Three-point transformation (lines only)
5 Four-point transformation (lines only)

In method **1**, a reference point is entered on or near the region to be transformed, and a second reference point (possibly coincidental with the first) is entered to locate the transformed region. The user then enters scale factors (for X, Y, and text height) and a rotation angle. The transformation proceeds in the following order:

1. Scale with respect to first reference point
2. Rotate with respect to first reference point
3. Translate to second reference point

If the first and second reference points are coincidental, no translation takes place.

For methods **2** and **3**, two points are entered to specify a “destination rectangle” and elements from the source region will be scaled so that they fit in the rectangle. The exact scaling depends on how closely the *source rectangle* was entered around the original elements. In method **2**, elements will be scaled differently in X and Y for the best fit to the destination rectangle. In this case, a text height scale factor must be entered by the user, since text cannot be scaled differently in X and Y. In method **3**, a single scale factor is used so that aspect ratio and text positioning is preserved. These methods also allow an optional rotation by a multiple of 90 degrees.

Method **4** uses a transformation of the form

$$x \longrightarrow A + Bx + Cy$$

$$y \longrightarrow D + Ex + Fy$$

This transformation can be used to simulate 3-d effects such as projections with different viewing angles. The transformation is specified by entering three “source points” and three “destination points” that the source points will be mapped into by the transformation.

Similarly, Method **5** uses a transformation of the form

$$x \longrightarrow A + Bx + Cy + Dxy$$

$$y \longrightarrow E + Fx + Gy + Hxy$$

which is specified by entering four source points and four destination points. This transformation can be used to simulate 3-d effects such as perspective projections with one or two vanishing points.

NOTE: Transformation methods 4 and 5 affect lines only. Text, arcs, and components can only be transformed if they are first converted to lines using the Convert to lines function (STROKE command).

To speed up processing, the display of elements as they are transformed can be stopped at any time by pressing `CTRL-C`. This affects only the display and does not affect the transformation process, which will continue to completion.

> **NOTE:** The transformation function is performed without element-by-element verification.

31.3 Duplicate

Command: `DUP`

This function makes one or more duplicate copies of a rectangular region of the drawing. The user first specifies the region, usually by entering two corners of the rectangle. Then, a reference point is entered to specify a "base point" for the region. The user is then prompted to enter any number of reference points for the copies.

Line segments that cross the edges of the source rectangle will be clipped at the edges. Text and arcs will not be clipped. A text element will be copied if its origin point lies in the source rectangle, and arcs will be copied if they cross the source rectangle

31.4 Extract

Command: `EXT drawing_name`

This function is used to extract a rectangular portion of the current drawing into the specified drawing file. The user simply enters a rectangle (the extraction rectangle) around the region to be extracted.

The drawing limits for the extracted drawing can be fitted to the extraction rectangle (default) or they can be set the same as the current drawing. In the former case, the limits are set so as to preserve the aspect ratio of drawing elements.

The absolute coordinates of elements extracted to the new drawing will be preserved. Line segments will be clipped at the borders of the extraction rectangle, but there is no clipping for arcs or text: these will be extracted entire if their origin/end points lie in the extracted area.

31.5 Insert

Command: `INS drawing_name`

This function allows the contents of the specified external drawing to be inserted into the current drawing. This operation can be done in three different ways:

1. scale to fit a rectangle
2. scale at a specified center point
3. overlay (no translation or scaling)

In option **1**, the user enters a rectangular area, and the inserted drawing will be scaled to fit completely within the area. The scaling is done so that the drawing domain of the inserted drawing is centered within the rectangle.

In option **2** the user enters a center point and enters a scale factor. The inserted drawing will be centered with respect to the entered point, and scaled according to the scale factor.

In option **3** the contents of the inserted drawing are simply copied wholesale into the current drawing, without any scaling or translation. If the inserted drawing's limits exceed those of the current drawing, some incoming elements may not be visible or editable, unless the current drawing's limits are expanded to include them.

Before the user selects an option, the drawing limits of the current drawing and the drawing to be inserted will be shown on the screen.

Components added to the current drawing may be defined or undefined. A component is considered to be defined if its name is on the list of loaded components in the current drawing. In this case, the added component will use the definition in the current drawing. If a component is undefined, its definition will automatically be copied from the inserted drawing file to the current drawing file.

31.6 Insert tiles

Command: `TILE nrow ncol`

This function offers an easy way to insert other drawings into the current drawing space in a regular and consistent fashion. The command-mode arguments `nrow` and `ncol` specify the number of rows and the number of columns into which the drawing space will be divided. This subdivision defines a set of equal-sized rectangular regions, called “tiles”. After the subdivision is specified, a grid is displayed to show the tile boundaries.

In X/Motif, a dialog pops up that allows the user to enter the number of tile rows and columns and use a file selector to pick the drawings to be inserted.

In command mode, after the subdivision is shown, the user is prompted to enter a series of drawing names for the inserted drawings. The format is:

```
>drawing_name   [row   column]
```

which causes the specified drawing to be inserted in the tile addressed by (`row,column`). The tile address is optional: if it is omitted, the tile address will be incremented from its previous value. By default, tiling will start at row 1, column 1, unless otherwise specified in the first record of the list file. Before tiling starts, the user interactively chooses whether tiles will be filled by rows or by columns, in the absence of specific addressing. Multiframe drawings will have all frames inserted into a sequential set of tiles starting at the current address.

Entering `*` or a drawing name containing `*` will generate a list of drawings matching the specification.

To end the input of drawing names, enter CTRL-Z on VMS and CTRL-D on UNIX.

After exit from the Insert tiles function, the tiling scheme (number or rows and columns) remains defined, so that the Zoom function and the area-editing functions can operate on tiles.

Multiple tiling procedures can be done. Each time, the desired subdivision must be entered. By entering different subdivisions, one can get complex layouts.

Drawings can be displayed on a tile-by-tile basis, using the Zoom function. Refer to Section II for details. Area-editing functions such as Area cut (`ACUT`), Area delete (`ADEL`), Area modify lines (`AMOD`), Duplicate (`DUP`), Extract (`EXT`), Insert (`INS`) and Transform (`TRAN`) can also use tiles as their input: when asked to enter the input rectangle, the user clicks on the Select tile toggle and clicks on the relevant tile in the drawing (in command mode, use the "T" key to select the tile). For the Transform and Duplicate functions, the destination rectangle can also be a tile, so that tiles can be moved and copied.

During the tiling process, components added to the current drawing may be defined or undefined. A component is considered to be defined if its name is on the list of loaded components in the current drawing. In this case, the added component will use the definition in the current drawing. If a component is undefined, its definition will automatically be copied from the inserted drawing file to the current drawing file.

32 Multiframe

The following functions are for accessing, decomposing and building multiframe drawing files. See Section I for details on the generation and use of these files.

When a multiframe drawing file is first called up in EDGR, the first frame will be displayed, and becomes the *current frame*. The number of the current frame and the total number of frames in the file are shown whenever a frame is accessed or displayed.

X/Motif: when a multiframe drawing is called up, a frame selection dialog will pop up and remain present until the drawing is closed. This dialog provides the same functionality as the `FRAME`, `FN`, and `FP` commands discussed below.

32.1 Go to frame

Command: `FRAME frame_number`

This function selects the specified frame for display.

32.2 Next frame

Command: `FN`

This function selects the next frame in the drawing file for display. No action is taken if we are already at the last frame in the file.

32.3 Previous frame

Command: `FP`

This function selects the previous frame in the drawing file for display. No action is taken if we are already at the first frame in the file.

32.4 Multiframe Make

Command: `FMAKE drawing_name list_file`

This function takes any collection of drawings and concatenates them into a new multiframe drawing with the specified name. The names of the source drawings to be concatenated are obtained from the list file, which contains one drawing name per line. Source drawings can be single or multiframe. The list file can be generated using EDGR's `LIST` command or the appropriate external system tools.

Components added to the multiframe drawing may be defined or undefined. A component is considered to be defined if its name is on the list of loaded components in the multiframe drawing. In this case, the added component will use the definition already loaded. If components are undefined, their definitions will automatically be copied from the source drawing files to the multiframe drawing file.

32.5 Multiframe Split

Command: `FSPLIT [split_name]`

This function takes the current drawing, which must be multiframe, and splits it into a series of single-frame drawings, one for each frame. This allows frames to be edited at the element level. The single-frame drawings will have the name specified by `split_name` with an additional three characters indicating the frame number: 001, 002, etc. If `split_name` is not specified, the current drawing name is used. The resulting set of single frame drawings, or any subset or superset, can later be recombined into a new multiframe drawing using the Multiframe Make (`FMAKE`) function.

33 Import/Export

The following functions deal with the transfer of graphics information from other packages into EDGR and the transfer of EDGR drawings over networks.

33.1 Convert plot file

Command: `CONV format filename`

`format:`

`HPGL`	for Hewlett Packard HPGL and compatible plotters
`DMPL`	for Houston Instruments DM/PL plotters
`TEK`	for Tektronix 4010/4014 and compatible devices

This function invokes a facility for converting plot files for the above devices to EDGR drawings. Many graphics packages and programs can output their images in one or more of these formats. Using CONV, these images can be brought into EDGR for editing, hardcopy production, or inclusion in other drawings.

After the specified plot file is opened, the user will be asked to enter the name of a new drawing to receive the converted graphical data.

In X/Motif, a dialog will be brought up with option toggles for setting the plot file format and with file selectors for choosing the plot file and the new drawing file.

If you have a choice of source formats, the **HPGL** format is the recommended one. The HPGL converter is more highly evolved than the others and is more tolerant of different logical record structures (or lack thereof) in the input file. Most of the HPGL command repertoire is supported.

33.1.1 Transferring files to VMS

In non-VMS environments, plot files are often produced as just a stream of characters without any logical record structure. If you want to transfer such plot files to VMS for reading by EDGR, you should use a **binary** mode of file transfer, resulting in a VMS file with fixed-length 512 byte records. For example, in ftp issue the `binary` command and in Kermit use the `set file type binary` command. If you are using VMS PCDISK, copy the file to VMS using `export/format=fixed`.

33.1.2 Plotter files

For the HPGL and DM/PL plotter files, after the new drawing is generated, the drawing format and limits will be set to appropriate values to suit the data, giving minimum 2.5% margins.

Pen changes in the plot file will result in layer changes in the EDGR drawing. Pen 1 is mapped to Layer 0, Pen 2 to Layer 1, and so on.

33.1.3 Tektronix files

For Tektronix plot files, the coordinate system used for the drawing is 1024×780, corresponding to the Tektronix 4010 address space. The 4010 protocol supports only integer addresses in this range, so the lack of precision will be evident on zooming in. The 4014 protocol supports an additional 2 bits of precision, so coordinates will be to the nearest 0.25 in the 1024×780 address space. For applications where precision and smooth rendering are important, the 4014 protocol should be used if it is available.

Both the 4010 and 4014 terminals support "hardware text," and this is used by some applications, which send ASCII strings to be plotted as text, rather than sending text as vectors. EDGR's Tek converter generally tries to filter out all non-graphics "garbage" in the plot file, including ASCII text, which is often conversational and not part of the graphics display. However, there is an option to try to interpret properly-encoded ASCII text as graphics text to be plotted. This option has been tested for simple cases, but it is not necessarily foolproof.

If the plot file has multiple images, it will be converted to a multiframe drawing file with one image per frame. The images must be properly separated by the Tektronix "reset" sequence ESC FF.

33.1.4 UGS plot files

The `CONV TEK` command also includes a special handler for image files from the SLAC Unified Graphics System (UGS). These files are essentially Tektronix plot files that may contain multiple images and header information. If there are multiple images, the resulting drawing will be multiframe. The frame number and name of each image will be displayed as it is processed.

33.2 Write net file

Command: `NETW drawing_name [net_file_name]`

This function encodes the specified drawing as an ASCII file that can be shipped over networks and then decoded at the destination using the Read net file (`NETR`) function. The file can be sent by a file transfer facility or as part of a mail message.

If the net file name is not given, it will be the same as the drawing name. The default extension for the net file is `.net`.

Since VMS and UNIX systems have different floating-point binary representations, the Write net file and Read net file functions are necessary to move drawings from one system to the other.

In X/Motif, a file selector is brought up to allow the choice of the drawing file to be encoded. The net file name will be derived from the drawing name.

33.3 Read net file

Command: `NETR net_file_name [drawing_name]`

This function reads the specified net file (created with the Write net file function) and decodes it into a new drawing with the specified name. The new drawing files will be identical with the originals that were used to make the net file. If the net file is embedded in a mail message, any leading material will be skipped until the header "BEGIN Edgr Net File" is found. Material after the "END Edgr Net File" trailer will also be ignored.

If the drawing name is not given, it will be the same as the net file name. The default extension for the net file is `.net`.

In X/Motif, a file selector is brought up to allow the selection of the net file to be read. The drawing name will be derived from the net file name.

34 Hardcopy

The following functions allow hardcopy files to be generated from the current drawing or a selected drawing.

34.1 Hardcopy Entire drawing

Command: `HARD [drawing_name]`

This function is used to make a hardcopy of the current drawing or any other existing drawing. In X/Motif, if there is a current drawing then a hardcopy will be made of it. If there is no current drawing then a drawing file selector will be brought up. The selected drawing will not be displayed and will be closed after hardcopy processing. In command mode, if the `drawing_name` parameter is omitted, a hardcopy of the current drawing is made. If a drawing name is specified, the current drawing (if any) is closed and the specified drawing is opened but is not displayed and will be closed after hardcopy processing.

The hardcopy produced is for the full drawing, as defined by the drawing limits. To make a hardcopy of a zoomed portion of a drawing, the Hardcopy Current zoom (`SNAP`) function must be used.

Supported hardcopy devices are:

- Tektronix plot file
- Printronix printer/plotter
- HP plotter (HPGL)
- HP Laserjet printer (PCL)
- HP Thinkjet printer (PCL)
- DEC LA100 printer
- Houston Instruments plotters (DM/PL)
- DEC LN03+ laser printer
- Imagen laser printer (imPRESS)
- HP Paintjet colour printer
- PostScript printers
- GKS Metafile (where GKS is available)
- Roland GL plotters

The size and orientation of the image of the drawing on the hardcopy page will depend on the drawing format: landscape, portrait, or square.

For landscape and portrait formats, the image will usually occupy most of the output page, with appropriate margins and orientation. For large-format 11×14 printers (Printronix and LA100) a smaller portion of the output page is used. For "bit-mapped" devices with variable resolution, such as the HP Laserjet and Thinkjet, a choice of hardcopy sizes and orientations may be offered, from which a selection is made prior to generating the hardcopy.

For square format, the image will be a square, usually centered on the output page and made as large as possible. For viewing, the output page will be in portrait orientation, except on the large-format printers where landscape orientation is used.

In X/Motif mode, a Hardcopy dialog is brought up which allows the device and output format, and other options, to be chosen. The hardcopy data can be directed to a file for later printing, or can be printed directly on a specified printer queue.

In command mode, the destination and disposition of the hardcopy file is determined by typing a **hardcopy command**. For the selected hardcopy device, a menu of possible commands is displayed. Example:

```
Device: HP Laserjet
Default queue: HP$LASER
HARDCOPY COMMANDS
 P  [que-name]     Print
 PC [que-name]     Print compressed
 S  [file-name]    Save
 SC [file-name]    Save compressed
 T  [file-name]    TeX output
 TJ [file-name]    TeX justified output
 TC  [file-name]   TeX output compressed
 TJC [file-name]   TeX justified output compressed
 Q                 Quit

Enter hardcopy command >
```

- The **Print** commands produce a plot file and queue it for printing on the specified queue. For the above device the default queue is determined by the logical name `HP$LASER` (or environment variable `HP_LASER` on UNIX systems). The plot file will be deleted after printing.

- The **Save** commands write out the plot file to disk, under the specified file name. If the file name is omitted, the file will be named according to the device type, eg. `HPLASER.PLT`.

- For the HP Laserjet models IIP and later, the **compressed** mode of output can be used to reduce plot file size and printing time.

- The **TeX output** commands perform the same function as **Save** but the plot file is produced in a suitable format for inclusion as a figure in a TeX document, rather than for direct printing.

- The **justified** option for TeX output removes all white space above and to the left of the image, so that the significant part of the image will start at the current writing location on the document page. This makes it easier to set figure positioning parameters.

- The **Quit** command exits from the hardcopy function without producing any plot file, and returns to EDGR command level.

The list of available commands varies from device to device, but all devices have basic **Print** and **Save** functions, with the exception of the generic Tektronix and GKS files, which have only **Save**.

34.1.1 Colour assignments

On colour hardcopy devices, each drawing layer can be assigned to a particular colour or plotter pen number on the device. Prior to output of the plot file, the list of these assignments can be displayed on the screen and altered as desired. The default assignments are arranged, as far as possible, to give the same colour on the display and the hardcopy.

34.1.2 Pen plotters

For pen plotters, the default assignment of layers is as follows: layer 0 is assigned to to pen 1, layer 1 to pen 2, and so on until the last pen has been assigned, after which additional layers are assigned to pen 1. An option is also provided for setting the pen velocity for the plot.

34.1.3 Line widths

For bit-mapped devices such as the HP Laserjet and HP Paintjet, and for PostScript devices, the user can specify the *line width* for the hardcopy. The line width specifies the thickness in pixels of the lines in the hardcopy image. Note that the actual size of a pixel may depend on the density mode (number of pixels per inch) being used on the printer.

The line width parameter must be an integer. Any fractional values will be rounded to the nearest whole number. On bit-mapped devices, for a given line width n, lines are rendered by painting into the bitmap with imaginary "brush" which is a **square** of $n \times n$ pixels. For line widths greater than 3 pixels, a **round** brush can also be used. This gives a more consistent width for lines at different angles, and in most cases improves the appearance of the image. To specify a round brush, enter $-n$ instead of n for the line width.

A single line width can be assigned to the entire drawing or, optionally, a different line width can be assigned to each layer in the drawing, in a similar way to the assignment of colours and plotter pens.

34.1.4 Postscript

The PostScript output conforms to PS-Adobe-2.0 EPSF-2.0 structuring conventions. The output file is produced in "encapsulated" format and is suitable both for direct printing and for inclusion in other PostScript documents. An option for colour output is provided. Note that some monochrome printers will fail to print files with colour commands, while others will map the colours into various shades of gray.

34.1.5 Tektronix plot files

This hardcopy option produces a plot file that contains a record of the drawing display in **Tektronix 4010** or **4014** format. This file can be used in various ways:

- It can be copied to a Tek-compatible terminal, e.g. using `TYPE` or `cat`, to display the drawing image.
- It can be read by a "slide show" program or script which copies it to the terminal.
- It can be sent to Tek-compatible hardcopy devices to produce a hardcopy of the drawing.

34.1.6 GKS metafiles

GKS metafile output is available only at sites where a local GKS library is available, and EDGR has been linked with this library. This type of metafile may be useful for obtaining hardcopies on devices not directly supported by EDGR, provided a local metafile replayer is available.

Note: the module `GKSDUM` must be removed from the graphics library `LIB.OLB` or `gplot.a` before linking EDGR with a GKS library.

34.2 Hardcopy Current zoom

Command: `SNAP`

This function makes a hardcopy of the current zoom area instead of the entire drawing. A single page in landscape mode, showing whatever is seen on the screen (for terminals) or the zoom window (for workstations), is produced on the selected output device. This feature can be used with the `Zoom` and `Pan` functions to print details of drawings. See the preceding section for details on hardcopy devices and output control.

35 Help and Utility

The following functions display information about the EDGR system and its current status and provide some general-purpose utility functions.

35.1 Info

Command: `INFO`

This function just provides brief "bootstrap" information for anyone running EDGR for the first time.

35.2 News

Command: `NEWS`

This gives immediate access to HELP topic NEWS, for users who wish to keep up-to-date with changes and new features in EDGR. Information under NEWS can be considered as an addendum to this User's Guide.

35.3 General help

Syntax: `HELP [topic]`

This gives the user access to complete on-line documentation for EDGR.

X/Motif: the help facility is implemented through a dialog that contains a scrolling text region for displaying the help text. Topics are selected by double-clicking with the mouse on the topic word. Buttons are provided for listing the topics at the current level, and going up one level to list the previous topics. The Help menu item General help is used to open the help dialog window and display the topic list. There are additional menu items to provide "instant help" on the following topics: Info, News, Mouse functions, and Action keys. The first two topics are equivalent to the command-mode `INFO` and `NEWS` commands.

Command mode: On VMS the standard Help Library facility is used. On UNIX systems the same help text is accessed through an emulation routine that behaves very similarly to the VMS Help interface. As an alternative to browsing through the Help library, including the parameter `topic` will bring up the help text for the topic and return immediately to EDGR command mode. For example, `HELP LAYER` will display documentation for the `LAYER` command.

At present, regardless of the which interface is being used, the help text for the various functions is accessed by the terminal-mode command names: for example, help on the Enter Lines menu item is stored under topic `LINES`. The correspondence between menu items and terminal commands is usually clear, e.g. `TEXT` = Enter text, `MODT` = Modify text, etc. A few non-obvious cases are listed here:

`CL`	Clear graphics
`DISP`	Refresh display [button]
`WHAT`	Show status
`SNAP`	Hardcopy Current zoom
`FRAME`	Go to frame [button in multiframe dialog]
`FN,FP`	[arrow buttons in multiframe dialog]
`FMAKE`	Multiframe Make
`FSPLIT`	Multiframe Split
`STROKE`	Convert to lines
`VER`	Confirm edits [toggle button]
`PLACE`	Enter Components
`LOAD`	Component Load
`CMOD`	Move layers
`LMOD`	Modify line types

35.4 Command summary [command mode only]

Command: `COMM`

This command displays on the screen a summary of all the EDGR commands. The commands are grouped according to their function. The display has been made as concise as possible, so it can be used to quickly identify the command name for a particular editing function.

35.5 Directory list [command mode only]

Command: `DIR [filespec]`

This command emulates the DCL `DIRECTORY` command or the Unix `ls` command. It produces a list of all files matching the filespec, which may contain disk and directory names and wild cards. In the absence of the filespec, a listing of the current working directory is given. On UNIX systems the `DIR` command simply invokes `ls` to generate the list of files.

35.6 Text editor

Command: `EDT filename`

On VMS this command invokes the EDT editor to edit the specified file. If you wish to use an EDT startup file the logical name EDTINI should be defined to point to this file. On UNIX the emacs editor is invoked.

35.7 DCL mode [VMS only]

Command: `DCL`

In command mode, this allows access to the operating system by attaching the terminal to a subprocess, where any desired programs or DCL commands can be executed. The command RESUME must be typed to exit the subprocess and resume the EDGR session. The first time the `DCL` command is issued there may be a noticeable delay due to the spawning of the subprocess, but subsequent transfers to the subprocess will be immediate.

35.8 Shell command [UNIX only]

Command: `%command`

In command mode, any shell command can be executed from within EDGR by prefixing it with a `%` character. The environment variable `SHELL` determines which shell is used, the default being `sh`. If you are using `csh` then access to the shell can also be gotten in the usual way by typing CTRL-Z to interrupt EDGR and resuming later with `fg`. To avoid difficulties in resuming EDGR operations after an interrupt, it is advisable, when in graphics input mode, to exit from this mode by pressing the `Q` key before typing CTRL-Z.

35.9 Calculator [VMS only]

Command: `CALC`

In command mode, this invokes a calculator facility which reads Fortran-like expressions and evaluates them numerically. Example:

```
EDGR > CALC
? DX=0.4
? DX*SIN(40*3.14159/180)
DX*SIN(40*3.14159/180) = 0.25711486318410744
?
EDGR >
```

Symbolic constants can be used and will be remembered throughout the session. To exit from the calculator and return to EDGR command mode, press `RETURN`. The calculator can also be invoked in graphics input mode by pressing the `#` key.

A Installing EDGR on VMS Systems

The files for EDGR are typically furnished in a backup save set or as part of a standard distribution VAX backup tape that also contains the Graphics Library and the PHYSICA graphics/analysis program. Included with the distribution is an installation guide for PLOTDATA and other TRIUMF programs. The following describes only the installation of the necessary components for EDGR.

Note: If you get the backup save set `EDGR.BCK` via ftp, you may have to change the record length before unpacking it. In general, this applies if you do not have Multinet ftp. In any case, the backup save set must be transferred in binary mode. To change the record length, get the file `FIXBCK.COM` via ascii ftp, and type the following:

```
$ @FIXBCK EDGR.BCK
```

A.1 Files for EDGR

The following files should be installed in a single directory on disk:

`EDGR.EXE`	graphics editor program
`EDGR.HLB`	help library
`EDGR.HLP`	help library text
`EDGR.HLI`	help library index
`EDGR.DAT`	app-defaults file
`EDGR.OLB`	object module library
`EDGR.LNK`	procedure to re-link EDGR (modify appropriately)

If desired, a text file `EDGR.USE` can be created in the same directory. If this file is present, it will be used to log each usage of EDGR. This file is optional and is not required to run EDGR. The file must have “world write” access to allow logging of all users.

The following steps are necessary to install EDGR:

1. Define the logical name `EDGR$DIR` to point to the directory containing the above files.
2. Define the foreign command to invoke the graphics editor:

   ```
   $ EDGR:==$EDGR$DIR:EDGR
   ```

3. Install the app-defaults file:

   ```
   $ COPY EDGR.DAT DECW$SYSTEM_DEFAULTS
   ```

A.2 Files for library support

The following additional files should be installed on disk:

`GPLOT.OLB`	graphics and analysis library
`VAXFONT.DAT`	font file

The following definitions should be made, preferably at the system level, for access to these files:

1. Define the logical name `GPLOT$DIR` to point to the directory containing the library `GPLOT.OLB`.
2. Define the logical name `TRIUMF$FONTS` to point to the directory containing the file `VAXFONT.DAT`.

A.3 GKS metafile support

If there is a local GKS library, a version of EDGR can be built to produce GKS metafile output as a hardcopy option. This is not included in the standard version of EDGR since not all sites have GKS facilities.

To build the new version of EDGR, do the following:

1. Remove the module `GKSDUM` from the graphics library `GPLOT.OLB`.
2. Add the GKS library at the end of the linking procedure `EDGR.LNK`.
3. Execute the link procedure.

B Installing EDGR on UNIX Systems

The files for EDGR will typically be furnished in a compressed tar file or on a standard distribution tar tape that also contains the Graphics Library and the PHYSICA graphics/analysis program. The following describes only the installation of the necessary components for EDGR.

B.1 List of required files

After extracting the contents of the tar tape to disk, the following files should be present:

`edgr`	graphics editor program
`edgr.hlp`	help library
`edgr.hli`	help library index
`Edgr`	app-defaults file
`edgr.a`	object module archive
`edgr.lnk`	shell script to re-link EDGR
`gplot.a`	Triumf graphics library
`vaxfont.dat`	font file

For ease of maintenance, these files can be kept in a common directory, e.g. `/usr/users/triumf`. One way to install EDGR is:

```
% setenv EDGR_DIR /usr/users/triumf
% setenv TRIUMF_FONTS /usr/users/triumf
% alias edgr /usr/users/triumf/edgr
% cp Edgr /usr/lib/X11/app-defaults
```

EDGR expects to find the files `edgr.hlp` and `edgr.hli` in the directory `EDGR_DIR`, and the file `vaxfont.dat`, in the directory `TRIUMF_FONTS`. If these environment variables are not defined, EDGR will look in `/usr/local/bin` for the files. Thus an alternative way to install EDGR is to put these files in `/usr/local/bin` or to define logical links such as:

```
% cd /usr/local/bin
% ln -s /usr/users/triumf/edgr
% ln -s /usr/users/triumf/edgr.hlp
% ln -s /usr/users/triumf/edgr.hli
% ln -s /usr/users/triumf/vaxfont.dat
% cd /usr/lib/X11/app-defaults
% ln -s /usr/users/triumf/Edgr
```

To complement this type of installation, the graphics library can also be installed in the standard place:

```
% cd /usr/local/lib
% ln -s /usr/users/triumf/gplot.a libgplot.a
```

allowing the loader flag `-lgplot` to be used for linking programs.

B.2 GKS metafile support

If there is a local GKS library, a version of EDGR can be built to produce GKS metafile output as a hardcopy option. This is not included in the standard version of EDGR since not all sites have GKS facilities.

To build the new version of EDGR, do the following:

1. Remove the module `gksdum.o` from the graphics library archive `gplot.a`.
2. Add the GKS library to the shell script `edgr.lnk`.
3. Execute the shell script.

Command Index

%command Execute shell command [UNIX], **83**
@filename Execute script file, **41**
ACL Clear alpha, **50**
ACUT Area cut lines, **44**
ADEL Area delete, **44**
AMOD Area modify lines, **46**
AMODT Area modify text, **47**
ARC Enter arcs, **57**
ARROW Enter arrows, **63**
BORDER Border display on/off, **55**
BOX Enter boxes, **64**
CALC Calculator [VMS], **83**
CL Clear graphics, **50**
CLOSE Close drawing, **39**
CMAKE Make component file, **66**
CMOD Move layers, **68**
COMM Command summary, **82**
COMP Compress current drawing, **40**
CONV Convert plot file to drawing, **74**
CROSS Activate terminal crosshairs, **56**
DCL Enter DCL mode [VMS], **83**
DEL Delete lines, **43**
DELC Delete components, **44**
DELT Delete text, **44**
DIR Directory list, **82**
DISP Refresh display, **50**
DUP Region duplicate, **71**
EDT Text editor, **82**
EXT Region extract, **71**
FAST Fast text on/off, **55**
FILL Area fill, **63**
FMAKE Make multiframe drawing, **74**
FN Select next frame, **73**
FORM Set drawing format and limits, **68**
FP Select previous frame, **74**
FRAME Go to frame, **73**
FSPLIT Split multiframe drawing, **74**
FULL Display entire drawing, **52**
GRID Grid set-up, **54**
HARD Make hardcopy, **77**
HELP Use help facility, **81**
INS Region insert, **71**
LAYER Layer control, **53**
LINE Enter lines, **56**
LIST List drawings, **40**
LMOD Modify line types, **68**
LOAD Load component, **66**
MOD Modify lines, **45**
MODC Modify components, **47**
MODT Modify text, **46**
MOVE Move region, **69**
NETW Write net file, **76**
NETR Read net file, **76**
NEW Edit new drawing, **38**
NEWS Show recent changes, **81**
OLD Edit existing drawing, **39**
VIEW Open drawing for read only, **39**
PAN Pan display, **52**
PLACE Enter components, **61**
SAVE Save current drawing, **40**
SNAP Hardcopy current zoom, **80**
STAMP Identification stamp, **65**
STOP Exit program, **42**
STROKE Convert to lines, **49**
SURVEY Make survey file, **41**
TEXT Enter text, **59**
TILE Insert tiles, **72**
TOL Set edit tolerance, **67**

TRAN Region transform, **69**
VER Set edit verification, **69**
WHAT Show drawing status, **67**
ZOOM Set zoom window, **50**

EDGR Command Summary

File

NEW	Edit new drawing
OLD	Edit existing drawing
VIEW	Open drawing for read only
CLOSE	Close current drawing
COMP	Compress current drawing
SAVE	Save current drawing
LIST	List drawings
@file	Execute script file
SURVEY	Execute survey file
STOP	Exit from graphics editor

View

CL	Clear graphics
ACL	Clear alpha
DISP	Refresh display
ZOOM	Set zoom window
PAN	Pan display
FULL	Display entire drawing
LAYER	Layer control
GRID	Set grid parameters
FAST	Fast text on/off
BORDER	Border display on/off
CROSS	Activate crosshairs

Enter

LINE	Enter lines
ARC	Enter arcs
TEXT	Enter text
PLACE	Place components
FILL	Area fill
ARROW	Enter arrows
BOX	Enter boxes
STAMP	Identification stamp

Edit

DEL	Delete lines
DELT	Delete text
DELC	Delete components
ADEL	Area delete
ACUT	Area cut lines
MOD	Modify lines
MODT	Modify text
MODC	Modify components
AMOD	Area modify lines
AMODT	Area modify text
SMOOTH	Smooth polyline
STROKE	Convert to lines

Drawing

WHAT	Show editing status
TOL	Set edit tolerance
FORM	Set drawing format and limits
CMOD	Move layers
LMOD	Modify line types
VER	Set edit verification

Region

MOVE	Move region
TRAN	Transform region
DUP	Duplicate region
EXT	Extract region from drawing
INS	Insert drawing in region
TILE	Define and insert tiles

Multiframe

FRAME	Select frame
FN	Select next frame
FP	Select previous frame
FMAKE	Make new multiframe drawing
FSPLIT	Split multiframe drawing

Component

CMAKE	Make component file
LOAD	Load component def
CMAINT	Component maintenance

Import-Export

CONV	Convert plotfile to drawing
NETW	Write net file
NETR	Read net file

Hardcopy

HARD	Make hardcopy of drawing
SNAP	Make snapshot of display

Help and Utility

COMM	List commands
HELP	Enter help facility
NEWS	Show recent changes
DIR	Directory list
EDT	Text editor
DCL	Enter DCL mode [VMS]
%command	Shell command [UNIX]
CALC	Calculator [VMS]