5. The Command Reference Guide ... 2
5.1 Terminology for the Schematic and Symbol Editors ... 2
5.1.1 Screen Areas ... 2
5.1.2 Design Fundamentals ... 3
5.1.3 Other Terms ... 3
5.2 Specifying Names When Using Schedit ... 5
5.3 Placing Points in the Design Area ... 5
5.4 Selecting CapFast Commands ... 5
5.5 The CapFast Commands in Menus ... 6
5.5.1 Editing Symbol and Schematic Files − The File Menu ... 7
5.5.2 Creating Hierarchical Designs − The Hierarchy Menu ... 8
5.5.3 Manipulating Elements − The Edit Menu ... 9
5.5.4 Commands for Viewing − The View Menu ... 11
5.5.5 Drawing Symbols − The Draw Menu (Symed Only) ... 12
5.5.6 Bringing Parts into a Schematic − The Part Menu (Schedit Only) ... 13
5.5.6.1 PCB Parts Library ... 14
5.5.6.2 SUSIE Parts Library ... 16
5.5.6.3 Hilo Primitive Library ... 16
5.5.6.4 Spice Parts Library ... 16
5.5.6.5 Actel Parts Library ... 17
5.5.6.6 Xilinx Parts Library ... 17
5.5.6.7 Flowchart Library ... 18
5.5.7 Documentation Aids − The Text Menu ... 18
5.5.8 Adding Ports to a Symbol − The Ports Menu (Symed Only) ... 20
5.5.9 Manipulating Wires and Buses − The Wire Menu (Schedit Only) ... 20
5.5.10 Controlling Symbol Characteristics − The Properties Menu ... 22
5.5.10.1 Assigning Symbol Characteristics − The Properties Window ... 23
5.5.11 Selecting and Finding Elements − The Select Menu ... 25
5.5.12 Creating Keyboard Shortcuts − The Macro Menu ... 26
5.5.13 Controlling the Design Area − The Configure Menu ... 27
5.5.14 Controlling Drawing Alignment − The Drawing Constraint Menu ... 28
5.5.15 Controlling the Editing Environment − The Options Menu ... 29
5.5.15.1 Saving Your Option Changes ... 29
5.5.15.2 The Options Menu ... 29
5.5.15.3 The Default Naming Scheme Sub−menu ... 31
5.5.15.4 The Name Prompting Sub−menu (Schedit Only) ... 33
5.5.15.5 The Object Visibility Sub−menu ... 33
5.5.15.6 The Search Path Sub−menu ... 34
5.5.15.7 The Text Options Sub−menu ... 35
5.5.15.8 The View Options Sub−menu ... 36
5.5.15.9 The Visible Grid Linestyle Sub−menu ... 37
5.5.15.10 The Wire Options Sub−menu (Schedit Only) ... 37
5.5.16 The About Menu ... 38
5.6 The CapFast Internal Commands ... 39

Chapter 5 The Command Reference Guide

This chapter contains an assortment of information about *Schedit* and *Symed*. While this chapter is organized by menu, you do not need to read it in any special order. You should consider this to be *Schedit* and *Symed* reference material rather than a thoroughly−structured presentation.

5.1 Terminology for the Schematic and Symbol Editors

These words have specific meanings with regard to the display screen and the design editors.

5.1.1 Screen Areas

Command and Response Area	The area at the bottom of the screen, where the editor displays prompts and feedback, and user input is echoed.
Design Area	The screen area where you view and alter the schematic or schematic symbols.
Mouse Button Area	The area at the top of the screen, describing the current functions of the mouse buttons. In the left corner of this area are the x,y coordinates of the last point placed; in the right corner is a continuous report of the current x,y cursor position. These two x,y locations are given in design units.
Pop−Up Menu	A menu that "pops up" when selected. A pop−up menu disappears when you type `<Esc>`, move the cursor outside the menu and press any mouse button, or invoke a command from the menu.
Menu Bar	The screen area where the main menu titles are displayed.
Status Area	The screen area providing information about the editing environment, including the name and release of the program, the file being edited, the alignment and adjustment factors and the distance between grid points.

5.1.2 Design Fundamentals

Design Unit	The minimum possible distance defined between points in the design area. Lengths and distances in the design area are expressed in terms of design units. One design unit corresponds to 1/160th of an inch.
Element	In *Schedit*, a wire, bus, bundle, annotation box, text string, or symbol. In *Symed*, a text string, annotation box, port, or basic graphic from which symbols are created.
Properties	Text strings that define the electrical, physical, and logical characteristics of the symbols, wires, ports, and connectors in a schematic.
Schematic Connector	A connector in a schematic. There are five types of schematic connectors: external physical, offpage, onpage, hierarchical, and global.
Symbol	A graphic representation of a functional circuit element. Symbols are created with *Symed*, and may represent a physical part, such as a diode, or an underlying schematic in a hierarchical design.
Primitive Symbol	A symbol with a `primitive` property. A primitive symbol is the fundamental unit to which electrical properties can be assigned for logic design and simulation. Primitive symbols are created using *Symed*. Examples include an AND gate, a transistor, and a flip−flop.
Symbol Port	The part of a symbol to which a wire is attached. For a symbol that represents an underlying schematic, the symbol ports have a one−to−one correspondence to the hierarchical connectors in the underlying schematic and are electrically connected to them.
Text	Refers to the instance names assigned to objects on the schematic and to names and labels not associated with any particular design element. Text commands and options work with either type of text.

5.1.3 Other Terms

Current Set	The current working set, consisting of elements that have been chosen for manipulation. The current set is displayed in a color different from that of the other items in the schematic. This color is usually red unless you have changed it. You can define any collection of things in the design area to be the current set. Once the current set is defined, you can use

editor commands to move it around, change its size (text only), change its orientation, turn it over, make copies of it, etc.

If a current set is not defined before invoking a command that requires you to, you are prompted to select items on which you want to operate. Press the Done mouse button when all items are selected, and they become the current set and are operated on.

Some ways of defining the current set "undefine" the previous current set. Other ways let you add to, or subtract from, the previous current set. You use the Select menu to define the current set.

Drag Mode
: An interactive editing mode used to move objects. You enter the drag mode by first selecting some objects and then entering the `Edit®Move` command. Pressing the Point button on the mouse fixes the object in place, exiting you from Drag Mode.

Flashlight Mode
: An interactive editing mode used to select objects. Press **`S I`** (`Select®Items`) on the keyboard to enter flashlight mode. In this mode, as you move the cursor over an object, the object becomes highlighted. You can then press a mouse button to include or remove it from the current set. You also enter this mode when you do not place points or do not have a selected set prior to invoking some commands.

Point
: A location on the screen used to designate a reference or positioning point for a line, wire, box, symbol, or other object. A point is displayed as a small cross with a sequence number.

Rubber Band Mode
: An interactive editing mode used to create lines and wires. In *Schedit*, type **`W W`** (`Wire®Draw Wire`) to enter this mode. In *Symed*, type **`D V`** (`Draw®Vector`) to enter this mode. When the rubber band mode is in effect, pressing the `POINT` button on the mouse establishes a reference point in the design area. The editor displays a wire/line on the screen between the reference point and the cursor. As you move the cursor, the wire/line between the reference point and the cursor stretches as if it were a rubber band. When the wire/line is in the desired position, you press the `POINT` button again to establish the endpoint of the wire/line.

Rubber Box Mode
: An interactive editing mode used to define rectangles. You enter this mode in *Schedit* with a command like **`V A`** (`View®Zoom Area`), with no points in the design area. This mode can also be used with the **`D R`** (`Draw®Rectangle`) command in *Symed*. When the rubber box mode is in effect, pressing the `POINT` button on the mouse establishes one corner of the box in the design area. The editor then displays a rectangle. One corner of the rectangle is at the initial reference point, and the other corner is controlled by the cursor. As you move the cursor, the

rectangle grows and shrinks. Pressing the `DRAG/STRETCH` button on the mouse will allow you to move the entire rectangle; pressing the `DRAG/STRETCH` button again returns you to the stretch mode. When the rectangle is the desired size and shape, you press the `POINT` button again to fix the rectangle in place.

5.2 Specifying Names When Using *Schedit*

As you use *Schedit* or *Symed*, you may assign names to objects, such as wires, ports, and symbol instances. Object names may contain characters (numbers or letters) in either uppercase or lowercase (case is significant), but should not contain spaces, tabs, or other control characters. In addition, these characters have reserved meanings and should not be used:

```
@  #  /  {  }  (  )  ,  !  .  <  >  ;
```

The following characters may only be used in conjunction with buses or bus components:

```
[  ]
```

5.3 Placing Points in the Design Area

One of the uses of the mouse is to place points. Points are placed in the design area to specify locations for elements to be drawn, instantiated, or manipulated in subsequent commands. In the design area, the default functions of the mouse buttons are:

```
( POINT | COMMAND | UNPOINT )
```

To place a point, move the cursor to the desired position and press the `POINT` button. To remove the most−recently−placed point, press the `UNPOINT` button. Pressing `UNPOINT` again removes the next−most−recently−placed point, and so on.

Whenever you place points in the design area, each is displayed with a number next to it. The numbers indicate the order in which the points were placed, beginning with 0 and continuing upward sequentially.

The x, y location of the last point placed is displayed in the left corner of the mouse button area. In the right corner of the mouse button area, the current x, y cursor position is displayed unless you have customized the editor; see the section on the `Options` menu. Both locations are expressed in design units, and may range from −32000, −32000 to 32000, 32000.

5.4 Selecting CapFast Commands

As described in *Chapter 3: CapFast Tutorial*, there are two ways to enter CapFast commands: through pop−up menus or with the keyboard.

Figure 5−1. The CapFast *Symed* commands

Figure 5−1. File menu in *Schedit*.

To invoke commands from the menu, place the cursor over the desired command to highlight it, and then press the left mouse button. If the command is in a sub−menu, click on the sub−menu's name in the parent menu and the sub−menu will appear.

To invoke a command with the keyboard, first type the letters necessary to display the menu that the command is on, and then type the underlined letter of the command. If the command does not have an underlined letter, type the first letter of the command. If there are more than one command starting with the same letter (with no underlines), then each time you type the letter, the highlight will move to the next command starting with that letter. In the last case, you will have to press <Enter> to invoke the command once it is highlighted.

For example, to exit the editor, type **F E**. To select a Vcc connector, type **P P V** (no <Enter> at the end). To select an output hierarchical connector, type **P C H H <Enter>**.

You can also use a combination of keyboard shortcuts and menus. For example, to select a bidirectional hierarchical connector, type **P C** and left−click on Hierarchical Bidirectional with the mouse.

Figure 5−2. Screen movement with the keypad.

CapFast uses the number keys to let you pan around the screen as you work. When *NumLock* is on, the numeric keypad can also be used. The function assigned to each number key is shown at left. The 5 key displays the entire design in the view area.

5.5 The CapFast Commands in Menus

This section consists of sub−sections which describe the CapFast commands in each menu, illustrating the hierarchy of the CapFast menus. See *Figure 5−3* for a representation of the CapFast *Schedit* menu structure, and *Figure 5−1* for a representation of the CapFast *Symed* menu structure. Each succeeding level of menu is indented an additional tab stop. For example, the main menu contains the menus not indented, beginning with File, Edit, View, etc. The File menu contains the commands indented one tab stop, beginning with New, Open..., Save, etc.

Figure 5−3. The CapFast *Schedit* commands

5.5.1 Editing Symbol and Schematic Files – The `File` Menu

The following commands may be accessed from the `File` menu.

`New` — This command clears the editing area for a new schematic or symbol. If you have made changes to the current file since the last time you saved it, you will be asked if you want to save changes.

`Open...` — Invoking this command lets you stop editing the current schematic or symbol file and begin work on a different file. A dialog box will appear with which you can navigate through your directory structure to find the file you want.

Note that if you select a design file (`.dsn`) to load, the current working directory will be changed to the directory that the design file is in, before loading the first file in the design. If you select a schematic or symbol file, the current directory will not change.

`Save` — Saves the current schematic or symbol. If you haven't named the schematic or symbol yet, you will need to enter a name for the file.

`Save As...` — Allows you to save the current schematic or symbol under a different name. A dialog box is used to select the path where the file is to be stored.

`Merge...` — This command allows you to merge the components of another schematic (*Schedit*) or symbol (*Symed*) with the current drawing. After you select the schematic or symbol to merge, a copy of it will appear in the design area and follow the mouse movements. Place it where you want with the left mouse button.

`Save Selection...` — This will save the currently selected elements to a new schematic or symbol file. A dialog box is used to get the file name and location.

`Hierarchy®` (*Schedit* only) — This brings up the `Hierarchy` menu, from which you can navigate up or down in the design's hierarchy. See the section below about the `Hierarchy` menu.

`QC/ERC` (*Schedit* only) — This command starts the Electrical Rules Checker to check the schematic for these common errors:

1. Floating inputs.
2. Unconnected output, bidirectional, or undefined ports.
3. Outputs and inputs on the same gate connected together.
4. Multiple outputs connected.

5. Outputs connected to GND, Vcc, or any global connector.

6. Wire−to−port width mismatches; that is, connecting a wire or bus to a symbol port, schematic connector, or another wire or bus of a different width. For example, wire `D[0−7]` connected to port `P[0−3]` produces an error.

7. Unused gates.

Schedit will inform you of errors the QC/ERC discovers in two ways: by writing a description of the errors it encountered (if any) into a file (whose name you are prompted for), and by placing properties on the offending symbol or port. The properties will look like this for unused gates:

```
(ERC)warn:unused gate
```

or like this, for ports:

```
(ERC)warn(PORT NAME):ERROR TYPE
```

It also selects the errant symbols, making them the current set.

`Print...` (Windows only)
This brings up the `Print` dialog, allowing you to print the current schematic or symbol. You can also enter `Print Setup` from here.

`Print Setup...` (Windows only)
This brings up the dialog box which allows you to set the options for the printer.

`Execute...`
This command allows you to execute a file containing *Schedit* or *Symed* internal commands. See *Appendix A: Customizing CapFast* for a list of the commands that may be used in a command file, and the section in this chapter on *Internal Commands* for detailed information.

`Exit`
This command terminates an editing session and exits the program. If any changes were made to the schematic or symbol since the last save, you will be asked if you would like to save them before exiting.

5.5.2 Creating Hierarchical Designs − The `Hierarchy` Menu

The commands in the `Hierarchy` menu, a sub−menu of the `File` menu, are used to create the associations between schematic and symbol files, or between various schematic files, in complex designs. These commands help you create the proper association between symbols and schematics while building a schematic hierarchy. As described in *Chapter 4: Creating Electrical Designs*, there are two basic approaches to creating a hierarchical schematic: top−down and bottom−up.

For either approach, each intermediate element of the hierarchy must have both a schematic representation (`.sch`) and a symbolic representation (`.sym`). The commands in the `Hierarchy` menu allow you to develop the two kinds of files, ensuring that corresponding schematic and symbol files have the same base file name, with ports in symbol files that match hierarchical connectors in schematic files.

The following commands may be accessed from the `Hierarchy` menu. See the section entitled *Hierarchical Designs* in *Chapter 4: Creating Electrical Designs*, for more information.

`Descend` (*Schedit* only)	Lets you edit a schematic at a lower hierarchy level than the current schematic without disturbing the editing environment. This command is normally used in top−down design. To use the command, the symbol representing the schematic must be the selected set.
`Ascend` (*Schedit* only)	Used in conjunction with the `Descend` command, this command returns you to the hierarchical level of the symbol whose underlying schematic was being edited.
`Connectors from Symbol...` (*Schedit* only)	Use this command to map symbol ports onto a schematic that represents the underlying circuitry of the symbol. Hierarchical connectors appear in the schematic that have the same names and functions as their corresponding symbol ports.
`New Symbol` (*Schedit* only)	Normally used in top−down design, this command produces a rectangular symbol. It allows you to design a symbol without leaving *Schedit*. You can control the size of the symbol either by specifying coordinates when invoking the command or by using the rubber box mode afterward. You must supply a name for the symbol.
`Add Port to New Symbol` *Schedit* only)	Lets you put ports on the symbol created with the `New symbol` command. Once placed, these ports cannot be changed in *Schedit*. The port type can be `input`, `output`, `bidirectional`, or `undefined`.
`Create Symbol from Schematic` (*Schedit* only)	Normally used in bottom−up design, this command creates a rectangular symbol, whose size you may adjust. The symbol represent the schematic being edited in a hierarchical structure.

5.5.3 Manipulating Elements − The `Edit` Menu

The following commands may be accessed from the `Edit` menu.

`Undo`	This command undoes the last change you made to the schematic or symbol, even if the last command was `Undo`. View manipulation and other commands that do not modify the schematic or symbol are not undone.

`Copy With...` There are four different ways to copy symbols or wires:

`Both Instance Properties and Wire Names` Copies the selected set, using the same instance properties and wire names assigned to the original set. When using this command to copy wires, remember that *schedit* assumes all wires with the same name are electrically connected.

`No Instance Properties or Wire Names` Copies the selected set to a new location, assigns new wire names, and resets the properties on these symbols and wires to their default values. The wire names will be automatically generated by *schedit*.

`Instance Properties` Copies the selected set, retaining all of the instance properties from the original set. New wire names are generated automatically by *schedit*.

`Wire Names` Copies the selected set, retaining wire names.

The `Copy With...` commands can be used in two different ways.

1. Invoke the command. Then you can create new copies of the current set by dragging the items with the cursor to the appropriate locations and pressing the `COPY` button on the mouse. If there is no current set, you will be prompted to select items first. When you have made all the copies you want, press the `DONE` button.

2. Place points before invoking the `Copy` command. If there is a current set, the first point marks a particular part of the current set and the remaining points specify where that part of the current set is to be placed when the copies are made. If there is no current set, the points will mark which items to select, and then you will be in the copy mode as described in the previous paragraph. When you have finished placing the points, invoke the `Copy` command and the editor will make all of the requested copies.

`Delete` This command deletes the current set from the symbol or schematic.

`Move` This command moves the current set without rotating it. You can use the `Move` command in the following ways:

1. Invoke the command, and with the cursor drag the current set items to the new location, then press the `POINT` button. If there is no current set, you will be prompted to select items first.

2. Place two points, and then invoke the command. The current set will move to a new location determined by its current location and the distance between two points. For

example, to move the current set 40 units to the right and 20 units down, place one point anywhere, and place a second point 40 units to the right of and 20 units down from the first point. Then invoke the `Move` command.

`Rotate` This command enables you to change the orientation of the current set in increments of 90 degrees, rotating counterclockwise.

`Flip X` Flips the current set over a horizontal line. If you place a point before invoking this command, `Flip X` flips the current set about a horizontal line through the point. If you do not place a point before invoking the command, the current set flips about its horizontal center line.

`Flip Y` Flips the current set over a vertical line. If you place a point before invoking this command, `Flip Y` flips the current set about a vertical line containing the point. If you do not place a point before invoking the command, the current set flips about its vertical center line.

`Clear Points` This deletes all points ("+" labeled with a number) currently being displayed in the design area.

5.5.4 Commands for Viewing – The `View` Menu

You have a great deal of control over what is displayed in the design area. By panning, you can move the drawing around the design area. By zooming, you can magnify or shrink what is visible in the design area, creating the impression that you are moving in closer or pulling back. By viewing specific areas of your design, you can see just the part of the schematic you want to work on in detail.

The following commands may be accessed from the `View` menu.

`Redraw` Clears and then redisplays the design area. Occasionally, when you remove elements such as wires from the display, you will find that little pieces of the remaining elements get written over. To restore an accurate display, select `Redraw`. Also, when you change the status of certain display options, for example, `Default Names Visible`, you must invoke the `Redraw` command to see the new display status.

`Redraw Area` This command is like `Redraw` but only redraws that portion of the design that you have selected. This is useful in large, complex schematics that take a long time to redraw entirely.

`View All` This command zooms the view out so that the entire schematic or symbol is visible in the window. If the grid dots would be too close together at this zoom level, they will not be displayed.

`Zoom In`	Invoking this command zooms the display in, keeping the same center. The amount of zoom is selected from the `Options®View Options` menu.
`Zoom In Under Cursor`	This command zooms the display in, centering the view around the cursor.
`Zoom Out`	Invoking this command zooms the display out, keeping the same center. The amount of zoom is selected from the `Options®View Options` menu.
`Zoom Area`	This command lets you zoom in on a specific area of the design. There are three ways to use `Zoom Area`. 1. Place two points in the design area and then enter the command. This causes the view to zoom in as far as it can without causing either of the two points to leave the design area. 2. Place a point in the design area and then invoke the command. This pans the view so that the point you placed is in the lower left corner of the window, without changing the magnification. 3. Enter the command without placing any points. In this case, you are placed in the rubber box mode and you may use it to specify a rectangle. Then the command zooms in as far as it can without forcing any part of the rectangle out of the design area.
`Center Specified Point`	Lets you select any point on the screen and center the screen on it. This command maintains the current display perspective. Place a point before selecting this command, or you may invoke the command and CapFast will center the location under the cursor (as in `Zoom In Under Cursor`).
`Save View`	If you want to save a view that you have obtained by zooming, panning, or viewing, enter the `Save View` command. To return to that view after panning, zooming, or viewing again, enter the `Restore View` command.
`Restore View`	Invoking this command returns you to the last view marked with the `Save View` command, at the original display settings. In other words, this command returns you to the marked view and displays it exactly as originally marked.

5.5.5 Drawing Symbols – The `Draw` Menu (*Symed* Only)

Symed lets you draw circles, arcs, rectangles, and series of straight line segments. The following commands may be accessed from the `Draw` menu:

`Linestyle` Displays the menu for specifying the linestyle of lines drawn with *Symed*. See *Chapter 7: Plotting Schematic Hardcopies* for more information. The possible linestyles are:

```
Dots
Dash dot dot
Dash dot
Dashes
Solid (the default)
Bold solid
Double bold solid
Triple bold solid
```

`Vector` You can draw a connected series of lines with *Symed*'s `Vector` command. If you place points before entering the command, the lines are drawn from point to point in the order in which the points were placed, just as with *Schedit*'s `Draw Wire` command. Otherwise, you can enter the command and then use rubberband mode to draw the lines.

`Circle/Arc` To draw a circle, you can use the `Circle/Arc` command without placing any points beforehand. When you do this, you enter the rubberband circle mode, which lets you draw a circle interactively. Click the left mouse button when the pointer is where you want the center of the circle to be. Click it again when the circle is the size you want. Alternatively, you can first place two points, one for the center of the circle and the other for a place on the circle's edge, and invoke the command.

Figure 5–4. Drawing an arc in *Symed*

To draw an arc with the `Circle/Arc` command, first place three points as follows. Imagine a circle containing the arc and place the first point at the center of the circle. Next, place a point on the circumference such that the arc is a path traced along the circumference beginning at that point and extending *counterclockwise*. Third, place a point to indicate the other end of the arc. Note that because it would not necessarily be possible to place the third point exactly on the circle's circumference, you may approximate it. When you draw the arc, *Symed* terminates it at the invisible line between the first point and the third point. After placing these three points, select the Circle/Arc command. See the example to the left about the order of placing points.

`Rectangle` You can draw a rectangle with *Symed*'s `Rectangle` command, either by first placing points that mark diagonally opposite corner points, or by entering the command first and then using the rubber box mode to define the rectangle.

5.5.6 Bringing Parts into a Schematic – The `Part` Menu (*Schedit* Only)

The `Part` menu lets you import a symbol, border, or schematic connector into your schematic. Before you can use the `Part` menu, you must first select which part library to use. This is done in `Configure®Select Part Library`. The part library you use will determine what is on the `Part` menu.

Specify the location desired for the lower−left corner of the symbol's bounding box either by placing a point there before invoking the `Part` command, or by dragging the symbol there after invoking the command.

You may bring a symbol into the schematic editor in one of three ways. One way is by selecting the name of the symbol from one of the sub−menus in the `Part` menu. (A complete list of all the symbols in the CapFast library is available upon request).

The second way to import a symbol is to use the `Get By Name` command which is available in the `Part` menu for all standard part libraries, and entering the file name at the prompt. Note that you do not need to include the file's extension `.sym`.

The third way is to type:

> **`:get`** *`symbol`*

which is actually the internal command that `Get By Name` uses.

The last two methods will work for most symbols, but not all. The symbols that they cannot find are those that use a common symbol file and just add instance properties. For example, all the diodes in the PCB library's `Part→Discrete Semiconductors→ Diodes and Rectifiers→1) 1N914 − 1N6098 Diodes` menu all use the symbol file `diode.sym`, with different instance properties. Typing **`:get 1N914`** won't work because there is not a symbol file by that name.

When getting symbols, the editor searches for the symbol file in the directories specified in the search path set from the `Options→Search Path` menu.

The following is a summary of the sub−menus available from the `Part` menu for different part libraries. The menus listed for each part library are only available after the `Part` menu has been configured to use that part library using `Configure→Select Part Library`.

5.5.6.1 PCB Parts Library

The PCB parts library lets you access the following `Part` sub−menus:

`Power`	Several standard options are available for power sources.
`Ground`	Several standard grounds are available.
`Schematic Connectors`	This menu contains offpage, onpage, and hierarchical connectors, which do the following:

> Offpage connectors pass signals between pages of a multi−page design.

Onpage connectors pass signals between points on the same page of a schematic design.

Hierarchical connectors pass signals between levels of hierarchy. This kind of schematic connector corresponds to a symbol port at a higher hierarchical level. There are four types of hierarchical connectors: `Input`, `Output`, `Bidirectional`, and `Undefined`.

`Border` — You may use the `Border` sub−menu to give your schematic a finished appearance when printed. Besides a border, the schematic can also have a title and revision history block. You can use the border and blocks provided, or you can create your own border and block symbols using *Symed*. Be aware, when using borders and blocks, that such symbols must have special properties to be acceptable to some netlisters.

A menu containing the following entries allows you to add to your design either a complete border which includes a title and revision block, or the separate pieces:

```
Complete border (with title + rev. blocks)
Border only
Revision block only
Title block only
```

Once you have selected an entry, a second sub−menu pops up which allows you to specify the size of the item selected. Borders may be of size `A`, `B`, `C`, `D`, or `E`. Title and revision blocks are one of two sizes: `A, B, or C` and `D or E`.

The amount of the design area encompassed by the border is dependent on the current inter−pin distance. A smaller inter−pin distance increases the area encompassed. The inter−pin distance must be set correctly before the border is placed so that the border is sized correctly. For more information on the inter−pin distance, see the `Options` menu. For information on printing schematics with borders, see *Chapter 7: Plotting Schematic Hardcopies*.

`TTL/Logic` — This menu separates the subsequent TTL/Logic part menus into:

```
Standard DIP
Chip Carrier
W type package
```

`ECL` — ECL parts (10K and 100K series).

`CMOS` — CMOS parts (4000 series).

`uP/Controllers` — Microprocessor and microcontroller symbols.

`Interface/Peripheral` — Interfaces and peripherals divided by manufacturer:

```
Intel
Motorola
AMD
Signetics
Other peripherals
```

`Memory` Various memory components.

`PLD` Programmable Logic Devices made by various manufacturers.

`Linear` The following menus are available when this menu item is chosen:

```
Analog
Connectors/switches
D/A / A/D converters
Operational amplifiers
Voltage comparators
Linear−other
```

`Discrete semiconductors` The following sub−menus are available when this menu item is chosen:

```
Bipolar transistors
Current regulator diodes
Diodes and rectifiers
Field effect transistors
Power mosfets
Quartz crystals
Voltage−variable capacitance diodes
Zener diodes
```

5.5.6.2 SUSIE Parts Library

Symbols specifically tailored to the SUSIE simulator are available from this library's `Part` sub−menus, in addition to the `Power`, `Ground`, `Schematic Connectors`, and `Border` sub−menus described in the PCB library section.

```
1) TTL 54/74xxyyy (y:00−73)
2) TTL 54/74xxyyy (y:74−175)
3) TTL 54/74xxyyy (y:191−688)
4) Microprocessors
5) Memory
6) PLDs
7) Switches + Control Symbols
```

5.5.6.3 Hilo Primitive Library

The Hilo sub−menus include `Power`, `Ground`, `Schematic Connectors`, `Border`, and a menu of Hilo primitive elements for use with the Hilo simulator.

5.5.6.4 Spice Parts Library

The following sub−menus or commands contain a hierarchy of menus containing symbols tailored specifically for use with the SPICE simulators. Symbols for specific SPICE simulators, such as ISSPICE and PSPICE are available from their respective sub−

menus. Also available are the `Power`, `Ground`, `Schematic Connectors`, and `Border` sub−menus described in the PCB library section.

```
Meters/Control
Sources
Active Devices
Passive Devices
Subcircuit connector
1) ISSPICE library
2) PSPICE library
```

5.5.6.5 Actel Parts Library

The Actel `Part` menu contains a `Power` and `Ground`, and the `Schematic Connectors` and `Border` menus described in the PCB library section. It also contains the following sub−menus:

```
Adders
Buffers
Clock Parts
Digital Logic (Gates)
D Flip−Flops
I/O Buffers
J−K Flip−Flops
Latches
Muxes/Latched
```

5.5.6.6 Xilinx Parts Library

The Xilinx `Part` sub−menus contain symbols tailored for use with the Xilinx interface. Three families of Xilinx parts are available: 2000, 3000, and 4000 families. Select one of them from `Configure`→`Select Parts Library`→`Xilinx Library` menu. In the `Part` menu for all three families are a `Power`, a `Ground`, and the `Schematic Connectors` and `Border` menus described in the PCB library section, in addition to the items listed below.

The 2000 family has the following `Part` menu items:

```
D Flip−Flop
Data Latch
Digital Logic (Gates)
IOB Schematic Elements
Miscellaneous
```

The 3000 family has the following `Part` menu items:

```
Counters
Digital Logic (Gates)
Encoders/Decoders
Flip−Flops
General
Digital Logic (Gates)
Latches
Multiplexers
Registers
Miscellaneous
```

The 4000 family has the following `Part` menu items:

```
Arithmetic Functions
Buffers
Comparators
Counters
Decoders
Encoders
Flip−Flops
Gates
I/O Functions and PADs
Memories
Multiplexers
```

5.5.6.7 Flowchart Library

The Flowchart library contains symbols used in creating flowcharts. In addition to the `Schematic Connectors` and `Border` sub−menus, the Flowchart `Part` menu contains:

```
Arrows
Document symbols
External storage
Main symbols
```

5.5.7 Documentation Aids − The `Text` Menu

The commands in the `Text` menu deal with text that is not a property name or value, but is used to label parts and document a design. Text defaults can be specified or changed in the `Options`→`Text Options` menu. If the `Commands Affect All Text` option is set to `YES`, changes will apply to all selected instance names and text strings; otherwise you are prompted for a change to each selected text string. The following commands may be accessed from the `Text` menu.

`Add` — To place a line of text, invoke the `Text`→`Add` command. You specify the text to be added when prompted. You position it by placing a reference point before invoking the command or by dragging the text into position after entering the text.

To place multiple lines of text, place points where the lines are to go before invoking the `Text`→`Add` command. CapFast prompts you for the text, going from point to point until you have entered text at all of the points. A carriage return ends each line of text.

`Relabel` — You can alter either unassociated text or text that is associated with the name of a port, symbol, or wire with the `Text`→`Relabel` command. To use this command, make the unassociated text, port, symbol, or wire the current set and invoke the command. Specify the replacement text at the prompt. When using `Text`→`Relabel` with ports, symbols, or wires, the text you change is the *instance name* of the port or symbol. When you use `Relabel` with non−associated text, it replaces the currently selected text. Even if the `Commands Affect All Text` option in the `Options`→`Text Options` menu is set to `YES`, you are still prompted for a label for each item.

`Move`

This command moves text and element names. To move displayed text, make it the current set and invoke the `Text`→`Move` command. There are two ways to specify the new position of the text. One is to place two points before invoking the command. The first point acts as a reference point for the text in its present position. The second specifies where the first point would come to rest if the text and the first point are moved together. The other way to specify the new position of the text is to invoke `Text`→`Move` first, then drag the text into position.

`Orientation`

Lets you specify how text is to be oriented. The possible values are horizontal (`normal`), up (`up`), and down (`down`). Rotation occurs with respect to the defining point of the text, based on the justification. The default is `normal`.

`Scale`

This command specifies the size of the characters, as a percentage of the default size. The possible values range from 25 through 10,000 (integers only). The default is `100`.

Figure 5−5. Text justification options in CapFast.

`Justification`

This command lets you specify how displayed text is positioned. CapFast allows five vertical and three horizontal text positions, as shown in the illustration. You will be prompted for the horizontal and then vertical justification. The CapFast text position default is `left` and `base`. The five vertical options are as follows: `top` is the top of the tallest letters like l, T, k; `cap` is the top of lower−case letters such as a, c, g; `center` is the vertical center of lower−case letters such as a, c, g; `base` is the bottom of lower−case letters such as a, c, e; and `bottom` is the bottom of the lowest letters such as p, g, y.

When you specify where to place text, you click on a particular point to place the text at. The text is drawn in a position relative to that point based on what the text `Justification` is. So, for example, if the `Justification` is `cap left`, the text would be drawn so that the left edge of the first character in the text is at the point, and the top of the lower−case letters (such as 'a') are at a horizontal line going through the point.

`Display`

This command specifies whether the selected text is to be visible or invisible. The possible values are `on` (for visible) and `off`. The default is `on`.

5.5.8 Adding Ports to a Symbol − The `Ports` Menu (*Symed* Only)

The following commands may be accessed from the *Symed* `Ports` menu:

`Visible` — This command brings up a sub−menu that lets you place a port with the chosen function on a symbol. You may place points for all port locations with the same function, then invoke the `Ports−Visible` command, or you may drag each port into position. The following port types are available: `Input`, `Output`, `Bidirectional`, and `Undefined`.

You should draw a vector to show attachment of the port to the main part of the symbol. This makes the symbol instance in *Schedit* look like its ports are wires that extend a short distance out from the symbol body, with port function icons shown at the end.

`Invisible` — This command is used to create a port that won't be displayed in a schematic. This command is used instead of the `Port`→`Visible` command when you don't want a port to be displayed in a design: for example, if the port is for GND or Vcc.

`From Schematic...` — Use this command to map the hierarchical connectors of a schematic into the ports of the symbol file.

`Change Definition` — This command changes the function and name of a symbol port. To use this command, select the ports that you want to change and invoke the command. Then enter the port definition or `<Enter>` for no change. Valid port functions are `Input`, `Output`, `Bidirectional`, and `Undefined`. You only need to enter the first letter of the function type. Then enter the new name or `<Enter>` for no change.

5.5.9 Manipulating Wires and Buses − The `Wire` Menu (*Schedit* Only)

The commands on the `Wire` menu let you draw a wire from one port, connector, or wire to another. Using *Schedit* consists primarily of bringing in symbols and connecting them together at their ports. The most common way to connect ports is with wires. You can also connect symbols' ports by placing one symbol's port directly on another symbol's port.

The following commands may be accessed from the `Wire` menu. (See the section entitled *Wires and Buses* in *Chapter 4: Creating Electrical Designs*, for more information.)

`Draw Wire` — This command places wires or buses along a path that you specify. There are two ways to use the `Draw Wire` command.

1. Before invoking the `Draw Wire` command, place points, in order, along the path that the wire is to follow. The first and last points mark the endpoints. Then, when you invoke the `Draw Wire` command, *Schedit* automatically routes the wire along the path specified by the points.

2. Invoke the `Draw Wire` command without placing any points beforehand. This puts you in the rubber band mode. Use the `POINT` button to mark one of the endpoints, and then use the same button to mark each corner of the wire until you mark the other endpoint. If you make a mistake, press the `BACK−UP` button on the mouse to remove the last point. Press the `DONE` button or `<Space>` to leave the rubber band mode. This way of using the `Draw Wire` command gives you more control over the route taken by the wire because of the immediate feedback of seeing exactly where the wire will be routed.

Each time you place a wire, you will be asked for a wire name unless:

1. The `Prompt for Wire Names` option is set to `NO` in `Options→Name Prompting`. If it is set to `YES`, you can supply a name, or you can just press `<Enter>` to tell *Schedit* to generate a name for you. The name it generates will by default be of the form `n#x`, where `x` is an integer from an on−going index starting at 0. This naming scheme can be changed in `Options→Default Naming Scheme`.

2. You are connecting the wire to another wire.

3. The `Options→Wire Options→Name Wires After Connectors` option is `YES` and you are wiring to a connector.

4. You are using the `Auto−name Bus Wires` feature.

`Re−Route`

`Re−Route` is used to re−route existing wires along new paths. If you are not satisfied with the placement of a wire, you can use the `Re−Route` command to move one or more of the vertices where the straight segments of the wire meet. To use `Re−Route`, first select the wire you want to re−route, then invoke the command and move the cursor to the vertex that is to be moved. Use the `POINT` button to pick up the vertex, then drag the vertex to its new location, and use the `POINT` button again to place the vertex in its new location.

`Alias Symbol`

This command produces an alias symbol. The alias symbol lets you give two names to a wire or bus, or electrically connect two wires with different names. It is placed between the differently

	named wires, busses, or signals. When joining differently named buses, make sure that they have the same line widths. See *Chapter 4: Creating Electrical Designs* for more information.
`Draw Bus`	This command does the same thing as the `Draw Wire` command.
`Bustap Symbol`	The `Bustap Symbol` command is used to separate a signal off of, or merge a signal onto, a bus. Use this command to connect wires with the same name, and with subscripts in the same range as those of the wider (bus) line. See *Chapter 4: Creating Electrical Designs* for more information.
`Auto-name Bus Wires...`	This command allows the editor to automatically name a series of signals with the same base name and with an index number that increments by one. You will be prompted for a base name and a starting index number. Each wire you draw thereafter will be automatically named. After placing the final wire, exit this command by selecting it again to toggle it off.
`Draw Bundle`	This command is used to draw a bundle. It is used the same way as the `Draw Wire` command. See the section on *Wires, Buses, and Bundles* in *Chapter 4: Creating Electrical Designs* for more information.

5.5.10 Controlling Symbol Characteristics – The `Properties` Menu

Properties are text strings that define the electrical, physical, and logical characteristics of the symbols, wires, ports, and connectors in a schematic design. Properties are used to pass information to netlisters.

Each symbol, wire, or connector can have a list of properties associated with it. Global properties are those assigned with *Symed* at symbol creation time, which are common to every instance of a particular symbol. Local properties are those assigned with *Schedit* to an instance of a symbol.

Property text strings take the following form:

```
[(QUALIFIER)]NAME:VALUE
```

or

```
[(QUALIFIER)]NAME(PORT):VALUE
```

where *`QUALIFIER`* indicates to a netlister whether the property is to be included in the netlist file; *`NAME`* is the name of the property; *`PORT`* is the symbol port name to which the property applies; and *`VALUE`* is one or more numbers or strings that specify the value of the property name.

See *Chapter 6: Properties* for more detailed information on properties.

You can select the following commands from the `Properties` menu:

`Edit...`	This command brings up the `Properties` window. For more information on this window, see its section below.
`Read Rules File...`	Rules files allow you to define rules or values which properties must meet to be valid. A property rule is used to assign to a specific property a range or enumeration (list) of values the property may assume. This limits the possible legal values that may inadvertently be assigned to a property. For more information, see the section on *Property Rules Files* in *Chapter 6: Properties*.

5.5.10.1 Assigning Symbol Characteristics − The `Properties` *Window*

You can edit properties by selecting a symbol, wire, or connector as the current set and invoking the `Edit...` command from the `Properties` menu. If the current set contains several symbols, the *Schedit* or *Symed* selects one of the symbols, temporarily changes its color, and displays its properties window. Each time you click on `OK`, another symbol in the current set is selected and its properties are displayed.

The `Properties` window is shown below for a 74LS00, with the `ref` property highlighted. Properties of the selected symbol, wire, or connector are displayed in the big box in the `Properties` window, sorted by property name.

You can scroll through the list of properties by using the standard scroll bar on the properties list box. If a property value is too long to see entirely in the box, then select the property, click in the `Value:` box, and use the cursor keys to scroll through it horizontally. To select a property, press the left mouse button (left−click) when the pointer is on the desired property. You can select a series of properties by pressing and holding the left mouse button down while dragging the pointer over the series, or by left−clicking on the first property in the series and then pressing `<Shift>` while left−clicking on the last property in the series. You can select multiple non−adjacent properties by pressing `<Control>` while left−clicking on the properties.

There are several functions available through the use of the buttons in the `Properties` window:

`OK` or `Close`	Clicking on this button will close the Properties window, keeping the changes you've made.
`Add/Modify`	Clicking on this button adds or modifies the property currently displayed in the two boxes labeled `Name:` and `Value:`. The colons (":") in each property are added by the editor; you do not need to insert them yourself. If the property you are adding has the same name as an existing property, the new property replaces the existing property. If you want to rename a property, you should change the name of the misnamed property, click on `Add/Modify` to add it with the correct name, and click on `Delete` to delete the misnamed one.

`Display...` This command pops up the `Property Display` window for changing the display characteristics of the properties selected. There are four display characteristics for a property: visibility, text scale, orientation, and justification. Click on `OK` when finished, or on `Cancel` to discard the display changes.

There are three possible values for a property's visibility, selected in the `Visible` box:

`All`	Both the property name and value are displayed in the schematic.
`Value Only`	Only the property value is displayed in the schematic.
`Nothing`	Neither the property name nor its value is displayed in the schematic.

The `Scale` value determines the size of the characters in the property when displayed on the schematic, as a percentage of the default size. The possible values range from 25 through 10,000 (integers only). The default is 100.

The `Orientation` of the property text can be `Normal`, `Up`, `Over`, and `Down`. This controls what direction the text runs. `Normal` is horizontal left to right, `Up` is vertical bottom to top, and `Down` is vertical top to bottom. `Over` is horizontal, but the direction depends on the value of `Options®Text®Link Text Rotation To Symbol`. See the `Text` sub−menu in the `Options` menu section of this chapter for a detailed explanation.

The `Justification` of the property text is the same as described in the `Justification` command in the `Text` menu. In the `Property Display` window, select one of the options in the `Horizontal` box and one in the `Vertical` box.

`Delete` In *Schedit*, this deletes the selected local properties or resets their values back to the global values. Global properties may not be deleted. In *Symed*, this command deletes any selected properties.

`Move` Invoking this command closes the `Properties` window and lets you move the selected properties with the mouse. If multiple properties are selected, they will be moved as a group. Once you have dragged the properties to their new position, press the left mouse button to place them. Selected properties with display visibility set to `Nothing` are invisible, but their location will still change if you move them.

`Copy` (*Schedit* only) This command copies the properties you have selected to another symbol. When you click on `Copy`, the `Properties` window disappears and you are prompted to select a symbol. Once selected, the properties are copied to that symbol.

5.5.11 Selecting and Finding Elements − The `Select` Menu

The `Select` menu contains various commands for selecting elements in the schematic or symbol. The commands that search for an element by name or property make the element, when found, the only member of the current set. They look for an exact match, so case is important.

The following commands may be accessed from the `Select` menu.

`Items`	Unselects the current set and lets you create a new current set by selecting items one at a time. You can either place the points on the items and invoke the command, or select the items via flashlight mode. When in flashlight mode, you use the left mouse button to select items, and the right mouse button to de−select items.
`More Items`	Lets you select items to add to or remove from the current set. If you place the points on the items before invoking the command, the items will be added to the current set. If you invoke the command first, you select the items via flashlight mode and can also remove already selected items from the current set.
`Area`	Unselects the current set and allows you to select an area as the current set. Everything entirely in the area becomes part of the current set. You may either specify the opposite corners of the area rectangle by points placed before invoking the command, or invoke the command and use the rubber box mode.
`Additional Area`	Lets you add items in a selected area to the current set. You may either specify the opposite corners of the area rectangle by points placed before invoking the command or use the rubber box mode.
`Symbol by Name...` (*Schedit* only)	This command finds a symbol or connector whose instance name begins with the search string. When the first symbol or connector is found, it becomes the current set. For example, if the search string is `R`, then the `Symbol by Name` command finds symbols and connectors whose names begin with `R`.
`Next Symbol` (*Schedit* only)	Used in conjunction with the `Symbol by Name` command, this command finds the next symbol whose instance name begins with the search string and adds it to the current set.
`Wire by Name` (*Schedit* only)	This command lets you designate all wire segments with the specified name as the current set.
`Elements by Property` (*Schedit* only)	This command finds all symbols, wires, and connectors that have a particular property and makes them the current set. You can specify the entire property, the property name, or a qualifier. If you specify a qualifier, it must be in parentheses.
`Clear`	Unselects the current set.

5.5.12 Creating Keyboard Shortcuts – The Macro Menu

You can save time as you design schematics or symbols by using this CapFast's macro feature to automate the more repetitive design tasks. These commands are useful when you wish to perform the same sequence of actions more than once.

The following commands may be accessed from the Macro menu.

Record...

Records a sequence of commands that can be played back. When you invoke this command, CapFast prompts you for a new command name. A command name may be one of the function keys <F1> through <F10>, <Alt>–<A> through <Alt>–<Z>, and <Shift>–<Alt>–<A> through <Shift>–<Alt>–<Z>. Press the key you want to assign the command to.

Once you have selected a name for the new command, all the succeeding commands you execute become part of the new command definition. New commands may be a series of actions offset from an initial reference point, or they may be interactively performed. To end a new command definition, Select Macro→Stop after you have invoked the last command. To execute the new command, simply press the key that you assigned the command to. For example, press the <F1> key to replay a command assigned to that key.

To offset actions from an initial reference point, begin the command definition by placing a point. For example, to wire a series of regularly spaced ports, start the macro recording and place an initial reference point on the top port. Before executing succeeding commands in the macro, place locating points first. End the command definition by selecting Macro→Stop.

To execute the new command, first place a new reference point. Select the new command by pressing the key corresponding to its name and all recorded commands will be offset from this point.

If you want to perform a series of actions interactively, do not place an initial reference point. This means that you cannot use the point method for executing commands to be recorded.

Stop

Selecting this command ends the current macro definition.

View Definition...

This command allows you to view the definition of a macro command in terms of the internal commands it represents. You will be prompted for the name of the command. Enter the name of the macro as follows. For function keys <F1> through <F10>, enter **:Func1** through **:Func10** (include the colon). For <Alt>–<A> through <Alt>–<Z>, enter **:Alta** through **:Altz**. For <Shift>–<Alt>–<A> through <Shift>–<Alt>–<Z>, enter **:AltA** through **:AltZ**.

`Save...`	This command lets you save the macro commands for future editing sessions. These commands are saved in the `schedit.rc` or `symed.rc` file which is read in at program startup time. Note that the CapFast editor deletes new macro commands at the end of the current editing session unless you explicitly save them with this command.
`Unsave...`	Invoking this command removes the definition of the selected macro command from the `schedit.rc` or `symed.rc` file which is read in at program startup time. CapFast prompts you for the name of the macro command that you want to delete. Note that the selected macro command will still work until it is redefined or until you exit CapFast.

5.5.13 Controlling the Design Area − The `Configure` Menu

Several factors influence how and where points can be placed. When you specify the desired location of a point, any of these factors can cause the actual location of the point to be different from the specified location.

Two particular factors are *alignment distance* which you can change with the `Snap Grid Size` command, and *alignment adjustment*, controlled through commands in the `Drawing Constraint` sub−menu. The `Snap Grid Size` constrains the placement of points by requiring points to be placed on integer multiples of the alignment distance, relative to the point with coordinates 0 0 (the origin). For example, if the `Snap Grid Size` is 5, then you can place points at any location where both coordinates are multiples of 5, such as −5 25, but you cannot place a point at 3 10, because the first coordinate is not a multiple of the `Snap Grid Size`. The point would actually be snapped to position 5 10. Thus, the `Snap Grid Size` specifies the minimum distance that can exist between points in the drawing space. Also, if you try to place a point too close to an existing point, the new point ends up on top of the old point. This can cause wires to be connected that you didn't intend to be, so be careful.

The following commands may be accessed from the `Configure` menu.

`Snap Grid Size`	Sets the design area's snap grid size − the minimum number of design units between points. The placement of points is also restricted so that their horizontal and vertical distances from the origin (the point with coordinates 0 0) are always integer multiples of the snap grid factor. The default is 16 design units. Valid values are integers in the range 1 through 256.
`Visible Grid Size`	This controls the distance between the grid dots that are displayed in the design area. If this value is too small for the current zoom level, the dots will not be displayed. The default is 32 design units. Valid values are integers in the range 2 through 256.

`Drawing Constraint`	This brings up a sub−menu of ways to constrain the placement of points. These commands do not modify parts or wires. See the section below about the `Drawing Constraint` menu.
`Select Parts Library`	This brings up a sub−menu of parts libraries to choose from. The parts libraries are described in the `Part` menu section. The default options are:

```
PCB Library
SUSIE Library
Hilo Library
Spice Library
Actel Library
Xilinx Library
    2000 Family
    3000 Family
    4000 Family
Flowchart Symbols
```

5.5.14 Controlling Drawing Alignment − The `Drawing Constraint` Menu

The commands in this sub−menu of the `Configure` menu constrain how points can be placed. This includes points placed interactively, such as the vertices of a wire when invoking the `Wire®Draw Wire` command without placing points first. `Orthogonal`, `Horizontal`, and `Vertical` are mutually exclusive, meaning that only one of the three commands may be in effect at one time. If one of these commands is in effect, selecting one of the others overrides it.

`None`	Selecting this command turns off `Orthogonal`, `Horizontal`, and `Vertical` alignment.
`Orthogonal`	`Orthogonal` alignment forces any point you place to be either directly above, below, to the left, or to the right of the last point placed (it has no effect when you are placing the first point). That is, points placed will be forced to lie on the two orthogonal lines going through the first point placed which are parallel to the x and y axes. For example, if the last point you placed was at (80 120), and you had alignment adjustment set to `Orthogonal`, you would only be able to place points whose x coordinates were 80, or whose y coordinates were 120. If you tried to place a point at 140 200, it would end up at 80 200. This can be useful for routing wires.
`Horizontal`	`Horizontal` alignment is like `Orthogonal` but more restrictive. It forces to points placed after the first one to fall on a horizontal line.
`Vertical`	`Vertical` alignment is similar to `Horizontal`, but of course it forces points to fall on a vertical line.

5.5.15 Controlling the Editing Environment − The `Options` Menu

The commands or sub−menus accessible from the `Options` menu help you customize the editing environment. Some of these commands and sub−menus are found in *Symed* and some are found in *Schedit*.

5.5.15.1 Saving Your Option Changes

By default, each time you change an option, the following prompt appears:

```
Variable_Name = Value: Save change [n]?
```

If you type **`<Enter>`** or **`n`**, then the option change is in effect temporarily, lasting only as long as the current invocation of the editor. If you type **`y`**, then the option change is put into effect and will be in effect each time you invoke *Symed* or *Schedit*. The editors save this change by writing into one of three control files which are read when the editors are started. These control files are `cad.rc`, `schedit.rc` and `symed.rc`. For more information, see the *Control Files* section of *Appendix A: Customizing CapFast*.

When you save a change made to the data file search path through the `Search Path` sub−menu, it is saved to the file `cad.rc`. All other preferences set through the `Options` menu, and all user−defined commands from the `Macro` menu, are saved to either the `schedit.rc` or `symed.rc` file, depending on which editor you are using.

By default, all changes are recorded in the files located in the directory ~. If there is no `cad.rc`, `schedit.rc`, or `symed.rc` file located in this directory, the most recently used `cad.rc`, `schedit.rc`, or `symed.rc` will be copied to the directory ~, and the changes written into the copy. If no `cad.rc`, `schedit.rc`, or `symed.rc` has been used, a new file is created in the directory ~.

The next time you start the editor, the control files `cad.rc` and `schedit.rc` or `symed.rc` will be looked for first in the current directory, next in the directory ~, and last in the directory `~p3/wcs/lib`. Once the editor locates the files in one place, it will look no further; the defaults included with the software are thus overridden since they are stored in the location last on the list.

5.5.15.2 The `Options` Menu

The following commands or sub−menus are available from the `Options` menu:

`Allow Message Beeps`	When set to `YES`, allows message beeps. The editor will make a beep when it wants to get your attention to answer a question or notify you of something. The default is `YES`.
`Display Cursor Position`	Toggles the continuous reporting of the current cursor position in the mouse button area of the screen. The x and y coordinates are reported in design units. The default is `YES`.
`Control File Location`	Specifies the directory where changes to control files are saved. Resetting the search path and saving the change through the `Library Options` menu writes the changes into a file named

`cad.rc`. Resetting and saving other options writes the changes to a file named `schedit.rc` (or `symed.rc`). Valid values are `HOME` which is your home directory ~; `LIB` which is the directory `~p3/wcs/lib`; or `CURRENT` which is your current directory. The default value is `HOME`. See the *Saving Your Option Changes* section earlier in this chapter for more information.

`Auto−Save Options`	When this option is set to `PROMPT`, CapFast prompts you with the following question each time you change a display parameter. *VARIABLE_NAME=VALUE*`: Save change [n]?` If you type `y` then the option change you have made continues to be in effect each subsequent time you invoke *Schedit* or *Symed*. The editors save this option by writing into a file which is read upon invocation. If you press `<Enter>` or type `n`, then the changed option is in effect temporarily, lasting only as long as the current invocation of the editor. If this prompt is set to `SAVE`, then no prompt appears and all changes are automatically recorded the same as if you had typed `y`. If the prompt is set to `IGNORE`, then no prompt appears and all changes are ignored, just as if you had pressed the `<Enter>` key for each one.
`Auto−backup`	Toggles the automatic backup feature. If set to `YES`, the editors will back up your work as frequently as specified by `Auto−backup Interval`; if set to `NO`, the editors will not back up your work. Files will be saved in the directory `~/bak`. The default is `YES`.
`Auto−backup Interval`	Lets you assign, in minutes, the interval between automatic backups when `Auto−backup` is set to `YES`. Valid values are in the range of 1 to 60 minutes. The default is 15 minutes.
`Default Naming Scheme`	This sub−menu contains commands to control the default names CapFast gives to connectors, symbols, and wires. The default names will be used only if you do not specify a name in each instance. You can name a connector, symbol, or wire when you are prompted for a name when you first place the element, or by using the `Text®Relabel` command. For more information on setting the default naming scheme, see the section below.

`Inter−pin Distance` (*Schedit* only)	Sets the inter−pin distance for plotting schematics. To be different from the default, this option should be set before a border is brought into a schematic to ensure the properly−sized border symbol is used. Valid options are (in mils):
	
	`100`
	`125`
	`200` (default)
	`250`

	If you find that even the largest border provided is not large enough, you can either draw your own, or decrease the inter−pin distance. A value of 100 gives you four times as much room as 200.
`Name Prompting`	This sub−menu contains commands to control when the editors prompt for names of objects.
`Object Visibility`	This sub−menu contains commands to control what objects are visible. They take effect when you redraw the screen, whether by the `View®Redraw` command, or the zoom commands.
`Search Path`	This sub−menu contains commands to alter and view the search path used by the editors to find schematic and symbol files. It also contains some commands to change and view the current working directory.
`Text Options`	This sub−menu contains commands to modify how text is displayed by default, and how commands affect text.
`View Options`	This sub−menu contains commands to modify the magnitude of the zoom and pan commands.
`Visible Grid Linestyle`	This sub−menu contains commands to modify what the visible grid looks like.
`Wire Options`	This sub−menu contains commands to control how wires may be drawn and named, and whether error checking is done automatically.

5.5.15.3 The `Default Naming Scheme` Sub−menu

If you want, the editors can automatically assign instance names to design elements, so you don’t have to think up names for them. *Schedit* automatically assigns names to instances of components, connectors, and wires, and *Symed* can automatically assign names to both visible and invisible ports. The automatically−assigned names all have a common format. They consist of some base characters followed by an append character, followed by some digits (an ongoing index). Here are examples of automatically−assigned names. In each pair of examples, the first is typical of default name−assignment scheme, and the second shows what automatically−assigned names might look like were you to change the scheme. In each example, the last characters that are digits are the ongoing index, and the non−alphabetic character is the append character.

Connector names:
```
c#12
q:12
```

Symbol names:
```
nand#12
instance=12
```

Wire names:
`n#22`
`w~22`

Invisible Port names:
`p#04`
`i%4`

Visible Port names:
`p#03`
`v^3`

You can control some aspects of how automatically−assigned names are generated. For all elements, you can control the append character (which is # by default) with the `Append character(s)` command. The base characters can be controlled for all element types, but note the exception for certain connector symbols, described below.

`Append Character(s)`	Lets you specify the character that the editor appends to the base characters to generate a prefix when automatically assigning instance names. The default string is `#`. **Note:** It's not advisable to change the append character to something other than `#`, because many CapFast netlisters assume that the append character is always `#`. If you were to create a schematic that used some other append character, those netlisters would not work correctly when processing that schematic.
`Connector Base Character(s)` (*Schedit* only)	Lets you specify the character string or number string that *Schedit* uses as a prefix when automatically generating connector names. If you leave it blank, no base characters are used when naming connectors. The default is `c`.
`Symbol Base Character(s)` (*Schedit* only)	Lets you specify the characters that *Schedit* uses as the base when generating names automatically, for most symbols. If you leave it blank (which is the default), the editor will use as the base characters the *filename* of the symbol being placed, unless there is an `id` property on the symbol. See *`id` Property* in *Chapter 6: Properties*. The CapFast symbol library contains some symbols that function as connectors; when *Schedit* assigns names to these symbols, it uses the `Connector Base Character(s)` instead of `Symbol Base Character(s)`. These symbols are the offpage, onpage, and hierarchical connectors and the SPICE subcircuit connector. Unlike other symbols, if you leave `Connector Base Character(s)` blank, *Schedit* will not use the name of the symbol as the base characters, but will in fact use no base characters.
`Wire Base`	Lets you select the alphanumeric string that *Schedit* uses as a

`Character(s)` (*Schedit* only)	prefix when automatically generating names for wires. The default prefix is `n`.
`Invisible Port Base Character(s)` (*Symed* only)	Lets you select the alphanumeric string that *Symed* uses as a prefix when automatically generating names for invisible ports. The default prefix is `p`.
`Visible Port Base Character(s)` (*Symed* only)	Lets you specify the character string or number string that the editor uses as a prefix when automatically generating port names. The default is `p`.

5.5.15.4 The `Name Prompting` Sub−menu (Schedit Only)

You can control when *Schedit* prompts you for names of elements. By default, you are only prompted for wire names. You can set these options so that you are also prompted for symbol and connector names when you place them:

`Prompt For Connector Names`	Lets you enable or disable the prompts that *Schedit* issues for connector names. If you disable the prompt, schematic connector names are assigned automatically, as if you had responded to each connector's prompt with **`<Enter>`**. The default is `YES`.
`Prompt For Symbol Names`	Lets you enable or disable the prompts that *Schedit* issues for symbol names. If you disable this option, schematic symbol names are assigned automatically when you place them, as if you had responded to each `Instance Name?` prompt with **`<Enter>`**. If you enable this prompt, you are asked to enter a symbol name. The default is `NO`.
`Prompt For Wire Names`	Lets you enable or disable the prompts that *Schedit* issues for wire names. If you disable this option, wire names are assigned automatically, as if you had responded to each wire name prompt with **`<Enter>`**. The default is `YES`.

5.5.15.5 The `Object Visibility` Sub−menu

You can control whether various things are displayed by default. These include default names, connector names and properties; symbol names, properties, and ports; wire names and properties; and the grid. The following commands are available from the `Object Visibility` sub−menu:

`Connector names visible`	Controls whether user−assigned connector names are visible. The default is `YES`.
`Default names visible`	Controls the visibility of default names assigned by the editors. The default is `NO`.
`Grid visible`	Controls the display of the grid on the screen. The default is `YES`.

`Properties visible`	Controls the display of properties. The default is `YES`. This command overrides the display option you set for a property in the `Properties` window.
`Symbol names visible`	Controls the visibility of user−assigned names for symbols other than connectors. The default is `YES`.
`Symbol text visible`	Controls the display of text that is associated with symbols (the symbol label). The default is `YES`.
`Symbol ports visible`	Controls the display of symbol ports. The default is `YES`.
`Wire names visible`	Lets you control whether user−assigned wire names are displayed. The default is `YES`.

5.5.15.6 The `Search Path` Sub−menu

The `Search Path` sub−menu contains commands to change and view the search path used by the CapFast editors to locate symbol files (whose names end with `.sym`), and schematic files (whose names end with `.sch`). It also contains commands to change and view the current working directory. The following commands are available from the `Search Path` sub−menu:

`Show Search Path` Displays the current list of directories that will be searched by CapFast for symbols or schematics whenever one is being loaded. The current search path is displayed at the bottom of the screen in the command and response area.

`Define New Search Path` Changes the search path used by CapFast when you load a symbol or schematic. This command changes the default search path to the path you specify.

If you create your own library of symbols and place them in a special directory, or place schematics in directories other than the current directory, you may need to change the CapFast search path.

The search path used by *Schedit* and *Symed* is set by the `cad.rc` file, which is read at startup. If the `cad.rc` does not specify the path for *Schedit* and *Symed*, or if the `cad.rc` file can't be found, the search path defaults to:

```
.+~+~p3/library/default
```

CapFast uses the plus sign (`+`) as a directory separator. The dot (`.`) indicates the current directory.

Note: Changing the CapFast search path affects only CapFast, not the way your operating system or other applications operate.

`Add Directory To Beginning`	Adds directories to the beginning of the search path used by the CapFast editors. The directories you add will be the first searched. At the prompt for the new directory, give the directory path. You may use ~ or `~p3`, and CapFast will expand them for you.
`Add Directory To End`	Adds directories to the end of the search path used by the CapFast editors. The directories you add will be the last searched. At the prompt for the new directory, give the directory path. You may use ~ or `~p3`, and CapFast will expand them for you.
`Remove Directory From Search Path`	Removes the specified directory from the search path used by the CapFast editors. At the prompt give the directory path to remove from the search path. You may use ~ or `~p3`, and CapFast will expand them for you. Once the directory has been entered, CapFast will display the complete, new search path.
`Print Current Directory`	Prints the full path name of the current working directory. This directory is also referred to as ’`.`’
`Change Current Directory`	Changes from the current directory to the specified directory.

5.5.15.7 *The `Text Options` Sub−menu*

You can specify text options for names of connectors, symbols, and wires, and for unassociated text strings with the `Text Options` sub−menu. The following commands may be accessed from the `Text Options` menu.

`Commands Affect All Text`	Lets you specify how text commands should be applied. You may specify that text commands function on the entire selected set (`YES`). Alternatively, you may specify that CapFast prompt you for each text element in the set (`NO`). The default is `NO`.
`Link Text Rotation To Symbol`	Toggles between the values `NOLINKAGE`, `LIMITEDLINKAGE`, and `FULLLINKAGE`. `NOLINKAGE` maintains the same text orientation regardless of the orientation of its associated object. `FULLLINKAGE` causes text to be rotated with the object it is associated with. `LIMITEDLINKAGE` rotates text with the object, except that text that would be displayed upside down is displayed right−side up instead. The default is `LIMITEDLINKAGE`. This option is useful if you are editing schematics that were created before October, 1989, in which case use `NOLINKAGE` to make the schematic appear as it would have appeared with older versions of *Schedit*.
`Default Property Display`	Controls what subsequently created properties’ display attributes will be by default (also assigned by clicking on `Display...` in the `Properties` window). The values toggle between `OFF`, `ON`, and `VALUE`. `OFF` assigns new properties the display attribute of `Nothing`. `ON` assigns subsequently created properties the

display attribute `All`. `VALUE` assigns subsequently created properties the display attribute `Value Only`. The default is `ON`.

Note: A newly created property may appear far from the symbol it is associated with. This is because CapFast assigns a default position for each property where it would be displayed if its display attribute were turned on. Each property is assigned a position that is about one character down and one and a half characters to the right of the previous property, even if the previous property is not displayed. This keeps properties from being displayed on top of each other when the `Object Visibility®Properties Visible` option is `ON`.

`Default Text Rotation` — Lets you specify how free text is rotated by default. The value toggles between `NORMAL`, `UP`, `DOWN`, and `OVER`. This option does not affect text associated with objects.

`Default Text Size` — Lets you specify text size as a percentage of the default size for the font. The default percentage is 100. Valid values are integers in the range 20 through 10000.

`Default Text Justification` — Lets you specify how displayed text is positioned, relative to its defining point. CapFast allows five vertical and three horizontal text justifications, as shown in the illustration. The defining point for the text is shown as a crosshair. The default is `Base`

Figure 5−6. Text justification options in CapFast.

`Left`.

5.5.15.8 The `View Options` Sub−menu

The following commands may be accessed from the `View Options` menu:

`Zoom In Magnification` — Lets you specify the zoom factor used when you select the `View®Zoom In` command. For example, if in factor is 5.0, then zooming in makes objects in the design area five times taller and five times wider. Thus the view is reduced to 1/25 of the previous area. The default in factor is 1.4. Valid values are real numbers in the range 1.0 through 10.0.

`Zoom Out Magnification` — Lets you specify the zoom factor when you enter the `View®Zoom Out` command. For example, if `Zoom Out Magnification` is 0.2, then zooming out makes objects in the design area 1/5 as tall and 1/5 as wide. The default is 0.7. Valid values are real numbers in the range 0.1 through 1.0.

`Pan Amount` — Lets you specify the fraction of the window width panned when you enter a pan command. The default value is 0.5 (half the window width). Valid values are real numbers in the range 0.1 through 5.0.

5.5.15.9 The `Visible Grid Linestyle` Sub−menu.

The `Visible Grid Linestyle` sub−menu contains the options for specifying the linestyle for the grid. The possible linestyles are:

`Dots` (the default)
`Dotted Lines`
`Dash Dot Dot Lines`
`Dash Dot Lines`
`Dashed Lines`
`Solid Lines`

To specify the distance between consecutive vertical or horizontal grid lines in the display, see `Configure®Visible Grid Size`.

5.5.15.10 The `Wire Options` Sub−menu (Schedit Only)

The following commands may be accessed from the `Wire Options` menu.

`Permit All−angle Wiring` Changing this option to `YES` allows you to draw new wires at any angle. When you toggle this option to `NO`, new wires and wires that you re−route are forced to be horizontal or vertical. All corners are therefore 90 degrees.

`ERC For Wires` If this is `YES`, *Schedit* automatically checks wires when you create them. Here are the things it will check for:

1. Two wires shorted.

   If you attempt to join two wires with different names without using an alias symbol, *Schedit* asks if this is correct. If you answer **`yes`**, and you have assigned both names, *Schedit* assigns the name of the wire at the first point to all wires connected to it.

   On the other hand, if you have named only one of the wires, *Schedit* assigns that name to all wires connected to it. Otherwise, if neither of the wires were named by you, *Schedit* picks the wire name of the wire at the first point and assigns that name to all connected wires.

2. Bus width correspondence to port width.

   Schedit checks a wire name as it is assigned, to see if its width (number of represented wires) is the same as the width of objects that it connects. Bundles and schematic connectors, including bus taps and aliases, have varying widths and can match any wire width up to 256. If widths do not match, *Schedit* does not allow the wire.

If `ERC For Wires` is `NO`, *Schedit* issues warnings but does not interrupt the editing session. The default is `YES`.

`Name Wires After Connectors`	This command tells the editor to name new wires the same as the connector they are attached to, if any. The connector must have been placed first. Changing the name of the connector after the wire has been connected to it, does not change the name of the wire. The default is `YES`.
`Draw Buses Thick`	Lets you make the display of buses bold (`YES`) or thin (`NO`). The default is `YES`.

5.5.16 The `About` Menu

This menu contains only one command, named `About Schedit` (*Schedit*) or `About Symed` (*Symed*). These commands display, in the command and response area, the version and copyright information.

5.6 The CapFast Internal Commands

This section contains a synopsis for each of CapFast's internal commands, whether specific to *Schedit* or *Symed*, or common to both. The commands are listed in alphabetical order.

The format of this section is as follows:

Command	`:`*`internal-command-name`*	Which editor's command
Arguments	*`required-arg1`* or [*`optional-arg2`*]	
Function	Brief explanation of what the command is for.	
Description	Description of the behaviour and functionality of the command, if needed.	
Usage	One or more examples of the command.	Menu example
See Also	Other commands that may be useful with this one.	

Internal commands are entered in the command and response area by just typing the command, making sure to include the leading colon (`:`). They may be abbreviated to the shortest string that would still be unique. For example, `:del` is an ambiguous command because both `:deletepoint` and `:delpathdir` start with "`del`". However, `:dele` would be unique and is equivalent to typing `:deletepoint`.

Most commands allow you to enter the arguments interactively. Some, however, perform a different action when you omit the argument(s). These will be noted in the commands' descriptions.

Command	`:altsymbol`	*Schedit*
Arguments	*`FILENAME`*	
Function	Substitute selected symbol with *`FILENAME`*`.sym`	
Description	If *`FILENAME`*`.sym` is found and is a valid symbol, the last symbol of the selected set will be changed to *`FILENAME`*. Make sure to have a symbol selected before issuing this command.	
Usage	**`:altsymbol`**	`Select®Symbol to Replace`
	`:altsymbol res`	
See Also	`:get`	

Command	`:arc`	*Symed*
Arguments	[Point P1] [Point P2] [Point P3]	
Function	Draw an arc defined by points P1, P2, and P3.	
Description	The way this command works is explained in the section on the menu command `Draw®Circle/Arc`.	

Usage	**`:arc`**	`Draw®Circle/Arc`
See Also	`:frame`, `:rectangle`, `:vector`	

Command	`:areaselect`	Both
Arguments	[Point P1 Point P2]	
Function	Add to the current selected set all symbols wires, and text in the rectangular area defined by points P1 and P2.	
Description	If this command is started with one or no points placed, you will be prompted to define a rectangular area. All items fully contained in the defined area will be added to the selected set. That is, the current selected set will not be cleared.	
Usage	**`:unselect;:areaselect`**	`Select®Area`
See Also	`:select`, `:unselect`	

Command	`:block`	*Schedit*
Arguments	[Point P1], [Point P2], [*`FILENAME`*]	
Function	Make a new symbol.	
Description	This command makes a new symbol "on−the−fly" without having to go into *Symed*. The points P1 and P2 define a rectangle which will be the shape of the symbol. The symbol is saved to *`FILENAME`*. Ports can be defined for it with `:newport`.	
Usage	**`:block`**	`Hierarchy®New Symbol`
	`:block inthndlr`	
See Also	`:altsymbol`, `:get`, `:newport`, `:symbol`, `:subedit`	

Command	`:bundle`	*Schedit*
Arguments	Points	
Function	Draw a bundle.	
Description	This function is for drawing a bundle, which is a group of related wires drawn in a "bundle" to reduce clutter on the diagram. See the section on *Wires, Buses, and Bundles* in *Chapter 4: Creating Electrical Designs* for more information.	
Usage	**`:bundle`**	`Wire®Draw Bundle`
See Also	`:bus`, `:rewire`, `:wire`	

Command	`:bus`	*Schedit*
Arguments	*`NAME START_INDEX`*	
Function	Toggle automatically naming of bus signals.	
Description	This command toggles whether wires are automatically named as bus signals. *`NAME`* is the name of the bus, such as `DATA`, and *`START_INDEX`* is the starting index, such as `0`. When toggling auto−naming on, a bus name and starting index are required. When toggling it off, arguments are ignored.	
Usage	**`:bus`**	`Wire®Auto−name Bus Wires...`
	`:bus ADDR 0`	
See Also	`:bundle`, `:wire`	

Command	`:cd`	Both
Arguments	*`NEW_DIRECTORY_NAME`*	
Function	Change current working directory.	
Description	This command changes the current working directory. You may use the abbreviated tilde (~) notation.	
Usage	**`:cd ~/work`**	
See Also	`:edit`, `:path`, `:pwd`	

Command	`:clearpoints`	Both
Arguments	None	
Function	Clear all points currently placed on the design.	
Usage	**`:clearpoints`**	`Edit®Clear Points`
See Also	`:deletepoint`, `:point`	

Command	`:constrainprops`	Both
Arguments	*`RULES_FILE_NAME`*	
Function	Cause *Schedit* or *Symed* to read a property rules file.	
Description	Schedit and Symed can use a property rules file to constrain the properties of all symbols in the current design to values defined by the rules in the rules file. For more information, see the section on *Property Rules Files* in *Chapter 6: Properties*.	
Usage	**`:constrainprops`**	`Properties®Read Rules File...`
	`:constrainprops prule`	
See Also	`:property`	

Command	`:copy`	Both
Arguments	None	
Function	Copy selected set.	
Description	This command copies the selected set of symbols, wires, or text. The internal variables `cpyinst` and `wcpynames` affect how it works. If `cpyinst` is set to 0, instance properties are not copied with items in the set; if it is set to 1, instance properties are copied. If `wcpynames` is set to 0, wire names are not copied with wires in the set; if it is set to 1, they are.	

Usage	**`:set cpyinst 1;::copy`**	`Edit®Copy With...Instance Properties`
See Also	`:move`, `:set`, `:undo`	

Command	`:deletepoint`	Both
Arguments	None	
Function	Delete the last point that was placed.	
Usage	**`:deletepoint`**	
See Also	`:point`, `:clearpoints`	

Command	`:delpathdir`	Both
Arguments	*`SEARCH_PATH`*	
Function	Delete directory from data file search path.	
Description	This command removes *`SEARCH_PATH`* from the data file search path, which affects *only* the editor you are using. You may use tilde (~) notation.	
Usage	**`:set A;:prompt A "Directory to remove? "; :delpathdir $(A);:path`**	
	`Options®Search Path®Remove Directory From Search Path`	
See Also	`:cd`, `:path`	

Command	`:echo`	Both
Arguments	*`STRING`* or `$`*`VARIABLE_NAME`*	
Function	Print string or contents of *`VARIABLE_NAME`*.	
Description	This command echoes (prints in the Command and Response area) *`STRING`* or *`VARIABLE_NAME`*. To print a variable's contents, you must have the `$` preceding *`VARIABLE_NAME`*. Note that you cannot echo the semicolon (`;`) character because it indicates the end of the current command and everything after it is interpreted as another command. For more information on referencing internal variables, see the section on internal variables in *Appendix A: Customizing CapFast*.	
Usage	**`:set A Yes;:echo A is set to: $A`**	
See Also	`:prompt`, `:set`	

Command	`:edit`	Both
Arguments	*`FILENAME`*	
Function	Edit a schematic or symbol file.	
Description	When *Schedit* or *Symed* look for *`FILENAME`*, they look for it in the directories in the search path, in the order they are listed, unless a path is given with *`FILENAME`*. *Schedit* also uses the search path to find the symbols in the schematic when it's loading. You can use tilde (~) notation. If you do not specify a *`FILENAME`*, the file Open dialog box will pop up, and you can select the file interactively.	
Usage	**`:edit`**	`File®Open...`
	`:edit ~p3/wcs/examples/all`	
See Also	`:empty`, `:get`, `:read`, `:source`, `:subedit`, `:write`	

Command	`:empty`	Both

Arguments	None	
Function	Start a new schematic or symbol.	
Description	You will be prompted to save the current schematic or symbol if it has changed, then the design area will be cleared.	
Usage	**:empty**	File®New
See Also	:edit, :read, :subedit, :write	

Command	:erase	Both
Arguments	None	
Function	Erase selected set.	
Usage	**:erase**	Edit®Delete
See Also	:copy, :move, :undo	

Command	:findbyprop	*Schedit*
Arguments	*PROPERTY* or (*QUALIFIER*) or :*VALUE*	
Function	Add to the selected set *all* symbols or wires that have the property *PROPERTY* or a property with qualifier *QUALIFIER* or value *VALUE*.	
Description	This command *adds* items to the selected set. Note that *QUALIFIER* must be surrounded by parentheses ’(’ and ’)’ and *VALUE* must be preceded by a colon (:).	
Usage	**:findbyprop**	Select®Elements by Property...
	:findbyprop (ASC)	
See Also	:findlabel, :next, :node, :property, :select, :unselect	

Command	:findlabel	*Schedit*
Arguments	*INSTANCE_NAME_STRING*	
Function	Add to the selected list a symbol whose instance name starts with *INSTANCE_NAME_STRING*.	
Description	This command finds only symbols, not wires, and it works with automatically generated names such as 00#4. It finds the *first* symbol whose instance name starts with *INSTANCE_NAME_STRING*. To find additional symbols whose instance names start with the same string, use the :next command.	
Usage	**:findlabel**	Select®Symbol By Name...
	:findlabel res	
See Also	:findbyprop, :next, :node, :property, :select, :unselect	

Command	`:frame`	*Schedit*
Arguments	*`NAME`*, [Point P1], [Point P2]	
Function	Draw a text frame with label *`NAME`*.	
Description	This command draws a rectangular frame (box) defined by points P1 and P2, and labeled *`NAME`*. A frame cannot have properties.	
Usage	**`:frame Test`**	
See Also	`:block`, `:text`	

Command	`:get`	*Schedit*
Arguments	*`FILENAME`* [*`INSTANCE_NAME`*]	
Function	Get a symbol.	
Description	Symbols are searched for in the directories in the data search path, in order. If you do not specify *`INSTANCE_NAME`*, and `asksnames` is `YES`, you will be prompted for an instance name.	
Usage	**`:get led D5`**	
See Also	`:altsymbol`, `:path`, `:text`	

Command	`:import`	Both
Arguments	*`FILENAME`*	
Function	Import ports or connectors.	
Description	This command imports ports or connectors from a symbol or schematic into the current design. This is useful with the `:block` command in *Schedit*, and for creating in *Symed* a symbol that represents a schematic.	
Usage	**`:import`**	`Ports®From Schematic...`
	`:import inthndlr`	
See Also	`:block`, `:invisible`, `:newport`, `:port`, `:report`, `:symbol`	

Command	`:invisible`	*Symed*
Arguments	<[`in`], [`out`], [`un`], or [`bi`]> [*`INSTANCE_NAME`*]	
Function	Get an invisible port.	
Description	This command gets an invisible port and names it *`INSTANCE_NAME`*. An invisible port is one that is not visible in *Schedit*, and should be named so that it is implicitly connected to a node (wire). Common examples of invisible ports are Vcc and GND in Ics such as 74LS00 (look at `00`). To connect a port to a node implicitly, the symbol must have a `connect(`*`NODE_NAME`*`):`*`NODE_NAME`*`!` property. For example, to connect an invisible Vcc port to the Vcc node, the symbol must have the property `connect(Vcc):Vcc!`.	
Usage	**`:invisibleport`**	`Ports®Invisible®Bidirectional`
	`:invisibleport bi Vcc`	
See Also	`:port`, `:newport`, `:report`	

Command	`:macro`	Both
Arguments	`:`*`COMMAND_NAME COMMAND_LIST`*	

Function	Define a new command.	
Description	This command allows you to define a new command, or macro. The macro may contain internal commands or other macros in *COMMAND_LIST*. After the macro is defined, you can invoke it by typing :*COMMAND_NAME* with any arguments you define. For example, the Usage example, when put in a command file which is then sourced (`:source` *FILENAME*), creates a macro named "`:chprop`" which allows you to change a property without popping up the Properties window. To use it, type **:chprop** or **:chprop** *PROPERTY_NAME* or **:chprop** *PROPERTY_NAME*:*VALUE*. It is quite useful to define macros in the `schedit.rc` or `symed.rc` file so you don't have to re−define them each time you load *Schedit* or *Symed*.	
Usage	**:macro :chprop :prop F+S $(1)+V $(1)**	
See Also	`:newcmd`	

Command	`:mark`	Both
Arguments	None	
Function	Mark the current view.	
Description	This command saves the current view so that you can go back to it after zooming and/or panning to a different view. Currently it is not possible to mark multiple views.	
Usage	**:mark**	`View®Save`
See Also	`:viewmark`	

Command	`:mload`	Both
Arguments	*MENU_NAME*	
Function	Load main menu.	
Description	This command loads the main menu from the file *MENU_NAME*. Currently it is only possible to load the main menu (the menu bar across the top of the design area).	
Usage	**:set libchoice pcb;:mload schedit2**	`Configure®Select Parts Library®PCB Library`
See Also	`:remmenu`	

Command	`:modtext`	Both
Arguments	[0 *ORIENTATION JUSTIFICATION SCALE DISPLAY_STATUS*] or [1 *ORIENTATION*] or [2 *JUSTIFICATION*] or [3 *SCALE*] or [4 *DISPLAY_STATUS*]	
Function	Change display attributes of selected text.	
Description	This command allows you to change the display attributes of selected text. For details, see the section earlier in *Chapter 5: The Command Reference Guide* on the `Text` menu.	
Usage	**:modtext 3**	`Text®Scale`
See Also	`:movetext`, `:relabel`, `:rotate`, `:text`	

Command	`:move`	Both
Arguments	None	

Function	Move selected set.	
Usage	**:move**	
See Also	:copy, :movetext, :rotate, :xmirror, :ymirror, :undo	

Command	:movetext	Both
Arguments	None	
Function	Move selected text.	
Description	For free−standing text, this works the same as the :move command. :movetext is required to move text associated with a symbol or wire, including instance names that are displayed.	
Usage	**:movetext**	Text®Move
See Also	:modtext, :move, :rotate, :undo	

Command	:newcmd	Both
Arguments	None	
Function	Start recording a macro.	
Description	This command is like the :macro command, except that it allows you to enter commands for the macro interactively, and from menus. It is useful if you don't know the internal command equivalents of menu commands, and to interactively place offset points. Stop the macro recording with :stopcmd.	
Usage	**:newcmd**	Macro®Record...
See Also	:macro, :savemac, :showdef, :stopcmd, :unsavemac	

Command	:newport	*Schedit*
Arguments	<[in], [out], [un], or [bi]> [*INSTANCE_NAME*]	
Function	Get a port for a new symbol.	
Description	This command is used after :block to define the ports for a symbol created in *Schedit*. The first argument is the type of port: in (input), out (output), bi (bidirectional), or un (undefined). *INSTANCE_NAME* is the name to give the port.	
Usage	**:newport out**	Hierarchy®Add Port to New Symbol®Output
See Also	:block	

Command	:next	*Schedit*
Arguments	None	
Function	Find next occurrence of the string last searched for with :findlabel.	
Description	The :findlabel command finds the first symbol whose instance name starts with the string you specify. The :next command finds the next instance that starts with the same string, and adds it to the selected set.	
Usage	**:next**	Select®Next Symbol
See Also	:findbyprop, :findlabel, :node, :select, :unselect	

Command	:node	*Schedit*
Arguments	*WIRE_NAME*	

Function Find a wire (node).

Description This command finds all wires (the entire node) with name *WIRE_NAME*, and adds them to the selected set.

Usage **:node** Select®Wire by Name

:node GND

See Also :findbyprop, :findlabel, :select, :unselect

Command :offset Both

Arguments *X Y*

Function Place a point offset *X,Y* from last point.

Description This command is useful in macros to place a group of points that will always be the same relative to each other, but can be placed anywhere in the schematic. If no point is placed before entering the command, the point will be offset from the last point erased, or 0 0. Points will actually be placed subject to the constraints of the snap grid size. See :point for a discussion on this.

Usage **:offset 128 156**

See Also :clearpoints, :deletepoint, :point

Command :pan Both

Arguments <[up], [upright], [upleft], [down], [downright], [downleft], [right], [left], Point>

Function Pan in specified direction.

Description This command causes the display to pan in the direction specified, by the amount specified in the internal variable panamount, as a fraction of a windowful. If you do not specify a direction, you will be prompted to click on the point you want to be the center of the display.

Usage **:pan upright**

See Also :mark, :redraw, :rzoom, :view, :viewmark, :zoom

Command :path Both

Arguments [+][*NEW_SEARCH_PATH*][+]

Function Show or modify the search path.

Description This command modifies the search path in the following manner. If you specify a *NEW_SEARCH_PATH* that is not preceded or followed by a '+', the current data file search path will be *replaced* with *NEW_SEARCH_PATH*. If it is preceded by a '+', *NEW_SEARCH_PATH* will be added to the *end* of the current search path (as in *CURRENT_SEARCH_PATH+NEW_SEARCH_PATH*). If it is followed by a '+', *NEW_SEARCH_PATH* will be added to the *beginning* of the current search path (as in *NEW_SEARCH_PATH+CURRENT_SEARCH_PATH*). If no argument is given, the current search path will be displayed.

Data files (schematic, symbol, and menu) will be searched for in the directories in the search path, in order. So, if your search path is c:\capfast\library\linear+c:\work and you have a symbol called pnp.sym in c:\work, the CapFast tools will find pnp.sym in c:\capfast\library\linear instead of in c:\work. If the search path

	is changed to `c:\work+c:\capfast\library\linear`, `pnp.sym` will be found in `c:\work`. The default search path for every CapFast program can be set in the `cad.rc` file. For more information, see *Appendix A: Customizing CapFast*.	
Usage	**`:path`**	`Options®Search Path®Show Search Path`
	`:path \work+`	`Options®Search Path®Add Directory to Beginning`
See Also	`:delpathdir, :`	

Command	`:plot`	Both (Windows only)
Arguments	None	
Function	Plot (print) current schematic or symbol.	
Description	This command will pop up the Print dialog box, where you can modify a few print settings, print the design, or go to the Print Setup dialog box. If you would like to have better control over the printing process, see *Chapter 7: Plotting Schematic Hardcopies* for information on how to use *schplot*.	
Usage	**`:plot`**	`File®Print...`
See Also	`:setupplotter`	

Command	`:point`	Both
Arguments	*`X Y`*	
Function	Place a point at *`X`*,*`Y`*.	
Description	The actual placement of points is subject to the constraints of the snap grid size (set in the internal variable `snapgridsize`). Points are placed on the closest snap grid point. So, if your snap grid size is `16` and you try to place a point at `8 8`, either with the mouse or with the `:point` command, the point would actually be placed at `16 16`. If you try to place a point at `7 7`, the point would actually be placed at `0 0`. The same is true for the `:offset` command − the resultant point is snapped to the closest snap grid point.	
Usage	**`:point 16 0`**	
See Also	`:clearpoints`, `:deletepoint`, `:offset`	

Command	`:popup`	Both
Arguments	*`MENU_NAME`*	
Function	Popup a menu.	
Description	This command pops up the menu contained in the file *`MENU_NAME`*`.mnu`, at the cursor (mouse pointer). The menu is searched for in the data search path. For information on menus, see *Appendix A: Customizing CapFast*.	
Usage	**`:popup nampromp`**	`Options®Name Prompting`
See Also	`:mload`, `:remmenu`	

Command	`:port`	*Symed*
Arguments	<[`in`], [`out`], [`bi`], or [`un`]> [*`INSTANCE NAME`*]	
Function	Get a visible port.	
Usage	**`:port in`**	`Ports®Visible®Input`

	:port bi A1	
See Also	`:invisible`, `:newport`, `:report`	

Command	`:prompt`	Both
Arguments	*`VARIABLE`* "*`PROMPT`*"	
Function	Prompt user for argument.	
Description	This command prompts the user with the string *`PROMPT`* and sets the variable *`VARIABLE`* to the string that the user enters. *`PROMPT`* must be in double or single quotes if it is more than one word, and *must not* contain a semicolon (`;`). If *`PROMPT`* is omitted, the editor will display the valid values or range (for internal variables), the current value, and a prompt for the new value. *`VARIABLE`* may be a user−defined variable or an internal variable that is modifiable. For more information, see *Appendix A: Customizing CapFast*.	
Usage	**:prompt append** Options®Default Naming Scheme®Append Character(s)	
See Also	`:echo`, `:set`	

Command	`:property`	Both
Arguments	[A *NAME*:*VALUE*]+[C]+[D [*VALUE*]]+[F]+[H *HEIGHT*]+[I]+[J *JUSTIFICATION*]+[M]+[O *ORIENTATION*]+[R]+[S *NAME*]+[V *NAME*:*VALUE*]	
Function	Edit properties (add, copy, change display, free, change scale, select instance, change justification, move, change orientation, remove, toggle selection, change value).	
Description	This command allows you to manipulate properties without using the Properties window. If no argument is specified, the Properties window will pop up. If more than one sub−command (A, C, D, F, H, I, J, M, O, R, S, V, which must all be upper−case) is given as an argument, they must be separated by a '+'. The sub−commands may be given in any order, and more than once per `:property` command. The sub−commands will be executed in the order they are entered. If a sub−command is expecting a value and you don't give it one, it will prompt you for the value. These things make this command quite useful in macros (see the `:macro` example).	

A *NAME*:*VALUE*	Add property *NAME* with value *VALUE*. If that property already exists, its value will be changed to *VALUE*.
C	Copy selected properties to another symbol or wire. You will be prompted to select an item to copy them to unless you place points prior to invoking the command.
D [*VALUE*]	Cycle through selected properties' display characteristics or set to *VALUE* (OFF, ON, VALUE).
F	Free (unselect) all selected properties.
H *HEIGHT*	Change scale of selected properties' text to *HEIGHT* (a percentage of default height)
I	Select all local properties associated with this symbol instance.
J *JUSTIFICATION*	Change selected properties' text justification ([bottom | base | center | cap | top] [left | center | right]). See the section earlier in

		this chapter on the Properties window.
	`M`	Move selected properties.
	`O` *`ORIENTATION`*	Change selected properties' text orientation ([normal | up | over | down]). See the section earlier in this chapter on the Properties window.
	`P`	Protect selected properties (*Symed* only). Protected properties cannot be changed or deleted in *Schedit*.
	`R`	Erase selected properties.
	`S` *`NAME`*	Toggle selection of property *`NAME`*. To be sure that a property is selected, you need to free all first, then toggle selection on the one(s) you want selected.
	`V` *`NAME`*`:`*`VALUE`*	Change value of property *`NAME`* to *`VALUE`*.
Usage	**`:prop F+S ref+S value+V ref:R1+V value:10K+D VALUE`**	
See Also	`:`	

Command	`:put`	Both
Arguments	[*`FILE_NAME`*]	
Function	Save design as *`FILE_NAME`* without changing `fullname`.	
Description	This command functions the same as the `:write` command, except that it does not modify the internal variable `fullname`.	
Usage	**`:put testsch`**	
See Also	`:write`	

Command	`:pwd`	Both
Arguments	None	
Function	Print current working directory.	
Usage	**`:pwd`**	`Options®Search Path®Print Current Directory`
See Also	`:cd`, `:delpathdir`, `:path`, `:writepath`	

Command	`:qcheck`	*Schedit*
Arguments	None	
Function	Do batch ERC.	
Description	This command first asks you whether you want to do a full ERC on the schematic or delete the ERC properties. ERC properties are properties added by the ERC (Electrical Rules Checker). Enter **`e`** to do an ERC, or **`d`** to delete the ERC properties. For more information on the Electrical Rules Checker, see the `QC/ERC` entry earlier in this chapter in the `File` menu section.	
Usage	**`:qcheck`**	`File®QC/ERC`
See Also	`:property`	

Command	`:quit`	Both
Arguments	None	
Function	Quit editor.	

Description	If the internal variable `exitwoprompt` is `NO`, you will be prompted to save the design if it has changed. If `exitwoprompt` is `YES`, the editor will quit without asking you to save changes.	
Usage	**`:quit`**	`File®Exit`
See Also		

Command	`:read`	Both
Arguments	*`FILENAME`*	
Function	Add schematic or symbol to current design.	
Description	This command allows you to merge another schematic or symbol with the current schematic or symbol.	
Usage	**`:read`**	`File®Merge`
See Also	`:edit`, `:get`, `:save`	

Command	`:rectangle`	*Symed*
Arguments	Point P1, Point P2	
Function	Draw a rectangle.	
Description	If you do not place both points before invoking the command, you will be put in rubber band mode after placing the first point.	
Usage	**`:rectangle`**	`Draw®Rectangle`
See Also	`:arc`, `:block`, `:frame`, `:vector`	

Command	`:redraw`	Both
Arguments	None	
Function	Redraw screen.	
Usage	**`:redraw`**	`View®Redraw`
See Also	`:redrawarea`	

Command	`:redrawarea`	Both
Arguments	Point P1, Point P2	
Function	Redraw an area.	
Description	This command is useful in large, schematics that take a long time to redraw, particularly when running CapFast across a network such as on an X terminal. Only the area you specify is redrawn.	
Usage	**`:redrawarea`**	`View®Redraw Area`
See Also	`:redraw`	

Command	`:relabel`	Both
Arguments	*`STRING`*	
Function	Relabel currently selected text or item name.	
Description	This command allows you to change the currently selected text or label. If the text is associated with a symbol or wire, the text is its label or name. Note that the text may have been automatically generated as an instance name, such as `res#4`, and may or may not be displayed.	

Usage	**`:relabel`**	`Text®Relabel`
See Also	`:modtext`, `:movetext`, `:text`	

Command	`:remmenu`	Both
Arguments	None	
Function	Remove the main menu.	
Description	This command removes the main menu bar across the top of the design area. This is useful if you need every bit of screen space and have keys assigned to macros to to the commands you need. To gain additional space at the bottom (in the command and response area), start *Schedit* or *Symed* with the `−w`*`NUM`* option, where *`NUM`* is the number of lines you want in the command and response area.	
Usage	**`:remmenu`**	
See Also	`:mload`, `:popup`	

Command	`:report`	*Symed*
Arguments	<[`in`], [`out`], [`bi`], or [`un`]> [*`INSTANCE NAME`*]	
Function	Change port function.	
Description	This command allows you to change the function and/or name of the ports selected. If more than one port is selected, both will be changed to the type you choose, but you can change the names independently.	
Usage	**`:report`**	`Ports®Change Definition`
	`:report in IN1`	
See Also	`:import`, `:invisible`, `:newport`, `:port`	

Command	`:return`	*Schedit*
Arguments	None	
Function	Ascend hierarchy (Return from subedit).	
Description	This command ascends the design hierarchy in a hierarchical design.	
Usage	**`:return`**	`File®Hierarchy®Ascend`
See Also	`:quit`, `:subedit`	

Command	`:rewire`	*Schedit*
Arguments	None	
Function	Reroute wire, leaving ends intact.	
Description	This command allows you to reroute the selected wire(s). It leaves the endpoints intact if you select one of the non−endpoint vertices as the vertex to be moved. You can also move the endpoints if you select them as the vertex to be moved.	
Usage	**`:rewire`**	`Wire®Re−Route`
See Also	`:bundle`, `:bus`, `:wire`	

Command	`:rotate`	Both
Arguments	*`DEGREES`*, Point	

Function	Rotate selected set in multiples of 90 degrees.	
Description	This command rotates the selected set `DEGREES` degrees *counter−clockwise*, rounded off to the nearest multiple of 90 degrees. The rotation is done about the point you specify, or the "Galbi point" of the last symbol selected in the selected list. Labels and other text are rotated according to the value of the internal variable `textrotation` (see `Options®Text Options®Link Text Rotation to Symbol`).	
Usage	**`:rotate 90`**	`Edit®Rotate`
See Also	`:modtext`, `:move`, `:movetext`, `:undo`, `:xmirror`, `:ymirror`	

Command	`:rzoom`	Both
Arguments	*`AMOUNT`*, `[C]`	
Function	Zoom relatively.	
Description	This command zooms the display so that it is *`AMOUNT`* times the current size. So, if *`AMOUNT`* is `1`, the display does not zoom, if it is greater than `1`, it zooms in, and if it is less than `1` it zooms out. If the constant '`C`' is given as an argument, the display zooms and centers about the given point, or wherever the mouse pointer is when the command is issued.	
Usage	**`:rzoom $zoomin C`**	`View®Zoom In Under Cursor`
	`:rzoom $zoomout`	`View®Zoom Out`
See Also	`:mark`, `:pan`, `:view`, `:viewmark`, `:zoom`	

Command	`:save`	Both
Arguments	*`FILENAME`*	
Function	Save selected set to a file.	
Description	If *`FILENAME`* is not specified, the Save As dialog box will appear, allowing you to choose the directory and file to save the selected set in.	
Usage	**`:save partsch`**	
See Also	`:edit`, `:read`, `:write`	

Command	`:savemac`	Both
Arguments	None	
Function	Save a macro defined with `:newcmd`.	
Description	This command allows you to save a macro that you defined with `:newcmd` (`Macro®Record...`). The macro is saved to the `schedit.rc` or `symed.rc` file in the directory defined by the `controldir` internal variable. You choose which macro to save, after you invoke the command, by pressing the key the macro was assigned to.	
Usage	**`:savemac`**	
See Also	`:macro`, `:newcmd`, `:stopcmd`, `:unsavemac`	

Command	`:select`	Both
Arguments	Points	
Function	Select items.	

Description This command allows you to select items. If you place points before you invoke the command, the editor will select the items that the points refer to. If no points are placed before invoking the command, you will be placed in "Flashlight Mode" and the items will turn green as you move the mouse pointer over them. If you click the left mouse button when an item is green, it will become selected and turn red.

Usage **`:select`** `Select®Items`

See Also `:areaselect`, `:findbyprop`, `:findlabel`, `:next`, `:node`, `:unselect`

Command `:set` Both

Arguments *`VARIABLE VALUE`*

Function Set variable.

Description This command allows you to set user variables and modifiable internal variables. User variables are any valid variable name you choose. Internal variables are ones that have special meaning to the editors. Some are read−only, such as `uniqueid` and `version`. Most of them you can modify to control certain functions of the editors. You can use your own variables to control how your macros work, or to store information. For more information, see *Appendix A: Customizing CapFast*.

Usage **`:set autotime 5`**

See Also `:echo`, `:prompt`, `:toggle`, `writerc`

Command `:setupplotter` Both, Windows only

Arguments None

Function Set up plotter or printer.

Usage **`:setupplotter`** `File®Print Setup...`

See Also `:plot`

Command `:showdef` Both

Arguments *`COMMAND_NAME`*

Function Show macro definition.

Description This command shows the list of commands contained in the macro definition of *`COMMAND_NAME`*. This can be a macro defined with the `:macro` command, or with the `:newcmd` command. If it is defined with the `:newcmd` command, specify the command by the key it was assigned to − **`:Func1`** − **`:Func10`** for `<F1>` − `<F10>`, **`:AltA`** − **`:AltZ`** for `<Alt><Shift><A>` − `<Alt><Shift><Z>`, and **`:Alta`** − **`:Altz`** for `<Alt><A>` − `<Alt><Z>`.

Usage **`:showdef`** `Macro®View Definition...`

`:showdef :chprop`

See Also `:macro`, `:newcmd`, `:savemac`, `:stopcmd`, `:unsavemac`

Command `:source` Both

Arguments *`FILENAME`*

Function Execute command file.

Description	This command executes the commands in *`FILENAME`*. This is done automatically with `schedit.rc` or `symed.rc` when you start the editors, but can be done at any other time, and with other command files. They may contain any internal commands.	
Usage	**`:source`**	`File®Execute...`
See Also	`:macro`, `:newcmd`	

Command	`:stopcmd`	Both
Arguments	None	
Function	Stop recording macro started with `:newcmd`.	
Usage	**`:stopcmd`**	`Macro®Stop`
See Also	`:macro`, `:newcmd`, `:savemac`, `:showdef`, `:unsavemac`	

Command	`:subedit`	*Schedit*
Arguments	*`FILENAME`*	
Function	Descend hierarchy (Subedit symbol).	
Description	If no filename is specified, *Schedit* will attempt to subedit the last symbol selected. If there is no schematic with the same name as the symbol, it will bring up an empty design.	
Usage	**`:subedit`**	`File®Hierarchy®Descend`
See Also	`:block`, `:return`, `:symbol`	

Command	`:symbol`	*Schedit*
Arguments	*`FILENAME`*, Point P1, Point P2	
Function	Create block from schematic.	
Description	This command creates a symbol file containing the rectangle defined by points P1 and P2.	
Usage	**`:symbol newsym`**	
See Also	`:block`	

Command	`:symbolmenu`	Both
Arguments	None	
Function	Pops up a menu associated with selected symbol.	
Description	This command pops up a menu for the selected symbol. It searches for `s`*`SYMBOL_NAME`*`.mnu` and `sdefault.mnu`.	
Usage	**`:symbolmenu`**	
See Also	`:mload`	

Command	`:text`	Both
Arguments	*`STRING`*	
Function	Add text.	
Description	This command adds text to the design. The text is free, or unassociated with any symbol or wire. This is used to make comments on the design or label areas of the design.	

Usage	**`:text`**	`Text®Add`
See Also	`:modtext`, `:movetext`, `:relabel`	

Command	`:toggle`	Both
Arguments	*`INTERNAL_VARIABLE`*	
Function	Toggle value of internal variable.	
Description	This command actually cycles through the possible values for *`INTERNAL_VARIABLE`*. If the variable only has `YES` and `NO` values, it appears to toggle.	
Usage	**`:toggle controldir`**	`Options®Control File Location`
See Also	`:echo`, `:prompt`, `:set`, `:writerc`	

Command	`:transferfocus`	Both
Arguments	*`WINDOW_NAME`*	
Function	Transfer focus to named window.	
Description	This command transfers focus to the named window. So, if you have *Symed* open, and you are in *Schedit*, you can transfer focus to *Symed* with this command.	
Usage	**`:transferfocus Symed`**	
See Also	:	

Command	`:undo`	Both
Arguments		
Function	Undo last command.	
Description	This command undoes the last command entered. It does not undo view changes.	
Usage	**`:undo`**	`Edit®Undo`
See Also	:	

Command	`:unsavemac`	Both
Arguments	*`MACRO_NAME`*	
Function	Removes a macro from a `.rc` file.	
Description	This command removes from a `schedit.rc` or `symed.rc` file a macro that was saved there with `:savemac`.	
Usage	**`:unsavemac`**	`Macro®Unsave...`
See Also	`:savemac`	

Command	`:unselect`	Both
Arguments	None	
Function	Clear selected set.	
Usage	**`:unselect`**	`Select®Clear`
See Also	`:areaselect`, `:findbyprop`, `:findlabel`, `:next`, `:select`	

Command	`:vector`	*Symed*

Arguments	Point P1, Point P2	
Function	Draw a vector.	
Description	This function draws a vector, or line, with the current linestyle. The linestyle is defined in the internal variable `brush`, which you can change through the `Draw®Linestyle` menu.	
Usage	**`:vector`**	`Draw®Vector`
See Also	`:arc`, `:rectangle`	

Command	`:view`	Both
Arguments	Point P1, Point P2	
Function	View an area.	
Description	See `View®Zoom Area` for more information.	
Usage	**`:view`**	`View®Zoom Area`
See Also	`:mark`, `:pan`, `:rzoom`, `:viewmark`, `:zoom`	

Command	`:viewmark`	Both
Arguments	None	
Function	Return to a marked view.	
Usage	**`:viewmark`**	`View®Restore`
See Also	`:mark`, `:pan`, `:rzoom`, `:view`, `:zoom`	

Command	`:wire`	*Schedit*
Arguments	[*`INSTANCE_NAME`*], Points	
Function	Draw a wire.	
Description	This command draws a wire defined by points. If less than two points are defined, you will be placed in "Rubber Band Mode", and the entry of points will end when you press the `DONE` button.	
Usage	**`:wire`**	`Wire®Draw Wire`
	`:wire Vin`	
See Also	:	

Command	`:write`	Both
Arguments	*`FILENAME`*	
Function	Write a schematic or symbol file.	
Description	This command writes out the current design to the file *`FILENAME`*. If *`FILENAME`* is not specified, it pops up the `Save As` dialog.	
Usage	**`:write $fullname`**	`File®Save`
See Also	`:edit`, `:read`, `:save`	

Command	`:writerc`	Both
Arguments	*`VARIABLE VALUE`*	
Function	Save variable in `.rc` file.	

Description	This command saves the variable *VARIABLE* (internal or user−defined) with value *VALUE* in the `schedit.rc` or `symed.rc` file in the directory defined by the internal variable `controldir`.	
Usage	**`:prompt autotime;:writerc autotime $autotime`**	Options®Auto−backup Interval
See Also	`:prompt`, `:set`, `:toggle`	

Command	`:xmirror`	Both
Arguments	[Point P1]	
Function	Xmirror a selected set.	
Description	This command mirrors the selected set about a horizontal line passing through a point you specify prior to invoking the command. If no point is specified, the mirror is done about the horizontal line passing through the "Galbi point" of the last symbol selected in the selected list. Labels and other text are not rotated, but their relative positions are moved with the other parts.	
Usage	**`:xmirror`**	Edit®Flip X
See Also	`:move`, `:rotate`, `:ymirror`	

Command	`:ymirror`	Both
Arguments	[Point P1]	
Function	Ymirror a selected set.	
Description	This command mirrors the selected set about a vertical line passing through a point you specify prior to invoking the command. If no point is specified, the mirror is done about the vertical line passing through the "Galbi point" of the last symbol selected in the selected list. Labels and other text are not rotated, but their relative positions are moved with the other parts.	
Usage	**`:ymirror`**	Edit®Flip Y
See Also	`:move`, `:rotate`, `:xmirror`	

Command	`:zoom`	Both
Arguments	*AMOUNT*	
Function	Zoom absolutely.	
Description	This command zooms so that the display is *AMOUNT* times the size of the entire design page. So, if *AMOUNT* is less than `1`, the design appears to get smaller (zoom out); if greater than `1`, the design appears to get larger (zoom in).	
Usage	**`:zoom .98`**	View® View All
See Also	`:mark`, `:rzoom`, `:viewmark`	