NAME

magic - VLSI layout editor

SYNOPSIS

magic [-T technology] [-d device_type] [-g graphics_port] [-m monitor_type] [-D] [file]

DESCRIPTION

Magic is an interactive editor for VLSI layouts that runs under all variants of UNIX, including Mac OS-X and Cygwin. This man page is a reference manual; if you are a first-time user, you should use the Magic tutorials to get acquainted with the system (see the online resources links below).

Magic uses two windows: one for text and a separate window for displaying layouts. Magic runs under the window system X11 (use under Cygwin requires the presence of an X11 server in Windows; the server that comes packaged with Cygwin works well). The command line switch "-d" can be used to tell Magic which kind of display you are running. Specifically, this is useful to tell magic to use OpenGL graphics instead of plain X11 ("-d OGL"), or to use 24 bit plane graphics instead of 8 bit planes ("-d 24BITS") if both are available on a video card with overlay support.

Here are the options accepted by Magic:

-T The next argument is the name of a technology. The tile types, display information, and design rules for this technology are read by Magic from a technology file when it starts up. The technology defaults to ``scmos''.

-d The next argument is the type of workstation or graphics display being used. Magic supports these types:

NULL A null device for running Magic without using a graphics display, such as a batch job. Note that the special case "-dnull" (lowercase, no space) has a more streamlined startup specifically for batch processing.

X11 X-windows, version 11. The is the preferred interface. Magic acts as a client to the X window server and interfaces to all graphics terminals supported by the X server.

OGL OpenGL graphics running under X11. This is the preferred interface on systems having hardware-accelerated 3D graphics video cards and drivers.

Addition information on Magic's X11 driver, including options for .Xdefaults files, may be found in ``Magic Maintainer's Manual #4: Using Magic Under X Windows''.

XWIND Simply another name for the X11 driver.

If no device is specified, Magic tries to guess which driver is appropriate (by checking the environment variables and by poking around in /dev).

When Magic starts up it looks for a command file in ${CAD_ROOT}/magic/sys/.magicrc and processes it if it exists. Then Magic looks for a file with the name ``.magicrc'' in the home directory and processes it if it exists. Finally, Magic looks for a .magicrc file in the current directory and reads it as a command file if it exists. The .magicrc file format is described under the source command. If magic is compiled with Tcl/Tk support, then any magic or Tcl/Tk commands may be used inside the startup file. The startup file name was changed to ".magicrc" to avoid conflicts with a common system file named ".magic". However, the name ".magic" will be searched after ".magicrc" for backward compatibility.

COMMANDS -- GENERAL INFORMATION

Magics uses types of commands: Key macros and long commands. The first form consists of single-key (or button) abbreviations called ``macros''; macros are invoked by pressing a single key or mouse button. Certain macros are predefined in the systemwide ${CAD_ROOT}/magic/sys/.magicrc file, but you can override them and add your own macros using the macro command (described below under COMMANDS FOR ALL WINDOWS). The special macro "." is reserved to mean "execute the last typed long command".

You can enter long commands in the terminal console at the console prompt, or from the layout window by typing a : or ; key, which are the two other reserved macros meaning "switch keyboard focus to the console window". After typing the : or ; key, type the text of the command, which will be written to the terminal window. Multiple commands may be specified on one line by separating them with semicolons.

Most commands deal with the window underneath the cursor, so if a command doesn't do what you expect make sure that you are pointing to the correct place on the screen. There are several different kinds of windows in Magic (layout, color, and netlist); each window has a different command set, described in a separate section below.

MOUSE BUTTONS FOR LAYOUT WINDOWS

Magic uses a three button mouse. The buttons are interpreted in a way that depends on the current tool, as indicated by the shape of the cursor (see the tool command). The various tools are described below. The initial tool is box. These interpretations apply only when mouse buttons are pressed in the interior of a layout window.

Box Tool

This is the default tool, and is indicated by a crosshair cursor. It is used to position the box and to paint and erase:

left This button is used to move the box by one of its corners. Normally, this button picks up the box by its lower-left corner. To pick the box up by a different corner, click the right button while the left button is down. This switches the pick-up point to the corner nearest the cursor. When the button is released, the box is moved to position the corner at the cursor location. If the box has been set to snap to the window's grid (see the :snap command), then the box corner is left aligned with the grid that the user has chosen for the window with the :grid command, even if that grid is not visible.

right Change the size of the box by moving one corner. Normally this button moves the upper-right corner of the box. To move a different corner, click the left button while the right button is down. This switches the corner to the one nearest the cursor. When you release the button, three corners of the box move in order to place the selected corner at the cursor location (the corner opposite the one you picked up remains fixed). Snapping to the window's grid is handled as for the left button.

middle (bottom)
Used to paint or erase. If the crosshair is over paint, then the area of the box is painted with the layer(s) underneath the crosshair. If the crosshair is over white space, then the area of the box is erased.

Wiring Tool

The wiring tool, indicated by an arrow cursor, is used to provide an efficient interface to the wiring commands:

left Same as the long command wire type.

right Same as the long command wire leg.

middle (bottom)
Same as the long command wire switch.

Netlist Tool

This tool is used to edit netlists interactively. It is indicated by a thick box cursor.

left   Select the net associated with the terminal nearest the cursor.

right Find the terminal nearest the cursor, and toggle it into the current net (if it wasn't already in the current net) or out of the current net (if it was previously in the net).

middle (bottom)
Find the terminal nearest the cursor, and join its net with the current net.

Rsim Tool
Used when running the IRSIM simulator under Magic. A pointing hand is used as the cursor.

left   Moves the box just like the box tool.

right Moves the box just like the box tool.

middle (bottom)
Displays the Rsim node values of the selected paint.

LONG COMMANDS FOR LAYOUT WINDOWS

These commands work if you are pointing to the interior of a layout window. Commands are invoked by typing a colon (``:'') or semi-colon (``;''), followed by a line containing a command name and zero or more parameters. In addition, macros may be used to invoke commands with single keystrokes. Useful default macros are set in the global .magicrc file (in ${CAD_ROOT}/magic/sys, normally /usr/local/lib/magic/sys). You can list all current macros with the macro command, described under ``LONG COMMANDS FOR ALL WINDOWS''. Unique abbreviations are acceptable for all keywords in commands. The commands are:

addpath searchpath
Add more directories to the end of Magic's cell search path. See the documentation for the path command for an explanation of the search path.

array xsize ysize
Make many copies of the selection. There will be xsize instances in the x-direction and ysize instances in the y-direction. Paint and labels are arrayed by copying them. Subcells are not copied, but instead each instance is turned into an array instance with elements numbered from 0 to xsize-1 in the x-direction, and from 0 to ysize-1 in the y-direction. The spacing between elements of the array is determined by the box x- and y-dimensions.

array xlo ylo xhi yhi
Identical to the form of array above, except that the elements of arrayed cells are numbered left-to-right from xlo to xhi and bottom-to-top from ylo to yhi. It is legal for xlo to be greater than xhi, and also for ylo to be greater than yhi.

box [args]
Used to change the size of the box or to find out its size. There are several sorts of arguments that may be given to this command:

(No arguments.)
Show the box size and its location in the edit cell, or root cell of its window if the edit cell isn't in that window.

direction [distance]
Move the box distance units in direction, which may be one of left, right, up, or down. Distance defaults to the width of the box if direction is right or left, and to the height of the box if direction is up or down.

width [size]

height [size]

Set the box to the width or height indicated. If size is not specified the width or height is reported.

x1 y1 x2 y2

Move the box to the coordinates specified (these are in edit cell coordinates if the edit cell is in the window under the cursor; otherwise these are in the root coordinates of the window). x1 and y1 are the coordinates of the lower left corner of the box, while x2 and y2 are the upper right corner. The coordinates must all be integers.

calma [option] [args]

This command is used to read and write files in Calma GDS II Stream format (version 3.0, corresponding to GDS II Release 5.1). This format is like CIF, in that it describes physical mask layers instead of Magic layers. In fact, the technology file specifies a correspondence between CIF and Calma layers. The current CIF output style (see cif ostyle) controls how Calma stream layers are generated from Magic layers. If no arguments are given, the calma command generates a Calma stream file for the layout in the window beneath the cursor in file.strm, where file is the name of the root cell. This stream file describes the entire cell hierarchy in the window. The name of the library is the same as the name of the root cell. Option and args may be used to invoke one of several additional operations:

calma flatten

Ordinarily, Magic arrays are output using the Calma ARRAY construct. After a calma flatten command, though, arrays will be output instead as a collection of individual cell uses, as occurs when writing CIF.

calma help

Print a short synopsis of all of the calma command options.

calma labels

Output labels whenever writing a Calma output file.

calma lower

Allow both upper and lower case to be output for label text. This is the default behavior; calma nolower causes lower case to be converted to upper case on output.

calma noflatten

Undoes the effect of a prior :calma flatten command, re-enabling the output of Magic arrays using the Calma ARRAY construct.

calma nolabels

Don't output labels when writing a Calma output file.

calma nolower

Convert lower to upper case when outputting labels.

calma read file

The file file.strm is read in Calma format and converted to a collection of Magic cells. The current CIF input style determines how the Calma layers are converted to Magic layers. The new cells are marked for design-rule checking. Calma format doesn't identify the root of the collection of cells read in, so none of the cells read will appear on the display; use load to make them visible. If the Calma file had been produced by Magic, then the name of the root cell is the same as the library name printed by the :calma read command.

calma write fileName

Writes a stream file just as if no arguments had been entered, except that the output is written into fileName.strm instead of using the root cell name for the file name.

channels

This command will run just the channel decomposition part of the Magic router, deriving channels for the area under the box. The channels will be displayed as outlined feedback areas over the edit cell.

cif [option] [args]

Read or write files in Caltech Intermediate Form (CIF). If no arguments are given, this command generates a CIF file for the window beneath the cursor in file.cif, where file is the name of the root cell. The CIF file describes the entire cell hierarchy in the window. Option and args may be used to invoke one of several additional CIF operations:

cif arealabels [yes | no]

Enables/disables the cif area-label extension. If enabled, area labels are written via the 95 cif extension. If disabled, labels are collapsed to points when writing cif and the 94 cif construct is used. Area-labels are disabled by default (many programs don't understand cif area-labels).

cif help

Print a short synopsis of all of the cif command options.

cif istyle [style]

Select the style to be used for CIF input. If no style argument is provided, then Magic prints the names of all CIF input styles defined in the technology file and identifies the current style. If style is provided, it is made the current style.

cif ostyle [style]

Select the style to be used for CIF output. If no style argument is provided, then Magic prints the names of all CIF output styles defined in the technology file and identifies the current style. If style is provided, it is made the current style.

cif read file

The file file.cif is read in CIF format and converted to a collection of Magic cells. The current input style determines how the CIF layers are converted to Magic layers. The new cells are marked for design-rule checking. Any information in the top-level CIF cell is copied into the edit cell. Note: this command is not undo-able (it would waste too much space and time to save information for undoing).

cif see layer

In this command layer must be the CIF name for a layer in the current output style. Magic will display on the screen all the CIF for that layer that falls under the box, using stippled feedback areas. It's a bad idea to look at CIF over a large area, since this command requires the area under the box to be flattened and therefore is slow.

cif statistics

Prints out statistics gathered by the CIF generator as it operates. This is probably not useful to anyone except system maintainers.

cif write fileName

Writes out CIF just as if no arguments had been entered, except that the CIF is written into fileName.cif instead of using the root cell name for the file name. The current output style controls how CIF layers are generated from Magic layers.

cif flat fileName

Writes out CIF as in the cif write command, but flattens the design first (e.g. creates an internal version with the cell hierarchy removed). This is useful if one wishes to use the and-not feature of the CIF output styles, but is having problems with interactions of overlapping cells.

clockwise [degrees]
Rotate the selection by 90, 180 or 270 degrees. After the rotation, the lower-left corner of the selection's bounding box will be in the same place as the lower-left corner of the bounding box before the rotation. Degrees defaults to 90. If the box is in the same window as the selection, it is rotated too. Only material in the edit cell is affected.

copy [direction [amount]]

copy to x y
If no arguments are given, a copy of the selection is picking up at the point lying underneath the box lower-left corner, and placed so that this point lies at the cursor position. If direction is given, it must be a Manhattan direction (e.g. north, see the ``DIRECTIONS'' section below). The copy of the selection is moved in that direction by amount. If the box is in the same window as the selection, it is moved too. Amount defaults to 1. The second form of the command behaves as though the cursor were pointing to (x, y) in the edit cell; a copy of the selection is picked up by the point beneath the lower-left corner of the box and placed so that this point lies at (x, y).

corner direction1 direction2 [layers]
This command is similar to fill, except that it generates L-shaped wires that travel across the box first in direction1 and then in direction2. For example, corner north east finds all paint under the bottom edge of the box and extends it up to the top of the box and then across to the right side of the box, generating neat corners at the top of the box. The box should be at least as tall as it is wide for this command to work correctly. Direction1 and direction2 must be Manhattan directions (see the section DIRECTIONS below) and must be orthogonal to each other. If layers is specified then only those layers are used; otherwise all layers are considered.

delete Delete all the information in the current selection that is in the edit cell. When cells are deleted, only the selected use(s) of the cell is (are) deleted: other uses of the cell remain intact, as does the disk file containing the cell. Selected material outside the edit cell is not deleted.

drc option [args]
This command is used to interact with the design rule checker. Option and args (if needed) are used to invoke a drc command in one of the following ways:

drc catchup
Let the checker process all the areas that need rechecking. This command will not return until design-rule checking is complete or an interrupt is typed. The checker will run even if the background checker has been disabled with drc off.

drc check
Mark the area under the box for rechecking in all cells that intersect the box. The recheck will occur in background after the command completes. This command is not normally necessary, since Magic automatically remembers which areas need to be rechecked. It should only be needed if the design rules are changed.

drc count
Print the number of errors in each cell under the box. Cells with no errors are skipped.

drc find [nth]
Place the box over the nth error area in the selected cell or edit cell, and print out information about the error just as if drc why had been typed. If nth isn't given (or is less than 1), the command moves to the next error area. Successive invocations of drc find cycle through all the error tiles in the cell. If multiple cells are selected, this command uses the upper-leftmost one. If no cells are selected, this command uses the edit cell.

drc help

Print a short synopsis of all the drc command options.

drc off
Turn off the background checker. From now on, Magic will not recheck design rules immediately after each command (but it will record the areas that need to be rechecked; the command drc on can be used to restart the checker).

drc on Turn on the background checker. The checker will check whatever modifications have not already been checked. From now on, the checker will reverify modified areas as they result from commands. The checker is run in the background, not synchronously with commands, so it may get temporarily behind if massive changes are made.

drc printrules [file]
Print out the compiled rule set in file, or on the text terminal if file isn't given. For system maintenance only.

drc rulestats
Print out summary statistics about the compiled rule set. This is primarily for use in writing technology files.

drc statistics
Print out statistics kept by the design-rule checker. For each statistic, two values are printed: the count since the last time drc statistics was invoked, and the total count in this editing session. This command is intended primarily for system maintenance purposes.

drc why
Recheck the area underneath the box and print out the reason for each violation found. Since this command causes a recheck, the box should normally be placed around a small area (such as an error area).

dump cellName [child refPointC] [parent refPointP]
Copy the contents of cell cellName into the edit cell so that refPointC in the child is positioned at point refPointP in the edit cell. The reference points can either be the name of a label, in which case the lower-left corner of the label's box is used as the reference point, or as a pair of numbers giving the (x, y) coordinates of a point explicitly. If refPointC is not specified, the lower-left corner of cellName cell is used. If refPointP is not specified, the lower-left corner of the box tool is used (the box must be in a window on the edit cell). After this command completes, the new information is selected.

edit Make the selected cell the edit cell, and edit it in context. The edit cell is normally displayed in brighter colors than other cells (see the see command to change this). If more than one cell is selected, or if the selected cell is an array, the cursor position is used to select one of those cells as the new edit cell. Generally, Magic commands modify only the current edit cell.

erase [layers]
For the area enclosed by the box, erase all paint in layers. (See the ``LAYERS'' section for the syntax of layer lists). If layers is omitted it defaults to *,labels. See your technology manual, or use the layers command, to find out about the available layer names.

expand [toggle]
If the keyword toggle is supplied, all of the selected cells that are unexpanded are expanded, and all of the selected cells that are expanded are unexpanded. If toggle isn't specified, then all of the cells underneath the box are expanded recursively until there is nothing but paint under the box.

extract option [args]
Extract a layout, producing one or more hierarchical .ext files that describe the electrical circuit implemented by the layout. The current extraction style (see extract style below) determines the pa-

rameters for parasitic resistance, capacitance, etc. that will be used. The extract command with no parameters checks timestamps and re-extracts as needed to bring all .ext files up-to-date for the cell in the window beneath the crosshair, and all cells beneath it. Magic displays any errors encountered during circuit extraction using stippled feedback areas over the area of the error, along with a message describing the type of error. Option and args are used in the following ways:

extract all

All cells in the window beneath the cursor are re-extracted regardless of whether they have changed since last being extracted.

extract cell name

Extract only the currently selected cell, placing the output in the file name. If more than one cell is selected, this command uses the upper-leftmost one.

extract do [option]

extract no option

Enable or disable various options governing how the extractor will work. Use :extract do with no arguments to print a list of available options and their current settings. When the adjust option is enabled, the extractor will compute compensating capacitance and resistance whenever cells overlap or abut; if disabled, the extractor will not compute these adjustments but will run faster. If capacitance is enabled, node capacitances to substrate (perimeter and area) are computed; otherwise, all node capacitances are set to zero. Similarly, resistance governs whether or not node resistances are computed. The coupling option controls whether coupling capacitances are computed or not; if disabled, flat extraction is significantly faster than if coupling capacitance computation is enabled. Finally, the length option determines whether or not pathlengths in the root cell are computed (see extract length below).

extract help

Prints a short synopsis of all the extract command options.

extract length [option args]

Provides several options for controlling which point-to-point path lengths are extracted explicitly. The extractor maintains two internal tables, one of drivers, or places where a signal is generated, and one of receivers, or places where a signal is sent. The components of each table are hierarchical label names, defined by means of the two commands extract length driver name1 [name2 ...] and extract length receiver name1 [name2 ...]. If extraction of pathlengths is enabled (``:extract do length''), then when the root cell in an extract command is being extracted, the extractor will compute the shortest and longest path between each driver and each receiver on the same electrical net, and output it to the .ext file for the root cell. Normally, one should create a file of these Magic commands for the circuit drivers and receivers of interest, and use source to read it in prior to circuit extraction. Extract length clear removes all the entries from both the driver and receiver tables.

extract parents

Extract the currently selected cell and all of its parents. All of its parents must be loaded in order for this to work correctly. If more than one cell is selected, this command uses the upper-leftmost one.

extract showparents

Like extract parents, but only print the cells that would be extracted; don't actually extract them.

extract style [style]

Select the style to be used for extraction parameters. If no style argument is provided, then Magic prints the names of all extraction parameter styles defined in the technology file and identifies the current style. If style is provided, it is made the current style.

extract unique [#]

For each cell in the window beneath the cursor, check to insure that no label is attached to more than one node. If the # keyword was not specified, whenever a label is attached to more than one node, the labels in all but one of the nodes are changed by appending a numeric suffix to make them unique. If the # keyword is specified, only names that end in a ``#'' are made unique; any other duplicate nodenames that don't end in a ``!'' are reported by leaving a warning feedback area. This command is provided for converting old designs that were intended for extraction with Mextra, which would automatically append unique suffixes to node names when they appeared more than once.

extract warn [[no] option | [no] all]

The extractor always reports fatal errors. This command controls the types of warnings that are reported. Option may be one of the following: dup, to warn about two or more unconnected nodes in the same cell that have the same name, fets, to warn about transistors with fewer than the minimum number of terminals, and labels, to warn when nodes are not labeled in the area of cell overlap. In addition, all may be used to refer to all warnings. If a warning is preceded by no, it is disabled. To disable all warnings, use ``extract warn no all''. To see which warning options are in effect, use ``extract warn''.

extresist [cell [threshold]]

Postprocessor for improving on the resistance calculation performed by the circuit extractor. To use this command, you first have to extract the design rooted at cell with :extract cell, and then flatten the design using ext2sim(1), producing the files cell.sim and cell.nodes. Then run :extresist cell to produce a file, cell.res.ext, containing differences between the network described by the .ext files produced the first time around, and a new network that incorporates explicit two-point resistors where appropriate (see below). This file may be appended to cell.ext, and then ext2simrun for a second time, to produce a new network with explicit resistors. The threshold parameter is used to control which nodes are turned into resistor networks: any node whose total resistance exceeds threshold times the smallest on-resistance of any transistor connected to that node will be approximated as a resistor network.

feedback option [args]

Examine feedback information that is created by several of the Magic commands to report problems or highlight certain things for users. Option and args are used in the following ways:

feedback add text [style]

Used to create a feedback area manually at the location of the box. This is intended as a way for other programs like Crystal to highlight things on a layout. They can generate a command file consisting of a feedback clear command, and a sequence of box and feedback add commands. Text is associated with the feedback (it will be printed by feedback why and feedback find). Style tells how to display the feedback, and is one of dotted, medium, outline, pale, and solid (if unspecified, style defaults to pale).

feedback clear

Clears all existing feedback information from the screen.

feedback count

Prints out a count of the current number of feedback areas.

feedback find [nth]

Used to locate a particular feedback area. If nth is specified, the box is moved to the location of the nth feedback area. If nth isn't specified, then the box is moved to the next sequential feedback area after the last one located with feedback find. In either event, the explanation associated with the feedback area is printed.

feedback help

Prints a short synopsis of all the feedback command options.

feedback save file
This option will save information about all existing feedback areas in file. The information is stored as a collection of Magic commands, so that it can be recovered with the command source file.

feedback why
Prints out the explanations associated with all feedback areas underneath the box.

fill direction [layers]
Direction is a Manhattan direction (see the section DIRECTIONS below). The paint visible under one edge of the box is sampled. Everywhere that the edge touches paint, the paint is extended in the given direction to the opposite side of the box. For example, if direction is north, then paint is sampled under the bottom edge of the box and extended to the top edge. If layers is specified, then only the given layers are considered; if layers isn't specified, then all layers are considered.

findbox [zoom]
Center the view on the box. If the optional zoom argument is present, zoom into the area specified by the box. This command will complain if the box is not in the window you are pointing to.

flush [cellname]
Cell cellname is reloaded from disk. All changes made to the cell since it was last saved are discarded. If cellname is not given, the edit cell is flushed.

garoute option [args]
This command, with no option or arg, is like the route command: it generates routing in the edit cell to make connections specified in the current netlist. (See the route command for further information). Unlike the route command, this command is intended to be used for routing types of circuits, such as gate-arrays, whose routing channels can be determined in advance, and which require the ability to river-route across the tops of cells. The channels must have been predefined using garoute channel commands prior to this command being invoked. Unlike the route command, where the box indicates the routing area, this command ignores the box entirely. The new wires are placed in the edit cell. The netlist used is that selected by the route netlist command, or the current netlist being edited in a netlist window if no route netlist command has been given. Options and args have the following effects:

garoute channel [type]

garoute channel xlo ylo xhi yhi [type]
Define a channel. If xlo, ylo, xhi, and yhi are provided, they are interpreted as the coordinates of the lower-left and upper-right of the bounding box for the channel respectively. Otherwise, the coordinates of the box are used. The boundary of each channel is adjusted inward to lie halfway between routing grid lines if it does not lie there already; if the channel is adjusted, a warning message is printed. The channel defined is an ordinary routing channel if type is not specified; such channels are identical to those used by the router of the route command. If type is given, it must be either h or v. The channel thereby created will be a river-routing channel inside which only left-to-right routes are possible (``h'') or top-to-bottom (``v''). Unlike a normal channel, a river-routing channel may contain terminals in its interior.

garoute generate type [file]
Provides a primitive form of channel decomposition for regular structures such as gate-array or standard-cell layouts. Generates a collection of garoute channel commands, either to the standard output, or to file if the latter is specified. The type parameter must be either h or v.
The entire area contained within the box is turned into routing channels. Each cell inside this area has its bounding box computed for purposes of routing by looking only at those layers con-

sidered to be ``obstacles'' to routing (see ``Tutorial #7: Netlists and Routing'' for details). The bounding box just computed is then extended all the way to the sides of the area of the box tool, vertically if type is h or horizontally if type is v. This extended area is then marked as belonging to a river-routing channel of type type; adjacent channels of this type are merged into a single channel. After all cells are processed, the areas not marked as being river-routing channels are output as normal channels.

garoute help

Print a short synopsis of all the garoute command options.

garoute nowarn

If a given terminal appears in more than one place inside a cell, the router can leave feedback if it is not possible to route to all of the places where the terminal appears. The garoute nowarn command instructs the router to leave feedback only if it is not possible to route to any of the locations of a terminal. (This is the default behavior of garoute router).

garoute route [netlist]

Route the edit cell. If netlist is not specified, the netlist used is the same as when garoute is given with no options. If netlist is given, then it is used instead.

garoute reset

Clear all channels defined by garoute channel in preparation for redefining a new set of channels.

garoute warn

The opposite of garoute nowarn, this command instructs the router to leave feedback if it is not possible to route to all of the places where a terminal appears when a terminal has more than one location, even if not all of those locations are actually selected for routing by the global router.

getcell cellName [child refPointC] [parent refPointP]

This command adds a child cell instance to the edit cell. The instance refers to the cell cellName; it is positioned so that refPointC in the child is at point refPointP in the edit cell. The reference points can either be the name of a label, in which case the lower-left corner of the label's box is used as the reference point, or as a pair of numbers giving the (x, y) coordinates of a point explicitly. If refPointC is not specified, the lower-left corner of cellName cell is used. If refPointP is not specified, the lower-left corner of the box tool is used (the box must be in a window on the edit cell). The new subcell is selected. The difference between this command and dump is that dump copies the contents of the cell, while getcell simply makes a reference to the original cell. Cellname must not be the edit cell or one of its ancestors.

getnode [alias on | alias off]

getnode [abort [str]]

Getnode prints out the node names (used by the extractor) for all selected paint. If aliasing turned on, getnode prints all the names it finds for a given node. It may not print every name that exists, however. When turned off, it just prints one name. The abort option allows the user to tell getnode that it is not important to completely search nodes that have certain names. For example, getnode abort Vdd will tell getnode not to continue searching the node if it determines that one of its names is Vdd. A getnode abort, without a string argument, will erase the list of names previously created by calling getnode abort with string arguments. Getnode can be safely aborted at any time by typing the interrupt character, usually ^C. See Tutorial #11: Using IRSIM and RSIM with Magic for more information on this command.

grid [xSpacing [ySpacing [xOrigin yOrigin]]]

grid off

If no arguments are given, a one-unit grid is toggled on or off in the window underneath the cursor. Grid off always turns the grid off, regardless of whether it was on or off previously. If numerical arguments are given, the arguments determine the grid spacing and origin for the window under the cursor. In its most general form, grid takes four integer arguments. XOrigin and yOrigin specify an origin for the grid: horizontal and vertical grid lines will pass through this point. XSpacing and ySpacing determine the number of units between adjacent grid lines. If xOrigin and yOrigin are omitted, they default to 0. If ySpacing is also omitted, the xSpacing value is used for both spacings. Grid parameters will be retained for a window until explicitly changed by another grid command. When the grid is displayed, a solid box is drawn to show the origin of the edit cell.

identify instance_id

Set the instance identifier of the selected cell use to instance_id. Instance_id must be unique among all instance identifiers in the parent of the selected cell. Initially, Magic guarantees uniqueness of identifiers by giving each cell an initial identifier consisting of the cell definition name followed by an underscore and a small integer.

iroute subcommand [args]

This command provides an interactive interface to the Magic maze-router. Routing is done one connection at a time. Three internal hint layers, magnet, fence, and rotate, allow the user to guide routing graphically. Routes are chosen close to magnets (if possible), routing does not cross fence boundaries, and rotate areas reverse the preferred routing directions for each layer. The maze-router seeks to find a lowest-cost path. Parameters specifying costs for horizontal and vertical routing on each layer, cost for jogs and contacts, and cost (per unit area) for distance between a path and magnets, help determine the nature of the routes. Several search parameters permit tuning to achieve acceptable routes in as short a time as possible. Routing can always be interrupted with ^C. The iroute subcommands are as follows:

iroute Routes from cursor to inside box.

iroute contact [type] [parameter] [value1] ... [valuen]

An asterisk, *, can be used for type and parameter. This command is for setting and examining parameters related to contacts.

iroute help [subcommand]

Summarizes irouter commands. If a subcommand is given, usage information for that subcommand is printed.

iroute layers [type] [parameter] [value1] ... [valuen]

An asterisk, *, can be used for type and parameter. This command is for setting and examining parameters related to route layers.

iroute route [options]

Invokes the router. Options are as follows:

-sLayers layers = layers route may start on
-sCursor = start route at cursor (DEFAULT)
-sLabel name = start route at label of given name
-sPoint x y = start route at given coordinates
-dLayers layers = layers route may end on
-dBox = route to box (DEFAULT)
-dLabel name = route to label of given name
-dRect xbot ybot xtop ytop = route to rectangle of given coordinates
-dSelection = route to selection

iroute saveParameters <filename>

Saves all current irouter parameter settings. The parameters can be restored to these values with the command ``source filename''.

iroute search [searchParameter] [value]

Allows parameters controlling the search to be modified. If routing is too slow try increasing rate. If the router is producing bad results, try reducing rate. Its a good idea to make width at least twice as big as rate.

iroute spacings [route-type] [type] [spacing] ... [typen spacingn]

Default minimum spacings between a route-type placed by the router and other types are derived from the drc section of the technology file. The defaults can be overridden by this command. The special type SUBCELL is used to specify minimum spacing to unexpanded subcells.

iroute verbosity [level]

Controls the number of messages printed during routing:

0 = errors and warnings only,
1 = brief,
2 = lots of statistics.

iroute version

Prints irouter version information.

iroute wizard [wizardparameter] [value]

Used to examine and set miscellaneous parameters. Most of these are best left alone by the un-adventurous user.

label string [pos [layer]]

A label with text string is positioned at the box location. Labels may cover points, lines, or areas, and are associated with specific layers. Normally the box is collapsed to either a point or to a line (when labeling terminals on the edges of cells). Normally also, the area under the box is occupied by a single layer. If no layer argument is specified, then the label is attached to the layer under the box, or space if no layer covers the entire area of the box. If layer is specified but layer doesn't cover the entire area of the box, the label will be moved to another layer or space. Labels attached to space will be considered by CIF processing programs to be attached to all layers overlapping the area of the label. Pos is optional, and specifies where the label text is to be displayed relative to the box (e.g. ``north''). If pos isn't given, Magic will pick a position to ensure that the label text doesn't stick out past the edge of the cell.

layers Prints out the names of all the layers defined for the current technology.

load [file]

Load the cell hierarchy rooted at file.mag into the window underneath the cursor. If no file is supplied, a new unnamed cell is created. The root cell of the hierarchy is made the edit cell unless there is already an edit cell in a different window.

move [direction [amount]]

move to x y

If no arguments are given, the selection is picked up by the point underneath the lower-left corner of the box and moved so that this point lies at the cursor location. If direction is given, it must be a Manhattan direction (e.g. north). The selection is moved in that direction by amount. If the box is in the same window as the selection, it is moved too. Amount defaults to 1. Selected material that is not in the edit cell, is not affected. The second form of the command is as though the cursor were pointing to (x, y) in the edit cell; the selection is picked up by the point beneath the lower-left corner of the box and moved so that this point lies at (x, y).

paint layers

The area underneath the box is painted in layers.

path [searchpath]

This command tells Magic where to look for cells. Searchpath contains a list of directories separated by colons or spaces (if spaces are used, then searchpath must be surrounded by quotes). When looking for a cell, Magic will check each directory in the path in order, until the cell is found. If the cell is not found anywhere in the path, Magic will look in the system library for it. If the path command is invoked with no arguments, the current search path is printed.

plot option [args]

Used to generate hardcopy plots direct from Magic. Options and args are used in the following ways:

plot gremlin file [layers]

Generate a Gremlin-format description of everything under the box, and write the description in file. If layers isn't specified, paint, labels, and unexpanded subcells are all included in the Gremlin file just as they appear on the screen. If layers is specified, then just the indicated layers are output in the Gremlin file. Layers may include the special layers labels and subcell. The Gremlin file is scaled to have a total size between 256 and 512 units; you should use the width and/or height Grn commands to ensure that the printed version is the size you want. Use the mg stipples in Grn. No plot parameters are used in Gremlin plotting.

plot help

Print a short synopsis of all the plot command options.

plot parameters [name value]

If plot parameters is invoked with no additional arguments, the values for all of the plot parameters are printed. If name and value are provided, then name is the name of a plot parameter and value is a new value for it. Plot parameters are used to control various aspects of plotting; all of them have ``reasonable'' initial values. Most of the parameters available now are used to control Versatec-style plotting. They are:

cellIdFont

The name of the font to use for cell instance ids in Versatec plots. This must be a file in Vfont format.

cellNameFont

The name of the font to use for cell names in Versatec plots. This must be a file in Vfont format.

color If this is set to true, the :plot versatec command will generate output suitable for a four-color Versatec plotter, using the styles defined in the colorversatec style of the plot section of the technology file. If color is false (the default), then :plot versatec generates normal black-and-white plots.

directory

The name of the directory in which to create raster files for the Versatec. The raster files have names of the form magicPlotXXXXXX, where XXXXXX is a process-specific identifier.

dotsPerInch

Indicates how many dots per inch there are on the Versatec printer. This parameter is used only for computing the scale factor for plotting. Must be an integer greater than zero.

labelFont

The name of the font to use for labels in Versatec plots. This must be a file in Vfont format.

printer

The name of the printer to which to spool Versatec raster files.

showcellnames
If ``true'' (the default) then the name and instance-identifier of each unexpanded subcell is displayed inside its bounding box. If this parameter is ``false'' then only the bounding box of the cell is displayed.

spoolCommand
The command used to spool Versatec raster files. This must be a text string containing two ``%s'' formatting fields. The first ``%s'' will be replaced with the printer name, and the second one will be replaced with the name of the raster file.

swathHeight
How many raster lines of Versatec output to generate in memory at one time. The raster file is generated in swaths in order to keep the memory requirements reasonable. This parameter determines the size of the swaths. It must be an integer greater than zero, and should be a multiple of 16 in order to avoid misalignment of stipple patterns.

width The number of pixels across the Versatec printer. Must be an integer greater than 0, and must be an even multiple of 32.

plot versatec [size [layers]]
Generate a raster file describing all the the information underneath the box in a format suitable for printing on Versatec black-and-white or color printers, and spool the file for printing. See the plot parameters above for information about the parameters that are used to control Versatec plotting. Size is used to scale the plot: a scalefactor is chosen so that the area of the box is size inches across on the printed page. Size defaults to the width of the printer. Layers selects which layers (including labels and subcells) to plot; it defaults to everything visible on the screen.

plow direction [layers]

plow option [args]
The first form of this command invokes the plowing operation to stretch and/or compact a cell. Direction is a Manhattan direction. Layers is an optional collection of mask layers, which defaults to *. One of the edges of the box is treated as a plow and dragged to the opposite edge of the box (e.g. the left edge is used as the plow when plow right is invoked). All edges on layers that lie in the plow's path are pushed ahead of it, and they push other edges ahead of them to maintain design rules, connectivity, and transistor and contact sizes. Subcells are moved in their entirety without being modified internally. Any mask information overlapping a subcell moved by plowing is also moved by the same amount. Option and args are used in the following ways:

plow boundary
The box specifies the area that may be modified by plowing. This area is highlighted with a pale stipple outline. Subsequent plows are not allowed to modify any area outside that specified by the box; if they do, the distance the plow moves is reduced by an amount sufficient to insure that no geometry outside the boundary gets affected.

plow help
Prints a short synopsis of all the plow command options.

plow horizon n

plow horizon
The first form sets the plowing jog horizon to n units. The second form simply prints the value of the jog horizon. Every time plowing considers introducing a jog in a piece of material, it looks up and down that piece of material for a distance equal to the jog horizon. If it finds an existing jog within this distance, it uses it. Only if no jog is found within the jog hori-

zon does plowing introduce one of its own. A jog horizon of zero means that plowing will always introduce new jogs where needed. A jog horizon of infinity (plow nojogs) means that plowing will not introduce any new jogs of its own.

plow jogs
Re-enable jog insertion with a horizon of 0. This command is equivalent to plow horizon 0.

plow noboundary
Remove any boundary specified with a previous plow boundary command.

plow nojogs
Sets the jog horizon to infinity. This means that plowing will not introduce any jogs of its own; it will only use existing ones.

plow nostraighten
Don't straighten jogs automatically after each plow operation.

plow selection [direction [distance]]
Like the move or stretch commands, this moves all the material in the selection that belongs to the edit cell. However, any material not in the selection is pushed out of its way, just as though each piece of the selection were plowed individually. If no arguments are given, the selection is picked up by the point underneath the lower-left corner of the box and plowed so that this point lies at the cursor location. The box is moved along with the selection. If direction is given, it must be a Manhattan direction (e.g. north). The selection is moved in that direction by amount. If the box is in the same window as the selection, it is moved too. Amount defaults to 1. If there is selected material that isn't in the edit cell, it is ignored (note that this is different from select and move). If direction isn't given and the cursor isn't exactly left, right, up, or down from the box corner, then Magic first rounds the cursor position off to a position that is one of those (whichever is closest).

plow straighten
Straighten jogs automatically after each plow operation. The effect will be as though the straighten command were invoked after each plow operation, with the same direction, and over the area changed by plowing.

resist cell [tolerance]
This command is similar to extresist above, but used for extracting resistance networks for individual nodes. Only the node underneath the box is processed. The network for this node is output to the file cell.res.ext. See the description for extresist for an explanation of tolerance.

route option [args]
This command, with no option or arg, is used to generate routing using the Magic router in the edit cell to make connections specified in the current netlist. The box is used to indicate the routing area: no routing will be placed outside the area of the box. The new wires are placed in the edit cell. Options and args have the following effects:

route end [real]
Print the value of the channel end constant used by the channel router. If a value is supplied, the channel end constant is set to that value. The channel end constant is a dimensionless multiplier used to compute how far from the end of a channel to begin preparations to make end connections.

route help
Print a short synopsis of all the route command options.

route jog [int]
Print the value of the minimum jog length used by the channel router. If a value is supplied,

the minimum jog length is set to that value. The channel router makes no vertical jogs shorter than the minimum jog length, measured in router grid units. Higher values for this constant may improve the quality of the routing by removing unnecessary jogs; however, prohibiting short jogs may make some channels unroutable.

route metal

Toggle metal maximization on or off. The route command routes the preferred routing layer (termed ``metal'') horizontally and the alternate routing layer vertically. By default wires on the alternate routing layer are then converted, as much as possible, to the preferred layer before being painted into the layout. Enabling metal maximization improves the quality of the resulting routing, since the preferred routing layer generally has better electrical characteristics; however, designers wishing to do hand routing after automatic routing may find it easier to disable metal maximization and deal with a layer-per-direction layout.

route netlist [file]

Print the name of the current netlist. If a file name is specified, it is opened if possible, and the new netlist is loaded. This option is provided primarily as a convenience so you need not open the netlist menu before routing.

route obstacle [real]

Print the obstacle constant used by the channel router. If a value is supplied, set the channel router obstacle constant to that value. The obstacle constant is a dimensionless multiplier used in deciding how far in front of an obstacle the channel router should begin jogging nets out of the way. Larger values mean that nets will jog out of the way earlier; however, if nets jog out of the way too early routing area is wasted.

route origin [x y]

Print the x- and y-coordinates of the origin of the routing grid. By default, the routing grid starts from (0,0). However, by supplying an x and y coordinate to the route origin command, the origin can be set to any other value. This command is primarily useful when routing a chip that has been designed with routing on the same pitch as the router will use, but where the left and bottom edges of the pre-existing routing don't line up with the routing grid lines (for example, the pre-existing routing might have been centered on routing grid lines). The alternative to specifying a different origin for the routing grid would be to translate all the material in the cell to be routed so that the prewiring lined up properly with routing grid lines.

route settings

Print the values of all router parameters.

route steady [int]

Print the value of the channel router's steady net constant. If a value is supplied, set the steady net constant to the value. The steady net constant, measured in router grid units, specifies how far beyond the next terminal the channel router should look for a conflicting terminal before deciding that a net is rising or falling. Larger values mean that the net rises and falls less often.

route tech

Print the router technology information. This includes information such as the names of the preferred and alternate routing layers, their wire widths, the router grid spacing, and the contact size.

route viamin

Minimize vias in (previously) routed netlist. This subcommand removes unnecessary layer changes in all nets in the current netlist to minimize via count. The preferred routing layer, layer1 in the router section of the technology file, is favored by the algorithm. Note that ``route viamin'' is an independent routing postpass that can be applied even if the routing was not generated by the route command, provided the layers and widths agree with the router section of the technology file.

route vias [int]

Print the value of the metal maximization via constant. If a value is supplied, set the via constant to the value. The via constant, measured in router grid units, represents the tradeoff between metal maximization and the via count. In many cases it is possible to convert wiring on the alternate routing layer into routing on the preferred routing layer (``metal'') at the expense of introducing one or two vias. The via constant specifies the amount of converted wiring that makes it worthwhile to add vias to the routing.

rsim [options] [filename]

Runs rsim under Magic. See Tutorial #11: Using IRSIM and RSIM with Magic for more information on what options and files are required by rsim. Normally, IRSIM requires a parameter file for the technology and a .sim file describing the circuit.

The rsim command without any options can be used to interact with a previously-started rsim. Type rsim and you will see the rsim prompt. To get back to magic, type q.

save [name]

Save the edit cell on disk. If the edit cell is currently the ``(UNNAMED)'' cell, name must be specified; in this case the edit cell is renamed to name as well as being saved in the file name.mag. Otherwise, name is optional. If specified, the edit cell is saved in the file name.mag; otherwise, it is saved in the file from which it was originally read.

see option

This command is used to control which layers are to be displayed in the window under the cursor. It has several forms:

see no layers

Do not display the given layers in the window under the cursor. If labels is given as a layer name, don't display labels in that window either. If errors is given as a layer, no design-rule violations will be displayed (the checker will continue to run, though). If layers is given as "*", all mask layers will be disabled, but errors and labels will still be shown. See the "LAYERS" section at the end of this manual page for an explanation of layer naming in Magic.

see layers

Reenable display of the given layers. Note that "*" expands to all mask layers, but does not include the label or error layers. See the "LAYERS" section at the end of this manual page for details.

see no Don't display any mask layers or labels. Only subcell bounding boxes will be displayed.

see Reenable display of all mask layers, labels, and errors.

see allSame

Display all cells the same way. This disables the facility where the edit cell is displayed in bright colors and non-edit cells are in paler colors. After see allSame, all mask information will be displayed in bright colors.

see no allSame

Reenable the facility where non-edit cells are drawn in paler colors.

select option

This command is used to select paint, labels, and subcells before operating on them with commands like move and copy and delete. It has several forms:

select If the cursor is over empty space, then this command is identical to select cell. Otherwise, paint is selected. The first time the command is invoked, a chunk of paint is selected: the largest rectangular area of material of the same type visible underneath the cursor. If the command is invoked again without moving the cursor, the selection is extended to include all material of the same type, regardless of shape. If the command is invoked a third time, the selection is extended again to include all material that is visible and electrically connected to the point underneath the cursor.

select more
: This command is identical to select except that the selection is not first cleared. The result is to add the newly-selected material to what is already in the selection.

select less
: This chooses material just as select does, but the material is removed from the selection, rather than added to it. The result is to deselect the chosen material.

select [more | less] area layers
: Select material by area. If layers are not specified, then all paint, labels, and unexpanded subcells visible underneath the box are selected. If layers is specified, then only those layers are selected. If more is specified, the new material is added to the current selection rather than replacing it. If less is specified, the new material is removed from the selection (deselected).

select [more | less] cell name
: Select a subcell. If name isn't given, this command finds a subcell that is visible underneath the cursor and selects it. If the command is repeated without moving the cursor then it will step through all the subcells under the cursor. If name is given, it is treated as a hierarchical instance identifier starting from the root of the window underneath the cursor. The named cell is selected. If more is specified, the new subcell is added to the current selection instead of replacing it. If less is specified, the new subcell is removed from the selection (deselected).

select clear
: Clear out the selection. This does not affect the layout; it merely deselects everything.

select help
: Print a short synopsis of the selection commands.

select save cell
: Save all the information in the selection as a Magic cell on disk. The selection will be saved in file cell.mag.

select and the see command
: Select interacts with the see command. When selecting individual pieces of material, only visible layers are candidates for selection. When selecting an entire area, however, both visible and non-visible material is selected. This behavior allows entire regions of material to be moved, even if see has been used to turn off the display of some of the layers.

sideways
: Flip the selection left-to-right about a vertical axis running through the center of the selection's area. If the box is in the same window as the selection, it is flipped too. Selected material not in the edit cell is not affected.

simcmd cmd
: Sends the command cmd to rsim for execution. See Tutorial #11: Using IRSIM and RSIM with Magic for more information.

snap [on]

snap [off]
Control whether the box and point are snapped to the grid selected for the windows in which they appear (the grid was set by the grid command), or to the standard 1x1 grid. The default is for snapping to be off, i.e., snapping to a 1x1 grid. With no arguments, snap prints whether snapping is enabled or not.

startrsim [options] [filename]
Similar to the rsim command, except it returns to Magic as soon as rsim is started. See Tutorial #11: Using IRSIM and RSIM with Magic for more information.

straighten direction
Straighten jogs in wires underneath the box by pulling them in direction. Jogs are only straightened if doing so will cause no additional geometry to move.

stretch [direction [amount]]
This command is identical to move except that simple stretching occurs as the selection is moved. Each piece of paint in the selection causes the area through which it's moved to be erased in that layer. Also, each piece of paint in the selection that touches unselected material along its back side causes extra material to be painted to fill in the gap left by the move. If direction isn't given and the cursor isn't exactly left, right, up, or down from the box corner, then Magic first rounds the cursor position off to a position that is one of those (whichever is closest).

tool [name | info]
Change the current tool. The result is that the cursor shape is different and the mouse buttons mean different things. The command tool info prints out the meanings of the buttons for the current tool. Tool name changes the current tool to name, where name is one of box, wiring, or netlist. If tool is invoked with no arguments, it picks a new tool in circular sequence: multiple invocations will cycle through all of the available tools.

unexpand
Unexpand all cells that touch the box but don't completely contain it.

upsidedown
Flip the selection upside down about a horizontal axis running through the center of the selection's area. If the box is in the same window as the selection then it is flipped too. Selected material that is not in the edit cell is not changed.

what Print out information about all the things that are selected.

wire option [args]
This command provides a centerline-wiring style user interface. Option and args specify a particular wiring option, as described below. Some of the options can be invoked via mouse buttons when the wiring tool is active.

wire help
Print out a synopsis of the various wiring commands.

wire horizontal
Just like wire leg except that the new segment is forced to be horizontal.

wire leg
Paint a horizontal or vertical segment of wire from one side of the box over to the cursor's x- or y-location (respectively). The direction (horizontal or vertical) is chosen so as to produce the longest possible segment. The segment is painted in the current wiring material and thickness. The new segment is selected, and the box is placed at its tip.

wire switch [layer width]

Switch routing layers and place a contact at the box location. The contact type is chosen to connect the old and new routing materials. The box is placed at the position of the contact, and the contact is selected. If layer and width are specified, they are used as the new routing material and width, respectively. If they are not specified, the new material and width are chosen to correspond to the material underneath the cursor.

wire type [layer width]

Pick a material and width for wiring. If layer and width are not given, then they are chosen from the material underneath the cursor, a square chunk of material is selected to indicate the layer and width that were chosen, and the box is placed over this chunk. If layer and width are given, then this command does not modify the box position.

wire vertical

Just like wire leg except that the new segment is forced to be vertical.

writeall [force]

This command steps through all the cells that have been modified in this edit session and gives you a chance to write them out. If the force option is specified, then ``autowrite'' mode is used: all modified cells are automatically written without asking for permission.

COMMANDS FOR ALL WINDOWS

These commands are not used for layout, but are instead used for overall, housekeeping functions. They are valid in all windows.

closewindow

The window under the cursor is closed. That area of the screen will now show other windows or the background.

echo [-n] str1 str2 ... strN

Prints str1 str2 ... strN in the text window, separated by spaces and followed by a newline. If the -n switch is given, no newline is output after the command.

help [pattern]

Displays a synopsis of commands that apply to the window you are pointing to. If pattern is given then only command descriptions containing the pattern are printed. Pattern may contain '*' and '?' characters, which match a string of non-blank characters or a single non-blank character (respectively).

logcommands [file [update]]

If file is given, all further commands are logged to that file. If no arguments are given, command logging is terminated. If the keyword update is present, commands are output to the file to cause the screen to be updated after each command when the command file is read back in.

macro [char [command]]

Command is associated with char such that typing char on the keyboard is equivalent to typing ``:'' followed by command. If command is omitted, the current macro for char is printed. If char is also omitted, then all current macros are printed. If command contains spaces, tabs, or semicolons then it must be placed in quotes. The semicolon acts as a command separator allowing multiple commands to be combined in a single macro.

openwindow [cell]

Open a new, empty window at the cursor position. Placement, sizing, and methods of manipulation are determined by the conventions of the window system in use. If cell is specified, then that cell is displayed in the new window. Otherwise the area of the box will be displayed in the new window.

pushbutton button action

Simulates a button push. Button should be left, middle, or right. Action is one of up, or down. This command is normally invoked only from command scripts produced by the logcommands command.

quit   Exit Magic and return to the shell. If any cells, colormaps, or netlists have changed since they were
last saved on disk, you are given a chance to abort the command and continue in Magic.

redo [n]
Redo the last n commands that were undone using undo (see below). The number of commands to redo defaults to 1 if n is not specified.

redraw Redraw the graphics screen.

scroll direction [amount]
The window under the cursor is moved by amount screenfulls in direction relative to the circuit. If amount is omitted, it defaults to 0.5.

send type command
Send a command to the window client named by type. The result is just as if command had been typed in a window of type type. See specialopen, below, for the allowable types of windows.

setpoint [x y [windowID]]
Fakes the location of the cursor up until after the next interactive command. Without arguments, just prints out the current point location. This command is normally invoked only from command scripts.

If windowID is given, then the point is assumed to be in that window's screen coordinate system rather than absolute screen coordinates.

sleep n
Causes Magic to go to sleep for n seconds.

source filename
Each line of filename is read and processed as one command. Any line whose last character is backslash is joined to the following line. The commands setpoint, pushbutton, echo, sleep, and updatedisplay are useful in command files, and seldom used elsewhere.

specialopen [x1 y1 x2 y2] type [args]
Open a window of type type. If the optional x1 y1 x2 y2 coordinates are given, then the new window will have its lower left corner at screen coordinates (x1, y1) and its upper right corner at screen coordinates (x2, y2). The args arguments are interpreted differently depending upon the type of the window. These types are known:

layout This type of window is used to edit a VLSI cell. The command takes a single argument which is used as the name of a cell to be loaded. The command

```
open filename
```

is a shorthand for the command

```
specialopen layout filename.
```

color This type of window allows the color map to be edited. See the section COMMANDS FOR COLORMAP EDITING below.

netlist
This type of window presents a menu that can be used to place labels, and to generate and edit net-lists. See the section COMMANDS FOR NETLIST EDITING below.

underneath
Move the window pointed at so that it lies underneath the rest of the windows.

undo [count]
Undoes the last count commands. Almost all commands in Magic are now undo-able. The only holdouts

left are cell expansion/unexpansion, and window modifications (change of size, zooming, etc.). If count is unspecified, it defaults to 1.

updatedisplay
Update the display. This command is normally invoked only from command scripts. Scripts that do not contain this command update the screen only at the end of the script.

view Choose a view for the window underneath the cursor so that everything in the window is visible.

windscrollbars [on|off]
Set the flag that determines if new windows will have scroll bars.

windowpositions [file]
Write out the positions of the windows in a format suitable for the source command. If file is specified, then write it out to that file instead of to the terminal.

zoom [factor]
Zoom the view in the window underneath the cursor by factor. If factor is less than 1, we zoom in; if it is greater than one, we zoom out.

MOUSE BUTTONS FOR NETLIST WINDOWS

When the netlist menu is opened using the command special netlist, a menu appears on the screen. The colored areas on the menu can be clicked with various mouse buttons to perform various actions, such as placing labels and editing netlists. For details on how to use the menu, see ``Magic Tutorial #7: Netlists and Routing''. The menu buttons all correspond to commands that could be typed in netlist or layout windows.

COMMANDS FOR NETLIST WINDOWS

The commands described below work if you are pointing to the interior of the netlist menu. They may also be invoked when you are pointing at another window by using the send netlist command. Terminal names in all of the commands below are hierarchical names consisting of zero or more cell use ids separated by slashes, followed by the label name, e.g. toplatch/shiftcell_1/in. When processing the terminal paths, the search always starts in the edit cell.

add term1 term2
Add the terminal named term1 to the net containing terminal term2. If term2 isn't in a net yet, make a new net containing just term1 and term2.

cleanup
Check the netlist to make sure that for every terminal named in the list there is at least one label in the design. Also check to make sure that every net contains at least two distinct terminals, or one terminal with several labels by the same name. When errors are found, give the user an opportunity to delete offending terminals and nets. This command can also be invoked by clicking the ``Cleanup'' menu button.

cull Examine the current netlist and the routing in the edit cell, and remove those nets from the netlist that are already routed. This command is often used after pre-routing nets by hand, so the router won't try to implement them again.

dnet name name ...
For each name given, delete the net containing that terminal. If no name is given, delete the currently-selected net, just as happens when the ``No Net'' menu button is clicked.

dterm name name ...
For each name given, delete that terminal from its net.

extract
Pick a piece of paint in the edit cell that lies under the box. Starting from this, trace out all the

electrically-connected material in the edit cell. Where this material touches subcells, find any terminals in the subcells and make a new net containing those terminals. Note: this is a different command from the extract command in layout windows.

find pattern [layers]
Search the area beneath the box for labels matching pattern, which may contain the regular-expression characters ``*'' ``?'', ``['', ``]'', and ``\'' (as matched by csh(1); see the description of the find button in ``Magic Tutorial #7: Netlists and Routing''). For each label found, leave feedback whose text is the layer on which the label appears, followed by a semicolon, followed by the full hierarchical pathname of the label. The feedback surrounds the area of the label by one unit on all sides. (The reason for the one-unit extension is that feedback rectangles must have positive area, while labels may have zero width or height). If layers are given, only labels attached to those layers are considered.

flush [netlist]
The netlist named netlist is reloaded from the disk file netlist.net. Any changes made to the netlist since the last time it was written are discarded. If netlist isn't given, the current netlist is flushed.

join term1 term2
Join together the nets containing terminals term1 and term2. The result is a single net containing all the terminals from both the old nets.

netlist [name]
Select a netlist to work on. If name is provided, read name.net (if it hasn't already been read before) and make it the current netlist. If name isn't provided, use the name of the edit cell instead.

print [name]
Print the names of all the terminals in the net containing name. If name isn't provided, print the terminals in the current net. This command has the same effect as clicking on the ``Print'' menu button.

ripup [netlist]
This command has two forms. If netlist isn't typed as an argument, then find a piece of paint in the edit cell under the box. Trace out all paint in the edit cell that is electrically connected to the starting piece, and delete all of this paint. If netlist is typed, find all paint in the edit cell that is electrically connected to any of the terminals in the current netlist, and delete all of this paint.

savenetlist [file]
Save the current netlist on disk. If file is given, write the netlist in file.net. Otherwise, write the netlist back to the place from which it was read.

shownet
Find a piece of paint in any cell underneath the box. Starting from this paint, trace out all paint in all cells that is electrically connected to the starting piece and highlight this paint on the screen. To make the highlights go away, invoke the command with the box over empty space. This command has the same effect as clicking on the ``Show'' menu button.

showterms
Find the labels corresponding to each of the terminals in the current netlist, and generate a feedback area over each. This command has the same effect as clicking on the ``Terms'' menu button.

trace [name]
This command is similar to shownet except that instead of starting from a piece of paint under the box, it starts from each of the terminals in the net containing name (or the current net if no name is given). All connected paint in all cells is highlighted.

verify Compare the current netlist against the wiring in the edit cell to make sure that the nets are implemented exactly as specified in the netlist. If there are discrepancies, feedback areas are created to describe them. This command can also be invoked by clicking the ``Verify'' menu button.

writeall
Scan through all the netlists that have been read during this editing session. If any have been modified, ask the user whether or not to write them out.

MOUSE BUTTONS FOR COLORMAP WINDOWS

Color windows display two sets of colored bars and a swatch of the color being edited. The left set of color bars is labeled Red, Green, and Blue; these correspond to the proportion of red, green, and blue in the color being edited. The right set of bars is labeled Hue, Saturation, and Value; these correspond to the same color but in a space whose axes are hue (spectral color), saturation (spectral purity vs. dilution with white), and value (light vs. dark).

The value of a color is changed by pointing inside the region spanned by one of the color bars and clicking any mouse button. The color bar will change so that it extends to the point selected by the crosshair when the button was pressed. The color can also be changed by clicking a button over one of the ``pumps'' next to a color bar. A left-button click makes a 1% increment or decrement, and a right-button click makes a 5% change.

The color being edited can be changed by pressing the left button over the current color box in the editing window, then moving the mouse and releasing the button over a point on the screen that contains the color to be edited. A color value can be copied from an existing color to the current color by pressing the right mouse button over the current color box, then releasing the button when the cursor is over the color whose value is to be copied into the current color.

COMMANDS FOR COLORMAP WINDOWS

These commands work if you are pointing to the interior of a colormap window. The commands are:

color [number]
Load number as the color being edited in the window. Number must be an octal number between 0 and 377; it corresponds to the entry in the color map that is to be edited. If no number is given, this command prints out the value of the color currently being edited.

load [techStyle displayStyle monitorType]
Load a new color map. If no arguments are specified, the color map for the current technology style (e.g, mos), display style (e.g, 7bit), and monitor type (e.g, std) is re-loaded. Otherwise, the color map is read from the file techStyle.displayStyle.monitorType.cmap in the current directory or in the system library directory.

save [techStyle displayStyle monitorType]
Save the current color map. If no arguments are specified, save the color map in a file determined by the current technology style, display style, and monitor type as above. Otherwise, save it in the file techStyle.displayStyle.monitorType.cmap in the current directory or in the system library directory.

DIRECTIONS

Many of the commands take a direction as an argument. The valid direction names are north, south, east, west, top, bottom, up, down, left, right, northeast, ne, southeast, se, northwest, nw, southwest, sw, and center. In some cases, only Manhattan directions are permitted, which means only north, south, east, west, and their synonyms, are allowed.

LAYERS

The mask layers are different for each technology, and are described in the technology manuals. The layers below are defined in all technologies:

*   All mask layers. Does not include special layers like the label layer and the error layer (see below).

$ All layers underneath the cursor.

errors Design-rule violations (useful primarily in the see command).

labels Label layer.

subcell
    Subcell layer.

Layer masks may be formed by constructing comma-separated lists of individual layer names. The individual layer names may be abbreviated, as long as the abbreviations are unique. For example, to indicate polysilicon and n-diffusion, use poly,ndiff or ndiff,poly. The special character - causes all subsequent layers to be subtracted from the layer mask. For example, *-p means ``all layers but polysilicon''. The special character + reverses the effect of a previous -; all subsequent layers are once again added to the layer mask.

SEE ALSO

ext2sim(1), ext2spice(1), cmap(5), dstyle(5), ext(5), sim(5), glyphs(5), magic(5), displays(5), net(5)

Online documentation can be found at the following URLs:
http://opencircuitdesign.com/magic/
http://vlsi.cornell.edu/magic/
The OpenCircuitDesign website contains HTML versions of all the documentation found in the Magic "doc" subdirectory, including tutorials, technology file manual; download, compile and install instructions, and command reference.

FILES

${CAD_ROOT}/magic/sys/.magicstartup file to create default macros
~/.magic   user-specific startup command file
${CAD_ROOT}/magic/nmos/*some standard nmos cells
${CAD_ROOT}/magic/scmos/*some standard scmos cells
${CAD_ROOT}/magic/sys/*.cmapcolormap files, see CMAP(5) man page
${CAD_ROOT}/magic/sys/*.dstyledisplay style files, see DSTYLE(5) man page
${CAD_ROOT}/magic/sys/*.glyphscursor and window bitmap files, see GLYPH(5) man page
${CAD_ROOT}/magic/sys/*.techtechnology files, see ``Maintainer's Manual
                #2: The Technology File''
${CAD_ROOT}/displaysconfiguration file for Magic serial-line displays

CAD_ROOT variable. If the shell environment variable CAD_ROOT is set, Magic uses that location instead of the installed location (/usr/local/lib, by default). Normally one would change the search path (see below) rather than redirect the entire CAD_ROOT location.

Search path. Magic's system and library files, such as technology files and display-style files, normally are placed in the ${CAD_ROOT}/magic area. However, Magic first tries to find them in the user's current directory. This makes it easier for an individual user to override installed system files. The search path is defined by the Magic command path,

AUTHORS

Original: Gordon Hamachi, Robert Mayo, John Ousterhout, Walter Scott, George Taylor

Contributors: Michael Arnold (Magic maze-router and Irouter command), Don Stark (new contact scheme, X11 interface, various other things), Mike Chow (Rsim interface). The X11 driver is the work of several people, including Don Stark, Walter Scott, and Doug Pan.

Developers: Ongoing development (magic version 6.5 and higher) made possible by Stefanos Sidiropolous, Tim Edwards, Rajit Manohar, Philippe Pouliquen, Michael Godfrey, and others.

Many other people have contributed to Magic, but it is impossible to list them all here. We appreciate their help!

BUGS

If Magic gets stuck for some reason, first try typing Control-C into the terminal window (in the Tcl/Tk version, this is the original terminal, not the Tk console window). Most of Magic's lengthy database searches are interruptible. If this doesn't work, kill the process. The Tcl/Tk version automatically creates periodic backups that may be recovered with "magic -r".

Report bugs to magic-dev@csl.cornell.edu. Please be specific: tell us exactly what you did to cause the problem, what you expected to happen, and what happened instead. If possible send along small files that we can use to reproduce the bug. A list of known bugs and fixes is also available from the above address. Tell us which version of magic you are running.