cadence®

Innovus Database Object Information

Product Version 20.10
March 2020

© 2013-2020 Cadence Design Systems, Inc. All rights reserved.
Printed in the United States of America.

Cadence Design Systems, Inc. (Cadence), 2655 Seely Ave., San Jose, CA 95134, USA.

Open SystemC, Open SystemC Initiative, OSCI, SystemC, and SystemC Initiative are trademarks or registered trademarks of Open SystemC Initiative, Inc. in the United States and other countries and are used with permission.

Trademarks: Trademarks and service marks of Cadence Design Systems, Inc. (Cadence) contained in this document are attributed to Cadence with the appropriate symbol. For queries regarding Cadence's trademarks, contact the corporate legal department at the address shown above or call 800.862.4522.

All other trademarks are the property of their respective holders.

Restricted Permission: This publication is protected by copyright law and international treaties and contains trade secrets and proprietary information owned by Cadence. Unauthorized reproduction or distribution of this publication, or any portion of it, may result in civil and criminal penalties. Except as specified in this permission statement, this publication may not be copied, reproduced, modified, published, uploaded, posted, transmitted, or distributed in any way, without prior written permission from Cadence. Unless otherwise agreed to by Cadence in writing, this statement grants Cadence customers permission to print one (1) hard copy of this publication subject to the following conditions:

1. The publication may be used only in accordance with a written agreement between Cadence and its customer.
2. The publication may not be modified in any way.
3. Any authorized copy of the publication or portion thereof must include all original copyright, trademark, and other proprietary notices and this permission statement.
4. The information contained in this document cannot be used in the development of like products or software, whether for internal or external use, and shall not be used for the benefit of any other party, whether or not for consideration.

Disclaimer: Information in this publication is subject to change without notice and does not represent a commitment on the part of Cadence. Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information.

Restricted Rights: Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor.

Contents

1

Types and Definitions

Attribute Type	Actual Definition
area	Single value; either int (dbGet -d), or float. Units are coord*coord (squared area, either micron^2 or, in the -d case, dbu^2)
bool (Boolean)	Boolean value (0=false and 1=true)
coord (Coordinate)	Single value; either int (dbGet -d), or float.
enum	Enumerated type with specified list of legal values
float	Floating point number
int	Integer number
list	List (can be any attribute type except object which uses the objList specifically)
list(list)	Space separated list
list(pt) (list of points)	{pt pt ...} { {coord coord} {coord coord} ...}
obj (object)	Single object pointer (object type is enclosed in the "(" & ")"). A single object can be one of a set of possible objects. This is different from the objList where there is a list of pointers. For example, term.cell (which is either a topCell or a libCell)
objList (object list)	List of object pointers (object type or types are enclosed in the "(" & ")").
pt (point)	{coord coord}
pwlList	Piecewise Linear list

rect (rectangle)	{ coord coord coord coord }
string	String

2

Database Objects

antennaData

Parent Object

term

Definition

Antenna information for terminals

Types and Definitions

Child Object or Attribute	Type	Edit	Description
area	area	No	Area value for non *Car type cases, 0 value used for *Car type cases as area is not applicable in those cases.
layer	obj(layer)	No	Pointer to layer. If layer is null(0x0), data applies to all layers.
model	enum	No	Oxide model, none is used for cases where oxide model does not apply. **Legal enum**: none, oxide1, oxide2, oxide3, oxide4
ratio	float	No	Ratio value for *Car type cases. 0 value used for non *Car enums as ratio is not applicable in those cases.
type	enum	No	Type of antenna data. Equivalent to LEF MACRO PIN ANTENNA* constructs. **Legal enum**: libDiffArea, libGateArea, libMaxAreaCar, libMaxCutCar, libMaxSideAreaCar, libPartialCutArea, libPartialMetalArea, libPartialMetalSideArea, topDiffArea, topGateArea, topMaxAreaCar, topMaxCutCar, topMaxSideAreaCar, topPartialCutArea, topPartialMetalArea, topPartialMetalSideArea

Note: In addition to the above entries, every object has an objType attribute

antennaModel

Parent Object

layer

Definition

Antenna model information for one layer & oxide

Types and Definitions

Child Object or Attribute	Type	Edit	Description
areaDiffReduceList	list(pwlList)	No	Indicates that the cut area is multiplied by this factor computed from a piece-wise linear (PWL) interpolation, based on the diffusion area attached to the cut (0x0 indicates that the attribute does not apply), LEF(ANTENNAAREADIFFREDUCEPWL)
areaFactor	float	No	Specifies the multiply factor for the antenna metal area calculation (default value 1.0), LEF(ANTENNAAREAFACTOR)
areaFactorDiffuseOnly	bool	No	Specifies that the current antenna area factor should only be used when the corresponding layer is connected to the diffusion, LEF(DIFFUSEONLY)
areaRatio	float	No	Specifies the maximum legal antenna ratio, using the area of the metal wire that is not connected to the diffusion diode (0 indicates that the attribute does not apply), LEF(ANTENNAAREARATIO)
cumAreaRatio	float	No	Specifies the cumulative antenna ratio, using the area of the wire that is not connected to the diffusion diode (0 indicates that the attribute does not apply), LEF(ANTENNACUMAREARATIO)
cumDiffAreaRatio	float	No	Specifies the cumulative antenna ratio, using the area of the metal wire that is connected to the diffusion diode in explicit ratio (0 indicates that the attribute does not apply), LEF(ANTENNACUMDIFFAREARATIO)
cumDiffAreaRatioList	list(pwlList)	No	Specifies the cumulative antenna ratio, using the area of the metal wire that is connected to the diffusion diode in piece-wise linear format (PWL) (0x0 indicates that the attribute does not apply), LEF(ANTENNACUMDIFFAREARATIO PWL)

cumDiffSideAreaRatio	float	No	Specifies the cumulative antenna ratio, using the side wall area of the metal wire that is connected to the diffusion diode in explicit ratio (0 indicates that the attribute does not apply), LEF(ANTENNACUMDIFFSIDEAREARATIO)
cumDiffSideAreaRatioList	list(pwlList)	No	Specifies the cumulative antenna ratio, using the side wall area of the metal wire that is connected to the diffusion diode in piece-wise linear format (PWL) (0x0 indicates that the attribute does not apply), LEF(ANTENNACUMDIFFSIDEAREARATIO PWL)
cumRoutingPlusCut	bool	No	Indicates that the cumulative ratio rules (ANTENNACUMAREARATIO and ANTENNACUMDIFFAREARATIO) accumulate with the previous cut layer instead of the previous metal layer, LEF(ANTENNACUMROUTINGPLUSCUT)
cumSideAreaRatio	float	No	Specifies the cumulative antenna ratio, using the side wall area of the metal wire that is not connected to the diffusion diode (0 indicates that the attribute does not apply), LEF(ANTENNACUMSIDEAREARATIO)
diffAreaRatio	float	No	Specifies the antenna ratio, using the area of the metal wire connected to the diffusion diode in explicit ratio (0 indicates that the attribute does not apply), LEF(ANTENNADIFFAREARATIO)
diffAreaRatioList	list(pwlList)	No	Specifies the antenna ratio, using the area of the metal wire connected to the diffusion diode in piece-wise linear format (PWL) (0x0 indicates that the attribute does not apply), LEF(ANTENNADIFFAREARATIO PWL)
diffSideAreaRatio	float	No	Specifies the antenna ratio, using the side wall area of the metal wire that is connected to the diffusion diode in explicit ratio (0 indicates that the attribute does not apply), LEF(ANTENNADIFFSIDEAREARATIO)

diffSideAreaRatioList	list(pwlList)	No	Specifies the antenna ratio, using the side wall area of the metal wire that is connected to the diffusion diode in piece-wise linear format (PWL) (0x0 indicates that the attribute does not apply), LEF(ANTENNADIFFSIDEAREARATIO PWL)
gateMinusDiff	float	No	Indicates that the antenna ratio metal area should subtract the diffusion area connected to it (0 indicates that the attribute does not apply), LEF(ANTENNAAREAMINUSDIFF)
gatePlusDiff	float	No	Indicates that the antenna ratio gate area includes the diffusion area multiplied by this factor (0 indicates that the attribute does not apply), LEF(ANTENNAGATEPLUSDIFF)
sideAreaFactor	float	No	Specifies the multiply factor for the antenna metal side wall area calculation (default value 1.0), LEF(ANTENNASIDEAREAFACTOR)
sideAreaFactorDiffuseOnly	bool	No	Specifies that the current antenna side area factor should only be used when the corresponding layer is connected to the diffusion, LEF(DIFFUSEONLY)
sideAreaRatio	float	No	Specifies the antenna ratio, using the side wall area of the metal wire that is not connected to the diffusion diode (0 indicates that the attribute does not apply), LEF(ANTENNASIDEAREARATIO)

Note: In addition to the above entries, every object has an objType attribute

bndry

Parent Object

fPlan, hInst

Definition

Boundary for placement control(fence, region, guide, etc.)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
area	area	No	Total area of the boundry as defined by the LEF MACRO SIZE or OVERLAP information
box	rect	No	Bounding box of the boxes specifing the boundary
box_area	area	No	Area of bounding box of the boxes specifing the boundary
box_ll	pt	No	Lower left (ll) of bounding box of the boxes specifing the boundary
box_llx	coord	No	Lower left X (llx) of bounding box of the boxes specifing the boundary
box_lly	coord	No	Lower left Y (lly) of bounding box of the boxes specifing the boundary
box_size	pt	No	Size of bounding box of the boxes specifing the boundary
box_sizex	coord	No	Size X of bounding box of the boxes specifing the boundary
box_sizey	coord	No	Size Y of bounding box of the boxes specifing the boundary
box_ur	pt	No	Upper right (ur) of bounding box of the boxes specifing the boundary
box_urx	coord	No	Upper Right X (urx) of bounding box of the boxes specifing the boundary
box_ury	coord	No	Upper Right Y (ury) of bounding box of the boxes specifing the boundary
boxes	list(rect)	No	List of rectangles that define the boundary
hInst	obj(hInst)	No	Pointer to parent hierarchical instance (if boundary is for a partition)

isFloating	bool	Yes	Only affects bndry with .type = fence or region. If true, the global placer can move the fence or region. The .box and .boxes value must also be set. The global placer will not change the size of the rect, but may move it.
type	enum	Yes	Fence: only members are allowed inside, region: members are placed inside and other insts can also be inside, guide: suggested placement boundary (placer gives high cost to be outside), cluster: keep members near each other (boxes field is ignored), none: undefined, which means the value was never set ("none" should not be used because it will cause placement errors-you must delete this object if you want it to have no effect). **Legal enum**: cluster, fence, guide, none, region

Note: In addition to the above entries, every object has an objType attribute

bump

Parent Object

topCell

Definition

Bump

Types and Definitions

Child Object or Attribute	Type	Edit	Description
bumpTerms	objList(bump)	No	The terms of the bump. A bump with ALLPINSCONNECTED LEF syntax can have multiple bumpTerms which are internally connected.

bump_shape_bbox	rect	No	Bounding box of the bump shape for a package connection, which is the widest shape found on the top routing layer. If there is more than one wide shape of the same size, the bbox of all the wide shapes is returned
bump_shape_bbox_area	area	No	Area of bounding box of the bump shape for a package connection, which is the widest shape found on the top routing layer. If there is more than one wide shape of the same size, the bbox of all the wide shapes is returned
bump_shape_bbox_ll	pt	No	Lower left (ll) of bounding box of the bump shape for a package connection, which is the widest shape found on the top routing layer. If there is more than one wide shape of the same size, the bbox of all the wide shapes is returned
bump_shape_bbox_llx	coord	No	Lower left X (llx) of bounding box of the bump shape for a package connection, which is the widest shape found on the top routing layer. If there is more than one wide shape of the same size, the bbox of all the wide shapes is returned
bump_shape_bbox_lly	coord	No	Lower left Y (lly) of bounding box of the bump shape for a package connection, which is the widest shape found on the top routing layer. If there is more than one wide shape of the same size, the bbox of all the wide shapes is returned
bump_shape_bbox_size	pt	No	Size of bounding box of the bump shape for a package connection, which is the widest shape found on the top routing layer. If there is more than one wide shape of the same size, the bbox of all the wide shapes is returned

bump_shape_bbox_sizex	coord	No	Size X of bounding box of the bump shape for a package connection, which is the widest shape found on the top routing layer. If there is more than one wide shape of the same size, the bbox of all the wide shapes is returned
bump_shape_bbox_sizey	coord	No	Size Y of bounding box of the bump shape for a package connection, which is the widest shape found on the top routing layer. If there is more than one wide shape of the same size, the bbox of all the wide shapes is returned
bump_shape_bbox_ur	pt	No	Upper right (ur) of bounding box of the bump shape for a package connection, which is the widest shape found on the top routing layer. If there is more than one wide shape of the same size, the bbox of all the wide shapes is returned
bump_shape_bbox_urx	coord	No	Upper Right X (urx) of bounding box of the bump shape for a package connection, which is the widest shape found on the top routing layer. If there is more than one wide shape of the same size, the bbox of all the wide shapes is returned
bump_shape_bbox_ury	coord	No	Upper Right Y (ury) of bounding box of the bump shape for a package connection, which is the widest shape found on the top routing layer. If there is more than one wide shape of the same size, the bbox of all the wide shapes is returned
bump_shape_center	pt	No	The center of the bump shape for a package connection, which is the widest shape found on the top routing layer. If there is more than one wide shape of the same size, the center of all the wide shapes is returned
bump_shape_center_x	coord	No	X of the center of the bump shape for a package connection, which is the widest shape found on the top routing layer. If there is more than one wide shape of the same size, the center of all the wide shapes is returned

bump_shape_center_y	coord	No	Y of the center of the bump shape for a package connection, which is the widest shape found on the top routing layer. If there is more than one wide shape of the same size, the center of all the wide shapes is returned
cell	obj(libCell)	No	Pointer to child cell master
isFixed	bool	No	Indicates that the bump is connected to one term
name	string	No	Name of the bump
net	obj(net)	No	Pointer to net connected to the terminal
orient	enum	Yes	Bump placement orientation **Legal enum**: MX, MX90, MY, MY90, R0, R180, R270, R90
pStatus	enum	Yes	Bump placement status **Legal enum**: cover, fixed, placed, softFixed, unplaced
props	objList(prop)	No	List of pointers to properties
pt	pt	Yes	Location of the bump
pt_x	coord	Yes	X of location of the bump
pt_y	coord	Yes	Y of location of the bump
terms	objList(term)	No	The terms the bump is assigned to.
type	enum	No	Bump terminal type **Legal enum**: analogTerm, asyncCtrlTerm, clockTerm, dQTerm, dTerm, fFQTerm, feedTerm, gatedClockTerm, groundTerm, latchQTerm, normalTerm, powerTerm, rSTerm, triStateTerm

Note: In addition to the above entries, every object has an objType attribute

bumpTerm

Parent Object

bump

Definition

PIN in bump LEF.

Types and Definitions

Child Object or Attribute	Type	Edit	Description
bump	obj(inst)	No	The bump that the bumpTerm belongs to.
cellTerm	obj(term)	No	The equivalent cell terminal.
defName	string	No	DEF name of the bumpTerm.
direction	enum	No	bumpTerm's direction from the corresponding cellTerm. **Legal enum**: bidi, input, output
layer	obj(layer)	No	The layer of the bumpTerm. For bumpTerm with more than one shape, it is the layer of the first shape (which is the same shape used for the .pt value).
name	string	No	The bumpTerm name including bump path.
pt	pt	No	The location of the bumpTerm. For bumpTerm with more than one shape, it is the location of the first shape (which is the same shape used for the .layer value).
pt_x	coord	No	X of the location of the bumpTerm. For bumpTerm with more than one shape, it is the location of the first shape (which is the same shape used for the .layer value).
pt_y	coord	No	Y of the location of the bumpTerm. For bumpTerm with more than one shape, it is the location of the first shape (which is the same shape used for the .layer value).
term	objList(term)	No	The top-level term that the bumpTerm connects to.

Note: In addition to the above entries, every object has an objType attribute

bus

Parent Object

net, term

Definition

Bus

Types and Definitions

Child Object or Attribute	Type	Edit	Description
baseName	string	No	Non-vector portion of the bus name (eg. A for bus A[7:0])
bits	objList(net, term)	No	List of pointers to bits of the bus (homogeneous, only one type per bus object)
lsb	int	No	Least significant bit ([msb:lsb] form)
msb	int	No	Most significant bit ([msb:lsb] form)

Note: In addition to the above entries, every object has an objType attribute

busGuide

Parent Object

fPlan

Definition

Bus Guide

Types and Definitions

Child Object or Attribute	Type	Edit	Description
area	area	No	Area of the busGuide as defined by the LEF MACRO SIZE or OVERLAP information
botLayer	obj(layer)	Yes	Pointer to bottom layer of allowed layer range
box	rect	No	Bounding box of the busGuide
box_area	area	No	Area of bounding box of the busGuide
box_ll	pt	No	Lower left (ll) of bounding box of the busGuide
box_llx	coord	No	Lower left X (llx) of bounding box of the busGuide
box_lly	coord	No	Lower left Y (lly) of bounding box of the busGuide
box_size	pt	No	Size of bounding box of the busGuide
box_sizex	coord	No	Size X of bounding box of the busGuide
box_sizey	coord	No	Size Y of bounding box of the busGuide
box_ur	pt	No	Upper right (ur) of bounding box of the busGuide
box_urx	coord	No	Upper Right X (urx) of bounding box of the busGuide
box_ury	coord	No	Upper Right Y (ury) of bounding box of the busGuide
netGroup	obj(netGroup)	No	Pointer to net group with list of nets to be routed within the busGuide
topLayer	obj(layer)	Yes	Pointer to top layer of allowed layer range
type	enum	Yes	The type of busGuide.Hard specifies the busGuide as a hard constraint for routing. eGR, NR and NRHF should obey the path of bus guide. Soft specifies the busGuide as a soft constraint for routing. eGR, NR and NRHF should be guided by the route path. But tool can route the net out of the bus guide. **Legal enum**: hard, soft

Note: In addition to the above entries, every object has an objType attribute

busSinkGroup

Parent Object

fPlan

Definition

BusSink group

Types and Definitions

Child Object or Attribute	Type	Edit	Description
name	string	No	BusSinkGroup name
sinks	objList(instTerm, term)	No	List of pointers to busSinks (terms and instTerms)

Note: In addition to the above entries, every object has an objType attribute

cellDensity

Parent Object

libCell

Definition

cellDensity

Types and Definitions

Child Object or Attribute	Type	Edit	Description
layer	obj(layer)	No	Pointer to layer
shapes	objList(densityShape)	No	List of pointers to shapes that define cellDensity

Note: In addition to the above entries, every object has an objType attribute

densityShape

Parent Object

cellDensity, pBlkg

Definition

Rectangle with metal/cut or placement density information

Types and Definitions

Child Object or Attribute	Type	Edit	Description
density	float	No	Density value (range: 0.0-1.0)
rect	rect	No	Rectangle. For cellDensity, the coordinates are local to cell, not relative to the design.
rect_area	area	No	Area of rectangle. For cellDensity, the coordinates are local to cell, not relative to the design.
rect_ll	pt	No	Lower left (ll) of rectangle. For cellDensity, the coordinates are local to cell, not relative to the design.
rect_llx	coord	No	Lower left X (llx) of rectangle. For cellDensity, the coordinates are local to cell, not relative to the design.
rect_lly	coord	No	Lower left Y (lly) of rectangle. For cellDensity, the coordinates are local to cell, not relative to the design.
rect_size	pt	No	Size of rectangle. For cellDensity, the coordinates are local to cell, not relative to the design.
rect_sizex	coord	No	Size X of rectangle. For cellDensity, the coordinates are local to cell, not relative to the design.
rect_sizey	coord	No	Size Y of rectangle. For cellDensity, the coordinates are local to cell, not relative to the design.
rect_ur	pt	No	Upper right (ur) of rectangle. For cellDensity, the coordinates are local to cell, not relative to the design.
rect_urx	coord	No	Upper Right X (urx) of rectangle. For cellDensity, the coordinates are local to cell, not relative to the design.
rect_ury	coord	No	Upper Right Y (ury) of rectangle. For cellDensity, the coordinates are local to cell, not relative to the design.

Note: In addition to the above entries, every object has an objType attribute

foreign

Parent Object

libCell

Definition

Foreign

Types and Definitions

Child Object or Attribute	Type	Edit	Description
instOrients	string	No	Orientation of the instance for the cell. If the instance matches one of the orientations in the specified list then the cell name specified in the foreign.name is instantiated during GDSII/Stream and OASIS generation. A value = "none" indicates that the specified FOREIGN applies in all cases that do not have specific .instOrients specified.
name	string	No	Name of the foreign cell (can be the same as the master if only a shift/offset is required)
orient	enum	No	Orientation of the foreign cell (non-R0 values are only allowed if the cell name is different from the master cell) **Legal enum**: MX, MX90, MY, MY90, R0, R180, R270, R90
pt	pt	No	Location (offset) of the foreign cell
pt_x	coord	No	X of location (offset) of the foreign cell
pt_y	coord	No	Y of location (offset) of the foreign cell

Note: In addition to the above entries, every object has an objType attribute

fPlan

Parent Object

topCell

Definition

Floorplan header with pointers to floorplan objects

Types and Definitions

Child Object or Attribute	Type	Edit	Description
allGuiShapes	objList(guiLine, guiRect)	No	List of pointers to guiRect and guiLine objects
area	area	No	Area of the floor plan as defined by the LEF MACRO SIZE or OVERLAP information
bndrys	objList(bndry)	No	List of pointers to boundaries (fence, region, etc.)
box	rect	No	Rectangle that defines the design size
box_area	area	No	Area of rectangle that defines the design size
box_ll	pt	No	Lower left (ll) of rectangle that defines the design size
box_llx	coord	No	Lower left X (llx) of rectangle that defines the design size
box_lly	coord	No	Lower left Y (lly) of rectangle that defines the design size

box_size	pt	No	Size of rectangle that defines the design size
box_sizex	coord	No	Size X of rectangle that defines the design size
box_sizey	coord	No	Size Y of rectangle that defines the design size
box_ur	pt	No	Upper right (ur) of rectangle that defines the design size
box_urx	coord	No	Upper Right X (urx) of rectangle that defines the design size
box_ury	coord	No	Upper Right Y (ury) of rectangle that defines the design size
boxes	list(rect)	No	List of rectangles that define the shape of the floorplan boundary
busGuides	objList(busGuide)	No	List of pointers to bus guide object
busSinkGroups	objList(busSinkGroup)	No	List of pointers to bus sink groups
core2Bot	coord	No	Distance between core edge and its die/io box
core2Left	coord	No	Distance between core edge and its die/io box
core2Right	coord	No	Distance between core edge and its die/io box
core2Top	coord	No	Distance between core edge and its die/io box
coreBox	rect	No	Rectangle that defines the core row area
coreBox_area	area	No	Area of rectangle that defines the core row area
coreBox_ll	pt	No	Lower left (ll) of rectangle that defines the core row area

coreBox_llx	coord	No	Lower left X (llx) of rectangle that defines the core row area
coreBox_lly	coord	No	Lower left Y (lly) of rectangle that defines the core row area
coreBox_size	pt	No	Size of rectangle that defines the core row area
coreBox_sizex	coord	No	Size X of rectangle that defines the core row area
coreBox_sizey	coord	No	Size Y of rectangle that defines the core row area
coreBox_ur	pt	No	Upper right (ur) of rectangle that defines the core row area
coreBox_urx	coord	No	Upper Right X (urx) of rectangle that defines the core row area
coreBox_ury	coord	No	Upper Right Y (ury) of rectangle that defines the core row area
coreSite	obj(site)	No	Pointer to the site object to use during floorplan creation
exclusiveGroupMinGap	coord	Yes	This is the minimum gap should be maintained between exclusiveGroups (all exclusive groups). The value is measured in microns. It can be set by command createExclusiveGroups -min_gap.
flipRows	enum	No	Specification of floorplan row creation, none = no flipping; first = first row is flipped, other rows alternate; second = first row is not flipped, other rows alternate **Legal enum**: first, none, second
gCellGrids	objList(gCellGridDef)	No	List of pointers to gCellGrid definitions
groups	objList(group)	No	List of pointers to groups

guiLines	objList(guiLine)	No	List of pointers to guiLine objects
guiPolys	objList(guiPoly)	No	List of pointers to guiPoly objects
guiRects	objList(guiRect)	No	List of pointers to guiRect objects
guiTexts	objList(guiText)	No	List of pointers to displayText objects
ioBox	rect	No	Rectangle that defines the IO area
ioBox_area	area	No	Area of rectangle that defines the IO area
ioBox_ll	pt	No	Lower left (ll) of rectangle that defines the IO area
ioBox_llx	coord	No	Lower left X (llx) of rectangle that defines the IO area
ioBox_lly	coord	No	Lower left Y (lly) of rectangle that defines the IO area
ioBox_size	pt	No	Size of rectangle that defines the IO area
ioBox_sizex	coord	No	Size X of rectangle that defines the IO area
ioBox_sizey	coord	No	Size Y of rectangle that defines the IO area
ioBox_ur	pt	No	Upper right (ur) of rectangle that defines the IO area
ioBox_urx	coord	No	Upper Right X (urx) of rectangle that defines the IO area
ioBox_ury	coord	No	Upper Right Y (ury) of rectangle that defines the IO area
ios	objList(io)	No	List of pointers to io objects

isCore2Io	bool	No	Indicates whether core2* attributes are measured between core edge (coreBox) and design boundary (box) or between core edge and io box edge (ioBox)
netGroups	objList(netGroup)	No	List of pointers to net groups
numRows	int	No	Number of rows within the core area
pBlkgs	objList(pBlkg)	No	List of pointers to placement blockages
pinBlkgs	objList(ptnPinBlkg)	No	List of pointers to partition pin blockage objects
pinGroups	objList(pinGroup)	No	List of pointers to pin groups
pinGuides	objList(pinGuide)	No	List of pointers to pin guide object
rBlkgs	objList(rBlkg)	No	List of pointers to routing blockages
rHaloBotLayer	obj(layer)	No	Pointer to the bottom layer for which routing halo will be created
rHaloSideSize	coord	Yes	Specifies routing halo inside the design boundary (honored by the signal router). Only positive values are used and indicate the halo is inside of the design boundary.
rHaloTopLayer	obj(layer)	No	Pointer to the top layer for which routing halo will be created
resizeBlkgs	objList(resizeBlkg)	No	List of pointers to sizeBlkg objects
rowSpacing	coord	No	Specification of the floorplan row spacing
rowSpacingType	enum	No	Indicates whether the rowSpacing is applied between each row (1) or between each pair of rows (2) **Legal enum**: 1, 2
rows	objList(row)	No	List of pointers to rows

topCell	obj(topCell)	No	Pointer to cell containing the floorplan data
topSdps	objList(sdp)	No	List of pointers to top data path objects
tracks	objList(trackDef)	No	List of pointers to trackDef

Note: In addition to the above entries, every object has an objType attribute

gCellGridDef

Parent Object

fPlan

Definition

Floorplan track information (DEF TRACKS equivalent)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
dir	enum	No	Specifies the location and direction of the first grid defined. x indicates vertical lines; y indicates horizontal lines. **Legal enum**: x, y
numGrids	int	No	Specifies the number of grid lines to create (number of rows or columns is numGrids-1)
start	coord	No	Specifies the coordinate of the first line
step	coord	No	Specifies the spacing between the grids

Note: In addition to the above entries, every object has an objType attribute

group

Parent Object

fPlan, hInst, inst, pd

Definition

Group of hInsts, insts, or groups

Types and Definitions

Child Object or Attribute	Type	Edit	Description
area	area	No	Area of the group as defined by the LEF MACRO SIZE or OVERLAP information
boxes	list(rect)	No	List of rectangles that define the shape of group
conType	enum	Yes	Fence: only members are allowed inside, region: members are placed inside and other insts can also be inside, guide: suggested placement boundary (placer gives high cost to be outside), cluster: keep members near each other (boxes field is ignored), none: undefined, which means the value was never set ("none" should not be used because it will cause placement errors-you must delete this object if you want it to have no effect). **Legal enum**: cluster, fence, guide, none, region
density	float	No	Group density (legal range: 0.0-1.0)
exclusiveGroupGap	coord	Yes	This is the gap should be maintained between exclusiveGroups (per safety island groups). The value is measured in microns. It can be set by command createExclusiveGroups -gap. It is only valid when the group's conType is region or fence.

exclusiveGroups	string	Yes	This attribute specifies a list of groups that are exclusive of this group. It can be used to implement safety islands in automotive application designs.
isFloating	bool	Yes	Only affects groups with .conType = fence or region. If true, the global placer can move the fence or region. The .boxes value must also be set, and is currently restricted to a single rect. The global placer will not change the size of the rect, but may move it.
members	objList(group, hInst, inst)	No	List of pointers to group's members
name	string	No	Name of group
parent	obj(group)	No	Pointer to the parent group if is a sub-group
pd	obj(pd)	No	Pointer to Power Domain (if group is pd)
props	objList(prop)	No	List of pointers to properties

Note: In addition to the above entries, every object has an objType attribute

guiLine

Parent Object

fPlan

Definition

A line that can be displayed on the GUI, and is not output to DEF.

Types and Definitions

Child Object or Attribute	Type	Edit	Description
guiLayerName	string	No	normally a GUI-only layer name that is not a tech-file layer. If the name is the same as a tech-file layer, it will be drawn on the GUI with other shapes on that layer, but it will not be output to DEF.
props	objList(prop)	No	List of pointers to properties
pts	list(pt)	No	2 points for line

Note: In addition to the above entries, every object has an objType attribute

guiPoly

Parent Object

fPlan

Definition

The polygon shape that can be displayed on the GUI, and is not output to DEF.

Types and Definitions

Child Object or Attribute	Type	Edit	Description
guiLayerName	string	No	normally a GUI-only layer name that is not a tech-file layer. If the name is the same as a tech-file layer, it will be drawn on the GUI with other shapes on that layer, but it will not be output to DEF.
props	objList(prop)	No	List of pointers to properties
pts	list(pt)	No	point list for poly

Note: In addition to the above entries, every object has an objType attribute

guiRect

Parent Object

fPlan

Definition

A rect that can be displayed on the GUI, and is not output to DEF.

Types and Definitions

Child Object or Attribute	Type	Edit	Description
box	rect	No	Bounding box for the rectangle
box_area	area	No	Area of bounding box for the rectangle
box_ll	pt	No	Lower left (ll) of bounding box for the rectangle
box_llx	coord	No	Lower left X (llx) of bounding box for the rectangle
box_lly	coord	No	Lower left Y (lly) of bounding box for the rectangle
box_size	pt	No	Size of bounding box for the rectangle
box_sizex	coord	No	Size X of bounding box for the rectangle
box_sizey	coord	No	Size Y of bounding box for the rectangle
box_ur	pt	No	Upper right (ur) of bounding box for the rectangle
box_urx	coord	No	Upper Right X (urx) of bounding box for the rectangle
box_ury	coord	No	Upper Right Y (ury) of bounding box for the rectangle
guiLayerName	string	No	normally a GUI-only layer name that is not a tech-file layer. If the name is the same as a tech-file layer, it will be drawn on the GUI with other shapes on that layer, but it will not be output to DEF.
props	objList(prop)	No	List of pointers to properties

Note: In addition to the above entries, every object has an objType attribute

guiText

Parent Object

fPlan

Definition

The text that can be displayed on the GUI, and is not output to DEF.

Types and Definitions

Child Object or Attribute	Type	Edit	Description
guiLayerName	string	No	normally a GUI-only layer name that is not a tech-file layer. If the name is the same as a tech-file layer, it will be drawn on the GUI with other shapes on that layer, but it will not be output to DEF.
height	coord	No	Text height
label	string	No	Text string
props	objList(prop)	No	List of pointers to properties
pt	pt	No	Text location (lower left)
pt_x	coord	No	X of text location (lower left)
pt_y	coord	No	Y of text location (lower left)

Note: In addition to the above entries, every object has an objType attribute

head

Parent Object

No Parents

Definition

Root/Head of the database

Types and Definitions

Child Object or Attribute	Type	Edit	Description
allCells	objList(libCell, ptnCell, topCell, vCell)	No	List of pointers to cells of all types (library cells and design cells)
bottomRoutingLayer	string	Yes	Lowest lef layer name for global and detail routing
cellEdgeSpacings	list(list)	No	List of cell edge spacing values, list form: {cell_edge_name1 cell_edge_name2 spacing_coord}.
dbUnits	int	No	Database units per user unit
eeqVariantSiteStart	int	Yes	Site variant_id of the first site on rows in core area. This is specified in library LEF58_CELLVARIANTS property by value of STARTVARIANT in 'CELLVARIANTS totalNum [STARTVARIANT num] YFLIPMAP {flippedVariantNum siteVariantNum} ...'
eeqVariantYFlipTable	list(list)	No	The LEF YFLIPMAP table defined in LEF58_CELLVARIANTS property as 'CELLVARIANTS variantTotalNum YFLIPMAP {flippedVariantNum siteVariantNum} ...'
finGridDirection	enum	Yes	Returns a string(horizontal, vertical) indicating the direction of the FINFET grid as described in LEF **Legal enum**: Horizontal, Vertical
finGridOffset	coord	Yes	Returns a single floating point number indicating the offset of the FINFET grid as described in LEF
finGridPitch	coord	Yes	Returns a single floating point number indicating the pitch of the FINFET grid as described in LEF

instMaskShiftLayers	objList(layer)	No	Ordered list of layer pointers to layers that are defined for instance mask shifting. Equivalent to DEF COMPONENTMASK shift ordering of layers.
layers	objList(layer)	No	List of pointers to layers
libCells	objList(libCell)	No	List of pointers to library cells
mfgGrid	coord	No	Manufacturing grid
numCellEeqVariants	int	No	Maximum number of EEQ variants of any cell in library, from LIBRARY LEF58_CELLVARIANTS property 'CELLVARIANTS variantTotalNum YFLIPMAP {flippedVariantNum siteVariantNum} ...'
props	objList(prop)	No	List of pointers to properties
ptnCells	objList(ptnCell)	No	List of pointers to partition cells
routeTypes	objList(routeType)	No	List of pointers to routeType rule objects
rules	objList(rule)	No	List of pointers to non-default rule objects
sites	objList(site)	No	List of pointers to sites
topCells	objList(topCell)	No	List of pointers to top design cells
topRoutingLayer	string	Yes	Highest lef layer name for global and detail routing
vCells	objList(vCell)	No	List of pointers to Verilog cells
viaRuleGenerates	objList(viaRuleGenerate)	No	All the viaRuleGenerate rules defined in the LEF or OpenAccess techfile
vias	objList(via)	No	List of pointers to via master

Note: In addition to the above entries, every object has an objType attribute

hInst

Parent Object

bndry, group, hInstTerm, hTerm, inst, ptn, topCell, vCell

Definition

Hierarchical instance (derived from netlist)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
allInsts	objList(hInst, inst)	No	List of pointers to all instances and hierarchical instances in the level referred to by the hInst
allTreeInsts	objList(hInst, inst)	No	List of pointers to all instances and hierarchical instances in the hInst
area	area	No	Area of the hInst as defined by the LEF MACRO SIZE or OVERLAP information
bndry	obj(bndry)	No	Pointer to the boundary
box	rect	No	Bounding box of the boxes
box_area	area	No	Area of bounding box of the boxes
box_ll	pt	No	Lower left (ll) of bounding box of the boxes
box_llx	coord	No	Lower left X (llx) of bounding box of the boxes
box_lly	coord	No	Lower left Y (lly) of bounding box of the boxes
box_size	pt	No	Size of bounding box of the boxes
box_sizex	coord	No	Size X of bounding box of the boxes
box_sizey	coord	No	Size Y of bounding box of the boxes

box_ur	pt	No	Upper right (ur) of bounding box of the boxes
box_urx	coord	No	Upper Right X (urx) of bounding box of the boxes
box_ury	coord	No	Upper Right Y (ury) of bounding box of the boxes
boxes	list(rect)	No	List of rectangles that define the boundary of hInst
cell	obj(vCell)	No	Pointer to module cell that corresponds to the hInst
defName	string	No	Fully qualified def name of the hInst from db
dontTouch	enum	Yes	This attributes defines the user preservation status of the hinst during optimization. Setting this attribute will set the dont_touch attribute on the parent module of this hinst and all hinsts of the same module. This setting will apply to all insts within the hinst unless overridden at a lower level hinst or on the inst object itself. The dont_touch_effective attribute on each child inst and hinst will return the resolved value. **Legal enum**: constPropDeleteOk, constPropSizeDeleteOk, deleteOk, false, none, sizeDeleteOk, sizeOk, sizeSameFootprintOk, sizeSameHeightOk, true
dontTouchEffective	enum	No	This attribute defines the effective preservation status of this hinst during optimization from the dont_touch_sources values. If the partition source is 'true' or the ilm source is 'true', then the effective value for the hinst is 'true'. Otherwise, the user value has precendence. If the user value is false, then the parent value is returned. **Legal enum**: constPropDeleteOk, constPropSizeDeleteOk, deleteOk, false, none, sizeDeleteOk, sizeOk, sizeSameFootprintOk, sizeSameHeightOk, true
dontTouchHports	enum	Yes	This attribute defines the user preservation status for the hports of this hinst during optimization. **Legal enum**: addInvertOk, addOk, deleteOk, false, invertOk, mapSizeOk, none, true

dontTouchSources	string	No	Dictionary of {source <value>} pairs contributing to the dont_touch_effective attribute for this object : `user <value>} {parent <value>} {read_only <value>} {parent_read_only <value>.`
dontUseCells	list(string)	Yes	List of cell names (wildcards supported) to disallow for this hinst during optimization. Setting on an hinst sets the attribute on the module and hinsts sharing the module. Overrides any library dont_use values. If this list is empty then the closest parent hinst that has a non-empty list is used. If I cell is added to this list that is already in the .use_cells list, it will be removed from the .use_cells list.
dontUseCellsEffective	string	No	The resolved list of all cell names to disallow during optimization for this hinst, based on the library dont_use and the dont_use_cells and use_cells attributes of this hinst or the closest parent hinst with a non-empty list. The precedence is: union of the use_cells of this hinst (or closest parent if empty), then dont_use_cells of this hinst (or closets parent if empty), then the library dont_use setting. When there are multiple hinsts that share the same module, the dontTouchEffective is calculated for the master hinst and the other (clone) hinsts inherit.
group	obj(group)	No	Pointer to the parent group
hInstTerms	objList(hInst)	No	List of pointers of the hierarchical terminals
hInsts	objList(hInst)	No	List of pointers to all hierarchical instances in the level referred to by the hInst
hNets	objList(hNet)	No	List of pointers to hierarchical nets in the hInst
ilmInst	obj(inst)	No	Specifies the inst object of this ilm when it is under unflatten view. This attribute is only valid when the ilm is under flatten view.
insts	objList(inst)	No	List of pointers to all instances in the level referred to by the hInst

isIlm	bool	No	This attribute is true if the hinst is an ILM. It will affect the readOnlyEffective and dontTouchEffective of this hinst and all hinsts within it, as well as the dontTouchEffective of all insts within it. It cannot be overridden by other hinst or inst values.
name	string	No	Fully qualified (path) name of the hInst
parent	obj(hInst)	No	Pointer to the parent hInst object from the current hInst
props	objList(prop)	No	List of pointers to properties
ptn	obj(ptn)	No	Pointer to the partition
readOnly	enum	No	This attribute is true if the hinst is a read_only. This is set by the set_module_view command. When true, this hinst cannot be optimized and cells inside will not be moved. setThis attribute will affect the dontTouchEffective and pStatusEffective attributes on all insts and hinsts within this hinst. It cannot be overridden by other hinst or inst values. **Legal enum**: false, none, true
readOnlyEffective	bool	No	This attribute defines the read_only status of this hinst. This can be true when the local .read_only attribute is true or if a parent .read_only attribute is true. The hinst cannot be optimized and cells inside will not be moved.
treeHInsts	objList(hInst)	No	List of pointers to all hierarchical instances in the hInst
treeInsts	objList(inst)	No	List of pointers to all instances in the hInst

useCells	list(string)	Yes	List of cell names to allow for this hinst during optimization. Setting on an hinst sets the attribute on the module and all hinst siblings inherit. All lib_cells of each base_cell will be allowed. Overrides cells in the dont_use_cells list and any library dont_use values. If this list is empty then the closest parent hinst that has a non-empty list is used. Any parent hinst settings are ignored if this list is defined (there is no merging of lists).

Note: In addition to the above entries, every object has an objType attribute

hInstTerm

Parent Object

hInst, hNet

Definition

Hierarchical instance terminal (seen from level above)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
defName	string	No	Fully qualified def name of the hierarchical instance terminal from db
dontTouch	enum	Yes	The preservation status of an hpin during optimization. A preserved hpin means the logical function of the hpin must be preserved to maintain a simulation or test-point hpin in the netlist. However, the name does not need to be preserved. **Legal enum**: addInvertOk, addOk, deleteOk, false, invertOk, none, true

dontTouchEffective	enum	No	This attribute defines the effective (most pessimistic) preservation status of an hpin during optimization based on the .dont_touch_sources. **Legal enum**: addInvertOk, addOk, deleteOk, false, invertOk, mapSizeOk, none, true
dontTouchSources	string	No	Dictionary of {source value>} pairs contributing to the dont_touch_effective attribute for this object : {user <value>} {power_intent <value>}.
downHNet	obj(hNet)	No	Ponter to the hierarchical net object below
hInst	obj(hInst)	No	Pointer to the hierarchical instance
hTerm	obj(hTerm)	No	Pointer to the hierarchical terminal
isTieHi	bool	No	Indicates that the hierarchical instance terminal is a tieHi
isTieLo	bool	No	Indicates that the hierarchical instance terminal is a tieLo
layer	obj(layer)	Yes	Pointer to layer of the instance terminal (yellow square in display)
name	string	No	Fully qualified (path) name of the hierarchical instance terminal
net	obj(net)	No	Pointer to the assoicated canonical (flat) net
props	objList(prop)	No	List of pointers to properties
pt	pt	Yes	Location of instance terminal (yellow square in display)
pt_x	coord	Yes	X of location of instance terminal (yellow square in display)
pt_y	coord	Yes	Y of location of instance terminal (yellow square in display)
term	obj(term)	No	Pointer to terminal
upHNet	obj(hNet)	No	Pointer to the hierarchical net object above

Note: In addition to the above entries, every object has an objType attribute

hNet

Parent Object

hInst, hInstTerm, hTerm, instTerm, net

Definition

Hierarchical net (derived from netlist)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
allTerms	objList(hInstTerm, hTerm, instTerm)	No	List of pointers to connections on the net
defName	string	No	Fully qualified def name of the hNet from db
dontTouch	enum	Yes	This attribute defines the preservation status of an hnet during optimization. Setting this will preserve all connections on the hnet at the level of hierarchy where the hnet exists (i.e. will stop at the hpins and hports connected to this hnet). **Legal enum**: deleteOk, false, true
hInstTerms	objList(hInstTerm)	No	List of pointers to hInstTerm connections on the net
hTerms	objList(hTerm)	No	List of pointers to hTerm connections on the net
instTerms	objList(InstTerm)	No	List of pointers to instTerm connections on the net
isGnd	bool	No	Indicates that net is ground
isPwrOrGnd	bool	No	Indicates that net is power or ground
name	string	No	Net name (local)
net	obj(net)	No	Pointer to canonical (flat) net associated with the hierarchical net
props	objList(prop)	No	List of pointers to properties

Note: In addition to the above entries, every object has an objType attribute

hTerm

Parent Object

hInstTerm, hNet

Definition

Hierarchical terminal (seen from level below)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
defName	string	No	Fully qualified def name of the hierarchical terminal from db
downHNet	obj(hNet)	No	Pointer to hierarchical net below
hInst	obj(hInst)	No	Pointer to hInst containing the hTerm
name	string	No	Terminal name (local)
net	obj(net)	No	Pointer to associated canonical (flat) net
props	objList(prop)	No	List of pointers to properties
term	obj(term)	No	Pointer to terminal
upHNet	obj(hNet)	No	Pointer to hierarchical net above

Note: In addition to the above entries, every object has an objType attribute

inst

Parent Object

bumpTerm, group, hInst, instTerm, io, pBlkg, pgInstTerm, ptn, rBlkg, sdp, topCell

Definition

Instance - canonical (flat), equivalent to DEF COMPONENT. Points to a libCell or ptnCell.

Types and Definitions

Child Object or Attribute	Type	Edit	Description
area	area	No	Area of the instance as defined by the LEF MACRO SIZE or OVERLAP information
box	rect	No	Bounding box of the instance
box_area	area	No	Area of bounding box of the instance
box_ll	pt	No	Lower left (ll) of bounding box of the instance
box_llx	coord	No	Lower left X (llx) of bounding box of the instance
box_lly	coord	No	Lower left Y (lly) of bounding box of the instance
box_size	pt	No	Size of bounding box of the instance
box_sizex	coord	No	Size X of bounding box of the instance
box_sizey	coord	No	Size Y of bounding box of the instance
box_ur	pt	No	Upper right (ur) of bounding box of the instance
box_urx	coord	No	Upper Right X (urx) of bounding box of the instance
box_ury	coord	No	Upper Right Y (ury) of bounding box of the instance
boxes	list(rect)	No	List of rectangles that define the shape of instance
cell	obj(libCell, ptnCell)	No	Pointer to child cell master or ptnCell
defName	string	No	Fully qualified def name of the instance from db
dontMergeMultibit	bool	Yes	This attribute is denotes whether the instance can be merged during multibit optimization, true indicates cannot merge, false indicates can be merged.
dontSplitMultibit	bool	Yes	This attribute is denotes whether the instance can be split (unmerged) during multibit optimization, true indicates cannot split, false indicates can be split

dontTouch	enum	Yes	This attribute defines the user preservation status of an instance during optimization. **Legal enum**: constPropDeleteOk, constPropSizeDeleteOk, deleteOk, false, mapSizeOk, none, sizeDeleteOk, sizeOk, sizeSameFootprintOk, sizeSameHeightOk, true
dontTouchEffective	enum	No	This attribute defines the effective (most pessimistic) preservation status of an instance during optimization based on the 'sources'. **Legal enum**: constPropDeleteOk, constPropSizeDeleteOk, deleteOk, false, mapSizeOk, none, sizeDeleteOk, sizeOk, sizeSameFootprintOk, sizeSameHeightOk, true
dontTouchSources	string	No	Dictionary of {source value>} pairs contributing to the dont_touch_effective attribute for this object : {user <value>} {lib <value>} {parent <value>} {scan <value>}
group	obj(group)	No	Pointer to the group that contains the instance
hInst	obj(hInst)	No	Pointer to the parent hierarhical instance of the current instance
instTerms	objList(inst)	No	List of pointers to instance terminals
isAlwaysOn	bool	No	Indicates that the instance is always on
isAlwaysOnBuffer	bool	No	Indicates that the instance is a MSV inst and its master cell type is buffer
isFixedMask	bool	No	Indicates the libCell of the inst has FIXEDMASK keyword in LEF, so mask-shifting is not allowed, except for the layers with LAYERMASKSHIFT keyword.
isHaloBlock	bool	No	Indicates that the instance is a halo block
isInsideIlm	bool	No	This attribute denotes whether the inst is the child of a parent hinst the has an ILM specified for it.
isIsolation	bool	No	Indicates that the instance is isolation
isJtagElem	bool	No	Indicates that the instance is a Jtag element

isLevelShifter	bool	No	Indicates that the instance is level shifter
isPhysOnly	bool	No	Indicates that the instance is physical only
isSpareGate	bool	Yes	Indicates that the instance is a spare gate
lithoHalo	bool	Yes	A lithohalo on an inst forces parallelrouting away from the block boundary to meet lithography DRC rules on a few routing layers but allows perpendicular access to pins. It is only allowed if LEF LITHOMACROHALO values exists for some routing layers. See the LEF manual for figures and more details of this DRC rule. It cannot be added to standard cells (cells with a CLASS CORE SITE).
maskShift	string	Yes	Digit encoded value indicates the mask shifting for the instance contents (0 = unshifted, for other shift cases refer to the DEF COMP + MASKSHIFT documentation).
name	string	No	Fully qualified (path) name of the instance
orient	enum	Yes	Instance placement orientation **Legal enum**: MX, MX90, MY, MY90, R0, R180, R270, R90
pHaloBot	coord	Yes	Specifies extra spacing around the inst that should not be used for placement
pHaloBox	rect	No	Bounding box of the inst placement halo
pHaloBox_area	area	No	Area of bounding box of the inst placement halo
pHaloBox_ll	pt	No	Lower left (ll) of bounding box of the inst placement halo
pHaloBox_llx	coord	No	Lower left X (llx) of bounding box of the inst placement halo
pHaloBox_lly	coord	No	Lower left Y (lly) of bounding box of the inst placement halo
pHaloBox_size	pt	No	Size of bounding box of the inst placement halo
pHaloBox_sizex	coord	No	Size X of bounding box of the inst placement halo

pHaloBox_sizey	coord	No	Size Y of bounding box of the inst placement halo
pHaloBox_ur	pt	No	Upper right (ur) of bounding box of the inst placement halo
pHaloBox_urx	coord	No	Upper Right X (urx) of bounding box of the inst placement halo
pHaloBox_ury	coord	No	Upper Right Y (ury) of bounding box of the inst placement halo
pHaloLeft	coord	Yes	Specifies extra spacing around the inst that should not be used for placement
pHaloPoly	list(pt)	No	Specifies the polygon shape of the inst placement halo.
pHaloRight	coord	Yes	Specifies extra spacing around the inst that should not be used for placement
pHaloTop	coord	Yes	Specifies extra spacing around the inst that should not be used for placement
pStatus	enum	Yes	This attribute is the placement status of an instance during placement and optimization. The placer will look at both pStatus and pStatusCTS and use the more restrictive value. **Legal enum**: cover, fixed, placed, softFixed, unplaced
pStatusCTS	enum	Yes	This attribute is the CTS placement status of an instance during placement and optimization. The placer will look at both pStatus and pStatusCTS and use the more restrictive value. **Legal enum**: fixed, softFixed, unset
pStatusEffective	enum	No	This attribute is the effective placement status for the instance. It is the worst case of the instance pStatus, instance pStatusCTS, and hinst parent read only attribute (closest hinst above that is not set to none). **Legal enum**: cover, fixed, placed, softFixed, unplaced

pd	obj(pd)	No	Pointer to the power domain that the instance should be placed within (equivalent to Design Browser locPD)
pgInstTerms	obj(pgInstTerm)	No	List of pgInstTerms for the inst.
props	objList(prop)	No	List of pointers to properties
pt	pt	Yes	Location of the instance
pt_x	coord	Yes	X of location of the instance
pt_y	coord	Yes	Y of location of the instance
rHaloBotLayer	obj(layer)	Yes	Pointer to the bottom layer for which routing halo will be created
rHaloPoly	list(pt)	No	Specifies the polygon shape of the inst routing halo.
rHaloSideSize	coord	Yes	Specifies routing halo around the inst (honored by signal router). Only positive values are used and indicate the halo is outside of the instance boundary.
rHaloTopLayer	obj(layer)	Yes	Pointer to the top layer for which routing halo will be created
sdp	obj(sdp)	No	Return parent sdp group the instance belongs to.
useCells	list(string)	Yes	This attribute is the list of allowable cells that this instance can be resized to. Wildcards are supported. If set, any parent (hinst) .dontUseCellsEffective values are ignored. This attribute only applies to operations on mapped designs. The .dontTouchEffective values sizeSameHeightOk and sizeSameFootprintOk will filter this list further based on height and footprint, respectively.

Note: In addition to the above entries, every object has an objType attribute

instTerm

Parent Object

busSinkGroup, hNet, inst, net

Definition

Instance terminal (used in flattened connectivity)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
cellTerm	obj(term)	No	Pointer to equivalent cell terminal
defName	string	No	Fully qualified def name of the instance terminal from db
dontTouch	enum	Yes	The preservation status of a pin during optimization. A preserved pin means the logical function of the pin must be preserved to maintain a simulation or test-point pin in the netlist. However, the name does not need to be preserved.. **Legal enum**: deleteOk, false, invertOk, none, true
effectiveStackVia	string	No	Reads the effective stack via name for this instterm, which is what the router would use if the design were routed now.
effectiveStackViaRule	obj(stackViaRule)	No	The stackViaRule that is expected to be used by the router for connecting to this instTerm. This value is derived from the other instTerm and term attributes related to stack via.

hNet	obj(hNet)	No	Pointer to the hierarchical net (hNet) connected to the instance term
hdlName	string	Yes	This is the original RTL name for this instterm. It is used to map RTL simulation results with RTL names to the current netlist for switching activity analysis. It is only maintained properly on the output pins of sequential cells. Optimization will copy this name during any multi-bit merge or splitting transforms to the equivalent instterm, but not on output pins of combinational cells. The is_phase_inverted attribute will be flipped if the phase is inverted.
initialName	string	Yes	This will be used for verification of initial input netlist (post-synthesis) to any other netlist generated during Innovus flow.
inst	obj(inst)	No	Pointer to Instance containing the instTerm
isHdlPhaseInverted	bool	Yes	This indicates if the original RTL phase has been inverted. It is combined with the .hdlName attribute to map RTL simulation results to the current netlist for switching activity analysis.
isInitialPhaseInverted	bool	Yes	This indicates if the initial RTL phase has been inverted.
isInput	bool	No	Indicates that the terminal is an input
isOutput	bool	No	Indicates that the terminal is an output
isSGNC	bool	No	Indicates tie connection was made by Single Inst GNC rule
isSpecial	bool	Yes	Indicates that the instTerm belongs in the DEF SPECIALNETS section (0 = NETS section, 1 = SPECIALNETS section)
isTieHi	bool	No	Indicates that the terminal is a tieHi
isTieLo	bool	No	Indicates that the terminal is a tieLo

layer	obj(layer)	No	Pointer to layer of the instance terminal (yellow square in display)
name	string	No	Fully qualified instance terminal name including instance path
net	obj(net)	No	Pointer to net connected to the instTerm
pd	obj(pd)	No	Pointer to the parent power domain of the instance terminal (equivalent to Design Browser effPD)
props	objList(prop)	No	List of pointers to propertie.
pt	pt	No	Location of instance terminal (yellow square in display)
pt_x	coord	No	X of location of instance terminal (yellow square in display)
pt_y	coord	No	Y of location of instance terminal (yellow square in display)
stackViaRequired	bool	Yes	Sets whether a stack via is required for this instTerm.
stackViaRule	obj(stackViaRule)	Yes	This value serves as a instTerm-specific override to use this stackViaRule when connecting to this instTerm. If set, it must match one of the elements from the stackViaRuleList attribute of the corresponding term. The interpretation of this value depends on this instTerm's stackViaRuleRequired attribute.
stackViaRule == {} && stackViaRuleRequired == false	Types and Definitions	No	
stackViaRule == stackrule1 && stackViaRuleRequired == false	Router will prefer stackrule1	No	but it may select another (or possibly none) if necessary to avoid design rule violations.

stackViaRule == {} && stackViaRuleRequired == true	The required stackViaRule is empty	No	so the router will not use any stackViaRule for connecting to this instTerm (even ignoring stackViaRuleRequired on the term if necessary).
stackViaRule == stackrule1 && stackViaRuleRequired == true	Router will use stackrule1	No	even if it leads to design rule violations.
stackViaRuleRequired	bool	Yes	Specifies whether the router must use this instTerm's stackViaRule value. If false, the stackViaRule value will be preferred by the router, if true the stackViaRule value is required by the router. See the stackViaRule attribute for more details.

Note: In addition to the above entries, every object has an objType attribute

io

Parent Object

fPlan

Definition

IO object for block (term) or chip design (inst) constraints

Types and Definitions

Child Object or Attribute	Type	Edit	Description
area	area	No	Area of the instance as defined by the LEF MACRO SIZE or OVERLAP information
box	rect	No	Bounding box of the instance (only used when 'type = inst')

box_area	area	No	Area of bounding box of the instance (only used when 'type = inst')
box_ll	pt	No	Lower left (ll) of bounding box of the instance (only used when 'type = inst')
box_llx	coord	No	Lower left X (llx) of bounding box of the instance (only used when 'type = inst')
box_lly	coord	No	Lower left Y (lly) of bounding box of the instance (only used when 'type = inst')
box_size	pt	No	Size of bounding box of the instance (only used when 'type = inst')
box_sizex	coord	No	Size X of bounding box of the instance (only used when 'type = inst')
box_sizey	coord	No	Size Y of bounding box of the instance (only used when 'type = inst')
box_ur	pt	No	Upper right (ur) of bounding box of the instance (only used when 'type = inst')
box_urx	coord	No	Upper Right X (urx) of bounding box of the instance (only used when 'type = inst')
box_ury	coord	No	Upper Right Y (ury) of bounding box of the instance (only used when 'type = inst')
indent	coord	No	Indent of instance from the boundary of the design
inst	obj(inst)	No	Pointer to IO instance (null/0x0 for block design case)
isAssigned	bool	No	Indicates that the IO location has been set by the IO placer
isCorner	bool	No	Indicates that the IO refers to a corner cell
isGapFixed	bool	No	Indicates that the IO spacing constraint is applied
isGround	bool	No	Indicates that the IO is a Ground
isOffsetFixed	bool	No	Indicates that the IO offset constraint is applied
isPower	bool	No	Indicates that the IO is a Power

name	string	No	IO name, same as term name or inst name depending on design style
offset	coord	No	IO relative location from left (North & South) or bottom (East & West) die edge
order	int	No	IO order per side. Order is left-to-right (North & South) or bottom-to-top (East & West)
row	int	No	IO row/ring number (0 is outtermost IO row)
side	enum	No	Side constraint of IO **Legal enum**: East, None, North, South, West
spacing	coord	No	Spacing between IO and previous IO (left or below)
term	obj(term)	No	Pointer to IO terminal (only used when 'type = term')
type	enum	No	Type of IO (endSpace indicates from last IO to corner, inst indicates real instance case, obs indicates obstruction between IOs, term indicates block design case) **Legal enum**: endSpace, inst, obs, term

Note: In addition to the above entries, every object has an objType attribute

layer

Parent Object

antennaData, bumpTerm, busGuide, cellDensity, fPlan, hInstTerm, head, inst, instTerm, layerRule, layerShape, marker, net, pWire, pinGuide, pinShape, ptn, ptnPinBlkg, rBlkg, routeType, sWire, term, text, trackDef, vWire, via, viaRuleGenerate, whatIfVia, whatIfWire, wire

Definition

Layer (from technology source, LEF or OpenAccess)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
antennaModel1	obj(antennaModel)	No	Pointer to antenna model object for Oxide1
antennaModel2	obj(antennaModel)	No	Pointer to antenna model object for Oxide2
antennaModel3	obj(antennaModel)	No	Pointer to antenna model object for Oxide3
antennaModel4	obj(antennaModel)	No	Pointer to antenna model object for Oxide4
area	area	No	Layer minimum area from LEF/OpenAccess. If AREA rule is not specified in LEF, the value of -1 in dbu will be returned.
backside	bool	No	Indicates that the layer is a backside (underside of the die) layer.
densityStepX	coord	No	Layer density window step from LEF/OpenAccess
densityStepY	coord	No	Layer density window step from LEF/OpenAccess
densityWindowX	coord	No	Layer density window length from LEF/OpenAccess
densityWindowY	coord	No	Layer density window width from LEF/OpenAccess
direction	enum	No	Layer pref. direction from LEF/OpenAccess **Legal enum**: Diag135, Diag45, HVUnassigned, Horizontal, Vertical
extName	string	No	Layer name from LEF/OpenAccess technology file definition.

fillActiveSpacing	coord	No	Layer fill minimum spacing from LEF/OpenAccess
fillGapSpacing	coord	No	Layer fill to fill spacing from LEF/OpenAccess
maxDensity	float	No	Layer fill maximum density from LEF/OpenAccess
maxWidth	coord	No	Layer maximum wire width from LEF/OpenAccess
mfgGrid	coord	No	Manufacturing grid
minDensity	float	No	Layer fill minimum density from LEF/OpenAccess
minSpacing	coord	No	Layer minimum spacing from LEF/OpenAccess
minWidth	coord	No	Layer minimum wire width from LEF/OpenAccess
name	string	No	By default is the LEF or OA layer name. If the layerNameNoAbbreviation global is set to 0, then .name uses an older style abbreviation that is no longer recommended (M1 for first routing layer, M2 for second routing layer, etc.).
num	int	No	Layer number for routing layers (fist routing layer = 1, second = 2, etc). All layers that are not type routing or have backside = 1 will return 0.
numMasks	int	No	Indicates how many masks will be used for the layer (1 = single mask, 2 = double-patterning, 3 = triple-patterning)

offsetX	coord	No	Layer offset X from LEF/OpenAccess
offsetY	coord	No	Layer offset Y from LEF/OpenAccess
pitchX	coord	No	Layer wire pitch X from LEF/OpenAccess
pitchY	coord	No	Layer wire pitch Y from LEF/OpenAccess
props	objList(prop)	No	List of pointers to properties
spacingTables	list(list)	No	List of spacing table in LEF format
type	enum	No	Layer type (cut, routing, etc.) **Legal enum**: MIMCap, MIMCapCut, TSV, aboveDieEdge, belowDieEdge, cut, cutRegion, diffusion, ignore, implant, invalid, masterslice, nWell, overlap, pWell, passivation, polyRouting, region, routing, routingRegion, stackedDie, trimMetal, trimMetalRegion, trimPoly
width	coord	No	Layer wire width from LEF/OpenAccess
wrongwayMinWidth	coord	No	min width in the non-preferred direction (from LEF LAYER MINWIDTH WRONGDIRECTION). A value of 0 indicates that there is no special value assigned for a wrong direction behavior.

wrongwaySpacing	coord	No	min spacing in the non-preferred direction (from LEF LAYER SPACING WRONGDIRECTION). A value of 0 indicates that there is no special value assigned for a wrong direction behavior.
wrongwayWidth	coord	No	min width in the non-preferred direction (from LEF LAYER WIDTH WRONGDIRECTION). A value of 0 indicates that there is no special value assigned for a wrong direction behavior.
wspOaWidthSpacingPattern	string	No	A list of Tcl dict style parameters that match the OA widthSpacingPattern parameters like this: {name <string> is_from_lib <bool> offset <coord>}. The 'name' and 'offset' match the corresponding OA parameters, and 'is_from_lib <bool>' is true if it is from the OA tech graph and false if it is from the cellview.

wspOaWidthSpacingSnapPatternDef	string	No	A list of Tcl dict style parameters that match the OA widthSpacingSnapPatternDef parameters like this: {name <string> is_from_lib <bool> offset <coord> offset_reference <string> wire_type <name> purpose <name>}. The 'name', 'offset', 'wire_type', and 'purpose' match the corresponding OA parameters. 'offset_reference' indicates whether the 'offset' is from the lower-left corner of the boundary or the origin. 'is_from_lib <bool>' is true if it is from the OA tech graph and false if it is from the cellview.
wspOffset	coord	No	Offset from lower-left corner of the core box to the first track. For a horizontal routing layer track, this is a Y offset.

wspPattern	list(string)	No	This is set by add_tracks or by reading an OA design with widthSpacingPatterns defined. It is a list of {width pitch repeat} values that created the tracks. For example, {1.5 2.5 1} {1.0 2.0 3} means a track of 1.5 width, 2.5 track-to-track pitch repeated 1 time, and then a track with width 1.0, track-to-track pitch 2.0 repeated 3 times, then the full pattern repeats. If this is not an SADP layer, there is no width assigned to the track, and a width value of 0.0 is returned. For a horizontal routing layer, the width and pitch are Y values.
wspPatternMasks	list(int)	No	A list of mask values for each track in the wsp_pattern after the pattern repeat sections are expanded.

Note: In addition to the above entries, every object has an objType attribute

layerRule

Parent Object

rule

Definition

Layer Rule

Types and Definitions

Child Object or Attribute	Type	Edit	Description
layer	obj(layer)	No	Pointer to layer
spacing	coord	No	min spacing (from NONDEFAULTRULE LAYER SPACING, LEF LAYER SPACING or SPACINGTABLE)
width	coord	No	wire width (from NONDEFAULTRULE LAYER WIDTH or LEF LAYER WIDTH)

Note: In addition to the above entries, every object has an objType attribute

layerShape

Parent Object

libCell, pin

Definition

layerShape

Types and Definitions

Child Object or Attribute	Type	Edit	Description
isExceptPGNet	bool	No	Indicates that Power/Ground routing is ignored when checking for DRC violations (including shorts) involving the current shape (equivalent to LEF MACRO OBS LAYER EXCEPTPGNET)
layer	obj(layer)	Yes	Pointer to layer of blockage
shapes	objList(shape)	No	List of pointers to shapes that define the blockage area
spacing	coord	No	LEF OBS SPACING equivalent minSpacing value, -1 if not specified in LEF.

Note: In addition to the above entries, every object has an objType attribute

libCell

Parent Object

bump, head, inst, term

Definition

Library cell (from LEF or OpenAccess)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
abstractLib	string	No	Library name for the OpenAccess abstract for the cell (LEF MACRO equivalent)

abstractView	string	No	View name for the OpenAccess abstract for the cell (LEF MACRO equivalent)
allObstructions	objList(layerShape, shapeVia)	No	List of pointers to layerShapes and shapeVias that define cell obstruction geometries.
baseClass	enum	No	Class type: derived from LEF/OpenAccess (refer to documentation for complete mapping) **Legal enum**: block, core, corner, cover, none, pad
bottomEdgeName	string	No	Name of cell edge type for the bottom edge of the cell (R0/N orientation), used to indicate which cells need extra spacing to other cells
bottomPadding	int	Yes	Value of Bottom Padding for the Cell.
cellDensities	objList(cellDensity)	No	List of pointers to cell density information. Equivalent to LEF MACRO DENSITY.
defName	string	No	Fully qualified def name of the cell from db
dontTouch	bool	Yes	This attribute says any inst of this base_cell cannot be modified during optimization. This is the effective dont_touch value for all lib_cells. It is set to the worst case of the lib_cells during init_design and can only be updated by set_db / set_dont_touch after that (subsequent library reads will not affect). This attribute will get restored back to the state during write_db regardless if the library files have been altered.

dontUse	bool	Yes	This attribute says do not use this base_cell during optimization. This is the effective dont_use value for all lib_cells. It is set to the worst case of the lib_cells during init_design and can only be updated by set_db / set_dont_touch after that (subsequent library reads will not affect). This attribute will get restored back to the state during write_db regardless if the library files have been altered.
eeqCells	objList(libCell)	No	List of pointers to electrically equivalent cells. Equivalent to LEF MACRO EEQ.
eeqVariant	int	Yes	The LEF EEQ cell variant number from the LEF58_EEQ property 'EEQ macroName VARIANT num'
foreigns	objList(foreign)	No	List of pointers to foreign references. Equivalent to LEF MACRO FOREIGN.
isAbstractDefined	bool	No	Indicates that the libCell is abstract defined
isAlwaysOn	bool	No	Specifies the cell is an always-on cell. An always-on cell normally has two power pins. One is primary which aligns with the normal cell power-rail, and the other is the secondary which actually powers the cell, even when the primary power is off. This attribute can be set by liberty files, or by CPF commands.
isBuffer	bool	No	Indicates that the libCell acts as a buffer
isFixedMask	bool	No	Indicates the cell has FIXEDMASK keyword in LEF.
isInverter	bool	No	Indicates that the libCell acts as a inverter

isIsoNor	bool	No	Specifies the cell is an ISONOR cell. An ISONOR cell is a kind of isolation cell, which has only one primary power pin and one primary ground pin. An ISONOR cell is defined by library files. In cell library, it has permit_power_down true for primary power pin, alive_during_power_up true for input signal pin, and alive_during_partial_power_down true for enable pin and output signal pin. The attribute should be queried after read and commit power intent.
isIsolationCell	bool	No	This attribute specifies the cell is an isolation cell. An isolation cell is used to clamp the signal to high or low when its input is shutoff (unknown). This attribute can be set by liberty files, or by CPF commands.
isLevelShifter	bool	No	Specifies the cell is a level shifter cell. A level-shifter cell is used to shift the signal voltage from low(high) to high(low). This attribute can be set by liberty files, or by CPF commands
isPowerSwitch	bool	No	Specifies the cell is a power switch cell. The power switch cell is used to switch off the power/ground during shutoff. This attribute can be set by liberty files, or by CPF commands.
isRetention	bool	No	Specifies the cell is a state-retention cell. A state-retention cell is used to retain its state during shutoff. It has a secondary power pin which powers the cell and retains its state, even when the primary power is off. This attribute can be set by liberty files, or by CPF commands.
isSequential	bool	No	If the cell is sequential or not.
isTimeDefined	bool	No	Indicates that the cell is defined in a timing library

isVDDOnBottom	bool	Yes	Indicates this standard cell has a power pin along the bottom of the cell. This is derived from the power and ground pin information in the cell. It is used by the placer to align multi-height cells properly to the rows. It it not meaningful for non standard cells. Modifications are not saved and are only valid for the current session.
layerShapeObstructions	objList(layerShape)	No	List of pointers to layerShapes that define cell obstruction geometries.
layoutLib	string	No	Library name for the OpenAccess layout for the cell (GDSII equivalent)
layoutView	string	No	View name for the OpenAccess layout for the cell (GDSII equivalent)
lefFileName	string	No	Lef file name of the cell
leftEdgeName	string	No	Name of cell edge type for the left edge of the cell (R0/N orientation), used to indicate which cells need extra spacing to other cells.
leftPadding	int	Yes	Value of Left Padding for the Cell.
name	string	No	Name of cell
numBidirs	int	No	Number of bidir terminals in the cell
numInputs	int	No	Number of input terminals in the cell
numOutputs	int	No	Number of output terminals in the cell
numPGTerms	int	No	Number of power/ground terminals in the cell
numPhysTerms	int	No	Number of physical terminals in the cell
numRefs	int	No	Number of times cell is used in design
numTerms	int	No	Number of terminals in the cell

pgTerms	objList(term)	No	List of pointers to power/ground terminals in the cell
physTerms	objList(term)	No	List of pointers to physical (unused) signal terminals in the cell
props	objList(prop)	No	List of pointers to properties
rightEdgeName	string	No	Name of cell edge type for the right edge of the cell (R0/N orientation), used to indicate which cells need extra spacing to other cells.
rightPadding	int	Yes	Value of Right Padding for the Cell.
shapeViaObstructions	objList(shapeVia)	No	List of pointers to shapeVias that define cell obstruction geometries.
site	obj(site)	Yes	Pointer to site of the cell. Modifications are valid only for the current session.
size	pt	No	Size of the cell
size_x	coord	No	X of size of the cell
size_y	coord	No	Y of size of the cell
subClass	enum	Yes	All the LEF CLASS and PROPERTY LEF58_CLASS values (and equivalent OpenAccess values). Refer to the LEF documentation for the complete list and descriptions. They are separated into 5 classes with a unique prefix based on their usage as described here: CLASS COVER types all start with cover. CLASS RING or BLOCK start with block. CLASS PAD start with pad. CLASS CORE start with core. CLASS ENDCAP that are corner cells (TOPLEFT, TOPRIGHT, BOTTOMLEFT, BOTTOMRIGHT) start with corner. CLASS ENDCAP that are not corner cells all start with core because they are all placed in the core rows like CLASS CORE cells.

			No CLASS means the value is none. Modifications are valid only for the current session. **Legal enum**: block, blockBlackBox, blockRing, blockSoft, core, coreAntenna, coreEndCapBottomEdge, coreEndCapLeftBottomCorner, coreEndCapLeftBottomCornerNeighbor, coreEndCapLeftBottomEdge, coreEndCapLeftBottomEdgeNeghbor, coreEndCapLeftBottomEvenSiteCorner, coreEndCapLeftBottomEvenSiteEdge, coreEndCapLeftBottomOddSiteCorner, coreEndCapLeftBottomOddSiteEdge, coreEndCapLeftEdge, coreEndCapLeftEdgeBottomBorder, coreEndCapLeftEdgeTopBorder, coreEndCapLeftEvenSiteEdge, coreEndCapLeftOddSiteEdge, coreEndCapLeftTopCorner, coreEndCapLeftTopCornerNeighbor, coreEndCapLeftTopEdge, coreEndCapLeftTopEdgeNeighbor, coreEndCapLeftTopEvenSiteCorner, coreEndCapLeftTopEvenSiteEdge, coreEndCapLeftTopOddSiteCorner, coreEndCapLeftTopOddSiteEdge, coreEndCapPost, coreEndCapPre, coreEndCapRightBottomCorner, coreEndCapRightBottomCornerNeighbor, coreEndCapRightBottomEdge, coreEndCapRightBottomEdgeNeighbor, coreEndCapRightBottomEvenSiteCorner, coreEndCapRightBottomEvenSiteEdge, coreEndCapRightBottomOddSiteCorner, coreEndCapRightBottomOddSiteEdge, coreEndCapRightEdge, coreEndCapRightEdgeBottomBorder, coreEndCapRightEdgeTopBorder, coreEndCapRightEvenSiteEdge, coreEndCapRightOddSiteEdge, coreEndCapRightTopCorner, coreEndCapRightTopCornerNeighbor, coreEndCapRightTopEdge,

			coreEndCapRightTopEdgeNeighbor, coreEndCapRightTopEvenSiteCorner, coreEndCapRightTopEvenSiteEdge, coreEndCapRightTopOddSiteCorner, coreEndCapRightTopOddSiteEdge, coreEndCapTopEdge, coreFeedthru, coreSpacer, coreTieHigh, coreTieLow, coreWellTap, cornerBottomLeft, cornerBottomRight, cornerTopLeft, cornerTopRight, cover, coverBump, coverFill, none, pad, padAreaIO, padInout, padInput, padOutput, padPower, padSpacer
symmetryR90	bool	Yes	Symmetry of the cell in R90. Modifications are valid only for the current session.
symmetryX	bool	Yes	Symmetry of the cell in X. Modifications are valid only for the current session.
symmetryY	bool	Yes	Symmetry of the cell in Y. Modifications are valid only for the current session.
tapType	string	Yes	Specifies the name of a well tap type for this cell. Various rules for well taps are grouped together for each tapType. See the LEF documentation on the TAPTYPE keyword for more details
tapWall	bool	Yes	Specifies a special well tap cell (LEF CLASS CORE WELLTAP) or a special endcap (LEF CLASS ENDCAP ...) cell that can be used for a tap wall purpose, which is used to break OD diffusion and aligned vertically to form a tap wall. See the LEF docs about the keyword TAPWALL for more details.
terms	objList(term)	No	List of pointers to signal terminals in the cell

topEdgeName	string	No	Name of cell edge type for the top edge of the cell (R0/N orientation), used to indicate which cells need extra spacing to other cells.
topPadding	int	Yes	Value of Top Padding for the Cell.

Note: In addition to the above entries, every object has an objType attribute

marker

Parent Object

topCell

Definition

DRC Marker

Types and Definitions

Child Object or Attribute	Type	Edit	Description
area	area	No	Area of the marker as defined by the LEF MACRO SIZE or OVERLAP information
box	rect	No	Bounding box for marker
box_area	area	No	Area of bounding box for marker
box_ll	pt	No	Lower left (ll) of bounding box for marker
box_llx	coord	No	Lower left X (llx) of bounding box for marker
box_lly	coord	No	Lower left Y (lly) of bounding box for marker
box_size	pt	No	Size of bounding box for marker

box_sizex	coord	No	Size X of bounding box for marker
box_sizey	coord	No	Size Y of bounding box for marker
box_ur	pt	No	Upper right (ur) of bounding box for marker
box_urx	coord	No	Upper Right X (urx) of bounding box for marker
box_ury	coord	No	Upper Right Y (ury) of bounding box for marker
layer	obj(layer)	No	Pointer to layer
message	string	No	DRC marker message
messageId	int	No	DRC marker message ID
originator	enum	No	Marker originator **Legal enum**: Ccopt, Check, CheckPlace, External, NRLitho, PG, RouteDesign, Unknown
polyPts	list(pt)	No	Polygon boundary for the marker, the first point is not repeated as the last point in the list
subType	enum	No	Marker subtype (system, when originator != External) **Legal enum**: AbuttedIBCellMaxLengthViolation, AcuteAngle, AdjacentOneSiteFillerViolation, AntAreaRatio, AntCAreaRatio, AntCSAreaRatio, AntSAreaRatio, Antenna, BTSDRV, BumpConnectTargetOpen, BusGap, BusGuideLayerRangeViolation, BusGuideOverlap, BusGuideViolation, BusRes, BusSpacing, BusWidth, CLPViolation, CTSDRV, Cap, CapViolation, CellContinuousConstraintViolation, CellStackConstraintViolation, CellVariantSiteMismatch, ClassBumpOpen, ClkHaloViolation, ClockTreeViolation, ConnectedMaxFloatingArea, ConnectivityAntenna, ConvexCornerViolation, CutSpacing, DamagedPin, DiffPairsMinCuts, DiffPairsRes, DiffusionWidthLengthRatioViolation, Distance, DontCross, DpLoopViolation, EndCapFill1Channel, EndCapNarrowChannel, EndCapNarrowEdge, EndCapNoInstanceViolation, EndCapWrongLocationViolation, EndCapWrongOrientViolation, EndCapWrongTypeViolation, EndOfLine, ExclusiveRegionViolation, FanoutViolation, FillerAbutPatternVio, ForbiddenSpacingVio, Geom,

			IBCellVerticalStackMaxLengthViolation, IRDrop, ImplantStrictCornerTouchViolation, InaccessiblePin, InstHaloOverlapViolation, InstancePlacement, InsufficentMetal, Loop, MEOLMaxLengthViolation, MGrid, MSBusSameLayer, MSDiffPairGap, MSDiffPairSpacing, MSDiffPairThreshold, MSDiffPairWidth, MSLengthMismatch, MSMatchLayerMatch, MSMatchPairsSpacing, MSMatchPairsWidth, MSNdr, MSRouteLayers, MSShieldGap, MSShieldSpacing, MSShieldWidth, MSVAllPGTermNotConnected, MSVDRV, MSVGndTermWrongConnection, MSVInstConnectedToSamePD, MSVInstNotInAnyPD, MSVIsoIOAreOfTheSamePD, MSVIsoInputTiedToPG, MSVIsoNotInRule, MSVNetMissingRequiredIsolation, MSVNetMissingRequiredShifter, MSVPGTermNotConnected, MSVPwrTermWrongConnection, MSVShifterIOAreOfTheSamePD, MSVShifterInputTiedToPG, MSVShifterNotInRule, MSVTermMissingRequiredIsolation, MSVTermMissingRequiredShifter, MSVTermNotConnectedToSNet, MSVTieHiWrongConnection, MSVTieLoWrongConnection, MSVWrongIsoLocation, MSVWrongShifterLocation, Mar, MatchPairsMinCuts, MatchPairsRes, MaxFloatingArea, MaxStackedVia, MaxWidth, MinArea, MinCuts, MinEnclosure, MinSize, MinSpacing, MinStep, MinVtEnclosedAreaVio, MinWidth, MissConn, MissingVia, MixedSignalRes, MultiplePorts, NRLithoViolation, NoFillerViolation, NoFillet, NoGrid, NoRouting, NoShield, NoSubtype, NoTurnVia, Notch, OffGrid, Open, OutsideBoundary, Overlap, PartialRouting, PinAccessDir, PinCheck_abutMutiPinViol, PinCheck_abutmentViol, PinCheck_busGuideViol, PinCheck_colorViol, PinCheck_constNet, PinCheck_dViol, PinCheck_drcSpViol, PinCheck_fenceViol, PinCheck_groupKeepOutSpViol, PinCheck_isPinExternal, PinCheck_marViol, PinCheck_missingShape, PinCheck_multiLevelViol, PinCheck_netGroupExclVoil, PinCheck_netGroupOrderVoil, PinCheck_netGroupPureExclVoil, PinCheck_overlapViol, PinCheck_pinGroupExclVoil, PinCheck_pinGroupOrderVoil, PinCheck_pinGroupPureExclVoil, PinCheck_pinGuideViol, PinCheck_routeTrackViol, PinCheck_spViol, PinCheck_wViol, PinCheck_zViol, PinWithoutPort, ProcessAntenna, Protrusion, RoutedMinCuts, RoutedRes, RoutedSpacing, RoutedWidth,

			SPCellKeepoutOverlapPGViol, SPContextConstraintViolation, SPDiffusionCoreEdgeLengthViolation, SPDiffusionWidthViolation, SPDoublePatternViolation, SPDynamicPaddingViolation, SPFillerGapViolation, SPIOOverlapViolation, SPImpltAreaViolation, SPImpltSpacingViolation, SPInstSpaceGroupViolation, SPMacroBlockageViolation, SPMaxCoreEdgeLengthViolation, SPNotOfFenceViolation, SPODForbiddenSpacingViolation, SPOffGridViolation, SPOrientationViolation, SPOutOfCoreViolation, SPOverlapViolation, SPPGHookupViolation, SPPinAccessViolation, SPPinTrackAlignmentMismatch, SPPinTrackMaskMismatchVio, SPPlacementBlkViolation, SPRFViolation, SPRowViolation, SPSiteOrientViolation, SPSoftBlockageViolation, SPSpacingRuleViolation, SPTPOLayerViolation, SPTechSiteViolation, ShieldMinCuts, ShieldOpen, ShieldRes, Short, SingleMaxFloatingArea, SpHPinTrackAlignmentViolation, SpHPinTrackMaskViolation, SpVia0AlignmentViolation, SpecifiedGapViolation, StackedVia, Tapering, TieHiLo, Tos, TranViolation, UnConnectedPin, UnexpectedBumpConnectTarget, VFCMissTSVBump, VerticalStackMaxLengthViolation, ViaEnclosure, ViaOverlap, WeaklyConnectedPin, WellAntennaNoProtectViolation, WellSeparation, WellTap, Width, WireExtn, WireGap, WireOverlap, WrongWay
type	enum	No	Marker type (system, when originator != External) **Legal enum**: ACLimit, Antenna, BusPlanning, Ccopt, Connectivity, Density, Electrical, Floorplan, Geometry, IRDrop, MSV, MixedSignal, NRLitho, None, Overlap, Placement, XTalk
userOriginator	string	No	Marker originator (tool, user specified, when originator = External)
userSubType	string	No	Marker subType (user specified, when originator = External)
userType	string	No	Marker type (user specified, when originator = External)

Note: In addition to the above entries, every object has an objType attribute

net

Parent Object

bump, bus, hInstTerm, hNet, hTerm, instTerm, netGroup, pWire, pd, pgInstTerm, resistor, routeType, sViaInst, sWire, term, topCell, vWire, viaInst, whatIfVia, whatIfWire, wire

Definition

Canonical (flat) net (equivalent to connectivity in DEF NETS and SPECIALNETS)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
allTerms	objList(instTerm, term)	No	List of pointers to connections (terms and instTerms)
area	area	No	Area of the net as defined by the LEF MACRO SIZE or OVERLAP information
avoidDetour	bool	Yes	Avoids detours of roughly more than a few gcell grids on the specified nets (affects global routing only).
botMaskLayerNum	int	Yes	Specify the bottom layer number that the mask constraint should be applied.
botOneSideSpacingLayerNum	int	Yes	Specify the bottom layer number of one side spacing, The spacing constraint could be applied to one side only instead of the typical both sides. This is an attempt to balance the timing benefit vs routability impact.

bottomPreferredLayer	obj(layer)	Yes	Pointer to the preferred lowest routing layer. This attribute is a soft limit; that is, NanoRoute might use a layer below the specified layer if necessary to complete routing.
box	rect	No	Bounding box of the net's wiring if routed, otherwise bounding box of the terms/instTerms on the net.
box_area	area	No	Area of bounding box of the net's wiring if routed, otherwise bounding box of the terms/instTerms on the net.
box_ll	pt	No	Lower left (ll) of bounding box of the net's wiring if routed, otherwise bounding box of the terms/instTerms on the net.
box_llx	coord	No	Lower left X (llx) of bounding box of the net's wiring if routed, otherwise bounding box of the terms/instTerms on the net.
box_lly	coord	No	Lower left Y (lly) of bounding box of the net's wiring if routed, otherwise bounding box of the terms/instTerms on the net.
box_size	pt	No	Size of bounding box of the net's wiring if routed, otherwise bounding box of the terms/instTerms on the net.
box_sizex	coord	No	Size X of bounding box of the net's wiring if routed, otherwise bounding box of the terms/instTerms on the net.
box_sizey	coord	No	Size Y of bounding box of the net's wiring if routed, otherwise bounding box of the terms/instTerms on the net.

box_ur	pt	No	Upper right (ur) of bounding box of the net's wiring if routed, otherwise bounding box of the terms/instTerms on the net.
box_urx	coord	No	Upper Right X (urx) of bounding box of the net's wiring if routed, otherwise bounding box of the terms/instTerms on the net.
box_ury	coord	No	Upper Right Y (ury) of bounding box of the net's wiring if routed, otherwise bounding box of the terms/instTerms on the net.
bus	obj(bus)	No	Pointer to bus object, Null (0x0) for scalar net
defName	string	No	Fully qualified def name of the net from db
dontTouch	enum	Yes	This attribute defines the preservation status of an net during optimization. Setting this attribute will preserve all connections on this net. When set, this overrides any setting on hnets of this net. **Legal enum**: deleteOk, false, true
hNets	objList(hNet)	No	List of pointers to hNets which make up this net.
initialBottomPreferredLayerNum	int	No	The initial lowest layer set by setAttribute or dbSet. Opt will use it as a reference to set bottomPreferredLayer to any value equal or greater than initialBottomPreferredLayer and smaller than topPreferredLayer for better performance.
instTerms	objList(instTerm)	No	List of pointers to instTerm connections

isAnalog	bool	Yes	Indicates that net is an analog net
isCTSClock	bool	Yes	Indicates that net is a cts clock net (equivalent to 'USE CLOCK' property in DEF)
isClock	bool	No	Indicates that net is a clock according to timing constraints and tracing
isEarlyGlobalRouted	bool	Yes	Indicate that the net is from early global route
isEdit	bool	Yes	Indicate that the wire/via on the net has been modified. This attribute only works when "setEditMode -create_is_edit_flag 1". Please look up for more details on "setEditMode -create_is_edit_flag" man page.
isExternal	bool	No	Indicates that net is connected to a terminal
isFixedBump	bool	No	Indicates that the net is connected to a bump
isGnd	bool	Yes	Indicates that net is ground
isIlmNet	bool	No	Specify if a top level net is connecting to ILM (Interface Logic Model) modules and some or all terms (either drive or sink) of the net are inside ILM module.
isMixedSignal	bool	No	Indicates that the net has one or more Mixed Signal constraints when the value is true
isPatternTrunk	bool	No	Indicates that the net is routed with a trunk pattern
isPhysOnly	bool	No	Indicates that the net is physical only

isPwr	bool	Yes	Indicates that net is power
isPwrOrGnd	bool	No	Indicates that net is power or ground
isScanNet	bool	No	Indicates that net is a scan net
mask	int	Yes	Indicates mask number for multiple mask layer usage. Refer to layer .numMask attributes for valid range, 0 indicates unconstrained. Layers that do not support the specified value will be treated as unconstrained. (Legal range: 0-3).
maxVoltage	float	No	Max voltage for the net, determined by the max net voltage of all active hold views. Net voltage is determined using the following procedure: voltage from net's driver instTerm, associated power from CPF related_power_pins command or Liberty related_pg_pin attribute or LEF SUPPLYSENSITIVITY statement; power domain operating voltage; or default system voltage.
minVoltage	float	No	Min voltage for the net, determined by the min net voltage of all active setup views.Net voltage is determined using the following procedure: voltage from net's driver instTerm, associated power from CPF related_power_pins command or Liberty related_pg_pin attribute or LEF SUPPLYSENSITIVITY statement; power domain operating voltage; or default system voltage.
name	string	No	Canonical (flat) name of the net

numInputTerms	int	No	Number of input (load) connections on the net (terms with .isOutput 1 + instTerms with .isInput 1)
numOutputTerms	int	No	Number of output (driver) connected to the net (terms with .isInput 1 + instTerms with .isOutput 1)
numTerms	int	No	Number of connections to the net (number of terms + number of instTerms)
pWires	objList(pWire)	No	List of pointers to wire patch objects (small rectangles that are connected to the symbolic wiring graph)
preferredExtraSpace	int	Yes	Gives additional pitch spacing to the specified net. Use this attribute to give critical nets extra space to reduce coupling. Legal range: 0-3
props	objList(prop)	No	List of pointers to properties
rule	obj(rule)	No	Pointer to non-default rule corresponding to the net, nets with the default routing rule will return NULL (0x0).
sVias	objList(sViaInst)	No	List of pointers to sViasInsts (DEF SPECIALNETS equivalent)
sWires	objList(sWire)	No	List of pointers to sWires (DEF SPECIALNETS equivalent)
shieldNets	objList(net)	No	Specifies the nets that are to be used in the creation of shielding wires (DEF NETS + SHIELDNET equivalent).
siPostRouteRepair	bool	Yes	Specifies that Signal Integrity issues should be repaired at the end of the routing phase.

skipAntennaRepair	bool	Yes	Specifies that antenna violations should not be corrected by the routing as the violation will be corrected in a different level of the design hierarchy.
skipRouting	bool	Yes	Specifies that Nanoroute should not route or re-route the net.
terms	objList(term)	No	List of pointers to top level term connections
topMaskLayerNum	int	Yes	Specify the top layer number that the mask constraint should be applied.
topOneSideSpacingLayerNum	int	Yes	Specify the top layer number of one side spacing, The spacing constraint could be applied to one side only instead of the typical both sides. This is an attempt to balance the timing benefit vs routability impact.
topPreferredLayer	obj(layer)	Yes	Pointer to the preferred highest routing layer. This attribute is a soft limit; that is, NanoRoute might use a layer above the specified layer if necessary to complete routing.
vWires	objList(vWire)	No	List of pointers to virtual wire objects (connections that are not real shapes used in the symbolic wiring graph)
vias	objList(viaInst)	No	List of pointers to viaInsts (DEF NETS equivalent)

weight	int	Yes	Specifies a relative weight for routing nets. In each switch box, the NanoRoute router routes nets with the highest weight first, then the next highest weight, and so on. Specify a value higher than 2 to ensure a net is routed before other nets.
whatIfVias	objList(whatIfVia)	No	List of 'what if' vias.
whatIfWires	objList(whatIfWire)	No	List of 'what if' wires.
wires	objList(wire)	No	List of pointers to wires (DEF NETS equivalent)

Note: In addition to the above entries, every object has an objType attribute

netGroup

Parent Object

busGuide, fPlan, pinGuide

Definition

Net group

Types and Definitions

Child Object or Attribute	Type	Edit	Description
excludeNet	enum	Yes	Indicates group net exclude type(allLayer, sameLayer, inclusive, allLayerInGuidedArea) **Legal enum**: allLayer, allLayerInGuidedArea, inclusive, sameLayer
isCompact	bool	Yes	Indicate whether the nets in the group are assigned tightly together. By default, nets which are part of a group and associated to a guide can be spread inside the guide based on the area available and alignment to targets
isGuided	bool	Yes	Indicates if net group is guided
isOptimizeOrder	bool	Yes	Indicates whether net order will be optimized
isSpread	bool	Yes	Indicates whether member nets distributed evenly
keepOutSpace	int	Yes	Minimum spacing with pin of foreign nets (Unit: track)
name	string	No	Group name
nets	objList(net)	No	List of pointers to member nets

Note: In addition to the above entries, every object has an objType attribute

pBlkg

Parent Object

fPlan

Definition

Placement blockage (hard, soft, partial, macroOnly)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
area	area	No	Area of the placement blockage as defined by the LEF MACRO SIZE or OVERLAP information
attr	enum	Yes	Indicates whether the blockage has been pushed down **Legal enum**: pushdown, undefined
boxes	list(rect)	Yes	List of rectangles that define the shape of placement blockage
density	float	Yes	The max placement density percent allowed inside this pBlkg. It must be in the range of 5 to 100, in steps of 5. It is only valid if the type = partial or soft. For example, a partial placement percentage of 75 percent means that up to 75 percent of placement density is allowed in the area. If the type is not partial or soft, a value of 0 is returned.
inst	obj(inst)	No	Pointer to the instance that the placement blockage is associated with (equivalent to DEF BLOCKAGES + COMPONENT)
isNoFlop	bool	Yes	Indicate a partial placement blockage to prevent flops to be placed underneath.
isPushdown	bool	Yes	Indicates that placement blockage has been pushed down from a higher level in the design hierarchy.
name	string	Yes	Name of placement blockage
shapes	objList(densityShape, shape)	No	List of pointers to shapes
type	enum	Yes	Blockage type (hard, macroOnly, partial, soft) **Legal enum**: hard, macroOnly, partial, soft

Note: In addition to the above entries, every object has an objType attribute

pd

Parent Object

group, inst, instTerm, term, topCell

Definition

Power domain

Types and Definitions

Child Object or Attribute	Type	Edit	Description
availableSupplyNets	objList(net)	No	Specifies the power nets physically available for this power_domain to use for secondary power pin connections.
baseDomains	objList(pd)	No	Specifies the base power domains (always-on domains) that supply the power to this switchable power_domain through power_switch cells.
core2Bot	coord	No	Distance between the power domain edge and its core box
core2Left	coord	No	Distance between the power domain edge and its core box
core2Right	coord	No	Distance between the power domain edge and its core box
core2Top	coord	No	Distance between the power domain edge and its core box
extBot	coord	Yes	Maximum search distance for power connections

extEdges	list(coord)	No	List of maximum search distances for power extension connections in clockwise order starting with the vertical edge at the lower-left corner (smallest Y, then smallest X)
extLeft	coord	Yes	Maximum search distance for power connections
extRight	coord	Yes	Maximum search distance for power connections
extTop	coord	Yes	Maximum search distance for power connections
gapBot	coord	No	Minimum spacing to other power domains or rows
gapEdges	list(coord)	No	List of minimum spacing values to other power domains or rows in clockwise order starting with the vertical edge at the lower-left corner (smallest Y, then smallest X)
gapLeft	coord	No	Minimum spacing to other power domains or rows
gapRight	coord	No	Minimum spacing to other power domains or rows
gapTop	coord	No	Minimum spacing to other power domains or rows
group	obj(group)	No	Pointer to group
isAlwaysOn	bool	No	Indicates that the power domain is always on
isDefault	bool	No	Indicates that the power domain is the default power domain
isPowerDomainMacroOnly	bool	No	Indicates whether the domain member has only hardmacro members.
isVirtual	bool	No	Specifies this is a virtual domain, meaning that this domain does not have any inst members.
name	string	No	Name of the power domain

powerSwitchRuleName	string	No	Indicates the domain's power switch rule name.
primaryGroundNet	obj(net)	No	Specifies the main ground rail net of the power_domain.
primaryPowerNet	obj(net)	No	Specifies the main power rail net of the power_domain.
shutoffCondition	string	No	Indicates the power domain's shutoff condition. It is a logic expression with pin or hpin names.

Note: In addition to the above entries, every object has an objType attribute

pgInstTerm

Parent Object

inst

Definition

A power or ground instance terminal that connects to a net.

Types and Definitions

Child Object or Attribute	Type	Edit	Description
defName	string	No	Fully qualified DEF name of the pgInstTerm.
inst	obj(inst)	No	Pointer to Instance containing the pgInstTerm.
isSGNC	bool	No	Indicates pg connection was made by Single Inst GNC rule
name	string	No	Fully qualified pgInstTerm name including instance path.
net	obj(net)	No	The net connected to this pgInstTerm.
term	obj(term)	No	The corresponding cell term for this pgInstTerm.

Note: In addition to the above entries, every object has an objType attribute

pin

Parent Object

term

Definition

Lef Port

Types and Definitions

Child Object or Attribute	Type	Edit	Description
allShapes	objList(layerShape, shapeVia)	No	List of pointers to layerShapes and shapeVias that define the terminal pin geometries.
class	enum	No	Pin class **Legal enum**: bump, core, none, undefined
layerShapeShapes	objList(layerShape)	No	List of pointers to layerShapes that define the terminal pin geometries.
portNumber	int	No	The port number for geometries under the pin.
shapeViaShapes	objList(shapeVia)	No	List of pointers to shapeVias that define the terminal pin geometries.

Note: In addition to the above entries, every object has an objType attribute

pinGroup

Parent Object

fPlan, pinGuide

Definition

Pin group

Types and Definitions

Child Object or Attribute	Type	Edit	Description
cell	obj(ptnCell, topCell)	No	Pointer to pin group cell (ptnCell or topCell)
excludePin	enum	Yes	Indicates group pin exclude type(allLayer, sameLayer, inclusive, allLayerInGuidedArea) **Legal enum**: allLayer, allLayerInGuidedArea, inclusive, sameLayer
isCompact	bool	Yes	Indicate whether the pins in the group are assigned tightly together. By default, pins which are part of a group and associated to a guide can be spread inside the guide based on the area available and alignment to targets
isGuided	bool	No	Indicates if pin group is guided
isSpread	bool	Yes	Indicates whether member pins distributed evenly
keepOutSpace	int	Yes	Minimum spacing with foreign pins (Unit: track)
name	string	No	Name of pinGroup
optimizeOrder	bool	No	Indicates whether pin order will be optimized
pinSpacing	int	Yes	Minimum spacing between adjacent pins (Unit: track)
pinWidth	coord	Yes	Minimum pin width (0 indicates default routing width for each layer is used)
terms	objList(term)	No	List of pointers of terms associated with the group

Note: In addition to the above entries, every object has an objType attribute

pinGuide

Parent Object

fPlan

Definition

pinGuide

Types and Definitions

Child Object or Attribute	Type	Edit	Description
area	area	No	Area of the pin guide as defined by the LEF MACRO SIZE or OVERLAP information
boxes	list(rect)	No	rectangle that defines the pin guide
cell	obj(topCell)	No	cell where pin guide exists
layerPriority	bool	No	layer priority
layers	objList(layer)	No	Pointers of layers where pinGuide is present
name	string	No	Name of pin guide
netGroup	obj(netGroup)	No	Pointer to netGroup, if pinGuide is based on netGroup otherwise NULL
pinGroup	obj(pinGroup)	No	Pointer to pinGroup, if pinGuide is based on pinGroup otherwise NULL

Note: In addition to the above entries, every object has an objType attribute

pinShape

Parent Object

term

Definition

This corresponds to one of a term's .pins.layerShapeShapes.shapes. It carries a link to the term object, that a layerShape object does not have. This allows GUI operations that need both the layerShape and the term object together as one object like delete_obj or wire-editing commands.

Types and Definitions

Child Object or Attribute	Type	Edit	Description
layer	obj(layer)	No	Layer of the pinShape.
mask	int	No	The mask number for this pinShape if its layer has multiple masks. 0 means it is uncolored. Refer to layer.numMasks for legal range.
name	string	No	Name of the term for this pinShape. Note that all the pin shapes for one term will have the same name.
polyPts	list(pt)	No	Points of the polygon of the pinShape for type rect or polygon.
rect	rect	No	Bounding box of the pinShape for type rect or polygon.
rect_area	area	No	Area of bounding box of the pinShape for type rect or polygon.
rect_ll	pt	No	Lower left (ll) of bounding box of the pinShape for type rect or polygon.
rect_llx	coord	No	Lower left X (llx) of bounding box of the pinShape for type rect or polygon.
rect_lly	coord	No	Lower left Y (lly) of bounding box of the pinShape for type rect or polygon.
rect_size	pt	No	Size of bounding box of the pinShape for type rect or polygon.
rect_sizex	coord	No	Size X of bounding box of the pinShape for type rect or polygon.
rect_sizey	coord	No	Size Y of bounding box of the pinShape for type rect or polygon.
rect_ur	pt	No	Upper right (ur) of bounding box of the pinShape for type rect or polygon.

rect_urx	coord	No	Upper Right X (urx) of bounding box of the pinShape for type rect or polygon.
rect_ury	coord	No	Upper Right Y (ury) of bounding box of the pinShape for type rect or polygon.
term	obj(term)	No	The term this pinShape belongs to.
type	enum	No	Type of the pinShape. **Legal enum**: polygon, rect

Note: In addition to the above entries, every object has an objType attribute

pkgComponent

Parent Object

topCell

Definition

Chip in the package

Types and Definitions

Child Object or Attribute	Type	Edit	Description
cellName	string	No	The name of other chip cell in the package
instName	string	No	The name of other chips in the package
pt	pt	No	Location of the other chips in the package
pt_x	coord	No	X of location of the other chips in the package
pt_y	coord	No	Y of location of the other chips in the package
refDes	string	No	Another identifying name of the chip in the package
size	pt	No	Size of the cell
size_x	coord	No	X of size of the cell
size_y	coord	No	Y of size of the cell

Note: In addition to the above entries, every object has an objType attribute

pkgObject

Parent Object

topCell

Definition

Object in the package

Types and Definitions

Child Object or Attribute	Type	Edit	Description
dieNetName	string	No	the name of die net which this package object connected to
layerName	string	No	The package layer that this package object belongs to
name	string	No	the name of this package object
pkgNetName	string	No	the name of package net which this package object connected to
pt	pt	No	Location of the package object
pt_x	coord	No	X of location of the package object
pt_y	coord	No	Y of location of the package object
size	pt	No	Size of the package object
size_x	coord	No	X of size of the package object
size_y	coord	No	Y of size of the package object
type	enum	No	the type of this package object **Legal enum**: BGA_BALL, BOND_FINGER, BOND_WIRE, COMPONENT, FLIGHT_LINE, OTHER, ROUTE

Note: In addition to the above entries, every object has an objType attribute

prop

Parent Object

bump, group, guiLine, guiPoly, guiRect, guiText, hInst, hInstTerm, hNet, hTerm, head, inst, instTerm, layer, libCell, net, ptnCell, sViaInst, sWire, term, topCell, vCell

Definition

Property

Types and Definitions

Child Object or Attribute	Type	Edit	Description
name	string	No	Property name
parent	obj(parent)	No	Pointer to the parent object referencing the property
value	value	No	Property value (depends on valueType)
valueType	enum	No	Type of value stored in the property (int, float, etc.) **Legal enum**: bits, float, int, obj, pt, rect, string

Note: In addition to the above entries, every object has an objType attribute

ptn

Parent Object

hInst, topCell

Definition

Partition Object

Types and Definitions

Child Object or Attribute	Type	Edit	Description
cloneHInsts	objList(hInst)	No	List of pointers to the clone hierarchical instances if the partition is not committed and not a blackbox, otherwise 0x0 will be returned.

cloneInsts	objList(inst)	No	List of pointers to the clone instances if the partition is committed or a blackbox, otherwise 0x0 will be returned.
clones	objList(hInst, inst)	No	List of pointers to the clone inst/hInsts. The objTypes returned will be inst if the partition is committed or a blackbox, otherwise the objType will be hInst.
coreSpacingBot	coord	Yes	Spacing between the partition boundary and core design area of the partition module
coreSpacingLeft	coord	Yes	Spacing between the partition boundary and core design area of the partition module
coreSpacingRight	coord	Yes	Spacing between the partition boundary and core design area of the partition module
coreSpacingTop	coord	Yes	Spacing between the partition boundary and core design area of the partition module
isBlackBox	bool	No	Specifies if partition is a blackbox
isCommited	bool	No	Specifies if partition is committed
master	obj(hInst, inst)	No	Pointer to the master inst/hInst. The objType returned will be inst if the partition is committed or a blackbox, otherwise the objType will be hInst.
masterHInst	obj(hInst)	No	Pointer to the master hierarchical instance if the partition is not commited and not a blackbox, otherwise 0x0 will be returned.
masterInst	obj(inst)	No	Pointer to the master instance if the partition is commited or a blackbox, otherwise 0x0 will be returned.
minPitchBot	int	Yes	Specifies the pin pitch (in tracks)
minPitchLeft	int	Yes	Specifies the pin pitch (in tracks)
minPitchRight	int	Yes	Specifies the pin pitch (in tracks)
minPitchTop	int	Yes	Specifies the pin pitch (in tracks)
name	string	No	name of partition.

pHaloBot	coord	Yes	Specifies extra spacing around the partition that should not be used for placement
pHaloLeft	coord	Yes	Specifies extra spacing around the partition that should not be used for placement
pHaloRight	coord	Yes	Specifies extra spacing around the partition that should not be used for placement
pHaloTop	coord	Yes	Specifies extra spacing around the partition that should not be used for placement
pinLayersBot	objList(layer)	Yes	List of pointers to the layers to use for the partition
pinLayersLeft	objList(layer)	Yes	List of pointers to the layers to use for the partition
pinLayersRight	objList(layer)	Yes	List of pointers to the layers to use for the partition
pinLayersTop	objList(layer)	Yes	List of pointers to the layers to use for the partition
pinToCornerDist	list(int)	No	List of distance constraints (in tracks) of pins from topCell/ptnCell corners where the lower left corner is listed first and the remaining corners are listed in clockwise order.
rHaloBotLayer	obj(layer)	Yes	Pointer to the bottom partition layer for which routing halo will be created
rHaloSideSize	coord	Yes	Specifies routing halo around the partition (honored by signal router). Positive values indicate the halo is outside the partition. Negative values indicate the halo is inside of the boundary of the partition and will be pushed into the partition when the partition is committed.
rHaloTopLayer	obj(layer)	Yes	Pointer to the top partition layer for which routing halo will be created
railWidth	coord	No	Specifies the cell rail width
reservedLayers	objList(layer)	Yes	List of pointers to the metal layers which are used for routing in the partition and generating partition pins. Any metal layers that are not specified, usually the top-most metal layers, are allowed to route over the partition

stdCellHeight	coord	Yes	Specifies the standard cell height for the partition

Note: In addition to the above entries, every object has an objType attribute

ptnCell

Parent Object

head, inst, pinGroup, term

Definition

Partition cell, created by partition command

Types and Definitions

Child Object or Attribute	Type	Edit	Description
defName	string	No	Fully qualified def name of the ptnCell from db
name	string	No	Name of cell
numBidirs	int	No	Number of bidir terminals in the cell
numInputs	int	No	Number of input terminals in the cell
numPGTerms	int	No	Number of power/ground terminals in the cell
numPhysTerms	int	No	Number of physical terminals in the cell
numRefs	int	No	Number of times cell is used in design
numTerms	int	No	Number of terminals in the cell
pgTerms	objList(term)	No	List of pointers to power/ground terminals in the cell
physTerms	objList(term)	No	List of pointers to physical (unused) signal terminals in the cell
pinToCornerDist	list(int)	No	List of distance constraints (in tracks) of pins from topCell/ptnCell corners where the lower left corner is listed first and the remaining corners are listed in clockwise order.
props	objList(prop)	No	List of pointers to properties
symmetryR90	bool	Yes	Symmetry of the cell in R90
symmetryX	bool	Yes	Symmetry of the cell in X
symmetryY	bool	Yes	Symmetry of the cell in Y
terms	objList(term)	No	List of pointers to signal terminals in the cell

Note: In addition to the above entries, every object has an objType attribute

ptnPinBlkg

Parent Object

fPlan

Definition

List of pointers to partition pin blockage objects

Types and Definitions

Child Object or Attribute	Type	Edit	Description
box	rect	No	Partition pin blockage rectangle
box_area	area	No	Area of partition pin blockage rectangle
box_ll	pt	No	Lower left (ll) of partition pin blockage rectangle
box_llx	coord	No	Lower left X (llx) of partition pin blockage rectangle
box_lly	coord	No	Lower left Y (lly) of partition pin blockage rectangle
box_size	pt	No	Size of partition pin blockage rectangle
box_sizex	coord	No	Size X of partition pin blockage rectangle
box_sizey	coord	No	Size Y of partition pin blockage rectangle
box_ur	pt	No	Upper right (ur) of partition pin blockage rectangle
box_urx	coord	No	Upper Right X (urx) of partition pin blockage rectangle
box_ury	coord	No	Upper Right Y (ury) of partition pin blockage rectangle
layer	obj(layer)	No	Pointer to the layer that is being blocked
name	string	No	Partition pin blockage name

Note: In addition to the above entries, every object has an objType attribute

pWire

Parent Object

net

Definition

Wire patch (equivalent to DEF NETS wiring RECT).

Types and Definitions

Child Object or Attribute	Type	Edit	Description
box	rect	No	Bounding box of the shape
boxTrimMetal	rect	No	The box of this patch wire trim metal
boxTrimMetal_area	area	No	Area of the box of this patch wire trim metal
boxTrimMetal_ll	pt	No	Lower left (ll) of the box of this patch wire trim metal
boxTrimMetal_llx	coord	No	Lower left X (llx) of the box of this patch wire trim metal
boxTrimMetal_lly	coord	No	Lower left Y (lly) of the box of this patch wire trim metal
boxTrimMetal_size	pt	No	Size of the box of this patch wire trim metal
boxTrimMetal_sizex	coord	No	Size X of the box of this patch wire trim metal
boxTrimMetal_sizey	coord	No	Size Y of the box of this patch wire trim metal
boxTrimMetal_ur	pt	No	Upper right (ur) of the box of this patch wire trim metal
boxTrimMetal_urx	coord	No	Upper Right X (urx) of the box of this patch wire trim metal
boxTrimMetal_ury	coord	No	Upper Right Y (ury) of the box of this patch wire trim metal
box_area	area	No	Area of bounding box of the shape
box_ll	pt	No	Lower left (ll) of bounding box of the shape
box_llx	coord	No	Lower left X (llx) of bounding box of the shape
box_lly	coord	No	Lower left Y (lly) of bounding box of the shape
box_size	pt	No	Size of bounding box of the shape
box_sizex	coord	No	Size X of bounding box of the shape

box_sizey	coord	No	Size Y of bounding box of the shape
box_ur	pt	No	Upper right (ur) of bounding box of the shape
box_urx	coord	No	Upper Right X (urx) of bounding box of the shape
box_ury	coord	No	Upper Right Y (ury) of bounding box of the shape
colorTrimMetal	int	No	The color of this patch wire trim metal
hasTrimMetal	bool	No	Indicates the wire has trim metal or not
layer	obj(layer)	No	Pointer to layer of patch
mask	int	Yes	Indicates mask number for multiple mask layer usage. Refer to layer's .numMask attribute for legal range, 0 indicates uncolored.
net	obj(net)	No	Pointer to net that the patch belongs to
pt	pt	No	Reference point to symbolic location
pt_x	coord	No	X of reference point to symbolic location
pt_y	coord	No	Y of reference point to symbolic location
rule	obj(rule)	No	Pointer to non-default rule corresponding to the wire, wires with the default routing rule will return NULL (0x0).
status	enum	Yes	Wiring status (equivalent to DEF NETS regular wiring status) **Legal enum**: cover, fixed, noshield, routed, unknown

Note: In addition to the above entries, every object has an objType attribute

rBlkg

Parent Object

fPlan

Definition

A routing blockage that corresponds to DEF routing BLOCKAGES. By default, a routing blockage prevents routing on the same layer in the area of the blockage. verify_drc and routeDesign will treat it like min-width routing for spacing checks unless other attributes are set as described for each attribute

Types and Definitions

Child Object or Attribute	Type	Edit	Description
area	area	No	The area of the routing blockage.
attr	enum	Yes	The routing blockage type. 'default' means the blockage is completely blocked and verify_drc will treat it like min-width routing for spacing checks. 'partial' means a percentage of the routing resource is available on the layer (see density attribute). 'fills' means do not add any metal fill in the area, and 'slots' means do not add slots to the area (slots is not used in Innovus but is allowed to match DEF). The type corresponds to the DEF BLOCKAGES + FILLS or + SLOTS value. There is no DEF syntax for partial, so it is currently lost when written to DEF. **Legal enum**: default, fills, partial, slots
boxes	list(rect)	No	List of rectangles that define the shape of routing blockage. If the blockage is defined by a polygon, the list of rectangles is derived from the polygon
density	int	Yes	The density percentage allowed for a partial routing blockage. It causes the global router to only use up to this percent of the routing resource on the layer in the blockage area, so the global router will see higher congestion and put fewer routes on that layer in the area.

designRuleWidth	coord	Yes	Specifies that the blockage has an effective width for the purposes of spacing calculations to other shapes on the same layer. A value of 0 indicates that there is no specified designRuleWidth value. The designRuleWidth and spacing attributes are not allowed to have non-default values at the same time, so the designRuleWidth value cannot be changed when the spacing attribute's value is not the default.
inst	obj(inst)	No	Pointer to the instance that the routing blockage is associated with (equivalent to DEF BLOCKAGES + COMPONENT)
isExceptPGNet	bool	Yes	Indicates that Power/Ground routing is ignored when checking for DRC violations (including shorts) involving the current shape (equivalent to DEF BLOCKAGES + EXCEPTPGNET)
isPGNetOnly	bool	Yes	Indicates this blockage only affects special-routes for Power/Ground nets. Other nets, and regular routing of PG nets are ignored when checking for DRC violations with this blockage (there is currently no DEF equivalent). Setting this value will clear isExceptPGNet if it is set.
isPushdown	bool	Yes	Indicates that routing blockage has been pushed down from a higher level in the design hierarchy. It is assumed to match the exact route shape from above, and is treated as a real routing shape by verify_drc and routeDesign rather than as min-width routing. It is equivalent to DEF BLOCKAGES + PUSHDOWN.
layer	obj(layer)	Yes	Pointer to layer of blockage
name	string	Yes	Name of the routing blockage. The same name can be given to many routing blockages to make it easy to track blockages for different purposes.
shapes	objList(shape)	No	List of pointers to shapes that define the blockage area

spacing	coord	Yes	Specifies the minimum spacing allowed between the blockage and any other shape on the same layer. A value = -1 (DBU value, micron value will be -1/[dbGet head.dbUnits]) indicates that there is no specified spacing value. The designRuleWidth and spacing attributes are not allowed to have non-default values at the same time, so the spacing value cannot be changed when the designRuleWidth attribute's value is not the default. To reset the value use dbSet -d <rblkgPtr>.spacing -1.

Note: In addition to the above entries, every object has an objType attribute

resistor

Parent Object

No Parents

Definition

Object Resistor

Types and Definitions

Child Object or Attribute	Type	Edit	Description
capacitance	list(double)	No	The capacitance for the two nodes before any effects of set_rail_what_if_capacitance in units of farads.
current	double	No	The current of the resistor in units of A. For static rail_analysis it is the average value, for dynamic power analysis it is the rms value.
name	string	No	This is an R followed by an integer to identify the resistor.
net	obj(net)	No	The net of the resistor.
nodeNames	string	No	This is an N followed by an integer to identify the two nodes at each end of the resistor.
nodeReff	list(double)	No	The values are in units of ohms and are valid whenever the node reff db is available in the loaded state directory.
nodeVoltageDrop	list(double)	No	The IR voltage drop of the two nodes in units of mV.
nodeVoltages	list(double)	No	The IR voltage drop of the two nodes in units of mV.
pt	list(pt)	No	The x, y location of the two nodes at each end of the resistor.
resistance	double	No	The resistance before any effects from set_rail_what_if_resistance in units of ohms.
switchNet	obj(net)	No	The switched net of the resistor.
whatIfCapacitance	list(double)	No	The capacitance after any effects of set_rail_what_if_capacitance. This is the capacitance actually used in the rail analysis in units of farads
whatIfResistance	double	No	The resistance after any effects of set_rail_what_if_resistance. This is the resistance actually used in the rail analysis in units of ohms.

Note: In addition to the above entries, every object has an objType attribute

resizeBlkg

Parent Object

fPlan

Definition

resizeBlkg

Types and Definitions

Child Object or Attribute	Type	Edit	Description
area	area	No	Area of the size blockage as defined by the LEF MACRO SIZE or OVERLAP information
box	rect	No	Rectangle that indicates the bounding box of the size blockage
box_area	area	No	Area of rectangle that indicates the bounding box of the size blockage
box_ll	pt	No	Lower left (ll) of rectangle that indicates the bounding box of the size blockage
box_llx	coord	No	Lower left X (llx) of rectangle that indicates the bounding box of the size blockage
box_lly	coord	No	Lower left Y (lly) of rectangle that indicates the bounding box of the size blockage
box_size	pt	No	Size of rectangle that indicates the bounding box of the size blockage
box_sizex	coord	No	Size X of rectangle that indicates the bounding box of the size blockage

box_sizey	coord	No	Size Y of rectangle that indicates the bounding box of the size blockage
box_ur	pt	No	Upper right (ur) of rectangle that indicates the bounding box of the size blockage
box_urx	coord	No	Upper Right X (urx) of rectangle that indicates the bounding box of the size blockage
box_ury	coord	No	Upper Right Y (ury) of rectangle that indicates the bounding box of the size blockage
isResizeable	bool	Yes	Specifies that the size blockage can be resized, however alignment and the minimum space between the objects in the blockage area will be maintained during floorplan resize.(1 = resizeable).
name	string	Yes	Name of the size blockage

Note: In addition to the above entries, every object has an objType attribute

routeType

Parent Object

head

Definition

Net attributes that configure routing for a class of clock net

Types and Definitions

Child Object or Attribute	Type	Edit	Description
botMaskLayerNum	int	Yes	Specify the bottom layer number that the mask constraint should be applied.

botOneSideLayerNum	int	Yes	Specify the bottom layer number that one side spacing constrain should be applied on. By default, NDR spacing is applied to both sides of an NDR net or wire. Use this to specify the layer range for wires that have only one neighboring wire with minimum spacing, which means only one side needs to follow larger NDR spacing.
bottomPreferredLayer	obj(layer)	No	Pointer to the preferred lowest routing layer. This attribute is a soft limit; that is, NanoRoute might use a layer below the specified layer if necessary to complete routing.
emNdrDist	double	Yes	Specify the distance related to the output pin, when emNdrRule is applied to the net. When routing outside this distance range, the router either uses regular wire or regular no-default routing, if the regular NDR setting is enabled in this net.
emNdrRule	string	Yes	Specifies the EM NDR rule to associate with this route type. When routing within the distance specified in -emDist from output pin, router will use this NDR rule to route. By default, it is NO EM NDR rule.
isTable	bool	No	is the routetype table based
mask	int	Yes	Indicates mask number for multiple mask layer usage. Refer to layers .numMask attribute for valid range, 0 indicates unconstrained. Layers that do not support the specified value will be treated as unconstrained. (Legal range: 0-3).
minStackLayer	obj(layer)	Yes	The net should use a stacked via from output pins up to the given layer before starting normal routing. This is normally used to force the routing to higher layers with wider widths to reduce wiring resistance or avoid EM current limits for high-drive outputs. If outputStackViaRule or inputStackViaRule is also specified, the specified stack via rule is used for the input or output pins accordingly. If both outputStackViaRule/inputStackViaRule are not specified, a single-cut stacked-via will be used for the output pins only.

name	string	No	Name of routeType object.
prefMultiCutVia	bool	Yes	The pref multi cut via rule attribute of this route type.
routingEffort	enum	Yes	routing effort **Legal enum**: high, low, medium
rule	obj(rule)	No	Pointer to non-default rule corresponding to the net, nets with the default routing rule will return NULL (0x0).
shield	obj(net)	No	shield net
shieldSide	enum	Yes	Specifies whether to perform one sided or two sided shielding for the route type specified. **Legal enum**: both_side, one_side
stackDistance	double	Yes	Specifies that the cut distance of cuts on adjacent layers in the stacked vias are defined in -minStackLayer.
topMaskLayerNum	int	Yes	Specify the top layer number that the mask constraint should be applied.
topOneSideLayerNum	int	Yes	Specify the top layer number that one side spacing constrain should be applied on. By default, NDR spacing is applied to both sides of an NDR net or wire. Use this to specify the layer range for wires that have only one neighboring wire with minimum spacing, which means only one side needs to follow larger NDR spacing.
topPreferredLayer	obj(layer)	No	Pointer to the preferred highest routing layer. This attribute is a soft limit; that is, NanoRoute might use a layer above the specified layer if necessary to complete routing.

Note: In addition to the above entries, every object has an objType attribute

row

Parent Object

fPlan

Definition

Row (core), constructed from sites (equivalent to DEF ROWS)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
area	area	No	Area of the row as defined by the LEF MACRO SIZE or OVERLAP information
box	rect	No	Bounding box of the row
box_area	area	No	Area of bounding box of the row
box_ll	pt	No	Lower left (ll) of bounding box of the row
box_llx	coord	No	Lower left X (llx) of bounding box of the row
box_lly	coord	No	Lower left Y (lly) of bounding box of the row
box_size	pt	No	Size of bounding box of the row
box_sizex	coord	No	Size X of bounding box of the row
box_sizey	coord	No	Size Y of bounding box of the row
box_ur	pt	No	Upper right (ur) of bounding box of the row
box_urx	coord	No	Upper Right X (urx) of bounding box of the row
box_ury	coord	No	Upper Right Y (ury) of bounding box of the row
name	string	No	Name of row (generated name)
numX	int	No	Number of sites in X direction (refer to DEF ROW syntax)
numY	int	No	Number of sites in Y direction (refer to DEF ROW syntax)
orient	enum	No	Orientation of the sites in the row **Legal enum**: MX, MX90, MY, MY90, R0, R180, R270, R90
site	obj(site)	No	Pointer to site used in the row
stepX	coord	No	Step in the X direction (refer to DEF ROW syntax)
stepY	coord	No	Step in the Y direction (refer to DEF ROW syntax)

Note: In addition to the above entries, every object has an objType attribute

rule

Parent Object

head, net, pWire, routeType, term, trackDef, viaInst, wire

Definition

Rule information (equivalent to LEF/DEF NONDEFAULTRULE)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
fromLib	bool	No	Indicates whether the rule came from the library technology (true: from LEF or OA tech) or from the design (false: from DEF, OA database, or createNdr).
hardSpacing	bool	No	Indicates that any spacing values that exceed the LEF LAYER spacing requirements are 'hard' rules instead of 'soft' rules.
layerRules	objList(layerRule)	No	List of pointers to the layers rules
minCuts	list(list)	No	List of cut layer and minimum number of cuts allowed for any via using the specified cut layer.
name	string	No	Name of non-default rule.
vias	objList(via)	No	List of pointers to via, default or USEVIA or derived from mincut

Note: In addition to the above entries, every object has an objType attribute

sdp

Parent Object

fPlan, inst,

Definition

A structured datapath object. Each sdp is formed hierarchically from a list of sdps below it. Each sdp can be a row, column, space or inst (see .type). At the leaf-level, an sdp can only be a space or inst. See the createSdpGroup command for more help.

Types and Definitions

Child Object or Attribute	Type	Edit	Description
area	area	No	Area of the data path as defined by the LEF MACRO SIZE or OVERLAP information
box	rect	No	Bounding box of the data path.
box_area	area	No	Area of bounding box of the data path.
box_ll	pt	No	Lower left (ll) of bounding box of the data path.
box_llx	coord	No	Lower left X (llx) of bounding box of the data path.
box_lly	coord	No	Lower left Y (lly) of bounding box of the data path.
box_size	pt	No	Size of bounding box of the data path.
box_sizex	coord	No	Size X of bounding box of the data path.
box_sizey	coord	No	Size Y of bounding box of the data path.
box_ur	pt	No	Upper right (ur) of bounding box of the data path.
box_urx	coord	No	Upper Right X (urx) of bounding box of the data path.
box_ury	coord	No	Upper Right Y (ury) of bounding box of the data path.

flip	enum	No	Flip SDP object.Its members will also be flipped. **Legal enum**: FlipX, FlipXY, FlipY, UnknownFlip
hierName	string	No	Name of hierarchical instance (only for isTop = 1).
insts	objList(insts)	No	Pointer to the instances members in the sdp group.
isPlaced	bool	No	Indicates that the data path is placed (only for isTop = 1).
isTop	bool	No	Indicates that the data path is a top data path group.
justifyBy	enum	No	Alignment **Legal enum**: E, MidDir, N, NE, NW, S, SE, SW, W
name	string	No	Name of Data Path.
orientation	enum	No	Orientation **Legal enum**: MX, MX90, MY, MY90, R0, R180, R270, R90, UnknownOrient
parent	obj(sdp)	No	Return parent sdp group that the sdp belongs to.
**Note	Types and Definitions	No	
pinAlignment	enum	No	Pin alignment (only available if pins are not NULL). **Legal enum**: E, Mid, Unknown, W
pinNames	list(string)	No	List of pin names (Not available if pinAlignment = Unknown).
pt	pt	No	Lower left location
pt_x	coord	No	X of lower left location
pt_y	coord	No	Y of lower left location
sdps	objList(inst, sdp)	No	List of pointers to row, column, or sdp objects or inst object in current sdp group. This attribute is null/0x0 if the current sdp is an inst object or a space type sdp object.
size	pt	No	Size of data path relative to lower left corner of SDP.
size_x	coord	No	X of size of data path relative to lower left corner of SDP.
size_y	coord	No	Y of size of data path relative to lower left corner of SDP.

space	int	No	empty space value (only for type = space).
treeInsts	objList(inst)	No	Get tree instances of sdp group.
type	enum	No	Type of Data Path object **Legal enum**: column, row, space, unknown

Note: In addition to the above entries, every object has an objType attribute

shape

Parent Object

layerShape, pBlkg, rBlkg

Definition

Shape

Types and Definitions

Child Object or Attribute	Type	Edit	Description
mask	int	No	Indicates mask number for multiple mask layer usage. Refer to layer's .numMask attribute for legal range, 0 indicates uncolored.
path	list(pt)	No	Points of the path (if type = path)
poly	list(pt)	No	Points of the polygon (if type = polygon)
rect	rect	No	Bounding box of shape
rect_area	area	No	Area of bounding box of shape
rect_ll	pt	No	Lower left (ll) of bounding box of shape
rect_llx	coord	No	Lower left X (llx) of bounding box of shape
rect_lly	coord	No	Lower left Y (lly) of bounding box of shape
rect_size	pt	No	Size of bounding box of shape
rect_sizex	coord	No	Size X of bounding box of shape
rect_sizey	coord	No	Size Y of bounding box of shape
rect_ur	pt	No	Upper right (ur) of bounding box of shape
rect_urx	coord	No	Upper Right X (urx) of bounding box of shape
rect_ury	coord	No	Upper Right Y (ury) of bounding box of shape
type	enum	No	Type of shape (path, rect, polygon) **Legal enum**: dim, path, poly, rect

Note: In addition to the above entries, every object has an objType attribute

shapeVia

Parent Object

libCell, pin

Definition

layer Shape via

Types and Definitions

Child Object or Attribute	Type	Edit	Description
botMask	int	Yes	Indicates mask number for bottom layer for multiple mask layer usage. Refer to layer's .numMask attribute for legal range, 0 indicates uncolored
botRects	list(rect)	No	List of rectangles (typically only one) on bottom routing layer in terms of design coordinates (equivalent attribute on the via master is in coordinates local to the via master)
cutMask	int	Yes	Indicates mask number for cut layer for multiple mask layer usage. Applies to lower left cut of the via, other cuts are rotated from the reference cut in the lower left corner. Refer to layer's .numMask attribute for legal range, 0 indicates uncolored
cutRects	list(rect)	No	List of rectangles on cut layer in terms of design coordinates (equivalent attribute on the via master is in coordinates local to the via master)
pt	pt	No	Via location
pt_x	coord	No	X of via location
pt_y	coord	No	Y of via location
topMask	int	Yes	Indicates mask number for top layer for multiple mask layer usage. Refer to layer's .numMask attribute for legal range, 0 indicates uncolored
topRects	list(rect)	No	List of rectangles (typically only one) on top routing layer in terms of design coordinates (equivalent attribute on the via master is in coordinates local to the via master)
via	obj(via)	No	Pointer to via master

Note: In addition to the above entries, every object has an objType attribute

site

Parent Object

fPlan, head, libCell, row

Definition

site

Types and Definitions

Child Object or Attribute	Type	Edit	Description
class	enum	No	Site class (equivalent to LEF SITE CLASS) **Legal enum**: core, pad
name	string	No	Name of site (equivalent to LEF SITE NAME)
size	pt	No	Size of the site (equivalent to LEF SITE SIZE)
size_x	coord	No	X of size of the site (equivalent to LEF SITE SIZE)
size_y	coord	No	Y of size of the site (equivalent to LEF SITE SIZE)
symmetryR90	bool	Yes	Symmetry of the site in R90 (equivalent to LEF SITE SYMMETRY)
symmetryX	bool	Yes	Symmetry of the site in X (equivalent to LEF SITE SYMMETRY)
symmetryY	bool	Yes	Symmetry of the site in Y (equivalent to LEF SITE SYMMETRY)

Note: In addition to the above entries, every object has an objType attribute

stackViaRule

Parent Object

instTerm, term

Definition

Stack Via Rule

Types and Definitions

Child Object or Attribute	Type	Edit	Description
name	string	No	Stack Via Rule name.

Note: In addition to the above entries, every object has an objType attribute

sViaInst

Parent Object

net

Definition

Special Via (equivalent to DEF SPECIALNETS via instances)

Types and Definitions

Child Object or Attribute	Type	Edit	Description

botMask	int	Yes	Is the mask number for the lower, left shape on the bottom layer of the via. Normally there is only one shape on the bottom layer of a via, but if there are two or more bottom layer shapes, then the mask for the other shapes on the bottom layer are derived from the corresponding via-master mask values by "shifting" the via-master's mask values to match. See the DEF manual section on 'Multi-Mask Layers with Special Wiring' for figures and examples. A value of 0 indicates the bottom layer is uncolored, or the layer is not a multi-mask layer.
botRects	list(rect)	No	List of rectangles (typically only one) on bottom routing layer in terms of design coordinates (equivalent attribute on the via master is in coordinates local to the via master)
botRectsMask	list(int)	No	List of mask values for each rect in botRects in the same order as botRects. A value of 0 means it is uncolored, or this layer is not a multi-mask layer.
cutMask	int	Yes	Is the mask number for the lower, left cut of the via. The mask for the other cuts of the sViaInst are derived from the via-master by "shifting" the via master's cut masks to match. So, if the via-master lower, left cut is mask 1, and the sViaInst cutMask is set to 3, then all the via-master cuts on mask 1 become mask 3 for this sViaInst and similarly cuts on 2 shift to 1, and cuts on 3 shift to 2. See the layer .numMasks attribute for the max mask value allowed. A value of 0 indicates the cut is uncolored if this layer has multiple masks. See the DEF manual section on 'Multi-Mask Layers with Special Wiring' for figures and more examples. A value of 0 indicates the cut layer is uncolored, or the layer is not a multi-mask layer.
cutRects	list(rect)	No	List of rectangles on cut layer in terms of design coordinates (equivalent attribute on the via master is in coordinates local to the via master)
cutRectsMask	list(int)	No	List of mask values for each rect in cutRects in the same order as cutRects. A value of 0 means it is uncolored, or this layer is not a multi-mask layer.
net	obj(net)	No	Pointer to net that the sViaInst belongs to

props	objList(prop)	No	List of pointers to properties
pt	pt	No	Location of Via
pt_x	coord	No	X of location of Via
pt_y	coord	No	Y of location of Via
shape	enum	Yes	DEF SPECIALNETS + SHAPE equivalents (ring, stripe, etc.) **Legal enum**: blockRing, blockagewire, blockwire, corewire, drcfill, fillopc, fillwire, followpin, iowire, notype, padRing, ring, stripe
shieldNet	obj(net)	No	Pointer to net that is shielded if status is shieldNet
status	enum	Yes	Wiring status (equivalent to DEF SPECIALNETS special wiring status) **Legal enum**: cover, fixed, routed, shield, unknown
topMask	int	Yes	Is the mask number for the lower, left shape on the top layer of the via. Normally there is only one shape on the top layer of a via, but if there are two or more top layer shapes, then the mask for the other shapes on the top layer are derived from the corresponding via-master mask values by "shifting" the via-master's mask values to match. See the DEF manual section on 'Multi-Mask Layers with Special Wiring' for figures and examples. A value of 0 indicates the top layer is uncolored, or the layer is not a multi-mask layer.
topRects	list(rect)	No	List of rectangles (typically only one) on top routing layer in terms of design coordinates (equivalent attribute on the via master is in coordinates local to the via master)
topRectsMask	list(int)	No	List of mask values for each rect in topRects in the same order as topRects. A value of 0 means it is uncolored, or this layer is not a multi-mask layer.
userClass	string	Yes	Subclass value ({} is returned if no subClass is specified)
via	obj(via)	No	Pointer to via master

Note: In addition to the above entries, every object has an objType attribute

sWire

Parent Object

net

Definition

Special wire (equivalent to DEF SPECIALNETS wiring)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
area	area	No	Area of the special wire as defined by the LEF MACRO SIZE or OVERLAP information
beginExt	coord	No	Extension of path at the first point (only on path type)
box	rect	Yes	Bounding box for shapes, actual shape for geomType = rect
boxTrimMetal	rect	No	The box of this special wire trim metal
boxTrimMetal_area	area	No	Area of the box of this special wire trim metal
boxTrimMetal_ll	pt	No	Lower left (ll) of the box of this special wire trim metal
boxTrimMetal_llx	coord	No	Lower left X (llx) of the box of this special wire trim metal
boxTrimMetal_lly	coord	No	Lower left Y (lly) of the box of this special wire trim metal
boxTrimMetal_size	pt	No	Size of the box of this special wire trim metal
boxTrimMetal_sizex	coord	No	Size X of the box of this special wire trim metal
boxTrimMetal_sizey	coord	No	Size Y of the box of this special wire trim metal

boxTrimMetal_ur	pt	No	Upper right (ur) of the box of this special wire trim metal
boxTrimMetal_urx	coord	No	Upper Right X (urx) of the box of this special wire trim metal
boxTrimMetal_ury	coord	No	Upper Right Y (ury) of the box of this special wire trim metal
box_area	area	No	Area of bounding box for shapes, actual shape for geomType = rect
box_ll	pt	Yes	Lower left (ll) of bounding box for shapes, actual shape for geomType = rect
box_llx	coord	Yes	Lower left X (llx) of bounding box for shapes, actual shape for geomType = rect
box_lly	coord	Yes	Lower left Y (lly) of bounding box for shapes, actual shape for geomType = rect
box_size	pt	Yes	Size of bounding box for shapes, actual shape for geomType = rect
box_sizex	coord	Yes	Size X of bounding box for shapes, actual shape for geomType = rect
box_sizey	coord	Yes	Size Y of bounding box for shapes, actual shape for geomType = rect
box_ur	pt	Yes	Upper right (ur) of bounding box for shapes, actual shape for geomType = rect
box_urx	coord	Yes	Upper Right X (urx) of bounding box for shapes, actual shape for geomType = rect
box_ury	coord	Yes	Upper Right Y (ury) of bounding box for shapes, actual shape for geomType = rect
colorTrimMetal	int	No	The color of this specical wire trim metal
endExt	coord	No	Extension of path at the last point (only on path type)
geomType	enum	No	Type of shape (rect, polygon, etc.) **Legal enum**: path, pathSeg, poly, rect

hasTrimMetal	bool	No	Indicates the speical wire has trim metal or not
layer	obj(layer)	Yes	Pointer to layer
mask	int	Yes	Indicates mask number for multiple mask layer usage. Refer to layer's .numMask attribute for legal range, 0 indicates uncolored.
net	obj(net)	No	Pointer to net that the sWire belongs to
polyPts	list(pt)	No	Polygon boundary for the object, the first point is not repeated as the last point in the list
props	objList(prop)	No	List of pointers to properties
pts	list(pt)	No	2 points for pathSeg center-line, n points for path center-line, n points for polygon, null for rect
shape	enum	Yes	DEF SPECIALNETS + SHAPE equivalents (ring, stripe, etc.) **Legal enum**: blockRing, blockagewire, blockwire, corewire, drcfill, fillopc, fillwire, followpin, iowire, notype, padRing, ring, stripe
shieldNet	obj(net)	No	Pointer to net that is shielded if status is shieldNet
status	enum	Yes	Wiring status (equivalent to DEF SPECIALNETS special wiring status) **Legal enum**: cover, fixed, routed, shield, unknown
userClass	string	Yes	Subclass value ({} is returned if no subClass is specified)
width	coord	Yes	Width of path/pathSeg type

Note: In addition to the above entries, every object has an objType attribute

term

Parent Object

bump, bumpTerm, bus, busSinkGroup, hInstTerm, hTerm, instTerm, io, libCell, net, pgInstTerm, pinGroup, pinShape, ptnCell, topCell

Definition

Terminal for libCell, ptnCell, or topCell

Types and Definitions

Child Object or Attribute	Type	Edit	Description
antennas	objList(antennaData)	No	List of pointers to antenna data for the terminal
bus	obj(bus)	No	Pointer to bus object, Null (0x0) for scalar terminal
cell	obj(libCell, ptnCell, topCell)	No	Pointer to the cell (libCell, topCell, or ptnCell) that contains the terminal
defName	string	No	Fully qualified def name of the terminal from db
depth	coord	No	Depth of terminal (not for libCell terms)
direction	enum	No	Specifies the direction of the power or ground terminal from liberty data. PG terminals with no liberty entry will have 'invalid'. **Legal enum**: bidi, input, internal, output, outputTriState, unknown

edge	coord	No	If this term is for a ptnCell (for an hinst that is a ptn), topCell, or inst that is a black-box, and the term is assigned (has pStatus of placed/fixed/cover) the edge value indicates along which edge of the boundary polygon the term is assigned. The edge number starts from the lowest Y, then left-most X vertex, staring with 0, and then counting clock-wise. See the setPinConstraint command document for a figure showing the edge numbering. If the term is not assigned, or not for a ptnCell, topCell, or black-box inst, the value of -1 is returned.
effectiveStackViaRule	obj(stackViaRule)	No	The stackViaRule that is expected, but not required, to be used by the router for connecting to the instance terms instantiated from this term. The actual stack via rule used (if any) may be effected by other instTerm and term attributes, or by choices made by the software (optimization, clock tree synthesis, the router etc.)
groundSensitivity	obj(term)	No	Pointer to reference ground terminal (LEF MACRO PIN GROUNDSENSITIVITY equivalent). Typically only used if the cell contains more than one ground terminal. Null (0x0) indicates default assumption will be used for reference terminal.
hdlName	string	Yes	This is the original RTL name for this term.
inOutDir	enum	No	Terminal direction **Legal enum**: bidi, input, internal, output, outputTriState, unknown
isAnalog	bool	No	Specifies the terminal is an analog signal. This attribute can be set by liberty files.
isClk	bool	No	Indicates that the terminal is a clock terminal

isInput	bool	No	Indicates that the terminal is an input
isIsolated	bool	No	Specifies the terminal is isolated internally in the cell. It is used for cells where some pins are internally isolated and some are not. This attribute can be set by liberty files.
isIsolationCellEnable	bool	No	Identifies the terminal is an isolation enable terminal. This terminal is used to control when to clamp the output and isolate it from the input. This attribute can be set by liberty files, or by CPF commands.
isLevelShifterEnable	bool	No	Identifies the terminal is a level shifter enable terminal. This terminal is used to control when to clamp the output and isolate it from the input. This attribute can be set by liberty files, or by CPF commands.
isOutput	bool	No	Indicates that the terminal is an output
isRetentionEnable	bool	No	Identifies the terminal as a retention cell enable terminal. This terminal is used to control when to retain the state and ignore other inputs. This attribute can be set by liberty files, or by CPF commands.
isScanClk	bool	No	Indicates that the terminal is a scan clock terminal
isSpecial	bool	Yes	Indicates that the terminal is Special(not set for libCell terms)
isSwitchEnable	bool	No	Identifies the terminal is a power switch enable terminal. This terminal is used to control when to turn on/off the power switch. This attribute can be set by liberty files, or by CPF commands.
isTieHi	bool	No	Indicates that the terminal is a tieHi

isTieLo	bool	No	Indicates that the terminal is a tieLo
isUnconnected	bool	No	Specifies the terminal is floating internally. This is used for cells where some of the inputs or outputs are unused by the cell. This attribute can be set by liberty files.
isViaInPinOnly	bool	No	Indicates that the pin has a LEF VIAINPINONLY property. It means that vias must be dropped inside the original pin shapes to connect to the pin. In some advanced nodes, the pin shapes can be extended for metal alignment purposes. However, via insertion is not allowed in that extended portion if this attribute is true.
layer	obj(layer)	Yes	Pointer to terminal layer
mustJoin	obj(term)	No	Pointer to must join term
name	string	No	Terminal name
net	obj(net)	No	Pointer to canonical (flat) net connected to the terminal
orient	enum	No	Orientation of the terminal **Legal enum**: MX, MX90, MY, MY90, R0, R180, R270, R90
pStatus	enum	Yes	Placement Status of terminal (not for libCell terms) **Legal enum**: cover, fixed, placed, softFixed, unplaced
pd	obj(pd)	No	Pointer to the power domain of the terminal (equivalent to Design Browser effPD)

pgType	enum	No	Specifies the type of the power or ground terminal from liberty data. Signal terminals and PG pins with no liberty entry will have 'invalid'. **Legal enum**: backupGround, backupPower, deepnWell, deeppWell, internalGround, internalPower, invalid, nWell, pWell, primaryGround, primaryPower
physicalConnection	string	No	Specifies the physical connection of the term.
pinShapes	objList(pinShape)	No	All the individual physical pin shapes of the term.
pins	objList(pin)	No	List of pointers to ports
props	objList(prop)	No	List of pointers to properties
pt	pt	Yes	Location of terminal (not for libCell terms)
pt_x	coord	Yes	X of location of terminal (not for libCell terms)
pt_y	coord	Yes	Y of location of terminal (not for libCell terms)
relatedBiasTerm	obj(term)	No	Specifies which bias terminal of this signal terminal. It must be one of the bias terminals defined for this cell. CPF, IEEE1801, liberty, or LEF/OA can set this, with CPF/IEEE1801 having highest precedence, then liberty, then LEF/OA.
relatedGroundTerm	obj(term)	No	Specifies which ground terminal drives this signal terminal. It must be one of the ground terminals defined for this cell. CPF, IEEE1801, liberty, or LEF/OA can set this, with CPF/IEEE1801 having highest precedence, then liberty, then LEF/OA.

relatedPowerTerm	obj(term)	No	Specifies which power terminal drives this signal terminal. It must be one of the power terminals defined for this cell. CPF, IEEE1801, liberty, or LEF/OA can set this, with CPF/IEEE1801 having highest precedence, then liberty, then LEF/OA.
rule	obj(rule)	No	Pointer to non-default rule of the terminal. This attribute is the equivalent of the LEF MACRO PIN TAPERRULE construct and only applies to terminals of libCells (LEF MACROs). A value of 0x0 will be return if the TAPERRULE is not specified (implies tapering the default wiring rule)
shape	enum	No	Terminal shape **Legal enum**: abutment, feedThru, none, ring
side	enum	No	Side constraint of terminal (not for libCell terms) **Legal enum**: East, None, North, South, Up, West
stackViaRequired	bool	Yes	Specifies whether a stack via is required when connecting to the instance terms instantiated from this term. If true, one of the stackViaRule from stackViaRuleList must be used to generate a stack via even if a design rule violation occurs. If false, a stack via is optional. Note that the instTerm stackViaRuleRequired value may override this term setting.
stackViaRuleList	string	Yes	List of stackViaRule name that are valid choices for connecting to the instance terms instantiated from this term. If the list is empty, no stack via is allowed.

supplySensitivity	obj(term)	No	Pointer to reference power terminal (LEF MACRO PIN SUPPLYSENSITIVITY equivalent). Typically only used if the cell contains more than one power terminal. Null (0x0) indicates default assumption will be used for reference terminal.
tiedTo	string	No	Specifies the PG pin name or 'empty' which the PG term tied to.
type	enum	No	Terminal type **Legal enum**: analogTerm, asyncCtrlTerm, clockTerm, dQTerm, dTerm, fFQTerm, feedTerm, gatedClockTerm, groundTerm, latchQTerm, normalTerm, powerTerm, rSTerm, triStateTerm
width	coord	No	Width of terminal (not for libCell terms)

Note: In addition to the above entries, every object has an objType attribute

text

Parent Object

topCell

Definition

Text

Types and Definitions

Child Object or Attribute	Type	Edit	Description
alignment	enum	No	Horizontal and vertical alignment of the text string **Legal enum**: centerCenter, centerLeft, centerRight, lowerCenter, lowerLeft, lowerRight, upperCenter, upperLeft, upperRight
drafting	bool	No	Indicates if the text is always displayed left-to-right or top-to-bottom. Text will remain readable even if rotated and mirrored if this value is true.
fontName	enum	No	Font name **Legal enum**: euroStyle, fixed, gothic, math, milSpec, roman, script, stick, swedish
fontNum	enum	No	Font Number **Legal enum**: 0, 1, 2, 3, 4, 5, 6, 7, 8
height	coord	Yes	Text height
label	string	Yes	Text string value
layer	obj(layer)	Yes	Pointer to layer
oaPurpose	string	Yes	User specified purpose name for OpenAccess text layer purpose pair support. Only values available that exist in the library's tech graph are allowed. The default value is 'drawing'.
orient	enum	Yes	Text Orientation **Legal enum**: MX, MX90, MY, MY90, R0, R180, R270, R90
pt	pt	Yes	Text location
pt_x	coord	Yes	X of text location
pt_y	coord	Yes	Y of text location

Note: In addition to the above entries, every object has an objType attribute

topCell

Parent Object

fPlan, head, pinGroup, pinGuide, term

Definition

Top cell, container for flattened connectivity

Types and Definitions

Child Object or Attribute	Type	Edit	Description
bumps	objList(bump)	No	List of pointers to bumps in the cell
designLib	string	No	Library name for the design in OpenAccess cellview
designView	string	No	View name for the design in OpenAccess cellview
fPlan	obj(fplan)	No	Pointer to the floorplan
hInst	obj(hInst)	No	Pointer to the top level hierarhical instance
insts	objList(inst)	No	List of pointers to instances in the cell
isProtoModelCommitted	bool	No	Indicates that the design has committed FlexModels.
isProtoModelSpecified	bool	No	Indicates that the design has specified FlexModels.
markers	objList(marker)	No	List of pointers to Markers
name	string	No	Name of cell

nets	objList(net)	No	List of pointers to canonical (flat) nets in the cell
numBidirs	int	No	Number of bidir terminals in the cell
numInputs	int	No	Number of input terminals in the cell
numInsts	int	No	Number of instances in the cell
numNets	int	No	Number of canonical (flat) nets in the cell
numPGTerms	int	No	Number of power/ground terminals in the cell
numPhysInsts	int	No	Number of physical instances in the cell
numPhysNets	int	No	Number of physical nets in the cell
numPhysTerms	int	No	Number of physical terminals in the cell
numTerms	int	No	Number of terminals in the cell
pds	objList(pd)	No	List of pointers to power domains (pd) in the design
pgNets	objList(net)	No	List of pointers to power ground net in the design
pgTerms	objList(term)	No	List of pointers to power/ground terminals in the cell
physInsts	objList(inst)	No	List of pointers to physical instances in the cell
physNets	objList(net)	No	List of pointers to physical nets in the cell
physTerms	objList(term)	No	List of pointers to physical (unused) signal terminals in the cell

pinToCornerDist	list(int)	No	List of distance constraints (in tracks) of pins from topCell/ptnCell corners where the lower left corner is listed first and the remaining corners are listed in clockwise order.
pkgComponents	objList(pkgComponent)	No	List of other chips in the package
pkgObjects	objList(pkgObject)	No	List of objects in the package
props	objList(prop)	No	List of pointers to properties
ptns	objList(ptn)	No	List of pointers to partitions in the design
readOnly	enum	No	This attribute is set by set_module_view to identify if the top_level is read_only or not. It means the partition cannot be optimized, and cells inside will not be moved. Setting this attribute will set the dont_touch_effective attribute on all insts and hinsts within the top level partition unless overridden at a lower level partition. It cannot be overridden by other hinst or inst values. **Legal enum**: false, none, true
restoreDbDesignName	string	No	The design name of the restored design.
restoreDbToolName	string	No	The tool name save the restored design in current session.
restoreDbToolVersion	string	No	The tool version number save the restored design in current session.
statusClockSynthesized	bool	No	Design status: Clock synthesized
statusDetailRouted	bool	No	Design status: detail routed, true if routeDesign was run.
statusIoPlaced	bool	No	Design status: IOs are Placed
statusPlaced	bool	No	Design status: Placed

statusPowerAnalyzed	bool	No	Design status: Power analyzed
statusRCExtracted	bool	No	Design status: Parasitic extracted
statusRouted	bool	No	Design status: routed. true if design has global routes from placeDesign/earlyGlobalRoute or detail routes from routeDesign.
statusScanOpted	bool	No	Design status: Scan optimized
symmetryR90	bool	Yes	Symmetry of the cell in R90
symmetryX	bool	Yes	Symmetry of the cell in X
symmetryY	bool	Yes	Symmetry of the cell in Y
terms	objList(term)	No	List of pointers to signal terminals in the cell
texts	objList(text)	No	List of pointers to text

Note: In addition to the above entries, every object has an objType attribute

trackDef

Parent Object

fPlan

Definition

Floorplan track information (DEF TRACKS equivalent)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
dir	enum	No	Specifies the location and direction of the first track defined. x indicates vertical lines; y indicates horizontal lines. **Legal enum**: x, y
isSameMask	bool	No	Indicates that all of the tracks are the same mask instead of alternating (1,2 for two mask layers, and 1,2,3 for three mask layers)
layers	objList(layer)	No	List of Pointers to layer
mask	int	No	Specifies the mask number for the first track
numTracks	int	No	Specifies the number of tracks to create for the grid
rule	obj(rule)	No	Pointer to rule object (DEF NONDEFAULTRULE) to be used as a constraint for wiring that can be created on the track. A null/0x0 value indicates that there is no constraint on the rules of the wires on the track. This value can be set with add_tracks and is intended for advanced nodes that do not allow different rules on the same track for lower routing layers.
start	coord	No	Specifies the coordinate of the first line
step	coord	No	Specifies the spacing between the tracks
width	coord	No	Width constraint for wiring that can be created on the track. A value of 0 indicates that there is no constraint on the width of wires on the track. This value can be set with add_tracks and is intended for advanced nodes that do not allow different widths on the same track for lower routing layers.

Note: In addition to the above entries, every object has an objType attribute

vCell

Parent Object

hInst, head

Definition

Intermediate module cell (equivalent to intermediate module cells in Verilog or OpenAccess module domain)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
allowIlmEco	bool	Yes	The attribute is only valid when the module cell is an ILM. If true, optimizer can optimize the ILM boundary interface logic to improve timing.
attributes	list(string)	No	List of Verilog module attributes in the form { {attr1 value1} {attr2 value2} ...}
dontTouch	enum	Yes	This attributes defines the user preservation status of the module during optimization. Setting this attribute will set the dont_touch attribute on all hinsts of the same module. This setting will apply to all insts within the hinst unless overridden at a lower level hinst or on the inst object itself. The dont_touch_effective attribute on each child inst and hinst will return the resolved value. **Legal enum**: constPropDeleteOk, constPropSizeDeleteOk, deleteOk, false, none, sizeDeleteOk, sizeOk, sizeSameFootprintOk, sizeSameHeightOk, true

dontTouchHports	enum	Yes	This attribute defines the user preservation status for the module object hports during optimization. **Legal enum**: addInvertOk, addOk, deleteOk, false, invertOk, mapSizeOk, none, true
dontUseCells	list(string)	Yes	List of base_cell names (wildcards supported) to disallow for this module during optimization. Setting this applies to all hinsts sharing the module. If a cell is added to this list that is already in the .use_cells list, it will be removed from the .use_cells list so that the lists are non-overlapping. Setting on hinsts of this module will update this attribute.
dontUseCellsEffective	string	No	The resolved list of all cell names to disallow during optimization for hinsts of this module, based on the library dont_use and the dont_use_cells and use_cells attributes of this module or the closest parent hinst with non-empty lists.
hInst	obj(hInst)	No	Pointer to the corresponding hierarchical instance If the netlist data is not uniquified, the pointer will be to one of the hierarchical instances.
isIlm	bool	No	This attribute is true if the module is a ILM. This attribute will affect the read_only_effective and dont_touch_effective attribute on all insts and hinsts within the hinsts of this module. It cannot be overridden by other hinst or inst values.
name	string	No	Name of cell
props	objList(prop)	No	List of pointers to properties
useCells	list(string)	Yes	List of base_cell names to allow for this module during optimization. Setting this applies to all hinsts sharing the module. All lib_cells of each base_cell will be allowed. If a cell is added to this list that is already in the .dont_use_cells list, it will be removed from the .dont_use_cells list so that the lists are non-overlapping.

Note: In addition to the above entries, every object has an objType attribute

via

Parent Object

head, rule, sViaInst, shapeVia, viaInst

Definition

Via cell (equivalent to LEF VIA or DEF VIA)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
botLayer	obj(layer)	No	Pointer to bottom routing layer
botRects	list(rect)	No	List of rectangles (typically only one) on bottom layer
botRectsMask	list(int)	Yes	List of mask values for each rect in botRects in the same order as botRects. A value of 0 means it is uncolored, or this layer is not a multi-mask layer.
cutClass	string	No	Returns the name of the CUTCLASS definition (from LEF or OA tech) for the cutRects in this via. It returns an empty string if no CUTCLASS exists for the cut layer, or no CUTCLASS matches the size of the cutRects.
cutColumns	int	No	The number of cut columns. It is only set for generated vias created from viaRuleGenerate parameters. See the LEF/DEF manual VIA definition with VIARULE and ROWCOL values for more details. It is 0 for fixed vias (e.g. a LEF/DEF VIA definition with only RECT values).
cutLayer	obj(layer)	No	Pointer to cut layer
cutRects	list(rect)	No	List of rectangles on cut layer

cutRectsMask	list(int)	Yes	List of mask values for each rect in cutRects in the same order as cutRects. A value of 0 means it is uncolored, or this layer is not a multi-mask layer.
cutRows	int	No	The number of cut rows. It is only set for generated vias created from viaRuleGenerate parameters. See the LEF/DEF manual VIA definition with VIARULE and ROWCOL values for more details. It is 0 for fixed vias (e.g. a LEF/DEF VIA definition with only RECT values).
cutSize	string	No	The {width height} of the first rect in cutRects in microns.
fromDesign	bool	No	Indicates that the via is from design
fromLib	bool	No	Indicates that the via is from library
isDefault	bool	No	Indicates that the via is a default via (LEF VIA DEFAULT)
isNonDefault	bool	No	Indicates that the via is declared in a LEF NONDEFAULTRULE
name	string	No	Via name
resistance	float	No	Via resistance in ohms that is derived from LEF or OA data. This may not match the resistance of RC extraction results derived from extraction coefficient data. If the via is a fixed via with a resistance value defined in the LEF VIA definition statement or OA oaViaDef, that value is returned. For vias without a resistance value, the resistance is computed from the cut-layer resistance_per_cut value and the number of cuts in the via (or equivalent cuts for a LEF CUTCLASS or OA cutClass via with different cut sizes). If both the via definition, and the cut-layer has no resistance value, then 0.0 is returned
topLayer	obj(layer)	No	Pointer to top routing layer
topRects	list(rect)	No	List of rectangles (typically only one) on top routing layer
topRectsMask	list(int)	Yes	List of mask values for each rect in topRects in the same order as topRects. A value of 0 means it is uncolored, or this layer is not a multi-mask layer.

viaRuleGenerate	obj(via)	No	The viaRuleGenerate for this via. It is only set for generated vias created from viaRuleGenerate parameters. See the LEF/DEF manual VIA statement with VIARULE for more details. It returns 0x0 for fixed vias (e.g. a LEF/DEF VIA with only RECT values).

Note: In addition to the above entries, every object has an objType attribute

viaInst

Parent Object

net

Definition

viaInst (equivalent to via in DEF NETS wiring)

Types and Definitions

Child Object or Attribute	Type	Edit	Description
botMask	int	Yes	Is the mask number for the lower, left shape on the bottom layer of the via. Normally there is only one shape on the bottom layer of a via, but if there are two or more bottom layer shapes, then the mask for the other shapes on the bottom layer are derived from the corresponding via-master mask values by "shifting" the via-master's mask values to match. See the DEF manual section on 'Multi-Mask Patterns for Routing Points' for figures and examples. A value of 0 indicates the bottom layer is uncolored, or the layer is not a multi-mask layer.
botRects	list(rect)	No	List of rectangles (typically only one) on bottom routing layer in terms of design coordinates (equivalent attribute on the via master is in coordinates local to the via master)

botRectsMask	list(int)	No	List of mask values for each rect in botRects in the same order as botRects. A value of 0 means it is uncolored, or this layer is not a multi-mask layer.
cutMask	int	Yes	Is the mask number for the lower, left cut of the via. The mask for the other cuts of the viaInst are derived from the via-master by "shifting" the via master's cut masks to match. So, if the via-master lower, left cut is mask 1, and the viaInst cutMask is set to 3, then all the via-master cuts on mask 1 become mask 3 for this viaInst and similarly cuts on 2 shift to 1, and cuts on 3 shift to 2. See the layer .numMasks attribute for the max mask value allowed. A value of 0 indicates the cut is uncolored if this layer has multiple masks. See the DEF manual section on 'Multi-Mask Patterns for Routing Points' for figures and more examples. A value of 0 indicates the cut layer is uncolored, or the layer is not a multi-mask layer.
cutRects	list(rect)	No	List of rectangles on cut layer in terms of design coordinates (equivalent attribute on the via master is in coordinates local to the via master)
cutRectsMask	list(int)	No	List of mask values for each rect in cutRects in the same order as cutRects. A value of 0 means it is uncolored, or this layer is not a multi-mask layer.
net	obj(net)	No	Pointer to net that the via belongs to
pt	pt	No	Location of Via
pt_x	coord	No	X of location of Via
pt_y	coord	No	Y of location of Via
rule	obj(rule)	No	Pointer to non-default rule corresponding to the via, vias with the default routing rule will return NULL (0x0).
status	enum	Yes	Wiring status (equivalent to DEF NETS regular wiring status) **Legal enum**: cover, fixed, noshield, routed, unknown

topMask	int	Yes	Is the mask number for the lower, left shape on the top layer of the via. Normally there is only one shape on the top layer of a via, but if there are two or more top layer shapes, then the mask for the other shapes on the top layer are derived from the corresponding via-master mask values by "shifting" the via-master's mask values to match. See the DEF manual section on 'Multi-Mask Patterns for Routing Points' for figures and examples. A value of 0 indicates the top layer is uncolored, or the layer is not a multi-mask layer.
topRects	list(rect)	No	List of rectangles (typically only one) on top routing layer in terms of design coordinates (equivalent attribute on the via master is in coordinates local to the via master)
topRectsMask	list(int)	No	List of mask values for each rect in topRects in the same order as topRects. A value of 0 means it is uncolored, or this layer is not a multi-mask layer.
via	obj(via)	No	Pointer to via cell

Note: In addition to the above entries, every object has an objType attribute

viaRuleGenerate

Parent Object

head, via

Definition

Via rule information defined in the LEF or OpenAccess techfile

Types and Definitions

Child Object or Attribute	Type	Edit	Description

botEnclosure	pt	No	Two minimum enclosure values for the cuts in the via. The order of the two values does not matter. The bottom layer shape must enclose all the cuts by one of the enclosure values in one direction (e.g. either X or Y), and by the other enclosure value in the other direction. If it is {0 0}, which is recommended for newer technologies, then only the DRC rules are used to compute the minimum enclosure..
botEnclosure_x	coord	No	X of two minimum enclosure values for the cuts in the via. The order of the two values does not matter. The bottom layer shape must enclose all the cuts by one of the enclosure values in one direction (e.g. either X or Y), and by the other enclosure value in the other direction. If it is {0 0}, which is recommended for newer technologies, then only the DRC rules are used to compute the minimum enclosure..
botEnclosure_y	coord	No	Y of two minimum enclosure values for the cuts in the via. The order of the two values does not matter. The bottom layer shape must enclose all the cuts by one of the enclosure values in one direction (e.g. either X or Y), and by the other enclosure value in the other direction. If it is {0 0}, which is recommended for newer technologies, then only the DRC rules are used to compute the minimum enclosure..
botLayer	obj(layer)	No	List of pointers to the layers rules
botWidth	pt	No	Optional min and max width. If given, this rule should only be used if the bottom wire width is greater than or equal to the first value (min-width), and less than or equal to the second value (max-width). For example, {1 2} means min-width is >= 1.0, and max_width is <= 2.0). If not given, the default is {0 0}, which means this rule can be used for any width.
botWidth_x	coord	No	X of optional min and max width. If given, this rule should only be used if the bottom wire width is greater than or equal to the first value (min-width), and less than or equal to the second value (max-width). For example, {1 2} means min-width is >= 1.0, and max_width is <= 2.0). If not given, the default is {0 0}, which means this rule can be used for any width.

botWidth_y	coord	No	Y of optional min and max width. If given, this rule should only be used if the bottom wire width is greater than or equal to the first value (min-width), and less than or equal to the second value (max-width). For example, {1 2} means min-width is >= 1.0, and max_width is <= 2.0). If not given, the default is {0 0}, which means this rule can be used for any width.
cutLayer	obj(layer)	No	The cut layer for the via.
cutRect	rect	No	The size of one cut rectangle.
cutRect_area	area	No	Area of the size of one cut rectangle.
cutRect_ll	pt	No	Lower left (ll) of the size of one cut rectangle.
cutRect_llx	coord	No	Lower left X (llx) of the size of one cut rectangle.
cutRect_lly	coord	No	Lower left Y (lly) of the size of one cut rectangle.
cutRect_size	pt	No	Size of the size of one cut rectangle.
cutRect_sizex	coord	No	Size X of the size of one cut rectangle.
cutRect_sizey	coord	No	Size Y of the size of one cut rectangle.
cutRect_ur	pt	No	Upper right (ur) of the size of one cut rectangle.
cutRect_urx	coord	No	Upper Right X (urx) of the size of one cut rectangle.
cutRect_ury	coord	No	Upper Right Y (ury) of the size of one cut rectangle.
cutSpacing	pt	No	Minimum center-to-center spacing in the X and Y directions..
cutSpacing_x	coord	No	X of minimum center-to-center spacing in the X and Y directions..
cutSpacing_y	coord	No	Y of minimum center-to-center spacing in the X and Y directions..
name	string	No	Via rule name.

resistancePerCut	double	No	Optional via resistance per cut in ohms that is defined in LEF or OA data. This value is useful for estimation, but will not match the resistance extracted by RC extraction commands that use more accurate coefficient files. It is 0.0 if not given in the library data.
topEnclosure	pt	No	Two minimum enclosure values for the cuts in the via. The order of the two values does not matter. The top layer shape must enclose all the cuts by one of the enclosure value in one direction (e.g. either X or Y), and by the other enclosure value in the other direction. If it is {0 0}, which is recommended, then only the DRC rules are used to compute the minimum enclosure.
topEnclosure_x	coord	No	X of two minimum enclosure values for the cuts in the via. The order of the two values does not matter. The top layer shape must enclose all the cuts by one of the enclosure value in one direction (e.g. either X or Y), and by the other enclosure value in the other direction. If it is {0 0}, which is recommended, then only the DRC rules are used to compute the minimum enclosure.
topEnclosure_y	coord	No	Y of two minimum enclosure values for the cuts in the via. The order of the two values does not matter. The top layer shape must enclose all the cuts by one of the enclosure value in one direction (e.g. either X or Y), and by the other enclosure value in the other direction. If it is {0 0}, which is recommended, then only the DRC rules are used to compute the minimum enclosure.
topLayer	obj(layer)	No	The top routing layer for the via.
topMinWidth	coord	No	Optional min-width. If given, this rule should only be used if the top wire width is greater than or equal to this value. If not given, the default is 0, which means this rule can be used for any width.
topWidth	pt	No	Optional min and max width. If given, this rule should only be used if the top wire width is greater than or equal to the first value (min-width), and less than or equal to the second value (max-width). For example,. {1 2} means min-width is >= 1.0, and max_width is <= 2.0). If not given, the default is {0 0}, which means this rule can be used for any width.

topWidth_x	coord	No	X of optional min and max width. If given, this rule should only be used if the top wire width is greater than or equal to the first value (min-width), and less than or equal to the second value (max-width). For example,. {1 2} means min-width is >= 1.0, and max_width is <= 2.0). If not given, the default is {0 0}, which means this rule can be used for any width.
topWidth_y	coord	No	Y of optional min and max width. If given, this rule should only be used if the top wire width is greater than or equal to the first value (min-width), and less than or equal to the second value (max-width). For example,. {1 2} means min-width is >= 1.0, and max_width is <= 2.0). If not given, the default is {0 0}, which means this rule can be used for any width.

Note: In addition to the above entries, every object has an objType attribute

vWire

Parent Object

net

Definition

Wire virtual connection (equivalent to DEF NETS wiring VIRTUAL).

Types and Definitions

Child Object or Attribute	Type	Edit	Description
beginLayer	obj(layer)	No	Pointer to begin layer for the virtual connection
beginPt	pt	No	Reference point to the begin symbolic location
beginPt_x	coord	No	X of reference point to the begin symbolic location
beginPt_y	coord	No	Y of reference point to the begin symbolic location
endLayer	obj(layer)	No	Pointer to end layer for the virtual connection
endPt	pt	No	Reference point to the end symbolic location
endPt_x	coord	No	X of reference point to the end symbolic location
endPt_y	coord	No	Y of reference point to the end symbolic location
net	obj(net)	No	Pointer to net that the virtual connection belongs to
status	enum	Yes	Wiring status (equivalent to DEF NET regular wiring status) **Legal enum**: cover, fixed, noshield, routed, unknown

Note: In addition to the above entries, every object has an objType attribute

whatIfVia

Parent Object

net

Definition

After loading a Voltus IR-drop analysis result, you can use create_what_if_shape to add 'what if' power vias to see how they would improve the IR-drop results without modifying the real power-mesh.

Types and Definitions

Child Object or Attribute	Type	Edit	Description
botLayer	obj(layer)	No	The bottom routing layer for this whatIfVia.
cutLayer	obj(layer)	No	The cut layer for this whatIfVia.
cutRects	list(rect)	No	The rects on the cut layer for this whatIfVia.
net	obj(net)	No	The net of this whatIfVia.
topLayer	obj(layer)	No	The top routing layer of this whatIfVia.

Note: In addition to the above entries, every object has an objType attribute

whatIfWire

Parent Object

net

Definition

After loading a Voltus IR-drop analysis result, you can use create_what_if_shape to add 'what if' power wires to see how they would improve the IR-drop results without modifying the real power-mesh.

Types and Definitions

Child Object or Attribute	Type	Edit	Description
box	rect	No	Bounding box of the wire.
box_area	area	No	Area of bounding box of the wire.
box_ll	pt	No	Lower left (ll) of bounding box of the wire.
box_llx	coord	No	Lower left X (llx) of bounding box of the wire.
box_lly	coord	No	Lower left Y (lly) of bounding box of the wire.
box_size	pt	No	Size of bounding box of the wire.
box_sizex	coord	No	Size X of bounding box of the wire.
box_sizey	coord	No	Size Y of bounding box of the wire.
box_ur	pt	No	Upper right (ur) of bounding box of the wire.
box_urx	coord	No	Upper Right X (urx) of bounding box of the wire.
box_ury	coord	No	Upper Right Y (ury) of bounding box of the wire.
layer	obj(layer)	No	The layer of the wire.
net	obj(net)	No	The net of the wire.
width	coord	No	The width of the wire.

Note: In addition to the above entries, every object has an objType attribute

wire

Parent Object

net

Definition

Wire (symbolic wire type, equivalent to DEF NETS wiring). Represents a wire segment.

Types and Definitions

Child Object or Attribute	Type	Edit	Description
area	area	No	Area of the wire as defined by the LEF MACRO SIZE or OVERLAP information
beginExt	coord	Yes	Extension of wire at the first point
box	rect	No	Bounding box of the shape
box_area	area	No	Area of bounding box of the shape
box_ll	pt	No	Lower left (ll) of bounding box of the shape
box_llx	coord	No	Lower left X (llx) of bounding box of the shape
box_lly	coord	No	Lower left Y (lly) of bounding box of the shape
box_size	pt	No	Size of bounding box of the shape
box_sizex	coord	No	Size X of bounding box of the shape
box_sizey	coord	No	Size Y of bounding box of the shape
box_ur	pt	No	Upper right (ur) of bounding box of the shape
box_urx	coord	No	Upper Right X (urx) of bounding box of the shape
box_ury	coord	No	Upper Right Y (ury) of bounding box of the shape
direction	enum	No	Direction of wire, consistent with layer direction from LEF/OpenAccess **Legal enum**: Horizontal, Other, Vertical
endExt	coord	Yes	Extension of wire at the second point
layer	obj(layer)	No	Pointer to layer of wire

length	coord	No	Centerline length of wire between the endpoints (not the box width/height)
mask	int	Yes	Indicates mask number for multiple mask layer usage. Refer to layer's .numMask attribute for legal range, 0 indicates uncolored.
net	obj(net)	No	Pointer to net that the wire belongs to
pts	list(pt)	No	2 points (center-line) for the wire
rule	obj(rule)	No	Pointer to non-default rule corresponding to the wire, wires with the default routing rule will return NULL (0x0).
status	enum	Yes	Wiring status (equivalent to DEF NETS regular wiring status) **Legal enum**: cover, fixed, noshield, routed, unknown
width	coord	No	Width of wire

Note: In addition to the above entries, every object has an objType attribute