Innovus Technology File

lef/def

DATABASE MICRONS **2000** ;

```
innovus #> dbGet head.dbUnits
    2000
```

dbu2uu

database integer value -> its user unit floating point equivalent

uu2dbu

user unit floating point value -> its database integer value equivalent

VIARULE

tlef	def
UNITS	VIAS 51 ;
CAPACITANCE PICOFARADS 1 ;	- M9_M8_1
DATABASE MICRONS **2000** ;	+ **VIARULE M9_M8**
END UNITS	+ CUTSIZE 140 140
VIARULE **M9_M8** GENERATE DEFAULT	+ LAYERS Metal8 Via8 Metal9
LAYER Metal8 ;	+ CUTSPACING 220 140
ENCLOSURE 0.005 0.03 ;	+ ENCLOSURE 30 130 30 130
LAYER Via8 ;	+ ROWCOL **71 56**
RECT -0.035 -0.035 0.035 0.035 ;	;
SPACING 0.14 BY 0.14 ;	- M9_M8_2
RESISTANCE 0.200000 ;	+ **VIARULE M9_M8**
LAYER Metal9 ;	+ CUTSIZE 140 140
ENCLOSURE 0.005 0.03 ;	+ LAYERS Metal8 Via8 Metal9
END M9_M8	+ CUTSPACING 220 140
	+ ENCLOSURE 10 130 10 130
VIARULE M8_M7 GENERATE DEFAULT	+ ROWCOL **71 45**
LAYER Metal7 ;	;
ENCLOSURE 0.005 0.03 ;	- M8_M7_1
LAYER Via7 ;	+ VIARULE M8_M7
RECT -0.035 -0.035 0.035 0.035 ;	+ CUTSIZE 140 140
SPACING 0.14 BY 0.14 ;	+ LAYERS Metal7 Via7 Metal8
RESISTANCE 5.000000 ;	+ CUTSPACING 140 220
LAYER Metal8 ;	+ ENCLOSURE 90 70 90 70
ENCLOSURE 0.005 0.03 ;	+ ROWCOL 28 57
END M8_M7	

Via

VIA viaName [DEFAULT]

```
        { VIARULE viaRuleName ;
        CUTSIZE xSize ySize ;
        LAYERS botMetalLayer cutLayer topMetalLayer ;
        CUTSPACING xCutSpacing yCutSpacing ;
        ENCLOSURE xBotEnc yBotEnc xTopEnc yTopEnc ;
        [ROWCOL numCutRows numCutCols ;]
        [ORIGIN xOffset yOffset ;]
        [OFFSET xBotOffset yBotOffset xTopOffset yTopOffset ;]
        [PATTERN cutPattern ;]
        }
        | {[RESISTANCE resistValue ;]
        {LAYER layerName ;
        { RECT [MASK maskNum] pt pt ;
        | POLYGON [MASK maskNum] pt pt pt ...;} ...
        } ...
        }
        [PROPERTY propName propVal ;] ...
    END viaName
```

Defines two types of vias: **fixed vias** and **generated vias**. All vias consist of shapes on three layers: a cut layer and two routing (or masterslice) layers that connect through that cut layer.

1) A fixed via is defined using rectangles or polygons, and **does not use a VIARULE**

    The fixed via name must mean the same via in all associated LEF and DEF files.

2) A generated via is defined using VIARULE parameters to indicate that it was derived from a **VIARULE GENERATE** statement.

DEFAULT

Identifies the via as the default via between the defined layers. Default vias are used for default routing by the signal routers.

CUTSIZE xSize ySize

Specifies the required width (xSize) and height (ySize) of the cut layer rectangles

CUTSPACING xCutSpacing yCutSpacing

Specifies the required x and y spacing between cuts. The spacing is measured from one cut edge to the next cut edge.

ENCLOSURE xBotEnc yBotEnc xTopEnc yTopEnc

Specifies the required x and y enclosure values for the bottom and top metal layers. The enclosure measures the distance from the **cut array edge** to the **metal edge** that encloses the cut array.

LAYERS botMetalLayer cutLayer topMetalLayer

Specifies the required names of the bottom routing (or masterslice) layer, cut layer, and top routing (or masterslice) layer. These layer names must be previously defined in layer definitions, and must match the layer names defined in the specified **LEF viaRuleName**.

generated via only

LAYER layerName

Specifies the layer on which to create the **rectangles** that make up the via. All vias consist of shapes on three layers: a cut layer and two routing (or masterslice) layers that connect through that cut layer. There should be at least one RECT or POLYGON on each of the three layers.

fixed via only

VIARULE viaRulename

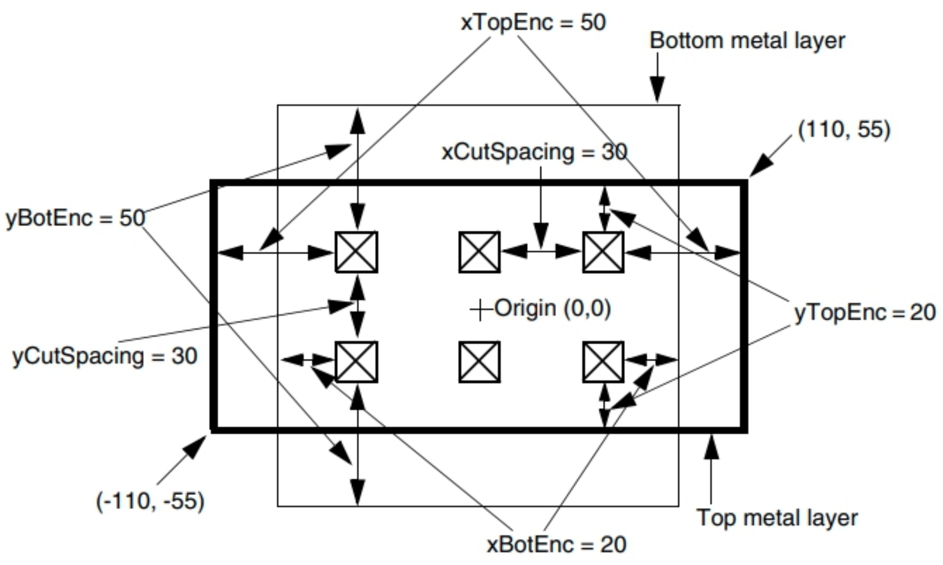
Specifies the name of the LEF VIARULE that produced this via. This indicates that the via is the result of automatic via generation, and that the via name is only used locally inside this LEF file. viaRuleName must be specified before you define any of the other parameters, and must refer to a previously defined VIARULE GENERATE rule name. It cannot refer to a VIARULE without a GENERATE keyword

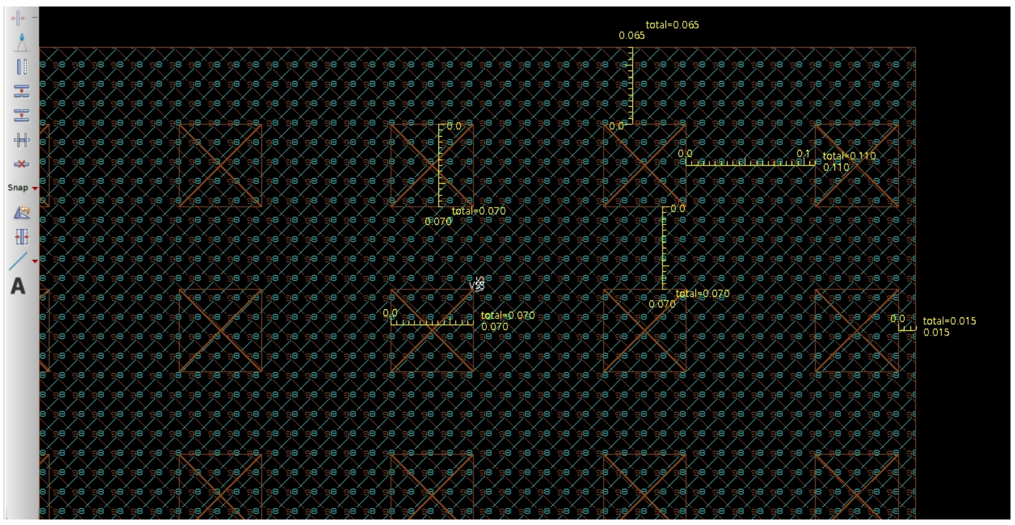
ROWCOL numCutRows numCutCols

Specifies the number of cut rows and columns that make up the via array.

Default: 1, for both values

Type: Positive integer, for both values





M9_M8_1 (via cell)

cutsize

```
innovus #> dbu2uu 140 140
    0.0700 0.0700
```

CUTSPACING

```
innovus #> dbu2uu 220 140
    0.1100 0.0700
```

ENCLOSURE

```
innovus #> dbu2uu 30 130 30 130
    0.0150 0.0650 0.0150 0.0650
```

Via Rule

```
VIARULE viaRuleName
    LAYER layerName ;
        DIRECTION {HORIZONTAL | VERTICAL} ;
        [WIDTH minWidth TO maxWidth ;]
    LAYER layerName ;
        DIRECTION {HORIZONTAL | VERTICAL} ;
        [WIDTH minWidth TO maxWidth ;]
    {VIA viaName ;} ...
    [PROPERTY propName propVal ;] ...
END viaRuleName
```

Defines which vias to use at the intersection of special wires of the same net.

Note:

You should only use VIARULE GENERATE statements to create a via for the intersection of two special wires. In earlier versions of LEF, VIARULE GENERATE was not complete enough to cover all situations. In those cases, a **fixed VIARULE (without a GENERATE keyword)** was sometimes used. This is no longer required

DIRECTION {HORIZONTAL | VERTICAL}

Specifies the wire direction. If you specify a WIDTH range, the rule applies to wires of the specified DIRECTION that fall within the range. Otherwise, the rule applies to all wires of the specified DIRECTION on the layer.

VIA viaName

Specifies a previously defined via to test for the current via rule. The first via in the list that can be placed at the location without design rule violations is selected. The vias must all have exactly three layers in them. The three layers must include the same routing or masterslice layers as listed in the LAYER statements of the VIARULE, and a cut layer that is between the two routing or masterslice layers.

e.g.

```
VIARULE viaRule1
    LAYER metal1 ;
        DIRECTION HORIZONTAL ;
        WIDTH 0.5 TO 1.0 ;
    LAYER metal2 ;
        DIRECTION VERTICAL ;
        WIDTH 1.0 TO 2.0 ;
    VIA via12_1 ;
    VIA via12_2 ;
END viaRule1
```

Via Rule Generate

```
VIARULE viaRuleName GENERATE [DEFAULT]
    LAYER routingLayerName ;
        ENCLOSURE overhang1 overhang2 ;
        [WIDTH minWidth TO maxWidth ;]
    LAYER routingLayerName ;
        ENCLOSURE overhang1 overhang2 ;
        [WIDTH minWidth TO maxWidth ;]
    LAYER cutLayerName ;
```

RECT pt pt ;
SPACING xSpacing BY ySpacing ;
[RESISTANCE resistancePerCut ;]
END viaRuleName

Defines formulas for **generating via arrays**. You can use the VIARULE GENERATE statement to cover special wiring that is not explicitly defined in the VIARULE statement. Rather than specifying a list of vias for the situation, you can create a formula to specify how to generate the cut layer geometries.

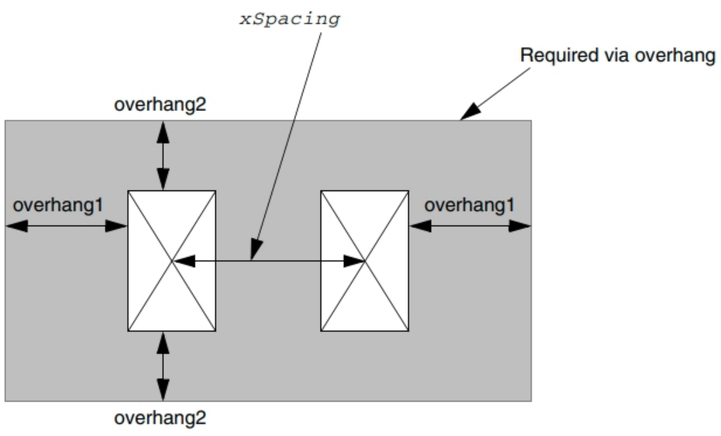
Note: Any vias created automatically from a VIARULE GENERATE rule that appear in the DEF NETS or SPECIALNETS sections must also appear in the DEF VIA section.

WIDTH minWidth TO maxWidth

Specifies a wire width range. If the widths of two intersecting special wires fall within the wire width range, the VIARULE is used. To fall within the range, the widths must be greater than or equal to minWidth and less than or equal to maxWidth

ENCLOSURE overhang1 overhang2

Specifies that the via must be covered by metal on two opposite sides by **at least** overhang1, and on the other two sides by at least overhang2 (see Figure 1-54 on page 161). The via generation code then chooses the direction of overhang that best maximizes the number of cuts that can fit in the via.



SPACING xSpacing BY ySpacing

Defines **center-to-center spacing** in the x and y dimensions to create an array of contact cuts.The number of cuts of an array in each direction is the most that can fit within the bounds of the intersection formed by the two special wires. Cuts are only generated where they do not violate stacked or adjacent via design rules.

Note: This value can be overridden by the SPACING ADJACENTCUTS value in the cut layer statement.

The cut layer SPACING ADJACENTCUTS statement can override the VIARULE cut layer SPACING statements.

Note: The spacing in VIARULE GENERATE is **center-to-center** spacing, whereas the spacing in ADJACENTCUTS is **edge-to-edge**.

DEFAULT

Specifies that the via rule can be used to generate vias for the default routing rule. There can only be one VIARULE GENERATE DEFAULT for a given routing-cut-routing (or masterslice-cut-masterslice) layer combination.

Layer (Cut)

LAYER layerName
TYPE CUT ;
[MASK maskNum ;]
[SPACING cutSpacing

```
    [CENTERTOCENTER]
    [SAMENET]
    [ LAYER secondLayerName [STACK]
    | ADJACENTCUTS {2 | 3 | 4} WITHIN cutWithin [EXCEPTSAMEPGNET]
    | PARALLELOVERLAP
    | AREA cutArea
    ]
    ;] ...
END layerName
```

Defines cut layers in the design. Each cut layer is defined by assigning it a name and design rules. You must define cut layers separately, with their own layer statements.

SPACING

Specifies the **minimum** spacing allowed between via cuts on the same net or different nets.

The SPACING syntax is defined as follows:

```
[SPACING cutSpacing
[ LAYER secondLayerName [STACK]
| ADJACENTCUTS {2 | 3 | 4} WITHIN cutWithin
;] ...
```

cutSpacing

Specifies the **default minimum** spacing between via cuts, in microns.

Type: Float

ADJACENTCUTS {2 |3 | 4} WITHIN cutWithin

Applies the spacing rule only when the cut has two, three, or four via cuts that are less than cutWithin distance, in microns, from each other. You can specify only one ADJACENTCUTS statement per cut layer. For more information, see "Adjacent Via Cuts."

Type: Float (distance)

Adjacent Via Cuts

A cut is considered adjacent if it is within distance of another cut in any direction (including a 45-degree angle). The ADJACENTCUTS rule overrides the cut-to-cut spacing used in VIARULE GENERATE statements for large vias if the ADJACENTCUTS spacing value is larger than the VIARULE spacing value.

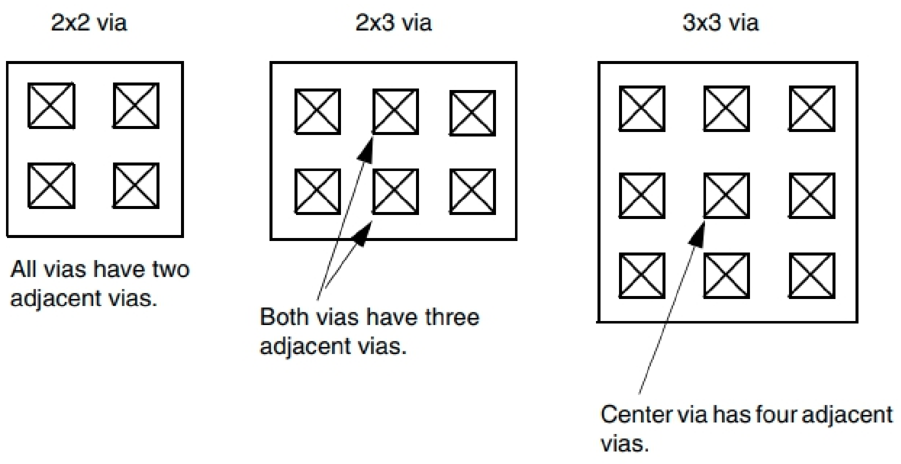
The following spacing rule specifies that extra space is needed for any via with more than three adjacent cuts, which happens if one via has more than 2x2 cuts (see Figure 2-6 on page 194). A cut that is within .25 μm of three other cuts requires spacing that is greater than or equal to 0.22 μm.

```
LAYER CUT12
    SPACING 0.20 ; #default cut spacing
    SPACING 0.22 ADJACENTCUTS 3 WITHIN 0.25 ;
    ...
END CUT12
```

The following spacing rule specifies that extra space is required for any via with 3x3 cuts or more (that is, a cut with four or more adjacent cuts – see Figure 2-6 on page 194). A cut that is within .25 μm of four other cuts requires spacing that is greater than or equal to 0.22 μm.

```
LAYER CUT12
```

```
        SPACING 0.20 ; #default cut spacing
        SPACING 0.22 ADJACENTCUTS 4 WITHIN 0.25 ;
        ...
    END CUT12
```


2x2 via
2x3 via
3x3 via
All vias have two adjacent vias.
Both vias have three adjacent vias.
Center via has four adjacent vias.

Nondefault Rule

```
    [NONDEFAULTRULE ruleName
        [HARDSPACING ;]
        {LAYER layerName
        WIDTH width ;
        [DIAGWIDTH diagWidth ;]
        [SPACING minSpacing ;]
        [WIREEXTENSION value ;]
        END layerName} ...
        [VIA viaStatement] ...
        [USEVIA viaName ;] ...
        [USEVIARULE viaRuleName ;] ...
        [MINCUTS cutLayerName numCuts ;] ...
        [PROPERTY propName propValue ;] ...
        [PROPERTY LEF58_USEVIACUTCLASS
          "USEVIACUTCLASS cutLayerName className
        [ROWCOL numCutRows numCutCols]
        ;...   " ;]
    END ruleName]
```

Defines the wiring width, design rule spacing, and via size for **regular (signal) nets**. You do not need to define cut layers for the non-default rule.

```
    NONDEFAULTRULE widewire

        LAYER Metal1
            WIDTH 0.48 ;
            SPACING 0.66 ;
        END Metal1

        LAYER Metal2
            WIDTH 0.96 ;
            SPACING 0.88 ;
        END Metal2
        LAYER Metal3
            WIDTH 0.96 ;
            SPACING 0.88 ;
        END Metal3
        ...
        VIA ndvia45
            LAYER Metal4 ; RECT -.22 -.22 .22 .22 ;
            LAYER Via45 ; RECT -.20 -.20 .20 .20 ;
            LAYER Metal5 ; RECT -.22 -.22 .22 .22 ;
        END ndvia45
        VIA ndvia56
            LAYER Metal5 ; RECT -.22 -.22 .22 .22 ;
            LAYER Via56 ; RECT -.20 -.20 .20 .20 ;
            LAYER Metal6 ; RECT -.22 -.22 .22 .22 ;
        END ndvia56
    END widewire
```