NAME

ext − format of .ext files produced by Magic's hierarchical extractor

DESCRIPTION

Magic's extractor produces a **.ext** file for each cell in a hierarchical design. The **.ext** file for cell *name* is *name***.ext**. This file contains three kinds of information: environmental information (scaling, timestamps, etc), the extracted circuit corresponding to the mask geometry of cell *name*, and the connections between this mask geometry and the subcells of *name*.

A **.ext** file consists of a series of lines, each of which begins with a keyword. The keyword beginning a line determines how the remainder of the line is interpreted. The following set of keywords define the environmental information:

tech *techname*
: Identifies the technology of cell *name* as *techname*, e.g, **nmos**, **cmos**.

timestamp *time*
: Identifies the time when cell *name* was last modified. The value *time* is the time stored by Unix, i.e, seconds since 00:00 GMT January 1, 1970. Note that this is *not* the time *name* was extracted, but rather the timestamp value stored in the **.mag** file. The incremental extractor compares the timestamp in each **.ext** file with the timestamp in each **.mag** file in a design; if they differ, that cell is re-extracted.

version *version*
: Identifies the version of **.ext** format used to write *name***.ext**. The current version is **5.1**.

style *style*
: Identifies the style that the cell has been extracted with.

scale *rscale cscale lscale*
: Sets the scale to be used in interpreting resistance, capacitance, and linear dimension values in the remainder of the **.ext** file. Each resistance value must be multiplied by *rscale* to give the real resistance in milliohms. Each capacitance value must be multiplied by *cscale* to give the real capacitance in attofarads. Each linear dimension (e.g, width, height, transform coordinates) must be multiplied by *lscale* to give the real linear dimension in centimicrons. Also, each area dimension (e.g, transistor channel area) must be multiplied by *scale*scale* to give the real area in square centimicrons. At most one **scale** line may appear in a **.ext** file. If none appears, all of *rscale*, *cscale*, and *lscale* default to 1.

resistclasses *r1 r2 ...*
: Sets the resistance per square for the various resistance classes appearing in the technology file. The values *r1*, *r2*, etc. are in milliohms; they are not scaled by the value of *rscale* specified in the **scale** line above. Each node in a **.ext** file has a perimeter and area for each resistance class; the values *r1*, *r2*, etc. are used to convert these perimeters and areas into actual node resistances. See ‘‘Magic Tutorial #8: Circuit Extraction’’ for a description of how resistances are computed from perimeters and areas by the program **ext2sim**.

The following keywords define the circuit formed by the mask information in cell *name*. This circuit is extracted independently of any subcells; its connections to subcells are handled by the keywords in the section after this one.

node *name R C x y type a1 p1 a2 p2 ... aN pN*
: Defines an electrical node in *name*. This node is referred to by the name *name* in subsequent **equiv** lines, connections to the terminals of transistors in **fet** lines, and hierarchical connections or adjustments using **merge** or **adjust**. The node has a total capacitance to ground of *C* attofarads, and a lumped resistance of *R* milliohms. For purposes of going back from the node name to the geometry defining the node, *(x, y)* is the coordinate of a point inside the node, and *type* is the layer on which this point appears. The values *a1*, *p1*, ... *aN*, *pN* are the area and perimeter for the

material in each of the resistance classes described by the **resistclasses** line at the beginning of the **.ext** file; these values are used to compute adjusted hierarchical resistances more accurately. **NOTE:** since many analysis tools compute transistor gate capacitance themselves from the transistor's area and perimeter, the capacitance between a node and substrate (GND!) normally does not include the capacitance from transistor gates connected to that node. If the **.sim** file was produced by *ext2sim* (1), check the technology file that was used to produce the original **.ext** files to see whether transistor gate capacitance is included or excluded; see ‘‘Magic Maintainer's Manual #2: The Technology File’’ for details.

attr *name xl yl xh yh type text*

One of these lines appears for each label ending in the character ‘‘@’’ that was attached to geometry in the node *name*. The location of each attribute label (*xl yl xh yh*) and the type of material to which it was attached (*type*) are given along with the text of the label minus the trailing ‘‘@’’ character (*text*).

equiv *node1 node2*

Defines two node names in cell *name* as being equivalent: *node1* and *node2*. In a collection of node names related by **equiv** lines, exactly one must be defined by a **node** line described above.

fet *type xl yl xh yh area perim sub GATE T1 T2 ...*

Defines a transistor in *name*. The kind of transistor is *type*, a string that comes from the technology file and is intended to have meaning to simulation programs. The coordinates of a square entirely contained in the gate region of the transistor are *(xl, yl)* for its lower-left and *(xh, yh)* for its upper-right. All four coordinates are in the *name*'s coordinate space, and are subject to scaling as described in **scale** above. The gate region of the transistor has area *area* square centimicrons and perimeter *perim* centimicrons. The substrate of the transistor is connected to node *sub*.

The remainder of a **fet** line consists of a series of triples: *GATE*, *T1*, ... Each describes one of the terminals of the transistor; the first describes the gate, and the remainder describe the transistor's non-gate terminals (e.g, source and drain). Each triple consists of the name of a node connecting to that terminal, a terminal length, and an attribute list. The terminal length is in centimicrons; it is the length of that segment of the channel perimeter connecting to adjacent material, such as polysilicon for the gate or diffusion for a source or drain.

The attribute list is either the single token ‘‘0’’, meaning no attributes, or a comma-separated list of strings. The strings in the attribute list come from labels attached to the transistor. Any label ending in the character ‘‘^’’ is considered a gate attribute and appears on the gate's attribute list, minus the trailing ‘‘^’’. Gate attributes may lie either along the border of a channel or in its interior. Any label ending in the character ‘‘**$**’’ is considered a non-gate attribute. It appears on the list of the terminal along which it lies, also minus the trailing ‘‘**$**’’. Non-gate attributes may only lie on the border of the channel.

The keywords in this section describe information that is not processed hierarchically: path lengths and accurate resistances that are computed by flattening an entire node and then producing a value for the flattened node.

killnode *node*

During resistance extraction, it is sometimes necessary to break a node up into several smaller nodes. The appearance of a **killnode** line during the processing of a **.ext** file means that all information currently accumulated about *node*, along with all fets that have a terminal connected to *node*, should be thrown out; it will be replaced by information later in the **.ext** file. The order of processing **.ext** files is important in order for this to work properly: children are processed before their parents, so a **killnode** in a parent overrides one in a child.

resist *node1 node2 R*

Defines a resistor of *R* milliohms between the two nodes *node1* and *node2*. Both names are hierarchical.

distance *name1 name2 dmin dmax*

Gives the distance between two electrical terminals *name1* (a driver) and *name2* (a receiver). Note that these are terminals and not nodes: the names (which are hierarchical label names) are used to specify two different locations on the same electrical node. The two distances, *dmin* and *dmax*, are the lengths (in lambda) of the shortest and longest acyclic paths between the driver and receiver.

The keywords in this last section describe the subcells used by *name*, and how connections are made to and between them.

use *def use-id TRANSFORM*

Specifies that cell *def* with instance identifier *use-id* is a subcell of cell *name*. If cell *def* is arrayed, then *use-id* will be followed by two bracketed subscript ranges of the form: **[***lo***,***hi***,***sep***]**. The first range is for x, and the second for y. The subscripts for a given dimension are *lo* through *hi* inclusive, and the separation between adjacent array elements is *sep* centimicrons.

TRANSFORM is a set of six integers that describe how coordinates in *def* are to be transformed to coordinates in the parent *name*. It is used by *ext2sim* (1) in transforming transistor locations to coordinates in the root of a design. The six integers of *TRANSFORM* (*ta, tb, tc, td, te, tf*) are interpreted as components in the following transformation matrix, by which all coordinates in *def* are post-multiplied to get coordinates in *name*:

ta	*td*	0
tb	*te*	0
tc	*tf*	1

merge *path1 path2 C a1 p1 a2 p2 ... aN pN*

Used to specify a connection between two subcells, or between a subcell and mask information of *name*. Both *path1* and *path2* are hierarchical node names. To refer to a node in cell *name* itself, its pathname is just its node name. To refer to a node in a subcell of *name*, its pathname consists of the *use-id* of the subcell (as it appeared in a **use** line above), followed by a slash (/), followed by the node name in the subcell. For example, if *name* contains subcell *sub* with use identifier *sub-id*, and *sub* contains node *n*, the full pathname of node *n* relative to *name* will be *sub-id/n*.

Connections between adjacent elements of an array are represented using a special syntax that takes advantage of the regularity of arrays. A use-id in a path may optionally be followed by a range of the form **[***lo***:***hi***]** (before the following slash). Such a use-id is interpreted as the elements *lo* through *hi* inclusive of a one-dimensional array. An element of a two-dimensional array may be subscripted with two such ranges: first the y range, then the x range.

Whenever one *path* in a **merge** line contains such a subscript range, the other must contain one of comparable size. For example,

merge sub-id[1:4,2:8]/a sub-id[2:5,1:7]/b

is acceptable because the range 1:4 is the same size as 2:5, and the range 2:8 is the same size as 1:7.

When a connection occurs between nodes in different cells, it may be that some resistance and capacitance has been recorded redundantly. For example, polysilicon in one cell may overlap polysilicon in another, so the capacitance to substrate will have been recorded twice. The values *C*, *a1*, *p1*, etc. in a **merge** line provide a way of compensating for such overlap. Each of *a1*, *p1*, etc. (usually negative) are added to the area and perimeter for material of each resistance class to give an adjusted area and perimeter for the aggregate node. The value *C* attofarads (also usually negative) is added to the sum of the capacitances (to substrate) of nodes *path1* and *path2* to give the capacitance of the aggregate node.

cap *node1 node2 C*

Defines a capacitor between the nodes *node1* and *node2*, with capacitance *C*. This construct is used to specify both internodal capacitance within a single cell and between cells.

AUTHOR

Walter Scott

SEE ALSO

ext2sim (1), magic (1)