NAME

xspectre - extract Spectre netlist out the database

SYNOPSIS

xspectre [-NTdfghikonptuvxy -w width -z name -O name
    -C file -D name -F outfile -S stimfile -X lib] cell

OPTIONS

-N Do not extract the sub-cell(s), but add a line **include** "*cell*.spectre" for each sub-cell to the output file (when option **-h** is used).

-T Tanner output, comments on same line.

-d Use original database names for instance names.

-f Send output to file *cell*.spectre instead of *stdout*.

-g Connect a large grounded resistor to each node of the two nodes that are connected by a net statement.

-h Also extract hierarchy (used local sub-cells).

-i Also extract imported sub-cells (when option **-h** is used).

-k Output all cells as sub-circuits.

-o Omit model definitions for devices in the output file.

-n Always add "nbulk" terminals for n-bulk connections.

-p Always add "pbulk" terminals for p-bulk connections.

-t Do not output unconnected instances.

-u Do not automatically add terminals for bulk connections.

-v Enable verbose mode.

-w Use *width* for the output listing. If zero, infinite width.

-x Use node number = 0 for nodes whose name start with "gnd" or "GND".

-y Use node number = 0 for nodes whose name start with "vss" or "VSS".

-z *prefix* Use node number = 0 for nodes whose name start with *prefix*.

-O *name*
Use node number = 0 for the node that has this name.

-C *file* Use the specified *file* as the control file.

-D *name*
Define a label *name* for conditional reading of the control file (always converted to uppercase).

-F *outfile*
Send output to the specified *file* instead of *stdout*.

-S *stimfile*
Append the file *stimfile* to the output file. The *stimfile* can contain input stimuli and simulation control statement (such as print and plot commands). This file is just copied, no checking or interpretation of its contents is done.

-X *lib* Exclude specific certain library cells from the listing (when options **-hi** are used). This is very useful if you have a design with your own imported libraries. This option may be given several times. There are two exclude possibilities: (1) all lib's starting with absolute path (for example "-X /usr/ocean"); (2) the lib with basename (for example "-X primitives").

DESCRIPTION

Xspectre is a program to extract a Spectre netlist (circuit description) out of the database. Default, only a netlist of the cell itself is extracted and no sub-cells are extracted. When using the **-h** option the program will also extract all (local) sub-cells that are not a function (see "SLS: Switch-Level Simulator User's Manual") or a device.

Whether or not a cell is a device is determined by (1) the contents of the control file, (2) the use of the program *xcontrol(1ICD) for new projects* or *device(1ICD) for old projects* and/or *putdevmod(1ICD)* (see the manual pages of these programs), or by (3) a set of built-in device names: "res" for resistors, "cap" for capacitors, "d" for diodes, "nenh" for n-enhancement transistors, "penh" for p-enhancement transistors, "ndep" for n-depletion transistors, "npn" and "pnp" for bipolar transistors.

When extracting a Spectre circuit descriptions, the program *xspectre* will automatically determine if "nbulk" and/or "pbulk" terminals should be added. This is determined from the bulk voltages that are required for the different devices that are part of the circuit. Under normal circumstances, this means that for circuits containing n-enhancement and depletion transistors n-bulk connections are added, and that for circuits containing p-enhancement transistors (also) p-bulk connections are added. For each bulk terminal, the program will automatically generate an appropriate voltage source, unless the **-u** option is used.

Default, *xspectre* will automatically add model descriptions to the network for the devices that are part of the network. These model descriptions are searched at the following places. First, model descriptions are searched in the library files that are specified in the control file. And second, the program will search for model descriptions that have been added to the database using the program *putdevmod.* Note that the device status must be set by *xcontrol* for new projects. For all devices that occur in the network and that have a model description available, these model descriptions will be added to the end of the network. The automatic inclusion of model descriptions can be omitted by using the **-o** option.

THE CONTROL FILE

For *xspectre* a control file may be used. This control file specifies the models that should be used for the devices, the location of the library files that contain the descriptions of these models, and the bulk voltage for devices. The default name of the control file is "xspectrerc". First, the program tries to read this file from the current working directory. Next, it tries to open the control file in the process directory.

The format of a specification of a library file in the control file is as follows:

include_library *file_name*

where *file_name* is the name of the library file. First, the program tries to read the library file from the current working directory. Otherwise, the program tries to open the library file in the process directory.

The format of a specification of a model for a device is as follows:

model *name orig_name type_name* [(*range_specs*)]

The *orig_name* has to be equal to the extracted device name, which is equal to the name specified in the element definition file. The *type_name* has to be equal to one of the SPICE standard device-model names, such as *npn*, *pnp*, *nmos*, *pmos*, *c* (= cap), *d* (= diode), *r* (= res), or *l* (= inductor). Optional, *range_specs* can be specified for device geometries for which the model is valid (initial implemented for bipolar models, for a description, see the Space User's Manual). If the *range_specs* are left out, then the model is always valid. Only the following geometrical parameters can be specified in the *range_specs*: *w*, *l*, *v*, *ae*, *pe* and *wb*.

The format of a specification of a bulk voltage for a device is as follows:

bulk *name value*

where *name* can be the *orig_name* or the *type_name* (npn, pnp, nmos, pmos) and where *value* is the bulk voltage for the device. Up to a maximum of 2 different bulk voltages may be specified in the control file. Node name "nbulk" is used for the lowest voltage and "pbulk" for the highest voltage.

For a device, instance parameters may be specified as follows:

params *name* [*model_name*] { *param_spec1 param_spec2 ...* }

The *name* can be the *orig_name* or the *type_name* and can optional be followed by the *model_name*. A new params-statement overrides a previous specified params-statement. With the params-statement the printing order of parameter values (with or without parameter name) can be changed. Normally invisible parameters can be made visible or used. Standard visible parameters can be left out or changed. The parameter specifications *param_spec1*, *param_spec2* etc. each must have one of the following forms:

parameter=*value*
value
parameter=*$intern_par*[<operator><value>]
$intern_par[<operator><value>]

with *$intern_par* denoting the actual value of a parameter that is internally (in the database) called 'intern_par' (for example $w, $l, $v, $area and $perim refer to respectively the width and length of a MOS transistor, the value of a resistor or capacitor and the area and perimeter of a junction capacitance). If the *$intern_par* does not exist in the instance attribute-list, the parameter specification is left out! If the *$intern_par* is a standard visible parameter, it is not more printed on the standard way. If the "*$intern_par*"-forms have a leading '!' sign, they are not printed. This is the way to skip a standard visible parameter. If the "*$intern_par*"-forms have two leading '!' signs, they are printed in the comment-part. The "*$intern_par*"-forms can optional be followed by an <operator> and a <value>. This <value> may also be another internal parameter. The operation is only done, if this internal parameter exists and is not zero. This <operator> can be a '+', '-', '*' and '/'. At last, you can additional use the '@' <operator> with a <string>. Denoting that the <string> must be printed after the value.

Other program build-in internal parameters are:

mname	the used model name
mdl	the model 'l' subtract value (default 0)
mdw	the model 'w' subtract value (default 0)
msf	the scale factor for scalable models (default 1)

As an example, the parameters of a junction diode **ndif** that was extracted using *space(1ICD),* and that will be simulated with *Spectre*, may be specified as follows:

params **ndif** { area=$area pj=$perim }

Further, if the control file contains the keyword:

as_subckt

all cells will be listed as sub-circuit (see option **-k**).

The keyword *name_ground* may be used to define the ground node name (is equivalent to using option **-O**):

name_ground *string*

The following specification is useful with the **-i** option:

exclude_project *path_name*

to exclude specific certain imported projects from the net-listing (see the **-X** option).

The keyword *rename* may be used to change the model simulation type name and to add some (optional) string:

```
rename  modeltype  newname  [add_string]
```

For other keywords, see also the description in the *xspice* manual page.

CONTROL FILE EXAMPLE

The following example demonstrates use of above keywords:

```
include_library spectre.lib

model dp pdiode d
model dw wdiode d
model dn ndiode d

params d { area=$area*1e12 pj=$perim*1e6 }

rename mos0 mos2
rename mos7 bsim3v3
```

TANNER OUTPUT

When using *xspectre* you can force instance line comments on the same line with the **-T** option. Such inline comments start with a '//' sign surrounded by space. An inline comment continuation is done with a '\' sign. With inline comments it is maybe useful to set very long line width. With option **-w 0** you can set infinite line width (no wrapping).
When using the **-T** option twice, you get node(net) coordinates in the inline comment for back-annotation.

THE LIBRARY FILE

The library file contains one or more simulation models according to the following format:

```
model name type_name ( par_list )
```

where *par_list* contains the model parameters. For a more extended description and examples, see the Space User's Manual and the Space Tutorial. The *type_name* may be different to a SPICE *type_name* used in the control file. For example: *capacitor* (for *c*), *diode* (for *d*), or *resistor* (for *r*).
For *nmos* and *pmos* transistors specify for example:

```
model nenh bsim4 ( type=n par_list )
model penh bsim4 ( type=p par_list )
```

Note that, when *nmos* or *pmos* is used and the first entry in the *par_list* is "level=N", then *nmos* or *pmos* is replaced by "mosN type=n|p" and level is removed from the *par_list*.
Note that the *type_name npn* or *pnp* automatically is replaced by "bjt type=npn|pnp".

THE SPICEMOD FILE

When option **-o** is not used, the *xspectre* program tries to read the *spicemod* file from the process directory. The contained SPICE models are converted and added to the extracted netlist. This file can also contain some control statements that should be used for the extracted devices. For more detail see the spicemod(4ICD) manual page. Note that a device control statement only works and a model is only added, when there is no "library model" or "devmod".

EXAMPLE

```
% xspectre -fhx invert
```

AUTHOR

S. de Graaf

FILES

cell`.spectre`	(output file, when option **-f** is used)
`xspectrerc`	(default control file)
`spicemod`	(models file for xspice/xspectre/...)

SEE ALSO

putdevmod(1ICD), space(1ICD), spicemod(4ICD), xspice(1ICD).
Spectre is a trademark of Cadence Design Systems, Inc.