1

Defects Described in Verilog-AMS

Module/model level defects

Since the most common device model description form is Verilog-A (an analog-only subset of Verilog-AMS), the easiest place to drop a defect description is in the device model itself. Put simply: the defect is a subcircuit of the module, with the option of breaking a port connections:

```
module DeviceName #(properties...) (ports…);
 analog begin
  // device model
  ….
 end
 defect DefectName1;
  break PortName; // => Open defect, “/PortName” to connect to outside
  StandardDefect #(properties...) (nodes…);
 enddefect;
 defect DefectName2;
  defparam Param = OOBvalue; // parametric defect
 ...
endmodule;
```

Multiple defects can be described and any active defect in simulation behaves the same as a submodule – the instance path being “...<device instance>.<defect name>...”

If a simulator doesn't understand the defect syntax the defects can be surrounded by a `ifdef...`endif to hide them.

The module reference “StandardDefect” would be replaced by something like “SimpleShort” from the standard defect library described below.

For annotating devices from outside the module the syntax is the same but “module” is preceded by “annotate”, e.g.:

```
annotate
module DeviceName #(properties...) (ports…);
 // no model constructs allowed
 defect DefectName1;
  break PortName; // => Open defect, “/PortName” to connect to outside
```

```
    StandardDefect #(properties...) (nodes…);
  enddefect;
  ...
endmodule;
```

"annotate" statements refer to the last module of that name if there is ambiguity, or the next if there is no immediate match (Verilog input is usually read in strict sequence). The module name in an annotate statement may be "(<regular expression>)" so that one set of defects may be used for multiple devices – e.g. "annotate module (*) ..." for all modules.

The base class of a defect (open or short) is implied by whether there are "breaks" in it, otherwise its name and/or submodules called (from a P2427 standard library) imply the defect type. Note: "break" is a Verilog keyword, and the "defparam" mechanism already exists, so scripting the generation of an individual defect model from the general device model is straightforward – rename the internal node for any breaks, and instantiate the given defect block(s). Verilog has a general attribute scheme - (* <attribute> *) - for any extra information needed.

While intended to be used in the device models, this mechanism can be used in any module, so wiring defects due to circuit construction can be add in higher level modules when delivering blocks of IP, and above. Likewise the "annotate" version can be used with SPICE subcircuits.

The "/" operator is not unary in existing Verilog, and the new use would be limited to defect blocks.

SystemVerilog and VHDL support configurations, where you can provide alternative implementations of a given instance. For (re)simulating a defect you just run the above module/defect definition through a simple script script to create the defective version to use. Noting that the syntax is mostly structural and analog behavior is in the defect library not necessarily the device/module definition (it works equally well for digital defects)

P2427 Standard Library

Most defects can be described with passive components that can be described as SPICE subcircuits. Verilog-A was designed to be SPICE compatible (Appendix E in the LRM), so using the Verilog-A device/defect description doesn't preclude the standard defect models being in other languages – module instance binding is by name and port order for most HDLs.

SystemVerilog supports "packages", using that syntax the standard defects can be provided in a file with that style and also "ifdef'd" for inclusion in standard Verilog-AMS -

```
package P2427_Defects;
```

```
1
2    module SimpleShort (inout a, inout b);
3      electrical a,b;
4      parameter real R = 1.0 from (0.0:inf];
5      analog I(a,b) <+ V(a,b)/R;
6    endmodule
7    module OpenResistance (inout a, inout b);
8      electrical a,b;
9      parameter real R = 1.0e9 from (0.0:inf];
10     analog I(a,b) <+ V(a,b)/R;
11   endmodule
12   ….
13 endpackage
```

Defect reporting

When reporting defects used in a test run, the same format can be used but with “annotate <instance path>: <module name>” instead of just “annotate <module name>”.

As above, the defect description may just be breaks and references to standard defect models, and the standard model descriptions would not need to be included in every report. Only the defect(s) chosen for simulation out of the complete set are reported, and specific parameter values if the source defect description was specified with a range.

References

https://www.hdlworks.com/hdl_corner/verilog_ref/items/DefParam.htm

https://verilog.renerta.com/mobile/source/vrg00027.htm

https://www.accellera.org/images/downloads/standards/v-ams/VAMS-LRM-2-4.pdf

http://www.ece.uah.edu/~gaede/cpe526/SystemVerilog_3.1a.pdf

https://www.hdlworks.com/hdl_corner/verilog_ref/items/Configuration.htm