Changing the Paradigm for Compact Model Integration in Circuit Simulators Using Verilog-A

M. Mierzwinski, P. O’Halloran, B. Troyanovsky, R. Dutton*

Tiburon Design Automation, Inc.
*Stanford University
Outline

• Current state of compact model development
• Verilog-A for compact modeling
• Examples of compact models in Verilog-A
• The paradigm shift
• Summary and conclusions
Typical model flow

Model Extracted or Translated

Foundry

Design Verified

Model and Simulation Flow Verified

Repeat...

Independent Implementations

Vendor A

Vendor B

Vendor C

Vendor D
Model Developer issues

- Complexity in model development has shifted from model creation to model implementation
- Fewer models implemented
- Models become more general to try to handle all processes
Model Developer issues

- Simulator model interfaces that are powerful are complex.
- Model interfaces promote errors due to complex math and program requirements.
- Software engineering aspects of implementation are time consuming.
Model Developer Issues

• Existing source-code interfaces are inherently non-portable

• High burden on the developer to:
  – Manually compute matrix stamp
  – Manually compute derivatives
  – Implement multiple, self-consistent entry points for analysis types

• Requires close cooperation with the simulator vendor
End User Issues

- Want accurate, robust, tried and tested models.
- Want to use models confidently across different simulation tools.
- Want all models to simulate efficiently.
- Want a wide range of models to be available and easily obtainable.
- Want to easily modify the underlying equations to suite their particular need.
Vendor Issues

• Standard models, such as BSIM3, are a necessary part of any analog simulator
  – Supporting standard models offers no competitive advantage.
• Multiple implementations of same standard model increases risk of errors
  – Always requires verification and testing.
• Potential model differences prevent new customers from evaluating their product.
Foundry Issues

• Standard models often do not provide accurate enough representation of the foundry process
• Foundries do not have sufficient resources to support and verify all vendors’ tools.
  – This can limit their customer base.
Verilog-A

- Natural language for model development
- Concise
- OVI standard (proposed to IEEE)
- Implemented in many simulators
Existing solutions

Current Verilog-A solutions
- Are not sufficiently fast for simulation with typical compact models
- Are not universally available
- May not offer IP protection
- Don’t support all analysis types
- Do not provide a simple distribution mechanism
Proposed Solution

• Use Verilog-A for analog model definition
• Develop a Verilog-A model compiler and support tools to create analog models that can be used in a wide range of simulation platforms.
Proposed Solution

- Maintain simulation performance comparable to existing C/C++ level interfaces.
- Provide robustness better than existing implementations.
- Support for all analysis types, e.g. transient, harmonic balance, shooting, nonlinear noise.
Proposed Architecture

- A Verilog-A compiler and simulator-specific run-time environment (RTE)

```
Verilog-A Source
    ↓
  Compiler
    ↓
   CML*

---

RTE Simulator 1
RTE Simulator 2
RTE Simulator 3
RTE Simulator 4

* CML = compiled model library
Benefits to End Users

• Solves the problem of model availability and compatibility.
• Models with source code available can be modified.
Benefits to Vendors

• Removes need to support complicated standard models.

• Proprietary models can also be developed Verilog-A and distributed in compiled form
  – Provides IP protection
Benefits to Foundries

• Model definition can be part of foundry kit
  – Can be completely independent of simulators and simulator versions.
• One parameter set to extract, distribute, and support.
• Models can be modified to fit the foundry process
  – Parameter information and model changes can be hidden to protect IP.
Benefits to Model Developers

• Concentrate on model development, not implementation issues.
• Models and updates can be made available immediately on all platforms.
• Model IP can be protected and licensed.
Demonstration of Verilog-A based Compact Models

• Support for popular models is necessary for language acceptance
• Provides templates for generation of similar models in Verilog-A
• Demonstration of language capability
• RTE implemented in two simulators: – SPICE 3F5, HBsim.
Examples: Gummel-Poon BJT

- SPICE Gummel-Poon
- Very old model
  - Yet very popular
- Users often like to modify model slightly
  - Kull-Nagel
  - Thermal
  - Avalanche
- ~300 lines in Verilog-A
Examples: MEXTRAM BJT

- Philips MEXTRAM 504 BJT in Verilog-A
- Colpitts oscillator
Examples: Compact models MOS

- UC Berkeley BSIM3 MOSFET
  - Only CMC “standard” model
  - Typically tens of thousands of lines of C-code
  - Example of simple op-amp using Verilog-A
  - Same results as SPICE built-in

Gain vs. Freq for Verilog-A and SPICE BSIM3 models in OpAmp
Examples Compact models MOS

- UC Berkeley BSIM4 MOSFET model
  - Next generation BSIM
  - Many more parameters
  - First released March 2000
  - April 2002 CMC still working on implementation problems between vendors.

DC-IV for both SPICE and Verilog-A versions of BSIM4
Examples Compact models MOS

- Philips MOS Model 9
  - BSIM3 counterpart
- Philips MOS Model 11
  - Next generation MOS model
Example of single model library used in multiple simulators

• HISIM
  – Hiroshima University STARC IGFET Model
• Popular drift-diffusion/surface potential based model
  – Smaller parameter set size
  – Physically based
• Uses local Newton-Raphson loop
HiSIM Implementation

HiSIM
Verilog-A
Source

Compiler

hisim.cml

RTE

SPICE 3F5

RTE

HBsim 1a4

One compiled model
library file shared by both
simulators
New design capability

- Circuit designers can control level of abstraction
- All levels can be combined in one hierarchical implementation
Behavioral model example: PLL

Complete PLL behavioral definition in one library
Performance

- Simulation speed of Verilog-A based compact models is critical for acceptance
- Myth of hand-coded equations

<table>
<thead>
<tr>
<th>Model</th>
<th>Simulation</th>
<th>Total simulation time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Verilog-A / SPICE</td>
<td></td>
</tr>
<tr>
<td>Resistor</td>
<td>DC</td>
<td>1.3</td>
</tr>
<tr>
<td></td>
<td>Transient</td>
<td>1.2</td>
</tr>
<tr>
<td>Diode</td>
<td>DC</td>
<td>1.3</td>
</tr>
<tr>
<td></td>
<td>Transient</td>
<td>1.5</td>
</tr>
<tr>
<td>BSIM3</td>
<td>DC</td>
<td>1.2</td>
</tr>
<tr>
<td></td>
<td>Transient</td>
<td>1.5</td>
</tr>
<tr>
<td>BJT adder</td>
<td>Transient</td>
<td>1.6</td>
</tr>
</tbody>
</table>
Outstanding issues

• Verilog-A is quite capable of describing compact models
• Addition improvements
  – Parameter attributes
  – Additional program functionality
    • Function parameters
    • Local declarations
New Modeling Ecosystem

Develop once, run everywhere

Model validated through design success

Single model extraction

Vendor independent Implementation

Verilog-A

Vendor A

Vendor B

Vendor D

Vendor C

Designer

Simulation flow verification only
New Paradigm

- Model interface is standardized on Verilog-A
- Model definition no longer needs to be static
  - Fit model to process
  - Remove unneeded functionality of standard models
- Models can be a competitive advantage
- Models can have revenue potential
- Users can choose simulators based on analysis algorithms rather than model set.
Conclusions

- Verilog-A has the capability to support complex compact model implementations
- A Verilog-A compiler provides fast execution and support for all analysis types
- Virtually all popular compact models have been implemented as a demonstration.
- Model implementation and distribution can be greatly simplified.
  - Model developers, simulation vendors, end users all benefit.