Automatic Model Compilation

An Idea Whose Time Has Come

Ken Kundert
Designer's Guide Consulting, Inc.

Version 1, 18 May 2002

Development of compilers that install compact device models described in Verilog-A into commercial circuit simulators are expected to dramatically improve the number and quality of device models available to designers. A open-source approach is proposed for developing such a compiler along with an associated set of models.

Last updated on May 12, 2006. You can find the most recent version at www.designers-guide.org. Contact the author via e-mail at ken@designers-guide.com.

1.0 Device Modeling Today

Currently an implementation of a device model like BSIM3v3 requires between 10-30K lines of C code, much of which is custom to each simulator in which it is installed. Once a model is released by its author, it may take several months of work to install and test it in a simulator, many more months may pass before the simulator passes through its release process and becomes available to customers, and then it can take even longer for the simulator to be qualified by the purchasing companies and made available to the designers.

Before the model is integrated, it must be written. With current practices, this is a slow and error prone process. The initial design of the model is done with Matlab. Typically, at this point a model requires on the order of a few hundred lines of code. Once the equations are set, the process of converting the model to C begins. This involves computing and coding all the derivatives, performing careful optimizations, and interfacing the model to the simulator. The fact that the models are hand coded implies that the optimization involves a difficult trade-off between efficiency and the time required to implement and support the model. Also, the conversion from Matlab to C is a potential source of many errors.

The combination of these factors imply that models are updated infrequently and so designers often do not get access to compiled models that include important effects until years after the effects have been identified. Further, the high cost of installing and maintaining models implies that only a few models are provided with commercial simulators. Innovative new models for MOS and BJT devices often do not have much impact because they never become available in the commercial simulators. Models for less common components generally never get compiled into commercial simulators. This implies that important devices, such as SiGe HBTs, varactors, PIN diodes, photo and laser diodes, MEMS devices, etc. are either not modeled, or are less accurately and less efficiently macro modeled.

The end result of the difficulty of implementing and supporting models is that few models are available in commercial simulators, which means that designers do not have the models they need, and yet model developers are discouraged from providing them because anything they do is unlikely to get used. Dramatically reducing the cost and time required to develop, install, and support models in simulators should substantially improve the variety, utility, and quality of models while reducing the time it takes to make them available to designers. Further, if these models can be installed without requiring the cooperation of the simulation vendors, then a significant barrier to the availability of a broad suite of models will be eliminated.

2.0 Model Compiler

At the Custom Integrated Circuits Conference this year, Lemaitre, McAndrew and Hamm, all from Motorola, presented ADMS, a model compiler that they are proposing to make publicly available through an open-source process [1, 2]. ADMS takes as input a Verilog-A description of a model and produces C code that compiles directly into a target simulator. Verilog-A is the industry standard language analog behavioral language[1] that is well suited to device modeling[2]. Currently, Verilog-A provides many of the benefits of Matlab, and provides substantial additional advantages. Current imple-

mentations of Verilog-A are interpreted, meaning that, as with Matlab, the time required to try ideas is very short. In addition, since Verilog-A is implemented in a circuit simulator, it is easy to test the models in real circuits, such as ring oscillators. In this way, model developers can more easily identify and focus on the effects that have an impact on target circuits. This is not easily accomplished in Matlab.

ADMS is structured with a front-end that compiles Verilog-A into an intermediate representation and a back-end that converts that to the actual C code that is compiled into a target simulator. Each simulator has its own back end. Currently, ADMS provides back-ends for Spectre, ADS, McSPICE[3], and NanoSim. The companies that own these simulators all consider their compiled-model interface to be proprietary, which will inhibit Motorola from effectively putting ADMS in the public domain without the consent and support of the simulation vendors. However, it is important to recognize that it not necessary for the back-ends to be in the public domain. As long as the intermediate representation is open and well specified, the back-ends can be closed and owned by the individual simulation vendors. The innovation that the open-source aspect brings would then be concentrated in the front-end compiler in the form of optimizations and in the models.

3.0 Benefits of the Model Compiler

The ADMS model compiler has the potential to address all of the issues presented in Section 1.0. When fully realized, it provides users the ability to quickly install new or enhanced models without needing to enlist the cooperation of the simulator vendor. Since the same model can be compiled into multiple simulators, compatibility between simulators is under user control. Design companies have the option of funding model development with the assurance that the models can be installed in all the simulators they use. Research groups can create models knowing that they only have to identify their merits to design groups, which will see the models as a benefit, and not simulation vendors, which see the models as a cost. As new effects become important, existing models can be supplemented without waiting for the original model authors and the simulator vendors to take action.

1. Verilog-AMS (a super set of Verilog and Verilog-A) and VHDL-AMS (a super set of VHDL) are also industry standard languages, but they are mixed-signal languages and so provide features that are not needed in this application and that serve to restrict the set of simulators that support them to single engine mixed-signal simulators, of which there are currently only a small number available. Verilog-A, being an analog only language, can also be supported in any SPICE simulator, is more widely available than either Verilog-AMS or VHDL-AMS, and is available now, or expected to be available soon, in all simulators that support, or are expected to support, either AMS language.
2. Some small extensions to Verilog-A will be needed to make the models easier to compile and more self documenting. The effort to develop these extensions is being led by Geoffrey Coram of Analog Devices, and it is expected to result in a new version of the Verilog-AMS language (vers. 2.2) in September 2004.
3. McSPICE is Motorola's internal simulator.

It is expected that wide availability of a Verilog-A model compiler will create an environment that is conducive to developing open-source models. Already models such as VBIC and MOS9 are available in source code form on the web [3]. When anyone can download such models, enhance them, and easily use their enhancements, there will be a natural tendency to want to share those enhancements as a way of maximizing their utility and minimizing support costs. The end result will be a broader variety of models that are more accurate and robust, and a more vibrant and healthy modeling community.

4.0 Realizing the Model Compiler

Successfully establishing a healthy open-source modeling community will involve creating an organization to lead the open source effort. Its would establish the infrastructure and ground-rules for sharing and adopting code, first for the model compiler, and then perhaps for the models themselves. More specifically, this would define and communicate the overall goals and direction of the effort, develop a plan for accomplishing the stated goals, set up and fund the machinery for sharing code, choose a license agreement, recruit individuals and companies that will contribute expertise, code, time and energy, proposed enhancements to Verilog-A, create a test suite, own and manage the rights to any jointly developed intellectual property, encourage support for the compiler in commercial simulators, and promote development of compact device models in Verilog-A. The organization can either be completely independent, or it can be associated with an existing similar organization.

Funding will be necessary to cover the expense of maintaining the code-sharing infrastructure and to fund research and development of new compiler technology and models. Both are critically important. The success of the model compiler will not be assured until it supports all of the popular MOS and BJT models with a run-time efficiency nearly as good as hand coded models. Funding may come either from companies that are members of the organization, from industrial consortia such as the SRC, or from governmental sources, such as the NSF or Darpa.

5.0 An Idea Whose Time Has Come

For many years Colin McAndrew of Motorola has been carrying on a one-man crusade to get automatic model compilation supported by mainstream simulators and models. Despite his first hand accounts of the tremendous productivity gains achieved by both the model and the simulator developers, few were willing to make the needed investment.

Widespread adoption of a standard modeling language combined with the ability to compile into mainstream commercial simulators has the potential to revolutionize circuit simulation for everyone that either writes or uses circuit simulators and their models. The time has come for both the modeling and the simulation community to start investing in and adopting this critically important technology. Doing so will revitalize the simulation and modeling and will dramatically improve the utility and effectiveness of simulation for designers.

Colin was right.

6.0 References

[1] Laurant Lemaitre, Colin McAndrew, Steve Hamm. ADMS — Automatic Device Model Synthesizer. *IEEE Custom Integrated Circuits Conference*, May 2002.

[2] ADMS Project Home Page. *mot-adms.sourceforge.net*.

[3] Verilog-A/MS semiconductor models. *www.verilog-ams.com*.