Latest Developments in the Xyce Large-Scale Analog Circuit Simulator

Jason C. Verley
Sandia National Laboratories
Albuquerque, New Mexico
jcverle@sandia.gov

Eric R. Keiter
Sandia National Laboratories
Albuquerque, New Mexico
erkeite@sandia.gov

Heidi K. Thornquist
Sandia National Laboratories
Albuquerque, New Mexico
hkthorn@sandia.gov

ABSTRACT

This paper provides an overview of the open source analog simulation tool, Xyce, which was designed from the ground-up to perform large-scale circuit analysis. Current capabilities of the simulation tool will be discussed, including the analysis methods, device models, and parallel implementation. Highlights of progress from the previous WOSET conference will also be presented, in addition to future directions for expanding Xyce's capabilities and improving performance.

CCS CONCEPTS

• **Computing methodologies → Massively parallel and high-performance simulations**; *Parallel algorithms*; *Distributed simulation*; • **Applied computing → Computer-aided design**; *Physical sciences and engineering*.

ACM Reference Format:
Jason C. Verley, Eric R. Keiter, and Heidi K. Thornquist. 2019. Latest Developments in the Xyce Large-Scale Analog Circuit Simulator. In *Proceedings of Workshop on Open-Source EDA Technology (WOSET'19).* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Analog circuit simulation is an established workhorse in the area of Electronic Design Automation (EDA). In fact, it could be argued that the original Berkeley SPICE 2 [5] analog circuit simulator, released in 1975, was the first widely successful EDA tool. SPICE, in its various commercial flavors, has become so ubiquitous that analog circuit simulations are called "spice" simulations, independent of the tool being used.

In the years since its release, SPICE and its successors continue to be an integral part of all analog and mixed-signal EDA flows. Even in digital flows, spice tools contribute to a range of tasks, from standard cell characterization at the front end of the flow, to design verification at the back-end. The tight requirements of current process nodes further pushes the ability of analog circuit simulators for efficiency and the ability to simulate increasingly larger systems.

Analog circuit simulators are based on solving a fully-coupled nonlinear differential algebraic equation (DAE)

$$f\left(x\left(t\right)\right) + \frac{dq\left(x\left(t\right)\right)}{dt} = b\left(t\right). \tag{1}$$

Traditional solution approaches rely on a large single matrix that is usually treated by direct matrix solvers. These methods do not scale well beyond tens of thousands of unknowns As a result, the analog runtime scales super-linearly with increasing circuit size. For RF simulation, the scalability can be even worse than for transient, because harmonic balance (HB) analysis generates larger matrices that lack sparsity for the nonlinear portions of the problem.

SPICE-accurate simulation is often a prohibitive bottleneck in circuit design and verification. The speed penalty is can be mitigated by simulating only individual modules. Alternatively, it is natural to consider techniques that accelerate the analog simulation by using numerical approximations at various levels throughout the simulator. "Fast-SPICE" simulators rely on circuit-level, hierarchical partitioning algorithms, event-driven simulation techniques, and efficient surrogate models for devices and/or sub-circuits to perform faster, large-scale circuit simulations. While effective in many cases, the numerical approximations inherent to such algorithms can break down for modern feature sizes, especially in post-layout simulations where parasitics play a large role.

2 THE XYCE PARALLEL SIMULATOR

Xyce is a SPICE-compatible analog simulation tool [3], which has been under development since 1999. It has been designed to use distributed memory parallelism to address the scalability issues inherent to solving large nonlinear DAEs (1). From the beginning, the focus of Xyce development has been to provide scalable, numerically accurate analog simulation for large-scale circuits through the development and improvement of the algorithms at the core of SPICE-style simulation. Furthermore, Xyce has been designed to have a modular framework for integrating device models and developing state-of-the-art continuation algorithms, analysis methods, preconditioned linear solvers, and parallel partitioning techniques.

While designed to be SPICE-compatible, Xyce is not a derivative of SPICE. It was designed from scratch to be a parallel simulation code, written in C++ and using a message-passing implementation (MPI). Xyce leverages Sandia's open-source solver library, Trilinos, which includes a number of circuit-specific solvers, such as the KLU direct solver. In this section, we will take a look at several essential components of Xyce and discuss how they support the simulation of large-scale circuits for modern process nodes.

WOSET'19, November 2019, Westminster, Colorado USA
© 2019 Copyright held by the owner/author(s).
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2.1 Analysis Capabilities

Xyce supports the standard analysis capabilities found in other analog circuit simulation codes, such as DC, transient, AC, and small-signal noise analyses. Other, less common capabilities include: single-tone and multi-tone harmonic balance (HB), multi-time PDE (MPDE), and model order reduction (MOR). More recent additions include sensitivity analysis (both direct and adjoint)—which computes sensitivities for a user-specified objective function with respect to a user-specified list of circuit parameters—and sampling analysis, which executes the primary analysis (.DC, .AC, .TRAN, etc.) inside a loop over randomly distributed parameters. The newer analysis capabilities are also active areas of research, so the capabilities are improving with each release.

In addition to the analysis methods, Xyce offers post-processing computation capabilities. Fourier analysis of transient analysis output is available in the .FOUR and .MEASURE commands. The .MEASURE capability has many other output variable computations, including maximum/minimum values, frequency, duty cycle, and derivatives.

2.2 Device Model Support

Xyce includes legacy SPICE models, industry standard models (BSIM, PSP, MEXTRAM, VBIC, e.g.), and non-traditional models, such as memristors. The extensible design of the device model package, which defines the interaction between the analysis methods and the device models, has also enabled Xyce to be used for biological/reaction/neural network simulation and power grid simulation. For the integration of new compact models, Xyce has a backend to the ADMS compiler, which translates Verilog-A to compilable Xyce C++ code. The C++ code can be compiled directly into Xyce's device model package, or dynamically linked to an existing Xyce binary. Alternatively, the Model and Algorithm Prototyping Platform (MAPP) [9], developed at Berkeley, provides translation from devices written in ModSpec [1], or the compilation of Verilog-A code via VAPP (the Verilog-A Parser and Processor). The resulting C++ device model can be passed into Xyce, using dynamic loading via an API.

2.3 Scalable Simulation Using MPI

To our knowledge, Xyce is unique as an analog circuit simulator in its use of MPI for parallel processing. Other simulators use shared-memory parallelism (threading or GPU, e.g.). Shared-memory parallel methods inherently restrict a simulation to a single machine, thus limiting the size of the problem a code can simulate. In contrast, distributed-memory parallel methods, such as message passing, can scale to much larger problems, since they are limited only by the performance of compute clusters.

Achieving scalable parallel circuit simulation often comes down to striking the right balance of device distribution and the right choice of a linear solver. Since the inception of Xyce, the simulator has been designed to use a separate partition for devices and the linear solver. That is because the cost of evaluating a device model can vary greatly between device types, and balancing that cost across processors can result in a matrix that is more challenging for linear solvers. Furthermore, as the scale of circuits increase—and assuming a reasonable distribution of devices—the dominant cost in the analysis quickly becomes the linear solver. To address this performance bottleneck, new parallel linear solvers and preconditioners [2, 6, 7] have been developed that enable the scalable transient simulation of postlayout ASICs with millions of devices.

An efficient, scalable linear solver is even more crucial for frequency-domain simulation than it is for transient simulation. Due to the fact that most device models can only be evaluated in the time domain, it is necessary to use Fourier expansions on the input and output variables of (1). After substitution and truncation (to M harmonics), the frequency-domain system is

$$H_{HB} = \Omega Q(X) + F(X) - B = 0, \tag{2}$$

where

$$\Omega = \begin{bmatrix} -Mj\omega_0 & & \\ & \ddots & \\ & & Mj\omega_0 \end{bmatrix}, \omega_0 = \frac{2\pi}{T}.$$

Compared to the linear system, of dimension n, generated by transient analysis, HB analysis will generate a linear system of dimension $n(2M+1)$ that is block structured, complex-valued, and possibly dense. Forming the matrix for HB analysis is not desirable, due to its size and structure, so iterative linear solvers are more appealing. Xyce provides efficient preconditioners and even a block direct linear solver that leverages sparsity from linear devices [4].

3 RECENT DEVELOPMENTS IN XYCE

Xyce has had several improvements since the 2018 Workshop on Open-Source EDA Technology [8]. In addition to significant performance improvements in several areas of the code, the build system was converted to CMake, new capabilities were added for frequency-domain simulations, output capabilities were added, and there were improvements with HSPICE compatibility.

Of importance for the simulation of modern process nodes is the support of foundry Process Development Kits (PDKs). The compact model libraries supplied with PDKs are typically in the HSPICE or Spectre netlist languages. Therefore, a focus of recent Xyce development work has been on HSPICE compatibility. In addition to syntax, several features were added to Xyce, including expression support in the capacitor model, and the general handling of expressions. Outside of the Xyce code, itself, a netlist translator has been packaged with Xyce as part of the 7.0 release. Called XDM, the translator allows a user to translate an HSPICE-based PDK, or an HSPICE netlist, into Xyce syntax. While some hand-processing of the netlists is still sometimes required, XDM marks the non-translatable parts (often simulation specifications) to reduce the amount of time the user spends in the translation process.

Enhancements to Xyce have also resulted in speed improvements and reduced memory usage. In particular, the ADMS backend was rewritten to provide analytic derivatives in the resulting C++ code. Previously, derivatives were calculated on-the-fly using Sacado (provided as part of the Trilinos library). This change has resulted in an approximately 8× speedup in the evaluation of Verilog-A–based devices. Various test circuits ran $\sim 5\times$ faster using the new code.

In addition to the device evaluations, two other performance improvements were done. The first focused on memory usage, which eliminated obsolete storage of various structures, and reduced the use of strings in the code. This resulted in a 20% reduction of memory use in several circuits of interest. For the second, the linear solver was improved with the removal of a matrix copy operation. This resulted in a $\sim 2\times$ speedup for certain memory circuits.

Another improvement, available soon, is a rewrite of the CMake build system. This rewrite allows for a more robust build process, especially against Trilinos and other third-party libraries. While the CMake system has proven robust on Sandia systems, the Autotools build system will remain until the CMake system has been proven robust on a larger number of platforms.

Finally, many other capabilities have been added to Xyce. Of note is the ability to perform S-parameter analyses. While still a beta capability, it will be enhanced and improved in future months. Other improvements include the ability use .MEASURE and parameter sensitives with AC analysis, enhancements to the output capabilities, and the implementation of a basic VPI interface.

3.1 Future Directions

While Xyce is a highly-capable analog circuit simulator, many opportunities for improvements exist. In addition to a continued focus on performance, future work will focus on continued enhancements for PDK compatibility. This includes support for more compact models, improvements to the XDM netlist translator, and code enhancements for parsing and feature compatibility. Other areas of development include support of coupling to other codes (including mixed-signal simulations), and enhancements to the frequency-domain capabilities, focus on HB analysis and improved S-parameter support.

4 LICENSING AND AVAILABILITY

Xyce is open source software, released under the GNU General Public License, Version 3.0, since the release of Xyce 6.0 in 2013. Version 7.0 of Xyce is scheduled for release in November 2018. More information about the Xyce project, including software downloads and documentation, can be found on the website: https://xyce.sandia.gov.

ACKNOWLEDGMENTS

We wish to acknowledge the Advanced Simulation and Computing (ASC) Project under the National Nuclear and Security Administration (NNSA), and the DARPA Posh Open Source Hardware (POSH) project for funding the development of Xyce.

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.

REFERENCES

[1] D. Amsallem and J. Roychowdhury. 2011. ModSpec: An open, flexible specification framework for multi-domain device modelling. In *2011 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 367–374.

[2] J. Booth, S. Rajamanickam, and H. Thornquist. 2016. Basker: A Threaded Sparse LU Factorization Utilizing Hierarchical Parallelism and Data Layouts. In *Proceedings of the 2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. ACM, 673–682.

[3] Eric R. Keiter, Heidi K. Thornquist, Robert J. Hoekstra, Thomas V. Russo, Richard L. Schiek, and Eric L. Rankin. 2011. Parallel Transistor-Level Circuit Simulation. In *Advanced Simulation and Verification of Electronic and Biological Systems*, Peng Li, Luís Miguel Silveira, and Peter Feldmann (Eds.). Springer, Chapter 1, 1–21.

[4] Amit Mehrotra and Abhishek Somani. 2009. A Robust and Efficient Harmonic Balance (HB) using Direct Solution of HB Jacobian. In *Proceedings of the 1009 Design Automation Conference*. ACM, 370–375.

[5] L. W. Nagel. 1975. *SPICE 2, a Computer Program to Simulate Semiconductor Circuits.* Technical Report Memorandum ERL-M250.

[6] Heidi K. Thornquist, Eric R. Keiter, Robert J. Hoekstra, David M. Day, and Erik G. Boman. 2009. A parallel preconditioning strategy for efficient transistor-level circuit simulation. In *2009 IEEE/ACM International Conference on Computer-Aided Design - Digest of Technical Papers*. 410–417.

[7] Heidi K. Thornquist and Siva Rajamanickam. 2014. A Hybrid Approach for Parallel Transistor-Level Full Chip Circuit Simulation. In *High Performance Computing for Computational Science - VECPAR 2014*, M. Dayde, O. Marques, and K. Nakajima (Eds.). Springer.

[8] Jason C. Verley, Eric R. Keiter, and Heidi K. Thornquist. 2018. Xyce: Open Source Simulation for Large-Scale Circuits. In *Articles presented in WOSET 2018*. https://woset-workshop.github.io

[9] T. Wang, K. Aadithya, B. Wu, and J. Roychowdhury. 2015. MAPP: A platform for prototyping algorithms and models quickly and easily. In *2015 IEEE MTT-S International Conference on Numerical Electromagnetic and Multiphysics Modeling and Optimization (NEMO)*. 1–3.