SAND2014-1538P

Xyce: an open source SPICE engine

Eric Keiter, Tom Russo, Heidi Thornquist, et al.
Sandia National Laboratories

Jaijeet Roychowdhury, Tianshi Wang
UC-Berkeley

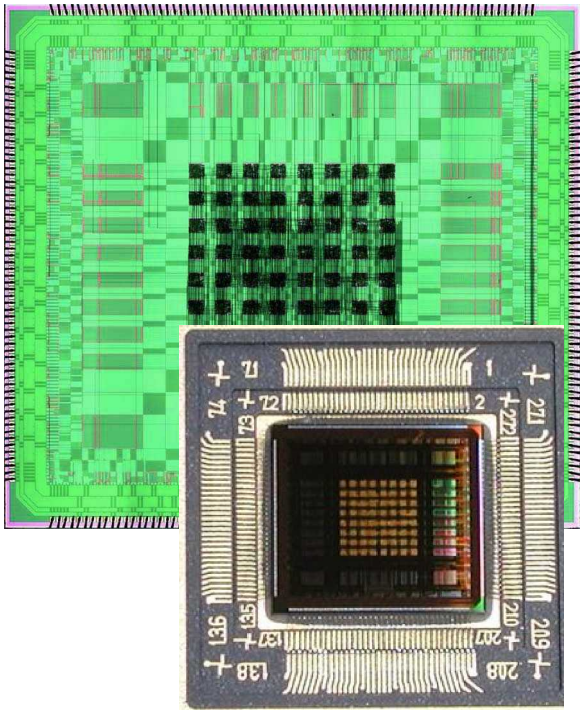
Outline

Xyce intro
Xyce requirements
Xyce design
- Object-oriented (C++ mostly)
- Spice-compatible
- But different than SPICE!
- Parallel implementation
- Analysis types (DC,Tran,AC, HB, MPDE, UQ)
- Open-Source (new, as of v6.0)

ADMS model compiler
ModSpec-Xyce
Xyce supports most SPICE models (BSIM, EKV, etc)

Xyce Parallel Circuit Simulator

- Xyce: Massively Parallel circuit simulator:
  - Distributed Memory Parallel (MPI-based)
  - Unique solver algorithms
  - SPICE-Compatible
  - Industry standard models (BSIM, PSP, EKV, VBIC, etc)
- Analysis types
  - DC, TRAN, AC
  - Harmonic Balance (HB)
  - Multi-time PDE (MPDE)
  - Model order reduction (MOR)
  - Direct and Adjoint sensitivity analysis
  - Uncertainty quantification (UQ) via Dakota.
- Sandia-specific models
  - Prompt Photocurrent
  - Prompt Neutron
  - Thermal
- Xyce Release 6.0
  - Open Source!
  - GPL v3 license

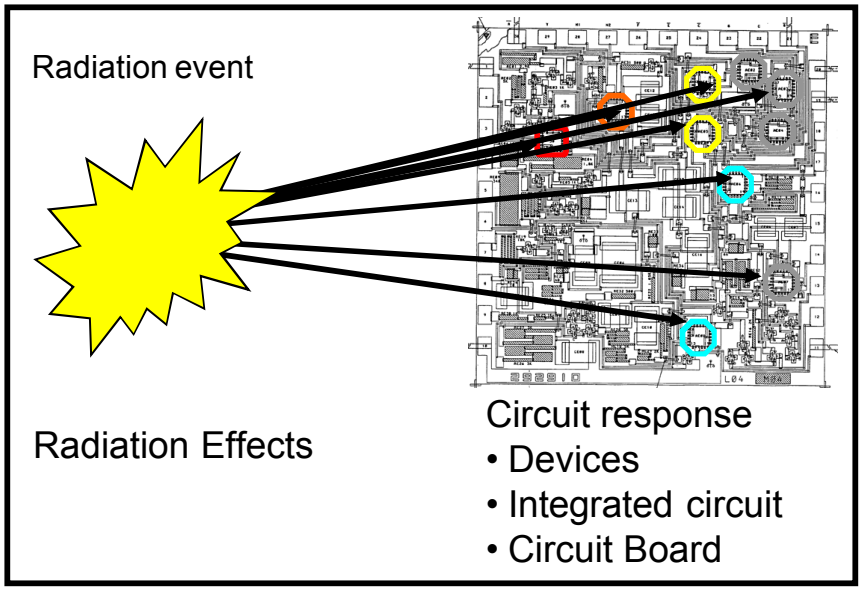


http://xyce.sandia.gov

Project Motivation: Why Xyce?

- Comprehensive Test Ban Treaty (CTBT), 1993
- Advanced Simulation & Computing (ASC), 1995
- Qualification Alternatives to SPR (QASPR), 2005
  - Offset lack of NW testing
  - Help qualify NW systems

- Unique Requirements ➡ Differentiating capabilities
  - Unique models: Radiation Effects
  - High fidelity: SPICE-level or higher
  - Large capacity: Massively-parallel
  - IP: Sandia owns it

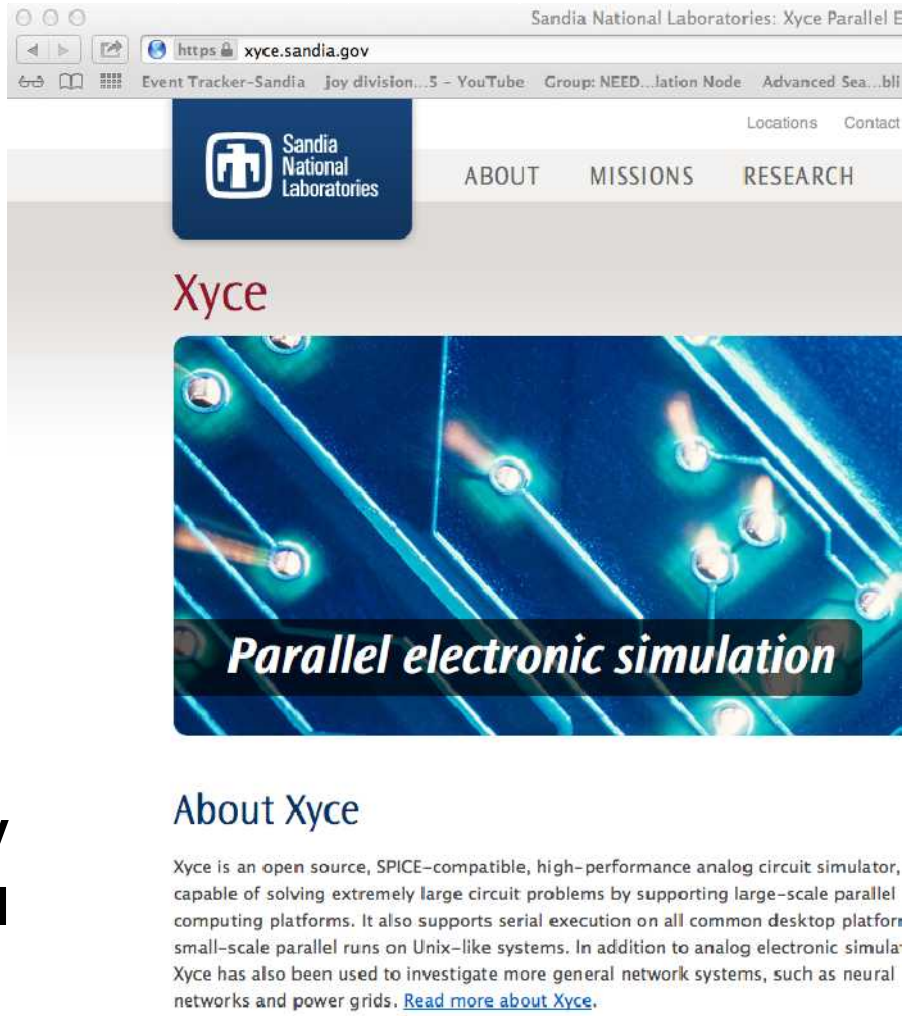


Why Open Source?

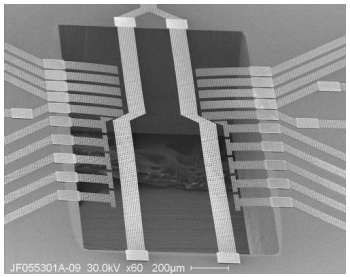
- First open source release, v6.0
- November 5, 2013.
- GPL license v3.0
- Source and binary downloads available
- xyce.sandia.gov

- Next release (v6.1) ~April 2014.

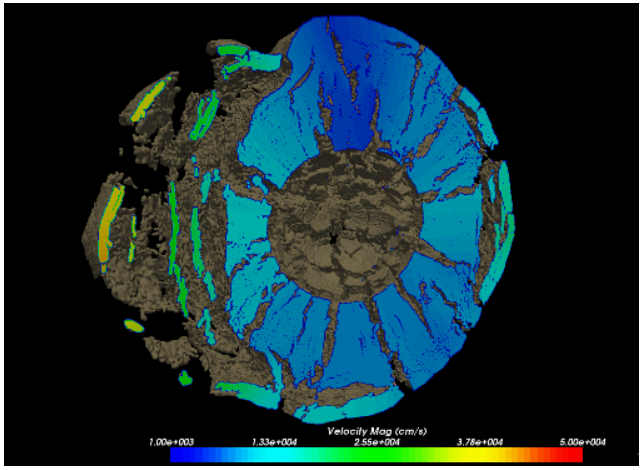
- Foster external collaboration
- Feedback from wider community
- Taxpayer funded, so encouraged to open source.



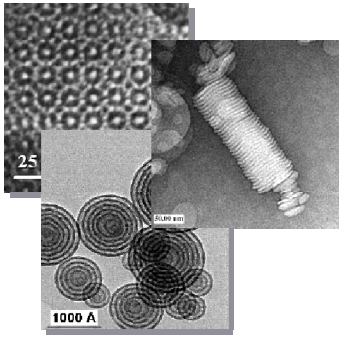
SNL has six core technical capabilities



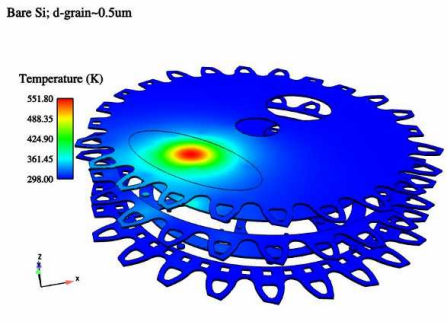
Microelectronics and Photonics



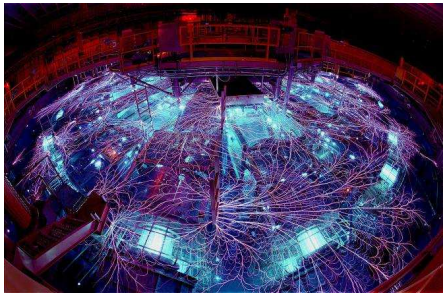
Computational & Informational Sciences



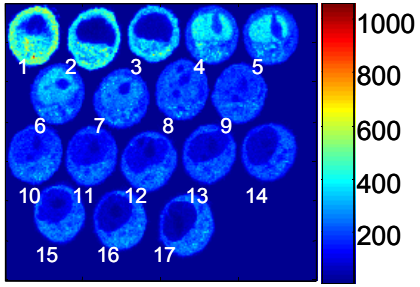
Materials Science & Technology



Engineering Sciences



Pulsed Power



Bioscience

Eric Keiter, Sandia National Laboratories
2014 NEEDS Workshop, Berkeley, CA

Hybrid-Hybrid solver result: 19x Speedup for Challenging IC

Necessary for efficient simulation of a CMOS logic circuit:

- 1.6M total devices, ~2M unknowns
- Xyce w/ KLU solver takes ~ **2 weeks**, w/ ShyLU solver takes ~ **1 day**
- ShyLU: Optimal # partitions = 64; number of rows in *S* = 1854 (4 MPI procs)

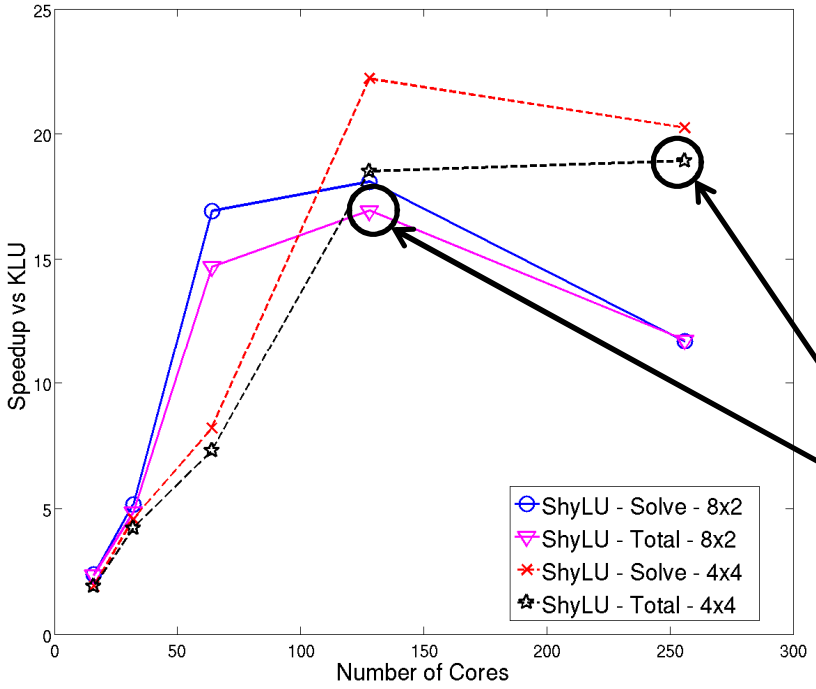


TABLE III
COMPARISON OF TOTAL LINEAR SOLVE TIME (SEC.) OF VARIOUS SPARSE DIRECT SOLVERS FOR OUR TEST CIRCUITS; (-) INDICATES SIMULATION FAILED TO COMPLETE.

	ckt1	ckt2	ckt3	ckt4	ckt5
KLU	**80.8**	162.2	9381.3	7060.8	**14222.7**
PARDISO (16)	128.6	**105.3**	**715.0**	**6690.5**	-
SuperLU	-	10294.1	-	-	72176.8
SuperLU_Dist (16)	-	-	-	-	-

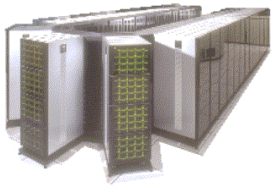
Strong scaling of Xyce's simulation time and ShyLU linear solve time for different configurations of MPI Tasks X Threads per node.

CIS has a rich history in the development and maturation of high performance computing hardware and software technology


CM-2


nCUBE-2


iPSC-860


Paragon


ASCI Red


Cplant


Red Storm

1987 1988 1989 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007

Gordon Bell Prize

R&D 100
Parallel Software

Patent
Meshing

R&D 100
Dense Solvers

R&D 100
Storage

Gordon Bell Prize

World Record
Teraflops

Gordon Bell Prize

R&D 100
Allocator

R&D 100
Trilinos

R&D 100
Xyce

Gordon Bell Prize

R&D 100
Signal Processing

SC96 Gold Medal
Networking

Mannheim
SuParCup

Karp Challenge

World Record
281 GFlops

R&D 100
Aztec

Patent
Data Mining

R&D 100
3D-Touch

R&D 100
Meshing

World Record
143 GFlops

R&D 100
Salvo

Patent
Parallel Software

Patent
Paving

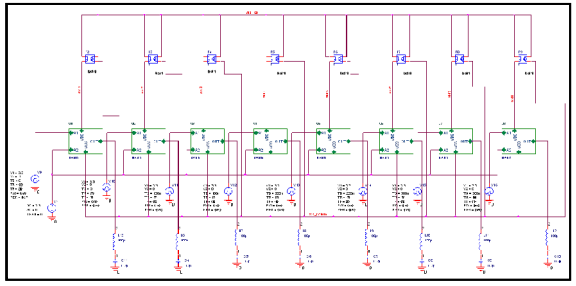
Patent
Decomposition

Fernbach
Award

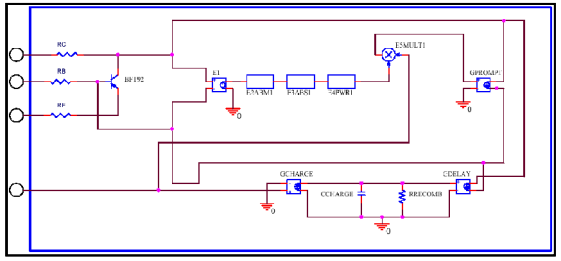
Eric Keiter, Sandia National Laboratories
2014 NEEDS Workshop, Berkeley, CA

Transforming Design Capabilities

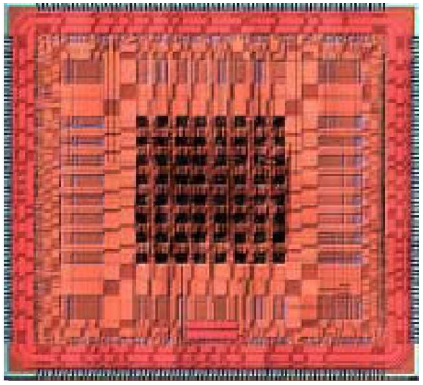
then ~ 2001
PSpice circuit model:
~300 devices



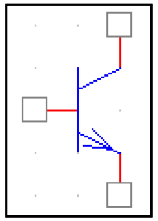
rad. model:
empirical/behavioral



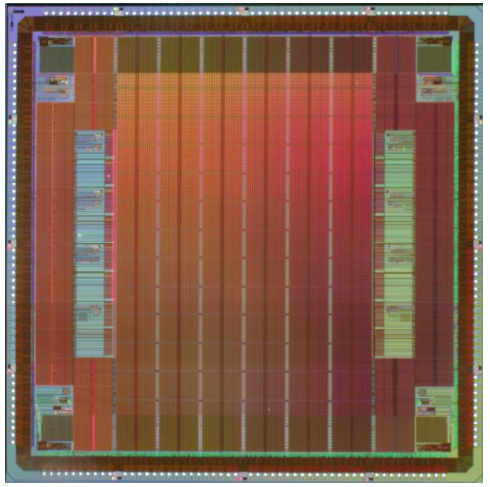
2006
Xyce circuit model:
300K+ devices



rad. model:
physics-based/built-in



201
2
Xyce circuit model:
40M+ devices



Xyce Requirements

Xyce started in 1999. Requirements/goals:

- SPICE-compatible
- Massively parallel
- Include special Sandia-specific physics models
- Useful to unsophisticated users
- Object-oriented, modular (C++)
- Extensible, lots of developers

Another way to put it:

- "Be the same as SPICE, but also be much better"
- *"I never want to see the "Time step too small" error again"* – Ken Marx, Sandia Electrical Analyst

Why not use SPICE?

SPICE: the original open-source simulator

de-facto standard

To be useful: modular, well-structured, flexible

separated numerics, algorithms, models, I/O

simple, clean interfaces

short, easy to read, easy to modify

i.e., **not SPICE!**

excerpt from *dioload.c* (SPICE3)

```
#ifdef SENSDEBUG
  printf("vd = %.7e \n",vd);
#endif /* SENSDEBUG */
  goto next1;
}
if(ckt->CKTmode & MODEINITSMSIG) {
  vd= *(ckt->CKTstate0 + here->DIOvoltage);
} else if (ckt->CKTmode & MODEINITTRAN) {
  vd= *(ckt->CKTstate1 + here->DIOvoltage);
} else if ( (ckt->CKTmode & MODEINITJCT) &&
  (ckt->CKTmode & MODETRANOP)
         && (ckt->CKTmode & MODEUIC) ) {
                vd=here->DIOinitCond;
} else if ( (ckt->CKTmode & MODEINITJCT) && here->DIOoff)
{
         vd=0;
} else if ( ckt->CKTmode & MODEINITJCT) {
         vd=here->DIOtVcrit;
} else if ( ckt->CKTmode & MODEINITFIX && here->DIOoff) {
         vd=0;
} else {
#ifndef PREDICTOR
         if (ckt->CKTmode & MODEINITPRED) {
```

Sensitivity analysis

AC analysis code

Transient analysis related code

Why Modularity is Necessary

Key to managing complexity

limits what each developer needs to know about the system

the “API” encapsulates rest of the simulator

Eg, to implement BSIM6.4, you don't need to know all about:

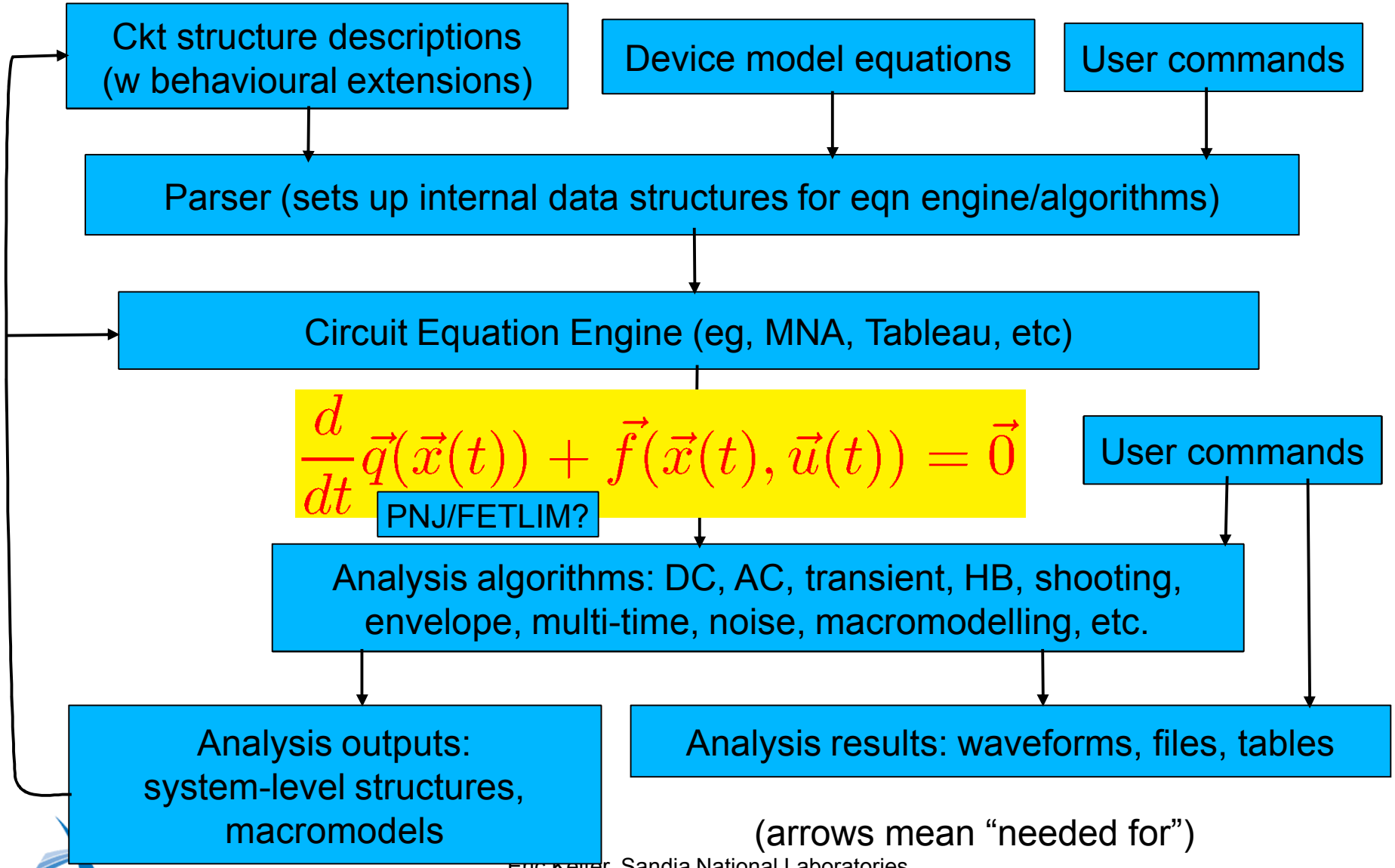
trapezoidal, Gear, etc. algorithms

shooting, harmonic balance, sensitivity, ...

Newton-Raphson, homotopy, ...

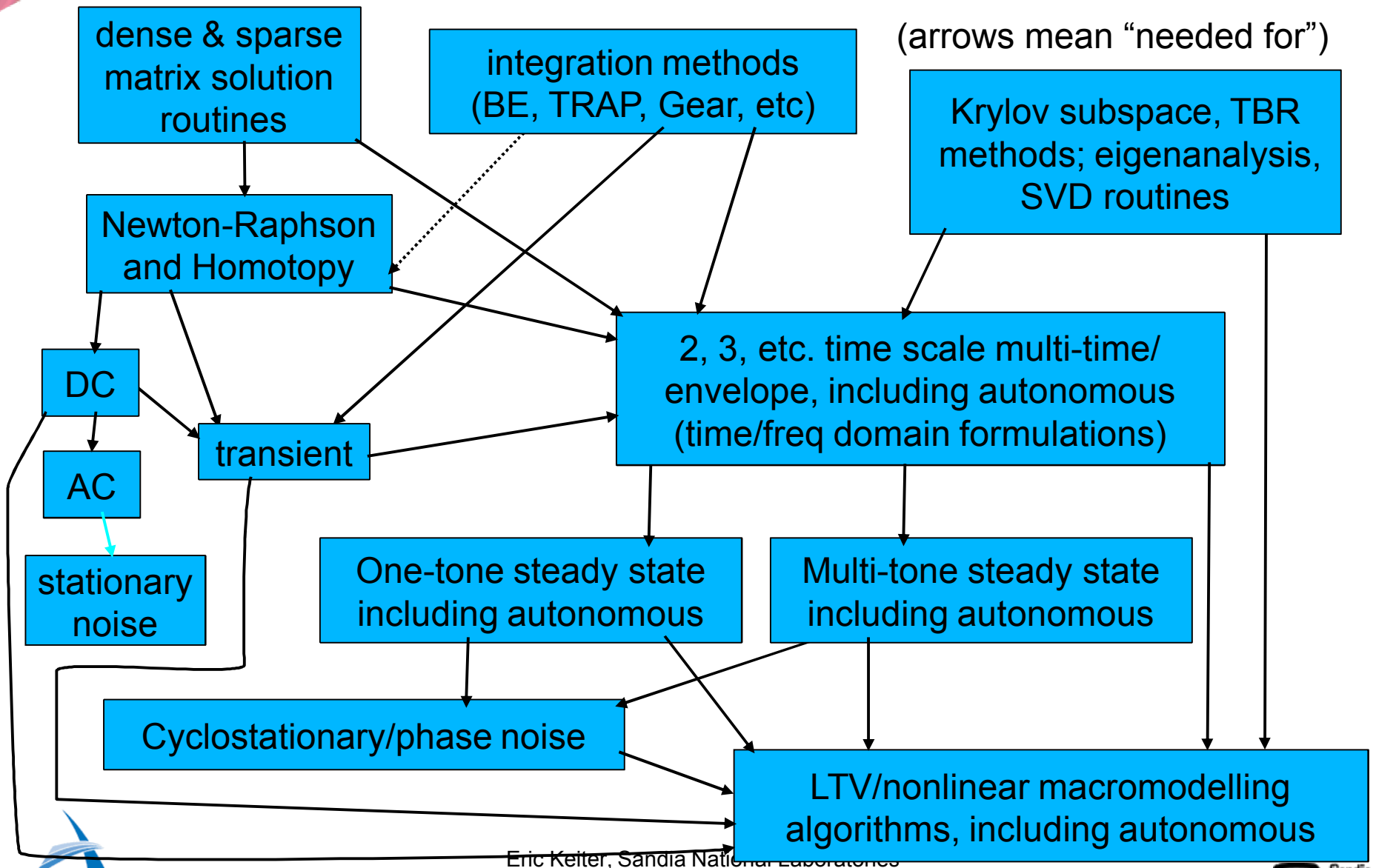
Errors and mistakes reduced

Modular Organization: The Big Picture

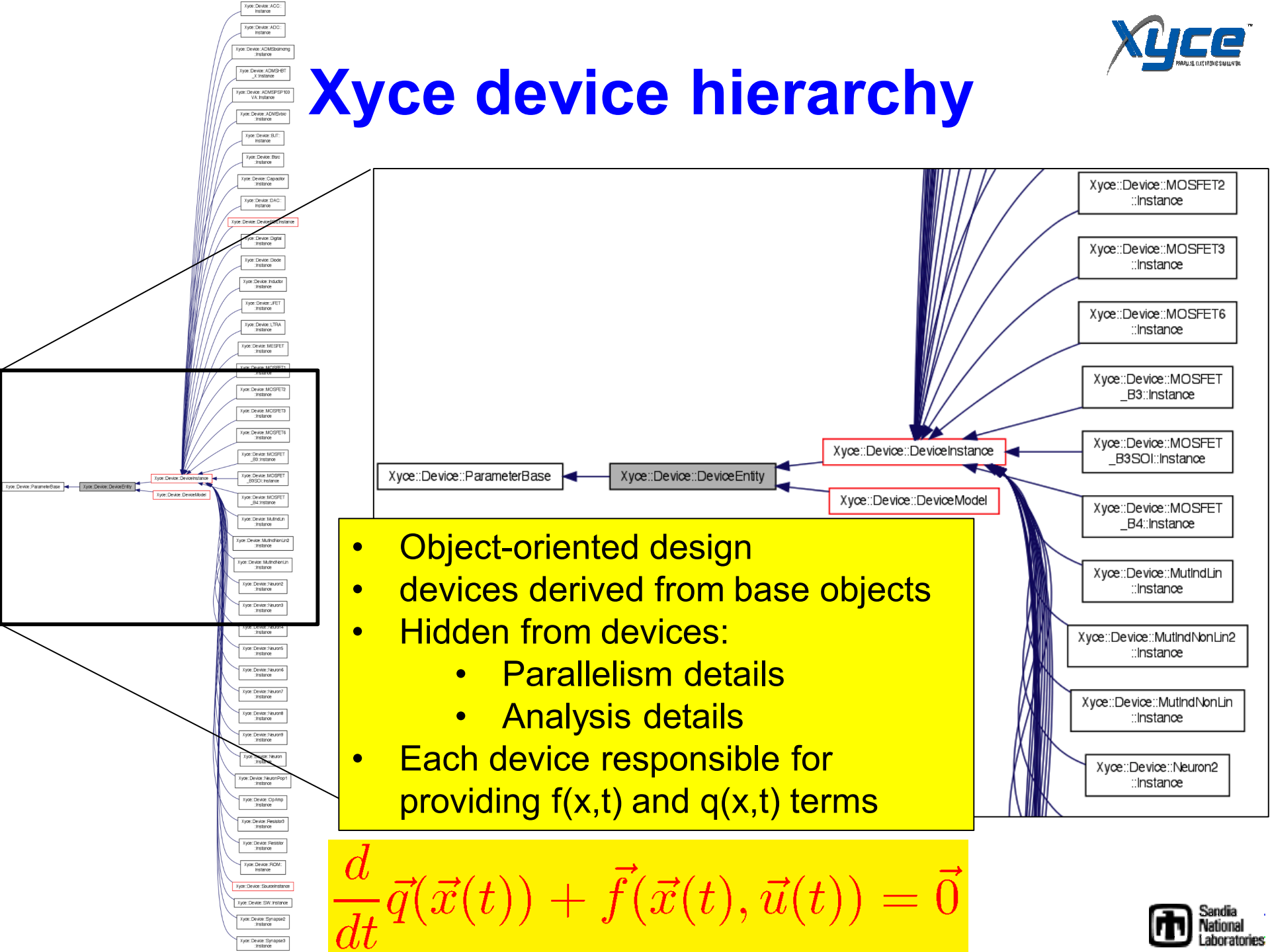


(arrows mean “needed for”)

Analysis Algorithms: Structure



Xyce device hierarchy



- Object-oriented design
- devices derived from base objects
- Hidden from devices:
  - Parallelism details
  - Analysis details
- Each device responsible for providing f(x,t) and q(x,t) terms

$$\frac{d}{dt}\vec{q}(\vec{x}(t)) + \vec{f}(\vec{x}(t), \vec{u}(t)) = \vec{0}$$

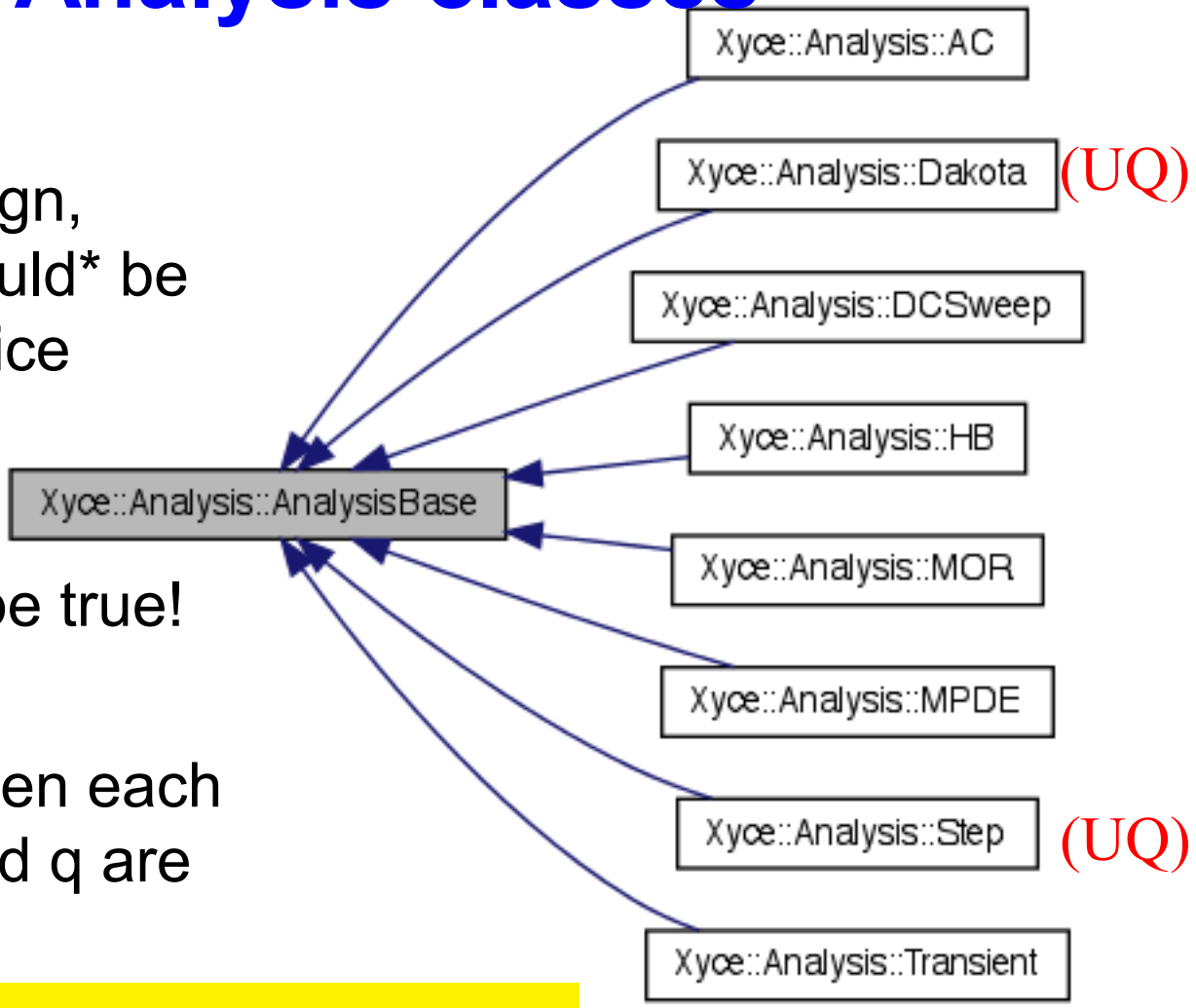
Sandia National Laboratories

Xyce Analysis classes

With proper code design, analysis types *should* be independent of device models

Reverse should also be true!

Main difference between each of these is how f and q are handled.



$$\frac{d}{dt}\vec{q}(\vec{x}(t)) + \vec{f}(\vec{x}(t), \vec{u}(t)) = \vec{0}$$

Harmonic Balance

- Expands state variables as a Fourier series; solves for the Fourier coefficients
  - Insensitive to widely spaced spectral components
  - Excellent for dealing with complex high-frequency passive (linear) components
  - Directly captures the large-signal quasi-periodic steady-state
- Standard set of circuit equations:

Circuit DAE: $$g(x(t)) + \frac{d}{dt} q(x(t)) = u(t)$$

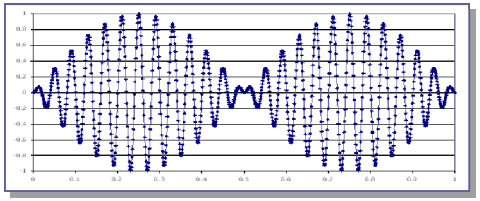
Approximate solution with Fourier expansion: $$x(t) = \sum_{h=-M}^{M} X_h \exp(j\omega_h t)$$

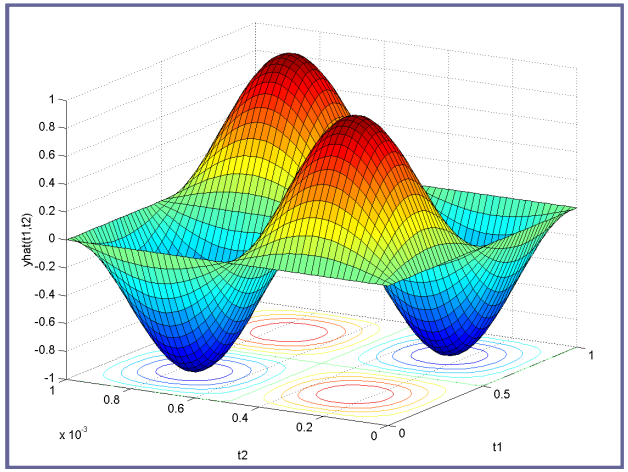
Frequency domain equation: $$G(X) + \Omega Q(X) = U$$

- Block linear system:
  - N=original system size
  - M=number of Fourier terms
  - Total size = N*M

MPDE Analysis

- Multi-time PDE (MPDE)
- Useful for problems with widely differing time scales
  - Slow envelope
  - Fast periodic
- Original DAE transformed into a PDE-like problem, where fast timescale solved all at once:





Original DAE $$\frac{d}{dt}q(x,t)+f(x,t)=b(t)$$

MPDE $$\frac{\partial}{\partial t_1}q(x,t_1,t_2)+\frac{\partial}{\partial t_2}q(x,t_1,t_2)+f(x,t_1,t_2)=b(t)$$

$$x(t)=\sin(2\pi t)\sin(2\pi 10^9 t)$$

$$\hat{x}(t_1,t_2)=\sin(2\pi t_1)\sin(2\pi 10^9 t_2)$$

$$x(t)=\hat{x}(t,t)$$

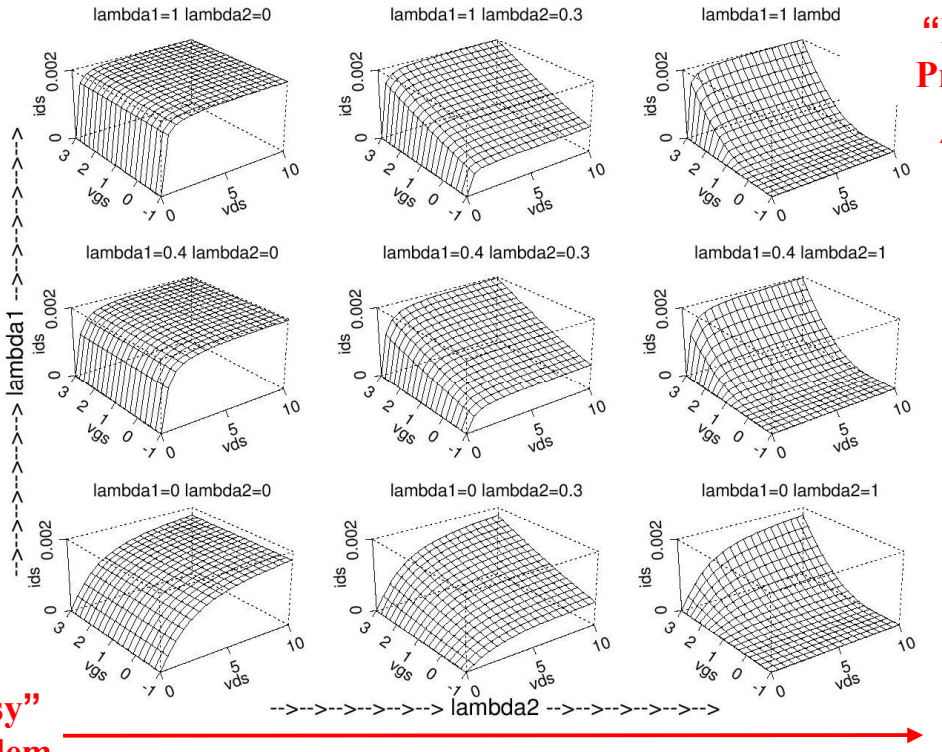
DCOP Non-linear solution techniques

- DCOP solution can be difficult
- Some circuits have multiple solns.
- Need to find stable solution.

- Homotopy:
  - Relax stiffness
  - continuation params
  - Obtain solution via continuat

- Common examples (Xyce & SPICE)
  - Voltage source stepping
  - GMIN stepping

- Less common (Xyce):
  - Pseudo-transient
  - MOSFET homotopy (see right)

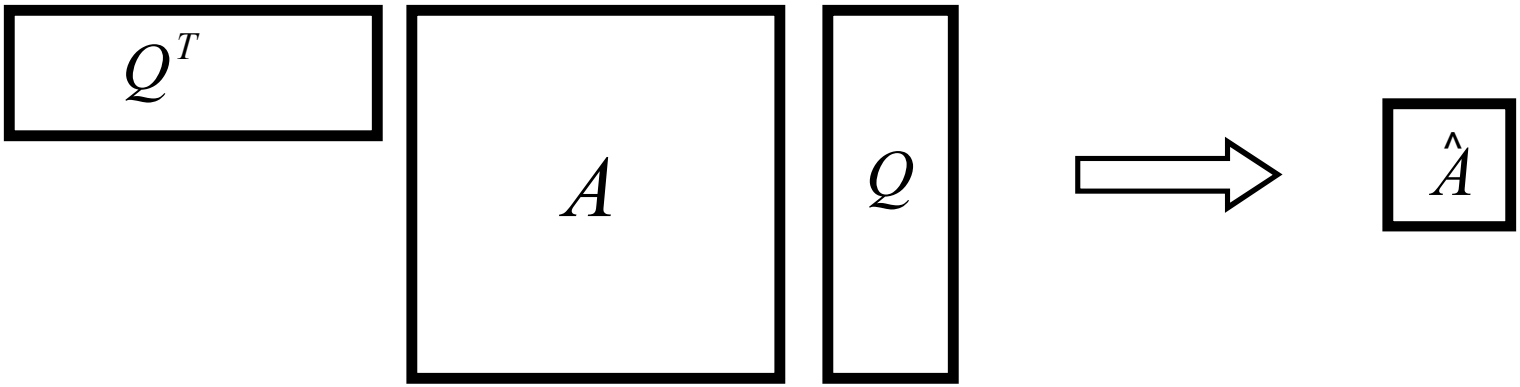


MOSFET continuation example

* Figure 12 from, Jaijeet Roychowdhury and Robert Melville, "Delivering Global DC Convergence for Large Mixed-Signal Circuits via Homotopy/Continuation Methods" IEEE Trans. CAD of ICAS, vol 25, no 1, 2006.

Model Order Reduction(MOR)

- Model reduction: in a rigorous manner, generate a system of the same form, but smaller dimension, with input-output behavior approximately the same.
- Very important for large linear networks, such as post-layout parasitics.
- Several projection methods are implemented in Xyce.
- Projection squashes matrices to smaller size

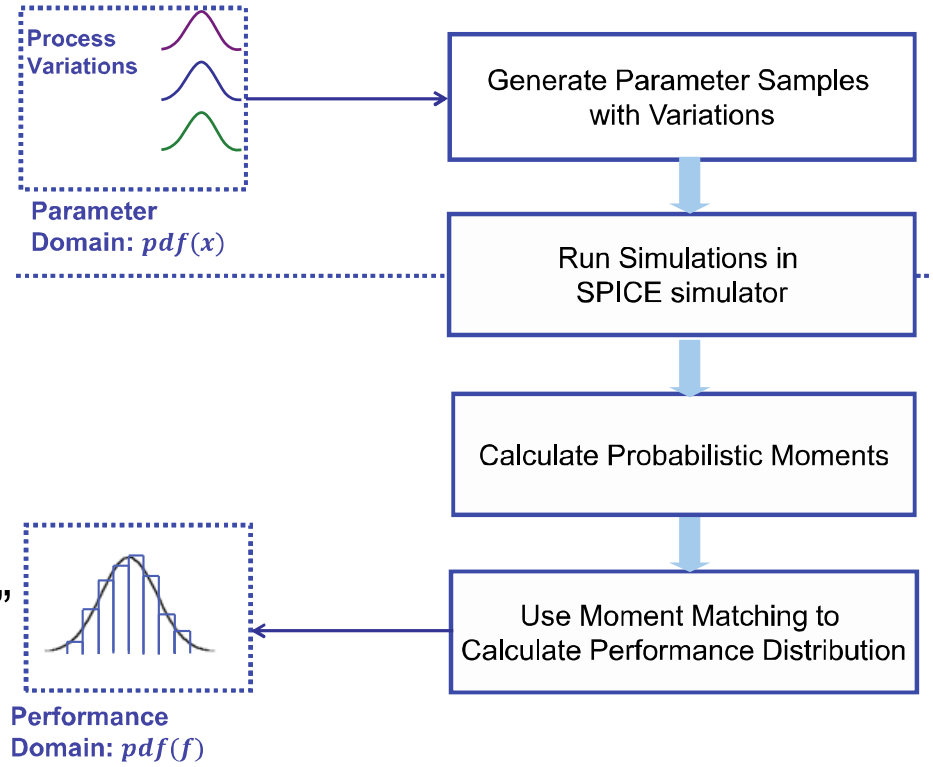


Uncertainty Quantification (UQ)

- Xyce has a number of capabilities to support uncertainty quantification (UQ)
- Mostly these come via the DAKOTA framework.

- General idea: given uncertainty on parameter inputs, how to estimate that on circuit outputs.

- Most simulators support “brute force” approaches based on sampling.
- Many much more sophisticated methods exist, including stochastic collocation methods, etc.



ADMS-Xyce

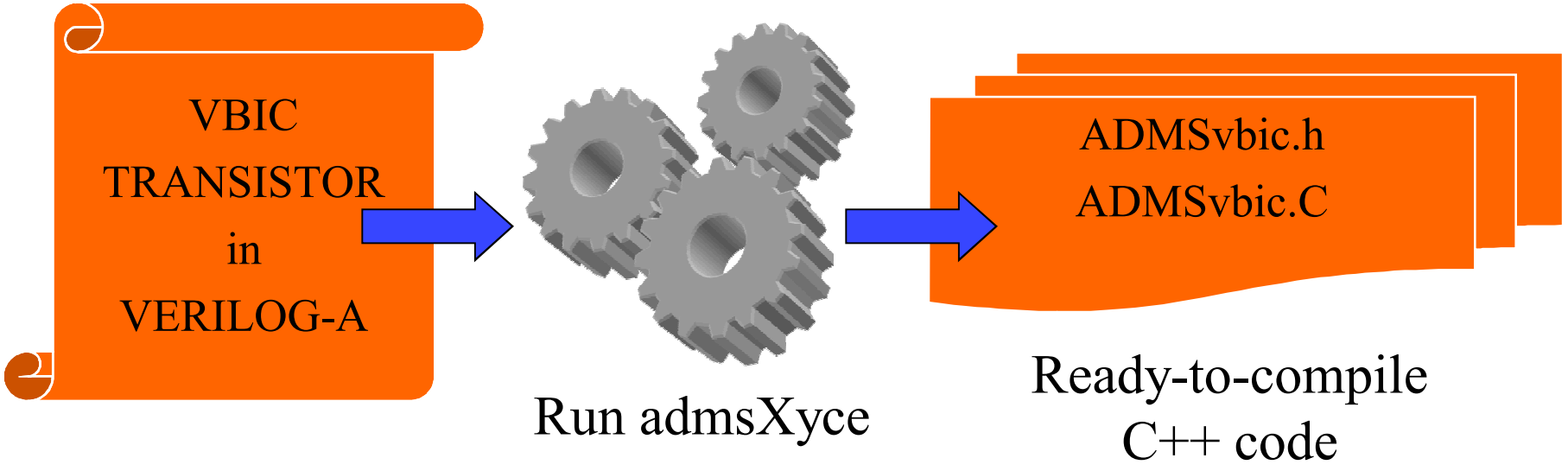
Verilog-A interface, via ADMS model compiler

- VBIC, Mextram, EKV, HiCUM, etc.

Verilog-A: industry standard format for new models

ADMS translates Verilog-A to compilable C/C++ code;

API automatically handles data structures, matrices, tedious details.



ADMS-Xyce

ADMS-converted models in Xyce:

- VBIC (the first model we did, only constant-phase so far)
- EKV (can't distribute thanks to NDA, so it's not in 6.0 open-source)
- PSP (version 103)
- BSIM-CMG 107 (only one of the many variants)
- FBH HBT_X version 2.1
- FBH HBT_X version 2.3 (we were able to process it into Xyce, but dropped this one after discussions with Matthias Rudolph (convergence problems))
- MEXTRAM (we don't support mextram, but the MEXTRAM verilog can be (and has been) processed by our ADMS back-end).
- MVS Silicon virtual source model from nanohub.com (processed last week in ADMS, not really tested yet)

ADMS-Xyce

What is unique about Xyce's use of ADMS?

- Sacado AD library: By using Sacado, we are able to do away with an enormous amount of "admst" code for computing derivatives (compare with ng-spice's or qucs ADMS back-ends). It is difficult to overstate how much back-end work this saves.
- Have added "$limit" support (simple mod to ADMS required to let "$limit" be recognized as a legal function, but otherwise all the work is in the back-end)
- Had to add some "attributes" for parameters and modules to provide certain metadata (descriptions, units, "model" vs. "instance", level number, whether it's a Q, M, Y device, etc.).

ADMS-Xyce

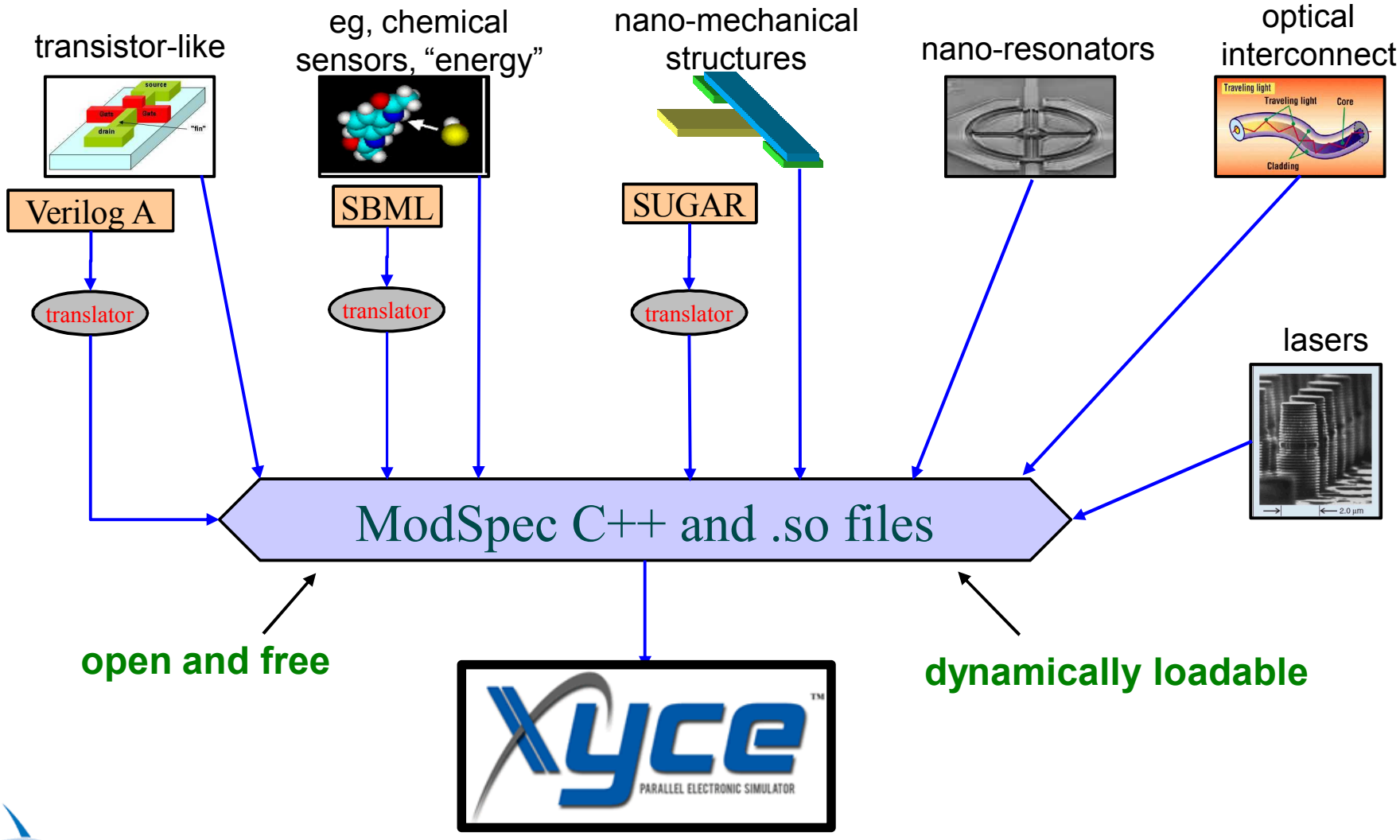
What challenges were there?

- ADMS back-ends are written in ADMST, a sort of XSLT superset.
- Xyce's linear system differs from SPICE so generating code for voltage limiting ($limit) is not as simple as generating a function call.
- ADMS has NO support for current branches. It generates datastructures representing the Jacobian resulting when all variables are nodal voltages, but you must do all the work yourself if you throw a branch current in the mix.
- Node collapse: since this has to happen at the time the device is instantiated (rather than the time its equations are evaluated), it is necessary to generate (separately from the rest of the evaluation) the parts of code needed to compute all the variables required to determine whether or not to collapse.
- Dynamic linking: this is on our roadmap, but hasn't been done yet.

Challenges with ADMS

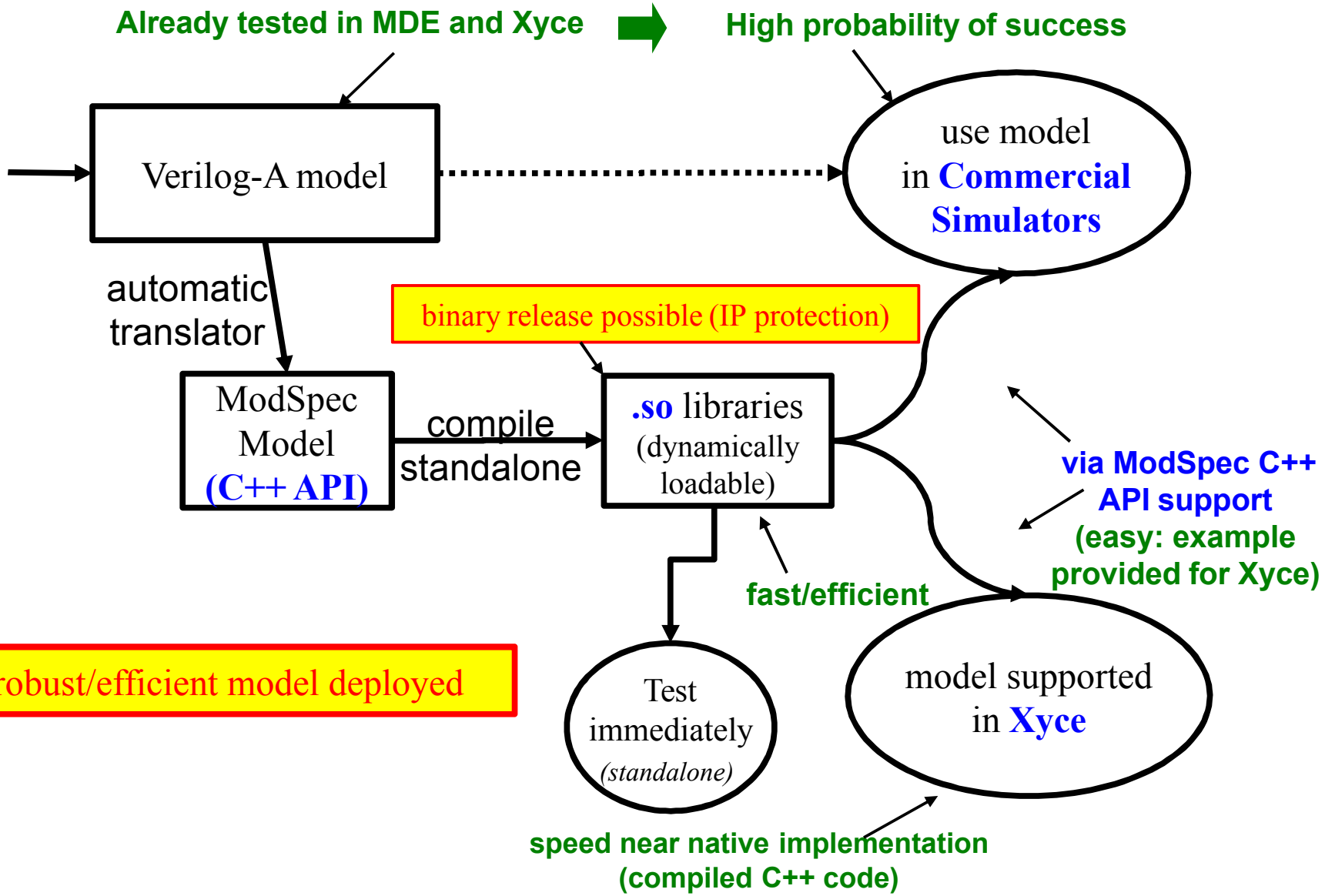
- not well suited for multi-physics domains
  - even for electrical domain, doesn't support structures like internal current unknowns
- no longer open-source
  - ADMS's website: "*Noovela Consulting offers - at interesting rate - support to implement your compact models into spice simulators*"
  - future support uncertain
- written in XSLT
  - difficult to work with or maintain
  - e.g. took 6-7 years by Tom Russo (working off and on on it) even with ADMS already existing
- dynamical linking not supported yet
  - compilation required

Multi-physics support in Xyce through ModSpec

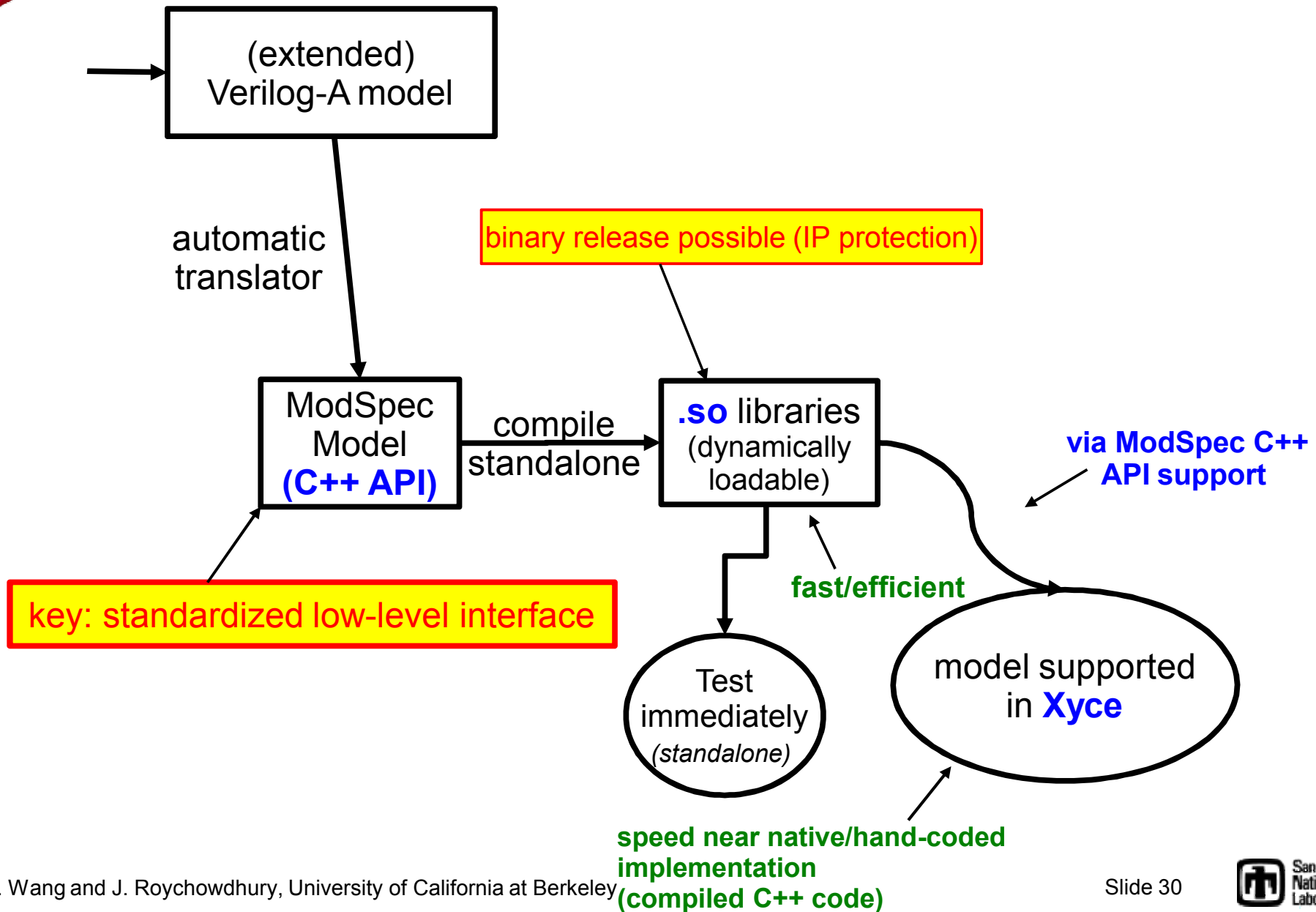


Eric Keiter, Sandia National Laboratories
2014 NEEDS Workshop, Berkeley, CA

Model Development: Verilog-A → ModSpec → Xyce



Using ModSpec in Xyce



Acknowledgements

Xyce team

xyce.sandia.gov

- Scott Hutchinson
- Tom Russo
- Heidi Thornquist
- Jason Verley
- Rich Schiek
- Ting Mei
- Dave Baur
- Sivasankaran Rajamanickam

Trilinos team

trilinos.sandia.gov

Dakota team

dakota.sandia.gov

Thanks!

Publications / Presentations

Publications

- *"Electrical Modeling and Simulation for Stockpile Stewardship"*, ACM XRDS, 2013
- *"ShyLU: A Hybrid-Hybrid Solver for Multicore Platforms"*, IPDPS 2012
- *"Parallel Transistor-Level Circuit Simulation"*, Simulation and Verification of Electronic and Biological Systems, Springer, 2011
- *"A Parallel Preconditioning Strategy for Efficient Transistor-Level Circuit Simulation"*, ICCAD 2009

Presentations

- *"Sparse Matrix Techniques for Next-Generation Parallel Transistor-Level Circuit Simulation"*

  Heidi K. Thornquist, Parallel Matrix Algorithms and Applications 2012
- *"Partitioning for Hybrid Solvers: ShyLU and HIPS"*

  Erik G. Boman, Siva Rajamanickam, and Jeremie Gaidamour, Copper Mtn. 2012
- *"Efficient Preconditioners for Large-Scale Parallel Circuit Simulation"*

  Heidi K. Thornquist, SIAM Computational Science & Engineering 2011
- *"Advances in Parallel Transistor-Level Circuit Simulation"*

  Heidi K. Thornquist, Scientific Computing in Electrical Engineering 2010
- *"Large Scale Parallel Circuit Simulation"*

  Heidi Thornquist and Eric Keiter, Circuit and Multi-Domain Simulation Workshop, ICCAD 2009