International Journal of Power Electronics and Drive Systems (IJPEDS)
Vol. 6, No. 4, December 2015, pp. 665 – 682
ISSN: 2088-8694

Co-Simulation Interfacing Capabilities in Device-Level Power Electronic Circuit Simulation Tools: An Overview

Wentao Wang*, Wenying Yang, and Venkata Dinavahi***

*Department of Electrical & Computer Engineering, University of Alberta, Edmonton, AB T6G 2V4 Canada.
**School of Electrical Engineering & Automation, Harbin Institute of Technology, Harbin 150001 China.

Article Info

Article history:
Received Jun 20, 2015
Revised Aug 13, 2015
Accepted Aug 29, 2015

Keyword:
Co-simulation
Circuit simulators
Device-level modeling
Interfacing,
Power electronics
System-level simulation.

ABSTRACT

Power electronic circuit simulation today has become increasingly more demanding in both the speed and accuracy. Whilst almost every simulator has its own advantages and disadvantages, co-simulations are becoming more prevalent. This paper provides an overview of the co-simulation capabilities of device-level circuit simulators. More specifically, a listing of device-level simulators with their salient features are compared and contrasted. The co-simulation interfaces between several simulation tools are discussed. A case study is presented to demonstrate the co-simulation between a device-level simulator (PSIM) interfacing a system-level simulator (Simulink), and a finite element simulation tool (FLUX). Results demonstrate the necessity and convenience as well as the drawbacks of such a comprehensive simulation.

Corresponding Author:
Venkata Dinavahi
Dept. of Electrical & Computer Engineering
University of Alberta
9107 116 St., Edmonton, AB T6G 2V4 Canada
Ph: (780) 492-1003
Email: dinavahi@ualberta.ca

1. INTRODUCTION

Device-level circuit simulators are essential to tackle the increasing demand of the accuracy and precision in the design and implementation of complex systems. The special advantage of such simulators is based on their conformity to fundamental physical principles, thereby achieving more accurate simulation results [1],[2]. However, since device-level simulators have to depict the waveform of a design node in great detail, they are relatively slow in execution speed, which is the key bottleneck in large design projects. Besides accuracy and speed, many system-level simulation projects even require specific analyses (e.g. transient, statistical analysis) in certain parts of the systems, which could only be achieved using circuit simulators [3, 4]. In such cases, hierarchical design structure strategies become necessary. For example, some large power system simulation tasks may require implementation in both system-level like Matlab/Simulink® and device-level circuit simulators to exploit the modeling strengths in their respective areas of simulation [5],[6]. Meanwhile, many of the design projects today may include analog, digital and mixed-signal simulations, which make it necessary for analog and mixed-signal simulators to support the function of co-simulation within standard digital simulators [7]. Furthermore, researchers have endeavored to devise methods to enable device-level circuit simulators to co-simulate with programming languages like SystemC™, which are especially useful during hardware realization [8],[9]. As a result of all aforementioned factors, the discussion of co-simulation issue for device-level circuit simulators becomes paramount.

Another important issue about device-level circuit simulators obviously is the numerical solution method. For nonlinear dc, transient and linear ac circuit simulations, the circuit may comprise of semiconductor devices such as diodes, BJTs, JFETs, MOSFETs and IGBTs; linear/nonlinear resistors, capacitors, inductors, independent/dependent voltage and current sources, etc. [1] All these actual devices can be approximated by built-in or user-defined models, where the complete system of models can be highly coupled. Standard methods exist to solve nonlinear differential

algebraic equations. Typically, numerical integration, the Newton-Raphson method, and sparse-matrix techniques are applied [10]. One of the circuit simulators, that applies the standard method is SPICE [11]. This is also true of other simulators including those among the SPICE family which are essentially based on the kernel of SPICE [12], [13], [14]. Although the standard methods are efficacious in many applications, they were invented several decades ago, and they have the drawback of slow computing speed, especially when used in conjunction with larger circuit simulations. Therefore, there is an urgent need to devise more efficient methods of solution. Actually, many of these kinds of novel simulators have been developed, which include MEDUSA and CODECS [15],[16]. The purpose of this paper is to enumerate co-simulation interfaces in device-level circuit simulators, and to identify their specific application in the literature. A PSIM®-Simulink®-FLUX® case study is provided to demonstrate the effectiveness of the co-simulation interfacing in device-level circuit simulators.

2. GENERIC NUMERICAL SOLUTION APPROACH IN DEVICE-LEVEL CIRCUIT SIMULATORS

While there are a handful of commonly used and popular device-level circuit simulators, this study has revealed a host of other tools that are less well known but nonetheless unique in their modeling and simulation capabilities. Table 1 enumerates the device-level circuit simulators found in this study with their salient features and relevant references, with a special emphasis on their advantages and disadvantages from a user's point of view. These simulators can be broadly categorized into research-oriented simulators (e.g., SPICE, Ngspice, QUCS, MEDUSA, and CODECS) and commercial simulators (e.g., PSpice®, Saber®, and PSIM®). In the tables, research-oriented simulators are marked by R and commercial ones are marked by C. Furthermore, a group of simulators, including SPICE, Spice2, Spice3, Ngspice, PSpice, HomSPICE, MultiSim™and so on can be categorized as SPICE-like simulators, which are essentially based on the kernel of SPICE [12], [13], [14], and their transient analysis operations are basically the same. In this section, the numerical solution approach for transient analysis used by a majority of device-level simulators is described.

The first step is the system matrix formulation. The modified nodal analysis (MNA) approach is taken by Saber and SPICE-like simulators, where the system is represented by a group of nonlinear first-order differential algebraic equations (DAEs):

$$N(\mathbf{x}(t), \frac{d\mathbf{x}(t)}{dt}, t) = \mathbf{0} \tag{1}$$

where $\mathbf{x}(t)$ is the vector of unknown circuit variables, and $N(\cdot)$ are the nonlinear vector functions [1], [11]. In contrast, other circuit simulators such as PECS, utilize a state-space approach to undertake the system matrix formulation step [30].

The nonlinear DAE group contain nonlinear ordinary differential equations (ODEs) of the form [2]:

$$\frac{d\mathbf{x}(t)}{dt} = o(\mathbf{x}(t), t) \tag{2}$$

To solve (2) at the next time-step t_{n+1}, numerical integration is applied. For instance, SPICE-like simulators and Saber use Gear or Trapezoidal methods.

For the Trapezoidal method,

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \frac{h}{2}[o(\mathbf{x}_{n+1}, t_{n+1}) + o(\mathbf{x}_n, t_n)] \tag{3}$$

where $\mathbf{x}_{n+1}$ is the solution of next time-step t_{n+1}, and $\mathbf{x}_n$ is the solution at the current time-step t_n [2].

For the second-order Gear method (the default method for Saber),

$$\mathbf{x}_{n+1} = \frac{4}{3}\mathbf{x}_n - \frac{1}{3}\mathbf{x}_{n-1} + \frac{2}{3}h \cdot o(\mathbf{x}_{n+1}, t_{n+1}). \tag{4}$$

Thus (2) is successfully transformed into nonlinear algebraic equations (NAEs), which have the general form of

$$F(\mathbf{x}) = \mathbf{0}, \tag{5}$$

where $\mathbf{x} \equiv \mathbf{x}_{n+1}$, and $F(\cdot)$ is the general nonlinear operator [1], [11].

To solve these NAEs, (5) needs to be linearized in this step. SPICE-like simulators and Saber use Newton-Raphson or the Katzenelson algorithms. (5) can be further written as

$$F(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x})]^T, \tag{6}$$

Table 1. Comparison of Device-Level Circuit Simulators

Simulators	★: Salient Features; ▲: Advantages; ▼: Disadvantages
SPICE (Spice2, Spice3) R	★: Simulation Program with Integrated Circuit Emphasis (SPICE), the most popular general purpose and open source analog circuit simulator[12]; ▲: i) Large model library; ii) Efficient solution method: finite difference numerical method; iii) Three time-step strategies: iteration count, dV/dt control, and local truncation error (LTE); iv) Simulation studies: DC and AC analysis, DC transfer curve analysis, Noise analysis, Transient analysis. ▼: i) Less simulation accuracy in a complex circuit environment; ii) Limited interfacing capability.
HomSPICE R	★: A member of SPICE-family circuit simulators, uses three homotopy algorithms: FIXPDF, FIXPNF, and FIXPQF, which is in favor of calculating a circuit's dc operating points and periodic steady-state response [13]. ▲: Parameter embedding methods are robust and accurate numerical techniques for solving nonlinear algebraic equations; ▼: Computational intensity is high.
JSPICE R	★: Based on Spice2, designed for superconductor and semiconductor circuits [17]; ▲: i) Can be incorporated with the Josephson junction model; ii) Supports the same Spice2 format and runs in the batch mode. ▼: i) Only valid for transient simulations; ii) DC operating point and AC small-signal analyses are not allowed.
MacSpice R	★: A Mac version of SPICE, open source and free for non-commercial use only [18]. ▲: i) An adaptive step-size algorithm is used to gain better convergence during a transient analysis; ii) Provides a robust multi-parameter optimizer and facility for interprocess communication with other applications. ▼: Limited interfacing capability.
XSpice R	★: Based on SPICE, but further incorporates arbitrary user models [19]. ▲: i) Supports for adding "code models" written in the C programming language and contain over 40 new functional blocks; ii) An embedded event-driven algorithm is coordinated with the analog simulation algorithm to provide fast and accurate simulation of mixed-signal circuits and systems; iii) An interprocess communication interface for connection to CAE system software. ▼: AC analysis is not supported for circuits with event-driven nodes.
Ngspice R	★: A mixed-level/mixed-signal circuit simulator [20]; Based on three software packages: Spice3f5, Xspice and Cider1b1; Ngspice is part of gEDA project, development from its users contribution; ▲: i) Provides additional C language code models to support analog behavioral modeling and co-simulation of digital components through a fast event driven algorithm; ii) Involves a numerical device simulator to couple the circuit level simulator to the device simulator to provide enhanced simulation accuracy (at the expense of increased simulation time); iii) Migrates to other commercial SPICE simulator flawlessly; iv) Supports both Windows and Linux platforms; ▼: i) Many SPICE parameters will not be supported by Ngspice and simulation results can be inaccurate; ii) Does not provide schematic capture function.
QUCS R	★: An integrated circuit simulator with a graphical user interface[21]; Simulates the large-signal, small-signal and noise behavior of the circuit. Pure digital simulations are also supported using VHDL and/or Verilog; ▲: i) A graphical interface for schematic capture. Simulation data can be represented in various types of diagrams, including Smith-Chart, Cartesian, Tabular and so on; ii) Existing SPICE models can be imported for use; ▼: Only supports the GNU/Linux OS.
HSpice® C	★:An analog circuit simulator similar to Spice3 but has better convergence, commercial product from Synopsys® [22]. ▲: i) Performs transient, steady-state, and frequency domain analyses; ii) Better convergence than SPICE3. ▼: i) GUI (graphic user interface) is not friendly; ii) Time-step setting is a little complicated when performing high frequency analysis.
Orcad®/PSpice C	★: As a PC version of SPICE, PSpice® is a dominant industrial standard for circuit and system analysis, works in analog and mixed signal environments, supports the functions for analog behavioral modeling [14] [23]; ▲: i) Comprehensive model library (about 30000); ii) Powerful waveform/simulation result analysis interface; iii) Offers models for all kinds of devices like Electromechanical systems (Resolvers, Brushless

Table 1. Comparison of Device-Level Circuit Simulators (Continued)

Simulators	★: Salient Features; ▲: Advantages; ▼: Disadvantages
	DC motors etc.), Mechanical systems (Flywheel etc.); iv) Support co-simulation with Matlab; v) With several Advance Analysis modes - check for reliability /stress (SMOKE Analysis), calculates Yield for multiple goals (Monte Carlo), Optimization Module with multiple optimization engines (Optimizer), Design Space explorations (Parametric Plotter); vi) Well integrated with complete system design and analysis tools (PCB Editor). ▼: i) Allow user to select specific components with industry standard part numbers and specifications. But searching for these components is time-consuming; ii) Complex circuit simulator; iii) The setting of simulation parameters is critical and difficult to set in order to avoid numerical convergence problems. iv) No data visualization during simulation.
Saber® C	★: A comprehensive mixed-signal simulator, provides a versatile modeling language named MAST, which makes it possible to divide specific models from simulation algorithms; applied to electrical, optical, thermal, mechanical systems [10], [24]. ▲: i) Comprehensive model library: 30,000+ models; ii) Accurate and efficient solution method: adopts the piecewise linear evaluation technique and subdivision; iii) Flexible time-step strategy: fixed and variable; iv) Very friendly GUI: for generating virtual prototypes of power system networks; v) Powerful interfacing capability: with popular 3D CAD tools (Catia V5, Siemens (UGS), Pro/E), MATLAB/Simulink, Zuken, Mentor Graphics, Cadence, Synopsys VCS; vi) Various simulation studies: DC and AC static (steady) or transient solution, robust design methods (e.g., Stress, Sensitivity, Monte Carlo, etc.) ; vii) Wide range of industrial applications, for design validation and optimization for automotive, aerospace, industrial power and energy systems; viii) Supports for MAST and VHDL-AMS language standards; ix) Verify the behavior of physical systems (i.e., electrical, mechanical, hydraulic etc.); x) Use grid computing to minimize time for compute intensive statistical analyses. ▼: i) No front end for algorithmic modeling; ii) No links to rapid prototyping or to operation with hardware-in-the-loop Simulation; iii) Co-simulation with Simulink requires that Saber is running.
PSIM® C	★: A strong simulation platform for power electronics and motor drive control [25]. ▲: i) With strong algorithm dedicated to electrical circuits (piecewise method, generic models and a fixed time-step) and simulation times are significantly reduced; ii) Friendly user interface; iii) Powerful interfacing capability: with Matlab/Simulink, JMag and Magnet; Link to external C/C++ Code via DLL. iv) Various simulation studies: AC Analysis, AC Sweep, Harmonic Analysis, Motor Drive Analysis, Switch Losses Calculation and Thermal Analysis. ▼: Complexity of the block diagrams used to simulate the power circuits can increase drastically with the number of semi-conductors in the circuit.
PLECS® C	★: A Piece-wise Linear Electrical Circuit Simulator (PLECS) based on the state-variable formulation working within Matlab/Simulink environment and integrating circuits entered into Simulink as S-functions [26]. ▲: i) Direct access to Matlab/Simulink environment. ii) Nonlinear component and thermal modeling library are included. ▼: i) Semiconductor models are not including all effects. ii) Only electrical domain simulation (some thermal domain components exist). iii) Limited to place electric components in restricted modeling area.
HSIM® C	★: Designed to meet the requirement of nanometer circuit analysis, able to perform hierarchical simulation, commercial product developed by Synopsys® [27]; ▲: i) Techniques of hierarchical storage and isomorphic matching in HSIM speed up the simulation of large circuits; ii) Full SPICE functionality: AC, DC, transient, Monte Carlo and FFT analyses. ▼: Needs to set many options to get the correct simulation.
XSIM R	★: An efficient crosstalk simulator, based on indigenous methodology in Visual C++[28]. ▲: i) Friendly graphical user interface; ii) Executes a parallel application with a virtual wall clock time. ▼: i) Simulation studies are limited; ii) Needs to enlarge the model library.
Multisim™ C	★: A updated version of SPICE; software simulation kit provides dynamic simulation models, with powerful interactivity; has powerful Design-Rules-Check and Connectivity Check with the breadboard tool [12], [29]. ▲: i) Comprehensive model library; ii) Friendly graphical user interface; iii) Various simulation studies. ▼: Interfacing capability should be further improved.

Table 1. Comparison of Device-Level Circuit Simulators (Continued)

Simulators	★: Salient Features; ▲: Advantages; ▼: Disadvantages
PECS R	★: Power Electronics Circuit Simulator (PECS), excels in time-domain simulation of switched networks with nonlinearity[30]. ▲: i) Friendly graphical user interface; ii) State-space method for the analysis of switched networks is adopted to achieve both high speed and accuracy. ▼: i) Model library is small; ii) Simulation studies are limited.
TITAN C	★: A complete customizable simulator, which provides the freedom to divide the whole circuit into arbitrary subcircuits [31]. ▲: i) Various simulation studies; ii) Circuit equations are solved by special vectorized solver. ▼: Interfacing capability should be improved.
PETS R	★: Power Electronics Transient Simulator (PETS), used for time-domain analysis, provides freedom to choose different degrees of complexity for piecewise-linear models[32]. ▲: i) Supports continuously differentiable nonlinear models by applying a delay approximation method with Newton-Raphson iteration. ii) Automatic time-step control. ▼: i) Model library is small; ii) Simulation studies are limited.
Spectre® C	★: An improved SPICE-like analog simulator from Cadence® [33]. ▲: i) Provides an adaptive time-step control algorithm that reliably follows rapid changes in the solution waveforms; ii) Improves simulation speed by increasing the efficiency of the simulator, typically two to three times faster than SPICE; iii) Various simulation studies: DC and AC analysis, Monte Carlo analysis, S-Parameter analysis, etc. ▼: Limited interfacing capability.
MEDUSA R	★: A user-oriented simulator for modular circuits, satisfies the needs for device and circuit simulations at the same time [15]. ▲: Solves the large circuits effectively by utilizing system modularity. ▼: i) Simulation studies are limited; ii) Interfacing capability could be improved.
CODECS R	★: A mixed-level circuit simulator, based on Spice3, while incorporates a set of numerical models under its main structure without changing its Spice3 core [16]. ▲: i) Supports various simulation studies: dc, transient, small-signal ac and pole-zero analysis; ii) Numerical models in CODECS include physical effects, such as bandgap narrowing, Shockley-Hall-Read and Auger recombination, etc. ▼: Simulation efficiency could be improved.
CASPOC® C	★: A multi-level simulator developed for the simulation/animation of power electronics and electrical drives, especially suited for the simulation of switching circuits with highly nonlinear models [34, 35]. ▲: i) Adds flexible robust non-linear function solver for modeling non-linear components; ii) Friendly user interface; iii) Flexible interfacing capability: coupling to Simulink, Tesla (a machine design tool) and ANSYS®. ▼: Simulation studies can be added more; ii) Simulation accuracy should be improved.
AWEswit C	★: A mixed analog and digital circuit simulator, special for the switched capacitor circuits [36]. ▲: i) Employs asymptotic waveform evaluation (AWE) technique to efficiently evaluate more detailed circuit models; ii) Provides flexibility in circuit formulations. ▼: i) Simulation studies are limited; ii) Interfacing capability should be improved.
DesignLab C	★: A Windows version of PSpice®, developed in the form of web pages with multimedia effect for analysis and design of circuits and electronics [37]. ▲: i) Comprehensive model library; ii) Various simulation studies: DC and AC, transient, Monte Carlo analysis, etc.; iii) Good interfacing capability. ▼: i) User interface still requires refinement; ii) No data visualization during simulation.
Eldo™ C	★: An analog, digital and mixed circuit simulator with a VHDL-based Analog Hardware Language [38]. ▲: i) Extensive device model libraries including leading MOS, bipolar and MESFET transistor; ii) Offers a unique partitioning scheme allowing the use of different algorithms on differing portions of design; iii) 3× to 10× gain in simulation speed over other commercial SPICE simulators, while maintaining same accuracy; iv) Various simulation studies: pole-zero, enhanced Monte-Carlo analysis and reliability simulation. ▼: Weak interface capability with other simulators.
IsSPICE4 C	★: An improved version from Spice3f5 and XSpice, adding some strong interactive features and extensions[39]. ▲: i) Adds a variety of new models: lossy transmission line model, GaAs Mesfet models, JFET model, etc.; ii) Friendly user interface; iii) Various simulation

Table 1. Comparison of Device-Level Circuit Simulators (Continued)

Simulators	★: Salient Features; ▲: Advantages; ▼: Disadvantages
	studies: DC and AC sensitivity, transient, pole-zero analysis, etc. ▼: Limited Interfacing capability.
Gnucap R	★: A general purpose mixed analog and digital circuit simulator, fully interactive, compatible to SPICE, containing a simple behavioral modeling language [40]. ▲: i) Employs plugins to make the simulation extremely flexible; ii) Flexible interface to other software; iii) Improves accuracy through rigorous error control method; iv) Multiple simulation languages, including Spice, Verilog and Spectre. ▼: User interface requires work.
LTSpice R	★: A high performance simulator, with many enhancements based on traditional SPICE simulator [41]. ▲: i) Stable SPICE circuit simulation with unlimited number of nodes; ii) Fast simulation of switching mode power supplies (SMPS); iii) Large model library: over 1100 macromodels of Linear Technology products and 500+ SMPS. ▼: i) More simulation studies are better to add; ii) Interfacing capability with other simulators can be improved.
TINA-TI™ C	★: A user-friendly, powerful circuit simulator based on a version of SPICE [42]. ▲: i) Includes more SPICE models and example circuits; ii) Supports multi-core processor and make simulations run 2-20 times faster; iii) Friendly user graphic interface; iv) Various simulation studies: DC, AC, transient, noise analysis. ▼: Interfacing capability with other simulators can be improved.

where $f_1(\cdot), f_2(\cdot), \ldots, f_m(\cdot)$ are all nonlinear operators [1].

For the Newton-Raphson algorithm, the Jacobian matrix must be formulated, which is given as

$$J(\mathbf{x}_k) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}|_{\mathbf{x}_k} & \frac{\partial f_1}{\partial x_2}|_{\mathbf{x}_k} & \cdots & \frac{\partial f_1}{\partial x_m}|_{\mathbf{x}_k} \\ \vdots & & & \vdots \\ \frac{\partial f_m}{\partial x_1}|_{\mathbf{x}_k} & \frac{\partial f_m}{\partial x_2}|_{\mathbf{x}_k} & \cdots & \frac{\partial f_m}{\partial x_m}|_{\mathbf{x}_k} \end{pmatrix} \tag{7}$$

where $\mathbf{x}_k$ is the solution at the kth Newton iteration [1]. Then, a linearized system of equations is obtained as:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - J(\mathbf{x}_k)^{-1} F(\mathbf{x}_k) \tag{8}$$

where $\mathbf{x}_{k+1}$ is the solution at the $(k+1)$th iteration [1].

Notably, when evaluating the Jacobian matrix, Saber applies a simplified subdivision technique to calculate the first derivatives thereby reducing the computational burden at every iteration, which is different from SPICE [10]. When applying the Newton-Raphson method, the SPICE-like simulators apply pre-specified tolerances (e.g. *ABSTOL*, *RELTOL*, and *CHGTOL*) to determine convergence to a valid solution. In contrast, Saber uses no tolerance to determine convergence, since the system of equations are evaluated piecewise linearly and solved exactly [10]. The CODECS simulator, however, use a modified two-level Newton algorithm [16] in this step. Apart from Newton-Raphson, the Katzenelson algorithm is based on piecewise linear systems. Sample points are used to find the linear regions for every nonlinear device [2]. Finally, in order to solve the linear algebraic equations (8), SPICE-like simulators and Saber use the methods of Gaussian elimination or LU decomposition techniques with forward and backward substitutions [1], [11], whilst the TITAN simulator uses a vectorized solver method [31]. The aforementioned transient analysis procedure (for e.g., in Saber) can be illustrated by the flowchart in Fig. 1.

For formulating the Jacobian matrix other circuit simulators mostly utilize the standard techniques mentioned above, however, but they also use alternate methods. For example, PETS uses a novel algorithm to decide the accurate element states of its piecewise-linear networks as well as an efficient way to avoid its piecewise-linear and reactive elements from changing values with each time-step, thereby keeping the system matrix constant [32].

3. CO-SIMULATION TECHNIQUES IN DEVICE-LEVEL CIRCUIT SIMULATORS

There are three main co-simulation patterns: co-simulation of device-level circuit simulators and system-level simulators, analog and digital co-simulation of circuit simulators, and co-simulation of circuit simulators with programming languages.

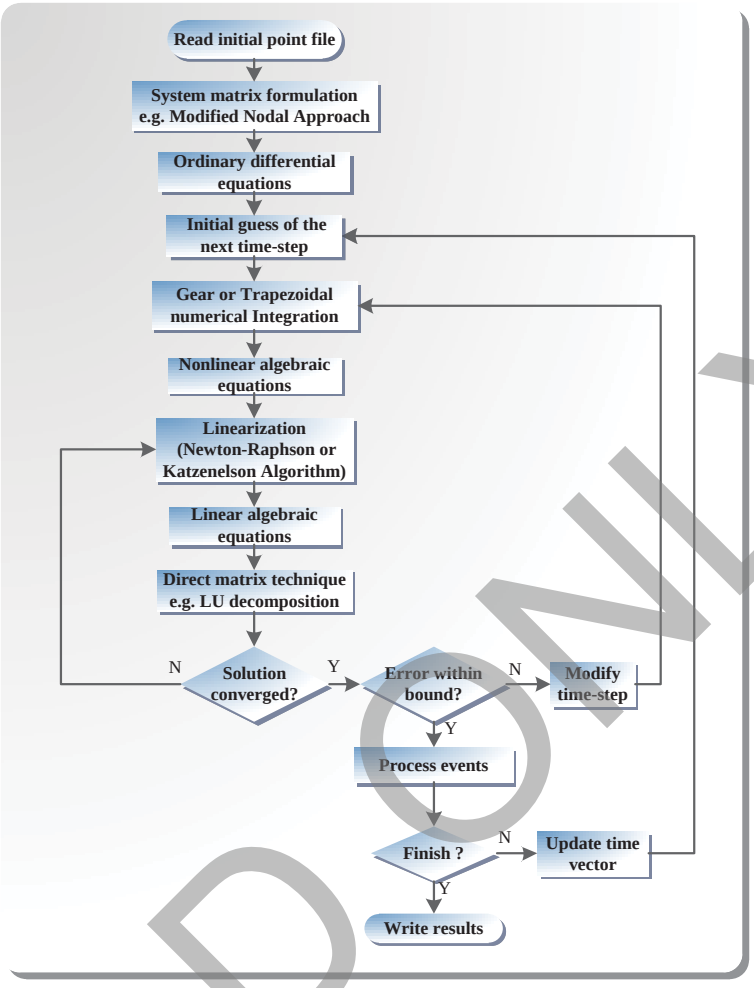

Figure 1. Flowchart of transient analysis operation [2].

3.1. Co-Simulation of Device-Level Circuit Simulators and System-Level Simulators

In this section, co-simulation examples for device-level circuit simulators are discussed.

3.1.1. Saber® and Matlab/Simulink®

Saber is a device-level circuit simulator which specializes in power electronic simulation, while Simulink is versatile in building control systems. Like other multi-domain designs, the co-simulation between Saber and Matlab/Simulink can be very effective in many circumstances. For instance, the Saber solution could calculate device losses which Simulink by itself may not be able to do. The procedure of using the Saber-Simulink co-simulation tool is discussed in [43]. The principle of Saber-Simulink interfacing is illustrated in Fig. 2.

Notably, the running processes of the two simulators are fully independent except when they need to exchange data at fixed period of intervals. This communication mechanism is realized by using an S-function. Additionally, a Saber Co-simulation block (SaberCosim.mdl) should be integrated into the Simulink model, which is then imported in to Saber interface using Saber-Simulink co-simulation tool. This tool is responsible for producing the required co-simulation interface symbol and the MAST template. This expressive user interface makes it possible for the co-simulation to run completely in the Saber interface.

Ref. [6] presented a method to construct a high-voltage source circuit system by the hybrid modeling of Saber and Simulink. In their designs, the Saber software was responsible for the power electronic circuit, while Simulink was in charge of building a fuzzy PID controller, because Saber is more specialized and powerful in switched-circuit analysis, while Matlab/Simulink is more powerful in control system design. The results showed that the co-simulation of Saber and Simulink was a highly efficient way to analyze and design a switching circuit system which included an intelligent control system.

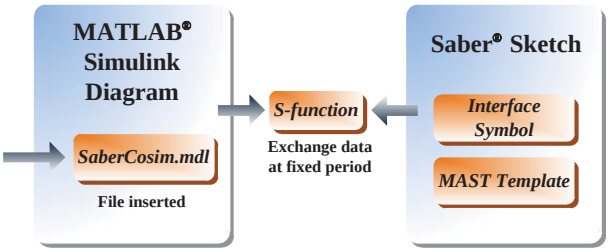

Figure 2. Illustration of Saber® Simulink® co-simulation interface [43].

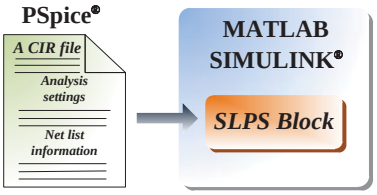

Figure 3. Illustration of PSpice® Simulink® co-simulation interface [47].

Similarly, [44] applied the Saber and Simulink co-simulation for the modeling of a high pulse power circuit. In their studies, the physical model of Reversely Switched Dynistor (RSD) was constructed using Simulink, and Saber was used to model the pulse power circuit and the magnetic switch. In fact, RSD physical model and circuit module can both be realized in M file form of Matlab and then the whole simulation can be finished just in only one software. But there are two disadvantages of this methodology, one is that a little change of the circuit structure leads to rewrite the code; the other is that the poor visualization of circuit structure. The Saber-Simulink Co-Simulation environment (SSCSE) allows the algorithm realized in Simulink and power circuit modeled in SABER, respectively, and the co-simulation method can overcome the two drawbacks above easily.

3.1.2. PSpice® and Matlab/Simulink®

PSpice is a simulation tool which can tackle models in analog and mixed-signal environments, whereas Matlab/Simulink, as a platform for multi-domain simulation, is mainly based on approximate continuous-time and discrete-time models of dynamic systems. Obviously, they both have advantages in a simulation project, since the former can make it possible for designers to perform simulation which includes realistic electrical models, such as the electrical circuits in switched reluctance motor in [45] and the zero voltage switch (ZVS) inverter in [46], while the latter is mainly focused on building the whole system.

The co-simulation between Matlab/Simulink and PSpice is realized by an interface tool named the PSpice SLPS Interface, which enables electro-mechanical system designers to perform system-level simulation which include specific device-level circuit simulation. More specifically, the PSpice SLPS Interface makes it possible for the designer to include realistic electrical PSpice models of actual components when performing system-level simulation. The detailed procedure to use the SLPS interface is available in [47] and the co-simulation is illustrated in Fig. 3.

Notably, the co-simulation of PSpice and Simulink is initialized by creating a CIR file, which specifies the PSpice analysis settings (e.g. the analysis time) and the net list information, thereby assigning the circuit built in PSpice to the SLPS block, which, in turn, is integrated into Simulink models [47].

The co-simulation is dominated by Simulink, which exchanges data with PSpice at its own time-step [48]. In general, the internal time steps of PSpice are much smaller than the Simulink time steps. If PSpice uses the minimum/maximum time step determined by Simulink, a convergence error will be likely to occur; that is, no result can be found. To solve this problem, SLPS calculates values suited to PSpice based on Simulink parameters, and then passing these to PSpice. Fig. 4 shows the mechanism clearly.

System-level simulation can be finished in a short time, but it can only verify the function of design and can not check the performance and non-idealities of the real circuits. Device level simulation can solve such problems but it usually needs a much longer simulation run time. So through the combination of the merits of the Simulink and PSpice, the simulation run time can be diminished greatly while the simulation precision can be well ensured and a satisfactory simulation result is generated. The PSpice SLPS Interface has been used in many electro-mechanical designs. These applications include the design presented in [5], which successfully simulates pipeline ADC circuits

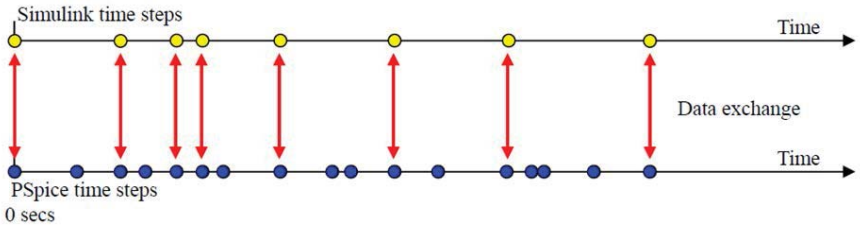

Figure 4. Data exchange mechanism of PSpice® and Simulink® [48]

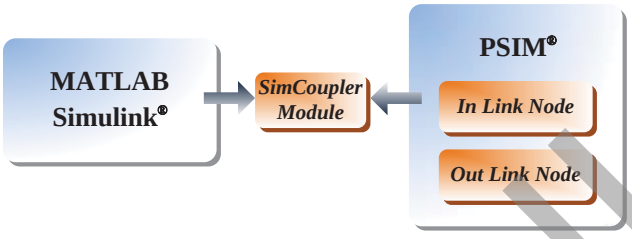

Figure 5. Illustration of PSIM® Simulink® co-simulation interface [51].

and obtains satisfactory results. Also, [49] presents a method for the simulation of solar photovoltaic (PV) cell by applying the PSpice and Simulink co-simulation interface; it builds a hybrid model of the PV module using PSpice and Simulink. Additionally, Ref. [50] presents a co-simulation solution for a high efficiency full-bridge DC-DC converter for fuel cell; because the Simulink has merits in building the feedback controller of the converter while PSpice excels in modeling the electronic circuits.

3.1.3. PSIM® and Matlab/Simulink®

Ref. [51] introduced the SimCoupler Module by which the co-simulation between PSIM and Matlab/Simulink was made possible and provided detailed procedures to use this module. The principle of PSIM-Simulink interfacing is illustrated in Fig. 5.

As a device-level circuit simulator which has special advantages in power electronics simulation [52, 53], PSIM has disadvantages to build a control subsystem. However, Matlab/Simulink is good at constructing control circuits. Therefore, the co-simulation between PSIM and Simulink has its unique value to be studied and applied in many circumstances. For example, [54] built a simulation platform for a three-level adjustable speed drive. In view of the merits of PSIM, the designer built the main circuit of the three-level adjustable speed drive using PSIM, whereas they constructed the control system, observed the output voltage and performed Fourier analysis in Matlab/Simulink.

Another case for the application of the co-simulation between PSIM and Simulink was presented in [55], where the designer first compared PSIM and Simulink separately in simulating single-phase uncontrolled rectifier and three-phase controlled rectifier. After discussing the pros and cons of both simulators, they concluded that the co-simulation between them was a better solution. The simulation results states that Matlab/Simulink is a suitable platform for control and regulation of the simulation processes, in additional to its dominant role in conducting research tasks. Conversely, PSIM is dedicated to power electronic circuits and machine simulation tasks with fast and robust algorithms. So the authors concluded that the co-simulation between them was a better solution.

3.1.4. SPICE and Matlab/Simulink®

Ref. [56] introduces a co-simulation interface between Simulink and SPICE, which is realized by a new Simulink block SLSP. The SLSP block is written in C MEX S-function, which is responsible for reading in the circuit file, initializing the simulation, performing the time-domain numerical integration and manipulating the SPICE-Simulink communication.

The co-simulation setting is simple to build up as an SLSP block in the Matlab/Simulink environment with a parameter for the name of a SPICE circuit file. This mechanism is described in Fig. 6.

Ref. [56] also elaborates an application to use Simulink-SPICE Interface to simulate a speed control system of a dc motor. Specifically, the whole system is modeled in the Simulink environment, except for the PI controller, the machine is modeled in SPICE.

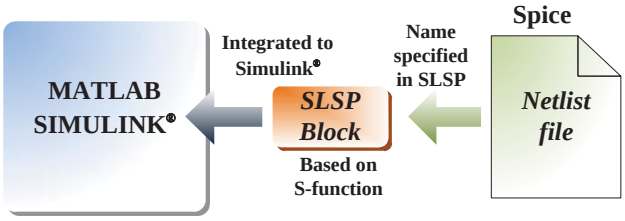

Figure 6. Simulink®-SPICE interface mechanism [56].

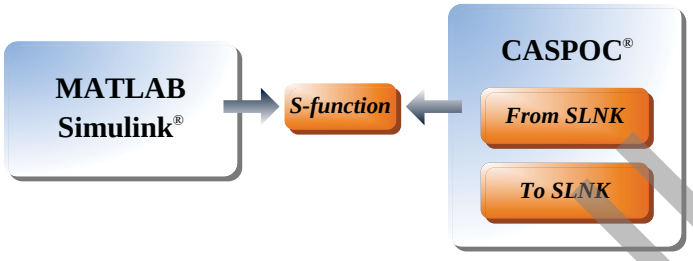

Figure 7. CASPOC®Simulink® co-simulation interface [35].

3.1.5. PLECS® and Matlab/Simulink®

PLECS, is a piece-wise linear electrical simulator that enters the circuit information as netlists, which are in turn integrated into Matlab/Simulink using S-functions. Compared to the Power System Blockset models of Simulink, the PLECS improves the performance greatly using the ideal switch models [26].

Ref. [57] applies Simulink and PLECS co-simulation in a photovoltaic energy conversion system. The control subsystem is modeled using Simulink, whilst the plant subsystem, including DC supply, inverter, LCL filter and utility grid, is modeled using PLECS. The simulation results show that the co-simulation between Simulink and PLECS is much faster than using Simulink transfer functions.

3.1.6. CASPOC® and Matlab/Simulink®

CASPOC is a circuit simulator dedicated to power electronics and electrical drives simulation [34]. Ref. [35] describes that simulations performed by coupling CASPOC and Matlab/Simulink offer the advantage of modeling of the control of Matlab/Simulink while connecting with CASPOC modeling the power electronics devices and circuits.

Fig. 7 illustrates CASPOC and Simulink co-simulation interface. The co-simulation is performed through data exchange between CASPOC and Simulink at every time step. Simulations in Simulink are controlled from CASPOC and the control then can be implemented in Simulink and co-simulated with a simulation in CASPOC. The co-simulation begins with creating two function blocks in CASPOC before starting the simulation in Simulink to communicate data. Then CASPOC is running in the background while data can be exchanged between the two simulators with the use of S-function block in Simulink. The S-function block represents the complete CASPOC simulation. It is required that the step-size in CASPOC and Simulink must be equal during the simulation.

3.2. Analog and Digital Co-Simulation of Circuit Simulators

Although co-simulation with some system-level simulators may be effective in handling large and complex simulation tasks, for those applications such as power and mechatronic system simulations, which may include a great many analog and digital subsystems, novel co-simulation platforms have become necessary [65]. For example, simulators such as Saber work in analog and mixed-signal environments, whereas many of the control subsystems are performed in digital logic. Therefore, co-simulations with other digital simulators such as ModelSim® become effective solutions [7]. In this section, several interfacing instances for analog and digital co-simulation of circuit simulators are discussed.

3.2.1. Saber® and ModelSim®

According to [7], the Saber/ModelSim co-simulation interface enables Saber, an analog and mixed-signal simulator, to support VHDL modeling. More specifically, a designer can model the analog part by using Saber MAST

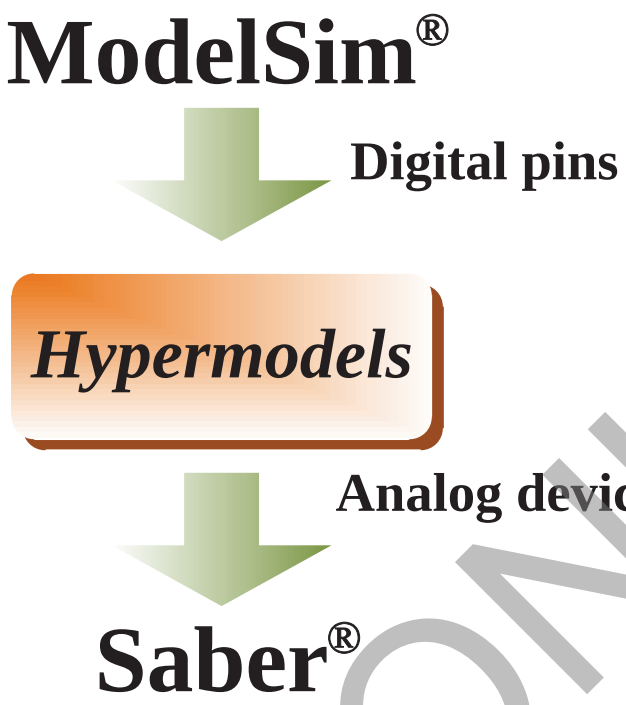

Figure 8. Saber®/ModelSim® and PSIM®/ModelSim® co-simulation interfaces [7] [58] [59].

and the digital part in VHDL using ModelSim VHDL or in VHDL/Verilog using ModelSim Plus®. The mechanism behind the Saber/ModelSim co-simulation is illustrated in Fig. 8.

The hypermodels shown in Fig. 8 that interface the digital pins from the models built in ModelSim with the analog device models in Saber are included in a library of more than 3,500 models, which guarantee the co-simulation to be accurate [7].

3.2.2. PSIM® and ModelSim®

According to [58] and [59], the ModCoupler-VHDL (or -Verilog) module provides a link for co-simulation between PSIM and ModelSim for VHDL or Verilog code support. With these modules, the power circuit can be implemented in PSIM, and the control circuit in VHDL code, which can then be simulated by ModelSim, for hardware implementation on FPGA. These modules provide a quick way to go from conceptual validation in PSIM to hardware implementation in a FPGA hardware. FPGAs are the preferred platform nowadays for real-time hardware emulation of power systems and power electronic systems due to the high degree of parallelism that can be exploited [60, 61, 62, 63]. The co-simulation between PSIM and Modelsim can be illustrated as in Fig. 8.

The co-simulation of PSIM and Modelsim has been applied to the modeling and analysis of DC/DC switching power converters and results show the validity of the co-simulation approach [64].

3.2.3. HSIMplus® and HDL

Ref. [65] introduces the co-simulation between HSIMplus and HDL, which connects digital and analog subsystems. Specifically, the HSIMplus supports interface to several different Verilog simulators such as Synopsys® VCS, Cadence® NC-Verilog, and Mentor Graphics® ModelSim. The co-simulation of HSIMplus and ModelSim can be described as in Fig. 9. The co-simulation can facilitate full integration of circuit information using different formats. Furthermore, the designers can decide which simulator is predominant for specific cases [65].

3.2.4. Multisim™and LabVIEW™

Ref. [66] introduces the co-simulation between Multisim and LabVIEW. The advantage of this co-simulation is that it integrates the analog and digital system in effectively, because Multisim is excellent in modeling analog and

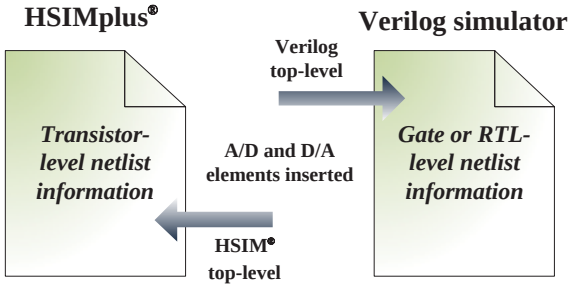

Figure 9. HSIMplus® HDL co-simulation interface mechanism [65].

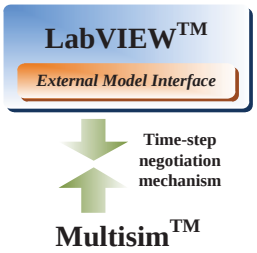

Figure 10. Multisim™LabVIEW™co-simulation interface [66].

mixed-signal circuitry whilst LabVIEW is prominent in implementing the control logic. With the new co-simulation capabilities between Multisim and LabVIEW, you can design an entire analog and digital system with accurate, closed-loop point-by-point simulation.

As illustrated in Fig. 10, the data exchange between the two simulators uses a variable time-step mechanism, which is decided by the conditions of both simulators. An External Model Interface in LabVIEW is responsible for all the data exchanged [66]. Also, [66] describes the detailed procedures on how to initiate and perform the co-simulation with an example.

3.3. Co-Simulation of Circuit Simulators with Programming Languages

The limitations of circuit simulators may sometimes require them to perform co-simulation with programming languages. For example, SystemC™(essentially C++) is a good option for multi-domain applications because of its advantages to undertake high levels of abstraction when modeling systems. As a programming language, it can act as a simulator as well as a connect and coordinate element between other simulators [8],[9]. A discussion of the co-simulation between circuit simulators and programming languages is of significance because traditional circuit simulators have several modeling limitations; custom built user-defined models can overcome such limitations.

3.3.1. PSpice® and C++

Although the co-simulation of PSpice and Simulink can be very effective to tackle many demanding designs, high level languages such as C++ sometimes can be applied to compile and store the external data for PSpice in order to simplify the modeling and reduce the iteration times.

For example, [67] applies the co-simulation technique of PSpice with C++ in the application of a three-phase PWM power converter. Specifically, the designers produce the PWM waveforms using C++ programming by interfacing the micro-controller with PSpice. The output data of the PWM pulses is stored in memory, which is then integrated into the PSpice listing program for the whole circuit. Finally, PSpice's graphic processor, named Probe, works as an oscilloscope to display the output waveforms.

3.3.2. SPICE and SystemC™

Ref. [8] introduces a SPICE and SystemC co-simulation method which is realized on the concept of loose simulator coupling. In practice, SystemC acts as the master simulator, keeping in charge of the whole simulation period, while SPICE acts as an integral part of SystemC, exchanging information with the master simulator in a loosely synchronized manner.

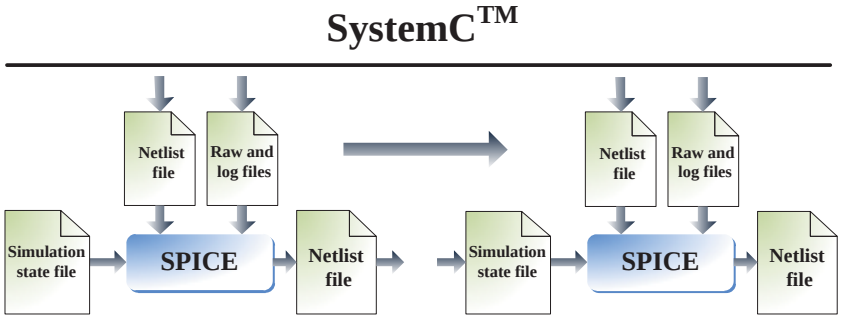

Figure 11. SPICE and SystemC™ co-simulation interface [8].

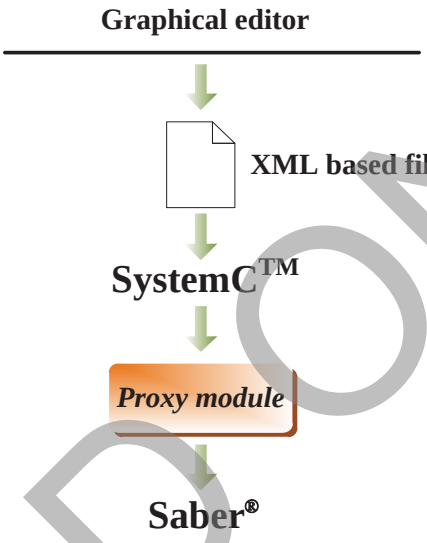

Figure 12. Saber® and SystemC™ co-simulation interface [9].

SystemC is active and dominant throughout the cycle of simulation, whilst SPICE, as a slave simulator, gets called by SystemC until exiting, when its current state is saved to a file that is used to resume the simulation when it is called next time. The mechanism of SPICE and SystemC co-simulation can be illustrated as in Fig. 11, where the information exchange is realized by several simulation source files. A specially designed wrapper program is responsible for the interfacing with SPICE, turning SystemC commands into the netlist file before sending to SPICE, and scanning the events and errors from SPICE.

3.3.3. Saber® and SystemC™

A successful example for the co-simulation of Saber and SystemC is mentioned in [9], where co-simulation is implemented by a proxy module responsible for transferring SystemC signals to Saber. A special synchronization method was applied to tackle the difference between analog and digital simulators. Furthermore, a graphical editor had been made it easy for the mixed signal design by Saber and SystemC [9]. The co-simulation interface of Saber and SystemC is shown in Fig. 12. Specifically, the graphical editor is capable of exporting the design to SystemC via a XML based file, which contains the structural information (e.g., components, interfaces and interconnections) [9].

4. CO-SIMULATION CASE STUDY, RESULTS AND DISCUSSION

4.1. PSIM®-Simulink®-FLUX® Co-simulation

In the section, a novel co-simulation model of PSIM-Simulink-FLUX was established and the schematic diagram of the model is illustrated in Fig. 13. The Simulink is in charge for the entire model, including the setting of

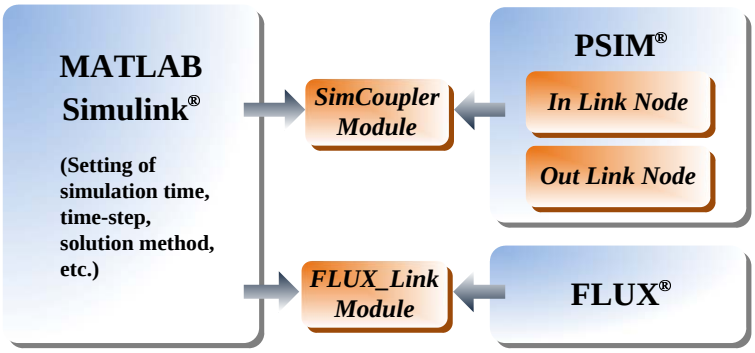

Figure 13. Schematic diagram of PSIM® -Simulink®-FLUX® co-simulation

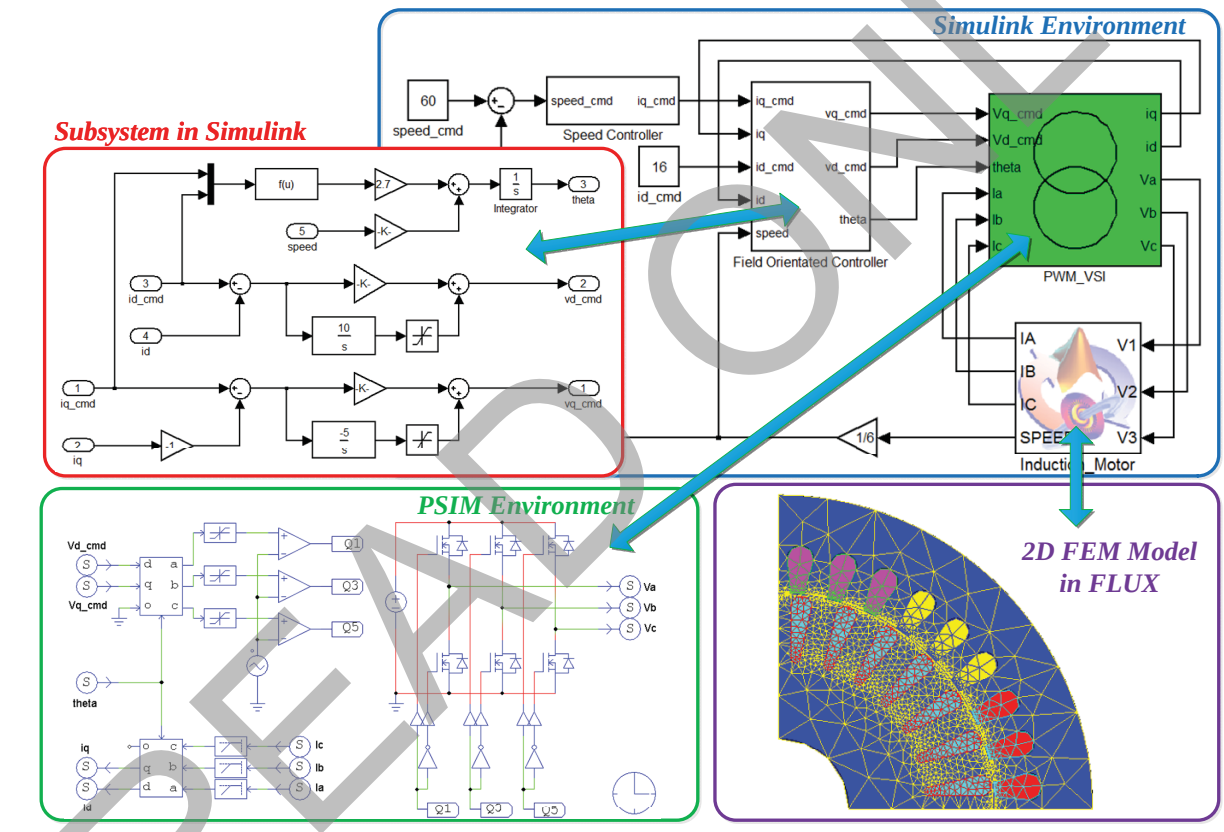

Figure 14. PSIM®-Simulink®-FLUX® co-simulation.

the simulation time, the time-step, the solution method, etc. The data exchange is fulfilled by the SimCouple module for PSIM-Simulink and the FLUX_Link module for FLUX-Simulink. The reason of choosing PSIM for the simulation rather than SimPower in Simulink is that the former is much faster and more accurate results can be achieved. It is worth mentioning that, in terms of the selection of the finite element software, FLUX and JMAG can both be the nice choice because they can both be conveniently connected to Simulink with the specific modules and the data exchange in the simulation is fast and stable. Compared with the above two FE software, Maxwell is not a good choice because the connection between Maxwell and Simulink is more complicated based on the script programming.

This section evaluates the performance of a field-oriented induction motor drive where the speed controller and field-oriented controller are implemented in Simulink; the transformations, PWM modulation, and the three-phase inverter are modeled in PSIM; and the induction motor and mechanical load are modeled in FLUX. This interfacing technique among different tools facilitated a detailed analysis of the system, and to obtain more specific characteristics

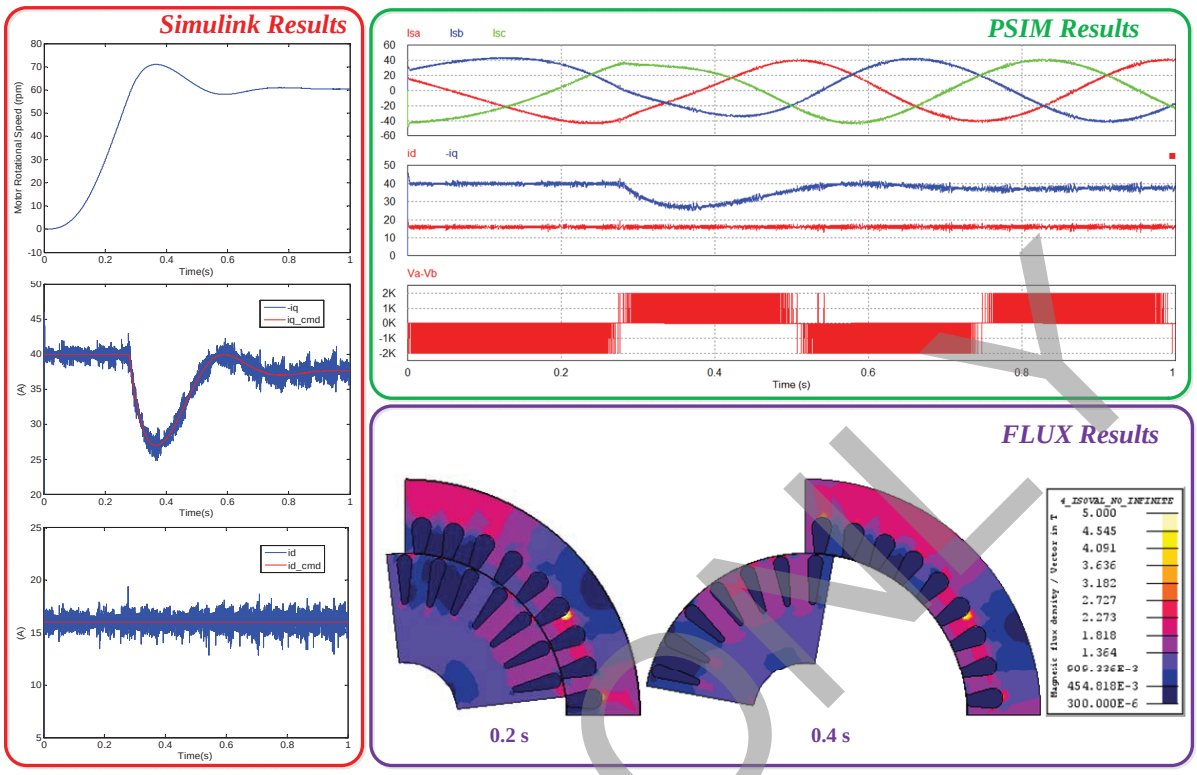

Figure 15. PSIM®-Simulink®-FLUX® co-simulation results.

Table 2. Model parameters and simulation setting

Model Parameters							
Inverter		Induction Motor					
DC Bus Voltage	Mosfet	Rs (Ω)	Ls (H)	Rr (Ω)	Lr (H)	No. of Poles	Moment of Inertia
2000V	Ideal	0.294	1.39e-3	0.156	7.4e-4	4	0.4 ($kg \cdot m^2$)
Setting of Simulation							
Simulation Time		Time-step		Solver in Simulink			
$1s$		Fixed-step : $10\mu s$		ode3 (Bogacki-Shampine)			
Computational Resource							
Computation Platform						Computation Time	
Dual-CPU Intel® Xeon® E5-2620 v2 @2.10GHz (12 cores)						∼70hours	

of the electromagnetic device, such as the magnetic flux density distribution of the motor. The whole co-simulation model is shown in Fig. 14. The specific parameters of the model and the setting of the simulation are shown in Table 2.

Simulink is the master simulator and controls the entire process. The links between PSIM-Simulink and FLUX-Simulink are responsible for exchanging data at a predetermined fixed period. In the simulation, the time-step is set to 10μs and the duration is set to 1s. Considering the symmetry of the induction motor, a 2D FEM model of a quarter of the machine was built in FLUX to speed-up the co-simulation. The FE model includes 7651 nodes and 3800 surface elements. Due to adopt self-adaptive mesh technique, the gap region between the stator and rotor can be meshed by very small elements to obtain highly accurate results and the minimum size of an element is less than 0.2mm.

4.2. Results and Discussion

In the simulation, the machine is subject to ramp up to the rotational speed of 60 rpm within 1s. Fig. 15 shows the co-simulation results, including the waveform of machine speed, phase voltages, q and d components of stator current, and the magnetic flux density distribution at certain moments. It is apparent that this co-simulation provides more information from inside of the machine than would be possible if a lumped machine model was used in Simulink.

By the co-simulation technique in the paper, we can get more details from the entire simulation, e.g. the distribution of magnetic flux density in the stator or rotor and which part of the motor is magnetically saturated, as shown in Fig. 15. We can also obtain the variation of flux density in the process. This cannot be achieved by the traditional method, i.e. co-simulation without FE model. The detail information from the FE model is very useful for the designer to improve the characteristic of the entire system. The effect of the change of size and material of inductor motor on the results can also be evaluated by this technique.

In addition, comparing to using only one tool alone, this co-simulation allowed the visualization of all results simultaneously using the Scope module in Simulink. The adjustment of the parameters in Simulink is more convenient than in PSIM, especially for running the control system. It can be interrupted and the parameters can be modified immediately, without waiting for the whole simulation to finish. This co-simulation made the model of the hierarchical system clearer and this is particularly important for building a detailed power electronic drive system, while making the analysis more flexible and providing a complete and detailed picture of the whole system.

In addition to the benefits of the approach, it should be noted that the co-simulation takes a relatively long time. From Table 2, we can find the FE model needs to be calculated 100,000 times because the time-step is set to be $10\mu s$; the actual execution time of the 2-D FE model is quite short– only 2∼3s. However, after 100,000 FE steps the total computation time is much longer than a circuit simulator such as PSIM. The advantage of this approach is that all the internal machine results are available for further analysis. Improving the co-simulation efficiency is a worthy future research goal.

5. CONCLUSION

This paper presented an overview of interfacing capabilities of device-level circuit simulation tools. A compendium of popular device-level circuit simulators are listed and compared with respect to their specialties. Many of them have been developed with versatile co-simulation interfaces to meet the higher demands from the simulation tasks today. More specifically, the researchers can now perform very large simulation tasks which can achieve desirable accuracy with satisfactory speed. At the same time, the transient information of specific part of the sub-circuits can also be collected and analyzed. At last, it is worth mentioning that some novel simulators have developed user-friendly GUIs, which enable the designers to handle the complicated simulation projects with more ease. While co-simulation still remains computationally onerous, with growing processor capabilities and opportunities for parallelism, this is not expected to remain obstacle for long. With growing emphasis on detailed simulation of multi-domain systems, this paper is provides a preliminary guideline in choosing the right simulation tool.

REFERENCES

[1] "*SPICE2: a computer program to simulate semiconductor circuits*," Electronics research laboratory report UCB/ERL M520, University of California, Berkeley, May 1975.
[2] "*Saber user guide*," Synopsys, Inc., USA, Sept. 2011.
[3] M. O. Faruque, V. Dinavahi, M. Steurer, A. Monti, K. Strunz, J. A. Martinez, G. W. Chang, J. Jatskevich, R. Iravani, and A. Davoudi, "Interfacing issues in multi-domain simulation tools," *IEEE Trans. on Power Del.*, vol. 27, no. 1, pp. 439-448, 2012.
[4] B. Asghari, V. Dinavahi, M. Rioual, J. A. Martinez, R. Iravani, "Interfacing Techniques for Electromagnetic Field and Circuit Simulation Programs," *IEEE Trans. on Power Del.*, vol. 24, no. 2, pp. 939-950, April 2009.
[5] Q. Guo, X. Jia, and H. Tang, "Co-simulation of pipeline ADC using Simulink and PSpice," *International Conference on Intelligent Computation Technology and Automation*, vol. 2, pp. 487-490, March 2011.
[6] J. Mo, S. He, and Y. Zou, "Fuzzy PID controller design for PWM-Buck EBW stabilized high-voltage source using Saber-Matlab co-simulation," *IEEE Int. Conf. Control Automation*, pp. 2003-2007, 2007.
[7] Analogy Introduces VHDL Co-Simulation Product,1998. Website: www.eetimes.com/ electronics-news/4149530/Analogy-Introduces-VHDL-Co-Simulation-Product
[8] T. Kirchner, N. Bannow, and C. Grimm, "Analogue mixed signal simulation using Spice and SystemC," *Design, Automation & Test Europe Conf. & Exhibition*, pp. 284-287, Apr. 2009.

[9] T. Kirchner, N. Bannow, C. Kerstan, and C. Grimm, "Mixed signal simulation with SystemC and Saber," *Forum Specification & Design Languages*, pp. 1-6, Sep. 2010.
[10] H. Mantooth and M. Vlach, "Beyond SPICE with Saber and MAST," *Proc. IEEE Int. Symp. Circuits Syst.*, vol. 1, pp. 77-80, May 1992.
[11] K. G. Nichols, T. J. Kazmierski, M. Zwolinski, and A. D. Brown, "Overview of SPICE-like circuit simulation algorithms," *IEE Proc. Circuits, Devices Syst.*, vol. 141, no. 4, pp. 242-250, Aug. 1994.
[12] "*SPICE/MultiSim Tutorial*," EECS Department, UC Berkeley, 2007.
[13] L. Trajkovic, E. fung, and S. Sanders, "HomSPICE: simulator with homotopy algorithms for finding dc and steady-state solutions of nonlinear circuits," *Proc. IEEE Int. Symp. Circuits Syst.*, vol. 6, pp. 227-231, 1998.
[14] I. M. Wilson, "Analog behavioral modeling using PSpice," *Proceedings of the 32rd Midwest Symposium on Circuits and Systems*, Aug. 1989.
[15] W. L. Engl, R. Laur, and H. K. Dirks, "MEDUSA-a simulator for modular circuits," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Syst.*, vol. 1, no. 2, pp. 85-93, Apr. 1982.
[16] K. Mayaram and D. O. Pederson, "CODECS: a mixed-level device and circuit simulator," *Proc. Int. Conf. Computer-Aided Design*, Santa Clara, CA, pp. 112-115, Nov. 1988.
[17] JSPICE, 2008. Website: www.eecs.berkeley.edu /IPRO/Software/Description/jspice.html
[18] MacSpice User's Guide. Website: www.macspice.com/ug/
[19] "*XSpice Software User's Manual*," Georgia Tech Research Corporation, USA, Dec. 1992.
[20] "*NGSPICE User Manual*," University of California, USA, 1996.
[21] Qucs project. Website: qucs.sourceforge.net/
[22] HSPICE, 1999. www.stanford.edu/class/ee133/spice/HSpice.pdf
[23] Introduction to OrCAD Capture and PSpice, Sep 2008. Website: user-web.eng.gla.ac.uk/john.davies/orcad/spiceintro160.pdf
[24] C. K. Chuang and C. G. Harrison, "Analogue behavioural modelling and simulation using VHDL and Saber-MAST," *Proc. IEE Colloq. Mixed Mode Modelling and Simul.*, pp. 1-5, Nov. 1994.
[25] "*PSIM user guide*," Powersim Inc., Jun. 2003.
[26] J. H. Allmeling and W. P. Hammer, "PLECS-piece-wise linear electrical circuit simulation for Simulink," *Proc. IEEE Int. Conf. Power Electron. Drive Syst.*, vol. 1, pp. 355-360, 1999.
[27] Hierarchical Full-chip Circuit Simulation and Analysis, 2012. Website: www.synopsys.com /Tools/Verification/AMSVerification/CircuitSimulation/HSIM/Pages/default.aspx
[28] A. K. Palit, K. K. Duganapalli, and W. Anheier, "XSIM: an effecient crosstalk simulator for analysis and modeling of signal integrity faults in both defective and defect-free interconnects," *IEEE Design Diagnostics Electro. Circuit Syst.*, pp. 1-4, Apr. 2007.
[29] Z. Cheng, "The application of Multisim in power electronic technology," *Int. Conf. Multimedia Technol.*, pp. 813-816, Jul. 2011.
[30] D. Li, R. Tymerski, and T. Ninomiya, "PECS-power electronics circuit simulator," *7th Workshop Computer Power Electron.*, pp. 159-165, 2000.
[31] U. Feldmann and R. Schultz, "TITAN: an universal circuit simulator with event control for latency exploitation," *4th European Solid-State Circuits Conf.*, pp. 183-185, Sep. 1988.
[32] P. Pejovic and D. Maksimovic, "PETS-a simulation tool for power electronics," *IEEE Workshop on Computer in Power Electron.*, Aug. 1996.
[33] "*Spectre circuit simulator reference*," Cadence Design Systems, 2003.
[34] P. J. van Duijsen, P. Bauer, and B. Davat, "Simulation and animation of power electronics and drives, requirements for education," 2006, www.simulation-research.com.
[35] CASPOC from Scratch - An Engineers Workbook. Website: www.simulation-research.com /support/howto.pdf
[36] R. J. Trihy and R. A. Rohrer, "A switched capacitor circuit simulator: AWEswit," *IEEE J. Solid-State Circuits,*, vol. 29, pp. 217-225, 1994.
[37] F. Ye and J. Cheng, "A multimedia software for the analysis and design of circuits and electronics," *Proc. Int. Symp. multimedia software eng.*, Taipei, pp. 378-381, Dec. 2000.
[38] "*Eldo users manual*," Mentor Graphics Corporation, USA, 2005.
[39] "*IsSPICE4 users guide*," intusoft, USA, 2007.
[40] "*Gnucap the gnu circuit analysis package users manual*," Albert Davis, Sep. 2006.
[41] "*SwitcherCAD III/LTspice getting started guide*," Linear Tech., 2008.
[42] "*Getting started with TINA-TI*," Texas Instruments, Inc., USA, 2008.
[43] "*Saber Simulink co-simulation interface user guide*," Synopsys, Inc., USA, Sept. 2004.
[44] Y. Peng, Y. Yu, L. Liang, C. Shang, J. Liu, and Y. Wang, "Simulation of high pulse power circuit based on

semiconductor switch RSD in SABER-Simulink co-simulation environment and experiment," *Int. Conf. Electrical Machines Syst.*, pp. 3840-3844, Oct. 2008.
[45] R. M. Autee, "Dynamic Testing & Simulation of 4 KW (5.5 Hp) Switched Reluctance Motor using PSpice," *International Journal of Power Electronics and Drive System (IJPEDS)*, vol. 1, no. 1, pp. 36-40, Sep. 2011.
[46] X. Y. Wang and Z. Yang, "Simulation of ZVS Converter and Analysis of Its Switching Loss Based on PSPICE," *International Journal of Power Electronics and Drive System (IJPEDS)*, vol. 2, no. 1, pp. 19-24, Mar. 2012.
[47] "*SLPS user guide*," Cadence Design System, USA, 2007.
[48] "*PSpice SLPS Interface User's Guide Version 2.5*," Cybernet Systems Co., Ltd., Oct. 2004.
[49] Y. Jiang, J. A. A. Qahouq, and M. Orabi, "MATLAB/Pspice hybrid simulation modeling of solar PV cell/module," *26th Annu. IEEE Applied Power Electron. Conf. Exposition*, pp. 1244-1250, March 2011.
[50] O. A. Ahmed and J. A. M. Bleijs, "Pspice and Simulink co-simulation for high efficiency dc-dc converter using SLPS interface software," *5th IET Int. Conf. Power Electron. Machines Drives*, pp. 1-6, Apr. 2010.
[51] "*Tutorial on how to use the SimCoupler Module*," Powersim Inc., 2009.
[52] O. H. Abdalla and M. Han, "Power Electronics Converters for Variable Speed Pump Storage," *International Journal of Power Electronics and Drive System (IJPEDS)*, vol. 3, no. 1, pp. 74-82, Mar. 2013.
[53] Y. Cho, "A Low Cost Single-Switch Bridgeless Boost PFC Converter," *International Journal of Power Electronics and Drive System (IJPEDS)*, vol. 4, no. 2, pp. 256-264, Jun. 2014.
[54] Y. Zhang, Z. Zhao, Baihua, L. Yuan, and H. Zhang, "Pspice and Simulink co-simulation for high efficiency dc-dc converter using SLPS interface software," *CES/IEEE 5th Int. Power Electron. Motion Control Conf.*, vol. 1, pp. 1-5, Aug. 2006.
[55] S. Khader, A. Hadad, and A. A. Abu-aisheh, "The application of PSIM & MATLAB/SIMULINK in power electronics courses," *IEEE Global Eng. Education Conf.*, pp. 118-121, Apr. 2011.
[56] "*Simulink-SPICE-interface*," BAUSCH-GALL GmbH, Germany, 2010.
[57] M. Ciobotaru, T. Kerekes, R. Teodorescu, and A. Bouscayrol, "PV inverter simulation using MATLAB/Simulink graphical environment and PLECS blockset," *32nd Annual Conf. IEEE Industrial Electron.*, pp. 5313-5318, Nov. 2006.
[58] "*ModCoupler-VHDL User's Guide*," Powersim Inc., May 2011.
[59] "*ModCoupler-Verilog User's Guide*," Powersim Inc., Oct. 2012.
[60] N. R. Tavana, V. Dinavahi, "A General Framework for FPGA-Based Real-Time Emulation of Electrical Machines for HIL Applications," *IEEE Trans. on Industrial Electronics*, vol. 62, no. 4, pp. 2041-2053, April 2015.
[61] W. Wang, Z. Shen, V. Dinavahi, "Physics-Based Device-Level Power Electronic Circuit Hardware Emulation on FPGA," *IEEE Trans. on Industrial Informatics*, vol. 10, no. 4, pp. 2166-2179, Nov. 2014.
[62] Y. Chen, V. Dinavahi, "Hardware Emulation Building Blocks for Real-Time Simulation of Large-Scale Power Grids," *IEEE Trans. on Industrial Informatics*, vol. 10, no. 1, pp. 373-381, Feb. 2014.
[63] Y. Chen, V. Dinavahi, "An Iterative Real-Time Nonlinear Electromagnetic Transient Solver on FPGA," *IEEE Trans. on Industrial Electronics*, vol. 58, no. 6, pp. 2547-2555, June 2011.
[64] P. Zumel, M. Garcła-Valderas, A. M. Roldan, C. Lopez-Ongil, and A. Barrado, "Co-Simulation Psim-Modelsim Oriented to Digitally Controlled Switching Power Converters," *12th IEEE Workshop on in Control and Modeling for Power Electronics*, pp. 1-7, Jun. 2010.
[65] HSIMplus HDL Co-Simulation, 2012. Website: www.synopsys.com/Tools/Verification/AMSVerification/DesignAnalysis/Pages/HDLCo-Simulation.aspx
[66] Introduction to Digital and Analog Co-simulation Between NI LabVIEW and NI Multisim, 2012. Website: www.ni.com/white-paper/13663/en
[67] S. Mekhilef, N. A. Rahim, and Z. A. Karim, "Analysis of different type of PWM for three phase power converter," *Proc. TENCON*, vol. 1, pp. 414-418, 2000.