The Qucs/QucsStudio and Qucs-S Graphical User Interface: An Evolving ”White-Board” for Compact Device Modeling and Circuit Simulation in the Current Era

Mike Brinson
Centre for Communications Technology
London Metropolitan University
UK
Email: mbrin72043@yahoo.co.uk

***Abstract*—The Qucs/QucsStudio and Qucs-S simulators share a common graphical user interface which has slowly evolved into an interactive platform for drawing circuit schematics, controlling simulation and displaying simulation output data and measured device/circuit parameters/properties. This interface acts as a window for accessing circuit simulation software and is in many ways similar to the ”White-Boards” that are popular among scientists and engineers for recording ideas when ”brainstorming” circuit design or analysis problems. This paper outlines the evolution of the Qucs device modeling and simulation ”white-Board” from concept to working media over the fifteen year period that Qucs, QucsStudio and Qucs-S have been under development. The operation of a number of the ”White-Board” features are introduced with a compact tunnel diode model and the simulation data obtained from tests using the QucsStudio and Qucs-S software packages.**

***Index Terms*—Qucs/QucsStudio, Qucs-S, compact device modeling, Verilog-A, Graphical User Interface, data visualization, ”White-Board” display,**

I. Introduction

A. Background

It is over fifty years since the industrial standard SPICE 2g6 [1] and 3f5 simulators [2] were first released as tools for integrated circuit design. Originally, these were developed as applications for main frame computers. Today the high performance Personal Computer (PC) has become the work horse for compact modeling and circuit simulation, which in turn has encouraged the development of a range of new commercial and Free Open Source Software (FOSS) circuit simulators similar to, or derived from, the Berkeley SPICE FORTRAN (SPICE 2g6) or C (SPICE 3f5) code. The move from centralized main frame computers to individual PC work station has had a profound effect on the analysis/design capabilities and simulation output data processing tools available to the compact modeling and circuit simulation communities. This paper outlines the evolution, over a period of roughly fifteen years, of the ”Quite universal circuit simulator” (Qucs) modeling and simulation facilities [3], placing particular emphasis on the evolution from ”SPICE text-in netlist input and simulation text-out output data” to highly interactive PC ”White-Board” controlled compact device modeling, circuit schematic drawing, circuit simulation and output data visualization. Illustrated in Fig. 1 is a block diagram that shows pictorially the links between Qucs and the ”forked” QucsStudio [4] and Qucs-S [5] circuit simulators. In this figure the vertical arrows signify modeling and simulation information flow, culminating in entries displayed on a PC Graphical User Interface (GUI) ”White-Board” window. Other GUI background information also indicates additional features, like for example where SPICE netlists apply. Throughout this paper a series of compact device modeling, circuit design/simulation and data visualization examples based on a tunnel diode compact model are presented. These have been chosen to demonstrate the QucsStudio and Qucs-S PC ”White-Board” features that exemplify, without complex detail, the application of this innovative approach to compact modeling and circuit simulation.

B. The Qucs Graphical User Interface (GUI)

The Qucs project was started by German engineer Michael Margraf [3] as a universal simulation tool with a Graphical User Interface (GUI) for drawing Qucs/Qucs-S and QucsStudio schematics, developing compact device models with Equation-Defined Devices (EDD) and Verilog-A module synthesis, launching multi-engine circuit simulation, undertaking parameter sweeps and device/component optimization, plus post-simulation data processing and visualization. The GUI acts as a wrapper for circuit schematic entry and post simulation data processing. It also gives access to a color highlighted text editor, 2D and 3D graph plotting, the Octave numerical analysis package [6] (for advanced output data processing and visualization), plus a group of drop down menus for launching simulations and undertaking other modeling and design tasks. The GUI is the foundation for a powerful user platform that functions as a ”White-Board” on a high resolution PC display window, allowing users to freely experiment with compact modeling, circuit simulation and output data analysis on a single graphics screen. The PC ”White-Board” is interpreted

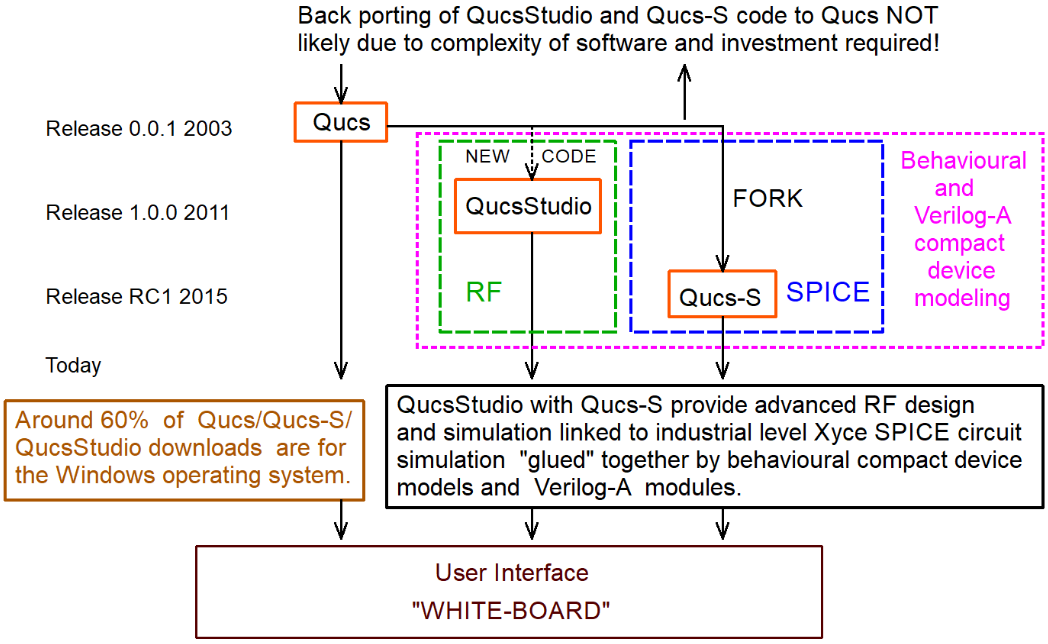

Back porting of QucsStudio and Qucs-S code to Qucs NOT likely due to complexity of software and investment required!
Release 0.0.1 2003
Release 1.0.0 2011
Release RC1 2015
Today
Qucs
NEW
CODE
QucsStudio
RF
FORK
Qucs-S
SPICE
Behavioural and Verilog-A compact device modeling
Around 60% of Qucs/Qucs-S/ QucsStudio downloads are for the Windows operating system.
QucsStudio with Qucs-S provide advanced RF design and simulation linked to industrial level Xyce SPICE circuit simulation "glued" together by behavioural compact device models and Verilog-A modules.
User Interface
"WHITE-BOARD"

Fig. 1. A block diagram illustrating links between Qucs, QucsStudio, Qucs-S and a common GUI ”White-Board”.

by the GUI software and its use is largely limited by a users imagination.

C. The relationship between Qucs, Qucs-S and QucsStudio

The QucsStudio and Qucs-S versions of the current Qucs compact modeling and circuit simulation software both derive from recent releases of the Qucs package. However, since their original release there has been inevitable divergence between the capabilities of the three software packages. This is largely the result of their target use; Qucs and QucsStudio are aimed squarely at RF circuit analysis and design, while Qucs-S links the Qucs GUI to different versions of SPICE. By combining the different facilities provided by individual packages a more versatile group of design and analysis tools has resulted. The diagram in Fig. 2 shows the relationship between QucsStudio and Qucs-S, highlighting the use of standardized simulator independent Verilog-A module code [7] as a model interchange vehicle between the two packages, and indeed other circuit simulators, effectively minimizing the effects of differences in model netlist formats.

II. Integrated Analysis and Design Blocks in Simulation Schematics

Subcircuits are essential building blocks when sectioning parts of a complex circuit schematic into smaller manageable units, such as circuit blocks that represent specific electronic functions or integrated circuits. Unfortunately, in the original SPICE 2g6 and 3f5 netlist format subcircuits were defined without the ability to pass parameters via an argument list. Later commercial and FOSS simulators corrected this omission, allowing both subcircuit parameters and, in the case of the Qucs series of simulators, blocks of algebraic and numeric equations for calculating component values from physical parameter values coupled to circuit design routines [8]. Similar capabilities are also available for circuit macromodels. Fig.3 shows a typical device model with component values derived from subcircuit parameters calculated by a Qucs style Equation block [9]. Each simulation schematic drawn on the PC ”White-Board” window is allowed one or more Equation blocks at the highest hierarchy design level, where the left hand equation variables may only be defined once across the blocks. Equation blocks are also allowed within subcircuits. With Qucs and QucsStudio ordering of the equations in a set of Equation blocks is not important. With Qucs-S however, when simulating with a SPICE engine, it is. The diagram sections labeled (a), (b) and (c) displayed in Fig. 3, are examples of fundamental items that can be combined with other component symbols to form a model schematic and placed on a Qucs style ”White-Board”. Allowed items include, simulation icons, data tables, data plots, text blocks, library models, predefined passive and active component symbols and subcircuit symbols.

III. The Qucs Equation Defined-Device

At the center of the Qucs/Qucs-S and QucsStudio compact device modeling capabilities is a new highly innovative component called an Equation-Defined Device (EDD) [10] [11]. The primary task of this multi-terminal device is to define a static or dynamic nonlinear component that represents, at electrical level, a physical process, where the nonlinear component is built from controlled current generators who’s properties are expressed as explicit algebraic functions of one or more EDD component terminal voltages, $I(Vn)$, or a differential time dependent function of branch stored charge $I(Q(Vn, In))$.

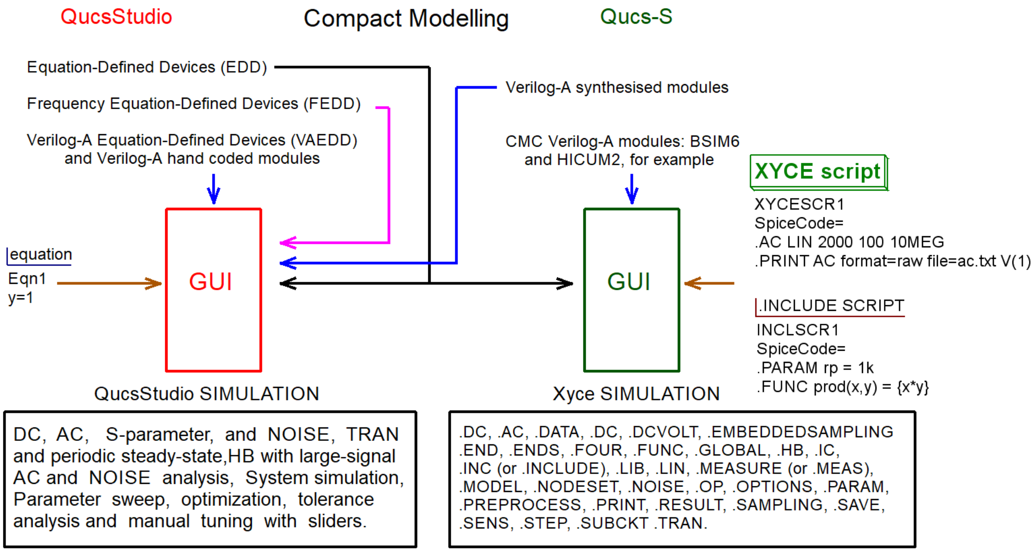

QucsStudio
Compact Modelling
Qucs-S
Equation-Defined Devices (EDD)
Frequency Equation-Defined Devices (FEDD)
Verilog-A Equation-Defined Devices (VAEDD)
and Verilog-A hand coded modules
Verilog-A synthesised modules
CMC Verilog-A modules: BSIM6
and HICUM2, for example
XYCE script
XYCESCR1
SpiceCode=
.AC LIN 2000 100 10MEG
.PRINT AC format=raw file=ac.txt V(1)
equation
Eqn1
y=1
GUI
GUI
.INCLUDE SCRIPT
INCLSCR1
SpiceCode=
.PARAM rp = 1k
.FUNC prod(x,y) = {x*y}
QucsStudio SIMULATION
Xyce SIMULATION
DC, AC, S-parameter, and NOISE, TRAN and periodic steady-state,HB with large-signal AC and NOISE analysis, System simulation, Parameter sweep, optimization, tolerance analysis and manual tuning with sliders.
.DC, .AC, .DATA, .DC, .DCVOLT, .EMBEDDEDSAMPLING .END, .ENDS, .FOUR, .FUNC, .GLOBAL, .HB, .IC, .INC (or .INCLUDE), .LIB, .LIN, .MEASURE (or .MEAS), .MODEL, .NODESET, .NOISE, .OP, .OPTIONS, .PARAM, .PREPROCESS, .PRINT, .RESULT, .SAMPLING, .SAVE, .SENS, .STEP, .SUBCKT .TRAN.

Fig. 2. A compact device modeling and simulation tool set derived from QucsStudio and Qucs-S.

Unfortunately, both SPICE 2g6 and 3f5 are only equipped with limited capabilities of this type (polynomial current source in SPICE 2g6 and B type controlled current source in SPICE 3f5), making dynamic current, $I = d/dt(Q(Vn, In))$, particularly difficult to model.

A. Explicit Equation-Defined Device (EDD) models

The component drawn in Fig. 4 represents a generalized EDD. Qucs and QucsStudio have a maximum of eight two terminal ports per EDD. Qucs-S has this number increased to twenty. All three packages allow more than one EDD per PC ”White-Board” schematic. This new nonlinear component allows interpretive modeling of both static and dynamic device properties, where terminal current In can be an algebraic function of branch voltages Vn plus a dynamic current component expressed as $d/dt(Qn(Vn, In))$, where $Qn(Vn, In)$ is the charge associated with EDD branch n. The explicit equations listed in Fig. 4 give a more complete specification. EDD can be combined with conventional component models, subcircuits, and $Equation\ \ Eqn$ blocks to construct compact device subcircuits or circuit macromodels. EDD is an advanced component that allows users to build prototype nonlinear compact device models, based on sets of physical properties defined as algebraic equations, attached to a schematic, and placed on a PC ”White-Board”. The d/dt operator is automatically evaluated by the circuit simulation software. Fig. 5 illustrates an example of this powerful interactive form of compact modeling applied to a tunnel diode. The tunnel diode static Id/Vd EDD equations are

$$I1 = I_p \cdot exp\left(\frac{-V_{pp}}{V1}\right) \cdot \left(exp\left(\frac{V1}{VTH}\right) - 1.0\right) \tag{1}$$

$$I2 = I_p \cdot \frac{V1}{Vp} \cdot exp\left(1.0 - \frac{V1}{Vp}\right) \tag{2}$$

$$I3 = I_p \cdot exp(V1 - V_v) \tag{3}$$

$$Id = I1 + I2 + I3 \tag{4}$$

where I_p and Vp are the diode peak current and voltage, Iv and Vv are the diode valley current and voltage respectively, V_{pp} is the protected peak voltage and VTH is the thermal voltage at $TempK$ Kelvin. The tunnel diode test bench in Fig. 5 shows the minimum set of items common to most PC ”White-Boards”, namely a test circuit (including components with lists of parameters), simulation Icons ($dc\ \ simulation$ and $Parameter\ \ sweep$ in the tunnel diode example), and output data (a d.c. 2D plot of $Itd(A)$ against $Vtd(V)$). The order and placement position of these items on a PC ”White-Board” is quite arbitrary. Notice also that the charge associated with a fixed capacitor Cp, defined as $Cp * \_v1$ and stored in branch one of EDD X1 Fig. 3, is given in (a), where $\_v1$ is the QucsStudio format for the voltage across branch one.

B. Debugging compact device models and circuits during development and simulation

During the development of complex compact device models, or simulation test circuits, it is likely that errors in a schematic drawing, parameter list or an equation block will occur. Finding and debugging such errors can often be difficult and indeed very time consuming, particularly in those case where a PC ”White-Board” includes many different elements. To minimize such problems complex compact modeling and simulation tasks are normally split into multiple self-contained sections and tested, when possible, separately. The use of subcircuits is particularly helpful when finding and eliminating bugs. The Qucs, QucsStudio and Qucs-S GUI have an additional aid for debugging problems during the development process. Fig. 6 shows the body of the tunnel diode model previously drawn in Fig. 3. In this diagram component Rp has a red cross

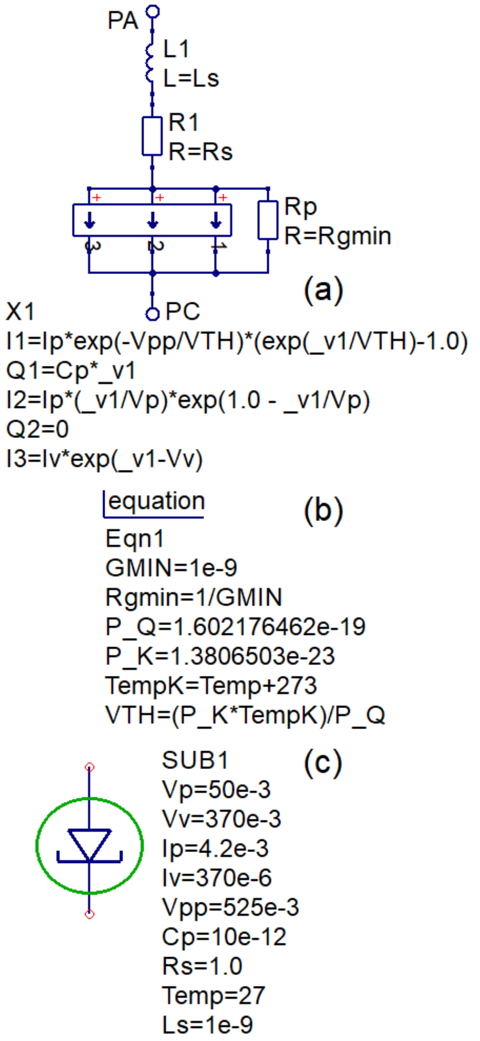


Fig. 3. Qucs-S subcircuit model of a tunnel diode: (a) subcircuit body, (b) design equation block, and (c) subcircuit schematic symbol with attached parameter list.

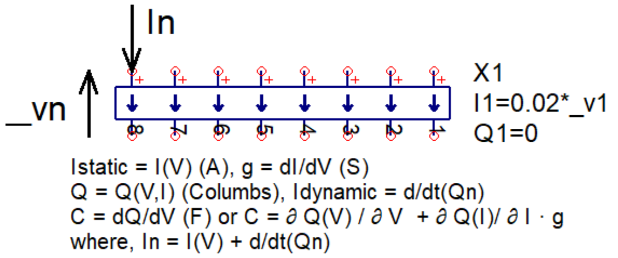

In
_vn
X1
I1=0.02*_v1
Q1=0
Istatic = I(V) (A), g = dI/dV (S)
Q = Q(V,I) (Columbs), Idynamic = d/dt(Qn)
C = dQ/dV (F) or C = ∂ Q(V) / ∂ V + ∂ Q(I)/ ∂ I · g
where, In = I(V) + d/dt(Qn)

Fig. 4. Specification of Qucs-S and QucsStudio EDD models with explicit equations.

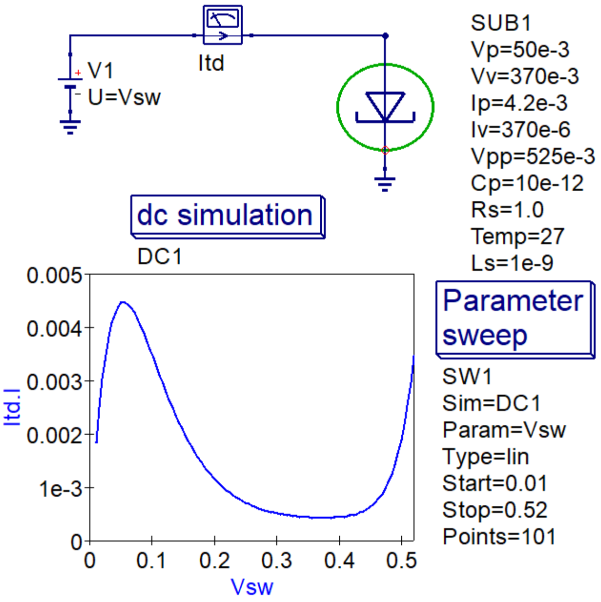


Fig. 5. Tunnel diode d.c. test bench and $Itd(A)$ plotted against $Vtd = Vsw$.

superimposed on it's symbol and component Ls has a green cross superimposed on it's symbol, indicating open circuit (red) and short circuit (green) respectively. Hence, it becomes possible to build and test subcircuits by removing single or groups of components from a compact device model. The short circuit and open circuit features are available at all hierarchical levels of a schematic placed on a PC "White-Board". They can also be used to remove complete subcircuits, icons or other valid items. Removing, the red and green crosses reconnects components in a device model or circuit. However, remember this feature is an aid for finding and eliminate drawing and textual syntax errors, having little or no direct effect when tracking down model physical property errors or omissions.

C. The Qucs Radio Frequency Equation-Defined Device (FEDD) and equivalent QucsStudio component

In some compact modeling instances the physical attributes of a device can be a nonlinear function of a.c. signal frequency

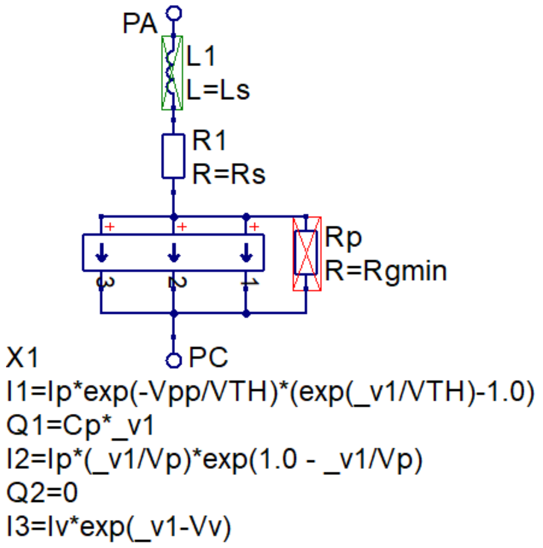
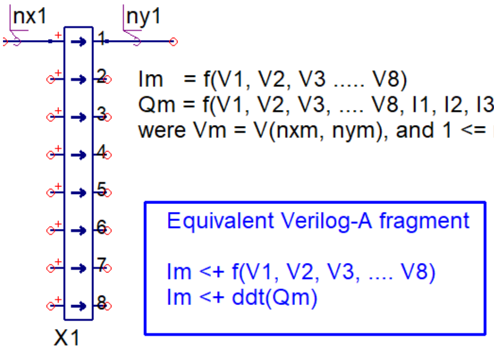

PA
L1
L=Ls
R1
R=Rs
Rp
R=Rgmin
X1
PC
I1=Ip*exp(-Vpp/VTH)*(exp(_v1/VTH)-1.0)
Q1=Cp*_v1
I2=Ip*(_v1/Vp)*exp(1.0 - _v1/Vp)
Q2=0
I3=Iv*exp(_v1-Vv)

Fig. 6. The tunnel diode compact device model showing Ls shorted (green crossed box) and Rp open circuit (red crossed box) for debugging purposes.



```
Im   = f(V1, V2, V3 ..... V8)
Qm = f(V1, V2, V3, .... V8, I1, I2, I3, ....I8),
were Vm = V(nxm, nym), and 1 <= m <= 8
```

```
Equivalent Verilog-A fragment

Im <+ f(V1, V2, V3, .... V8)
Im <+ ddt(Qm)
```

Fig. 7. An explicit EDD with eight two terminal ports and equivalent Verilog-A code fragment.

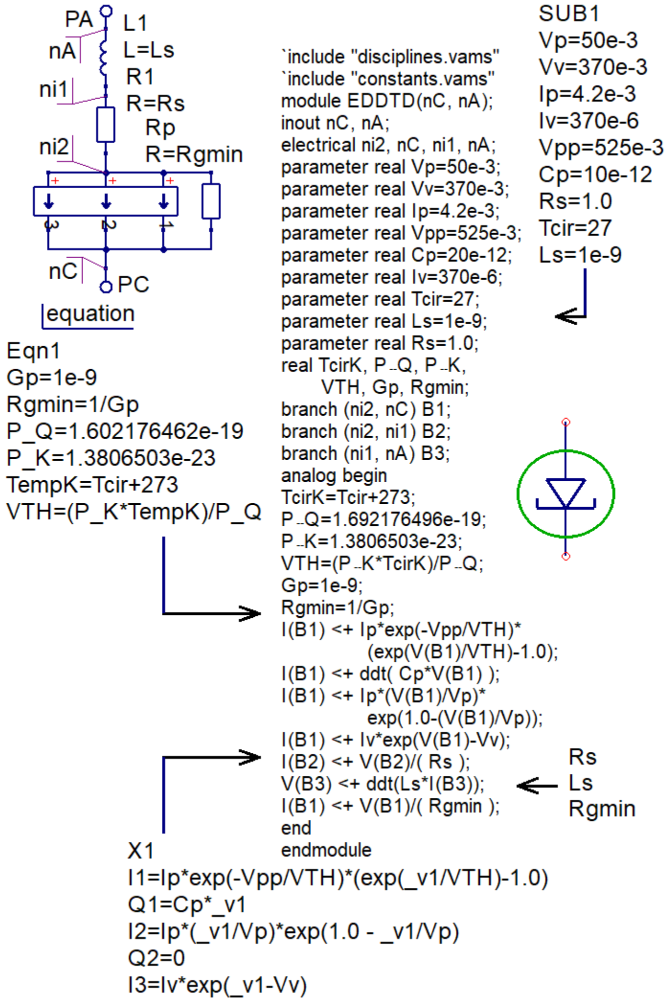
f where a model element is expressed as an algebraic function of f. To handle this form of modeling an extended form of the EDD has been implemented in Qucs, called FEDD, and as a ”Frequency Domain” component in QucsStudio. These devices add frequency domain two port and multi-port nonlinear modeling to the Qucs group of simulators, including S, Y, Z, H, G, A, T and VCVS types. Examples of the use of these elements can be found in reference [12].

IV. Compact modeling with Verilog-A

The previously described process for constructing compact device models with the built in components is essentially one of placing all the required elements on a PC ”White-Board”, simulating and making any required changes in an interactive fashion. Such an approach provides an ideal environment for prototyping and testing device models and circuits. Unfortunately, compact device models or circuits that include significant numbers of EDD and FEDD tend to simulate slowly. Moreover, their simulation performance also deteriorates as the number of nonlinear components increases. To overcome this limitation, and indeed other factors, the compact modeling community has adopted the Verilog-A [14] analog hardware description language for defining and constructing C++ machine code models. Qucs and QucsStudio employ the Analog Device Model Synthesizer (ADMS) [15] to translate Verilog-A module code to C++ code. Finally, the C++ version of a model is compiled to dynamic machine code and linked to the main body of simulator code. A schematic symbol has been specially developed which allows compiled Verilog-A models to be combined with EDD and other conventional component symbols. The function and details of this advanced modeling technique are introduced in the following sections.

A. The relationship between EDD and Verilog-A modules

The design of the EDD has been tailored to match the functions provided by the Verilog-A static and dynamic current contribution statements. The generalized form of these relationships are defined in Fig. 7. In Fig. 8 a complete tunnel diode subcircuit is drawn along side its equivalent Verilog-A module code with each of the different sections indicated by horizontal arrow pointers. Note the structure of the EDDTD Verilog-A module. This is typical for a large number of compact models, allowing the module code to be written by hand without difficulty, as is the case in Fig. 8. In QucsStudio the PC ”White-Board” concept has been extended to add a second window which acts as a color highlighted text editor similar to the popular Windows ”notepad++” text editor package. Simply simulating the Verilog-A hardware description language listed in the second window causes QucsStudio to convert the module code from Verilog-A to C++, followed by an automatic C++ compile and link sequence, attaching the machine code model to the verilog-A file name. Indeed any time a change occurs in the Verilog-A module code QucsStudio automatically recompiles and links the new version of the model to the main body of the simulator C++ code. Fig. 9 shows the simple tunnel diode test set with the



```
Eqn1
Gp=1e-9
Rgmin=1/Gp
P_Q=1.602176462e-19
P_K=1.3806503e-23
TempK=Tcir+273
VTH=(P_K*TempK)/P_Q
```

```
SUB1
Vp=50e-3
Vv=370e-3
Ip=4.2e-3
Iv=370e-6
Vpp=525e-3
Cp=10e-12
Rs=1.0
Tcir=27
Ls=1e-9
```

```
`include "disciplines.vams"
`include "constants.vams"
module EDDTD(nC, nA);
inout nC, nA;
electrical ni2, nC, ni1, nA;
parameter real Vp=50e-3;
parameter real Vv=370e-3;
parameter real Ip=4.2e-3;
parameter real Vpp=525e-3;
parameter real Cp=20e-12;
parameter real Iv=370e-6;
parameter real Tcir=27;
parameter real Ls=1e-9;
parameter real Rs=1.0;
real TcirK, P_Q, P_K,
      VTH, Gp, Rgmin;
branch (ni2, nC) B1;
branch (ni2, ni1) B2;
branch (ni1, nA) B3;
analog begin
TcirK=Tcir+273;
P_Q=1.692176496e-19;
P_K=1.3806503e-23;
VTH=(P_K*TcirK)/P_Q;
Gp=1e-9;
Rgmin=1/Gp;
I(B1) <+ Ip*exp(-Vpp/VTH)*
              (exp(V(B1)/VTH)-1.0);
I(B1) <+ ddt( Cp*V(B1) );
I(B1) <+ Ip*(V(B1)/Vp)*
              exp(1.0-(V(B1)/Vp));
I(B1) <+ Iv*exp(V(B1)-Vv);
I(B2) <+ V(B2)/( Rs );
V(B3) <+ ddt(Ls*I(B3));
I(B1) <+ V(B1)/( Rgmin );
end
endmodule
```

```
X1
I1=Ip*exp(-Vpp/VTH)*(exp(_v1/VTH)-1.0)
Q1=Cp*_v1
I2=Ip*(_v1/Vp)*exp(1.0 - _v1/Vp)
Q2=0
I3=Iv*exp(_v1-Vv)
```

Fig. 8. A QucsStudio tunnel diode compact model drawn at both symbol and internal body hierarchy level, showing Verilog-A module structure and positioning of the Equation:Eqn1 block, subcircuit parameters, components Rs, Ls, and $Rgmin$ within the module code. Note that parameter $Temp$ has been renamed $Tcir$ in the Verilog-A code to remove a clash of names between the EDD and Verilog-A naming conventions.

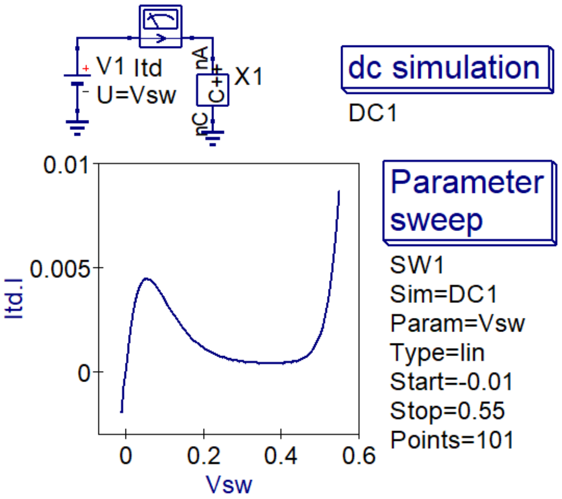


Fig. 9. Tunnel diode d.c. test bench showing compiled C++ compact model symbol and Itd plotted against $Vtd = Vsw$.

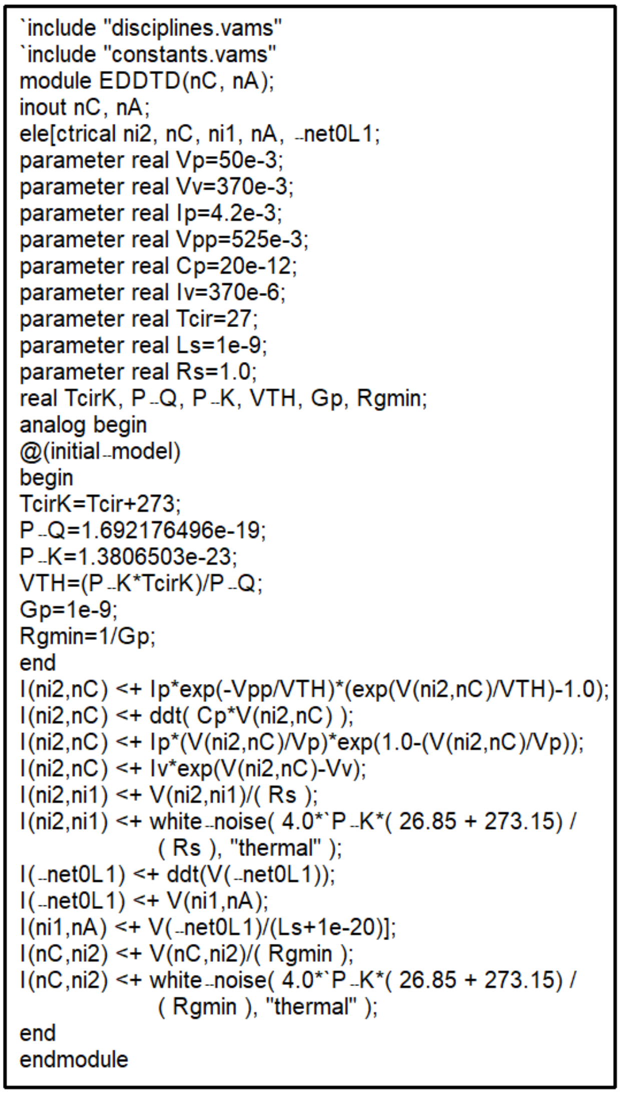
EDD subcircuit model replaced by the compiled Verilog-A equivalent.

B. Qucs-S Verilog-A module synthesis

Qucs-S has a number of important extensions when compared to Qucs. One of these is a built in synthesizer for generating Verilog-A module code from a subcircuit schematic [5]. In this process the Qucs-S subcircuit must only be constructed with the following Qucs components: R, C, L, $VCCS$, $CCCS$, $VCVS$, $CCVS$, a $subcircuit$ wrapper, a set of $subcircuit\ parameters$, EDD, $Equation \quad Eqn$, $pinspx$ and the SPICE B style nonlinear current source. Fig. 10 lists the tunnel diode Verilog-A module code generated by the Qucs-S synthesizer. Notice that there appears to be a significant number of differences between the hand crafted Verilog-A code listed in Fig. 8 and the synthesized code, for example in Fig. 10 the current contributions are written in terms of node names, rather than branches, and the ADMS Verilog-A statement $@(initial\_model)$ $begin.....end$; is included. Notice also that in Fig. 10 the inductor is synthesized by three Verilog-A statements, based on the electrical equivalent circuits given in Fig.11, rather than the single statement in Fig.8. However, in reality both sets of code provide the same overall function. The synthesized Verilog-A code is in a format from which it is possible to hand adjust the statements to meet the capabilities of different Verilog-A to C++ translators. Synthesized Verilog-A modules can be added to both Qucs and QucsStudio projects, compiled and used like standard built-in components. The Qucs/QucsStudio circuit simulators also provide facilities that allow Verilog-A modules to be stored in libraries of new components.

C. The Verilog-A Equation-Defined Device (VAEDD)

Compact modeling of nonlinear devices with EDD and Verilog-A modules represent two techniques at opposite ends of a scale going from purely interpreted models to compiled C++ models. From a practical point of view what is often required is an approach to modeling that has the convenience of interpreted EDD models coupled with the high simulation speed obtained with compiled C++ models synthesized from Verilog-A code. It is also worth noting that in a large number of models, often with more than one EDD, simulation time is largely determined by the complexity of the $In(Vn)$ and $Qn(Vn, In)$ equations for each of the EDD branches. Hence, significant improvement in simulation speed can be obtained by replacing one or more of the most complex EDD branches with an equivalent Verilog-A module called a VAEDD [13]. The tunnel diode compact model shown in Fig. 12 has branch one of the original EDD, see Fig. 8, replaced by a VAEDD branch, see Fig. 12 (a) and (c). This is in reality a tiny Verilog-A module with a structure that is simple to construct, follows a standard template and simulates at speeds significantly faster



```
`include "disciplines.vams"
`include "constants.vams"
module EDDTD(nC, nA);
inout nC, nA;
ele[ctrical ni2, nC, ni1, nA, _net0L1;
parameter real Vp=50e-3;
parameter real Vv=370e-3;
parameter real Ip=4.2e-3;
parameter real Vpp=525e-3;
parameter real Cp=20e-12;
parameter real Iv=370e-6;
parameter real Tcir=27;
parameter real Ls=1e-9;
parameter real Rs=1.0;
real TcirK, P_Q, P_K, VTH, Gp, Rgmin;
analog begin
@(initial_model)
begin
TcirK=Tcir+273;
P_Q=1.692176496e-19;
P_K=1.3806503e-23;
VTH=(P_K*TcirK)/P_Q;
Gp=1e-9;
Rgmin=1/Gp;
end
I(ni2,nC) <+ Ip*exp(-Vpp/VTH)*(exp(V(ni2,nC)/VTH)-1.0);
I(ni2,nC) <+ ddt( Cp*V(ni2,nC) );
I(ni2,nC) <+ Ip*(V(ni2,nC)/Vp)*exp(1.0-(V(ni2,nC)/Vp));
I(ni2,nC) <+ Iv*exp(V(ni2,nC)-Vv);
I(ni2,ni1) <+ V(ni2,ni1)/( Rs );
I(ni2,ni1) <+ white_noise( 4.0*`P_K*( 26.85 + 273.15) /
              ( Rs ), "thermal" );
I(_net0L1) <+ ddt(V(_net0L1));
I(_net0L1) <+ V(ni1,nA);
I(ni1,nA) <+ V(_net0L1)/(Ls+1e-20)];
I(nC,ni2) <+ V(nC,ni2)/( Rgmin );
I(nC,ni2) <+ white_noise( 4.0*`P_K*( 26.85 + 273.15) /
              ( Rgmin ), "thermal" );
end
endmodule
```

Fig. 10. Qucs-S synthesized Verilog-A module code for the EDD tunnel diode model.

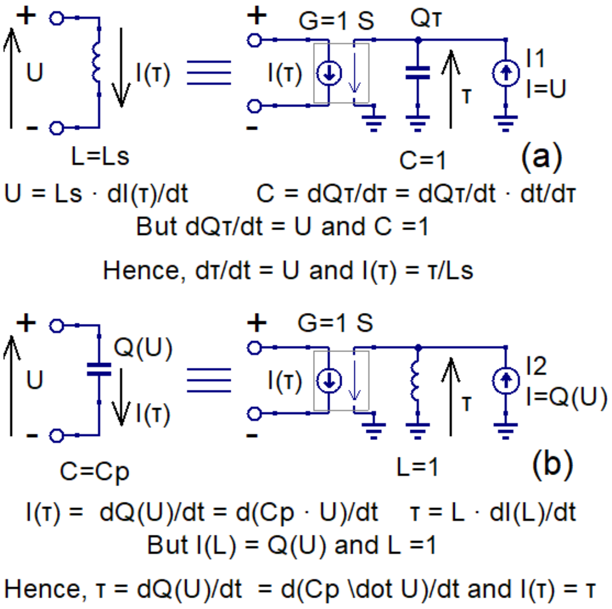


Fig. 11. Synthesized nonlinear inductor and capacitor models: (a) inductor and (b) capacitor.

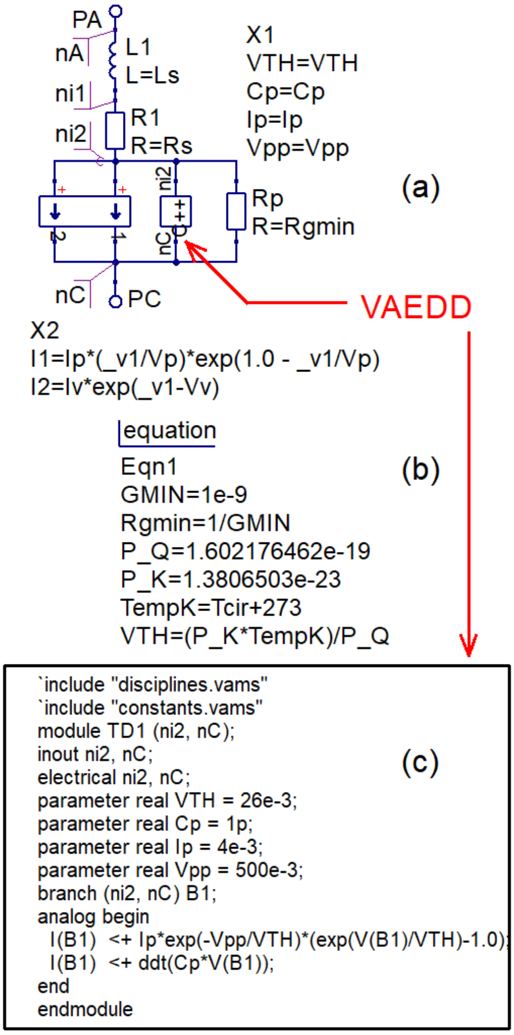
than its EDD equivalent. Further gains in simulation speed can be achieved by replacing more than one EDD branch with equivalent Verilog-A VAEDD.

V. SPICE Multi-simulator engine compact modeling

Qucs-S is a ground-breaking circuit simulation package in that it allows users to select a simulation engine from (1) the Qucs built-in Qucsator simulator, (2) the SPICE 3f5 compatible SPICE OPUS simulator, (3) the next generation SPICE Ngspice circuit simulator, and (4) the new SPICE compatible Xyce circuit simulator. As these SPICE 3f5 related simulation engines all have some form of extensions when compared to the original Berkeley software, Qucs-S selects which built-in models and library models are allowed with each package. In order for users to be made aware of which model works with each package schematic symbols are color coded; dark blue denoting a legacy Qucs item, red/brown signifying a Ngspice/SPICE OPUS item and dark green an Xyce item. This makes identification of components/devices placed on the PC "White-Board" straight forward. Qucs-S automatically synthesizes the different netlist formats for each of the SPICE engines, taking into account individual package extensions. This process is not simply a one-to-one translation of the Qucs-S symbols to a single line SPICE statement but is a more complex procedure that takes into account component function, often resulting in more than one SPICE statement per symbol. A typical Xyce translation output netlist is given in Fig. 13 for the tunnel diode test circuit shown in Fig. 5. Note the use of Qucs-S "*dc simulation*" and "*Parameter sweep*" icons in both Fig. 5 and Fig. 9. Qucs-S icons are implemented for all the fundamental SPICE simulation types. However, with advance mature packages like Xyce continuous development is underway, regularly adding a range of new simulation types, for example Harmonic Balance analysis, and output data manipulation statements like .MEASURE, making it difficult to keep up with the volume of changes. With Qucs-S this situation was anticipated and a scripting icon developed that allows users to incorporate into Qucs-S future changes made by the Xyce development team. Fig. 14 (a) indicates how scripts of Xyce simulation and output data processing are added to a Qucs-S PC "White-Board". Fig. 14 (b) introduces a number of SPICE related icons, who's names identify their function, for adding and manipulating Xyce netlist elements.

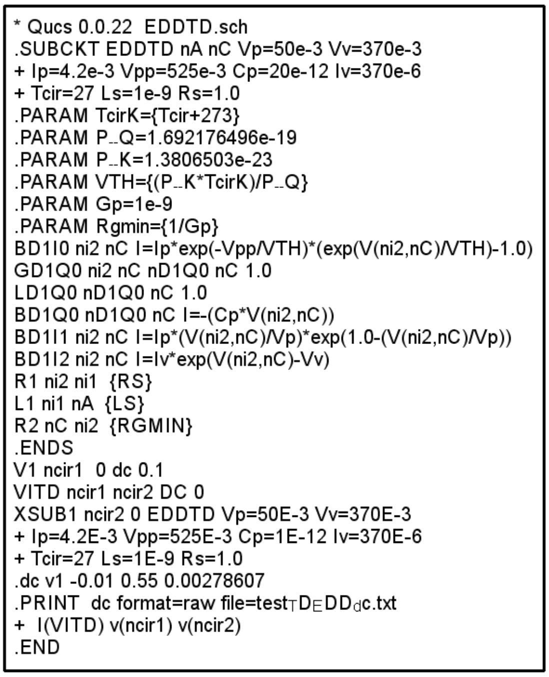
VI. Current and future developments

Both Qucs-S and QucsStudio are active projects with new features, improvements and bug fixes regularly released by



```
`include "disciplines.vams"
`include "constants.vams"
module TD1 (ni2, nC);
inout ni2, nC;
electrical ni2, nC;
parameter real VTH = 26e-3;
parameter real Cp = 1p;
parameter real Ip = 4e-3;
parameter real Vpp = 500e-3;
branch (ni2, nC) B1;
analog begin
  I(B1) <+ Ip*exp(-Vpp/VTH)*(exp(V(B1)/VTH)-1.0)
  I(B1) <+ ddt(Cp*V(B1));
end
endmodule
```

Fig. 12. The tunnel diode compact model with branch one of the EDD replaced with a VAEDD: (a) and (b) the revised model and (c) the VAEDD Verilog-A code.



```
* Qucs 0.0.22  EDDTD.sch
.SUBCKT EDDTD nA nC Vp=50e-3 Vv=370e-3
+ Ip=4.2e-3 Vpp=525e-3 Cp=20e-12 Iv=370e-6
+ Tcir=27 Ls=1e-9 Rs=1.0
.PARAM TcirK={Tcir+273}
.PARAM P_Q=1.692176496e-19
.PARAM P_K=1.3806503e-23
.PARAM VTH={(P_K*TcirK)/P_Q}
.PARAM Gp=1e-9
.PARAM Rgmin={1/Gp}
BD1I0 ni2 nC I=Ip*exp(-Vpp/VTH)*(exp(V(ni2,nC)/VTH)-1.0)
GD1Q0 ni2 nC nD1Q0 nC 1.0
LD1Q0 nD1Q0 nC 1.0
BD1Q0 nD1Q0 nC I=-(Cp*V(ni2,nC))
BD1I1 ni2 nC I=Ip*(V(ni2,nC)/Vp)*exp(1.0-(V(ni2,nC)/Vp))
BD1I2 ni2 nC I=Iv*exp(V(ni2,nC)-Vv)
R1 ni2 ni1  {RS}
L1 ni1 nA  {LS}
R2 nC ni2  {RGMIN}
.ENDS
V1 ncir1  0 dc 0.1
VITD ncir1 ncir2 DC 0
XSUB1 ncir2 0 EDDTD Vp=50E-3 Vv=370E-3
+ Ip=4.2E-3 Vpp=525E-3 Cp=1E-12 Iv=370E-6
+ Tcir=27 Ls=1E-9 Rs=1.0
.dc v1 -0.01 0.55 0.00278607
.PRINT  dc format=raw file=testTDEDDdc.txt
+  I(VITD) v(ncir1) v(ncir2)
.END
```

Fig. 13. Xyce SPICE netlist for the tunnel diode test bench given in Fig. 5.

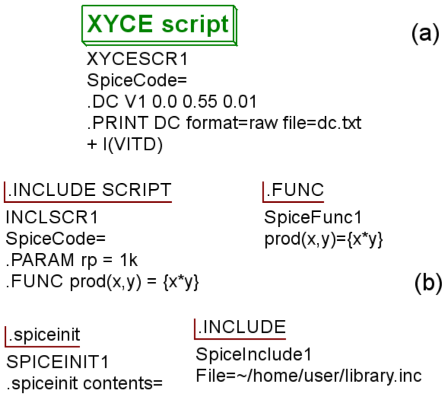


Fig. 14. Qucs-S SPICE netlist manipulation icons: (a) $XYCE\ script$, and (b) other related SPICE netlist handling scripts.

their Development Teams. The same is true for the Ngspice and Xyce circuit simulators. SPICE OPUS has been include for use with Qucs-S because it provides a base line simulator who's properties are essentially the same as the published Berkeley SPICE 3f5 software. The following sections introduce a number of current and proposed developments that aim at improving the Qucs-S/Xyce and Qucsstudio compact modeling and simulation capabilities, and the PC "White-Board" platform.

A. Xyce

Since the release of Xyce version 6.0, as open source software, under the General Public License (GPL) 3.0, new versions of the software have been appeared roughly every six months. As of March 2020 Version 7.0 is the current stable package. For the Qucs-S Development Team this is both a good and a bad feature of the Xyce software. Good in the sense that the Xyce package offers new features at every release plus bug fixes. Bad in the sense that continuous changes and adaptions of the Qucs-S code are needed to keep in step with Xyce. It was largely for this reason that the $XYCE\ script$, $.INCLUDE$, $.FUNC$, $.INCLUDE$, and $spiceinit$ control icons have been implemented. These allow Xyce SPICE netlists to be placed on a PC "White-Board" and interpreted at simulation run time. Qucs-S has in fact a two level GUI system; items common to SPICE 3f5, and other equivalent simulators, operate via built-in Icons or a $XYCE\ script$, while the less used or recently added features, can only be accessed via a $XYCE\ script$. For example, since 2018 approximately 20 important additions to Xyce functionality have been implemented, including Monte Carlo analysis and Lattice hypercube sampling via a new $.SAMPLING$ feature, Transient simulation direct sensitivity analysis that supports $.FOUR$, $.LIN$ for S parameter multiport analysis with Y and Z output data in Touchstone level 1 and level 2 format, and a new charge expression variant for capacitors that is similar to the EDD branch charge implementation. The tunnel diode model test circuit shown in Fig. 15 illustrates both the use of the $XYCE\ script$ icon and a number of new Xyce features: firstly the diode compact model has been built from SPICE B style current sources with the capacitor Cp modeled with a Q style component ($Q = Cp \cdot V(ni2)$), secondly note the extensive use of $\{...\}$ round equations, thirdly the use of

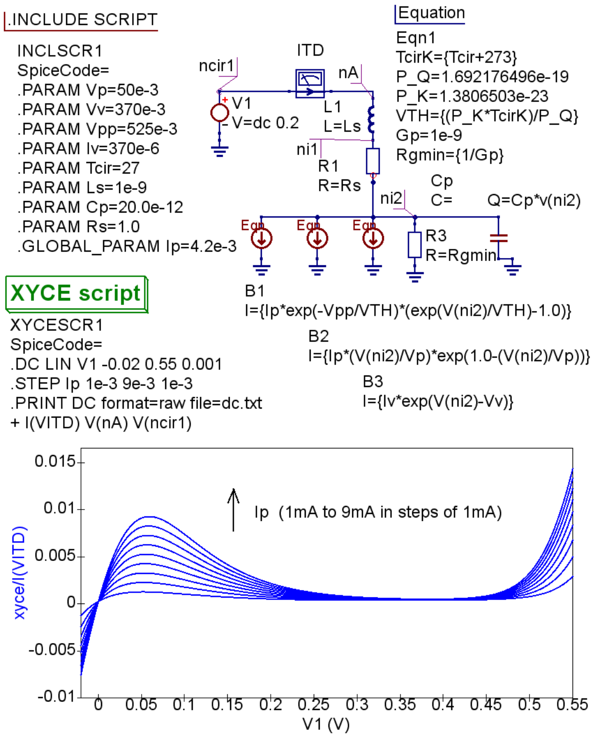
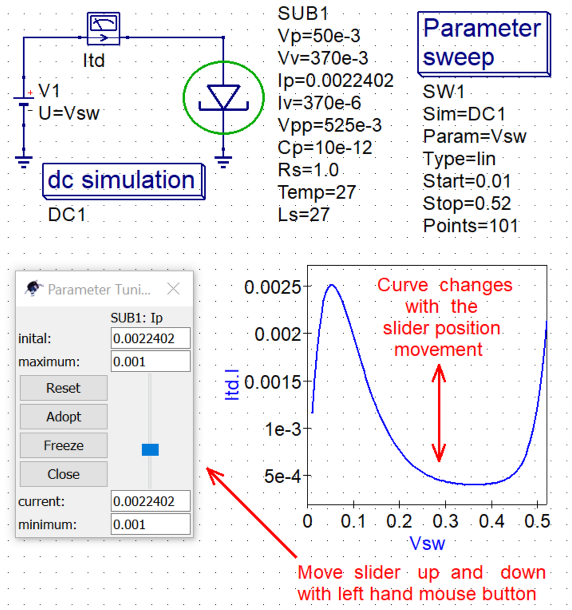


Fig. 15. A Xyce SPICE style tunnel diode compact model with a test bench for investigating the effects of stepped device parameters.

$.GLOBAL\_PARAM$ to identify parameters to be stepped, and finally the combination of SPICE directives $.DC$, $.STEP$ and $.PRINT$ to direct simulation and output data.



SUB1
Vp=50e-3
Vv=370e-3
Ip=0.0022402
Iv=370e-6
Vpp=525e-3
Cp=10e-12
Rs=1.0
Temp=27
Ls=27

SW1
Sim=DC1
Param=Vsw
Type=lin
Start=0.01
Stop=0.52
Points=101

SUB1: Ip
inital: 0.0022402
maximum: 0.001
current: 0.0022402
minimum: 0.001

Curve changes with the slider position movement

Move slider up and down with left hand mouse button

Fig. 16. The QucsStudio EDD tunnel diode model test bench with an interactive slider for investigating the effects of changing device parameter Ip over range of values: in this example parameter Ip is changed over the range 1mA (minimum = slider down) to 10mA (maximum = slider up).

B. QucsStudio

The current version of QucsStudio is 2.57. This package has reached an advanced stage of development in that it offers an almost complete set of circuit simulation routines covering the d.c. to transient domains with significant additions beyond SPICE 3f5 like multi-tone Harmonic Balance analysis, Monte Carlo analysis, parameter sweep, multi-port S parameter and noise simulation, optimization and system simulation. Full ”turn-key” Verilog-A compact modelling is also offered via the ADMS software. In terms of White-Board” development QucsStudio is particularly interesting in that it is the first of the Qucs series of circuit simulators to introduce interactive animation as a tool for advanced circuit simulation. Illustrated in Figure 16 is the basic tunnel diode d.c. test circuit, introduced previously, where the value of current peak parameter I_p can be set by changing the position of a slider with the left hand mouse button. QucsStudio allows one or more parameter values to be simultaneously controlled by sliders. With the computational power of a modern PC changes in simulation output data can be observed as movements in plotted curves as the sliders are moved. The parameter slider technique is particularly useful as a process for obtaining (good guess) starting values in parameter sets prior to full computer controlled optimization.

C. Merging simulation and measurements

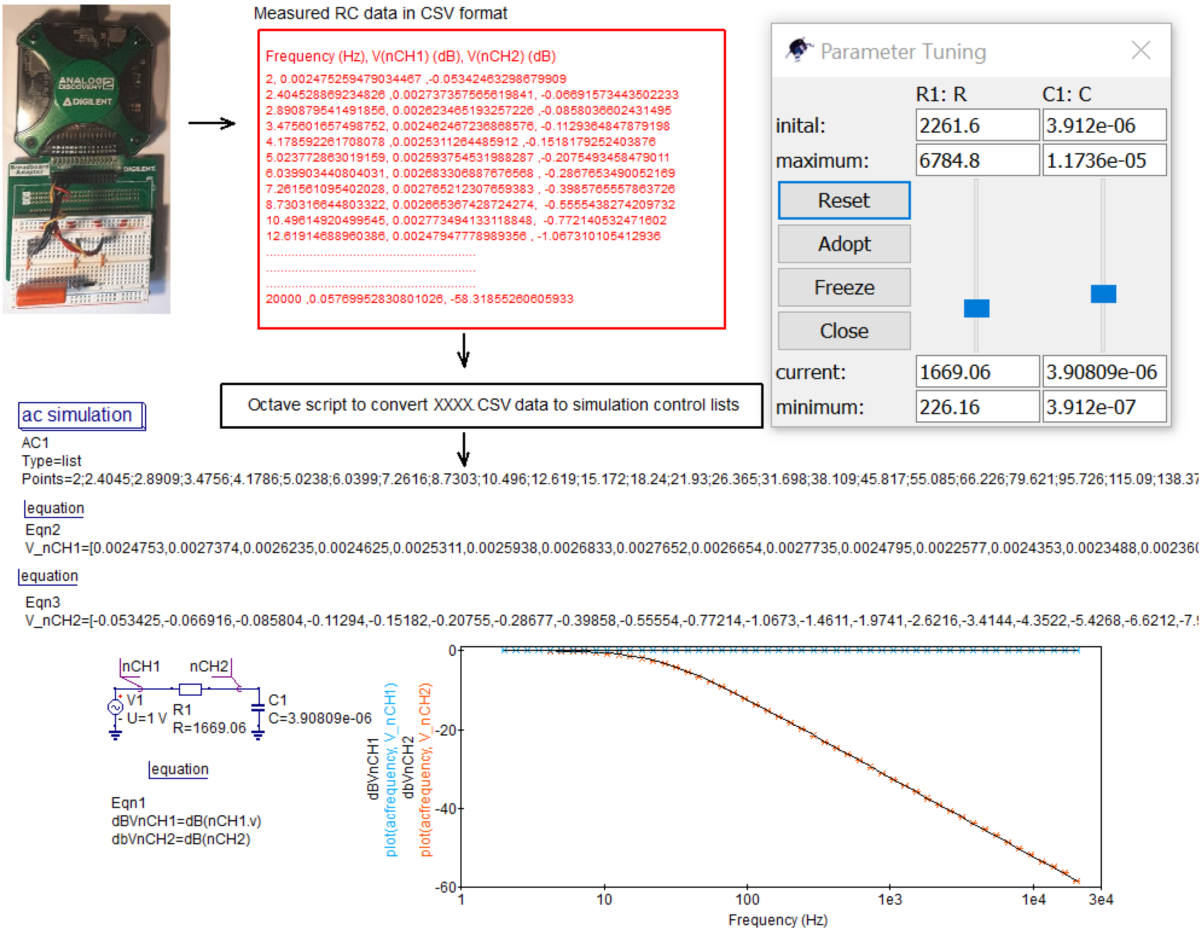
Illustrated in Fig. 17 is an example of a comprehensive ”White-Board” for simulating the a.c. performance of a simple passive first order low pass RC filter. The latest extensions to the Qucs ”White-Board” repertoire includes pictures (in Fig. 17 a picture of an ”Analog Discovery 2” transfer function measurement system [16]), information pointer directed data flow diagrams and associated text blocks (in Fig. 17 a note explaining the use of an Octave script for converting measured CSV formatted output data to simulation control icons with data lists ($AC\ simulation\ AC1$, $Equation\ Eq2$ and $Equation\ Equ3$)), a QucsStudio simulation schematic plus plotted measured and simulated output data controlled by parameter tuning (of values R1 and C1). In this example two points are worth noting, firstly the AC simulation frequency range is determined by the ”Points” list, synchronizing the measured and simulation frequency values, and secondly the picture, text and arrow pointers shown on the ”White-Board” are transparent during simulation and play no part in computing the RC voltage transfer function. Moreover, their primary role is to provide a clear indication of the processes involved in the function of the test bench, making the concept of a self-documenting ”White-Board” an important step in the development of the next generation compact modeling and circuit simulation tools.

VII. Summary

Low cost high performance PC engineering work stations have encouraged the development of compact device modeling and circuit simulation tools centered on high resolution graphics interfaces for schematic drawing, simulation and output data visualization. This paper outlines the structure and capabilities of a ”White-Board” display system developed for the Qucs series of circuit simulators and modeling tools. These tools allow interactive prototyping of compact device models and their testing using the Qucs/QucsStudio and Qucs-S ”White-Board” environment as a central platform in the construction of production level Verilog-A device models. The ”White-Board” environment offers significant interactive circuit entry and data display improvements when compared to those implemented in previous generations of circuit simulator, namely high resolution schematic drawing, simulation control Icons, output data visualization, attached design equations and self documenting text and flow diagram features. Each of these, when coupled with established, or new, compact modeling techniques, like non-linear EDD, FEDD, mixed Equation-Defined Device and Verilog-A models (VAEDD) plus verilog-A modules, make the evolving ”White-Board” a highly flexible and innovative platform for compact modeling and circuit simulation in the current era. Future expansion of the ”White-Board” concept indicates that by merging device parameter measurements with established circuit simulation will significantly extend the scope of traditional circuit simulation.

References

[1] A.R. Newton, D. O. Pederson and A Sangiovanni-Vincentelli, ”SPICE Version 2g User’s Guide”, Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA, 1981.



Measured RC data in CSV format

```
Frequency (Hz), V(nCH1) (dB), V(nCH2) (dB)
2, 0.002475259479034467 ,-0.05342463298679909
2.404528869234826 ,0.002737357565619841, -0.06691573443502233
2.890879541491856, 0.002623465193257226 ,-0.0858036602431495
3.475601657498752, 0.002462467236868576, -0.1129364847879198
4.178592261708078 ,0.0025311264485912 ,-0.1518179252403876
5.023772863019159, 0.002593754531988287 ,-0.2075493458479011
6.039903440804031, 0.002683306887676568 , -0.2867653490052169
7.261561095402028, 0.002765212307659383 , -0.3985765557863726
8.730316644803322, 0.002665367428724274,  -0.5555438274209732
10.49614920499545, 0.002773494133118848,  -0.772140532471602
12.61914688960386, 0.002479477789893356 , -1.067310105412936
.................................................................
.................................................................
.................................................................
20000 ,0.05769952830801026, -58.31855260605933
```

Octave script to convert XXXX.CSV data to simulation control lists

Parameter Tuning

	R1: R	C1: C
inital:	2261.6	3.912e-06
maximum:	6784.8	1.1736e-05
current:	1669.06	3.90809e-06
minimum:	226.16	3.912e-07

ac simulation
AC1
Type=list
Points=2;2.4045;2.8909;3.4756;4.1786;5.0238;6.0399;7.2616;8.7303;10.496;12.619;15.172;18.24;21.93;26.365;31.698;38.109;45.817;55.085;66.226;79.621;95.726;115.09;138.37

equation
Eqn2
V_nCH1=[0.0024753,0.0027374,0.0026235,0.0024625,0.0025311,0.0025938,0.0026833,0.0027652,0.0026654,0.0027735,0.0024795,0.0022577,0.0024353,0.0023488,0.00236

equation
Eqn3
V_nCH2=[-0.053425,-0.066916,-0.085804,-0.11294,-0.15182,-0.20755,-0.28677,-0.39858,-0.55554,-0.77214,-1.0673,-1.4611,-1.9741,-2.6216,-3.4144,-4.3522,-5.4268,-6.6212,-7.

equation
Eqn1
dBVnCH1=dB(nCH1.v)
dbVnCH2=dB(nCH2)

Fig. 17. A simple RC low pass passive filter example showing an advanced "White-Board" with a computer controlled transfer function measurement system where the measured output data is converted from CSV format to Qucs simulation control ICONS. (Note the right hand side of the $AC\ simulation$ and the two $Equation$ Icons have been truncated.)

[2] B. Johnson, T. Quarles, A.R. Newton, D. O. Pederson and A Sangiovanni-Vincentelli, "SPICE3 Version 3f User's Manual", Department of Electrical Engineering and Computer Science, University of California, Berkeley, CA, 1992.

[3] M. Margraf et al. Qucs (Quite Universal Circuit Simulator). Version 0.0.19, 2020. Available from http://qucs.sourceforge.net/index.html. [Accessed May 2020].

[4] M. Margraf, "QucsStudio Version 2.5.7. Available from http://dd6um.darc.de/ QucsStudio/qucsstudio.html, 2018, [Accessed May 2020].

[5] V. Kusnetsov and M. Brinson, "Qucs-S: Qucs with SPICE". Version 0.0.22. Available from https://ra3xdh.github.io/, 2020. [Accessed January 2020].

[6] John W. Eaton, David Bateman, Søren Hauberg and Rik Wehbring. "GNU Octave version 5.1.0 manual: a high-level interactive language for numerical computations". 2019, CreateSpace Independent Publishing Platform. ISBN 144141300. Available from http://www.gnu.org/software/octave/doc/interpreter/, [Accessed January 2020].

[7] S. Jahn, M. Brinson, H. Parruitte, B. Ardouin, P. Nenzi and L. Lemaitre, "GNU simulators supporting Verilog-A compact model standardization", MOS-AK meeting, Premstaetten 2007. Available from http://www.mos-ak.org/premstaetten/papers/MOSAK_QUCS_ngspice_ADMS.pdf. [Accessed May 2020].

[8] M. Brinson and H. Nabijou, "Adaptive subcircuits and compact Verilog-A macromodels as integrated design and analysis blocks in Qucs circuit simulation", International Journal of Electronics, 2011, 98(5), pp. 631-645.

[9] M. Brinson, "Qucs:Component, compact device and circuit modelling using symbolic equations", 2007. Available from http://qucs.sourceforge.net/docs/tutorial/equations.pdf. [Accessed May 2020].

[10] S. Jahn and M. Brinson, "Interactive compact device modelling using Qucs equation-defined devices", International Journal of Numerical Modelling: Electronic Networks, Devices and Fields, 2008, 21(5), pp 335-349.

[11] M. Brinson and S. Jahn, Qucs: "A GPL software package for circuit simulation, compact modelling and circuit macromodelling from DC to RF and beyond", International Journal of Numerical Modelling: Electronic Networks, Devices and Fields, 2009, 22(5), pp 297-319.

[12] M. Brinson and S. Jahn, "Modelling of high-frequency inductance with Qucs non-linear radio frequency equation defined devices", International Journal of Electronics, 2009, 96(3), pp. 307-321.

[13] M. Brinson, "FOSS Compact Model Prototyping with Verilog-A Equation-Defined Devices (VAEDD)", 2019 MIXDES - 26th International Conference "Mixed Design of Integrated Circuits and Systems", Rzeszów, Poland, 2019, pp. 92-97.

[14] Accellera, "Verilog-AMS Language Reference Manual, version 2.2". Available from: http://www.accellera.org, [Accessed May 2020].

[15] L. Lemaitre, C. McAndrew and S. Hamm, "ADMS- Automatic device model synthesizer", IEEE Custom Integrated Circuits Conference, 2002, pp. 27-30.

[16] Digilent (A National Instruments Company), "Analog Discovery 2: 100MS/s USB Oscilloscope, Logic Analyzer and Variable Power Supply", Digilent Inc. 1300 Henley Ct 3, Pullman, WA 99163, 2020.