A modular approach to next generation Qucs

Felix Salfelder\textsuperscript{1} and Mike Brinson\textsuperscript{2}

Qucs Development Team\textsuperscript{1}

Centre for Communications Technology, London Metropolitan University, UK\textsuperscript{2}
A modular approach to next generation Qucs: presentation topics

• Background and timeline
• Next generation Qucs
  Qucs - a modular approach
  But why plug-ins?
  Comparison and plans (exemplified)
  Library design
  Development
  Example slides
• Qucs-S (Qucs for SPICE)
  Qucs-S: a FOSS circuit simulator formed from the Qucs GUI plus Xyce simulation engine
  Compact device modeling: Equation Defined Device model and Verilog-A module development
• Merging Qucs(Qt5) GUI and Qucs-S
  Qucs(Qt5)/Xyce schematic symbols and models
  Evolving Qucs(Qt5)/Xyce modeling and simulation capabilities
• Summary
• References
A modular approach to next generation Qucs: background and timeline

- Release 0.0.1 2003
- Release 0.0.17 2013
- Release 0.0.18 2014
- Release 0.0.19 2017
- Release 0.0.20 2019

- FORK
  - Behavioural and Verilog-A compact device modeling

- Qucs

- RF

- SPICE

- Qucs-S

- Next generation Qucs (Qt5)
  - User loadable device models via plugins:
    1. Qucs/Quc salvator
    2. Xyce
    3. Ngspice
    4. WRspice
  - User loadable simulation data visualization plugins:
    1. Qucs/Quc salvator
    2. Octave
    3. Gnuplot
    4. Spreadsheets
  - User loadable simulation engine control plugins:
    1. Qucs/Quc salvator
    2. Xyce
    3. Ngspice
    4. WRspice

- Release 0.0.19 2017
- Release 0.0.20 2017
- Release 0.0.21 2018
- Release 0.0.22 2020
A modular approach to next generation Qucs: Qucs – a modular approach

- Evident challenges
  - Over 100,000 lines of code
  - A zoo of features
  - Moving targets (dependencies and applications)

- Approach: Small library and extensions
  - Separate data from UI and interactions
  - Establish internal and external interfaces
  - Opt-in extensions (including default features)
  - User extensions (no approval required)

- Goals
  - Growth without forking
  - Encourage alternatives and improvement
  - Avoid review and integration overhead
  - Keep/support the legacy (“0.0.20”) but move forward
A modular approach to next generation Qucs: But why plug-ins?

A Plug-in extends a program

- Loaded at run time
- Provides nontrivial features
- Allows customisation
- Does anything (aka. Turing completeness)

Known software with plug-in support

- `linux> modprobe pcsPkr`  # I can play sound
- `python> import numpy`    # fancy arrays and maths
- `gnucap> attach bsimXYZ.so`  # use transistor model
- Firefox, Gimp, Inkscape, Emacs(especially)
A modular approach to next generation Qucs: Comparison and plans - exemplified

<table>
<thead>
<tr>
<th></th>
<th>GnuCap</th>
<th>Qucs</th>
<th>Xyce</th>
<th>future Qucs</th>
</tr>
</thead>
<tbody>
<tr>
<td>plugins</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>interactions</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>devices</td>
<td>yes</td>
<td>many</td>
<td>many</td>
<td>plugin</td>
</tr>
<tr>
<td>GUI</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>plugins</td>
</tr>
<tr>
<td>HB simulation</td>
<td>n/a</td>
<td>no</td>
<td>yes</td>
<td>plugin</td>
</tr>
<tr>
<td>Verilog-AMS</td>
<td>n/a</td>
<td>no</td>
<td>?</td>
<td>plugin</td>
</tr>
<tr>
<td>Python library</td>
<td>YOU</td>
<td>GET</td>
<td>THE</td>
<td>IDEA</td>
</tr>
</tbody>
</table>
A modular approach to next generation Qucs: Intended plug-ins

Devices
- Legacy (C++)
- Legacy "lib" files
- "multi view"
- (any)

Schematic file
- Legacy
- Verilog
- (any)

Netlist file
- Qucsator
- Verilog
- Spice ...
- (any)

Simulation
- Subprocess
- Shared library
- (any)

Data
- Legacy ("dat")
- hdf5 (wip)
- shared mem (wip)
- (any)
A modular approach to next generation Qucs: Library design

- Few base classes, broad inheritance
- Clear distinction between data and UI
  - Circuit representation
  - UI (cli) + GUI (Qt)
- Explicit objective circuit model
  - Symbols with ports and parameters
  - Nodes, Nets & Hierarchy
- Interfaces for
  - Parsers, netlisters, subprocesses
  - Data sets and manipulation
  - Elaborate symbols ("multiview", wire, equation...)
  - GUI (Dialog widgets, menus, drawings ...)
  - (Non-C++ plugins intended, need wrapper.)
A modular approach to next generation Qucs: Qt5 benefits

- Broadly available across platforms (Qt3/4 has reached EOL years ago)
- Fast & scalable graphics scene
- Support for high resolution displays
- Built-in manipulation routines
- 2D/3D data visualisation options
- Standardised undo/redo stack etc.
- (expecting easy transition to Qt6)
A modular approach to next generation Qucs: Development

Process
1. Identify relevant library, infrastructure
2. Move everything else to plugins

Status/Summary
- Library is converging (minor changes expected)
- new: Simulator interface
- new: Symbol plug-ins
- new: Pluggable Data
- new: Nets are explicit
- new: (Circuit & Schematic) language support
- Schematic editor (rewrite) usable, needs work.
- Diagrams, need work, QWidget rewrite intended
- Legacy schematics: Lacks some fancy stuff (labels...)
A modular approach to next generation Qucs:
Feature preview: Explicit nets

- “Which net does this wire belong to?”
- ... introduce (persistent) nets and keep track
- i.e. follow circuit alterations

- Simpler API and possible UI improvements
- Traditional "netlisting" is now trivial
- Labels, feedback, DRC ...
A modular approach to next generation Qucs: Finally - Qucs-S

Qucs-S plugins

Work in progress...
• A modular approach to next generation Qucs: Qucs-S = Qucs GUI plus Xyce

Qucs
Equation-Defined Devices (EDD)
Frequency Equation-Defined Devices (FEDD)
Verilog-A Equation-Defined Devices (VAEDD) and Verilog-A hand coded modules

Compact Modelling

Qucs-S
Verilog-A synthesised modules
CMC Verilog-A modules: BSIM6 and HICUM2, for example

GUI

ADMS
Automated Model Synthesizer, Open source Verilog-A translator

XYCE script

XYCESCR1
SpiceCode=
.AC LIN 2000 100 10MEG
.PRINT AC format=raw file=ac.txt V(1)

INCLUDE SCRIPT
INCLSCR1
SpiceCode=
.PARAM rp = 1k
.FUNC prod(x,y) = {x*y}

DC, AC and noise, S-parameter, TRAN
single tone Harmonic Balance, Parameter sweep, and optimization.
A modular approach to next generation Qucs: Equation Defined Device (EDD) models 1

\[ I = I(V), \quad g = \frac{dI}{dV} \]

\[ Q = Q(V,I), \quad C = \frac{dQ}{dV} = \frac{\partial Q(V)}{\partial V} + \frac{\partial Q(I)}{\partial I} \cdot g, \text{ where} \]

the current flowing in branch \( n \) is \( I_n = I(V_n) + \frac{d}{dt}(Q_n) \), and \( 1 \leq n \leq 20 \).

- EDD is a multiterminal nonlinear component with branch currents that can be functions of EDD branch voltage, and stored charge that can be a function of both EDD branch voltages and currents
- EDD is similar, but more advanced to the SPICE 3f5 B type I or V controlled sources
- EDD can be combined with conventional circuit components and Qucs-S equation blocks when constructing compact device models and subcircuit macromodels
- EDD is an advanced component, allowing users to construct prototype experimental models from a set of equations derived from physical device properties
- EDD operator \( \frac{d}{dt} \) is undertaken internally by Qucs-S
- Qucs-S EDD can have a maximum of 20 two terminal branches
A modular approach to next generation Qucs: Equation Defined Device (EDD) models 2

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

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

Parameter sweep
SW1
Sim=DC1
Param=Vsw
Type=lin
Start=0.01
Stop=0.5
Points=101
A modular approach to next generation Qucs: EDD models translated to Verilog-A modules

\( I_m = f(V_1, V_2, V_3, \ldots, V_8) \)
\( Q_m = f(V_1, V_2, V_3, \ldots, V_8, I_1, I_2, I_3, \ldots, I_8) \)

were \( V_m = V(nxm, nym) \), and \( 1 \leq m \leq 8 \)

**Equivalent Verilog-A fragment**

\[ I_m \leftrightarrow f(V_1, V_2, V_3, \ldots, V_8) \]
\[ I_m \leftrightarrow \text{ddt}(Q_m) \]

**Equation**

\[ V_{pp} = 50 \times 10^{-3} \]
\[ V_V = 370 \times 10^{-3} \]
\[ I_{p} = 4.2 \times 10^{-3} \]
\[ I_{v} = 370 \times 10^{-6} \]
\[ V_{pp} = 525 \times 10^{-3} \]
\[ R_s = 1.0 \]
\[ \text{Temp} = 27 \]
\[ L_s = 1.0 \times 10^{-9} \]
A modular approach to next generation Qucs: Synthesis of Qucs-S schematics to Verilog-A modules

Subcircuit components

Verilog-A module code

```verilog
// include “disciplines.vams”
// include “constants.vams”
module EDDTD(nC, nA);
inout nC, nA;
ele[nC], nC, nI, nA, _net0L1;
parameter real Vp=500-3;
parameter real Vv=3700-3;
parameter real Ip=4.2e-3;
parameter real Vpp=4260-3;
parameter real Cpr=20e-12;
parameter real Vo=3700-6;
parameter real TcIr=27;
parameter real L=1e-9;
parameter real R=1.0;
real TcIrK, P, _Q, P, _K, VTH, Gp, Rgmin;

analog begin
    @(initial_model)
    begin
        TcIrK=TcIr/273;
        P = 1.892176486e-10;
        P, _K = 1.38059e-33;
        VTH=P, _K, TcIrK/P, _Q;
        Gp=1e-9;
        Rgmin=1/Gp;
        end

        ![Eqn1](image)
y=1

        ![Equation](image)

        ![Subcircuit](image)
```

Synthesis

Build XSPICE IFS file from subcircuit
A modular approach to next generation Qucs:
Qucs/Qucs-S/Xyce Verilog-A compact device modeling

GUI
Graphical User Interface

Qucs
1. Text editor
2. Schematic capture
3. Launch circuit simulator

Qucs netlist

Qucs-S
1. Text editor
2. Schematic capture
3. Launch circuit simulator
4. Verilog-A module synthesis

EDD behavioural and Verilog-A compact device modeling

ADMS
Automatic Device Model Synthesizer

ADMS model prototypes

SPICE netlist

Compiled Verilog module C++ code

Output data processing and visualization

Xyce

Xyce output data processing and visualization

* CMC and other compact device models:
  1. DIODE.CMC, 2. BJT – VBIC 1.3, FBH HBT.X, HICUMLO/L2, MEXTRAM.
  3. MOSFET – BSIM3, BSIM4, BSIM6, BSIM.SOI, BSIM.CMG, MVS, PSP
  4. Memristor-TEAM, -Yakopcic, -PEM
A modular approach to next generation Qucs: Qucs(Qt5)/Xyce schematic symbols and models

Points to note:
1. Component and device models are no longer hard wired into Qucs.
2. Xyce SPICE dialect becomes the Qucs(Qt5)/Xyce netlist format.
3. The body of each schematic symbol holds defining Xyce SPICE code

Basic test library for Xyce
A modular approach to next generation Qucs: Qucs(Qt5)/Xyce schematic symbols and models 2

Non-linear cap: Algebraic equation passed as a parameter that is a function of internal nodes p1, p2

\[ c_{c1} = 10^7 \times (1 + v(p1,p2))^2 \times (\alpha + \beta \times v(p1,p2)) \]

Parameters (equivalent to variables): alpha, beta and dcsweep can be changed during simulation.

\[
\begin{align*}
\text{XYCSCR1} \\
\text{SpiceCode=} \\
&\cdot \text{global_param alpha=0.3} \\
&\cdot \text{global_param beta=0.03} \\
&\cdot \text{global_param dcsweep=0.0} \\
&\cdot \text{ac dec 300 1e4 1e7} \\
&\cdot \text{step lin dcsweep 0.0 1.0 0.2} \\
&\cdot \text{print ac format=std file=ac.dat dcsweep} \\
&\phantom{=} + \text{vm(xammeter1:1) pm} \\
&\phantom{=} + \text{vm(xvoltagemeter1:1) pm} \\
\end{align*}
\]

\(\text{dcsweep}\) value changed by \'.step\'
A modular approach to next generation Qucs: Qucs(Qt5)/Xyce tunnel diode EDD compact model

Introducing an extended EDD for Xyce/SPICE circuit simulation

EDD3B1
i1=q*(vp3,p4)/vp*exp(1.0-vp3,p4)/vp)
q=0.0

Qucs/Xyce generated SPICE netlist

TD test bench and simulation output
A modular approach to next generation Qucs: Simulation with CMC Verilog-A standardised device models

Problem: Xyce nested .dc sweep data output is a continuous list
A modular approach to next generation Qucs: Evolving Qucs(Qt5)/Xyce simulation capabilities

S parameter analysis

0.1V <= dcsweep <= 0.3V

0.45V <= dcsweep <= 0.55V
A modular approach to next generation Qucs: Evolving Qucs(Qt5)/Xyce simulation capabilities

Harmonic Balance analysis

**XYCE script**

```text
XYCESCR1
SpiceCode= .options hint numfreq=10 startupperiods=10 numptps=100 .tb 0.5e6 .print hb_fd file=hb.txt vm(xprobe_v2:p1) + vm(xprobe_v2:p1) im(xmeter1:vpulse)
V0 (dc)=0.15V, Va(ac)=0.05V peak
V0 (dc)=0.15V, Va(ac)=0.1V peak
V0 (dc)=0.15V, Va(ac)=0.2V peak
```

24
QucsStudio and Qucs-S Verilog-A compact device modeling
Summary

This presentation introduced the background, concepts and current progress in developing a modular Qt5 version of Qucs, which is

● easy to maintain
● makes extensive use of plugins
● and is driven by discipline and user needs

The package is in its early stages of development and it is envisaged that the first general release will not before the end of 2021.
QucsStudio and Qucs-S Verilog-A compact device modeling - References

Qucs (0.0.19/20): https://github.com/Qucs/qucs/
Download - version (Linux, Windows or Mac) as required from home page.

Qucs-S (0.0.22): Qucs with SPICE - https://ra3xdh.github.io/;
Download – version (Linux or Windows) as required from home page.

Xyce: https://xyce.sandia.gov/
Download - version (Linux, Windows or Mac) as required from home page.