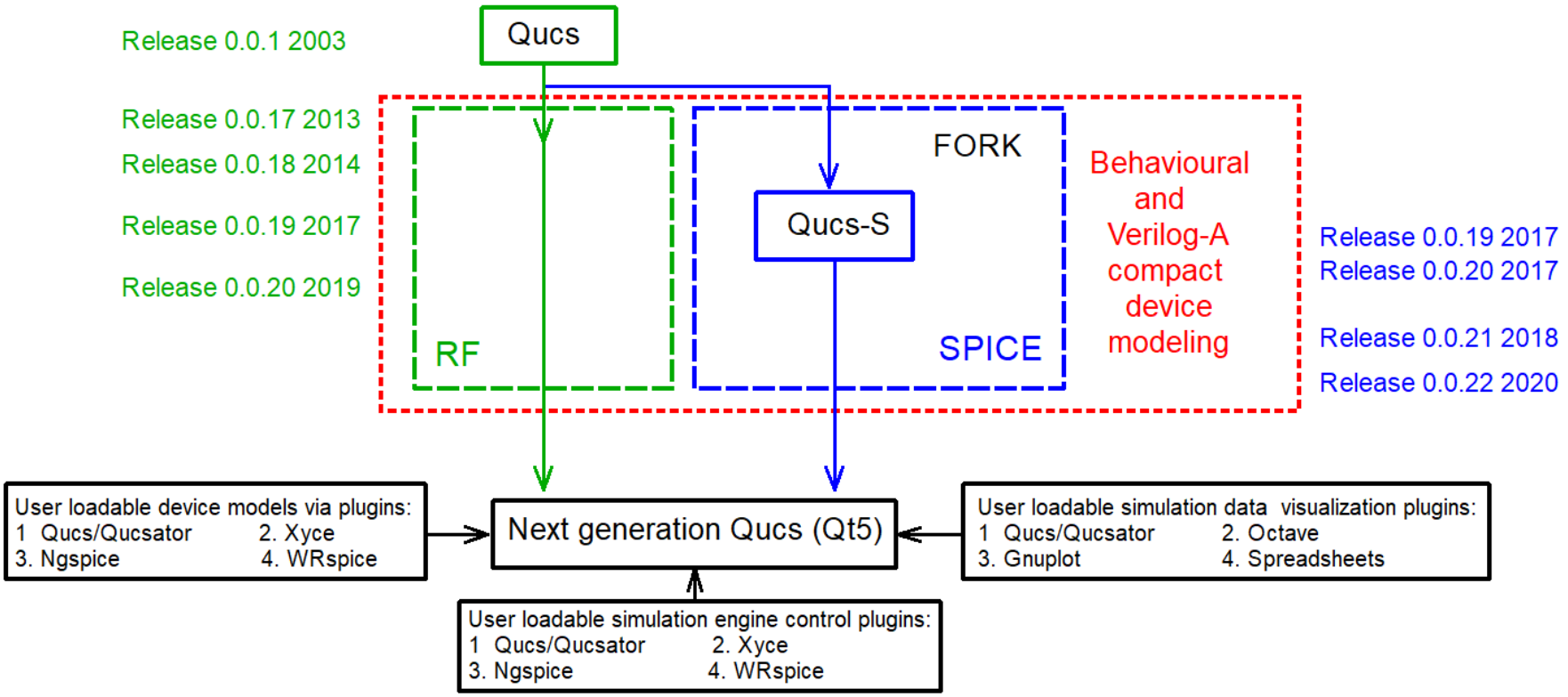
A modular approach to next generation Qucs

Felix Salfelder
Qucs Development Team
and
Mike Brinson
Centre for Communications Technology,
London Metropolitan University, UK

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



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

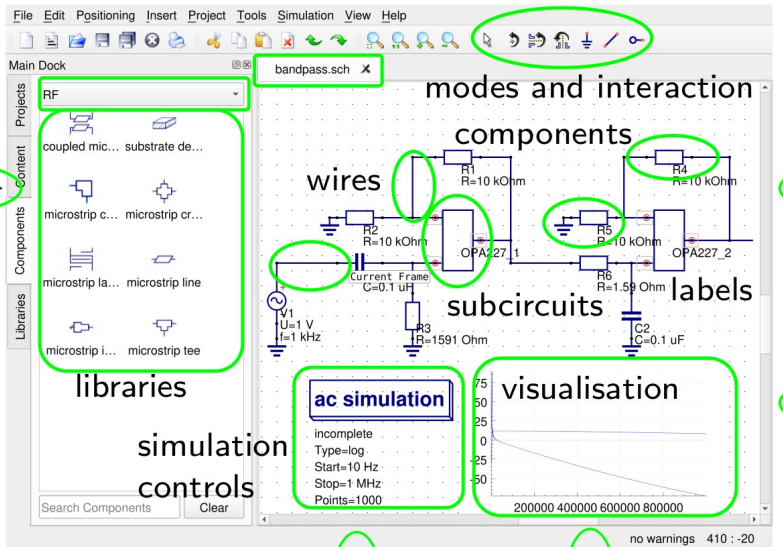
- linux> modprobe pcspkr # I can play sound
- python> import numpy # fancy arrays and maths
- gnucap> attach bsimXYZ.so # use transistor model
- Firefox, Gimp, Inkscape, Emacs(endless)

A modular approach to next generation Qucs: Comparison and plans - exemplified

	Gnucap	Qucs	Xyce	future Qucs
plugins	yes	no	no	yes
interactions	yes	yes	no	yes
devices		many	many	plugin
GUI	no	yes	no	plugins
HB simulation	n/a	no	yes	plugin
Verilog-AMS	n/a	no	?	plugin
Python library	YOU	GET	THE	IDEA

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

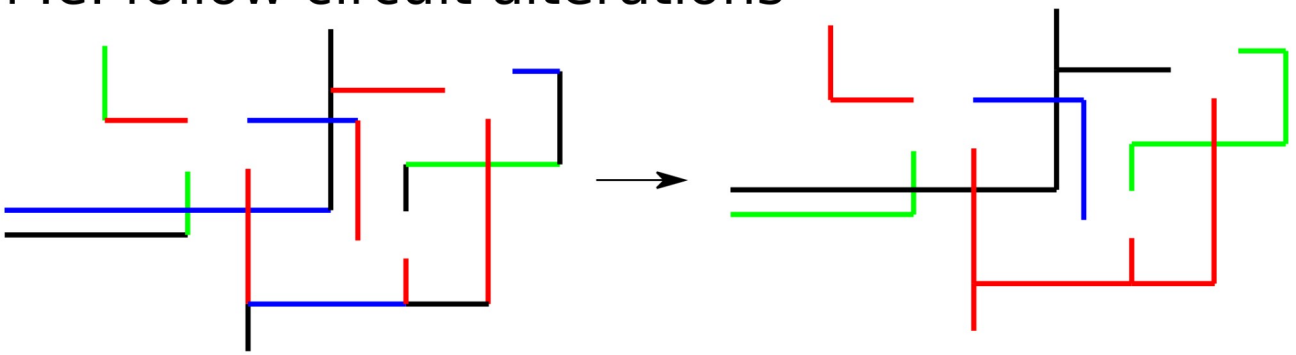
1. Identify relevant library, infrastructure
2. Move everything else to plugins

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, QtWidget rewrite intended
- Legacy schematics: Lacks some fancy stuff (labels...)

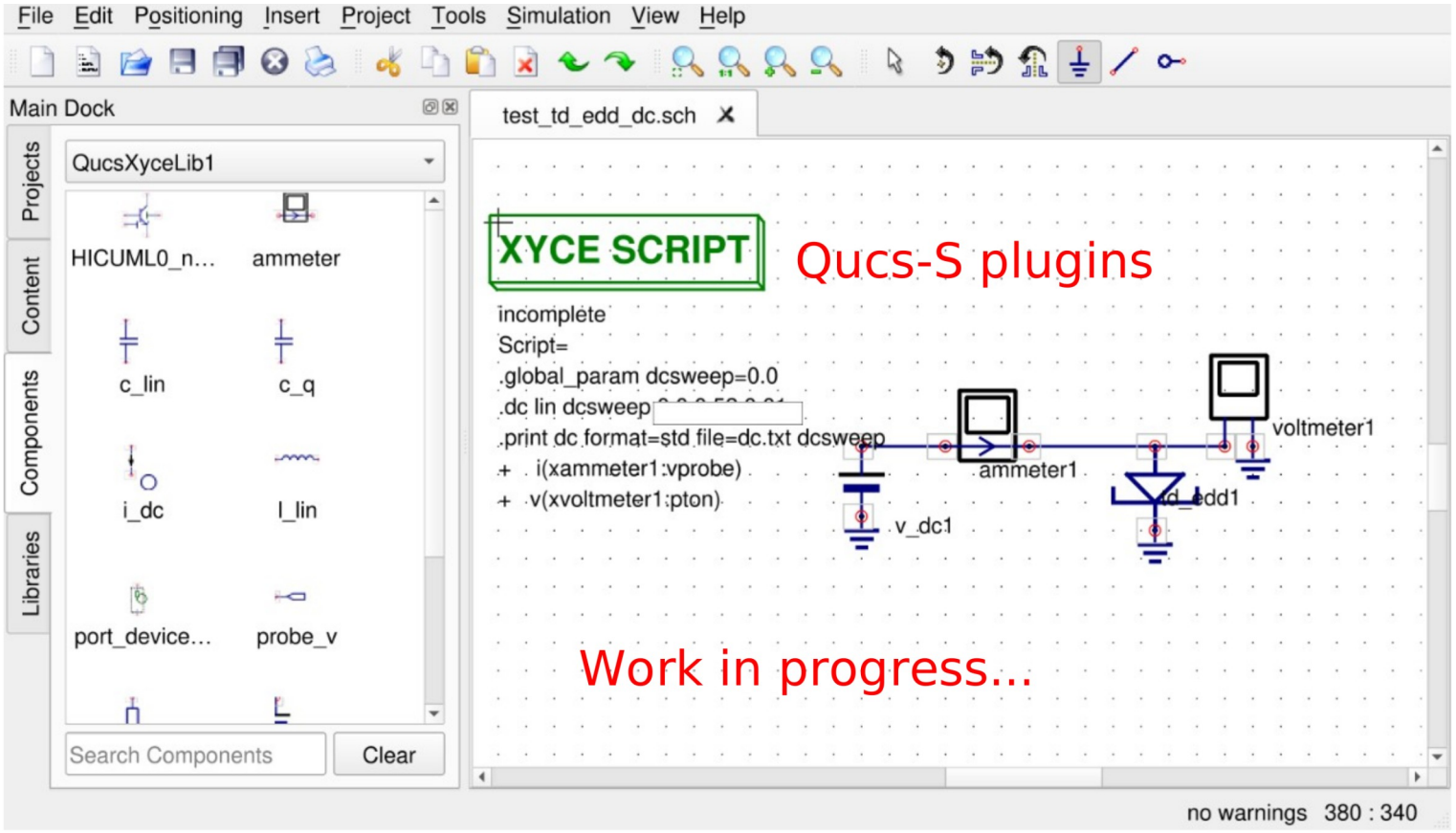
A modular approach to next generation Qucs: Feature preview: Explicit nets

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

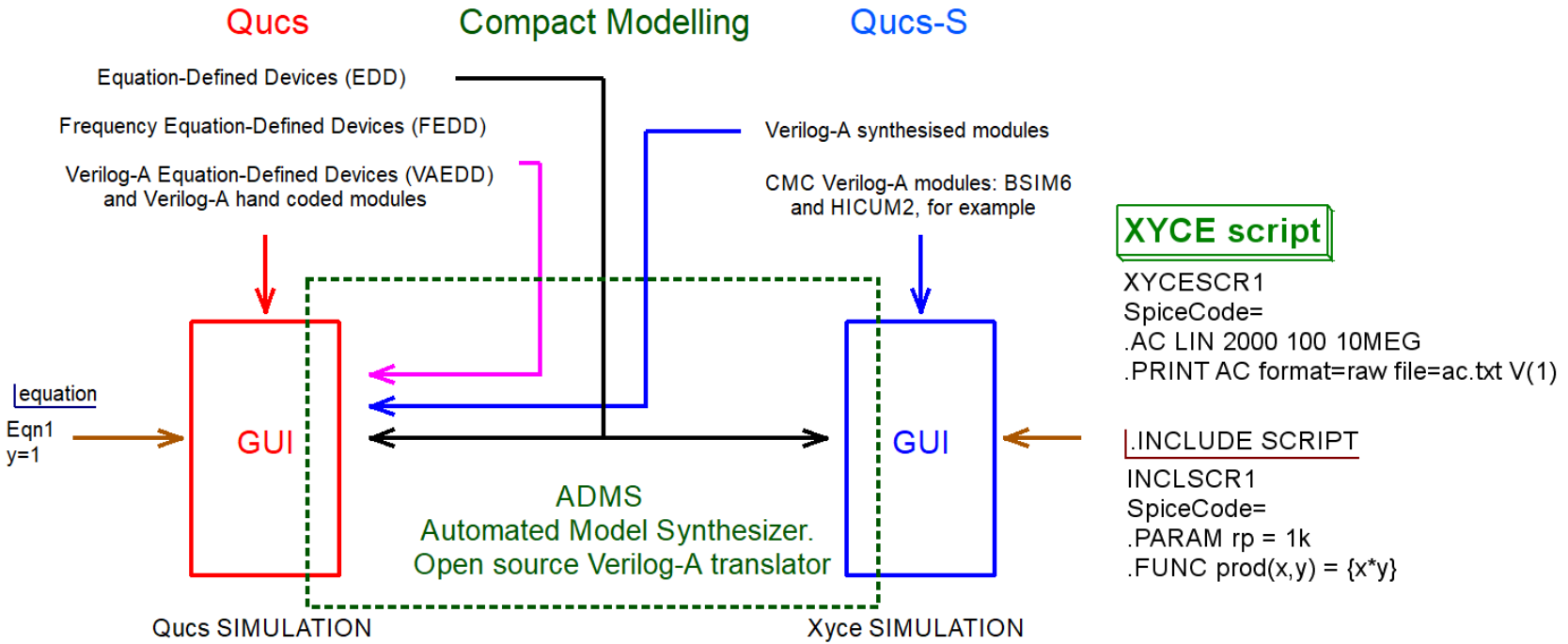


- Simpler API and possible UI improvements
- Traditional ”netlisting” is now trivial
- Labels, feedback, DRC ...

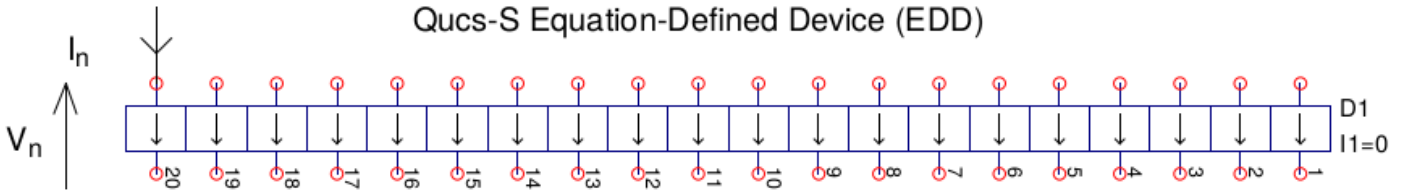
A modular approach to next generation Qucs: Finally - Qucs-S



• A modular approach to next generation Qucs: Qucs-S = Qucs GUI plus Xyce



A modular approach to next generation Qucs: Equation Defined Device (EDD) models 1



$I = I(V),\ g = dI/dV$

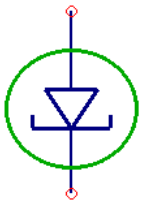
$Q = Q(V,I),\ C = dQ/dV = \partial Q(V)/\partial V + \partial Q(I)/\partial I \cdot g$, where

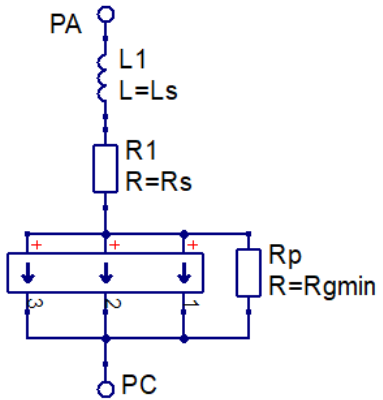
the current flowing in branch n is $I_n = I(V_n) + d/dt(Q_n)$, and $1 <= n <= 20$.

- 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 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

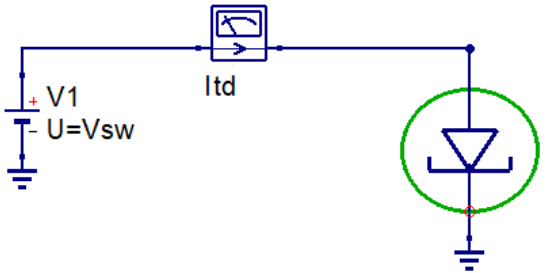




X1
I1=Ip*exp(-Vpp/VTH)*(exp(v1/VTH)-1.0)
Q1=Cp*v1
I2=Ip*(v1/Vp)*exp(1.0 - v1/Vp)
Q2=0
I3=Iv*exp(v1-Vv)

equation

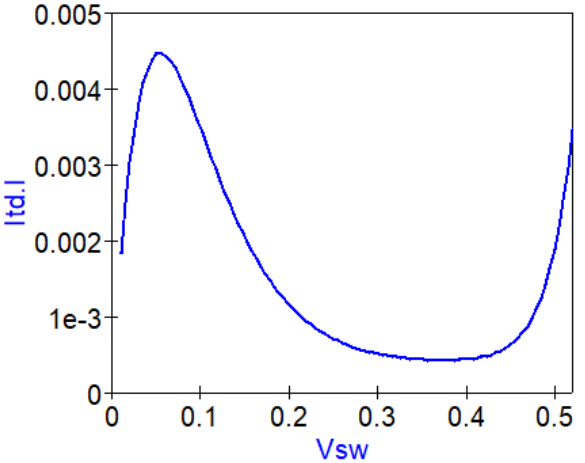
Eqn1
GMIN=1e-9
Rgmin=1/GMIN
P_Q=1.602176462e-19
P_K=1.3806503e-23
TempK=Temp+273
VTH=(P_K*TempK)/P_Q



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

dc simulation

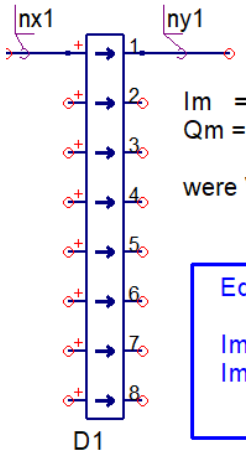
DC1



Parameter sweep

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

A modular approach to next generation Qucs: EDD models translated to Verilog-A modules

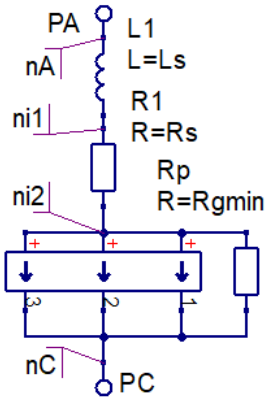


Im = f(V1, V2, V3 ..... V8)
Qm = f(V1, V2, V3, ... V8, I1, I2, I3, ...I8),

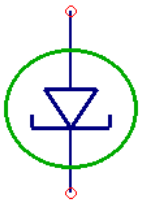
were Vm = V(nxm, nym), and 1 <= m <= 8

Equivalent Verilog-A fragment

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




```
`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
```



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

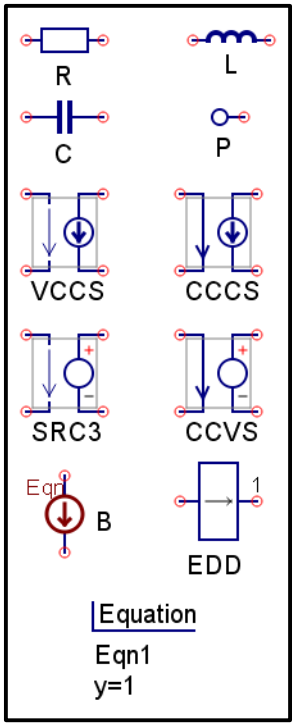
equation

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

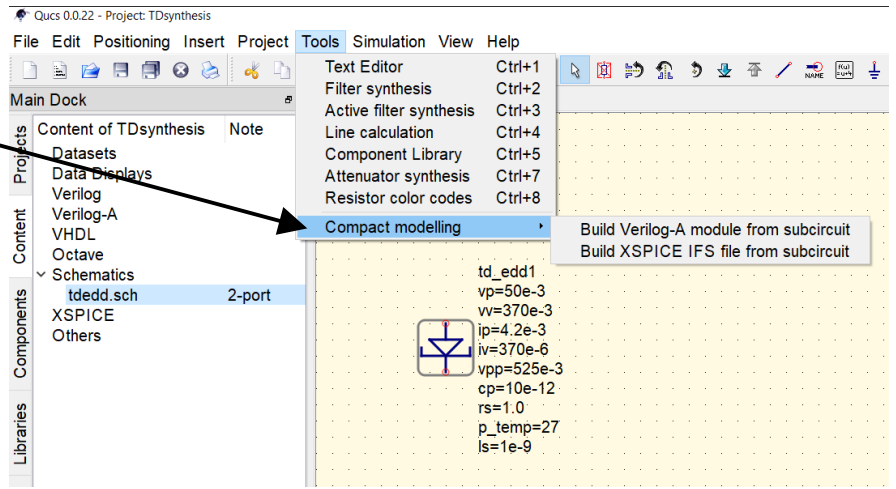
D1
I1=Ip*exp(-Vpp/VTH)*(exp(v1/VTH)-1.0)
Q1=Cp*v1
I2=Ip*(v1/Vp)*exp(1.0 - v1/Vp)
Q2=0
I3=Iv*exp(v1-Vv)

Rs
Ls
Rgmin

A modular approach to next generation Qucs: Synthesis of Qucs-S schematics to Verilog-A modules



Subcircuit components

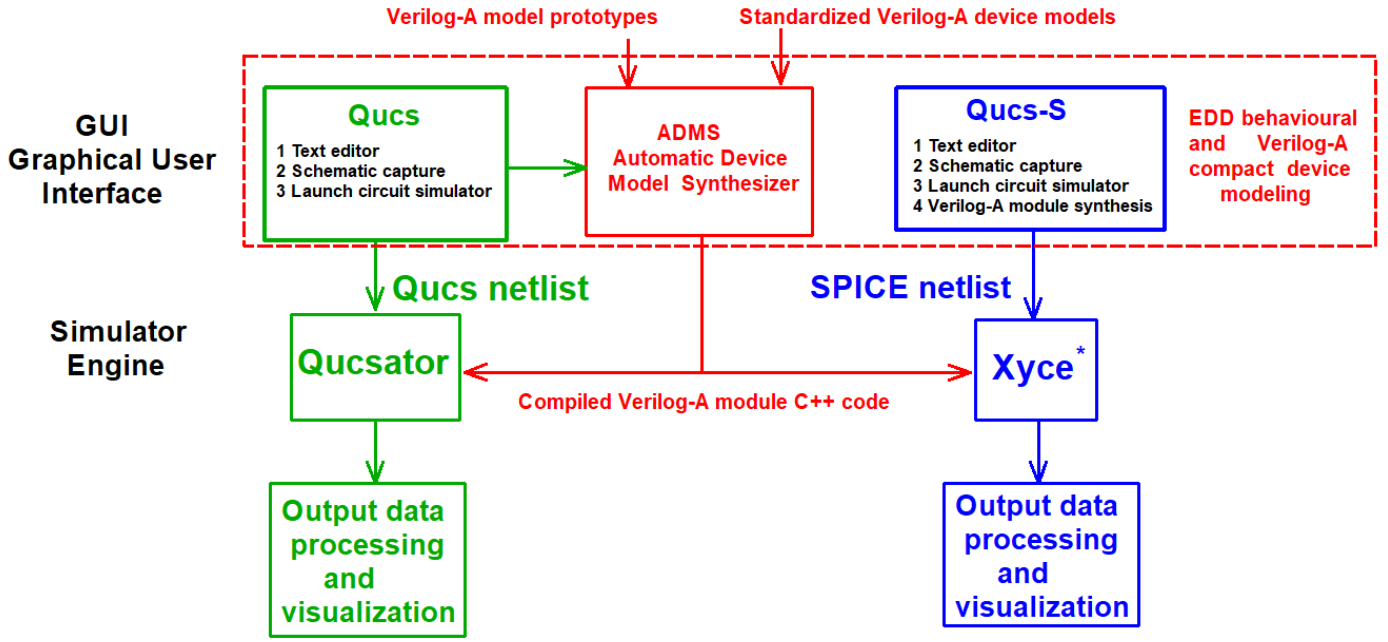


Synthesis

```
`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
```

Verilog-A module code

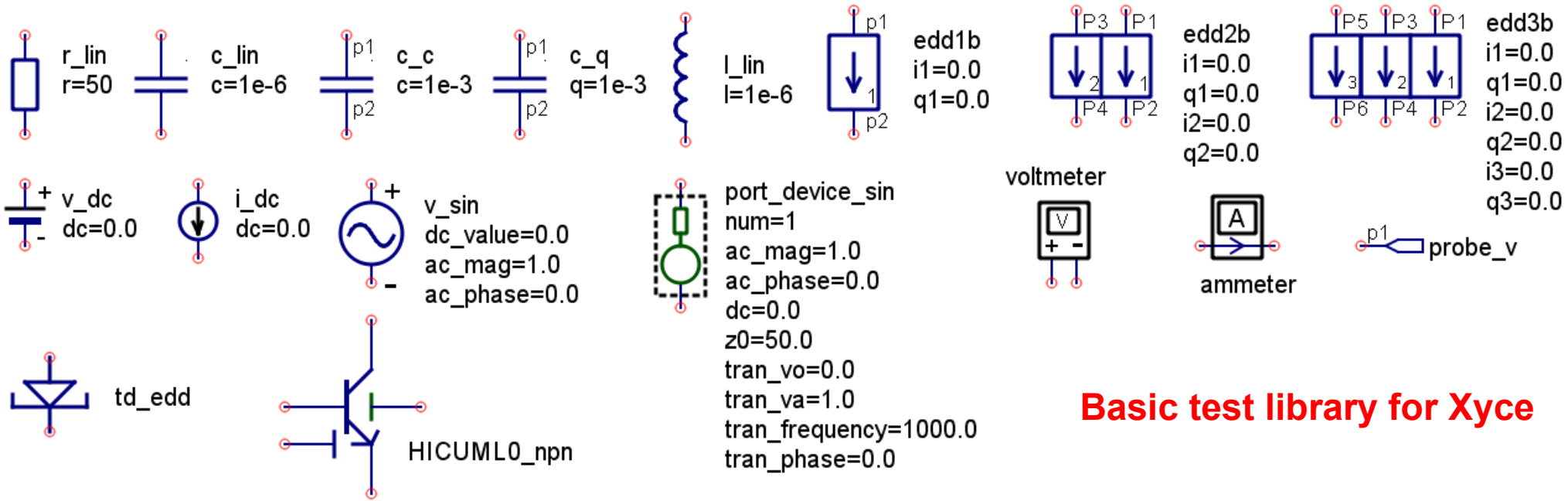
A modular approach to next generation Qucs: Qucs/Qucs-S/Xyce Verilog-A compact device modeling



* CMC and other compact device models:

1. DIODE_CMC , 2. BJT – VBIC 1.3, FBH HBT_X, HICUML0/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 1



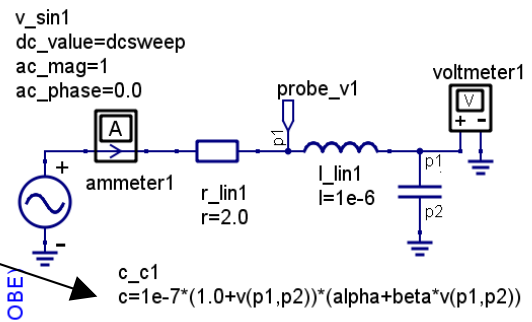
Basic test library for Xyce

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

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

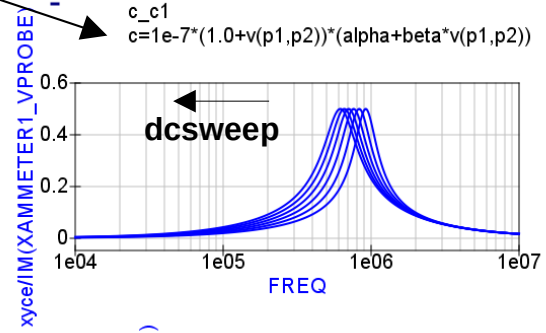
dcsweep value changed by .step

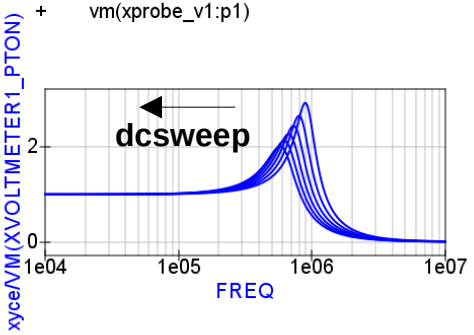


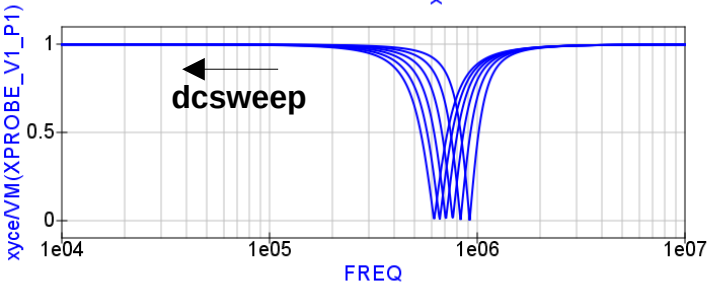
XYCE script

```
XYCESCR1
SpiceCode=
.global_param alpha=0.3
.global_param beta=0.03
.global_param dcsweep=0.0
.ac dec 300 1e4 1e7
.step lin dcsweep 0.0 1.0 0.2
.print ac format=std file=ac.txt  dcsweep
+       im(xammeter1:vprobe)
+       vm(xvoltmeter1:pton)
+       vm(xprobe_v1:p1)
```

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





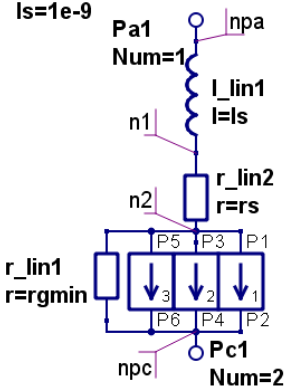


A modular approach to next generation Qucs: Qucs(Qt5)/Xyce tunnel diode EDD compact model

Introducing an extended EDD for Xyce/SPICE circuit simulation

```
td_edd1
vp=50e-3
vv=370e-3
ip=4.2e-3
iv=370e-6
vpp=525e-3
cp=10e-12
rs=1.0
p_temp=27
ls=1e-9
```

```
.INCLUDE SCRIPT
INCLSCR1
SpiceCode=
.param rgmin=1e9
.param p_q=1.60217646e-19
.param p_k=1.3806503e-23
.param tempK={p_temp+273}
.param vth={(p_k*tempK)/p_q}
```



TD EDD compact device model

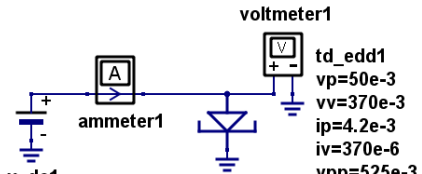
```
EDD3B1
i1=ip*exp(-vpp/vth)*(exp(v(p1,p2)/vth)-1.0)
q1=cp*v(p1,p2)
i2=ip*(v(p3,p4)/vp)*exp(1.0-v(p3,p4)/vp)
q2=0.0
i3=iv*exp(v(p5,p6)-vv)
q3=0.0
```

```
.SUBCKT r_lin p1 p2 r=50 m=1 tc1=0.0 tc2=0.0 tce=0.0 p_temp=26.58 dtemp=0.0
rlinear p1 p2 {r} m={m} tc1={tc1} tc2={tc2} tce={tce} temp={p_temp} dtemp={dtemp}
.ENDS
*
.SUBCKT l_lin p1 p2 l=1e-6 m=1.0 tc1=0.0 tc2=0.0 p_temp=26.58
rl_lin p1 p2 1e9
l_lin p1 p2 {l} m={m} tc1={tc1} tc2={tc2} temp={p_temp}
.ENDS
*
.SUBCKT eddb3 p1 p2 p3 p4 p5 p6 i1=0.0 q1=0.0 i2=0.0 q2=0.0 i3=0.0 q3=0.0
rpb1 p1 p2 1e9
bb1   p1 p2  i=i1
cb1   p1 p2 q=q1
rpb2  p3 p4 1e9
bb2   p3 p4 i=i2
cb2   p3 p4 q=q2
rpb3  p5 p6 1e9
bb3   p5 p6 i=i3
cb3   p5 p6 q=q2
.ENDS
```

```
.SUBCKT v_dc p1 p2 dc=0.0
v_dc p1 p2 dc {dc}
.ENDS
```

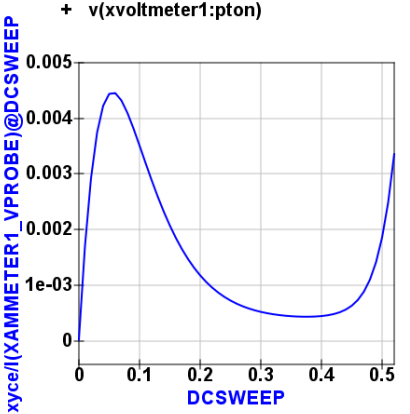
Qucs/Xyce generated SPICE netlist

```
.SUBCKT tdedd npa npc vp=50e-3 vv=370e-3 ip=4.2e-3 iv=370e-6 vpp=525e-3 cp=10e-12
+   rs=1.0 p_temp=27 ls=1e-9
.param rgmin=1e9
.param p_q=1.60217646e-19
.param p_k=1.3806503e-23
.param tempK={p_temp+273}
.param vth={(p_k*tempK)/p_q}
Xr_lin2 n2 npc r_lin r={RGMIN} m=1 tc1=0.0 tc2=0.0 tce=0.0 p_temp=26.58 dtemp=0.0
Xl_lin1 n1 npa l_lin l={LS} m=1.0 tc1=0.0 tc2=0.0 p_temp=26.58
Xr_lin1 n1 n2 r_lin r={RS} m=1 tc1=0.0 tc2=0.0 tce=0.0 p_temp=26.58 dtemp=0.0
XEDD3B1 n2 npc n2 npc n2 npc eddb3 i1={IP*EXP(-VPP/VTH)*(EXP(V(P1,P2)/VTH)-1.0)}
+   q1={CP*V(P1,P2)} i2={IP*(V(P3,P4)/VP)*EXP(1.0-V(P3,P4)/VP)} q2=0.0
+   i3={IV*EXP(V(P5,P6)-VV)} q3=0.0
.ENDS
Xtd_edd1 n2 0 tdedd vp=50E-3 vv=370E-3 ip=4.2E-3 iv=370E-6 vpp=525E-3 cp=10E-12
+    rs=1.0 p_temp=27 ls=1E-9
Xr_lin1 n2 n1 r_lin r=1 m=1 tc1=0.0 tc2=0.0 tce=0.0 p_temp=26.58 dtemp=0.0
Xv_dc1 n1 0 v_dc dc={DCSWEEP}
.global_param dcsweep = 0.0
.dc lin dcsweep 0.0 0.55 0.02
.print dc format=std file=dc.txt v(n1) v(n2)
.END
```



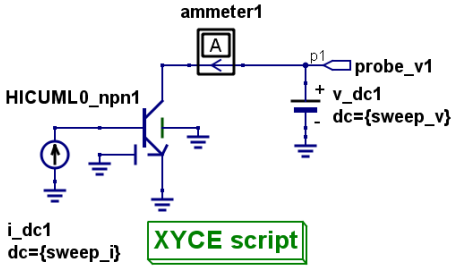
XYCE script

```
XYCESCR1
SpiceCode=
.global_param dcsweep=0.0
.dc lin dcsweep 0.0 0.52 0.01
.print dc format=std file=dc.txt dcsweep
+   i(xammeter1:vprobe)
+   v(xvoltmeter1:pton)
```



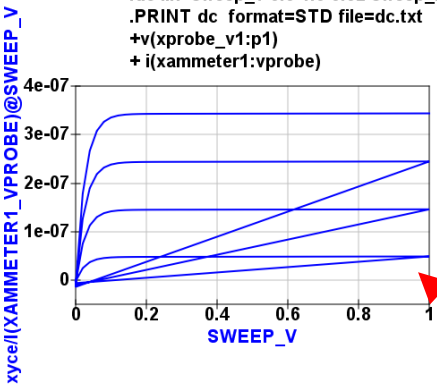
TD test bench and simulation output

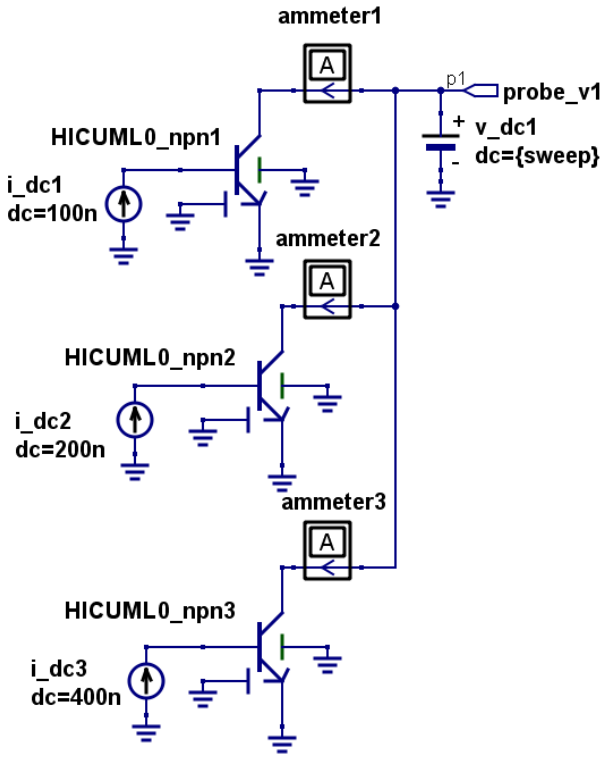
A modular approach to next generation Qucs: Simulation with CMC Verilog-A standardised device models



XYCE script

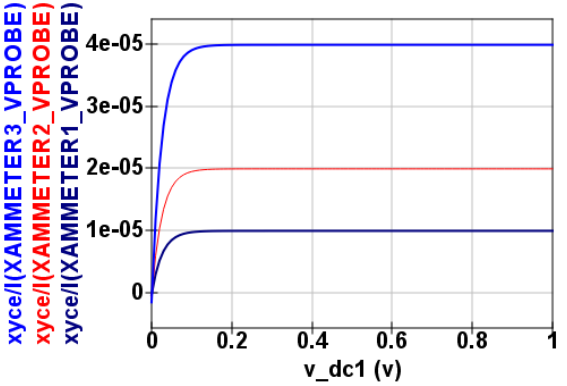
```
XYCESCR1
SpiceCode=
.global_param sweep_v =0.0
.global_param sweep_i=0.0
.dc lin  sweep_v 0.0 1.0 0.02 sweep_i  1n 4n 1n
.PRINT dc  format=STD file=dc.txt   sweep_i sweep_v
+v(xprobe_v1:p1)
+ i(xammeter1:vprobe)
```





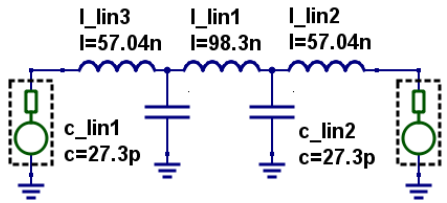
XYCE script

```
XYCESCR1
SpiceCode=
.global_param sweep =0.0
.dc lin sweep 0.0 1.0 0.01
.PRINT dc  format=STD file=dc.txt   sweep v(xprobe_v1:p1)
+ i(xammeter1:vprobe) i(xammeter2:vprobe) i(xammeter3:vprobe)
```



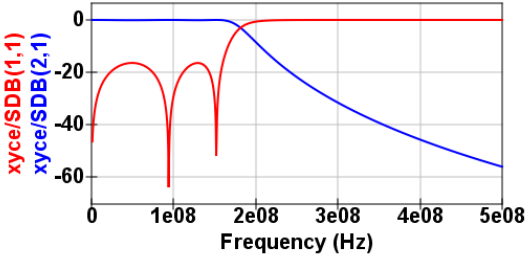
Problem: Xyce nested .dc sweep data output is a continuous list

A modular approach to next generation Qucs: Evolving Qucs(Qt5)/Xyce simulation capabilities 1

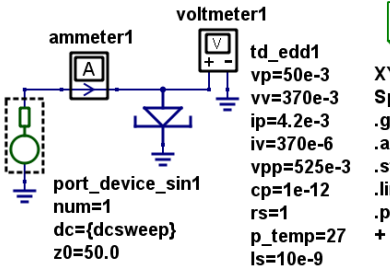


XYCE script

```
XYCESCR2
SpiceCode=
.ac lin  1000 1e6 500e6
.lin format=touchstone sparcalc=1
.print ac format=std file=sparam.txt sdb(1,1)  sdb(2,1)
```



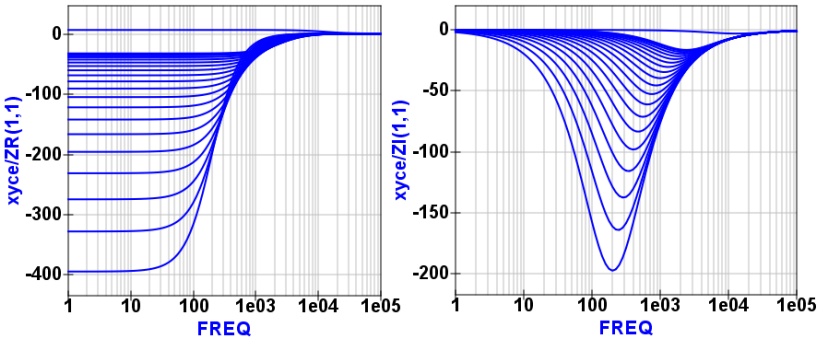
S parameter analysis



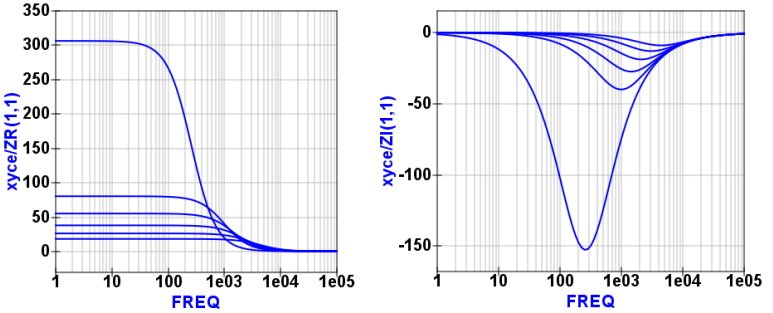
XYCE script

```
XYCESCR2
SpiceCode=
.global_param dcsweep=0.0
.ac dec 20 1.0 1e5
.step dcsweep 0.1 0.3 0.01
.lin format=touchstone sparcalc=1 lintype=z
.print ac format=std file=sparam.txt   dcsweep
+ zr(1,1) zi(1,1) zm(1,1) zp(1,1)
```

0.1V <= dcsweep <= 0.3V

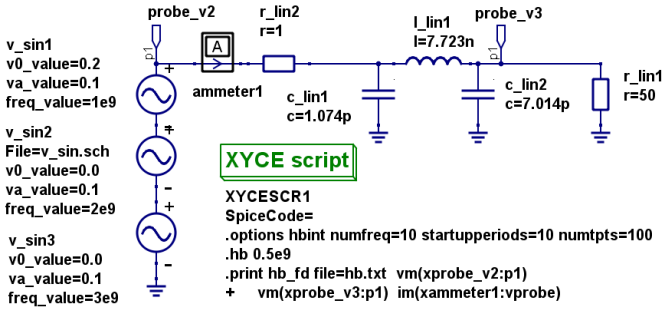


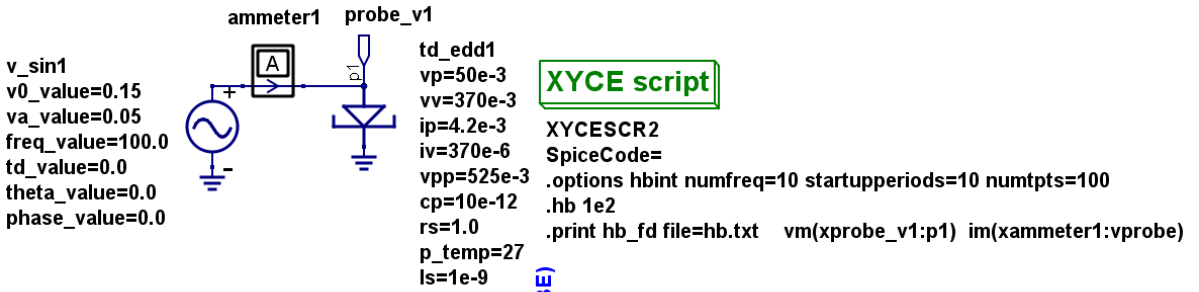
0.45V <= dcsweep <= 0.55V

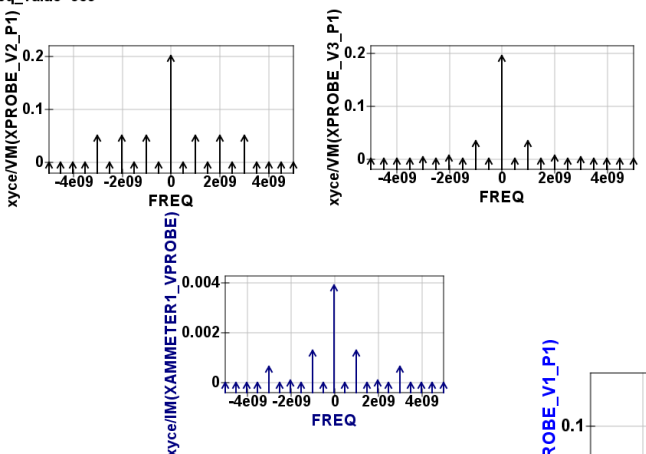


A modular approach to next generation Qucs: Evolving Qucs(Qt5)/Xyce simulation capabilities 2

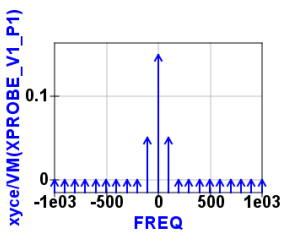
Harmonic Balance analysis

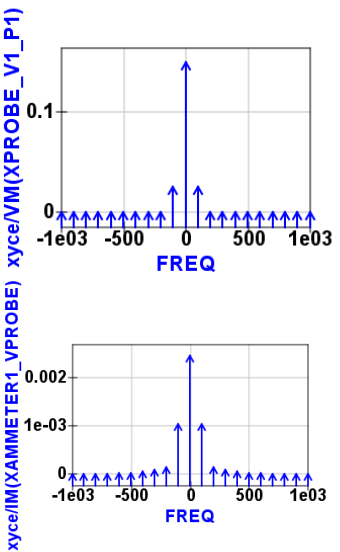




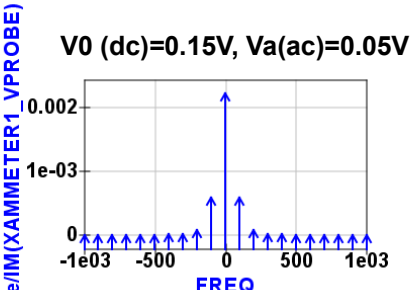


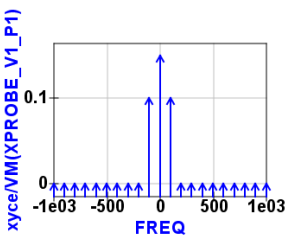
V0 (dc)=0.15V, Va(ac)=0.1V peak



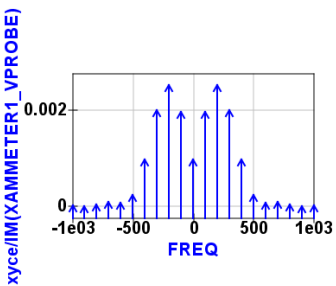


V0 (dc)=0.15V, Va(ac)=0.05V peak





V0 (dc)=0.15V, Va(ac)=0.2V peak



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 it's 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.