

Qucs Roadmap: background to the new features in release 0.0.18 and an outline of future software development directions

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

Richard Crozier, The University of Edinburgh, UK, richard.crozier@yahoo.co.uk.

Clemens Novak, Qucs Developer, clemens@familie-novak.net.

Bastien Roucaries, Laboratoire SATIE – CNRS UMR 8929, Université de Cergy-Pontoise, ENS Cachan, FR, bastien.roucaries@satie.ens-cauchan.fr.

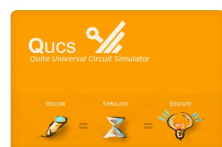
Frans Schreuder, Nikhef, Amsterdam, NL, fransschreuder@gmail.com.

Guilherme Brondani Torri, Qucs Developer, guitorri@gmail.com.

Plus contributions from Developers in the Qucs user community.

Listed in “Projects of the week”, September 8, 2014, by Sourceforge.

- **Qucs release 0.0.18: an overview of new simulation and compact device modelling features**
- **Compact model development with the Qucs/ADMS “Turn-key” modelling system**
- **Qucs experimental implementation of the BSIM 6.1 MOS transistor model**
- **Model parameter extraction: linking Qucs with low-cost measurement instrumentation; RF components**
- **Qucs swept Harmonic Balance simulation: Verilog-A and EDD diode models**
- **Current and possible future directions for Qucs development**



Qucs release 0.0.18: an overview of new simulation and compact device modelling features

Qucs 0.0.18 was released at the end of August 2014. It can be downloaded from the Sourceforge.net web site in binary and source code. The 0.0.18 release is available as binary ready to run code for the Linux (Ubuntu), MAC OSX (Darwin) and Windows (XP, 7, 8 and 8.1) operating systems.

Qucs 0.0.18 Release Notes

This release brings improvements to many parts of Qucs. The port from Qt3 to Qt4 is still ongoing and many bugs and usability issues have been fixed in the process. A few new features were added and others are in development. See below a short summary of the most noticeable enhancements. Qucs and Qucsator.

- **new:** Dynamic compilation and loading of Verilog-A modules (beta feature). Addition of a full ADMS/Qucs "turn key" Verilog-A compact device modelling system to Qucs. Users are no longer required to manually edit C++ code and the build system to be able to run Verilog-A models. Uses ADMS 2.3.4 (only a subset of Verilog-A is supported).
- **fix:** Spice file handling and translator.
- **new:** Addition of BSIM 3 and BSIM 4 models to Qucs MOS models.
- **new:** Release of the first model (RFresistor) in a new series of RF models for Harmonic Balance and other forms of Qucs simulation.
- **new:** HICUM npn and pnp Level 0 V1.3 Verilog-A devices.
- **new:** HICUM npn level 2 V2.32 Verilog-A device.



Qucs 0.0.18 release

Qucs

- **new/fix** : Ported several classes from Qt3 Support to Qt4 (work in progress).
- **new**: use QSettings to store user preferences and application settings.
- **new**: Copy/duplicate project content files.
- **new**: Export schematic and diagrams to SVG, PNG, JPG, JPEG, PDF, PDF+Latex, EPS (Inkscape required).
- **new**: Component library is not included on the Dock Window.
- **new**: Added checkbox to settings, which allows to load schematics from future versions.
- **new**: Open Recent menu option.
- **new**: Set ADMS, ASCO and Octave paths on the application settings.
- **new**: Use environment variables ADMSXMLBINDIR, ASCOBINDIR, OCTAVEBINDIR to set tools paths.
- **new**: Updated translations: pt-BR, pt-PT, ru.

Qucsator

- **new**: Refactoring of Qucsator namespace. Most of the core functions are under the namespace qucs::
- **new**: Allow to run autotest by adding bugon/assert operator.
- **new**: initial support for unit testing with googletest.
- **new**: added a Python scripts for parsing Qucs data files.
- **fix**: enable BJT models with ($V_{tf} < 0$).

QucsFilter, QucsAttenuator, QucsHelp, QucsEdit

- **new/fix** : Ported several classes from Qt3Support to Qt4 (work in progress)

QucsResCodes

- **new**: QucsResCodes is a program to compute color codes for resistors and resistance values from color codes. The program computes the closest standard resistor value. Users can paste the computed resistor in the schematic.



Qucs 0.0.18 release

Interfaces

- Improved Qucs/Octave interface for GUI and Qucssator netlist entry (including a start on a Octave library of plotting functions for Qucs post simulation data visualisation).
- First steps in developing a combined Qucs/Python Qucs simulation capability with high quality data visualisation using matplotlib. Applications include basic sensitivity analysis and circuit performance optimisation.
- Preliminary version of Qucs/Octave linked transient simulation capability.

Build system

- Add ADMS as a Git submodule to the Qucs Git repository.
- Added --disable-asco and --disable-adms option to skip the build of included sources.
- Added CMake (experimental) for advanced users building from Git repository.
- OSX: options to build against 10.5 to 10.9 SDK.
- Included packages: ASCO 0.4.9 (patched) and ADMS 2.3.4.

Miscellaneous

- Project website is now under version control.
- Source code is now under Travis continuous integration service.
- Source code documentation (Doxygen) available on the website.
- Preliminary regression and test suite (qucs-test).

• Bug fixes

- Hundreds of bug fixes and code improvements.



Compact model development with the Qucs/ADMS “Turn-key” modelling system: 1. Build Verilog-A module

Qucs 0.0.18 - Project: bsim6

File Edit Positioning Insert Project Tools Simulation View Help

bsim6MOD.va x

```
// *****  
// * BSIM Bulk MOSFET Model Equations (Verilog-A)  
// *****  
\`include "constants.vams"  
\`include "disciplines.vams"  
  
// Disable strobe for improved performance speed  
//\`define DISABLE_STROBE //To Use DISABLE_STROBE, Activate it here. Used Only  
\`ifdef DISABLE_STROBE  
  \`define STROBE(X)  
  \`define STROBE2(X,Y)  
\`else  
  \`define STROBE(X) $strobe(X)  
  \`define STROBE2(X,Y) $strobe(X,Y)  
\`endif  
  
// Not all Verilog-A compilers are able to properly collapse  
// internal nodes by shorting branches. To ensure minimal node  
// count, comment out the following lines:  
  
//\`define __RDSMOD__  
//\`define __TNOISW__  
//\`define __RGATEMOD__  
//\`define __RBODYMOD__  
//\`define __SHMOD__  
  
// Normalized pinch-off voltage including PD  
\`define PO_psisip(vg_vfb,gamma,DPD,phif,psip) \  
  T1 = 1.0 + DPD; \  
  vgfbPD = vg_vfb/T1; \  
  gammaPD = gamma/T1; \  
  T1 = 0.5*vgfbPD - 3.0*(1.0 + gammaPD/`M_SQRT2); \  
  T2 = T1 + sqrt(T1*T1 + 6.0*vgfbPD); \  
  if (vgfbPD < 0.0) begin \  
    T3 = (vgfbPD - T2) / gammaPD; \  
  end
```

[warning] please ensure extra code to be added to the interface
[info...] last argument 'dyload' provided, emitting code for the dynamic loader
[info...] bsim6MOD.core.cpp and bsim6MOD.core.h: files created
[info...] bsim6MOD.defs.h: file created
[info...] bsim6MOD.analogfunction.h created
[info...] bsim6MOD.analogfunction.cpp created
[info...] elapsed time: 5 (second)
[info...] admst iterations: 9794271 (9410697 freed)

admsXml Compiler

Select from Project drop-down menu

“Build Verilog-A module”

ADMS compiles XXX.va to C++ code

C++ code compiled and linked to Qucs

Compact model development with the Qucs/ADMS “Turn-key” modelling system: 2. Edit text symbol

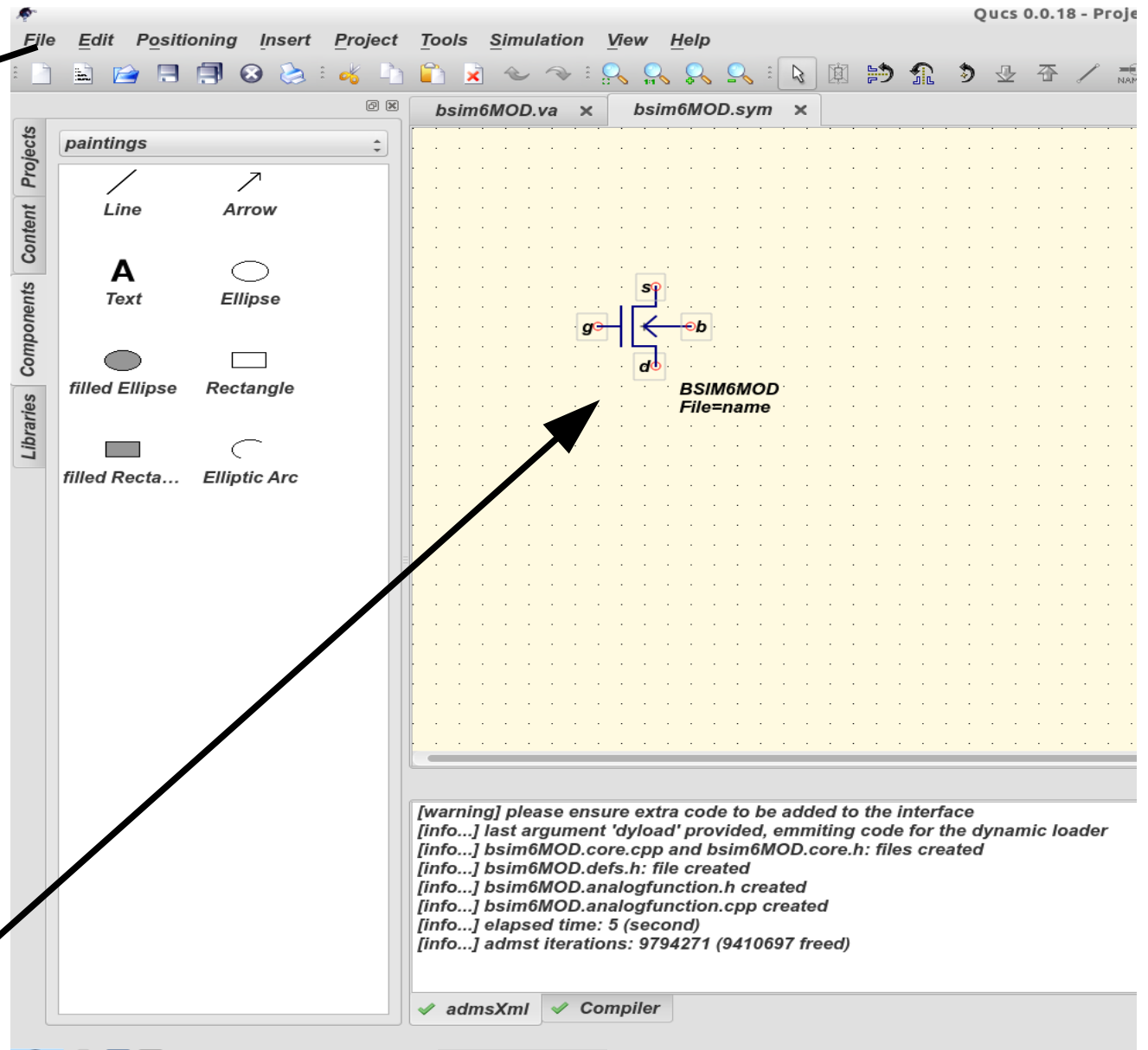
Select from File drop-down menu [with xxx.va code displayed]

“Edit text symbol”

Qucs displays symbol template

EDIT

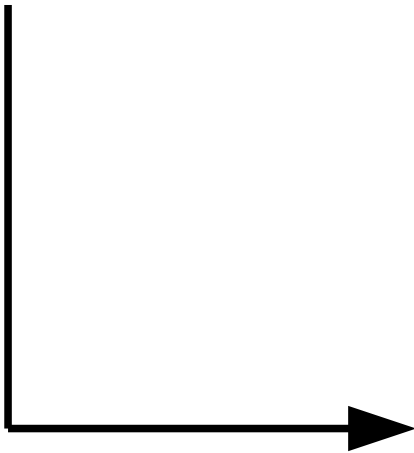
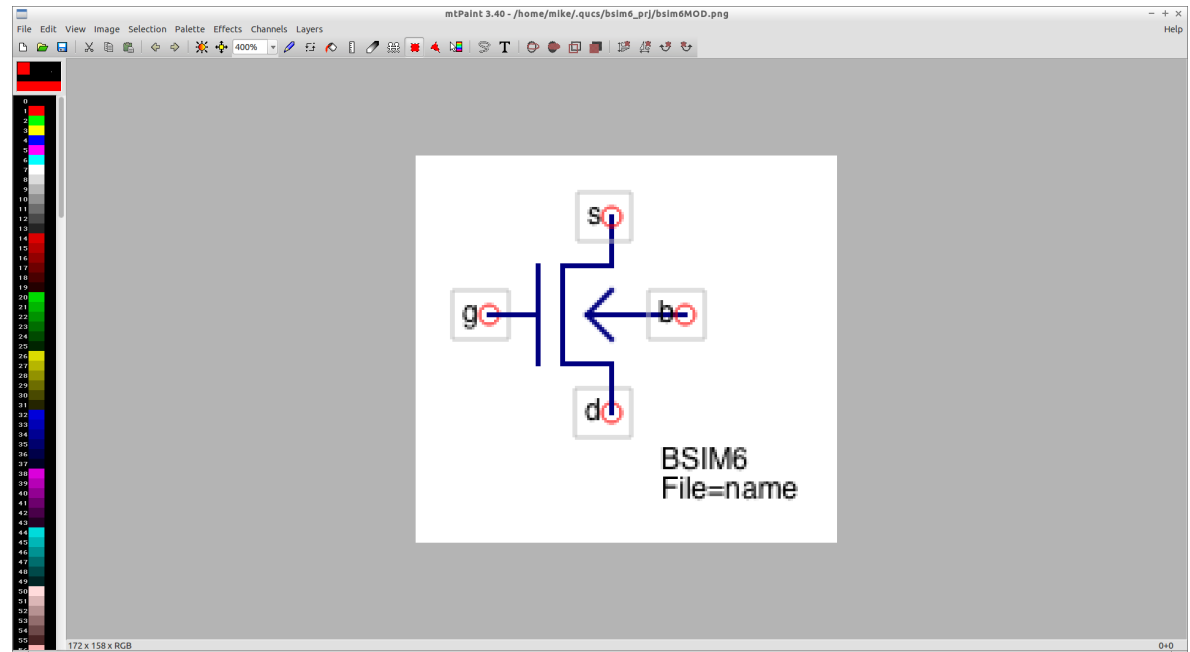
Then save to current project



Compact model development with the Qucs/ADMS “Turn-key” modelling system: 3. Generate and add icon to current project

Generate icon bitmap file

XXXX.png



Save XXXX.png file in working project,
For example BSIM6MOD_prj



Compact model development with the Qucs/ADMS “Turn-key” modelling system: 4. Load new model before use

Select from Project drop-down menu
↓
“Load Verilog-A module”
↓
Load Verilog-A Symbol file(s)

The screenshot displays the Qucs 0.0.18-Pr interface. On the left, the 'Libraries' panel shows a list of components under 'lumped components'. A black arrow points from the 'Project' drop-down menu to the 'Load Verilog-A symbols' dialog box. The dialog box is titled 'Load Verilog-A symbols' and contains a list of files with 'bsim6MOD_symbol.json' selected. A preview window shows a circuit symbol for a BSim6 model. Below the list are buttons for 'Select All', 'Deselect All', 'Cancel', and 'Ok'. A status bar at the bottom shows 'admsXml' and 'Compiler' with green checkmarks. On the right, the 'Libraries' panel shows 'verilog-a user devices' with 'bsim6MOD' listed. A black arrow points from the 'bsim6MOD' component to the 'Load Verilog-A symbols' dialog box. A warning message is visible at the bottom right of the interface.

```
[warning] please ensure extra code to be added to the interface
[info...] last argument 'dyload' provided, emitting code for the dynamic loader
[info...] bsim6MOD.core.cpp and bsim6MOD.core.h: files created
[info...] bsim6MOD.defs.h: file created
[info...] bsim6MOD.analogfunction.h created
[info...] bsim6MOD.analogfunction.cpp created
[info...] elapsed time: 5 (second)
[info...] admst iterations: 9794271 (9410697 freed)
```

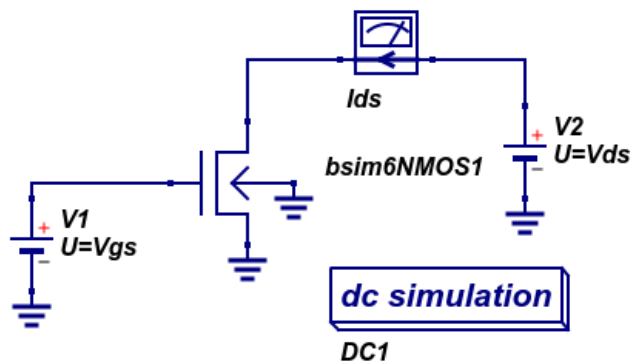


Qucs experimental implementation of the BSIM 6.1 MOS transistor model

The Qucs Development Team has taken over maintaining the LGPL version of ADMS:
Latest version is ADMS 2.3.4; <http://sourceforge.net/projects/mot-adms/>
[Includes contributions from Xyce and WRspice ADMS patches]
Source code repository has migrated to: <https://github.com/Qucs/ADMS>
Download : git clone <https://github.com/Qucs/ADMS.git>

BSIM 6.1 compiles with ADMS 2.3.4 : Roughly 4000 lines of Verilog-A code generates around 27,000 lines of C++ code for compiling and linking to Qucs 0.0.18

Examples of initial test results for the Qucs version of a BSIM 6.1 MOS n and p devices: the new implementation is currently under test and will be published once the Qucs “Development Team” is satisfied that it functions correctly and accurately. Likely publication will be with the release of the next Qucs snapshot.

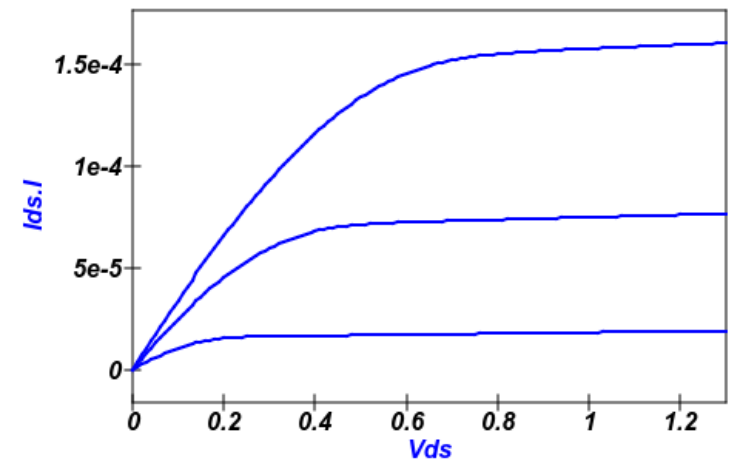


Parameter sweep

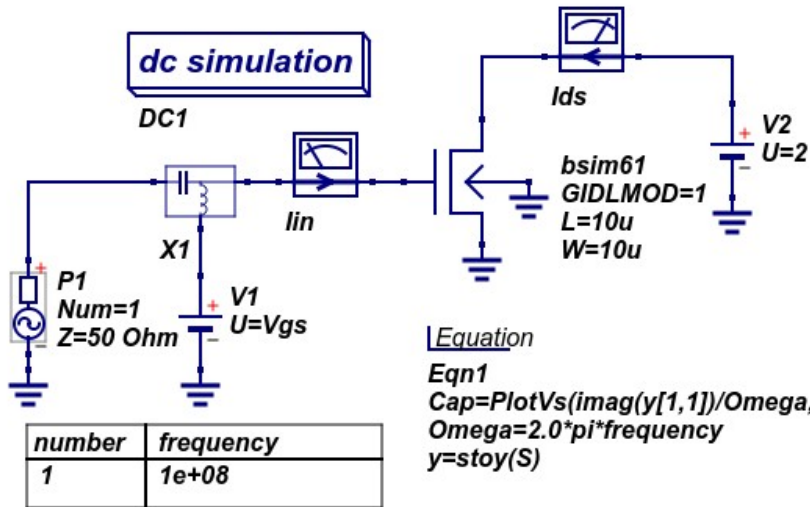
SW2
Sim=SW1
Type=lin
Param=Vgs
Start=0.4
Stop=1.0
Points=3

Parameter sweep

SW1
Sim=DC1
Type=lin
Param=Vds
Start=0
Stop=1.3
Points=131



Qucs experimental implementation of the BSIM 6.1 MOS transistor model : continued



number	frequency
1	1e+08

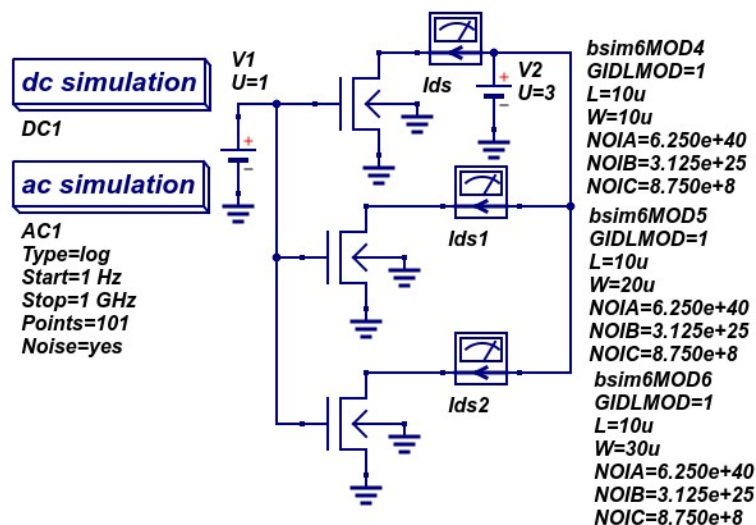
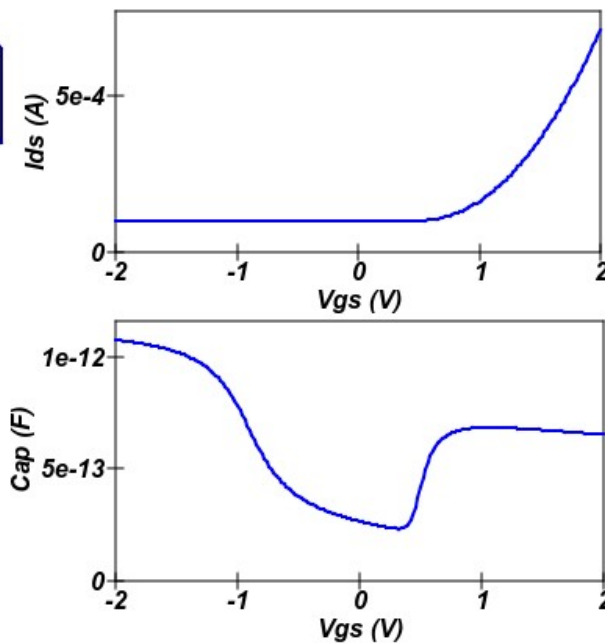
Equation
 Eqn1
 $Cap = PlotVs(imag(y[1,1])/Omega, Vgs)$
 $Omega = 2.0 * pi * frequency$
 $y = stoy(S)$

S parameter simulation

SP1
 Type=const
 Values=[1e8]

Parameter sweep

SW2
 Sim=SP1
 Type=lin
 Param=Vgs
 Start=-2
 Stop=2
 Points=201



dc simulation

DC1

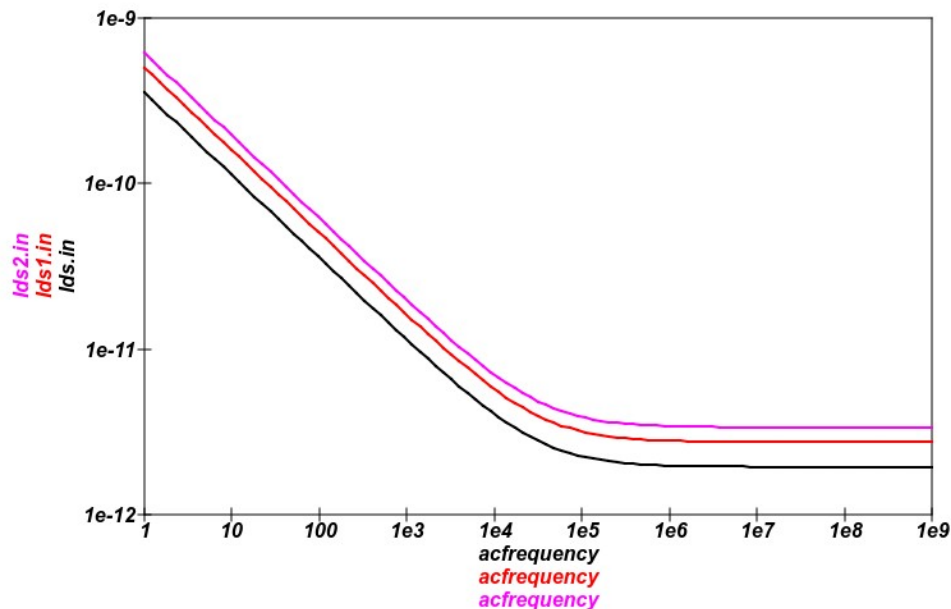
ac simulation

AC1
 Type=log
 Start=1 Hz
 Stop=1 GHz
 Points=101
 Noise=yes

bsim6MOD4
 GIDLMOD=1
 L=10u
 W=10u
 NOIA=6.250e+40
 NOIB=3.125e+25
 NOIC=8.750e+8

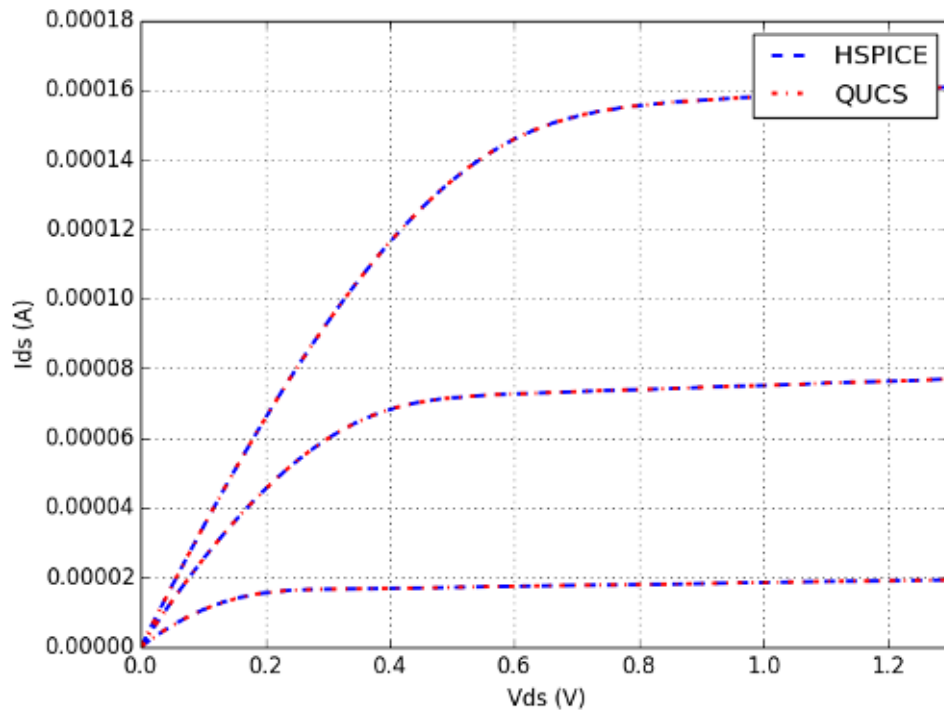
bsim6MOD5
 GIDLMOD=1
 L=10u
 W=20u
 NOIA=6.250e+40
 NOIB=3.125e+25
 NOIC=8.750e+8

bsim6MOD6
 GIDLMOD=1
 L=10u
 W=30u
 NOIA=6.250e+40
 NOIB=3.125e+25
 NOIC=8.750e+8

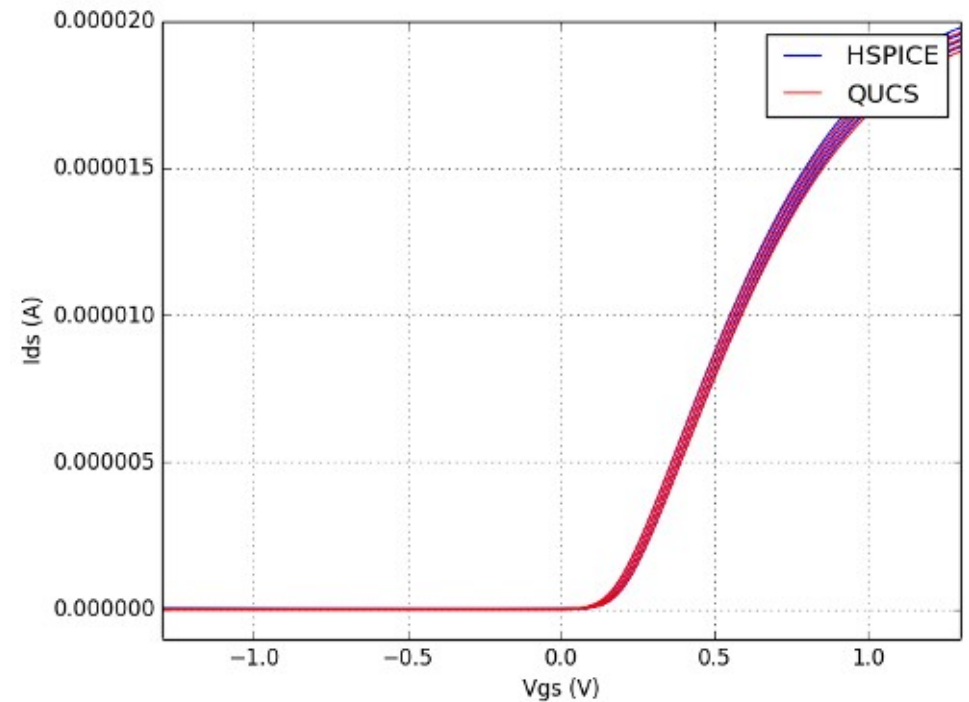


Qucs experimental implementation of the BSIM 6.1 MOS transistor model : continued

Qucs/ HSPICE: comparison of simulated nMOS I/V characteristics



I_{ds} (A) against V_{ds} (V)



I_{ds} (A) against V_{gs} (V)



Qucs experimental implementation of the BSIM 6.1 MOS transistor model : continued

Qucs/ HSPICE: comparison of simulated pMOS I/V characteristics

dc simulation

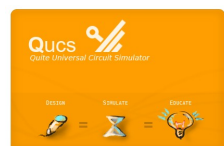
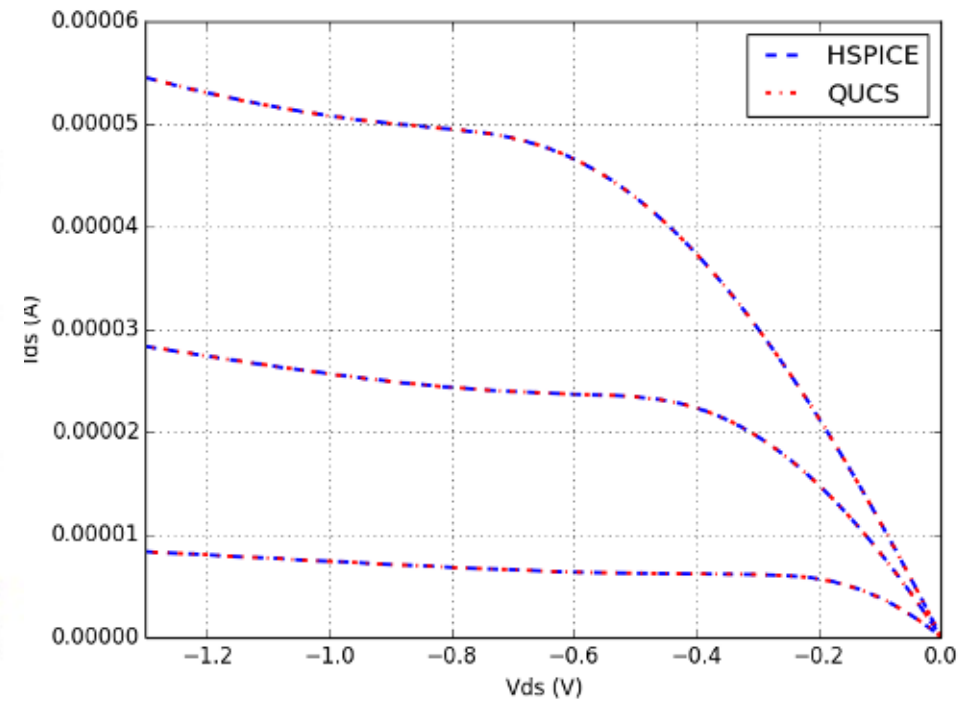
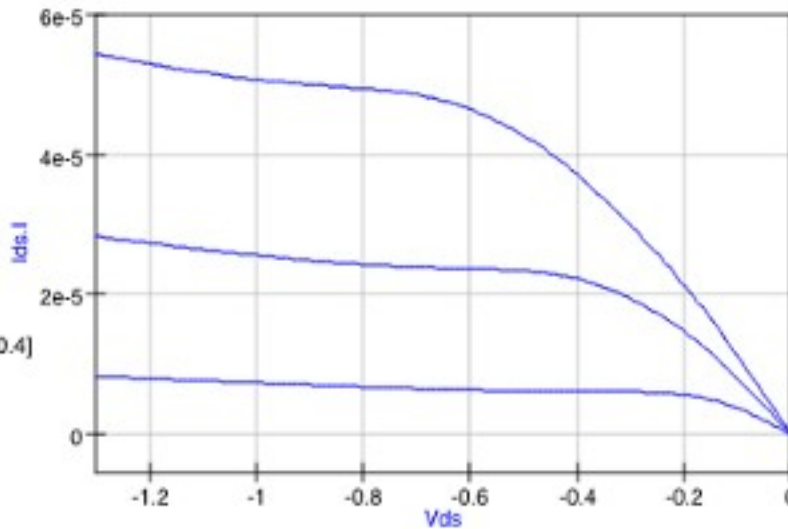
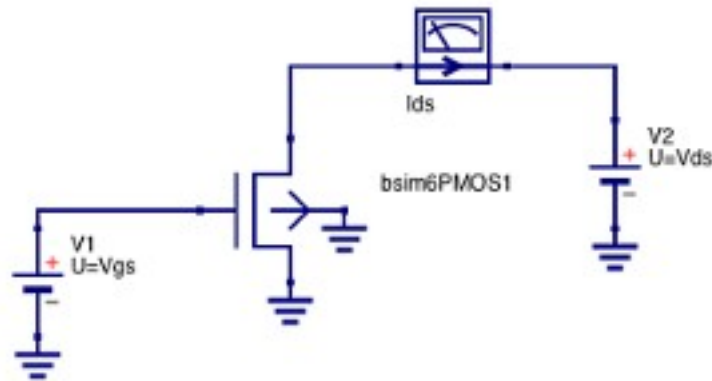
DC1

Parameter sweep

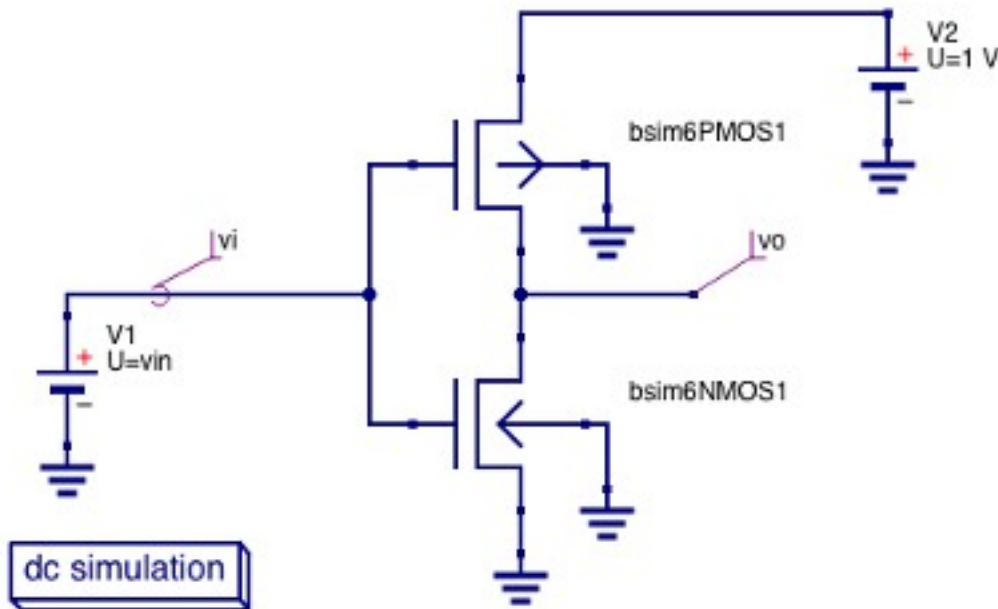
SW1
Sim=DC1
Type=In
Param=Vds
Start=0
Stop=-1.3
Points=131

Parameter sweep

SW2
Sim=SW1
Type=lst
Param=Vgs
Values=[-1.0; -0.7; -0.4]



Qucs experimental implementation of the BSIM 6.1 MOS transistor model : continued

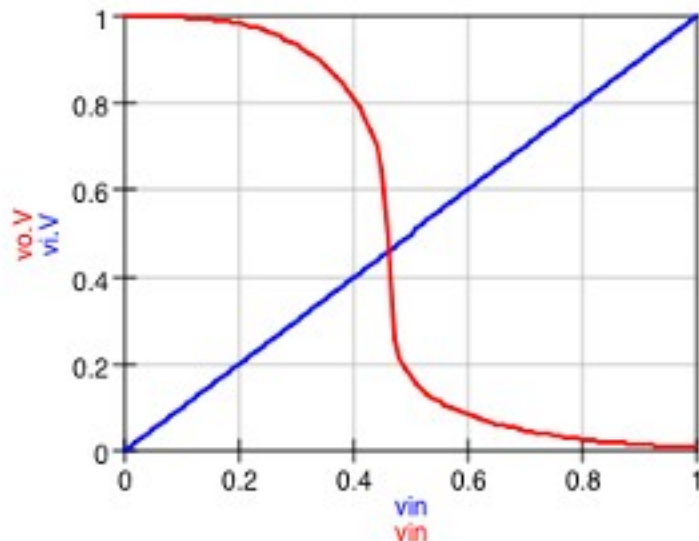


dc simulation

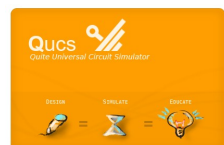
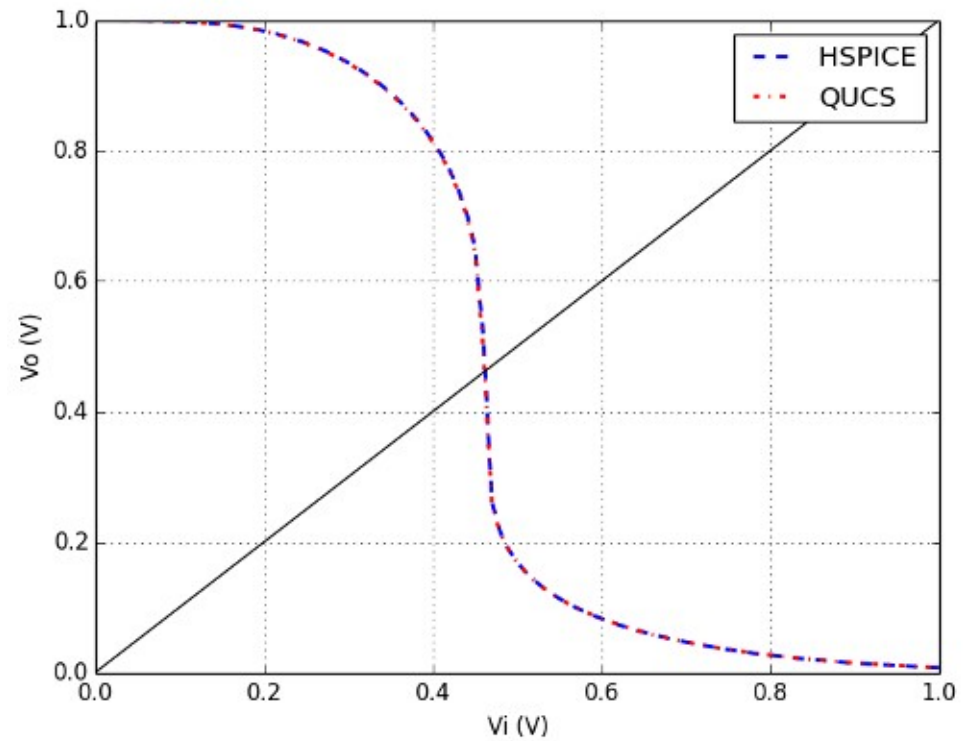
DC1

Parameter sweep

SW1
Sim=DC1
Type=lin
Param=vin
Start=0
Stop=1
Points=101

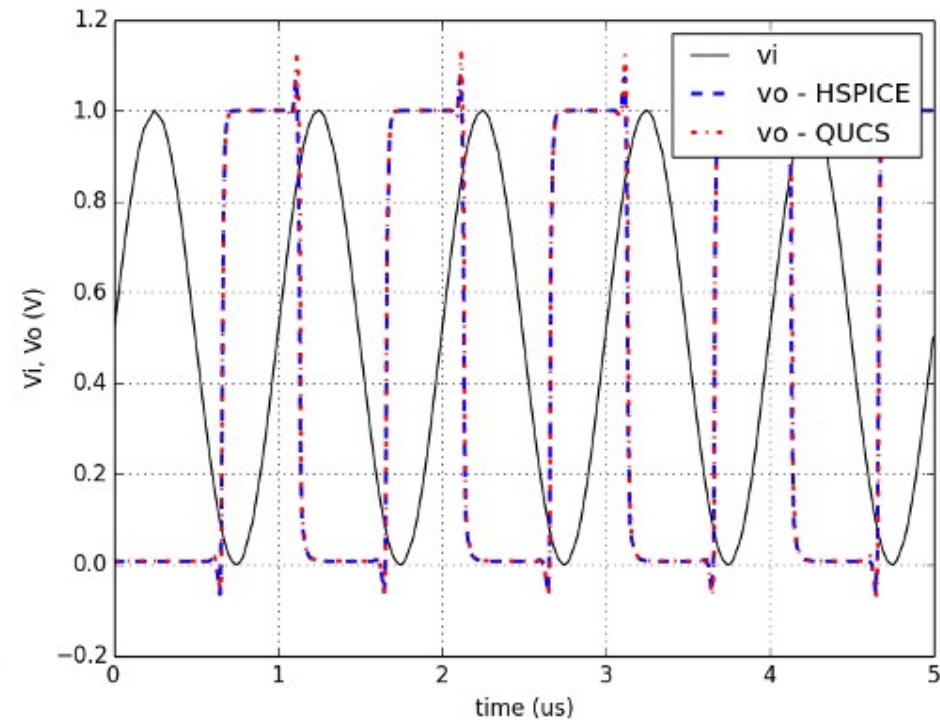
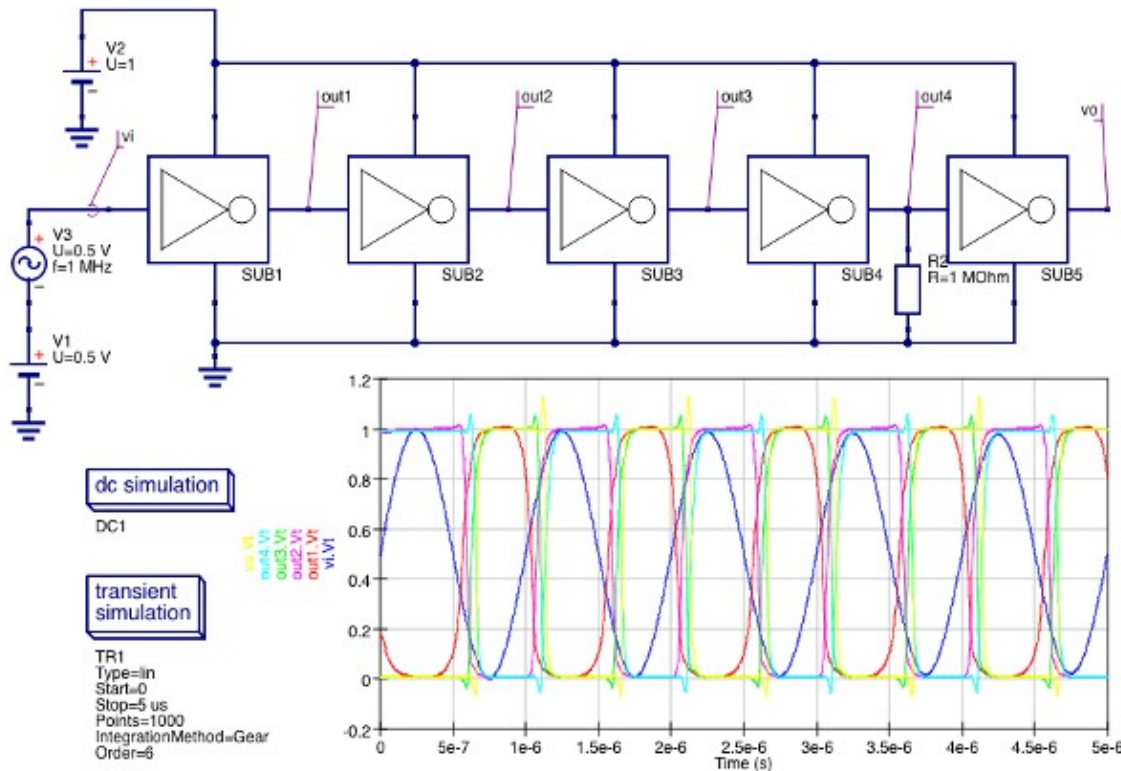


Qucs/ HSPICE: comparison of simulated CMOS inverter characteristics



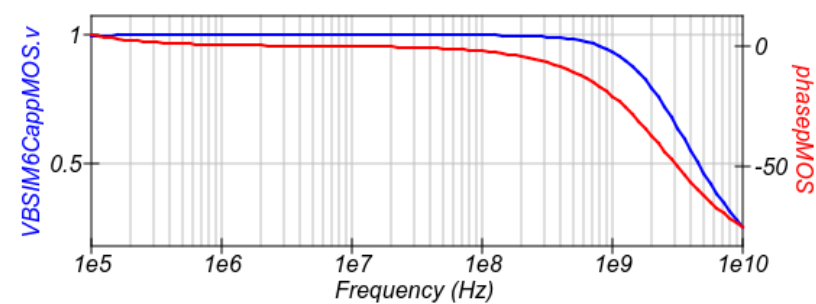
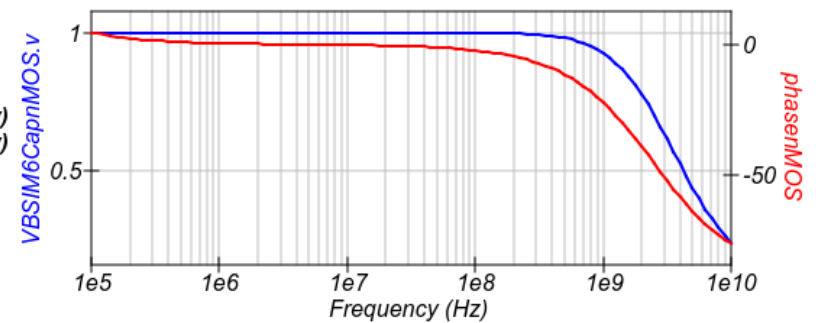
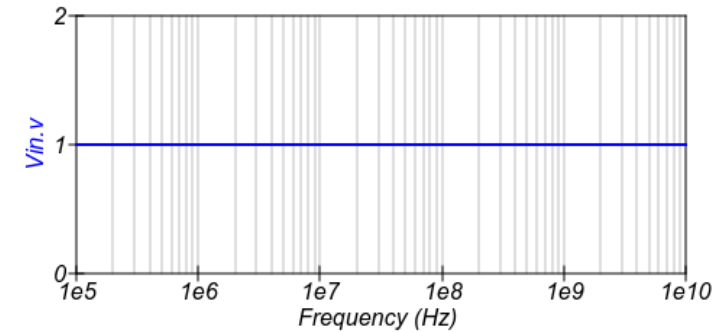
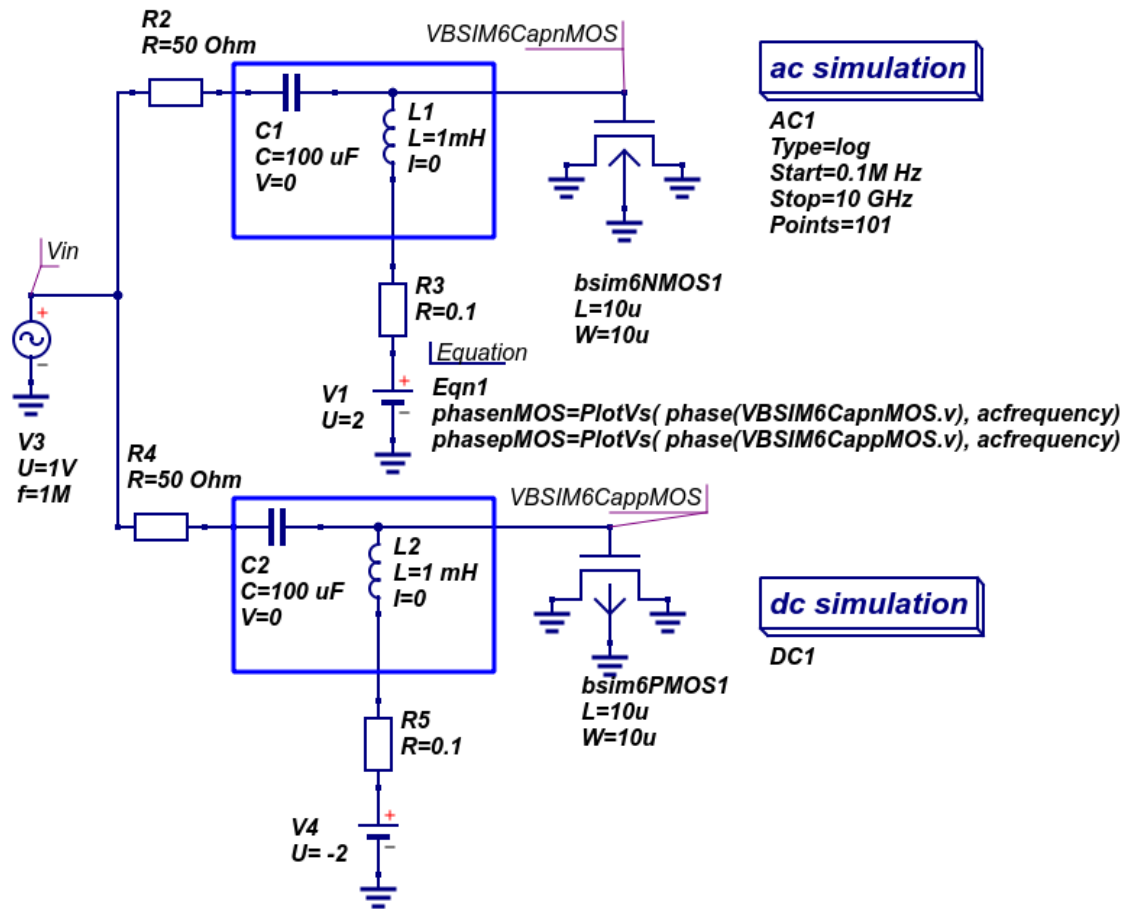
Qucs experimental implementation of the BSIM 6.1 MOS transistor model : continued

Qucs/ HSPICE: comparison of simulated CMOS inverter transient characteristics

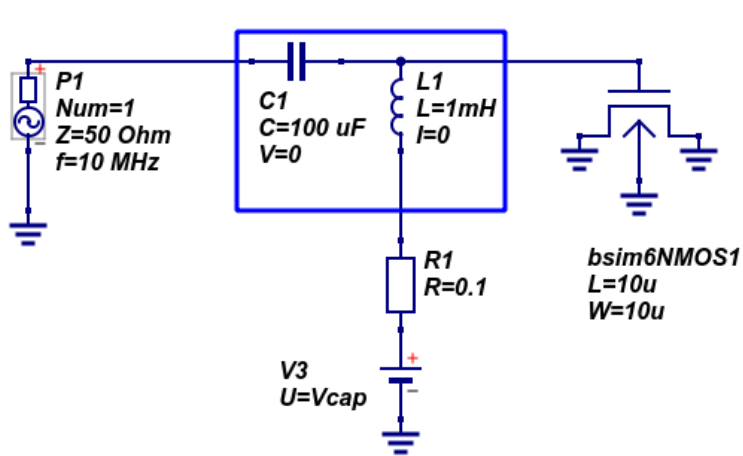


NOTE: resistor R2 has been added to the original HSPICE test circuit to aid DC convergence.

Qucs experimental implementation of the BSIM 6.1 MOS transistor model : more examples; AC small signal simulation



Qucs experimental implementation of the BSIM 6.1 MOS transistor model : more examples; AC small signal S parameter simulation



Equation

```
Eqn2
y=stoy(S)
Omega=2*pi*frequency
Capn=PlotVs(imag(y[1,1]/Omega), Vcap)
Capp=PlotVs(imag(y[2,2]/Omega), Vcap)
```

dc simulation

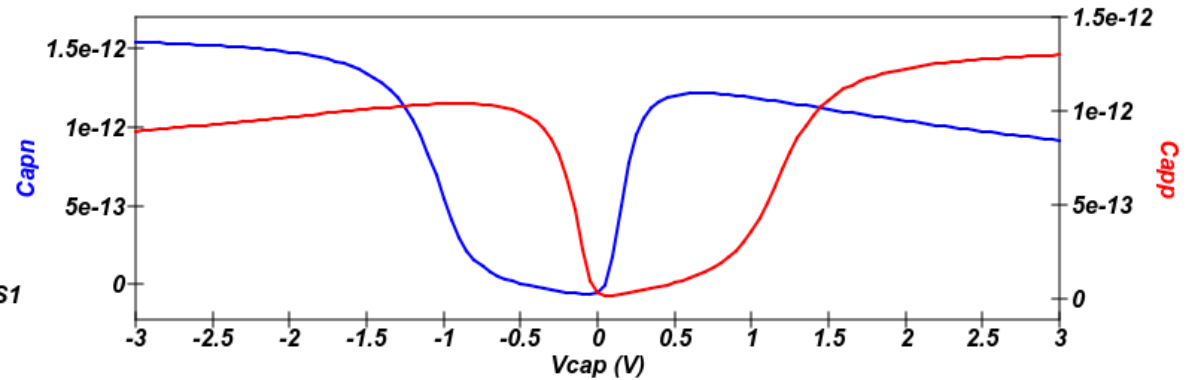
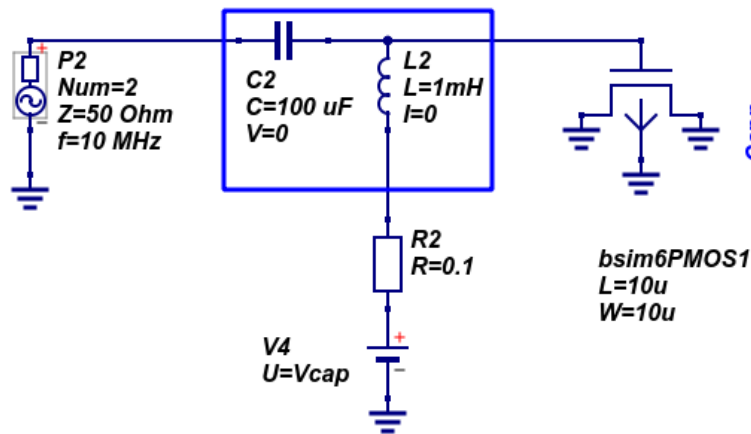
DC1

S parameter simulation

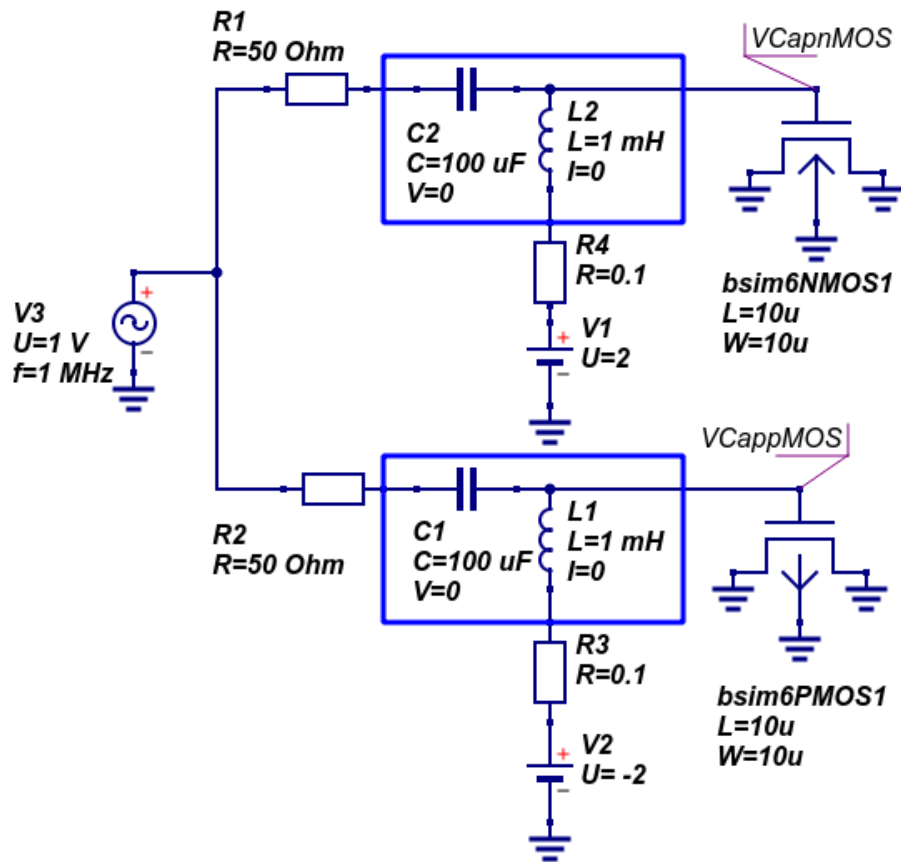
SP1
Type=const
Values=[10 M]

Parameter sweep

SW1
Sim=SP1
Type=lin
Param=Vcap
Start=-3
Stop=+3
Points=121



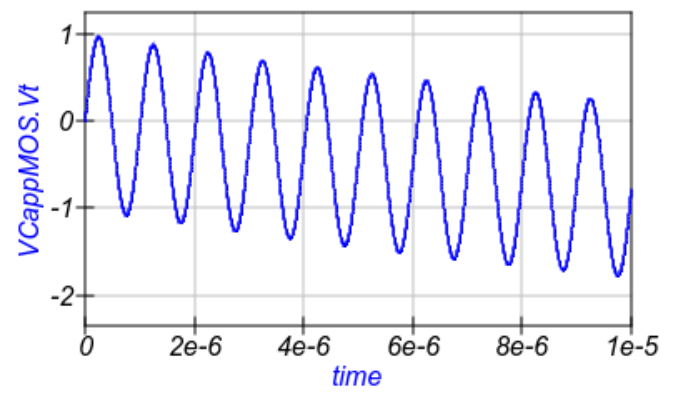
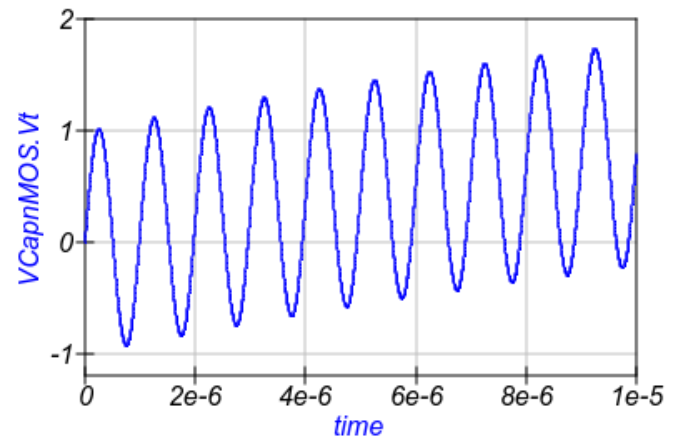
Qucs experimental implementation of the BSIM 6.1 MOS transistor model : more examples; Transient simulation



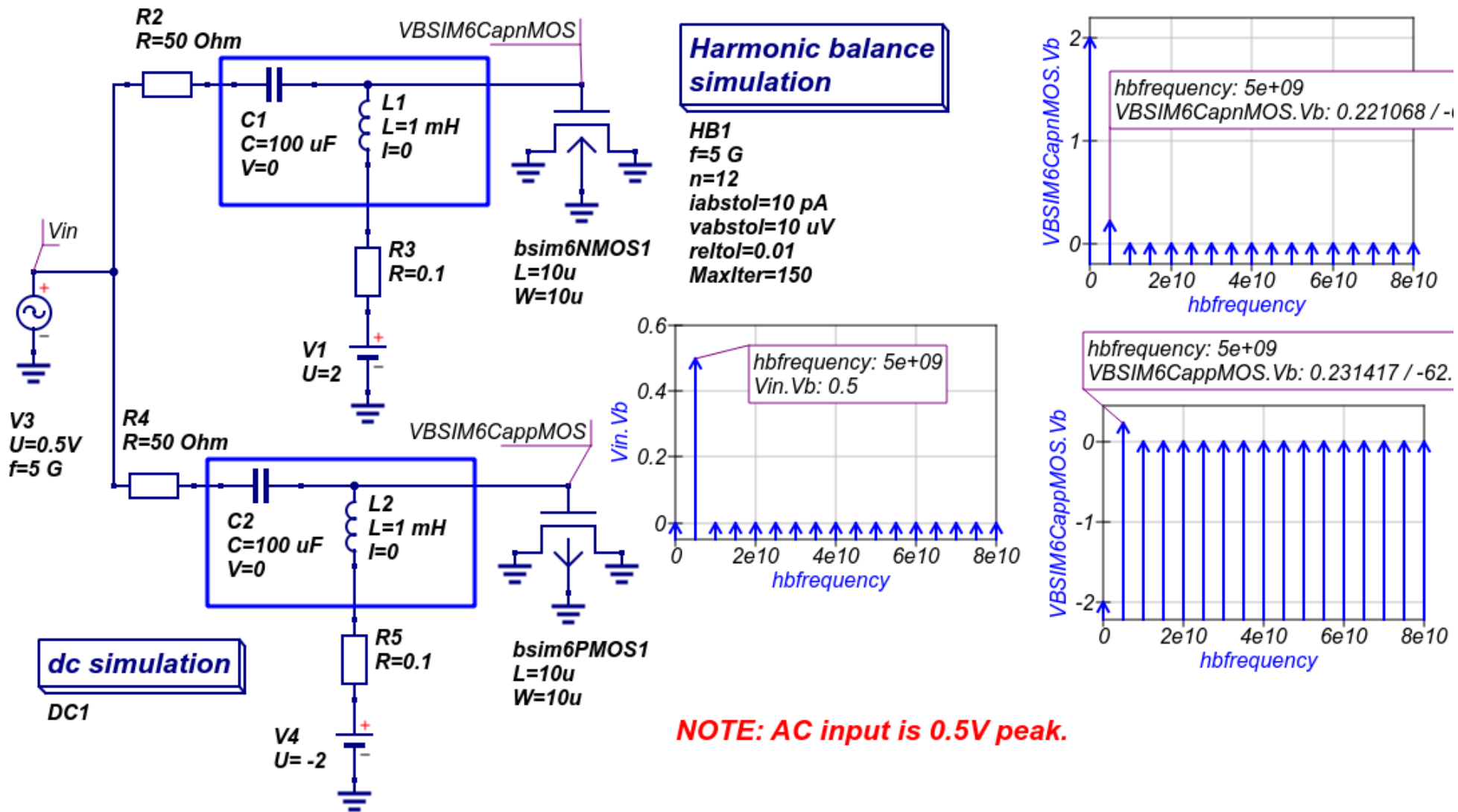
dc simulation
DC1

transient simulation
TR1

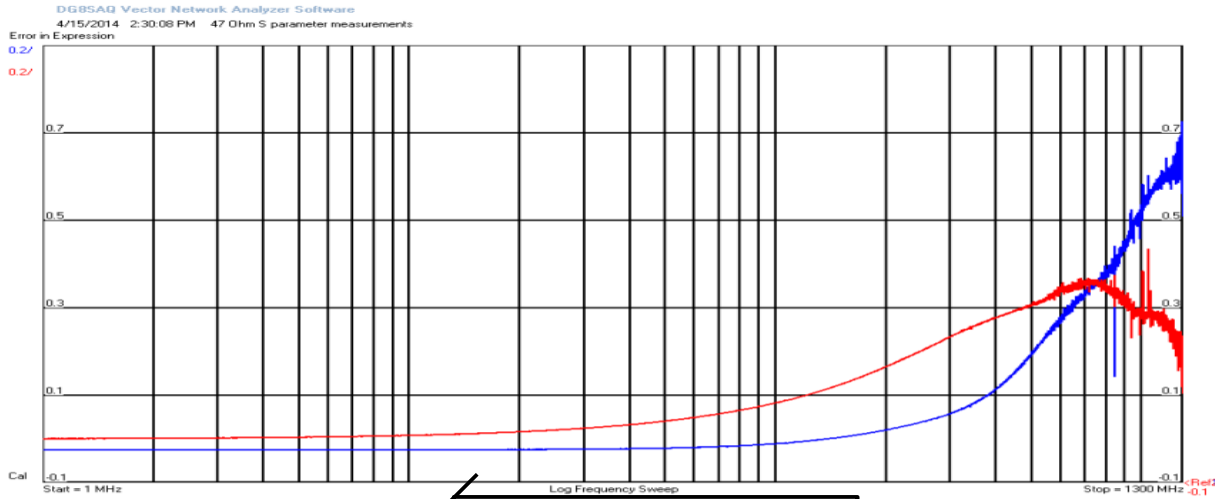
Type=lin
Start=0
Stop=10 us
IntegrationMethod=Gear
Order=4



Qucs experimental implementation of the BSIM 6.1 MOS transistor model : more examples; Harmonic Balance simulation



Model parameter extraction: linking Qucs with low-cost measurement instrumentation; RF components



Measured S parameters

Qucs device model parameter extraction system: Mike Harrison, April 2014 – RF RESISTOR MODEL Version 0.0.4

Optimization

Opt1

Equation

Eqn3

$$Cf1=A*(\text{real}(S[2,2])-\text{real}(S[1,1]))^2$$

$$Cf2=B*(\text{imag}(S[2,2])-\text{imag}(S[1,1]))^2$$

S parameter simulation

SP1

Type=log

Start=1e6

Stop=1.3e9

Points=1501

Simulation Controls

Test circuits

Measurements

Simulation

Equation

Eqn1	Eqn4	Eqn5
Zr=real(stoz(S))	A=1	Sr_1=real(S[1,1])
Zi=imag(stoz(S))	Rm=47.3	Si_1=imag(S[1,1])
	B=1	Sr_2=real(S[2,2])
		Si_2=imag(S[2,2])

Optimized values of model parameters are listed by displaying Optimization icon menu "Variables"

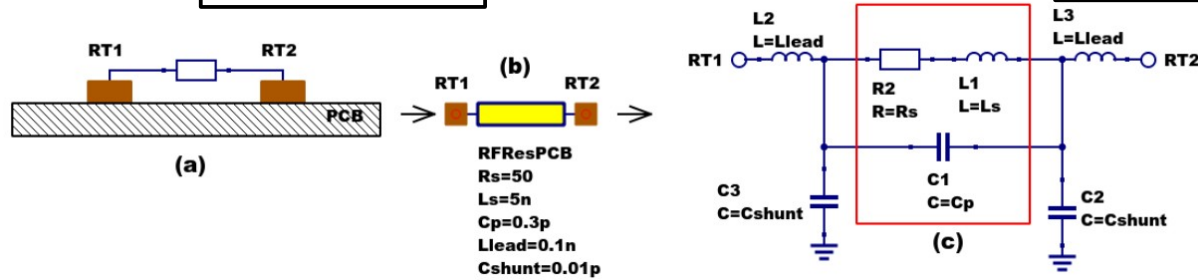
ASCO optimization

Verilog-A model

```

include "disciplines.vams"
include "constants.vams"
module RFResPCB(RT1, RT2);
inout RT1, RT2; electrical n1, n2, n3, nx, ny, nz;
define attr(txt) (*txt*)
parameter real Rs = 50 from [1e-20 : inf]
  attr(info="RF resistance" unit="Ohm's");
parameter real Cp = 0.3e-12 from [0 : inf]
  attr(info="Resistor shunt capacitance" unit="F");
parameter real Ls = 8.5e-9 from [1e-20 : inf]
  attr(info="Series inductance" unit="H");
parameter real Llead = 0.1e-9 from [1e-20 : inf]
  attr(info="Parasitic lead inductance" unit="H");
parameter real Cshunt = 1e-10 from [1e-20 : inf]
  attr(info="Parasitic shunt capacitance" unit="F");
parameter real Tc1 = 0.0 from [-100 : 100]
  attr(info="First order temperature coefficient" unit="Ohm/Celsius");
parameter real Tc2 = 0.0 from [-100 : 100]
  attr(info="Second order temperature coefficient" unit="(Ohm/Celsius)^2");
parameter real Tnom = 26.85 from [-273.15 : 300]
  attr(info="Parameter extraction temperature" unit="Celsius");
parameter real Temp = 26.85 from [-273.15 : 300]
  attr(info="Simulation temperature" unit="Celsius");
branch (RT1, n1) bRT1n1; branch (n1, n2) bn1n2;
branch (n1, n3) bn1n3; branch (n2, n3) bn2n3;
branch (n3, RT2) bn3RT2;
real Rst, FourKT, n, Tdiff, Rn;
analog begin
@ (initial_model) begin
  Tdiff = Temp-Tnom; FourKT = 4.0*P_K*Temp;
  Rst = Rs*(1.0+Tc1*Tdiff+Tc2*Tdiff*Tdiff);
  Rn = FourKT/Rst;
end
I(n1) <+ ddt(Cshunt*V(n1)); I(bn1n2) <+ V(bn1n2)/Rst;
I(bn1n3) <+ ddt(Cp*V(bn1n3)); I(n3) <+ ddt(Cshunt*V(n3));
I(bRT1n1) <+ -V(nx); I(nx) <+ V(bRT1n1);
I(nx) <+ ddt(Llead*V(nx));
I(bn2n3) <+ -V(ny); I(ny)
<+ V(bn2n3); I(ny)
<+ ddt(Ls*V(ny)); I(bn3RT2) <+ -V(nz);
I(nz) <+ V(bn3RT2);
I(nz)n<+ ddt(Llead*V(nz));
I(bn1n2) <+ white_noise(Rn, "thermal");
end
endmodule
    
```

Verilog-A model code

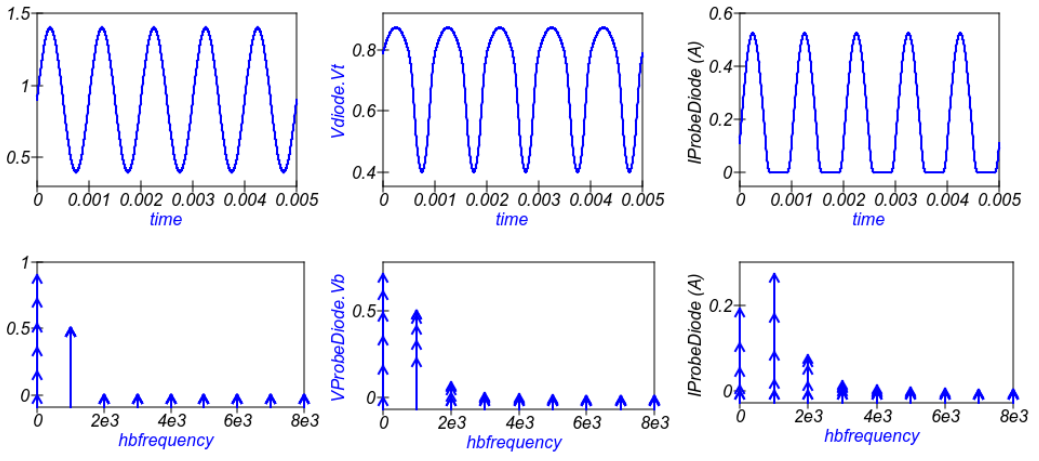
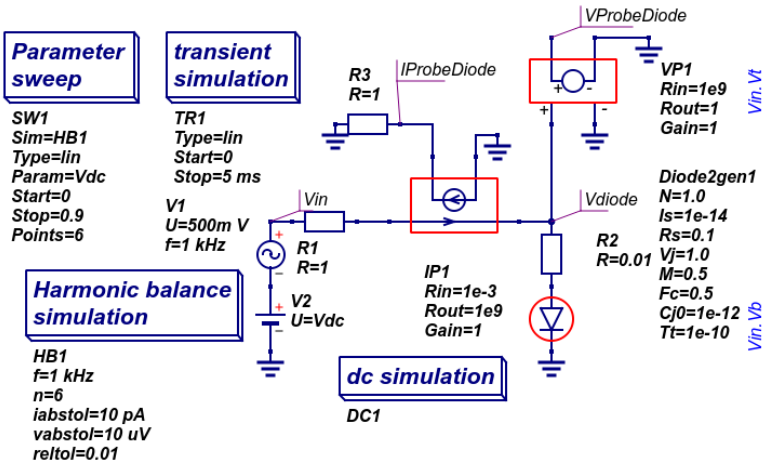


DG8SAQ VNWA 3 & 3E- Vector Network Analysers, SDR Kits Limited, Grangeside Business Centre, 129 Devizes Road, Trowbridge, Wilts BA14-7sZ, United Kingdom, 2014. www.SDR-Kits.net.

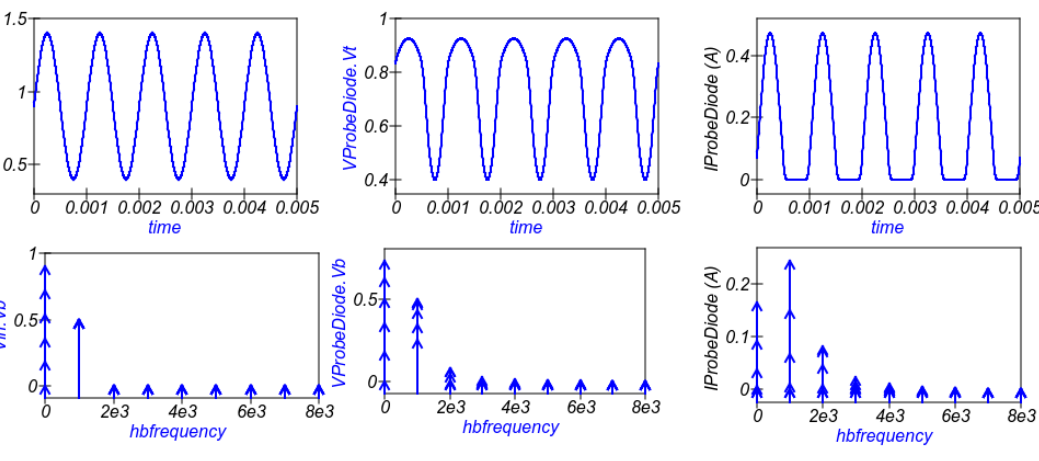
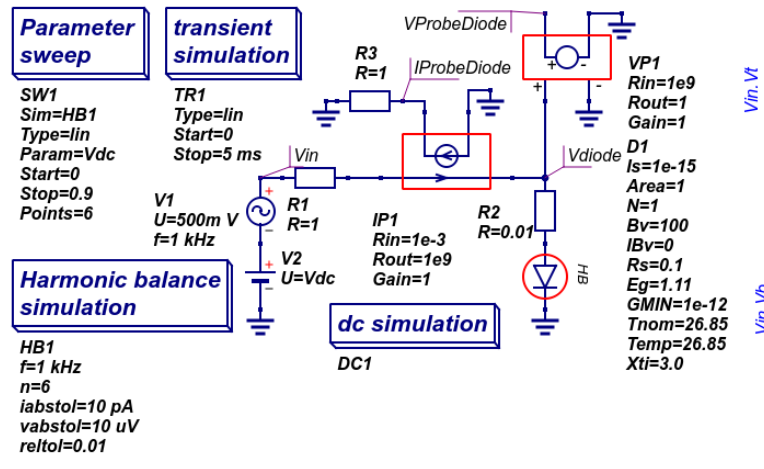


Qucs swept Harmonic Balance simulation: Verilog-A and EDD diode models

Verilog-A diode model



EDD diode model



Current and possible future directions for Qucs development

Current projects

- Finish Qt3 to Qt4 port.
- Continue development of the Qucs “Turn-key” Verilog-A modelling system.
- Improve Qucs simulation engines: particularly transient and Harmonic Balance simulation code.
- Improve the Qucs/Octave interface.
- Improve the Qucs/Python interface and continue work on high quality post simulation data visualisation capabilities.
- Add more industrial standard compact semiconductor device models to Qucs.
- Continue development of Qucs RF component models and macromodels.
- Continue work on device model parameter extraction.

Future projects

- Collect ideas from (1) the Qucs development team and (2) the Qucs user community.
- Update the Qucs project roadmap as a guide to future developments.
- Implement agreed roadmap in a structured way.

