Building device models and circuit macromodels with the Qucs GPL circuit simulator: A demonstration

M.E. Brinson: Faculty of Computing, London Metropolitan University, UK; Qucs development team: mbrin72043@yahoo.co.uk

S. Jahn: Qucs project manager, Munich, Germany; stefan@lkcc.org

- Background
- Introduction to circuit simulation with Qucs
- Building a compact semiconductor model of an npn bipolar phototransistor
- Compiling and linking Verilog-A code to Qucs
- Summary

Presented on April 2, 2009, at the COMON meeting, IHP, Frankfurt (Oder), Germany
Background

• Qucs is an open source circuit simulator with a graphical user interface (GUI) based on QT® by Trolltech®
• Qucs is available from http://qucs.sourceforge.net/
• Qucs aims to support a wide range of circuit simulation, including DC, AC, transient, S-parameter, noise analysis, Harmonic Balance analysis and digital system simulation
• Developed using GNU/Linux under the General Public License (GPL)
• Available for most of the popular computer operating systems, including GNU/Linux, Windows®, Solaris®, NetBSD, FreeBSD and MacOS®
• Qucs is one of the GPL circuit simulators taking part in the MOS-AK Verilog-A standardization initiative
• Developers: Loose group of international engineers, scientists and programmers
• Qucs is multilingual and has been translated into: Romanian, German, Italian, Polish, French, Portuguese, Spanish, Japanese, Hungarian, Hebrew, Swedish, Turkish, Russian, Czech and Catalan
Introduction to circuit simulation with Qucs

Qucs current versions: production 0.0.14; development 0.0.15

Start Qucs by issuing the command `# qucs`
OR by clicking the appropriate icon on your start menu or desktop

File -> Application settings
Open projects folders

Drawing tool bar
Drop-down menus
Schematic circuit drawing window
Introduction to circuit simulation with Qucs

Tool suite: Qucs consists of a group of standalone programs which interact with each other through a graphical user interface (GUI)

- **the GUI**: used to create schematics, setup simulations, display simulation results and writing VHDL/Verilog code amongst other tasks
- **the analogue simulator**: a command line program which is run by the GUI to simulate user generated schematics
- **a simple text editor**: which is used to display netlists, simulation log files and to edit SPICE, Touchstone and other text files
- **a filter synthesis program**: used for the design of passive filters
- **a transmission line calculator**: used for the design and analysis of transmission lines
- **a component library**
- **an attenuator synthesis program**: used for the design of passive attenuators
- **a command line conversion program**: this program acts as a tool for the import and export of data sets, netlists and schematics from and to other CAD/EDA software

The GUI is also used by Qucs to launch other EDA tools: FreeHDL (http://www.freehdl.seul.org) for digital VHDL, Icurus Verilog (http://icarus.com/eda/verilog/) for digital Verilog and ASCO (http://asco.sourceforge.net/) for circuit optimization
Introduction to circuit simulation with Qucs

Setting up schematics

Either press the **New** button above the projects folder or use the menu entry **Project → New Project** and enter the project name.

Confirm the dialog by pressing **Create** button. Qucs then opens the Phototransistor project and displays a blank project window with the **Content** tab open.
Introduction to circuit simulation with Qucs
Building a simple schematic

- Component tab ON
- Empty drawing pad
- Drag and drop
- From component library
- Place components
- Built-in components
- Add simulation controls
- Wiring tool
- Position components, connect components and edit values
- Rotation tools
- Post simulation data processing
Introduction to circuit simulation with Qucs
Qucs built-in analogue components: 1

Lumped components

Sources

R1
R=50 Ohm
C1
C=1 pF

R2
R=50 Ohm
L1
L=1 nH

P1
Num=1

V1
U=1 V
I1
I=1 mA

V2
U=1 V
I2
I=1 mA

V3
u=1e-6
i=1e-6

V4
U1=0 V
U2=1 V
T1=0
T2=1 ms

V5
U1=0 V
U2=1 V
T1=0
T2=1 ms

V6
U=1 V
TH=1 ms
TL=1 ms

I4
I=1 mA
Num=1
Z=50 Ohm

SRC4
G=1 S

SRC5
i1=1e-6
i2=1e-6
C=0.5

SRC6
i1=1e-6
i2=1e-6
C=0.5

SRC7
i1=1e-6
i2=1e-6
C=0.5

V7
U=1 V

V8
U=1 V

V9
File=vfile.dat

I5
I=1 I
I2=1 A
T1=0
T2=1 ms

I6
I=1 I
I2=1 A
T1=0
T2=1 ms

Pr1

Pr2

RF1
RF2

RF

1
RF

1

2

2

X1
L=10 dB

X2
L=50 Ohm

X3
phi=90

X4
phi=90

X5
phi=90

X6
k=0.7071

phi=180
Introduction to circuit simulation with Qucs

Qucs built-in analogue components: 2

Non-linear components

<table>
<thead>
<tr>
<th>Component</th>
<th>Type</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>npn</td>
<td>Is=1e-16, Nf=1, Va=0, Bf=100</td>
</tr>
<tr>
<td>T2</td>
<td>npn</td>
<td>Is=1e-16, Nf=1, Va=0, Bf=100</td>
</tr>
<tr>
<td>T3</td>
<td>npn</td>
<td>Is=1e-16, Nf=1, Va=0, Bf=100</td>
</tr>
<tr>
<td>T4</td>
<td>pnp</td>
<td>Is=1e-16, Nf=1, Va=0, Bf=100</td>
</tr>
<tr>
<td>T5</td>
<td>nfet</td>
<td>Vt=0, Vt=0, Beta=1e-4, Lambda=0.0</td>
</tr>
<tr>
<td>T6</td>
<td>pft</td>
<td>Vt=0, Vt=0, Beta=1e-4, Lambda=0.0</td>
</tr>
<tr>
<td>T7</td>
<td></td>
<td>Vt=1.0 V, Kp=2e-5, Lambda=0.0</td>
</tr>
<tr>
<td>T8</td>
<td></td>
<td>Vt=1.0 V, Kp=2e-5, Lambda=0.0</td>
</tr>
<tr>
<td>T9</td>
<td></td>
<td>Vt=1.0 V, Kp=2e-5, Lambda=0.0</td>
</tr>
<tr>
<td>T10</td>
<td></td>
<td>Vt=1.0 V, Kp=2e-5, Lambda=0.0</td>
</tr>
<tr>
<td>T11</td>
<td></td>
<td>Vt=1.0 V, Kp=2e-5, Lambda=0.0</td>
</tr>
<tr>
<td>T12</td>
<td></td>
<td>Vt=1.0 V, Kp=2e-5, Lambda=0.0, G=1e6</td>
</tr>
</tbody>
</table>

Verilog-A components

<table>
<thead>
<tr>
<th>Component</th>
<th>Type</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>LA1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>T2</td>
<td>npn</td>
<td>Is=1e-16, Nf=1, Va=0, Bf=100</td>
</tr>
<tr>
<td>T4</td>
<td></td>
<td>POT1</td>
</tr>
<tr>
<td>T5</td>
<td>pnp</td>
<td>Is=1e-16, Nf=1, Va=0, Bf=100</td>
</tr>
<tr>
<td>M1</td>
<td>nmos</td>
<td>Is=1e-16, Nf=1, Va=0, Bf=100</td>
</tr>
<tr>
<td>M2</td>
<td>pmos</td>
<td>Is=1e-16, Nf=1, Va=0, Bf=100</td>
</tr>
<tr>
<td>T7</td>
<td>npn</td>
<td>Is=1e-16, Nf=1, Va=0, Bf=100</td>
</tr>
<tr>
<td>T8</td>
<td>pnp</td>
<td>Is=1e-16, Nf=1, Va=0, Bf=100</td>
</tr>
<tr>
<td>T9</td>
<td></td>
<td>PD1</td>
</tr>
</tbody>
</table>

File components

<table>
<thead>
<tr>
<th>Component</th>
<th>File</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>test.s1p</td>
</tr>
<tr>
<td>X2</td>
<td>test.s2p</td>
</tr>
<tr>
<td>X3</td>
<td>test.s3p</td>
</tr>
<tr>
<td>X4</td>
<td>test.s1p</td>
</tr>
</tbody>
</table>

Qucs built-in analogue components: 2
Introduction to circuit simulation with Qucs

Qucs built-in analogue components: 3

- Line1
  Z=50 Ohm
  L=1 mm

- Line2
  Z=50 Ohm
  L=1 mm

- Line3
  d=0.5 mm
  D=0.8 mm
  L=1.5 mm

- Line4
  er=2.29
  L=1500 mm

- Line5
  a=2.95 mm
  b=0.9 mm
  L=1500 mm
  h=1 mm
  t=35 um
  tand=2e-4
  rho=0.022e-6
  D=0.15e-6

- MS1
  W=1 mm
  L=10 mm

- MS2
  W=1 mm
  L=10 mm
  W2=1 mm
  W3=2 mm

- MS3
  Subst=Subst1
  W=1 mm

- MS4
  Subst=Subst1
  W=1 mm

- MS5
  Subst=Subst1
  W=1 mm
  W2=1 mm

- MS6
  W=1 mm
  W2=1 mm
  W3=2 mm

- MS7
  Subst=Subst1
  W1=1 mm
  W2=2 mm
  W3=1 mm
  W4=2 mm

- MS8
  Subst=Subst1
  W=1 mm

- MS9
  Subst=Subst1
  W1=1 mm
  W2=1 mm
  S=1 mm

- CL3
  Subst=Subst1
  W=1 mm
  S=1 mm

- CL4
  Subst=Subst1
  W=1 mm
  S=1 mm
  G=0.5 mm

- CL5
  Subst=Subst1
  W=1 mm
  W2=2 mm
  S=3 mm

- CL6
  Subst=Subst1
  L=50 um
  D=50 mm
  H=2 mm
  L=10 mm

- CL7
  Subst=Subst1
  G=5 mm
Introduction to circuit simulation with Qucs

Qucs built-in analogue components: 4

Simulations

- **dc simulation**
  - DC1

- **transient simulation**
  - TR1
  - Type=lin
  - Start=0
  - Stop=1 ms

- **ac simulation**
  - AC1
  - Type=lin
  - Start=1 GHz
  - Stop=10 GHz
  - Points=19

- **S parameter simulation**
  - SP1
  - Type=lin
  - Start=1 GHz
  - Stop=10 GHz
  - Points=19

- **Harmonic balance simulation**
  - HB1
  - n=4

- **Parameter sweep**
  - SW1
  - Sim=
  - Type=lin
  - Param=R1
  - Start=5 Ohm
  - Stop=50 Ohm
  - Points=20

- **digital simulation**
  - Digi1
  - Type=TruthTable

- **Optimization**
  - Opt1

Diagrams

- Cartesian
- Polar
- Tabular
- Smith Chart
- Admittance
- Smith
- Polar-Smith Combi
- Smith-Polar Combi
- 3D-Cartesian
- Locus curve
- Timing Diagram
- Truth Table

Paintings

- Line
- Arrow
- Text
- Ellipse
- Rectangle
- filled Ellipse
- filled Rectangle
- Elliptic Arc
Introduction to circuit simulation with Qucs

Qucs built-in libraries of components

- Diode Bridges
  - D1
  - D_1N4148_1
  - Is=222p
  - N=1.65

- MOSFETs
  - T_BSS123_1
  - Vt0=1

- Semiconductor Diodes
  - T2D1

- Ideal functional components
  - VADD1

- JFETs
  - J201_1

- LEDs
  - D2

- NMOSFETs
  - M2N3796_1

- PMOSFETs
  - T1

- OPAMPs
  - OP1

- Regulators
  - PVR100AZ_B12V_1

- Substrates
  - RO3003_1
  - er=3.00
  - h=0.8 mm
  - t=35 um
  - tand=0.0013
  - rho=0.022e-6
  - D=1.4e-6

- Transistors
  - T_2DA1774R_1

- Varistors
  - VDR1
  - Tol=0

- Zener Diodes
  - D3
Introduction to circuit simulation with Qucs

Generation of circuits using Qucs design tool suite:
 passive filter synthesis example

Low pass filter synthesis

Other synthesis/design tools:
Text Editor, Line Calculator, Matching Circuits and Attenuator Synthesis
Introduction to circuit simulation with Qucs

Launching a simulation

Simulation control menu

Plotted data

Post-simulation processed and plotted data

Data sets

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S[1,1]</td>
<td>S-parameter value</td>
</tr>
<tr>
<td>nodename.V</td>
<td>DC voltage at node nodename</td>
</tr>
<tr>
<td>name.I</td>
<td>DC current through component name</td>
</tr>
<tr>
<td>nodename.v</td>
<td>AC voltage at node nodename</td>
</tr>
<tr>
<td>name.i</td>
<td>AC current through component name</td>
</tr>
<tr>
<td>nodename.vn</td>
<td>AC noise voltage at node nodename</td>
</tr>
<tr>
<td>name.in</td>
<td>AC noise current through component name</td>
</tr>
<tr>
<td>nodename.Vt</td>
<td>Transient voltage at node nodename</td>
</tr>
<tr>
<td>name.It</td>
<td>Transient current through component name</td>
</tr>
</tbody>
</table>
Introduction to circuit simulation with Qucs
Qucs: Post simulation data processing – MATLAB®*/Octave style**

Equation blocks > data processing > Plots/Tables

Simulation data sets plus any of the following

Constants: i, j, pi, e, kB, q
Number suffixes: E, P, T, G, M, k, m, u, n, p, f, a
Immediate: 2.5, 1.4+j5.1, [1, 3, 4, 5, 7], [11, 12; 21, 22]
Matrices: M, M[2,3], M[:3]
Ranges: Lo:Hi, :Hi, Lo:, :
Arithmetic operators: +x, -x, x+y, x-y, x*y, x/y, x%y, x^y
Logical operators: !x, x&&y, x||y, x^^y, x?y:z, x==y, x!=y, x<y, x<=y,x>y, x>=y

Functions:
abs adjoint angle arccos arccosec arccot arccosech arccosh arccoth arccsc arcsin arctan arg arsech
arsinh artanh avg besseli0 besselj bessely ceil conj cos cosec cosech cosh cot coth cumavg cumprod
cumsum dB dbm dbm2w deg2rad det dft diff erf erfc erfcinv erfinv exp eye fft fix floor Freq2Time GaCircle
GpCircle hypot idft ifft imag integrate interpolate inverse kbd limexp linspace ln log10 log2 logspace mag
max min Mu Mu2 NoiseCircle norm phase PlotVs polar prod rad2deg random real rms Rollet round rtoswr
rtoz runavg sec sech sign sin sinc sinh sqr sqrt srandom StabCircleL StabCircleS StabFactor
StabMeasure stddev step stos stoy stoz sum tan tanh Time2Freq transpose twoport unwrap variance vt
w2dbm xvalue ytor ytos ytoz yvalue ztor ztos ztoy

* MATLAB, Mathworks, http://www.mathworks.com/
Building a compact semiconductor model of an npn bipolar phototransistor

The phototransistor consists of an Ebers-Moll bipolar junction transistor model which has been extended to include depletion and diffusion capacitance, forward and reverse Early effects, high current forward and reverse beta degradation, thermal and shot noise, plus a light bus which connects external light signals to the phototransistor.
Building a compact semiconductor model of an npn bipolar phototransistor

**Phototransistor model parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Symbol</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>BF</td>
<td>$\beta_f$</td>
<td>Forward beta</td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>BR</td>
<td>$\beta_r$</td>
<td>Reverse beta</td>
<td></td>
<td>0.1</td>
</tr>
<tr>
<td>Is</td>
<td>Is</td>
<td>Saturation current</td>
<td>A</td>
<td>$1e^{-10}$</td>
</tr>
<tr>
<td>Nf</td>
<td>$N_f$</td>
<td>Forward emission coefficient</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Nr</td>
<td>$N_r$</td>
<td>Reverse emission coefficient</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Var</td>
<td>$Var$</td>
<td>Reverse Early voltage</td>
<td>V</td>
<td>100</td>
</tr>
<tr>
<td>Vaf</td>
<td>$Vaf$</td>
<td>Forward Early voltage</td>
<td>V</td>
<td>100</td>
</tr>
<tr>
<td>Mje</td>
<td>$M_{je}$</td>
<td>Base-emitter exponential factor</td>
<td></td>
<td>0.33</td>
</tr>
<tr>
<td>Vje</td>
<td>$V_{je}$</td>
<td>Base-emitter built-in potential</td>
<td>V</td>
<td>0.75</td>
</tr>
<tr>
<td>Cje</td>
<td>$C_{je}$</td>
<td>Base-emitter zero-bias depletion capacitance</td>
<td>F</td>
<td>1p</td>
</tr>
<tr>
<td>Mjc</td>
<td>$M_{jc}$</td>
<td>Base-collector exponential factor</td>
<td></td>
<td>0.33</td>
</tr>
<tr>
<td>Vjc</td>
<td>$V_{jc}$</td>
<td>Base-collector built-in potential</td>
<td>V</td>
<td>0.75</td>
</tr>
<tr>
<td>Cjc</td>
<td>$C_{jc}$</td>
<td>Base-collector zero-bias depletion capacitance</td>
<td>F</td>
<td>1p</td>
</tr>
<tr>
<td>Tr</td>
<td>$Tr$</td>
<td>Ideal reverse transit time</td>
<td>s</td>
<td>100n</td>
</tr>
<tr>
<td>Tf</td>
<td>$Tf$</td>
<td>Ideal forward transit time</td>
<td>s</td>
<td>0.1n</td>
</tr>
<tr>
<td>Ikf</td>
<td>$I_{kf}$</td>
<td>High current corner for forward beta</td>
<td>A</td>
<td>0.5</td>
</tr>
<tr>
<td>Ikfr</td>
<td>$I_{kr}$</td>
<td>High current corner for reverse beta</td>
<td>A</td>
<td>0.5</td>
</tr>
<tr>
<td>Rc</td>
<td>$R_c$</td>
<td>Collector series resistance</td>
<td>$\Omega$</td>
<td>2</td>
</tr>
<tr>
<td>Re</td>
<td>$R_e$</td>
<td>Emitter series resistance</td>
<td>$\Omega$</td>
<td>1</td>
</tr>
<tr>
<td>Rb</td>
<td>$R_b$</td>
<td>Base series resistance</td>
<td>$\Omega$</td>
<td>100</td>
</tr>
<tr>
<td>Kf</td>
<td>$K_f$</td>
<td>Flicker noise coefficient</td>
<td></td>
<td>$1e^{-12}$</td>
</tr>
<tr>
<td>Ffe</td>
<td>$F_{fe}$</td>
<td>Flicker noise frequency exponent</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Af</td>
<td>$A_f$</td>
<td>Flicker noise exponent</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Responsivity</td>
<td>Responsivity</td>
<td>Responsivity at peak wavelength</td>
<td>A/W</td>
<td>1.5</td>
</tr>
<tr>
<td>P0</td>
<td>$P_0$</td>
<td>Relative selectivity polynomial coefficient</td>
<td>%</td>
<td>$2.6122x10^3$</td>
</tr>
<tr>
<td>P1</td>
<td>$P_1$</td>
<td>Relative selectivity polynomial coefficient</td>
<td>%/nm</td>
<td>$-1.4893x10^1$</td>
</tr>
<tr>
<td>P2</td>
<td>$P_2$</td>
<td>Relative selectivity polynomial coefficient</td>
<td>%/nm²</td>
<td>$3.0332x10^{-2}$</td>
</tr>
<tr>
<td>P3</td>
<td>$P_3$</td>
<td>Relative selectivity polynomial coefficient</td>
<td>%/nm³</td>
<td>$-2.5708x10^{-5}$</td>
</tr>
<tr>
<td>P4</td>
<td>$P_4$</td>
<td>Relative selectivity polynomial coefficient</td>
<td>%/nm⁴</td>
<td>$7.6923x10^{-9}$</td>
</tr>
</tbody>
</table>
Building a compact semiconductor model of an npn bipolar phototransistor

Bipolar phototransistor model equations

\[
\begin{align*}
IEC &= Is \cdot \exp \left( \frac{V(BI, CI)}{N_r \cdot vt(300)} - 1 \right) \\
ICC &= Is \cdot \exp \left( \frac{V(BI, EI)}{N_f \cdot vt(300)} - 1 \right)
\end{align*}
\]

\[
\begin{align*}
IDC &= \frac{IEC}{\beta r} + GMIN \cdot V(BI, CI) \\
IDE &= \frac{ICC}{\beta f} + GMIN \cdot V(BI, EI)
\end{align*}
\]

\[
\begin{align*}
q1 &= 1 + \frac{V(BI, CI)}{Vaf} + \frac{V(BI, EI)}{Var} \\
q2 &=IBC + IEC \cdot V(T) = \frac{K \cdot T}{q}
\end{align*}
\]

\[
\begin{align*}
CBE &= \frac{dQ|BI, EI|}{dV|BI, EI|} = Cje \left[ 2 \cdot Mje \cdot V|BI, EI| \right] + Tr \cdot \frac{dIEC}{dV|BI, CI|} \\
ICT &= \frac{ICC - IEC}{q1} \cdot \frac{1}{1 + \sqrt{1 + 4 \cdot q2}}
\end{align*}
\]

\[
\begin{align*}
CBC &= \frac{dQ|BI, CI|}{dV|BI, CI|} = Cjc \left[ 1 - \frac{V|BI, CI|}{Vjc} \right]^{Mjc} + Tr \cdot \frac{dIEC}{dV|BI, CI|} \\
\forall \ V|BI, CI| < \frac{Vjc}{2}
\end{align*}
\]

\[
\begin{align*}
CBE &= \frac{dQ|BI, EI|}{dV|BI, EI|} = Cje \left[ 1 - \frac{V|BI, EI|}{Vje} \right]^{Mje} + Tr \cdot \frac{dICC}{dV|BI, EI|} \\
\forall \ V|BI, EI| < \frac{Vje}{2}
\end{align*}
\]

\[
\begin{align*}
I_{opt} &= Gpbc \cdot P_{opt} \\
Gpbc &= \frac{RelSensitivity \cdot Responsivity}{\beta f \cdot 100}
\end{align*}
\]

\[
\begin{align*}
RelSensitivity &= P0 + P1 \cdot \lambda + P2 \cdot \lambda^2 + P3 \cdot \lambda^3 + P4 \cdot \lambda^4
\end{align*}
\]

\[
\begin{align*}
iRcn^2 &= \frac{4 \cdot K \cdot T}{Rc} \cdot \Delta f \\
iRbn^2 &= \frac{8 \cdot K \cdot T}{Rb} \cdot \Delta f \\
iICTsn^2 &= 2 \cdot q \cdot IC \cdot \Delta f
\end{align*}
\]

Where K is the Boltzmann constant, T is the temperature in Kelvin, q is the electron charge, GMIN is a small admittance in parallel with the device junctions, \( \Delta f \) is the noise frequency bandwidth in Hz and \( \lambda \) is the light wavelength in nm. Other symbols and node names are defined in the previous slides.
Building a compact semiconductor model of an npn bipolar phototransistor

Construction: stage 1 – the basic npn transistor model

Equation defined device

Large signal DC model

Generate symbol

Click right-hand mouse button on drawing pad

Equation defined device

Large signal DC model

Generate symbol

Click right-hand mouse button on drawing pad

$$I = I(V), \ g = \frac{dI}{dV}$$

$$Q = Q(I, V)$$

$$C = \frac{dQ}{dV} = \frac{dQ(V)}{dV} + \frac{dQ(I)}{dI} \cdot g$$
Building a compact semiconductor model of an npn bipolar phototransistor

Construction: stage 2 – DC simulation tests

Parameter sweep
SW1
Sim=SW2
Type=log
Param=Ib
Start=1e-6
Stop=1m
Points=16

Parameter sweep
SW2
Sim=DC1
Type=lin
Param=Vce
Start=0
Stop=5
Points=51

Equation
Eqn1
Beta=Ic/Ib
PLBeta=PlotVs(Beta, Ic)
Building a compact semiconductor model of an npn bipolar phototransistor

**Construction: stage 3 – Adding capacitance to the phototransistor model**

**Charge equations**

\[
Q_{BI,CI} = \int_0^{V_{BI,CI}} CBC \cdot dV = Tr \cdot IEC + 2^{Mjc} \cdot Cjc \cdot \left[ Mjc \cdot \frac{V_{BI,CI}^2}{Vjc} + (1 - Mjc) \cdot V_{BI,CI} \right] \quad \forall V_{BI,CI} > \frac{Vjc}{2}
\]

\[
= Tr \cdot IEC + Cjc \cdot Vjc \cdot \left[ 1 - \left( 1 - \frac{V_{BI,CI}}{Vjc} \right)^{1-Mjc} \right] \quad \forall V_{BI,CI} \leq \frac{Vjc}{2}
\]

\[
Q_{BI,IE} = \int_0^{V_{BI,IE}} CBE \cdot dV = Tf \cdot ICC + 2^{Mje} \cdot Cje \cdot \left[ Mje \cdot \frac{V_{BI,IE}^2}{Vje} + (1 - Mje) \cdot V_{BI,IE} \right] \quad \forall V_{BI,IE} > \frac{Vje}{2}
\]

\[
= Tf \cdot ICC + Cje \cdot Vje \cdot \left[ 1 - \left( 1 - \frac{V_{BI,IE}}{Vje} \right)^{1-Mje} \right] \quad \forall V_{BI,IE} \leq \frac{Vje}{2}
\]

**EDD blocks**

**Equation**

Eqn2
con3=1-Mje
con4=1-Mjc
con5=2^Mje
con6=2^Mjc
Building a compact semiconductor model of an npn bipolar phototransistor

Construction: stage 4 – simulating capacitive effects

AC gain

Cbe and Rbe extraction
Building a compact semiconductor model of an npn bipolar phototransistor

Construction: stage 5 – Adding photoelectric effects

Si phototransistor relative sensitivity data

Measured data

<table>
<thead>
<tr>
<th>Wavelength (nm)</th>
<th>Relative responsivity (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>15</td>
</tr>
<tr>
<td>550</td>
<td>25</td>
</tr>
<tr>
<td>600</td>
<td>40</td>
</tr>
<tr>
<td>650</td>
<td>58</td>
</tr>
<tr>
<td>700</td>
<td>77</td>
</tr>
<tr>
<td>750</td>
<td>95</td>
</tr>
<tr>
<td>800</td>
<td>100</td>
</tr>
<tr>
<td>850</td>
<td>95</td>
</tr>
<tr>
<td>900</td>
<td>80</td>
</tr>
<tr>
<td>950</td>
<td>65</td>
</tr>
<tr>
<td>1000</td>
<td>35</td>
</tr>
</tbody>
</table>

Curve fitting program

```
## Simple phototransistor responsivity fitting
## program - works with Octave or MATLAB
data = load('data.dat');
Ldata = data(:,1);
Sdata = data(:,2);
## Fit polynomial
Order = 4;
p = polyfit(Ldata, Sdata, Order);
## Evaluate the fitted polynomial
x = linspace(min(Ldata), max(Ldata), 101);
y = polyval(p, x);
## Plot
plot(x, y, '.', Ldata, Sdata,'x');
legend('Fitted polynomial', 'Original data');
```

**RelSensitivity** = \( P_0 + P_1 \cdot \lambda + P_2 \cdot \lambda^2 + P_3 \cdot \lambda^3 + P_4 \cdot \lambda^4 \)

Where \( P_0 = 2.6122 \times 10^3 \), \( P_1 = -1.4893 \times 10^1 \), \( P_2 = 3.0332 \times 10^{-2} \), \( P_3 = -2.5708 \times 10^{-5} \), \( P_4 = 7.69 \times 10^{-9} \)

EDD light bus model

**Iopt** = \( \frac{\text{RelSensitivity} \cdot \text{Responsivity} \cdot \text{power}}{\beta_f \cdot 100} \)
Building a compact semiconductor model of an npn bipolar phototransistor

Construction: stage 6 – simulating photoelectric effects

Phototransistor output characteristics

Light_source1
Power=P_sweep
Wavelength=800

Parameter sweep
DC1
SW1
Sim=SW2
Type=lin
Param=P_sweep
Start=0
Stop=10m
Points=7

P
W
I1
I=0
P
W
I1
I=0

V1
U=Vce

Ptran1
Bf=100
Br=0.1
Is=1e-10
Ni=1
Nr=1
Var=100
Vaf=100
Mje=0.33
Vje=0.75
Cje=1p
Mjc=0.33
Vjc=0.75
Cjc=2p
Tr=100n
Tj=0.1n
Ikf=0.5
Ikr=0.5
Rc=2
Re=1
Rb=100
Kf=1e-12
Fle=1.0
Af=1.0
Responsivity=1.5
P0=2.6122e3
P1=1.4893e1
P2=3.0332e-2
P3=2.5708e-5
P4=7.6923e-9

Power = 10 mW

Phototransistor responsivity characteristics

Light_source1
Power=10m
Wavelength=W_sweep

Parameter sweep
DC1
SW1
Sim=DC1
Type=lin
Param=W_sweep
Start=500
Stop=1000
Points=51

Ptran1
Bf=100
Br=0.1
Is=1e-10
Ni=1
Nr=1
Var=100
Vaf=100
Mje=0.33
Vje=0.75
Cje=1p
Mjc=0.33
Vjc=0.75
Cjc=2p
Tr=100n
Tj=0.1n
Ikf=0.5
Ikr=0.5
Rc=2
Re=1
Rb=100
Kf=1e-12
Fle=1.0
Af=1.0
Responsivity=1.5
P0=2.6122e3
P1=1.4893e1
P2=3.0332e-2
P3=2.5708e-5
P4=7.6923e-9
Building a compact semiconductor model of an npn bipolar phototransistor

Construction: stage 7 – adding noise to the phototransistor model

Qucs noise voltage source

VPSD\( (f) = \frac{U}{a + c \cdot f^e} \)

\( i_{bsn} = \sqrt{2 \cdot q \cdot IB} \ \frac{A}{\sqrt{Hz}} \)

\( i_{bfn} = \sqrt{\frac{Kf}{f \cdot Ffe}} \ \frac{A}{\sqrt{Hz}} \)

Where VPSD\( (f) \) is the voltage spectral density at frequency \( f \) in \( V^2 / \text{Hz} \), \( U \) is the voltage spectral density as \( f \) goes to zero, \( a, c \) and \( e \) are coefficients that determine the type of noise generated; white and shot noise with \( U=1, e=0, c=1 \) and \( a=0 \), and flicker noise with \( U=Kf, e=Ffe, c=1 \) and \( a=0 \).

\( i_{ICTsn} = \sqrt{2 \cdot q \cdot IC} \ \frac{A}{\sqrt{Hz}} \)

Note: Resistor thermal noise is automatically generated by Qucs.
Building a compact semiconductor model of an npn bipolar phototransistor

Construction: stage 8 – simulating phototransistor noise

Variable light power

Parameter sweep

SW1
Sim=AC1
Type=lin
Param=sw_power
Start=0
Stop=20m
Points=11

AC1
Type=log
Start=1 Hz
Stop=100MHz
Points=51
Noise=yes

Variable wavelength

Parameter sweep

SW1
Sim=AC1
Type=lin
Param=sw_wavelength
Start=500
Stop=800
Points=51
Noise=yes

Ptran1
Bf=100
Br=0.1
Is=1e-10
Nf=1
Nr=1
Var=100
Va=100
Mje=0.33
Vje=0.75
Cje=1p
Mjc=0.33
Vjc=0.75
Cjc=2p
Tr=100n
 Tf=0.1n
Ikf=0.5
Ikr=0.5
Rc=2
Re=1
Rb=100
Kf=1e-12
Fte=1.0
A1=1.0
Responsivity=1.5
P0=2.6122e3
P1=1.4893e1
P2=3.0332e2
P3=2.5708e-5
P4=7.6923e-9

DC simulation

Light_source1
Power=sw_power
Wavelength=800

P
W
\( \text{IC} \)
V1
U=3 V

Ptran1
Bf=100
Br=0.1
Is=1e-10
Nf=1
Nr=1
Var=100
Va=100
Mje=0.33
Vje=0.75
Cje=1p
Mjc=0.33
Vjc=0.75
Cjc=2p
Tr=100n
 Tf=0.1n
Ikf=0.5
Ikr=0.5
Rc=2
Re=1
Rb=100
Kf=1e-12
Fte=1.0
A1=1.0
Responsivity=1.5
P0=2.6122e3
P1=1.4893e1
P2=3.0332e2
P3=2.5708e-5
P4=7.6923e-9

AC simulation

Collector noise current (\( \text{A}/\sqrt{\text{Hz}} \))

Frequency (Hz)

Collector noise current (\( \text{A}/\sqrt{\text{Hz}} \))

Frequency (Hz)
Building a compact semiconductor model of an npn bipolar phototransistor

Construction: stage 9 – producing a distribution standard model

Model

Generate Verilog-A code

Store in Qucs library

Use “Create Library” from Project menu

Phototransistor.va file

```
`include "disciplines.vams"
`include "constants.vams"
module phototransistor (Collector, Base, Emitter, Power, Wavelength);
inout Collector, Base, Emitter, Power, Wavelength;
electrical Collector, Base, Emitter, Power, Wavelength;
// Definition of internal local nodes
electrical CI, BI, B12, EI;
// Parameter values and description text
define attr (txt) (*txt*)
parameter real Bf=100 from [1:inf] `attr(info="forward beta");
p Parameter real Br=0.1 from [1e-6:inf] `attr(info="reverse beta");

parameter real P3=-2.5708e-5 from[-inf:inf] `attr(info="relative selectivity polynomial coefficient");
parameter real P4=7.6923e-9 from[-inf:inf] `attr(info="relative selectivity polynomial coefficient");
// Definition of internal variables and quantities
real VT, con1, con2, con3, con4, con5, con6, con7, con8, con9, con10, TwoQ, FourKT, GMIN;
real ICC, IEC, q1, q2, IB, IC, IE, Q1, RelSensitivity; // Quantities
analog begin
// Module initialisation code
@ (initial_model)
begin
VT = `P_K*300/`P_Q; con1=1/(Nf*VT); con2=1/(Nr*VT); con3=1-Mje; // VT = vt(300)
con4=1-Mjc, con5=pow(2, Mje); con6=pow(2, Mjc); con7=Rb/2; con8=2/Rb;
con9=1/RC; con10=1/Re; TwoQ=2*`P_Q; GMIN=1e-12; // TwoQ = 2^q
FourKT=4*`P_K*$temperature; // FourKT = 4*K*T
end;
// Model quantity equations and current contributions
ICC=Is*(limexp(V(BI,EI)*con1)-1); IEC=Is*(limexp(V(BI,CI)*con2)-1);
q1=1+V(BI,CI)/Vaf + V(BI,EI)/Var; q2=(ICC/Ifk) + (IEC/IKr); IB=V(B2,BI)*con8;
IC=V(Collector,CI)*con9; IE=V(EI,Emitter)*con10; l (Collector,CI) <+ IC;
I(Base,B12) <+ V(Base, B12)*con8; I(B12, BI) <+ IB; I(EI, Emitter) <+ IE;
I(B1, CI) <+ (IEC/Be) + GMIN*V(B1,CI); I(B1, EI) <+ (ICC/BE) + GMIN*V(B1, EI);
Q1=(V(B1,CI) >Vjc/2) ? Tr*IEC+Cjc*con6*(Mjc*V(B1,CI)*V(B1,CI)/Vjc+con4*V(B1,CI))
: Tr*IEC+Cjc*V(Vj/con4)*((1-pow(1-(Vj/Vje),con4)));
Q1=(V(B1, EI) >Vje/2) ? Tr*ICC+Cje*con5*(Mje*V(B1,EL)*V(B1,EL)/Vje+con3*V(B1,EL))
: Tr*ICC+Cje*V(Vje/con3)*((1-pow(1-(Vj/Vje),con3)));
I(B1, EI) <+ ddt(Q1);
RelSensitivity = P0+P1*V(Wavelength)+P2*V(Wavelength)^2;
I(Cl, B12) <+ (Responsivity*RelSensitivity)/(Bf*100) *V(Power);
// Noise contributions
l (Collector,CI) <+ white_noise(FourKT*con9, "thermal");
l (Base,B12) <+ white_noise(FourKT*con8, "thermal");
l (B12, BI) <+ white_noise(FourKT*con8, "thermal");
l (EI, Emitter) <+ white_noise(FourKT*con10, "thermal");
l (Cl, EI) <+ white_noise(TwoQ^4IC, "shot");
l (B1, EI) <+ white_noise(TwoQ^4IB, "shot");
l (B1, EI) <+ flicker_noise(Kf*pow(IB, A), Ffe, "flicker");
end
endmodule
```
Building a compact semiconductor model of an npn bipolar phototransistor

Construction: stage 10 – Compiling Verilog-A code to C++

Compile phototransistor.va file

with ADMS* using

1. admsXml phototransistor.va -e qucsVersion.xml -e qucsMODULEcore.xml
   : generates files phototransistor.core.cpp and phototransistor.core.h

2. admsXml phototransistor.va -e qucsVersion.xml -e qucsMODULEdefs.xml
   : generates file phototransistor.defs.h

3. admsXml phototransistor.va -e qucsVersion.xml -e qucsMODULEgui.xml
   : generates files phototransistor.gui.cpp phototransistor.gui.h

4. admsXml phototransistor.va -e analogfunction.xml
   : generates files phototransistor.analogfunction.cpp and
     phototransistor.analogfunction.h

**File phototransistor.va and the generated compiled files must be in Qucs directory**
qucs-core/src/components/verilog


Building a compact semiconductor model of an npn bipolar phototransistor

Construction: stage 11 – Merging phototransistor C++ code with the Qucsator analogue circuit simulator code

11.1 Modify Makefile.am in directory qucs-core/src/components/Verilog
(a) Add `phototransistor.analogfunction.cpp` and `phototransistor.core.cpp` to entry `libverilog_a_SOURCES`
(b) Add `phototransistor.analogfunction.h`, `phototransistors.defs.h` and `phototransistor.core.h` to entry `noinst_HEADERS`
(c) Add `phototransistor.va` to entry `VERILOG_FILES`

11.2 Add `#include “verilog/phototransistor.core.h”` to file `qucs-core/src/components/components.h`

11.3 Add under heading // circuit components
   `REGISTER_CIRCUIT(phototransistor)` to file `qucs-core/src/module.cpp`

11.4 Add under heading // verilog devices (this step is optional but recommended)
   `CIR_phototransistor` to file `qucs-core/src/components/component_id.h`
Building a compact semiconductor model of an npn bipolar phototransistor

Construction: stage 12 – Adding the phototransistor to the Qucs GUI code: part 1

12.1 Add to array pInfoFunc VerilogAComps[ ] in file qucs/qucs/qucs.cpp entry &phototransistor::info before 0 at and of list

12.2 Rename phototransistor.gui.cpp to phototransistor.cpp and phototransistor.gui.h to phototransistor.h, then copy renamed files to directory qucs/qucs/components

12.3 Edit phototransistor.cpp to include GUI drawing information

```c++
#include "phototransistor.h"
phototransistor::phototransistor()
{
    Description = QObject::tr("phototransistor verilog device");
    Props.append (new Property ("Bf", "100", false, QObject::tr ("forward beta")));
    Props.append (new Property ("Br", "0.1", false, QObject::tr ("reverse beta")));
    Props.append (new Property ("Is", "1e-10", false, QObject::tr ("dark current") + " (" + QObject::tr ("A") + ")");
    ...
    Props.append (new Property ("Temp", "26.85", false, QObject::tr ("simulation temperature")));
    createSymbol ();
    tx = x2 + 4;
    ty = y1 + 4;
    Model = "phototransistor";
    Name = "PT";
}
Component * phototransistor::newOne() {
    phototransistor * p = new phototransistor();
    p->Props.getFirst()->Value = Props.getFirst()->Value;
    p->recreate(0);
    return p;
}
Element * phototransistor::info(QString& Name, char * &BitmapFile, bool getNewOne) {
    Name = QObject::tr("photo transistor");
    BitmapFile = (char *) "phototransistor";
    if(getNewOne) return new phototransistor();
    return 0;
}
```
Building a compact semiconductor model of an npn bipolar phototransistor

Construction: stage 12 – Adding the phototransistor to the Qucs GUI code: part 2

Editing `phototransistor.cpp` continued

```cpp
void phototransistor::createSymbol()
{
    Arcs.append(new Arc(-25,-20, 40, 0,20*360,QPen(QPen::red,2)));
    Lines.append(new Line(-10,-15, 10,15,QPen(QPen::darkBlue,3)));
    Lines.append(new Line(-30, 0,10, 0,QPen(QPen::darkBlue,2)));
    Lines.append(new Line(-10,-5, 0,15,QPen(QPen::darkBlue,2)));
    Lines.append(new Line( 0,-15, 10, 0,QPen(QPen::darkBlue,2)));
    Lines.append(new Line(-10, 5, 10,15,QPen(QPen::darkBlue,2)));
    Lines.append(new Line( 0, 15, 10,15,QPen(QPen::darkBlue,2)));
    Lines.append(new Line(-6, 15, 0,15,QPen(QPen::darkBlue,2)));
    Lines.append(new Line( 0, 5, 0,15,QPen(QPen::darkBlue,2)));
    Lines.append(new Line(-15, 9, 0,15,QPen(QPen::darkBlue,2)));
    Lines.append(new Line(-55,-55, -40,-55,QPen(QPen::green,2)));
    Lines.append(new Line(-40,-55, -40,-25,QPen(QPen::green,2)));
    Lines.append(new Line(-40,-25, -55,-25,QPen(QPen::green,2)));
    Lines.append(new Line(-40,-40, -15,-15,QPen(QPen::green,2)));
    Lines.append(new Line(-15,-15, -15,-20,QPen(QPen::green,2)));
    Lines.append(new Line(-15,-15, -20,-15,QPen(QPen::green,2)));
    Lines.append(new Line(-60,-60, -60,-70,QPen(QPen::black,2)));
    Lines.append(new Line(-60,-70, -55,-70,QPen(QPen::black,2)));
    Lines.append(new Line(-55,-70, -55,-65,QPen(QPen::black,2)));
    Lines.append(new Line(-60,-65, -55,-65,QPen(QPen::black,2)));
    Lines.append(new Line(-63,-35, -60,-30,QPen(QPen::black,2)));
    Lines.append(new Line(-60,-30, -57,-35,QPen(QPen::black,2)));
    Lines.append(new Line(-57,-35, -54,-30,QPen(QPen::black,2)));
    Lines.append(new Line(-54,-30, -51,-35,QPen(QPen::black,2)));
    Ports.append(new Port(0, -30)); // Collector
    Ports.append(new Port(-30, 0)); // Base
    Ports.append(new Port( 0, 30)); // Emitter
    Ports.append(new Port(-55,-55)); // Power
    Ports.append(new Port(-55,-25)); // Wavelength
    x1 = -35; y1 = -55;
    x2 = 35; y2 = 50;
}
```
Building a compact semiconductor model of an npn bipolar phototransistor

Construction: stage 12 – Adding the phototransistor to the Qucs graphical user interface code: part 3

12.3 (a) Draw a 32 bit x 32 bit icon picture for the phototransistor and add it to directory qucs/qucs/bitmaps
(b) Add phototransistor.png to XPMS entry in file qucs/qucs/bitmaps/Makefile.am

12.4 Add to file qucs/qucs/components/component.cpp
(in getComponentFromName, in switch (first) statement)
as part of case 'p'
else if (cstr == "phototransistor") c = new phototransistor();

12.5 Modify Makefile.am in directory qucs/qucs/components
(a) Add phototransistor.cpp to entry libverilog_a_SOURCES
(b) Add phototransistor.h to entry noinst_HEADERS

12.6 Add #include “phototransistor.h”
to file qucs/qucs/components/components.h

FINAL STEP
Compile qucs and qucs-core
Run qucs and test new phototransistor model
Summary

1. Qucs is a freely available circuit simulator distributed as open source software under the GNU/Linux General Public Licence (GPL).

2. This demonstration has attempted to outline the fundamental features of the package, its available components, libraries, built-in design aids, and analysis types.

3. The demonstration also introduces a number of basic approaches to circuit simulation using Qucs.

4. The presentation also showed how the compact semiconductor modeling and circuit macromodeling features implemented in the current Qucs release can be used to develop equation-defined component models of established and emerging technology devices.

5. The latter sections of the demonstration concentrated on using ADMS to compile Verilog-A compact device models and the steps needed to link such models with the main body of the Qucs C++ code.