

Model implementation into ADS

Rüdiger Follmann



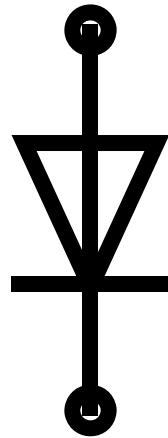
Contents

- PN-diode, equivalent circuit and equations
- Linear part
- Non-linear part
- Linearized part
- Noise
- Interface
- Consistent implementation



PN diode

Implementation of a pn-diode model
into Agilent's ADS



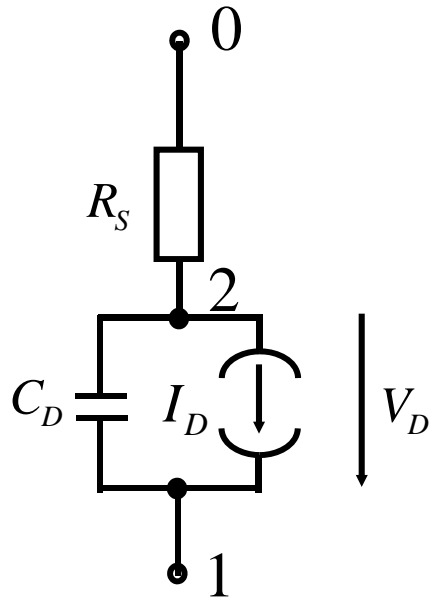


How to start?

A non-linear model consists of the parts

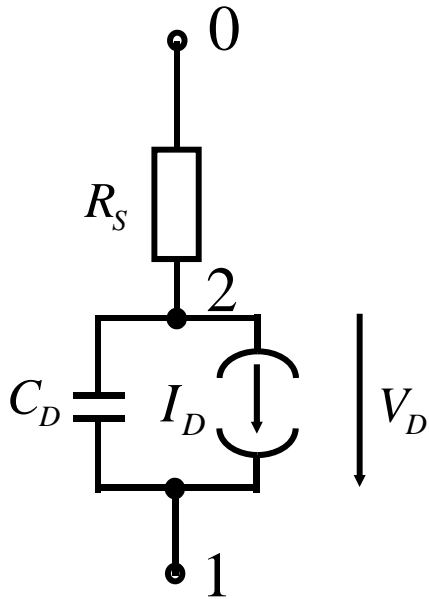
- Equivalent circuit and associated equations
- Linear part containing linear elements
- Same with non-linear part
- Bias point dependent linearized part (Jacobian matrix)
- Bias dependent noise part

ADS specific things



- All nodes are numbered
- Outer nodes get smallest numbers

Equivalent circuit and equations



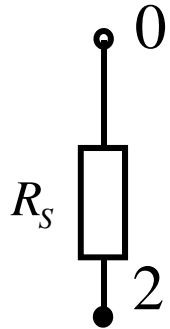
$$I_D(V_D) = I_S \left(e^{\frac{V_D}{V_{te}}} - 1 \right) + 10^{-12} \frac{\text{A}}{\text{V}} V_D$$

$$\frac{dI_D}{dV_D} = \frac{I_S e^{\frac{V_D}{V_{te}}}}{V_{te}} + 10^{-12} \frac{\text{A}}{\text{V}}$$

$$C_D(V_D) = TT \frac{dI_D}{dV_D} + CJ0 \left(1 - \frac{V_D}{VJ} \right)^{-M}$$

$$Q_D(V_D) = \int_0^{V_D} C_D(\tilde{V}_D) d\tilde{V}_D = TT \cdot I_D + \frac{VJ \cdot CJ0}{1-M} \left(1 - \left(1 - \frac{V_D}{VJ} \right)^{1-M} \right)$$

Linear part



R_s is the only linear element between nodes 0-2

COMPLEX y ;

y .real = $1/R_s$; y .imag = 0.0;

status = add_y_branch(userInst, 0, 2,
 y);

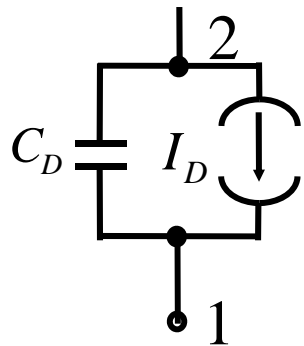
Admittances here

nodes

Y-matrix

Non-linear part

- Number all nodes in order to get the voltages
- Start with outer nodes
- Use code calculates charges at each node and non-linear currents out of each node



I_D and C_D are non-linear elements between nodes 1-2



Linearized part

- Calculation of partial derivatives for non-linear elements with respect to the nodes to get capacitances and conductances for Jacobian matrix
- The linearized part does not influence the final harmonic balance solution
- The linearized part influences the convergence speed



Noise

→ Noise part contains the bias dependent correlation parameters - normalized to $4kT_0$

→ This part is only needed for noise analysis

Possible noise sources:

Thermal noise $\langle i_{th}^2 \rangle = \frac{4kT_0}{R} \Delta f$

Shot noise $\langle i_{shot}^2 \rangle = 2qI_D \Delta f$

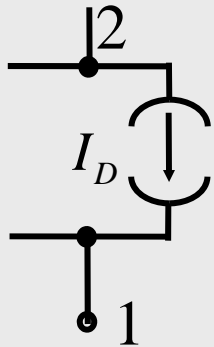
1/f noise $\langle i_{1/f}^2 \rangle = KF \frac{I_D^{AF}}{f^{FFE}} \Delta f$

Burst or popcorn noise

$$\langle i_{burst}^2 \rangle = KB \frac{I_D^{CF}}{1 + \left(\frac{f}{f_{CF}} \right)^2} \Delta f$$

Non-linear part

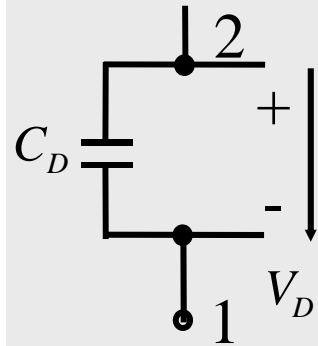
Calculation of node currents and charges



Node 1: $i_1 = -I_D$

Node 2: $i_2 = +I_D$

Current from node:+, to node:-



Node 1: $q_1 = -Q_D$

Node 2: $q_2 = +Q_D$

Node i q



```
status = add_nl_iq(userInst, 1, -id, -qd) &&  
         add_nl_iq(userInst, 2, +id, +qd);
```

Jacobian matrix

$$\frac{\partial I_1}{\partial V_1} = -1 \cdot \left(-\frac{dI_D}{dV_D} \right) = \frac{dI_D}{dV_D}$$

$$\frac{\partial I_1}{\partial V_2} = -\frac{dI_D}{dV_D}$$

Real part

$$\frac{\partial I_2}{\partial V_1} = -\frac{dI_D}{dV_D}$$

$$\frac{\partial I_2}{\partial V_2} = \frac{dI_D}{dV_D}$$

Imaginary part

$$\frac{\partial Q_1}{\partial V_1} = C_D$$

$$\frac{\partial Q_1}{\partial V_2} = -C_D$$

$$\frac{\partial Q_2}{\partial V_1} = -C_D$$

$$\frac{\partial Q_2}{\partial V_2} = C_D$$

partial
derivatives

g

c

+
-
↓



$$V_D = V_2 - V_1$$

```
status = add_nl_gc(userInst, 1, 1, +gd, +cd) &&
add_nl_gc(userInst, 1, 2, -gd, -cd) &&
add_nl_gc(userInst, 2, 1, -gd, -cd) &&
add_nl_gc(userInst, 2, 2, +gd, +cd);
```

Noise

```
COMPLEX thermal, dNoise;
```

```
thermal.imag = dNoise.imag = 0.0;
```

```
thermal.real = 1.0/Rs*DEV_TEMP/NOISE_REF_TEMP;
```

Thermal noise

```
dNoise.real = 2 * CHARGE * id;
```

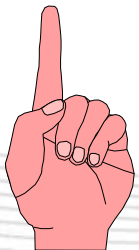
Shot noise

```
dNoise.real = kf * pow(id, AF)*pow(omega/TWOPI, -FFE);
```

1/f noise

```
dNoise.real /= FOUR_K_T0;
```

```
status =      add_n_branch(userInst, 0, 2, thermal) &&  
              add_n_branch(userInst, 1, 2, dNoise);
```



Attention: All noise currents are normalized to $4kT_0$

Interface

`nonlin_ele.h`

```
// name      numPars   params   compute_y   post_analysis   senior Info
//   numExtNodes  pre_analysis  compute_n   devDef      tranDef
```

```
{"PNDIODE", 0, size(PNDIODE), PNDIODE, NULL, NULL, NULL, NULL,
NULL, NULL, NULL}
```

`nonlin_def.h`

```
static UserNonLinDef PNDIODE =
{
    0,          /* numIntNodes */
    NULL,      /* analyze_lin() */
    sdiode_nl, /* analyze_nl() */
    sdiode_ac, /* analyze_ac() */
    NULL,     /* modelDef */
    NULL     /* analyze_ac_n */
};
```

Interface

nonlin_typ.h

```
PNDIODE[] =  
{ /* P-N junction diode */  
  {"AREA",  REAL_data},  
  {"IS",    REAL_data},  
  {"RS",    REAL_data},  
  {"N",     REAL_data},  
  {"TT",    REAL_data},  
  ...  
}
```

nonlin_fun.h

```
* NON-LINEAR PN-DIODE MODEL */  
EXTERN_FUNCTION( extern boolean  pndiode_lin, (UserInstDef *userInst, double omega));  
EXTERN_FUNCTION( extern boolean  pndiode_nl,  (UserInstDef *userInst, double *vPin));  
EXTERN_FUNCTION( extern boolean  pndiode_ac,  (UserInstDef *userInst, double *vPin, double omega));  
EXTERN_FUNCTION( extern boolean  pndiode_ac_n,(UserInstDef *userInst, double *vPin, double omega));
```

Interface

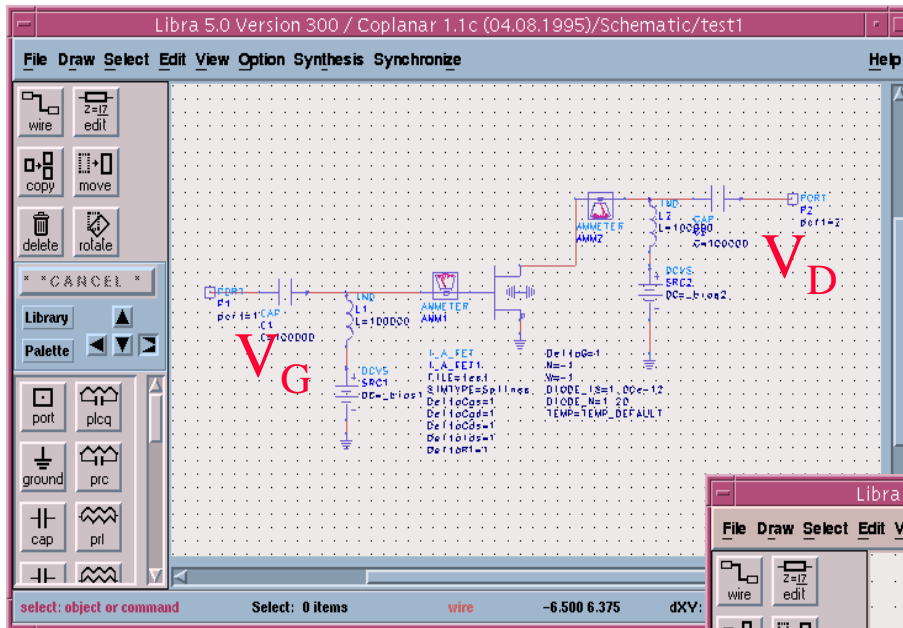
→ AEL programming language is the GUI for the simulator



Summary

- Different parts of a non-linear model
- Nonlinear equations and their derivatives
- Implementation into circuit simulation software
- Noise sources
- Jacobian matrix

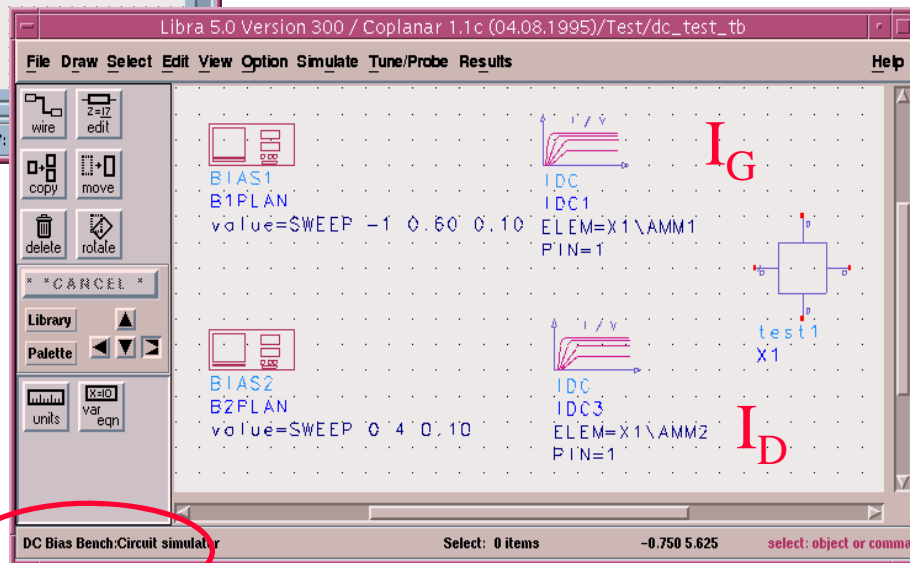
DC tests



All tests are shown for a transistor device. They are valid for diode devices as well.

Perform a DC analysis first!

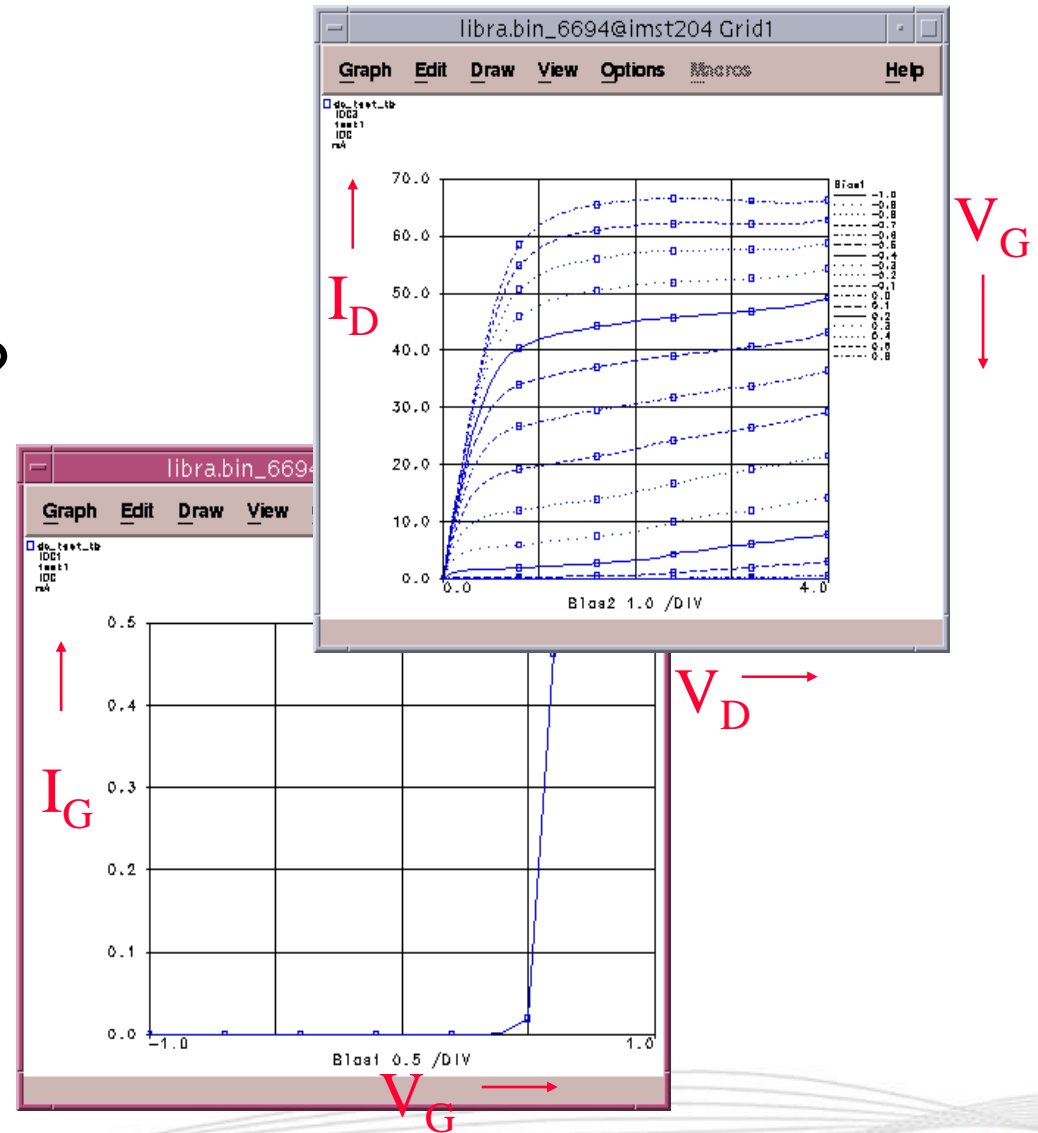
Only currents are calculated



DC testbench

DC tests

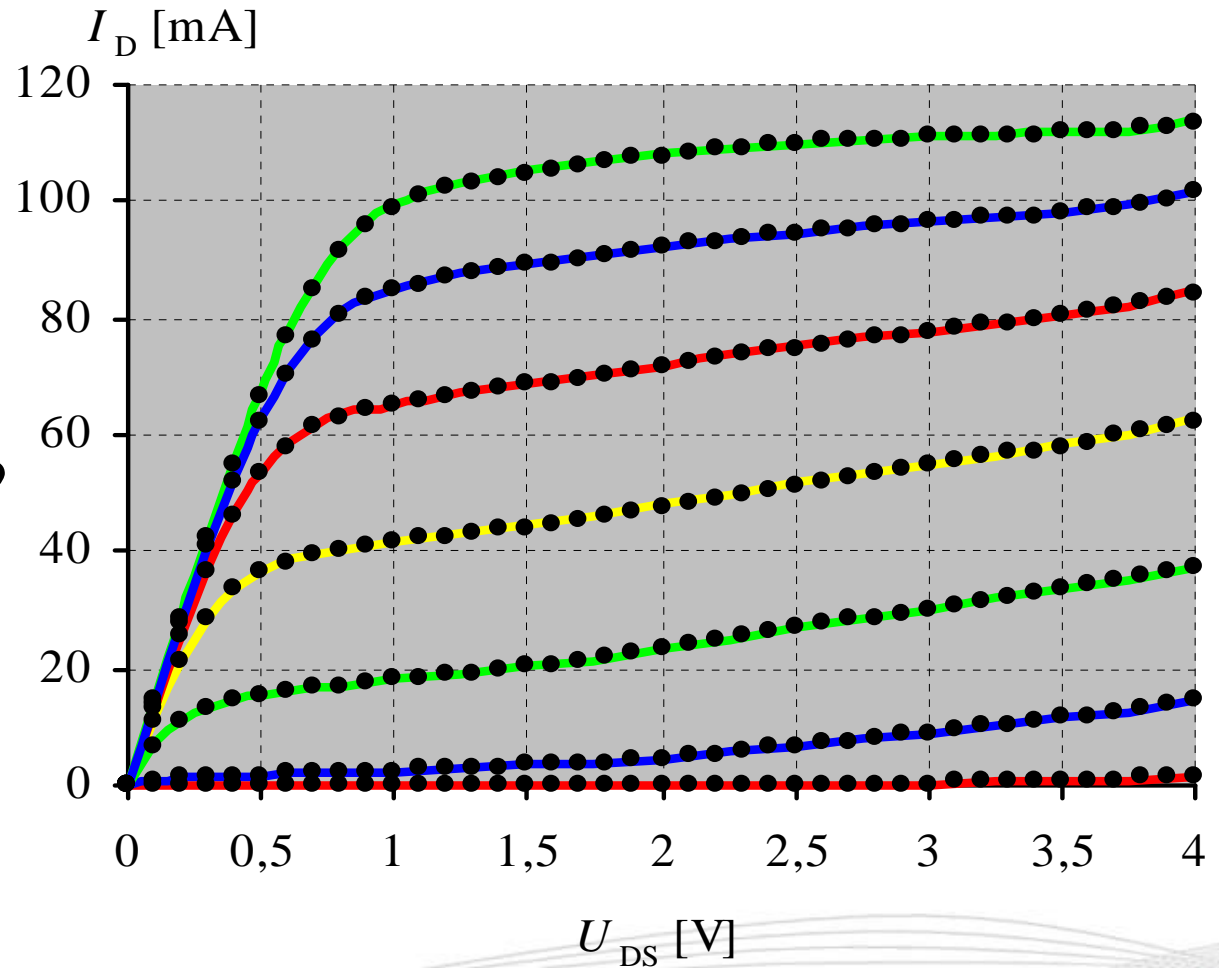
- IV output curves OK?
- IV input curves OK?
- Curves typical for element?
- Show measurements similar behaviour?



DC tests

T4x50 μm HEMT, simulation vs. measurements

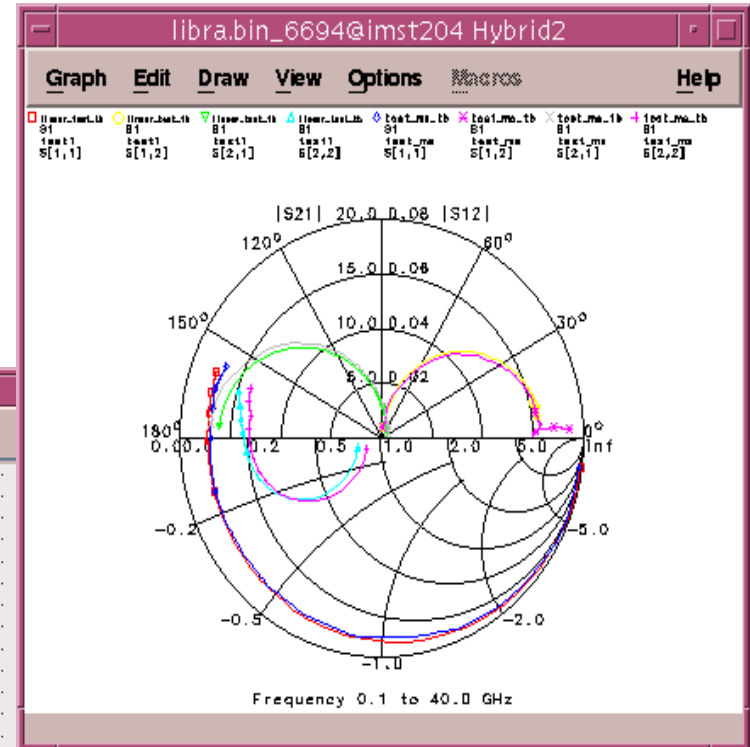
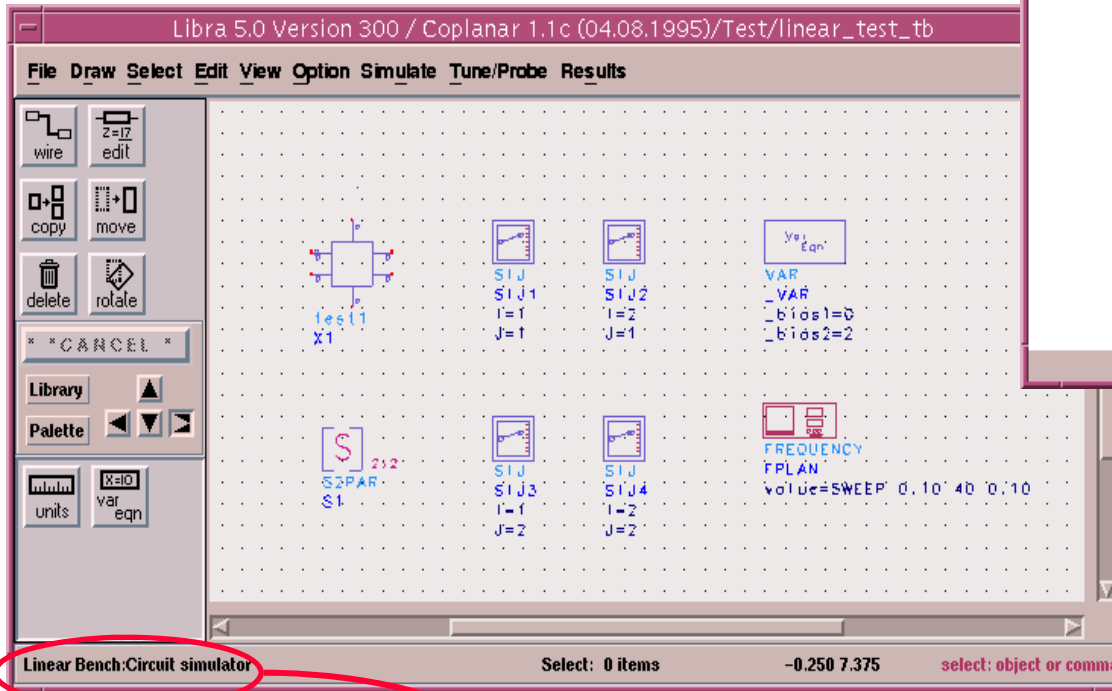
Show simulation
and measurements
similar behaviour?



RF Tests

Are measured and simulated S-parameters similar ?

Transistor: Gain, pinch-off S-parameters OK?



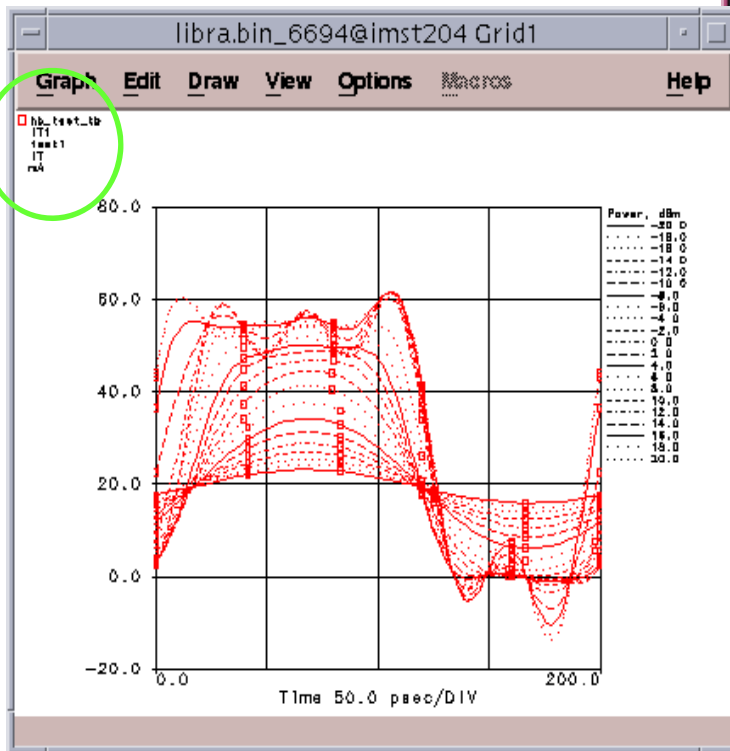
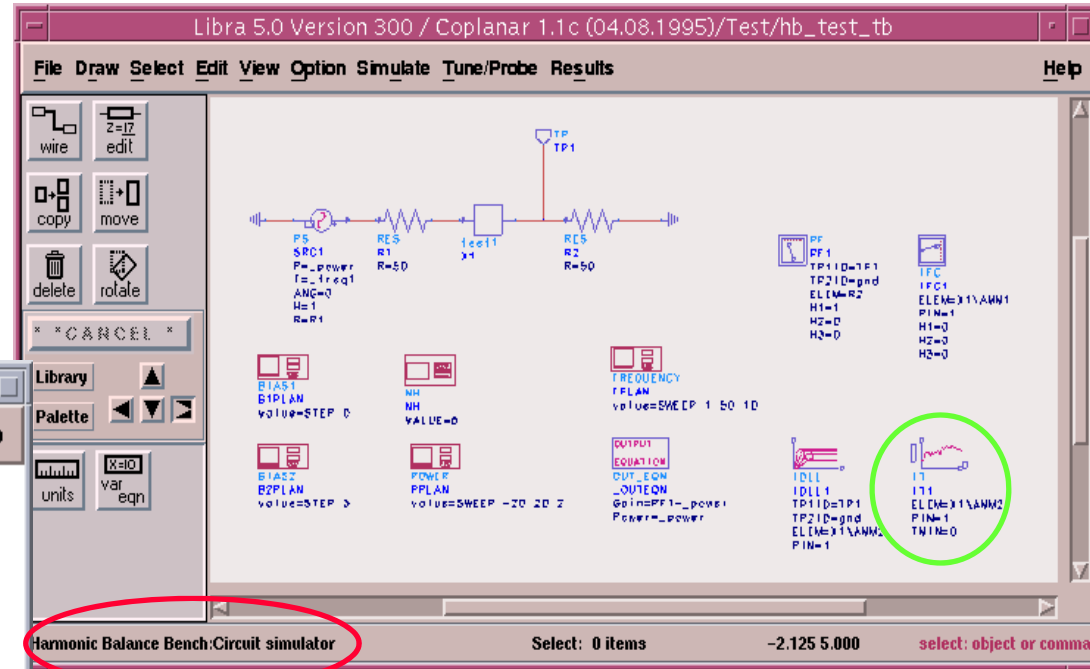
Diode: Forward and reverse S-parameters OK?

Lineares testbench



HB tests

Simulate a power sweep

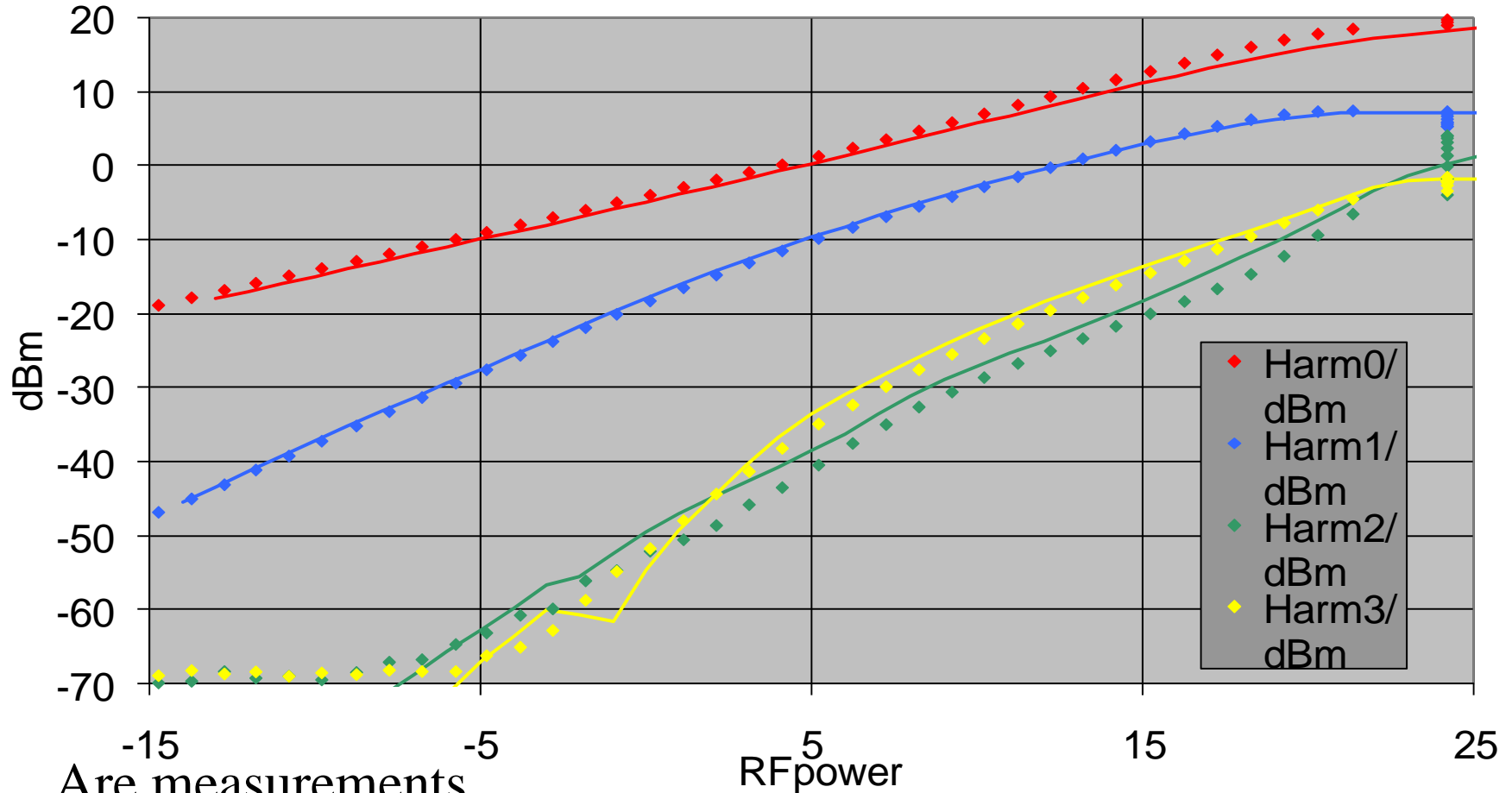


Harmonic Balance testbench

Is the behaviour as expected?
Are the nonlinearities dominant
for high input power

HB tests

Comp 1x600 2.1 GHz , 26 V, 2.1 mA



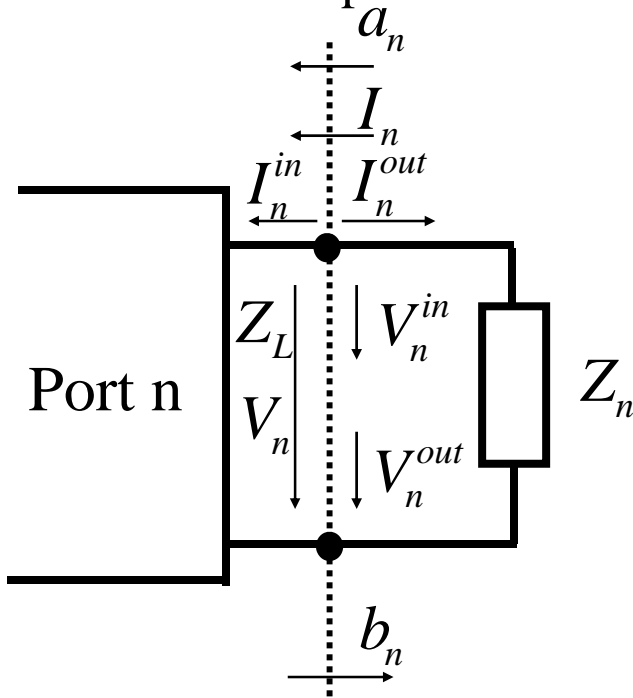
Are measurements
and simulations
similar ?

How good is the
quality of the model ?



Consistent implementation

Calculation of s-parameters using voltage and current definition



Waves in and out of port n

$$a_n = \frac{V_n^{in}}{\sqrt{Z_{Ln}}} = I_n^{in} \sqrt{Z_{Ln}}$$

$$b_n = \frac{V_n^{out}}{\sqrt{Z_{Ln}}} = I_n^{out} \sqrt{Z_{Ln}}$$

Currents and voltages

$$u_n = \frac{V_n}{\sqrt{Z_{Ln}}} = \frac{V_n^{in} + V_n^{out}}{\sqrt{Z_{Ln}}} = a_n + b_n$$

$$i_n = I_n \sqrt{Z_{Ln}} = (I_n^{in} - I_n^{out}) \sqrt{Z_{Ln}} = a_n - b_n$$

$$\Rightarrow \begin{cases} a_n = \frac{1}{2}(u_n + i_n) \\ b_n = \frac{1}{2}(u_n - i_n) \end{cases}$$

S matrix

$$\|b\| = \|s\| \cdot \|a\|$$

Example

$$s_{11} = \left. \frac{b_1}{a_1} \right|_{a_2=0}$$

$$s_{11,dB} = 20 \cdot \log \frac{u_1 - i_1}{u_1 + i_1}$$

Consistent implementation

Calculation of small signal S-parameters using a large signal test bench

The image displays two windows from a circuit simulator. The top window, titled 'Libra 5.0 Version 300 / Coplanar 1.1c (04.08.1995)/Schematic/FET_LS', shows a schematic diagram with two test benches. The top test bench is circled in red and contains a block labeled 'test1 X2' connected between ports 'P3' and 'P4'. The bottom test bench contains a block labeled 'test1 X1' connected between ports 'P1' and 'P2'. The bottom window, titled 'Libra 5.0 Version 300 / Coplanar 1.1c (04.08.1995)/Test/LS_test_tb', shows a more detailed circuit implementation of the test bench, also circled in red. This implementation includes a central block labeled 'FET_118' connected to various components like resistors (R1, R2, R3, R4), capacitors (C1, C2), and inductors (L1, L2). The status bar at the bottom of the bottom window reads 'Harmonic Balance Bench: Circuit simulator'.

Harmonic balance
testbench

Consistent implementation

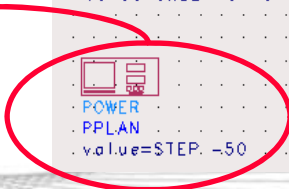
Calculation of small signal s-parameters using a large signal testbench

```
OUTPUT
EQUATION
OUT_EQN
_QUEQN
a11=VFC1/7.07+IFC1*7.07*1e-3
a22=VFC4/7.07+IFC4*7.07*1e-3
b11=VFC1/7.07-IFC1*7.07*1e-3
b22=VFC4/7.07-IFC4*7.07*1e-3
b21=VFC2/7.07-IFC2*7.07*1e-3
b12=VFC3/7.07-IFC3*7.07*1e-3
s11=20*log(b11/a11)
s11_ang=arg(b11/a11)
s22=20*log(b22/a22)
s22_ang=arg(b22/a22)
s21=20*log(b21/a11)
s21_ang=arg(b21/a11)
s12=20*log(b12/a22)
s12_ang=arg(b12/a22)
```

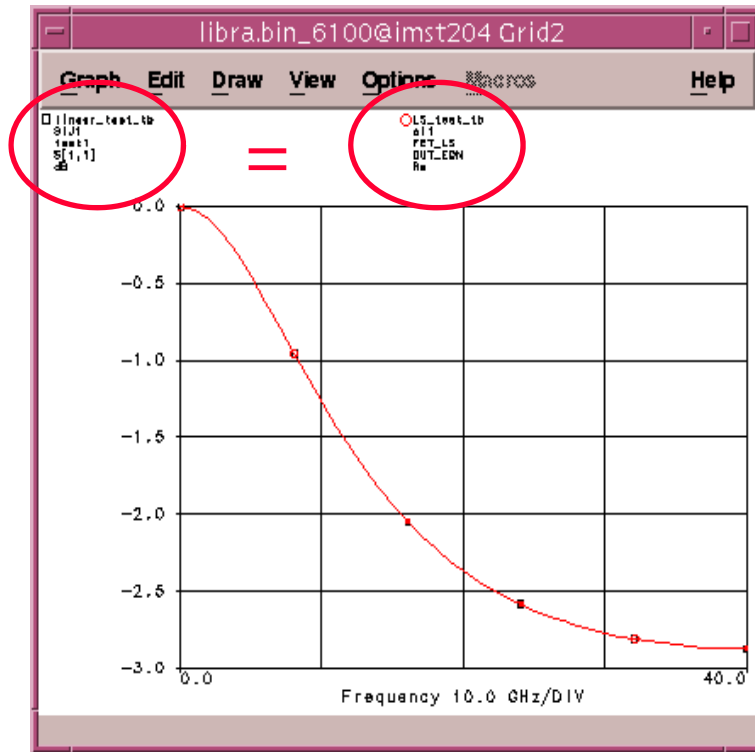
 VFC VFC1 TP1 ID=TP1 TP2 ID=gnd H1=1 H2=0 H3=0	 IFC IFC1 ELEM=R1 PIN=1 H1=1 H2=0 H3=0	 VFC VFC2 TP1 ID=TP2 TP2 ID=gnd H1=1 H2=0 H3=0	 IFC IFC2 ELEM=R2 PIN=1 H1=1 H2=0 H3=0	 VFC VFC3 TP1 ID=TP3 TP2 ID=gnd H1=1 H2=0 H3=0	 IFC IFC3 ELEM=R3 PIN=1 H1=1 H2=0 H3=0
 VFC VFC4 TP1 ID=TP4 TP2 ID=gnd H1=1 H2=0 H3=0	 IFC IFC4 ELEM=R4 PIN=1 H1=1 H2=0 H3=0				

 NH NH VALUE=5	 PF Pout_RF TP1 ID=TP2 TP2 ID=gnd ELEM=R2 H1=1 H2=0 H3=0	 PF Pin TP1 ID=TP1 TP2 ID=gnd ELEM=R1 H1=1 H2=0 H3=0	 PSPEC PSPEC1 TP1 ID=TP2 TP2 ID=gnd ELEM=R2
 FREQUENCY FPLAN value=SWEEP: 0:10:40:1			
 POWER PPLAN value=STEP: -50	 BIAS1 B1PLAN value=STEP: 0	 BIAS2 B2PLAN value=STEP: 2	

Small signal!
-50 dBm

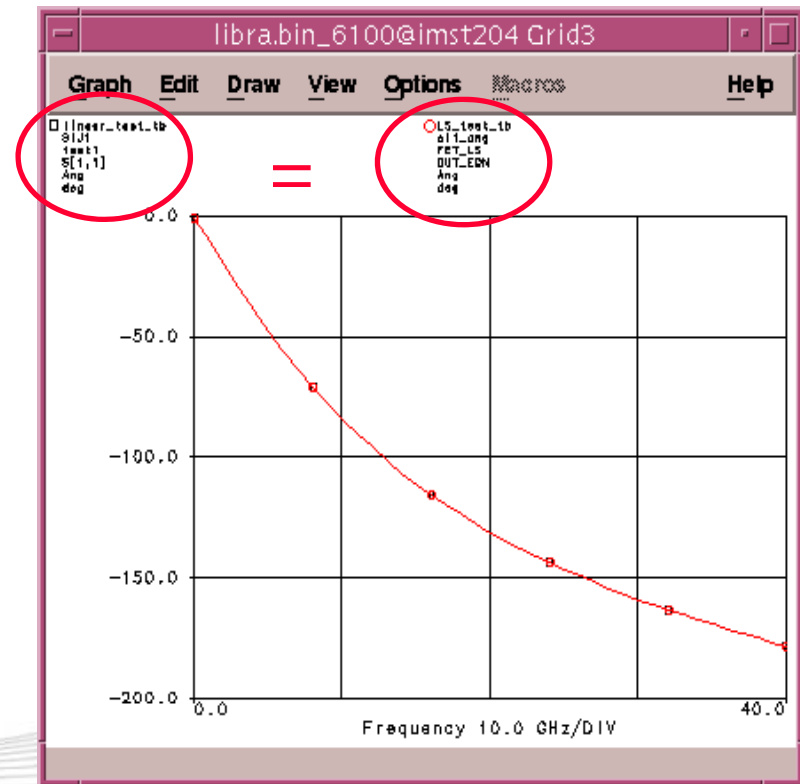


Consistent implementation



Mag(s_{11}), simulated using
Small signal and large signal
testbench for very low
input power (-50 dBm)

Phase(s_{11}), simulated using
Small signal and large signal
testbench for very low
input power (-50 dBm)



Needful things

Linearize exp() function for better conversion and avoid errors such as numerical overflows

$$f(x) = \begin{cases} \exp(x) & x \leq x_0 \\ mx + b & x > x_0 \end{cases}$$

Function must be continuous at x_0

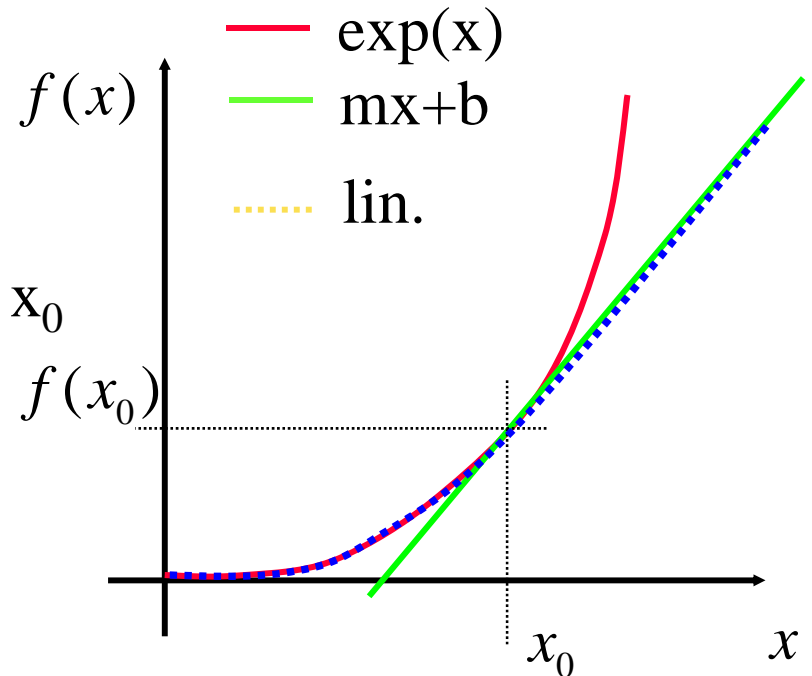
$$f_1(x_0) = f_2(x_0)$$

$$\left. \frac{df_1}{dx} \right|_{x_0} = \left. \frac{df_2}{dx} \right|_{x_0}$$

$$\Rightarrow \exp(x_0) = mx_0 + b$$

$$m = \exp(x_0)$$

$$\Rightarrow b = \exp(x_0)(1 - x_0) \Rightarrow f_2(x) = \exp(x_0)(x + 1 - x_0)$$



Summary

- Perform different tests (DC, RF, harmonic balance)
- Typical results for a FET
- Calculation of small signal s-parameters using a large signal testbench
- Linearization of functions

